무료 전자 책

배우기

Java Language

Free unaffiliated eBook created from
Stack Overflow contributors.

#java

1:

...

- : .
- : . ( ). JavaScript .
- : . , `String` .
- : Java , () ( ) .
- **Portable** : `javac` JVM .

Java "WORA (Write Once, Run Anywhere)" , Java Java .

Java (JVM) (`.class`) . JVM . JVM ( ). "JIT (Just-In-Time) ".

Java Edition

Sun / Oracle Java .

- *Java Standard Edition (SE)* .
- *Java Enterprise Edition (EE)* Java " " . Java EE .
- *Java Micro Edition (ME)* *Java SE* .

Java SE / EE / ME .

. Java SE .

Java

Java (Standard Edition) .

Java

:

-
- JAR Java
- Java
-

?

. Java .

- Java
- Java
-
- Java
-
- Java
- Java

Java . .

- JUnit ( )
- TestNG ( )

- .
- .
- Java Enterprise Edition Java EE .
- Oracle JavaFX JavaFX .

1. () Oracle Java SE . Java SE Oracle Java SE .

Java SE		([1])	
Java SE 9 (Early Access)			2017-07-27
Java SE 8			2014-03-18
Java SE 7		2015-04-14	2011 7 28
Java SE 6		2013-04-16	2006-12-23
Java SE 5		2009-11-04	2004-10-04
Java SE 1.4		2009-11-04	2002-02-06
Java SE 1.3		2009-11-04	2000-05-08
Java SE 1.2		2009-11-04	1998-12-08
Java SE 1.1		2009-11-04	1997-02-19
Java SE 1.0		2009-11-04	1996-01-21

Examples

`HelloWorld.java`

IDE . .

```
public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello, World!");
    }
}
```

Ideone

: Java `public class` `.java` (`HelloWorld`). `public` .

Java . (_) (`$`) .

`HelloWorld.java` .

```
cd /path/to/containing/folder/
```

: cd .

`javac` .

```
$ javac HelloWorld.java
```

```
'javac' is not recognized as an internal or external command, operable program or batch file.
'javac' is not recognized as an internal or external command, operable program or batch file.
'javac' is not recognized as an internal or external command, operable program or batch file.
```

JDK IDE `.eclipse` . .

Windows `javac.exe` `C:\Program Files\Java\jdk(version number)\bin` . .

```
$ C:\Program Files\Java\jdk(version number)\bin\javac HelloWorld.java
```

`javac` . OS `javac` . . PATH .

Windows XP / Vista / 7 / 8 / 10 `PATH` .

- ⇒ ⇒
- "" ⇒
- " " "PATH" . ⇒

. . `javac` `javac` . `c:\Program Files\Java\jdk1.8.0_xx\bin` `c:\Program Files\Java\jdk1.8.0_xx\bin`

"Variable value" **IN FRONT** (;) . .

```
Variable name  : PATH
Variable value : c:\Program Files\Java\jdk1.8.0_xx\bin;[Existing Entries...]
```

.

Linux .

: `javac` *Java .*

JVM (Java Virtual Machine) `HelloWorld.class` . Java `javac` Java `bytecode` . `Pluggable` Annotation Processing API . Java API .

`java main` (`HelloWorld`). `.class` .

```
$ java HelloWorld
```

: `java` *Java .*

:

, !

Java !

: Java (`java` , `javac`) .

- JDK (: Oracle , OpenJDK)
- .

JVM (`javac`) executor (`java`) . `java -version javac -version` . (: `1.8.0_73`).

Hello World .

"Hello World" `HelloWorld` , `main` `main` .

```
public class HelloWorld {
```

`class HelloWorld` . Java ( ).

```
    public static void main(String[] args)  {
```

`JVM` (`public static void main(String[])` `public static void main(String[])`). Java . :

- `public` : . .
- `static` : .
- `void` : . : `int` C C ++ *(Java* `System.exit()` *).*

.

- (:) `String` (`args`).

Java .

:

- `args` `args` .
- varargs (`String[] args`) Varargs (`String... args`) .

: (`main`) . `java` .

main .

```
        System.out.println("Hello, World!");
```

.

`System`	`java.lang` `System` .
.	" ". [1] . () . , `out` `System` .
`out`	`System` `PrintStream` .
.	. `out` `println` .
`println`	PrintStream . .
(	( ) `println` .
`"Hello, World!"`	`println` String . .
)	`println` .
;	.

: Java (;) .

.

```
    }  // end of main function scope
}      // end of class HelloWorld scope
```

OO . (, !) . , .

`Team` .

```
public class Team {
    Member member;
    public Team(Member member) {  // who is in this Team?
        this.member = member;  //  one 'member' is in this Team!
    }
}
```

Member .

```
class Member {
    private String name;
    private String type;
    private int level; // note the data type here
    private int rank; // note the data type here as well

    public Member(String name, String type, int level, int rank) {
        this.name = name;
        this.type = type;
        this.level = level;
        this.rank = rank;
    }
}
```

private ? , , . private . getter ( ) private .

getters main , :

```
public class Team {
    Member member;
    public Team(Member member) {
        this.member = member;
    }

    // here's our main method
    public static void main(String[] args) {
       Member myMember = new Member("Aurieel", "light", 10, 1);
       Team myTeam = new Team(myMember);
       System.out.println(myTeam.member.getName());
       System.out.println(myTeam.member.getType());
       System.out.println(myTeam.member.getLevel());
       System.out.println(myTeam.member.getRank());
    }
}

class Member {
    private String name;
    private String type;
    private int level;
    private int rank;

    public Member(String name, String type, int level, int rank) {
        this.name = name;
        this.type = type;
        this.level = level;
        this.rank = rank;
    }

    /* let's define our getter functions here */
    public String getName() { // what is your name?
        return this.name; // my name is ...
    }

    public String getType() { // what is your type?
        return this.type; // my type is ...
    }
```

```
    public int getLevel() { // what is your level?
        return this.level; // my level is ...
    }

    public int getRank() { // what is your rank?
        return this.rank; // my rank is
    }
}
```

:

```
Aurieel
light
10
1
```

, `Test main` . `main` JVM (Java Virtual Machine) .

---

1 - `HelloWorld System public` .

: https://riptutorial.com/ko/java/topic/84/--

2: BigDecimal

BigDecimal (, , ,), , , , . BigDecimal , 10 . .

Examples

BigDecimal .

BigDecimal BigDecimal .

```
BigDecimal a = new BigDecimal("42.23");
BigDecimal b = new BigDecimal("10.001");

a.add(b); // a will still be 42.23

BigDecimal c = a.add(b); // c will be 52.231
```

BigDecimal

compareTo BigDecimals .

```
BigDecimal a = new BigDecimal(5);
a.compareTo(new BigDecimal(0));    // a is greater, returns 1
a.compareTo(new BigDecimal(5));    // a is equal, returns 0
a.compareTo(new BigDecimal(10));   // a is less, returns -1
```

equals BigDecimals :

```
BigDecimal a = new BigDecimal(5);
a.equals(new BigDecimal(5));       // value and scale are equal, returns true
a.equals(new BigDecimal(5.00));    // value is equal but scale is not, returns false
```

BigDecimal

BigDecimal .

1.

```
BigDecimal a = new BigDecimal("5");
BigDecimal b = new BigDecimal("7");

//Equivalent to result = a + b
BigDecimal result = a.add(b);
System.out.println(result);
```

 : 12

2.

```
BigDecimal a = new BigDecimal("5");
BigDecimal b = new BigDecimal("7");

//Equivalent to result = a - b
BigDecimal result = a.subtract(b);
System.out.println(result);
```

: -2

3.

2 `BigDecimal` (), .

```
BigDecimal a = new BigDecimal("5.11");
BigDecimal b = new BigDecimal("7.221");

//Equivalent to result = a * b
BigDecimal result = a.multiply(b);
System.out.println(result);
```

: 36.89931

, `MathContext` multiply . Oracle .

```
BigDecimal a = new BigDecimal("5.11");
BigDecimal b = new BigDecimal("7.221");

MathContext returnRules = new MathContext(4, RoundingMode.HALF_DOWN);

//Equivalent to result = a * b
BigDecimal result = a.multiply(b, returnRules);
System.out.println(result);
```

: 36.90

4.

Division . .

```
BigDecimal a = new BigDecimal("5");
BigDecimal b = new BigDecimal("7");

BigDecimal result = a.divide(b);
System.out.println(result);
```

0.7142857142857143 .

: **java.lang.ArithmeticException : 10 ; .**

. 5 2 . `ArithmeticException` . BigDecimal **Scale Rounding Mode** . Oracle .

.

```
BigDecimal a = new BigDecimal("5");
BigDecimal b = new BigDecimal("7");

//Equivalent to result = a / b (Upto 10 Decimal places and Round HALF_UP)
BigDecimal result = a.divide(b,10,RoundingMode.HALF_UP);
System.out.println(result);
```

: 0.7142857143

5.Remainder Modulus

```
BigDecimal a = new BigDecimal("5");
BigDecimal b = new BigDecimal("7");

//Equivalent to result = a % b
BigDecimal result = a.remainder(b);
System.out.println(result);
```

: 5

6.Power

```
BigDecimal a = new BigDecimal("5");

//Equivalent to result = a^10
BigDecimal result = a.pow(10);
System.out.println(result);
```

: 9765625

7.

```
BigDecimal a = new BigDecimal("5");
BigDecimal b = new BigDecimal("7");

//Equivalent to result = MAX(a,b)
BigDecimal result = a.max(b);
System.out.println(result);
```

: 7

8.

```
BigDecimal a = new BigDecimal("5");
BigDecimal b = new BigDecimal("7");

//Equivalent to result = MIN(a,b)
BigDecimal result = a.min(b);
System.out.println(result);
```

: 5

.

```
BigDecimal a = new BigDecimal("5234.49843776");

//Moves the decimal point to 2 places left of current position
BigDecimal result = a.movePointLeft(2);
System.out.println(result);
```

: 52.3449843776

10.

```
BigDecimal a = new BigDecimal("5234.49843776");

//Moves the decimal point to 3 places right of current position
BigDecimal result = a.movePointRight(3);
System.out.println(result);
```

: 5234498.43776

(, 6). .

float BigDecimal

. . . . Java 7 BigDecimal .

```
import java.math.BigDecimal;

public class FloatTest {

public static void main(String[] args) {
    float accountBalance = 10000.00f;
    System.out.println("Operations using float:");
    System.out.println("1000 operations for 1.99");
    for(int i = 0; i<1000; i++){
        accountBalance -= 1.99f;
    }
```

```
    System.out.println(String.format("Account balance after float operations: %f",
accountBalance));

    BigDecimal accountBalanceTwo = new BigDecimal("10000.00");
    System.out.println("Operations using BigDecimal:");
    System.out.println("1000 operations for 1.99");
    BigDecimal operation = new BigDecimal("1.99");
    for(int i = 0; i<1000; i++){
        accountBalanceTwo = accountBalanceTwo.subtract(operation);
    }
    System.out.println(String.format("Account balance after BigDecimal operations: %f",
accountBalanceTwo));
}
```

.

```
Operations using float:
1000 operations for 1.99
Account balance after float operations: 8009,765625
Operations using BigDecimal:
1000 operations for 1.99
Account balance after BigDecimal operations: 8010,000000
```

10000.00 , 1.99 1000 8010.00 . float 8009.77 . . BigDecimal .

BigDecimal.valueOf ()

BigDecimal (: 0 ~ 10) . BigDecimal.valueOf () . , a b .

```
BigDecimal a = BigDecimal.valueOf(10L); //Returns cached Object reference
BigDecimal b = new BigDecimal(10L); //Does not return cached Object reference

BigDecimal a = BigDecimal.valueOf(20L); //Does not return cached Object reference
BigDecimal b = new BigDecimal(20L); //Does not return cached Object reference


BigDecimal a = BigDecimal.valueOf(15.15); //Preferred way to convert a double (or float) into
a BigDecimal, as the value returned is equal to that resulting from constructing a BigDecimal
from the result of using Double.toString(double)
BigDecimal b = new BigDecimal(15.15); //Return unpredictable result
```

0, 1 10 BigDecimal

BigDecimal 0, 1 10 . .

- BigDecimal.ZERO
- BigDecimal.ONE
- BigDecimal.TEN

'' .

```
//Bad example:
BigDecimal bad0 = new BigDecimal(0);
BigDecimal bad1 = new BigDecimal(1);
```

```
BigDecimal bad10 = new BigDecimal(10);

//Good Example:
BigDecimal good0 = BigDecimal.ZERO;
BigDecimal good1 = BigDecimal.ONE;
BigDecimal good10 = BigDecimal.TEN;
```

BigDecimal : https://riptutorial.com/ko/java/topic/1667/bigdecimal

3: BigInteger

BigInteger . 100 158 . long . BigInteger Java java.lang.Math .

- BigInteger variable_name = BigInteger ("12345678901234567890"); // 10
- BigInteger variable_name = new BigInteger ("10101011010101001010100110001100111010110001111100001010110100 10", 2) // .
- BigInteger variable_name = new BigInteger ("ab54a98ceb1f0800", 16) // 16
- BigInteger variable_name = BigInteger (64, ()); // 64
- BigInteger variable_name = new BigInteger ([] {0, -85, 84, -87, -116, -21, 31, 10, -46}); // 2 ( )
- BigInteger variable_name = new BigInteger (1, [] {- 85, 84, -87, -116, -21, 31, 10, -46}); // 2 ( )

BigInteger . . , sum .

```
BigInteger sum = BigInteger.ZERO;
for(int i = 1; i < 5000; i++) {
   sum.add(BigInteger.valueOf(i));
}
```

sum .

```
sum = sum.add(BigInteger.valueOf(i));
```

Java SE 8

BigInteger , BigInteger -2 2147483647 2 2147483647 ( ) . BigInteger 20 !

Examples

java.math.BigInteger java.lang.Math . java.math java.math.BigInteger .

long int BigInteger .

```
long longValue = Long.MAX_VALUE;
BigInteger valueFromLong = BigInteger.valueOf(longValue);
```

:

```
int intValue = Integer.MIN_VALUE; // negative
BigInteger valueFromInt = BigInteger.valueOf(intValue);
```

intValue , , .

String BigInteger .

```
String decimalString = "-1";
BigInteger valueFromDecimalString = new BigInteger(decimalString);
```

a `BigInteger` `BigInteger`.

```
String binaryString = "10";
int binaryRadix = 2;
BigInteger valueFromBinaryString = new BigInteger(binaryString , binaryRadix);
```

Java `BigInteger` . .

```
byte[] bytes = new byte[] { (byte) 0x80 };
BigInteger valueFromBytes = new BigInteger(bytes);
```

-128 `BigInteger` .

```
byte[] unsignedBytes = new byte[] { (byte) 0x80 };
int sign = 1; // positive
BigInteger valueFromUnsignedBytes = new BigInteger(sign, unsignedBytes);
```

1, 128 `BigInteger` .

.

- `BigInteger.ZERO` - "0".
- `BigInteger.ONE` - "1".
- `BigInteger.TEN` - "10".

`BigInteger.TWO` ("2") `private` .

BigIntegers

Java `String` `BigIntegers` .

:

```
BigInteger one = BigInteger.valueOf(1);
BigInteger two = BigInteger.valueOf(2);

if(one.equals(two)){
    System.out.println("Equal");
}
else{
    System.out.println("Not Equal");
}
```

:

```
Not Equal
```

:

`==` BigIntegers **.**

- `==` : . ,
- `equals()` : 2 BigInteger .

, BigInteger :

```
if (firstBigInteger == secondBigInteger) {
  // Only checks for reference equality, not content equality!
}
```

`==` . BigInteger , **.** `equals` BigIntegers .

`BigInteger` 0,1,10 .

:

```
BigInteger reallyBig = BigInteger.valueOf(1);
if(BigInteger.ONE.equals(reallyBig)){
    //code when they are equal.
}
```

`compareTo()` BigInteger . `compareTo()` 3 .

- **0 :** .
- **1 :** ( ).
- **-1 :** .

```
BigInteger reallyBig = BigInteger.valueOf(10);
BigInteger reallyBig1 = BigInteger.valueOf(100);

if(reallyBig.compareTo(reallyBig1) == 0){
    //code when both are equal.
}
else if(reallyBig.compareTo(reallyBig1) == 1){
    //code when reallyBig is greater than reallyBig1.
}
else if(reallyBig.compareTo(reallyBig1) == -1){
    //code when reallyBig is less than reallyBig1.
}
```

BigInteger

BigInteger , BigInteger .

: *10 + 10 = 20*

```
BigInteger value1 = new BigInteger("10");
BigInteger value2 = new BigInteger("10");
```

```
BigInteger sum = value1.add(value2);
System.out.println(sum);
```

: 20

: *10 - 9 = 1*

```
BigInteger value1 = new BigInteger("10");
BigInteger value2 = new BigInteger("9");

BigInteger sub = value1.subtract(value2);
System.out.println(sub);
```

: 1

: *10 / 5 = 2*

```
BigInteger value1 = new BigInteger("10");
BigInteger value2 = new BigInteger("5");

BigInteger div = value1.divide(value2);
System.out.println(div);
```

: 2

Division : *17/4 = 4*

```
BigInteger value1 = new BigInteger("17");
BigInteger value2 = new BigInteger("4");

BigInteger div = value1.divide(value2);
System.out.println(div);
```

: 4

: *10 * 5 = 50*

```
BigInteger value1 = new BigInteger("10");
BigInteger value2 = new BigInteger("5");

BigInteger mul = value1.multiply(value2);
System.out.println(mul);
```

: 50

: *10 ^ 3 = 1000*

```
BigInteger value1 = new BigInteger("10");
BigInteger power = value1.pow(3);
System.out.println(power);
```

: 1000

: *10 % 6 = 4*

```
BigInteger value1 = new BigInteger("10");
BigInteger value2 = new BigInteger("6");

BigInteger power = value1.remainder(value2);
System.out.println(power);
```

: 4

GCD : `12` `18` GCD (Greatest Divisor) `6` .

```
BigInteger value1 = new BigInteger("12");
BigInteger value2 = new BigInteger("18");

System.out.println(value1.gcd(value2));
```

: 6

2 BigInteger :

```
BigInteger value1 = new BigInteger("10");
BigInteger value2 = new BigInteger("11");

System.out.println(value1.max(value2));
```

: 11

2 BigInteger :

```
BigInteger value1 = new BigInteger("10");
BigInteger value2 = new BigInteger("11");

System.out.println(value1.min(value2));
```

: 10

BigInteger

BigInteger `Number` . .

:

```
BigInteger val1 = new BigInteger("10");
BigInteger val2 = new BigInteger("9");

val1.or(val2);
```

: 11 (`10 | 9`)

:

```
BigInteger val1 = new BigInteger("10");
BigInteger val2 = new BigInteger("9");

val1.and(val2);
```

: 8 (`10 & 9`)

Xor :

```
BigInteger val1 = new BigInteger("10");
BigInteger val2 = new BigInteger("9");

val1.xor(val2);
```

: 3 (`10 ^ 9`)

RightShift :

```
BigInteger val1 = new BigInteger("10");

val1.shiftRight(1);   // the argument be an Integer
```

: 5 (`10 >> 1`)

LeftShift :

```
BigInteger val1 = new BigInteger("10");

val1.shiftLeft(1);   // here parameter should be Integer
```

: 20 (`10 << 1`)

(Not) :

```
BigInteger val1 = new BigInteger("10");

val1.not();
```

: 5

*NAND (And-Not) : **

```
BigInteger val1 = new BigInteger("10");
BigInteger val2 = new BigInteger("9");

val1.andNot(val2);
```

: 7

BigIntegers

`BigInteger`

, java.util.Random BigInteger int , random BigIntegers BigInteger . . BigInteger(int, Random) :

```
BigInteger randomBigInt = new BigInteger(bitCount, sourceOfRandomness);
```

0 () 2 bitCount ( ) BigInteger .

new BigInteger(2147483647, sourceOfRandomness) BigInteger .

---

sourceOfRandomness . , new Random() .

```
new BigInteger(32, new Random());
```

, new SecureRandom () :

```
import java.security.SecureRandom;

// somewhere in the code...
new BigInteger(32, new SecureRandom());
```

! **RNG** *()* , , BigInteger , .

```
new BigInteger(32, new Random() {
    int seed = 0;

    @Override
    protected int next(int bits) {
        seed = ((22695477 * seed) + 1) & 2147483647; // Values shamelessly stolen from
Wikipedia
        return seed;
    }
});
```

BigInteger : https://riptutorial.com/ko/java/topic/1514/biginteger

4: BufferedWriter

- BufferedWriter (Writer); //
- BufferedWriter.write (int c); // .
- BufferedWriter.write (String str); // .
- BufferedWriter.newLine (); // .
- BufferedWriter.close (); // BufferedWriter .

- `BufferedWriter` (`BufferedWriter.write()`) `BufferedWriter` (`BufferedWriter.close()`), `IOException` .
- `BufferedWriter(Writer)` `IOException` . `FileWriter(File)` `IOException` `FileNotFoundException` . `IOException` `FileNotFoundException` , `FileNotFoundException` catch .

Examples

. `finally` .

```
  public void writeLineToFile(String str) throws IOException {
    File file = new File("file.txt");
    BufferedWriter bw = null;
    try {
      bw = new BufferedWriter(new FileWriter(file));
      bw.write(str);
    } finally {
      if (bw != null) {
        bw.close();
      }
    }
  }
```

`write(String s)` . `newLine()` .

Java SE 7

Java 7 `java.nio.file` try-with-resources .

```
public void writeLineToFile(String str) throws IOException {
    Path path = Paths.get("file.txt");
    try (BufferedWriter bw = Files.newBufferedWriter(path)) {
        bw.write(str);
    }
}
```

BufferedWriter : https://riptutorial.com/ko/java/topic/3063/bufferedwriter

5: ByteBuffer

`ByteBuffer` Java 1.4 . . `byte[]`

- byte [] arr = [1000];
- ByteBuffer buffer = ByteBuffer.wrap (arr);
- ByteBuffer buffer = ByteBuffer.allocate (1024);
- ByteBuffer buffer = ByteBuffer.allocateDirect (1024);
- b = buffer.get ();
- b = buffer.get (10);
- s = buffer.getShort (10);
- buffer.put ((byte) 120);
- buffer.putChar ('a');

Examples

- ByteBuffer

`ByteBuffer` .

`byte[]` `ByteBuffer` " " .

```
byte[] reqBuffer = new byte[BUFFER_SIZE];
int readBytes = socketInputStream.read(reqBuffer);
final ByteBuffer reqBufferWrapper = ByteBuffer.wrap(reqBuffer);
```

`byte[]` `ByteBuffer` :

```
final ByteBuffer respBuffer = ByteBuffer.allocate(RESPONSE_BUFFER_SIZE);
putResponseData(respBuffer);
socketOutputStream.write(respBuffer.array());
```

`ByteBuffer #allocateDirect()` .

-

`ByteBuffer` (), `put` `put` . `put` `put` .

"" . .

```
buffer.putInt(0xCAFEBABE).putChar('c').putFloat(0.25).putLong(0xDEADBEEFCAFEBABE);
```

.

```
buffer.putInt(0xCAFEBABE);
buffer.putChar('c');
```

```
buffer.putFloat(0.25);
buffer.putLong(0xDEADBEEFCAFEBABE);
```

byte S . ByteBuffer byte[] put . , put .

: put* byte .

- DirectByteBuffer

DirectByteBuffer byte[] ByteBuffer .

ByteBuffer .

```
ByteBuffer directBuffer = ByteBuffer.allocateDirect(16);
```

16 . .

ByteBuffer .

```
directBuffer.isDirect(); // true
```

DirectByteBuffer JVM ByteBuffers .

IO DirectByteBuffer DirectByteBuffer .

array() UnsupportedOperationException . ByteBuffer ByteBuffer .

```
byte[] arrayOfBytes;
if(buffer.hasArray()) {
    arrayOfBytes = buffer.array();
}
```

byte JNI interop. byte[] . Java .

JNI . NIO .

ByteBuffer : https://riptutorial.com/ko/java/topic/702/bytebuffer

6: C ++

Java C ++ . Java C ++ .

#

C ++

[ref] ( )

```
class Outer {
   class Inner {
      public:
         Inner(Outer* o) :outer(o) {}

      private:
         Outer*  outer;
   };
};
```

[] ( )

```
class OuterClass {
    ...
    class InnerClass {
        ...
    }
}
```

C ++

```
class Outer {
   class Inner {
      ...
   };
};
```

( ) [ref]

```
class OuterClass {
    ...
    static class StaticNestedClass {
        ...
    }
}
```

(:)

C ++

[ref]

```
void fun() {
   class Test {
      /* members of Test class */
   };
}
```

[ref]

```
class Test {
    void f() {
        new Thread(new Runnable() {
            public void run() {
                doSomethingBackgroundish();
            }
        }).start();
    }
}
```

C ++ Java .

- .
- .
- . "Java " Stackoverflow - 1 ; 2

(Polymorphism) . .

- Shape
- Square Circle .
- .

C ++ . Java .

I

C ++ .

Java finalize C ++ . finalizers (GC). - "" .

```
protected void close() {
```

```
    try {
       // do subclass cleanup
    }
    finally {
       isClosed = true;
       super.close();
    }
}

protected void finalize() {
    try {
       if(!isClosed) close();
    }
    finally {
       super.finalize();
    }
}
```

	C ++	
.	`virtual void eat(void) = 0;`	`abstract void draw();`
.	. `class AB {public: virtual void f() = 0;};`	. . `abstract class GraphicObject {}`
	"" Java	1) . 2) `interface TestInterface {}`

	C ++	
-		
-		
- .		
	private. struct public.	.
	- ( )	

C ++

```
class Node {
  private:
    int key;  Node *next;
    // LinkedList::search() can access "key" & "next"
    friend int LinkedList::search();
};
```

B C A D B C . A B C D D ? B C? ( )

C ++ Java Java 8 . Java Java "" .

java.lang.Object

Java Object . Java Object .

C ++ "Object" .

Java Collections & C ++

C ++ .

Java Collections

C ++

			C ++ (LLP64 LP64)	Java
8	-2 (8-1) = -128	2 (8-1) -1 = 127		
8	0	2 (8) -1 = 255		-
16	-2 (16-1) = -32,768	2 (16-1) -1 = 32,767		
16	0 (\ u0000)	2 (16) -1 = 65,535 (\ uFFFF)		char ()
32	-2 (32-1) = -24.77	2 (32-1) -1 = 2147	int	int
32	0	2 (32) -1 = 42 5		-
64	-2 (64-1)	2 (16-1) -1	*	.
64	0	2 (16) -1	long * long long	-

* Win64 API 32 .

C ++

Examples

.

C ++

```
// define in header
class Singleton {
   public:
      static Singleton *getInstance();

   private:
      Singleton() {}
      static Singleton *instance;
};

// initialize in .cpp
Singleton* Singleton::instance = 0;
```

Java

```
public class Singleton {
    private static Singleton instance;

    private Singleton() {}

    public static Singleton getInstance() {
        if(instance == null) {
            instance = new Singleton();
        }
        return instance;
    }
}
```

C ++

[ref] ( )

```
class Outer {
   class Inner {
      public:
         Inner(Outer* o) :outer(o) {}

      private:
         Outer*  outer;
   };
};
```

[] ( )

```
class OuterClass {
    ...
    class InnerClass {
        ...
    }
}
```

C ++

```
class Outer {
   class Inner {
      ...
   };
};
```

( ) [ref]

```
class OuterClass {
    ...
    static class StaticNestedClass {
        ...
    }
}
```

(:)

C ++

[ref]

```
void fun() {
   class Test {
      /* members of Test class */
   };
}
```

[ref]

```
class Test {
    void f() {
        new Thread(new Runnable() {
            public void run() {
                doSomethingBackgroundish();
            }
        }).start();
    }
}
```

, . C ++ Java ( ) ( ) .

C ++ ( )

```
// passes a COPY of the object
static void passByCopy(PassIt obj) {
   obj.i = 22;  // only a "local" change
}

// passes a pointer
static void passByPointer(PassIt* ptr) {
   ptr->i = 33;
   ptr = 0; // better to use nullptr instead if '0'
}

// passes an alias (aka reference)
static void passByAlias(PassIt& ref) {
   ref.i = 44;
}

// This is an old-school way of doing it.
// Check out std::swap for the best way to do this
static void swap(PassIt** pptr1, PassIt** pptr2) {
   PassIt* tmp = *pptr1;
   *pptr1 = *pptr2;
   *pptr2 = tmp;
}
```

Java ( )

```
// passes a copy of the variable
// NOTE: in java only primitives are pass-by-copy
public static void passByCopy(int copy) {
   copy = 33;  // only a "local" change
}

// No such thing as pointers in Java
/*
public static void passByPointer(PassIt *ptr) {
   ptr->i = 33;
   ptr = 0; // better to use nullptr instead if '0'
}
*/

// passes an alias (aka reference)
public static void passByAlias(PassIt ref) {
   ref.i = 44;
}

// passes aliases (aka references),
// but need to do "manual", potentially expensive copies
public static void swap(PassIt ref1, PassIt ref2) {
   PassIt tmp = new PassIt(ref1);
   ref1.copy(ref2);
   ref2.copy(tmp);
}
```

C ++ Java .

Downcasting (,) . instanceof & downcasting & .

C ++

```
// explicit type case required
Child *pChild =  (Child *) &parent;
```

Java

```
if(mySubClass instanceof SubClass) {
   SubClass mySubClass = (SubClass)someBaseClass;
   mySubClass.nonInheritedMethod();
}
```

.

C ++

```
virtual void eat(void) = 0;
```

```
abstract void draw();
```

.

C ++

.

```
class AB {public: virtual void f() = 0;};
```

. .

```
abstract class GraphicObject {}
```

C ++

.

1) . 2)

```
interface TestInterface {}
```

C ++ : https://riptutorial.com/ko/java/topic/10849/c-plusplus-

7: Dequeue

Deque .

deque "double ended queue" "deck" .

Deque deques .

Deque .

Deque .

```
Deque<Object> deque = new LinkedList<Object>();
```

FIFO (First-In-First-Out) .

Deques LIFO (Last-In-First-Out) .

.

Examples

Deque

```
Deque deque = new LinkedList();

//Adding element at tail
deque.add("Item1");

//Adding element at head
deque.addFirst("Item2");

//Adding element at tail
deque.addLast("Item3");
```

Deque

```
//Retrieves and removes the head of the queue represented by this deque
Object headItem = deque.remove();

//Retrieves and removes the first element of this deque.
Object firstItem = deque.removeFirst();

//Retrieves and removes the last element of this deque.
Object lastItem = deque.removeLast();
```

```
//Retrieves, but does not remove, the head of the queue represented by this deque
Object headItem = deque.element();

//Retrieves, but does not remove, the first element of this deque.
```

```
Object firstItem = deque.getFirst();

//Retrieves, but does not remove, the last element of this deque.
Object lastItem  = deque.getLast();
```

Deque

```
//Using Iterator
Iterator iterator = deque.iterator();
while(iterator.hasNext(){
  String Item = (String) iterator.next();
}

//Using For Loop
for(Object object : deque) {
    String Item = (String) object;
}
```

Dequeue : https://riptutorial.com/ko/java/topic/10156/dequeue-

8: EnumSet

Java EnumSet enum Set . AbstractSet Set .

Examples

```
import java.util.*;
enum days {
  SUNDAY, MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, SATURDAY
}
public class EnumSetExample {
  public static void main(String[] args) {
    Set<days> set = EnumSet.of(days.TUESDAY, days.WEDNESDAY);
    // Traversing elements
    Iterator<days> iter = set.iterator();
    while (iter.hasNext())
      System.out.println(iter.next());
  }
}
```

EnumSet : https://riptutorial.com/ko/java/topic/10159/enumset-

9: Executor, ExecutorService Thread

Java Executor , . Executor . Executors .

- ScheduledExecutorService . ScheduledExecutorService .

Examples

-

Executors `java.lang.Runnable` . `java.lang.Runnable` (( ) .

.

```
Executor exec = anExecutor;
exec.execute(new Runnable() {
    @Override public void run() {
        //offloaded work, no need to get result back
    }
});
```

.
Java 8 lambda .

Java SE 8

```
Executor exec = anExecutor;
exec.execute(() -> {
    //offloaded work, no need to get result back
});
```

ThreadPoolExecutor

Executor `ThreadPoolExecutor` , `ThreadPoolExecutor` . ( ) . .

```
ThreadPoolExecutor pool = new ThreadPoolExecutor(
    1,                                      // keep at least one thread ready,
                                            // even if no Runnables are executed
    5,                                      // at most five Runnables/Threads
                                            // executed in parallel
    1, TimeUnit.MINUTES,                    // idle Threads terminated after one
                                            // minute, when min Pool size exceeded
    new ArrayBlockingQueue<Runnable>(10)); // outstanding Runnables are kept here

pool.execute(new Runnable() {
    @Override public void run() {
        //code to run
    }
});
```

`ThreadPoolExecutor` , `corePoolSize` .

ThreadPoolExecutor

```
ThreadPoolExecutor(int corePoolSize, int maximumPoolSize, long keepAliveTime,
TimeUnit unit, BlockingQueue<Runnable> workQueue, ThreadFactory threadFactory,
RejectedExecutionHandler handler)
```

JavaDoc

corePoolSize maximumPoolSize , .

:

1. BlockingQueue . .
2. Rejection Handler Rejection Handler .
    1. ThreadPoolExecutor.AbortPolicy RejectedExecutionException .
    2. `ThreadPoolExecutor.CallerRunsPolicy` execute . .
    3. `ThreadPoolExecutor.DiscardPolicy` , .
    4. `ThreadPoolExecutor.DiscardOldestPolicy` executor ( ).
3. `ThreadFactory` .
    1.
    2.
    3.

ThreadPoolExecutor .

-

Runnable . `ExecutorService.submit(` `Callable` `<T>)` .

`Future` .

```
// Submit a callable for execution
ExecutorService pool = anExecutorService;
Future<Integer> future = pool.submit(new Callable<Integer>() {
    @Override public Integer call() {
        //do some computation
        return new Random().nextInt();
    }
});
// ... perform other tasks while future is executed in a different thread
```

`future.get()` `future.get()`

- .

```
try {
    // Blocks current thread until future is completed
    Integer result = future.get();
catch (InterruptedException || ExecutionException e) {
    // handle appropriately
}
```

- . .

```
try {
    // Blocks current thread for a maximum of 500 milliseconds.
    // If the future finishes before that, result is returned,
    // otherwise TimeoutException is thrown.
    Integer result = future.get(500, TimeUnit.MILLISECONDS);
catch (InterruptedException || ExecutionException || TimeoutException e) {
    // handle appropriately
}
```

`Future.cancel(boolean)` .

- `cancel(false)` .
- `cancel(true)` .

`ScheduledExecutorService` . `pool` .

```
ScheduledExecutorService pool = Executors.newScheduledThreadPool(2);
```

`ExecutorService` `ScheduledExecutorService` API `ScheduledFuture` 4 . ( ) .

10 .

```
ScheduledFuture<Integer> future = pool.schedule(new Callable<>() {
        @Override public Integer call() {
            // do something
            return 42;
        }
    },
    10, TimeUnit.MINUTES);
```

10 1 .

```
ScheduledFuture<?> future = pool.scheduleAtFixedRate(new Runnable() {
        @Override public void run() {
            // do something
        }
    },
    10, 1, TimeUnit.MINUTES);
```

`pool` , `future` , .

`scheduledAtFixedRate` . .

10 1 .

```
ScheduledFuture<?> future = pool.scheduleWithFixedDelay(new Runnable() {
        @Override public void run() {
            // do something
        }
    },
    10, 1, TimeUnit.MINUTES);
```

`pool` , `future` , .

1. Executor
2. ( ) .

`RejectedExecutionHandler.rejectedExecution(Runnable, ThreadPoolExecutor)` .

Throw RejectedExecutionException . .

- **ThreadPoolExecutor.AbortPolicy** (, REE)
- **ThreadPoolExecutor.CallerRunsPolicy** (-)
- **ThreadPoolExecutor.DiscardPolicy** ( )
- **ThreadPoolExecutor.DiscardOldestPolicy** ( )

ThreadPool .

```
public ThreadPoolExecutor(int corePoolSize,
                          int maximumPoolSize,
                          long keepAliveTime,
                          TimeUnit unit,
                          BlockingQueue<Runnable> workQueue,
                          RejectedExecutionHandler handler) // <--

public ThreadPoolExecutor(int corePoolSize,
                          int maximumPoolSize,
                          long keepAliveTime,
                          TimeUnit unit,
                          BlockingQueue<Runnable> workQueue,
                          ThreadFactory threadFactory,
                          RejectedExecutionHandler handler) // <--
```

RejectedExecutionHandler .

```
void rejectedExecution(Runnable r, ThreadPoolExecutor executor)
```

submit () vs execute ()

execute () fire forget call ( ) submit () Future .

.

submit () .

:

1 : Exception execute () Runnable .

```
import java.util.concurrent.*;
import java.util.*;

public class ExecuteSubmitDemo {
    public ExecuteSubmitDemo() {
        System.out.println("creating service");
        ExecutorService service = Executors.newFixedThreadPool(2);
        //ExtendedExecutor service = new ExtendedExecutor();
        for (int i = 0; i < 2; i++){
            service.execute(new Runnable(){
                 public void run(){
                    int a = 4, b = 0;
                    System.out.println("a and b=" + a + ":" + b);
                    System.out.println("a/b:" + (a / b));
                    System.out.println("Thread Name in Runnable after divide by
zero:"+Thread.currentThread().getName());
                 }
            });
        }
        service.shutdown();
    }
    public static void main(String args[]){
        ExecuteSubmitDemo demo = new ExecuteSubmitDemo();
    }
}

class ExtendedExecutor extends ThreadPoolExecutor {

   public ExtendedExecutor() {
       super(1, 1, 60, TimeUnit.SECONDS, new ArrayBlockingQueue<Runnable>(100));
   }
   // ...
   protected void afterExecute(Runnable r, Throwable t) {
     super.afterExecute(r, t);
     if (t == null && r instanceof Future<?>) {
       try {
         Object result = ((Future<?>) r).get();
       } catch (CancellationException ce) {
           t = ce;
       } catch (ExecutionException ee) {
           t = ee.getCause();
       } catch (InterruptedException ie) {
           Thread.currentThread().interrupt(); // ignore/reset
       }
     }
     if (t != null)
       System.out.println(t);
   }
 }
```

:

```
creating service
a and b=4:0
a and b=4:0
Exception in thread "pool-1-thread-1" Exception in thread "pool-1-thread-2"
java.lang.ArithmeticException: / by zero
```

```
        at ExecuteSubmitDemo$1.run(ExecuteSubmitDemo.java:15)
        at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1145)
        at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:615)
        at java.lang.Thread.run(Thread.java:744)
java.lang.ArithmeticException: / by zero
        at ExecuteSubmitDemo$1.run(ExecuteSubmitDemo.java:15)
        at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1145)
        at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:615)
        at java.lang.Thread.run(Thread.java:744)
```

2 : execute () submit () . `service.submit(new Runnable(){` run () catch .

:

```
creating service
a and b=4:0
a and b=4:0
```

3 : newFixedThreadPool ExtendedExecutor

```
//ExecutorService service = Executors.newFixedThreadPool(2);
 ExtendedExecutor service = new ExtendedExecutor();
```

:

```
creating service
a and b=4:0
java.lang.ArithmeticException: / by zero
a and b=4:0
java.lang.ArithmeticException: / by zero
```

. ThreadPoolExecutor Exectpion ThreadPoolExecutor .

: ExecutorService submit submit () Future Future () get () API . afterExecute . ThreadPoolExecutor : ThreadPoolExecutor .

1. ExecutorService

   `ExecutorService executor = Executors.newFixedThreadPool(50);`

   `.ThreadPoolExecutor .`

   `Callable/Runnable .CPU/Memory RejectedExecutionHandler` & `RejectedExecutionHandler ThreadPoolExecutor .`

2. CountDownLatch

   `CountDownLatch . countDown() .` 0 `await() await().await()` 0 . , *Java thread, thread ().*

   :

   1. :

2. N .

3. .

3. ThreadPoolExecutor : . Runnable / Callable . RejectionHandler . Java `RejectedExecutionHandler` .

1. `ThreadPoolExecutor.AbortPolicy` , RejectedExecutionException throw.

2. ThreadPoolExecutor.CallerRunsPolicy` execute . .

3. `ThreadPoolExecutor.DiscardPolicy` , .

4. `ThreadPoolExecutor.DiscardOldestPolicy` executor ( ).

`CountDownLatch` `invokeAll()` .

4. ForkJoinPool.

`ForkJoinPool` Java 7 Java . `ForkJoinPool` Java `ExecutorService` . `ForkJoinPool` `ForkJoinPool` . `ForkJoinPool` .

Java 8 ExecutorService API . `RecursiveTask` `RecursiveAction` `ForkJoinPool` .

```
public static ExecutorService newWorkStealingPool()
```

.

CPU .

4 . .

ExecutorService .

Executor .

1. ExecutorService `invokeAll()`

, Future .

:

```
import java.util.concurrent.*;
import java.util.*;

public class InvokeAllDemo{
    public InvokeAllDemo(){
        System.out.println("creating service");
        ExecutorService service =
Executors.newFixedThreadPool(Runtime.getRuntime().availableProcessors());

        List<MyCallable> futureList = new ArrayList<MyCallable>();
```

```
        for (int i = 0; i < 10; i++){
            MyCallable myCallable = new MyCallable((long)i);
            futureList.add(myCallable);
        }
        System.out.println("Start");
        try{
            List<Future<Long>> futures = service.invokeAll(futureList);
        } catch(Exception err){
            err.printStackTrace();
        }
        System.out.println("Completed");
        service.shutdown();
    }
    public static void main(String args[]){
        InvokeAllDemo demo = new InvokeAllDemo();
    }
    class MyCallable implements Callable<Long>{
        Long id = 0L;
        public MyCallable(Long val){
            this.id = val;
        }
        public Long call(){
            // Add your business logic
            return id;
        }
    }
}
```

2. CountDownLatch

   .

   CountDownLatch . await `countDown()` 0 . . . . **CyclicBarrier** .

3. Executors ForkJoinPool `newWorkStealingPool()`

4. `ExecutorService` `Future` .

5. ExecutorService :

```
void shutdownAndAwaitTermination(ExecutorService pool) {
    pool.shutdown(); // Disable new tasks from being submitted
    try {
      // Wait a while for existing tasks to terminate
      if (!pool.awaitTermination(60, TimeUnit.SECONDS)) {
        pool.shutdownNow(); // Cancel currently executing tasks
        // Wait a while for tasks to respond to being cancelled
        if (!pool.awaitTermination(60, TimeUnit.SECONDS))
            System.err.println("Pool did not terminate");
      }
    } catch (InterruptedException ie) {
      // (Re-)Cancel if current thread also interrupted
      pool.shutdownNow();
      // Preserve interrupt status
      Thread.currentThread().interrupt();
    }
```

`shutdown()`: .

`shutdownNow()`: .

while

```
if (!pool.awaitTermination(60, TimeUnit.SECONDS))
```

```
while(!pool.awaitTermination(60, TimeUnit.SECONDS)) {
  Thread.sleep(60000);
```

}

ExecutorService

Executors ThreadPool .

1. `public static ExecutorService newSingleThreadExecutor()`

   thread Executor .

   `newFixedThreadPool(1)` `newSingleThreadExecutor()` java .

   newFixedThreadPool (1) , executor, thread () .

   , `newFixedThreadPool` . `((ThreadPoolExecutor) fixedThreadPool).setMaximumPoolSize(10)` `newSingleThreadExecutor` .

   :

   1. .
   2. .

   :

   1. .

2. `public static ExecutorService newFixedThreadPool(int nThreads)`

   . nThreads . .

   :

   1. . `nThreads` `Runtime.getRuntime().availableProcessors()`
   2.

   :

   1. .

3. `public static ExecutorService newCachedThreadPool()`

thread thread , thread

:

1.

:

1. .
2. . . : `newFixedThreadPool`

4. `public static ScheduledExecutorService newScheduledThreadPool(int corePoolSize)`

.

:

1. ( )

:

1. .

5. `public static ExecutorService newWorkStealingPool()`

.

:

1. .
2. . .

:

1. .

ExecutorService , . ThreadPoolExecutor .

```
ThreadPoolExecutor(int corePoolSize, int maximumPoolSize, long keepAliveTime,
TimeUnit unit, BlockingQueue<Runnable> workQueue, ThreadFactory threadFactory,
RejectedExecutionHandler handler)
```

`ThreadPoolExecutor`

1.
2. `BlockingQueue`
3. `RejectionExecutionHander` .
4. Thread `CustomThreadFactory (public Thread newThread(Runnable r)`

`ExecutorService` .

.

`submit`	.
`execute`	.
`invokeAll`	.
`invokeAny`	().

`shutdown()` `shutdown()` . . `awaitTermination` `isShutdown()` .

Executor, ExecutorService Thread : https://riptutorial.com/ko/java/topic/143/executor--executorservice--thread-

10: FileUpload to AWS

Spring Rest API AWS s3 .

Examples

s3

Java rest API S3 APi .

: - s3 .

: - DocumentController.java

```
@RestController
@RequestMapping("/api/v2")
public class DocumentController {

    private static String bucketName = "pharmerz-chat";
    //   private static String keyName        = "Pharmerz"+ UUID.randomUUID();

    @RequestMapping(value = "/upload", method = RequestMethod.POST, consumes =
MediaType.MULTIPART_FORM_DATA)
    public URL uploadFileHandler(@RequestParam("name") String name,
                                 @RequestParam("file") MultipartFile file) throws IOException
{

/******* Printing all the possible parameter from @RequestParam *************/

        System.out.println("*****************************");

        System.out.println("file.getOriginalFilename() " + file.getOriginalFilename());
        System.out.println("file.getContentType()" + file.getContentType());
        System.out.println("file.getInputStream() " + file.getInputStream());
        System.out.println("file.toString() " + file.toString());
        System.out.println("file.getSize() " + file.getSize());
        System.out.println("name " + name);
        System.out.println("file.getBytes() " + file.getBytes());
        System.out.println("file.hashCode() " + file.hashCode());
        System.out.println("file.getClass() " + file.getClass());
        System.out.println("file.isEmpty() " + file.isEmpty());

        /*************Parameters to b pass to s3 bucket put Object **************/
        InputStream is = file.getInputStream();
        String keyName = file.getOriginalFilename();


// Credentials for Aws
        AWSCredentials credentials = new BasicAWSCredentials("AKIA*************",
"zr**********************");

        /****************** DocumentController.uploadfile(credentials);
*************************/
```

```
        AmazonS3 s3client = new AmazonS3Client(credentials);
        try {
            System.out.println("Uploading a new object to S3 from a file\n");
            //File file = new File(awsuploadfile);
            s3client.putObject(new PutObjectRequest(
                    bucketName, keyName, is, new ObjectMetadata()));


            URL url = s3client.generatePresignedUrl(bucketName, keyName,
Date.from(Instant.now().plus(5, ChronoUnit.MINUTES)));
            // URL url=s3client.generatePresignedUrl(bucketName,keyName,
Date.from(Instant.now().plus(5, ChronoUnit.)));
            System.out.println("************************************");
            System.out.println(url);

            return url;

        } catch (AmazonServiceException ase) {
            System.out.println("Caught an AmazonServiceException, which " +
                    "means your request made it " +
                    "to Amazon S3, but was rejected with an error response" +
                    " for some reason.");
            System.out.println("Error Message:    " + ase.getMessage());
            System.out.println("HTTP Status Code: " + ase.getStatusCode());
            System.out.println("AWS Error Code:   " + ase.getErrorCode());
            System.out.println("Error Type:       " + ase.getErrorType());
            System.out.println("Request ID:       " + ase.getRequestId());
        } catch (AmazonClientException ace) {
            System.out.println("Caught an AmazonClientException, which " +
                    "means the client encountered " +
                    "an internal error while trying to " +
                    "communicate with S3, " +
                    "such as not being able to access the network.");
            System.out.println("Error Message: " + ace.getMessage());
        }

        return null;

    }


}
```

```
var form = new FormData();
form.append("file", "image.jpeg");

var settings = {
 "async": true,
 "crossDomain": true,
 "url": "http://url/",
 "method": "POST",
 "headers": {
   "cache-control": "no-cache"
 },
 "processData": false,
 "contentType": false,
 "mimeType": "multipart/form-data",
 "data": form
}
```

```
$.ajax(settings).done(function (response) {
 console.log(response);
});
```

FileUpload to AWS : https://riptutorial.com/ko/java/topic/10589/fileupload-to-aws

11: FTP ( )

- FTPClient connect (InetAddress , int)
- FTPClient (,)
- FTPClient disconnect ()
- FTPReply getReplyStrings ()
- storeFile (String remote, InputStream local)
- OutputStream storeFileStream (String)
- setFileType (int fileType)
- completePendingCommand ()

	FTP IP
	FTP
	FTP
	FTP

Examples

FTP

Java FTP `FTPClient .connect(String server, int port) .login(String username, String password)` .

```
import java.io.IOException;
import org.apache.commons.net.ftp.FTPClient;
import org.apache.commons.net.ftp.FTPReply;
//Import all the required resource for this project.

public class FTPConnectAndLogin {
    public static void main(String[] args) {
        // SET THESE TO MATCH YOUR FTP SERVER //
        String server = "www.server.com"; //Server can be either host name or IP address.
        int port = 21;
        String user = "Username";
        String pass = "Password";

        FTPClient ftp = new FTPClient;
        ftp.connect(server, port);
        ftp.login(user, pass);
    }
}
```

. ? . .

```
try {
```

```
    ftp.connect(server, port);
    showServerReply(ftp);
    int replyCode = ftp.getReplyCode();
    if (!FTPReply.isPositiveCompletion(replyCode)) {
        System.out.printIn("Operation failed. Server reply code: " + replyCode)
        return;
    }
    ftp.login(user, pass);
} catch {

}
```

.

```
showServerReply(ftp);
```

.

```
int replyCode = ftp.getReplyCode();
```

/ .

```
if (!FTPReply.isPositiveCompletion(replyCode)) {
    System.out.printIn("Operation failed. Server reply code: " + replyCode)
    return;
}
```

. , " : :" . try / catch . , `ftp.login()` .

```
boolean success = ftp.login(user, pass);
showServerReply(ftp);
if (!success) {
    System.out.println("Failed to log into the server");
        return;
    } else {
        System.out.println("LOGGED IN SERVER");
    }
```

.

```
boolean success = ftp.login(user, pass);
```

FTP .

```
showServerReply(ftp);
```

.

```
if (!success) {
System.out.println("Failed to log into the server");
    return;
} else {
```

```
    System.out.println("LOGGED IN SERVER");
}
```

. "", " ". .

```
import java.io.IOException;
import org.apache.commons.net.ftp.FTPClient;
import org.apache.commons.net.ftp.FTPReply;

public class FTPConnectAndLogin {
    public static void main(String[] args) {
        // SET THESE TO MATCH YOUR FTP SERVER //
        String server = "www.server.com";
        int port = 21;
        String user = "username"
        String pass = "password"

        FTPClient ftp = new FTPClient
        try {
            ftp.connect(server, port)
            showServerReply(ftp);
            int replyCode = ftpClient.getReplyCode();
            if (!FTPReply.isPositiveCompletion(replyCode)) {
                    System.out.println("Operation failed. Server reply code: " + replyCode);
                    return;
                }
            boolean success = ftp.login(user, pass);
            showServerReply(ftp);
            if (!success) {
                System.out.println("Failed to log into the server");
                return;
            } else {
                System.out.println("LOGGED IN SERVER");
            }
        } catch {

        }
    }
}
```

Catch .

```
} catch (IOException ex) {
    System.out.println("Oops! Something went wrong.");
    ex.printStackTrace();
}
```

catch "Oops! Something went wrong" . . `showServerReply()` .

```
private static void showServerReply(FTPClient ftp) {
    String[] replies = ftp.getReplyStrings();
    if (replies != null && replies.length > 0) {
    for (String aReply : replies) {
        System.out.println("SERVER: " + aReply);
        }
    }
}
```

`FTPClient` "ftp". . . "SERVER : [reply]" . . :

```
import java.io.IOException;
import org.apache.commons.net.ftp.FTPClient;
import org.apache.commons.net.ftp.FTPReply;

public class FTPConnectAndLogin {
    private static void showServerReply(FTPClient ftp) {
        String[] replies = ftp.getReplyStrings();
        if (replies != null && replies.length > 0) {
        for (String aReply : replies) {
            System.out.println("SERVER: " + aReply);
            }
        }
    }

    public static void main(String[] args) {
        // SET THESE TO MATCH YOUR FTP SERVER //
        String server = "www.server.com";
        int port = 21;
        String user = "username"
        String pass = "password"

        FTPClient ftp = new FTPClient
        try {
            ftp.connect(server, port)
            showServerReply(ftp);
            int replyCode = ftpClient.getReplyCode();
            if (!FTPReply.isPositiveCompletion(replyCode)) {
                System.out.println("Operation failed. Server reply code: " + replyCode);
                return;
            }
            boolean success = ftp.login(user, pass);
            showServerReply(ftp);
            if (!success) {
                System.out.println("Failed to log into the server");
                return;
            } else {
                System.out.println("LOGGED IN SERVER");
            }
        } catch (IOException ex) {
            System.out.println("Oops! Something went wrong.");
            ex.printStackTrace();
        }
    }
}
```

`FTPClient` `.connect(String server, int port)` `.login(String username, String password)` . try / catch . . " `showServerReply(FTPClient ftp)` ".

```
import java.io.IOException;
import org.apache.commons.net.ftp.FTPClient;
import org.apache.commons.net.ftp.FTPReply;

public class FTPConnectAndLogin {
    private static void showServerReply(FTPClient ftp) {
        if (replies != null && replies.length > 0) {
        for (String aReply : replies) {
            System.out.println("SERVER: " + aReply);
```

```
            }
        }
    }

public static void main(String[] args) {
    // SET THESE TO MATCH YOUR FTP SERVER //
    String server = "www.server.com";
    int port = 21;
    String user = "username"
    String pass = "password"

FTPClient ftp = new FTPClient
try {
    ftp.connect(server, port)
    showServerReply(ftp);
    int replyCode = ftpClient.getReplyCode();
    if (!FTPReply.isPositiveCompletion(replyCode)) {
            System.out.println("Operation failed. Server reply code: " + replyCode);
            return;
        }
    boolean success = ftp.login(user, pass);
    showServerReply(ftp);
    if (!success) {
        System.out.println("Failed to log into the server");
            return;
        } else {
            System.out.println("LOGGED IN SERVER");
        }
    } catch (IOException ex) {
        System.out.println("Oops! Something went wrong.");
        ex.printStackTrace();
        }
    }
}
```

FTP Java .

FTP ( ) : https://riptutorial.com/ko/java/topic/5228/ftp-----

12: Hashtable

Hashtable Map Dictionary Java .

.

Examples

Hashtable

```
import java.util.*;
public class HashtableDemo {
   public static void main(String args[]) {
   // create and populate hash table
   Hashtable<Integer, String> map = new Hashtable<Integer, String>();
   map.put(101,"C Language");
   map.put(102, "Domain");
   map.put(104, "Databases");
   System.out.println("Values before remove: "+ map);
   // Remove value for key 102
   map.remove(102);
   System.out.println("Values after remove: "+ map);
   }
}
```

Hashtable : https://riptutorial.com/ko/java/topic/10709/hashtable

13: HttpURLConnection

- Android HttpUrlConnection `AndroidManifest.xml` .
- Square OkHttp Java HTTP .

Examples

String

```
String getText(String url) throws IOException {
    HttpURLConnection connection = (HttpURLConnection) new URL(url).openConnection();
    //add headers to the connection, or check the status if desired..

    // handle error response code it occurs
    int responseCode = conn.getResponseCode();
    InputStream inputStream;
    if (200 <= responseCode && responseCode <= 299) {
        inputStream = connection.getInputStream();
    } else {
        inputStream = connection.getErrorStream();
    }

    BufferedReader in = new BufferedReader(
        new InputStreamReader(
            inputStream));

    StringBuilder response = new StringBuilder();
    String currentLine;

    while ((currentLine = in.readLine()) != null)
        response.append(currentLine);

    in.close();

    return response.toString();
}
```

URL String .

:

- `new URL(url).openConnection()` URL `HttpUrlConnection` . `HttpUrlConnection` `UrlConnection` (:) . (.)
- `InputStream` ( )
- `InputStream` `BufferedReader` .
- `StringBuilder` .
- `InputStream` , String .

:

- , ( ) `IoException` , URL , `MalformedUrlException` Throw.
- (HTML), JSON XML REST API URL .
- : Java String to String .

:

:

```
String text = getText("http://example.com");
//Do something with the text from example.com, in this case the HTML.
```

```
public static void post(String url, byte [] data, String contentType) throws IOException {
    HttpURLConnection connection = null;
    OutputStream out = null;
    InputStream in = null;

    try {
        connection = (HttpURLConnection) new URL(url).openConnection();
        connection.setRequestProperty("Content-Type", contentType);
        connection.setDoOutput(true);

        out = connection.getOutputStream();
        out.write(data);
        out.close();

        in = connection.getInputStream();
        BufferedReader reader = new BufferedReader(new InputStreamReader(in));
        String line = null;
        while ((line = reader.readLine()) != null) {
            System.out.println(line);
        }
        in.close();

    } finally {
        if (connection != null) connection.disconnect();
        if (out != null) out.close();
        if (in != null) in.close();
    }
}
```

URL POST .

- `URL HttpURLConnection` .
- `setRequestProperty` . `application/x-www-form-urlencoded`
- `setDoOutput(true)` .
- `getOutputStream()` `OutputStream` . .
- .

```
   public static void delete (String urlString, String contentType) throws IOException {
        HttpURLConnection connection = null;
```

```
        try {
            URL url = new URL(urlString);
            connection = (HttpURLConnection) url.openConnection();
            connection.setDoInput(true);
            connection.setRequestMethod("DELETE");
            connection.setRequestProperty("Content-Type", contentType);

            Map<String, List<String>> map = connection.getHeaderFields();
            StringBuilder sb = new StringBuilder();
            Iterator<Map.Entry<String, String>> iterator =
responseHeader.entrySet().iterator();
            while(iterator.hasNext())
            {
                Map.Entry<String, String> entry = iterator.next();
                sb.append(entry.getKey());
                sb.append('=').append('"');
                sb.append(entry.getValue());
                sb.append('"');
                if(iterator.hasNext())
                {
                    sb.append(',').append(' ');
                }
            }
            System.out.println(sb.toString());

        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            if (connection != null) connection.disconnect();
        }
    }
```

URL .

- `URL HttpURLConnection .`
- `setRequestProperty . application/x-www-form-urlencoded`
- `setDoInput(true)` URL .
- HTTP `setRequestMethod("DELETE")`

.

.

```
/**
 * Checks if a resource exists by sending a HEAD-Request.
 * @param url The url of a resource which has to be checked.
 * @return true if the response code is 200 OK.
 */
public static final boolean checkIfResourceExists(URL url) throws IOException {
    HttpURLConnection conn = (HttpURLConnection) url.openConnection();
    conn.setRequestMethod("HEAD");
    int code = conn.getResponseCode();
    conn.disconnect();
    return code == 200;
}
```

:

GET HEAD . .

`200 true true` . (: 3XX) .

:

```
checkIfResourceExists(new URL("http://images.google.com/")); // true
checkIfResourceExists(new URL("http://pictures.google.com/")); // false
```

HttpURLConnection : https://riptutorial.com/ko/java/topic/156/httpurlconnection

14: InputStream OutputStream

- int read (byte [] b) IOException throw.

, `InputStream` , `BufferedStream` `BufferedStream` . `InputStream` read . CPU .

Examples

InputStream String

String . `byte` `char` "native Java"UTF-16 Codepoints . `InputStreamReader` .

"" .

Java SE 7

```
public String inputStreamToString(InputStream inputStream) throws Exception {
     StringWriter writer = new StringWriter();

     char[] buffer = new char[1024];
     try (Reader reader = new BufferedReader(new InputStreamReader(inputStream, "UTF-8"))) {
           int n;
           while ((n = reader.read(buffer)) != -1) {
                // all this code does is redirect the output of `reader` to `writer` in
                // 1024 byte chunks
                writer.write(buffer, 0, n);
           }
     }
     return writer.toString();
}
```

Java SE 6 () .

OutputStream

1 `OutputStream`

```
OutputStream stream = object.getOutputStream();

byte b = 0x00;
stream.write( b );
```

```
byte[] bytes = new byte[] { 0x00, 0x00 };

stream.write( bytes );
```

```
int offset = 1;
int length = 2;
byte[] bytes = new byte[] { 0xFF, 0x00, 0x00, 0xFF };
```

```
stream.write( bytes, offset, length );
```

. . .

Java SE 7

```
try(FileWriter fw = new FileWriter("outfilename");
    BufferedWriter bw = new BufferedWriter(fw);
    PrintWriter out = new PrintWriter(bw))
{
    out.println("the text");
    //more code
    out.println("more text");
    //more code
} catch (IOException e) {
    //handle this however you
}
```

try-with-resources , , .

Java SE 6

Java 6 . `finally` .

```
FileWriter fw = null;
BufferedWriter bw = null;
PrintWriter out = null;
try {
    fw = new FileWriter("myfile.txt");
    bw = new BufferedWriter(fw);
    out = new PrintWriter(bw);
    out.println("the text");
    out.close();
} catch (IOException e) {
    //handle this however you want
}
finally {
    try {
        if(out != null)
            out.close();
    } catch (IOException e) {
        //typically not much you can do here...
    }
}
```

., .

.

```
void copy(InputStream in, OutputStream out) throws IOException {
    byte[] buffer = new byte[8192];
    while ((bytesRead = in.read(buffer)) > 0) {
        out.write(buffer, 0, bytesRead);
    }
}
```

-

```
 // reading from System.in and writing to System.out
copy(System.in, System.out);
```

/

OutputStream InputStream . .

, .

```
File myFile = new File("targetFile.txt");
PrintWriter writer = new PrintWriter(new BufferedOutputStream(new FileOutputStream(myFile)));
```

```
Cipher cipher = ... // Initialize cipher
File myFile = new File("targetFile.enc");
BufferedOutputStream outputStream = new BufferedOutputStream(new DeflaterOutputStream(new
CipherOutputStream(new FileOutputStream(myFile), cipher)));
```

/

BufferedOutputStream / BufferedInputStream	OutputStream BufferedOutputStream . .
DeflaterOutputStream / DeflaterInputStream	.
InflaterOutputStream / InflaterInputStream	.
CipherOutputStream / CipherInputStream	/ .
DigestOutputStream / DigestInputStream	Message Digest .
CheckedOutputStream / CheckedInputStream	CheckSum . CheckSum Message Digest .
DataOutputStream / DataInputStream	. . .
PrintStream	. . .
OutputStreamWriter	OutputStream . Writer OutputStream .
PrintWriter	OutputStreamWriter . . .

DataInputStream

```
package com.streams;
import java.io.*;
public class DataStreamDemo {
  public static void main(String[] args) throws IOException {
    InputStream input = new FileInputStream("D:\\datastreamdemo.txt");
    DataInputStream inst = new DataInputStream(input);
    int count = input.available();
    byte[] arr = new byte[count];
    inst.read(arr);
    for (byte byt : arr) {
      char ki = (char) byt;
      System.out.print(ki+"-");
    }
  }
}
```

InputStream OutputStream : https://riptutorial.com/ko/java/topic/110/inputstream--outputstream

15: Iterator Iterable

`java.util.Iterator` , Iterator Java SE . `java.lang.Iterable` , .

Java Iterable , `for` each . JVM .

Examples

for Iterable

`Iterable<>` `for` . . , .

```
public class UsingIterable {

    public static void main(String[] args) {
        List<Integer> intList = Arrays.asList(1,2,3,4,5,6,7);

        // List extends Collection, Collection extends Iterable
        Iterable<Integer> iterable = intList;

        // foreach-like loop
        for (Integer i: iterable) {
            System.out.println(i);
        }

        // pre java 5 way of iterating loops
        for(Iterator<Integer> i = iterable.iterator(); i.hasNext(); ) {
            Integer item = i.next();
            System.out.println(item);
        }
    }
}
```

foreach ("extended for loop") iterator . .

```
List<String> yourData = //...
Iterator<String> iterator = yourData.iterator();
while (iterator.hasNext()){
    // next() "moves" the iterator to the next entry and returns it's value.
    String entry = iterator.next();
    System.out.print(entry);
    if (iterator.hasNext()){
        // If the iterator has another element after the current one:
        System.out.print(",");
    }
}
```

`isLastEntry` .

.

Iterable . `Iterable iterator()` . `Iterator` 3 . .

```
public static class Alphabet implements Iterable<Character> {

    @Override
    public Iterator<Character> iterator() {
        return new Iterator<Character>() {
            char letter = 'a';

            @Override
            public boolean hasNext() {
                return letter <= 'z';
            }

            @Override
            public Character next() {
                return letter++;
            }

            @Override
            public void remove() {
                throw new UnsupportedOperationException("Doesn't make sense to remove a
letter");
            }
        };
    }
}
```

:

```
public static void main(String[] args) {
    for(char c : new Alphabet()) {
        System.out.println("c = " + c);
    }
}
```

`Iterator` , next `.hasNext()` . , `remove()` , , .

`Iterator.remove()` `Iterator.next()` . , .

```
List<String> names = new ArrayList<>();
names.add("name 1");
names.add("name 2");
names.add("");
names.add("name 3");
names.add("");
System.out.println("Old Size : " + names.size());
Iterator<String> it = names.iterator();
while (it.hasNext()) {
  String el = it.next();
  if (el.equals("")) {
    it.remove();
  }
}
System.out.println("New Size : " + names.size());
```

:

```
Old Size : 5
```

```
New Size : 3
```

. .

```
for (String el: names) {
    if (el.equals("")) {
        names.remove(el); // WRONG!
    }
}
```

(: `ArrayList`) `ConcurrentModificationException` .

`remove()` `next()` ( ). `next()` `next()` `remove()` `IllegalStateException` .

`remove` ., . , . `remove()` `UnsupportedOperationException` .

Iterator Iterable : https://riptutorial.com/ko/java/topic/172/iterator--iterable

16: Java (Standard Edition)

Windows , Linux macOS java standard edition .

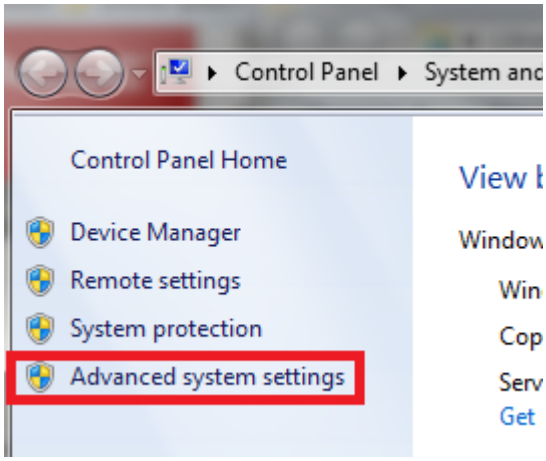
Examples

Windows % PATH % % JAVA_HOME %

:

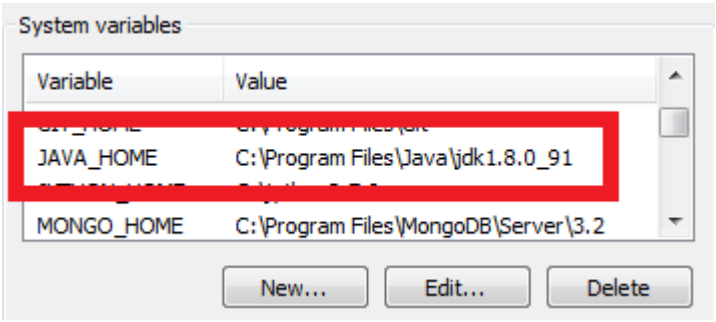
- Oracle JDK .
- JDK .

1. Windows .
2. *This PC* (Windows) . Windows . Win + Pause .
3. . .



Run (Win + R) SystemPropertiesAdvanced ( ) Enter .

4. ... .
5. JAVA_HOME JDK ... . .



6. Path .
7.

: Windows `Path` . Windows . Windows JDK . `Path` "Edit ..." .
`%JAVA_HOME%\bin;` `%JAVA_HOME%\bin; Path` .

Java `Path` . .

1. `cmd` `Enter` .
2. `javac -version` . JDK .

: . `Path` .

Java SE

1995 Java 1.0 Java . (Java .) . (Oracle) / . (/ .)

Java SE . Java 8 . Java 9 2017 . (Java 7 End of Life 2015 4 . Java 7 .)

Java Java . Java . Java Java . (/ / .)

Java Java . Java . Java .

Java

. .

JDK	
jdk-1.0	JDK 1.0
jdk-1.1	JDK 1.1
JDK-1.2	J2SE 1.2
...	...
jdk-1.5	J2SE 1.5 Java SE 5
JDK 1.6	Java SE 6
jdk-1.7	Java SE 7
jdk-1.8	Java SE 8
jdk-9 [1]	Java SE 9 ( )

1 - Java " " . .

"SE" Standard Edition . , PC Java (Android).

Java . "Java ME" Micro Edition "Java EE" Enterprise Edition. Java Android Java SE . Java ME, Java EE Android Java .

Java .

```
1.8.0_101-b13
```

JDK 1.8.0, Update 101, Build # 13 . Oracle .

```
Java™ SE Development Kit 8, Update 101 (JDK 8u101)
```

. , ( ) . . 8 1.8 . Java 8 Java 1.8 "" .

JDK Java . (, .)

.

- Eclipse, Intellij IDEA NetBeans Java IDE
- Ant, Gradle Maven Java
- ( )
- CI ( )

Linux Java JDK

OpenJDK Oracle JDK / JRE Linux . (.)

. ("root" ... `sudo` . .)

() Java .

`apt-get` , ( )

Oracle Java 8 .

```
$ sudo add-apt-repository ppa:webupd8team/java
$ sudo apt-get update
$ sudo apt-get install oracle-java8-installer
```

: Java 8 .

```
$ sudo apt-get install oracle-java8-set-default
```

`.deb`

Oracle `.tar.gz` `.deb` `.tar.gz` `./<jdk>.tar.gz` .

```
$ sudo apt-get install java-package # might not be available in default repos
$ make-jpkg ./<jdk>.tar.gz          # should not be run as root
$ sudo dpkg -i *j2sdk*.deb
```

: ".tar.gz" .

`slackpkg` , Slackware

```
sudo slapt-get install default-jdk
```

`yum` , , CentOS

```
sudo yum install java-1.8.0-openjdk-devel.x86_64
```

`dnf` ,

Fedora `yum dnf` .

```
sudo dnf install java-1.8.0-openjdk-devel.x86_64
```

Fedora Java 7 .

`pacman` ,

```
sudo pacman -S jdk8-openjdk
```

`sudo` .

Gentoo Java Gentoo Java portage USE wiki .

Redhat, CentOS, Fedora Oracle JDK

Oracle JDK JRE `tar.gz` JDK .

1. Oracle Java Oracle ("tar.gz") .
2. .
3. .

```
tar xzvf jdk-8u67-linux-x64.tar.gz
```

Oracle Java RPM .

1. Oracle Java RPM .
2. `rpm` . :

```
$ sudo rpm -ivh jdk-8u67-linux-x644.rpm
```

Windows Java JDK JRE

Windows Oracle JDK JRE . .

1. Oracle Java .
2. JDK , JRE Server JRE . Java JDK . JDK JRE .
3. . .
4. "Accept License Agreement" .
5. Windows x86 (32) Windows x64 (64) .
6. ... Windows .

Windows Java Chocolately .

1. https://chocolatey.org/ Chocolately
2. cmd . `Win` + `R` "" "cmd" .
3. cmd Java 8 JDK .

```
C:\> choco install jdk8
```

VM JDK / JRE JDK / JRE (x64 / x86) . (.EXE) . (JDK / JRE 7).

1. Java 2 .
2. 7-Zip .
3. 7-Zip Java EXE .
4. `Shift` `Right-Click` .
5. . `jdk-7u25-windows-x64` . `cd jdk-7u25-windows-x64` . .

   ```
   cd .rsrc\JAVA_CAB10

   extrac32 111
   ```

6. `tools.zip` . 7-Zip `tools.zip` `tools` .
7. .

   ```
   cd tools

   for /r %x in (*.pack) do .\bin\unpack200 -r "%x" "%~dx%~px%~nx.jar"
   ```

8. . `tools` .

JDK / JRE .

: http://stackoverflow.com/a/6571736/1448252

MacOS JDK

Oracle Java 7 Java 8

MacOS Java 7 Java 8 Oracle . Oracle Mac Java . 7u25 Java 7 Apple .

Oracle Java (7) MacOS 10.7.3 Intel Mac .

Oracle Java

macOS Java 7 & 8 JDK JRE Oracle .

- Java 8 - Java SE
- Java 7 - Oracle Java Archive.

. JDK :

```
/Library/Java/JavaVirtualMachines/<version>.jdk/Contents/Home
```

.

Java . .

```
export JAVA_HOME=/usr/libexec/java_home -v 1.6 #Or 1.7 or 1.8
```

`~/.bash_profile` (Bash) .

```
function java_version {
   echo 'java -version';
}

function java_set {
   if [[ $1 == "6" ]]
   then
       export JAVA_HOME='/usr/libexec/java_home -v 1.6';
       echo "Setting Java to version 6..."
       echo "$JAVA_HOME"
   elif [[ $1 == "7" ]]
   then
       export JAVA_HOME='/usr/libexec/java_home -v 1.7';
       echo "Setting Java to version 7..."
       echo "$JAVA_HOME"
   elif [[ $1 == "8" ]]
   then
       export JAVA_HOME='/usr/libexec/java_home -v 1.8';
       echo "Setting Java to version 8..."
       echo "$JAVA_HOME"
   fi
}
```

MacOS Apple Java 6

MacOS (10.11 El Capitan) Apple Java 6 . .

```
/System/Library/Java/JavaVirtualMachines/1.6.0.jdk/Contents/Home
```

Java 6 . Oracle Apple Java 6 .

Linux Java

Linux `alternatives` . Java .

1. $ JDK JDK .

```
$ JDK=/Data/jdk1.8.0_67
```

2. `alternatives --install` Java SDK `alternatives --install` .

```
 $ sudo alternatives --install /usr/bin/java java $JDK/bin/java 2
 $ sudo alternatives --install /usr/bin/javac javac $JDK/bin/javac 2
 $ sudo alternatives --install /usr/bin/jar jar $JDK/bin/jar 2
```

.

Java .

```
$ sudo alternatives --config javac

There is 1 program that provides 'javac'.

  Selection    Command
-----------------------------------------------
*+ 1           /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.101-1.b14.fc23.x86_64/bin/javac
   2           /Data/jdk1.8.0_67/bin/javac

Enter to keep the current selection[+], or type selection number: 2
$
```

`alternatives` (8) .

Arch Linux `archlinux-java` Java .

```
$ archlinux-java status
Available Java environments:
  java-7-openjdk (default)
  java-8-openjdk/jre
```

```
# archlinux-java set <JAVA_ENV_NAME>
```

:

```
# archlinux-java set java-8-openjdk/jre
```

.

Linux

Java SDK . .

```
$ java -version
$ javac -version
```

JRE JDK . JDK / JRE .

- "command not found" JRE JDK . **PATH** .
- " " . .
- . **JAVA_HOME** .

PATH

`java` `javac` .

Java . **Java** .

, `bash` `~/.bash_profile` `~/.bashrc` (Bash) .

```
JAVA_HOME=<installation directory>
PATH=$JAVA_HOME/bin:$PATH

export JAVA_HOME
export PATH
```

... `<installation directory>` Java . `bin` , `bin` `java` `javac` .

, .

```
$ source ~/.bash_profile
```

, `java` `javac` . , `which java` `which javac` .

ptopagate. .

`java -version` `javac -version` . `which` `ls -l` .

```
$ ls -l `which java`
```

:

```
lrwxrwxrwx. 1 root root 22 Jul 30 22:18 /usr/bin/java -> /etc/alternatives/java
```

`alternatives` . `PATH` `PATH` .

- Linux Java
-

"PATH " .

Java ?

Java .

- Oracle RPM Java "/ usr / java" .
- Fedora "/ usr / lib / jvm".
- Java ZIP JAR .

`find` (`slocate`) . :

```
$ find / -name java -type f 2> /dev/null
```

`java` . "/ dev / null" .

tar Linux oracle java

tar Oracle JDK .

1. tar - Java SE Development Kit 8u112 .
2. sudo .

   ```
   sudo su
   ```

3. jdk dir :

   ```
   mkdir /opt/jdk
   ```

4. tar .

   ```
   tar -zxf jdk-8u5-linux-x64.tar.gz -C /opt/jdk
   ```

5. .

   ```
   ls /opt/jdk
   ```

6. Oracle JDK JVM :

   ```
   update-alternatives --install /usr/bin/java java /opt/jdk/jdk1.8.0_05/bin/java 100
   ```

   ```
   update-alternatives --install /usr/bin/javac javac /opt/jdk/jdk1.8.0_05/bin/javac 100
   ```

7. Java :

:

```
java version "1.8.0_111"
Java(TM) SE Runtime Environment (build 1.8.0_111-b14)
Java HotSpot(TM) 64-Bit Server VM (build 25.111-b14, mixed mode)
```

Java (Standard Edition) : https://riptutorial.com/ko/java/topic/4754/java--standard-edition--

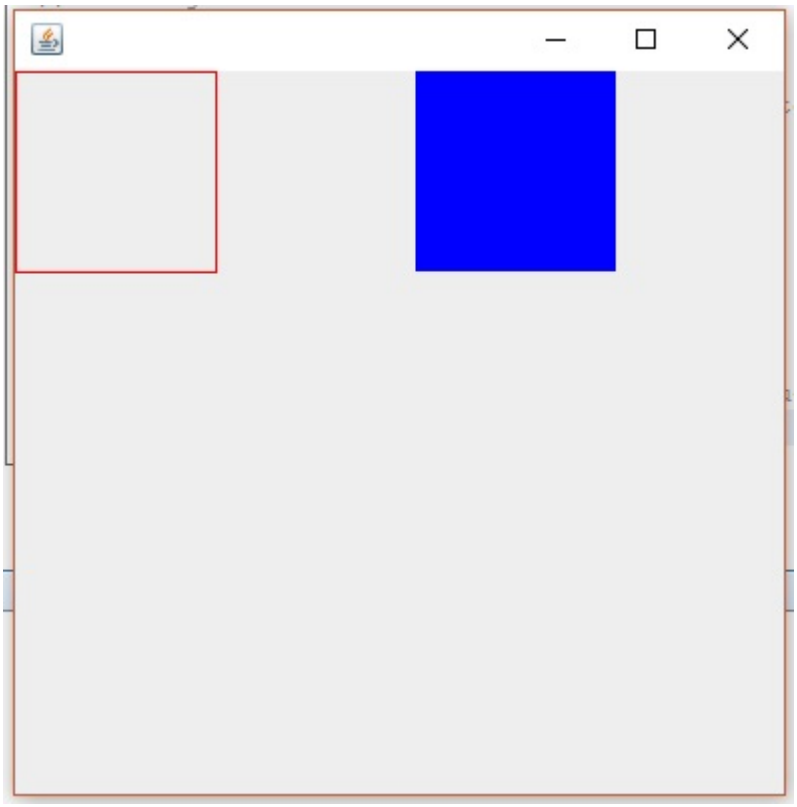
17: Java 2D

, , , . , , . .

Java 2D API . Java 2D .

Examples

1 :



. https://i.stack.imgur.com/dlC5v.jpg

Graphics 2 .

1. ,
2.

: . . MyPanel Test. MyPanel drawRect () & fillRect () mathods Color . setColor (Color.blue) . . p = new MyPanel () MyPanel . Test Rectangle Blue Color Filled Rectangle .

: MyPanel

```
import javax.swing.*;
import java.awt.*;
// MyPanel extends JPanel, which will eventually be placed in a JFrame
public class MyPanel extends JPanel {
    // custom painting is performed by the paintComponent method
    @Override
```

```
    public void paintComponent(Graphics g){
        // clear the previous painting
        super.paintComponent(g);
        // cast Graphics to Graphics2D
        Graphics2D g2 = (Graphics2D) g;
        g2.setColor(Color.red); // sets Graphics2D color
        // draw the rectangle
        g2.drawRect(0,0,100,100); // drawRect(x-position, y-position, width, height)
        g2.setColor(Color.blue);
        g2.fillRect(200,0,100,100); // fill new rectangle with color blue
    }
}
```

2 :

```
import javax.swing.;
import java.awt.;
public class Test { //the Class by which we display our rectangle
    JFrame f;
    MyPanel p;
    public Test(){
        f = new JFrame();
        // get the content area of Panel.
        Container c = f.getContentPane();
        // set the LayoutManager
        c.setLayout(new BorderLayout());
        p = new MyPanel();
        // add MyPanel object into container
        c.add(p);
        // set the size of the JFrame
        f.setSize(400,400);
        // make the JFrame visible
        f.setVisible(true);
        // sets close behavior; EXIT_ON_CLOSE invokes System.exit(0) on closing the JFrame
        f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

    public static void main(String args[ ]){
        Test t = new Test();
    }
}
```

https://docs.oracle.com/javase/tutorial/uiswing/layout/border.html .

paintComponent ()

- .
- .
- , (,)

Graphic2D Graphic2D .

: Java 2D API .

- , .
- .
-

.

- .
- .

2 :

```
import javax.swing.*;
import java.awt.*;

public class MyPanel extends JPanel {
    @Override
    public void paintComponent(Graphics g){
        // clear the previous painting
        super.paintComponent(g);
        Graphics2D g2 = (Graphics2D)g;
        g2.setColor(Color.blue);
        g2.drawOval(0, 0, 20,20);
        g2.fillOval(50,50,20,20);
    }
}
```

g2.drawOval (int x, int y, int height, int width);
x y .

g2.fillOval (int x, int y, int height, int width); x y .

Java 2D : https://riptutorial.com/ko/java/topic/10127/java-2d-

18: Java Editions, ,

Examples

Java SE JRE Java SE JDK

Java SE Sun / Oracle JRE JDK . , JRE Java JDK Java .

Java Runtime Environment JRE Java . JRE .

- JVM (Java Virtual Machine) Java `java`
- Nashorn Javascript `jjs` .
- Java `keytool` .
- `policytool` .
- "pack200" `pack200` `unpack200` .
- Java CORBA RMI `orbd` , `rmid` , `rmiregistry` `tnameserv`

" JRE" Java . "Server JRE" .linux syscall read benchmarku

Java 7 update 6 JRE JavaFX (2.2) .

Java Development Kit JDK JRE Java . .

- Java (".java") (".class") `javac` .
- `jar` `jarsigner` JAR
- :
  - `appletviewer` `appletviewer`
  - `idlj` CORBA IDL to Java
  - `javah` JNI
  - Java `native2ascii`
  - `schemagen` Java XML (JAXB)
  - `serialver` Java Object Serialization .
  - JAX-WS `wsgen` `wsimport`
- :
  - `jdb`
  - `jmap` `jhat` Java .
  - `jstack` .
  - ".class" `javap` .
- :
  - `jconsole` ,
  - `jstat` , `jstatd` , `jinfo` `jps`

Sun / Oracle JDK Java ZIP . Java 6 Java .

Java 6 OpenJDK OpenJDK . (Linux) JDK .

Oracle OpenJDK ?

JRE JDK Java .

- Oracle Oracle .
- OpenJDK OpenJDK ( ) .

OpenJDK . () "" OpenJDK .

OpenJDK Hotspot Hotspot . OpenJDK . , OpenJDK QA .

(fipside) Linux . , Linux Java .

Java EE, Java SE, Java ME JavaFX

Java . Java . Java Java .

Java . , Java .

Java

Java .

- Java , Standard Edition (Java SE)
- , (Java EE)
- , (Java ME)
- Java FX

Java Java Virtual Machine (VM) API (Application Programming Interface) . Java Virtual Machine Java . API . Java API , , , Java . .

Java SE

Java Java SE API . Java SE API Java . Java , , , (GUI) XML .

API Java SE , , Java .

Java EE

Java EE Java SE . Java EE , , API .

Java ME

Java ME Java API . API Java SE API . Java ME Java EE .

Java FX

Java FX Java FX ScriptTM . Java FX Script Java , Java VM . Java FX Java Java EE .

-

Java SE

Java SE

Java SE .

Java SE [1]		([2])	
Java SE 9 (Early Access)			2017-07-27 ()
Java SE 8			2014-03-18
Java SE 7		2015-04-14	2011 7 28
Java SE 6		2013-04-16	2006-12-23
Java SE 5		2009-11-04	2004-10-04
Java SE 1.4.2		2009-11-04	2003-06-26
Java SE 1.4.1	/	2009-11-04	2002-09-16
Java SE 1.4		2009-11-04	2002-02-06
Java SE 1.3.1		2009-11-04	2001-05-17
Java SE 1.3		2009-11-04	2000-05-08
Java SE 1.2		2009-11-04	1998-12-08
Java SE 1.1		2009-11-04	1997-02-19
Java SE 1.0		2009-11-04	1996-01-21

:

1. . Oracle Java Archives .
2. SE "end of life" . Java Oracle . .

:

- Mind Products Roedy Green JDK .

Java SE

Java SE	
Java SE 8	MapReduce . Nashorn Javascript . . . API. JNI JavaFX . PermGen .
Java SE 7	, *try-with-resource* , (`<>`), / . . . . ECC . 2D (GPU) . .
Java SE 6	JVM . API Rhino . JDBC 4.0. API. JAXB 2.0. (JAX-WS)
Java SE 5	, , , `enum` , varargs, `for` Java . RMI . `java.util.concurrent.*` `Scanner` .
Java SE 1.4	`assert` . . . NIO API - I / O, `Buffer` `Channel` . `java.util.logging.*` API. I / O API. XML XSLT (JAXP). (JCE, JSSE, JAAS) Java Web Start. API.
Java SE 1.3	HotSpot JVM . CORBA / RMI . JNDI (Java Naming and Directory Interface). (JPDA). JavaSound API. API.
Java SE 1.2	`strictfp` . API. ( ). CORBA . .
Java SE 1.1	. . JDBC. RMI. / . . AWT . JavaBeans.

:

- Wikipedia : Java

Java Editions, , : https://riptutorial.com/ko/java/topic/8973/java-editions------

19: Java Pitfalls - Null NullPointerException

`null` , .

`NullPointerException` (NPE) `null` throw . .

- `null` ()
- `null` ,
- `null` `null` ,
- `synchronized null` ,
- `null` .
- `null` ,
- `null` .

NPE .

- ,
-
- `null` API .

`null` .

- `Map` API `get(key)` `null` .
- `ClassLoader` `Class` API `getResource(path)` `getResourceAsStream(path)` `null` .
- , `Reference` API `get()` `null` `null` .
- , Java EE API `getXxxx` `null` .

`null` "Yoda Notation" NPE .

Examples

Pitfall - Primitive Wrappers NullPointerExceptions .

Java . . .

```
public class MyRecord {
    public int a, b;
    public Integer c, d;
}

...
MyRecord record = new MyRecord();
record.a = 1;               // OK
record.b = record.b + 1;    // OK
record.c = 1;               // OK
record.d = record.d + 1;    // throws a NullPointerException
```

`MyRecord` 1 . `new` `a` `b` 0 `c` `d` `null` .

`int`

Integer .,(d) , null NullPointerException .

.

- c d , , null . null .
- . .

.

1- . , . .

Pitfall - null .

null . . .

.

```
/**
 * Sum the values in an array of integers.
 * @arg values the array to be summed
 * @return the sum
 **/
public int sum(int[] values) {
    int sum = 0;
    for (int value : values) {
        sum += value;
    }
    return sum;
}
```

null .

```
/**
 * Sum the values in an array of integers.
 * @arg values the array to be summed, or null.
 * @return the sum, or zero if the array is null.
 **/
public int sum(int[] values) {
    int sum = 0;
    if (values != null) {
        for (int value : values) {
            sum += value;
        }
    }
    return sum;
}
```

. null .

null) . null . null NullPointerException . null . .

.

```
int[] values = new int[0];                      // always empty
List<Integer> list = new ArrayList();           // initially empty
List<Integer> list = Collections.emptyList();  // always empty
```

, .

- ""

StackOverflow Answers .

```
public String joinStrings(String a, String b) {
    if (a == null) {
        a = "";
    }
    if (b == null) {
        b = "";
    }
    return a + ": " + b;
}
```

`NullPointerException` `null` "" .

? : .

`joinStrings` .

"a" "b" null ?

`String` 0 . `null` "" ? .

null ?

`null` , . . .

null "" " " ?

`null` . "". Java 8 `Optional` .

( ) ""?

`null` "" . ? `NullPointerException` ? .

"" .

?

"make good" ""null . . " " .

null , null . null , javadoc null .

null NullPointerException .

Pitfall - throw null .

. .

```
public Reader getReader(String pathname) {
    try {
        return new BufferedReader(FileReader(pathname));
    } catch (IOException ex) {
        System.out.println("Open failed: " + ex.getMessage());
        return null;
    }
```

}

?

getReader Reader null . null . NullPointerException .

.

1. IOException .
2. .
3. 「」 Reader , null .

, null .

1. NullReader . "" API .
2. Java 8 getReader Optional<Reader> .

Pitfall - I / O

. catch try - catch - finally .

```
void writeNullBytesToAFile(int count, String filename) throws IOException {
    FileOutputStream out = null;
    try {
        out = new FileOutputStream(filename);
        for(; count > 0; count--)
            out.write(0);
    } finally {
        out.close();
    }
}
```

. out (out = new FileOutputStream(filename)) out null out.close() NullPointerException !

null .

```
void writeNullBytesToAFile(int count, String filename) throws IOException {
    FileOutputStream out = null;
    try {
        out = new FileOutputStream(filename);
        for(; count > 0; count--)
            out.write(0);
    } finally {
        if (out != null)
            out.close();
    }
}
```

-with-resource `try . finally` NPE 0 .

```
void writeNullBytesToAFile(int count, String filename) throws IOException {
    try (FileOutputStream out = new FileOutputStream(filename)) {
        for(; count > 0; count--)
            out.write(0);
    }
}
```

Pitfall - "Yoda " NullPointerException

StackOverflow .

```
if ("A".equals(someString)) {
    // do something
}
```

`someString null NullPointerException "" "". ,`

```
    "A".equals(someString)
```

:

```
    someString != null && someString.equals("A")
```

(. , .)

`NullPointerExceptions` Yoda .

`"A".equals(someString) someString null " ".` (Pitfall - " " `null`) , "" `null` .

" " [1] . `null` `NullPointerException` ( ) . / `null` .

Yoda `null` API `null` . , . `null` .

---

1 - (Wikipedia) " , ." . Yoda . .

Java Pitfalls - Null NullPointerException : https://riptutorial.com/ko/java/topic/5680/java-pitfalls---null--nullpointerexception

20: Java SE 7

Java SE 7 Java . JDBC JVM (Java Virtual Machine) . .

Java SE 7

Examples

Java SE 7

- 2 : (byte, short, int long) . 2 0b 0B .
- switch : switch String .
- try-with-resources : try-with-resources try . . try-with-resources . java.io.Closeable java.lang.AutoCloseable .
- : catch . .
- : (_) . .
- : (<>) . .
- Varargs

```
// An 8-bit 'byte' value:
byte aByte = (byte)0b00100001;

// A 16-bit 'short' value:
short aShort = (short)0b1010000101000101;

// Some 32-bit 'int' values:
int anInt1 = 0b10100001010001011010000101000101;
int anInt2 = 0b101;
int anInt3 = 0B101; // The B can be upper or lower case.

// A 64-bit 'long' value. Note the "L" suffix:
long aLong = 0b1010000101000101101000010100010110100001010001011010000101000101L;
```

. BufferedReader . BufferedReader .

```
static String readFirstLineFromFile(String path) throws IOException {
  try (BufferedReader br = new BufferedReader(new FileReader(path))) {
    return br.readLine();
  }
}
```

try-with-resources BufferedReader . try . Java SE 7 BufferedReader java.lang.AutoCloseable java.lang.AutoCloseable . BufferedReader try-with-resource try (IOException throw BufferedReader.readLine) .

.

```
long creditCardNumber = 1234_5678_9012_3456L;
long socialSecurityNumber = 999_99_9999L;
```

```
float pi = 3.14_15F;
long hexBytes = 0xFF_EC_DE_5E;
long hexWords = 0xCAFE_BABE;
long maxLong = 0x7fff_ffff_ffff_ffffL;
byte nybbles = 0b0010_0101;
long bytes = 0b11010010_01101001_10010100_10010010;
```

. .

-
-
- F L
-

```
Map<String, List<String>> myMap = new HashMap<>();
```

```
Map<String, List<String>> myMap = new HashMap<String, List<String>>();
```

, .

```
List<String> list = new ArrayList<>();
list.add("A");

  // The following statement should fail since addAll expects
  // Collection<? extends String>

list.addAll(new ArrayList<>());
```

. .

```
public String getTypeOfDayWithSwitchStatement(String dayOfWeekArg) {
     String typeOfDay;
     switch (dayOfWeekArg) {
         case "Monday":
             typeOfDay = "Start of work week";
             break;
         case "Tuesday":
         case "Wednesday":
         case "Thursday":
             typeOfDay = "Midweek";
             break;
         case "Friday":
             typeOfDay = "End of work week";
             break;
         case "Saturday":
         case "Sunday":
             typeOfDay = "Weekend";
             break;
         default:
             throw new IllegalArgumentException("Invalid day of the week: " + dayOfWeekArg);
     }
     return typeOfDay;
}
```

Java SE 7 : https://riptutorial.com/ko/java/topic/8272/java-se-7-

21: Java SE 8

Java SE 8 Java . JDBC Java Virtual Machine (JVM) .

: Java SE 8

Examples

Java SE 8

- Lambda Expressions . . ( ) .
  - .
  - .
  - Java SE 8 API .
- - Java . Java . , Java SE 7 . Java SE 8 .
  -
  -
- .
- Type Annotations . .
- - `java.lang.reflect.Executable.getParameters` . (Method Constructor Executable `Executable.getParameters` .) `.class` . `.class` Reflection API javac `-parameters` .
- Date-time-api - `java.time` API `java.time` . .

Java SE 8 : https://riptutorial.com/ko/java/topic/8267/java-se-8-

22: Java

Examples

JNA

JNA ?

Java Native Access (JNA) Java (Windows `.dll` , Unix `.so` ...) .

?

- JNA CLASSPATH jna.jar .
- , Java , .

  Windows . `"msvcrt" "c" "msvcrt".`

Java C `printf` .

CRuntimeLibrary.java

```
package jna.introduction;

import com.sun.jna.Library;
import com.sun.jna.Native;

// We declare the printf function we need and the library containing it (msvcrt)...
public interface CRuntimeLibrary extends Library {

   CRuntimeLibrary INSTANCE =
       (CRuntimeLibrary) Native.loadLibrary("msvcrt", CRuntimeLibrary.class);

   void printf(String format, Object... args);
}
```

MyFirstJNAProgram.java

```
package jna.introduction;

// Now we call the printf function...
public class MyFirstJNAProgram {
    public static void main(String args[]) {
         CRuntimeLibrary.INSTANCE.printf("Hello World from JNA !");
    }
}
```

?

.

Java : https://riptutorial.com/ko/java/topic/5244/java--

23: Java

Java . Java .

Java . , .

Examples

Java `finalize` . Java . .

```
public class MyClass {

    //Methods for the class

    @Override
    protected void finalize() throws Throwable {
        // Cleanup code
    }
}
```

Java .

- Java `finalize()` .
- `finalize()` .
- . .

( ) `finalize` . , .

, .

▪

. . 1 .

```
public class CaptainJack {
    public static CaptainJack notDeadYet = null;

    protected void finalize() {
        // Resurrection!
        notDeadYet = this;
    }
}
```

`CaptainJack` `finalize()` `notDeadYet` `notDeadYet` . .

: ?

: .

catch JVM finalizer . `notDeadYet` `notDeadYet` `null` `finalize()` .

1 - https://en.wikipedia.org/wiki/Jack_Harkness .

GC

.

```
System.gc();
```

Java . JVM (Java Virtual Machine) "" .

. JVM `-XX:+DisableExplicitGC` `System.gc()` . `System.gc()` JVM / .

C ++ -

C ++ . `new` C ++ `delete` `delete` .

- `delete` . .
- `delete` `delete`

C ++ `new` `delete` . .

Java -

Java . `delete` Java . Java . GC (*garbage collector*) GC .

Java [1] .

- JLS .

  .

  `static` .

- .

. . :

- [1] .
- .

Java JVM . () JVM .

JLS .

?

, . . GC .

GC .

1. `Reference` , .
2. *finalizable* finalize. .
3. .

.

.

```
// A node in simple "open" linked-list.
public class Node {
    private static int counter = 0;

    public int nodeNumber = ++counter;
    public Node next;
}

public class ListTest {
    public static void main(String[] args) {
        test();                     // M1
        System.out.prinln("Done"); // M2
    }

    private static void test() {
        Node n1 = new Node();      // T1
        Node n2 = new Node();      // T2
        Node n3 = new Node();      // T3
        n1.next = n2;              // T4
        n2 = null;                 // T5
        n3 = null;                 // T6
    }
}
```

`test()` . T1, T2 T3 `Node` `n1` , `n2` `n3` . T4 `Node` `Node` `next` . `Node` .

```
 n2 -> Node2
 n1 -> Node1, Node1.next -> Node2
```

T5 `n2` `null` . `Node2` `Node2` `Node2` .

T6 `n3` `null` . `Node3` , `Node3` . `Node1` `Node2` `n1` .

`test()` `n1` , `n2` `n3` . `Node1` `Node2` `Node` .

---

1 - finalization `Reference` . Java .

, PermGen

Java , . `java` . Java 8 Java PermGen .

PermGen Java 8 PermGen ( ).

JVM . . .

, PermGen :

JVM .

- `-Xms<size>` - .
- `-Xmx<size>` - .
- `-XX:PermSize<size>` - PermGen .
- `-XX:MaxPermSize<size>` - PermGen .
- `-Xss<size>` - .

`<size>  k , m  g  .  ,  .`

:

```
$ java -Xms512m -Xmx1024m JavaApp
$ java -XX:PermSize=64m -XX:MaxPermSize=128m JavaApp
$ java -Xss512k JavaApp
```

:

`-XX:+printFlagsFinal` JVM . .

- Linux, Unix, Solaris Mac OSX

  $ java -XX : + PrintFlagsFinal -version | grep -iE 'HeapSize | PermSize | ThreadStackSize'

- Windows :

  java -XX : + PrintFlagsFinal -version | findstr / i "HeapSize PermSize ThreadStackSize"

.

```
uintx InitialHeapSize                          := 20655360        {product}
uintx MaxHeapSize                              := 331350016       {product}
uintx PermSize                                  = 21757952        {pd product}
uintx MaxPermSize                               = 85983232        {pd product}
 intx ThreadStackSize                           = 1024            {pd product}
```

.

Java . . .

▪

.

```
public class NaiveStack {
    private Object[] stack = new Object[100];
    private int top = 0;

    public void push(Object obj) {
        if (top >= stack.length) {
            throw new StackException("stack overflow");
        }
        stack[top++] = obj;
    }

    public Object pop() {
        if (top <= 0) {
            throw new StackException("stack underflow");
        }
        return stack[--top];
    }

    public boolean isEmpty() {
        return top == 0;
    }
}
```

`push` `pop` `stack` .

API . (), 「*thread* 」 . JVM . JVM . (.)

, `NaiveStack` "" . .

.

```
    public Object pop() {
        if (top <= 0) {
            throw new StackException("stack underflow");
        }
        Object popped = stack[--top];
        stack[top] = null;              // Overwrite popped reference with null.
        return popped;
    }
```

▪

. . . .

. . .

```
public class RequestHandler {
    private Map<Task, Result> cache = new HashMap<>();

    public Result doRequest(Task task) {
        Result result = cache.get(task);
        if (result == null) {
            result == doRequestProcessing(task);
            cache.put(task, result);
        }
        return result;
```

```
    }
}
```

`doRequest` . . .

. . `WeakHashMap` JVM .

Java : https://riptutorial.com/ko/java/topic/2804/java--

24: Java

Java , JLS , thread thread . "JLS 17.4 "(Java 8 , Java 7 , Java 6)

Java 5 Java ( ) `volatile volatile` . .

Examples

.

```
public class Example {
    public int a, b, c, d;

    public void doIt() {
       a = b + 1;
       c = d + 1;
    }
}
```

, . :

```
public class SingleThreaded {
    public static void main(String[] args) {
        Example eg = new Example();
        System.out.println(eg.a + ", " + eg.c);
        eg.doIt();
        System.out.println(eg.a + ", " + eg.c);
    }
}
```

:

```
0, 0
1, 1
```

"main" , `main() doIt()` . Java (JLS) .

.

```
public class MultiThreaded {
    public static void main(String[] args) {
        final Example eg = new Example();
        new Thread(new Runnable() {
            public void run() {
                while (true) {
                    eg.doIt();
                }
            }
        }).start();
        while (true) {
            System.out.println(eg.a + ", " + eg.c);
        }
```

```
    }
}
```

?

, JLS , .

- `0, 0` .
- `N, N N, N + 1` .
- `N + 1, N` .
- `0, 0` 1 .

1 - `println` . .

?

JIT `doIt()` . JLS, thread *(*) . , `doIt()` () . JIT .

?

, ... JLS , *thread* , .

. . . .

. , , . JLS Java JIT " " " " . , .

JLS JIT . ( ) ?

Java . . .

- mutex `synchronized` .
- `volatile` .
- ; , `java.util.concurrent`

. Java .

Java , JLS , thread thread . , () . " " . "happen before" JIT .

Java .

-

(Java Language Specification . .)

Happens-Before . JLS (JLS 17.4.5) :

 " . ."

?

JLS 17.4.2 . 5 .

- : .
- : .
- :
  - : .
  - : .
  - .
  - . .
  - () .
  - .
- . .
- . .

(JLS 17.4.3 JLS 17.4.4) Java .

.

.

- .
- .
- (, `Thread.start()`) (`run()`) .
- . JLS .
- . `true` `join()` `isTerminated()` `isTerminated()` .
- .

(JLS 17.4.5), , .

, `v` `v` `write(v)` - `read(v)` `v` . `read(v)` .

.

- **# 1** - x y x y x y .
- **Rule # 2** - finalizer .
-

3 - x y x y .

- **# 4** - x - y y - z x - z .

Java . .

-

.

, .

```
public class SingleThreadExample {
    public int a, b;

    public int add() {
       a = 1;          // write(a)
       b = 2;          // write(b)
       return a + b;   // read(a) followed by read(b)
    }
}
```

Rule by Happen-Before 1 :

1. write(a) write(b) .
2. write(b) read(a) .
3. read(a) - read(a) .

4 :

4. write(a) - write(b) AND write(b) - read(a) write(a) - read(a) .
5. write(b) - read(a) AND read(a) - read(b) write(b) - read(b) .

:

6. write(a) - read(a) a + b a .
7. write(b) - read(b) a + b b .

2 'volatile'

" .

```
public class VolatileExample {
    private volatile int a;
    private int b;          // NOT volatile

    public void update(int first, int second) {
       b = first;          // write(b)
       a = second;          // write-volatile(a)
    }

    public int observe() {
```

```
        return a + b;       // read-volatile(a) followed by read(b)
    }
}
```

, 2 :

1. VolatileExample . ve ,
2. ve.update(1, 2)
3. ve.observe() .

Rule by Happen-Before 1 :

1. write(a) volatile-write(a) .
2. volatile-read(a) read(b) .

Rule by Happen-Before 2 :

3. volatile-write(a) volatile-read(a) .

4 :

4. write(b) read(b) .

, b ., update , observe() b a - . volatile-read(a) volatile-write(a) .

observe() b .

3

.

1. VolatileExample . ve ,
2. update .
   - ve.update(1, 2) .
   - ve.update(3, 4) ,
3. ve.observe() .

1 2 ., .

1 - update(1, 2) update(3,4) .

```
write(b, 1), write-volatile(a, 2)     // first thread
write(b, 3), write-volatile(a, 4)     // second thread
read-volatile(a), read(b)             // third thread
```

, write(b, 3) read(b) chain . b . b 3 .

2 - update(1, 2) update(3,4) .

```
write(b, 3)                           // second thread
```

```
write(b, 1)                          // first thread
write-volatile(a, 2)                 // first thread
write-volatile(a, 4)                 // second thread
read-volatile(a), read(b)            // third thread
```

`write(b, 3)` `read(b)` `write(b, 1)` . `read(b)` `read(b)` .

(: `volatile` .)

, . .

, .

- . , .
- " " .
- mutex `Lock` [1] .
- `Executor` / `ExecutorService` fork join .
- wait / notify / notifyAll , , java.util.concurrent .
- , ,, `java.util.concurrent` .

" " Java . .

---

1 - . .

Java : https://riptutorial.com/ko/java/topic/6829/java--

25: Java - 'java' 'javaw'

- `java [ <opt> ... ] <class-name> [ <argument> ... ]`
- `java [ <opt> ... ] -jar <jar-file-pathname> [ <argument> ... ]`

`java` Java . Java SE JRE JDK .

Windows `java` .

- `java` .
- `javaw` .

(: Linux, Mac OSX, UNIX) `java` .

`<opt>` `java` . "Java Options" "Heap and stack sizing options" . JVM Flags .

Examples

JAR

JAR Java . * (: JAR .) *

`<jar-path>` JAR .

```
java -jar <jar-path>
```

`<jar-path>` . :

```
java -jar <jar-path> arg1 arg2 arg3
```

`java` JVM , `-jar` . `-jar` `-cp` / `-classpath` . JAR .

"main" Java

JAR `java` .

HelloWorld

"HelloWorld" Java . `HelloWorld` .

( ) "HelloWorld.class" .

```
java HelloWorld
```

.

- ".class" ".java" .
- (Java `java` . `SomeClass` `com.example` `com.example.SomeClass` .

`java -jar` `java` . . ( ) . `-cp` `-cp` .

```
java -cp . HelloWorld
```

(".") .

`-cp` `java` . `java` classname . Java , `main` `String[]` .

(`-cp` `java` `CLASSPATH` . , `java` "." .)

Java `main` .

```
public static void main(String[] args)
```

Sidenote : , `(String args[])`

`java` `main` . Java `String` . `main` , `null` `null` .

.

- `main` ( )
- `public static`
- `void`
- `String[]` . .
- : .
- - ( ) .

`public` . Java 5 `main` `String` varargs . `main` throw `args` .

JavaFX

Java 8 `java` JavaFX . JavaFX JavaFX JavaFX .

- `javafx.application.Application`
- `public abstract abstract`
-
- `public` no-args .

'java'

'java' .

" "

:

```
java: command not found
```

`java` , `java` . .

- Java JRE JDK .
- () `PATH` ,
- " " .

: Java ; .

" "

`java` . .

- .
- .
- classpath Java .

.

1. .
    - . "Main" "com.example.myapp" "com.example.myapp.Main".
    - `javap` .
    - .
    - JAR ZIP JAR ZIP .
2. `java` . `java` .
    - .
    - ".java" ".class" .
    - Java [1] [1] .
    - .
3. .
    - . .
    - classpath . :
    - classpath JAR ZIP .
    - .

`-Xdiag -XshowSettings` . JVM .

, .

- `Main-Class` JAR .
- `Class-Path` JAR .
- classname [2] `java` .
- Java
-

.

" <> "

`java` .

.

- JAR JAR "Main-Class" .
- `java` .
- . .

- " " ?
- http://docs.oracle.com/javase/tutorial/getStarted/problems/index.html

---

1 - Java 8 `java` ("/" "") (".") . .

2 - " " .

Java

"main class" "executable JAR" .

Java . JAR .

Java . Java JVM . , .

- JAR .
- JAR JVM .

JAR "Class-Path" . *(: `jar` .)* `-cp CLASSPATH` .

, `com.example.MyApp` "myApp.jar" Java . JAR "lib / library1.jar" "lib / library2.jar" . `java` .

```
$ # Alternative 1 (preferred)
$ java -cp myApp.jar:lib/library1.jar:lib/library2.jar com.example.MyApp

$ # Alternative 2
$ export CLASSPATH=myApp.jar:lib/library1.jar:lib/library2.jar
$ java com.example.MyApp
```

(Windows classpath `:` `;` `export` `set` () `CLASSPATH` `set` .)

Java " ". (Windows) . , "myApp" .

```
#!/bin/bash
# The 'myApp' wrapper script

export DIR=/usr/libexec/myApp
```

```
export CLASSPATH=$DIR/myApp.jar:$DIR/lib/library1.jar:$DIR/lib/library2.jar
java com.example.MyApp
```

.

```
$ myApp arg1 arg2 ...
```

`"$@"` Java . (Windows .)

, Java . Java .

.

```
import java.io.File;

public class PrintFileSizes {

    public static void main(String[] args) {
        for (String name: args) {
            File file = new File(name);
            System.out.println("Size of '" + file + "' is " + file.size());
        }
    }
}
```

. : `/home/steve/Test File.txt` . :

```
$ java PrintFileSizes /home/steve/Test File.txt
```

`/home/steve/Test File.txt` . Java . Java () .

POSIX

POSIX `sh` `bash` `ksh` . .

```
$ java PrintFileSizes "/home/steve/Test File.txt"
```

. . .

```
$ java PrintFileSizes '/home/steve/Test File.txt'
```

() .

```
$ java PrintFileSizes /home/steve/Test\ File.txt
```

.

Bash .

Windows

Windows OS ( ) . , ( ) . .

Java :

- `java` .
- `java` .
- `SET` .
- `cmd.exe` . `^` escapes .

.

Java

`java` .

- (`-`) . "" `--` GNU / Linux .
- `<classname> -jar <jarfile>` . Java .
- `-X -XX` . Java [1] .
- `-X` Java .
- `-XX` .

-D

`-D<property>=<value>` Properties . .

,

, PermGen . *(: .)*

`-ea -da` Java `assert` .

- .
- `-ea` .
- `-ea:<packagename>...` .
- `-ea:<classname>...` .
- `-da` .
- `-da:<packagename>...` .
- `-da:<classname>...` .
- `-esa` .

- `-dsa` .

. .

```
$ # Enable all assertion checking in non-system classes
$ java -ea -dsa MyApp

$ # Enable assertions for all classes in a package except for one.
$ java -ea:com.wombat.fruitbat... -da:com.wombat.fruitbat.Brickbat MyApp
```

Java .

- .
- .
- - .
- `assert` .

VM

`-client -server` HotSpot VM .

- "" .
- "" . JVM JVM "warm up" .

JVM 64 . `-d32 -d64` .

---

1 - `java` . " " .

Java - 'java' 'javaw' : https://riptutorial.com/ko/java/topic/5791/java-----java---javaw-

26: Java

( ). Java `float` (4) `double` (8) . Java .

Examples

(`float` `double`) (`==` `!=` , `<`). . :

```
public class CompareTest {
    public static void main(String[] args) {
        double oneThird = 1.0 / 3.0;
        double one = oneThird * 3;
        System.out.println(one == 1.0);      // prints "false"
    }
}
```

`oneThird` `oneThird` `3` `1.0` .

`double` `float` . :

```
public class CompareTest2 {
    public static void main(String[] args) {
        float floatVal = 0.1f;
        double doubleVal = 0.1;
        double doubleValCopy = floatVal;

        System.out.println(floatVal);      // 0.1
        System.out.println(doubleVal);     // 0.1
        System.out.println(doubleValCopy); // 0.10000000149011612

        System.out.println(floatVal == doubleVal); // false
        System.out.println(doubleVal == doubleValCopy); // false
    }
}
```

`float` `double` . `float` 23 8 . `double` 52 15 10 . . . .

```
if (Math.abs(v1 - v2) < delta)
```

:

```
public class DeltaCompareExample {

    private static boolean deltaCompare(double v1, double v2, double delta) {
        // return true iff the difference between v1 and v2 is less than delta
        return Math.abs(v1 - v2) < delta;
    }

    public static void main(String[] args) {
        double[] doubles = {1.0, 1.0001, 1.0000001, 1.000000001, 1.0000000000001};
        double[] deltas = {0.01, 0.00001, 0.0000001, 0.0000000001, 0};
```

```
        // loop through all of deltas initialized above
        for (int j = 0; j < deltas.length; j++) {
            double delta = deltas[j];
            System.out.println("delta: " + delta);

            // loop through all of the doubles initialized above
            for (int i = 0; i < doubles.length - 1; i++) {
                double d1 = doubles[i];
                double d2 = doubles[i + 1];
                boolean result = deltaCompare(d1, d2, delta);

                System.out.println("" + d1 + " == " + d2 + " ? " + result);

            }

            System.out.println();
        }
    }
}
```

:

```
delta: 0.01
1.0 == 1.0001 ? true
1.0001 == 1.0000001 ? true
1.0000001 == 1.000000001 ? true
1.000000001 == 1.0000000000001 ? true

delta: 1.0E-5
1.0 == 1.0001 ? false
1.0001 == 1.0000001 ? false
1.0000001 == 1.000000001 ? true
1.000000001 == 1.0000000000001 ? true

delta: 1.0E-7
1.0 == 1.0001 ? false
1.0001 == 1.0000001 ? false
1.0000001 == 1.000000001 ? true
1.000000001 == 1.0000000000001 ? true

delta: 1.0E-10
1.0 == 1.0001 ? false
1.0001 == 1.0000001 ? false
1.0000001 == 1.000000001 ? false
1.000000001 == 1.0000000000001 ? false

delta: 0.0
1.0 == 1.0001 ? false
1.0001 == 1.0000001 ? false
1.0000001 == 1.000000001 ? false
1.000000001 == 1.0000000000001 ? false
```

`double` `float` `compare` . :

```
double a = 1.0;
double b = 1.0001;

System.out.println(Double.compare(a, b));//-1
System.out.println(Double.compare(b, a));//1
```

. . . . (Numerical Analysis .)

float 32 IEEE 754 .

`Float`

`3.4028235e+38` , `Infinity` .

```
float f = 3.4e38f;
float result = f*2;
System.out.println(result); //Infinity
```

`Float`

1.4e-45f, `0.0` .

```
    float f = 1e-45f;
    float result = f/1000;
    System.out.println(result);
```

double `64-bit` IEEE 754 .

`Double`

`1.7976931348623157e+308` `Infinity` .

```
double d = 1e308;
double result=d*2;
System.out.println(result); //Infinity
```

`Double`

4.9e-324, `0.0` .

```
    double d = 4.8e-323;
    double result = d/1000;
    System.out.println(result); //0.0
```

`'f'` `String.format` 10 .

```
    //Two digits in fracttional part are rounded
    String format1 = String.format("%.2f", 1.2399);
    System.out.println(format1); // "1.24"

    // three digits in fractional part are rounded
    String format2 = String.format("%.3f", 1.2399);
    System.out.println(format2); // "1.240"

    //rounded to two digits, filled with zero
    String format3 = String.format("%.2f", 1.2);
    System.out.println(format3); // returns "1.20"

    //rounder to two digits
```

```
    String format4 = String.format("%.2f", 3.19999);
    System.out.println(format4); // "3.20"
```

`DecimalFormat` 10 .

```
   // rounded with one digit fractional part
    String format = new DecimalFormat("0.#").format(4.3200);
    System.out.println(format); // 4.3

   // rounded with two digit fractional part
    String format = new DecimalFormat("0.##").format(1.2323000);
    System.out.println(format); //1.23

    // formatting floating numbers to decimal number
    double dv = 123456789;
    System.out.println(dv); // 1.23456789E8
    String format =  new DecimalFormat("0").format(dv);
    System.out.println(format); //123456789
```

IEEE

`float` `double` IEEE 754 . . .

`strictfp` . , , , , , , , `strictfp` , float double IEEE 754 .

`strictfp` .

`strictfp` (CPU). . .

```
public class StrictFP { // No strictfp -> default lenient
    public strictfp float strict(float input) {
        return input * input / 3.4f; // Strictly adheres to the spec.
                                     // May be less accurate and may be slower.
    }

    public float lenient(float input) {
        return input * input / 3.4f; // Can sometimes be more accurate and faster,
                                     // but results may not be reproducable.
    }

    public static final strictfp class Ops { // strictfp affects all enclosed entities
        private StrictOps() {}

        public static div(double dividend, double divisor) { // implicitly strictfp
            return dividend / divisor;
        }
    }
}
```

Java : https://riptutorial.com/ko/java/topic/6167/java---

27: Java

Examples

Java . . :

- .
- .
- .

. . JVM JVisualVM JDK .

, .

Java String . GC String .

String .

```
myString += "foo";
```

```
for (int i = 0; i < N; i++) {
    myString += "foo" + i;
}
```

+ String ( ). `StringBuilder` `StringBuffer` .

```
StringBuffer sb = new StringBuffer(myString);
for (int i = 0; i < N; i++) {
    sb.append("foo").append(i);
}
myString = sb.toString();
```

(: SQL) String API .

:

- `replace` , `substring` .
- `String.toArray()` .
- (:) ( ).
- .
- ( ) StringBuilder .

Java

Donald Knuth .

" , , . *97 %* : . 3 % . "

sage .

1. , . .
2. () .
3. . , .
4. .
5. . (" " .)
6. . . (.)
7. .
8. ( ) " " . .
9. .
10. , .
11. .
    - 4 .
    - 9 . .

. . 1 % , 50 % 0.5 % .

. . :

- .
- .
- CPU .
- .

.

Java : https://riptutorial.com/ko/java/topic/4160/java--

28: Java

, .

.

.
1. TCP - 2. UDP -

TCP .
UDP .

TCP , .

UDP . UDP .

-
TCP . UDP .

TCP .
UDP .

Examples

TCP

TCP . . .

```
public class CAPECHOServer extends Thread{

    // This class implements server sockets. A server socket waits for requests to come
    // in over the network only when it is allowed through the local firewall
    ServerSocket serverSocket;

    public CAPECHOServer(int port, int timeout){
        try {
            // Create a new Server on specified port.
            serverSocket = new ServerSocket(port);
            // SoTimeout is basiacally the socket timeout.
            // timeout is the time until socket timeout in milliseconds
            serverSocket.setSoTimeout(timeout);
        } catch (IOException ex) {
            Logger.getLogger(CAPECHOServer.class.getName()).log(Level.SEVERE, null, ex);
        }
    }

    @Override
    public void run(){
        try {
            // We want the server to continuously accept connections
            while(!Thread.interrupted()){
```

```
            }
            // Close the server once done.
            serverSocket.close();
        } catch (IOException ex) {
            Logger.getLogger(CAPECHOServer.class.getName()).log(Level.SEVERE, null, ex);
        }
    }

}
```

. run .

```
@Override
public void run(){
    while(!Thread.interrupted()){
        try {
            // Log with the port number and machine ip
            Logger.getLogger((this.getClass().getName())).log(Level.INFO, "Listening for
Clients at {0} on {1}", new Object[]{serverSocket.getLocalPort(),
InetAddress.getLocalHost().getHostAddress()});
            Socket client = serverSocket.accept();  // Accept client conncetion
            // Now get DataInputStream and DataOutputStreams
            DataInputStream istream = new DataInputStream(client.getInputStream()); // From
client's input stream
            DataOutputStream ostream = new DataOutputStream(client.getOutputStream());
            // Important Note
            /*
                The server's input is the client's output
                The client's input is the server's output
            */
            // Send a welcome message
            ostream.writeUTF("Welcome!");

            // Close the connection
            istream.close();
            ostream.close();
            client.close();
        } catch (IOException ex) {
            Logger.getLogger(CAPECHOServer.class.getName()).log(Level.SEVERE, null, ex);
        }
    }

    // Close the server once done

    try {
        serverSocket.close();
    } catch (IOException ex) {
        Logger.getLogger(CAPECHOServer.class.getName()).log(Level.SEVERE, null, ex);
    }
}
```

.

IP .

.

```
Welcome!
```

```
Connection to host lost.
```

. TCP . .

, .

```
public class CAPECHOClient extends Thread{

Socket server;
Scanner key; // Scanner for input

    public CAPECHOClient(String ip, int port){
        try {
            server = new Socket(ip, port);
            key = new Scanner(System.in);
        } catch (IOException ex) {
            Logger.getLogger(CAPECHOClient.class.getName()).log(Level.SEVERE, null, ex);
        }
    }

    @Override
    public void run(){
        DataInputStream istream = null;
        DataOutputStream ostream = null;
        try {
            istream = new DataInputStream(server.getInputStream()); // Familiar lines
            ostream = new DataOutputStream(server.getOutputStream());
            System.out.println(istream.readUTF());  // Print what the server sends
            System.out.print(">");
            String tosend = key.nextLine();
            ostream.writeUTF(tosend);   // Send whatever the user typed to the server
            System.out.println(istream.readUTF());  // Finally read what the server sends
before exiting.
        } catch (IOException ex) {
            Logger.getLogger(CAPECHOClient.class.getName()).log(Level.SEVERE, null, ex);
        } finally {
            try {
                istream.close();
                ostream.close();
                server.close();
            } catch (IOException ex) {
                Logger.getLogger(CAPECHOClient.class.getName()).log(Level.SEVERE, null, ex);
            }
        }
    }
}
```

.

```
ostream.writeUTF("Welcome!");

String inString = istream.readUTF();    // Read what the user sent
String outString = inString.toUpperCase();  // Change it to caps
ostream.writeUTF(outString);

// Close the connection
istream.close();
```

.

```
Welcome!
>
```

Java : https://riptutorial.com/ko/java/topic/9923/java-

29: Java

Examples

, Manifest.mf .

```
Can-Redefine-Classes: true
Can-Retransform-Classes: true
```

Java Instrumentation . Instrumentation *addTransformer (ClassFileTransformer transformer)* . ClassFileTransformers . ClassLoader, , java.lang.Class , ProtectionDomain .

.

```
byte[] transform(ClassLoader loader, String className, Class<?> classBeingRedefined,
          ProtectionDomain protectionDomain, byte[] classfileBuffer)
```

. .

ASM Javassist BCEL .

```
ClassNode getNode(byte[] bytes) {
    // Create a ClassReader that will parse the byte array into a ClassNode
    ClassReader cr = new ClassReader(bytes);
    ClassNode cn = new ClassNode();
    try {
        // This populates the ClassNode
        cr.accept(cn, ClassReader.EXPAND_FRAMES);
        cr = null;
    } catch (Exception e) {
        e.printStackTrace();
    }
    return cn;
}
```

ClassNode . / . ASM Tree API .

ClassNode .

```
public static byte[] getNodeBytes(ClassNode cn, boolean useMaxs) {
    ClassWriter cw = new ClassWriter(useMaxs ? ClassWriter.COMPUTE_MAXS :
ClassWriter.COMPUTE_FRAMES);
    cn.accept(cw);
    byte[] b = cw.toByteArray();
    return b;
}
```

JVM . Attach API *VirtualMachine.attatch (String id)* . jar .

```
public static void loadAgent(String agentPath) {
```

```
    String vmName = ManagementFactory.getRuntimeMXBean().getName();
    int index = vmName.indexOf('@');
    String pid = vmName.substring(0, index);
    try {
        File agentFile = new File(agentPath);
        VirtualMachine vm = VirtualMachine.attach(pid);
        vm.loadAgent(agentFile.getAbsolutePath(), "");
        VirtualMachine.attach(vm.id());
    } catch (Exception e) {
        throw new RuntimeException(e);
    }
}
```

premain (String agentArgs, Instrumentation inst) *agentmain (String agentArgs, Instrumentation inst)* . Manifest.mf .

Premain *"premain (String agentArgs Instrumentation inst)"* .

.

```
import java.lang.instrument.Instrumentation;

public class PremainExample {
    public static void premain(String agentArgs, Instrumentation inst) {
        System.out.println(agentArgs);
    }
}
```

jar Manifest Premain-Class .

.

```
Premain-Class: PremainExample
```

Java "myProgram" JVM .

```
java -javaagent:PremainAgent.jar -jar myProgram.jar
```

Java : https://riptutorial.com/ko/java/topic/1265/java-

30: Java

Java Print Service API .

IETF RFC 2911 (IPP) 1.1 .

Examples

`PrintServiceLookup` . :

```
import javax.print.PrintService;
import javax.print.PrintServiceLookup;

public class DiscoveringAvailablePrintServices {

    public static void main(String[] args) {
        discoverPrintServices();
    }

    public static void discoverPrintServices() {
        PrintService[] allPrintServices = PrintServiceLookup.lookupPrintServices(null, null);

        for (Printservice printService : allPrintServices) {
            System.out.println("Print service name: " + printService.getName());
        }
    }

}
```

Windows .

```
Print service name: Fax
Print service name: Microsoft Print to PDF
Print service name: Microsoft XPS Document Viewer
```

`PrintServiceLookup` . ::

```
import javax.print.PrintService;
import javax.print.PrintServiceLookup;

public class DiscoveringDefaultPrintService {

    public static void main(String[] args) {
        discoverDefaultPrintService();
    }

    public static void discoverDefaultPrintService() {
        PrintService defaultPrintService = PrintServiceLookup.lookupDefaultPrintService();
        System.out.println("Default print service name: " + defaultPrintService.getName());
    }

}
```

. .

- ( )
-

.

```
DocPrintJob printJob = printService.createPrintJob();
```

DocPrintJob print .

```
printJob.print(doc, pras);
```

doc Doc : .

pras PrintRequestAttributeSet . PrintRequestAttribute . .

- (1, 2),
- ( )
- (,)
- (, ,)
- (,)
- ...

print PrintException .

Doc , Java Print Service API SimpleDoc .

Doc .

- (, ,)
- DocFlavor (MIME +) .

Doc . .

```
FileInputStream pdfFileInputStream = new FileInputStream("something.pdf");
```

DocFlavor DocFlavor . DocFlavor . INPUT_STREAM.PDF .

```
DocFlavor pdfDocFlavor = DocFlavor.INPUT_STREAM.PDF;
```

SimpleDoc .

```
Doc doc = new SimpleDoc(pdfFileInputStream, pdfDocFlavor , null);
```

doc ( ).

. .

.

- (1, 2),
- ( )
- (,)
- (, ,)
- (,)
- ...

.

```
PrintRequestAttributeSet pras = new HashPrintRequestAttributeSet();
```

. .

```
pras.add(new Copies(5));
pras.add(MediaSize.ISO_A4);
pras.add(OrientationRequested.PORTRAIT);
pras.add(PrintQuality.NORMAL);
```

`pras` ( ).

.

Java Print Service API . .

- `PrintJobListener`
- .

. . .

- ,
- ,
- ,
- .

:

```
import javax.print.event.PrintJobEvent;
import javax.print.event.PrintJobListener;

public class LoggerPrintJobListener implements PrintJobListener {

    // Your favorite Logger class goes here!
    private static final Logger LOG = Logger.getLogger(LoggerPrintJobListener.class);


    public void printDataTransferCompleted(PrintJobEvent pje) {
        LOG.info("Print data transfer completed ;) ");
    }

    public void printJobCompleted(PrintJobEvent pje) {
```

```
        LOG.info("Print job completed =) ");
    }

    public void printJobFailed(PrintJobEvent pje) {
        LOG.info("Print job failed =( ");
    }

    public void printJobCanceled(PrintJobEvent pje) {
        LOG.info("Print job canceled :| ");
    }

    public void printJobNoMoreEvents(PrintJobEvent pje) {
        LOG.info("No more events to the job ");
    }

    public void printJobRequiresAttention(PrintJobEvent pje) {
        LOG.info("Print job requires attention :O ");
    }
}
```

.

```
DocPrintJob printJob = printService.createPrintJob();

printJob.addPrintJobListener(new LoggerPrintJobListener());

printJob.print(doc, pras);
```

PrintJobEvent

`PrintJobEvent pje` . . :

```
pje.getPrintJob().getAttributes();
```

`PrintJobAttributeSet` for each way .

`PrintJobAdapter` . `PrintJobListener` . . . :

```
import javax.print.event.PrintJobEvent;
import javax.print.event.PrintJobAdapter;

public class LoggerPrintJobAdapter extends PrintJobAdapter {

    // Your favorite Logger class goes here!
    private static final Logger LOG = Logger.getLogger(LoggerPrintJobAdapter.class);

    public void printJobCompleted(PrintJobEvent pje) {
        LOG.info("Print job completed =) ");
    }

    public void printJobFailed(PrintJobEvent pje) {
```

```
        LOG.info("Print job failed =( ");
    }
}
```

.

PrintJobListener .

```
printJob.addPrintJobListener(new LoggerPrintJobAdapter());

printJob.print(doc, pras);
```

Java : https://riptutorial.com/ko/java/topic/10178/java--

31: Java

javadoc . Javadoc java HTML API Sun Microsystems . HTML .

- / ** - , , JavaDoc
- @author // , enum . .
- @version // , enum . . % I % % G %
- @param // ( ) . @param .
- @return // void .
- @exception // . . ArrayIndexOutOfBoundsException catch . Throw @exception .
- @throws // @exception .
- @see // , , . package.Class # something .
- @since // , . , java.util.Optional <T> JDK-8.
- @serial, @serialField, @serialData // serialVersionUID .
- @deprecated // , . , java.io.StringBufferInputStream . .
- {@link} // @see . JFrame # setDefaultCloseOperation . {@link #setDefaultCloseOperation (int closeOperation))
- {@linkplain} // {@link} .
- {@code} // HTML . : {@code <html> </ html>}. . {@literal} .
- {@literal} // {@code} .
- {@value} // . JFrame # EXIT_ON_CLOSE {@value}. . {@value AppConstants # APP_NAME} .
- {@docRoot} // JavaDoc HTML . : .
- HTML : <code> "Hi cookies".substring (3) </ code>.
- * / - JavaDoc

Javadoc HTML JDK . Java API Javadoc . .

Examples

Javadoc (`/**`) (`*/`) . , . .

```
/**
 * Brief summary of this class, ending with a period.
 *
 * It is common to leave a blank line between the summary and further details.
 * The summary (everything before the first period) is used in the class or package
 * overview section.
 *
 * The following inline tags can be used (not an exhaustive list):
 * {@link some.other.class.Documentation} for linking to other docs or symbols
 * {@link some.other.class.Documentation Some Display Name} the link's appearance can be
 * customized by adding a display name after the doc or symbol locator
 * {@code code goes here} for formatting as code
 * {@literal <>[]()foo} for interpreting literal text without converting to HTML markup
 * or other tags.
 *
```

```
 * Optionally, the following tags may be used at the end of class documentation
 * (not an exhaustive list):
 *
 * @author John Doe
 * @version 1.0
 * @since 5/10/15
 * @see some.other.class.Documentation
 * @deprecated This class has been replaced by some.other.package.BetterFileReader
 *
 * You can also have custom tags for displaying additional information.
 * Using the @custom.<NAME> tag and the -tag custom.<NAME>:htmltag:"context"
 * command line option, you can create a custom tag.
 *
 * Example custom tag and generation:
 * @custom.updated 2.0
 * Javadoc flag: -tag custom.updated:a:"Updated in version:"
 * The above flag will display the value of @custom.updated under "Updated in version:"
 *
 */
public class FileReader {
}
```

```
Classes     Enums  Interfaces  .
```

Javadoc (`/**`) (`*/`) . , . .

```
/**
 * Brief summary of method, ending with a period.
 *
 * Further description of method and what it does, including as much detail as is
 * appropriate.  Inline tags such as
 * {@code code here}, {@link some.other.Docs}, and {@literal text here} can be used.
 *
 * If a method overrides a superclass method, {@inheritDoc} can be used to copy the
 * documentation
 * from the superclass method
 *
 * @param stream Describe this parameter.  Include as much detail as is appropriate
 *               Parameter docs are commonly aligned as here, but this is optional.
 *               As with other docs, the documentation before the first period is
 *               used as a summary.
 *
 * @return Describe the return values.  Include as much detail as is appropriate
 *         Return type docs are commonly aligned as here, but this is optional.
 *         As with other docs, the documentation before the first period is used as a
 *         summary.
 *
 * @throws IOException Describe when and why this exception can be thrown.
 *                     Exception docs are commonly aligned as here, but this is
 *                     optional.
 *                     As with other docs, the documentation before the first period
 *                     is used as a summary.
 *                     Instead of @throws, @exception can also be used.
 *
 * @since 2.1.0
 * @see some.other.class.Documentation
 * @deprecated  Describe why this method is outdated. A replacement can also be specified.
 */
public String[] read(InputStream stream) throws IOException {
```

```
    return null;
}
```

Javadoc (`/**`) (`*/`) . , . .

```
/**
 * Fields can be documented as well.
 *
 * As with other javadocs, the documentation before the first period is used as a
 * summary, and is usually separated from the rest of the documentation by a blank
 * line.
 *
 * Documentation for fields can use inline tags, such as:
 * {@code code here}
 * {@literal text here}
 * {@link other.docs.Here}
 *
 * Field documentation can also make use of the following tags:
 *
 * @since 2.1.0
 * @see some.other.class.Documentation
 * @deprecated Describe why this field is outdated
 */
public static final String CONSTANT_STRING = "foo";
```

Java SE 5

Javadoc `package-info.java` . . ( ),

```
/**
 * Package documentation goes here; any documentation before the first period will
 * be used as a summary.
 *
 * It is common practice to leave a blank line between the summary and the rest
 * of the documentation; use this space to describe the package in as much detail
 * as is appropriate.
 *
 * Inline tags such as {@code code here}, {@link reference.to.other.Documentation},
 * and {@literal text here} can be used in this documentation.
 */
package com.example.foo;

// The rest of the file must be empty.
```

`package-info.java` Java `com.example.foo` .

Javadocs `@link` .

```
/**
 * You can link to the javadoc of an already imported class using {@link ClassName}.
 *
 * You can also use the fully-qualified name, if the class is not already imported:
 *  {@link some.other.ClassName}
 *
 * You can link to members (fields or methods) of a class like so:
 *  {@link ClassName#someMethod()}
```

```
 *  {@link ClassName#someMethodWithParameters(int, String)}
 *  {@link ClassName#someField}
 *  {@link #someMethodInThisClass()} - used to link to members in the current class
 *
 * You can add a label to a linked javadoc like so:
 *  {@link ClassName#someMethod() link text}
 */
```

You can link to the javadoc of an already imported class using ClassName.

You can also use the fully-qualified name, if the class is not already imported: some.other.ClassName

You can link to members (fields or methods) of a class like so:

ClassName.someMethod()

ClassName.someMethodWithParameters(int, String)

ClassName.someField

someMethodInThisClass() - used to link to members in the current class

You can add a label to a linked javadoc like so: link text

`@see` *See also* `. @param @return ,  . @return .`

```
/**
 * This method has a nice explanation but you might found further
 * information at the bottom.
 *
 * @see ClassName#someMethod()
 */
```

This method has a nice explanation but you might found furthe

See Also:

ClassName.someMethod()

HTML `<a> . @link @see  .`

```
/**
 * Wondering how this works? You might want
 * to check this <a href="http://stackoverflow.com/">great service</a>.
 *
 * @see <a href="http://stackoverflow.com/">Stack Overflow</a>
 */
```

Wondering how this works? You might want to check this great service.

See Also:

Stack Overflow

Javadocs

IDE Javadocs HTML . (: Maven Gradle) HTML .

Javadoc HTML . `javadoc` .

.

```
javadoc JavaFile.java
```

`JavaFile.java` Javadoc HTML .

`[source-directory]` java `[package.name]` `[package.name]` `[docs-directory]` :

```
javadoc -d [docs-directory] -subpackages -sourcepath [source-directory] [package.name]
```

Javadoc .

`//` .

```
public void method() {

  //single line comment
  someMethodCall(); //single line comment after statement

}
```

`/* */` . .

```
public void method(Object object) {

  /*
    multi
    line
    comment
  */
  object/*inner-line-comment*/.method();
}
```

JavaDocs `/**` .

.

TAG . . .

- `//TODO`
- `//FIXME`

Jira .

- `//PRJ-1234`

`{@code }` . `<pre></pre>` .

```
/**
 * The Class TestUtils.
 * <p>
 * This is an {@code inline("code example")}.
 * <p>
```

```
 * You should wrap it in pre tags when writing multiline code.
 * <pre>{@code
 *  Example example1 = new FirstLineExample();
 *  example1.butYouCanHaveMoreThanOneLine();
 * }</pre>
 * <p>
 * Thanks for reading.
 */
class TestUtils {
```

javadoc . @ . {@literal } <code> .

```
/**
 * Usage:
 * <pre><code>
 * class SomethingTest {
 * {@literal @}Rule
 *  public SingleTestRule singleTestRule = new SingleTestRule("test1");
 *
 * {@literal @}Test
 *  public void test1() {
 *      // only this test will be executed
 *  }
 *
 *  ...
 * }
 * </code></pre>
 */
class SingleTestRule implements TestRule { }
```

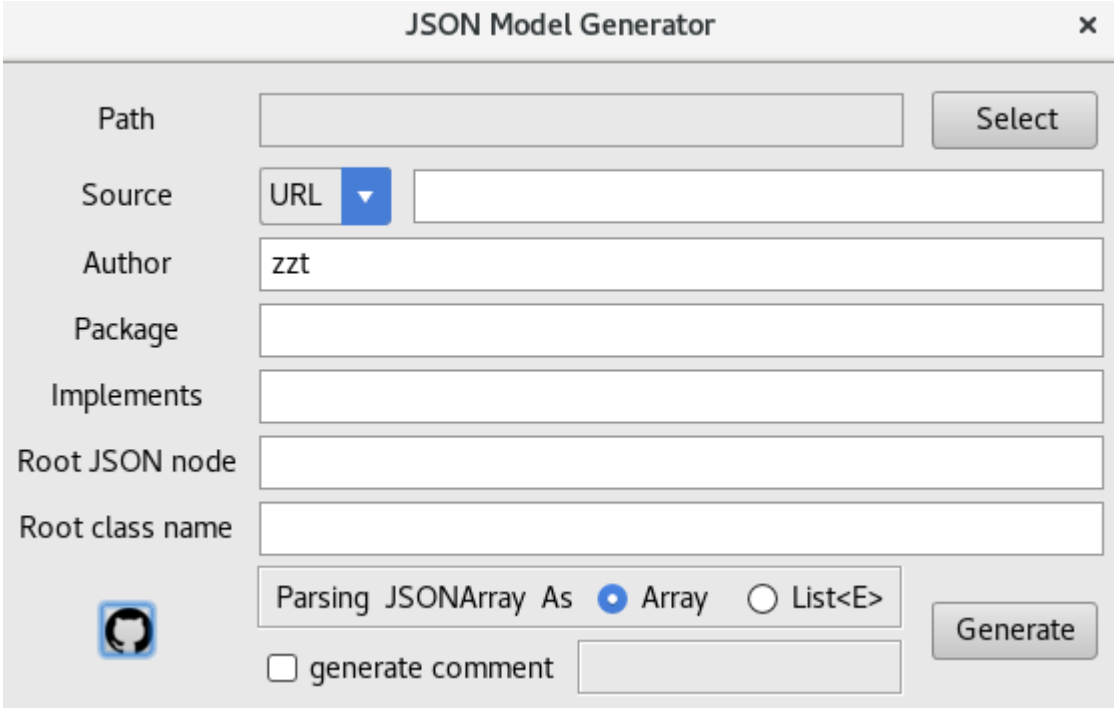
Java : https://riptutorial.com/ko/java/topic/140/java--

32: Java

Examples

JSON POJO

- Intellij Intellij JSON Model Genrator .
- " .
- UI ('Path', 'Source', 'Package').



- " .

Java : https://riptutorial.com/ko/java/topic/9400/java--

33: Java -

Java ""( ) .

""Java . , .

Examples

-

TRACE DEBUG . ,"" .

.

```
// Processing a request of some kind, logging the parameters
LOG.debug("Request coming from " + myInetAddress.toString()
          + " parameters: " + Arrays.toString(veryLongParamArray));
```

INFO debug() . .

- String : String .
- InetAddress DNS .
- veryLongParamArray . String .

. .:

```
// No toString() evaluation, no string concatenation if debug is disabled
LOG.debug("Request coming from {} parameters: {}", myInetAddress, parameters));
```

String.valueOf (Object) . compuation .

```
if (LOG.isDebugEnabled()) {
    // Argument expression evaluated only when DEBUG is enabled
    LOG.debug("Request coming from {}, parameters: {}", myInetAddress,
              Arrays.toString(veryLongParamArray);
}
```

Arrays.toString(Obect[]) LOG.debug() DEBUG .

(Pitfall) - .

.

```
public String joinWords(List<String> words) {
    String message = "";
    for (String word : words) {
        message = message + " " + word;
    }
```

```
    return message;
}
```

`words` . .

```
message = message + " " + word;
```

`message` `message` . .

`joinWords` M N O (N) O (MN ^{2}) . N ^{2} .

1 `StringBuilder` .

```
public String joinWords2(List<String> words) {
    StringBuilder message = new StringBuilder();
    for (String word : words) {
        message.append(" ").append(word);
    }
    return message.toString();
}
```

`joinWords2` `StringBuilder` "" . O (logN) O (MN) . `toString()` .

(`StringBuilder` .)

`joinWords` , Java .

```
  StringBuilder tmp = new StringBuilder();
  tmp.append(message).append(" ").append(word);
  message = tmp.toString();
```

Java `joinWords2` `joinWords2` `StringBuilder` " ".

:

- " '+' ?"

---

1 - Java 8 `Joiner` . .

Pitfall - 'new' .

Java `Integer` , `Boolean` `new` . autoboxing (Java 5) `valueOf` .

```
 Integer i1 = new Integer(1);      // BAD
 Integer i2 = 2;                   // BEST (autoboxing)
 Integer i3 = Integer.valueOf(3);  // OK
```

`new Integer(int)` JIT . , autoboxing `valueOf` Java `Integer` . " (hit)" . GC .

:

1. Java autoboxing `valueOf` `Boolean` , `Byte` , `Short` , `Integer` , `Long` `Character` .
2. Java .

Pitfall - ' ()' .

`new String(String)` `new String(String)` .

- .
- Java `String` `String` . () () . Java 7 `String` backing arrays .

`new String(String)` .

- CPU .
- GC .
- String `equals(Object)` `hashCode()` `equals(Object)` .

Pitfall - System.gc () .

`System.gc()` ( ) .

`gc()` javadoc .

" `gc` Java Virtual Machine . Java Virtual Machine . "

:

1. "" "" JVM . JVM ( ) JVM `-XX:+DisableExplicitGC` .
2. " " `gc` "" `gc` .

`System.gc()` `System.gc()` ?

, . GC "" . . GC .

, "" . . . . . . , .

, . .

JVM GC . JVM GC .

"... ( ) ...". .

1. (: finalizers weak / soft / phantom) `System.gc()` .
2. . "" . .

Pitfall - .

.

```
int a = 1000;
int b = a + 1;
```

```
Integer a = 1000;
Integer b = a + 1;
```

: ?

: .

. .

```
Integer a = Integer.valueOf(1000);               // box 1000
Integer b = Integer.valueOf(a.intValue() + 1);   // unbox 1000, add 1, box 1001
```

`int` `Integer` . `valueOf` `Integer` . .

`valueOf` . `Integer` `Integer` 16 . `int` `a` `b` .

---

boxed equivalent .

`int[]` `Integer[]` `int[]` `int` . `Integer[]` () `Integer` , `int` .

. `int[]` CPU . `Integer[]` CPU .

, CPU . .

- .

.

```
Map<String, String> map = new HashMap<>();
for (String key : map.keySet()) {
    String value = map.get(key);
    // Do something with key and value
}
```

(`get()`) . (HashMap `hashCode` `equals` `equals`). .

. .

Pitfall - size () .

Java Collections Framework `Collection` .

- `size()` `Collection`
- `isEmpty()` `Collection` true .

. :

```
Collection<String> strings = new ArrayList<>();
boolean isEmpty_wrong = strings.size() == 0; // Avoid this
boolean isEmpty = strings.isEmpty();         // Best
```

. size() . :

- (java.util.LinkedList) .
- ConcurrentHashMap "" .
- .

isEmpty() . .

size() == 0 isEmpty() isEmpty() size() == 0 . isEmpty() .

-

(Java) . .

Pattern Matcher .

.

```
/**
 * Test if all strings in a list consist of English letters and numbers.
 * @param strings the list to be checked
 * @return 'true' if an only if all strings satisfy the criteria
 * @throws NullPointerException if 'strings' is 'null' or a 'null' element.
 */
public boolean allAlphanumeric(List<String> strings) {
    for (String s : strings) {
        if (!s.matches("[A-Za-z0-9]*")) {
            return false;
        }
    }
    return true;
}
```

. matches(...) . s.matches("[A-Za-z0-9]*") .

```
Pattern.matches(s, "[A-Za-z0-9]*")
```

```
Pattern.compile("[A-Za-z0-9]*").matcher(s).matches()
```

Pattern.compile("[A-Za-z0-9]*") Pattern . . s Matcher . match() .

. .

```
private static Pattern ALPHA_NUMERIC = Pattern.compile("[A-Za-z0-9]*");

public boolean allAlphanumeric(List<String> strings) {
    Matcher matcher = ALPHA_NUMERIC.matcher("");
```

```
    for (String s : strings) {
        matcher.reset(s);
        if (!matcher.matches()) {
            return false;
        }
    }
    return true;
}
```

`Pattern` javadoc , .

, thread . `Matcher` .

find () match () .

`s` 3 . .

```
  if (s.matches(".*[0-9]{3}.*")) {
      System.out.println("matches");
  }
```

```
  if (Pattern.compile("[0-9]{3}").matcher(s).find()) {
      System.out.println("matches");
  }
```

. . ".*" "" "" .

, 3 .

, . . :

```
Pattern.compile("ABC").matcher(s).find()
```

.

```
s.contains("ABC")
```

( ).

. , `matches()` `allAlplanumeric` .

```
public boolean matches(String s) {
    for (char c : s) {
        if ((c >= 'A' && c <= 'Z') ||
            (c >= 'a' && c <= 'z') ||
            (c >= '0' && c <= '9')) {
             return false;
        }
    }
    return true;
}
```

Matcher .

(Pattern .)

( ) .

```
Pattern pat = Pattern.compile("(A+)+B");
System.out.println(pat.matcher("AAAAAAAAAAAAAAAAAAAAAAAAAAAAAB").matches());
System.out.println(pat.matcher("AAAAAAAAAAAAAAAAAAAAAAAAAAAAAC").matches());
```

println true true . false .. C A .

. .

(A+)+B . " A B " A . .

- 'AB' : '(A) B'
- 'AAB' '(AA) B' '(A) (A) B' .
- 'AAAB' '(AAA) B' '(AA) (A) B or '(A)(AA)B '(A) (A) (A) B'
-

, 2 N. N A .

, (, K O(2^N) O(N^K) . .

- .
- .
- .
- . .

, API . " " .

:

- , http://www.riptutorial.com/regex/topic/259/getting-started-with-regular-expressions/977/backtracking#t=20161001033913136116 3 http://www.riptutorial.com/ / / 259 / - / 4527 / # t = 20161001033959356491 3
- Jeff Atwood "Regex Performance"
- Andreas Haufler " Java "

Pitfall - == () .

:

" == ( )"

== . (== String.equals(...) .)

.

, == String . JLS, String . Java SE API String.intern(String) String.intern(String) . String , . .

'intern ()'

String . . (HashMap , HashTable) intern CPU .

equals == . "" .

(: , .)

CPU , .

7 Java "PermGen" . PermGen () . PermGen JVM .

Java 7 "PermGen" . , . .

. JVM .

. .

. intern .

Java 6 . Java 6 . (-XX:StringTableSize) Java 6 . Java 7 1009 60013 .

intern , . intern .

. DoS (Denial of Service) . intern O(N) . N .

(DoS DoS DoS .)

Pitfall - / . .

.

```
import java.io.*;

public class FileCopy {

    public static void main(String[] args) throws Exception {
        try (InputStream is = new FileInputStream(args[0]);
             OutputStream os = new FileOutputStream(args[1])) {
           int octet;
           while ((octet = is.read()) != -1) {
               os.write(octet);
           }
        }
    }
}
```

(, .)

. OS .

(*!*)

( ) 1 1 . . :

```
import java.io.*;

public class FileCopy {

    public static void main(String[] args) throws Exception {
        try (InputStream is = new BufferedInputStream(
                    new FileInputStream(args[0]));
             OutputStream os = new BufferedOutputStream(
                    new FileOutputStream(args[1]))) {
            int octet;
            while ((octet = is.read()) != -1) {
                os.write(octet);
            }
        }
    }
}
```

. . .

- `is` , . `read()` `read()` , , 1 . , read .
- `os` . `os.write(int)` `os.write(int)` 1 . `os` .

?

I / O API .

- `InputStream` `OutputStream` I / O API.
- `Reader` `Writer` I / O API.

I / O, `BufferedReader` `BufferedWriter` `BufferedInputStream` `BufferedOutputStream` .

?

.

- Java Java JVM . .
- JVM . " " . :
    1. syscall .
    2. SYSENTER .
    3. . .
    4. syscall .
    5. syscall . `read` .

1. .
2. ( ) ,
3. JVM
4. thstream

6. . VM .

. , . (3 .)

. `read()` syscall . `read()` `byte` . .

(I / O , . OS . .)

?

. "" . `byte[]` `char[]` . () .

Java ?

. Java API - . NIO `ByteBuffer` `Channel` API . .

Java - : https://riptutorial.com/ko/java/topic/5455/java-----

34: Java -

Java . .

Java . f .

Examples

-

Java Java 3 . 4! ( ) .

. API. (SOLID). .

public .

public . public . . public API . API .

Pitfall - " " .

Java . switch fallthrough . "" . "case 0" "break" , "Zero" "One" . "switch" "" . :

```
public static void switchCasePrimer() {
        int caseIndex = 0;
        switch (caseIndex) {
            case 0:
                System.out.println("Zero");
            case 1:
                System.out.println("One");
                break;
            case 2:
                System.out.println("Two");
                break;
            default:
                System.out.println("Default");
        }
}
```

( ) .

switch-statement ""fallthrough . .

```
switch(caseIndex) {
    [...]
    case 2:
        System.out.println("Two");
        // fallthrough
    default:
        System.out.println("Default");
```

-

Java . :

```
if (feeling == HAPPY)
    System.out.println("Smile");
else
    System.out.println("Frown");
```

:

```
if (feeling == HAPPY);
    System.out.println("Smile");
else
    System.out.println("Frown");
```

Java `else` . .

```
if (feeling == HAPPY)
    /*empty statement*/ ;
System.out.println("Smile");   // This is unconditional
else                           // This is misplaced.  A statement cannot
                               // start with 'else'
System.out.println("Frown");
```

. :

```
for (int i = 0; i < 5; i++);
    System.out.println("Hello");
```

"Hello" . , `for` . , `println` M.

:

```
for (int i = 0; i < 5; i++);
    System.out.println("The number is " + i);
```

`i` " " . `println` `i` .

. . .

1. Java " " . . .
2. . Java .
3. . IDE Java .
4. . Java "then" "else" . (`{`) .

, `if` .

```
if (feeling == HAPPY); {
    System.out.println("Smile");
} else {
    System.out.println("Frown");
}
```

. .

```
if (feeling == HAPPY); {
                           System.out.println("Smile");
                       } else {
                           System.out.println("Frown");
                       }
```

.

- : " " ""

Oracle Java Style Guide `if` "then" "else" "" "" . .

```
if (a) {           // <- open brace
    doSomething();
    doSomeMore();
}                  // <- close brace
```

Java . () "" , `if` ,

```
if (a)
    doSomething();
```

```
if (a) doSomething();
```

Java . .

"dangling if" :

.

```
if (a)
   doSomething();
   doSomeMore();
```

`doSomething doSomeMore a true true.,` `.doSomeMore() if` Java . .

```
if (a)
   doSomething();
doSomeMore();
```

"dangling else"

`else` . .

```
if (a)
   if (b)
      doX();
   else if (c)
      doY();
else
   doZ();
```

doZ a false., . .

```
if (a)
   if (b)
      doX();
   else if (c)
      doY();
   else
      doZ();
```

Java .

```
if (a) {
   if (b) {
      doX();
   } else if (c) {
      doY();
   } else {
      doZ();
   }
}
```

. .

```
if (a) {                        if (a) {
   if (b) {                         if (b) {
      doX();                           doX();
   } else if (c) {                  } else if (c) {
      doY();                           doY();
} else {                        } else {
   doZ();                           doZ();
}                                   }
}                               }
```

, Java " ".

-

.

```
public final class Person {
    private final String firstName;
    private final String lastName;

    public Person(String firstName, String lastName) {
        this.firstName = (firstName == null) ? "" : firstName;
        this.lastName = (lastName == null) ? "" : lastName;
```

```
    }

    public boolean equals(String other) {
        if (!(other instanceof Person)) {
            return false;
        }
        Person p = (Person) other;
        return firstName.equals(p.firstName) &&
                lastName.equals(p.lastName);
    }

    public int hashcode() {
        return firstName.hashCode() + 31 * lastName.hashCode();
    }
}
```

. Person equals hashcode Object .

- equals .equals(Object) equals(String) .
- hashcode .hashCode() (**C** hashCode() .

Person .

(Java 5). @Override .

Java SE 5

```
public final class Person {
    ...

    @Override
    public boolean equals(String other) {
        ....
    }

    @Override
    public hashcode() {
        ....
    }
}
```

@Override ( ) . . .

- 8

.

```
// Print the sum of the numbers 1 to 10
int count = 0;
for (int i = 1; i < 010; i++) {    // Mistake here ....
    count = count + i;
}
System.out.println("The sum of 1 to 10 is " + count);
```

Java . 1 - 8 .

('0') 8 . `010` 8 10 8.

Pitfall -

Java . :

```
package com.example;

/**
 * My string utilities
 */
public class String {
    ....
}
```

. :

```
package com.example;

public class Test {
    public static void main(String[] args) {
        System.out.println("Hello world!");
    }
}
```

.

```
$ javac com/example/*.java
$ java com.example.Test
Error: Main method not found in class test.Test, please define the main method as:
   public static void main(String[] args)
or a JavaFX application class must extend javafx.application.Application
```

`Test  main  java  . , java  .`

`Test  String  java.lang.String  . Test.main`

```
void main(com.example.String[] args)
```

```
void main(java.lang.String[] args)
```

`java  .`

: `java.lang` Java SE . , .

Pitfall - '=='

.

```
public void check(boolean ok) {
    if (ok == true) {           // Note 'ok == true'
```

```
        System.out.println("It is OK");
    }
}
```

.

```
public void check(boolean ok) {
    if (ok) {
       System.out.println("It is OK");
    }
}
```

`ok == true` . .

```
public void check(boolean ok) {
    if (ok = true) {           // Oooops!
        System.out.println("It is OK");
    }
}
```

`==` as `=` ... . . `x = true` `true` `x` `true` . , `check` "It is OK" .

`== false` `== true` . , .

: `ok == true` Yoda ., (`true == ok`). Yoda . `ok` (`!ok`) .

- .

.

```
import com.example.somelib.*;
import com.acme.otherlib.*;

public class Test {
    private Context x = new Context();   // from com.example.somelib
    ...
}
```

`somelib` 1.0 `somelib` 1.0 `otherlib` . `otherlib` 2.0 . 1.0 2.0 `otherlib` `Context` `otherlib` .

`Test` `Context` .

, . , . .

. . :

- , Java .
- . IDE .

. ( ) IDE .

Pitfall : 'assert'

StackOverflow `assert` .

.

.

- IllegalArgumentException Throw
- Google Guava `Preconditions`
- Apache Commons Lang3 `Validate` .

Java (JLS 14.10, Java 8) .

.

. . . .

, . , .

(: `IllegalArgumentException` , `ArrayIndexOutOfBoundsException` `NullPointerException`) .
throw . . `AssertionError` . try throw try .

Null

```
public class Foobar {
    public static void main(String[] args) {

        // example:
        Boolean ignore = null;
        if (ignore == false) {
            System.out.println("Do not ignore!");
        }
    }
}
```

`null` `false` . `boolean` `Boolean` , `Boolean` `Object` . `null` `NullPointerException` .

Java `NullPointerException` `null` . . `false == null` ; `incomparable types: int and <null>` .

Java - : https://riptutorial.com/ko/java/topic/5382/java-----

35: Java -

Java . .

Examples

-

"" . . .

(" ") ( ) . ( ) . . . .

IDE "" . .

```
try {
    inputStream = new FileInputStream("someFile");
} catch (IOException e) {
    /* add exception handling code here */
}
```

IDE . (, . "", ; `inputStream null NullPointerException` .)

, . . .

```
try {
    selfie.show();
} catch (InterruptedException e) {
    // It doesn't matter if showing the selfie is interrupted.
}
```

.

```
try {
    selfie.show();
} catch (InterruptedException ignored) {  }
```

`ignored` IntelliJ IDEA IDE catch .

- Throwable, Exception, Error RuntimeException

"" "" . .

```
....
try {
    InputStream is = new FileInputStream(fileName);
    // process the input
} catch (Exception ex) {
    System.out.println("Could not open file " + fileName);
}
```

. catch . fileName null . FileInputStream NullPointerException . catch .

```
    Could not open file null
```

. , ( ) " " . I / O .

Exception . .

- Catching Exception catch.
- RuntimeException catch.
- Catching Error JVM . .
- Catching Throwable catch.

. Exception . .

, . , .

```
try {
    InputStream is = new FileInputStream(fileName);
    // process the input
} catch (FileNotFoundException ex) {
    System.out.println("Could not open file " + fileName);
}
```

thrown .

Exception . .

```
public static void main(String[] args) {
    try {
        // do stuff
    } catch (Exception ex) {
        System.err.println("Unfortunately an error has occurred. " +
                           "Please report this to X Y Z");
        // Write stacktrace to a log file.
        System.exit(1);
    }
}
```

. Exception (Throwable) .

---

1 - .

- Throwable, Exception, Error RuntimeException Throw

Throwable , Exception , Error RuntimeException .

.

```
try {
```

```
    InputStream is = new FileInputStream(someFile);  // could throw IOException
    ...
    if (somethingBad) {
        throw new Exception();  // WRONG
    }
} catch (IOException ex) {
    System.err.println("cannot open ...");
} catch (Exception ex) {
    System.err.println("something bad happened");  // WRONG
}
```

`Exception` . `Exception` . `somethingBad` `true` throw `Exception` " " `NullPointerException` .

.

- / .
- `Exception` `Throwable` `throws` . .

, . " " . .

"throws" Throwable Exception .

`throws` `Exception` Throwable . .

1. `Exception` ( ).
2. .
3. `Exception` . . .
4. `Throwable` . .

. :

```
try {
    doSomething();
} catch (Exception ex) {
    report(ex);
    throw ex;
}
```

. , Java 7 `throw ex;` `Exception` . `throws Exception` `throws Exception` . Java 7 ( ) .

- InterruptedException

```
try {
    // Some code
} catch (Exception) {
    // Some error handling
}
```

. perticular .

.

```
Thread t = new Thread(new Runnable() {
    public void run() {
         while (true) {
             //Do something indefinetely
         }
    }
}

t.start();

//Do something else

// The thread should be canceld if it is still active.
// A Better way to solve this is with a shared variable that is tested
// regularily by the thread for a clean exit, but for this example we try to
// forcibly interrupt this thread.
if (t.isAlive()) {
   t.interrupt();
   t.join();
}

//Continue with program
```

`t.interrupt()` InterruptedException . ? InterruptedException catch .

```
 Thread t = new Thread(new Runnable() {
    public void run() {
        try {
            while (true) {
                //Do something indefinetely
            }
        } catch (InterruptedException ex) {
            //Do some quick cleanup

            // In this case a simple return would do.
            // But if you are not 100% sure that the thread ends after
            // catching the InterruptedException you will need to raise another
            // one for the layers surrounding this code.
            Thread.currentThread().interrupt();
        }
    }
}
```

catch-all InterruptedException catch . `t.join()` `t.join()` .

```
 Thread t = new Thread(new Runnable() {
    public void run() {
        try {
            while (true) {
                try {
                    //Do something indefinetely
                }
                catch (Exception ex) {
                    ex.printStackTrace();
                }
            }
        } catch (InterruptedException ex) {
            // Dead code as the interrupt exception was already caught in
            // the inner try-catch
```

```
            Thread.currentThread().interrupt();
        }
    }
}
```

. catch-all InterruptedException catch.

```
Thread t = new Thread(new Runnable() {
    public void run() {
        try {
            while (true) {
                try {
                    //Do something indefinetely
                } catch (InterruptedException ex) {
                    throw ex; //Send it up in the chain
                } catch (Exception ex) {
                    ex.printStackTrace();
                }
            }
        } catch (InterruptedException ex) {
            // Some quick cleanup code

            Thread.currentThread().interrupt();
        }
    }
}
```

Pitfall - flowcontrol

Java .

" ."

(: http://programmers.stackexchange.com/questions/184654)

(Java). null .

```
public String truncateWordOrNull(String word, int maxLength) {
    if (word == null) {
        return "";
    } else {
        return word.substring(0, Math.min(word.length(), maxLength));
    }
}

public String truncateWordOrNull(String word, int maxLength) {
    try {
        return word.substring(0, Math.min(word.length(), maxLength));
    } catch (NullPointerException ex) {
        return "";
    }
}
```

() `word null` . *if ... else* *try ... catch* `null` . ?

. . `Math.min String.substring NullPointerException` throw .

. Java 8 Java . , .

" " . , " " . . .

-

`printStackTrace()` .

`printStackTrace()` stacktrace .

- Java .
- .

`printStackTrace` .

. , injudiciously . :

```
public void method1() throws SomeException {
    try {
        method2();
        // Do something
    } catch (SomeException ex) {
        Logger.getLogger().warn("Something bad in method1", ex);
        throw ex;
    }
}

public void method2() throws SomeException {
    try {
        // Do something else
    } catch (SomeException ex) {
        Logger.getLogger().warn("Something bad in method2", ex);
        throw ex;
    }
}
```

`method2` .

, ( ). .

Pitfall -`Throwable`

`Throwable` 2 `Exception` `Error` . `Throwable` `Exception` `Error` .

`Throwable` . `Exception` `Exception` .

Java - : https://riptutorial.com/ko/java/topic/5381/java-----

36: java.util.Objects

Examples

null

```
Object nullableObject = methodReturnObject();
if (Objects.isNull(nullableObject)) {
    return;
}
```

not null

```
Object nullableObject = methodReturnObject();
if (Objects.nonNull(nullableObject)) {
    return;
}
```

API Objects.nonNull ()

null

```
List<Object> someObjects = methodGetList();
for (Object obj : someObjects) {
    if (obj == null) {
        continue;
    }
    doSomething(obj);
}
```

`Objects.nonNull` Java8 Stream API .

```
List<Object> someObjects = methodGetList();
someObjects.stream()
           .filter(Objects::nonNull)
           .forEach(this::doSomething);
```

java.util.Objects : https://riptutorial.com/ko/java/topic/5768/java-util-objects-

37: Java

Java . JSR223 (Java Specification Request 223)

Java Scripting API Java .

Scripting API java . .

JavaScript (ECMAScript) nashorn . , . Writer .

JRuby . .

. , . . EngineFactory

Examples

nashorn

```
public class JSEngine {

    /*
    * Note Nashorn is only available for Java-8 onwards
    * You can use rhino from ScriptEngineManager.getEngineByName("js");
    */

    ScriptEngine engine;
    ScriptContext context;
    public Bindings scope;

    // Initialize the Engine from its factory in scripting mode
    public JSEngine(){
        engine = new NashornScriptEngineFactory().getScriptEngine("-scripting");
        // Script context is an interface so we need an implementation of it
        context = new SimpleScriptContext();
        // Create bindings to expose variables into
        scope = engine.createBindings();
    }

    // Clear the bindings to remove the previous variables
    public void newBatch(){
        scope.clear();
    }

    public void execute(String file){
        try {
            // Get a buffered reader for input
            BufferedReader br = new BufferedReader(new FileReader(file));
            // Evaluate code, with input as bufferdReader
            engine.eval(br);
        } catch (FileNotFoundException ex) {
            Logger.getLogger(JSEngine.class.getName()).log(Level.SEVERE, null, ex);
        } catch (ScriptException ex) {
            // Script Exception is basically when there is an error in script
            Logger.getLogger(JSEngine.class.getName()).log(Level.SEVERE, null, ex);
```

```
        }
    }

    public void eval(String code){
        try {
            // Engine.eval basically treats any string as a line of code and evaluates it,
executes it
            engine.eval(code);
        } catch (ScriptException ex) {
            // Script Exception is basically when there is an error in script
            Logger.getLogger(JSEngine.class.getName()).log(Level.SEVERE, null, ex);
        }
    }

    // Apply the bindings to the context and set the engine's default context
    public void startBatch(int SCP){
        context.setBindings(scope, SCP);
        engine.setContext(context);
    }

    // We use the invocable interface to access methods from the script
    // Invocable is an optional interface, please check if your engine implements it
    public Invocable invocable(){
        return (Invocable)engine;
    }

}
```

```
public static void main(String[] args) {
    JSEngine jse = new JSEngine();
    // Create a new batch probably unecessary
    jse.newBatch();
    // Expose variable x into script with value of hello world
    jse.scope.put("x", "hello world");
    // Apply the bindings and start the batch
    jse.startBatch(ScriptContext.ENGINE_SCOPE);
    // Evaluate the code
    jse.eval("print(x);");
}
```

.

```
hello world
```

x . .

test.js .

```
print(x);
function test(){
    print("hello test.js:test");
}
test();
```

main

```
public static void main(String[] args) {
    JSEngine jse = new JSEngine();
```

```
    // Create a new batch probably unecessary
    jse.newBatch();
    // Expose variable x into script with value of hello world
    jse.scope.put("x", "hello world");
    // Apply the bindings and start the batch
    jse.startBatch(ScriptContext.ENGINE_SCOPE);
    // Evaluate the code
    jse.execute("./test.js");
}
```

test.js . .

```
hello world
hello test.js:test
```

Java : https://riptutorial.com/ko/java/topic/9926/java----

38: Java JSON

JSON (JavaScript Object Notation) . JSON . JSON Ajax , , RESTful . JSON Java API JSON , , API .

Google Gson , Jackson Object Mapper Java JSON .

. Java JSON

Examples

JSON

JSONObject .

```
// Create a new javax.json.JSONObject instance.
JSONObject first = new JSONObject();

first.put("foo", "bar");
first.put("temperature", 21.5);
first.put("year", 2016);

// Add a second object.
JSONObject second = new JSONObject();
second.put("Hello", "world");
first.put("message", second);

// Create a new JSONArray with some values
JSONArray someMonths = new JSONArray(new String[] { "January", "February" });
someMonths.put("March");
// Add another month as the fifth element, leaving the 4th element unset.
someMonths.put(4, "May");

// Add the array to our object
object.put("months", someMonths);

// Encode
String json = object.toString();

// An exercise for the reader: Add pretty-printing!
/* {
       "foo":"bar",
       "temperature":21.5,
       "year":2016,
       "message":{"Hello":"world"},
       "months":["January","February","March",null,"May"]
   }
*/
```

JSON

JSONObject JSONObject .

```
String json =
"{\"foo\":\"bar\",\"temperature\":21.5,\"year\":2016,\"message\":{\"Hello\":\"world\"},\"months\":[\"Ja


// Decode the JSON-encoded string
JSONObject object = new JSONObject(json);

// Retrieve some values
String foo = object.getString("foo");
double temperature = object.getDouble("temperature");
int year = object.getInt("year");

// Retrieve another object
JSONObject secondary = object.getJSONObject("message");
String world = secondary.getString("Hello");

// Retrieve an array
JSONArray someMonths = object.getJSONArray("months");
// Get some values from the array
int nMonths = someMonths.length();
String february = someMonths.getString(1);
```

optXXX getXXX

JSONObject JSONArray .

```
JSONObject obj = new JSONObject();
obj.putString("foo", "bar");

// For existing properties of the correct type, there is no difference
obj.getString("foo");        // returns "bar"
obj.optString("foo");        // returns "bar"
obj.optString("foo", "tux"); // returns "bar"

// However, if a value cannot be coerced to the required type, the behavior differs
obj.getInt("foo");      // throws JSONException
obj.optInt("foo");      // returns 0
obj.optInt("foo", 123); // returns 123

// Same if a property does not exist
obj.getString("undefined");        // throws JSONException
obj.optString("undefined");        // returns ""
obj.optString("undefined", "tux"); // returns "tux"
```

JSONArray getXXX / optXXX .

JSON (Gson)

name Person .

```
private class Person {
    public String name;

    public Person(String name) {
        this.name = name;
    }
```

```
}
```

:

```
Gson g = new Gson();

Person person = new Person("John");
System.out.println(g.toJson(person)); // {"name":"John"}
```

Gson classpath .

JSON To Object (Gson)

name Person .

```
private class Person {
    public String name;

    public Person(String name) {
        this.name = name;
    }
}
```

:

```
Gson gson = new Gson();
String json = "{\"name\": \"John\"}";

Person person = gson.fromJson(json, Person.class);
System.out.println(person.name); //John
```

gson .

JSON

```
String json = "{\"name\": \"John\", \"age\":21}";

JsonObject jsonObject = new JsonParser().parse(json).getAsJsonObject();

System.out.println(jsonObject.get("name").getAsString()); //John
System.out.println(jsonObject.get("age").getAsInt()); //21
```

Jackson Object Mapper

```
public class Model {
    private String firstName;
    private String lastName;
    private int age;
    /* Getters and setters not shown for brevity */
}
```

: String to Object

```
Model outputObject = objectMapper.readValue(
     "{\"firstName\":\"John\",\"lastName\":\"Doe\",\"age\":23}",
     Model.class);
System.out.println(outputObject.getFirstName());
//result: John
```

: Object to String

```
String jsonString = objectMapper.writeValueAsString(inputObject));
//result: {"firstName":"John","lastName":"Doe","age":23}
```

---

import :

```
import com.fasterxml.jackson.databind.ObjectMapper;
```

: jackson-databind

`ObjectMapper`

```
//creating one
ObjectMapper objectMapper = new ObjectMapper();
```

- `ObjectMapper` threadsafe.
- : .

:

```
<T> T readValue(String content, Class<T> valueType)
```

- `valueType` . .
-
  - `IOException` -
  - `JsonParseException` -
  - `JsonMappingException` - JSON

(jsonString .) :

```
Model fromJson = objectMapper.readValue(jsonString, Model.class);
```

:

writeValueAsString ( )

-
  - `JsonProcessingException`
  - : 2.1 , IOException throws . 2.1 .

JSON

JSONObject

```
JSONObject obj = new JSONObject("{\"isMarried\":\"true\", \"name\":\"Nikita\",
\"age\":\"30\"}");
Iterator<String> keys = obj.keys();//all keys: isMarried, name & age
while (keys.hasNext()) {                        //as long as there is another key
      String key = keys.next();                 //get next key
      Object value = obj.get(key);              //get next value by key
      System.out.println(key + " : " + value);//print key : value
}
```

JSONArray

```
JSONArray arr = new JSONArray();        //Initialize an empty array
//push (append) some values in:
arr.put("Stack");
arr.put("Over");
arr.put("Flow");
for (int i = 0; i < arr.length(); i++) {//iterate over all values
    Object value = arr.get(i);          //get value
    System.out.println(value);          //print each value
}
```

JSON Builder -

`JSONObject` `JSONArray` .

JSONObject

```
JSONObject obj = new JSONObject();//Initialize an empty JSON object
//Before: {}
obj.put("name","Nikita").put("age","30").put("isMarried","true");
//After: {"name":"Nikita","age":30,"isMarried":true}
```

JSONArray

```
JSONArray arr = new JSONArray();//Initialize an empty array
//Before: []
arr.put("Stack").put("Over").put("Flow");
//After: ["Stack","Over","Flow"]
```

JSONObject.NULL

`null` `JSONObject.NULL` Java `null` .

`JSONObject.NULL` .

```
JSONObject obj = new JSONObject();
obj.put("some", JSONObject.NULL);   //Creates: {"some":null}
System.out.println(obj.get("some"));//prints: null
```

```
JSONObject.NULL.equals(null); //returns true
```

`Java.equals()` .

null x , x.equals (null) false .

JsonArray Java (Gson)

Java `ArrayList` JsonArray.

```
{
    "list": [
                "Test_String_1",
                "Test_String_2"
            ]
}
```

`JsonArray` 'list' Java `ArrayList` .

```
public ArrayList<String> getListString(String jsonList){
    Type listType = new TypeToken<List<String>>() {}.getType();
    //make sure the name 'list' matches the name of 'JsonArray' in your 'Json'.
    ArrayList<String> list = new Gson().fromJson(jsonList, listType);
    return list;
}
```

`POM.xml` Maven .

```
<!-- https://mvnrepository.com/artifact/com.google.code.gson/gson -->
<dependency>
    <groupId>com.google.code.gson</groupId>
    <artifactId>gson</artifactId>
    <version>2.7</version>
</dependency>
```

classpath `com.google.code.gson:gson:jar:<version>` .

Jackson JSON

pojo `Person` .

```
public class Person {
    public String name;

    public Person(String name) {
        this.name = name;
    }
}
```

JSON Person . `List<Person>` `Map<String, Person>` *(JSON deserialize)* . `TypeFactory` `TypeReference` .

TypeFactory

. (,) .

. TypeReference `List<Person>` . TypeReference .

JSON Java . JSON . . .

JSON

```
String jsonString = "[{\"name\": \"Alice\"}, {\"name\": \"Bob\"}]"
```

```
CollectionType listType =
    factory.constructCollectionType(List.class, Person.class);
List<Preson> list = mapper.readValue(jsonString, listType);
```

```
TypeReference<Person> listType = new TypeReference<List<Person>>() {};
List<Person> list = mapper.readValue(jsonString, listType);
```

JSON

```
String jsonString = "{\"0\": {\"name\": \"Alice\"}, \"1\": {\"name\": \"Bob\"}}"
```

```
CollectionType mapType =
    factory.constructMapLikeType(Map.class, String.class, Person.class);
List<Person> list = mapper.readValue(jsonString, mapType);
```

```
TypeReference<Person> mapType = new TypeReference<Map<String, Person>>() {};
Map<String, Person> list = mapper.readValue(jsonString, mapType);
```

---

import :

```
import com.fasterxml.jackson.core.type.TypeReference;
import com.fasterxml.jackson.databind.ObjectMapper;
import com.fasterxml.jackson.databind.type.CollectionType;
```

:

```
ObjectMapper mapper = new ObjectMapper();
TypeFactory factory = mapper.getTypeFactory();
```

`TypeReference` .

1. `TypeReference` .
2.

.

.

Java JSON : https://riptutorial.com/ko/java/topic/840/java-json

39: JAXB

JAXB JAXB (Java Architecture for XML Binding) Java Java XML . XML JAXB .

- JAXB.marshall (object, fileObjOfXML);
- Object obj = JAXB.unmarshall (fileObjOfXML, className);

fileObjOfXML	XML `File`
className	`.class`

JDK XJC xml (`.xsd`) xml Java (XJC).

Examples

XML ( )

```
import javax.xml.bind.annotation.XmlRootElement;

@XmlRootElement
public class User {

    private long userID;
    private String name;

    // getters and setters
}
```

`XMLRootElement` XML .

```
import java.io.File;
import javax.xml.bind.JAXB;

public class XMLCreator {
    public static void main(String[] args) {
        User user = new User();
        user.setName("Jon Skeet");
        user.setUserID(8884321);

        try {
            JAXB.marshal(user, new File("UserDetails.xml"));
        } catch (Exception e) {
            System.err.println("Exception occurred while writing in XML!");
        } finally {
            System.out.println("XML created");
        }
    }
}
```

`marshal()` XML . `user` `File` `marshal()` .

class-path `UserDetails.xml` XML .

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<user>
    <name>Jon Skeet</name>
    <userID>8884321</userID>
</user>
```

XML ( )

`UserDetails.xml` XML

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<user>
    <name>Jon Skeet</name>
    <userID>8884321</userID>
</user>
```

`User.java` POJO .

```
import javax.xml.bind.annotation.XmlRootElement;

@XmlRootElement
public class User {

    private long userID;
    private String name;

    // getters and setters
}
```

XML . `XmlRootElement` .

```
public class XMLReader {
    public static void main(String[] args) {
        try {
            User user = JAXB.unmarshal(new File("UserDetails.xml"), User.class);
            System.out.println(user.getName());   // prints Jon Skeet
            System.out.println(user.getUserID()); // prints 8884321
        } catch (Exception e) {
            System.err.println("Exception occurred while reading the XML!");
        }
    }
}
```

`unmarshal()` XML . XML . getter .

XmlAdapter xml

XML Java XmlAdapter XML . .

```
public class XmlAdapterExample {

    @XmlAccessorType(XmlAccessType.FIELD)
    public static class NodeValueElement {

        @XmlAttribute(name="attrValue")
        String value;

        public NodeValueElement() {
        }

        public NodeValueElement(String value) {
            super();
            this.value = value;
        }

        public String getValue() {
            return value;
        }

        public void setValue(String value) {
            this.value = value;
        }
    }

    public static class ValueAsAttrXmlAdapter extends XmlAdapter<NodeValueElement, String> {

        @Override
        public NodeValueElement marshal(String v) throws Exception {
            return new NodeValueElement(v);
        }

        @Override
        public String unmarshal(NodeValueElement v) throws Exception {
            if (v==null) return "";
            return v.getValue();
        }
    }

    @XmlRootElement(name="DataObject")
    @XmlAccessorType(XmlAccessType.FIELD)
    public static class DataObject {

        String elementWithValue;

        @XmlJavaTypeAdapter(value=ValueAsAttrXmlAdapter.class)
        String elementWithAttribute;
    }

    public static void main(String[] args) {
        DataObject data = new DataObject();
        data.elementWithValue="value1";
        data.elementWithAttribute ="value2";

        ByteArrayOutputStream baos = new ByteArrayOutputStream();
        JAXB.marshal(data, baos);

        String xmlString = new String(baos.toByteArray(), StandardCharsets.UTF_8);

        System.out.println(xmlString);
    }
```

```
}
```

/ XML (@XmlAccessorType)

Annotation `@XmlAccessorType` / XML serialize . `@XmlElement` , `@XmlAttribute` `@XmlTransient` .

```
public class XmlAccessTypeExample {

@XmlAccessorType(XmlAccessType.FIELD)
static class AccessorExampleField {
    public String field="value1";

    public String getGetter() {
        return "getter";
    }

    public void setGetter(String value) {}
}

@XmlAccessorType(XmlAccessType.NONE)
static class AccessorExampleNone {
    public String field="value1";

    public String getGetter() {
        return "getter";
    }

    public void setGetter(String value) {}
}

@XmlAccessorType(XmlAccessType.PROPERTY)
static class AccessorExampleProperty {
    public String field="value1";

    public String getGetter() {
        return "getter";
    }

    public void setGetter(String value) {}
}

@XmlAccessorType(XmlAccessType.PUBLIC_MEMBER)
static class AccessorExamplePublic {
    public String field="value1";

    public String getGetter() {
        return "getter";
    }

    public void setGetter(String value) {}
}

public static void main(String[] args) {
    try {
        System.out.println("\nField:");
        JAXB.marshal(new AccessorExampleField(), System.out);
        System.out.println("\nNone:");
        JAXB.marshal(new AccessorExampleNone(), System.out);
        System.out.println("\nProperty:");
```

```
        JAXB.marshal(new AccessorExampleProperty(), System.out);
        System.out.println("\nPublic:");
        JAXB.marshal(new AccessorExamplePublic(), System.out);
    } catch (Exception e) {
        System.err.println("Exception occurred while writing in XML!");
    }
}

} // outer class end
```

```
Field:
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<accessorExampleField>
    <field>value1</field>
</accessorExampleField>

None:
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<accessorExampleNone/>

Property:
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<accessorExampleProperty>
    <getter>getter</getter>
</accessorExampleProperty>

Public:
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<accessorExamplePublic>
    <field>value1</field>
    <getter>getter</getter>
</accessorExamplePublic>
```

/ XML

`@XmlElement` , `@XmlAttribute` `@XmlTransient` `javax.xml.bind.annotation` `@XmlElement` , (
`@XmlTransient` .

```
@XmlAccessorType(XmlAccessType.NONE) // we want no automatic field/property marshalling
public class ManualXmlElementsExample {

    @XmlElement
    private String field="field value";

    @XmlAttribute
    private String attribute="attr value";

    @XmlAttribute(name="differentAttribute")
    private String oneAttribute="other attr value";

    @XmlElement(name="different name")
    private String oneName="different name value";

    @XmlTransient
    private String transientField = "will not get serialized ever";


    @XmlElement
```

```
    public String getModifiedTransientValue() {
        return transientField.replace(" ever", ", unless in a getter");
    }

    public void setModifiedTransientValue(String val) {} // empty on purpose


    public static void main(String[] args) {
        try {
            JAXB.marshal(new ManualXmlElementsExample(), System.out);
        } catch (Exception e) {
            System.err.println("Exception occurred while writing in XML!");
        }
    }
}
```

XmlAdapter

. static .

Unmarshaller XmlAdapter . 0 XmlAdapter / .

.

```
import java.awt.image.BufferedImage;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.XmlRootElement;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

@XmlRootElement
public class User {

    private String name;
    private BufferedImage image;

    @XmlAttribute
    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    @XmlJavaTypeAdapter(value=ImageCacheAdapter.class)
    @XmlAttribute
    public BufferedImage getImage() {
        return image;
    }

    public void setImage(BufferedImage image) {
        this.image = image;
    }

    public User(String name, BufferedImage image) {
        this.name = name;
        this.image = image;
    }
```

```
    public User() {
        this("", null);
    }

}
```

.

Java SE 7

Java 7 `getImage` .

```
public BufferedImage getImage(URL url) {
    BufferedImage image = imageCache.get(url);
    if (image == null) {
        try {
            image = ImageIO.read(url);
        } catch (IOException ex) {
            Logger.getLogger(ImageCacheAdapter.class.getName()).log(Level.SEVERE, null, ex);
            return null;
        }
        imageCache.put(url, image);
        reverseIndex.put(image, url);
    }
    return image;
}
```

```
import java.awt.image.BufferedImage;
import java.io.IOException;
import java.net.URL;
import java.util.HashMap;
import java.util.Map;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.imageio.ImageIO;
import javax.xml.bind.annotation.adapters.XmlAdapter;

public class ImageCacheAdapter extends XmlAdapter<String, BufferedImage> {

    private final Map<URL, BufferedImage> imageCache = new HashMap<>();
    private final Map<BufferedImage, URL> reverseIndex = new HashMap<>();

    public BufferedImage getImage(URL url) {
        // using a single lookup using Java 8 methods
        return imageCache.computeIfAbsent(url, s -> {
            try {
                BufferedImage img = ImageIO.read(s);
                reverseIndex.put(img, s);
                return img;
            } catch (IOException ex) {
                Logger.getLogger(ImageCacheAdapter.class.getName()).log(Level.SEVERE, null,
ex);
                return null;
            }
        });
    }
```

```
    @Override
    public BufferedImage unmarshal(String v) throws Exception {
        return getImage(new URL(v));
    }

    @Override
    public String marshal(BufferedImage v) throws Exception {
        return reverseIndex.get(v).toExternalForm();
    }

}
```

XML

2 xml *Jon Skeet* 2 . .

```
<?xml version="1.0" encoding="UTF-8"?>

<user name="Jon Skeet"
image="https://www.gravatar.com/avatar/6d8ebb117e8d83d74ea95fbdd0f87e13?s=328&amp;d=identicon&amp;r=PG"
```

```
<?xml version="1.0" encoding="UTF-8"?>

<user name="Jon Skeet (Earth 2)"
image="https://www.gravatar.com/avatar/6d8ebb117e8d83d74ea95fbdd0f87e13?s=328&amp;d=identicon&amp;r=PG"
```

```
ImageCacheAdapter adapter = new ImageCacheAdapter();

JAXBContext context = JAXBContext.newInstance(User.class);

Unmarshaller unmarshaller = context.createUnmarshaller();

// specifiy the adapter instance to use for every
// @XmlJavaTypeAdapter(value=ImageCacheAdapter.class)
unmarshaller.setAdapter(ImageCacheAdapter.class, adapter);

User result1 = (User) unmarshaller.unmarshal(Main.class.getResource("user.xml"));

// unmarshal second xml using the same adapter instance
Unmarshaller unmarshaller2 = context.createUnmarshaller();
unmarshaller2.setAdapter(ImageCacheAdapter.class, adapter);
User result2 = (User) unmarshaller2.unmarshal(Main.class.getResource("user2.xml"));

System.out.println(result1.getName());
System.out.println(result2.getName());

// yields true, since image is reused
System.out.println(result1.getImage() == result2.getImage());
```

XML Java .

XML Java `package-info.java` . Java .

```
/**
 * A package containing serializable classes.
 */
@XmlSchema
(
    xmlns =
    {
        @XmlNs(prefix = MySerializableClass.NAMESPACE_PREFIX, namespaceURI =
MySerializableClass.NAMESPACE)
    },
    namespace = MySerializableClass.NAMESPACE,
    elementFormDefault = XmlNsForm.QUALIFIED
)
package com.test.jaxb;

import javax.xml.bind.annotation.XmlNs;
import javax.xml.bind.annotation.XmlNsForm;
import javax.xml.bind.annotation.XmlSchema;
```

XmlAdapter .

```
package com.example.xml.adapters;

import javax.xml.bind.annotation.adapters.XmlAdapter;

public class StringTrimAdapter extends XmlAdapter<String, String> {
    @Override
    public String unmarshal(String v) throws Exception {
        if (v == null)
            return null;
        return v.trim();
    }

    @Override
    public String marshal(String v) throws Exception {
        if (v == null)
            return null;
        return v.trim();
    }
}
```

package-info.java .

```
@XmlJavaTypeAdapter(value = com.example.xml.adapters.StringTrimAdapter.class, type =
String.class)
package com.example.xml.jaxb.bindings;// Packge where you intend to apply trimming filter

import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;
```

JAXB : https://riptutorial.com/ko/java/topic/147/jaxb

40: JAXP API XML

XML "", "", "", " " XML .

Java JAXP XML API XML Java API . JAXP Java 1.4 (JAXP v1.1) Java . Java 8 JAXP 1.6 .

API XML .

- DOM (Document Object Model)
- SAX (XML Simple API)
- StAX (XML API)

DOM

DOM XML W3C DOM . JAXP DOM (3) .

Document Object Model XML "Document Element" . API `Node` , `Node` , (`Node` , `Text` . .) XML `Element` , `Node` .

DOM XML " " , (, ,) , ( ) .) . . DOM XML . XML . XML .

SAX

SAX API XML API. XML (: " ", " ", " ", " "). ..

SAX API "push parsing" . SAX `Parser` XML XML (`ContentHandler`) . XML .

SAX (: ,) DOM `ContentHandler` . " / XML " . DOM `Node` SAX .

StAX

StAX API XML SAX API (,) . StAX (SAX). SAX `Parser` `ContentHandler` . Stax XML "" / .

API XMLStreamReader (XMLEventReader) , `nextEvent()` .

Examples

DOM API

.

```
<?xml version='1.0' encoding='UTF-8' ?>
<library>
   <book id='1'>Effective Java</book>
   <book id='2'>Java Concurrency In Practice</book>
</library>
```

`String` DOM .

```
import org.w3c.dom.Document;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;
import org.xml.sax.InputSource;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import java.io.StringReader;

public class DOMDemo {

public static void main(String[] args) throws Exception {
    String xmlDocument = "<?xml version='1.0' encoding='UTF-8' ?>"
            + "<library>"
            + "<book id='1'>Effective Java</book>"
            + "<book id='2'>Java Concurrency In Practice</book>"
            + "</library>";

    DocumentBuilderFactory documentBuilderFactory = DocumentBuilderFactory.newInstance();
    // This is useless here, because the XML does not have namespaces, but this option is
usefull to know in cas
    documentBuilderFactory.setNamespaceAware(true);
    DocumentBuilder documentBuilder = documentBuilderFactory.newDocumentBuilder();
    // There are various options here, to read from an InputStream, from a file, ...
    Document document = documentBuilder.parse(new InputSource(new StringReader(xmlDocument)));

    // Root of the document
    System.out.println("Root of the XML Document: " +
document.getDocumentElement().getLocalName());

    // Iterate the contents
    NodeList firstLevelChildren = document.getDocumentElement().getChildNodes();
    for (int i = 0; i < firstLevelChildren.getLength(); i++) {
        Node item = firstLevelChildren.item(i);
        System.out.println("First level child found, XML tag name is: " +
item.getLocalName());
        System.out.println("\tid attribute of this tag is : " +
item.getAttributes().getNamedItem("id").getTextContent());
    }

    // Another way would have been
    NodeList allBooks = document.getDocumentElement().getElementsByTagName("book");
}
}
```

.

```
Root of the XML Document: library
First level child found, XML tag name is: book
id attribute of this tag is : 1
First level child found, XML tag name is: book
```

```
id attribute of this tag is : 2
```

StAX API

.

```
<?xml version='1.0' encoding='UTF-8' ?>
<library>
   <book id='1'>Effective Java</book>
   <book id='2'>Java Concurrency In Practice</book>
   <notABook id='3'>This is not a book element</notABook>
</library>
```

ID .

```
import javax.xml.stream.XMLInputFactory;
import javax.xml.stream.XMLStreamConstants;
import javax.xml.stream.XMLStreamReader;
import java.io.StringReader;
import java.util.HashMap;
import java.util.Map;

public class StaxDemo {

public static void main(String[] args) throws Exception {
    String xmlDocument = "<?xml version='1.0' encoding='UTF-8' ?>"
            + "<library>"
                + "<book id='1'>Effective Java</book>"
                + "<book id='2'>Java Concurrency In Practice</book>"
                + "<notABook id='3'>This is not a book element </notABook>"
            + "</library>";

    XMLInputFactory xmlInputFactory = XMLInputFactory.newFactory();
    // Various flavors are possible, e.g. from an InputStream, a Source, ...
    XMLStreamReader xmlStreamReader = xmlInputFactory.createXMLStreamReader(new
StringReader(xmlDocument));

    Map<Integer, String> bookTitlesById = new HashMap<>();

    // We go through each event using a loop
    while (xmlStreamReader.hasNext()) {
        switch (xmlStreamReader.getEventType()) {
            case XMLStreamConstants.START_ELEMENT:
                System.out.println("Found start of element: " +
xmlStreamReader.getLocalName());
                // Check if we are at the start of a <book> element
                if ("book".equals(xmlStreamReader.getLocalName())) {
                    int bookId = Integer.parseInt(xmlStreamReader.getAttributeValue("",
"id"));
                    String bookTitle = xmlStreamReader.getElementText();
                    bookTitlesById.put(bookId, bookTitle);
                }
                break;
            // A bunch of other things are possible : comments, processing instructions,
Whitespace...
            default:
                break;
        }
```

```
            xmlStreamReader.next();
        }

        System.out.println(bookTitlesById);
    }
```

.

```
Found start of element: library
Found start of element: book
Found start of element: book
Found start of element: notABook
{1=Effective Java, 2=Java Concurrency In Practice}
```

.

1. `xmlStreamReader.getAttributeValue` `START_ELEMENT` . (`ATTRIBUTES`), () , `IllegalStateException` (`ATTRIBUTES`) .
2. `xmlStreamReader.getTextContent()` . `START_ELEMENT` . `<book>` .

( ) `BookParser` : `BookParser`) "" `BookParser` XML START_ELEMENT END_ELEMENT

`Stack` .

JAXP API XML : https://riptutorial.com/ko/java/topic/3943/jaxp-api--xml--

41: JAX-WS

Examples

JAX-WS .

Service Port .

```
Service s = new Service();
Port port = s.getPort();

BindingProvider prov = (BindingProvider)port;
prov.getRequestContext().put(BindingProvider.USERNAME_PROPERTY, "myusername");
prov.getRequestContext().put(BindingProvider.PASSWORD_PROPERTY, "mypassword");

port.call();
```

JAX-WS : https://riptutorial.com/ko/java/topic/4105/jax-ws

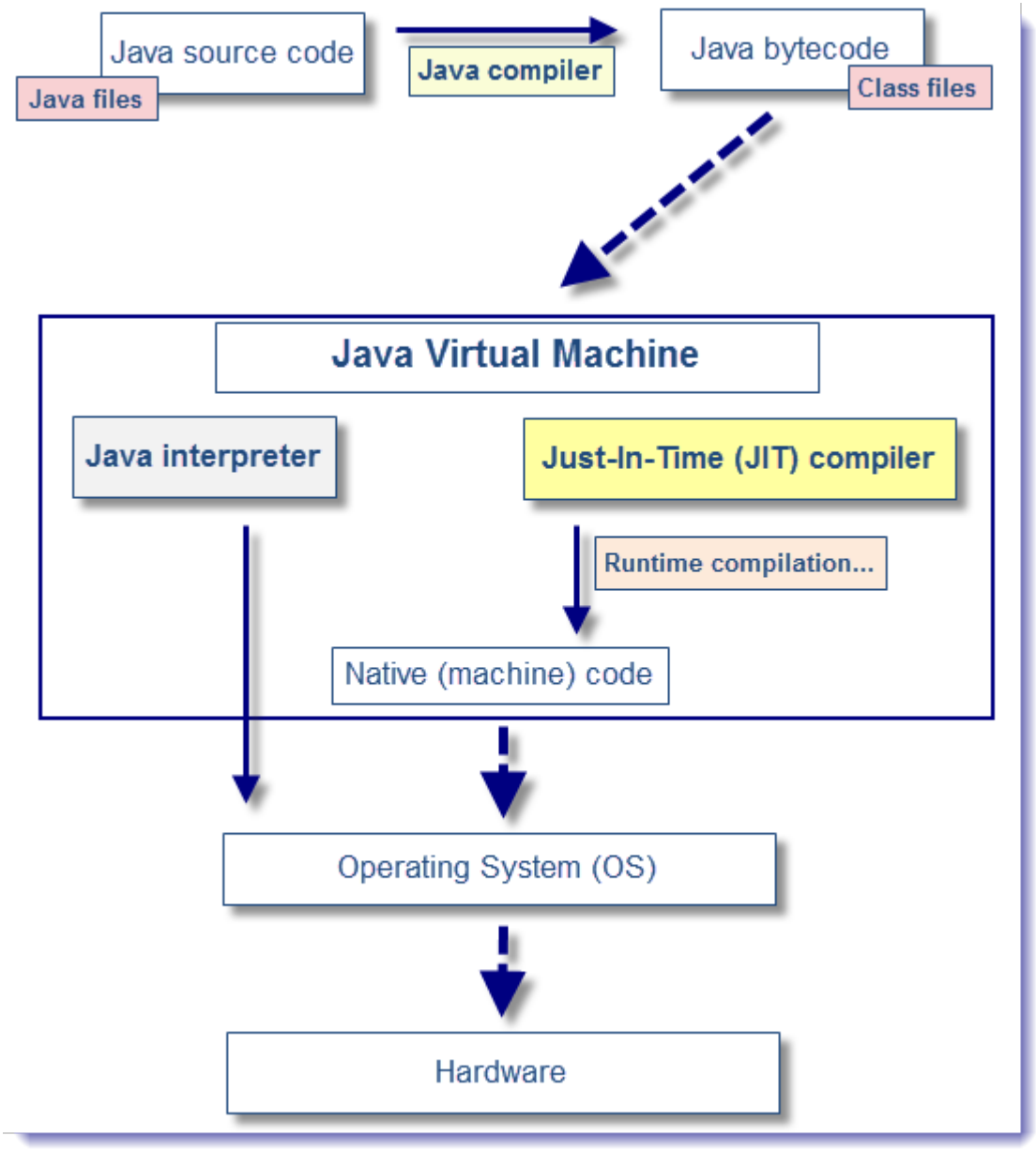
42: JIT (Just in Time)

Symantec JIT Sun Java 1.1.5 .

Hotspot JIT 1.2.2 Sun Java . Java 1.3 JIT .

(: JIT ?

Examples



Just-In-Time (JIT) Java Java ™ Runtime Environment .

- Java JVM .
- JVM .

Java .

JIT Java .

JIT Java . JIT "just in time" .

JVM . Java .

JIT . JVM . .

- . JVM call count .
- JVM JIT .
- JVM .
- JIT JVM .
- .
- 0 .
- JIT JIT .
- .

Java JIT .

JIT operational data at run time .

JIT , Java . JIT JIT .

JIT (Just in Time) : https://riptutorial.com/ko/java/topic/5152/jit--just-in-time--

43: JMX

JMX , . , , , .

Examples

MBean

. .

, .

```
public interface UserCounterMBean {
    long getSleepTime();

    void setSleepTime(long sleepTime);

    int getUserCount();

    void setUserCount(int userCount);

    String getGreetingString();

    void setGreetingString(String greetingString);

    void stop();
}
```

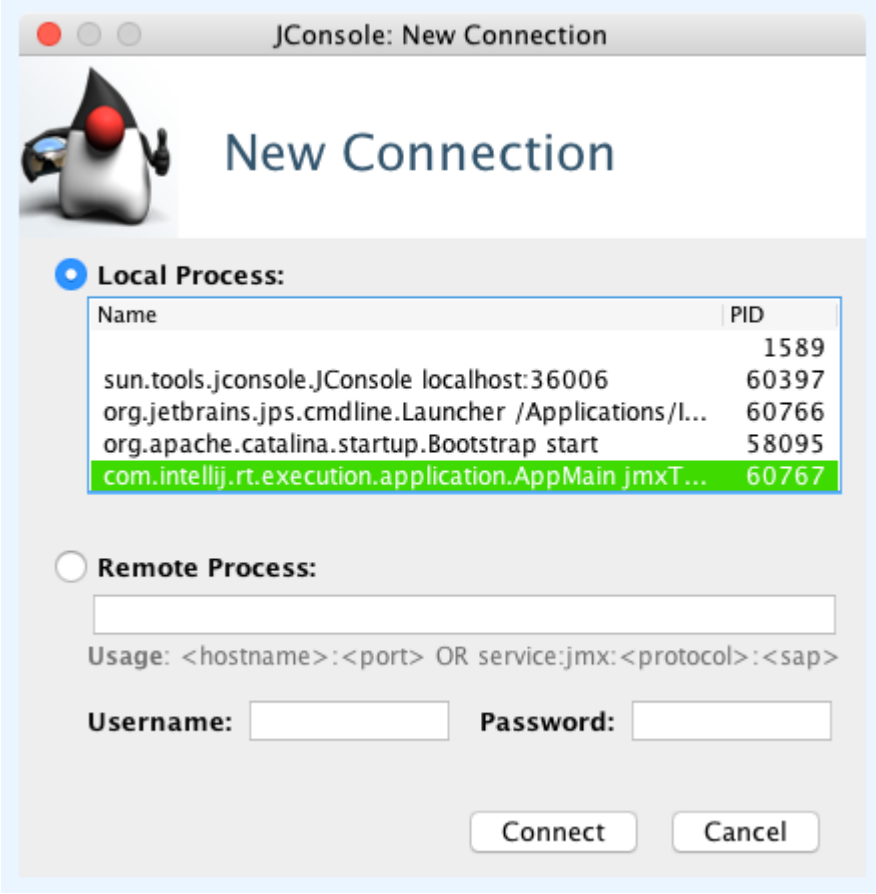
```
public class UserCounter implements UserCounterMBean, Runnable {
    private AtomicLong sleepTime = new AtomicLong(10000);
    private AtomicInteger userCount = new AtomicInteger(0);
    private AtomicReference<String> greetingString = new AtomicReference<>("welcome");
    private AtomicBoolean interrupted = new AtomicBoolean(false);

    @Override
    public long getSleepTime() {
        return sleepTime.get();
    }

    @Override
    public void setSleepTime(long sleepTime) {
        this.sleepTime.set(sleepTime);
    }

    @Override
    public int getUserCount() {
        return userCount.get();
    }

    @Override
    public void setUserCount(int userCount) {
        this.userCount.set(userCount);
    }
```

```
    @Override
    public String getGreetingString() {
        return greetingString.get();
    }

    @Override
    public void setGreetingString(String greetingString) {
        this.greetingString.set(greetingString);
    }

    @Override
    public void stop() {
        this.interrupted.set(true);
    }

    @Override
    public void run() {
        while (!interrupted.get()) {
            try {
                System.out.printf("User %d, %s%n", userCount.incrementAndGet(),
greetingString.get());
                Thread.sleep(sleepTime.get());
            } catch (InterruptedException ignored) {
            }
        }
    }
}
```

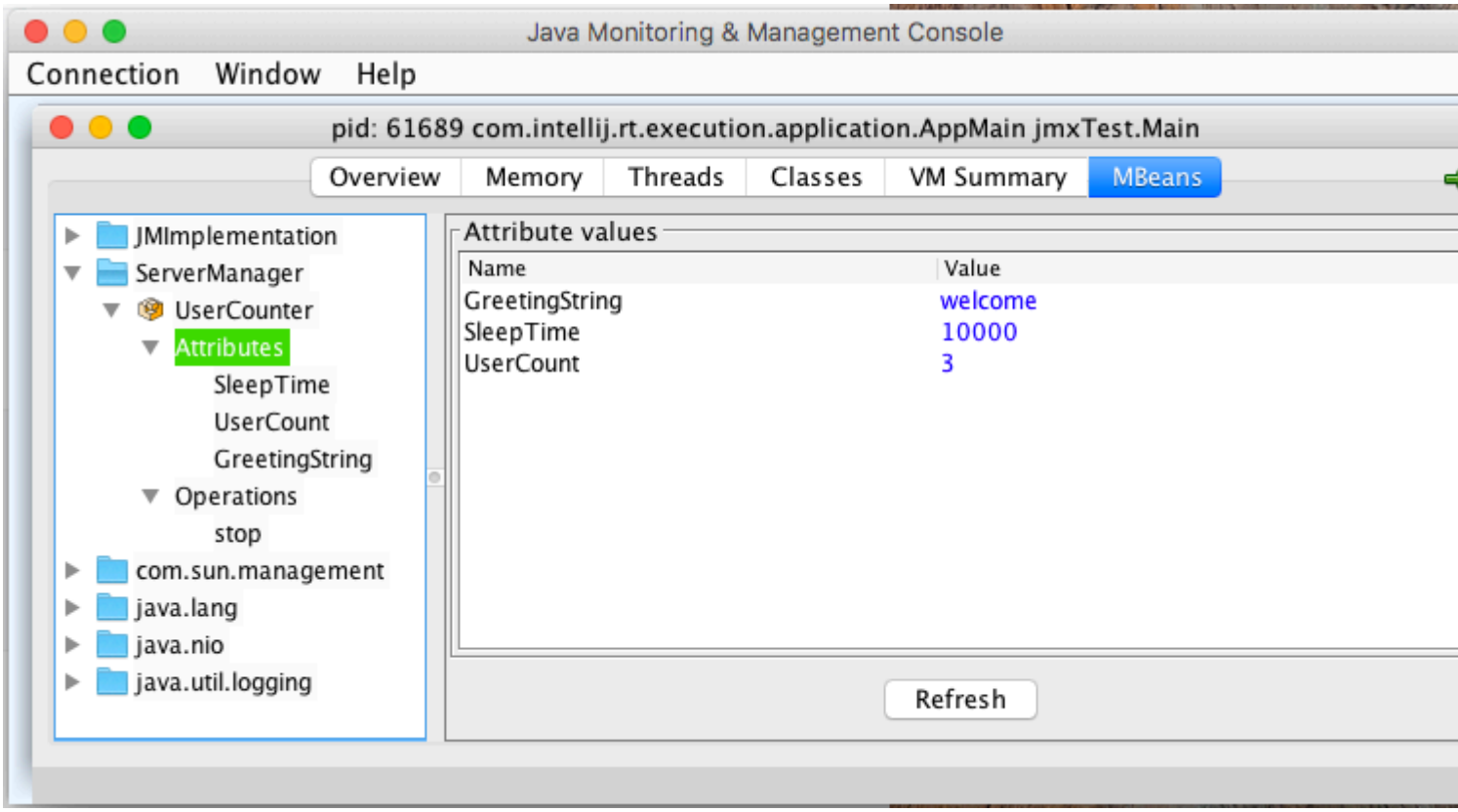
MBean .

```
import javax.management.InstanceAlreadyExistsException;
import javax.management.MBeanRegistrationException;
import javax.management.MBeanServer;
import javax.management.MalformedObjectNameException;
import javax.management.NotCompliantMBeanException;
import javax.management.ObjectName;
import java.lang.management.ManagementFactory;

public class Main {
    public static void main(String[] args) throws MalformedObjectNameException,
NotCompliantMBeanException, InstanceAlreadyExistsException, MBeanRegistrationException,
InterruptedException {
        final UserCounter userCounter = new UserCounter();
        final MBeanServer mBeanServer = ManagementFactory.getPlatformMBeanServer();
        final ObjectName objectName = new ObjectName("ServerManager:type=UserCounter");
        mBeanServer.registerMBean(userCounter, objectName);

        final Thread thread = new Thread(userCounter);
        thread.start();
        thread.join();
    }
}
```

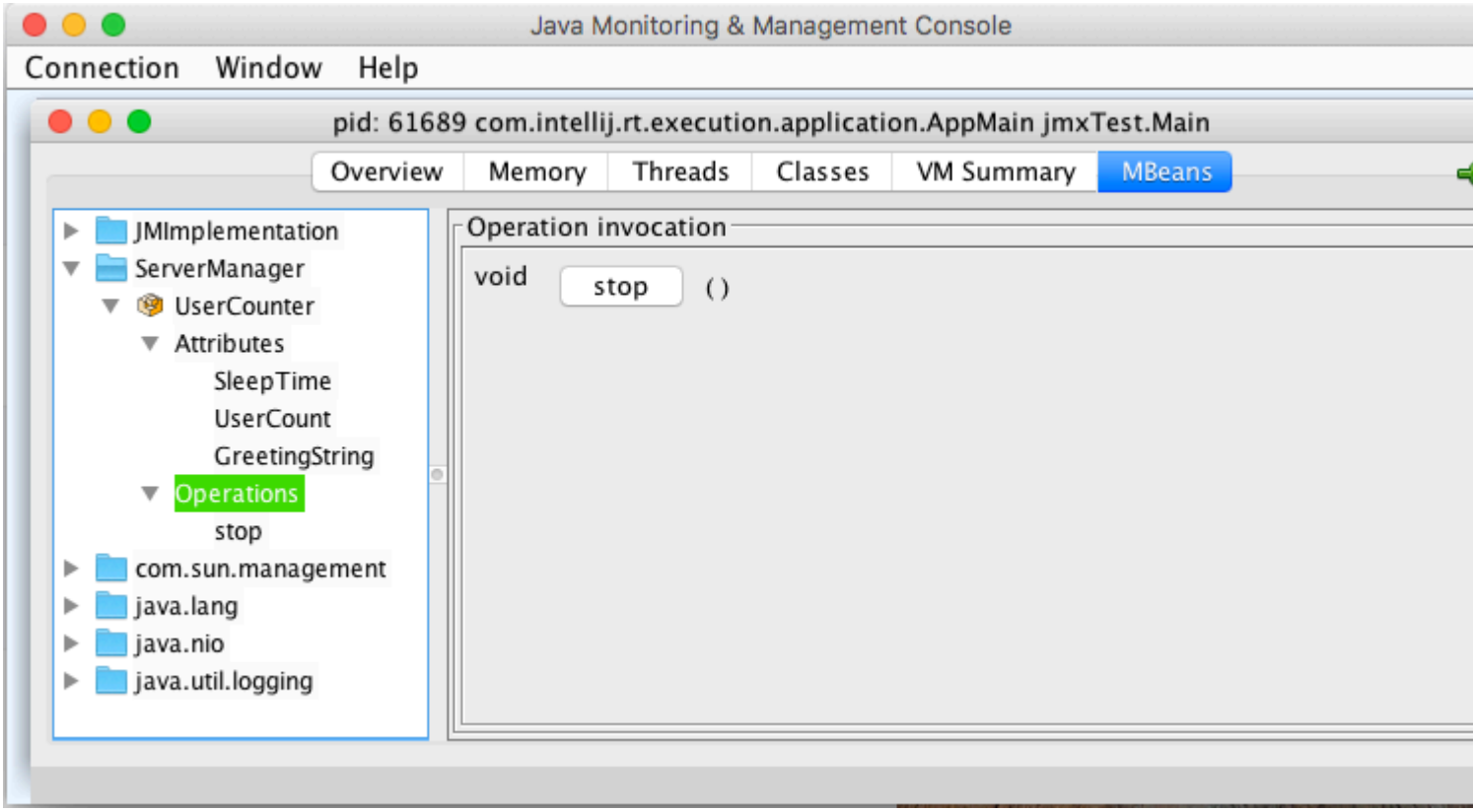
`$JAVA_HOME/bin` jConsole . , .



MBean Main MBean `ObjectName` (`ServerManager`). `Attributes` . get , . get set .



`Operations` .



JVM .

```
-Dcom.sun.management.jmxremote=true //true by default
-Dcom.sun.management.jmxremote.port=36006
-Dcom.sun.management.jmxremote.authenticate=false
-Dcom.sun.management.jmxremote.ssl=false
```

JMX 2 . jConsole `jconsole host:port` jConsole jConsole GUI `host:port` `service:jmx:rmi:///jndi/rmi://hostName:portNum/jmxrmi` .

:

- JMX
- JMX

JMX : https://riptutorial.com/ko/java/topic/9278/jmx

44: JNDI

Examples

JNDI RMI

RMI JNDI . :

- RMI Registry bind / unbind / rebind API
- RMI / API .

RMI JNDI RMI .

(), `java.rmi.registry.CreateRegistry()` RMI .

1. Server.java (JNDI)

```
package com.neohope.jndi.test;

import javax.naming.Context;
import javax.naming.InitialContext;
import javax.naming.NamingException;
import java.io.IOException;
import java.rmi.RemoteException;
import java.rmi.registry.LocateRegistry;
import java.util.Hashtable;

/**
 * JNDI Server
 * 1.create a registry on port 1234
 * 2.bind JNDI
 * 3.wait for connection
 * 4.clean up and end
 */
public class Server {
    private static Registry registry;
    private static InitialContext ctx;

    public static void initJNDI() {
        try {
            registry = LocateRegistry.createRegistry(1234);
            final Hashtable jndiProperties = new Hashtable();
            jndiProperties.put(Context.INITIAL_CONTEXT_FACTORY,
"com.sun.jndi.rmi.registry.RegistryContextFactory");
            jndiProperties.put(Context.PROVIDER_URL, "rmi://localhost:1234");
            ctx = new InitialContext(jndiProperties);
        } catch (NamingException e) {
            e.printStackTrace();
        } catch (RemoteException e) {
            e.printStackTrace();
        }
    }

    public static void bindJNDI(String name, Object obj) throws NamingException {
```

```
        ctx.bind(name, obj);
    }

    public static void unbindJNDI(String name) throws NamingException {
        ctx.unbind(name);
    }

    public static void unInitJNDI() throws NamingException {
        ctx.close();
    }

    public static void main(String[] args) throws NamingException, IOException {
        initJNDI();
        NMessage msg = new NMessage("Just A Message");
        bindJNDI("/neohope/jndi/test01", msg);
        System.in.read();
        unbindJNDI("/neohope/jndi/test01");
        unInitJNDI();
    }
}
```

2. Client.java (JNDI)

```
package com.neohope.jndi.test;

import javax.naming.Context;
import javax.naming.InitialContext;
import javax.naming.NamingException;
import java.util.Hashtable;

/**
 * 1.init context
 * 2.lookup registry for the service
 * 3.use the service
 * 4.end
 */
public class Client {
    public static void main(String[] args) throws NamingException {
        final Hashtable jndiProperties = new Hashtable();
        jndiProperties.put(Context.INITIAL_CONTEXT_FACTORY,
"com.sun.jndi.rmi.registry.RegistryContextFactory");
        jndiProperties.put(Context.PROVIDER_URL, "rmi://localhost:1234");

        InitialContext ctx = new InitialContext(jndiProperties);
        NMessage msg = (NeoMessage) ctx.lookup("/neohope/jndi/test01");
        System.out.println(msg.message);
        ctx.close();
    }
}
```

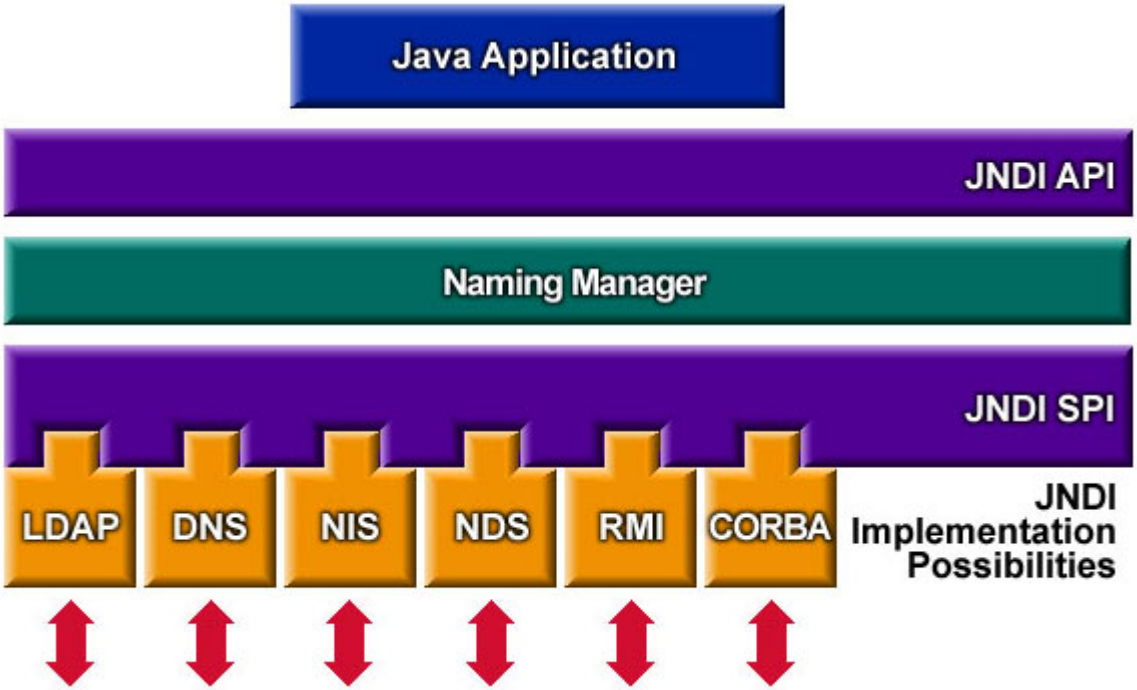
3. NMessage.java (RMI)

```
package com.neohope.jndi.test;

import java.io.Serializable;
import java.rmi.Remote;

/**
```

```
 * NMessage
 * RMI server class
 * must implements Remote and Serializable
 */
public class NMessage implements Remote, Serializable {
    public String message = "";

    public NMessage(String message)
    {
        this.message = message;
    }
}
```

eaxmple :

1.
2.



JNDI (Java Naming and Directory Interface) Java Java API. .

JNDI **API** (Application Programming Interface) **SPI** (Service Provider Interface) . Java API . SPI JNDI API Java .

JNDI LDAP, DNS, NIS, NDS, RMI CORBA . .

Java RMI JNDI API . .

-

RMI .

-

? , Name .

1. . `java.rmi.registry.LocateRegistry` .

```
//This will start a registry on localhost, port 1234
registry = LocateRegistry.createRegistry(1234);
```

2. . . .

```
//We use com.sun.jndi.rmi.registry.RegistryContextFactory as the InitialContextFactory
final Hashtable jndiProperties = new Hashtable();
jndiProperties.put(Context.INITIAL_CONTEXT_FACTORY,
"com.sun.jndi.rmi.registry.RegistryContextFactory");
//the registry usrl is "rmi://localhost:1234"
jndiProperties.put(Context.PROVIDER_URL, "rmi://localhost:1234");
InitialContext ctx = new InitialContext(jndiProperties);
```

3. .

```
//The jndi name is "/neohope/jndi/test01"
bindJNDI("/neohope/jndi/test01", msg);
```

4. "/ neohope / jndi / test01" .

```
//look up the object by name "java:com/neohope/jndi/test01"
NeoMessage msg = (NeoMessage) ctx.lookup("/neohope/jndi/test01");
```

5. .

6. .

```
ctx.unbind("/neohope/jndi/test01");
ctx.close();
```

JNDI : https://riptutorial.com/ko/java/topic/5720/jndi

45: JShell

JShell JDK 9 Java REPL. Java . jdk 9 . http://jdk.java.net/9/

- $ jshell - JShell REPL
- jshell> / <command> - JShell .
- jshell> / exit - JShell
- jshell> / help - JShell
- jshell> <java_expression> - Java ( ).
- jshell> / vars OR / methods OR / types - , .
- jshell> / open <file> - .
- jshell> / edit <identifier> -
- jshell> / set editor <command> - / edit
- jshell> / drop <> -
- jshell> / reset - JVM .

JShell (3 2017) (9) JDK jdk9.java.net . `jshell Unable to locate an executable JAVA_HOME` .

JShell .

```
import java.io.*
import java.math.*
import java.net.*
import java.nio.file.*
import java.util.*
import java.util.concurrent.*
import java.util.function.*
import java.util.prefs.*
import java.util.regex.*
import java.util.stream.*
```

Examples

JShell

JShell

JShell `JAVA_HOME` JDK 9 . JShell .

```
$ jshell
```

`jshell>` .

JShell

JShell JShell .

```
jshell> /exit
```

JShell Java . :

```
jshell> 4+2
jshell> System.out.printf("I am %d years old.\n", 421)
```

.

```
jshell> for (int i = 0; i<3; i++) {
   ...> System.out.println(i);
   ...> }
```

.

JShell .

```
jshell> String s = "hi"
jshell> int i = s.length
```

. JShell .

```
jshell> String var = "hi"
jshell> int var = 3
```

JShell `/vars` .

JShell .

```
jshell> void speak() {
   ...> System.out.println("hello");
   ...> }

jshell> class MyClass {
   ...> void doNothing() {}
   ...> }
```

. . . JShell `/methods /types` .

JShell . / / . `/edit` . , , `bar` `Foo` .

```
jshell> class Foo {
   ...> void bar() {
   ...> }
   ...> }
```

, . .

```
jshell> /edit Foo
```

. JShell .

```
jshell> /set editor emacs
jshell> /set editor vi
jshell> /set editor nano
jshell> /set editor -default
```

. / . .

```
jshell> String st = String 3
//error omitted
jshell> /edit st
|  No such snippet: st
```

(:) . .

```
jshell> int i = "hello"
//error omitted
jshell> /edit i
```

/drop .

```
jshell> int i = 13
jshell> /drop i
jshell> System.out.println(i)
|  Error:
|  cannot find symbol
|    symbol:   variable i
|  System.out.println(i)
|
```

JVM \reset \reset .

```
jshell> int i = 2

jshell> String s = "hi"

jshell> /reset
|  Resetting state.

jshell> i
|  Error:
|  cannot find symbol
|    symbol:   variable i
|  i
|  ^

jshell> s
|  Error:
|  cannot find symbol
|    symbol:   variable s
|  s
|  ^
```

JShell : https://riptutorial.com/ko/java/topic/9511/jshell

46: JVM

JVM TM

1.2

http://docs.oracle.com/javase/7/docs/platform/jvmti/jvmti.html

Examples

(Heap 1.0).

```
#include <vector>
#include <string>


#include "agent_util.hpp"
//this file can be found in Java SE Development Kit 8u101 Demos and Samples
//see http://download.oracle.com/otn-pub/java/jdk/8u101-b13-demos/jdk-8u101-windows-x64-
demos.zip
//jdk1.8.0_101.zip!\demo\jvmti\versionCheck\src\agent_util.h

/*
*   Struct used for jvmti->SetTag(object, <pointer to tag>);
*   http://docs.oracle.com/javase/7/docs/platform/jvmti/jvmti.html#SetTag
*
*/
typedef struct Tag
{
    jlong referrer_tag;
    jlong size;
    char* classSignature;
    jint hashCode;
} Tag;

/*
* Utility function: jlong -> Tag*
*/
static Tag* pointerToTag(jlong tag_ptr)
{
    if (tag_ptr == 0)
    {
        return new Tag();
    }
    return (Tag*)(ptrdiff_t)(void*)tag_ptr;
}

/*
* Utility function: Tag* -> jlong
*/
static jlong tagToPointer(Tag* tag)
{
    return (jlong)(ptrdiff_t)(void*)tag;
}
```

```
/*
*  Heap 1.0 Callback
*  http://docs.oracle.com/javase/7/docs/platform/jvmti/jvmti.html#jvmtiObjectReferenceCallback
*/
static jvmtiIterationControl JNICALL heabObjectReferencesCallback(
    jvmtiObjectReferenceKind reference_kind,
    jlong class_tag,
    jlong size,
    jlong* tag_ptr,
    jlong referrer_tag,
    jint referrer_index,
    void* user_data)
{
    //iterate only over reference field
    if (reference_kind != JVMTI_HEAP_REFERENCE_FIELD)
    {
        return JVMTI_ITERATION_IGNORE;
    }
    auto tag_ptr_list = (std::vector<jlong>*)(ptrdiff_t)(void*)user_data;
    //create and assign tag
    auto t = pointerToTag(*tag_ptr);
    t->referrer_tag = referrer_tag;
    t->size = size;
    *tag_ptr = tagToPointer(t);
    //collect tag
    (*tag_ptr_list).push_back(*tag_ptr);

    return JVMTI_ITERATION_CONTINUE;
}

/*
*  Main function for demonstration of Iterate Over Objects Reachable From Object
*
http://docs.oracle.com/javase/7/docs/platform/jvmti/jvmti.html#IterateOverObjectsReachableFromObject

*
*/
void iterateOverObjectHeapReferences(jvmtiEnv* jvmti, JNIEnv* env, jobject object)
{
    std::vector<jlong> tag_ptr_list;

    auto t = new Tag();
    jvmti->SetTag(object, tagToPointer(t));
    tag_ptr_list.push_back(tagToPointer(t));

    stdout_message("tag list size before call callback:  %d\n", tag_ptr_list.size());
    /*
    *    Call Callback for every reachable object reference
    *    see
http://docs.oracle.com/javase/7/docs/platform/jvmti/jvmti.html#IterateOverObjectsReachableFromObject

    */
    jvmti->IterateOverObjectsReachableFromObject(object, &heabObjectReferencesCallback,
(void*)&tag_ptr_list);
    stdout_message("tag list size after call callback:  %d\n", tag_ptr_list.size());

    if (tag_ptr_list.size() > 0)
    {
        jint found_count = 0;
        jlong* tags = &tag_ptr_list[0];
        jobject* found_objects;
```

```
        jlong* found_tags;

        /*
        *    collect all tagged object (via *tag_ptr = pointer to tag )
        *    see
http://docs.oracle.com/javase/7/docs/platform/jvmti/jvmti.html#GetObjectsWithTags
        */
        jvmti->GetObjectsWithTags(tag_ptr_list.size(), tags, &found_count, &found_objects,
&found_tags);
        stdout_message("found %d objects\n", found_count);

        for (auto i = 0; i < found_count; ++i)
        {
            jobject found_object = found_objects[i];

            char* classSignature;
            jclass found_object_class = env->GetObjectClass(found_object);
            /*
            *    Get string representation of found_object_class
            *    see
http://docs.oracle.com/javase/7/docs/platform/jvmti/jvmti.html#GetClassSignature
            */
            jvmti->GetClassSignature(found_object_class, &classSignature, nullptr);

            jint hashCode;
            /*
            *    Getting hash code for found_object
            *    see
http://docs.oracle.com/javase/7/docs/platform/jvmti/jvmti.html#GetObjectHashCode
            */
            jvmti->GetObjectHashCode(found_object, &hashCode);

            //save all it in Tag
            Tag* t = pointerToTag(found_tags[i]);
            t->classSignature = classSignature;
            t->hashCode = hashCode;
        }

        //print all saved information
        for (auto i = 0; i < found_count; ++i)
        {
            auto t = pointerToTag(found_tags[i]);
            auto rt = pointerToTag(t->referrer_tag);

            if (t->referrer_tag != 0)
            {
                stdout_message("referrer object %s#%d --> object %s#%d (size: %2d)\n",
                    rt->classSignature, rt->hashCode, t->classSignature, t->hashCode, t-
>size);
            }
        }
    }
}
```

JVM TI

Agent_OnLoad :

```
jvmtiEnv* jvmti;
```

```
/* Get JVMTI environment */
vm->GetEnv(reinterpret_cast<void **>(&jvmti), JVMTI_VERSION);
```

Agent_OnLoad

```
/* Callback for JVMTI_EVENT_VM_INIT */
static void JNICALL vm_init(jvmtiEnv* jvmti, JNIEnv* env, jthread thread)
{
    jint runtime_version;
    jvmti->GetVersionNumber(&runtime_version);
    stdout_message("JVMTI Version: %d\n", runtime_verision);
}

/* Agent_OnLoad() is called first, we prepare for a VM_INIT event here. */
JNIEXPORT jint JNICALL
Agent_OnLoad(JavaVM* vm, char* options, void* reserved)
{
    jint rc;
    jvmtiEventCallbacks callbacks;
    jvmtiCapabilities capabilities;
    jvmtiEnv* jvmti;

    /* Get JVMTI environment */
    rc = vm->GetEnv(reinterpret_cast<void **>(&jvmti), JVMTI_VERSION);
    if (rc != JNI_OK)
    {
        return -1;
    }

    /* Immediately after getting the jvmtiEnv* we need to ask for the
    *  capabilities this agent will need.
    */
    jvmti->GetCapabilities(&capabilities);
    capabilities.can_tag_objects = 1;
    jvmti->AddCapabilities(&capabilities);

    /* Set callbacks and enable event notifications */
    memset(&callbacks, 0, sizeof(callbacks));
    callbacks.VMInit = &vm_init;

    jvmti->SetEventCallbacks(&callbacks, sizeof(callbacks));
    jvmti->SetEventNotificationMode(JVMTI_ENABLE, JVMTI_EVENT_VM_INIT, nullptr);

    return JNI_OK;
}
```

JVM : https://riptutorial.com/ko/java/topic/3316/jvm--

47: JVM

.

- .
- .

.

Examples

-XXaggressive

`-XXaggressive` JVM . JVM . . , .

:

```
-XXaggressive:<param>
```

<param>	
`opt`	. .
`memory`	. JRockit JVM .

-XXallocClearChunks

TLA TLA . , , 0 ( ). . TLA .

:

```
-XXallocClearChunks
```

```
-XXallocClearChunks=<true | false>
```

IA64 . . `-XXallocClearChunkSize` . `true` .

-XXallocClearChunkSize

`-XXallocClearChunkSize` `-XXallocClearChunkSize` . 512 .

:

```
-XXallocClearChunks -XXallocClearChunkSize=<size>[k|K][m|M][g|G]
```

-XXcallProfiling

. JVM .

: JRockit JVM R27.3.0 . .

:

```
java -XXcallProfiling myApp
```

. .

-XXdisableFatSpin

Java fat lock fat lock sleep .

Java . (,). " " . "" . . `-XXdisableFatSpin` .

:

```
-XXdisableFatSpin
```

-XXdisableGCHeuristics

. . .

:

```
-XXdisableFatSpin
```

-XXdumpSize

(, ,) .

:

```
-XXdumpsize:<size>
```

<>	
`none`	.
`small`	Windows (Linux). . JRockit JVM 8.1 1 2 3 7.0 .
`normal`	. Java . JRockit JVM 1.4.2 .
`large`	Java . `-XXdumpSize` `-XXdumpFullState` .

-XXexitOnOutOfMemory

JRockit JVM . JRockit JVM . JRockit JVM .

:

```
-XXexitOnOutOfMemory
```

JVM : https://riptutorial.com/ko/java/topic/2500/jvm-

48: log4j / log4j2

Apache Log4j Java Java . Log4j Java .

- Logger.debug ("text to log"); //
- Logger.info ("text to log"); //
- Logger.error ("text to log"); //
- Logger.warn (" "); //
- Logger.trace ("text to log"); //
- Logger.fatal ("text to log"); //
- Log4j2 :
- Logger.debug ("Debug params {} {} {,"param1, param2, param3); //
- Logger.info ("Info params {} {} {}", param1, param2, param3); //
- Logger.error (" params {} {} {}", param1, param2, param3); //
- Logger.warn (" params {} {} {,"param1, param2, param3); //
- Logger.trace ("Trace params {} {} {}", param1, param2, param3); //
- Logger.fatal (" params {} {} {}", param1, param2, param3); //
- Logger.error (" :",); // ( )

Log4j 1

2015 8 5 (Logging Services Project Management Committee) Log4j 1.x .
Apache . **Log4j 1 Apache Log4j 2** .

: http://logging.apache.org/log4j/1.2/

Examples

Log4j

(log4j2)

Maven :

`POM.xml` .

```
<dependencies>
  <dependency>
    <groupId>org.apache.logging.log4j</groupId>
    <artifactId>log4j-api</artifactId>
    <version>2.6.2</version>
  </dependency>
  <dependency>
    <groupId>org.apache.logging.log4j</groupId>
    <artifactId>log4j-core</artifactId>
    <version>2.6.2</version>
  </dependency>
</dependencies>
```

Ivy :

```
<dependencies>
  <dependency org="org.apache.logging.log4j" name="log4j-api" rev="2.6.2" />
  <dependency org="org.apache.logging.log4j" name="log4j-core" rev="2.6.2" />
</dependencies>
```

Gradle :

```
dependencies {
  compile group: 'org.apache.logging.log4j', name: 'log4j-api', version: '2.6.2'
  compile group: 'org.apache.logging.log4j', name: 'log4j-core', version: '2.6.2'
}
```

---

log4j 1.x

: Log4j 1.x EOL (End-of-Life) ( ).

Maven :

`POM.xml` .

```
<dependency>
    <groupId>log4j</groupId>
    <artifactId>log4j</artifactId>
    <version>1.2.17</version>
</dependency>
```

Ivy :

```
<dependency org="log4j" name="log4j" rev="1.2.17"/>
```

Gradle :

```
compile group: 'log4j', name: 'log4j', version: '1.2.17'
```

Buildr :

```
'log4j:log4j:jar:1.2.17'
```

:

Log4j

Log4j Java

`final static logger .`

```
final static Logger logger = Logger.getLogger(classname.class);
```

.

```
//logs an error message
logger.info("Information about some param: " + parameter); // Note that this line could throw
a NullPointerException!

//in order to improve performance, it is advised to use the `isXXXEnabled()` Methods
if( logger.isInfoEnabled() ){
    logger.info("Information about some param: " + parameter);
}

// In log4j2 parameter substitution is preferable due to readability and performance
// The parameter substitution only takes place if info level is active which obsoletes the use
of isXXXEnabled().
logger.info("Information about some param: {}" , parameter);

//logs an exception
logger.error("Information about some error: ", exception);
```

Log4j . `log4j.properties` `log4j.properties` .

```
# Root logger option
log4j.rootLogger=DEBUG, stdout, file

# Redirect log messages to console
log4j.appender.stdout=org.apache.log4j.ConsoleAppender
log4j.appender.stdout.Target=System.out
log4j.appender.stdout.layout=org.apache.log4j.PatternLayout
log4j.appender.stdout.layout.ConversionPattern=%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n

# Redirect log messages to a log file, support file rolling.
log4j.appender.file=org.apache.log4j.RollingFileAppender
log4j.appender.file.File=C:\\log4j-application.log
log4j.appender.file.MaxFileSize=5MB
log4j.appender.file.MaxBackupIndex=10
log4j.appender.file.layout=org.apache.log4j.PatternLayout
log4j.appender.file.layout.ConversionPattern=%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n
```

Maven , :

```
/ProjectFolder/src/java/resources
```

log4j2.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<Configuration>
  <Appenders>
    <Console name="STDOUT" target="SYSTEM_OUT">
      <PatternLayout pattern="%d %-5p [%t] %C{2} %m%n"/>
    </Console>
  </Appenders>
```

```
  <Loggers>
    <Root level="debug">
      <AppenderRef ref="STDOUT"/>
    </Root>
  </Loggers>
</Configuration>
```

appender root logger log4j2.xml . .
log4j2.xml `status = <WARN | DEBUG | ERROR | FATAL | TRACE | INFO>` log4j2.xml .
. . `monitorInterval = 30` . 30 .

log4j 1.x 2.x

log4j 1.x log4j 2.x log4j 1.x .

Log4j 1.x API Bridge

```
<dependencies>
  <dependency>
    <groupId>org.apache.logging.log4j</groupId>
    <artifactId>log4j-1.2-api</artifactId>
    <version>2.6.2</version>
  </dependency>
</dependencies>
```

```
<dependencies>
  <dependency org="org.apache.logging.log4j" name="log4j-1.2-api" rev="2.6.2" />
</dependencies>
```

Gradle Build

```
dependencies {
  compile group: 'org.apache.logging.log4j', name: 'log4j-1.2-api', version: '2.6.2'
}
```

Apache Commons Logging Bridge log4j 1.x Apache Commons Logging log4j 2.x .

```
<dependencies>
  <dependency>
    <groupId>org.apache.logging.log4j</groupId>
    <artifactId>log4j-jcl</artifactId>
    <version>2.6.2</version>
  </dependency>
</dependencies>
```

```
<dependencies>
  <dependency org="org.apache.logging.log4j" name="log4j-jcl" rev="2.6.2" />
</dependencies>
```

Gradle Build

```
dependencies {
```

```
  compile group: 'org.apache.logging.log4j', name: 'log4j-jcl', version: '2.6.2'
}
```

: Apache .

: https://logging.apache.org/log4j/2.x/maven-artifacts.html

- DB

JDBC . IBM System i DB2 opensource . JT400

DB2 JDBC URL .

```
# Root logger option
log4j.rootLogger= ERROR, DB

# Redirect log messages to a DB2
# Define the DB appender
log4j.appender.DB=org.apache.log4j.jdbc.JDBCAppender

# Set JDBC URL (!!! adapt to your target system !!!)
log4j.appender.DB.URL=jdbc:as400://10.10.10.1:446/DATABASENAME;naming=system;errors=full;

# Set Database Driver (!!! adapt to your target system !!!)
log4j.appender.DB.driver=com.ibm.as400.access.AS400JDBCDriver

# Set database user name and password
log4j.appender.DB.user=USER
log4j.appender.DB.password=PASSWORD

# Set the SQL statement to be executed.
log4j.appender.DB.sql=INSERT INTO DB.TABLENAME VALUES('%d{yyyy-MM-
dd}','%d{HH:mm:ss}','%C','%p','%m')

# Define the layout for file appender
log4j.appender.DB.layout=org.apache.log4j.PatternLayout
```

(log4j 1.x)

`ERROR` "" . **Filter PropertyConfigurator . XML** . log4j-Wiki .

" "

```
<appender name="info-out" class="org.apache.log4j.FileAppender">
            <param name="File" value="info.log"/>
            <layout class="org.apache.log4j.PatternLayout">
                    <param name="ConversionPattern" value="%m%n"/>
            </layout>
            <filter class="org.apache.log4j.varia.LevelMatchFilter">
                    <param name="LevelToMatch" value="info" />
                    <param name="AcceptOnMatch" value="true"/>
            </filter>
            <filter class="org.apache.log4j.varia.DenyAllFilter" />
</appender>
```

" "

```
<appender name="info-out" class="org.apache.log4j.FileAppender">
            <param name="File" value="info.log"/>
            <layout class="org.apache.log4j.PatternLayout">
                    <param name="ConversionPattern" value="%m%n"/>
            </layout>
            <filter class="org.apache.log4j.varia.LevelRangeFilter">
                    <param name="LevelMax" value="info"/>
                    <param name="LevelMin" value="info"/>
                    <param name="AcceptOnMatch" value="true"/>
            </filter>
</appender>
```

log4j / log4j2 : https://riptutorial.com/ko/java/topic/2472/log4j---log4j2

49: MultiThreaded ThreadPoolExecutor .

Performant . ThreadPoolExecutors .

Examples

"Fire & Forget" (: , ,).

, Runnable main () 2 .

AsyncMaintenanceTaskCompleter.java

```
import lombok.extern.java.Log;

import java.util.concurrent.ThreadLocalRandom;
import java.util.concurrent.TimeUnit;

@Log
public class AsyncMaintenanceTaskCompleter implements Runnable {
    private int taskNumber;

    public AsyncMaintenanceTaskCompleter(int taskNumber) {
        this.taskNumber = taskNumber;
    }

    public void run() {
        int timeout = ThreadLocalRandom.current().nextInt(1, 20);
        try {
            log.info(String.format("Task %d is sleeping for %d seconds", taskNumber,
timeout));
            TimeUnit.SECONDS.sleep(timeout);
            log.info(String.format("Task %d is done sleeping", taskNumber));

        } catch (InterruptedException e) {
            log.warning(e.getMessage());
        }
    }
}
```

AsyncExample1

```
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;

public class AsyncExample1 {
    public static void main(String[] args){
        ExecutorService executorService = Executors.newCachedThreadPool();
        for(int i = 0; i < 10; i++){
            executorService.execute(new AsyncMaintenanceTaskCompleter(i));
        }
        executorService.shutdown();
    }
}
```

AsyncExample1.main () .

```
Dec 28, 2016 2:21:03 PM AsyncMaintenanceTaskCompleter run
INFO: Task 8 is sleeping for 18 seconds
Dec 28, 2016 2:21:03 PM AsyncMaintenanceTaskCompleter run
INFO: Task 6 is sleeping for 4 seconds
Dec 28, 2016 2:21:03 PM AsyncMaintenanceTaskCompleter run
INFO: Task 2 is sleeping for 6 seconds
Dec 28, 2016 2:21:03 PM AsyncMaintenanceTaskCompleter run
INFO: Task 3 is sleeping for 4 seconds
Dec 28, 2016 2:21:03 PM AsyncMaintenanceTaskCompleter run
INFO: Task 9 is sleeping for 14 seconds
Dec 28, 2016 2:21:03 PM AsyncMaintenanceTaskCompleter run
INFO: Task 4 is sleeping for 9 seconds
Dec 28, 2016 2:21:03 PM AsyncMaintenanceTaskCompleter run
INFO: Task 5 is sleeping for 10 seconds
Dec 28, 2016 2:21:03 PM AsyncMaintenanceTaskCompleter run
INFO: Task 0 is sleeping for 7 seconds
Dec 28, 2016 2:21:03 PM AsyncMaintenanceTaskCompleter run
INFO: Task 1 is sleeping for 9 seconds
Dec 28, 2016 2:21:03 PM AsyncMaintenanceTaskCompleter run
INFO: Task 7 is sleeping for 8 seconds
Dec 28, 2016 2:21:07 PM AsyncMaintenanceTaskCompleter run
INFO: Task 6 is done sleeping
Dec 28, 2016 2:21:07 PM AsyncMaintenanceTaskCompleter run
INFO: Task 3 is done sleeping
Dec 28, 2016 2:21:09 PM AsyncMaintenanceTaskCompleter run
INFO: Task 2 is done sleeping
Dec 28, 2016 2:21:10 PM AsyncMaintenanceTaskCompleter run
INFO: Task 0 is done sleeping
Dec 28, 2016 2:21:11 PM AsyncMaintenanceTaskCompleter run
INFO: Task 7 is done sleeping
Dec 28, 2016 2:21:12 PM AsyncMaintenanceTaskCompleter run
INFO: Task 4 is done sleeping
Dec 28, 2016 2:21:12 PM AsyncMaintenanceTaskCompleter run
INFO: Task 1 is done sleeping
Dec 28, 2016 2:21:13 PM AsyncMaintenanceTaskCompleter run
INFO: Task 5 is done sleeping
Dec 28, 2016 2:21:17 PM AsyncMaintenanceTaskCompleter run
INFO: Task 9 is done sleeping
Dec 28, 2016 2:21:21 PM AsyncMaintenanceTaskCompleter run
INFO: Task 8 is done sleeping

Process finished with exit code 0
```

.

1. .
2. () .

.

Callable <T> (T) main () .

AsyncValueTypeTaskCompleter.java

```
import lombok.extern.java.Log;
```

```
import java.util.concurrent.Callable;
import java.util.concurrent.ThreadLocalRandom;
import java.util.concurrent.TimeUnit;

@Log
public class AsyncValueTypeTaskCompleter implements Callable<Integer> {
    private int taskNumber;

    public AsyncValueTypeTaskCompleter(int taskNumber) {
        this.taskNumber = taskNumber;
    }

    @Override
    public Integer call() throws Exception {
        int timeout = ThreadLocalRandom.current().nextInt(1, 20);
        try {
            log.info(String.format("Task %d is sleeping", taskNumber));
            TimeUnit.SECONDS.sleep(timeout);
            log.info(String.format("Task %d is done sleeping", taskNumber));

        } catch (InterruptedException e) {
            log.warning(e.getMessage());
        }
        return timeout;
    }
}
```

AsyncExample2.java

```
import lombok.extern.java.Log;

import java.util.ArrayList;
import java.util.List;
import java.util.concurrent.ExecutionException;
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;
import java.util.concurrent.Future;

@Log
public class AsyncExample2 {
    public static void main(String[] args) {
        ExecutorService executorService = Executors.newCachedThreadPool();
        List<Future<Integer>> futures = new ArrayList<>();
        for (int i = 0; i < 10; i++){
            Future<Integer> submittedFuture = executorService.submit(new
AsyncValueTypeTaskCompleter(i));
            futures.add(submittedFuture);
        }
        executorService.shutdown();
        while(!futures.isEmpty()){
            for(int j = 0; j < futures.size(); j++){
                Future<Integer> f = futures.get(j);
                if(f.isDone()){
                    try {
                        int timeout = f.get();
                        log.info(String.format("A task just completed after sleeping for %d
seconds", timeout));
                        futures.remove(f);
                    } catch (InterruptedException | ExecutionException e) {
                        log.warning(e.getMessage());
```

```
                    }
                }
            }
        }
    }
}
```

AsyncExample2.main () .

```
Dec 28, 2016 3:07:15 PM AsyncValueTypeTaskCompleter call
INFO: Task 7 is sleeping
Dec 28, 2016 3:07:15 PM AsyncValueTypeTaskCompleter call
INFO: Task 8 is sleeping
Dec 28, 2016 3:07:15 PM AsyncValueTypeTaskCompleter call
INFO: Task 2 is sleeping
Dec 28, 2016 3:07:15 PM AsyncValueTypeTaskCompleter call
INFO: Task 1 is sleeping
Dec 28, 2016 3:07:15 PM AsyncValueTypeTaskCompleter call
INFO: Task 4 is sleeping
Dec 28, 2016 3:07:15 PM AsyncValueTypeTaskCompleter call
INFO: Task 9 is sleeping
Dec 28, 2016 3:07:15 PM AsyncValueTypeTaskCompleter call
INFO: Task 0 is sleeping
Dec 28, 2016 3:07:15 PM AsyncValueTypeTaskCompleter call
INFO: Task 6 is sleeping
Dec 28, 2016 3:07:15 PM AsyncValueTypeTaskCompleter call
INFO: Task 5 is sleeping
Dec 28, 2016 3:07:15 PM AsyncValueTypeTaskCompleter call
INFO: Task 3 is sleeping
Dec 28, 2016 3:07:16 PM AsyncValueTypeTaskCompleter call
INFO: Task 8 is done sleeping
Dec 28, 2016 3:07:16 PM AsyncExample2 main
INFO: A task just completed after sleeping for 1 seconds
Dec 28, 2016 3:07:17 PM AsyncValueTypeTaskCompleter call
INFO: Task 2 is done sleeping
Dec 28, 2016 3:07:17 PM AsyncExample2 main
INFO: A task just completed after sleeping for 2 seconds
Dec 28, 2016 3:07:17 PM AsyncValueTypeTaskCompleter call
INFO: Task 9 is done sleeping
Dec 28, 2016 3:07:17 PM AsyncExample2 main
INFO: A task just completed after sleeping for 2 seconds
Dec 28, 2016 3:07:19 PM AsyncValueTypeTaskCompleter call
INFO: Task 3 is done sleeping
Dec 28, 2016 3:07:19 PM AsyncExample2 main
INFO: A task just completed after sleeping for 4 seconds
Dec 28, 2016 3:07:20 PM AsyncValueTypeTaskCompleter call
INFO: Task 0 is done sleeping
Dec 28, 2016 3:07:20 PM AsyncExample2 main
INFO: A task just completed after sleeping for 5 seconds
Dec 28, 2016 3:07:21 PM AsyncValueTypeTaskCompleter call
INFO: Task 5 is done sleeping
Dec 28, 2016 3:07:21 PM AsyncExample2 main
INFO: A task just completed after sleeping for 6 seconds
Dec 28, 2016 3:07:25 PM AsyncValueTypeTaskCompleter call
INFO: Task 1 is done sleeping
Dec 28, 2016 3:07:25 PM AsyncExample2 main
INFO: A task just completed after sleeping for 10 seconds
Dec 28, 2016 3:07:27 PM AsyncValueTypeTaskCompleter call
INFO: Task 6 is done sleeping
Dec 28, 2016 3:07:27 PM AsyncExample2 main
```

```
INFO: A task just completed after sleeping for 12 seconds
Dec 28, 2016 3:07:29 PM AsyncValueTypeTaskCompleter call
INFO: Task 7 is done sleeping
Dec 28, 2016 3:07:29 PM AsyncExample2 main
INFO: A task just completed after sleeping for 14 seconds
Dec 28, 2016 3:07:31 PM AsyncValueTypeTaskCompleter call
INFO: Task 4 is done sleeping
Dec 28, 2016 3:07:31 PM AsyncExample2 main
INFO: A task just completed after sleeping for 16 seconds
```

:

.

1. ExecutorService.submit () Future .
2. Future isDone () . Future.get () .
3. Future executorService.shutdown () . . executorService.shutdown () ExecutorService Queue .

Lambdas

, .

main () . Callable Runnable <T> .

AsyncExample3.java

```
import lombok.extern.java.Log;

import java.util.ArrayList;
import java.util.List;
import java.util.concurrent.*;

@Log
public class AsyncExample3 {
    public static void main(String[] args) {
        ExecutorService executorService = Executors.newCachedThreadPool();
        List<Future<Integer>> futures = new ArrayList<>();
        for(int i = 0; i < 5; i++){
            final int index = i;
            executorService.execute(() -> {
                int timeout = getTimeout();
                log.info(String.format("Runnable %d has been submitted and will sleep for %d
seconds", index, timeout));
                try {
                    TimeUnit.SECONDS.sleep(timeout);
                } catch (InterruptedException e) {
                    log.warning(e.getMessage());
                }
                log.info(String.format("Runnable %d has finished sleeping", index));
            });
            Future<Integer> submittedFuture = executorService.submit(() -> {
                int timeout = getTimeout();
                log.info(String.format("Callable %d will begin sleeping", index));
                try {
                    TimeUnit.SECONDS.sleep(timeout);
```

```
                } catch (InterruptedException e) {
                    log.warning(e.getMessage());
                }
                log.info(String.format("Callable %d is done sleeping", index));
                return timeout;
            });
            futures.add(submittedFuture);
        }
        executorService.shutdown();
        while(!futures.isEmpty()){
            for(int j = 0; j < futures.size(); j++){
                Future<Integer> f = futures.get(j);
                if(f.isDone()){
                    try {
                        int timeout = f.get();
                        log.info(String.format("A task just completed after sleeping for %d
seconds", timeout));
                        futures.remove(f);
                    } catch (InterruptedException | ExecutionException e) {
                        log.warning(e.getMessage());
                    }
                }
            }
        }
    }

    public static int getTimeout(){
        return ThreadLocalRandom.current().nextInt(1, 20);
    }
}
```

:

.

1. ( ) .
2. Callable Runnable <T> .

MultiThreaded ThreadPoolExecutor . : https://riptutorial.com/ko/java/topic/8646/multithreaded---threadpoolexecutor--

50: Nashorn

Nashorn Java Nashorn JSR-223 8 , ECMA .

- ScriptEngineManager // ScriptEngine . SPI ( )
- ScriptEngineManager.ScriptEngineManager () //
- ScriptEngine // .
- ScriptEngine ScriptEngineManager. getEngineByName (String shortName) // .
- Object ScriptEngine.eval (String script) // .
- Object ScriptEngine.eval (Reader reader) // .
- ScriptContext ScriptEngine.getContext () // , .
- void ScriptContext. setWriter (Writer writer) // .

Nashorn Java Java 8 JavaScript . `javax.script` .

`ScriptEngineManager` (: JavaScript Nashorn) API .

Examples

```
// Obtain an instance of JavaScript engine
ScriptEngineManager manager = new ScriptEngineManager();
ScriptEngine engine = manager.getEngineByName("nashorn");

// Define a global variable
engine.put("textToPrint", "Data defined in Java.");

// Print the global variable
try {
    engine.eval("print(textToPrint);");
} catch (ScriptException ex) {
    ex.printStackTrace();
}

// Outcome:
// 'Data defined in Java.' printed on standard output
```

Nashorn

```
// Obtain an instance of JavaScript engine
ScriptEngineManager manager = new ScriptEngineManager();
ScriptEngine engine = manager.getEngineByName("nashorn");

// Execute an hardcoded script
try {
    engine.eval("print('Hello Nashorn!');");
} catch (ScriptException ex) {
    // This is the generic Exception subclass for the Scripting API
    ex.printStackTrace();
}

// Outcome:
```

```
// 'Hello Nashorn!' printed on standard output
```

JavaScript

```
// Required imports
import javax.script.ScriptEngineManager;
import javax.script.ScriptEngine;
import javax.script.ScriptException;
import java.io.FileReader;
import java.io.FileNotFoundException;

// Obtain an instance of the JavaScript engine
ScriptEngineManager manager = new ScriptEngineManager();
ScriptEngine engine = manager.getEngineByName("nashorn");

// Load and execute a script from the file 'demo.js'
try {
    engine.eval(new FileReader("demo.js"));
} catch (FileNotFoundException ex) {
    ex.printStackTrace();
} catch (ScriptException ex) {
    // This is the generic Exception subclass for the Scripting API
    ex.printStackTrace();
}

// Outcome:
// 'Script from file!' printed on standard output
```

demo.js :

```
print('Script from file!');
```

```
// Obtain an instance of JavaScript engine
ScriptEngineManager manager = new ScriptEngineManager();
ScriptEngine engine = manager.getEngineByName("nashorn");

// Setup a custom writer
StringWriter stringWriter = new StringWriter();
// Modify the engine context so that the custom writer is now the default
// output writer of the engine
engine.getContext().setWriter(stringWriter);

// Execute some script
try {
    engine.eval("print('Redirected text!');");
} catch (ScriptException ex) {
    ex.printStackTrace();
}

// Outcome:
// Nothing printed on standard output, but
// stringWriter.toString() contains 'Redirected text!'
```

```
// Obtain an instance of JavaScript engine
ScriptEngineManager manager = new ScriptEngineManager();
ScriptEngine engine = manager.getEngineByName("JavaScript");
```

```
//String to be evaluated
String str = "3+2*4+5";
//Value after doing Arithmetic operation with operator precedence will be 16

//Printing the value
try {
    System.out.println(engine.eval(str));
} catch (ScriptException ex) {
    ex.printStackTrace();
}

//Outcome:
//Value of the string after arithmetic evaluation is printed on standard output.
//In this case '16.0' will be printed on standard output.
```

Nashorn Java

Nashorn Java Java . JavaScript (Nashorn) Java .

Java .

:

1. if . if JS Java . falsy (null, undefined, 0,) false .
2. Nashorn JS Java .
3. . JS .

:

			.
Java null			
Java			0
		.	
Java / Long	! = 0		
Java ArrayList		.	
Java HashMap		.	
Java HashSet		.	

:

- null `if (some_string)` `if (some_string)` .
- `for each` , ,
- null undefined (Java)

```
import java.io.FileReader;
import java.io.IOException;

import javax.script.ScriptEngine;
import javax.script.ScriptEngineManager;
import javax.script.ScriptException;

public class InterfaceImplementationExample {
    public static interface Pet {
        public void eat();
    }

    public static void main(String[] args) throws IOException {
        // Obtain an instance of JavaScript engine
        ScriptEngineManager manager = new ScriptEngineManager();
        ScriptEngine engine = manager.getEngineByName("nashorn");

        try {
            //evaluate a script
            /* pet.js */
            /*
                var Pet = Java.type("InterfaceImplementationExample.Pet");

                new Pet() {
                    eat: function() { print("eat"); }
                }
            */

            Pet pet = (Pet) engine.eval(new FileReader("pet.js"));

            pet.eat();
        } catch (ScriptException ex) {
            ex.printStackTrace();
        }

        // Outcome:
        // 'eat' printed on standard output
    }
}
```

```
// Obtain an instance of JavaScript engine
ScriptEngineManager manager = new ScriptEngineManager();
ScriptEngine engine = manager.getEngineByName("nashorn");

try {
    // Set value in the global name space of the engine
    engine.put("name","Nashorn");
    // Execute an hardcoded script
    engine.eval("var value='Hello '+name+'!';");
    // Get value
    String value=(String)engine.get("value");
    System.out.println(value);
} catch (ScriptException ex) {
    // This is the generic Exception subclass for the Scripting API
    ex.printStackTrace();
}

// Outcome:
// 'Hello Nashorn!' printed on standard output
```

Nashorn : https://riptutorial.com/ko/java/topic/166/nashorn---

51: NIO -

SelectionKey . .

OP_READ . OP_WRITE . . OP_WRITE ( ).

, OP_CONNECT (, . SO). finishConnect() OP_CONNECT .

Examples

(: OP_CONNECT)

NIO Java 1.4 "" . I / O . SelectableChannel . C select() . .

- (OP_ACCEPT)
- (OP_CONNECT)
- FIFO (OP_READ)
- FIFO (OP_WRITE)

(/ / ...) I / O I / O (/ / ...) . I / O () I / O . .

.

1. Selector
2. SocketChannel
3. Selector SocketChannel
4. Selector

```
Selector sel = Selector.open(); // Create the Selector
SocketChannel sc = SocketChannel.open(); // Create a SocketChannel
sc.configureBlocking(false); // ... non blocking
sc.setOption(StandardSocketOptions.SO_KEEPALIVE, true); // ... set some options

// Register the Channel to the Selector for wake-up on CONNECT event and use some description
as an attachement
sc.register(sel, SelectionKey.OP_CONNECT, "Connection to google.com"); // Returns a
SelectionKey: the association between the SocketChannel and the Selector
System.out.println("Initiating connection");
if (sc.connect(new InetSocketAddress("www.google.com", 80)))
    System.out.println("Connected"); // Connected right-away: nothing else to do
else {
    boolean exit = false;
    while (!exit) {
        if (sel.select(100) == 0) // Did something happen on some registered Channels during
the last 100ms?
            continue; // No, wait some more

        // Something happened...
        Set<SelectionKey> keys = sel.selectedKeys(); // List of SelectionKeys on which some
registered operation was triggered
        for (SelectionKey k : keys) {
            System.out.println("Checking "+k.attachment());
```

```
            if (k.isConnectable()) { // CONNECT event
                System.out.print("Connected through select() on "+k.channel()+" -> ");
                if (sc.finishConnect()) { // Finish connection process
                    System.out.println("done!");
                    k.interestOps(k.interestOps() & ~SelectionKey.OP_CONNECT); // We are
already connected: remove interest in CONNECT event
                    exit = true;
                } else
                    System.out.println("unfinished...");
            }
            // TODO: else if (k.isReadable()) { ...
        }
        keys.clear(); // Have to clear the selected keys set once processed!
    }
}
System.out.print("Disconnecting ... ");
sc.shutdownOutput(); // Initiate graceful disconnection
// TODO: emtpy receive buffer
sc.close();
System.out.println("done");
```

:

```
Initiating connection
Checking Connection to google.com
Connected through 'select()' on java.nio.channels.SocketChannel[connection-pending
remote=www.google.com/216.58.208.228:80] -> done!
Disconnecting ... done
```

NIO - : https://riptutorial.com/ko/java/topic/5513/nio---

52: NumberFormat

Examples

NumberFormat

Java Locale . .

```
Locale locale = new Locale("en", "IN");
NumberFormat numberFormat = NumberFormat.getInstance(locale);
```

.

1. `numberFormat.format(10000000.99);`
2. `NumberFormat currencyFormat = NumberFormat.getCurrencyInstance(locale);`
   `currencyFormat.format(10340.999);`
3. `NumberFormat percentageFormat = NumberFormat.getPercentInstance(locale);`
   `percentageFormat.format(10929.999);`
4.

```
numberFormat.setMinimumIntegerDigits(int digits)
numberFormat.setMaximumIntegerDigits(int digits)
numberFormat.setMinimumFractionDigits(int digits)
numberFormat.setMaximumFractionDigits(int digits)
```

NumberFormat : https://riptutorial.com/ko/java/topic/7399/numberformat

53: RSA

Examples

OAEP GCM

128 AES AES GCM OAEP .

OAEP PKCS # 1 v1.5 . GCM .

. RSA . AES / GCM .

.

1. (`RSAPrivateKey getKeySize() getKeySize()`).
2. / . RSA .
3. GCM 128 (Java).

:

- 2048 RSA . ( ).
- AES-256 .
- Cryptographic Message Syntax (CMS / PKCS # 7) PGP .

:

```
/**
 * Encrypts the data using a hybrid crypto-system which uses GCM to encrypt the data and OAEP
to encrypt the AES key.
 * The key size of the AES encryption will be 128 bit.
 * All the default parameter choices are used for OAEP and GCM.
 *
 * @param publicKey the RSA public key used to wrap the AES key
 * @param plaintext the plaintext to be encrypted, not altered
 * @return the ciphertext
 * @throws InvalidKeyException if the key is not an RSA public key
 * @throws NullPointerException if the plaintext is null
 */
public static byte[] encryptData(PublicKey publicKey, byte[] plaintext)
        throws InvalidKeyException, NullPointerException {

    // --- create the RSA OAEP cipher ---

    Cipher oaep;
    try {
        // SHA-1 is the default and not vulnerable in this setting
        // use OAEPParameterSpec to configure more than just the hash
        oaep = Cipher.getInstance("RSA/ECB/OAEPwithSHA1andMGF1Padding");
    } catch (NoSuchAlgorithmException e) {
        throw new RuntimeException(
                "Runtime doesn't have support for RSA cipher (mandatory algorithm for
runtimes)", e);
    } catch (NoSuchPaddingException e) {
```

```
        throw new RuntimeException(
                "Runtime doesn't have support for OAEP padding (present in the standard Java
runtime sinze XX)", e);
    }
    oaep.init(Cipher.WRAP_MODE, publicKey);

    // --- wrap the plaintext in a buffer

    // will throw NullPointerException if plaintext is null
    ByteBuffer plaintextBuffer = ByteBuffer.wrap(plaintext);

    // --- generate a new AES secret key ---

    KeyGenerator aesKeyGenerator;
    try {
        aesKeyGenerator = KeyGenerator.getInstance("AES");
    } catch (NoSuchAlgorithmException e) {
        throw new RuntimeException(
                "Runtime doesn't have support for AES key generator (mandatory algorithm for
runtimes)", e);
    }
    // for AES-192 and 256 make sure you've got the rights (install the
    // Unlimited Crypto Policy files)
    aesKeyGenerator.init(128);
    SecretKey aesKey = aesKeyGenerator.generateKey();

    // --- wrap the new AES secret key ---

    byte[] wrappedKey;
    try {
        wrappedKey = oaep.wrap(aesKey);
    } catch (IllegalBlockSizeException e) {
        throw new RuntimeException(
                "AES key should always fit OAEP with normal sized RSA key", e);
    }

    // --- setup the AES GCM cipher mode ---

    Cipher aesGCM;
    try {
        aesGCM = Cipher.getInstance("AES/GCM/Nopadding");
        // we can get away with a zero nonce since the key is randomly generated
        // 128 bits is the recommended (maximum) value for the tag size
        // 12 bytes (96 bits) is the default nonce size for GCM mode encryption
        GCMParameterSpec staticParameterSpec = new GCMParameterSpec(128, new byte[12]);
        aesGCM.init(Cipher.ENCRYPT_MODE, aesKey, staticParameterSpec);
    } catch (NoSuchAlgorithmException e) {
        throw new RuntimeException(
                "Runtime doesn't have support for AES cipher (mandatory algorithm for
runtimes)", e);
    } catch (NoSuchPaddingException e) {
        throw new RuntimeException(
                "Runtime doesn't have support for GCM (present in the standard Java runtime
sinze XX)", e);
    } catch (InvalidAlgorithmParameterException e) {
        throw new RuntimeException(
                "IvParameterSpec not accepted by this implementation of GCM", e);
    }

    // --- create a buffer of the right size for our own protocol ---
```

```
    ByteBuffer ciphertextBuffer = ByteBuffer.allocate(
            Short.BYTES
            + oaep.getOutputSize(128 / Byte.SIZE)
            + aesGCM.getOutputSize(plaintext.length));

    // - element 1: make sure that we know the size of the wrapped key
    ciphertextBuffer.putShort((short) wrappedKey.length);

    // - element 2: put in the wrapped key
    ciphertextBuffer.put(wrappedKey);

    // - element 3: GCM encrypt into buffer
    try {
        aesGCM.doFinal(plaintextBuffer, ciphertextBuffer);
    } catch (ShortBufferException | IllegalBlockSizeException | BadPaddingException e) {
        throw new RuntimeException("Cryptographic exception, AES/GCM encryption should not
fail here", e);
    }

    return ciphertextBuffer.array();
}
```

. .

```
/**
 * Decrypts the data using a hybrid crypto-system which uses GCM to encrypt
 * the data and OAEP to encrypt the AES key. All the default parameter
 * choices are used for OAEP and GCM.
 *
 * @param privateKey
 *            the RSA private key used to unwrap the AES key
 * @param ciphertext
 *            the ciphertext to be encrypted, not altered
 * @return the plaintext
 * @throws InvalidKeyException
 *             if the key is not an RSA private key
 * @throws NullPointerException
 *             if the ciphertext is null
 * @throws IllegalArgumentException
 *             with the message "Invalid ciphertext" if the ciphertext is invalid (minimize
information leakage)
 */
public static byte[] decryptData(PrivateKey privateKey, byte[] ciphertext)
        throws InvalidKeyException, NullPointerException {

    // --- create the RSA OAEP cipher ---

    Cipher oaep;
    try {
        // SHA-1 is the default and not vulnerable in this setting
        // use OAEPParameterSpec to configure more than just the hash
        oaep = Cipher.getInstance("RSA/ECB/OAEPwithSHA1andMGF1Padding");
    } catch (NoSuchAlgorithmException e) {
        throw new RuntimeException(
                "Runtime doesn't have support for RSA cipher (mandatory algorithm for
runtimes)",
                e);
    } catch (NoSuchPaddingException e) {
        throw new RuntimeException(
                "Runtime doesn't have support for OAEP padding (present in the standard Java
```

```
runtime sinze XX)",
                e);
    }
    oaep.init(Cipher.UNWRAP_MODE, privateKey);

    // --- wrap the ciphertext in a buffer

    // will throw NullPointerException if ciphertext is null
    ByteBuffer ciphertextBuffer = ByteBuffer.wrap(ciphertext);

    // sanity check #1
    if (ciphertextBuffer.remaining() < 2) {
        throw new IllegalArgumentException("Invalid ciphertext");
    }
    // - element 1: the length of the encapsulated key
    int wrappedKeySize = ciphertextBuffer.getShort() & 0xFFFF;
    // sanity check #2
    if (ciphertextBuffer.remaining() < wrappedKeySize + 128 / Byte.SIZE) {
        throw new IllegalArgumentException("Invalid ciphertext");
    }

    // --- unwrap the AES secret key ---

    byte[] wrappedKey = new byte[wrappedKeySize];
    // - element 2: the encapsulated key
    ciphertextBuffer.get(wrappedKey);
    SecretKey aesKey;
    try {
        aesKey = (SecretKey) oaep.unwrap(wrappedKey, "AES",
                Cipher.SECRET_KEY);
    } catch (NoSuchAlgorithmException e) {
        throw new RuntimeException(
                "Runtime doesn't have support for AES cipher (mandatory algorithm for
runtimes)",
                e);
    } catch (InvalidKeyException e) {
        throw new RuntimeException(
                "Invalid ciphertext");
    }

    // --- setup the AES GCM cipher mode ---

    Cipher aesGCM;
    try {
        aesGCM = Cipher.getInstance("AES/GCM/Nopadding");
        // we can get away with a zero nonce since the key is randomly
        // generated
        // 128 bits is the recommended (maximum) value for the tag size
        // 12 bytes (96 bits) is the default nonce size for GCM mode
        // encryption
        GCMParameterSpec staticParameterSpec = new GCMParameterSpec(128,
                new byte[12]);
        aesGCM.init(Cipher.DECRYPT_MODE, aesKey, staticParameterSpec);
    } catch (NoSuchAlgorithmException e) {
        throw new RuntimeException(
                "Runtime doesn't have support for AES cipher (mandatory algorithm for
runtimes)",
                e);
    } catch (NoSuchPaddingException e) {
        throw new RuntimeException(
                "Runtime doesn't have support for GCM (present in the standard Java runtime
```

```
sinze XX)",
                e);
    } catch (InvalidAlgorithmParameterException e) {
        throw new RuntimeException(
                "IvParameterSpec not accepted by this implementation of GCM",
                e);
    }

    // --- create a buffer of the right size for our own protocol ---

    ByteBuffer plaintextBuffer = ByteBuffer.allocate(aesGCM
            .getOutputSize(ciphertextBuffer.remaining()));

    // - element 3: GCM ciphertext
    try {
        aesGCM.doFinal(ciphertextBuffer, plaintextBuffer);
    } catch (ShortBufferException | IllegalBlockSizeException
            | BadPaddingException e) {
        throw new RuntimeException(
                "Invalid ciphertext");
    }

    return plaintextBuffer.array();
}
```

RSA : https://riptutorial.com/ko/java/topic/1889/rsa-

54: ServiceLoader

`ServiceLoader` `.jar` (=) . / .

/ `ServiceLoader` 0 .

`ServiceLoader` jar `META-INF/services` . .

Examples

`ServiceLoader` .

```
package servicetest;

import java.io.IOException;

public interface Logger extends AutoCloseable {

    void log(String message) throws IOException;
}
```

`System.err` .

```
package servicetest.logger;

import servicetest.Logger;

public class ConsoleLogger implements Logger {

    @Override
    public void log(String message) {
        System.err.println(message);
    }

    @Override
    public void close() {
    }

}
```

.

```
package servicetest.logger;

import java.io.BufferedWriter;
import java.io.FileWriter;
import java.io.IOException;
import servicetest.Logger;

public class FileLogger implements Logger {
```

```
    private final BufferedWriter writer;

    public FileLogger() throws IOException {
        writer = new BufferedWriter(new FileWriter("log.txt"));
    }

    @Override
    public void log(String message) throws IOException {
        writer.append(message);
        writer.newLine();
    }

    @Override
    public void close() throws IOException {
        writer.close();
    }

}
```

META-INF / services / servicetest.Logger

`META-INF/services/servicetest.Logger` Logger .

```
servicetest.logger.ConsoleLogger
servicetest.logger.FileLogger
```

`main` . ServiceLoader .

```
public static void main(String[] args) throws Exception {
    final String message = "Hello World!";

    // get ServiceLoader for Logger
    ServiceLoader<Logger> loader = ServiceLoader.load(servicetest.Logger.class);

    // iterate through instances of available loggers, writing the message to each one
    Iterator<Logger> iterator = loader.iterator();
    while (iterator.hasNext()) {
        try (Logger logger = iterator.next()) {
            logger.log(message);
        }
    }
}
```

ServiceLoader

ServiceLoader . - ( ) - Java SE . ServiceLoader . Java API ServiceLoader .

.

-
- `ServiceLoader` `ServiceLoader`
-

`accounting-api.jar` jar .

```
package example;

public interface AccountingService {

  long getBalance();
}
```

`accounting-impl.jar` .

```
package example.impl;
import example.AccountingService;

public interface DefaultAccountingService implements AccouningService {

  public long getBalance() {
    return balanceFromDB();
  }

  private long balanceFromDB(){
    ...
  }
}
```

, `accounting-impl.jar` jar `AccountingService` . `META-INF/services/` .

- `META-INF/services/example.AccountingService`

.

```
example.impl.DefaultAccountingService
```

jar classpath `AccountingService` ServiceLauncher

```
ServiceLoader<AccountingService> loader = ServiceLoader.load(AccountingService.class)
AccountingService service = loader.next();
long balance = service.getBalance();
```

`ServiceLoader` `Iterable` .

```
ServiceLoader<AccountingService> loader = ServiceLoader.load(AccountingService.class)
for(AccountingService service : loader) {
   //...
}
```

`next()` . ServiceLoader for-each `iterator()` .

ServiceLoader : https://riptutorial.com/ko/java/topic/5433/serviceloader

55: SortedMap

.

Examples

.

:-

- SortedMap .
- .

:

- ().
- firstKey ().
- SortedMap headMap (Object end).
- Object lastKey ().
- SortedMap subMap (,).
- SortedMap tailMap (Object start).

```
public static void main(String args[]) {
      // Create a hash map
      TreeMap tm = new TreeMap();

      // Put elements to the map
      tm.put("Zara", new Double(3434.34));
      tm.put("Mahnaz", new Double(123.22));
      tm.put("Ayan", new Double(1378.00));
      tm.put("Daisy", new Double(99.22));
      tm.put("Qadir", new Double(-19.08));

      // Get a set of the entries
      Set set = tm.entrySet();

      // Get an iterator
      Iterator i = set.iterator();

      // Display elements
      while(i.hasNext()) {
         Map.Entry me = (Map.Entry)i.next();
         System.out.print(me.getKey() + ": ");
         System.out.println(me.getValue());
      }
      System.out.println();

      // Deposit 1000 into Zara's account
      double balance = ((Double)tm.get("Zara")).doubleValue();
      tm.put("Zara", new Double(balance + 1000));
      System.out.println("Zara's new balance: " + tm.get("Zara"));
   }
```

SortedMap : https://riptutorial.com/ko/java/topic/10748/sortedmap

56: String Tokenizer

java.util.StringTokenizer . .

( ) .

Examples

StringTokenizer

```
import java.util.StringTokenizer;
public class Simple{
 public static void main(String args[]){
   StringTokenizer st = new StringTokenizer("apple ball cat dog"," ");
     while (st.hasMoreTokens()) {
         System.out.println(st.nextToken());
     }
   }
}
```

:

StringTokenizer ','

```
public static void main(String args[]) {
        StringTokenizer st = new StringTokenizer("apple,ball cat,dog", ",");
        while (st.hasMoreTokens()) {
            System.out.println(st.nextToken());
        }
    }
```

:

String Tokenizer : https://riptutorial.com/ko/java/topic/10563/string-tokenizer

57: StringBuffer

Java StringBuffer .

Examples

:-

- mutable ( ) .
- : - .
- .

: -

- public synchronized StringBuffer append (String s)
- public synchronized StringBuffer insert (int offset, String s) .
- public synchronized StringBuffer replace (int startIndex, int endIndex, String str) .
- public synchronized StringBuffer delete (int startIndex, int endIndex) .
- public synchronized StringBuffer reverse ()
- public int capacity ()
- public void ensureCapacity (int minimumCapacity)
- char charAt (int index)
- public int length ()
- (int beginIndex)
- (int beginIndex, int endIndex)

diffrence : -

```
class Test {
 public static void main(String args[])
 {
  String str = "study";
  str.concat("tonight");
  System.out.println(str);      // Output: study

  StringBuffer strB = new StringBuffer("study");
  strB.append("tonight");
  System.out.println(strB);    // Output: studytonight
 }
```

```
}
```

StringBuffer : https://riptutorial.com/ko/java/topic/10757/stringbuffer

58: StringBuilder

Java StringBuilder ( ) . Java StringBuilder StringBuffer . JDK 1.5 .

- StringBuilder ()
- StringBuilder (int)
- StringBuilder (CharSequence seq)
- StringBuilder (StringBuilder)
- StringBuilder ( )
- StringJoiner (CharSequence)
- StringJoiner (CharSequence , CharSequence , CharSequence)

char StringBuilder int capacity String string .

```
 StringBuilder v = new StringBuilder('I'); //'I' is a character, "I" is a String.
 System.out.println(v.capacity()); --> output 73
 System.out.println(v.toString()); --> output nothing
```

Examples

n .

: String s n String .

String .

```
final int n = ...
final String s = ...
String result = "";

for (int i = 0; i < n; i++) {
    result += s;
}
```

n 1 n s O(s.length() * n²) = O(s.length() * (1+2+...+(n-1)+n)) .

StringBuilder O(s.length() * n) String .

```
final int n = ...
final String s = ...

StringBuilder builder = new StringBuilder();
```

```
for (int i = 0; i < n; i++) {
    builder.append(s);
}

String result = builder.toString();
```

StringBuffer, StringBuilder, Formatter StringJoiner

`StringBuffer` , `StringBuilder` , `Formatter` `StringJoiner` Java SE .

- `StringBuffer` , Java 1.0 , 「」 .
- `StringBuffer` Java 5 `StringBuilder` . clases API . `StringBuffer` `StringBuilder` .

`StringBuilder` .

```
int one = 1;
String color = "red";
StringBuilder sb = new StringBuilder();
sb.append("One=").append(one).append(", Color=").append(color).append('\n');
System.out.print(sb);
// Prints "One=1, Colour=red" followed by an ASCII newline.
```

(`StringBuffer` . `StringBuilder` `StringBuffer` .)

`StringBuffer` `StringBuilder` . , . remaming .

- `Formatter` Java 5 C `sprintf` . . .
- `StringJoiner` Java 8 . , . API Java 8 .

`Formatter` .

```
// This does the same thing as the StringBuilder example above
int one = 1;
String color = "red";
Formatter f = new Formatter();
System.out.print(f.format("One=%d, colour=%s%n", one, color));
// Prints "One=1, Colour=red" followed by the platform's line separator

// The same thing using the `String.format` convenience method
System.out.print(String.format("One=%d, color=%s%n", one, color));
```

`StringJoiner` .

```
StringJoiner sj = new StringJoiner(", ", "[", "]");
for (String s : new String[]{"A", "B", "C"}) {
    sj.add(s);
}
System.out.println(sj);
// Prints "[A, B, C]"
```

4 .

- `StringBuilder` .
- `StringBuffer` `StringBuilder` .
- `Formatter` `StringBuilder` . `Formatter.format(...)` .
  - `format` ,
  - *varargs* .
  - .
- `StringJoiner` `StringJoiner` .

StringBuilder : https://riptutorial.com/ko/java/topic/1037/stringbuilder

59: sun.misc.Unsafe

`Unsafe` Java . `Unsafe` .

1. `Unsafe` API JVM .
2. `Unsafe` API . Java .

Examples

sun.misc.Unsafe

```
public static Unsafe getUnsafe() {
    try {
        Field unsafe = Unsafe.class.getDeclaredField("theUnsafe");
        unsafe.setAccessible(true);
        return (Unsafe) unsafe.get(null);
    } catch (IllegalAccessException e) {
        // Handle
    } catch (IllegalArgumentException e) {
        // Handle
    } catch (NoSuchFieldException e) {
        // Handle
    } catch (SecurityException e) {
        // Handle
    }
}
```

`sun.misc.Unsafe` Private `getUnsafe()` . .

bootclasspath sun.misc.Unsafe

```
public class UnsafeLoader {
    public static Unsafe loadUnsafe() {
        return Unsafe.getUnsafe();
    }
}
```

Unsafe . JVM .

```
java -Xbootclasspath:$JAVA_HOME/jre/lib/rt.jar:./UnsafeLoader.jar foo.bar.MyApp
```

`foo.bar.MyApp` `UnsafeLoader.loadUnsafe()` .

. private `public static Unsafe getUnsafe()` `public static Unsafe getUnsafe()` . .

```
public static final Unsafe UNSAFE;

static {
    Unsafe unsafe = null;
```

```
    try {
        final PrivilegedExceptionAction<Unsafe> action = () -> {
            final Field f = Unsafe.class.getDeclaredField("theUnsafe");
            f.setAccessible(true);

            return (Unsafe) f.get(null);
        };

        unsafe = AccessController.doPrivileged(action);
    } catch (final Throwable t) {
        throw new RuntimeException("Exception accessing Unsafe", t);
    }

    UNSAFE = unsafe;
}
```

.

	API
/ ,	`allocateMemory(bytes)` , `reallocateMemory(address, bytes)` `freeMemory(address)`
	`loadFence()` , `storeFence()` , `fullFence()`
	`park(isAbsolute, time)` , `unpark(thread)`
/	`get*` `put*`
.	`throwException(e)`
CAS	`compareAndSwap*`
	`setMemory`
	`get*Volatile` , `put*Volatile` , `putOrdered*`

get put . null , .

```
// Putting a value to a field
protected static long fieldOffset = UNSAFE.objectFieldOffset(getClass().getField("theField"));
UNSAFE.putLong(this, fieldOffset , newValue);

// Puting an absolute value
 UNSAFE.putLong(null, address, newValue);
 UNSAFE.putLong(address, newValue);
```

int long . `floatToRawIntBits` , `intBitsToFloat`, doubleToRawLongBits , longBitsToDouble` float double .

sun.misc.Unsafe : https://riptutorial.com/ko/java/topic/6771/sun-misc-unsafe

60: ThreadLocal

`SimpleDateFormat` , `Marshaller` stateless .

`Random` ThreadLocal , `ThreadLocalRandom`

Examples

ThreadLocal Java 8

```
public static class ThreadLocalExample
{
    private static final ThreadLocal<SimpleDateFormat> format =
        ThreadLocal.withInitial(() -> new SimpleDateFormat("yyyyMMdd_HHmm"));

    public String formatDate(Date date)
    {
        return format.get().format(date);
    }
}
```

ThreadLocal

Java `ThreadLocal` . Object . ThreadLocal . , ThreadLocal .

`ThreadLocal` . . `SimpleDateFormat` `SimpleDateFormat` Static `SimpleDateFormat` `SimpleDateFormat` ThreadLocal .

.

`ThreadLocal` `get()` `set()` Thread .

`ThreadLocal` .

Java ThreadLocal `ThreadLocal` .

```
package com.examples.threads;

import java.text.SimpleDateFormat;
import java.util.Random;

public class ThreadLocalExample implements Runnable{

    // SimpleDateFormat is not thread-safe, so give one to each thread
 // SimpleDateFormat is not thread-safe, so give one to each thread
    private static final ThreadLocal<SimpleDateFormat> formatter = new
ThreadLocal<SimpleDateFormat>(){
        @Override
        protected SimpleDateFormat initialValue()
        {
            return new SimpleDateFormat("yyyyMMdd HHmm");
        }
```

```
    };

    public static void main(String[] args) throws InterruptedException {
        ThreadLocalExample obj = new ThreadLocalExample();
        for(int i=0 ; i<10; i++){
            Thread t = new Thread(obj, ""+i);
            Thread.sleep(new Random().nextInt(1000));
            t.start();
        }
    }

    @Override
    public void run() {
        System.out.println("Thread Name= "+Thread.currentThread().getName()+" default
Formatter = "+formatter.get().toPattern());
        try {
            Thread.sleep(new Random().nextInt(1000));
        } catch (InterruptedException e) {
            e.printStackTrace();
        }

        formatter.set(new SimpleDateFormat());

        System.out.println("Thread Name= "+Thread.currentThread().getName()+" formatter =
"+formatter.get().toPattern());
    }

}
```

:

```
Thread Name= 0 default Formatter = yyyyMMdd HHmm

Thread Name= 1 default Formatter = yyyyMMdd HHmm

Thread Name= 0 formatter = M/d/yy h:mm a

Thread Name= 2 default Formatter = yyyyMMdd HHmm

Thread Name= 1 formatter = M/d/yy h:mm a

Thread Name= 3 default Formatter = yyyyMMdd HHmm

Thread Name= 4 default Formatter = yyyyMMdd HHmm

Thread Name= 4 formatter = M/d/yy h:mm a

Thread Name= 5 default Formatter = yyyyMMdd HHmm

Thread Name= 2 formatter = M/d/yy h:mm a

Thread Name= 3 formatter = M/d/yy h:mm a

Thread Name= 6 default Formatter = yyyyMMdd HHmm

Thread Name= 5 formatter = M/d/yy h:mm a

Thread Name= 6 formatter = M/d/yy h:mm a

Thread Name= 7 default Formatter = yyyyMMdd HHmm

Thread Name= 8 default Formatter = yyyyMMdd HHmm
```

```
Thread Name= 8 formatter = M/d/yy h:mm a

Thread Name= 7 formatter = M/d/yy h:mm a

Thread Name= 9 default Formatter = yyyyMMdd HHmm

Thread Name= 9 formatter = M/d/yy h:mm a
```

Thread-0 thread-2 .

/ . . ( ).

```
public class Test {
    public static void main(String[] args) {
        Foo foo = new Foo();
        new Thread(foo, "Thread 1").start();
        new Thread(foo, "Thread 2").start();
    }
}
```

Foo 0 . ThreadLocal . ThreadLocal .

```
public class Foo implements Runnable {
    private static final int ITERATIONS = 10;
    private static final ThreadLocal<Integer> threadLocal = new ThreadLocal<Integer>() {
        @Override
        protected Integer initialValue() {
            return 0;
        }
    };

    @Override
    public void run() {
        for (int i = 0; i < ITERATIONS; i++) {
            synchronized (threadLocal) {
                //Although accessing a static field, we get our own (previously saved) value.
                int value = threadLocal.get();
                System.out.println(Thread.currentThread().getName() + ": " + value);

                //Update our own variable
                threadLocal.set(value + 1);

                try {
                    threadLocal.notifyAll();
                    if (i < ITERATIONS - 1) {
                        threadLocal.wait();
                    }
                } catch (InterruptedException ex) {
                }
            }
        }
    }
}
```

:

```
Thread 1: 0
Thread 2: 0
```

```
Thread 1: 1
Thread 2: 1
Thread 1: 2
Thread 2: 2
Thread 1: 3
Thread 2: 3
Thread 1: 4
Thread 2: 4
Thread 1: 5
Thread 2: 5
Thread 1: 6
Thread 2: 6
Thread 1: 7
Thread 2: 7
Thread 1: 8
Thread 2: 8
Thread 1: 9
Thread 2: 9
```

ThreadLocal : https://riptutorial.com/ko/java/topic/2001/threadlocal

61: TreeMap TreeSet

`TreeMap` `TreeSet` Java 1.2 Java . `TreeMap` `Map` . `TreeSet` `Set` .

`TreeMap` `O(log n)` Red-Black . `TreeSet` , `TreeMap` .

.

Examples

Java TreeMap

, .

Java SE 7

```
TreeMap<Integer, String> treeMap = new TreeMap<>();
```

Java SE 7

```
TreeMap<Integer, String> treeMap = new TreeMap<Integer, String>();
```

```
treeMap.put(10, "ten");
treeMap.put(4, "four");
treeMap.put(1, "one");
treeSet.put(12, "twelve");
```

.

```
System.out.println(treeMap.firstEntry()); // Prints 1=one
System.out.println(treeMap.lastEntry()); // Prints 12=twelve
System.out.println(treeMap.size()); // Prints 4, since there are 4 elemens in the map
System.out.println(treeMap.get(12)); // Prints twelve
System.out.println(treeMap.get(15)); // Prints null, since the key is not found in the map
```

Iterator foreach . .

Java SE 7

```
for (Entry<Integer, String> entry : treeMap.entrySet()) {
    System.out.print(entry + " "); //prints 1=one 4=four 10=ten 12=twelve
}
```

```
Iterator<Entry<Integer, String>> iter = treeMap.entrySet().iterator();
while (iter.hasNext()) {
    System.out.print(iter.next() + " "); //prints 1=one 4=four 10=ten 12=twelve
}
```

Java TreeSet

.

Java SE 7

```
TreeSet<Integer> treeSet = new TreeSet<>();
```

Java SE 7

```
TreeSet<Integer> treeSet = new TreeSet<Integer>();
```

```
treeSet.add(10);
treeSet.add(4);
treeSet.add(1);
treeSet.add(12);
```

.

```
System.out.println(treeSet.first()); // Prints 1
System.out.println(treeSet.last()); // Prints 12
System.out.println(treeSet.size()); // Prints 4, since there are 4 elemens in the set
System.out.println(treeSet.contains(12)); // Prints true
System.out.println(treeSet.contains(15)); // Prints false
```

Iterator foreach . .

Java SE 7

```
for (Integer i : treeSet) {
    System.out.print(i + " "); //prints 1 4 10 12
}
```

```
Iterator<Integer> iter = treeSet.iterator();
while (iter.hasNext()) {
    System.out.print(iter.next() + " "); //prints 1 4 10 12
}
```

Java TreeMap / TreeSet

`TreeMap` `TreeSet` / `TreeMap` `TreeSet` .

`Person` .

```
public class Person  {

    private int id;
    private String firstName, lastName;
    private Date birthday;

    //... Constuctors, getters, setters and various methods
}
```

`TreeSet` (TreeMap Key) , .

```
TreeSet<Person2> set = ...
set.add(new Person(1,"first","last",Date.from(Instant.now())));
```

Exception .

```
Exception in thread "main" java.lang.ClassCastException: Person cannot be cast to
java.lang.Comparable
    at java.util.TreeMap.compare(TreeMap.java:1294)
    at java.util.TreeMap.put(TreeMap.java:538)
    at java.util.TreeSet.add(TreeSet.java:255)
```

, id (`private int id`) `Person` . :

1. `Person` Comparable .

```
public class Person implements Comparable<Person> {
    private int id;
    private String firstName, lastName;
    private Date birthday;

    //... Constuctors, getters, setters and various methods

    @Override
    public int compareTo(Person o) {
        return Integer.compare(this.id, o.id); //Compare by id
    }
}
```

2. `TreeSet` A :

Java SE 8

```
TreeSet<Person> treeSet = new TreeSet<>((personA, personB) -> Integer.compare(personA.getId(),
personB.getId()));
```

```
TreeSet<Person> treeSet = new TreeSet<>(new Comparator<Person>(){
    @Override
    public int compare(Person personA, Person personB) {
        return Integer.compare(personA.getId(), personB.getId());
    }
});
```

.

1. `TreeSet` / `TreeMap` . `id` .

2. . Javadoc :

   x y `sgn(x.compareTo(y)) == -sgn(y.compareTo(x))` . (`y.compareTo(x)` throw `x.compareTo(y)` throw .

   : `(x.compareTo(y)>0 && y.compareTo(z)>0) x.compareTo(z)>0` .

   , z `x.compareTo(y)==0` `sgn(x.compareTo(z)) == sgn(y.compareTo(z))` .

TreeMap TreeSet

TreeMap TreeSet , .

TreeMap TreeSet . .

, thread , ConcurrentModificationException Throw . / .

1. Collections.synchronizedSorted.. :

```
SortedSet<Integer> set = Collections.synchronizedSortedSet(new TreeSet<Integer>());
SortedMap<Integer,String> map = Collections.synchronizedSortedMap(new
TreeMap<Integer,String>());
```

, SortedSet / SortedMap . .

2. .

```
 TreeSet<Integer> set = new TreeSet<>();
```

...

```
//Thread 1
synchronized (set) {
    set.add(4);
}
```

...

```
//Thread 2
synchronized (set) {
    set.remove(5);
}
```

3. ReentrantReadWriteLock :

```
 TreeSet<Integer> set = new TreeSet<>();
 ReentrantReadWriteLock lock = new ReentrantReadWriteLock();
```

...

```
 //Thread 1
 lock.writeLock().lock();
 set.add(4);
 lock.writeLock().unlock();
```

...

```
 //Thread 2
 lock.readLock().lock();
```

```
set.contains(5);
lock.readLock().unlock();
```

ReadWriteLock .

TreeMap TreeSet : https://riptutorial.com/ko/java/topic/9905/treemap--treeset

62: XJC

XJC XML Java Java SE .

JDK `/bin/xjc` .

- xjc [options] / URL / dir / jar ... [-b bindinfo] ...

Java xsd

XJC JDK . JAXB (un) Java .

Examples

XSD Java

XSD (schema.xsd)

xml (xsd) `name` `reputation` .

```
<?xml version="1.0"?>

<xs:schema version="1.0"
           xmlns:xs="http://www.w3.org/2001/XMLSchema"
           xmlns:ns="http://www.stackoverflow.com/users"
           elementFormDefault="qualified"
           targetNamespace="http://www.stackoverflow.com/users">
    <xs:element name="users" type="ns:Users"/>

    <xs:complexType name="Users">
        <xs:sequence>
            <xs:element type="ns:User" name="user" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="User">
        <xs:attribute name="name" use="required" type="xs:string"/>
        <xs:attribute name="reputation" use="required">
            <xs:simpleType>
                <xs:restriction base="xs:int">
                    <xs:minInclusive value="1"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:attribute>
    </xs:complexType>
</xs:schema>
```

xjc

xjc (JDK 2) OS .

.

```
xjc schema.xsd
```

java .

java .

```
package com.stackoverflow.users;

import java.util.ArrayList;
import java.util.List;
import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlType;

@XmlAccessorType(XmlAccessType.FIELD)
@XmlType(name = "Users", propOrder = {
    "user"
})
public class Users {

    protected List<User> user;

    public List<User> getUser() {
        if (user == null) {
            user = new ArrayList<User>();
        }
        return this.user;
    }

}
```

```
package com.stackoverflow.users;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.XmlType;

@XmlAccessorType(XmlAccessType.FIELD)
@XmlType(name = "User")
public class User {

    @XmlAttribute(name = "name", required = true)
    protected String name;
    @XmlAttribute(name = "reputation", required = true)
    protected int reputation;

    public String getName() {
        return name;
```

```
    }

    public void setName(String value) {
        this.name = value;
    }

    public int getReputation() {
        return reputation;
    }

    public void setReputation(int value) {
        this.reputation = value;
    }

}
```

```
package com.stackoverflow.users;

import javax.xml.bind.JAXBElement;
import javax.xml.bind.annotation.XmlElementDecl;
import javax.xml.bind.annotation.XmlRegistry;
import javax.xml.namespace.QName;

@XmlRegistry
public class ObjectFactory {

    private final static QName _Users_QNAME = new QName("http://www.stackoverflow.com/users",
"users");

    public ObjectFactory() {
    }

    public Users createUsers() {
        return new Users();
    }

    public User createUser() {
        return new User();
    }

    @XmlElementDecl(namespace = "http://www.stackoverflow.com/users", name = "users")
    public JAXBElement<Users> createUsers(Users value) {
        return new JAXBElement<Users>(_Users_QNAME, Users.class, null, value);
    }

}
```

package-info.java

```
@javax.xml.bind.annotation.XmlSchema(namespace = "http://www.stackoverflow.com/users",
elementFormDefault = javax.xml.bind.annotation.XmlNsForm.QUALIFIED)
package com.stackoverflow.users;
```

XJC : https://riptutorial.com/ko/java/topic/4538/xjc

63: XML XPath

XPath XML .

W3C .

Examples

XML NodeList

XML :

```
<documentation>
    <tags>
        <tag name="Java">
            <topic name="Regular expressions">
                <example>Matching groups</example>
                <example>Escaping metacharacters</example>
            </topic>
            <topic name="Arrays">
                <example>Looping over arrays</example>
                <example>Converting an array to a list</example>
            </topic>
        </tag>
        <tag name="Android">
            <topic name="Building Android projects">
                <example>Building an Android application using Gradle</example>
                <example>Building an Android application using Maven</example>
            </topic>
            <topic name="Layout resources">
                <example>Including layout resources</example>
                <example>Supporting multiple device screens</example>
            </topic>
        </tag>
    </tags>
</documentation>
```

Java `example` (XML XPath , XML XML).

```
XPathFactory xPathFactory = XPathFactory.newInstance();
XPath xPath = xPathFactory.newXPath(); //Make new XPath
InputSource inputSource = new InputSource("path/to/xml.xml"); //Specify XML file path

NodeList javaExampleNodes = (NodeList)
xPath.evaluate("/documentation/tags/tag[@name='Java']//example", inputSource,
XPathConstants.NODESET); //Evaluate the XPath
...
```

XML XPath

XML NodeList XPath .

XML :

```
<documentation>
    <tags>
        <tag name="Java">
            <topic name="Regular expressions">
                <example>Matching groups</example>
                <example>Escaping metacharacters</example>
            </topic>
            <topic name="Arrays">
                <example>Looping over arrays</example>
                <example>Converting an array to a list</example>
            </topic>
        </tag>
        <tag name="Android">
            <topic name="Building Android projects">
                <example>Building an Android application using Gradle</example>
                <example>Building an Android application using Maven</example>
            </topic>
            <topic name="Layout resources">
                <example>Including layout resources</example>
                <example>Supporting multiple device screens</example>
            </topic>
        </tag>
    </tags>
</documentation>
```

XPath .

```
XPath xPath = XPathFactory.newInstance().newXPath(); //Make new XPath
DocumentBuilder builder = DocumentBuilderFactory.newInstance();
Document doc = builder.parse(new File("path/to/xml.xml")); //Specify XML file path

NodeList javaExampleNodes = (NodeList)
xPath.evaluate("/documentation/tags/tag[@name='Java']//example", doc, XPathConstants.NODESET);
//Evaluate the XPath
xPath.reset(); //Resets the xPath so it can be used again
NodeList androidExampleNodes = (NodeList)
xPath.evaluate("/documentation/tags/tag[@name='Android']//example", doc,
XPathConstants.NODESET); //Evaluate the XPath

...
```

XPath XML

, evaluate compile . .

```
XPath xPath = XPathFactory.newInstance().newXPath(); //Make new XPath
XPathExpression exp = xPath.compile("/documentation/tags/tag[@name='Java']//example");
DocumentBuilder builder = DocumentBuilderFactory.newInstance();
Document doc = builder.parse(new File("path/to/xml.xml")); //Specify XML file path

NodeList javaExampleNodes = (NodeList) exp.evaluate(doc, XPathConstants.NODESET); //Evaluate
the XPath from the already-compiled expression

NodeList javaExampleNodes2 = (NodeList) exp.evaluate(doc, XPathConstants.NODESET); //Do it
again
```

`XPathExpression.evaluate()` `XPath.evaluate()` .

XML XPath : https://riptutorial.com/ko/java/topic/4148/xml-xpath-

64: XOM - XML

Examples

XML

XOM XML `Document` `Builder` .

```
Builder builder = new Builder();
Document doc = builder.build(file);
```

---

xml `Document getRootElement()` .

```
Element root = doc.getRootElement();
```

Element xml . .

- `getChildElements(String name)` - `Elements` `Elements` .
- `getFirstChildElement(String name)` - .
- `getValue()` - .
- `getAttributeValue(String name)` - .

`getChildElements()` `Elements` . `get(int index)` .

```
Elements colors = root.getChildElements("color");
for (int q = 0; q < colors.size(); q++){
    Element color = colors.get(q);
}
```

: XML .

XML :

```xml
<example>
    <person>
        <name>
            <first>Dan</first>
            <last>Smith</last>
        </name>
        <age unit="years">23</age>
        <fav_color>green</fav_color>
    </person>
    <person>
        <name>
            <first>Bob</first>
            <last>Autry</last>
        </name>
        <age unit="months">3</age>
        <fav_color>N/A</fav_color>
    </person>
</example>
```

:

```
import java.io.File;
import java.io.IOException;
import nu.xom.Builder;
import nu.xom.Document;
import nu.xom.Element;
import nu.xom.Elements;
import nu.xom.ParsingException;

public class XMLReader {

    public static void main(String[] args) throws ParsingException, IOException{
        File file = new File("insert path here");
        // builder builds xml data
        Builder builder = new Builder();
        Document doc = builder.build(file);

        // get the root element <example>
        Element root = doc.getRootElement();

        // gets all element with tag <person>
        Elements people = root.getChildElements("person");

        for (int q = 0; q < people.size(); q++){
            // get the current person element
            Element person = people.get(q);

            // get the name element and its children: first and last
            Element nameElement = person.getFirstChildElement("name");
            Element firstNameElement = nameElement.getFirstChildElement("first");
            Element lastNameElement = nameElement.getFirstChildElement("last");

            // get the age element
```

```
            Element ageElement = person.getFirstChildElement("age");

            // get the favorite color element
            Element favColorElement = person.getFirstChildElement("fav_color");

            String fName, lName, ageUnit, favColor;
            int age;

            try {
                fName = firstNameElement.getValue();
                lName = lastNameElement.getValue();
                age = Integer.parseInt(ageElement.getValue());
                ageUnit = ageElement.getAttributeValue("unit");
                favColor = favColorElement.getValue();

                System.out.println("Name: " + lName + ", " + fName);
                System.out.println("Age: " + age + " (" + ageUnit + ")");
                System.out.println("Favorite Color: " + favColor);
                System.out.println("----------------");

            } catch (NullPointerException ex){
                ex.printStackTrace();
            } catch (NumberFormatException ex){
                ex.printStackTrace();
            }
        }
    }

}
```

:

```
Name: Smith, Dan
Age: 23 (years)
Favorite Color: green
----------------
Name: Autry, Bob
Age: 3 (months)
Favorite Color: N/A
----------------
```

XML

XOM XML .

Element `Element(String name)` . Document .

```
Element root = new Element("root");
```

`Element` `Element` . .

- `appendChild(String name)` - name .
- `appendChild(Node node)` - `node` . .
- `addAttribute(Attribute attribute)` - .

Attribute . Attribute(String name, String value) .

Document . Document Element .

Serializer XML . Serializer .

```
FileOutputStream fileOutputStream = new FileOutputStream(file);
Serializer serializer = new Serializer(fileOutputStream, "UTF-8");
serializer.setIndent(4);
serializer.write(doc);
```

:

```
import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.UnsupportedEncodingException;
import nu.xom.Attribute;
import nu.xom.Builder;
import nu.xom.Document;
import nu.xom.Element;
import nu.xom.Elements;
import nu.xom.ParsingException;
import nu.xom.Serializer;

public class XMLWriter{

    public static void main(String[] args) throws UnsupportedEncodingException,
            IOException{
        // root element <example>
        Element root = new Element("example");

        // make a array of people to store
        Person[] people = {new Person("Smith", "Dan", "years", "green", 23),
            new Person("Autry", "Bob", "months", "N/A", 3)};

        // add all the people
        for (Person person : people){

            // make the main person element <person>
            Element personElement = new Element("person");

            // make the name element and it's children: first and last
            Element nameElement = new Element("name");
            Element firstNameElement = new Element("first");
            Element lastNameElement = new Element("last");

            // make age element
            Element ageElement = new Element("age");

            // make favorite color element
            Element favColorElement = new Element("fav_color");

            // add value to names
            firstNameElement.appendChild(person.getFirstName());
            lastNameElement.appendChild(person.getLastName());
```

```
            // add names to name
            nameElement.appendChild(firstNameElement);
            nameElement.appendChild(lastNameElement);

            // add value to age
            ageElement.appendChild(String.valueOf(person.getAge()));

            // add unit attribute to age
            ageElement.addAttribute(new Attribute("unit", person.getAgeUnit()));

            // add value to favColor
            favColorElement.appendChild(person.getFavoriteColor());

            // add all contents to person
            personElement.appendChild(nameElement);
            personElement.appendChild(ageElement);
            personElement.appendChild(favColorElement);

            // add person to root
            root.appendChild(personElement);
        }

        // create doc off of root
        Document doc = new Document(root);

        // the file it will be stored in
        File file = new File("out.xml");
        if (!file.exists()){
            file.createNewFile();
        }

        // get a file output stream ready
        FileOutputStream fileOutputStream = new FileOutputStream(file);

        // use the serializer class to write it all
        Serializer serializer = new Serializer(fileOutputStream, "UTF-8");
        serializer.setIndent(4);
        serializer.write(doc);
    }

    private static class Person {

        private String lName, fName, ageUnit, favColor;
        private int age;

        public Person(String lName, String fName, String ageUnit, String favColor, int age){
            this.lName = lName;
            this.fName = fName;
            this.age = age;
            this.ageUnit = ageUnit;
            this.favColor = favColor;
        }

        public String getLastName() { return lName; }
        public String getFirstName() { return fName; }
        public String getAgeUnit() { return ageUnit; }
        public String getFavoriteColor() { return favColor; }
        public int getAge() { return age; }
    }

}
```

"out.xml" .

```xml
<?xml version="1.0" encoding="UTF-8"?>
<example>
    <person>
        <name>
            <first>Dan</first>
            <last>Smith</last>
        </name>
        <age unit="years">23</age>
        <fav_color>green</fav_color>
    </person>
    <person>
        <name>
            <first>Bob</first>
            <last>Autry</last>
        </name>
        <age unit="months">3</age>
        <fav_color>N/A</fav_color>
    </person>
</example>

```

XOM - XML : https://riptutorial.com/ko/java/topic/5091/xom---xml----

65: ( )

- [=];
- [=];
- [=];
- [=];
- public class name {
- {

:

. .

- `public` ( ).
- - `public` , `private` , `protected` *package-private* ( ).

`public` . . (, *package-private*) .

`public` (*package-private*) . `private` `protected` . `private` . `protected` , (*package-private*) , () .

.

:

public				
protected				
private				

Examples

```
public interface MyInterface {
    public void foo();
    int bar();

    public String TEXT = "Hello";
    int ANSWER = 42;

    public class X {
    }

    class Y {
    }
}
```

`public` . `foo()` , `bar()` , `TEXT` , `ANSWER` , `X` `Y` . `MyInterface` `MyInterface` .

, .

Test .

```
public class Test{
    public int number = 2;

    public Test(){

    }
}
```

. public number .

```
public class Other{

    public static void main(String[] args){
        Test t = new Test();
        System.out.println(t.number);
    }

}
```

private visibility . public getter setters .

```
class SomeClass {
    private int variable;

    public int getVariable() {
        return variable;
    }

    public void setVariable(int variable) {
        this.variable = variable;
    }
}

public class SomeOtherClass {
    public static void main(String[] args) {
        SomeClass sc = new SomeClass();

        // These statement won't compile because SomeClass#variable is private:
        sc.variable = 7;
        System.out.println(sc.variable);

        // Instead, you should use the public getter and setter:
        sc.setVariable(7);
        System.out.println(sc.getVariable());
    }
}
```

. *Java Documentation* "[package visibility] . javax.swing ,

```
package javax.swing;
public abstract class JComponent extends Container … {
    …
```

```
    static boolean DEBUG_GRAPHICS_LOADED;
    …
}
```

DebugGraphics DEBUG_GRAPHICS_LOADED .

```
package javax.swing;
public class DebugGraphics extends Graphics {
    …
    static {
        JComponent.DEBUG_GRAPHICS_LOADED = true;
    }
    …
}
```

.

.

:

```
package com.stackexchange.docs;
public class MyClass{
    protected int variable; //This is the variable that we are trying to access
    public MyClass(){
        variable = 2;
    };
}
```

---

protected .

```
package some.other.pack;
import com.stackexchange.docs.MyClass;
public class SubClass extends MyClass{
    public SubClass(){
        super();
        System.out.println(super.variable);
    }
}
```

protected .

.

		.
/		.

private protected ( ) private. Java 1.0 .

( ) : https://riptutorial.com/ko/java/topic/134/-------

66:

. . (:) .

clone Cloneable , Cloneable clone . clone Object Cloneable , Cloneable clone . clone . , clone clone . , throw .

. Joshua Bloch Effective Java Item 11 : .

Examples

.

```
public class Sheep {

    private String name;

    private int weight;

    public Sheep(String name, int weight) {
        this.name = name;
        this.weight = weight;
    }

    // copy constructor
    // copies the fields of other into the new object
    public Sheep(Sheep other) {
        this.name = other.name;
        this.weight = other.weight;
    }

}

// create a sheep
Sheep sheep = new Sheep("Dolly", 20);
// clone the sheep
Sheep dolly = new Sheep(sheep); // dolly.name is "Dolly" and dolly.weight is 20
```

Clonable

Cloneable .

```
public class Sheep implements Cloneable {

    private String name;

    private int weight;

    public Sheep(String name, int weight) {
        this.name = name;
        this.weight = weight;
    }
```

```
    @Override
    public Object clone() throws CloneNotSupportedException {
        return super.clone();
    }

}

// create a sheep
Sheep sheep = new Sheep("Dolly", 20);
// clone the sheep
Sheep dolly =  (Sheep) sheep.clone(); // dolly.name is "Dolly" and dolly.weight is 20
```

. .

.

```
import java.util.List;

public class Sheep implements Cloneable {

    private String name;

    private int weight;

    private List<Sheep> children;

    public Sheep(String name, int weight) {
        this.name = name;
        this.weight = weight;
    }

    @Override
    public Object clone() throws CloneNotSupportedException {
        return super.clone();
    }

    public List<Sheep> getChildren() {
        return children;
    }

    public void setChildren(List<Sheep> children) {
        this.children = children;
    }

}

import java.util.Arrays;
import java.util.List;

// create a sheep
Sheep sheep = new Sheep("Dolly", 20);

// create children
Sheep child1 = new Sheep("Child1", 4);
Sheep child2 = new Sheep("Child2", 5);

sheep.setChildren(Arrays.asList(child1, child2));

// clone the sheep
```

```
Sheep dolly =  (Sheep) sheep.clone();
List<Sheep> sheepChildren = sheep.getChildren();
List<Sheep> dollysChildren = dolly.getChildren();
for (int i = 0; i < sheepChildren.size(); i++) {
    // prints true, both arrays contain the same objects
    System.out.println(sheepChildren.get(i) == dollysChildren.get(i));
}
```

.

```
import java.util.ArrayList;
import java.util.List;

public class Sheep implements Cloneable {

    private String name;

    private int weight;

    private List<Sheep> children;

    public Sheep(String name, int weight) {
        this.name = name;
        this.weight = weight;
    }

    @Override
    public Object clone() throws CloneNotSupportedException {
        Sheep clone = (Sheep) super.clone();
        if (children != null) {
            // make a deep copy of the children
            List<Sheep> cloneChildren = new ArrayList<>(children.size());
            for (Sheep child : children) {
                cloneChildren.add((Sheep) child.clone());
            }
            clone.setChildren(cloneChildren);
        }
        return clone;
    }

    public List<Sheep> getChildren() {
        return children;
    }

    public void setChildren(List<Sheep> children) {
        this.children = children;
    }

}

import java.util.Arrays;
import java.util.List;

// create a sheep
Sheep sheep = new Sheep("Dolly", 20);

// create children
Sheep child1 = new Sheep("Child1", 4);
Sheep child2 = new Sheep("Child2", 5);
```

```
sheep.setChildren(Arrays.asList(child1, child2));

// clone the sheep
Sheep dolly =  (Sheep) sheep.clone();
List<Sheep> sheepChildren = sheep.getChildren();
List<Sheep> dollysChildren = dolly.getChildren();
for (int i = 0; i < sheepChildren.size(); i++) {
    // prints false, both arrays contain copies of the objects inside
    System.out.println(sheepChildren.get(i) == dollysChildren.get(i));
}
```

```
public class Sheep {

    private String name;

    private int weight;

    public Sheep(String name, int weight) {
        this.name = name;
        this.weight = weight;
    }

    public static Sheep newInstance(Sheep other);
        return new Sheep(other.name, other.weight)
    }

}
```

: https://riptutorial.com/ko/java/topic/2830/-

67:

"Null Pointer Exception" . null . , ...

Examples

. . , , , "on" .

```
public class Person {

  private String name;

  public void setName(String name) { this.name = name; }

  public String getName() { return name; }

  public static void main(String [] arguments) {
    Person person = new Person();
    person.setName("Bob");

    int i = 5;
    setPersonName(person, i);

    System.out.println(person.getName() + " " + i);
  }

  private static void setPersonName(Person person, int num) {
    person.setName("Linda");
    num = 99;
  }
}
```

Java . Java .

, `person` `main` , `name` `"Bob"` . `person` . 5 .

`name` "Linda" 99 .

?

```
Linda 5
```

`person` `main` ? .

main `person` `setPersonName` . `setAnotherName` .

: `person` ( ). " " . .

. int (Integer) " ". , . .

Java . . , .

. f} . , .

.

```
private static void getAnotherObjectNot(Person person) {
  person = new Person();
  person.setName("George");
}
```

`main` `setAnotherName` println .

```
getAnotherObjectNot(person);
System.out.println(person.getName());
```

:

```
Linda 5
Linda
```

? , Linda . `getAnotherObjectNot` `getAnotherObjectNot` Linda George . Linda () , `main` `getAnotherObjectNot` . `getAnotherObjectNot` . .

, .

```
private static Person getAnotherObject() {
  Person person = new Person();
  person.setName("Mary");
  return person;
}
```

.

```
Person mary;
mary = getAnotherObject();
System.out.println(mary.getName());
```

.

```
Linda 5
Linda
Mary
```

.

```
public class Person {
  private String name;

  public void setName(String name) { this.name = name; }
  public String getName() { return name; }

  public static void main(String [] arguments) {
    Person person = new Person();
```

```
    person.setName("Bob");

    int i = 5;
    setPersonName(person, i);
    System.out.println(person.getName() + " " + i);

    getAnotherObjectNot(person);
    System.out.println(person.getName());

    Person person;
    person = getAnotherObject();
    System.out.println(person.getName());
  }

  private static void setPersonName(Person person, int num) {
    person.setName("Linda");
    num = 99;
  }

  private static void getAnotherObjectNot(Person person) {
    person = new Person();
    person.setMyName("George");
  }

  private static person getAnotherObject() {
    Person person = new Person();
    person.setMyName("Mary");
    return person;
  }
}
```

: https://riptutorial.com/ko/java/topic/5454/-

68:

Object .

- final <?> getClass ()
- public notify ()
- public notifyAll ()
- public final native void wait (long timeout) throws InterruptedException
- public final void wait () throws InterruptedException
- public final void wait (long timeout, int nanos) InterruptedException .
- int hashCode ()
- public boolean equals (Object obj)
- public String toString ()
- protected native Object clone () CloneNotSupportedException .
- protected void finalize () Throwable .

Examples

toString ()

toString() String . .toString() "hello " + anObject .

:

```
public class User {
    private String firstName;
    private String lastName;

    public User(String firstName, String lastName) {
        this.firstName = firstName;
        this.lastName = lastName;
    }

    @Override
    public String toString() {
        return firstName + " " + lastName;
    }

    public static void main(String[] args) {
        User user = new User("John", "Doe");
        System.out.println(user.toString()); // Prints "John Doe"
    }
}
```

Object toString() User .

println() toString() . .

```
System.out.println(user); // toString() is implicitly called on `user`
System.out.println(user.toString());
```

User toString() System.out.println(user) User@659e0bfd String . User Java Object toString() . .

equals ()

TL, DR

== ( )

.equals() ("").

---

equals() equals() . Object equals() true true . == .

```
public class Foo {
    int field1, field2;
    String field3;

    public Foo(int i, int j, String k) {
        field1 = i;
        field2 = j;
        field3 = k;
    }

    public static void main(String[] args) {
        Foo foo1 = new Foo(0, 0, "bar");
        Foo foo2 = new Foo(0, 0, "bar");

        System.out.println(foo1.equals(foo2)); // prints false
    }
}
```

foo1 foo2 . equals() false .

, equals() (override equals() .

```
public class Foo {
    int field1, field2;
    String field3;

    public Foo(int i, int j, String k) {
        field1 = i;
        field2 = j;
        field3 = k;
    }

    @Override
    public boolean equals(Object obj) {
        if (this == obj) {
            return true;
        }
        if (obj == null || getClass() != obj.getClass()) {
            return false;
        }

        Foo f = (Foo) obj;
```

```
        return field1 == f.field1 &&
               field2 == f.field2 &&
               (field3 == null ? f.field3 == null : field3.equals(f.field3));
    }

    @Override
    public int hashCode() {
        int hash = 1;
        hash = 31 * hash + this.field1;
        hash = 31 * hash + this.field2;
        hash = 31 * hash + (field3 == null ? 0 : field3.hashCode());
        return hash;
    }

    public static void main(String[] args) {
        Foo foo1 = new Foo(0, 0, "bar");
        Foo foo2 = new Foo(0, 0, "bar");

        System.out.println(foo1.equals(foo2)); // prints true
    }
}
```

`equals()` .

`hashCode()` . . `hashCode()` `equals()` .

`equals` . `Foo obj` `Object obj` `Foo obj` . `equals` .

`equals()` `hashCode()` . IDE .

`equals()` Java API `String` . `String` `String` .

Java SE 7

Java 1.7, `equals` `java.util.Objects` , `null` 2 , `equals` .

```
@Override
public boolean equals(Object obj) {
    if (this == obj) {
        return true;
    }
    if (obj == null || getClass() != obj.getClass()) {
        return false;
    }

    Foo f = (Foo) obj;
    return field1 == f.field1 && field2 == f.field2 && Objects.equals(field3, f.field3);
}
```

equals (`null`) .

```
@Override
public boolean equals(Object obj) {
    //...check for null
    if (getClass() != obj.getClass()) {
        return false;
```

```
    }
    //...compare fields
}
```

. . . JPA .

```
Foo detachedInstance = ...
Foo mergedInstance = entityManager.merge(detachedInstance);
if (mergedInstance.equals(detachedInstance)) {
    //Can never get here if equality is tested with getClass()
    //as mergedInstance is a proxy (subclass) of Foo
}
```

`instanceof`

```
@Override
public final boolean equals(Object obj) {
    if (!(obj instanceof Foo)) {
        return false;
    }
    //...compare fields
}
```

`instanceof` . Foo `equals()` `Foo` `FooSubclass` `FooSubclass` `Foo` .

```
Foo foo = new Foo(7);
FooSubclass fooSubclass = new FooSubclass(7, false);
foo.equals(fooSubclass) //true
fooSubclass.equals(foo) //false
```

`equals()` . `instanceof` `equals()` `final` . , `equals()` `equals()` override) , .

hashCode ()

Java `equals` (override) , `hashCode` (override) . :

- Java 2 `hashCode` , equals , . .
- 2 `equals(Object)` `equals(Object)` , 2 `hashCode` `equals(Object)` , .
- `equals(Object)` 2 , 2 `hashCode` `equals(Object)` , . .

`HashMap` , `HashTable` `HashSet` . `hashCode` . `hashCode` . `hashCode` `hashCode` . , .

.

```
public class Foo {
    private int field1, field2;
    private String field3;

    public Foo(int field1, int field2, String field3) {
        this.field1 = field1;
        this.field2 = field2;
        this.field3 = field3;
    }
```

```
    @Override
    public boolean equals(Object obj) {
        if (this == obj) {
            return true;
        }
        if (obj == null || getClass() != obj.getClass()) {
            return false;
        }

        Foo f = (Foo) obj;
        return field1 == f.field1 &&
               field2 == f.field2 &&
               (field3 == null ? f.field3 == null : field3.equals(f.field3);
    }

    @Override
    public int hashCode() {
        int hash = 1;
        hash = 31 * hash + field1;
        hash = 31 * hash + field2;
        hash = 31 * hash + (field3 == null ? 0 : field3.hashCode());
        return hash;
    }
}
```

Arrays.hashCode ()

Java SE 1.2

Java 1.2 , , Object `java.util.Arrays#hashCode` .

```
@Override
public int hashCode() {
    return Arrays.hashCode(new Object[] {field1, field2, field3});
}
```

Java SE 7

Java 1.7 `hash(Object... objects)` `hash(Object... objects)` `java.util.Objects` . `java.util.Arrays#hashCode` .

```
@Override
public int hashCode() {
    return Objects.hash(field1, field2, field3);
}
```

: `hashCode()` .

- `Object[]` . `Objects.hash()` "varargs" .
- .
- .
- `Arrays.hashCode` `Objects.hash` .
- `Arrays.hashCode` `Objects.hash` () `Objects.hash` `Object.hashCode()` .

.

```
public final class ImmutableArray {
    private int[] array;
    private volatile int hash = 0;

    public ImmutableArray(int[] initial) {
        array = initial.clone();
    }

    // Other methods

    @Override
    public boolean equals(Object obj) {
         // ...
    }

    @Override
    public int hashCode() {
        int h = hash;
        if (h == 0) {
            h = Arrays.hashCode(array);
            hash = h;
        }
        return h;
    }
}
```

( ) . (look up) .

`ImmutableArray` 0 (1 2 ^{32}) .

, . . .

wait () notify ()

`wait()` `notify()` . `wait()` `notify()` `notifyAll()` .

(: wait () / notify ())

```
package com.example.examples.object;

import java.util.concurrent.atomic.AtomicBoolean;

public class WaitAndNotify {

    public static void main(String[] args) throws InterruptedException {
        final Object obj = new Object();
        AtomicBoolean aHasFinishedWaiting = new AtomicBoolean(false);

        Thread threadA = new Thread("Thread A") {
            public void run() {
                System.out.println("A1: Could print before or after B1");
                System.out.println("A2: Thread A is about to start waiting...");
                try {
                    synchronized (obj) { // wait() must be in a synchronized block
                        // execution of thread A stops until obj.notify() is called
```

```
                        obj.wait();
                    }
                    System.out.println("A3: Thread A has finished waiting. "
                            + "Guaranteed to happen after B3");
                } catch (InterruptedException e) {
                    System.out.println("Thread A was interrupted while waiting");
                } finally {
                    aHasFinishedWaiting.set(true);
                }
            }
        };

        Thread threadB = new Thread("Thread B") {
            public void run() {
                System.out.println("B1: Could print before or after A1");

                System.out.println("B2: Thread B is about to wait for 10 seconds");
                for (int i = 0; i < 10; i++) {
                    try {
                        Thread.sleep(1000); // sleep for 1 second
                    } catch (InterruptedException e) {
                        System.err.println("Thread B was interrupted from waiting");
                    }
                }

                System.out.println("B3: Will ALWAYS print before A3 since "
                        + "A3 can only happen after obj.notify() is called.");

                while (!aHasFinishedWaiting.get()) {
                    synchronized (obj) {
                        // notify ONE thread which has called obj.wait()
                        obj.notify();
                    }
                }
            }
        };

        threadA.start();
        threadB.start();

        threadA.join();
        threadB.join();

        System.out.println("Finished!");
    }
}
```

:

```
A1: Could print before or after B1
B1: Could print before or after A1
A2: Thread A is about to start waiting...
B2: Thread B is about to wait for 10 seconds
B3: Will ALWAYS print before A3 since A3 can only happen after obj.notify() is called.
A3: Thread A has finished waiting. Guaranteed to happen after B3
Finished!

B1: Could print before or after A1
B2: Thread B is about to wait for 10 seconds
A1: Could print before or after B1
```

```
A2: Thread A is about to start waiting...
B3: Will ALWAYS print before A3 since A3 can only happen after obj.notify() is called.
A3: Thread A has finished waiting. Guaranteed to happen after B3
Finished!

A1: Could print before or after B1
A2: Thread A is about to start waiting...
B1: Could print before or after A1
B2: Thread B is about to wait for 10 seconds
B3: Will ALWAYS print before A3 since A3 can only happen after obj.notify() is called.
A3: Thread A has finished waiting. Guaranteed to happen after B3
Finished!
```

getClass ()

`getClass()` . .

```
public class User {

    private long userID;
    private String name;

    public User(long userID, String name) {
        this.userID = userID;
        this.name = name;
    }
}

public class SpecificUser extends User {
    private String specificUserID;

    public SpecificUser(String specificUserID, long userID, String name) {
        super(userID, name);
        this.specificUserID = specificUserID;
    }
}

public static void main(String[] args){
    User user = new User(879745, "John");
    SpecificUser specificUser = new SpecificUser("1AAAA", 877777, "Jim");
    User anotherSpecificUser = new SpecificUser("1BBBB", 812345, "Jenny");

    System.out.println(user.getClass()); //Prints "class User"
    System.out.println(specificUser.getClass()); //Prints "class SpecificUser"
    System.out.println(anotherSpecificUser.getClass()); //Prints "class SpecificUser"
}
```

`getClass()` anotherSpecificUser `getClass()` User class SpecificUser .

`getClass` .

```
public final native Class<?> getClass();
```

`getClass` static `Class<? extends T> T getClass` .

.

```
Class<? extends String> cls = "".getClass();
```

clone ()

clone() . clone() .

Cloneable .

Cloneable Cloneable native clone() . CloneNotSupportedException
CloneNotSupportedException .

Object Object CloneNotSupportedException .

. "" . . .

```
class Foo implements Cloneable {
    int w;
    String x;
    float[] y;
    Date z;

    public Foo clone() {
        try {
            Foo result = new Foo();
            // copy primitives by value
            result.w = this.w;
            // immutable objects like String can be copied by reference
            result.x = this.x;

            // The fields y and z refer to a mutable objects; clone them recursively.
            if (this.y != null) {
              result.y = this.y.clone();
            }
            if (this.z != null) {
              result.z = this.z.clone();
            }

            // Done, return the new object
            return result;

        } catch (CloneNotSupportedException e) {
            // in case any of the cloned mutable fields do not implement Cloneable
            throw new AssertionError(e);
        }
    }
}
```

finalize ()

Object . .

doc , , .

finalize() . .

Java FileInputStream.java .

```
protected void finalize() throws IOException {
    if ((fd != null) &&  (fd != FileDescriptor.in)) {
        /* if fd is shared, the references in FileDescriptor
         * will ensure that finalizer is only called when
         * safe to do so. All references using the fd have
         * become unreachable. We can call close()
         */
        close();
    }
}
```

.

finalize() .

. . .

. java.lang.ref.WeakReference<T> . finalize() WeakReference . .

finalize() " ".

Java Object . super() . . ..

super() .

super()

```
public class MyClass {

    public MyClass() {
        super();
    }
}
```

super()

```
public class MyClass {

    public MyClass() {
        // empty
    }
}
```

```
public class MyClass {

}
```

- ?

. .

```
public class MyClass {

    public MyClass(int size) {

        doSomethingWith(size);

    }

    public MyClass(Collection<?> initialValues) {

        this(initialValues.size());
        addInitialValues(initialValues);
    }
}
```

`MyClass(Arrays.asList("a", "b", "c"))` List (, `super()`) `addInitialValues(int size)` .
.

?

`new MyClass("argument")` `new MyClass("argument", 0)` ., , .

Object ?

(`super()`) .

.

Object ?

JVM . Java `<init>` . `<init>` `<init>` Java .

JVM `<init>` ?

JVM `invokespecial` `<init>` .

JVM Java .

- (JVM) - JVMS - 2.9
- - JLS - 8.8

: https://riptutorial.com/ko/java/topic/145/---

69:

getter setter . Java .

Examples

OOP . . `private` Getters Setters .

```
public class Sample {
  private String  name;
  private int age;

  public int getAge() {
    return age;
  }

  public void setAge(int age) {
    this.age = age;
  }

  public String getName() {
    return name;
  }

  public void setName(String name) {
    this.name = name;
  }
}
```

. . Getters Setters .

`getXxx()` `xxx` , `xxx` `setXxx()` .

(`variableName`)

- `boolean`

```
getVariableName()   //Getter, The variable name should start with uppercase
setVariableName(..) //Setter, The variable name should start with uppercase
```

- `boolean`

```
 isVariableName()     //Getter, The variable name should start with uppercase
 setVariableName(...) //Setter, The variable name should start with uppercase
```

Public Getters Setters Java Bean .

setter getter

Setters Getters private . ,

```
public class Person {

    private String name;

    public String getName() {
        return name;
    }

    public void setName(String name) {
        if(name!=null && name.length()>2)
           this.name = name;
    }
}
```

`Person` `name` . `getName()` `setName(String)` 2 null . `name` public setter `name` . getter .

```
public String getName(){
   if(name.length()>16)
      return "Name is too large!";
   else
      return name;
}
```

`getName()` , 16 `name` . , `"Name is too large"` . .

?

Java getter setter .

```
public class CountHolder {
  private int count = 0;

  public int getCount() { return count; }
  public void setCount(int c) { count = c; }
}
```

`count` . `getCount()` `setCount(int)` . . ? ?

```
public class CountHolder {
  public int count = 0;
}
```

. . .

- , " `int` , `int` ".
- : " `int` `int` ."

, ... API (API).

. .

```
public class CountHolder {
```

```
  private int count = 0;

  public synchronized int getCount() { return count; }
  public synchronized void setCount(int c) { count = c; }
}
```

`count` . , .

. (, ?)

. (IE : `System.out` `out`). . .

1. . ( ), getters / setter . / / API .
2. . . 99.9 % .

. getters / setter .

: https://riptutorial.com/ko/java/topic/3560/-

70:

`extends` . `implements` .

- ClassB ClassA . {...}
- ClassB InterfaceA {...} .
- InterfaceB InterfaceA {...} .
- class ClassB extends ClassA, InterfaceC, InterfaceD {...} .
- abstract class AbstractClassB ClassA . {...}
- abstract class AbstractClassB AbstractClassA . {...}
- abstract class AbstractClassB extends ClassA, InterfaceC, InterfaceD {...} .

. .

Examples

`abstract` . . .

. () .

:

```
public abstract class Component {
    private int x, y;

    public setPosition(int x, int y) {
        this.x = x;
        this.y = y;
    }

    public abstract void render();
}
```

. . .

.

```
//error: Component is abstract; cannot be instantiated
Component myComponent = new Component();
```

`Component` .

```
public class Button extends Component {

    @Override
    public void render() {
        //render a button
    }
}
```

```
public class TextBox extends Component {

    @Override
    public void render() {
        //render a textbox
    }
}
```

( ) .

```
Component myButton = new Button();
Component myTextBox = new TextBox();

myButton.render(); //renders a button
myTextBox.render(); //renders a text box
```

abstract / .

.

- .
- (`static`) `static` .

Java SE 8

, .

Java SE 8

Java 8 .

java . .

```
Component myAnonymousComponent = new Component() {
    @Override
    public void render() {
        // render a quick 1-time use component
    }
}
```

" " . , .

```
public class BaseClass {

    public static int num = 5;

    public static void sayHello() {
        System.out.println("Hello");
    }

    public static void main(String[] args) {
        BaseClass.sayHello();
        System.out.println("BaseClass's num: " + BaseClass.num);
```

```
        SubClass.sayHello();
        //This will be different than the above statement's output, since it runs
        //A different method
        SubClass.sayHello(true);

        StaticOverride.sayHello();
        System.out.println("StaticOverride's num: " + StaticOverride.num);
    }
}

public  class SubClass extends BaseClass {

    //Inherits the sayHello function, but does not override it
    public static void sayHello(boolean test) {
        System.out.println("Hey");
    }
}

public static class StaticOverride extends BaseClass {

    //Hides the num field from BaseClass
    //You can even change the type, since this doesn't affect the signature
    public static String num = "test";

    //Cannot use @Override annotation, since this is static
    //This overrides the sayHello method from BaseClass
    public static void sayHello() {
        System.out.println("Static says Hi");
    }

}
```

.

```
Hello
BaseClass's num: 5
Hello
Hey
Static says Hi
StaticOverride's num: test
```

, . BaseClass.sayHello() sayHello . . 2 , .

.

'final'

class final extend . final "".

```
// This declares a final class
final class MyFinalClass {
    /* some code */
}

// Compilation error: cannot inherit from final MyFinalClass
class MySubClass extends MyFinalClass {
    /* more code */
```

```
}
```

`private` . " " . .

```
public final class UtilityClass {

    // Private constructor to replace the default visible constructor
    private UtilityClass() {}

    // Static members can still be used as usual
    public static int doSomethingCool() {
        return 123;
    }

}
```

`final` . (, , ⌜ ⌟ .) (), . " " Liskov Substitution Principle .

`final` . . , `String` `final` .

`final` Mockito (mocking) . : Mockito 2 .

`final` , (override) .

```
public class MyClassWithFinalMethod {

    public final void someMethod() {
    }
}

public class MySubClass extends MyClassWithFinalMethod {

    @Override
    public void someMethod() { // Compiler error (overridden method is final)
    }
}
```

.

`final` `final` .

Liskov

Substitutability 1987 Barbara Liskov , `B` `A` `A` `B` .

```
class A {...}
class B extends A {...}

public void method(A obj) {...}

A a = new B(); // Assignment OK
method(new B()); // Passing as parameter OK
```

. .

```
interface Foo {
    void bar();
}

class A implements Foo {
    void bar() {...}
}

class B implements Foo {
    void bar() {...}
}

List<Foo> foos = new ArrayList<>();
foos.add(new A()); // OK
foos.add(new B()); // OK
```

.

extends ( ) ( ).

```
public class BaseClass {

    public void baseMethod(){
        System.out.println("Doing base class stuff");
    }
}

public class SubClass extends BaseClass {

}
```

SubClass baseMethod() baseMethod() .

```
SubClass s = new SubClass();
s.baseMethod();  //Valid, prints "Doing base class stuff"
```

. .

```
public class Subclass2 extends BaseClass {

    public void anotherMethod() {
        System.out.println("Doing subclass2 stuff");
    }
}

Subclass2 s2 = new Subclass2();
s2.baseMethod(); //Still valid , prints "Doing base class stuff"
s2.anotherMethod(); //Also valid, prints "Doing subclass2 stuff"
```

.

```
public class BaseClassWithField {
```

```
    public int x;

}

public class SubClassWithField extends BaseClassWithField {

    public SubClassWithField(int x) {
        this.x = x; //Can access fields
    }
}
```

private .

```
public class BaseClassWithPrivateField {

    private int x = 5;

    public int getX() {
        return x;
    }
}

public class SubClassInheritsPrivateField extends BaseClassWithPrivateField {

    public void printX() {
        System.out.println(x); //Illegal, can't access private field x
        System.out.println(getX()); //Legal, prints 5
    }
}

SubClassInheritsPrivateField s = new SubClassInheritsPrivateField();
int x = s.getX(); //x will have a value of 5.
```

Java .

```
public class A{}
public class B{}
public class ExtendsTwoClasses extends A, B {} //Illegal
```

(multiple inheritance) Java .

, Object .

. . :

```
class StaticMethodTest {

  // static method and inheritance
  public static void main(String[] args) {
    Parent p = new Child();
    p.staticMethod(); // prints Inside Parent
    ((Child) p).staticMethod(); // prints Inside Child
  }

  static class Parent {
    public static void staticMethod() {
```

```
      System.out.println("Inside Parent");
    }
  }

  static class Child extends Parent {
    public static void staticMethod() {
      System.out.println("Inside Child");
    }
  }
}
```

. staticMethod() p staticMethod() Parent staticMethod() . p Child , Child staticMethod() .

SHADOWED OVERRIDDEN. . .

```
class Car {
    public int gearRatio = 8;

    public String accelerate() {
        return "Accelerate : Car";
    }
}

class SportsCar extends Car {
    public int gearRatio = 9;

    public String accelerate() {
        return "Accelerate : SportsCar";
    }

    public void test() {

    }


    public static void main(String[] args) {

        Car car = new SportsCar();
        System.out.println(car.gearRatio + "  " + car.accelerate());
        // will print out 8  Accelerate : SportsCar
    }
}
```

. b = (B) a; ( ) *narrowing* .

. A a = b; .

.

```
class Vehicle {
}

class Car extends Vehicle {
}

class Truck extends Vehicle {
}
```

```
class MotorCycle extends Vehicle {
}

class Test {

    public static void main(String[] args) {

        Vehicle vehicle = new Car();
        Car car = new Car();

        vehicle = car; // is valid, no cast needed

        Car c = vehicle // not valid
        Car c = (Car) vehicle; //valid
    }
}
```

`Vehicle vehicle = new Car();` Java . `Car` `Vehicle` . .

, `Car c = vehicle;` . `vehicle` `Vehicle` `Car` , , , or any other current or future subclass of . (Or indeed, an instance of itself, since we did not declare it as an class.) The assignment cannot be allowed, since that might lead to itself, since we did not declare it as an . (Or indeed, an instance of Vehicle itself, since we did not declare it as an . (Or indeed, an instance of class.) The assignment cannot be allowed, since that might lead to . referring to a Truck referring to a class.) The assignment cannot be allowed, since that might lead to .

.

```
Car c = (Car) vehicle;
```

`vehicle` `Car` `Car` . . (), () `ClassCastException` Throw.

. :

```
String s = (String) vehicle;  // not valid
```

`Vehicle` `String` . JLS .

. ( ) :

```
public <T> Set<T> toSet(Collection<T> collection) {
  return Sets.newHashSet(collection);
}
```

.

```
public <T> HashSet<T> toSet(ArrayList<T> collection) {
  return Sets.newHashSet(collection);
}
```

. .

- .
- .
- .
- ( ).

? .

-
-
-

RPC .

```
public interface RemoteInvoker {
  <RQ, RS> CompletableFuture<RS> invoke(RQ request, Class<RS> responseClass);
}
```

. HTTP AMQP . . :

```
public interface AmqpInvoker extends RemoteInvoker {
  static AmqpInvokerBuilder with(String instanceId, ConnectionFactory factory) {
    return new AmqpInvokerBuilder(instanceId, factory);
  }
}
```

AMQP `RemoteInvoker` ( ) .

```
RemoteInvoker invoker = AmqpInvoker.with(instanceId, factory)
  .requestRouter(router)
  .build();
```

.

```
Response res = invoker.invoke(new Request(data), Response.class).get();
```

Java 8 `AmqpInvoker.with()` `AmqpInvoker.with()` . 8 Java `Factory` .

```
public interface AmqpInvoker extends RemoteInvoker {
  class Factory {
    public static AmqpInvokerBuilder with(String instanceId, ConnectionFactory factory) {
      return new AmqpInvokerBuilder(instanceId, factory);
    }
  }
}
```

.

```
RemoteInvoker invoker = AmqpInvoker.Factory.with(instanceId, factory)
  .requestRouter(router)
  .build();
```

(15). public `AmqpInvoker` .

```
public class AmqpInvokerBuilder {
  ...
  AmqpInvokerBuilder(String instanceId, ConnectionFactory factory) {
    this.instanceId = instanceId;
    this.factory = factory;
  }

  public AmqpInvokerBuilder requestRouter(RequestRouter requestRouter) {
    this.requestRouter = requestRouter;
    return this;
  }

  public AmqpInvoker build() throws TimeoutException, IOException {
    return new AmqpInvokerImpl(instanceId, factory, requestRouter);
  }
}
```

FreeBuilder .

, ( ) , .

```
class AmqpInvokerImpl implements AmqpInvoker {
  AmqpInvokerImpl(String instanceId, ConnectionFactory factory, RequestRouter requestRouter) {
    ...
  }

  @Override
  public <RQ, RS> CompletableFuture<RS> invoke(final RQ request, final Class<RS> respClass) {
    ...
  }
}
```

.

: "Is-a" vs "Has-a"

:

:

"" "".

.

```
public class InterfaceAndAbstractClassDemo{
    public static void main(String args[]){

        Dog dog = new Dog("Jack",16);
        Cat cat = new Cat("Joe",20);

        System.out.println("Dog:"+dog);
        System.out.println("Cat:"+cat);
```

```
        dog.remember();
        dog.protectOwner();
        Learn dl = dog;
        dl.learn();

        cat.remember();
        cat.protectOwner();

        Climb c = cat;
        c.climb();

        Man man = new Man("Ravindra",40);
        System.out.println(man);

        Climb cm = man;
        cm.climb();
        Think t = man;
        t.think();
        Learn l = man;
        l.learn();
        Apply a = man;
        a.apply();
    }
}

abstract class Animal{
    String name;
    int lifeExpentency;
    public Animal(String name,int lifeExpentency ){
        this.name = name;
        this.lifeExpentency=lifeExpentency;
    }
    public abstract void remember();
    public abstract void protectOwner();

    public String toString(){
        return this.getClass().getSimpleName()+":"+name+":"+lifeExpentency;
    }
}
class Dog extends Animal implements Learn{

    public Dog(String name,int age){
        super(name,age);
    }
    public void remember(){
        System.out.println(this.getClass().getSimpleName()+" can remember for 5 minutes");
    }
    public void protectOwner(){
        System.out.println(this.getClass().getSimpleName()+ " will protect owner");
    }
    public void learn(){
        System.out.println(this.getClass().getSimpleName()+ " can learn:");
    }
}
class Cat extends Animal implements Climb {
    public Cat(String name,int age){
        super(name,age);
    }
    public void remember(){
        System.out.println(this.getClass().getSimpleName() + " can remember for 16 hours");
    }
```

```
    public void protectOwner(){
        System.out.println(this.getClass().getSimpleName()+ " won't protect owner");
    }
    public void climb(){
        System.out.println(this.getClass().getSimpleName()+ " can climb");
    }
}
interface Climb{
    void climb();
}
interface Think {
    void think();
}

interface Learn {
    void learn();
}
interface Apply{
    void apply();
}

class Man implements Think,Learn,Apply,Climb{
    String name;
    int age;

    public Man(String name,int age){
        this.name = name;
        this.age = age;
    }
    public void think(){
        System.out.println("I can think:"+this.getClass().getSimpleName());
    }
    public void learn(){
        System.out.println("I can learn:"+this.getClass().getSimpleName());
    }
    public void apply(){
        System.out.println("I can apply:"+this.getClass().getSimpleName());
    }
    public void climb(){
        System.out.println("I can climb:"+this.getClass().getSimpleName());
    }
    public String toString(){
        return "Man :"+name+":Age:"+age;
    }
}
```

:

```
Dog:Dog:Jack:16
Cat:Cat:Joe:20
Dog can remember for 5 minutes
Dog will protect owner
Dog can learn:
Cat can remember for 16 hours
Cat won't protect owner
Cat can climb
Man :Ravindra:Age:40
I can climb:Man
I can think:Man
I can learn:Man
```

```
I can apply:Man
```

:

1. `Animal` `name` `lifeExpectancy` abstract : `remember()` `protectOwner()` . `Dog` `Cat` `remember()` `protectOwner()` `Animals` .
2. `Cat` `climb()` `Dog` . `Dog` `think()` `Cat` . `Cat` `Dog` .
3. `Man` `Animal` `Think` `Learn` `Apply` `Climb` .
4. `Cat` `Man` `Climb` .
5. `Dog` `Man` `Learn`
6. `Man` `Cat` `Dog` `Animal` , `Cat` `Dog` . .
7. `Animal` .

TL; DR :

.

Java .

.

1. .
2. public (: protected private) .
3. .

.

1. . , `Serializable` .
2. .
3. .

(override) . (Overriding) .

, `ClassB` , `ClassA` (override) .

:

```
public static void main(String[] args) {
    ClassA a = new ClassA();
    ClassA b = new ClassB();
    a.printing();
    b.printing();
}

class ClassA {
```

```
    public void printing() {
        System.out.println("A");
    }
}

class ClassB extends ClassA {
    public void printing() {
         System.out.println("B");
    }
}
```

:

: https://riptutorial.com/ko/java/topic/87/

71:

Java 8+ (Object) . JLS §9.8 . .

Examples

()		
()		
()		
()	int	IntSupplier
()		LongSupplier
()		DoubleSupplier
()		<T>
()		UnaryOperator <T>
()		<T, R>
()		<T>
()	int	ToIntFunction <T>
()		ToLongFunction <T>
()		ToDoubleFunction <T>
(T, T)		BinaryOperator <T>
(T, U)		BiConsumer <T, U>
(T, U)		BiFunction <T, U, R>
(T, U)		BiPredicate <T, U>
(T, U)	int	ToIntBiFunction <T, U>
(T, U)		ToLongBiFunction <T, U>
(T, U)		ToDoubleBiFunction <T, U>
(T, int)		ObjIntConsumer <T>

(T,)		ObjLongConsumer <T>
(T, double)		ObjDoubleConsumer <T>
(int)		IntConsumer
(int)		IntFunction <R>
(int)		IntPredicate
(int)	int	IntUnaryOperator
(int)		IntToLongFunction
(int)		IntToDoubleFunction
(int, int)	int	IntBinaryOperator
()		LongConsumer
()		LongFunction <R>
()		LongPredicate
()	int	LongToIntFunction
()		LongUnaryOperator
()		LongToDoubleFunction
(,)		LongBinaryOperator
()		DoubleConsumer
()		DoubleFunction <R>
()		
()	int	DoubleToIntFunction
()		DoubleToLongFunction
()		DoubleUnaryOperator
(,)		DoubleBinaryOperator

: https://riptutorial.com/ko/java/topic/10001/-

72:

Java 8 **Default Method** . , .

- methodName () {/ * * /}

- .
- .
- java.lang.Object (override) .
- .
- . .

- .
- ( ) . .
- java.lang.Object (override) ( ).

.

-	SUPER_CLASS-INSTANCE-METHOD	SUPER_CLASS-STATIC-METHOD
SUB_CLASS-INSTANCE-METHOD		-
SUB_CLASS-STATIC-METHOD	-	

.

-	-	-
IMPL_CLASS-INSTANCE-METHOD		
IMPL_CLASS-STATIC-METHOD	-	

:

- http://www.journaldev.com/2752/java-8-interface-changes-static-method-default-method
- https://docs.oracle.com/javase/tutorial/java/IandI/override.html

Examples

```
/**
 * Interface with default method
 */
public interface Printable {
    default void printString() {
        System.out.println( "default implementation" );
    }
}

/**
 * Class which falls back to default implementation of {@link #printString()}
 */
public class WithDefault
    implements Printable
{
}

/**
 * Custom implementation of {@link #printString()}
 */
public class OverrideDefault
    implements Printable {
    @Override
    public void printString() {
        System.out.println( "overridden implementation" );
    }
}
```

```
    new WithDefault().printString();
    new OverrideDefault().printString();
```

.

```
default implementation
overridden implementation
```

.

```
public interface Summable {
    int getA();

    int getB();

    default int calculateSum() {
        return getA() + getB();
    }
}

public class Sum implements Summable {
    @Override
    public int getA() {
```

```
        return 1;
    }

    @Override
    public int getB() {
        return 2;
    }
}
```

3 .

```
System.out.println(new Sum().calculateSum());
```

.

```
public interface Summable {
    static int getA() {
        return 1;
    }

    static int getB() {
        return 2;
    }

    default int calculateSum() {
        return getA() + getB();
    }
}

public class Sum implements Summable {}
```

3 .

```
System.out.println(new Sum().calculateSum());
```

`super.foo()` . super super `Fooable.super.foo()` .

```
public interface Fooable {
    default int foo() {return 3;}
}

public class A extends Object implements Fooable {
    @Override
    public int foo() {
        //return super.foo() + 1; //error: no method foo() in java.lang.Object
        return Fooable.super.foo() + 1; //okay, returns 4
    }
}
```

?

.

, 20 `Swim` .

```
public interface Swim {
    void backStroke();
}
```

, . .

```
public class FooSwimmer implements Swim {
    public void backStroke() {
         System.out.println("Do backstroke");
    }
}
```

20 .

Java 8 Default .

Swim .

```
public interface Swim {
    void backStroke();
    default void sideStroke() {
        System.out.println("Default sidestroke implementation. Can be overridden");
    }
}
```

. .

. Iterable Iterable `foreach` . Iterable .

,

.

- .

```
public interface Swim {
    default void backStroke() {
        System.out.println("Swim.backStroke");
    }
}

public abstract class AbstractSwimmer implements Swim {
    public void backStroke() {
        System.out.println("AbstractSwimmer.backStroke");
    }
}

public class FooSwimmer extends AbstractSwimmer {
}
```

```
new FooSwimmer().backStroke();
```

```
AbstractSwimmer.backStroke
```

-

```
public interface Swim {
    default void backStroke() {
        System.out.println("Swim.backStroke");
    }
}

public abstract class AbstractSwimmer implements Swim {
}

public class FooSwimmer extends AbstractSwimmer {
    public void backStroke() {
        System.out.println("FooSwimmer.backStroke");
    }
}
```

```
new FooSwimmer().backStroke();
```

```
FooSwimmer.backStroke
```

.

```
public interface A {
    default void foo() { System.out.println("A.foo"); }
}

public interface B {
    default void foo() { System.out.println("B.foo"); }
}
```

default foo .

extend Java .

abstract foo . A B .

```
public interface ABExtendsAbstract extends A, B {
    @Override
    void foo();
}
```

class ABExtendsAbstract implement foo .

```
public class ABExtendsAbstractImpl implements ABExtendsAbstract {
    @Override
    public void foo() { System.out.println("ABImpl.foo"); }
}
```

, default . A B foo .

```
public interface ABExtends extends A, B {
```

```
    @Override
    default void foo() { System.out.println("ABExtends.foo"); }
}
```

class ABExtends implement foo not .

```
public class ABExtendsImpl implements ABExtends {}
```

: https://riptutorial.com/ko/java/topic/113/-

73:

. {} .

.

, . .

:

```
// valid, but discouraged
Scanner scan = new Scanner(System.in);
int val = scan.nextInt();
if(val % 2 == 0)
    System.out.println("Val was even!");


// invalid; will not compile
// note the misleading indentation here
for(int i = 0; i < 10; i++)
    System.out.println(i);
    System.out.println("i is currently: " + i);
```

Examples

If / Else If / Else Control

```
if (i < 2) {
  System.out.println("i is less than 2");
} else if (i > 2) {
  System.out.println("i is more than 2");
} else {
  System.out.println("i is not less than 2, and not more than 2");
}
```

`if` `i` 1 .

`else if` `else if` (`else if` parent `if`) `false` `if`. `else if` `i` 2 .

`true` `else if` `else` .

`if` `else if` `true` `if` `else` .

For

```
for (int i = 0; i < 100; i++) {
    System.out.println(i);
}
```

`for` (;) / (`int i = 0`), (`i < 100`) (`i++`). { { . , `true` , `false` . } increment

.

. } ( ). .

for . for .

```
int i = obj.getLastestValue(); // i value is fetched from a method

for (; i < 100; i++) { // here initialization is not done
    System.out.println(i);
}
```

for (;;) { function-body } while (true) .

Nested For Loops

. 'for nested for loop'.

```
    for(;;){
        //Outer Loop Statements
        for(;;){
            //Inner Loop Statements
        }
        //Outer Loop Statements
    }
```

for .

```
for(int i=9;i>0;i--){//Outer Loop
    System.out.println();
    for(int k=i;k>0;k--){//Inner Loop -1
        System.out.print(" ");
    }
    for(int j=i;j<=9;j++){//Inner Loop -2
        System.out.print(" "+j);
    }
 }
```

While

```
int i = 0;
while (i < 100) { // condition gets checked BEFORE the loop body executes
    System.out.println(i);
    i++;
}
```

while true . " ".

.

do ... while

do...while . " ".

```
int i = 0;
do {
    i++;
    System.out.println(i);
} while (i < 100); // Condition gets checked AFTER the content of the loop executes.
```

`100` (`i < 100` `i <= 100`).

.

```
String theWord;
Scanner scan = new Scanner(System.in);
do {
    theWord = scan.nextLine();
} while (!theWord.equals("Bird"));

System.out.println(theWord);
```

`theWord` `theWord` .

Java SE 5

Java 5 for-each (for) .

```
List strings = new ArrayList();

strings.add("This");
strings.add("is");
strings.add("a for-each loop");

for (String string : strings) {
    System.out.println(string);
}
```

`Iterable` `List` `Set` Collections .

.

`Iterable<T>` `T[]` for `S` . if

- `T extends S`
- `T` `S` .
- `S` `T` `S` .
- `T` autoboxing `S` .

:

```
T elements = ...
for (S s : elements) {
}
```

int []		
[]	int	
`Iterable<Byte>`		
`Iterable<String>`	CharSequence	
`Iterable<CharSequence>`		
int []		
int []		

```
int i = 2;
if (i < 2) {
  System.out.println("i is less than 2");
} else {
  System.out.println("i is greater than 2");
}
```

`if` . `{ }` if . if .

`else` . `if` `false` `if` `if` .

: If

`switch` Java . long `if` - `else if` - `else` . `if` . .

`switch` .

- `case` : `switch` .
- `default` : `case` `true` `case` catch-all `true` .
- `case` ; `break` : `case` .

`continue` . .

- `break`
- `return`
- `throw`

`switch` `default` .

```
Scanner scan = new Scanner(System.in);
int i = scan.nextInt();
switch (i) {
    case 0:
        System.out.println("i is zero");
        break;
    case 1:
        System.out.println("i is one");
        break;
    case 2:
```

```
        System.out.println("i is two");
        break;
    default:
        System.out.println("i is less than zero or greater than two");
}
```

`break` "fall-through" . .

```
Scanner scan = new Scanner(System.in);
int foo = scan.nextInt();
switch(foo) {
    case 1:
        System.out.println("I'm equal or greater than one");
    case 2:
    case 3:
        System.out.println("I'm one, two, or three");
        break;
    default:
        System.out.println("I'm not either one, two, or three");
}
```

`foo == 1` .

```
I'm equal or greater than one
I'm one, two, or three
```

`foo == 3` .

```
I'm one, two, or three
```

Java SE 5

switch `enum` .

```
enum Option {
    BLUE_PILL,
    RED_PILL
}

public void takeOne(Option option) {
    switch(option) {
        case BLUE_PILL:
            System.out.println("Story ends, wake up, believe whatever you want.");
            break;
        case RED_PILL:
            System.out.println("I show you how deep the rabbit hole goes.");
            break;

    }
}
```

Java SE 7

`switch String` .

```
public void rhymingGame(String phrase) {
    switch (phrase) {
        case "apples and pears":
            System.out.println("Stairs");
            break;
        case "lorry":
            System.out.println("truck");
            break;
        default:
            System.out.println("Don't know any more");
    }
}
```

.

.

```
String name;

if (A > B) {
    name = "Billy";
} else {
    name = "Jimmy";
}
```

.

```
String name = A > B ? "Billy" : "Jimmy";
```

. .

`break` (`for` , `while`) switch .

:

```
while(true) {
   if(someCondition == 5) {
       break;
   }
}
```

. someCondition 5 .

break .

... ...

`try { ... } catch ( ... ) { ... }` Exceptions .

```
String age_input = "abc";
try {
    int age = Integer.parseInt(age_input);
    if (age >= 18) {
```

```
        System.out.println("You can vote!");
    } else {
        System.out.println("Sorry, you can't vote yet.");
    }
} catch (NumberFormatException ex) {
    System.err.println("Invalid input.  '" + age_input + "' is not a valid integer.");
}
```

.

. 'abc' .

`catch` `finally` . `finally` .

```
try { ... } catch ( ... ) { ... } finally { ... }
```

```
String age_input = "abc";
try {
    int age = Integer.parseInt(age_input);
    if (age >= 18) {
        System.out.println("You can vote!");
    } else {
        System.out.println("Sorry, you can't vote yet.");
    }
} catch (NumberFormatException ex) {
    System.err.println("Invalid input.  '" + age_input + "' is not a valid integer.");
} finally {
    System.out.println("This code will always be run, even if an exception is thrown");
}
```

.

. 'abc' .

.

/

label `break` `continue` .

```
outerloop:
for(...) {
    innerloop:
    for(...) {
        if(condition1)
            break outerloop;

        if(condition2)
            continue innerloop; // equivalent to: continue;
    }
}
```

Java .

continue . `continue` ( ) ( ).

```
String[] programmers = {"Adrian", "Paul", "John", "Harry"};

    //john is not printed out
    for (String name : programmers) {
        if (name.equals("John"))
            continue;
        System.out.println(name);
    }
```

`continue` () .

```
Outer: // The name of the outermost loop is kept here as 'Outer'
for(int i = 0; i < 5; )
{
    for(int j = 0; j < 5; j++)
    {
        continue Outer;
    }
}
```

: https://riptutorial.com/ko/java/topic/118/--

74:

. Java Java Collections Framework . `java.util.List` , `java.util.Collection` .

- ls.add (E); // .
- ls.remove (E); // .
- for (E element : ls) {} // .
- ls.toArray (new String [ls.length]); // .
- ls.get (int index); // .
- ls.set (int index, E element); // .
- ls.isEmpty (); // true false .
- ls.indexOf (Object o); // o , -1 .
- ls.lastIndexOf (Object o); // o , -1 .
- ls.size (); // List .

. " " . . "". Java .

- 0 .
- . , .
- . , , .
- . 0 , 1 .
- , .
- . , O (n) .

"" "". , *n* , *N + 1* *N* . :

```
List<String> list = new ArrayList<>();
list.add("world");
System.out.println(list.indexOf("world"));      // Prints "0"
// Inserting a new value at index 0 moves "world" to index 1
list.add(0, "Hello");
System.out.println(list.indexOf("world"));      // Prints "1"
System.out.println(list.indexOf("Hello"));      // Prints "0"
```

Examples

`Collections` .

- `sort(List<T> list) T extends Comparable<? super T>` `T extends Comparable<? super T>` ,
- `sort(List<T> list, Comparator<? super T> c)` .

. . , .

.

```
public class User {
    public final Long id;
    public final String username;
```

```
    public User(Long id, String username) {
        this.id = id;
        this.username = username;
    }

    @Override
    public String toString() {
        return String.format("%s:%d", username, id);
    }
}
```

Collections.sort(List<User> list) User Comparable .

```
public class User implements Comparable<User> {
    public final Long id;
    public final String username;

    public User(Long id, String username) {
        this.id = id;
        this.username = username;
    }

    @Override
    public String toString() {
        return String.format("%s:%d", username, id);
    }

    @Override
    /** The natural ordering for 'User' objects is by the 'id' field. */
    public int compareTo(User o) {
        return id.compareTo(o.id);
    }
}
```

String , Long , Integer Java Comparable compare compareTo .

User . (id .) :

```
List<User> users = Lists.newArrayList(
    new User(33L, "A"),
    new User(25L, "B"),
    new User(28L, ""));
Collections.sort(users);

System.out.print(users);
// [B:25, C:28, A:33]
```

User id name . Comparable .

Comparator sort .

```
Collections.sort(users, new Comparator<User>() {
    @Override
    /* Order two 'User' objects based on their names. */
    public int compare(User left, User right) {
        return left.username.compareTo(right.username);
```

```
    }
});
System.out.print(users);
// [A:33, B:25, C:28]
```

Java SE 8

Java 8 . one-liner .

```
Collections.sort(users, (l, r) -> l.username.compareTo(r.username));
```

, Java 8 `List sort` .

```
users.sort((l, r) -> l.username.compareTo(r.username))
```

(, : `String`) . `List` . `List` . :

```
List<String> strings;
```

`"string1" , "hello world!" "hello world!" , "goodbye" 9.2` .

```
List<Double> doubles;
```

`9.2 "hello world!" "hello world!" .`

`strings doubles` **null** NullPointerException !

.

1 : List

`List` . `ArrayList List LinkedList . .`

```
List<String> strings = new ArrayList<String>();
```

```
List<String> strings = new LinkedList<String>();
```

Java SE 7

Java SE 7 .

```
List<String> strings = new ArrayList<>();
```

```
List<String> strings = new LinkedList<>();
```

2 : Collections

`Collections List` List .

- `emptyList()` : .
- `singletonList(T)` : `singletonList(T)` .

`List`

- `addAll(L, T...)` : .

:

```
import java.util.List;
import java.util.Collections;

List<Integer> l = Collections.emptyList();
List<Integer> l1 = Collections.singletonList(42);
Collections.addAll(l1, 1, 2, 3);
```

List API 8 .

- `add(T type)`
- `add(int index, T type)`
- `remove(Object o)`
- `remove(int index)`
- `get(int index)`
- `set(int index, E element)`
- `int indexOf(Object o)`
- `int lastIndexOf(Object o)`

:

```
List<String> strings = new ArrayList<String>();
```

"Hello world!" . " !" , :

```
strings.add("Hello world!");
strings.add("Goodbye world!");
```

. " " . . :

```
strings.add(0, "Program starting!");
```

: 0.

, " !" , :

```
strings.remove("Goodbye world!");
```

(" !" . :

```
strings.remove(0);
```

:

1. , `ConcurrentModificationException` .

2. , , / `O(1)` `O(N)` .

`E get(int index);` List API :

```
strings.get(0);
```

.

`set(int index, E element);` `set(int index, E element);` . :

```
strings.set(0,"This is a replacement");
```

"This is a replacement" .

: set 0 . String 0 1 .

`int indexOf(Object o);` . _ -1 . .

```
strings.indexOf("This is a replacement")
```

0 0 "This is a replacement" . `int indexOf(Object o);` .`int lastIndexOf(Object o)` . "This is a replacement" :

```
strings.add("This is a replacement");
strings.lastIndexOf("This is a replacement");
```

1 0 .

, "hello", "how", "are", "you?" String List .

for-each .

```
public void printEachElement(List<String> list){
    for(String s : list){
        System.out.println(s);
    }
}
```

?

```
hello,
how
are
you?
```

StringBuilder .

```
public void printAsLine(List<String> list){
    StringBuilder builder = new StringBuilder();
    for(String s : list){
        builder.append(s);
    }
    System.out.println(builder.toString());
}
```

:

```
hello, how are you?
```

(ArrayList i) . : .

A B

A B , **B**

```
 List.removeAll(Collection c);
```

#:

```
public static void main(String[] args) {
    List<Integer> numbersA = new ArrayList<>();
    List<Integer> numbersB = new ArrayList<>();
    numbersA.addAll(Arrays.asList(new Integer[] { 1, 3, 4, 7, 5, 2 }));
    numbersB.addAll(Arrays.asList(new Integer[] { 13, 32, 533, 3, 4, 2 }));
    System.out.println("A: " + numbersA);
    System.out.println("B: " + numbersB);

    numbersB.removeAll(numbersA);
    System.out.println("B cleared: " + numbersB);
    }
```

.

A : [1, 3, 4, 7, 5, 2]

B : [13, 32, 533, 3, 4, 2]

B : [13, 32, 533]

2

A B .

`List.retainAll()` .

:

```
public static void main(String[] args) {
    List<Integer> numbersA = new ArrayList<>();
```

```
    List<Integer> numbersB = new ArrayList<>();
    numbersA.addAll(Arrays.asList(new Integer[] { 1, 3, 4, 7, 5, 2 }));
    numbersB.addAll(Arrays.asList(new Integer[] { 13, 32, 533, 3, 4, 2 }));

    System.out.println("A: " + numbersA);
    System.out.println("B: " + numbersB);
    List<Integer> numbersC = new ArrayList<>();
    numbersC.addAll(numbersA);
    numbersC.retainAll(numbersB);

    System.out.println("List A : " + numbersA);
    System.out.println("List B : " + numbersB);
    System.out.println("Common elements between A and B: " + numbersC);

}
```

```
List<Integer> nums = Arrays.asList(1, 2, 3);
List<String> strings = nums.stream()
    .map(Object::toString)
    .collect(Collectors.toList());
```

:

1.
2. `Object::toString`
3. `Collectors.toList()` `List` `String` .

ArrayList ,

`ArrayList` Java inbuilt . () ( ).

`AbstractList` `List` . `ArrayList` . `ArrayList` `ArrayList` . `ArrayList` . `ArrayList` `ArrayList` .

`ArrayList` .

```
List<T> myArrayList = new ArrayList<>();
```

`T` (Generics) `ArrayList` .

`ArrayList` Object . ( ).

`ArrayList` add `add()` `add()` .

```
myArrayList.add(element);
```

:

```
myArrayList.add(index, element); //index of the element should be an int (starting from 0)
```

`ArrayList` remove `remove()` .

```
myArrayList.remove(element);
```

.

```
myArrayList.remove(index); //index of the element should be an int (starting from 0)
```

List

`List` .

.

- set (int index, T type)
- int indexOf (T)

"Program starting!", "Hello world!" `ArrayList` . " !"

```
List<String> strings = new ArrayList<String>();
strings.add("Program starting!");
strings.add("Hello world!");
strings.add("Goodbye world!");
```

`set` .

```
strings.set(1, "Hi world");
```

. :

```
int pos = strings.indexOf("Goodbye world!");
if (pos >= 0) {
    strings.set(pos, "Goodbye cruel world!");
}
```

:

1. `set` `ConcurrentModificationException` .
2. `set` `ArrayList` (`O(1)`), `LinkedList` (`O(N)`).
3. `ArrayList` `LinkedList` `indexOf` (`O(N)`).

Collections .

```
List<String> ls = new ArrayList<String>();
List<String> unmodifiableList = Collections.unmodifiableList(ls);
```

.

```
List<String> unmodifiableList = Collections.singletonList("Only string in the list");
```

.

Collections (ls) .

:

```
Collections.reverse(ls);
```

rotate . . .

```
List<String> ls = new ArrayList<String>();
ls.add(" how");
ls.add(" are");
ls.add(" you?");
ls.add("hello,");
Collections.rotate(ls, 1);

for(String line : ls) System.out.print(line);
System.out.println();
```

", ?" .

.

```
Collections.shuffle(ls);
```

java.util.Random :

```
Random random = new Random(12);
Collections.shuffle(ls, random);
```

List -

`List` . .

List

`java.util.List` Java SE 8 `public` .

1. **:**
   - AbstractList
   - AbstractSequentialList
2. **:**
   - ArrayList
   - AttributeList
   - CopyOnWriteArrayList
   - LinkedList
   - RoleList

- RoleUnresolvedList
-
-

ArrayList

```
public class ArrayList<E>
extends AbstractList<E>
implements List<E>, RandomAccess, Cloneable, Serializable
```

ArrayList , List . , *ArrayList* (*List*) .

Integer ArrayList 100 .

```
List<Integer> myList = new ArrayList<Integer>(100); // Constructs an empty list with the
specified initial capacity.
```

- PROS :

, isEmpty, ***get*** , ***set*** , iterator listIterator . .

```
int e1 = myList.get(0);  //   \
int e2 = myList.get(10); //    | => All the same constant cost => O(1)
myList.set(2,10);        //   /
```

- :

( ) . :

add . , n O (n) .

O (n) .

AttributeList

CopyOnWriteArrayList

LinkedList

```
public class LinkedList<E>
extends AbstractSequentialList<E>
implements List<E>, Deque<E>, Cloneable, Serializable
```

LinkedList .

LinkedList

```
List<Integer> myList = new LinkedList<Integer>(); // Constructs an empty list.
```

- PROS :

.

```
myList.add(10);  // \
myList.add(0,2); //  | => constant time => O(1)
myList.remove(); // /
```

- : :

.

:

```
myList.get(10);    // \
myList.add(11,25); //  | => worst case done in O(n/2)
myList.set(15,35); // /
```

RoleList

RoleUnresolvedList

: https://riptutorial.com/ko/java/topic/2989/

75:

javadoc . .

Examples

Java `utils Random` . (`int` , `float`) . . .

```
import java.util.Random;

...

Random random = new Random();
int randInt = random.nextInt();
long randLong = random.nextLong();

double randDouble = random.nextDouble(); //This returns a value between 0.0 and 1.0
float randFloat = random.nextFloat(); //Same as nextDouble

byte[] randBytes = new byte[16];
random.nextBytes(randBytes); //nextBytes takes a user-supplied byte array, and fills it with
random bytes. It returns nothing.
```

: (`SecureRandom` ,). "" " " .

`Random nextInt(int bound)` , . , `nextInt` . `nextLong` , `nextDouble` .

```
Random random = new Random();
random.nextInt(1000); // 0 - 999

int number = 10 + random.nextInt(100); // number is in the range of 10 to 109
```

Java 1.7 `ThreadLocalRandom` (source) . thread PRNG ( ) . `nextInt` .

```
import java.util.concurrent.ThreadLocalRandom;

// nextInt is normally exclusive of the top value,
// so add 1 to make it inclusive
ThreadLocalRandom.current().nextInt(min, max + 1);
```

`bound` $2^{30}+1$ ( ) `nextInt(int bound)` .

. (2 ^ 31 n) . n . **n = 2 ^ 30 + 1, 1/2 2.**

, `nextInt` `nextInt` , `bound` int `bound` .

`Random ThreadLocalRandom` . . . .

PRNG `java.security.SecureRandom` . .

```
import java.security.SecureRandom;
import java.util.Arrays;

public class Foo {
    public static void main(String[] args) {
        SecureRandom rng = new SecureRandom();
        byte[] randomBytes = new byte[64];
        rng.nextBytes(randomBytes); // Fills randomBytes with random bytes (duh)
        System.out.println(Arrays.toString(randomBytes));
    }
}
```

`SecureRandom` 2^{48} `Random` 2^{160} . Mersenne Twister Xorshift `Random` PRNG .

SecureRandom . `SecureRandom` (`sun.security.provider.SecureRandom SUN`).

- `/dev/random` / `/dev/urandom` .
- Windows CryptoAPI `CryptGenRandom()` .

```
/**
 * returns a array of random numbers with no duplicates
 * @param range the range of possible numbers for ex. if 100 then it can be anywhere from 1-
100
 * @param length the length of the array of random numbers
 * @return array of random numbers with no duplicates.
 */
public static int[] getRandomNumbersWithNoDuplicates(int range, int length){
    if (length<range){
        // this is where all the random numbers
        int[] randomNumbers = new int[length];

        // loop through all the random numbers to set them
        for (int q = 0; q < randomNumbers.length; q++){

            // get the remaining possible numbers
            int remainingNumbers = range-q;

            // get a new random number from the remainingNumbers
            int newRandSpot = (int) (Math.random()*remainingNumbers);

            newRandSpot++;

            // loop through all the possible numbers
            for (int t = 1; t < range+1; t++){

                // check to see if this number has already been taken
                boolean taken = false;
                for (int number : randomNumbers){
                    if (t==number){
                        taken = true;
                        break;
                    }
                }

                // if it hasnt been taken then remove one from the spots
                if (!taken){
                    newRandSpot--;

                    // if we have gone though all the spots then set the value
```

```
                    if (newRandSpot==0){
                        randomNumbers[q] = t;
                    }
                }
            }
        }
        return randomNumbers;
    } else {
        // invalid can't have a length larger then the range of possible numbers
    }
    return null;
}
```

. newRandSpot . .

---

5 3 2 4 1 4 (5) . (2).

1 4 3 . 1 . 3 1 2. 2 . 4 . 1 0 randomNumber 4 .

```
//Creates a Random instance with a seed of 12345.
Random random = new Random(12345L);

//Gets a ThreadLocalRandom instance
ThreadLocalRandom tlr = ThreadLocalRandom.current();

//Set the instance's seed.
tlr.setSeed(12345L);
```

Random .

Long System.currentTimeMillis() .

```
Random random = new Random(System.currentTimeMillis());
ThreadLocalRandom.current().setSeed(System.currentTimeMillis());
```

apache-common lang3

org.apache.commons.lang3.RandomUtils .

```
int x = RandomUtils.nextInt(1, 1000);
```

nextInt(int startInclusive, int endExclusive) .

int long , double , float bytes .

RandomUtils .

```
static byte[] nextBytes(int count) //Creates an array of random bytes.
static double nextDouble() //Returns a random double within 0 - Double.MAX_VALUE
static double nextDouble(double startInclusive, double endInclusive) //Returns a random double
within the specified range.
```

```
static float nextFloat() //Returns a random float within 0 - Float.MAX_VALUE
static float nextFloat(float startInclusive, float endInclusive) //Returns a random float
within the specified range.
static int nextInt() //Returns a random int within 0 - Integer.MAX_VALUE
static int nextInt(int startInclusive, int endExclusive) //Returns a random integer within the
specified range.
static long nextLong() //Returns a random long within 0 - Long.MAX_VALUE
static long nextLong(long startInclusive, long endExclusive) //Returns a random long within
the specified range.
```

: https://riptutorial.com/ko/java/topic/890/-

76:

- `Date object = new Date();`
- `Date object = new Date(long date);`

	( ) Date .
	"the epoch"(1970 1 1 , 00:00:00 GMT) Date .

Java Date long . ( ) . Java Date Calendar .

Date . Calendar , , , , . . "" .

JVM . 10 "" .

Import Statement

```
import java.util.Date;
```

`Date` `java.util` .

`Date` . , `getDate()` .

```
public final class Transaction {
  private final Date date;

  public Date getTransactionDate() {
    return date;
  }
}
```

`date` Java 8 `java.time` API .

`Date` . `Calendar` .

Java 8

Java 8 LocalDate LocalTime `java.time` API . `java.time` API . Java 8 API . Dates and Time (java.time. *) .

Examples

Date

```
Date date = new Date();
System.out.println(date); // Thu Feb 25 05:03:59 IST 2016
```

Date .

```
Calendar calendar = Calendar.getInstance();
calendar.set(90, Calendar.DECEMBER, 11);
Date myBirthDate = calendar.getTime();
System.out.println(myBirthDate); // Mon Dec 31 00:00:00 IST 1990
```

Date Calendar . Factory Calendar Calendar . , Calendar .

```
calendar.set(90, Calendar.DECEMBER, 11, 8, 32, 35);
Date myBirthDatenTime = calendar.getTime();
System.out.println(myBirthDatenTime); // Mon Dec 31 08:32:35 IST 1990
```

, , .

Date

, LocalDate

Java SE 8

before, after, compareTo equals

```
//Use of Calendar and Date objects
final Date today = new Date();
final Calendar calendar = Calendar.getInstance();
calendar.set(1990, Calendar.NOVEMBER, 1, 0, 0, 0);
Date birthdate = calendar.getTime();

final Calendar calendar2 = Calendar.getInstance();
calendar2.set(1990, Calendar.NOVEMBER, 1, 0, 0, 0);
Date samebirthdate = calendar2.getTime();

//Before example
System.out.printf("Is %1$tF before %2$tF? %3$b%n", today, birthdate,
Boolean.valueOf(today.before(birthdate)));
System.out.printf("Is %1$tF before %1$tF? %3$b%n", today, today,
Boolean.valueOf(today.before(today)));
System.out.printf("Is %2$tF before %1$tF? %3$b%n", today, birthdate,
Boolean.valueOf(birthdate.before(today)));

//After example
System.out.printf("Is %1$tF after %2$tF? %3$b%n", today, birthdate,
Boolean.valueOf(today.after(birthdate)));
System.out.printf("Is %1$tF after %1$tF? %3$b%n", today, birthdate,
Boolean.valueOf(today.after(today)));
System.out.printf("Is %2$tF after %1$tF? %3$b%n", today, birthdate,
Boolean.valueOf(birthdate.after(today)));

//Compare example
System.out.printf("Compare %1$tF to %2$tF: %3$d%n", today, birthdate,
Integer.valueOf(today.compareTo(birthdate)));
```

```
System.out.printf("Compare %1$tF to %1$tF: %3$d%n", today, birthdate,
Integer.valueOf(today.compareTo(today)));
System.out.printf("Compare %2$tF to %1$tF: %3$d%n", today, birthdate,
Integer.valueOf(birthdate.compareTo(today)));

//Equal example
System.out.printf("Is %1$tF equal to %2$tF? %3$b%n", today, birthdate,
Boolean.valueOf(today.equals(birthdate)));
System.out.printf("Is %1$tF equal to %2$tF? %3$b%n", birthdate, samebirthdate,
            Boolean.valueOf(birthdate.equals(samebirthdate)));
System.out.printf(
            "Because birthdate.getTime() -> %1$d is different from samebirthdate.getTime() ->
%2$d, there are millisecondes!%n",
            Long.valueOf(birthdate.getTime()), Long.valueOf(samebirthdate.getTime()));

//Clear ms from calendars
calendar.clear(Calendar.MILLISECOND);
calendar2.clear(Calendar.MILLISECOND);
birthdate = calendar.getTime();
samebirthdate = calendar2.getTime();

System.out.printf("Is %1$tF equal to %2$tF after clearing ms? %3$b%n", birthdate,
samebirthdate,
            Boolean.valueOf(birthdate.equals(samebirthdate)));
```

Java SE 8

isBefore, isAfter, compareTo equals

```
//Use of LocalDate
final LocalDate now = LocalDate.now();
final LocalDate birthdate2 = LocalDate.of(2012, 6, 30);
final LocalDate birthdate3 = LocalDate.of(2012, 6, 30);

//Hours, minutes, second and nanoOfsecond can also be configured with an other class
LocalDateTime
//LocalDateTime.of(year, month, dayOfMonth, hour, minute, second, nanoOfSecond);

//isBefore example
System.out.printf("Is %1$tF before %2$tF? %3$b%n", now, birthdate2,
Boolean.valueOf(now.isBefore(birthdate2)));
System.out.printf("Is %1$tF before %1$tF? %3$b%n", now, birthdate2,
Boolean.valueOf(now.isBefore(now)));
System.out.printf("Is %2$tF before %1$tF? %3$b%n", now, birthdate2,
Boolean.valueOf(birthdate2.isBefore(now)));

//isAfter example
System.out.printf("Is %1$tF after %2$tF? %3$b%n", now, birthdate2,
Boolean.valueOf(now.isAfter(birthdate2)));
System.out.printf("Is %1$tF after %1$tF? %3$b%n", now, birthdate2,
Boolean.valueOf(now.isAfter(now)));
System.out.printf("Is %2$tF after %1$tF? %3$b%n", now, birthdate2,
Boolean.valueOf(birthdate2.isAfter(now)));

//compareTo example
System.out.printf("Compare %1$tF to %2$tF %3$d%n", now, birthdate2,
Integer.valueOf(now.compareTo(birthdate2)));
System.out.printf("Compare %1$tF to %1$tF %3$d%n", now, birthdate2,
```

```
Integer.valueOf(now.compareTo(now)));
System.out.printf("Compare %2$tF to %1$tF %3$d%n", now, birthdate2,
Integer.valueOf(birthdate2.compareTo(now)));

//equals example
System.out.printf("Is %1$tF equal to %2$tF? %3$b%n", now, birthdate2,
Boolean.valueOf(now.equals(birthdate2)));
System.out.printf("Is %1$tF to %2$tF? %3$b%n", birthdate2, birthdate3,
Boolean.valueOf(birthdate2.equals(birthdate3)));

//isEqual example
System.out.printf("Is %1$tF equal to %2$tF? %3$b%n", now, birthdate2,
Boolean.valueOf(now.isEqual(birthdate2)));
System.out.printf("Is %1$tF to %2$tF? %3$b%n", birthdate2, birthdate3,
Boolean.valueOf(birthdate2.isEqual(birthdate3)));
```

Java 8

Java 8 java.util.Calendar java.util.Date . Date 4 .

- ()
- ()
- compareTo (anotherDate)
- equals (Object obj)

`after` , `before` , `compareTo` `equals` getTime () .

`compareTo` .

- 0 : Date Date
- 0 : Date Date
- 0 : Date Date

`equals` , .

8

Java 8 Date Object java.time.LocalDate . `LocalDate` ChronoLocalDate .

java.time.LocalDateTime . `LocalDate` `LocalDateTime` .

`LocalDate` `ChronoLocalDate` `ChronoLocalDate` .

`LocalDate` `ChronoLocalDate` . .

[ChronoLocalDate] , LocalDate .

`LocalDate` 5 .

- isAfter (ChronoLocalDate)

- isBefore (ChronoLocalDate)
- isEqual (ChronoLocalDate)
- compareTo (ChronoLocalDate)
- equals (Object obj)

LocalDate isAfter , isBefore , isEqual , equals compareTo .

```
int compareTo0(LocalDate otherDate) {
    int cmp = (year - otherDate.year);
    if (cmp == 0) {
        cmp = (month - otherDate.month);
        if (cmp == 0) {
            cmp = (day - otherDate.day);
        }
    }
    return cmp;
}
```

equals isEqual compareTo0 .

ChronoLocalDate Epoch Day . Epoch 0 1970-01-01 (ISO) .

SimpleDateFormat format() Date String .

```
Date today = new Date();

SimpleDateFormat dateFormat = new SimpleDateFormat("dd-MMM-yy"); //pattern is specified here
System.out.println(dateFormat.format(today)); //25-Feb-16
```

applyPattern() .

```
dateFormat.applyPattern("dd-MM-yyyy");
System.out.println(dateFormat.format(today)); //25-02-2016

dateFormat.applyPattern("dd-MM-yyyy HH:mm:ss E");
System.out.println(dateFormat.format(today)); //25-02-2016 06:14:33 Thu
```

: mm (m) MM (M) . . "Y"(Y) " " "y"(y) .

SimpleDateFormat parse() String Date SimpleDateFormat .

```
DateFormat dateFormat = DateFormat.getDateInstance(DateFormat.SHORT, Locale.US);
String dateStr = "02/25/2016"; // input String
Date date = dateFormat.parse(dateStr);
System.out.println(date.getYear()); // 116
```

SHORT , MEDIUM (), LONG FULL ( ) 4 . .

	Locale.US	Locale.France
	6/30/09	30/06/09

	Locale.US	Locale.France
	2009 6 30	2009 9 30
	2009 6 30	2009 9 30
	2009 6 30	2009 6 30

yyyy/MM/dd hh:mm.ss .

2016/04/19 11 : 45.36

```
// define the format to use
String formatString = "yyyy/MM/dd hh:mm.ss";

// get a current date object
Date date = Calendar.getInstance().getTime();

// create the formatter
SimpleDateFormat simpleDateFormat = new SimpleDateFormat(formatString);

// format the date
String formattedDate = simpleDateFormat.format(date);

// print it
System.out.println(formattedDate);


// single-line version of all above code
System.out.println(new SimpleDateFormat("yyyy/MM/dd
hh:mm.ss").format(Calendar.getInstance().getTime()));
```

Date

Date .

```
    /**
     * Parses the date using the given format.
     *
     * @param formattedDate the formatted date string
     * @param dateFormat the date format which was used to create the string.
     * @return the date
     */
    public static Date parseDate(String formattedDate, String dateFormat) {
        Date date = null;
        SimpleDateFormat objDf = new SimpleDateFormat(dateFormat);
        try {
          date = objDf.parse(formattedDate);
        } catch (ParseException e) {
          // Do what ever needs to be done with exception.
        }
        return date;
    }
```

Java Date . Date long ( ) . Date .

Calendar . Calendar .

```
Calendar c = Calendar.getInstance();
```

Calendar . . .

```
c.set(1974, 6, 2, 8, 0, 0);
Date d = c.getTime();
```

`getTime` Date . ., .

/ .

- `(1974, 6, 2, 8, 0, 0)` ,
- 0 Calendar .

Java 8 LocalDate LocalDateTime

Date LocalDate Date LocalDate . .

LocalDate .

```
// Create a default date
LocalDate lDate = LocalDate.now();

// Creates a date from values
lDate = LocalDate.of(2017, 12, 15);

// create a date from string
lDate = LocalDate.parse("2017-12-15");

// creates a date from zone
LocalDate.now(ZoneId.systemDefault());
```

LocalDateTime .

```
// Create a default date time
LocalDateTime lDateTime = LocalDateTime.now();

// Creates a date time from values
lDateTime = LocalDateTime.of(2017, 12, 15, 11, 30);

// create a date time from string
lDateTime = LocalDateTime.parse("2017-12-05T11:30:30");

// create a date time from zone
LocalDateTime.now(ZoneId.systemDefault());
```

LocalDate to Date

```
Date date = Date.from(Instant.now());
ZoneId defaultZoneId = ZoneId.systemDefault();
```

```
// Date to LocalDate
LocalDate localDate = date.toInstant().atZone(defaultZoneId).toLocalDate();

// LocalDate to Date
Date.from(localDate.atStartOfDay(defaultZoneId).toInstant());
```

LocalDateTime

```
Date date = Date.from(Instant.now());
ZoneId defaultZoneId = ZoneId.systemDefault();

// Date to LocalDateTime
LocalDateTime localDateTime =  date.toInstant().atZone(defaultZoneId).toLocalDateTime();

// LocalDateTime to Date
Date out = Date.from(localDateTime.atZone(defaultZoneId).toInstant());
```

java.util.Date

`java.util.Date` .

- .
- Date .
- `new Date()` Date .

`java.text.SimpleDateFormat` Date .

```
Date date = new Date();
//print default time zone
System.out.println(TimeZone.getDefault().getDisplayName());
SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss"); //note: time zone not in
format!
//print date in the original time zone
System.out.println(sdf.format(date));
//current time in London
sdf.setTimeZone(TimeZone.getTimeZone("Europe/London"));
System.out.println(sdf.format(date));
```

:

```
Central European Time
2016-07-21 22:50:56
2016-07-21 21:50:56
```

java.util.Date java.sql.Date .

`java.util.Date` to `java.sql.Date` Date .

`java.sql.Date` , , `JDBC SQL DATE`

`java.util.Date()` Date . , `java.sql.Date convert(java.util.Date utilDate)`
`convert(java.util.Date utilDate)` .

```
public class UtilToSqlConversion {

    public static void main(String args[])
    {
        java.util.Date utilDate = new java.util.Date();
        System.out.println("java.util.Date is : " + utilDate);
        java.sql.Date sqlDate = convert(utilDate);
        System.out.println("java.sql.Date is : " + sqlDate);
        DateFormat df = new SimpleDateFormat("dd/MM/YYYY - hh:mm:ss");
        System.out.println("dateFormated date is : " + df.format(utilDate));
    }

    private static java.sql.Date convert(java.util.Date uDate) {
        java.sql.Date sDate = new java.sql.Date(uDate.getTime());
        return sDate;
    }

}
```

```
java.util.Date is : Fri Jul 22 14:40:35 IST 2016
java.sql.Date is : 2016-07-22
dateFormated date is : 22/07/2016 - 02:40:35
```

`java.util.Date` , `java.sql.Date`

Date LocalTime . LocalTime .

1. `LocalTime time = LocalTime.now();`
2. `time = LocalTime.MIDNIGHT;`
3. `time = LocalTime.NOON;`
4. `time = LocalTime.of(12, 12, 45);`

`LocalTime` toString .

```
System.out.println(time);
```

LocalTime , , , , .

```
time.plusMinutes(1);
time.getMinutes();
time.minusMinutes(1);
```

Date .

```
LocalTime lTime = LocalTime.now();
Instant instant = lTime.atDate(LocalDate.of(A_YEAR, A_MONTH, A_DAY)).
        atZone(ZoneId.systemDefault()).toInstant();
Date time = Date.from(instant);
```

.

: https://riptutorial.com/ko/java/topic/164/-

77: (java.time. *)

Examples

.

```
LocalDate.now()
```

.

```
LocalDate y = LocalDate.now().minusDays(1);
```

```
LocalDate t = LocalDate.now().plusDays(1);
```

.

```
LocalDate t = LocalDate.of(1974, 6, 2, 8, 30, 0, 0);
```

`plus minus LocalDate` "with" .

```
LocalDate.now().withMonth(6);
```

6 (`setMonth` 6 5 0 `java.util.Date`).

LocalDate LocalDate .

```
LocalDate ld = LocalDate.now().plusDays(1).plusYears(1);
```

1 .

```
LocalDateTime dateTime = LocalDateTime.of(2016, Month.JULY, 27, 8, 0);
LocalDateTime now = LocalDateTime.now();
LocalDateTime parsed = LocalDateTime.parse("2016-07-27T07:00:00");
```

```
ZoneId zoneId = ZoneId.of("UTC+2");
ZonedDateTime dateTime = ZonedDateTime.of(2016, Month.JULY, 27, 7, 0, 0, 235, zoneId);
ZonedDateTime composition = ZonedDateTime.of(localDate, localTime, zoneId);
ZonedDateTime now = ZonedDateTime.now(); // Default time zone
ZonedDateTime parsed = ZonedDateTime.parse("2016-07-27T07:00:00+01:00[Europe/Stockholm]");
```

(, DST)

```
ZoneOffset zoneOffset = ZoneOffset.ofHours(2);
OffsetDateTime dateTime = OffsetDateTime.of(2016, 7, 27, 7, 0, 0, 235, zoneOffset);
OffsetDateTime composition = OffsetDateTime.of(localDate, localTime, zoneOffset);
OffsetDateTime now = OffsetDateTime.now(); // Offset taken from the default ZoneId
OffsetDateTime parsed = OffsetDateTime.parse("2016-07-27T07:00:00+02:00");
```

```
LocalDate tomorrow = LocalDate.now().plusDays(1);
LocalDateTime anHourFromNow = LocalDateTime.now().plusHours(1);
Long daysBetween = java.time.temporal.ChronoUnit.DAYS.between(LocalDate.now(),
LocalDate.now().plusDays(3)); // 3
Duration duration = Duration.between(Instant.now(), ZonedDateTime.parse("2016-07-
27T07:00:00+01:00[Europe/Stockholm]"))
```

. Unix .

```
Instant now = Instant.now();
Instant epoch1 = Instant.ofEpochMilli(0);
Instant epoch2 = Instant.parse("1970-01-01T00:00:00Z");
java.time.temporal.ChronoUnit.MICROS.between(epoch1, epoch2); // 0
```

Date Time API

.

```
import java.time.Clock;
import java.time.Duration;
import java.time.Instant;
import java.time.LocalDate;
import java.time.LocalDateTime;
import java.time.LocalTime;
import java.time.ZoneId;
import java.time.ZonedDateTime;
import java.util.TimeZone;
public class SomeMethodsExamples {

/**
 * Has the methods of the class {@link LocalDateTime}
 */
public static void checkLocalDateTime() {
    LocalDateTime localDateTime = LocalDateTime.now();
    System.out.println("Local Date time using static now() method ::: >>> "
            + localDateTime);

    LocalDateTime ldt1 = LocalDateTime.now(ZoneId.of(ZoneId.SHORT_IDS
            .get("AET")));
    System.out
            .println("LOCAL TIME USING now(ZoneId zoneId) method ::: >>>>"
                    + ldt1);

    LocalDateTime ldt2 = LocalDateTime.now(Clock.system(ZoneId
            .of(ZoneId.SHORT_IDS.get("PST"))));
    System.out
            .println("Local TIME USING now(Clock.system(ZoneId.of())) ::: >>>> "
                    + ldt2);

    System.out
            .println("Following is a static map in ZoneId class which has mapping of short
timezone names to their Actual timezone names");
    System.out.println(ZoneId.SHORT_IDS);

}

/**
```

```
 * This has the methods of the class {@link LocalDate}
 */
public static void checkLocalDate() {
    LocalDate localDate = LocalDate.now();
    System.out.println("Gives date without Time using now() method. >> "
            + localDate);
    LocalDate localDate2 = LocalDate.now(ZoneId.of(ZoneId.SHORT_IDS
            .get("ECT")));
    System.out
            .println("now() is overridden to take ZoneID as parametere using this we can get
the same date under different timezones. >> "
                    + localDate2);
}

/**
 * This has the methods of abstract class {@link Clock}. Clock can be used
 * for time which has time with {@link TimeZone}.
 */
public static void checkClock() {
    Clock clock = Clock.systemUTC();
    // Represents time according to ISO 8601
    System.out.println("Time using Clock class : " + clock.instant());
}

/**
 * This has the {@link Instant} class methods.
 */
public static void checkInstant() {
    Instant instant = Instant.now();

    System.out.println("Instant using now() method :: " + instant);

    Instant ins1 = Instant.now(Clock.systemUTC());

    System.out.println("Instants using now(Clock clock) :: " + ins1);

}

/**
 * This class checks the methods of the {@link Duration} class.
 */
public static void checkDuration() {
    // toString() converts the duration to PTnHnMnS format according to ISO
    // 8601 standard. If a field is zero its ignored.

    // P is the duration designator (historically called "period") placed at
    // the start of the duration representation.
    // Y is the year designator that follows the value for the number of
    // years.
    // M is the month designator that follows the value for the number of
    // months.
    // W is the week designator that follows the value for the number of
    // weeks.
    // D is the day designator that follows the value for the number of
    // days.
    // T is the time designator that precedes the time components of the
    // representation.
    // H is the hour designator that follows the value for the number of
    // hours.
    // M is the minute designator that follows the value for the number of
    // minutes.
```

```
    // S is the second designator that follows the value for the number of
    // seconds.

    System.out.println(Duration.ofDays(2));
}

/**
 * Shows Local time without date. It doesn't store or represenet a date and
 * time. Instead its a representation of Time like clock on the wall.
 */
public static void checkLocalTime() {
    LocalTime localTime = LocalTime.now();
    System.out.println("LocalTime :: " + localTime);
}

/**
 * A date time with Time zone details in ISO-8601 standards.
 */
public static void checkZonedDateTime() {
    ZonedDateTime zonedDateTime = ZonedDateTime.now(ZoneId
            .of(ZoneId.SHORT_IDS.get("CST")));
    System.out.println(zonedDateTime);

}
}
```

Java 8 `java.text` `DateFormat` `SimpleDateFormat` `DateFormat` .

Java 8 .

`String` `DateTimeFormatter` .

```
import java.time.*;
import java.time.format.*;

class DateTimeFormat
{
    public static void main(String[] args) {

        //Parsing
        String pattern = "d-MM-yyyy HH:mm";
        DateTimeFormatter dtF1 = DateTimeFormatter.ofPattern(pattern);

        LocalDateTime ldp1 = LocalDateTime.parse("2014-03-25T01:30"), //Default format
                      ldp2 = LocalDateTime.parse("15-05-2016 13:55",dtF1); //Custom format

        System.out.println(ldp1 + "\n" + ldp2); //Will be printed in Default format

        //Formatting
        DateTimeFormatter dtF2 = DateTimeFormatter.ofPattern("EEE d, MMMM, yyyy HH:mm");

        DateTimeFormatter dtF3 = DateTimeFormatter.ISO_LOCAL_DATE_TIME;

        LocalDateTime ldtf1 = LocalDateTime.now();

        System.out.println(ldtf1.format(dtF2) +"\n"+ldtf1.format(dtF3));
    }
}
```

. ISO8061 .

2 LocalDates

`LocalDate` `ChronoUnit` :

```
LocalDate d1 = LocalDate.of(2017, 5, 1);
LocalDate d2 = LocalDate.of(2017, 5, 18);
```

`ChronoUnit` `between` `between` 2 `Temporal` `LocalDate`

```
long days = ChronoUnit.DAYS.between(d1, d2);
System.out.println( days );
```

(java.time. *) : https://riptutorial.com/ko/java/topic/4813/---java-time----

78:

- ("localhost", 1234); // "localhost" 1234 .
- SocketServer ("localhost", 1234); // localhost 1234 .
- socketServer.accept (); // Socket .

Examples

: .

```
//Open a listening "ServerSocket" on port 1234.
ServerSocket serverSocket = new ServerSocket(1234);

while (true) {
    // Wait for a client connection.
    // Once a client connected, we get a "Socket" object
    // that can be used to send and receive messages to/from the newly
    // connected client
    Socket clientSocket = serverSocket.accept();

    // Here we'll add the code to handle one specific client.
}
```

:

. (OS 1,000).

```
new Thread(() -> {
    // Get the socket's InputStream, to read bytes from the socket
    InputStream in = clientSocket.getInputStream();
    // wrap the InputStream in a reader so you can read a String instead of bytes
    BufferedReader reader = new BufferedReader(
            new InputStreamReader(in, StandardCharsets.UTF_8));
    // Read text from the socket and print line by line
    String line;
    while ((line = reader.readLine()) != null) {
        System.out.println(line);
    }
    }).start();
```

:

```
// 127.0.0.1 is the address of the server (this is the localhost address; i.e.
// the address of our own machine)
// 1234 is the port that the server will be listening on
Socket socket = new Socket("127.0.0.1", 1234);

// Write a string into the socket, and flush the buffer
OutputStream outStream = socket.getOutputStream();
PrintWriter writer = new PrintWriter(
```

```
        new OutputStreamWriter(outStream, StandardCharsets.UTF_8));
writer.println("Hello world!");
writer.flush();
```

.

1. , OS . `socket.close()` .

2. I / O (/) . ? ? . , `IOException` .

.

```
// "try-with-resources" will close the socket once we leave its scope
try (Socket socket = new Socket("127.0.0.1", 1234)) {
    OutputStream outStream = socket.getOutputStream();
    PrintWriter writer = new PrintWriter(
            new OutputStreamWriter(outStream, StandardCharsets.UTF_8));
    writer.println("Hello world!");
    writer.flush();
} catch (IOException e) {
    //Handle the error
}
```

-

:

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.net.ServerSocket;
import java.net.Socket;
import java.nio.charset.StandardCharsets;

public class Server {
    public static void main(String args[]) {
        try (ServerSocket serverSocket = new ServerSocket(1234)) {
            while (true) {
                // Wait for a client connection.
                Socket clientSocket = serverSocket.accept();

                // Create and start a thread to handle the new client
                new Thread(() -> {
                    try {
                        // Get the socket's InputStream, to read bytes
                        // from the socket
                        InputStream in = clientSocket.getInputStream();
                        // wrap the InputStream in a reader so you can
                        // read a String instead of bytes
                        BufferedReader reader = new BufferedReader(
                             new InputStreamReader(in, StandardCharsets.UTF_8));
                        // Read from the socket and print line by line
                        String line;
                        while ((line = reader.readLine()) != null) {
                            System.out.println(line);
```

```
                        }
                    }
                    catch (IOException e) {
                        e.printStackTrace();
                    } finally {
                        // This finally block ensures the socket is closed.
                        // A try-with-resources block cannot be used because
                        // the socket is passed into a thread, so it isn't
                        // created and closed in the same block
                        try {
                            clientSocket.close();
                        } catch (IOException e) {
                            e.printStackTrace();
                        }
                    }
                }).start();
            }
        }
        catch (IOException e) {
            e.printStackTrace();
        }

    }
}
```

:

```
import java.io.IOException;
import java.io.OutputStream;
import java.io.OutputStreamWriter;
import java.io.PrintWriter;
import java.net.Socket;
import java.nio.charset.StandardCharsets;

public class Client {
    public static void main(String args[]) {
        try (Socket socket = new Socket("127.0.0.1", 1234)) {
            // We'll reach this code once we've connected to the server

            // Write a string into the socket, and flush the buffer
            OutputStream outStream = socket.getOutputStream();
            PrintWriter writer = new PrintWriter(
                    new OutputStreamWriter(outStream, StandardCharsets.UTF_8));
            writer.println("Hello world!");
            writer.flush();
        } catch (IOException e) {
            // Exception should be handled.
            e.printStackTrace();
        }
    }
}
```

InputStream TrustStore KeyStore

```
public class TrustLoader {

    public static void main(String args[]) {
        try {
```

```
                //Gets the inputstream of a a trust store file under ssl/rpgrenadesClient.jks
                //This path refers to the ssl folder in the jar file, in a jar file in the
same directory
                //as this jar file, or a different directory in the same directory as the jar
file
                InputStream stream =
TrustLoader.class.getResourceAsStream("/ssl/rpgrenadesClient.jks");
                //Both trustStores and keyStores are represented by the KeyStore object
                KeyStore trustStore = KeyStore.getInstance(KeyStore.getDefaultType());
                //The password for the trustStore
                char[] trustStorePassword = "password".toCharArray();
                //This loads the trust store into the object
                trustStore.load(stream, trustStorePassword);

                //This is defining the SSLContext so the trust store will be used
                //Getting default SSLContext to edit.
                SSLContext context = SSLContext.getInstance("SSL");
                //TrustMangers hold trust stores, more than one can be added
                TrustManagerFactory factory =
TrustManagerFactory.getInstance(TrustManagerFactory.getDefaultAlgorithm());
                //Adds the truststore to the factory
                factory.init(trustStore);
                //This is passed to the SSLContext init method
                TrustManager[] managers = factory.getTrustManagers();
                context.init(null, managers, null);
                //Sets our new SSLContext to be used.
                SSLContext.setDefault(context);
            } catch (KeyStoreException | IOException | NoSuchAlgorithmException
                    | CertificateException | KeyManagementException ex) {
                //Handle error
                ex.printStackTrace();
            }

    }
}
```

KeyStore , `Trust Key` , . `KeyManager[] SSLContext.init` . `SSLContext.init(keyMangers, trustMangers, null)`

-

```
import java.io.*;
import java.net.Socket;

public class Main {

    public static void main(String[] args) throws IOException {//We don't handle Exceptions in
this example
        //Open a socket to stackoverflow.com, port 80
        Socket socket = new Socket("stackoverflow.com",80);

        //Prepare input, output stream before sending request
        OutputStream outStream = socket.getOutputStream();
        InputStream inStream = socket.getInputStream();
        BufferedReader reader = new BufferedReader(new InputStreamReader(inStream));
        PrintWriter writer = new PrintWriter(new BufferedOutputStream(outStream));

        //Send a basic HTTP header
        writer.print("GET / HTTP/1.1\nHost:stackoverflow.com\n\n");
```

```
        writer.flush();

        //Read the response
        System.out.println(readFully(reader));

        //Close the socket
        socket.close();
    }

    private static String readFully(Reader in) {
        StringBuilder sb = new StringBuilder();
        int BUFFER_SIZE=1024;
        char[] buffer = new char[BUFFER_SIZE]; // or some other size,
        int charsRead = 0;
        while ( (charsRead  = rd.read(buffer, 0, BUFFER_SIZE)) != -1) {
          sb.append(buffer, 0, charsRead);
        }
    }
}
```

HTTP `HTTP/1.1 200 OK` HTTP HTML .

EOF `readFully()` . `readLine()` Apache . commons-IOUtils

. Java Apache HTTP Google HTTP HTTP

UDP ( ) /

Client.java

```
import java.io.*;
import java.net.*;

public class Client{
    public static void main(String [] args) throws IOException{
        DatagramSocket clientSocket = new DatagramSocket();
        InetAddress address = InetAddress.getByName(args[0]);

        String ex = "Hello, World!";
        byte[] buf = ex.getBytes();

        DatagramPacket packet = new DatagramPacket(buf,buf.length, address, 4160);
        clientSocket.send(packet);
    }
}
```

(`args[0]`) . 4160.

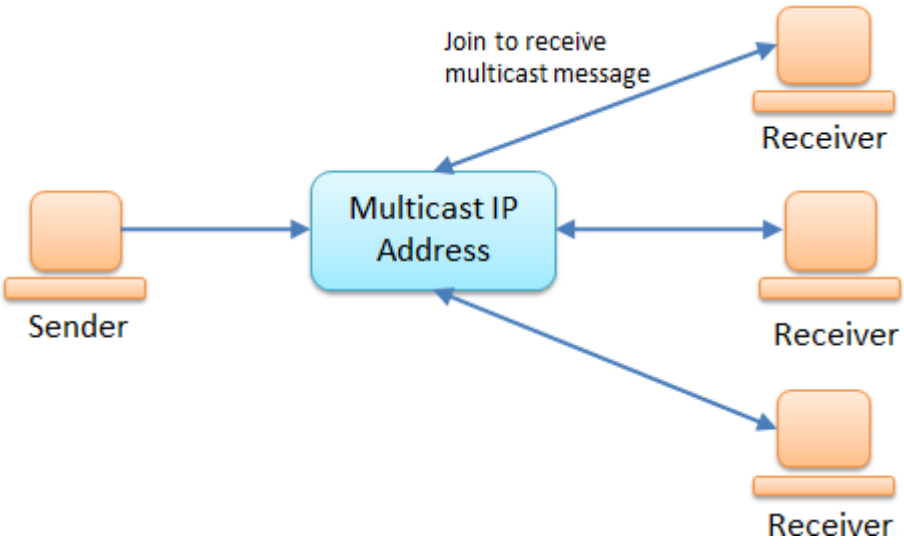
Server.java

```
import java.io.*;
import java.net.*;

public class Server{
    public static void main(String [] args) throws IOException{
        DatagramSocket serverSocket = new DatagramSocket(4160);
```

```
        byte[] rbuf = new byte[256];
        DatagramPacket packet = new DatagramPacket(rbuf, rbuf.length);
        serverSocket.receive(packet);
        String response = new String(packet.getData());
        System.out.println("Response: " + response);
    }
}
```

(4160) DatagramSocket .

Datagram Socket . IP .



:

```
public class Server {

    private DatagramSocket serverSocket;

    private String ip;

    private int port;

    public Server(String ip, int port) throws SocketException, IOException{
        this.ip = ip;
        this.port = port;
        // socket used to send
        serverSocket = new DatagramSocket();
    }

    public void send() throws IOException{
        // make datagram packet
        byte[] message = ("Multicasting...").getBytes();
        DatagramPacket packet = new DatagramPacket(message, message.length,
            InetAddress.getByName(ip), port);
        // send packet
        serverSocket.send(packet);
    }

    public void close(){
        serverSocket.close();
    }
```

```
}
```

:

```
public class Client {

    private MulticastSocket socket;

    public Client(String ip, int port) throws IOException {

        // important that this is a multicast socket
        socket = new MulticastSocket(port);

        // join by ip
        socket.joinGroup(InetAddress.getByName(ip));
    }

    public void printMessage() throws IOException{
        // make datagram packet to recieve
        byte[] message = new byte[256];
        DatagramPacket packet = new DatagramPacket(message, message.length);

        // recieve the packet
        socket.receive(packet);
        System.out.println(new String(packet.getData()));
    }

    public void close(){
        socket.close();
    }
}
```

:

```
public static void main(String[] args) {
    try {
        final String ip = args[0];
        final int port = Integer.parseInt(args[1]);
        Server server = new Server(ip, port);
        server.send();
        server.close();
    } catch (IOException ex) {
        ex.printStackTrace();
    }
}
```

:

```
public static void main(String[] args) {
    try {
        final String ip = args[0];
        final int port = Integer.parseInt(args[1]);
        Client client = new Client(ip, port);
        client.printMessage();
        client.close();
    } catch (IOException ex) {
        ex.printStackTrace();
```

```
    }
}
```

: IP . `send()` (`printMessage()`) . .

SSL ( )

SSL .

" " SSL .

```
try {
   // Create a trust manager that does not validate certificate chains
   TrustManager[] trustAllCerts = new TrustManager[] {
      new X509TrustManager() {
       public X509Certificate[] getAcceptedIssuers() {
           return null;
       }
       public void checkClientTrusted(X509Certificate[] certs, String authType) {
       }
       public void checkServerTrusted(X509Certificate[] certs, String authType) {
       }
      }
   };

   // Install the all-trusting trust manager
   SSLContext sc = SSLContext.getInstance("SSL");
   sc.init(null, trustAllCerts, new java.security.SecureRandom());
   HttpsURLConnection.setDefaultSSLSocketFactory(sc.getSocketFactory());

   // Create all-trusting host name verifier
   HostnameVerifier allHostsValid = new HostnameVerifier() {
       public boolean verify(String hostname, SSLSession session) {
           return true;
       }
   };

   // Install the all-trusting host verifier
   HttpsURLConnection.setDefaultHostnameVerifier(allHostsValid);
} catch (NoSuchAlgorithmException | KeyManagementException e) {
    e.printStackTrace();
}
```

!

```
String fileName     = "file.zip";                  // name of the file
String urlToGetFrom = "http://www.mywebsite.com/"; // URL to get it from
String pathToSaveTo = "C:\\Users\\user\\";         // where to put it

//If the file already exists, it will be overwritten!

//Opening OutputStream to the destination file
try (ReadableByteChannel rbc =
      Channels.newChannel(new URL(urlToGetFrom + fileName).openStream()) ) {
    try ( FileChannel channel =
        new FileOutputStream(pathToSaveTo + fileName).getChannel(); ) {
      channel.transferFrom(rbc, 0, Long.MAX_VALUE);
```

```
    }
    catch (FileNotFoundException e) { /* Output directory not found */ }
    catch (IOException e)           { /* File IO error */ }
}
catch (MalformedURLException e)     { /* URL is malformed */ }
catch (IOException e)               { /* IO error connecting to website */ }
```

- catch !
- .
- .

: https://riptutorial.com/ko/java/topic/149/

79:

?

. Java . .

.

- =
- (Base)
- .
- .
- . abstract .
- . java .
- Upcasting : 3 .
- =

Examples

-

:

```
package base;

/*
Abstract classes cannot be instantiated, but they can be subclassed
*/
public abstract class ClsVirusScanner {

    //With One Abstract method
    public abstract void fnStartScan();

    protected void fnCheckForUpdateVersion(){
        System.out.println("Perform Virus Scanner Version Check");
    }

    protected void fnBootTimeScan(){
        System.out.println("Perform BootTime Scan");
    }
    protected void fnInternetSecutiry(){
        System.out.println("Scan for Internet Security");
    }

    protected void fnRealTimeScan(){
        System.out.println("Perform RealTime Scan");
    }

    protected void fnVirusMalwareScan(){
        System.out.println("Detect Virus & Malware");
    }
}
```

:

```
import base.ClsVirusScanner;

//All the 3 child classes inherits the base class ClsVirusScanner
//Child Class 1
class ClsPaidVersion extends ClsVirusScanner{
    @Override
    public void fnStartScan() {
        super.fnCheckForUpdateVersion();
        super.fnBootTimeScan();
        super.fnInternetSecutiry();
        super.fnRealTimeScan();
        super.fnVirusMalwareScan();
    }
};    //ClsPaidVersion IS-A ClsVirusScanner
//Child Class 2

class ClsTrialVersion extends ClsVirusScanner{
    @Override
    public void fnStartScan() {
        super.fnInternetSecutiry();
        super.fnVirusMalwareScan();
    }
};    //ClsTrialVersion IS-A ClsVirusScanner

//Child Class 3
class ClsFreeVersion extends ClsVirusScanner{
    @Override
    public void fnStartScan() {
        super.fnVirusMalwareScan();
    }
};     //ClsTrialVersion IS-A ClsVirusScanner
```

/ .

```
//Calling Class
public class ClsRunTheApplication {

    public static void main(String[] args) {

        final String VIRUS_SCANNER_VERSION = "TRIAL_VERSION";

        //Parent Refers Null
        ClsVirusScanner objVS=null;

        //String Cases Supported from Java SE 7
        switch (VIRUS_SCANNER_VERSION){
        case "FREE_VERSION":

            //Parent Refers Child Object 3
            //ClsFreeVersion IS-A ClsVirusScanner
            objVS = new ClsFreeVersion(); //Dynamic or Runtime Binding
            break;
        case "PAID_VERSION":

            //Parent Refers Child Object 1
            //ClsPaidVersion IS-A ClsVirusScanner
            objVS = new ClsPaidVersion(); //Dynamic or Runtime Binding
```

```
            break;
        case "TRIAL_VERSION":

            //Parent Refers Child Object 2
            objVS = new ClsTrialVersion(); //Dynamic or Runtime Binding
            break;
        }

        //Method fnStartScan() is the Version of ClsTrialVersion()
        objVS.fnStartScan();

    }
}
```

:

```
Scan for Internet Security
Detect Virus & Malware
```

:

```
objVS = new ClsFreeVersion();
objVS = new ClsPaidVersion();
objVS = new ClsTrialVersion()
```

: https://riptutorial.com/ko/java/topic/9204/--

80: JAR

Java 9 Jar Java Jar (MRJAR). JEP 238 .

Examples

Jar

MANIFEST.MF `Multi-Release: true` Jar Jar Java (MRJAR) .

Jar .

```
jar root
  - A.class
  - B.class
  - C.class
  - D.class
  - META-INF
     - versions
        - 9
           - A.class
           - B.class
        - 10
           - A.class
```

- JDK <9 Java .
- JDK 9 A B `root/META-INF/versions/9` C D .
- JDK 10 A `root/META-INF/versions/10` .

jar Jar

`jar` Java 8 Java 9 Jar .

```
C:\Users\manouti>jar --create --file MR.jar -C sampleproject-base demo --release 9 -C
sampleproject-9 demo
Warning: entry META-INF/versions/9/demo/SampleClass.class contains a class that
is identical to an entry already in the jar
```

`--release 9 sampleproject-9` , `root/META-INF/versions/9 jar` (`sampleproject-9 demo`) . .

```
jar root
  - demo
     - SampleClass.class
  - META-INF
     - versions
        - 9
           - demo
              - SampleClass.class
```

`SampleClass` URL Java 9 Main .

```
package demo;

import java.net.URL;

public class Main {

    public static void main(String[] args) throws Exception {
        URL url = Main.class.getClassLoader().getResource("demo/SampleClass.class");
        System.out.println(url);
    }
}
```

jar .

```
C:\Users\manouti>jar --create --file MR.jar -C sampleproject-base demo --release 9 -C
sampleproject-9 demoentry: META-INF/versions/9/demo/Main.class, contains a new public class
not found in base entries
Warning: entry META-INF/versions/9/demo/Main.java, multiple resources with same name
Warning: entry META-INF/versions/9/demo/SampleClass.class contains a class that
is identical to an entry already in the jar
invalid multi-release jar file MR.jar deleted
```

`jar` . MRJAR Java API . . `jar` . , `Main` . Java 9 .

`Main` Java 9 `Main` . `javac --release` .

```
C:\Users\manouti\sampleproject-base\demo>javac --release 8 Main.java
C:\Users\manouti\sampleproject-base\demo>cd ../..
C:\Users\manouti>jar --create --file MR.jar -C sampleproject-base demo --release 9 -C
sampleproject-9 demo
```

Main SampleClass .

```
C:\Users\manouti>java --class-path MR.jar demo.Main
jar:file:/C:/Users/manouti/MR.jar!/META-INF/versions/9/demo/SampleClass.class
```

Jar URL

Jar :

```
jar root
  - demo
     - SampleClass.class
  - META-INF
     - versions
        - 9
           - demo
              - SampleClass.class
```

`SampleClass` URL .

```
package demo;
```

```
import java.net.URL;

public class Main {

    public static void main(String[] args) throws Exception {
        URL url = Main.class.getClassLoader().getResource("demo/SampleClass.class");
        System.out.println(url);
    }
}
```

MRJAR Java 9 .

```
C:\Users\manouti>java --class-path MR.jar demo.Main
jar:file:/C:/Users/manouti/MR.jar!/META-INF/versions/9/demo/SampleClass.class
```

JAR : https://riptutorial.com/ko/java/topic/9866/--jar-

81:

(Polymorphism) OOP ( ) . "poly" "morphs" . Poly "many" morph "forms"( ) .

. .

Interfaces Java . API . `interface` `implement` .

Examples

.

.

.

1.
2.
3.

.

. .

```
class Polymorph {

    public int add(int a, int b){
        return a + b;
    }

    public int add(int a, int b, int c){
        return a + b + c;
    }

    public float add(float a, float b){
        return a + b;
    }

    public static void main(String... args){
        Polymorph poly = new Polymorph();
        int a = 1, b = 2, c = 3;
        float d = 1.5, e = 2.5;

        System.out.println(poly.add(a, b));
        System.out.println(poly.add(a, b, c));
        System.out.println(poly.add(d, e));
    }

}
```

.

```
2
```

```
6
4.000000
```

. .

```
public class Polymorph {

    private static void methodOverloaded()
    {
        //No argument, private static method
    }

    private int methodOverloaded(int i)
    {
        //One argument private non-static method
        return i;
    }

    static int methodOverloaded(double d)
    {
        //static Method
        return 0;
    }

    public void methodOverloaded(int i, double d)
    {
        //Public non-static Method
    }
}
```

.

```
public class Polymorph {

void methodOverloaded(){
    //No argument and No return type
}

int methodOverloaded(){
    //No argument and int return type
    return 0;
}
```

() .

. references . references . .

reference , .

```
class SuperType {
    public void sayHello(){
        System.out.println("Hello from SuperType");
    }

    public void sayBye(){
        System.out.println("Bye from SuperType");
```

```
    }
}

class SubType extends SuperType {
    // override the superclass method
    public void sayHello(){
        System.out.println("Hello from SubType");
    }
}

class Test {
    public static void main(String... args){
        SuperType superType = new SuperType();
        superType.sayHello(); // -> Hello from SuperType

        // make the reference point to an object of the subclass
        superType = new SubType();
        // behaviour is governed by the object, not by the reference
        superType.sayHello(); // -> Hello from SubType

        // non-overridden method is simply inherited
        superType.sayBye(); // -> Bye from SuperType
    }
}
```

(override), (override) ( ) .

-
- (,).
-
- throws throw . throw . . , void methodX (int i) void methodX (int k) .
- . .

```
import java.util.ArrayList;
import java.util.List;

import static java.lang.System.out;

public class PolymorphismDemo {

    public static void main(String[] args) {
        List<FlyingMachine> machines = new ArrayList<FlyingMachine>();
        machines.add(new FlyingMachine());
        machines.add(new Jet());
        machines.add(new Helicopter());
        machines.add(new Jet());

        new MakeThingsFly().letTheMachinesFly(machines);
    }
}

class MakeThingsFly {
    public void letTheMachinesFly(List<FlyingMachine> flyingMachines) {
        for (FlyingMachine flyingMachine : flyingMachines) {
            flyingMachine.fly();
        }
    }
}
```

```
class FlyingMachine {
    public void fly() {
        out.println("No implementation");
    }
}

class Jet extends FlyingMachine {
    @Override
    public void fly() {
        out.println("Start, taxi, fly");
    }

    public void bombardment() {
        out.println("Fire missile");
    }
}

class Helicopter extends FlyingMachine {
    @Override
    public void fly() {
        out.println("Start vertically, hover, fly");
    }
}
```

a) `MakeThingsFly` `FlyingMachine` .

b) `letTheMachinesFly` (: `PropellerPlane`) (!).

```
public void letTheMachinesFly(List<FlyingMachine> flyingMachines) {
        for (FlyingMachine flyingMachine : flyingMachines) {
            flyingMachine.fly();
        }
    }
}
```

. .

Final Java . Java . . .

. , private final .

StackOverflow Virtual Methods . C # Java ? :

```
public class A{
    public void hello(){
        System.out.println("Hello");
    }

    public void boo(){
        System.out.println("Say boo");

    }
}

public class B extends A{
     public void hello(){
        System.out.println("No");
```

```
     }

    public void boo(){
        System.out.println("Say haha");

    }
}
```

B hello () boo () B A "No" "Say haha" . A .

abstract . "abstract" . boo () abstract .

```
public class A{
   public void hello(){
        System.out.println("Hello");
    }

    abstract void boo();
}

public class B extends A{
     public void hello(){
        System.out.println("No");
     }

    public void boo(){
        System.out.println("Say haha");

    }
}
```

B boo () B boo () boo () "Say haha" "Say haha" .

:

C # Java ?

.

/ ?

. Java . .

.

1. - ()
2. .
3. `super.methodName()` .
4. ( )

:

```
import java.util.HashMap;

abstract class Game implements Runnable{
```

```
    protected boolean runGame = true;
    protected Player player1 = null;
    protected Player player2 = null;
    protected Player currentPlayer = null;

    public Game(){
        player1 = new Player("Player 1");
        player2 = new Player("Player 2");
        currentPlayer = player1;
        initializeGame();
    }

    /* Type 1: Let subclass define own implementation. Base class defines abstract method to
force
        sub-classes to define implementation
    */

    protected abstract void initializeGame();

    /* Type 2: Sub-class can change the behaviour. If not, base class behaviour is applicable
*/
    protected void logTimeBetweenMoves(Player player){
        System.out.println("Base class: Move Duration: player.PlayerActTime -
player.MoveShownTime");
    }

    /* Type 3: Base class provides implementation. Sub-class can enhance base class
implementation by calling
        super.methodName() in first line of the child class method and specific implementation
later */
    protected void logGameStatistics(){
        System.out.println("Base class: logGameStatistics:");
    }
    /* Type 4: Template method: Structure of base class can't be changed but sub-class can
some part of behaviour */
    protected void runGame() throws Exception{
        System.out.println("Base class: Defining the flow for Game:");
        while (runGame) {
            /*
            1. Set current player
            2. Get Player Move
            */
            validatePlayerMove(currentPlayer);
            logTimeBetweenMoves(currentPlayer);
            Thread.sleep(500);
            setNextPlayer();
        }
        logGameStatistics();
    }
    /* sub-part of the template method, which define child class behaviour */
    protected abstract void validatePlayerMove(Player p);

    protected void setRunGame(boolean status){
        this.runGame = status;
    }
    public void setCurrentPlayer(Player p){
        this.currentPlayer = p;
    }
    public void setNextPlayer(){
        if (currentPlayer == player1) {
```

```
            currentPlayer = player2;
        }else{
            currentPlayer = player1;
        }
    }
    public void run(){
        try{
            runGame();
        }catch(Exception err){
            err.printStackTrace();
        }
    }
}

class Player{
    String name;
    Player(String name){
        this.name = name;
    }
    public String getName(){
        return name;
    }
}

/* Concrete Game implementation  */
class Chess extends Game{
    public Chess(){
        super();
    }
    public void initializeGame(){
        System.out.println("Child class: Initialized Chess game");
    }
    protected void validatePlayerMove(Player p){
        System.out.println("Child class: Validate Chess move:" + p.getName());
    }
    protected void logGameStatistics(){
        super.logGameStatistics();
        System.out.println("Child class: Add Chess specific logGameStatistics:");
    }
}
class TicTacToe extends Game{
    public TicTacToe(){
        super();
    }
    public void initializeGame(){
        System.out.println("Child class: Initialized TicTacToe game");
    }
    protected void validatePlayerMove(Player p){
        System.out.println("Child class: Validate TicTacToe move:" + p.getName());
    }
}

public class Polymorphism{
    public static void main(String args[]){
        try{

            Game game = new Chess();
            Thread t1 = new Thread(game);
            t1.start();
            Thread.sleep(1000);
            game.setRunGame(false);
```

```
            Thread.sleep(1000);

            game = new TicTacToe();
            Thread t2 = new Thread(game);
            t2.start();
            Thread.sleep(1000);
            game.setRunGame(false);

        }catch(Exception err){
            err.printStackTrace();
        }
    }
}
```

:

```
Child class: Initialized Chess game
Base class: Defining the flow for Game:
Child class: Validate Chess move:Player 1
Base class: Move Duration: player.PlayerActTime - player.MoveShownTime
Child class: Validate Chess move:Player 2
Base class: Move Duration: player.PlayerActTime - player.MoveShownTime
Base class: logGameStatistics:
Child class: Add Chess specific logGameStatistics:

Child class: Initialized TicTacToe game
Base class: Defining the flow for Game:
Child class: Validate TicTacToe move:Player 1
Base class: Move Duration: player.PlayerActTime - player.MoveShownTime
Child class: Validate TicTacToe move:Player 2
Base class: Move Duration: player.PlayerActTime - player.MoveShownTime
Base class: logGameStatistics:
```

: https://riptutorial.com/ko/java/topic/980/

82:

. Java . .

Java . JUnit. .

JUnit

JUnit4 - *JUnit4* . .

.

TestNG

.

Mockito - ; . .

JBehave - BDD . (/). . .

Examples

?

. . .

. .

() - .

```
public class Example {
  public static void main (String args[]) {
    new Example();
  }

  // Application-level test.
  public Example() {
    Consumer c = new Consumer();
    System.out.println("VALUE = " + c.getVal());
  }

  // Your Module.
  class Consumer {
    private Capitalizer c;

    public Consumer() {
      c = new Capitalizer();
    }
```

```
    public String getVal() {
      return c.getVal();
    }
  }

  // Another team's module.
  class Capitalizer {
    private DataReader dr;

    public Capitalizer() {
      dr = new DataReader();
    }

    public String getVal() {
      return dr.readVal().toUpperCase();
    }
  }

  // Another team's module.
  class DataReader {
    public String readVal() {
      // Refers to a file somewhere in your application deployment, or
      // perhaps retrieved over a deployment-specific network.
      File f;
      String s = "data";
      // ... Read data from f into s ...
      return s;
    }
  }
}
```

. DataReader Capitalizer Consumer . DataReader .

getVal() Capitalizer getVal() toUpperCase() toLowerCase() .

```
  // Another team's module.
  class Capitalizer {
    ...

    public String getVal() {
      return dr.readVal().toLowerCase();
    }
  }
```

. DataReader . / / Consumer ." ? ?", . . . ?

., . ?

.

.

. . 5 . 5 . 5 ..

. . , ( ) .

. , . .

.

. (ping) `Capitalizer` `Capitalizer` .

. Java . . `Capitalizer` JAR . ( ). ( ).

, (,) . () .

`Capitalizer` `getVal()` `Capitalizer` `getVal()` . . (, `toUpperCase()` `toLowerCase()`) .
, , .

( ) . , .

: https://riptutorial.com/ko/java/topic/8155/-

83:

Java 8, `Calendar` java.time . API Calendar .

Examples

`Calendar getInstance()` `GregorianCalendar` .

`Calendar` 1 0 `int` 0 `Calendar` , `Calendar.JANUARY` .

```
Calendar calendar = Calendar.getInstance();
Calendar gregorianCalendar = new GregorianCalendar();
Calendar gregorianCalendarAtSpecificDay = new GregorianCalendar(2016, Calendar.JANUARY, 1);
Calendar gregorianCalendarAtSpecificDayAndTime = new GregorianCalendar(2016, Calendar.JANUARY,
1, 6, 55, 10);
```

: . . `Calendar.JANUARY` 0

/

`add()` `roll()` `Calendar` .

```
Calendar calendar = new GregorianCalendar(2016, Calendar.MARCH, 31); // 31 March 2016
```

`add()` .

```
calendar.add(Calendar.MONTH, -6);
```

6 2015 9 30 .

`roll()` .

```
calendar.roll(Calendar.MONTH, -6);
```

6 9 . . .

/

Calendar AM PM .

```
  Calendar cal = Calendar.getInstance();
  cal.setTime(new Date());
  if (cal.get(Calendar.AM_PM) == Calendar.PM)
      System.out.println("It is PM");
```

2 `Calendar` , `getTimeInMillis()` `getTimeInMillis()` .

```
Calendar c1 = Calendar.getInstance();
Calendar c2 = Calendar.getInstance();
c2.set(Calendar.DATE, c2.get(Calendar.DATE) + 1);

System.out.println(c2.getTimeInMillis() - c1.getTimeInMillis()); //outputs 86400000 (24 * 60 *
60 * 1000)
```

: https://riptutorial.com/ko/java/topic/165/----

84:

Examples

PriorityQueue

PriorityQueue . SortedSet PriorityQueue PriorityQueue . . PriorityQueue comparable comparator .

```
//The type of the PriorityQueue is Integer.
PriorityQueue<Integer> queue = new PriorityQueue<Integer>();

//The elements are added to the PriorityQueue
queue.addAll( Arrays.asList( 9, 2, 3, 1, 3, 8 ) );

//The PriorityQueue sorts the elements by using compareTo method of the Integer Class
//The head of this queue is the least element with respect to the specified ordering
System.out.println( queue );  //The Output: [1, 2, 3, 9, 3, 8]
queue.remove();
System.out.println( queue );  //The Output: [2, 3, 3, 9, 8]
queue.remove();
System.out.println( queue );  //The Output: [3, 8, 3, 9]
queue.remove();
System.out.println( queue );  //The Output: [3, 8, 9]
queue.remove();
System.out.println( queue );  //The Output: [8, 9]
queue.remove();
System.out.println( queue );  //The Output: [9]
queue.remove();
System.out.println( queue );  //The Output: []
```

FIFO LinkedList

java.util.LinkedList java.util.List java.util.Queue FIFO (,) .

offer() LinkedList . enqueue . while FIFO Queue . dequeue .

```
Queue<String> queue = new LinkedList<String>();

queue.offer( "first element" );
queue.offer( "second element" );
queue.offer( "third element" );
queue.offer( "fourth. element" );
queue.offer( "fifth. element" );

while ( !queue.isEmpty() ) {
  System.out.println( queue.poll() );
}
```

.

```
first element
```

```
second element
third element
fourth element
fifth element
```

" " .

?

Java Stacks LIFO (Last In, First Out) .

Stack API

Java Stack API .

```
Stack()            //Creates an empty Stack
isEmpty()          //Is the Stack Empty?            Return Type: Boolean
push(Item item)    //push an item onto the stack
pop()              //removes item from top of stack  Return Type: Item
size()             //returns # of items in stack     Return Type: Int
```

```
import java.util.*;

public class StackExample {

   public static void main(String args[]) {
      Stack st = new Stack();
      System.out.println("stack: " + st);

      st.push(10);
      System.out.println("10 was pushed to the stack");
      System.out.println("stack: " + st);

      st.push(15);
      System.out.println("15 was pushed to the stack");
      System.out.println("stack: " + st);

      st.push(80);
      System.out.println("80 was pushed to the stack");
      System.out.println("stack: " + st);

      st.pop();
      System.out.println("80 was popped from the stack");
      System.out.println("stack: " + st);

      st.pop();
      System.out.println("15 was popped from the stack");
      System.out.println("stack: " + st);

      st.pop();
      System.out.println("10 was popped from the stack");
      System.out.println("stack: " + st);
```

```
     if(st.isEmpty())
      {
         System.out.println("empty stack");
      }
   }
}
```

:

```
stack: []
10 was pushed to the stack
stack: [10]
15 was pushed to the stack
stack: [10, 15]
80 was pushed to the stack
stack: [10, 15, 80]
80 was popped from the stack
stack: [10, 15]
15 was popped from the stack
stack: [10]
10 was popped from the stack
stack: []
empty stack
```

BlockingQueue

BlockingQueue . . . .

BlockingQueue . (null false) .), , .

	.			
	()	(e)	put (e)	(e, ,)
	()	()	()	(,)
	()	()	N / A	N / A

`BlockingQueue` . BlockingQueue, .

```
BlockingQueue<String> bQueue = new ArrayBlockingQueue<String>(2);
```

put () .

Queue . Integer.MAX_VALUE .

`BlockingQueue` .

1. ArrayBlockingQueue
2. LinkedBlockingQueue

3. PriorityBlockingQueue

`ArrayBlockingQueue` .

```
BlockingQueue<String> bQueue = new ArrayBlockingQueue<>(2);
bQueue.put("This is entry 1");
System.out.println("Entry one done");
bQueue.put("This is entry 2");
System.out.println("Entry two done");
bQueue.put("This is entry 3");
System.out.println("Entry three done");
```

.

```
Entry one done
Entry two done
```

.

`Queue` . FIFO ( ) .

. FIFO .

```
public interface Queue<E> extends Collection<E> {
    boolean add(E e);

    boolean offer(E e);

    E remove();

    E poll();

    E element();

    E peek();
}
```

`Queue` .

- Throw.
- other (`null` `false` .

	throw.	.
	`add(e)`	`offer(e)`
	`remove()`	`poll()`
	`element()`	`peek()`

Deque

Deque " ". .

Deque Queue ., , Deque . . .

getFirst()	.
getLast()	.
addFirst(E e)	.
addLast(E e)	.
removeFirst()	.
removeLast()	

offer , poll peek . .

Deque , add() . deque head tail addFirst() addLast() .

```
Deque<String> dequeA = new LinkedList<>();

dequeA.add("element 1");      //add element at tail
dequeA.addFirst("element 2"); //add element at head
dequeA.addLast("element 3");  //add element at tail
```

. element() . Deque x W getFirst() W getLast() . .

```
String firstElement0 = dequeA.element();
String firstElement1 = dequeA.getFirst();
String lastElement = dequeA.getLast();
```

---

remove remove() , removeFirst() removeLast() . .

```
String firstElement = dequeA.remove();
String firstElement = dequeA.removeFirst();
String lastElement  = dequeA.removeLast();
```

: https://riptutorial.com/ko/java/topic/7196/--

85:

, , : Multiset, Bag, Multimap, lib utils .

Examples

Apache HashBag, Guava HashMultiset Eclipse HashBag

Bag / ultiset . . JDK analog HashMap <T, Integer>, .

		Apache Commons Collections	GS	JDK
.	HashMultiset	HashBag	HashBag	HashMap
	TreeMultiset	TreeBag	TreeBag	
	LinkedHashMultiset	-	-	
	HashMultiset	SynchronizedBag	SynchronizedBag	`Collections.synchron`
	-	SynchronizedSortedBag	SynchronizedSortedBag	`Collections.synchron`
		UnmodifiableBag	UnmodifiableBag	`Collections.unmodifi`
	ImmutableSortedMultiset			`Collections.unmodifi`) `Collections.unmodi` `Integer>`

:

1. Apache SynchronizedSortedBag :

```
    // Parse text to separate words
    String INPUT_TEXT = "Hello World! Hello All! Hi World!";
    // Create Multiset
    Bag bag = SynchronizedSortedBag.synchronizedBag(new
TreeBag(Arrays.asList(INPUT_TEXT.split(" "))));

    // Print count words
    System.out.println(bag); // print [1:All!,2:Hello,1:Hi,2:World!]- in natural (alphabet)
order
    // Print all unique words
    System.out.println(bag.uniqueSet());    // print [All!, Hello, Hi, World!]- in natural
(alphabet) order


    // Print count occurrences of words
    System.out.println("Hello = " + bag.getCount("Hello"));    // print 2
    System.out.println("World = " + bag.getCount("World!"));    // print 2
```

```
    System.out.println("All = " + bag.getCount("All!"));    // print 1
    System.out.println("Hi = " + bag.getCount("Hi"));    // print 1
    System.out.println("Empty = " + bag.getCount("Empty"));    // print 0

    // Print count all words
    System.out.println(bag.size());    //print 6

    // Print count unique words
    System.out.println(bag.uniqueSet().size());    //print 4
```

2. Eclipse (GC) TreeBag :

```
    // Parse text to separate words
    String INPUT_TEXT = "Hello World! Hello All! Hi World!";
    // Create Multiset
    MutableSortedBag<String> bag =  TreeBag.newBag(Arrays.asList(INPUT_TEXT.split(" ")));

    // Print count words
    System.out.println(bag); // print [All!, Hello, Hello, Hi, World!, World!]- in natural
order
    // Print all unique words
    System.out.println(bag.toSortedSet());    // print [All!, Hello, Hi, World!]- in natural
order

    // Print count occurrences of words
    System.out.println("Hello = " + bag.occurrencesOf("Hello"));    // print 2
    System.out.println("World = " + bag.occurrencesOf("World!"));    // print 2
    System.out.println("All = " + bag.occurrencesOf("All!"));    // print 1
    System.out.println("Hi = " + bag.occurrencesOf("Hi"));    // print 1
    System.out.println("Empty = " + bag.occurrencesOf("Empty"));    // print 0

    // Print count all words
    System.out.println(bag.size());    //print 6

    // Print count unique words
    System.out.println(bag.toSet().size());    //print 4
```

3. LinkedHashMultiset :

```
    // Parse text to separate words
    String INPUT_TEXT = "Hello World! Hello All! Hi World!";
    // Create Multiset
    Multiset<String> multiset = LinkedHashMultiset.create(Arrays.asList(INPUT_TEXT.split("
")));

    // Print count words
    System.out.println(multiset); // print [Hello x 2, World! x 2, All!, Hi]- in predictable
iteration order
    // Print all unique words
    System.out.println(multiset.elementSet());    // print [Hello, World!, All!, Hi] - in
predictable iteration order

    // Print count occurrences of words
    System.out.println("Hello = " + multiset.count("Hello"));    // print 2
    System.out.println("World = " + multiset.count("World!"));    // print 2
    System.out.println("All = " + multiset.count("All!"));    // print 1
    System.out.println("Hi = " + multiset.count("Hi"));    // print 1
```

```
    System.out.println("Empty = " + multiset.count("Empty"));    // print 0

    // Print count all words
    System.out.println(multiset.size());    //print 6

    // Print count unique words
    System.out.println(multiset.elementSet().size());    //print 4
```

:

I. Apache Collection :

1. HashBag -
2. SynchronizedBag - .
3. SynchronizedSortedBag - -
4. TreeBag -

II. GS /

5. MutableBag -
6. MutableSortedBag -

III.

7. HashMultiset -
8. TreeMultiset -
9. LinkedHashMultiset -
10. ConcurrentHashMultiset -

, Eclipse

- . JDK , HashMap <K, List>, HashMap <K, Set> .

						Eclipse (GS
			HashMap	ArrayList	ArrayListMultimap	`MultiValueMap`	`FastListMulti`
			HashMap	HashSet	HashMultimap	`MultiValueMap. multiValueMap( new HashMap<K, Set>(), HashSet.class);`	`UnifiedSetMul`
			HashMap	TreeSet	`Multimaps. newMultimap( HashMap, Supplier <TreeSet>)`	`MultiValueMap.multiValueMap( new HashMap<K, Set>(), TreeSet.class)`	`TreeSortedSet Multimap`
				ArrayList	LinkedListMultimap	MultiValueMap. multiValueMap (LinkedHashMap <K, List> (), ArrayList.class);	

						Eclipse (GS
				LinkedHashSet	LinkedHashMultimap	`MultiValueMap. multiValueMap(new LinkedHashMap<K, Set>(), LinkedHashSet.class)`
				TreeSet	TreeMultimap	`MultiValueMap. multiValueMap( new TreeMap<K, Set>(),TreeSet.class)`

Multimap

: "Hello World! Hello All! !" MultiMap (: Hello = [0, 2], World! = [1, 5])

1. Apache MultiValueMap

```
    String INPUT_TEXT = "Hello World! Hello All! Hi World!";
    // Parse text to words and index
    List<String> words = Arrays.asList(INPUT_TEXT.split(" "));
    // Create Multimap
    MultiMap<String, Integer> multiMap = new MultiValueMap<String, Integer>();


    // Fill Multimap
    int i = 0;
    for(String word: words) {
        multiMap.put(word, i);
        i++;
    }

    // Print all words
    System.out.println(multiMap); // print {Hi=[4], Hello=[0, 2], World!=[1, 5], All!=[3]} -
in random orders
    // Print all unique words
    System.out.println(multiMap.keySet());    // print [Hi, Hello, World!, All!] - in random
orders

    // Print all indexes
    System.out.println("Hello = " + multiMap.get("Hello"));    // print [0, 2]
    System.out.println("World = " + multiMap.get("World!"));    // print [1, 5]
    System.out.println("All = " + multiMap.get("All!"));    // print [3]
    System.out.println("Hi = " + multiMap.get("Hi"));    // print [4]
    System.out.println("Empty = " + multiMap.get("Empty"));    // print null

    // Print count unique words
    System.out.println(multiMap.keySet().size());    //print 4
```

2. GS / Eclipse Collection HashBiMap

```
    String[] englishWords = {"one", "two", "three","ball","snow"};
    String[] russianWords = {"jeden", "dwa", "trzy", "kula", "snieg"};

    // Create Multiset
    MutableBiMap<String, String> biMap = new HashBiMap(englishWords.length);
    // Create English-Polish dictionary
    int i = 0;
    for(String englishWord: englishWords) {
        biMap.put(englishWord, russianWords[i]);
```

```
        i++;
    }

    // Print count words
    System.out.println(biMap); // print {two=dwa, ball=kula, one=jeden, snow=snieg,
three=trzy} - in random orders
    // Print all unique words
    System.out.println(biMap.keySet());    // print [snow, two, one, three, ball] - in random
orders
    System.out.println(biMap.values());    // print [dwa, kula, jeden, snieg, trzy] - in
random orders

    // Print translate by words
    System.out.println("one = " + biMap.get("one"));    // print one = jeden
    System.out.println("two = " + biMap.get("two"));    // print two = dwa
    System.out.println("kula = " + biMap.inverse().get("kula"));    // print kula = ball
    System.out.println("snieg = " + biMap.inverse().get("snieg"));    // print snieg = snow
    System.out.println("empty = " + biMap.get("empty"));    // print empty = null

    // Print count word's pair
    System.out.println(biMap.size());    //print 5
```

3. HashMultiMap

```
 String INPUT_TEXT = "Hello World! Hello All! Hi World!";
 // Parse text to words and index
 List<String> words = Arrays.asList(INPUT_TEXT.split(" "));
 // Create Multimap
 Multimap<String, Integer> multiMap = HashMultimap.create();

 // Fill Multimap
 int i = 0;
 for(String word: words) {
     multiMap.put(word, i);
     i++;
 }

 // Print all words
 System.out.println(multiMap); // print {Hi=[4], Hello=[0, 2], World!=[1, 5], All!=[3]} -
keys and values in random orders
 // Print all unique words
 System.out.println(multiMap.keySet());    // print [Hi, Hello, World!, All!] - in random
orders

 // Print all indexes
 System.out.println("Hello = " + multiMap.get("Hello"));    // print [0, 2]
 System.out.println("World = " + multiMap.get("World!"));    // print [1, 5]
 System.out.println("All = " + multiMap.get("All!"));    // print [3]
 System.out.println("Hi = " + multiMap.get("Hi"));    // print [4]
 System.out.println("Empty = " + multiMap.get("Empty"));    // print []

 // Print count all words
 System.out.println(multiMap.size());    //print 6

 // Print count unique words
 System.out.println(multiMap.keySet().size());    //print 4
```

:

I. Apache Collection :

1. MultiValueMap
2. MultiValueMapLinked
3. MultiValueMapTree

II. GS /

1. FastListMultimap
2. HashBag
3. TreeSortedSetMultimap
4. UnifiedSetMultimap

III.

1. HashMultiMap
2. LinkedHashMultimap
3. LinkedListMultimap
4. TreeMultimap
5. ArrayListMultimap

-

-

1.

	JDK		GS
	`new ArrayList<> ()`	`Lists.newArrayList()`	`FastList.newList()`
	`Arrays.asList("1", "2", "3")`	`Lists.newArrayList("1", "2", "3")`	`FastList.newListWith("1", "2", "3")`
= 100	`new ArrayList<>(100)`	`Lists.newArrayListWithCapacity(100)`	`FastList.newList(100)`
collectin	`new ArrayList<>(collection)`	`Lists.newArrayList(collection)`	`FastList.newList(collection)`
	-	`Lists.newArrayList(iterable)`	`FastList.newList(iterable)`
Iterator	-	`Lists.newArrayList(iterator)`	-
	`Arrays.asList(array)`	`Lists.newArrayList(array)`	`FastList.newListWith(array)`
	-	-	`FastList.newWithNValues(10, () -> "1")`

:

```
    System.out.println("createArrayList start");
    // Create empty list
```

```
    List<String> emptyGuava = Lists.newArrayList(); // using guava
    List<String> emptyJDK = new ArrayList<>(); // using JDK
    MutableList<String>  emptyGS = FastList.newList(); // using gs

    // Create list with 100 element
    List < String > exactly100 = Lists.newArrayListWithCapacity(100); // using guava
    List<String> exactly100JDK = new ArrayList<>(100); // using JDK
    MutableList<String>  empty100GS = FastList.newList(100); // using gs

    // Create list with about 100 element
    List<String> approx100 = Lists.newArrayListWithExpectedSize(100); // using guava
    List<String> approx100JDK = new ArrayList<>(115); // using JDK
    MutableList<String>  approx100GS = FastList.newList(115); // using gs

    // Create list with some elements
    List<String> withElements = Lists.newArrayList("alpha", "beta", "gamma"); // using guava
    List<String> withElementsJDK = Arrays.asList("alpha", "beta", "gamma"); // using JDK
    MutableList<String>  withElementsGS = FastList.newListWith("alpha", "beta", "gamma"); //
using gs

    System.out.println(withElements);
    System.out.println(withElementsJDK);
    System.out.println(withElementsGS);

    // Create list from any Iterable interface (any collection)
    Collection<String> collection = new HashSet<>(3);
    collection.add("1");
    collection.add("2");
    collection.add("3");

    List<String> fromIterable = Lists.newArrayList(collection); // using guava
    List<String> fromIterableJDK = new ArrayList<>(collection); // using JDK
    MutableList<String>  fromIterableGS = FastList.newList(collection); // using gs

    System.out.println(fromIterable);
    System.out.println(fromIterableJDK);
    System.out.println(fromIterableGS);
    /* Attention: JDK create list only from Collection, but guava and gs can create list from
Iterable and Collection */

    // Create list from any Iterator
    Iterator<String> iterator = collection.iterator();
    List<String> fromIterator = Lists.newArrayList(iterator); // using guava
    System.out.println(fromIterator);

    // Create list from any array
    String[] array = {"4", "5", "6"};
    List<String> fromArray = Lists.newArrayList(array); // using guava
    List<String> fromArrayJDK = Arrays.asList(array); // using JDK
    MutableList<String>  fromArrayGS = FastList.newListWith(array); // using gs
    System.out.println(fromArray);
    System.out.println(fromArrayJDK);
    System.out.println(fromArrayGS);

    // Create list using fabric
    MutableList<String>  fromFabricGS = FastList.newWithNValues(10, () ->
String.valueOf(Math.random())); // using gs
    System.out.println(fromFabricGS);

    System.out.println("createArrayList end");
```

2

	JDK		GS
	`new HashSet<>()`	`Sets.newHashSet()`	`UnifiedSet.newSet()`
	`new HashSet<>(Arrays.asList("alpha", "beta", "gamma"))`	`Sets.newHashSet("alpha", "beta", "gamma")`	`UnifiedSet.newSetWith("alpha", "beta", "gamma")`
	`new HashSet<>(collection)`	`Sets.newHashSet(collection)`	`UnifiedSet.newSet(collection)`
	-	`Sets.newHashSet(iterable)`	`UnifiedSet.newSet(iterable)`
	-	`Sets.newHashSet(iterator)`	-
	`new HashSet<>(Arrays.asList(array))`	`Sets.newHashSet(array)`	`UnifiedSet.newSetWith(array)`

:

```
    System.out.println("createHashSet start");
    // Create empty set
    Set<String> emptyGuava = Sets.newHashSet(); // using guava
    Set<String> emptyJDK = new HashSet<>(); // using JDK
    Set<String> emptyGS = UnifiedSet.newSet(); // using gs

    // Create set with 100 element
    Set<String> approx100 = Sets.newHashSetWithExpectedSize(100); // using guava
    Set<String> approx100JDK = new HashSet<>(130); // using JDK
    Set<String> approx100GS = UnifiedSet.newSet(130); // using gs

    // Create set from some elements
    Set<String> withElements =  Sets.newHashSet("alpha", "beta", "gamma"); // using guava
    Set<String> withElementsJDK = new HashSet<>(Arrays.asList("alpha", "beta", "gamma")); //
using JDK
    Set<String> withElementsGS  = UnifiedSet.newSetWith("alpha", "beta", "gamma"); // using gs

    System.out.println(withElements);
    System.out.println(withElementsJDK);
    System.out.println(withElementsGS);

    // Create set from any Iterable interface (any collection)
    Collection<String> collection = new ArrayList<>(3);
    collection.add("1");
    collection.add("2");
    collection.add("3");

    Set<String> fromIterable = Sets.newHashSet(collection); // using guava
    Set<String> fromIterableJDK = new HashSet<>(collection); // using JDK
    Set<String> fromIterableGS  = UnifiedSet.newSet(collection); // using gs

    System.out.println(fromIterable);
    System.out.println(fromIterableJDK);
    System.out.println(fromIterableGS);
    /* Attention: JDK create set only from Collection, but guava and gs can create set from
Iterable and Collection */

    // Create set from any Iterator
    Iterator<String> iterator = collection.iterator();
    Set<String> fromIterator = Sets.newHashSet(iterator); // using guava
```

```
    System.out.println(fromIterator);

    // Create set from any array
    String[] array = {"4", "5", "6"};
    Set<String> fromArray = Sets.newHashSet(array); // using guava
    Set<String> fromArrayJDK = new HashSet<>(Arrays.asList(array)); // using JDK
    Set<String> fromArrayGS  = UnifiedSet.newSetWith(array); // using gs
    System.out.println(fromArray);
    System.out.println(fromArrayJDK);
    System.out.println(fromArrayGS);

    System.out.println("createHashSet end");
```

3

	JDK		GS
	`new HashMap<>()`	`Maps.newHashMap()`	`UnifiedMap.newMap()`
= 130	`new HashMap<>(130)`	`Maps.newHashMapWithExpectedSize(100)`	`UnifiedMap.newMap(130)`
	`new HashMap<>(map)`	`Maps.newHashMap(map)`	`UnifiedMap.newMap(map)`
	-	-	`UnifiedMap.newWithKeysValues("1", "a", "2", "b")`

:

```
    System.out.println("createHashMap start");
    // Create empty map
    Map<String, String> emptyGuava = Maps.newHashMap(); // using guava
    Map<String, String> emptyJDK = new HashMap<>(); // using JDK
    Map<String, String> emptyGS = UnifiedMap.newMap(); // using gs

    // Create map with about 100 element
    Map<String, String> approx100 = Maps.newHashMapWithExpectedSize(100); // using guava
    Map<String, String> approx100JDK = new HashMap<>(130); // using JDK
    Map<String, String> approx100GS = UnifiedMap.newMap(130); // using gs

    // Create map from another map
    Map<String, String> map = new HashMap<>(3);
    map.put("k1","v1");
    map.put("k2","v2");
    Map<String, String> withMap =  Maps.newHashMap(map); // using guava
    Map<String, String> withMapJDK = new HashMap<>(map); // using JDK
    Map<String, String> withMapGS = UnifiedMap.newMap(map); // using gs

    System.out.println(withMap);
    System.out.println(withMapJDK);
    System.out.println(withMapGS);

    // Create map from keys
    Map<String, String> withKeys =  UnifiedMap.newWithKeysValues("1", "a", "2", "b");
    System.out.println(withKeys);

    System.out.println("createHashMap end");
```

: CreateCollectionTest

1. CollectionCompare
2.
3. JavaTransform

: https://riptutorial.com/ko/java/topic/2958/-

86:

/ I / O .

Examples

BufferedReader

`BufferedReader` `Reader` .

1. `BufferedReader` `Reader` . I / O .
2. `BufferedReader` .

BufferedReader

`BufferedReader` ., `Reader` . `Reader` `BufferedReader` . . `Reader` `BufferedReader` . :

```
File someFile = new File(...);
int aCount = 0;
try (FileReader fr = new FileReader(someFile);
     BufferedReader br = new BufferedReader(fr)) {
    // Count the number of 'a' characters.
    int ch;
    while ((ch = br.read()) != -1) {
        if (ch == 'a') {
            aCount++;
        }
    }
    System.out.println("There are " + aCount + " 'a' characters in " + someFile);
}
```

`Reader` .

:

1. Java 7 ( ) *try-with-resources* . . Java `finally` `BufferedReader` .
2. `try` `FileReader` ., `BufferedReader` `Reader` . .

BufferedReader

BufferedReader.readLine ()

:

List<String> . .

```
public List<String> getAllLines(String filename) throws IOException {
    List<String> lines = new ArrayList<String>();
    try (BufferedReader br = new BufferedReader(new FileReader(filename))) {
        String line = null;
        while ((line = reader.readLine) != null) {
            lines.add(line);
        }
    }
    return lines;
}
```

Java 8 `lines()` .

```
public List<String> getAllLines(String filename) throws IOException {
    try (BufferedReader br = new BufferedReader(new FileReader(filename))) {
        return br.lines().collect(Collectors.toList());
    }
    return Collections.empty();
}
```

StringWriter

Java StringWriter .

StringWriter Writer .

StringWriter StringWriter .

```
import java.io.*;
public class StringWriterDemo {
    public static void main(String[] args) throws IOException {
        char[] ary = new char[1024];
        StringWriter writer = new StringWriter();
        FileInputStream input = null;
        BufferedReader buffer = null;
        input = new FileInputStream("c://stringwriter.txt");
        buffer = new BufferedReader(new InputStreamReader(input, "UTF-8"));
        int x;
        while ((x = buffer.read(ary)) != -1) {
                    writer.write(ary, 0, x);
        }
        System.out.println(writer.toString());
        writer.close();
        buffer.close();
    }
}
```

BufferedReader StringWriter .

: https://riptutorial.com/ko/java/topic/10618/-

87:

(concurrent collection) [collection] [1]. . . [1] :
http://stackoverflow.com/documentation/java/90/collections#t=20161222193649729848 4

Examples

thread-safe

Collection .

.

```
List<String> threadSafeList = Collections.synchronizedList(new ArrayList<String>());
Set<String> threadSafeSet = Collections.synchronizedSet(new HashSet<String>());
Map<String, String> threadSafeMap = Collections.synchronizedMap(new HashMap<String,
String>());
```

.

Java SE 5

Java 5 `java.util.collections` `Collections.synchronized` .

```
List<String> threadSafeList = new CopyOnWriteArrayList<String>();
Set<String> threadSafeSet = new ConcurrentHashSet<String>();
Map<String, String> threadSafeMap = new ConcurrentHashMap<String, String>();
```

.

(/).

.

`addAll` , `removeAll` ( ), , .

, Java SE 5 `java.util.concurrent.CopyOnWriteArrayList` , thread `Lis` , javadoc .

" " iterator iterator . , , , ConcurrentModificationException throw .

.

```
public class ThreadSafeAndConcurrent {

public static final List<Integer> LIST = new CopyOnWriteArrayList<>();

public static void main(String[] args) throws InterruptedException {
    Thread modifier = new Thread(new ModifierRunnable());
    Thread iterator = new Thread(new IteratorRunnable());
    modifier.start();
```

```
    iterator.start();
    modifier.join();
    iterator.join();
}

public static final class ModifierRunnable implements Runnable {
    @Override
    public void run() {
        try {
            for (int i = 0; i < 50000; i++) {
                LIST.add(i);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}

public static final class IteratorRunnable implements Runnable {
    @Override
    public void run() {
        try {
            for (int i = 0; i < 10000; i++) {
                long total = 0;
                for(Integer inList : LIST) {
                    total += inList;
                }
                System.out.println(total);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
}
```

`ConcurrentLinkedQueue` .

, . java.util.ConcurrentModificationException . .

javadocs . `iterator()` ("fail fast", "weakly consistent" `iterator()` `iterator()` .

`LIST`

```
public static final List<Integer> LIST = Collections.synchronizedList(new ArrayList<>());
```

CPU / ( ) .

`Collections` / ( ).

ConcurrentHashMap

```
public class InsertIntoConcurrentHashMap
{
```

```
    public static void main(String[] args)
    {
        ConcurrentHashMap<Integer, SomeObject> concurrentHashMap = new ConcurrentHashMap<>();

        SomeObject value = new SomeObject();
        Integer key = 1;

        SomeObject previousValue = concurrentHashMap.putIfAbsent(1, value);
        if (previousValue != null)
        {
            //Then some other value was mapped to key = 1. 'value' that was passed to
            //putIfAbsent method is NOT inserted, hence, any other thread which calls
            //concurrentHashMap.get(1) would NOT receive a reference to the 'value'
            //that your thread attempted to insert. Decide how you wish to handle
            //this situation.
        }

       else
       {
            //'value' reference is mapped to key = 1.
       }
    }
}
```

: https://riptutorial.com/ko/java/topic/8363/-

88: ()

. Java . . .

StackOverflow :

-
- Executor, ExecutorService Thread
- `Thread` `Runnable`

Examples

. .

```
class CountAndPrint implements Runnable {

    private final String name;

    CountAndPrint(String name) {
        this.name = name;
    }

    /** This is what a CountAndPrint will do */
    @Override
    public void run() {
        for (int i = 0; i < 10000; i++) {
            System.out.println(this.name + ": " + i);
        }
    }

    public static void main(String[] args) {
        // Launching 4 parallel threads
        for (int i = 1; i <= 4; i++) {
            // `start` method will call the `run` method
            // of CountAndPrint in another thread
            new Thread(new CountAndPrint("Instance " + i)).start();
        }

        // Doing some others tasks in the main Thread
        for (int i = 0; i < 10000; i++) {
            System.out.println("Main: " + i);
        }
    }
}
```

`CountAndPrint` run . .

```
Instance 4: 1
Instance 2: 1
Instance 4: 2
Instance 1: 1
Instance 1: 2
Main: 1
Instance 4: 3
```

```
Main: 2
Instance 3: 1
Instance 4: 4
...
```

-

- . JDK (`AtomicBoolean` `BlockingQueue`) . BlockingQueue Javadoc . (DelayQueue Priority Queue) .

```
public class Producer implements Runnable {

    private final BlockingQueue<ProducedData> queue;

    public Producer(BlockingQueue<ProducedData> queue) {
        this.queue = queue;
    }

    public void run() {
        int producedCount = 0;
        try {
            while (true) {
                producedCount++;
                //put throws an InterruptedException when the thread is interrupted
                queue.put(new ProducedData());
            }
        } catch (InterruptedException e) {
            // the thread has been interrupted: cleanup and exit
            producedCount--;
            //re-interrupt the thread in case the interrupt flag is needeed higher up
            Thread.currentThread().interrupt();
        }
        System.out.println("Produced " + producedCount + " objects");
    }
}

public class Consumer implements Runnable {

    private final BlockingQueue<ProducedData> queue;

    public Consumer(BlockingQueue<ProducedData> queue) {
        this.queue = queue;
    }

    public void run() {
        int consumedCount = 0;
        try {
            while (true) {
                //put throws an InterruptedException when the thread is interrupted
                ProducedData data = queue.poll(10, TimeUnit.MILLISECONDS);
                // process data
                consumedCount++;
            }
        } catch (InterruptedException e) {
            // the thread has been interrupted: cleanup and exit
            consumedCount--;
            //re-interrupt the thread in case the interrupt flag is needeed higher up
            Thread.currentThread().interrupt();
        }
```

```
        System.out.println("Consumed " + consumedCount + " objects");
    }
}


public class ProducerConsumerExample {
    static class ProducedData {
        // empty data object
    }

    public static void main(String[] args) throws InterruptedException {
        BlockingQueue<ProducedData> queue = new ArrayBlockingQueue<ProducedData>(1000);
        // choice of queue determines the actual behavior: see various BlockingQueue
implementations

        Thread producer = new Thread(new Producer(queue));
        Thread consumer = new Thread(new Consumer(queue));

        producer.start();
        consumer.start();

        Thread.sleep(1000);
        producer.interrupt();
        Thread.sleep(10);
        consumer.interrupt();
    }
}
```

ThreadLocal

Java `ThreadLocal` . . .

, (:) . .

```
private static final ThreadLocal<MyUserContext> contexts = new ThreadLocal<>();

public static MyUserContext getContext() {
    return contexts.get(); // get returns the variable unique to this thread
}

public void doGet(...) {
    MyUserContext context = magicGetContextFromRequest(request);
    contexts.put(context); // save that context to our thread-local - other threads
                           // making this call don't overwrite ours
    try {
        // business logic
    } finally {
        contexts.remove(); // 'ensure' removal of thread-local variable
    }
}
```

`MyUserContext` `MyServlet.getContext()` . .

`contexts` . ( ) .

CountDownLatch

CountDownLatch

.

1. `CountDownLatch` .
2. await `countDown()` 0 . .
3. . . `CyclicBarrier` .

:

```
public void await() throws InterruptedException
```

0 .

```
public void countDown()
```

, 0 .

:

```
import java.util.concurrent.*;

class DoSomethingInAThread implements Runnable {
    CountDownLatch latch;
    public DoSomethingInAThread(CountDownLatch latch) {
        this.latch = latch;
    }
    public void run() {
        try {
            System.out.println("Do some thing");
            latch.countDown();
        } catch(Exception err) {
            err.printStackTrace();
        }
    }
}

public class CountDownLatchDemo {
    public static void main(String[] args) {
        try {
            int numberOfThreads = 5;
            if (args.length < 1) {
                System.out.println("Usage: java CountDownLatchDemo numberOfThreads");
                return;
            }
            try {
                numberOfThreads = Integer.parseInt(args[0]);
            } catch(NumberFormatException ne) {

            }
            CountDownLatch latch = new CountDownLatch(numberOfThreads);
            for (int n = 0; n < numberOfThreads; n++) {
                Thread t = new Thread(new DoSomethingInAThread(latch));
                t.start();
            }
            latch.await();
```

```
            System.out.println("In Main thread after completion of " + numberOfThreads + "
threads");
        } catch(Exception err) {
            err.printStackTrace();
        }
    }
}
```

:

```
java CountDownLatchDemo 5
Do some thing
Do some thing
Do some thing
Do some thing
Do some thing
In Main thread after completion of 5 threads
```

:

1. `CountDownLatch` 5 .
2. `await()` `await()` .
3. `DoSomethingInAThread` . `countDown()` .
4. 0 .

Java . , Java `synchronized` (, Java).

. `synchronized` .

```
private static int t = 0;
private static Object mutex = new Object();

public static void main(String[] args) {
    ExecutorService executorService = Executors.newFixedThreadPool(400); // The high thread
count is for demonstration purposes.
    for (int i = 0; i < 100; i++) {
        executorService.execute(() -> {
            synchronized (mutex) {
                t++;
                System.out.println(MessageFormat.format("t: {0}", t));
            }
        });
    }
    executorService.shutdown();
}
```

`synchronized` . ( ) t . 1 100 .

Java (*mutex)* (:). . : () . , .

Java . , . .

```
public void bar(){
    synchronized(this){
```

```
        ...
    }
}
public void foo(){
    synchronized(this){
        bar();
    }
}
```

`synchronized` `synchronized` `synchronized` .

( ).

1. `synchronized` `this` :

```
public void foo() {
    synchronized(this) {
        doStuff();
    }
}
```

2. `synchronized` :

```
public synchronized void foo() {
    doStuff();
}
```

`static` .

```
class MyClass {
    ...
    public static void bar() {
        synchronized(MyClass.class) {
            doSomeOtherStuff();
        }
    }
}
```

.

```
class MyClass {
    ...
    public static synchronized void bar() {
        doSomeOtherStuff();
    }
}
```

"" .

.

```
private static int t = 0;
```

```
public static void main(String[] args) {
    ExecutorService executorService = Executors.newFixedThreadPool(400); // The high thread
count is for demonstration purposes.
    for (int i = 0; i < 100; i++) {
        executorService.execute(() -> {
            t++;
            System.out.println(MessageFormat.format("t: {0}", t));
        });
    }
    executorService.shutdown();
}
```

. . . 1 . t: 100 . . , t 10 . t t 11 .

t . t .

java.util.concurrent.atomic.AtomicInteger . java.util.concurrent.atomic.AtomicInteger .

```
private static AtomicInteger t = new AtomicInteger(0);

public static void main(String[] args) {
    ExecutorService executorService = Executors.newFixedThreadPool(400); // The high thread
count is for demonstration purposes.
    for (int i = 0; i < 100; i++) {
        executorService.execute(() -> {
            int currentT = t.incrementAndGet();
            System.out.println(MessageFormat.format("t: {0}", currentT));
        });
    }
    executorService.shutdown();
}
```

AtomicInteger incrementAndGet . println println . AtomicInteger .

. java . synchronized . synchronized .

2 1 , 1 2 , 2 R1 R2 . R1 R2 Second R2 R1 .

t = 0 ,

R1 R2. R2 R2 . .

```
public class Example2 {

    public static void main(String[] args) throws InterruptedException {
        final DeadLock dl = new DeadLock();
        Thread t1 = new Thread(new Runnable() {

            @Override
            public void run() {
                // TODO Auto-generated method stub
                dl.methodA();
            }
        });

        Thread t2 = new Thread(new Runnable() {
```

```
            @Override
            public void run() {
                // TODO Auto-generated method stub
                try {
                    dl.method2();
                } catch (InterruptedException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                }
            }
        });
        t1.setName("First");
        t2.setName("Second");
        t1.start();
        t2.start();
    }
}

class DeadLock {

    Object mLock1 = new Object();
    Object mLock2 = new Object();


    public void methodA() {
        System.out.println("methodA wait for mLock1  " + Thread.currentThread().getName());
        synchronized (mLock1) {
            System.out.println("methodA mLock1 acquired   " +
Thread.currentThread().getName());
            try {
                Thread.sleep(100);
                method2();
            } catch (InterruptedException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
        }
    }
    public void method2() throws InterruptedException {
        System.out.println("method2 wait for mLock2  " + Thread.currentThread().getName());
        synchronized (mLock2) {
            System.out.println("method2  mLock2 acquired   " +
Thread.currentThread().getName());
            Thread.sleep(100);
            method3();
        }
    }
    public void method3() throws InterruptedException {
        System.out.println("method3  mLock1  "+ Thread.currentThread().getName());
        synchronized (mLock1) {
            System.out.println("method3   mLock1 acquired  " +
Thread.currentThread().getName());
        }
    }
}
```

:

```
methodA wait for mLock1  First
```

```
method2 wait for mLock2  Second
method2  mLock2 acquired   Second
methodA mLock1 acquired   First
method3  mLock1  Second
method2 wait for mLock2  First
```

`Thread.sleep` . . Thread `sleep` .

```
public static void sleep(long millis) throws InterruptedException
```

```
public static void sleep(long millis, int nanos)
```

1

```
Thread.sleep(1000);
```

. , ( ) .

try / catch `Thread.sleep` `InterruptedException` catch .

/ /

`synchronized` . `synchronized volatile` . / . / ? .

```
class Counter {

  private Integer count = 10;

  public synchronized void incrementCount() {
    count++;
  }

  public Integer getCount() {
    return count;
  }
}
```

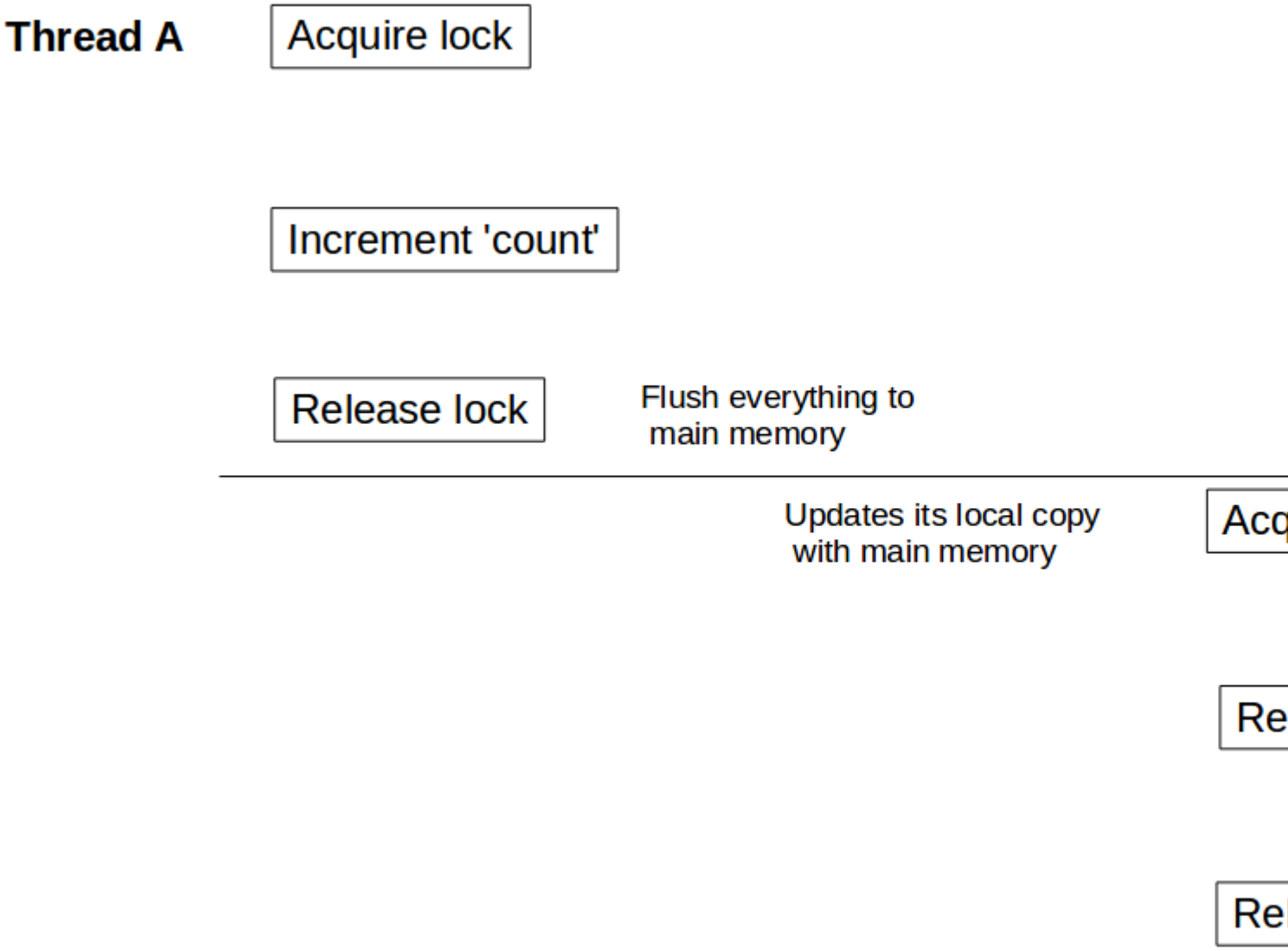
A `incrementCount()` *B* `getCount()` . B `count` . `count` `10` . `count` `count` .

Java . Java . : . . / . `synchronized` (`volatile`) `synchronized` `synchronized` . / .

B `count` `count` A . B count `getCount()` .

```
public synchronized Integer getCount() {
  return count;
}
```

A `count` `Counter` . B `Counter` `count` .



`volatile` / . `volatile` `volatile` .

java.lang.Thread

Java . . .

(java.lang.Thread) . *run ()* .

: java.lang.Thread .

```
class MyThread extends Thread {
    @Override
    public void run() {
        for (int i = 0; i < 10; i++) {
            System.out.println("Thread running!");
        }
    }
}
```

.

```
MyThread t = new MyThread();
```

Thread , . .

```
class MyThread extends Thread {
    public MyThread(String name) {
        super(name);
    }

    @Override
    public void run() {
        for (int i = 0; i < 10; i++) {
            System.out.println("Thread running! ");
        }
    }
}

MyThread t = new MyThread("Greeting Producer");
```

java.lang.Runnable *run ()* . Thread . , Runnable thread .

```
Thread t = new Thread(aRunnable);
```

(Java 8) .

```
Thread t = new Thread(operator::hardWork);
```

.

```
Thread t = new Thread(operator::hardWork, "Pi operator");
```

Practicaly , . .

, java.lang.ThreadGroup .

```
ThreadGroup tg = new ThreadGroup("Operators");
Thread t = new Thread(tg, operator::hardWork, "PI operator");
```

ThreadGroup . ThreadGroup . .

sumarize Thread public :

```
Thread()
Thread(String name)
Thread(Runnable target)
Thread(Runnable target, String name)
Thread(ThreadGroup group, String name)
Thread(ThreadGroup group, Runnable target)
Thread(ThreadGroup group, Runnable target, String name)
Thread(ThreadGroup group, Runnable target, String name, long stackSize)
```

.

. java.util.concurrent.ThreadFactory . *newThread (Runnable)* .

```
class WorkerFactory implements ThreadFactory {
    private int id = 0;

    @Override
    public Thread newThread(Runnable r) {
        return new Thread(r, "Worker " + id++);
    }
}
```

/

Java Thread , false. . . . ? . , . . thread WAITING TIMED_WAITING . InterruptedException throw RUNNABLE. InterruptedException . WAIT . (: IO) . , (,), InterruptedException .

Java . . . . (,) . , . InterruptedException . , InterruptedException . . . . InterruptedException . . . .

:

```
class TaskHandler implements Runnable {

    private final BlockingQueue<Task> queue;

    TaskHandler(BlockingQueue<Task> queue) {
        this.queue = queue;
    }

    @Override
    public void run() {
        while (!Thread.currentThread().isInterrupted()) { // check for interrupt flag, exit
loop when interrupted
            try {
                Task task = queue.take(); // blocking call, responsive to interruption
                handle(task);
            } catch (InterruptedException e) {
                Thread.currentThread().interrupt(); // cannot throw InterruptedException (due
to Runnable interface restriction) so indicating interruption by setting the flag
            }
        }
    }

    private void handle(Task task) {
        // actual handling
    }
}
```

:

```
class MustFinishHandler implements Runnable {

    private final BlockingQueue<Task> queue;
```

```
    MustFinishHandler(BlockingQueue<Task> queue) {
        this.queue = queue;
    }

    @Override
    public void run() {
        boolean shouldInterrupt = false;

        while (true) {
            try {
                Task task = queue.take();
                if (task.isEndOfTasks()) {
                    if (shouldInterrupt) {
                        Thread.currentThread().interrupt();
                    }
                    return;
                }
                handle(task);
            } catch (InterruptedException e) {
                shouldInterrupt = true; // must finish, remember to set interrupt flag when
we're done
            }
        }
    }

    private void handle(Task task) {
        // actual handling
    }
}
```

```
class GetAsFarAsPossible implements Runnable {

    private final List<Task> tasks = new ArrayList<>();

    @Override
    public void run() {
        for (Task task : tasks) {
            if (Thread.currentThread().isInterrupted()) {
                return;
            }
            handle(task);
        }
    }

    private void handle(Task task) {
        // actual handling
    }
}
```

/

Producer / Consumer . Producer Consumer .

```
import java.util.concurrent.*;
import java.util.Random;

public class ProducerConsumerWithES {
```

```
    public static void main(String args[]) {
        BlockingQueue<Integer> sharedQueue = new LinkedBlockingQueue<Integer>();

        ExecutorService pes = Executors.newFixedThreadPool(2);
        ExecutorService ces = Executors.newFixedThreadPool(2);

        pes.submit(new Producer(sharedQueue, 1));
        pes.submit(new Producer(sharedQueue, 2));
        ces.submit(new Consumer(sharedQueue, 1));
        ces.submit(new Consumer(sharedQueue, 2));

        pes.shutdown();
        ces.shutdown();
    }
}

/* Different producers produces a stream of integers continuously to a shared queue,
which is shared between all Producers and consumers */

class Producer implements Runnable {
    private final BlockingQueue<Integer> sharedQueue;
    private int threadNo;
    private Random random = new Random();
    public Producer(BlockingQueue<Integer> sharedQueue,int threadNo) {
        this.threadNo = threadNo;
        this.sharedQueue = sharedQueue;
    }
    @Override
    public void run() {
        // Producer produces a continuous stream of numbers for every 200 milli seconds
        while (true) {
            try {
                int number = random.nextInt(1000);
                System.out.println("Produced:" + number + ":by thread:"+ threadNo);
                sharedQueue.put(number);
                Thread.sleep(200);
            } catch (Exception err) {
                err.printStackTrace();
            }
        }
    }
}
/* Different consumers consume data from shared queue, which is shared by both producer and
consumer threads */
class Consumer implements Runnable {
    private final BlockingQueue<Integer> sharedQueue;
    private int threadNo;
    public Consumer (BlockingQueue<Integer> sharedQueue,int threadNo) {
        this.sharedQueue = sharedQueue;
        this.threadNo = threadNo;
    }
    @Override
    public void run() {
        // Consumer consumes numbers generated from Producer threads continuously
        while(true){
            try {
                int num = sharedQueue.take();
                System.out.println("Consumed: "+ num + ":by thread:"+threadNo);
            } catch (Exception err) {
               err.printStackTrace();
            }
```

```
        }
    }
}
```

:

```
Produced:69:by thread:2
Produced:553:by thread:1
Consumed: 69:by thread:1
Consumed: 553:by thread:2
Produced:41:by thread:2
Produced:796:by thread:1
Consumed: 41:by thread:1
Consumed: 796:by thread:2
Produced:728:by thread:2
Consumed: 728:by thread:1
```

.....

:

1. `LinkedBlockingQueue sharedQueue` Producer Consumer .
2. 200 `sharedQueue`
3. `Consumer sharedQueue` .
4. `synchronized Lock` . BlockingQueue .

   BlockingQueue - .

   BlockingQueue . .

/

"" .

`ReadWriteLock ReentrantReadWriteLock` .

1. . .
2. . .

.

```
import java.util.concurrent.locks.ReadWriteLock;
import java.util.concurrent.locks.ReentrantReadWriteLock;
public class Sample {

// Our lock. The constructor allows a "fairness" setting, which guarantees the chronology of
lock attributions.
protected static final ReadWriteLock RW_LOCK = new ReentrantReadWriteLock();

// This is a typical data that needs to be protected for concurrent access
protected static int data = 0;

/** This will write to the data, in an exclusive access */
public static void writeToData() {
```

```
    RW_LOCK.writeLock().lock();
    try {
        data++;
    } finally {
        RW_LOCK.writeLock().unlock();
    }
}

public static int readData() {
    RW_LOCK.readLock().lock();
    try {
        return data;
    } finally {
        RW_LOCK.readLock().unlock();
    }
}

}
```

NOTE 1 : `AtomicInteger` . Atomic variant .

2 : , . .

1. JVM , `data` 64 long , 64 write 32
2. *Happen Before* JVM .

Java SE 8

, `StampedLock` `StampedLock` , . `ReadWriteLock` `ReadWriteLock` .

Runnable

`Runnable` `run()` .

`Runnable` `Thread` . Thread `start()` .

```
public class HelloRunnable implements Runnable {

    @Override
    public void run() {
        System.out.println("Hello from a thread");
    }

    public static void main(String[] args) {
        new Thread(new HelloRunnable()).start();
    }
}
```

Java8 :

```
public static void main(String[] args) {
    Runnable r = () -> System.out.println("Hello world");
    new Thread(r).start();
}
```

Runnable Thread

`Runnable` , `Runnable` `Thread` .

`Thread` `Thread` .

`Runnable` API .

. . `0` ( ).

.

```
Semaphore semaphore = new Semaphore(1); // The int value being the number of permits
```

Semaphore . false (), thread (permit) . true (), acquire thread, () . .

```
Semaphore semaphore = new Semaphore(1, true);
```

javadoc . `getItem()` .

```
class Pool {
    /*
     * Note that this DOES NOT bound the amount that may be released!
     * This is only a starting value for the Semaphore and has no other
     * significant meaning UNLESS you enforce this inside of the
     * getNextAvailableItem() and markAsUnused() methods
     */
    private static final int MAX_AVAILABLE = 100;
    private final Semaphore available = new Semaphore(MAX_AVAILABLE, true);

    /**
     * Obtains the next available item and reduces the permit count by 1.
     * If there are no items available, block.
     */
    public Object getItem() throws InterruptedException {
        available.acquire();
        return getNextAvailableItem();
    }

    /**
     * Puts the item into the pool and add 1 permit.
     */
    public void putItem(Object x) {
        if (markAsUnused(x))
            available.release();
    }

    private Object getNextAvailableItem() {
        // Implementation
    }

    private boolean markAsUnused(Object o) {
        // Implementation
    }
}
```

`int` .

.

Threadpool `int` .

Java SE 8

```
int[] firstArray = { 2, 4, 6, 8 };
int[] secondArray = { 1, 3, 5, 7 };
int[] result = { 0, 0, 0, 0 };

ExecutorService pool = Executors.newCachedThreadPool();

// Setup the ThreadPool:
// for each element in the array, submit a worker to the pool that adds elements
for (int i = 0; i < result.length; i++) {
    final int worker = i;
    pool.submit(() -> result[worker] = firstArray[worker] + secondArray[worker] );
}

// Wait for all Workers to finish:
try {
    // execute all submitted tasks
    pool.shutdown();
    // waits until all workers finish, or the timeout ends
    pool.awaitTermination(12, TimeUnit.SECONDS);
}
catch (InterruptedException e) {
    pool.shutdownNow(); //kill thread
}

System.out.println(Arrays.toString(result));
```

:

1. . . .

2. Java 7 .

:

```
import java.util.Set;

public class ThreadStatus {
    public static void main(String args[]) throws Exception {
        for (int i = 0; i < 5; i++){
            Thread t = new Thread(new MyThread());
            t.setName("MyThread:" + i);
            t.start();
        }
        int threadCount = 0;
        Set<Thread> threadSet = Thread.getAllStackTraces().keySet();
        for (Thread t : threadSet) {
            if (t.getThreadGroup() == Thread.currentThread().getThreadGroup()) {
                System.out.println("Thread :" + t + ":" + "state:" + t.getState());
                ++threadCount;
            }
        }
        System.out.println("Thread count started by Main thread:" + threadCount);
```

```
    }
}

class MyThread implements Runnable {
    public void run() {
        try {
            Thread.sleep(2000);
        } catch(Exception err) {
            err.printStackTrace();
        }
    }
}
```

:

```
Thread :Thread[MyThread:1,5,main]:state:TIMED_WAITING
Thread :Thread[MyThread:3,5,main]:state:TIMED_WAITING
Thread :Thread[main,5,main]:state:RUNNABLE
Thread :Thread[MyThread:4,5,main]:state:TIMED_WAITING
Thread :Thread[MyThread:0,5,main]:state:TIMED_WAITING
Thread :Thread[MyThread:2,5,main]:state:TIMED_WAITING
Thread count started by Main thread:6
```

:

`Thread.getAllStackTraces().keySet()` `Thread.` `Thread .`

ThreadGroup System Threads .

```
Reference Handler
Signal Dispatcher
Attach Listener
Finalizer
```

`Runnable` . `Runnable` `Runnable` .

`Callable` Java 5 `Runnable` (peer) . `Callable` `run` `call` . `call` .

.

`Future` `Callable` . `Future` .

```
public interface Callable<V> {
    V call() throws Exception;
}
```

```
interface Future<V> {
    V get();
    V get(long timeout, TimeUnit unit);
    boolean cancel(boolean mayInterruptIfRunning);
    boolean isCancelled();
    boolean isDone();
}
```

:

```
public static void main(String[] args) throws Exception {
    ExecutorService es = Executors.newSingleThreadExecutor();

    System.out.println("Time At Task Submission : " + new Date());
    Future<String> result = es.submit(new ComplexCalculator());
    // the call to Future.get() blocks until the result is available.So we are in for about a
10 sec wait now
    System.out.println("Result of Complex Calculation is : " + result.get());
    System.out.println("Time At the Point of Printing the Result : " + new Date());
}
```

Callable

```
public class ComplexCalculator implements Callable<String> {

    @Override
    public String call() throws Exception {
        // just sleep for 10 secs to simulate a lengthy computation
        Thread.sleep(10000);
        System.out.println("Result after a lengthy 10sec calculation");
        return "Complex Result"; // the result
    }
}
```

```
Time At Task Submission : Thu Aug 04 15:05:15 EDT 2016
Result after a lengthy 10sec calculation
Result of Complex Calculation is : Complex Result
Time At the Point of Printing the Result : Thu Aug 04 15:05:25 EDT 2016
```

`get()` Future .

- `get(long timeout, TimeUnit unit)` .
- `cancel(mayInterruptIfRunning)` . mayInterrupt .
- `isDone()` / .
- `isCancelled()` `isCancelled()` .

Java 5 . / .

.

```
int count = 0; // shared among multiple threads

public void doSomething() {
    synchronized(this) {
        ++count; // a non-atomic operation
    }
}
```

```
int count = 0; // shared among multiple threads

Lock lockObj = new ReentrantLock();
public void doSomething() {
```

```
    try {
        lockObj.lock();
        ++count; // a non-atomic operation
    } finally {
        lockObj.unlock(); // sure to release the lock without fail
    }
}
```

.

```
class Locky {
    int count = 0; // shared among multiple threads

    Lock lockObj = new ReentrantLock();

    public void doSomething() {
        try {
            try {
                lockObj.lockInterruptibly();
                ++count; // a non-atomic operation
            } catch (InterruptedException e) {
                Thread.currentThread().interrupt(); // stopping
            }
        } finally {
            if (!Thread.currentThread().isInterrupted()) {
                lockObj.unlock(); // sure to release the lock without fail
            }
        }
    }
}
```

.

```
public class Locky2 {
    int count = 0; // shared among multiple threads

    Lock lockObj = new ReentrantLock();

    public void doSomething() {
        boolean locked = lockObj.tryLock(); // returns true upon successful lock
        if (locked) {
            try {
                ++count; // a non-atomic operation
            } finally {
                lockObj.unlock(); // sure to release the lock without fail
            }
        }
    }
}
```

. API .

() : https://riptutorial.com/ko/java/topic/121/----

89:

. . , . .

. .

- () -> {return expression; } // 0 .
- () -> expression // ; .
- () -> {function-body} // .
- parameterName -> expression // - . .
- (type parameterName, Type secondParameterName, ...) -> expression // lambda
- (parameterName, secondParameterName, ...) -> expression // . .

Examples

(8) , `java.util.Comparator` (1) ( ) :

Java SE 1.2

```
List<Person> people = ...
Collections.sort(
    people,
    new Comparator<Person>() {
        public int compare(Person p1, Person p2){
            return p1.getFirstName().compareTo(p2.getFirstName());
        }
    }
);
```

Java 8 . `p1 p2` .

```
Collections.sort(
    people,
    (p1, p2) -> p1.getFirstName().compareTo(p2.getFirstName())
);
```

`Comparator.comparing` :: ( ) .

```
Collections.sort(
    people,
    Comparator.comparing(Person::getFirstName)
);
```

.

```
import static java.util.Collections.sort;
import static java.util.Comparator.comparing;
//...
```

```
sort(people, comparing(Person::getFirstName));
```

. thenComparing .

```
sort(people, comparing(Person::getFirstName).thenComparing(Person::getLastName));
```

---

1 - Collections.sort (...) `List` . `Set` `Collection` API .

`HashMap` `HashMap` . `LinkedHashMap` . `HashMap` .

```
Map<String, Integer> map = new HashMap();  // ... or any other Map class
// populate the map
map = map.entrySet()
    .stream()
    .sorted(Map.Entry.<String, Integer>comparingByValue())
    .collect(Collectors.toMap(k -> k.getKey(), v -> v.getValue(),
                              (k, v) -> k, LinkedHashMap::new));
```

Java lambdas

(Lambdas) . `default` `static` . (Single Abstract Method Interfaces SAM Interfaces).

```
interface Foo1 {
    void bar();
}

interface Foo2 {
    int bar(boolean baz);
}

interface Foo3 {
    String bar(Object baz, int mink);
}

interface Foo4 {
    default String bar() { // default so not counted
        return "baz";
    }
    void quux();
}
```

`@FunctionalInterface` . , .

```
@FunctionalInterface
interface Foo5 {
    void bar();
}

@FunctionalInterface
interface BlankFoo1 extends Foo3 { // inherits abstract method from Foo3
}
```

```
@FunctionalInterface
interface Foo6 {
    void bar();
    boolean equals(Object obj); // overrides one of Object's method so not counted
}
```

, :

```
interface BadFoo {
    void bar();
    void quux(); // <-- Second method prevents lambda: which one should
                 // be considered as lambda?
}
```

.

```
interface BlankFoo2 { }
```

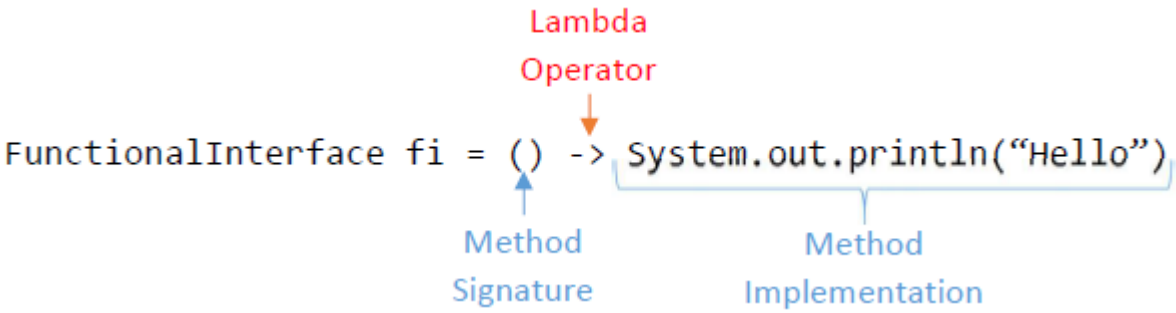
. .

```
interface Parent { public int parentMethod(); }

interface Child extends Parent { public int ChildMethod(); }
```

Child .

Java 8 java.util.function . , Predicate<T> T boolean .

.



fi FunctionalInterface , { System.out.println("Hello"); } . .

```
FunctionalInterface fi = new FunctionalInterface() {
    @Override
    public void theOneMethod() {
        System.out.println("Hello");
    }
};
```

this , super toString() " ".

. Java

(:) . .

```
Object fooHolder = (Foo1) () -> System.out.println("Hello");
System.out.println(fooHolder instanceof Foo1); // returns true
```

. (U). .

```
Foo2 longFoo = new Foo2() {
    @Override
    public int bar(boolean baz) {
        return baz ? 1 : 0;
    }
};
Foo2 shortFoo = (x) -> { return x ? 1 : 0; };
```

} .

```
Foo2 np = x -> { return x ? 1 : 0; }; // okay
Foo3 np2 = x, y -> x.toString() + y // not okay
```

Java . 2 .

```
IntUnaryOperator addOneShort = (x) -> (x + 1);
IntUnaryOperator addOneLong = (x) -> { return (x + 1); };
```

( )

, . final .

```
IntUnaryOperator makeAdder(int amount) {
    return (x) -> (x + amount); // Legal even though amount will go out of scope
                                // because amount is not modified
}

IntUnaryOperator makeAccumulator(int value) {
    return (x) -> { value += x; return value; }; // Will not compile
}
```

, . Java Closure .

: .

```
public void passMeALambda(Foo1 f) {
```

```
    f.bar();
}
passMeALambda(() -> System.out.println("Lambda called"));
```

. . lambda .

- (: `myPredicate = s -> s.isEmpty()`
- `stream.filter(s -> s.isEmpty())` : `stream.filter(s -> s.isEmpty())`
- (: `return s -> s.isEmpty()`
- (: `(Predicate<String>) s -> s.isEmpty()`

. , `o -> o.isEmpty()` . λ .

```
Predicate<String> javaStringPred = o -> o.isEmpty();
Function<String, Boolean> javaFunc = o -> o.isEmpty();
Predicate<List> javaListPred = o -> o.isEmpty();
Consumer<String> javaStringConsumer = o -> o.isEmpty(); // return value is ignored!
com.google.common.base.Predicate<String> guavaPredicate = o -> o.isEmpty();
```

, .

.

.

```
class Person {
    private final String name;
    private final String surname;

    public Person(String name, String surname){
        this.name = name;
        this.surname = surname;
    }

    public String getName(){ return name; }
    public String getSurname(){ return surname; }
}

List<Person> people = getSomePeople();
```

( )

```
people.stream().map(Person::getName)
```

:

```
people.stream().map(person -> person.getName())
```

`Person getName()` . .

( )

```
people.forEach(System.out::println);
```

System.out PrintStream , .

:

```
people.forEach(person -> System.out.println(person));
```

.

```
List<Integer> numbers = Arrays.asList(1, 2, 3, 4, 5, 6);
numbers.stream().map(String::valueOf)
```

String valueOf() . valueOf() .

:

```
 numbers.stream().map(num -> String.valueOf(num))
```

```
List<String> strings = Arrays.asList("1", "2", "3");
strings.stream().map(Integer::new)
```

.

Integer String . . :

```
strings.stream().map(s -> new Integer(s));
```

	TypeName::method	(args) -> TypeName.method(args)
(*)	instance::method	(args) -> instance.method(args)
( )	TypeName::method	(instance, args) -> instance.method(args)
**	TypeName::new	(args) -> new TypeName(args)
	TypeName[]::new	(int size) -> new TypeName[size]

---

* instance instance getInstance()::method : getInstance()::method , this::method

** TypeName ,

. (: java.io.Serializable) .

, TreeSet Comparator . :

```
TreeSet<Long> ts = new TreeSet<>((x, y) -> Long.compare(y, x));
```

Serializable . .

```
TreeSet<Long> ts = new TreeSet<>(
    (Comparator<Long> & Serializable) (x, y) -> Long.compare(y, x));
```

Apache Spark .

```
public interface SerializableComparator extends Comparator<Long>, Serializable {}

public class CustomTreeSet {
  public CustomTreeSet(SerializableComparator comparator) {}
}
```

.

lambdas FunctionalInterface . Execute-Around . / . io, io, try / catch .

```
interface DataProcessor {
    void process( Connection connection ) throws SQLException;;
}

public void doProcessing( DataProcessor processor ) throws SQLException{
    try (Connection connection = DBUtil.getDatabaseConnection();) {
        processor.process(connection);
        connection.commit();
    }
}
```

(lambda) .

```
public static void updateMyDAO(MyVO vo) throws DatabaseException {
    doProcessing((Connection conn) -> MyDAO.update(conn, ObjectMapper.map(vo)));
}
```

I / O . / . DRY ( ) .

Lambda . .

, Runnable , Thread () .

```
//Old way
new Thread(
        new Runnable(){
            public void run(){
                System.out.println("run logic...");
            }
```

```
        }
).start();

//lambdas, from Java 8
new Thread(
        ()-> System.out.println("run logic...")
).start();
```

.

```
interface TwoArgInterface {
    int operate(int a, int b);
}
```

lambda ? Runnable . :

```
public class CustomLambda {
    public static void main(String[] args) {

        TwoArgInterface plusOperation = (a, b) -> a + b;
        TwoArgInterface divideOperation = (a,b)->{
            if (b==0) throw new IllegalArgumentException("Divisor can not be 0");
            return a/b;
        };

        System.out.println("Plus operation of 3 and 5 is: " + plusOperation.operate(3, 5));
        System.out.println("Divide operation 50 by 25 is: " + divideOperation.operate(50,
25));

    }
}
```

`return` .

return .

!

```
void threeTimes(IntConsumer r) {
  for (int i = 0; i < 3; i++) {
    r.accept(i);
  }
}

void demo() {
  threeTimes(i -> {
    System.out.println(i);
    return; // Return from lambda to threeTimes only!
  });
}
```

for return :

```
void demo2() {
  for (int i = 0; i < 3; i++) {
```

```
    System.out.println(i);
    return; // Return from 'demo2' entirely
  }
}
```

Scala Kotlin `demo` `demo` `demo2` `0` . . Java . `return` .

```
void demo3() {
  threeTimes(new MyIntConsumer());
}

class MyIntConsumer implements IntConsumer {
  public void accept(int i) {
    System.out.println(i);
    return;
  }
}
```

, `return` , `for` `while` , .

Java .

```
IntStream.range(1, 4)
    .map(x -> x * x)
    .forEach(System.out::println);
IntStream.range(1, 4)
    .map(x -> { return x * x; })
    .forEach(System.out::println);
```

, Java try-with-resources .

```
class Resource implements AutoCloseable {
  public void close() { System.out.println("close()"); }
}

void executeAround(Consumer<Resource> f) {
  try (Resource r = new Resource()) {
    System.out.print("before ");
    f.accept(r);
    System.out.print("after ");
  }
}

void demo4() {
  executeAround(r -> {
    System.out.print("accept() ");
    return; // Does not return from demo4, but frees the resource.
  });
}
```

`before accept() after close()` . Scala Kotlin , try-with-resources , `before accept()` .

Java Closure.

( ) . .

`final` . Java 8 ( ) `final` . :

```
int n = 0; // With Java 8 there is no need to explicit final
Runnable r = () -> { // Using lambda
    int i = n;
    // do something
};
```

`n` . .

" ."

:

```
int n = 0;
Runnable r = () -> { // Using lambda
    int i = n;
    // do something
};
n++; // Will generate an error.
```

`final` .

```
int n = 0;
final int k = n; // With Java 8 there is no need to explicit final
Runnable r = () -> { // Using lambda
    int i = k;
    // do something
};
n++;      // Now will not generate an error
r.run();  // Will run with i = 0 because k was 0 when the lambda was created
```

.

Java true closure . . . :

```
// Does not compile ...
public IntUnaryOperator createAccumulator() {
    int value = 0;
    IntUnaryOperator accumulate = (x) -> { value += x; return value; };
    return accumulate;
}
```

. .

```
// Compiles, but is incorrect ...
public class AccumulatorGenerator {
    private int value = 0;

    public IntUnaryOperator createAccumulator() {
        IntUnaryOperator accumulate = (x) -> { value += x; return value; };
        return accumulate;
    }
}
```

IntUnaryOperator IntUnaryOperator . . . .

```
// Correct ...
public class Accumulator {
   private int value = 0;

   public int accumulate(int x) {
      value += x;
      return value;
   }
}
```

-

Java 8 JButton . btn.addActionListener .

```
JButton btn = new JButton("My Button");
btn.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        System.out.println("Button was pressed");
    }
});
```

ActionListener actionPerformed() . .

```
JButton btn = new JButton("My Button");
btn.addActionListener(e -> {
    System.out.println("Button was pressed");
});
```

```
interface MathOperation{
    boolean unaryOperation(int num);
}

public class LambdaTry {
    public static void main(String[] args) {
        MathOperation isEven = new MathOperation() {
            @Override
            public boolean unaryOperation(int num) {
                return num%2 == 0;
            }
        };

        System.out.println(isEven.unaryOperation(25));
        System.out.println(isEven.unaryOperation(20));
    }
}
```

1. .

```
public class LambdaTry {
    public static void main(String[] args) {
        MathOperation isEven = (int num) -> {
            return num%2 == 0;
```

```
        };

        System.out.println(isEven.unaryOperation(25));
        System.out.println(isEven.unaryOperation(20));
    }
}
```

2.

```
MathOperation isEven = (num) -> {
    return num%2 == 0;
};
```

3.

```
MathOperation isEven = num -> {
    return num%2 == 0;
};
```

4. (function body)
5. return

```
MathOperation isEven = num -> num%2 == 0;
```

Java lambda "" . lambdas `s -> s.length()` lambdas *stateless* - . lambda `() -> j` .

```
public static void main(String[] args) throws Exception {
    for (int i = 0; i < 1000000000; i++) {
        int j = i;
        doSomethingWithLambda(() -> j);
    }
}
```

`new` `() -> j` 1,000,000,000 . Java [1] .

---

1 - Java 9 Java "" .

Java 8 .

: . 18 .

:

```
public class Person {
    private String name;
    private int age;

    public Person(String name, int age) {
        this.name = name;
        this.age = age;
    }
```

```
    public int getAge() { return age; }
    public String getName() { return name; }
}
```

java.util.function.Predicate Predicate `boolean test(T t)` .

:

```
import java.util.ArrayList;
import java.util.List;
import java.util.function.Predicate;

public class LambdaExample {
    public static void main(String[] args) {
        List<Person> personList = new ArrayList<Person>();
        personList.add(new Person("Jeroen", 20));
        personList.add(new Person("Jack", 5));
        personList.add(new Person("Lisa", 19));

        print(personList, p -> p.getAge() >= 18);
    }

    private static void print(List<Person> personList, Predicate<Person> checker) {
        for (Person person : personList) {
            if (checker.test(person)) {
                System.out.print(person + " matches your expression.");
            } else {
                System.out.println(person  + " doesn't match your expression.");
            }
        }
    }
}
```

`print(personList, p -> p.getAge() >= 18);` (Predicate). : `checker.test(person)` .

. `print(personList, p -> p.getName().startsWith("J"));` . "J" .

: https://riptutorial.com/ko/java/topic/91/-

90:

Examples

. addShutdownHook .

```
Runtime.getRuntime().addShutdownHook(new Thread(() -> {
    ImportantStuff.someImportantIOStream.close();
}));
```

: https://riptutorial.com/ko/java/topic/7304/-

91: (java.util.logging)

Examples

API .

```
import java.util.logging.Level;
import java.util.logging.Logger;

public class MyClass {

    // retrieve the logger for the current class
    private static final Logger LOG = Logger.getLogger(MyClass.class.getName());

    public void foo() {
        LOG.info("A log message");
        LOG.log(Level.INFO, "Another log message");

        LOG.fine("A fine message");

        // logging an exception
        try {
            // code might throw an exception
        } catch (SomeException ex) {
            // log a warning printing "Something went wrong"
            // together with the exception message and stacktrace
            LOG.log(Level.WARNING, "Something went wrong", ex);
        }

        String s = "Hello World!";

        // logging an object
        LOG.log(Level.FINER, "String s: {0}", s);

        // logging several objects
        LOG.log(Level.FINEST, "String s: {0} has length {1}", new Object[]{s, s.length()});
    }

}
```

API 7 . .

- `SEVERE` ( )
- `WARNING`
- `INFO`
- `CONFIG`
- `FINE`
- `FINER`
- `FINEST` ()

`INFO` ( ).

: `OFF` ( ) `ALL` (`OFF`) .

:

```
import java.util.logging.Logger;

public class Levels {
    private static final Logger logger = Logger.getLogger(Levels.class.getName());

    public static void main(String[] args) {

        logger.severe("Message logged by SEVERE");
        logger.warning("Message logged by WARNING");
        logger.info("Message logged by INFO");
        logger.config("Message logged by CONFIG");
        logger.fine("Message logged by FINE");
        logger.finer("Message logged by FINER");
        logger.finest("Message logged by FINEST");

        // All of above methods are really just shortcut for
        // public void log(Level level, String msg):
        logger.log(Level.FINEST, "Message logged by FINEST");
    }
}
```

CONFIG .

```
Jul 23, 2016 9:16:11 PM LevelsExample main
SEVERE: Message logged by SEVERE
Jul 23, 2016 9:16:11 PM LevelsExample main
WARNING: Message logged by WARNING
Jul 23, 2016 9:16:11 PM LevelsExample main
INFO: Message logged by INFO
```

()

.

```
public class LoggingComplex {

    private static final Logger logger =
        Logger.getLogger(LoggingComplex.class.getName());

    private int total = 50, orders = 20;
    private String username = "Bob";

    public void takeOrder() {
        // (...) making some stuff
        logger.fine(String.format("User %s ordered %d things (%d in total)",
                                  username, orders, total));
        // (...) some other stuff
    }

    // some other methods and calculations
}
```

Java VM . , .

Java , `FINE` , `FINER` , `FINEST` `.takeOrder()` .

`javap -c LoggingComplex.class` .

```
public void takeOrder();
    Code:
       0: getstatic     #27 // Field logger:Ljava/util/logging/Logger;
       3: ldc           #45 // String User %s ordered %d things (%d in total)
       5: iconst_3
       6: anewarray     #3  // class java/lang/Object
       9: dup
      10: iconst_0
      11: aload_0
      12: getfield      #40 // Field username:Ljava/lang/String;
      15: aastore
      16: dup
      17: iconst_1
      18: aload_0
      19: getfield      #36 // Field orders:I
      22: invokestatic  #47 // Method java/lang/Integer.valueOf:(I)Ljava/lang/Integer;
      25: aastore
      26: dup
      27: iconst_2
      28: aload_0
      29: getfield      #34 // Field total:I
      32: invokestatic  #47 // Method java/lang/Integer.valueOf:(I)Ljava/lang/Integer;
      35: aastore
      36: invokestatic  #53 // Method
java/lang/String.format:(Ljava/lang/String;[Ljava/lang/Object;)Ljava/lang/String;
      39: invokevirtual #59 // Method java/util/logging/Logger.fine:(Ljava/lang/String;)V
      42: return
```

39 . (, , `format`) `FINE` () . .

.

.

```
public void takeOrder() {
    // making some stuff
    if (logger.isLoggable(Level.FINE)) {
        // no action taken when there's no need for it
        logger.fine(String.format("User %s ordered %d things (%d in total)",
                                  username, orders, total));
    }
    // some other stuff
}
```

8 :

Logger `Supplier<String>` as .

```
public void takeOrder() {
    // making some stuff
    logger.fine(() -> String.format("User %s ordered %d things (%d in total)",
            username, orders, total));
    // some other stuff
```

```
}
```

Suppliers `get()` `get()`) `if` .

(java.util.logging) : https://riptutorial.com/ko/java/topic/2010/--java-util-logging-

92:

java . .

Examples

```
public class localInner1{
 private int data=30;//instance variable
 void display(){
  class Local{
   void msg(){System.out.println(data);}
  }
  Local l=new Local();
  l.msg();
 }
 public static void main(String args[]){
  localInner1 obj=new localInner1();
  obj.display();
 }
}
```

: https://riptutorial.com/ko/java/topic/10160/--

93: ( )

Java JAR . .

. , ( ) .

ClassLoader.getResource ClassLoader.getResourceAsStream . . Class.getResource Class.getResourceAsStream .

getResource getResourceAsStream URL , URL InputStream .

ClassLoader ClassLoader .

- . jar , jar .
- .

URL . (/) . .

Class .

- (slash) , ClassLoader .
- getResource getResourceAsStream . *package* / *name* . *package* *name* .

:

```
package com.example;

public class ExampleApplication {
    public void readImage()
    throws IOException {

        URL imageURL = ExampleApplication.class.getResource("icon.png");

        // The above statement is identical to:
        // ClassLoader loader = ExampleApplication.class.getClassLoader();
        // URL imageURL = loader.getResource("com/example/icon.png");

        Image image = ImageIO.read(imageURL);
    }
}
```

.jar . .jar `config.properties` getResource getResourceAsStream .jar config.properties . classpath. Java EE .

, getResource getResourceAsStream `null` . . .

. . IDE (Eclipse) . . .jar .war . . URL getFile , URL .

. . .

Examples

:

```
package com.example;

public class ExampleApplication {
    private Image getIcon() throws IOException {
        URL imageURL = ExampleApplication.class.getResource("icon.png");
        return ImageIO.read(imageURL);
    }
}
```

.

```
package com.example;

public class ExampleApplication {
    private Properties getDefaults() throws IOException {
        Properties defaults = new Properties();

        try (InputStream defaultsStream =
            ExampleApplication.class.getResourceAsStream("config.properties")) {

            defaults.load(defaultsStream);
        }

        return defaults;
    }
}
```

JAR

JAR . . .

- META-INF / MANIFEST.MF
- META-INF / beans.xml (CDI)
- ServiceLoader

ClassLoader . ClassLoader . `Enumeration` Collections `List` .

```
Enumeration<URL> resEnum = MyClass.class.getClassLoader().getResources("META-
INF/MANIFEST.MF");
ArrayList<URL> resources = Collections.list(resEnum);
```

Java .

1. `Class ClassLoader` .
2. .
3. .
4. .
5. .

URL . `getResource` . . `getResourceAsStream` .

. UNIX / Linux . (/) . .

Classpath JVM JAR ZIP . / . "" . .

Class ClassLoader .Class .Class . :

- Java Class . : String.class String Class .
- Object.getClass() Class . : "hello".getClass() String Class .
- Class.forName(String) ( ) Class . : Class.forName("java.lang.String") .

ClassLoader Class getClassLoader() getClassLoader() . ClassLoader.getSystemClassLoader() JVM .

get

Class ClassLoader .

ClassLoader.getResource(path) ClassLoader.getResources(path)	URL .
ClassLoader.getResources(path) Class.getResources(path)	foo.bar URL Enumeration<URL> ..
ClassLoader.getResourceAsStream(path) Class.getResourceStream(path)	foo.bar InputStream .

:

- ClassLoader Class .
  - Class "" .
  - ClassLoader ., " ".
- ( ) getResource getResourceAsStream methods return null methods return , and the getResources methods return an empty Enumeration` methods return an empty .
- URL URL.toStream() . file: URL URL, JAR , jar: JAR URL.
- getResourceAsStream (URL.toStream()) InputStream . .

( ) : https://riptutorial.com/ko/java/topic/2433/-----

94:

Java (, Java). `1` , `0.333F` , `false` , `'X'` `"Hello world\n"` .

Examples

16 , 8 2

`hexadecimal` `hexadecimal` . 16 , `0-9` `AF` ( ) . `AF` `10-16` .

`octal` `octal` `0-7` .

`binary` `binary` `0` `1` .

. `110` :

```
int dec = 110;            // no prefix   --> decimal literal
int bin = 0b1101110;      // '0b' prefix --> binary literal
int oct = 0156;           // '0' prefix  --> octal literal
int hex = 0x6E;           // '0x' prefix --> hexadecimal literal
```

Java 7 .

8 . `'0'` .

```
int a = 0100;        // Instead of 100, a == 64
```

Java 7 (_) .

, .

Java SE 7

```
int i1 = 123456;
int i2 = 123_456;
System.out.println(i1 == i2); // true
```

:

Java SE 7

```
byte color = 1_2_3;
short yearsAnnoDomini= 2_016;
int socialSecurtyNumber = 999_99_9999;
long creditCardNumber = 1234_5678_9012_3456L;
float piFourDecimals =  3.14_15F;
double piTenDecimals =  3.14_15_92_65_35;
```

2 , 8 16 .

Java SE 7

```
short binary= 0b0_1_0_1;
int octal = 07_7_7_7_7_7_7_0;
long hexBytes = 0xFF_EC_DE_5E;
```

.

- (: `_123 123_`)
- (: `1._23 1_.23`)
- `1.23_F` F L (: `1.23_F 9999999_L`).
- (: `0_xFFFF`)

. (\ ) . .

.

`\\`	(\ ) .
`\'`	(') .
`\"`	(") .
`\n`	(LF) .
`\r`	(CR) .
`\t`	(HT) .
`\f`	(FF) .
`\b`	(BS) .
`\<octal>`	0 255 .

`<octal>` 0 255 (10) 1, 2 3 8 ('0' '7') .

. JLS .

:

- JLS 3.10.6.

Java JLS 3.3 . . .

```
'\' 'u' <hex-digit> <hex-digit> <hex-digit> <hex-digit>
```

`<hex-digit>` `'0','1','2','3','4','5','6','7','8','9','a','b','c','d','e','f','A','B','C','D','E','F'` .

Java (16) . ASCII .

`byte` , `short` , `int` , `long` `char` . (. 8 , 16 2 .)

. :

```
0       // The decimal number zero       (type 'int')
1       // The decimal number one        (type 'int')
42      // The decimal number forty two  (type 'int')
```

0. 0 *8* .

```
077    // This literal actually means 7 x 8 + 7 ... or 63 decimal!
```

. `-10` `+10` , - + .

`int` intrinsic 0 2 ^{31} 2,147,483,648 .

2 ^{31} `Integer.MAX_VALUE` 1 . 0 `2147483647` `2147483648` - . (, `Integer.MIN_VALUE` .)

```
int max = 2147483647;     // OK
int min = -2147483648;    // OK
int tooBig = 2147483648;  // ERROR
```

`long` `L` . :

```
0L          // The decimal number zero       (type 'long')
1L          // The decimal number one        (type 'long')
2147483648L // The value of Integer.MAX_VALUE + 1

long big = 2147483648;    // ERROR
long big2 = 2147483648L;  // OK
```

`int` `long` .

```
int i = 2147483647;
long l = i + 1;           // Produces a negative value because the operation is
                          // performed using 32 bit arithmetic, and the
                          // addition overflows
long l2 = i + 1L;         // Produces the (intuitively) correct value.
```

: JLS 3.10.1 -

Boolean Java . `boolean` `true` `false` . / . :

```
boolean flag = true;    // using the 'true' literal
flag = false;           // using the 'false' literal
```

Java . .

-

(") .
- 0 . (\ ) .
- .

:

```
"Hello world"   // A literal denoting an 11 character String
""              // A literal denoting an empty (zero length) String
"\""            // A literal denoting a String consisting of one
                //     double quote character
"1\t2\t3\n"     // Another literal with escape sequences
```

. ( ) . :

```
"Jello world    // Compilation error (at the end of the line!)
```

, (+) .

```
String typingPractice = "The quick brown fox " +
                        "jumped over " +
                        "the lazy dog"
```

+ . String .

JVM , String . , .

.

Null

Null (null) null (null) .

```
    MyClass object = null;
    MyClass[] objects = new MyClass[]{new MyClass(), null, new MyClass()};

    myMethod(null);

    if (objects != null) {
         // Do something
    }
```

null . Java . (.)

null . . .

null null instanceof <SomeReferenceType> false .

float double . .

-
-

- 16

JLS 10 . .

`float double` . . .

(.) (`f` , `F` , `d D`). `float` (`f F`) `double` (`d D`) . ( ) `double` .

```
0.0     // this denotes zero
.0      // this also denotes zero
0.      // this also denotes zero
3.14159 // this denotes Pi, accurate to (approximately!) 5 decimal places.
1.0F    // a `float` literal
1.0D    // a `double` literal.  (`double` is the default if no suffix is given)
```

.

```
1F      // means the same thing as 1.0F
```

10 IEEE . IEEE . 10 `Double.valueOf(String)` `Float.valueOf(String)` javadocs .

`E e` . 10 . `float double` . :

```
1.0E1    // this means 1.0 x 10^1 ... or 10.0 (double)
1E-1D    // this means 1.0 x 10^(-1) ... or 0.1 (double)
1.0e10f  // this means 1.0 x 10^(10) ... or 10000000000.0 (float)
```

(`float double`) . .

16

Java 6 16 . 16 .

1. 16 (`0`) `x X` .
2. (!) `a f` 16 .
3. `e E p` (`P`). 10 2 .

:

```
0x0.0p0f    // this is zero expressed in hexadecimal form (`float`)
0xff.0p19   // this is 255.0 x 2^19 (`double`)
```

: 16 Java .

Java 7 . "" . .

. , + INF -INF 0.0 . 0 .

INF NaN IEEE 754 . , `java.lang.Float` `java.lang.Double` . `Float.NaN` ,

`Float.NEGATIVE_INFINITY` `Float.POSITIVE_INFINITY`

Java `char` . .

- (').
- . (\ ) . .
- (').

:

```
char a = 'a';
char doubleQuote = '"';
char singleQuote = '\'';
```

.

```
char newline = '
// Compilation error in previous line
char newLine = '\n';  // Correct
```

: https://riptutorial.com/ko/java/topic/8250/

95: API

JVM . Java Reflection API , , . .

.

Java Reflection API :

Java . .

Examples

API . .

public :

```
import java.lang.reflect.Constructor;
import java.lang.reflect.Method;

// This is a object representing the String class (not an instance of String!)
Class<String> clazz = String.class;

Constructor<?>[] constructors = clazz.getConstructors(); // returns all public constructors of
String
Method[] methods = clazz.getMethods(); // returns all public methods from String and parents
```

.

.

- `getGenericParameterTypes()` .
- `getGenericReturnType()` .
- `getGenericType` .

.

```
import java.lang.reflect.Field;
import java.lang.reflect.Method;
import java.lang.reflect.ParameterizedType;
import java.lang.reflect.Type;
import java.util.List;
import java.util.Map;

public class GenericTest {

    public static void main(final String[] args) throws Exception {
        final Method method = GenericTest.class.getMethod("testMethod", Map.class);
        final Field field = GenericTest.class.getField("testField");

        System.out.println("Method parameter:");
        final Type parameterType = method.getGenericParameterTypes()[0];
```

```
        displayGenericType(parameterType, "\t");

        System.out.println("Method return type:");
        final Type returnType = method.getGenericReturnType();
        displayGenericType(returnType, "\t");

        System.out.println("Field type:");
        final Type fieldType = field.getGenericType();
        displayGenericType(fieldType, "\t");

    }

    private static void displayGenericType(final Type type, final String prefix) {
        System.out.println(prefix + type.getTypeName());
        if (type instanceof ParameterizedType) {
            for (final Type subtype : ((ParameterizedType) type).getActualTypeArguments()) {
                displayGenericType(subtype, prefix + "\t");
            }
        }

    }

    public Map<String, Map<Integer, List<String>>> testField;

    public List<Number> testMethod(final Map<String, Double> arg) {
        return null;
    }

}
```

.

```
Method parameter:
    java.util.Map<java.lang.String, java.lang.Double>
        java.lang.String
        java.lang.Double
Method return type:
    java.util.List<java.lang.Number>
        java.lang.Number
Field type:
    java.util.Map<java.lang.String, java.util.Map<java.lang.Integer,
java.util.List<java.lang.String>>>
        java.lang.String
        java.util.Map<java.lang.Integer, java.util.List<java.lang.String>>
            java.lang.Integer
            java.util.List<java.lang.String>
                java.lang.String
```

.

String .

```
import java.lang.reflect.InvocationTargetException;
import java.lang.reflect.Method;

String s = "Hello World!";

// method without parameters
```

```
// invoke s.length()
Method method1 = String.class.getMethod("length");
int length = (int) method1.invoke(s); // variable length contains "12"

// method with parameters
// invoke s.substring(6)
Method method2 = String.class.getMethod("substring", int.class);
String substring = (String) method2.invoke(s, 6); // variable substring contains "World!"
```

API . , API OS . `final` `final` .

Class # getField () .

```
// Get the field in class SomeClass "NAME".
Field nameField = SomeClass.class.getDeclaredField("NAME");

// Get the field in class Field "modifiers". Note that it does not
// need to be static
Field modifiersField = Field.class.getDeclaredField("modifiers");

// Allow access from anyone even if it's declared private
modifiersField.setAccessible(true);

// Get the modifiers on the "NAME" field as an int.
int existingModifiersOnNameField = nameField.getModifiers();

// Bitwise AND NOT Modifier.FINAL (16) on the existing modifiers
// Readup here https://en.wikipedia.org/wiki/Bitwise_operations_in_C
// if you're unsure what bitwise operations are.
int newModifiersOnNameField = existingModifiersOnNameField & ~Modifier.FINAL;

// Set the value of the modifiers field under an object for non-static fields
modifiersField.setInt(nameField, newModifiersOnNameField);

// Set it to be accessible. This overrides normal Java
// private/protected/package/etc access control checks.
nameField.setAccessible(true);

 // Set the value of "NAME" here. Note the null argument.
 // Pass null when modifying static fields, as there is no instance object
nameField.set(null, "Hacked by reflection...");

// Here I can directly access it. If needed, use reflection to get it. (Below)
System.out.println(SomeClass.NAME);
```

. Field # get () .

```
// Get the field in class SomeClass "NAME".
Field nameField = SomeClass.class.getDeclaredField("NAME");

// Set accessible for private fields
nameField.setAccessible(true);

// Pass null as there is no instance, remember?
String name = (String) nameField.get(null);
```

:

Class # getDeclaredField .

```
class HackMe extends Hacked {
    public String iAmDeclared;
}

class Hacked {
    public String someState;
}
```

`HackMe#iAmDeclared` . `HackMe#someState` Hacked .

`Class` `Constructor` .

```
Class myClass = ... // get a class object
Constructor[] constructors = myClass.getConstructors();
```

`constructors` public `Constructor` .

. , `Integer` , public .

```
Class myClass = ... // get a class object
Constructor constructor = myClass.getConstructor(new Class[]{Integer.class});
```

`NoSuchMethodException` .

```
Class myClass = MyObj.class // get a class object
Constructor constructor = myClass.getConstructor(Integer.class);
MyObj myObj = (MyObj) constructor.newInstance(Integer.valueOf(123));
```

:

```
enum Compass {
    NORTH(0),
    EAST(90),
    SOUTH(180),
    WEST(270);
    private int degree;
    Compass(int deg){
        degree = deg;
    }
    public int getDegree(){
        return degree;
    }
}
```

Java . `values()` `valueOf(String)` .
Reflection .

```
for(Field f : Compass.class.getDeclaredFields())
    System.out.println(f.getName());
```

.

ENUM $ VALUES

Reflection enum . Reflection API .

```
Compass.class.isEnum();
```

enum true .

```
Object[] values = Compass.class.getEnumConstants();
```

Compass.values () enum .

enum constant check

```
for(Field f : Compass.class.getDeclaredFields()){
    if(f.isEnumConstant())
        System.out.println(f.getName());
}
```

enum .

()

String Class Class.forName .

```
Class clazz = null;
try {
    clazz = Class.forName("java.lang.Integer");
} catch (ClassNotFoundException ex) {
    throw new IllegalStateException(ex);
}
```

Java SE 1.2

(forName) ClassLoader ( ) .

```
ClassLoader classLoader = ...
boolean initialize = ...
Class clazz = null;
try {
    clazz = Class.forName("java.lang.Integer", initialize, classLoader);
} catch (ClassNotFoundException ex) {
    throw new IllegalStateException(ex);
}
```

:

```
import java.lang.reflect.*;

class NewInstanceWithReflection{
    public NewInstanceWithReflection(){
        System.out.println("Default constructor");
    }
    public NewInstanceWithReflection( String a){
        System.out.println("Constructor :String => "+a);
    }
    public static void main(String args[]) throws Exception {

        NewInstanceWithReflection object =
(NewInstanceWithReflection)Class.forName("NewInstanceWithReflection").newInstance();
        Constructor constructor = NewInstanceWithReflection.class.getDeclaredConstructor( new
Class[] {String.class});
        NewInstanceWithReflection object1 =
(NewInstanceWithReflection)constructor.newInstance(new Object[]{"StackOverFlow"});

    }
}
```

:

```
Default constructor
Constructor :String => StackOverFlow
```

:

1. `Class.forName` : .
2. `Class array` , `getDeclaredConstructor` .
3. `newInstance` `Object array` `newInstance` .

Reflection API

. .

.

```
import java.lang.reflect.*;

public class ReflectionDemo{
    public static void main(String args[]){
        try{
            Field[] fields = A.class.getDeclaredFields();
            A a = new A();
            for ( Field field:fields ) {
                if(field.getName().equalsIgnoreCase("name")){
                    field.setAccessible(true);
                    field.set(a, "StackOverFlow");
                    System.out.println("A.name="+field.get(a));
                }
                if(field.getName().equalsIgnoreCase("age")){
                    field.set(a, 20);
```

```
                    System.out.println("A.age="+field.get(a));
                }
                if(field.getName().equalsIgnoreCase("rep")){
                    field.setAccessible(true);
                    field.set(a,"New Reputation");
                    System.out.println("A.rep="+field.get(a));
                }
                if(field.getName().equalsIgnoreCase("count")){
                    field.set(a,25);
                    System.out.println("A.count="+field.get(a));
                }
            }
        }catch(Exception err){
            err.printStackTrace();
        }
    }
}

class A {
    private String name;
    public int age;
    public final String rep;
    public static int count=0;

    public A(){
        name = "Unset";
        age = 0;
        rep = "Reputation";
        count++;
    }
}
```

:

```
A.name=StackOverFlow
A.age=20
A.rep=New Reputation
A.count=25
```

:

`private` getter setter . `final` .

`Reflection` .

`field.setAccessible(true)` .

getDeclaredConstructor .

```
public class Enclosing{
    public class Nested{
    public Nested(String a){
            System.out.println("Constructor :String => "+a);
        }
    }
    public static void main(String args[]) throws Exception {
        Class<?> clazzEnclosing = Class.forName("Enclosing");
```

```
        Class<?> clazzNested = Class.forName("Enclosing$Nested");
        Enclosing objEnclosing = (Enclosing)clazzEnclosing.newInstance();
        Constructor<?> constructor = clazzNested.getDeclaredConstructor(new
Class[]{Enclosing.class, String.class});
        Nested objInner = (Nested)constructor.newInstance(new Object[]{objEnclosing,
"StackOverFlow"});
    }
}
```

.

API . . .

.

```
public class DynamicProxyTest {

    public interface MyInterface1{
        public void someMethod1();
        public int someMethod2(String s);
    }

    public interface MyInterface2{
        public void anotherMethod();
    }

    public static void main(String args[]) throws Exception {
        // the dynamic proxy class
        Class<?> proxyClass = Proxy.getProxyClass(
                ClassLoader.getSystemClassLoader(),
                new Class[] {MyInterface1.class, MyInterface2.class});
        // the dynamic proxy class constructor
        Constructor<?> proxyConstructor =
            proxyClass.getConstructor(InvocationHandler.class);

        // the invocation handler
        InvocationHandler handler = new InvocationHandler(){
            // this method is invoked for every proxy method call
            // method is the invoked method, args holds the method parameters
            // it must return the method result
            @Override
            public Object invoke(Object proxy, Method method, Object[] args) throws Throwable
{
                String methodName = method.getName();

                if(methodName.equals("someMethod1")){
                    System.out.println("someMethod1 was invoked!");
                    return null;
                }
                if(methodName.equals("someMethod2")){
                    System.out.println("someMethod2 was invoked!");
                    System.out.println("Parameter: " + args[0]);
                    return 42;
                }
                if(methodName.equals("anotherMethod")){
                    System.out.println("anotherMethod was invoked!");
                    return null;
                }
                System.out.println("Unkown method!");
                return null;
```

```
            }
        };

        // create the dynamic proxy instances
        MyInterface1 i1 = (MyInterface1) proxyConstructor.newInstance(handler);
        MyInterface2 i2 = (MyInterface2) proxyConstructor.newInstance(handler);

        // and invoke some methods
        i1.someMethod1();
        i1.someMethod2("stackoverflow");
        i2.anotherMethod();
    }
}
```

.

```
someMethod1 was invoked!
someMethod2 was invoked!
Parameter: stackoverflow
anotherMethod was invoked!
```

Reflection API JDK . .

. Reflection . Java `SecurityManager` . java.lang.System .

```
private static volatile SecurityManager security = null;
```

System .

```
for(Field f : System.class.getDeclaredFields())
    System.out.println(f);
```

`System.class` `sun.reflect.Reflection` `fieldFilterMap` `sun.reflect.Reflection` Reflection . `SecurityManager` .

Java String JVM `String` . JVM . `String.intern()` "interned" . `"hello"` .

"" . `value` .

```
public class CrazyStrings {
    static {
        try {
            Field f = String.class.getDeclaredField("value");
            f.setAccessible(true);
            f.set("hello", "you stink!".toCharArray());
        } catch (Exception e) {
        }
    }
    public static void main(String args[])  {
        System.out.println("hello");
    }
}
```

" !" .

1 = 42

.

```
public class CrazyMath {
    static {
        try {
            Field value = Integer.class.getDeclaredField("value");
            value.setAccessible(true);
            value.setInt(Integer.valueOf(1), 42);
        } catch (Exception e) {
        }
    }
    public static void main(String args[])  {
        System.out.println(Integer.valueOf(1));
    }
}
```

.

stackoverflow .

```
public class Evil {
    static {
        try {
            Field field = Boolean.class.getField("FALSE");
            field.setAccessible(true);
            Field modifiersField = Field.class.getDeclaredField("modifiers");
            modifiersField.setAccessible(true);
            modifiersField.setInt(field, field.getModifiers() & ~Modifier.FINAL);
            field.set(null, true);
        } catch (Exception e) {
        }
    }
    public static void main(String args[]){
        System.out.format("Everything is %s", false);
    }
}
```

JVM . .

API : https://riptutorial.com/ko/java/topic/629/-api

96:

- ([] args)

args	. `main` Java `args` null `null` .

Java `java` ( ) `args` `main` .

Java SE . .

- Java .
- .

3 . StackOverflow . "Java " .

Examples

GWT ToolBase

google GWT . .

https://gwt.googlesource.com/gwt/+/2.8.0-beta1/dev/core/src/com/google/gwt/util/tools/ToolBase.java

`myprogram -dir "~/Documents" -port 8888` .

```
public class MyProgramHandler extends ToolBase {
   protected File dir;
   protected int port;
   // getters for dir and port
   ...

   public MyProgramHandler() {
       this.registerHandler(new ArgHandlerDir() {
            @Override
            public void setDir(File dir) {
                this.dir = dir;
            }
       });
       this.registerHandler(new ArgHandlerInt() {
            @Override
            public String[] getTagArgs() {
               return new String[]{"port"};
            }
            @Override
            public void setInt(int value) {
               this.port = value;
            }
       });
   }
```

```
   public static void main(String[] args) {
      MyProgramHandler myShell = new MyProgramHandler();
      if (myShell.processArgs(args)) {
         // main program operation
         System.out.println(String.format("port: %d; dir: %s",
            myShell.getPort(), myShell.getDir()));
      }
      System.exit(1);
   }
}
```

`ArgHandler isRequired()` . (`false`).

.

. `System.exit(1)` . (Java "myapp" .)

. `args.length` `args.length` .

```
public class Main {
    public static void main(String[] args) {
        if (args.length > 0) {
            System.err.println("usage: myapp");
            System.exit(1);
        }
        // Run the application
        System.out.println("It worked");
    }
}
```

, .

```
public class Main {
    public static void main(String[] args) {
        if (args.length != 2) {
            System.err.println("usage: myapp <arg1> <arg2>");
            System.exit(1);
        }
        // Run the application
        System.out.println("It worked: " + args[0] + ", " + args[1]);
    }
}
```

`args.length` , .

""

() .

```
package tommy;
public class Main {
    public static void main(String[] args) {
        boolean feelMe = false;
        boolean seeMe = false;
```

```
        int index;
        loop: for (index = 0; index < args.length; index++) {
            String opt = args[index];
            switch (opt) {
            case "-c":
               seeMe = true;
               break;
            case "-f":
               feelMe = true;
               break;
            default:
               if (!opts.isEmpty() && opts.charAt(0) == '-') {
                   error("Unknown option: '" + opt + "');
               }
               break loop;
            }
        }
        if (index >= args.length) {
            error("Missing argument(s)");
        }

        // Run the application
        // ...
    }

    private static void error(String message) {
        if (message != null) {
            System.err.println(message);
        }
        System.err.println("usage: myapp [-f] [-c] [ <arg> ...]");
        System.exit(1);
    }
}
```

." " . .

: https://riptutorial.com/ko/java/topic/4775/---

97:

- java.xml .
- public java.xml . # .
- com.example.foo; # .
- com.example.foo.impl com.example.bar . # .

. Java 9 . .

. **exports** .

.

(: `module`) .

Examples

`module-info.java` . :

```
 |-- module-info.java
 |-- com
     |-- example
         |-- foo
             |-- Foo.java
         |-- bar
             |-- Bar.java
```

.

```
module com.example {
    requires java.httpclient;
    exports com.example.foo;
}
```

Reverse-DNS .

Java `java.base` .

`requires` , `java.httpclient` .

`exports` .

`exports` `com.example.foo` . `com.example.foo` `export` .

`exports` `com.example.bar` .

: https://riptutorial.com/ko/java/topic/5286/

98: SET

List Set Java List Java Set

Examples

```
import java.util.ArrayList;
```

import java.util.HashSet; import java.util.List; import java.util.Set;

SetAndListExample {public static void main (String [] args) {System.out.println ("List example ....."); = ArrayList (); list.add ("1"); list.add ("2"); list.add ("3"); list.add ("4"); list.add ("1");

```
    for (String temp : list){
        System.out.println(temp);
    }

    System.out.println("Set example .....");
    Set<String> set = new HashSet<String>();
    set.add("1");
    set.add("2");
    set.add("3");
    set.add("4");
    set.add("1");
    set.add("2");
    set.add("5");

    for (String temp : set){
        System.out.println(temp);
    }
}
```

}

..... 1 2 3 4 1 ..... 3 2 10 5 4

SET : https://riptutorial.com/ko/java/topic/10125/--set

99:

Examples

UTF-8

```
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.IOException;
import java.nio.charset.StandardCharsets;
import java.nio.file.Files;
import java.nio.file.Paths;

public class ReadingUTF8TextFile {

    public static void main(String[] args) throws IOException {
        //StandardCharsets is available since Java 1.7
        //for ealier version use Charset.forName("UTF-8");
        try (BufferedWriter wr = Files.newBufferedWriter(Paths.get("test.txt"),
StandardCharsets.UTF_8)) {
            wr.write("Strange cyrillic symbol Ы");
        }
        /* First Way. For big files */
        try (BufferedReader reader = Files.newBufferedReader(Paths.get("test.txt"),
StandardCharsets.UTF_8)) {

            String line;
            while ((line = reader.readLine()) != null) {
                System.out.print(line);
            }
        }

        System.out.println(); //just separating output

        /* Second way. For small files */
        String s = new String(Files.readAllBytes(Paths.get("test.txt")),
StandardCharsets.UTF_8);
        System.out.print(s);
    }
}
```

UTF-8

```
import java.io.BufferedWriter;
import java.io.IOException;
import java.nio.charset.StandardCharsets;
import java.nio.file.Files;
import java.nio.file.Paths;

public class WritingUTF8TextFile {
    public static void main(String[] args) throws IOException {
        //StandardCharsets is available since Java 1.7
        //for ealier version use Charset.forName("UTF-8");
        try (BufferedWriter wr = Files.newBufferedWriter(Paths.get("test2.txt"),
StandardCharsets.UTF_8)) {
```

```
            wr.write("Cyrillic symbol Ы");
        }
    }
}
```

UTF-8

```
import java.nio.charset.StandardCharsets;
import java.util.Arrays;

public class GetUtf8BytesFromString {

    public static void main(String[] args) {
        String str = "Cyrillic symbol Ы";
        //StandardCharsets is available since Java 1.7
        //for ealier version use Charset.forName("UTF-8");
        byte[] textInUtf8 = str.getBytes(StandardCharsets.UTF_8);

        System.out.println(Arrays.toString(textInUtf8));
    }
}
```

: https://riptutorial.com/ko/java/topic/2735/-

100:

(`java.lang.String`) . Java Java .

Java . (.)

Java , `String` **`String`** . `String` . C C ++ `String` substring .

`String` `String` `StringBuilder` . `StringBuilder` `String` .

`StringBuffer` `String` . . `StringBuilder` .

`+` . `+` `String.concat(...)` `StringBuilder` . `StringBuilder` .

. `String` . `String` . . `String` . `char[]` (: `'\000'`).

`String` . JVM (,) `String` (JLS) . JVM . ( ) . `String` . `String` ( ) . (internation).

Java .

Java SE 7

Java 7 Java "PermGen" . .

Java SE 7

Java 7 "PermGen" .

. , `String` .

Java 8 `String` UTF-16 char (char 2) . Java 9 `String` (LATIN-1) (UTF-16) .

Examples

, String `equals` `equalsIgnoreCase` .

`String` .

```
String firstString = "Test123";
String secondString = "Test" + 123;

if (firstString.equals(secondString)) {
   // Both Strings have the same content.
}
```

:

```
String firstString = "Test123";
String secondString = "TEST123";

if (firstString.equalsIgnoreCase(secondString)) {
    // Both Strings are equal, ignoring the case of the individual characters.
}
```

`equalsIgnoreCase` `Locale` . , "Taki" "TAKI" . (`I` ı). `Locale` ( ) `equals` .

```
String firstString = "Taki";
String secondString = "TAKI";

System.out.println(firstString.equalsIgnoreCase(secondString)); //prints true

Locale locale = Locale.forLanguageTag("tr-TR");

System.out.println(firstString.toLowerCase(locale).equals(
                   secondString.toLowerCase(locale))); //prints false
```

==

( ) `==` `!=` . , `String` String .

`String.equals(Object)` . String . Pitfall : == .

---

switch

Java SE 7

1.7, `switch` String . String null . `NullPointerException` throw. `String.equals`) .

```
String stringToSwitch = "A";

switch (stringToSwitch) {
    case "a":
        System.out.println("a");
        break;
    case "A":
        System.out.println("A"); //the code goes here
        break;
    case "B":
        System.out.println("B");
        break;
    default:
        break;
}
```

String equals String null NullPointerException .

```
"baz".equals(foo)
```

foo null , foo.equals("baz") NullPointerException , "baz".equals(foo) false .

Java SE 7

Objects.equals(foo, "baz") null Objects.equals() .

(: NullPointerExceptions , ( ) "" .

String String.compareTo Comparable<String> ( ). String . String CASE_INSENSITIVE_ORDER Comparator<String> .

---

interned

Java (JLS 3.10. 6), .

String . String.intern String.intern ."

, == . String.intern() String .

:

```
String strObj = new String("Hello!");
String str = "Hello!";

// The two string references point two strings that are equal
if (strObj.equals(str)) {
    System.out.println("The strings are equal");
}

// The two string references do not point to the same object
if (strObj != str) {
    System.out.println("The strings are not the same object");
}

// If we intern a string that is equal to a given literal, the result is
// a string that has the same reference as the literal.
String internedStr = strObj.intern();

if (internedStr == str) {
    System.out.println("The interned string and the literal are the same object");
}
```

. String intern() .

- .
- .

== interning . . Pitfall - . == . .

String /

String .

- toUpperCase - .
- toLowerCase

String . String Java String . .

```
String string = "This is a Random String";
String upper = string.toUpperCase();
String lower = string.toLowerCase();

System.out.println(string);   // prints "This is a Random String"
System.out.println(lower);    // prints "this is a random string"
System.out.println(upper);    // prints "THIS IS A RANDOM STRING"
```

. .

: , . , HTML .

, "TITLE".toLowerCase() "tıtle". ı (\u0131) DOTLESS I. Locale.ROOT Locale.ROOT (: toLowerCase(Locale.ROOT) toUpperCase(Locale.ROOT)) Locale.ROOT .

Locale.ENGLISH Locale.ENGLISH , Locale.ROOT .

Unicode Consortium .

ASCII / :

ASCII / .

:

1. .
2. .
3. .
4. .
5. .
6. .
    - ASCII 32 .
    - ASCII 32 .
7. .
8. .

, .

.

```
Scanner scanner = new Scanner(System.in);
System.out.println("Enter the String");
String s = scanner.next();
char[] a = s.toCharArray();
System.out.println("Enter the character you are looking for");
System.out.println(s);
String c = scanner.next();
char d = c.charAt(0);

for (int i = 0; i <= s.length(); i++) {
    if (a[i] == d) {
        if (d >= 'a' && d <= 'z') {
            d -= 32;
        } else if (d >= 'A' && d <= 'Z') {
            d += 32;
        }
        a[i] = d;
        break;
    }
}
s = String.valueOf(a);
System.out.println(s);
```

a b String.contains() .

```
b.contains(a); // Return true if a is contained in b, false otherwise
```

String.contains() CharSequence . a b .

```
String str1 = "Hello World";
String str2 = "Hello";
String str3 = "helLO";

System.out.println(str1.contains(str2)); //prints true
System.out.println(str1.contains(str3)); //prints false
```

Ideone

String String String.indexOf() .

```
String s = "this is a long sentence";
int i = s.indexOf('i');    // the first 'i' in String is at index 2
int j = s.indexOf("long"); // the index of the first occurrence of "long" in s is 10
int k = s.indexOf('z');    // k is -1 because 'z' was not found in String s
int h = s.indexOf("LoNg"); // h is -1 because "LoNg" was not found in String s
```

Ideone

String.indexOf() char String String . -1 .

: String.indexOf() / .

:

```
String str1 = "Hello World";
String str2 = "wOr";
str1.indexOf(str2);                                 // -1
str1.toLowerCase().contains(str2.toLowerCase());  // true
str1.toLowerCase().indexOf(str2.toLowerCase());    // 6
```

Ideone

String .

a `String length()` . UTF-16 () .

```
String str = "Hello, World!";
System.out.println(str.length()); // Prints out 13
```

Ideone

String `char` UTF-16 . 0x1000 (:) . UTF-16 `char` String `codePointCount` .

```
int length = str.codePointCount(0, str.length());
```

8 .

```
int length = str.codePoints().count();
```

```
String s = "this is an example";
String a = s.substring(11); // a will hold the string starting at character 11 until the end
("example")
String b = s.substring(5, 10); // b will hold the string starting at character 5 and ending
right before character 10 ("is an")
String b = s.substring(5, b.length()-3); // b will hold the string starting at character 5
ending right before b' s lenght is out of 3  ("is an exam")
```

String / . , .

```
String datestring = "2015□11□17□"
datestring = datestring.substring(0, 4) + "-" + datestring.substring(5,7) + "-" +
datestring.substring(8,10);
//Result will be 2015-11-17
```

substring `String` . `String` . ( ). .

Java SE 7

JDK <7u6 , `substring` , `String` `char[]` `String` , `offset` `count` start length . `new String(s.substring(...))` `char[]` .

Java SE 7

JDK 7u6 `substring` `char[]` .

n

```
String str = "My String";

System.out.println(str.charAt(0)); // "M"
System.out.println(str.charAt(1)); // "y"
System.out.println(str.charAt(2)); // " "
System.out.println(str.charAt(str.length-1)); // Last character "g"
```

n , `charAt(n)` A `String` , `n`

: `n 0` n = 0 .

, (`\n` `\r\n`). Java .

```
System.getProperty("line.separator")
```

Java SE 7

Java 7 .

```
System.lineSeparator()
```

: .

`String.format` `\n` '\ r \ n' `%n` .

```
System.out.println(String.format('line 1: %s.%nline 2: %s%n', lines[0],lines[1]));
```

toString ()

`Person` .

```
public class Person {

    String name;
    int age;

    public Person (int age, String name) {
        this.age = age;
        this.name = name;
    }
}
```

`Person`

```
Person person = new Person(25, "John");
```

.

```
System.out.println(person.toString());
```

Ideone

.

```
Person@7ab89d
```

Person Person Object toString() . Object.toString() .

> Object toString , at `@' 16 . , .
>
> ```
> getClass().getName() + '@' + Integer.toHexString(hashCode())
> ```

toString() .

```
@Override
public String toString() {
    return "My name is " + this.name + " and my age is " + this.age;
}
```

.

```
My name is John and my age is 25
```

.

```
System.out.println(person);
```

Ideone

println() toString .

String String.split() .

```
public String[] split(String regex)
```

.

:

```
String lineFromCsvFile = "Mickey;Bolton;12345;121216";
String[] dataCells = lineFromCsvFile.split(";");
// Result is dataCells = { "Mickey", "Bolton", "12345", "121216"};
```

:

```
String lineFromInput = "What    do you need    from me?";
String[] words = lineFromInput.split("\\s+"); // one or more space chars
```

```
// Result is words = {"What", "do", "you", "need", "from", "me?"};
```

String .

```
String[] firstNames = "Mickey, Frank, Alicia, Tom".split(", ");
// Result is firstNames = {"Mickey", "Frank", "Alicia", "Tom"};
```

: .

```
"aaa.bbb".split("."); // This returns an empty array
```

. .

( ) .

```
  < > - = ! ( ) [ ] { } \ ^ $ | ? * + .
```

\\ Pattern.quote() .

- Pattern.quote() :

  ```
  String s = "a|b|c";
  String regex = Pattern.quote("|");
  String[] arr = s.split(regex);
  ```

- :

  ```
  String s = "a|b|c";
  String[] arr = s.split("\\|");
  ```

Split .

split(delimiter) . split(delimiter, limit) split(delimiter, limit) .

```
String[] split = data.split("\\|", -1);
```

split(regex) split(regex, 0) .

limit .
n 0 *n - 1* *n* .
n , .
n , .

StringTokenizer

split() StringTokenizer .

StringTokenizer String.split() . (String) . . String.split() .

(\t\n\r\f). .

```
String str = "the lazy fox jumped over the brown fence";
StringTokenizer tokenizer = new StringTokenizer(str);
while (tokenizer.hasMoreTokens()) {
    System.out.println(tokenizer.nextToken());
}
```

.

```
the
lazy
fox
jumped
over
the
brown
fence
```

.

```
String str = "jumped over";
// In this case character `u` and `e` will be used as delimiters
StringTokenizer tokenizer = new StringTokenizer(str, "ue");
while (tokenizer.hasMoreTokens()) {
    System.out.println(tokenizer.nextToken());
}
```

.

```
j
mp
d ov
r
```

Java SE 8

String.join() .

```
String[] elements = { "foo", "bar", "foobar" };
String singleString = String.join(" + ", elements);

System.out.println(singleString); // Prints "foo + bar + foobar"
```

Iterable String.join() .

StringJoiner .

```
StringJoiner sj = new StringJoiner(", ", "[", "]");
    // The last two arguments are optional,
```

```
    // they define prefix and suffix for the result string

sj.add("foo");
sj.add("bar");
sj.add("foobar");

System.out.println(sj); // Prints "[foo, bar, foobar]"
```

.

```
Stream<String> stringStream = Stream.of("foo", "bar", "foobar");
String joined = stringStream.collect(Collectors.joining(", "));
System.out.println(joined); // Prints "foo, bar, foobar"
```

.

```
Stream<String> stringStream = Stream.of("foo", "bar", "foobar");
String joined = stringStream.collect(Collectors.joining(", ", "{", "}"));
System.out.println(joined); // Prints "{foo, bar, foobar}"
```

.

1. StringBuilder / StringBuffer :

```
 String code = "code";
 System.out.println(code);

 StringBuilder sb = new StringBuilder(code);
 code = sb.reverse().toString();

 System.out.println(code);
```

2. :

```
String code = "code";
System.out.println(code);

char[] array = code.toCharArray();
for (int index = 0, mirroredIndex = array.length - 1; index < mirroredIndex; index++,
mirroredIndex--) {
    char temp = array[index];
    array[index] = array[mirroredIndex];
    array[mirroredIndex] = temp;
}

// print reversed
System.out.println(new String(array));
```

org.apache.commons.lang3.StringUtils `countMatches` `String` .

```
import org.apache.commons.lang3.StringUtils;

String text = "One fish, two fish, red fish, blue fish";
```

```
// count occurrences of a substring
String stringTarget = "fish";
int stringOccurrences = StringUtils.countMatches(text, stringTarget); // 4

// count occurrences of a char
char charTarget = ',';
int charOccurrences = StringUtils.countMatches(text, charTarget); // 3
```

Java API .

```
import java.util.regex.Matcher;
import java.util.regex.Pattern;

String text = "One fish, two fish, red fish, blue fish";
System.out.println(countStringInString("fish", text)); // prints 4
System.out.println(countStringInString(",", text)); // prints 3


public static int countStringInString(String search, String text) {
    Pattern pattern = Pattern.compile(search);
    Matcher matcher = pattern.matcher(text);

    int stringOccurrences = 0;
    while (matcher.find()) {
      stringOccurrences++;
    }
    return stringOccurrences;
}
```

StringBuilders

+ . :

```
String s1 = "a";
String s2 = "b";
String s3 = "c";
String s = s1 + s2 + s3; // abc
```

StringBuilder . .

```
StringBuilder sb = new StringBuilder("a");
String s = sb.append("b").append("c").toString();
```

StringBuilder . , String int . , .

```
String s1 = "a";
String s2 = "b";
String s = s1 + s2 + 2; // ab2
```

:

```
StringBuilder sb = new StringBuilder("a");
String s = sb.append("b").append(2).toString();
```

. StringBuilder . .

```
String result = "";
for(int i = 0; i < array.length; i++) {
    result += extractElement(array[i]);
}
return result;
```

StringBuilder . StringBuilder .

```
StringBuilder result = new StringBuilder();
for(int i = 0; i < array.length; i++) {
    result.append(extractElement(array[i]));
}
return result.toString();
```

StringBuilder 16 . .

```
StringBuilder buf = new StringBuilder(30); // Default is 16 characters
buf.append("0123456789");
buf.append("0123456789"); // Would cause a reallocation of the internal buffer otherwise
String result = buf.toString(); // Produces a 20-chars copy of the string
```

StringBuilder .

```
StringBuilder buf = new StringBuilder(100);
for (int i = 0; i < 100; i++) {
    buf.setLength(0); // Empty buffer
    buf.append("This is line ").append(i).append('\n');
    outputfile.write(buf.toString());
}
```

() StringBuffer A, synchronized StringBuilder . StringBuilder .

concat () :

```
String string1 = "Hello ";
String string2 = "world";
String string3 = string1.concat(string2);  // "Hello world"
```

string2 string1 . concat () .

```
"My name is ".concat("Buyya");
```

.

: String String .

:

```
String replace(char oldChar, char newChar)
```

oldChar newChar .

```
String s = "popcorn";
System.out.println(s.replace('p','W'));
```

:

```
WoWcorn
```

.

```
String replace(CharSequence target, CharSequence replacement)
```

.

```
String s = "metal petal et al.";
System.out.println(s.replace("etal","etallica"));
```

:

```
metallica petallica et al.
```

: $ $1 .

:

```
String replaceAll(String regex, String replacement)
```

.

```
String s = "spiral metal petal et al.";
System.out.println(s.replaceAll("(\\w*etal)","$1lica"));
```

:

```
spiral metallica petallica et al.
```

:

```
String replaceFirst(String regex, String replacement)
```

.

```
String s = "spiral metal petal et al.";
System.out.println(s.replaceAll("(\\w*etal)","$1lica"));
```

:

```
spiral metallica petal et al.
```

`trim()` String .

```
String s = new String("   Hello World!!  ");
String t = s.trim();  // t = "Hello World!!"
```

`trim` String .

`trim()` `Character.isWhitespace()` .

- `U+0000` `U+0020` ASCII `trim()` . `U+0020` 'SPACE', `U+0009` 'CHARACTER TABULATION', `U+000A` 'LINE FEED' `U+000D` 'CARRIAGE RETURN' `U+0007` 'BELL' .
- `U+00A0` 'NO-BREAK SPACE' `U+2003` 'EM SPACE' `trim()` .

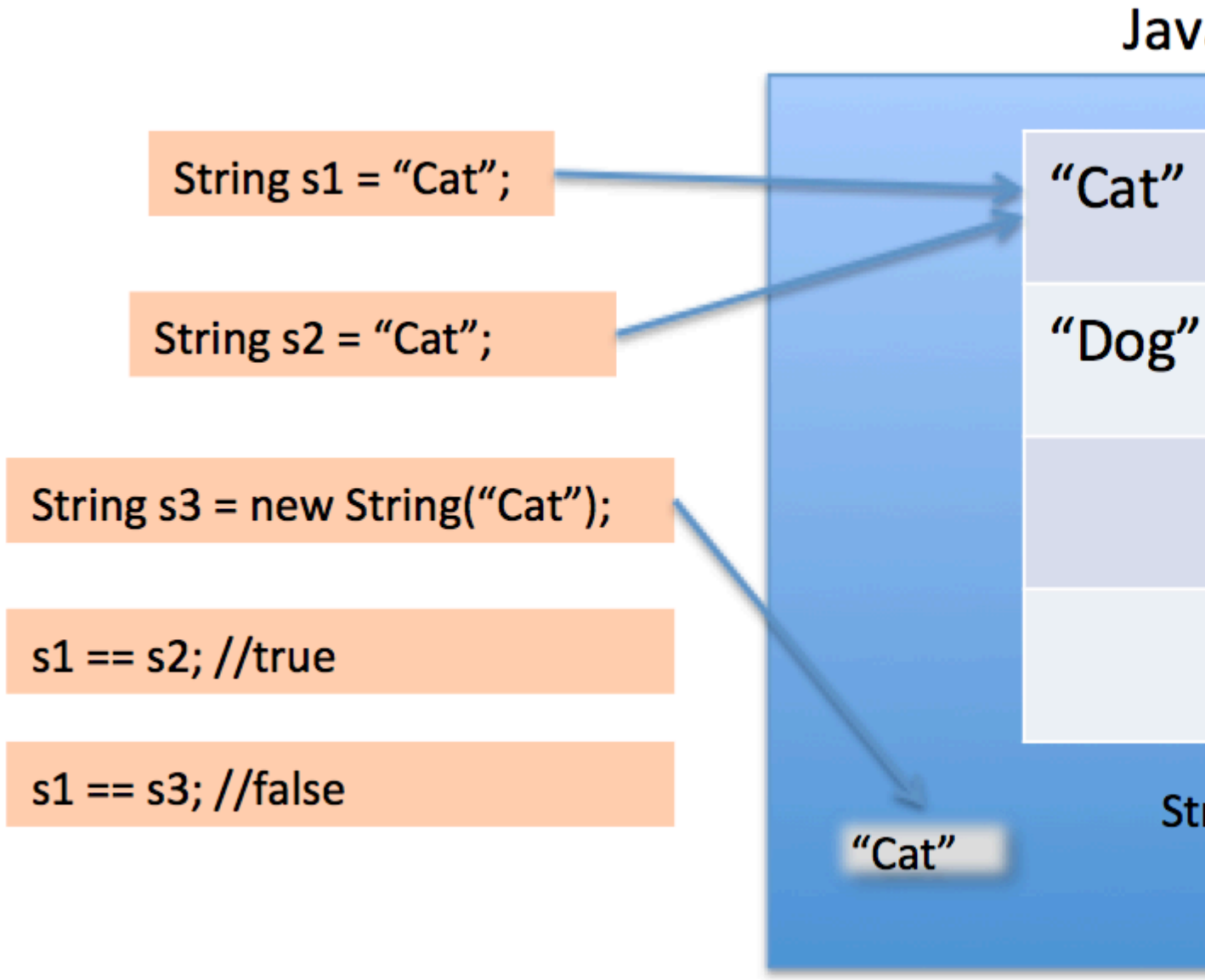
Java , `String` , . JVM `String` JVM `String` `String` . `String` `String` .

.

```
class Strings
{
    public static void main (String[] args)
    {
        String a = "alpha";
        String b = "alpha";
        String c = new String("alpha");

        //All three strings are equivalent
        System.out.println(a.equals(b) && b.equals(c));

        //Although only a and b reference the same heap object
        System.out.println(a == b);
        System.out.println(a != c);
        System.out.println(b != c);
    }
}
```

.

```
true
true
true
true
```



String String . String .

new String String . intern () String .

String .

Java SE 7

Java 7 `String` `PermGen` .

String `PermGen` .

Java SE 7

RFC : 6962931

> JDK 7 interned Java Java ( ) . Java . `String.intern()` .

Java SE 7

`switch` / `toLowerCase()` `toUpperCase` .

```
switch (myString.toLowerCase()) {
     case "case1" :
        ...
     break;
     case "case2" :
        ...
     break;
}
```

- `Locale` !
- . !

: https://riptutorial.com/ko/java/topic/109/

101:

Examples

String

- String `valueOf` String .

  :

```
int i = 42;
String string = String.valueOf(i);
//string now equals "42”.
```

  `float` , `double` , `boolean` `Object` .

- `.toString` Object ( ) String . `toString()` . Java `Date` .

  :

```
Foo foo = new Foo(); //Any class.
String stringifiedFoo = foo.toString().
```

  `stringifiedFoo` `foo` String .

String .

```
int i = 10;
String str = i + "";
```

```
String str = 10 + "";
```

Java `String#getBytes()` .

```
byte[] bytes = "test".getBytes(StandardCharsets.UTF_8);
```

:

```
String testString = new String(bytes, StandardCharsets.UTF_8);
```

.

```
import static java.nio.charset.StandardCharsets.UTF_8;
...
byte[] bytes = "test".getBytes(UTF_8);
```

.

```
byte[] bytes = "test".getBytes("UTF-8");
```

:

```
String testString = new String (bytes, "UTF-8");
```

UnsupportedCharsetException .

. Windows, Mac Linux .

```
byte[] bytes = "test".getBytes();
```

:

```
String testString = new String(bytes);
```

, . CharsetEncoder CharsetDecoder .

Base64 /

base64 .

javax.xml.bind DatatypeConverter .

```
import javax.xml.bind.DatatypeConverter;
import java.util.Arrays;

// arbitrary binary data specified as a byte array
byte[] binaryData = "some arbitrary data".getBytes("UTF-8");

// convert the binary data to the base64-encoded string
String encodedData = DatatypeConverter.printBase64Binary(binaryData);
// encodedData is now "c29tZSBhcmJpdHJhcnkgZGF0YQ=="

// convert the base64-encoded string back to a byte array
byte[] decodedData = DatatypeConverter.parseBase64Binary(encodedData);

// assert that the original data and the decoded data are equal
assert Arrays.equals(binaryData, decodedData);
```

Apache commons-codec

Apache commons-codec Base64 .

```
import org.apache.commons.codec.binary.Base64;

// your blob of binary as a byte array
```

```
byte[] blob = "someBinaryData".getBytes();

// use the Base64 class to encode
String binaryAsAString = Base64.encodeBase64String(blob);

// use the Base64 class to decode
byte[] blob2 = Base64.decodeBase64(binaryAsAString);

// assert that the two blobs are equal
System.out.println("Equal : " + Boolean.toString(Arrays.equals(blob, blob2)));
```

`c29tZUJpbmFyeURhdGE=` `c29tZUJpbmFyeURhdGE=` *UTF-8* String `c29tZUJpbmFyeURhdGE=` `someBinaryData` .

Java SE 8

Base64 .

```
// encode with padding
String encoded = Base64.getEncoder().encodeToString(someByteArray);

// encode without padding
String encoded = Base64.getEncoder().withoutPadding().encodeToString(someByteArray);

// decode a String
byte [] barr = Base64.getDecoder().decode(encoded);
```

.

, String parseXxx . Integer.parseInt String int .

```
String string = "59";
int primitive = Integer.parseInteger(string);
```

String valueOf .

```
String string = "59";
Integer wrapper = Integer.valueOf(string);
```

(5) :

```
String string = "59";
Integer wrapper = Integer.parseInteger(string);  // 'int' result is autoboxed
```

byte , short , int , long , float double (Byte , Short , Integer , Long , Float Double) .

String to Integer :

```
String integerAsString = "0101"; // binary representation
int parseInt = Integer.parseInt(integerAsString,2);
Integer valueOfInteger = Integer.valueOf(integerAsString,2);
System.out.println(valueOfInteger); // prints 5
System.out.println(parseInt); // prints 5
```

`valueOf(String)` `valueOf(String)` `parseXxx(...)` NumberFormatException .

`InputStream``String` .

`InputStream` `String` .

```
import java.io.*;

public String readString(InputStream input) throws IOException {
    byte[] bytes = new byte[50]; // supply the length of the string in bytes here
    input.read(bytes);
    return new String(bytes);
}
```

charset charset .

```
return new String(bytes, Charset.forName("UTF-8"));
```

.

Java .

int String :

```
String number = "12";
int num = Integer.parseInt(number);
```

:

```
String number = "12.0";
float num = Float.parseFloat(number);
```

:

```
String double = "1.47";
double num = Double.parseDouble(double);
```

:

```
String falseString = "False";
boolean falseBool = Boolean.parseBoolean(falseString);   // falseBool = false

String trueString = "True";
boolean trueBool = Boolean.parseBoolean(trueString);     // trueBool = true
```

:

```
String number = "47";
long num = Long.parseLong(number);
```

String to BigInteger :

```
String bigNumber = "21";
BigInteger reallyBig = new BigInteger(bigNumber);
```

String to BigDecimal :

```
String bigFraction = "17.21455";
BigDecimal reallyBig = new BigDecimal(bigFraction);
```

:

( ) NumberFormatException . .

tryParse... .

```
boolean tryParseInt (String value) {
    try {
        String somechar = Integer.parseInt(value);
        return true;
     } catch (NumberFormatException e) {
        return false;
     }
}
```

tryParse... () .parse... .

: https://riptutorial.com/ko/java/topic/6678/---

102:

() . . 8 (:).

```
a b c d e f g h i j k l m n o p q r s t u v w x y z
              ^               ^               ^
```

.

Examples

.

"" \G .

```
String[] parts = str.split("(?<=\\G.{8})");
```

8. 0 " 8" .

\G .

.

.

```
int length = 5;
String[] parts = str.split("(?<=\\G.{" + length + "})");
```

: https://riptutorial.com/ko/java/topic/5613/----

103:

Examples

?

JVM . Hello World .

```
public static void main(String[] args){
    System.out.println("Hello World");
}
```

.

```
public static main([Ljava/lang/String; args)V
    getstatic java/lang/System out Ljava/io/PrintStream;
    ldc "Hello World"
    invokevirtual java/io/PrintStream print(Ljava/lang/String;)V
```

?

getstatic - . *PrintStream* "" .

ldc - . , "Hello World"

invokevirtual - . .

, ?

255 opcode . opcode Java .

?

. , , .

:

-
-

:

-
  - ASM
  - Javassist
  - BCEL - *Java 8+* .
  -
-

-
  - JBytedit
  - reJ - *Java 8+ .*
  - JBE - *Java 8+ .*

!

. .

ASM jar

. .

- loadClasses (File)
- readJar (JarFile, JarEntry, Map)
- getNode (byte [])

```
Map<String, ClassNode> loadClasses(File jarFile) throws IOException {
    Map<String, ClassNode> classes = new HashMap<String, ClassNode>();
    JarFile jar = new JarFile(jarFile);
    Stream<JarEntry> str = jar.stream();
    str.forEach(z -> readJar(jar, z, classes));
    jar.close();
    return classes;
}

Map<String, ClassNode> readJar(JarFile jar, JarEntry entry, Map<String, ClassNode> classes) {
    String name = entry.getName();
    try (InputStream jis = jar.getInputStream(entry)){
        if (name.endsWith(".class")) {
            byte[] bytes = IOUtils.toByteArray(jis);
            String cafebabe = String.format("%02X%02X%02X%02X", bytes[0], bytes[1], bytes[2],
bytes[3]);
            if (!cafebabe.toLowerCase().equals("cafebabe")) {
                // This class doesn't have a valid magic
                return classes;
            }
            try {
                ClassNode cn = getNode(bytes);
                classes.put(cn.name, cn);
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    } catch (IOException e) {
        e.printStackTrace();
    }
    return classes;
}

ClassNode getNode(byte[] bytes) {
    ClassReader cr = new ClassReader(bytes);
    ClassNode cn = new ClassNode();
    try {
        cr.accept(cn, ClassReader.EXPAND_FRAMES);
```

```
    } catch (Exception e) {
        e.printStackTrace();
    }
    cr = null;
    return cn;
}
```

jar ClassNodes . jar Tree API .

```
File jarFile = new File("sample.jar");
Map<String, ClassNode> nodes = loadClasses(jarFile);
// Iterate ClassNodes
for (ClassNode cn : nodes.values()){
    // Iterate methods in class
    for (MethodNode mn : cn.methods){
        // Iterate instructions in method
        for (AbstractInsnNode ain : mn.instructions.toArray()){
            // If the instruction is loading a constant value
            if (ain.getOpcode() == Opcodes.LDC){
                // Cast current instruction to Ldc
                // If the constant is a string then capitalize it.
                LdcInsnNode ldc = (LdcInsnNode) ain;
                if (ldc.cst instanceof String){
                    ldc.cst = ldc.cst.toString().toUpperCase();
                }
            }
        }
    }
}
```

ClassNode . .

- ClassNodes
- jar *(: Manifest.mf / jar 2)* .
- .

.

- processNodes (Map <String, ClassNode> nodes)
- loadNonClasses (File jarFile)
- saveAsJar (Map <String, byte []> outBytes, fileName)

:

```
Map<String, byte[]> out = process(nodes, new HashMap<String, MappedClass>());
out.putAll(loadNonClassEntries(jarFile));
saveAsJar(out, "sample-edit.jar");
```

:

```
static Map<String, byte[]> processNodes(Map<String, ClassNode> nodes, Map<String, MappedClass>
mappings) {
    Map<String, byte[]> out = new HashMap<String, byte[]>();
    // Iterate nodes and add them to the map of <Class names , Class bytes>
    // Using Compute_Frames ensures that stack-frames will be re-calculated automatically
```

```
    for (ClassNode cn : nodes.values()) {
        ClassWriter cw = new ClassWriter(ClassWriter.COMPUTE_FRAMES);
        out.put(mappings.containsKey(cn.name) ? mappings.get(cn.name).getNewName() : cn.name,
cw.toByteArray());
    }
    return out;
}

static Map<String, byte[]> loadNonClasses(File jarFile) throws IOException {
    Map<String, byte[]> entries = new HashMap<String, byte[]>();
    ZipInputStream jis = new ZipInputStream(new FileInputStream(jarFile));
    ZipEntry entry;
    // Iterate all entries
    while ((entry = jis.getNextEntry()) != null) {
        try {
            String name = entry.getName();
            if (!name.endsWith(".class") && !entry.isDirectory()) {
                // Apache Commons - byte[] toByteArray(InputStream input)
                //
                // Add each entry to the map <Entry name , Entry bytes>
                byte[] bytes = IOUtils.toByteArray(jis);
                entries.put(name, bytes);
            }
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            jis.closeEntry();
        }
    }
    jis.close();
    return entries;
}

static void saveAsJar(Map<String, byte[]> outBytes, String fileName) {
    try {
        // Create jar output stream
        JarOutputStream out = new JarOutputStream(new FileOutputStream(fileName));
        // For each entry in the map, save the bytes
        for (String entry : outBytes.keySet()) {
            // Appent class names to class entries
            String ext = entry.contains(".") ? "" : ".class";
            out.putNextEntry(new ZipEntry(entry + ext));
            out.write(outBytes.get(entry));
            out.closeEntry();
        }
        out.close();
    } catch (IOException e) {
        e.printStackTrace();
    }
}
```

. "sample-edit.jar" .

ClassNode

```
/**
 * Load a class by from a ClassNode
 *
 * @param cn
```

```
 *            ClassNode to load
 * @return
 */
public static Class<?> load(ClassNode cn) {
    ClassWriter cw = new ClassWriter(ClassWriter.COMPUTE_FRAMES);
    return new ClassDefiner(ClassLoader.getSystemClassLoader()).get(cn.name.replace("/", "."),
cw.toByteArray());
}

/**
 * Classloader that loads a class from bytes.
 */
static class ClassDefiner extends ClassLoader {
    public ClassDefiner(ClassLoader parent) {
        super(parent);
    }

    public Class<?> get(String name, byte[] bytes) {
        Class<?> c = defineClass(name, bytes, 0, bytes.length);
        resolveClass(c);
        return c;
    }
}
```

jar

```
public static void main(String[] args) throws Exception {
    File jarFile = new File("Input.jar");
    Map<String, ClassNode> nodes = JarUtils.loadClasses(jarFile);

    Map<String, byte[]> out = JarUtils.loadNonClassEntries(jarFile);
    Map<String, String> mappings = new HashMap<String, String>();
    mappings.put("me/example/ExampleClass", "me/example/ExampleRenamed");
    out.putAll(process(nodes, mappings));
    JarUtils.saveAsJar(out, "Input-new.jar");
}

static Map<String, byte[]> process(Map<String, ClassNode> nodes, Map<String, String> mappings)
{
    Map<String, byte[]> out = new HashMap<String, byte[]>();
    Remapper mapper = new SimpleRemapper(mappings);
    for (ClassNode cn : nodes.values()) {
        ClassWriter cw = new ClassWriter(ClassWriter.COMPUTE_FRAMES);
        ClassVisitor remapper = new ClassRemapper(cw, mapper);
        cn.accept(remapper);
        out.put(mappings.containsKey(cn.name) ? mappings.get(cn.name) : cn.name,
cw.toByteArray());
    }
    return out;
}
```

SimpleRemapper ASM . . Remapper .

Javassist Basic

Javassist , Javassist / Java .

"com.my.to.be.instrumented.MyClass" .

```
import java.lang.instrument.ClassFileTransformer;
import java.lang.instrument.IllegalClassFormatException;
import java.security.ProtectionDomain;
import javassist.ClassPool;
import javassist.CtClass;
import javassist.CtMethod;

public class DynamicTransformer implements ClassFileTransformer {

    public byte[] transform(ClassLoader loader, String className, Class classBeingRedefined,
        ProtectionDomain protectionDomain, byte[] classfileBuffer) throws
IllegalClassFormatException {

        byte[] byteCode = classfileBuffer;

        // into the transformer will arrive every class loaded so we filter
        // to match only what we need
        if (className.equals("com/my/to/be/instrumented/MyClass")) {

            try {
                // retrive default Javassist class pool
                ClassPool cp = ClassPool.getDefault();
                // get from the class pool our class with this qualified name
                CtClass cc = cp.get("com.my.to.be.instrumented.MyClass");
                // get all the methods of the retrieved class
                CtMethod[] methods = cc.getDeclaredMethods()
                for(CtMethod meth : methods) {
                    // The instrumentation code to be returned and injected
                    final StringBuffer buffer = new StringBuffer();
                    String name = meth.getName();
                    // just print into the buffer a log for example
                    buffer.append("System.out.println(\"Method " + name + " executed\" );");
                    meth.insertBefore(buffer.toString())
                }
                // create the byteclode of the class
                byteCode = cc.toBytecode();
                // remove the CtClass from the ClassPool
                cc.detach();
            } catch (Exception ex) {
                ex.printStackTrace();
            }
        }

        return byteCode;
    }
}
```

(JVM) .

```
import java.lang.instrument.Instrumentation;

public class EasyAgent {

    public static void premain(String agentArgs, Instrumentation inst) {

        // registers the transformer
        inst.addTransformer(new DynamicTransformer());
```

```
    }
}
```

JVM . .

```
java -javaagent:myAgent.jar MyJavaApplication
```

, / "com.my.to.be.instrumented.MyClass" MyJavaApplication jar .

: https://riptutorial.com/ko/java/topic/3747/--

104:

API 1.5 `ProcessBuilder.start()` .

`waitFor` / . , **calc.exe** **notepad.exe** .

Examples

(Java <1.5)

. / .

```
package process.example;

import java.io.IOException;

public class App {

    public static void main(String[] args) {
        try {
            // Executes windows calculator
            Process p = Runtime.getRuntime().exec("calc.exe");

            // Wait for process until it terminates
            int exitCode = p.waitFor();

            System.out.println(exitCode);
        } catch (IOException e) {
            e.printStackTrace();
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }
}
```

ProcessBuilder

ProcessBuilder . . ProcessBuilder start () .

Add.exe .

```
List<String> cmds = new ArrayList<>();
cmds.add("Add.exe"); //the name of the application to be run
cmds.add("1"); //the first argument
cmds.add("5"); //the second argument

ProcessBuilder pb = new ProcessBuilder(cmds);

//Set the working directory of the ProcessBuilder so it can find the .exe
//Alternatively you can just pass in the absolute file path of the .exe
File myWorkingDirectory = new File(yourFilePathNameGoesHere);
pb.workingDirectory(myWorkingDirectory);
```

```
try {
    Process p = pb.start();
} catch (IOException e) {
    e.printStackTrace();
}
```

:

- .
- (, .exe)
- String .

.

(:).

```
Process waitFor()    .
```

:

```
//code setting up the commands omitted for brevity...

ProcessBuilder pb = new ProcessBuilder(cmds);

try {
    Process p = pb.start();
    p.waitFor();
} catch (IOException e) {
    e.printStackTrack();
} catch (InterruptedException e) {
    e.printStackTrace();
}

//more lines of code here...
```

ch.vorburger.exec

java.lang.ProcessBuilder API Java . Apache Commons Exec . ch.vorburger.exec Commons Exec .

```
 ManagedProcess proc = new ManagedProcessBuilder("path-to-your-executable-binary")
     .addArgument("arg1")
     .addArgument("arg2")
     .setWorkingDirectory(new File("/tmp"))
     .setDestroyOnShutdown(true)
     .setConsoleBufferMaxLines(7000)
     .build();

proc.start();
int status = proc.waitForExit();
int status = proc.waitForExitMaxMsOrDestroy(3000);
String output = proc.getConsole();

proc.startAndWaitForConsoleMessageMaxMs("started!", 7000);
// use service offered by external process...
```

```
proc.destroy();
```

Pitfall : Runtime.exec, Process ProcessBuilder .

`Runtime.exec(String ...)` `Runtime.exec(String)` [1] . Java OS . .

, `exec(String)` :

```
Process p = Runtime.exec("mkdir /tmp/testDir");
p.waitFor();
if (p.exitValue() == 0) {
    System.out.println("created the directory");
}
```

.

```
Process p = Runtime.exec("mkdir " + dirPath);
// ...
```

`dirPath` "/ home / user / My Documents" . `exec(String)` . :

```
"mkdir /home/user/My Documents"
```

.

```
"mkdir", "/home/user/My", "Documents"
```

"mkdir" .

. :

```
"mkdir \"/home/user/My Documents\""
```

.

```
"mkdir", "\"/home/user/My", "Documents\""
```

"" ., .

`exec(String ...)` .

```
Process p = Runtime.exec("mkdir", dirPath);
// ...
```

`dirpath` `dirpath exec` . OS `exec` .

,

. :

```
Process p = Runtime.exec("find / -name *.java -print 2>/dev/null");
```

```
Process p = Runtime.exec("find source -name *.java | xargs grep package");
```

Java "" Java `package` [2] .

. "find" "2> / dev / null" . . ("|") "" .

`exec ProcessBuilder` . , , , globbing .

(:) `ProcessBuilder` . . . :

```
Process p = Runtime.exec("bash", "-c",
                         "find / -name *.java -print 2>/dev/null");
```

```
Process p = Runtime.exec("bash", "-c",
                         "find source -name \\*.java | xargs grep package");
```

("*") . "" .

▪

UNIX .

```
Process p = Runtime.exec("cd", "/tmp");     // Change java app's home directory
```

```
Process p = Runtime.exec("export", "NAME=value");  // Export NAME to the java app's
environment
```

.

1. "cd" "export" . .
2. (: ,) . (Java) . , "cd" "java" . , `exec` .

.

---

1 - `ProcessBuilder` .

2 - . .

: https://riptutorial.com/ko/java/topic/4682/

105:

. . . int Object .

- `ArrayType[] myArray;` //
- `ArrayType myArray[];` // ( )
- `ArrayType[][][] myArray;` // ([])
- `ArrayType myVar = myArray[index];` // ()
- `myArray[index] = value;` // index .
- `ArrayType[] myArray = new ArrayType[arrayLength];` //
- `int[] ints = {1, 2, 3};` // : {[value1 [, value2] *]}
- `new int[]{4, -5, 6} // Can be used as argument, without a local variable`
- `int[] ints = new int[3]; // same as {0, 0, 0}`
- `int[][] ints = {{1, 2}, {3}, null};` // . int [] Object anyType [] null .

`ArrayType`	. (`int` , `long` , `byte`) Objects (`String` , `MyObject`) .
	.
	. (`new int[3]`) (`{1, 2, 3}`) .

Examples

```
int[]   numbers1 = new int[3];                // Array for 3 int values, default value is 0
int[]   numbers2 = { 1, 2, 3 };               // Array literal of 3 int values
int[]   numbers3 = new int[] { 1, 2, 3 };     // Array of 3 int values initialized
int[][] numbers4 = { { 1, 2 }, { 3, 4, 5 } }; // Jagged array literal
int[][] numbers5 = new int[5][];              // Jagged array, one dimension 5 long
int[][] numbers6 = new int[5][4];             // Multidimensional array: 5x4
```

.

```
float[]  boats = new float[5];          // Array of five 32-bit floating point numbers.
double[] header = new double[] { 4.56, 332.267, 7.0, 0.3367, 10.0 };
                                        // Array of five 64-bit floating point numbers.
String[] theory = new String[] { "a", "b", "c" };
                                        // Array of three strings (reference type).
Object[] dArt = new Object[] { new Object(), "We love Stack Overflow.", new Integer(3) };
                                        // Array of three Objects (reference type).
```

.

.

```
UserDefinedClass[] udType = new UserDefinedClass[5];
```

,

Java SE 1.2

```
// Parameters require objects, not primitives

// Auto-boxing happening for int 127 here
Integer[]       initial        = { 127, Integer.valueOf( 42 ) };
List<Integer>   toList         = Arrays.asList( initial );  // Fixed size!

// Note: Works with all collections
Integer[]       fromCollection = toList.toArray( new Integer[toList.size()] );

//Java doesn't allow you to create an array of a parameterized type
List<String>[]  list = new ArrayList<String>[2];  // Compilation error!
```

Java SE 8

```
// Streams - JDK 8+
Stream<Integer> toStream       = Arrays.stream( initial );
Integer[]       fromStream     = toStream.toArray( Integer[]::new );
```

.

. .

```
int size = 42;
int[] array = new int[size];
```

. . ArrayList Collection .ArrayList .

,

```
int[] array1 = { 1,2,3 };
int[] array2 = new int[10];
```

. array2 0 .

, .

```
SomeClassOrInterface[] array = new SomeClassOrInterface[10];
```

SomeClassOrInterface . SomeClassOrInterface () () *SomeClassOrInterface* . null .

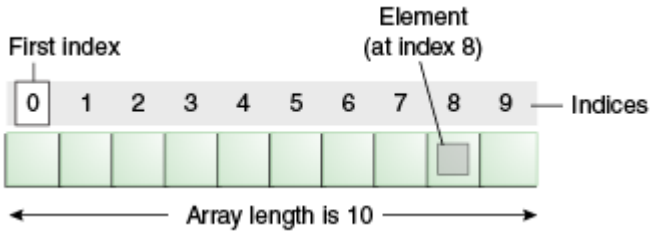
int -indexed, int . long .

```
long size = 23L;
int[] array = new int[size]; // Compile-time error:
                             // incompatible types: possible lossy conversion from
```

```
                                // long to int
```

0 length - 1 0 .

10 . 1 10 ( ). 0 9 .



. , , () .

Java . new Type[length] .

- 0 : byte , short , int , long , float , double .
- char '\u0000' (null).
- boolean false
- null

```
int[] array1 = new int[] { 1, 2, 3 }; // Create an array with new operator and
                                      // array initializer.
int[] array2 = { 1, 2, 3 };           // Shortcut syntax with array initializer.
int[] array3 = new int[3];            // Equivalent to { 0, 0, 0 }
int[] array4 = null;                  // The array itself is an object, so it
                                      // can be set as null.
```

[] ( ) ( ) .

```
int array5[];       /* equivalent to */  int[] array5;
int a, b[], c[][];  /* equivalent to */  int a; int[] b; int[][] c;
int[] a, b[];       /* equivalent to */  int[] a; int[][] b;
int a, []b, c[][];  /* Compilation Error, because [] is not part of the type at beginning
                       of the declaration, rather it is before 'b'. */
// The same rules apply when declaring a method that returns an array:
int foo()[] { ... } /* equivalent to */  int[] foo() { ... }
```

. Java Google Java . . .

```
float array[]; /* and */ int foo()[] { ... } /* are discouraged */
float[] array; /* and */ int[] foo() { ... } /* are encouraged */
```

C C .

Java 0 .

```
int[] array = new int[0]; // Compiles and runs fine.
int[] array2 = {};        // Equivalent syntax.
```

.

```
array[0] = 1;     // Throws java.lang.ArrayIndexOutOfBoundsException.
int i = array2[0]; // Also throws ArrayIndexOutOfBoundsException.
```

NullPointerException null .

.

```
int[] array = new int[-1]; // Throws java.lang.NegativeArraySizeException
```

length public .

```
System.out.println(array.length); // Prints 0 in this case.
```

: array.length ArrayList.size() .

.

```
int[][] a = new int[2][3];
```

a[0] a[1] 3 int . C .

.

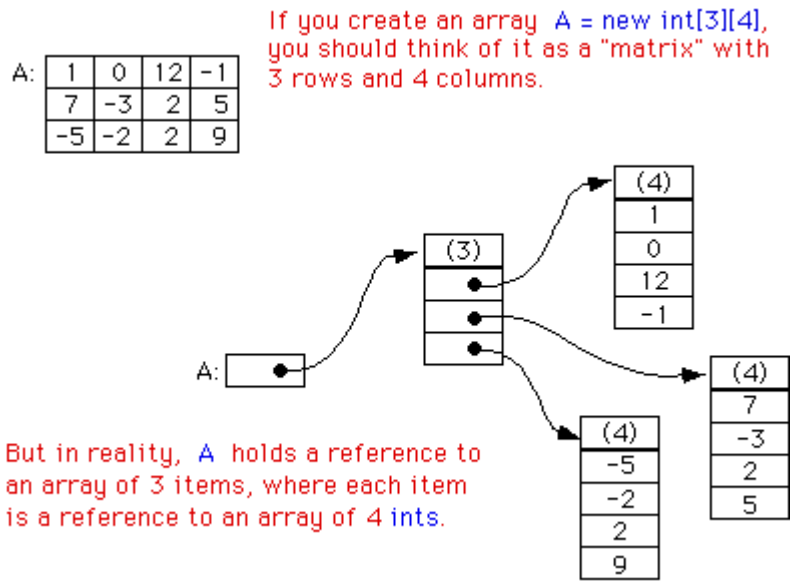
```
int[][] a = { {1, 2}, {3, 4}, {5, 6} };
```

C .

```
int[][] a = { {1}, {2, 3}, null };
```

a[0] 1 int a[1] 2 int a[2] null . . , . Java . , array[i][j][k] ((array[i])[j])[k] . C # array[i,j] Java .

Java



- Ideone

```
String[] array6 = new String[] { "Laurel", "Hardy" }; // Create an array with new
                                                      // operator and array initializer.
String[] array7 = { "Laurel", "Hardy" };              // Shortcut syntax with array
                                                      // initializer.
String[] array8 = new String[3];                      // { null, null, null }
String[] array9 = null;                               // null
```

`String` `Object` .

```
Object[] array10 = { new Object(), new Object() };
```

(covariant) `String` `ArrayStoreException` throw .

```
Object[] array11 = new String[] { "foo", "bar", "baz" };
array11[1] = "qux"; // fine
array11[1] = new StringBuilder(); // throws ArrayStoreException
```

`Object[]` .

`String[] emptyArray = new String[0]` 0 . 0 Array Collection Array .

(: `String[] array8 = new String[3]`) .

.

```
public class MyGenericClass<T> {
    private T[] a;

    public MyGenericClass() {
        a = new T[5]; // Compile time error: generic array creation
    }
```

```
}
```

( )

1. `Object` generic .

   ```
   a = (T[]) new Object[5];
   ```

   `Object[]` `Object[]` . .

2. `Array.newInstance` :

   ```
   public MyGenericClass(Class<T> clazz) {
       a = (T[]) Array.newInstance(clazz, 5);
   }
   ```

   `T` . `Array.newInstance` `Object` . `T[]` .

Java SE 1.2

`Arrays.fill()` :

```
Arrays.fill(array8, "abc");         // { "abc", "abc", "abc" }
```

`fill()` .

```
Arrays.fill(array8, 1, 2, "aaa");  // Placing "aaa" from index 1 to 2.
```

Java SE 8

Java 8, `setAll` `Concurrent parallelSetAll` . .

.

```
int[] array = new int[5];
Arrays.setAll(array, i -> i); // The array becomes { 0, 1, 2, 3, 4 }.
```

(0, 1, 2, 3, 4, ...) (0). , ArrayIndexOutOfBoundsException Throw.

```
int[] array9;              // Array declaration - uninitialized
array9 = new int[3];       // Initialize array  - { 0, 0, 0 }
array9[0] = 10;            // Set index 0 value - { 10, 0, 0 }
array9[1] = 20;            // Set index 1 value - { 10, 20, 0 }
array9[2] = 30;            // Set index 2 value - { 10, 20, 30 }
```

.

.

. . new .

```
// First initialization of array
int[] array = new int[] { 1, 2, 3 };

// Prints "1 2 3 ".
for (int i : array) {
    System.out.print(i + " ");
}

// Re-initializes array to a new int[] array.
array = new int[] { 4, 5, 6 };

// Prints "4 5 6 ".
for (int i : array) {
    System.out.print(i + " ");
}

array = { 1, 2, 3, 4 }; // Compile-time error! Can't re-initialize an array via shortcut
                        // syntax with array initializer.
```

java.util.Collection .

- Object[] toArray()
- <T> T[] toArray(T[] a)

Object[] toArray() .

Java SE 5

```
Set<String> set = new HashSet<String>();
set.add("red");
set.add("blue");

// although set is a Set<String>, toArray() returns an Object[] not a String[]
Object[] objectArray = set.toArray();
```

<T> T[] toArray(T[] a) .

Java SE 5

```
Set<String> set = new HashSet<String>();
set.add("red");
set.add("blue");

// The array does not need to be created up front with the correct size.
// Only the array type matters. (If the size is wrong, a new array will
// be created with the same type.)
String[] stringArray = set.toArray(new String[0]);
```

```
// If you supply an array of the same size as collection or bigger, it
// will be populated with collection values and returned (new array
// won't be allocated)
String[] stringArray2 = set.toArray(new String[set.size()]);
```

. ( ).

- `Object[] toArray()` arraycopy , ,arraycopy `T[] toArray(T[] a)` .
- `T[] toArray(new T[non-zero-size])` `T[] toArray(new T[0])` `T[] toArray(new T[0])` . . :
  .

Java SE 8

`Stream` Java SE 8 `Stream.toArray` Array `Stream` .

```
String[] strings = list.stream().toArray(String[]::new);
```

Java ArrayList 'String []' (*1* , *2*)

Java SE 5

Java 1.5 `String` . `Arrays.toString(Object[])` `Arrays.deepToString(Object[])` .

```
int[] arr = {1, 2, 3, 4, 5};
System.out.println(Arrays.toString(arr));      // [1, 2, 3, 4, 5]

int[][] arr = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};
System.out.println(Arrays.deepToString(arr));  // [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
```

`Arrays.toString()` `Object.toString()` `String` . . :

```
public class Cat { /* implicitly extends Object */
    @Override
    public String toString() {
      return "CAT!";
    }
}

Cat[] arr = { new Cat(), new Cat() };
System.out.println(Arrays.toString(arr));        // [CAT!, CAT!]
```

`toString()` ,Object `toString()` . . .

```
public class Dog {
    /* implicitly extends Object */
}

Dog[] arr = { new Dog() };
System.out.println(Arrays.toString(arr));        // [Dog@17ed40e0]
```

`Arrays.asList()` `List` . `List` .

```
String[] stringArray = {"foo", "bar", "baz"};
List<String> stringList = Arrays.asList(stringArray);
```

: ( ) . , . .

`java.util.ArrayList` `Collection` .

Java SE 5

```
String[] stringArray = {"foo", "bar", "baz"};
List<String> stringList = new ArrayList<String>(Arrays.asList(stringArray));
```

Java SE 7

Java SE 7 `<>` ( ) . . , `ArrayList` . Java Generics .

```
// Using Arrays.asList()

String[] stringArray = {"foo", "bar", "baz"};
List<String> stringList = new ArrayList<>(Arrays.asList(stringArray));

// Using ArrayList.addAll()

String[] stringArray = {"foo", "bar", "baz"};
ArrayList<String> list = new ArrayList<>();
list.addAll(Arrays.asList(stringArray));

// Using Collections.addAll()

String[] stringArray = {"foo", "bar", "baz"};
ArrayList<String> list = new ArrayList<>();
Collections.addAll(list, stringArray);
```

.

Java SE 8

```
// Using Streams

int[] ints = {1, 2, 3};
List<Integer> list = Arrays.stream(ints).boxed().collect(Collectors.toList());

String[] stringArray = {"foo", "bar", "baz"};
List<Object> list = Arrays.stream(stringArray).collect(Collectors.toList());
```

Arrays.asList ()

- `List` , `Arrays$ArrayList` (`Arrays`) `java.util.ArrayList` . `List` . , `UnsupportedOperationException` throw.

```
stringList.add("something"); // throws java.lang.UnsupportedOperationException
```

- `List` array-backed `List` `List` . .

```
List<String> modifiableList = new ArrayList<>(Arrays.asList("foo", "bar"));
```

- `int[]` (primitive) `<T> List<T> asList(T... a)` `<T> List<T> asList(T... a)` , `List<int[]>`

  . `List` , , (, `int[]` `Integer[]` `Arrays.asList`).

  `false` :

```
int[] arr = {1, 2, 3};      // primitive array of int
System.out.println(Arrays.asList(arr).contains(1));
```

  , `true` :

```
Integer[] arr = {1, 2, 3};  // object array of Integer (wrapper for int)
System.out.println(Arrays.asList(arr).contains(1));
```

  `Integer[]` `true` `true` :

```
System.out.println(Arrays.asList(1,2,3).contains(1));
```

. .

`[]` . , 2 `int` `int[][]` . 3 (`int[][][]`) .

---

2 :

```
int rows = 3;
int columns = 3;
int[][] table = new int[rows][columns];
```

. , `int` 0 .

```
table[0][0] = 0;
table[0][1] = 1;
table[0][2] = 2;
```

. .

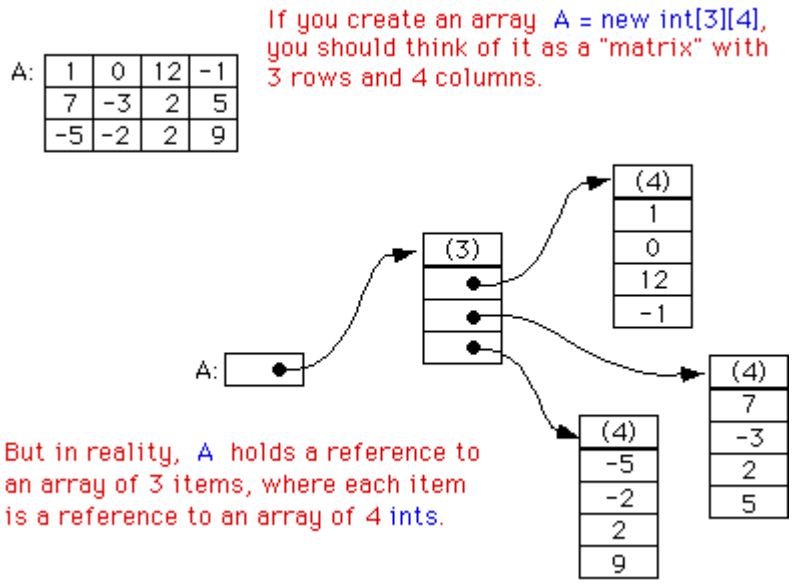
```
int[][] nonRect = new int[4][];
```

.

```
// valid
String[][] employeeGraph = new String[30][];
```

```
// invalid
int[][] unshapenMatrix = new int[][10];

// also invalid
int[][][] misshapenGrid = new int[100][][10];
```

Java



: *http://math.hws.edu/eck/cs124/javanotes3/c8/s5.html*

. 2x3 `int` .

```
int[][] table = {
    {1, 2, 3},
    {4, 5, 6}
};
```

: `null` . , `null` 0 2 `int` .

```
int[][] table = {
    null,
    {},
    {1},
    {1,2}
};
```

.

```
int[][][] arr = new int[3][3][3];
int[][] arr1 = arr[0]; // get first 3x3-dimensional array from arr
int[] arr2 = arr1[0]; // get first 3-dimensional array from arr1
int[] arr3 = arr[0]; // error: cannot convert from int[][] to int[]
```

ArrayIndexOutOfBoundsException

ArrayIndexOutOfBoundsException .

0 0 1 (array.length - 1).

, i , 0 <= i < array.length . , ArrayIndexOutOfBoundsException Throw.

ArrayIndexOutOfBoundsException .

```
String[] people = new String[] { "Carol", "Andy" };

// An array will be created:
// people[0]: "Carol"
// people[1]: "Andy"

// Notice: no item on index 2. Trying to access it triggers the exception:
System.out.println(people[2]);  // throws an ArrayIndexOutOfBoundsException.
```

:

```
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 2
    at your.package.path.method(YourClass.java:15)
```

(2). .

.

```
int index = 2;
if (index >= 0 && index < people.length) {
    System.out.println(people[index]);
}
```

. .

```
int[] arr1 = new int[0];
int[] arr2 = new int[2];
int[] arr3 = new int[]{1, 2, 3, 4};
int[] arr4 = {1, 2, 3, 4, 5, 6, 7};

int len1 = arr1.length; // 0
int len2 = arr2.length; // 2
int len3 = arr3.length; // 4
int len4 = arr4.length; // 7
```

length . final .

length .

```
public static void main(String[] args) {
```

```
    Integer arr[] = new Integer[] {1,2,3,null,5,null,7,null,null,null,11,null,13};

    int arrayLength = arr.length;
    int nonEmptyElementsCount = 0;

    for (int i=0; i<arrayLength; i++) {
        Integer arrElt = arr[i];
        if (arrElt != null) {
            nonEmptyElementsCount++;
        }
    }

    System.out.println("Array 'arr' has a length of "+arrayLength+"\n"
                           + "and it contains "+nonEmptyElementsCount+" non-empty values");
}
```

:

```
Array 'arr' has a length of 13
and it contains 7 non-empty values
```

`equals()` (`hashCode()` java.lang.Object) , `equals()` true . , `java.util.Arrays.equals` . , .

```
int[] a = new int[]{1, 2, 3};
int[] b = new int[]{1, 2, 3};
System.out.println(a.equals(b)); //prints "false" because a and b refer to different objects
System.out.println(Arrays.equals(a, b)); //prints "true" because the elements of a and b have
the same values
```

, `Arrays.equals()` `equals()` . ID . `Arrays.deepEquals()` .

```
int a[] = { 1, 2, 3 };
int b[] = { 1, 2, 3 };

Object[] aObject = { a }; // aObject contains one element
Object[] bObject = { b }; // bObject contains one element

System.out.println(Arrays.equals(aObject, bObject)); // false
System.out.println(Arrays.deepEquals(aObject, bObject));// true
```

`equals()` `hashCode()` set . `equals()` `hashCode()` List List .

Java SE 8

`Stream` :

```
String[] arr = new String[] {"str1", "str2", "str3"};
Stream<String> stream = Arrays.stream(arr);
```

`Arrays.stream()` `Stream` Stream .

```
int[] intArr = {1, 2, 3};
IntStream intStream = Arrays.stream(intArr);
```

`Stream` . .

```
int[] values = {1, 2, 3, 4};
IntStream intStream = Arrays.stream(values, 2, 4);
```

`Arrays.stream()` `Stream` . `Stream.of()` . `Stream.of()` varargs .

```
Stream<Integer> intStream = Stream.of(1, 2, 3);
Stream<String> stringStream = Stream.of("1", "2", "3");
Stream<Double> doubleStream = Stream.of(new Double[]{1.0, 2.0});
```

for (foreach) .

```
int[] array = new int[10];

// using indices: read and write
for (int i = 0; i < array.length; i++) {
    array[i] = i;
}
```

Java SE 5

```
// extended for: read only
for (int e : array) {
    System.out.println(e);
}
```

Array Iterator . Array `Iterable` .

Arrays.asList .

```
Integer[] boxed = {1, 2, 3};
Iterable<Integer> boxedIt = Arrays.asList(boxed); // list-backed iterable
Iterator<Integer> fromBoxed1 = boxedIt.iterator();
```

(8) (Arrays.stream -> IntStream) .

```
int[] primitives = {1, 2, 3};
IntStream primitiveStream = Arrays.stream(primitives); // list-backed iterable
PrimitiveIterator.OfInt fromPrimitive1 = primitiveStream.iterator();
```

(java 8) google .

```
Iterable<Integer> fromPrimitive2 = Ints.asList(primitives);
```

---

2 .

:

```
int[][] array = new int[10][10];
```

```
for (int indexOuter = 0; indexOuter < array.length; indexOuter++) {
    for (int indexInner = 0; indexInner < array[indexOuter].length; indexInner++) {
        array[indexOuter][indexInner] = indexOuter + indexInner;
    }
}
```

Java SE 5

```
for (int[] numbers : array) {
    for (int value : numbers) {
        System.out.println(value);
    }
}
```

.

`while` `do-while` .

: `0` `array.length - 1` ( ) . . .

:

```
int[] numbers = {1, 2, 3, 4};

public void incrementNumbers() {
    // DO THIS :
    for (int i = 0; i < numbers.length; i++) {
        numbers[i] += 1; //or this: numbers[i] = numbers[i] + 1; or numbers[i]++;
    }

    // DON'T DO THIS :
    for (int i = 0; i < 4; i++) {
        numbers[i] += 1;
    }
}
```

.

:

```
public void fillArrayWithDoubleIndex(int[] array) {
    // DO THIS :
    for (int i = 0; i < array.length; i++) {
        array[i] = i * 2;
    }

    // DON'T DO THIS :
    int doubleLength = array.length * 2;
    for (int i = 0; i < doubleLength; i += 2) {
        array[i / 2] = i;
    }
}
```

```
int[] array = {0, 1, 1, 2, 3, 5, 8, 13};
for (int i = array.length - 1; i >= 0; i--) {
   System.out.println(array[i]);
}
```

. .

```
// we want to print out all of these
String name = "Margaret";
int eyeCount = 16;
double height = 50.2;
int legs = 9;
int arms = 5;


// copy-paste approach:
System.out.println(name);
System.out.println(eyeCount);
System.out.println(height);
System.out.println(legs);
System.out.println(arms);


// temporary array approach:
for(Object attribute : new Object[]{name, eyeCount, height, legs, arms})
    System.out.println(attribute);

// using only numbers
for(double number : new double[]{eyeCount, legs, arms, height})
    System.out.println(Math.sqrt(number));
```

.

Java .

for

```
int[] a = { 4, 1, 3, 2 };
int[] b = new int[a.length];
for (int i = 0; i < a.length; i++) {
    b[i] = a[i];
}
```

Object .

Object.clone ()

Java `Object` `Object.clone()` .

```
int[] a = { 4, 1, 3, 2 };
int[] b = a.clone(); // [4, 1, 3, 2]
```

Object.clone ., .

Arrays.copyOf ()

java.util.Arrays . .

```
int[] a = {4, 1, 3, 2};
int[] b = Arrays.copyOf(a, a.length); // [4, 1, 3, 2]
```

Arrays.copyOf .

```
Double[] doubles = { 1.0, 2.0, 3.0 };
Number[] numbers = Arrays.copyOf(doubles, doubles.length, Number[].class);
```

System.arraycopy ()

public static void arraycopy(Object src, int srcPos, Object dest, int destPos, int length) .

```
int[] a = { 4, 1, 3, 2 };
int[] b = new int[a.length];
System.arraycopy(a, 0, b, 0, a.length); // [4, 1, 3, 2]
```

Arrays.copyOfRange ()

Array , Array Array .

```
int[] a = { 4, 1, 3, 2 };
int[] b = Arrays.copyOfRange(a, 0, a.length); // [4, 1, 3, 2]
```

. A[] T[] A[] A T . :

```
public static <T, A> T[] castArray(T[] target, A[] array) {
    for (int i = 0; i < array.length; i++) {
        target[i] = (T) array[i];
    }
    return target;
}
```

, A[] :

```
T[] target = new T[array.Length];
target = castArray(target, array);
```

Java SE, `Arrays.copyOf(original, newLength, newType)` .

```
Double[] doubles = { 1.0, 2.0, 3.0 };
Number[] numbers = Arrays.copyOf(doubles, doubles.length, Number[].class);
```

Java `java.util.Arrays` . .

ArrayList

`java.util.List` , , .

```
String[] array = new String[]{"foo", "bar", "baz"};

List<String> list = new ArrayList<>(Arrays.asList(array));
list.remove("foo");

// Creates a new array with the same size as the list and copies the list
// elements to it.
array = list.toArray(new String[list.size()]);

System.out.println(Arrays.toString(array)); //[bar, baz]
```

System.arraycopy

`System.arraycopy()` . :

```
int[] array = new int[] { 1, 2, 3, 4 }; // Original array.
int[] result = new int[array.length - 1]; // Array which will contain the result.
int index = 1; // Remove the value "2".

// Copy the elements at the left of the index.
System.arraycopy(array, 0, result, 0, index);
// Copy the elements at the right of the index.
System.arraycopy(array, index + 1, result, index, array.length - index - 1);

System.out.println(Arrays.toString(result)); //[1, 3, 4]
```

Apache Commons Lang

Apache Commons Lang , `ArrayUtils` `removeElement()` . :

```
int[] array = new int[]{1,2,3,4};
array = ArrayUtils.removeElement(array, 2); //remove first occurrence of 2
System.out.println(Arrays.toString(array)); //[1, 3, 4]
```

(covariant) `Integer` `Number` `Integer[]` `Number[]` . .

```
Integer[] integerArray = {1, 2, 3};
```

```
Number[] numberArray = integerArray;  // valid
Number firstElement = numberArray[0]; // valid
numberArray[0] = 4L;                  // throws ArrayStoreException at runtime
```

`Integer[]` `Number[]`, `Integer`, `Long` .

generic `List` .

```
List<Integer> integerList = Arrays.asList(1, 2, 3);
//List<Number> numberList = integerList;  // compile error
List<? extends Number> numberList = integerList;
Number firstElement = numberList.get(0);
//numberList.set(0, 4L);                  // compile error
```

.

```
interface I {}

class A implements I {}
class B implements I {}
class C implements I {}

I[] array10 = new I[] { new A(), new B(), new C() }; // Create an array with new
                                                     // operator and array initializer.

I[] array11 = { new A(), new B(), new C() };         // Shortcut syntax with array
                                                     // initializer.

I[] array12 = new I[3];                              // { null, null, null }

I[] array13 = new A[] { new A(), new A() };          // Works because A implements I.

Object[] array14 = new Object[] { "Hello, World!", 3.14159, 42 }; // Create an array with
                                                     // new operator and array initializer.

Object[] array15 = { new A(), 64, "My String" };     // Shortcut syntax
                                                     // with array initializer.
```

?

. . "" .

```
String[] listOfCities = new String[3];   // array created with size 3.
listOfCities[0] = "New York";
listOfCities[1] = "London";
listOfCities[2] = "Berlin";
```

`listOfCities` . .

1. 4 ,
2. 3 0, 1 2
3. 3 .

. Java 6 .

```
String[] newArray = new String[listOfCities.length + 1];
System.arraycopy(listOfCities, 0, newArray, 0, listOfCities.length);
newArray[listOfCities.length] = "Sydney";
```

Java 6 `Arrays.copyOf` `Arrays.copyOfRange` `Arrays.copyOfRange` .

```
String[] newArray = Arrays.copyOf(listOfCities, listOfCities.length + 1);
newArray[listOfCities.length] = "Sydney";
```

. .

-

.

- . . .
- "" .

. . .

Java SE . , Java SE 「」 , properties `List` , `Set` `Map` API . `ArrayList` (: O (N) , O (1) , O (N)) .

(`ArrayList` . . `O(1)` . .)

. .

```
String[] strings = new String[] { "A", "B", "C" };
int[] ints = new int[] { 1, 2, 3, 4 };
```

`index` `index2` . `-1` .

`Arrays.binarySearch` ( )

```
int index = Arrays.binarySearch(strings, "A");
int index2 = Arrays.binarySearch(ints, 1);
```

`Arrays.asList` ((primitive))

```
int index = Arrays.asList(strings).indexOf("A");
int index2 = Arrays.asList(ints).indexOf(1);  // compilation error
```

`Stream`

Java SE 8

```
int index = IntStream.range(0, strings.length)
```

```
                        .filter(i -> "A".equals(strings[i]))
                        .findFirst()
                        .orElse(-1); // If not present, gives us -1.
  // Similar for an array of primitives
```

```
  int index = -1;
  for (int i = 0; i < array.length; i++) {
      if ("A".equals(array[i])) {
          index = i;
          break;
      }
  }
  // Similar for an array of primitives
```

org.apache.commons

```
  int index = org.apache.commons.lang3.ArrayUtils.contains(strings, "A");
  int index2 = org.apache.commons.lang3.ArrayUtils.contains(ints, 1);
```

: .

0 .

.

```
boolean isPresent = Arrays.asList(strings).contains("A");
```

Java SE 8

```
boolean isPresent = Stream<String>.of(strings).anyMatch(x -> "A".equals(x));
```

```
boolean isPresent = false;
for (String s : strings) {
    if ("A".equals(s)) {
        isPresent = true;
        break;
    }
}

boolean isPresent = org.apache.commons.lang3.ArrayUtils.contains(ints, 4);
```

Arrays API .

```
import java.util.Arrays;

// creating an array with integers
int[] array = {7, 4, 2, 1, 19};
// this is the sorting part just one function ready to be used
Arrays.sort(array);
// prints [1, 2, 4, 7, 19]
System.out.println(Arrays.toString(array));
```

:

`String` `.sort()` String array Comparable `sort()`).

```
String[] names = {"John", "Steve", "Shane", "Adam", "Ben"};
System.out.println("String array before sorting : " + Arrays.toString(names));
Arrays.sort(names);
System.out.println("String array after sorting in ascending order : " +
Arrays.toString(names));
```

:

```
String array before sorting : [John, Steve, Shane, Adam, Ben]
String array after sorting in ascending order : [Adam, Ben, John, Shane, Steve]
```

```
Arrays.sort(names, 0, names.length, Collections.reverseOrder());
System.out.println("String array after sorting in descending order : " +
Arrays.toString(names));
```

:

```
String array after sorting in descending order : [Steve, Shane, John, Ben, Adam]
```

`Comparable` `Comparator` .

`sort(Object[])` `Comparable` .

`e1.compareTo(e2)` e1 e2 *`ClassCastException`* . `sort(T[], Comparator)` Object .

```
// How to Sort Object Array in Java using Comparator and Comparable
Course[] courses = new Course[4];
courses[0] = new Course(101, "Java", 200);
courses[1] = new Course(201, "Ruby", 300);
courses[2] = new Course(301, "Python", 400);
courses[3] = new Course(401, "Scala", 500);

System.out.println("Object array before sorting : " + Arrays.toString(courses));

Arrays.sort(courses);
System.out.println("Object array after sorting in natural order : " +
Arrays.toString(courses));

Arrays.sort(courses, new Course.PriceComparator());
System.out.println("Object array after sorting by price : " + Arrays.toString(courses));

Arrays.sort(courses, new Course.NameComparator());
System.out.println("Object array after sorting by name : " + Arrays.toString(courses));
```

:

```
Object array before sorting : [#101 Java@200 , #201 Ruby@300 , #301 Python@400 , #401
Scala@500 ]
Object array after sorting in natural order : [#101 Java@200 , #201 Ruby@300 , #301 Python@400
```

```
, #401 Scala@500 ]
Object array after sorting by price : [#101 Java@200 , #201 Ruby@300 , #301 Python@400 , #401
Scala@500 ]
Object array after sorting by name : [#101 Java@200 , #301 Python@400 , #201 Ruby@300 , #401
Scala@500 ]
```

.

(Java 8).

Java SE 8

```
int[] primitiveArray = {1, 2, 3, 4};
Integer[] boxedArray =
    Arrays.stream(primitiveArray).boxed().toArray(Integer[]::new);
```

.

Java SE 8

```
int[] primitiveArray = {1, 2, 3, 4};
Integer[] boxedArray = new Integer[primitiveArray.length];
for (int i = 0; i < primitiveArray.length; ++i) {
    boxedArray[i] = primitiveArray[i]; // Each element is autoboxed here
}
```

, .

Java SE 8

```
Integer[] boxedArray = {1, 2, 3, 4};
int[] primitiveArray =
    Arrays.stream(boxedArray).mapToInt(Integer::intValue).toArray();
```

Java SE 8

```
Integer[] boxedArray = {1, 2, 3, 4};
int[] primitiveArray = new int[boxedArray.length];
for (int i = 0; i < boxedArray.length; ++i) {
    primitiveArray[i] = boxedArray[i]; // Each element is outboxed here
}
```

: https://riptutorial.com/ko/java/topic/99/

106:

`System.currentTimeMillis()` . .

Examples

JMH

JMH . `HashSet` `TreeSet` .

JHM Maven Shade Plugin . JHM `pom.xml` .

```
<build>
    <plugins>
        <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-shade-plugin</artifactId>
            <version>3.0.0</version>
            <executions>
                <execution>
                    <phase>package</phase>
                    <goals>
                        <goal>shade</goal>
                    </goals>
                    <configuration>
                        <finalName>/benchmarks</finalName>
                        <transformers>
                            <transformer

implementation="org.apache.maven.plugins.shade.resource.ManifestResourceTransformer">
                                <mainClass>org.openjdk.jmh.Main</mainClass>
                            </transformer>
                        </transformers>
                        <filters>
                            <filter>
                                <artifact>*:*</artifact>
                                <excludes>
                                    <exclude>META-INF/*.SF</exclude>
                                    <exclude>META-INF/*.DSA</exclude>
                                    <exclude>META-INF/*.RSA</exclude>
                                </excludes>
                            </filter>
                        </filters>
                    </configuration>
                </execution>
            </executions>
        </plugin>
    </plugins>
</build>

<dependencies>
    <dependency>
        <groupId>org.openjdk.jmh</groupId>
        <artifactId>jmh-core</artifactId>
        <version>1.18</version>
```

```
    </dependency>
    <dependency>
        <groupId>org.openjdk.jmh</groupId>
        <artifactId>jmh-generator-annprocess</artifactId>
        <version>1.18</version>
    </dependency>
</dependencies>
```

.

```
package benchmark;

import org.openjdk.jmh.annotations.*;
import org.openjdk.jmh.infra.Blackhole;

import java.util.HashSet;
import java.util.Random;
import java.util.Set;
import java.util.TreeSet;
import java.util.concurrent.TimeUnit;

@State(Scope.Thread)
public class CollectionFinderBenchmarkTest {
    private static final int SET_SIZE = 10000;

    private Set<String> hashSet;
    private Set<String> treeSet;

    private String stringToFind = "8888";

    @Setup
    public void setupCollections() {
        hashSet = new HashSet<>(SET_SIZE);
        treeSet = new TreeSet<>();

        for (int i = 0; i < SET_SIZE; i++) {
            final String value = String.valueOf(i);
            hashSet.add(value);
            treeSet.add(value);
        }

        stringToFind = String.valueOf(new Random().nextInt(SET_SIZE));
    }

    @Benchmark
    @BenchmarkMode(Mode.AverageTime)
    @OutputTimeUnit(TimeUnit.NANOSECONDS)
    public void testHashSet(Blackhole blackhole) {
        blackhole.consume(hashSet.contains(stringToFind));
    }

    @Benchmark
    @BenchmarkMode(Mode.AverageTime)
    @OutputTimeUnit(TimeUnit.NANOSECONDS)
    public void testTreeSet(Blackhole blackhole) {
        blackhole.consume(treeSet.contains(stringToFind));
    }
}
```

`blackhole.consume()`

. . .

```
package benchmark;

import org.openjdk.jmh.runner.Runner;
import org.openjdk.jmh.runner.RunnerException;
import org.openjdk.jmh.runner.options.Options;
import org.openjdk.jmh.runner.options.OptionsBuilder;

public class BenchmarkMain {
    public static void main(String[] args) throws RunnerException {
        final Options options = new OptionsBuilder()
                .include(CollectionFinderBenchmarkTest.class.getSimpleName())
                .forks(1)
                .build();

        new Runner(options).run();
    }
}
```

. `mvn package` (`/target benchmarks.jar`) :

```
java -cp target/benchmarks.jar benchmark.BenchmarkMain
```

.

```
# Run complete. Total time: 00:01:21

Benchmark                                    Mode  Cnt    Score    Error  Units
CollectionFinderBenchmarkTest.testHashSet    avgt   20    9.940 ±  0.270  ns/op
CollectionFinderBenchmarkTest.testTreeSet    avgt   20   98.858 ± 13.743  ns/op
```

`blackhole.consume()` `blackhole.consume()` . java . `Blackhole` .

Aleksey Shipilëv , Jacob Jenkov Java (1 , 2) .

: https://riptutorial.com/ko/java/topic/9514/-

107:

( ) .

`final` . , thread (Java Concurrency in Practice, 3.4. 1).

.

Examples

.

Java . `java.util.Data` . .

. .

```
public class ImmutableIntArray {
    private final int[] array;

    public ImmutableIntArray(int[] array) {
        this.array = array.clone();
    }

    public int[] getValue() {
        return this.clone();
    }
}
```

(`int[]`) .

- `clone()` . `ImmutableIntArray` `ImmutableIntArray` .
- `getValue()` `clone()` . `ImmutableIntArray` .

wrapped `ImmutableIntArray` . , , .

, . .

. , . Java java.lang.String .

.

```
public final class Person {
    private final String name;
    private final String ssn;     // (SSN == social security number)

    public Person(String name, String ssn) {
        this.name = name;
        this.ssn = ssn;
    }
```

```
    public String getName() {
        return name;
    }

    public String getSSN() {
        return ssn;
    }
}
```

`private public static` factory .

:

- .
- .
- setter throw.
- `private final` .
- :
  -
  - getter .
- `final` .

:

- nullable . `null String` .
- `final` . .

.

```
public final class Person { // example of a bad immutability
    private final String name;
    private final String surname;
    public Person(String name) {
        this.name = name;
      }
    public String getName() { return name;}
    public String getSurname() { return surname;}
    public void setSurname(String surname) { this.surname = surname); }
}
```

`Person` .

```
Person person = new Person("Joe");
person.setSurname("Average"); // NOT OK, change surname field after creation
```

`setSurname()` .

```
public Person(String name, String surname) {
    this.name = name;
    this.surname = surname;
  }
```

.

```
public final class Person {
    public String name;
    public Person(String name) {
        this.name = name;
     }
    public String getName() {
        return name;
    }

}
```

.

```
Person person = new Person("Average Joe");
person.name = "Magic Mike"; // not OK, new name for person after creation
```

name `private final` .

---

getter

.

```
import java.util.List;
import java.util.ArrayList;
public final class Names {
    private final List<String> names;
    public Names(List<String> names) {
        this.names = new ArrayList<String>(names);
    }
    public List<String> getNames() {
        return names;
    }
    public int size() {
        return names.size();
    }
}
```

`Names` .

```
List<String> namesList = new ArrayList<String>();
namesList.add("Average Joe");
Names names = new Names(namesList);
System.out.println(names.size()); // 1, only containing "Average Joe"
namesList = names.getNames();
namesList.add("Magic Mike");
System.out.println(names.size()); // 2, NOT OK, now names also contains "Magic Mike"
```

`getNames()` `Names` .

, :

```
public List<String> getNames() {
   return new ArrayList<String>(this.names); // copies elements
}
```

getter :

```
public String getName(int index) {
    return names.get(index);
}
public int size() {
    return names.size();
}
```

. .

```
import java.util.List;
public final class NewNames {
    private final List<String> names;
    public Names(List<String> names) {
        this.names = names;
    }
    public String getName(int index) {
        return names.get(index);
    }
    public int size() {
        return names.size();
    }
}
```

Names , NewNames NewNames .

```
List<String> namesList = new ArrayList<String>();
namesList.add("Average Joe");
NewNames names = new NewNames(namesList);
System.out.println(names.size()); // 1, only containing "Average Joe"
namesList.add("Magic Mike");
System.out.println(names.size()); // 2, NOT OK, now names also contains "Magic Mike"
```

., .

```
    public Names(List<String> names) {
        this.names = new ArrayList<String>(names);
    }
```

.

```
public class Person {
    private final String name;
    public Person(String name) {
        this.name = name;
      }
    public String getName() { return name;}
}
```

`Person` , `Person` :

```
public class MutablePerson extends Person {
    private String newName;
    public MutablePerson(String name) {
        super(name);
    }
    @Override
    public String getName() {
        return newName;
    }
    public void setName(String name) {
        newName = name;
    }
}
```

`Person` (im) mutability .

```
Person person = new MutablePerson("Average Joe");
System.out.println(person.getName()); prints Average Joe
person.setName("Magic Mike"); // NOT OK, person has now a new name!
System.out.println(person.getName()); // prints Magic Mike
```

, `final` () `private` .

: https://riptutorial.com/ko/java/topic/2807/--

108:

. , String .

Java :

1. java.lang.String
2. : java.lang.Integer, java.lang.Byte, java.lang.Character, java.lang.Short, java.lang.Boolean, java.lang.Long, java.lang.Double, java.lang.Float
3. .
4. java.math.BigInteger java.math.BigDecimal ( )
5. java.io.File. VM ( ) . , . File .

Examples

.

1. "setter" .
2. .
3. (override) . final . private .
4. .
5. .
6. . . . , .

ref

```
public final class Color {
    final private int red;
    final private int green;
    final private int blue;

    private void check(int red, int green, int blue) {
        if (red < 0 || red > 255 || green < 0 || green > 255 || blue < 0 || blue > 255) {
            throw new IllegalArgumentException();
        }
    }

    public Color(int red, int green, int blue) {
        check(red, green, blue);
        this.red = red;
        this.green = green;
        this.blue = blue;
    }

    public Color invert() {
        return new Color(255 - red, 255 - green, 255 - blue);
    }
}
```

ref

Point .

```
class Point {
    private int x, y;

    public Point(int x, int y) {
        this.x = x;
        this.y = y;
    }

    public int getX() {
        return x;
    }

    public void setX(int x) {
        this.x = x;
    }

    public int getY() {
        return y;
    }

    public void setY(int y) {
        this.y = y;
    }
}

//...

public final class ImmutableCircle {
    private final Point center;
    private final double radius;

    public ImmutableCircle(Point center, double radius) {
        // we create new object here because it shouldn't be changed
        this.center = new Point(center.getX(), center.getY());
        this.radius = radius;
    }
```

?

. . ..

, , .

: https://riptutorial.com/ko/java/topic/10561/--

109: ()

"varargs" . ASCII (. . .) .

. varargs . ( ).

varargs :

1. Varargs .
2. Varargs .

. .

Examples

varargs

```
void doSomething(String... strings) {
    for (String s : strings) {
        System.out.println(s);
    }
}
```

. Varargs .

Varargs

varargs . .

`printVarArgArray()` .

```
public class VarArgs {

    // this method will print the entire contents of the parameter passed in

    void printVarArgArray(int... x) {
        for (int i = 0; i < x.length; i++) {
            System.out.print(x[i] + ",");
        }
    }

    public static void main(String args[]) {
        VarArgs obj = new VarArgs();

        //Using an array:
        int[] testArray = new int[]{10, 20};
        obj.printVarArgArray(testArray);

        System.out.println(" ");

        //Using a sequence of arguments
        obj.printVarArgArray(5, 6, 5, 8, 6, 31);
```

```
    }
}
```

:

```
10,20,
5,6,5,8,6,31
```

.

```
void method(String... a, int... b , int c){} //Compile time error (multiple varargs )

void method(int... a, String b){} //Compile time error (varargs must be the last argument
```

() : https://riptutorial.com/ko/java/topic/1948/---

110:

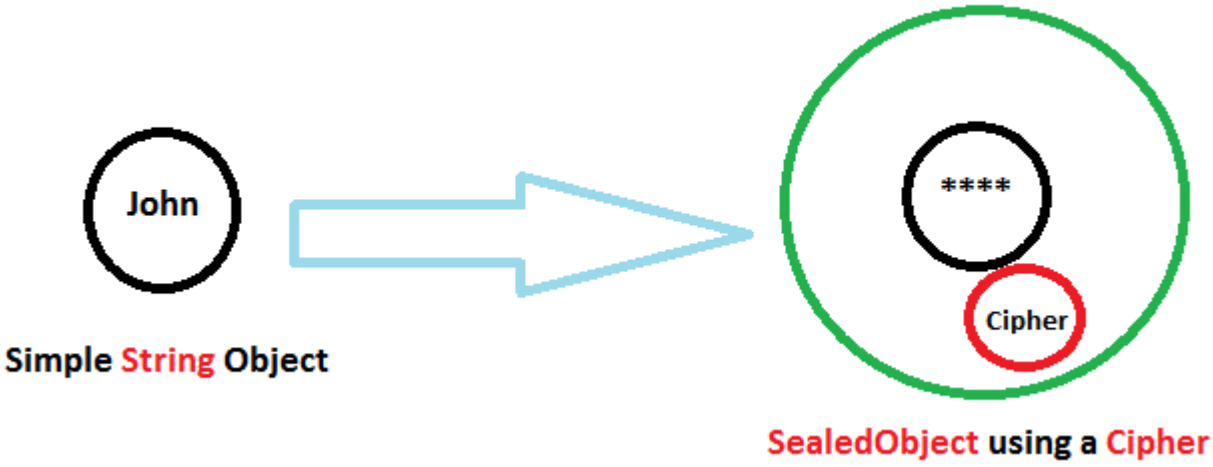
- SealedObject sealedObject = SealedObject (obj, cipher);
- SignedObject signedObject = SignedObject (obj, signingKey, signingEngine);

Examples

SealedObject (javax.crypto.SealedObject)

, , ().

Serializable AES, DES (, " (deep) ") () **SealedObject** . .

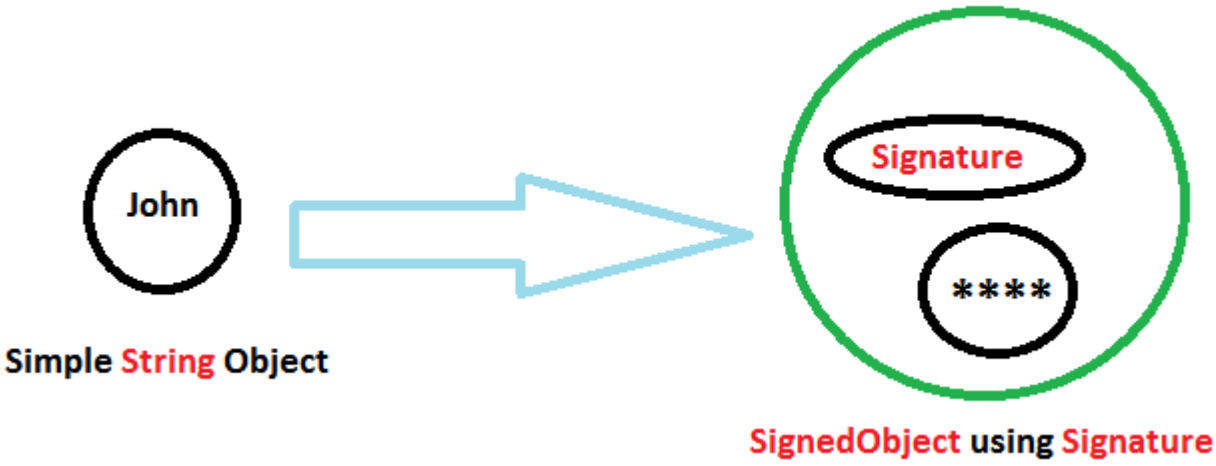



```
Serializable obj = new String("John");
// Generate key
KeyGenerator kgen = KeyGenerator.getInstance("AES");
kgen.init(128);
SecretKey aesKey = kgen.generateKey();
Cipher cipher = Cipher.getInstance("AES");
cipher.init(Cipher.ENCRYPT_MODE, aesKey);
SealedObject sealedObject = new SealedObject(obj, cipher);
System.out.println("sealedObject-" + sealedObject);
System.out.println("sealedObject Data-" + sealedObject.getObject(aesKey));
```

SignedObject (java.security.SignedObject)

SignedObject .

, SignedObject, , , .




```
//Create a key
KeyPairGenerator keyGen = KeyPairGenerator.getInstance("DSA", "SUN");
SecureRandom random = SecureRandom.getInstance("SHA1PRNG", "SUN");
keyGen.initialize(1024, random);
// create a private key
PrivateKey signingKey = keyGen.generateKeyPair().getPrivate();
// create a Signature
Signature signingEngine = Signature.getInstance("DSA");
signingEngine.initSign(signingKey);
// create a simple object
Serializable obj = new String("John");
// sign our object
SignedObject signedObject = new SignedObject(obj, signingKey, signingEngine);

System.out.println("signedObject-" + signedObject);
System.out.println("signedObject Data-" + signedObject.getObject());
```

: https://riptutorial.com/ko/java/topic/5528/-

111:

Examples

SecurityManager

JVM (Java Virtual Machine) SecurityManager . SecurityManager JVM .

JVM java.security.manager SecurityManager .

```
java -Djava.security.manager <main class name>
```

Java :

```
System.setSecurityManager(new SecurityManager())
```

Java SecurityManager . `$JAVA_HOME/lib/security/java.policy` .

ClassLoader

ClassLoader, `ProtectionDomain` .

```
public class PluginClassLoader extends ClassLoader {
    private final ClassProvider provider;

    private final ProtectionDomain pd;


    public PluginClassLoader(ClassProvider provider) {
        this.provider = provider;
        Permissions permissions = new Permissions();

        this.pd = new ProtectionDomain(provider.getCodeSource(), permissions, this, null);
    }


    @Override
    protected Class<?> findClass(String name) throws ClassNotFoundException {
        byte[] classDef = provider.getClass(name);
        Class<?> clazz = defineClass(name, classDef, 0, classDef.length, pd);
        return clazz;
    }
}
```

`loadClass` `findClass` PluginClassLoader .

:

```
public class PluginSecurityPolicy extends Policy {
    private final Permissions appPermissions = new Permissions();
    private final Permissions pluginPermissions = new Permissions();

    public PluginSecurityPolicy() {
```

```
        // amend this as appropriate
        appPermissions.add(new AllPermission());
        // add any permissions plugins should have to pluginPermissions
    }

    @Override
    public Provider getProvider() {
        return super.getProvider();
    }

    @Override
    public String getType() {
        return super.getType();
    }

    @Override
    public Parameters getParameters() {
        return super.getParameters();
    }

    @Override
    public PermissionCollection getPermissions(CodeSource codesource) {
        return new Permissions();
    }

    @Override
    public PermissionCollection getPermissions(ProtectionDomain domain) {
        return isPlugin(domain)?pluginPermissions:appPermissions;
    }

    private boolean isPlugin(ProtectionDomain pd){
        return pd.getClassLoader() instanceof PluginClassLoader;
    }

}
```

SecurityManager ( ).

```
    Policy.setPolicy(new PluginSecurityPolicy());
    System.setSecurityManager(new SecurityManager());
```

`ProtectionDomain` , , `Permission` . . 「」 `Policy` (repository) () 「」 , 「」 .

:

- , ., .
- `Policy` (`com.sun.security.provider.PolicyFile`) .
- . "" .

DeniedPermission

```
package com.example;

import java.lang.reflect.Constructor;
import java.lang.reflect.InvocationTargetException;
import java.lang.reflect.Modifier;
```

```
import java.security.BasicPermission;
import java.security.Permission;
import java.security.UnresolvedPermission;
import java.text.MessageFormat;

/**
 * A representation of a "negative" privilege.
 * <p>
 * A <code>DeniedPermission</code>, when "granted" (to some <code>ProtectionDomain</code>
and/or
 * <code>Principal</code>), represents a privilege which <em>cannot</em> be exercised,
regardless of
 * any positive permissions (<code>AllPermission</code> included) possessed. In other words,
if a
 * set of granted permissions, <em>P</em>, contains a permission of this class, <em>D</em>,
then the
 * set of effectively granted permissions is<br/>
 * <br/>
 *     <em>{ P<sub>implied</sub> - D<sub>implied</sub> }</em>.
 * </p>
 * <p>
 * Each instance of this class encapsulates a <em>target permission</em>, representing the
 * "positive" permission being denied.
 * </p>
 * Denied permissions employ the following naming scheme:<br/>
 * <br/>
 *
    <em>&lt;target_class_name&gt;:&lt;target_name&gt;(:&lt;target_actions&gt;)</em>

 * <br/>
 * where:
 * <ul>
 * <li><em>target_class_name</em> is the name of the target permission's class,</li>
 * <li><em>target_name</em> is the name of the target permission, and</li>
 * <li><em>target_actions</em> is, optionally, the actions string of the target
permission.</li>
 * </ul>
 * A denied permission, having a target permission <em>t</em>, is said to <em>imply</em>
another
 * permission <em>p</em>, if:
 * <ul>
 * <li>p <em>is not</em> itself a denied permission, and <code>(t.implies(p) == true)</code>,
 * or</li>
 * <li>p <em>is</em> a denied permission, with a target <em>t1</em>, and
 * <code>(t.implies(t1) == true)</code>.
 * </ul>
 * <p>
 * It is the responsibility of the policy decision point (e.g., the <code>Policy</code>
provider) to
 * take denied permission semantics into account when issuing authorization statements.
 * </p>
 */
public final class DeniedPermission extends BasicPermission {

    private final Permission target;
    private static final long serialVersionUID = 4736251638698006 79L;

    /**
     * Instantiates a <code>DeniedPermission</code> that encapsulates a target permission of
the
     * indicated class, specified name and, optionally, actions.
```

```
     *
     * @throws IllegalArgumentException
     *             if:
     *             <ul>
     *             <li><code>targetClassName</code> is <code>null</code>, the empty string,
does not
     *             refer to a concrete <code>Permission</code> descendant, or refers to
     *             <code>DeniedPermission.class</code> or
<code>UnresolvedPermission.class</code>.</li>
     *             <li><code>targetName</code> is <code>null</code>.</li>
     *             <li><code>targetClassName</code> cannot be instantiated, and it's the
caller's fault;
     *             e.g., because <code>targetName</code> and/or <code>targetActions</code> do
not adhere
     *             to the naming constraints of the target class; or due to the target class
not
     *             exposing a <code>(String name)</code>, or <code>(String name, String
actions)</code>
     *             constructor, depending on whether <code>targetActions</code> is
<code>null</code> or
     *             not.</li>
     *             </ul>
     */
    public static DeniedPermission newDeniedPermission(String targetClassName, String
targetName,
            String targetActions) {
        if (targetClassName == null || targetClassName.trim().isEmpty() || targetName == null)
{
            throw new IllegalArgumentException(
                    "Null or empty [targetClassName], or null [targetName] argument was
supplied.");
        }
        StringBuilder sb = new StringBuilder(targetClassName).append(":").append(targetName);
        if (targetName != null) {
            sb.append(":").append(targetName);
        }
        return new DeniedPermission(sb.toString());
    }

    /**
     * Instantiates a <code>DeniedPermission</code> that encapsulates a target permission of
the class,
     * name and, optionally, actions, collectively provided as the <code>name</code> argument.
     *
     * @throws IllegalArgumentException
     *             if:
     *             <ul>
     *             <li><code>name</code>'s target permission class name component is empty,
does not
     *             refer to a concrete <code>Permission</code> descendant, or refers to
     *             <code>DeniedPermission.class</code> or
<code>UnresolvedPermission.class</code>.</li>
     *             <li><code>name</code>'s target name component is <code>empty</code></li>
     *             <li>the target permission class cannot be instantiated, and it's the
caller's fault;
     *             e.g., because <code>name</code>'s target name and/or target actions
component(s) do
     *             not adhere to the naming constraints of the target class; or due to the
target class
     *             not exposing a <code>(String name)</code>, or
     *             <code>(String name, String actions)</code> constructor, depending on
```

```
whether the
     *             target actions component is empty or not.</li>
     *             </ul>
     */
    public DeniedPermission(String name) {
        super(name);
        String[] comps = name.split(":");
        if (comps.length < 2) {
            throw new IllegalArgumentException(MessageFormat.format("Malformed name [{0}]
argument.", name));
        }
        this.target = initTarget(comps[0], comps[1], ((comps.length < 3) ? null : comps[2]));
    }

    /**
     * Instantiates a <code>DeniedPermission</code> that encapsulates the given target
permission.
     *
     * @throws IllegalArgumentException
     *             if <code>target</code> is <code>null</code>, a
<code>DeniedPermission</code>, or an
     *             <code>UnresolvedPermission</code>.
     */
    public static DeniedPermission newDeniedPermission(Permission target) {
        if (target == null) {
            throw new IllegalArgumentException("Null [target] argument.");
        }
        if (target instanceof DeniedPermission || target instanceof UnresolvedPermission) {
            throw new IllegalArgumentException("[target] must not be a DeniedPermission or an
UnresolvedPermission.");
        }
        StringBuilder sb = new
StringBuilder(target.getClass().getName()).append(":").append(target.getName());
        String targetActions = target.getActions();
        if (targetActions != null) {
            sb.append(":").append(targetActions);
        }
        return new DeniedPermission(sb.toString(), target);
    }

    private DeniedPermission(String name, Permission target) {
        super(name);
        this.target = target;
    }

    private Permission initTarget(String targetClassName, String targetName, String
targetActions) {
        Class<?> targetClass;
        try {
            targetClass = Class.forName(targetClassName);
        }
        catch (ClassNotFoundException cnfe) {
            if (targetClassName.trim().isEmpty()) {
                targetClassName = "<empty>";
            }
            throw new IllegalArgumentException(
                    MessageFormat.format("Target Permission class [{0}] not found.",
targetClassName));
        }
        if (!Permission.class.isAssignableFrom(targetClass) ||
Modifier.isAbstract(targetClass.getModifiers())) {
```

```
            throw new IllegalArgumentException(MessageFormat
                    .format("Target Permission class [{0}] is not a (concrete) Permission.",
targetClassName));
        }
        if (targetClass == DeniedPermission.class || targetClass ==
UnresolvedPermission.class) {
            throw new IllegalArgumentException("Target Permission class cannot be a
DeniedPermission itself.");
        }
        Constructor<?> targetCtor;
        try {
            if (targetActions == null) {
                targetCtor = targetClass.getConstructor(String.class);
            }
            else {
                targetCtor = targetClass.getConstructor(String.class, String.class);
            }
        }
        catch (NoSuchMethodException nsme) {
            throw new IllegalArgumentException(MessageFormat.format(
                    "Target Permission class [{0}] does not provide or expose a (String name)
or (String name, String actions) constructor.",
                    targetClassName));
        }
        try {
            return (Permission) targetCtor
                    .newInstance(((targetCtor.getParameterCount() == 1) ? new Object[] {
targetName }
                            : new Object[] { targetName, targetActions }));
        }
        catch (ReflectiveOperationException roe) {
            if (roe instanceof InvocationTargetException) {
                if (targetName == null) {
                    targetName = "<null>";
                }
                else if (targetName.trim().isEmpty()) {
                    targetName = "<empty>";
                }
                if (targetActions == null) {
                    targetActions = "<null>";
                }
                else if (targetActions.trim().isEmpty()) {
                    targetActions = "<empty>";
                }
                throw new IllegalArgumentException(MessageFormat.format(
                        "Could not instantiate target Permission class [{0}]; provided target
name [{1}] and/or target actions [{2}] potentially erroneous.",
                        targetClassName, targetName, targetActions), roe);
            }
            throw new RuntimeException(
                    "Could not instantiate target Permission class [{0}]; an unforeseen error
occurred - see attached cause for details",
                    roe);
        }
    }

    /**
     * Checks whether the given permission is implied by this one, as per the {@link
DeniedPermission
     * overview}.
     */
```

```
    @Override
    public boolean implies(Permission p) {
        if (p instanceof DeniedPermission) {
            return target.implies(((DeniedPermission) p).target);
        }
        return target.implies(p);
    }

    /**
     * Returns this denied permission's target permission (the actual positive permission
which is not
     * to be granted).
     */
    public Permission getTargetPermission() {
        return target;
    }

}
```

DenyingPolicy

```
package com.example;

import java.security.CodeSource;
import java.security.NoSuchAlgorithmException;
import java.security.Permission;
import java.security.PermissionCollection;
import java.security.Policy;
import java.security.ProtectionDomain;
import java.security.UnresolvedPermission;
import java.util.Enumeration;

/**
 * Wrapper that adds rudimentary {@link DeniedPermission} processing capabilities to the
standard
 * file-backed <code>Policy</code>.
 */
public final class DenyingPolicy extends Policy {

    {
        try {
            defaultPolicy = Policy.getInstance("javaPolicy", null);
        }
        catch (NoSuchAlgorithmException nsae) {
            throw new RuntimeException("Could not acquire default Policy.", nsae);
        }
    }

    private final Policy defaultPolicy;

    @Override
    public PermissionCollection getPermissions(CodeSource codesource) {
        return defaultPolicy.getPermissions(codesource);
    }

    @Override
    public PermissionCollection getPermissions(ProtectionDomain domain) {
        return defaultPolicy.getPermissions(domain);
    }
```

```
    /**
     * @return
     *         <ul>
     *         <li><code>true</code> if:</li>
     *         <ul>
     *         <li><code>permission</code> <em>is not</em> an instance of
     *         <code>DeniedPermission</code>,</li>
     *         <li>an <code>implies(domain, permission)</code> invocation on the system-
default
     *         <code>Policy</code> yields <code>true</code>, and</li>
     *         <li><code>permission</code> <em>is not</em> implied by any
<code>DeniedPermission</code>s
     *         having potentially been assigned to <code>domain</code>.</li>
     *         </ul>
     *         <li><code>false</code>, otherwise.
     *         </ul>
     */
    @Override
    public boolean implies(ProtectionDomain domain, Permission permission) {
        if (permission instanceof DeniedPermission) {
            /*
             * At the policy decision level, DeniedPermissions can only themselves imply, not
be implied (as
             * they take away, rather than grant, privileges). Furthermore, clients aren't
supposed to use this
             * method for checking whether some domain _does not_ have a permission (which is
what
             * DeniedPermissions express after all).
             */
            return false;
        }

        if (!defaultPolicy.implies(domain, permission)) {
            // permission not granted, so no need to check whether denied
            return false;
        }

        /*
         * Permission granted--now check whether there's an overriding DeniedPermission. The
following
         * assumes that previousPolicy is a sun.security.provider.PolicyFile (different
implementations
         * might not support #getPermissions(ProtectionDomain) and/or handle
UnresolvedPermissions
         * differently).
         */

        Enumeration<Permission> perms = defaultPolicy.getPermissions(domain).elements();
        while (perms.hasMoreElements()) {
            Permission p = perms.nextElement();
            /*
             * DeniedPermissions will generally remain unresolved, as no code is expected to
check whether other
             * code has been "granted" such a permission.
             */
            if (p instanceof UnresolvedPermission) {
                UnresolvedPermission up = (UnresolvedPermission) p;
                if (up.getUnresolvedType().equals(DeniedPermission.class.getName())) {
                    // force resolution
                    defaultPolicy.implies(domain, up);
```

```
                    // evaluate right away, to avoid reiterating over the collection
                    p = new DeniedPermission(up.getUnresolvedName());
                }
            }
            if (p instanceof DeniedPermission && p.implies(permission)) {
                // permission denied
                return false;
            }
        }
        // permission granted
        return true;
    }

    @Override
    public void refresh() {
        defaultPolicy.refresh();
    }

}
```

```
package com.example;

import java.security.Policy;

public class Main {

    public static void main(String... args) {
        Policy.setPolicy(new DenyingPolicy());
        System.setSecurityManager(new SecurityManager());
        // should fail
        System.getProperty("foo.bar");
    }

}
```

:

```
grant codeBase "file:///path/to/classes/bin/-"
    permission java.util.PropertyPermission "*", "read,write";
    permission com.example.DeniedPermission "java.util.PropertyPermission:foo.bar:read";
};
```

`Main` `grant` (`PropertyPermission`) "" (`DeniedPermission`) . "" "foo.bar" `setProperty("foo.baz", "xyz")` .

: https://riptutorial.com/ko/java/topic/5712/-

112:

Examples

.

```
final MessageDigest md5 = MessageDigest.getInstance("MD5");
final MessageDigest sha1 = MessageDigest.getInstance("SHA-1");
final MessageDigest sha256 = MessageDigest.getInstance("SHA-256");

final byte[] data = "FOO BAR".getBytes();

System.out.println("MD5    hash: " + DatatypeConverter.printHexBinary(md5.digest(data)));
System.out.println("SHA1   hash: " + DatatypeConverter.printHexBinary(sha1.digest(data)));
System.out.println("SHA256 hash: " + DatatypeConverter.printHexBinary(sha256.digest(data)));
```

.

```
MD5    hash: E99E768582F6DD5A3BA2D9C849DF736E
SHA1   hash: 0135FAA6323685BA8A8FF8D3F955F0C36949D8FB
SHA256 hash: 8D35C97BCD902B96D1B551741BBE8A7F50BB5A690B4D0225482EAA63DBFB9DED
```

.

.

```
final byte[] sample = new byte[16];

new SecureRandom().nextBytes(sample);

System.out.println("Sample: " + DatatypeConverter.printHexBinary(sample));
```

.

```
Sample: E4F14CEA2384F70B706B53A6DF8C5EFE
```

`nextBytes()` .

```
final byte[] sample = new byte[16];
final SecureRandom randomness = SecureRandom.getInstance("SHA1PRNG", "SUN");

randomness.nextBytes(sample);

System.out.println("Provider: " + randomness.getProvider());
System.out.println("Algorithm: " + randomness.getAlgorithm());
System.out.println("Sample: " + DatatypeConverter.printHexBinary(sample));
```

.

```
Provider: SUN version 1.8
```

```
Algorithm: SHA1PRNG
Sample: C80C44BAEB352FD29FBBE20489E4C0B9
```

/

.

```
final KeyPairGenerator dhGenerator = KeyPairGenerator.getInstance("DiffieHellman");
final KeyPairGenerator dsaGenerator = KeyPairGenerator.getInstance("DSA");
final KeyPairGenerator rsaGenerator = KeyPairGenerator.getInstance("RSA");

dhGenerator.initialize(1024);
dsaGenerator.initialize(1024);
rsaGenerator.initialize(2048);

final KeyPair dhPair = dhGenerator.generateKeyPair();
final KeyPair dsaPair = dsaGenerator.generateKeyPair();
final KeyPair rsaPair = rsaGenerator.generateKeyPair();
```

.

.

```
final KeyPairGenerator generator = KeyPairGenerator.getInstance("RSA");

generator.initialize(2048, SecureRandom.getInstance("SHA1PRNG", "SUN"));

final KeyPair pair = generator.generateKeyPair();
```

.

```
final PrivateKey privateKey = keyPair.getPrivate();
final byte[] data = "FOO BAR".getBytes();
final Signature signer = Signature.getInstance("SHA1withRSA");

signer.initSign(privateKey);
signer.update(data);

final byte[] signature = signer.sign();
```

.

.

```
final PublicKey publicKey = keyPair.getPublic();
final Signature verifier = Signature.getInstance("SHA1withRSA");

verifier.initVerify(publicKey);
verifier.update(data);

System.out.println("Signature: " + verifier.verify(signature));
```

.

```
Signature: true
```

/

.

```
final Cipher rsa = Cipher.getInstance("RSA");

rsa.init(Cipher.ENCRYPT_MODE, keyPair.getPublic());
rsa.update(message.getBytes());
final byte[] result = rsa.doFinal();

System.out.println("Message: " + message);
System.out.println("Encrypted: " + DatatypeConverter.printHexBinary(result));
```

.

```
Message: Hello
Encrypted: 5641FBB9558ECFA9ED...
```

`Cipher` . JCA . RSA `message.getBytes()` . SO .

.

```
final Cipher rsa = Cipher.getInstance("RSA");

rsa.init(Cipher.DECRYPT_MODE, keyPair.getPrivate());
rsa.update(cipherText);
final String result = new String(rsa.doFinal());

System.out.println("Decrypted: " + result);
```

.

```
Decrypted: Hello
```

: https://riptutorial.com/ko/java/topic/7529/--

113:

Java . Java Java

Java JVM . JCE .

Java Java . , .

, .

1. JCE /
2.

Examples

JCE

JCE (Java Cryptography Extension) JVM . JCE , .

JCE . .

`byte` . Java "" . . . , `byte` . .

Java

.

.

.

, `java.utils.Random` "" . "" `Random` .

, `Random` . `SecureRandom` .

.

. , .

: . , X = Y f (x) f f (y) .

(Uniqueness) : . , X Y X! = Y, f (x)! = f (y).

: "" . , f (X) , X f (brute-force) . f1 (f (X)) = X f1 .

. , MD5 SHA1 . , . SHA-256 SHA-512.

: https://riptutorial.com/ko/java/topic/9371/--

114:

- javap [options] <classes>

`<classes>`	`.package1.package2.Classname` `package1/package2/Classname` `.` `.class` `.`
`-help` , `--help` , `-?`	
`-version`	
`-v` , `-verbose`	
`-l`	
`-public`	
`-protected`	/
`-package`	/ / ()
`-p` , `-private`	
`-c`	
`-s`	
`-sysinfo`	(, , , MD5)
`-constants`	
`-classpath <path>`	
`-cp <path>`	
`-bootclasspath <path>`	

Examples

javap

Java `javap` .

Java .

```
package com.stackoverflow.documentation;

import org.springframework.stereotype.Service;
```

```
import java.io.IOException;
import java.io.InputStream;
import java.util.List;

@Service
public class HelloWorldService {

    public void sayHello() {
        System.out.println("Hello, World!");
    }

    private Object[] pvtMethod(List<String> strings) {
        return new Object[]{strings};
    }

    protected String tryCatchResources(String filename) throws IOException {
        try (InputStream inputStream = getClass().getResourceAsStream(filename)) {
            byte[] bytes = new byte[8192];
            int read = inputStream.read(bytes);
            return new String(bytes, 0, read);
        } catch (IOException | RuntimeException e) {
            e.printStackTrace();
            throw e;
        }
    }

    void stuff() {
        System.out.println("stuff");
    }
}
```

.

```
cd <directory containing classes> (e.g. target/classes)
javap com/stackoverflow/documentation/SpringExample
```

.

```
Compiled from "HelloWorldService.java"
public class com.stackoverflow.documentation.HelloWorldService {
  public com.stackoverflow.documentation.HelloWorldService();
  public void sayHello();
  protected java.lang.String tryCatchResources(java.lang.String) throws java.io.IOException;
  void stuff();
}
```

. .

```
javap -p -c -s -constants -l -v com/stackoverflow/documentation/HelloWorldService
```

:

```
Classfile /Users/pivotal/IdeaProjects/stackoverflow-spring-
docs/target/classes/com/stackoverflow/documentation/HelloWorldService.class
  Last modified Jul 22, 2016; size 2167 bytes
  MD5 checksum 6e33b5c292ead21701906353b7f06330
```

```
  Compiled from "HelloWorldService.java"
public class com.stackoverflow.documentation.HelloWorldService
  minor version: 0
  major version: 51
  flags: ACC_PUBLIC, ACC_SUPER
Constant pool:
    #1 = Methodref          #5.#60         // java/lang/Object."<init>":()V
    #2 = Fieldref           #61.#62        // java/lang/System.out:Ljava/io/PrintStream;
    #3 = String             #63            // Hello, World!
    #4 = Methodref          #64.#65        // java/io/PrintStream.println:(Ljava/lang/String;)V
    #5 = Class              #66            // java/lang/Object
    #6 = Methodref          #5.#67         // java/lang/Object.getClass:()Ljava/lang/Class;
    #7 = Methodref          #68.#69        //
java/lang/Class.getResourceAsStream:(Ljava/lang/String;)Ljava/io/InputStream;
    #8 = Methodref          #70.#71        // java/io/InputStream.read:([B)I
    #9 = Class              #72            // java/lang/String
   #10 = Methodref          #9.#73         // java/lang/String."<init>":([BII)V
   #11 = Methodref          #70.#74        // java/io/InputStream.close:()V
   #12 = Class              #75            // java/lang/Throwable
   #13 = Methodref          #12.#76        //
java/lang/Throwable.addSuppressed:(Ljava/lang/Throwable;)V
   #14 = Class              #77            // java/io/IOException
   #15 = Class              #78            // java/lang/RuntimeException
   #16 = Methodref          #79.#80        // java/lang/Exception.printStackTrace:()V
   #17 = String             #55            // stuff
   #18 = Class              #81            // com/stackoverflow/documentation/HelloWorldService
   #19 = Utf8               <init>
   #20 = Utf8               ()V
   #21 = Utf8               Code
   #22 = Utf8               LineNumberTable
   #23 = Utf8               LocalVariableTable
   #24 = Utf8               this
   #25 = Utf8               Lcom/stackoverflow/documentation/HelloWorldService;
   #26 = Utf8               sayHello
   #27 = Utf8               pvtMethod
   #28 = Utf8               (Ljava/util/List;)[Ljava/lang/Object;
   #29 = Utf8               strings
   #30 = Utf8               Ljava/util/List;
   #31 = Utf8               LocalVariableTypeTable
   #32 = Utf8               Ljava/util/List<Ljava/lang/String;>;
   #33 = Utf8               Signature
   #34 = Utf8               (Ljava/util/List<Ljava/lang/String;>;)[Ljava/lang/Object;
   #35 = Utf8               tryCatchResources
   #36 = Utf8               (Ljava/lang/String;)Ljava/lang/String;
   #37 = Utf8               bytes
   #38 = Utf8               [B
   #39 = Utf8               read
   #40 = Utf8               I
   #41 = Utf8               inputStream
   #42 = Utf8               Ljava/io/InputStream;
   #43 = Utf8               e
   #44 = Utf8               Ljava/lang/Exception;
   #45 = Utf8               filename
   #46 = Utf8               Ljava/lang/String;
   #47 = Utf8               StackMapTable
   #48 = Class              #81            // com/stackoverflow/documentation/HelloWorldService
   #49 = Class              #72            // java/lang/String
   #50 = Class              #82            // java/io/InputStream
   #51 = Class              #75            // java/lang/Throwable
   #52 = Class              #38            // "[B"
   #53 = Class              #83            // java/lang/Exception
```

```
   #54 = Utf8               Exceptions
   #55 = Utf8               stuff
   #56 = Utf8               SourceFile
   #57 = Utf8               HelloWorldService.java
   #58 = Utf8               RuntimeVisibleAnnotations
   #59 = Utf8               Lorg/springframework/stereotype/Service;
   #60 = NameAndType        #19:#20       // "<init>":()V
   #61 = Class              #84           // java/lang/System
   #62 = NameAndType        #85:#86       // out:Ljava/io/PrintStream;
   #63 = Utf8               Hello, World!
   #64 = Class              #87           // java/io/PrintStream
   #65 = NameAndType        #88:#89       // println:(Ljava/lang/String;)V
   #66 = Utf8               java/lang/Object
   #67 = NameAndType        #90:#91       // getClass:()Ljava/lang/Class;
   #68 = Class              #92           // java/lang/Class
   #69 = NameAndType        #93:#94       //
getResourceAsStream:(Ljava/lang/String;)Ljava/io/InputStream;
   #70 = Class              #82           // java/io/InputStream
   #71 = NameAndType        #39:#95       // read:([B)I
   #72 = Utf8               java/lang/String
   #73 = NameAndType        #19:#96       // "<init>":([BII)V
   #74 = NameAndType        #97:#20       // close:()V
   #75 = Utf8               java/lang/Throwable
   #76 = NameAndType        #98:#99       // addSuppressed:(Ljava/lang/Throwable;)V
   #77 = Utf8               java/io/IOException
   #78 = Utf8               java/lang/RuntimeException
   #79 = Class              #83           // java/lang/Exception
   #80 = NameAndType        #100:#20      // printStackTrace:()V
   #81 = Utf8               com/stackoverflow/documentation/HelloWorldService
   #82 = Utf8               java/io/InputStream
   #83 = Utf8               java/lang/Exception
   #84 = Utf8               java/lang/System
   #85 = Utf8               out
   #86 = Utf8               Ljava/io/PrintStream;
   #87 = Utf8               java/io/PrintStream
   #88 = Utf8               println
   #89 = Utf8               (Ljava/lang/String;)V
   #90 = Utf8               getClass
   #91 = Utf8               ()Ljava/lang/Class;
   #92 = Utf8               java/lang/Class
   #93 = Utf8               getResourceAsStream
   #94 = Utf8               (Ljava/lang/String;)Ljava/io/InputStream;
   #95 = Utf8               ([B)I
   #96 = Utf8               ([BII)V
   #97 = Utf8               close
   #98 = Utf8               addSuppressed
   #99 = Utf8               (Ljava/lang/Throwable;)V
  #100 = Utf8               printStackTrace
{
  public com.stackoverflow.documentation.HelloWorldService();
    descriptor: ()V
    flags: ACC_PUBLIC
    Code:
      stack=1, locals=1, args_size=1
         0: aload_0
         1: invokespecial #1                  // Method java/lang/Object."<init>":()V
         4: return
      LineNumberTable:
        line 10: 0
      LocalVariableTable:
        Start  Length  Slot  Name   Signature
```

```
            0       5     0  this   Lcom/stackoverflow/documentation/HelloWorldService;

  public void sayHello();
    descriptor: ()V
    flags: ACC_PUBLIC
    Code:
      stack=2, locals=1, args_size=1
         0: getstatic     #2                  // Field
java/lang/System.out:Ljava/io/PrintStream;
         3: ldc           #3                  // String Hello, World!
         5: invokevirtual #4                  // Method
java/io/PrintStream.println:(Ljava/lang/String;)V
         8: return
      LineNumberTable:
        line 13: 0
        line 14: 8
      LocalVariableTable:
        Start  Length  Slot  Name   Signature
            0       9     0  this   Lcom/stackoverflow/documentation/HelloWorldService;

  private java.lang.Object[] pvtMethod(java.util.List<java.lang.String>);
    descriptor: (Ljava/util/List;)[Ljava/lang/Object;
    flags: ACC_PRIVATE
    Code:
      stack=4, locals=2, args_size=2
         0: iconst_1
         1: anewarray     #5                  // class java/lang/Object
         4: dup
         5: iconst_0
         6: aload_1
         7: aastore
         8: areturn
      LineNumberTable:
        line 17: 0
      LocalVariableTable:
        Start  Length  Slot  Name   Signature
            0       9     0  this   Lcom/stackoverflow/documentation/HelloWorldService;
            0       9     1 strings   Ljava/util/List;
      LocalVariableTypeTable:
        Start  Length  Slot  Name   Signature
            0       9     1 strings   Ljava/util/List<Ljava/lang/String;>;
    Signature: #34                          //
(Ljava/util/List<Ljava/lang/String;>;)[Ljava/lang/Object;

  protected java.lang.String tryCatchResources(java.lang.String) throws java.io.IOException;
    descriptor: (Ljava/lang/String;)Ljava/lang/String;
    flags: ACC_PROTECTED
    Code:
      stack=5, locals=10, args_size=2
         0: aload_0
         1: invokevirtual #6                  // Method
java/lang/Object.getClass:()Ljava/lang/Class;
         4: aload_1
         5: invokevirtual #7                  // Method
java/lang/Class.getResourceAsStream:(Ljava/lang/String;)Ljava/io/InputStream;
         8: astore_2
         9: aconst_null
        10: astore_3
        11: sipush        8192
        14: newarray       byte
        16: astore        4
```

```
        18: aload_2
        19: aload         4
        21: invokevirtual #8                  // Method java/io/InputStream.read:([B)I
        24: istore        5
        26: new           #9                  // class java/lang/String
        29: dup
        30: aload         4
        32: iconst_0
        33: iload         5
        35: invokespecial #10                 // Method java/lang/String."<init>":([BII)V
        38: astore        6
        40: aload_2
        41: ifnull        70
        44: aload_3
        45: ifnull        66
        48: aload_2
        49: invokevirtual #11                 // Method java/io/InputStream.close:()V
        52: goto          70
        55: astore        7
        57: aload_3
        58: aload         7
        60: invokevirtual #13                 // Method
java/lang/Throwable.addSuppressed:(Ljava/lang/Throwable;)V
        63: goto          70
        66: aload_2
        67: invokevirtual #11                 // Method java/io/InputStream.close:()V
        70: aload         6
        72: areturn
        73: astore        4
        75: aload         4
        77: astore_3
        78: aload         4
        80: athrow
        81: astore        8
        83: aload_2
        84: ifnull        113
        87: aload_3
        88: ifnull        109
        91: aload_2
        92: invokevirtual #11                 // Method java/io/InputStream.close:()V
        95: goto          113
        98: astore        9
       100: aload_3
       101: aload         9
       103: invokevirtual #13                 // Method
java/lang/Throwable.addSuppressed:(Ljava/lang/Throwable;)V
       106: goto          113
       109: aload_2
       110: invokevirtual #11                 // Method java/io/InputStream.close:()V
       113: aload         8
       115: athrow
       116: astore_2
       117: aload_2
       118: invokevirtual #16                 // Method
java/lang/Exception.printStackTrace:()V
       121: aload_2
       122: athrow
      Exception table:
         from    to  target type
            48    52    55   Class java/lang/Throwable
            11    40    73   Class java/lang/Throwable
```

```
            11    40    81   any
            91    95    98   Class java/lang/Throwable
            73    83    81   any
             0    70   116   Class java/io/IOException
             0    70   116   Class java/lang/RuntimeException
            73   116   116   Class java/io/IOException
            73   116   116   Class java/lang/RuntimeException
      LineNumberTable:
        line 21: 0
        line 22: 11
        line 23: 18
        line 24: 26
        line 25: 40
        line 21: 73
        line 25: 81
        line 26: 117
        line 27: 121
      LocalVariableTable:
        Start  Length  Slot  Name   Signature
           18      55     4 bytes   [B
           26      47     5  read   I
            9     107     2 inputStream   Ljava/io/InputStream;
          117       6     2     e   Ljava/lang/Exception;
            0     123     0  this   Lcom/stackoverflow/documentation/HelloWorldService;
            0     123     1 filename   Ljava/lang/String;
      StackMapTable: number_of_entries = 9
        frame_type = 255 /* full_frame */
          offset_delta = 55
          locals = [ class com/stackoverflow/documentation/HelloWorldService, class
java/lang/String, class java/io/InputStream, class java/lang/Throwable, class "[B", int, class
java/lang/String ]
          stack = [ class java/lang/Throwable ]
        frame_type = 10 /* same */
        frame_type = 3 /* same */
        frame_type = 255 /* full_frame */
          offset_delta = 2
          locals = [ class com/stackoverflow/documentation/HelloWorldService, class
java/lang/String, class java/io/InputStream, class java/lang/Throwable ]
          stack = [ class java/lang/Throwable ]
        frame_type = 71 /* same_locals_1_stack_item */
          stack = [ class java/lang/Throwable ]
        frame_type = 255 /* full_frame */
          offset_delta = 16
          locals = [ class com/stackoverflow/documentation/HelloWorldService, class
java/lang/String, class java/io/InputStream, class java/lang/Throwable, top, top, top, top,
class java/lang/Throwable ]
          stack = [ class java/lang/Throwable ]
        frame_type = 10 /* same */
        frame_type = 3 /* same */
        frame_type = 255 /* full_frame */
          offset_delta = 2
          locals = [ class com/stackoverflow/documentation/HelloWorldService, class
java/lang/String ]
          stack = [ class java/lang/Exception ]
    Exceptions:
      throws java.io.IOException

  void stuff();
    descriptor: ()V
    flags:
    Code:
```

```
      stack=2, locals=1, args_size=1
         0: getstatic     #2                  // Field
java/lang/System.out:Ljava/io/PrintStream;
         3: ldc           #17                 // String stuff
         5: invokevirtual #4                  // Method
java/io/PrintStream.println:(Ljava/lang/String;)V
         8: return
      LineNumberTable:
        line 32: 0
        line 33: 8
      LocalVariableTable:
        Start  Length  Slot  Name   Signature
            0       9     0  this   Lcom/stackoverflow/documentation/HelloWorldService;
}
SourceFile: "HelloWorldService.java"
RuntimeVisibleAnnotations:
  0: #59()
```

: https://riptutorial.com/ko/java/topic/2318/---

115:

.

Examples

Java `final` , . .

- .
- .
- .

. .

`final` . .

```
public int sumup(final List<Integer> ints);
```

`sumup ints` .

`final` .

```
  public IPrintName printName(){
    String name;
    return new IPrintName(){
        @Override
        public void printName(){
            System.out.println(name);
        }
    };
}
```

`name` .

Java SE 8

. . .

`final static`

`final foo static static` :

```
class TestFinal {
    private final static List foo;

    public Test() {
        foo = new ArrayList();
    }
}
```

, . . foo TestFinal . TestFinal TestFinal foo . foo .

```
class TestFinal {
    private static final List foo = new ArrayList();
    //..
}
```

:

```
class TestFinal {
    private static final List foo;
    static {
        foo = new ArrayList();
    }
    //..
}
```

final . .

Java Integer , Long . Integer Integer . . final .

volatile . volatile . , volatile (volatile long double) / .

```
public class MyRunnable implements Runnable
{
    private volatile boolean active;

    public void run(){ // run is called in one thread
        active = true;
        while (active){
            // some code here
        }
    }

    public void stop(){ // stop() is called from another thread
        active = false;
    }
}
```

static , .

- . .
- , .
- . .

```
public class TestStatic
{
    static int staticVariable;

    static {
        // This block of code is run when the class first loads
        staticVariable = 11;
    }

    int nonStaticVariable = 5;
```

```
    static void doSomething() {
        // We can access static variables from static methods
        staticVariable = 10;
    }

    void add() {
        // We can access both static and non-static variables from non-static methods
        nonStaticVariable += staticVariable;
    }

    static class StaticInnerClass {
        int number;
        public StaticInnerClass(int _number) {
            number = _number;
        }

        void doSomething() {
            // We can access number and staticVariable, but not nonStaticVariable
            number += staticVariable;
        }

        int getNumber() {
            return number;
        }
    }
}


// Static fields and methods
TestStatic object1 = new TestStatic();

System.out.println(object1.staticVariable); // 11
System.out.println(TestStatic.staticVariable); // 11

TestStatic.doSomething();

TestStatic object2 = new TestStatic();

System.out.println(object1.staticVariable); // 10
System.out.println(object2.staticVariable); // 10
System.out.println(TestStatic.staticVariable); // 10

object1.add();

System.out.println(object1.nonStaticVariable); // 15
System.out.println(object2.nonStaticVariable); // 10

// Static inner classes
StaticInnerClass object3 = new TestStatic.StaticInnerClass(100);
StaticInnerClass object4 = new TestStatic.StaticInnerClass(200);

System.out.println(object3.getNumber()); // 100
System.out.println(object4.getNumber()); // 200

object3.doSomething();

System.out.println(object3.getNumber()); // 110
System.out.println(object4.getNumber()); // 200
```

. . .

```
abstract class Car
{
    abstract void tagLine();
}

class Honda extends Car
{
    void tagLine()
    {
        System.out.println("Start Something Special");
    }
}

class Toyota extends Car
{
    void tagLine()
    {
        System.out.println("Drive Your Dreams");
    }
}
```

. . synchronized , . .

```
class Shared
{
    int i;

    synchronized void SharedMethod()
    {
        Thread t = Thread.currentThread();

        for(int i = 0; i <= 1000; i++)
        {
            System.out.println(t.getName()+" : "+i);
        }
    }

    void SharedMethod2()
    {
        synchronized (this)
        {
            System.out.println("Thais access to currect object is synchronize "+this);
        }
    }
}

public class ThreadsInJava
{
    public static void main(String[] args)
    {
        final Shared s1 = new Shared();

        Thread t1 = new Thread("Thread - 1")
        {
            @Override
            public void run()
            {
                s1.SharedMethod();
            }
        };
```

```
        Thread t2 = new Thread("Thread - 2")
        {
            @Override
            public void run()
            {
                s1.SharedMethod();
            }
        };

        t1.start();

        t2.start();
    }
}
```

transient .

```
public transient int limit = 55;   // will not persist
public int b; // will persist
```

Java SE 1.2

strictfp . 32 64 ( ), IEEE 754 . , .

```
// strictfp keyword can be applied on methods, classes and interfaces.

strictfp class A{}

strictfp interface M{}

class A{
    strictfp void m(){}
}
```

: https://riptutorial.com/ko/java/topic/4401/---

116:

- MyClass Comparable `<MyClass >`
- MyComparator Comparator `<SomeOtherClass >`
- int compareTo (MyClass)
- int compare (SomeOtherClass o1, SomeOtherClass o2)

`double compareTo(..)` .

```
public int comareTo(MyClass other) {
    return (int)(doubleField - other.doubleField); //THIS IS BAD
}
```

`(int)` 0 .

Double.compare .

```
public int comareTo(MyClass other) {
    return Double.compare(doubleField,other.doubleField); //THIS IS GOOD
}
```

`Comparable<T>` , `Comparable` , 1.2 . , `Comparable<T>` .

.

```
public class A implements Comparable<A>
```

.

`Comparable<T>` , `Comparator<T>` . `Comparator` . `Comparable` .

Examples

Comparable

Person . `equals` `hashCode` .

```
public class Person {

    private final String lastName; //invariant - nonnull
    private final String firstName; //invariant - nonnull

    public Person(String firstName, String lastName){
        this.firstName = firstName != null ? firstName : "";
        this.lastName = lastName != null ? lastName : "";
    }
```

```
    public String getFirstName() {
        return firstName;
    }

    public String getLastName() {
        return lastName;
    }

    public String toString() {
        return lastName + ", " + firstName;
    }

    @Override
    public boolean equals(Object o) {
        if (! (o instanceof Person)) return false;
        Person p = (Person)o;
        return firstName.equals(p.firstName) && lastName.equals(p.lastName);
    }

    @Override
    public int hashCode() {
        return Objects.hash(firstName, lastName);
    }
}
```

`Person` .

```
public static void main(String[] args) {
    List<Person> people = Arrays.asList(new Person("John", "Doe"),
                                        new Person("Bob", "Dole"),
                                        new Person("Ronald", "McDonald"),
                                        new Person("Alice", "McDonald"),
                                        new Person("Jill", "Doe"));
    Collections.sort(people); //This currently won't work.
}
```

, , . `Collections.sort(..)` .

: `1,3,5,4,2` , `1,2,3,4,5` . (Java) *(natural ordering)*, . Person `compareTo(Person p)`: `Comparable<Person> compareTo(Person p)`:

```
public class Person implements Comparable<Person> {

    private final String lastName; //invariant - nonnull
    private final String firstName; //invariant - nonnull

    public Person(String firstName, String lastName) {
        this.firstName = firstName != null ? firstName : "";
        this.lastName = lastName != null ? lastName : "";
    }

    public String getFirstName() {
        return firstName;
    }

    public String getLastName() {
        return lastName;
```

```
    }

    public String toString() {
        return lastName + ", " + firstName;
    }

    @Override
    public boolean equals(Object o) {
        if (! (o instanceof Person)) return false;
        Person p = (Person)o;
        return firstName.equals(p.firstName) && lastName.equals(p.lastName);
    }

    @Override
    public int hashCode() {
        return Objects.hash(firstName, lastName);
    }

    @Override
    public int compareTo(Person other) {
        // If this' lastName and other's lastName are not comparably equivalent,
        // Compare this to other by comparing their last names.
        // Otherwise, compare this to other by comparing their first names
        int lastNameCompare = lastName.compareTo(other.lastName);
        if (lastNameCompare != 0) {
            return lastNameCompare;
        } else {
            return firstName.compareTo(other.firstName);
        }
    }
}
```

.

```
public static void main(String[] args) {
    List<Person> people = Arrays.asList(new Person("John", "Doe"),
                                        new Person("Bob", "Dole"),
                                        new Person("Ronald", "McDonald"),
                                        new Person("Alice", "McDonald"),
                                        new Person("Jill", "Doe"));
    Collections.sort(people); //Now functions correctly

    //people is now sorted by last name, then first name:
    // --> Jill Doe, John Doe, Bob Dole, Alice McDonald, Ronald McDonald
}
```

Person , Person Comparator<T> . circle, square, rectangle, triangle, hexagon .,

.

```
public class PersonComparator implements Comparator<Person> {

    public int compare(Person p1, Person p2) {
        // If p1's lastName and p2's lastName are not comparably equivalent,
        // Compare p1 to p2 by comparing their last names.
        // Otherwise, compare p1 to p2 by comparing their first names
        if (p1.getLastName().compareTo(p2.getLastName()) != 0) {
            return p1.getLastName().compareTo(p2.getLastName());
        } else {
```

```
            return p1.getFirstName().compareTo(p2.getFirstName());
        }
    }
}

//Assume the first version of Person (that does not implement Comparable) is used here
public static void main(String[] args) {
    List<Person> people = Arrays.asList(new Person("John", "Doe"),
                                        new Person("Bob", "Dole"),
                                        new Person("Ronald", "McDonald"),
                                        new Person("Alice", "McDonald"),
                                        new Person("Jill", "Doe"));
    Collections.sort(people); //Illegal, Person doesn't implement Comparable.
    Collections.sort(people, new PersonComparator()); //Legal

    //people is now sorted by last name, then first name:
    // --> Jill Doe, John Doe, Bob Dole, Alice McDonald, Ronald McDonald
}
```

Comparator / .

```
//Assume the first version of Person (that does not implement Comparable) is used here
public static void main(String[] args) {
    List<Person> people = Arrays.asList(new Person("John", "Doe"),
                                        new Person("Bob", "Dole"),
                                        new Person("Ronald", "McDonald"),
                                        new Person("Alice", "McDonald"),
                                        new Person("Jill", "Doe"));
    Collections.sort(people); //Illegal, Person doesn't implement Comparable.

    Collections.sort(people, new PersonComparator()); //Legal

    //people is now sorted by last name, then first name:
    // --> Jill Doe, John Doe, Bob Dole, Alice McDonald, Ronald McDonald

    //Anonymous Class
    Collections.sort(people, new Comparator<Person>() { //Legal
        public int compare(Person p1, Person p2) {
            //Method code...
        }
    });
}
```

Java SE 8

8, .

```
    //Lambda
    Collections.sort(people, (p1, p2) -> { //Legal
        //Method code....
    });
```

Comparator

Comparator Comparator . lastName firstName .

```
Collections.sort(people, Comparator.comparing(Person::getLastName)
                                .thenComparing(Person::getFirstName));
```

---

reversedMethod .

compareTo compare

Comparable<T> .

```
public interface Comparable<T> {

    public int compareTo(T other);

}
```

Comparator<T> .

```
public interface Comparator<T> {

    public int compare(T t1, T t2);

}
```

: compareTo this other , compare t1 t2 this .

. (compareTo), . , 0 . , a b a :

- a < b , a.compareTo(b) compare(a,b) b.compareTo(a) compare(b,a)
- a > b , a.compareTo(b) compare(a,b) b.compareTo(a) compare(b,a)
- a b , 0 .

( ) ( )

Collections.sort() .

- List<T> T Comparable compareTo() .
- Comparator List Comparator

, Comparable Person .

```
public class Person implements Comparable<Person> {
    private String name;
    private int age;

    public String getName() {
        return name;
    }
    public void setName(String name) {
```

```
        this.name = name;
    }
    public int getAge() {
        return age;
    }
    public void setAge(int age) {
        this.age = age;
    }

    @Override
    public int compareTo(Person o) {
        return this.getAge() - o.getAge();
    }
    @Override
    public String toString() {
        return this.getAge()+"-"+this.getName();
    }

}
```

compareTo() .

```
//-- usage
List<Person> pList = new ArrayList<Person>();
            Person p = new Person();
            p.setName("A");
            p.setAge(10);
            pList.add(p);
            p = new Person();
            p.setName("Z");
            p.setAge(20);
            pList.add(p);
            p = new Person();
            p.setName("D");
            p.setAge(30);
            pList.add(p);

            //-- natural sorting i.e comes with object implementation, by age
            Collections.sort(pList);

            System.out.println(pList);
```

Comparator Comparable List List .

```
            //-- explicit sorting, define sort on another property here goes with name
            Collections.sort(pList, new Comparator<Person>() {

                @Override
                public int compare(Person o1, Person o2) {
                    return o1.getName().compareTo(o2.getName());
                }
            });
            System.out.println(pList);
```

Java 8 map `Map.Entry` .

Java SE 8

```
Map<String, Integer> numberOfEmployees = new HashMap<>();
numberOfEmployees.put("executives", 10);
numberOfEmployees.put("human ressources", 32);
numberOfEmployees.put("accounting", 12);
numberOfEmployees.put("IT", 100);

// Output the smallest departement in terms of number of employees
numberOfEmployees.entrySet().stream()
    .sorted(Map.Entry.comparingByValue())
    .limit(1)
    .forEach(System.out::println);   // outputs : executives=10
```

API .

Java SE 8

```
List<Map.Entry<String, Integer>> entries = new ArrayList<>(numberOfEmployees.entrySet());
Collections.sort(entries, Map.Entry.comparingByValue());
```

Comparator

```
Comparator.comparing(Person::getName)
```

`Person` . long, int double . :

```
Comparator.comparingInt(Person::getAge)
```

, `reversed()` .

```
Comparator.comparing(Person::getName).reversed()
```

```
Comparator.comparing(Person::getLastName).thenComparing(Person::getFirstName)
```

firs . .

: https://riptutorial.com/ko/java/topic/3137/---

117:

- C / C ++ Java . . .
- `byte` -128 +127 . 0xFF `(b & 0xFF)` .

Examples

/

. .

RGB 3 int .

```
// Raw bytes as input
byte[] b = {(byte)0x65, (byte)0xFF, (byte)0x31};

// Packed in big endian: x == 0x65FF31
int x = (b[0] & 0xFF) << 16  // Red
      | (b[1] & 0xFF) <<  8  // Green
      | (b[2] & 0xFF) <<  0; // Blue

// Packed in little endian: y == 0x31FF65
int y = (b[0] & 0xFF) <<  0
      | (b[1] & 0xFF) <<  8
      | (b[2] & 0xFF) << 16;
```

```
// Raw int32 as input
int x = 0x31FF65;

// Unpacked in big endian: {0x65, 0xFF, 0x31}
byte[] c = {
    (byte)(x >> 16),
    (byte)(x >>  8),
    (byte)(x & 0xFF)
};

// Unpacked in little endian: {0x31, 0xFF, 0x65}
byte[] d = {
    (byte)(x & 0xFF),
    (byte)(x >>  8),
    (byte)(x >> 16)
};
```

, , long

`n` `i` (byte, short, char, int long) :

```
(i & 1 << n) != 0 // checks bit 'n'
i |= 1 << n;      // sets bit 'n' to 1
i &= ~(1 << n);   // sets bit 'n' to 0
i ^= 1 << n;      // toggles the value of bit 'n'
```

long / int / short / byte :

```
public class BitMaskExample {
    private static final long FIRST_BIT = 1L << 0;
    private static final long SECOND_BIT = 1L << 1;
    private static final long THIRD_BIT = 1L << 2;
    private static final long FOURTH_BIT = 1L << 3;
    private static final long FIFTH_BIT = 1L << 4;
    private static final long BIT_55 = 1L << 54;

    public static void main(String[] args) {
        checkBitMask(FIRST_BIT | THIRD_BIT | FIFTH_BIT | BIT_55);
    }

    private static void checkBitMask(long bitmask) {
        System.out.println("FIRST_BIT: " + ((bitmask & FIRST_BIT) != 0));
        System.out.println("SECOND_BIT: " + ((bitmask & SECOND_BIT) != 0));
        System.out.println("THIRD_BIT: " + ((bitmask & THIRD_BIT) != 0));
        System.out.println("FOURTh_BIT: " + ((bitmask & FOURTH_BIT) != 0));
        System.out.println("FIFTH_BIT: " + ((bitmask & FIFTH_BIT) != 0));
        System.out.println("BIT_55: " + ((bitmask & BIT_55) != 0));
    }
}
```

```
FIRST_BIT: true
SECOND_BIT: false
THIRD_BIT: true
FOURTh_BIT: false
FIFTH_BIT: true
BIT_55: true
```

`checkBitMask` `checkBitMask` `FIRST_BIT | THIRD_BIT | FIFTH_BIT | BIT_55` .

2

2 (2 ^ n) `n` .

.

```
int pow2 = 1<<n;
```

:

```
int twoExp4 = 1<<4; //2^4
int twoExp5 = 1<<5; //2^5
int twoExp6 = 1<<6; //2^6
...
int twoExp31 = 1<<31; //2^31
```

16 10 2 .

```
int twoExp4 = 0x10; //hexadecimal
int twoExp5 = 0x20; //hexadecimal
int twoExp6 = 64; //decimal
```

```
...
int twoExp31 = -2147483648; //is that a power of 2?
```

2 int

```
int pow2(int exp){
    return 1<<exp;
}
```

2

x 2 , 1 , x-1 . , 4 100 3 011 . 0 2 .

```
boolean isPowerOfTwo(int x)
{
    return (x != 0) && ((x & (x - 1)) == 0);
}
```

READ , **WRITE** **EXECUTE** . 0 7 (4)

RESOURCE = READ WRITE EXECUTE (12)

RESOURCE = 0100 0110 0101 = 4 6 5 (12)

(12) ?

0100 0110 0101

0000 0000 0111 (&)

0000 0000 0101 = 5

RESOURCE **EXECUTE** . ?

0100 0110 0101

0111 0000 0000 (&)

0100 0000 0000 = 1024

? ? 1024. . . . , 8 , READ ? ?

0100 0000 0000 >> 8 => 0000 0000 0100 (0)

4 .

, **READ** , **WRITE** , **EXECUTE** ?

. (4)

READ = 0001

= 0100

EXECUTE = 0110

:

`READ | WRITE | EXECUTE` , . READ ? |

0001 | 0100 | 0110 => 0111

0001 (0001 0110 0110).

READ 8 **WRITE** 4 **PERMISSIONS** . **RESOURCE** 12 ( ). .

(<< 8) | (<< 4) | ()

0000 0000 0001 << 8 (READ)

0001 0000 0000 (8)

0000 0000 0100 << 4 (WRITE)

0000 0100 0000 (4)

0000 0000 0001 ()

.

0001 0000 0000 (READ)

0000 0100 0000 (WRITE)

0000 0000 0001 ()

0001 0100 0001 ()

java.util.BitSet

1.7 java.util.BitSet .

```
final BitSet bitSet = new BitSet(8); // by default all bits are unset

IntStream.range(0, 8).filter(i -> i % 2 == 0).forEach(bitSet::set); // {0, 2, 4, 6}

bitSet.set(3); // {0, 2, 3, 4, 6}

bitSet.set(3, false); // {0, 2, 4, 6}

final boolean b = bitSet.get(3); // b = false

bitSet.flip(6); // {0, 2, 4}

bitSet.set(100); // {0, 2, 4, 100} - expands automatically
```

BitSet Clonable Serializable long[] words .

and , or , xor , andNot :

```
bitSet.and(new BitSet(8));
bitSet.or(new BitSet(8));
bitSet.xor(new BitSet(8));
bitSet.andNot(new BitSet(8));
```

Java . int [-2 ^ 31 - 1, 2 ^ 31] -1, 0 .

>> << . .

. int [0, 2 ^ 32 - 1] int 2 .

. Java >>> , .

```
                initial value:                4 (                                100)
     signed left-shift: 4 << 1                8 (                               1000)
    signed right-shift: 4 >> 1                2 (                                 10)
 unsigned right-shift: 4 >>> 1                2 (                                 10)
                initial value:               -4 (   11111111111111111111111111111100)
    signed left-shift: -4 << 1               -8 (   11111111111111111111111111111000)
   signed right-shift: -4 >> 1               -2 (   11111111111111111111111111111110)
unsigned right-shift: -4 >>> 1      2147483646 (    1111111111111111111111111111110)
```

<<< ?

. . 2 .

.

: https://riptutorial.com/ko/java/topic/1177/-

118: I / O

- Paths.get (String first, String ... more) // String Path .
- Paths.get (URI uri) // URI Path .

Examples

Path ( )

Paths .

```
Path p1 = Paths.get("/var/www");
Path p2 = Paths.get(URI.create("file:///home/testuser/File.txt"));
Path p3 = Paths.get("C:\\Users\\DentAr\\Documents\\HHGTDG.odt");
Path p4 = Paths.get("/home", "arthur", "files", "diary.tex");
```

, Path .

- toString() toString() .

  ```
  Path p1 = Paths.get("/var/www"); // p1.toString() returns "/var/www"
  ```

- getFileName() (getFileName() .

  ```
  Path p1 = Paths.get("/var/www"); // p1.getFileName() returns "www"
  Path p3 = Paths.get("C:\\Users\\DentAr\\Documents\\HHGTDG.odt"); // p3.getFileName()
  returns "HHGTDG.odt"
  ```

- getNameCount() .

  ```
  Path p1 = Paths.get("/var/www"); // p1.getNameCount() returns 2
  ```

- getName(int index) .

  ```
  Path p1 = Paths.get("/var/www"); // p1.getName(0) returns "var", p1.getName(1) returns
  "www"
  ```

- getParent() .

  ```
  Path p1 = Paths.get("/var/www"); // p1.getParent().toString() returns "/var"
  ```

- getRoot() .

  ```
  Path p1 = Paths.get("/var/www"); // p1.getRoot().toString() returns "/"
  Path p3 = Paths.get("C:\\Users\\DentAr\\Documents\\HHGTDG.odt"); //
  p3.getRoot().toString() returns "C:\\"
  ```

`resolve()` . .

```
Path p5 = Paths.get("/home/");
Path p6 = Paths.get("arthur/files");
Path joined = p5.resolve(p6);
Path otherJoined = p5.resolve("ford/files");
```

```
joined.toString() == "/home/arthur/files"
otherJoined.toString() == "/home/ford/files"
```

. ( ) .. ( ).

, ...

Paths API `.normalize()` .

```
Path p7 = Paths.get("/home/./arthur/../ford/files");
Path p8 = Paths.get("C:\\Users\\.\\..\\Program Files");
```

```
p7.normalize().toString() == "/home/ford/files"
p8.normalize().toString() == "C:\\Program Files"
```

`Files` .

.

```
Files.exists(Path path)
```

```
Files.notExists(Path path)
```

`!Files.exists(path)` `Files.notExists(path)` . .

- (`exists true notExists false`)
- `false` (`exists false notExists true`)
- (:) : `exists nonExists` false .

`Files.isDirectory(Path path)` `Files.isRegularFile(Path path)` .

```
Path p1 = Paths.get("/var/www");
Path p2 = Paths.get("/home/testuser/File.txt");
```

```
Files.isDirectory(p1) == true
Files.isRegularFile(p1) == false

Files.isDirectory(p2) == false
```

```
Files.isRegularFile(p2) == true
```

.

```
Files.isReadable(Path path)
Files.isWritable(Path path)
Files.isExecutable(Path path)

Files.isHidden(Path path)
Files.isSymbolicLink(Path path)
```

MIME

```
Files.probeContentType(Path path)
```

MIME . MIME .

- `text/plain` `text/plain`
- HTML `text/html`
- PDF `application/pdf`
- PNG `image/png`

`Files` .

```
Path p2 = Paths.get(URI.create("file:///home/testuser/File.txt"));
byte[] content = Files.readAllBytes(p2);
List<String> linesOfContent = Files.readAllLines(p2);
```

`Files.readAllLines()` charset (`StandardCharsets.UTF_8`).

```
List<String> linesOfContent = Files.readAllLines(p2, StandardCharsets.ISO_8859_1);
```

bite- `Files`

```
Path p2 = Paths.get("/home/testuser/File.txt");
List<String> lines = Arrays.asList(
    new String[]{"First line", "Second line", "Third line"});

Files.write(p2, lines);
```

```
Files.write(Path path, byte[] bytes)
```

.

I / O : https://riptutorial.com/ko/java/topic/5519/--i---o

119:

Java . Java .

Java , . JLS §8.8 .

Examples

"" . .
, .

```
public class TestClass {
    private String test;
}
```

```
public class TestClass {
    private String test;
    public TestClass() {

    }
}
```

. package-private

. :

```
public class TestClass {
    private String test;
    public TestClass(String arg) {
    }
}
```

```
public class TestClass {
    private String test;
    public TestClass() {
    }
    public TestClass(String arg) {
    }
}
```

. .

```
public class TestClass {

    private String testData;

    public TestClass() {
        testData = "Test"
    }
}
```

.

```
TestClass testClass = new TestClass();
```

.

```
public class TestClass {

    private String testData;

    public TestClass(String testData) {
        this.testData = testData;
    }
}
```

:

```
TestClass testClass = new TestClass("Test Data");
```

. this() this() .

```
public class TestClass {

    private String testData;

    public TestClass(String testData) {
        this.testData = testData;
    }

    public TestClass() {
        this("Test"); // testData defaults to "Test"
    }
}
```

:

```
TestClass testClass1 = new TestClass("Test Data");
TestClass testClass2 = new TestClass();
```

Parent Child . Child Child gebinning . Child `super(...)` . Child Parent .

`super(...)` :

(, no-args `super()`)

```
class Parent {
    private String name;
    private int age;

    public Parent() {} // necessary because we call super() without arguments

    public Parent(String tName, int tAge) {
        name = tName;
```

```
        age = tAge;
    }
}

// This does not even compile, because name and age are private,
// making them invisible even to the child class.
class Child extends Parent {
    public Child() {
        // compiler implicitly calls super() here
        name = "John";
        age = 42;
    }
}
```

`super()` :

```
class Parent {
    private String name;
    private int age;
    public Parent(String tName, int tAge) {
        name = tName;
        age = tAge;
    }
}

class Child extends Parent {
    public Child() {
        super("John", 42);   // explicit super-call
    }
}
```

: () .

`super(...)` (.).

`super(...)` .

```
class Parent{
    public Parent(String tName, int tAge) {}
}

class Child extends Parent{
    public Child(){}
}
```

Parent Child `super` . . `super` .

```
class Child extends Parent{
    public Child(){
          super("",0);
    }
}
```

: https://riptutorial.com/ko/java/topic/682/

120:

`Optional` null . `isPresent()` `true` `get()` .

`ifPresent()` `orElse()` .

- Optional.empty () // .
- Optional.of (value) // null Optional . null , NullPointerException Throw.
- Optional.ofNullable (value) // null Optional .

Examples

Optional

`NoSuchElementException` `Optional.get()` . `Optional.orElse(T)` `Optional.orElseGet(Supplier<? extends T>)` Optional .

```
String value = "something";

return Optional.ofNullable(value).orElse("defaultValue");
// returns "something"

return Optional.ofNullable(value).orElseGet(() -> getDefaultValue());
// returns "something" (never calls the getDefaultValue() method)
```

```
String value = null;

return Optional.ofNullable(value).orElse("defaultValue");
// returns "defaultValue"

return Optional.ofNullable(value).orElseGet(() -> getDefaultValue());
// calls getDefaultValue() and returns its results
```

`orElse` `orElseGet` . `orElse` .

`Optional` `map()` `null` `null` .

(`map()` `filter()` Stream .)

:

```
public <U> Optional<U> map(Function<? super T,? extends U> mapper)
```

:

```
String value = null;

return Optional.ofNullable(value).map(String::toUpperCase).orElse("NONE");
// returns "NONE"
```

```
String value = "something";

return Optional.ofNullable(value).map(String::toUpperCase).orElse("NONE");
// returns "SOMETHING"
```

(NULL) Optional.map () (optional) ., map () (null) . **Null** .

.

```
String value = foo.getBar().getBaz().toString();
```

getBar , getBaz toString NullPointerException .

Optional toString() .

```
String value = Optional.ofNullable(foo)
                       .map(Foo::getBar)
                       .map(Bar::getBaz)
                       .map(Baz::toString)
                       .orElse("");
```

null .

. .

```
Optional.ofNullable(foo)
        .map(Foo::getBar)
        .map(Bar::getBaz)
        .map(Baz::toString)
        .ifPresent(System.out::println);
```

throw.

orElseThrow() Optional , throw . get() . .

.

```
Optional optional = Optional.of("something");

return optional.orElseThrow(IllegalArgumentException::new);
// returns "something" string
```

throw.

```
Optional optional = Optional.empty();

return optional.orElseThrow(IllegalArgumentException::new);
// throws IllegalArgumentException
```

throw .

```
optional.orElseThrow(() -> new IllegalArgumentException("Illegal"));
```

`filter()` .

`if (!somePredicate(x)) { x = null; }` .

:

```
String value = null;
Optional.ofNullable(value) // nothing
        .filter(x -> x.equals("cool string"))// this is never run since value is null
        .isPresent(); // false
```

```
String value = "cool string";
Optional.ofNullable(value) // something
        .filter(x -> x.equals("cool string"))// this is run and passes
        .isPresent(); // true
```

```
String value = "hot string";
Optional.ofNullable(value) // something
        .filter(x -> x.equals("cool string"))// this is run and fails
        .isPresent(); // false
```

`OptionalDouble` , `OptionalInt` `OptionalLong` `Optional` , :

```
OptionalInt presentInt = OptionalInt.of(value);
OptionalInt absentInt = OptionalInt.empty();
```

null . .

```
presentInt.isPresent(); // Is true.
absentInt.isPresent(); // Is false.
```

.

```
// Prints the value since it is provided on creation.
presentInt.ifPresent(System.out::println);

// Gives the other value as the original Optional is empty.
int finalValue = absentInt.orElseGet(this::otherValue);

// Will throw a NoSuchElementException.
int nonexistentValue = absentInt.getAsInt();
```

```
Optional<String> optionalWithValue = Optional.of("foo");
optionalWithValue.ifPresent(System.out::println);//Prints "foo".

Optional<String> emptyOptional = Optional.empty();
emptyOptional.ifPresent(System.out::println);//Does nothing.
```

`orElse` `Object` `Supplier` (`orElseGet`) .

:

```
String value = "something";
return Optional.ofNullable(value)
               .orElse(getValueThatIsHardToCalculate()); // returns "something"
```

getValueThatIsHardToCalculate() . . .

.

```
String value = "something";
return Optional.ofNullable(value)
               .orElseGet(() -> getValueThatIsHardToCalculate()); // returns "something"
```

getValueThatIsHardToCalculate() Optional .

flatMap map . javadoc .

map(Function) Optional flatMap , flatMap Optional .

, Optional , Optional.flatMap Optionals .

, .

```
public class Foo {
    Optional<Bar> getBar(){
        return Optional.of(new Bar());
    }
}

public class Bar {
}
```

Optional.map Optional . ie Optional<Optional<Bar>> .

```
Optional<Optional<Bar>> nestedOptionalBar =
    Optional.of(new Foo())
        .map(Foo::getBar);
```

Optional.flatMap Optional . ie Optional<Bar> .

```
Optional<Bar> optionalBar =
    Optional.of(new Foo())
        .flatMap(Foo::getBar);
```

: https://riptutorial.com/ko/java/topic/152/-

121:

Examples

HashSet

HashSet :

```
Set<String> h = new HashSet<String>() {{
    add("a");
    add("b");
}};
```

:

```
Set<String> h = new HashSet<String>(Arrays.asList("a", "b"));
```

:

```
Sets.newHashSet("a", "b", "c")
```

:

```
Set<String> set3 = Stream.of("a", "b", "c").collect(toSet());
```

. equals() hashCode() .

.

HashSet

Java SE 7

```
Set<String> set = new HashSet<> ();
set.add("Banana");
set.add("Banana");
set.add("Apple");
set.add("Strawberry");

// Set Elements: ["Strawberry", "Banana", "Apple"]
```

LinkedHashSet

Java SE 7

```
Set<String> set = new LinkedHashSet<> ();
set.add("Banana");
set.add("Banana");
set.add("Apple");
set.add("Strawberry");

// Set Elements: ["Banana", "Apple", "Strawberry"]
```

TreeSet - compareTo() Comparator

Java SE 7

```
Set<String> set = new TreeSet<> ();
set.add("Banana");
set.add("Banana");
set.add("Apple");
set.add("Strawberry");

// Set Elements: ["Apple", "Banana", "Strawberry"]
```

Java SE 7

```
Set<String> set = new TreeSet<> ((string1, string2) -> string2.compareTo(string1));
set.add("Banana");
set.add("Banana");
set.add("Apple");
set.add("Strawberry");

// Set Elements: ["Strawberry", "Banana", "Apple"]
```

. .

`Set` `HashSet` , `TreeSet` , `LinkedHashSet` .

:

HashSet :

```
Set<T> set = new HashSet<T>();
```

`T` `String` , `Integer` . **HashSet** O (1) .

TreeSet :

O (lg (n)) . .

```
TreeSet<T> sortedSet = new TreeSet<T>();
```

LinkedHashSet :

`HashSet` . . . O (1) `HashSet` .

```
LinkedHashSet<T> linkedhashset = new LinkedHashSet<T>();
```

?

.

:

1. **HashSet :** (HashMap) .
2. **HashSet :**
3. **TreeSet :** TreeMap NavigableSet .

```
Set<Integer> set = new HashSet<Integer>(); // Creates an empty Set of Integers

Set<Integer> linkedHashSet = new LinkedHashSet<Integer>(); //Creates a empty Set of Integers,
with predictable iteration order
```

`add()` .

```
 set.add(12); //  - Adds element 12 to the set
 set.add(13); //  - Adds element 13 to the set
```

:

```
set = [12,13]
```

```
set.clear();  //Removes all objects from the collection.
```

:

```
set = []
```

.

`contains()` .

```
set.contains(0);  //Returns true if a specified object is an element within the set.
```

: `False`

.

`isEmpty()` Set .

```
set.isEmpty();  //Returns true if the set has no elements
```

: True

```
 set.remove(0); // Removes first occurrence of a specified object from the collection
```

.

```
set.size(); //Returns the number of elements in the collection
```

: 0

```
List<String> list = new ArrayList<String>(listOfElements);
```

List.addAll ()

```
    Set<String> set = new HashSet<String>();
    set.add("foo");
    set.add("boo");

    List<String> list = new ArrayList<String>();
    list.addAll(set);
```

Java 8 Steam API

```
List<String> list = set.stream().collect(Collectors.toList());
```

```
elements
```

```
Collection<Type> noDuplicates = new HashSet<Type>(elements);
```

:

```
List<String> names = new ArrayList<>(
        Arrays.asList("John", "Marco", "Jenny", "Emily", "Jenny", "Emily", "John"));
Set<String> noDuplicates = new HashSet<>(names);
System.out.println("noDuplicates = " + noDuplicates);
```

:

```
noDuplicates = [Marco, Emily, John, Jenny]
```

: https://riptutorial.com/ko/java/topic/3102/

122:

.

Examples

```
String hostName = args[0];
int portNumber = Integer.parseInt(args[1]);

try (
    Socket echoSocket = new Socket(hostName, portNumber);
    PrintWriter out =
        new PrintWriter(echoSocket.getOutputStream(), true);
    BufferedReader in =
        new BufferedReader(
            new InputStreamReader(echoSocket.getInputStream()));
    BufferedReader stdIn =
        new BufferedReader(
            new InputStreamReader(System.in))
) {
    //Use the socket
}
```

: https://riptutorial.com/ko/java/topic/9918/

123:

properties key value . java.util.Properties Hashtable .

. Properties . , .

.

- :
- =
- #

Properties String Map . Map getProperty , setProperty stringPropertyNames .

Java . Properties.load . :

- / (=), (:). equals .
- .
- (u).
- . .
- , \u 16 UTF-16 .

java.util.ResourceBundle Java SE InputStream. InputStream ISO 8859-1 (, 0-255) . \u . JDK native2ascii .

, Charset Reader (InputStreamReader) , . (InputStream) load (Reader) .

XML . loadFromXML . XML DTD http://java.sun.com/dtd/properties.dtd .

Examples

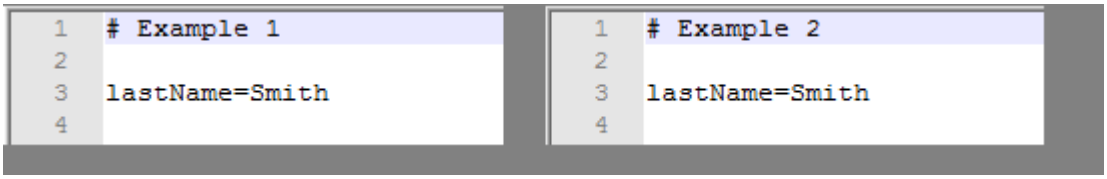
.

```
public class Defaults {

    public static Properties loadDefaults() {
        try (InputStream bundledResource =
            Defaults.class.getResourceAsStream("defaults.properties")) {

            Properties defaults = new Properties();
            defaults.load(bundledResource);
            return defaults;
        } catch (IOException e) {
            // Since the resource is bundled with the application,
            // we should never get here.
            throw new UncheckedIOException(
                "defaults.properties not properly packaged"
                + " with application", e);
        }
    }
```

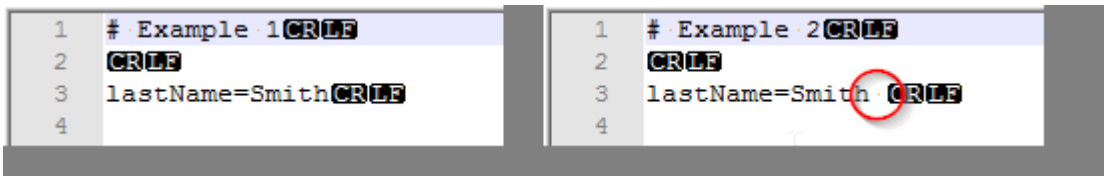
```
}
```

:

.



, .



(Notepad ++.)

lastName "Smith" "Smith " .

, Properties . Properties . **TrimmedProperties** . .

```
import java.io.FileInputStream;
import java.io.FileReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.Reader;
import java.util.Map.Entry;
import java.util.Properties;

/**
 * Properties class where values are trimmed for trailing whitespace if the
 * properties are loaded from a file.
 *
 * <p>
 * In the standard {@link java.util.Properties Properties} class trailing
 * whitespace is always preserved. When loading properties from a file such
 * trailing whitespace is almost always <i>unintentional</i>. This class fixes
 * this problem. The trimming of trailing whitespace only takes place if the
 * source of input is a file and only where the input is line oriented (meaning
 * that for example loading from XML file is <i>not</i> changed by this class).
 * For this reason this class is almost in all cases a safe drop-in replacement
 * for the standard <tt>Properties</tt>
 * class.
 *
 * <p>
 * Whitespace is defined here as any of space (U+0020) or tab (U+0009).
 * * *
 */
public class TrimmedProperties extends Properties {

    /**
     * Reads a property list (key and element pairs) from the input byte stream.
```

```
     *
     * <p>Behaves exactly as {@link java.util.Properties#load(java.io.InputStream) }
     * with the exception that trailing whitespace is trimmed from property values
     * if <tt>inStream</tt> is an instance of <tt>FileInputStream</tt>.
     *
     * @see java.util.Properties#load(java.io.InputStream)
     * @param inStream the input stream.
     * @throws IOException if an error occurred when reading from the input stream.
     */
    @Override
    public void load(InputStream inStream) throws IOException {
        if (inStream instanceof FileInputStream) {
            // First read into temporary props using the standard way
            Properties tempProps = new Properties();
            tempProps.load(inStream);
            // Now trim and put into target
            trimAndLoad(tempProps);
        } else {
            super.load(inStream);
        }
    }

    /**
     * Reads a property list (key and element pairs) from the input character stream in a
simple line-oriented format.
     *
     * <p>Behaves exactly as {@link java.util.Properties#load(java.io.Reader)}
     * with the exception that trailing whitespace is trimmed on property values
     * if <tt>reader</tt> is an instance of <tt>FileReader</tt>.
     *
     * @see java.util.Properties#load(java.io.Reader) }
     * @param reader the input character stream.
     * @throws IOException if an error occurred when reading from the input stream.
     */
    @Override
    public void load(Reader reader) throws IOException {
        if (reader instanceof FileReader) {
            // First read into temporary props using the standard way
            Properties tempProps = new Properties();
            tempProps.load(reader);
            // Now trim and put into target
            trimAndLoad(tempProps);
        } else {
            super.load(reader);
        }
    }

    private void trimAndLoad(Properties p) {
        for (Entry<Object, Object> entry : p.entrySet()) {
            if (entry.getValue() instanceof String) {
                put(entry.getKey(), trimTrailing((String) entry.getValue()));
            } else {
                put(entry.getKey(), entry.getValue());
            }
        }
    }

    /**
     * Trims trailing space or tabs from a string.
     *
     * @param str
```

```
     * @return
     */
    public static String trimTrailing(String str) {
        if (str != null) {
            // read str from tail until char is no longer whitespace
            for (int i = str.length() - 1; i >= 0; i--) {
                if ((str.charAt(i) != ' ') && (str.charAt(i) != '\t')) {
                    return str.substring(0, i + 1);
                }
            }
        }
        return str;
    }
}
```

XML

XML

XML f} `.properties` f} `.store()` `storeToXML()` .

```
public void saveProperties(String location) throws IOException{
    // make new instance of properties
    Properties prop = new Properties();

    // set the property values
    prop.setProperty("name", "Steve");
    prop.setProperty("color", "green");
    prop.setProperty("age", "23");

    // check to see if the file already exists
    File file = new File(location);
    if (!file.exists()){
        file.createNewFile();
    }

    // save the properties
    prop.storeToXML(new FileOutputStream(file), "testing properties with xml");
}
```

.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE properties SYSTEM "http://java.sun.com/dtd/properties.dtd">
<properties>
<comment>testing properties with xml</comment>
<entry key="age">23</entry>
<entry key="color">green</entry>
<entry key="name">Steve</entry>
</properties>
```

XML

properties load() .propeties load() loadFromXML() .

```
public static void loadProperties(String location) throws FileNotFoundException, IOException{
    // make new properties instance to load the file into
    Properties prop = new Properties();

    // check to make sure the file exists
    File file = new File(location);
    if (file.exists()){
        // load the file
        prop.loadFromXML(new FileInputStream(file));

        // print out all the properties
        for (String name : prop.stringPropertyNames()){
            System.out.println(name + "=" + prop.getProperty(name));
        }
    } else {
        System.err.println("Error: No file found at: " + location);
    }
}
```

.

```
age=23
color=green
name=Steve
```

: https://riptutorial.com/ko/java/topic/576/-

124: AppDynamics TIBCO BusinessWorks

AppDynamics , () . AppD TIBCO BW . BW .

Examples

Appdynamics BW

1. TIBCO_HOME / bw / 5.12 / bin / bwengine.tra (Linux) TIBCO BW bwengine.tra .
2. .

*** . . ***

3. . tibco.deployment = % tibco.deployment %
4. . java.extended.properties = -javaagent : /opt/appd/current/appagent/javaagent.jar -Dappdynamics.http.proxyHost =? -Dappdynamics.http.proxyPort =? -Dappdynamics.agent.applicationName =? -Dappdynamics.agent.tierName =? -Dappdynamics.agent.nodeName = % tibco.deployment % -Dappdynamics.controller.ssl.enabled =? -Dappdynamics.controller.sslPort =? -Dappdynamics.agent.logs.dir =? -Dappdynamics.agent.runtime.dir =? -Dappdynamics.controller.hostName =? -Dappdynamics.controller.port =? -Dappdynamics.agent.accountName =? -Dappdynamics.agent.accountAccessKey =?
5. . .

AppDynamics TIBCO BusinessWorks : https://riptutorial.com/ko/java/topic/10602/--appdynamics--tibco-businessworks-

125:

Java enum 100A, 25K . _ () .

Examples

```
public enum BookCode {
    _10A("Simon Haykin", "Communication System"),
    _42B("Stefan Hakins", "A Brief History of Time"),
    E1("Sedra Smith", "Electronics Circuits");

    private String author;
    private String title;

    BookCode(String author, String title) {
        this.author = author;
        this.title = title;
    }

    public String getName() {
        String name = name();
        if (name.charAt(0) == '_') {
            name = name.substring(1, name.length());
        }
        return name;
    }

    public static BookCode of(String code) {
        if (Character.isDigit(code.charAt(0))) {
            code = "_" + code;
        }
        return BookCode.valueOf(code);
    }
}
```

: https://riptutorial.com/ko/java/topic/10719/---

126:

Examples

3 .

1.
2.
3.

`super` . .

- : `super();`
- : `super(int no, double amount, String name);`

```
class Parentclass
{
   Parentclass(){
      System.out.println("Constructor of Superclass");
   }
}
class Subclass extends Parentclass
{
   Subclass(){
    /* Compile adds super() here at the first line
     * of this constructor implicitly
     */
    System.out.println("Constructor of Subclass");
   }
   Subclass(int n1){
    /* Compile adds super() here at the first line
     * of this constructor implicitly
     */
    System.out.println("Constructor with arg");
   }
   void display(){
    System.out.println("Hello");
   }
   public static void main(String args[]){
    // Creating object using default constructor
    Subclass obj= new Subclass();
    //Calling sub class method
       obj.display();
       //Creating object 2 using arg constructor
       Subclass obj2= new Subclass(10);
       obj2.display();
  }
}
```

: `super()` . .

`super` (override) . `super` .

```
class Parentclass
{
   //Overridden method
   void display(){
    System.out.println("Parent class method");
   }
}
class Subclass extends Parentclass
{
   //Overriding method
   void display(){
    System.out.println("Child class method");
   }
   void printMsg(){
    //This would call Overriding method
    display();
    //This would call Overridden method
    super.display();
   }
   public static void main(String args[]){
    Subclass obj= new Subclass();
    obj.printMsg();
   }
}
```

: super .

super . . / / super .

```
//Parent class or Superclass
class Parentclass
{
    int num=100;
}
//Child class or subclass
class Subclass extends Parentclass
{
    /* I am declaring the same variable
     * num in child class too.
     */
    int num=110;
    void printNumber(){
     System.out.println(num); //It will print value 110
     System.out.println(super.num); //It will print value 100
    }
    public static void main(String args[]){
       Subclass obj= new Subclass();
       obj.printNumber();
    }
}
```

: super .

: https://riptutorial.com/ko/java/topic/5764/-

127:

- = ( );
- = (System.in);

Source String, File InputStream .

`Scanner` Java 5 . Java 6 `reset()` () `Path` Java 7 .

Examples

```
Scanner scanner = new Scanner(System.in); //Scanner obj to read System input
String inputTaken = new String();
while (true) {
    String input = scanner.nextLine(); // reading one line of input
    if (input.matches("\\s+"))         // if it matches spaces/tabs, stop reading
        break;
    inputTaken += input + " ";
}
System.out.println(inputTaken);
```

. keyboar , `Reading from keyboard` .

```
Reading
from
keyboard
  //space
```

Scanner

```
Scanner scanner = null;
try {
    scanner = new Scanner(new File("Names.txt"));
    while (scanner.hasNext()) {
        System.out.println(scanner.nextLine());
    }
} catch (Exception e) {
    System.err.println("Exception occurred!");
} finally {
    if (scanner != null)
        scanner.close();
}
```

`Scanner` `File` . File scanner . `scanner.hasNext()` . `while` `Names.txt` . `nextLine()` , `nextInt()` , `nextBoolean()` . `scanner.nextLine()` . `nextLine()` `scanner` . `.close()` .

(Java 7) try .

```
try (Scanner scanner = new Scanner(new File("Names.txt"))) {
    while (scanner.hasNext()) {
        System.out.println(scanner.nextLine());
    }
} catch (Exception e) {
    System.err.println("Exception occurred!");
}
```

Scanner String .

Scanner \Z ( ) . , .

```
String content = new Scanner(new File("filename")).useDelimiter("\\Z").next();
System.out.println(content);
```

Scanner IoException throw IoException .

.useDelimiter(",") Scanner (.useDelimiter(",") . String.split(...) . Scanner .

```
Scanner scanner = null;
try{
    scanner = new Scanner("i,like,unicorns").useDelimiter(",");;
    while(scanner.hasNext()){
        System.out.println(scanner.next());
    }
}catch(Exception e){
    e.printStackTrace();
}finally{
    if (scanner != null)
        scanner.close();
}
```

. CSV CSV (Java CSV).

java.util.Scanner System.in (stdin , C, C ++ Unix Linux). .

```
package com.stackoverflow.scanner;

import javax.annotation.Nonnull;
import java.math.BigInteger;
import java.net.MalformedURLException;
import java.net.URL;
import java.util.*;
import java.util.regex.Pattern;

import static java.lang.String.format;

public class ScannerExample
{
    private static final Set<String> EXIT_COMMANDS;
    private static final Set<String> HELP_COMMANDS;
    private static final Pattern DATE_PATTERN;
    private static final String HELP_MESSAGE;

    static
```

```
    {
        final SortedSet<String> ecmds = new TreeSet<String>(String.CASE_INSENSITIVE_ORDER);
        ecmds.addAll(Arrays.asList("exit", "done", "quit", "end", "fino"));
        EXIT_COMMANDS = Collections.unmodifiableSortedSet(ecmds);
        final SortedSet<String> hcmds = new TreeSet<String>(String.CASE_INSENSITIVE_ORDER);
        hcmds.addAll(Arrays.asList("help", "helpi", "?"));
        HELP_COMMANDS = Collections.unmodifiableSet(hcmds);
        DATE_PATTERN = Pattern.compile("\\d{4}([-\\/])\\d{2}\\1\\d{2}"); //
http://regex101.com/r/xB8dR3/1
        HELP_MESSAGE = format("Please enter some data or enter one of the following commands
to exit %s", EXIT_COMMANDS);
    }

    /**
     * Using exceptions to control execution flow is always bad.
     * That is why this is encapsulated in a method, this is done this
     * way specifically so as not to introduce any external libraries
     * so that this is a completely self contained example.
     * @param s possible url
     * @return true if s represents a valid url, false otherwise
     */
    private static boolean isValidURL(@Nonnull final String s)
    {
        try { new URL(s); return true; }
        catch (final MalformedURLException e) { return false; }
    }

    private static void output(@Nonnull final String format, @Nonnull final Object... args)
    {
        System.out.println(format(format, args));
    }

    public static void main(final String[] args)
    {
        final Scanner sis = new Scanner(System.in);
        output(HELP_MESSAGE);
        while (sis.hasNext())
        {
            if (sis.hasNextInt())
            {
                final int next = sis.nextInt();
                output("You entered an Integer = %d", next);
            }
            else if (sis.hasNextLong())
            {
                final long next = sis.nextLong();
                output("You entered a Long = %d", next);
            }
            else if (sis.hasNextDouble())
            {
                final double next = sis.nextDouble();
                output("You entered a Double = %f", next);
            }
            else if (sis.hasNext("\\d+"))
            {
                final BigInteger next = sis.nextBigInteger();
                output("You entered a BigInteger = %s", next);
            }
            else if (sis.hasNextBoolean())
            {
                final boolean next = sis.nextBoolean();
```

```
                output("You entered a Boolean representation = %s", next);
            }
            else if (sis.hasNext(DATE_PATTERN))
            {
                final String next = sis.next(DATE_PATTERN);
                output("You entered a Date representation = %s", next);
            }
            else // unclassified
            {
                final String next = sis.next();
                if (isValidURL(next))
                {
                    output("You entered a valid URL = %s", next);
                }
                else
                {
                    if (EXIT_COMMANDS.contains(next))
                    {
                        output("Exit command %s issued, exiting!", next);
                        break;
                    }
                    else if (HELP_COMMANDS.contains(next)) { output(HELP_MESSAGE); }
                    else { output("You entered an unclassified String = %s", next); }
                }
            }
        }
        /*
           This will close the underlying Readable, in this case System.in, and free those
resources.
           You will not be to read from System.in anymore after this you call .close().
           If you wanted to use System.in for something else, then don't close the Scanner.
        */
        sis.close();
        System.exit(0);
    }
}
```

int

```
import java.util.Scanner;

Scanner s = new Scanner(System.in);
int number = s.nextInt();
```

int . System.in Scanner . Scanner int . int .

System.in , InputStream . .) `java.util.NoSuchElementException` `java.lang.IllegalStateException` throw.

:

```
    Scanner sc1 = new Scanner(System.in);
    Scanner sc2 = new Scanner(System.in);
    int x1 = sc1.nextInt();
    sc1.close();
    // java.util.NoSuchElementException
    int x2 = sc2.nextInt();
```

```
    // java.lang.IllegalStateException
    x2 = sc1.nextInt();
```

: https://riptutorial.com/ko/java/topic/551/

128: API

Java 9 `sun.reflect.Reflection` . `sun.reflect.Reflection::getCallerClass` . .

API JDK (9) `java.lang.StackWalker` . API .

Examples

.

```
1  package test;
2
3  import java.lang.StackWalker.StackFrame;
4  import java.lang.reflect.InvocationTargetException;
5  import java.lang.reflect.Method;
6  import java.util.List;
7  import java.util.stream.Collectors;
8
9  public class StackWalkerExample {
10
11    public static void main(String[] args) throws NoSuchMethodException, SecurityException,
IllegalAccessException, IllegalArgumentException, InvocationTargetException {
12        Method fooMethod = FooHelper.class.getDeclaredMethod("foo", (Class<?>[])null);
13        fooMethod.invoke(null, (Object[]) null);
14    }
15 }
16
17 class FooHelper {
18    protected static void foo() {
19        BarHelper.bar();
20    }
21 }
22
23 class BarHelper {
24    protected static void bar() {
25        List<StackFrame> stack = StackWalker.getInstance()
26                .walk((s) -> s.collect(Collectors.toList()));
27        for(StackFrame frame : stack) {
28            System.out.println(frame.getClassName() + " " + frame.getLineNumber() + " " +
frame.getMethodName());
29        }
30    }
31 }
```

:

```
test.BarHelper 26 bar
test.FooHelper 19 foo
test.StackWalkerExample 13 main
```

. , `StackWalker` `RETAIN_CLASS_REFERENCE` Class `StackFrame` . .

```
public class StackWalkerExample {
```

```
    public static void main(String[] args) {
        FooHelper.foo();
    }

}

class FooHelper {
    protected static void foo() {
        BarHelper.bar();
    }
}

class BarHelper {
    protected static void bar() {

System.out.println(StackWalker.getInstance(Option.RETAIN_CLASS_REFERENCE).getCallerClass());
    }
}
```

:

```
class test.FooHelper
```

/ . . , StackWalker SHOW_REFLECT_FRAMES .

```
package test;

import java.lang.StackWalker.Option;
import java.lang.StackWalker.StackFrame;
import java.lang.reflect.InvocationTargetException;
import java.lang.reflect.Method;
import java.util.List;
import java.util.stream.Collectors;

public class StackWalkerExample {

    public static void main(String[] args) throws NoSuchMethodException, SecurityException,
IllegalAccessException, IllegalArgumentException, InvocationTargetException {
        Method fooMethod = FooHelper.class.getDeclaredMethod("foo", (Class<?>[])null);
        fooMethod.invoke(null, (Object[]) null);
    }
}

class FooHelper {
    protected static void foo() {
        BarHelper.bar();
    }
}

class BarHelper {
    protected static void bar() {
        // show reflection methods
        List<StackFrame> stack = StackWalker.getInstance(Option.SHOW_REFLECT_FRAMES)
                .walk((s) -> s.collect(Collectors.toList()));
        for(StackFrame frame : stack) {
            System.out.println(frame.getClassName() + " " + frame.getLineNumber() + " " +
frame.getMethodName());
```

```
        }
    }
}
```

:

```
test.BarHelper 27 bar
test.FooHelper 20 foo
jdk.internal.reflect.NativeMethodAccessorImpl -2 invoke0
jdk.internal.reflect.NativeMethodAccessorImpl 62 invoke
jdk.internal.reflect.DelegatingMethodAccessorImpl 43 invoke
java.lang.reflect.Method 563 invoke
test.StackWalkerExample 14 main
```

`StackFrame.getLineNumber()` .

API : https://riptutorial.com/ko/java/topic/9868/--api

129:

`Stream` . 8 `Collection` `Stream` : `stream()` `parallelStream()` . `Stream` . `Stream` `Stream` . void .

- collection.stream ()
- Array.stream ()
- Stream.iterate (firstValue, currentValue -> nextValue)
- Stream.generate (() -> value)
- Stream.of (elementOfT [, elementOfT, ...])
- Stream.empty ()
- StreamSupport.stream (iterable.spliterator (), false)

Examples

`Stream` . `Stream` . `Stream` . .

: (Ideone)

```
Stream<String> fruitStream = Stream.of("apple", "banana", "pear", "kiwi", "orange");

fruitStream.filter(s -> s.contains("a"))
           .map(String::toUpperCase)
           .sorted()
           .forEach(System.out::println);
```

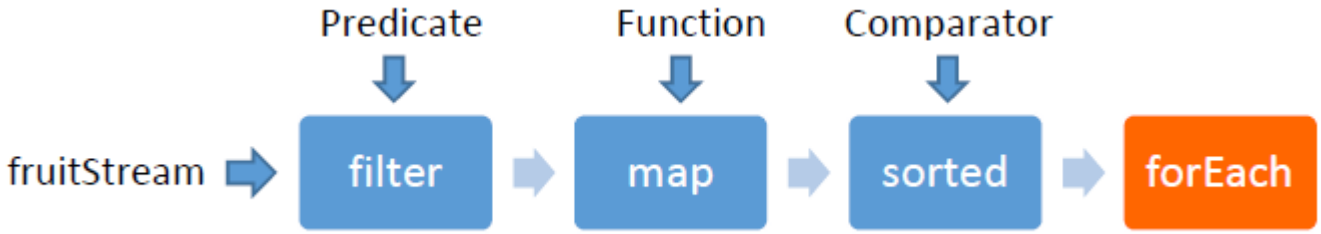
:

.

1. `Stream.of(values)` `String` `Stream` `Stream<String>` .
2. `filter()` ( ). `"a"` . .
3. `map()` (mapper) . , fruit `String` method-reference `String::toUppercase` `String` .

   `map()` . `Stream<String> .map(String::isEmpty)` `Stream<Boolean>`
4. `sorted()` , `Stream` `sorted()` `String` ,).
5. , `forEach(action)` `Stream` . . .

   `Stream` . . . `Stream` .



.

Stream . IO . Stream .

Stream AutoCloseable . close try-with-resource .

Stream Stream .

```
try (Stream<String> lines = Files.lines(Paths.get("somePath"))) {
    lines.forEach(System.out::println);
}
```

Stream Runnable Stream.onClose() . .

```
public Stream<String>streamAndDelete(Path path) throws IOException {
    return Files.lines(path).onClose(() -> someClass.deletePath(path));
}
```

try-with-resources close() .

Stream .

Stream . SortedMap List , Stream , . . Java HashMap keySet() .

:

```
List<Integer> integerList = Arrays.asList(0, 1, 2, 3, 42);

// sequential
long howManyOddNumbers = integerList.stream()
                                    .filter(e -> (e % 2) == 1)
                                    .count();

System.out.println(howManyOddNumbers); // Output: 2
```

.

Stream . , Stream 1 sort . Stream - Stream .

:

```
// parallel
```

```
long howManyOddNumbersParallel = integerList.parallelStream()
                                            .filter(e -> (e % 2) == 1)
                                            .count();

System.out.println(howManyOddNumbersParallel); // Output: 2
```

( )

. . , Stream . . Stream Collection .

toList() toSet() ▪

Stream Stream.collect .

```
System.out.println(Arrays
    .asList("apple", "banana", "pear", "kiwi", "orange")
    .stream()
    .filter(s -> s.contains("a"))
    .collect(Collectors.toList())
);
// prints: [apple, banana, pear, orange]
```

Set Collectors . , Collectors.toSet() Stream Set .

List Set

Collectors#toList() Collectors#toSet() List Set , , .

, Collectors#toCollection(Supplier) .

```
// syntax with method reference
System.out.println(strings
        .stream()
        .filter(s -> s != null && s.length() <= 3)
        .collect(Collectors.toCollection(ArrayList::new))
);

// syntax with lambda
System.out.println(strings
        .stream()
        .filter(s -> s != null && s.length() <= 3)
        .collect(Collectors.toCollection(() -> new LinkedHashSet<>()))
);
```

toMap

Map . key ID Value .

```
  List<Student> students = new ArrayList<Student>();
```

```
    students.add(new Student(1,"test1"));
    students.add(new Student(2,"test2"));
    students.add(new Student(3,"test3"));

    Map<Integer, String> IdToName = students.stream()
        .collect(Collectors.toMap(Student::getId, Student::getName));
    System.out.println(IdToName);
```

:

```
{1=test1, 2=test2, 3=test3}
```

Collector.toMap `Collector<T, ?, Map<K,U>> toMap(Function<? super T, ? extends K> keyMapper, Function<? super T, ? extends U> valueMapper, BinaryOperator<U> mergeFunction)` . mergeFunction .

mergeFunction `(s1, s2) -> s1` `(s1, s2) -> s2` .

: ArrayList Map <String, List <>>

. : , .

```
    List<Student> list = new ArrayList<>();
    list.add(new Student("Davis", SUBJECT.MATH, 35.0));
    list.add(new Student("Davis", SUBJECT.SCIENCE, 12.9));
    list.add(new Student("Davis", SUBJECT.GEOGRAPHY, 37.0));

    list.add(new Student("Sascha", SUBJECT.ENGLISH, 85.0));
    list.add(new Student("Sascha", SUBJECT.MATH, 80.0));
    list.add(new Student("Sascha", SUBJECT.SCIENCE, 12.0));
    list.add(new Student("Sascha", SUBJECT.LITERATURE, 50.0));

    list.add(new Student("Robert", SUBJECT.LITERATURE, 12.0));

    Map<String, List<SUBJECT>> map = new HashMap<>();
    list.stream().forEach(s -> {
                map.computeIfAbsent(s.getName(), x -> new ArrayList<>()).add(s.getSubject());
            });
    System.out.println(map);
```

:

```
{ Robert=[LITERATURE],
Sascha=[ENGLISH, MATH, SCIENCE, LITERATURE],
Davis=[MATH, SCIENCE, GEOGRAPHY] }
```

: ArrayList Map <String, Map <>>

```
    List<Student> list = new ArrayList<>();
    list.add(new Student("Davis", SUBJECT.MATH, 1, 35.0));
    list.add(new Student("Davis", SUBJECT.SCIENCE, 2, 12.9));
    list.add(new Student("Davis", SUBJECT.MATH, 3, 37.0));
    list.add(new Student("Davis", SUBJECT.SCIENCE, 4, 37.0));
```

```
    list.add(new Student("Sascha", SUBJECT.ENGLISH, 5, 85.0));
    list.add(new Student("Sascha", SUBJECT.MATH, 1, 80.0));
    list.add(new Student("Sascha", SUBJECT.ENGLISH, 6, 12.0));
    list.add(new Student("Sascha", SUBJECT.MATH, 3, 50.0));

    list.add(new Student("Robert", SUBJECT.ENGLISH, 5, 12.0));

    Map<String, Map<SUBJECT, List<Double>>> map = new HashMap<>();

    list.stream().forEach(student -> {
        map.computeIfAbsent(student.getName(), s -> new HashMap<>())
                .computeIfAbsent(student.getSubject(), s -> new ArrayList<>())
                .add(student.getMarks());
    });

    System.out.println(map);
```

:

```
{ Robert={ENGLISH=[12.0]},
Sascha={MATH=[80.0, 50.0], ENGLISH=[85.0, 12.0]},
Davis={MATH=[35.0, 37.0], SCIENCE=[12.9, 37.0]} }
```

List	Collectors.toList()
ArrayList	Collectors.toCollection(() -> new ArrayList<>(size))
Set	Collectors.toSet()
Set	Collectors.toCollection(() -> new LinkedHashSet<>())
/ Set<String>	Collectors.toCollection(() -> new TreeSet<>(String.CASE_INSENSITIVE_ORDER))
EnumSet<AnEnum> ( )	Collectors.toCollection(() -> EnumSet.noneOf(AnEnum.class))
Map<K,V>	Collectors.toMap(keyFunc,valFunc)
MyObject.getter () MyObject	Collectors.toMap(MyObject::getter, Function.identity())
MyObject.getter () MyObject	Collectors.groupingBy(MyObject::getter)

Stream . Stream Stream .Stream limit , Java Stream .

Stream 1 , Stream . Stream Stream .

```
// Generate infinite stream - 1, 2, 3, 4, 5, 6, 7, ...
IntStream naturalNumbers = IntStream.iterate(1, x -> x + 1);

// Print out only the first 5 terms
```

```
naturalNumbers.limit(5).forEach(System.out::println);
```

:

1
2

4
5

Stream.generate . Supplier .

```
// Generate an infinite stream of random numbers
Stream<Double> infiniteRandomNumbers = Stream.generate(Math::random);

// Print out only the first 10 random numbers
infiniteRandomNumbers.limit(10).forEach(System.out::println);
```

Stream count() , collect() forEach() . Stream .

Stream peek() filter() .

```
IntStream.range(1, 10).filter(a -> a % 2 == 0).peek(System.out::println);
```

Stream .

peek forEach .

```
IntStream.range(1, 10).filter(a -> a % 2 == 0).forEach(System.out::println);
```

:

2
4
6
8

Stream .

Iterable . .

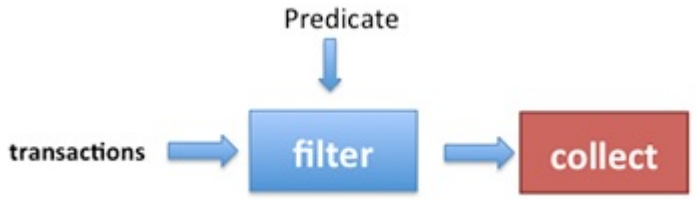
```
List<String> list = Arrays.asList("FOO", "BAR");
Iterable<String> iterable = () -> list.stream().map(String::toLowerCase).iterator();

for (String str : iterable) {
    System.out.println(str);
}
for (String str : iterable) {
    System.out.println(str);
```

```
}
```

:

`Iterable` `Iterator<T> iterator()` . (lambda) .

`Stream` .



: .

```
try {
    IntStream.range(1, 10).filter(null);
} catch (NullPointerException e) {
    System.out.println("We got a NullPointerException as null was passed as an argument to
filter()");
}
```

:

filter () null NullPointerException .

`groupingBy(classifier, downstream)` `Stream` `Map` .

`Map` `Stream` . . `counting()` `counting()` .

```
Stream.of("apple", "orange", "banana", "apple")
      .collect(Collectors.groupingBy(Function.identity(), Collectors.counting()))
      .entrySet()
      .forEach(System.out::println);
```

String `Long` (`Collectors.counting()`). `collect` `Map<String, Long>` .

.

= 1
= 1
= 2

: Stream ParallelStream Sequential Stream .

Stream .

```
List<String> data = Arrays.asList("One", "Two", "Three", "Four", "Five");
Stream<String> aParallelStream = data.stream().parallel();
```

:

```
Stream<String> aParallelStream = data.parallelStream();
```

.

```
aParallelStream.forEach(System.out::println);
```

Stream () :

( ). parallelStream .

Stream Stream - .

Stream CPU .

Stream Optional A Stream Optional . (, null Optional.empty()).

```
 Optional<String> op1 = Optional.empty();
 Optional<String> op2 = Optional.of("Hello World");

 List<String> result = Stream.of(op1, op2)
                             .filter(Optional::isPresent)
                             .map(Optional::get)
                             .collect(Collectors.toList());

 System.out.println(result); //[Hello World]
```

Java Collection<E> Stream<E> stream() parallelStream() .

```
Collection<String> stringList = new ArrayList<>();
Stream<String> stringStream = stringList.parallelStream();
```

Stream<E> .

```
String[] values = { "aaa", "bbbb", "ddd", "cccc" };
Stream<String> stringStream = Arrays.stream(values);
Stream<String> stringStreamAlternative = Stream.of(values);
```

Arrays.stream() Stream.of() Stream.of() varargs .

```
Stream<Integer> integerStream = Stream.of(1, 2, 3);
```

Stream . :

```
IntStream intStream = IntStream.of(1, 2, 3);
DoubleStream doubleStream = DoubleStream.of(1.0, 2.0, 3.0);
```

Arrays.stream() .

```
IntStream intStream = Arrays.stream(new int[]{ 1, 2, 3 });
```

Stream .

```
int[] values= new int[]{1, 2, 3, 4, 5};
IntStream intStram = Arrays.stream(values, 1, 3);
```

boxed boxed .

```
Stream<Integer> integerStream = intStream.boxed();
```

Collector collect .

. . .

```
Supplier<Stream<String>> streamSupplier = () -> Stream.of("apple", "banana","orange",
"grapes", "melon","blueberry","blackberry")
.map(String::toUpperCase).sorted();

  streamSupplier.get().filter(s ->   s.startsWith("A")).forEach(System.out::println);

// APPLE

  streamSupplier.get().filter(s -> s.startsWith("B")).forEach(System.out::println);

  // BANANA
  // BLACKBERRY
  // BLUEBERRY
```

int[] List<Integer> .

```
int[] ints = {1,2,3};
List<Integer> list = IntStream.of(ints).boxed().collect(Collectors.toList());
```

Java 8 count , min , max , sum average IntSummaryStatistics , DoubleSummaryStatistics LongSummaryStatistics .

Java SE 8

```
List<Integer> naturalNumbers = Arrays.asList(1, 2, 3, 4, 5, 6, 7, 8, 9, 10);
IntSummaryStatistics stats = naturalNumbers.stream()
                                           .mapToInt((x) -> x)
                                           .summaryStatistics();
System.out.println(stats);
```

.

Java SE 8

```
IntSummaryStatistics{count=10, sum=55, min=1, max=10, average=5.500000}
```

: 21 50 30 `Stream` .

```
final long n = 20L; // the number of elements to skip
final long maxSize = 30L; // the number of elements the stream should be limited to
final Stream<T> slice = collection.stream().skip(n).limit(maxSize);
```

:

- `n` , `maxSize` , `IllegalArgumentException` Throw
- `skip(long)` `limit(long)` .
- `n` `skip(n)` .
- `skip(long)` `limit(long)` ,

:

```
Collection<String> abc = Arrays.asList("a", "b", "c");
Collection<String> digits = Arrays.asList("1", "2", "3");
Collection<String> greekAbc = Arrays.asList("alpha", "beta", "gamma");
```

1 - `Stream`

```
final Stream<String> concat1 = Stream.concat(abc.stream(), digits.stream());

concat1.forEach(System.out::print);
// prints: abc123
```

2 - `Stream`

```
final Stream<String> concat2 = Stream.concat(
    Stream.concat(abc.stream(), digits.stream()),
    greekAbc.stream());

System.out.println(concat2.collect(Collectors.joining(", ")));
// prints: a, b, c, 1, 2, 3, alpha, beta, gamma
```

`concat()` `Stream` `flatMap()` .

```
final Stream<String> concat3 = Stream.of(
    abc.stream(), digits.stream(), greekAbc.stream())
```

```
    .flatMap(s -> s);
    // or `.flatMap(Function.identity());` (java.util.function.Function)

System.out.println(concat3.collect(Collectors.joining(", ")));
// prints: a, b, c, 1, 2, 3, alpha, beta, gamma
```

Stream StackOverflowException Stream .

IntStream String

, String Stream Character , IntStream String.codePoints() . IntStream :

```
public IntStream stringToIntStream(String in) {
  return in.codePoints();
}
```

, IntStreamToString . :

```
public String intStreamToString(IntStream intStream) {
  return intStream.collect(StringBuilder::new, StringBuilder::appendCodePoint,
StringBuilder::append).toString();
}
```

```
List<String> data = new ArrayList<>();
data.add("Sydney");
data.add("London");
data.add("New York");
data.add("Amsterdam");
data.add("Mumbai");
data.add("California");

System.out.println(data);

List<String> sortedData = data.stream().sorted().collect(Collectors.toList());

System.out.println(sortedData);
```

:

```
[Sydney, London, New York, Amsterdam, Mumbai, California]
[Amsterdam, California, London, Mumbai, New York, Sydney]
```

sorted .

.

```
List<String> sortedData2 = data.stream().sorted((s1,s2) ->
s2.compareTo(s1)).collect(Collectors.toList());
```

[Sydney, New York, Mumbai, London, California, Amsterdam]

Comparator.reverseOrder() Comparator.reverseOrder() , reverse .

```
List<String> reverseSortedData =
data.stream().sorted(Comparator.reverseOrder()).collect(Collectors.toList());
```

Java `IntStream` (`int`), `LongStream` (`long`) `DoubleStream` (`double` 3 (primitive) `Stream` . , .
:

```
IntStream is = IntStream.of(10, 20, 30);
double average = is.average().getAsDouble(); // average is 20.0
```

Analog `collect()` `Stream` , `Stream.toArray()` :

```
List<String> fruits = Arrays.asList("apple", "banana", "pear", "kiwi", "orange");

String[] filteredFruits = fruits.stream()
    .filter(s -> s.contains("a"))
    .toArray(String[]::new);

// prints: [apple, banana, pear, orange]
System.out.println(Arrays.toString(filteredFruits));
```

`String[]::new` : .

`Stream` .

`50000` `Integer` .

```
IntStream.iterate(1, i -> i + 1) // Generate an infinite stream 1,2,3,4...
    .filter(i -> (i*i) > 50000) // Filter to find elements where the square is >50000
    .findFirst(); // Find the first filtered element
```

`OptionalInt` .

`Stream` Java . `Stream` Java `OptionalInt` .

IntStream

`Stream` . `ArrayList` `IntStream.range(start, endExclusive)` .

```
String[] names = { "Jon", "Darin", "Bauke", "Hans", "Marc" };

IntStream.range(0, names.length)
    .mapToObj(i -> String.format("#%d %s", i + 1, names[i]))
    .forEach(System.out::println);
```

`range(start, endExclusive)` `ÌntStream` `mapToObj(mapper)` `String` .

:

1 Jon
2 Darin
3 Bauke

4
5 Marc

for .

```
for (int i = 0; i < names.length; i++) {
    String newName = String.format("#%d %s", i + 1, names[i]);
    System.out.println(newName);
}
```

flatMap ()

Stream Stream .

.

```
List<String> list1 =  Arrays.asList("one", "two");
      List<String> list2 =  Arrays.asList("three","four","five");
      List<String> list3 =  Arrays.asList("six");
          List<String> finalList = Stream.of(list1, list2,
list3).flatMap(Collection::stream).collect(Collectors.toList());
System.out.println(finalList);

// [one, two, three, four, five, six]
```

.

```
Map<String, List<Integer>> map = new LinkedHashMap<>();
map.put("a", Arrays.asList(1, 2, 3));
map.put("b", Arrays.asList(4, 5, 6));

List<Integer> allValues = map.values() // Collection<List<Integer>>
        .stream()                      // Stream<List<Integer>>
        .flatMap(List::stream)         // Stream<Integer>
        .collect(Collectors.toList());

System.out.println(allValues);
// [1, 2, 3, 4, 5, 6]
```

Map List Stream .

```
List<Map<String, String>> list = new ArrayList<>();
Map<String,String> map1 = new HashMap();
map1.put("1", "one");
map1.put("2", "two");

Map<String,String> map2 = new HashMap();
map2.put("3", "three");
map2.put("4", "four");
list.add(map1);
list.add(map2);


Set<String> output= list.stream()  //  Stream<Map<String, String>>
    .map(Map::values)              // Stream<List<String>>
```

```
    .flatMap(Collection::stream)   // Stream<String>
    .collect(Collectors.toSet());  //Set<String>
// [one, two, three,four]
```

```
Stream<String> characters = Stream.of("A", "B", "C");

Map<Integer, String> map = characters
            .collect(Collectors.toMap(element -> element.hashCode(), element -> element));
// map = {65=A, 66=B, 67=C}
```

Function - Function.identity() . element -> element Function.identity() .

Collectors.toMap javadoc .

Object.equals(Object) , IllegalStateException Throw. toMap(Function, Function, BinaryOperator) .

```
Stream<String> characters = Stream.of("A", "B", "B", "C");

Map<Integer, String> map = characters
            .collect(Collectors.toMap(
                element -> element.hashCode(),
                element -> element,
                (existingVal, newVal) -> (existingVal + newVal)));

// map = {65=A, 66=BB, 67=C}
```

Collectors.toMap(...) BinaryOperator . :

- (), .
-
-

"" Collectors.groupingBy . , .

```
List<Person> people = Arrays.asList(
    new Person("Sam", "Rossi"),
    new Person("Sam", "Verdi"),
    new Person("John", "Bianchi"),
    new Person("John", "Rossi"),
    new Person("John", "Verdi")
);

Map<String, List<String>> map = people.stream()
        .collect(
                // function mapping input elements to keys
                Collectors.groupingBy(Person::getName,
                // function mapping input elements to values,
                // how to store values
                Collectors.mapping(Person::getSurname, Collectors.toList()))
        );

// map = {John=[Bianchi, Rossi, Verdi], Sam=[Rossi, Verdi]}
```

ID Strings . Stream .

. String SecureRandom .

: SecureRandom , setSeed() setSeed() 1 .

```
private static final SecureRandom rng = new SecureRandom(SecureRandom.generateSeed(20));
//20 Bytes as a seed is rather arbitrary, it is the number used in the JavaDoc example
```

String ,, ( ) . Stream boolean .

```
//returns true for all chars in 0-9, a-z and A-Z
boolean useThisCharacter(char c){
    //check for range to avoid using all unicode Letter (e.g. some chinese symbols)
    return c >= '0' && c <= 'z' && Character.isLetterOrDigit(c);
}
```

RNG useThisCharacter charset String .

```
public String generateRandomString(long length){
    //Since there is no native CharStream, we use an IntStream instead
    //and convert it to a Stream<Character> using mapToObj.
    //We need to specify the boundaries for the int values to ensure they can safely be cast
to char
    Stream<Character> randomCharStream = rng.ints(Character.MIN_CODE_POINT,
Character.MAX_CODE_POINT).mapToObj(i -> (char)i).filter(c ->
this::useThisCharacter).limit(length);

    //now we can use this Stream to build a String utilizing the collect method.
    String randomString = randomCharStream.collect(StringBuilder::new, StringBuilder::append,
StringBuilder::append).toString();
    return randomString;
}
```

Stream , IntStream , (Σ) . Stream .

, Madhava Pi (: wikipedia) .

$$\pi = \sqrt{12}\sum_{k=0}^{\infty}\frac{(-3)^{-k}}{2k+1} = \sqrt{12}\sum_{k=0}^{\infty}\frac{(-\frac{1}{3})^{k}}{2k+1} = \sqrt{12}\left(\frac{1}{1\cdot 3^0} - \frac{1}{3\cdot 3^1} + \frac{1}{5\cdot 3^2} - \frac{1}{7\cdot 3^3} + \cdots\right)$$

. , 101 :

```
double pi = Math.sqrt(12) *
            IntStream.rangeClosed(0, 100)
                     .mapToDouble(k -> Math.pow(-3, -1 * k) / (2 * k + 1))
                     .sum();
```

: double 29 Math.Pi .

, Stream . .

```
public interface Ordered {
    default int getOrder(){
        return 0;
```

```
    }
}

public interface Valued<V extends Ordered> {
    boolean hasPropertyTwo();
    V getValue();
}

public interface Thing<V extends Ordered> {
    boolean hasPropertyOne();
    Valued<V> getValuedProperty();
}

public <V extends Ordered> List<V> myMethod(List<Thing<V>> things) {
    List<V> results = new ArrayList<V>();
    for (Thing<V> thing : things) {
        if (thing.hasPropertyOne()) {
            Valued<V> valued = thing.getValuedProperty();
            if (valued != null && valued.hasPropertyTwo()){
                V value = valued.getValue();
                if (value != null){
                    results.add(value);
                }
            }
        }
    }
    results.sort((a, b)->{
        return Integer.compare(a.getOrder(), b.getOrder());
    });
    return results;
}
```

Stream . .

```
public <V extends Ordered> List<V> myMethod(List<Thing<V>> things) {
    return things.stream()
        .filter(Thing::hasPropertyOne)
        .map(Thing::getValuedProperty)
        .filter(Objects::nonNull)
        .filter(Valued::hasPropertyTwo)
        .map(Valued::getValue)
        .filter(Objects::nonNull)
        .sorted(Comparator.comparing(Ordered::getOrder))
        .collect(Collectors.toList());
}
```

Map.Entry

Stream Stream Map.Entry<K,V> .

```
public static <K, V> Function<K, Map.Entry<K, V>> entryMapper(Function<K, V> mapper){
    return (k)->new AbstractMap.SimpleEntry<>(k, mapper.apply(k));
}
```

Stream .

```
Set<K> mySet;
Function<K, V> transformer = SomeClass::transformerMethod;
Stream<Map.Entry<K, V>> entryStream = mySet.stream()
    .map(entryMapper(transformer));
```

Stream . .

.

:

Stream.map . . .

```
Arrays.asList(1, 2 ,3).stream().map(i -> {
    throw new RuntimeException("not gonna happen");
    return i;
});
```

, .

. Stream.forEach Stream.collect . .

. Stateless . Stream.map Stream.filter .

Statefulness . . . Stream.sorted . ( ) .

```
// works - stateless stream
long BIG_ENOUGH_NUMBER = 999999999;
IntStream.iterate(0, i -> i + 1).limit(BIG_ENOUGH_NUMBER).forEach(System.out::println);
```

Stream.sorted .

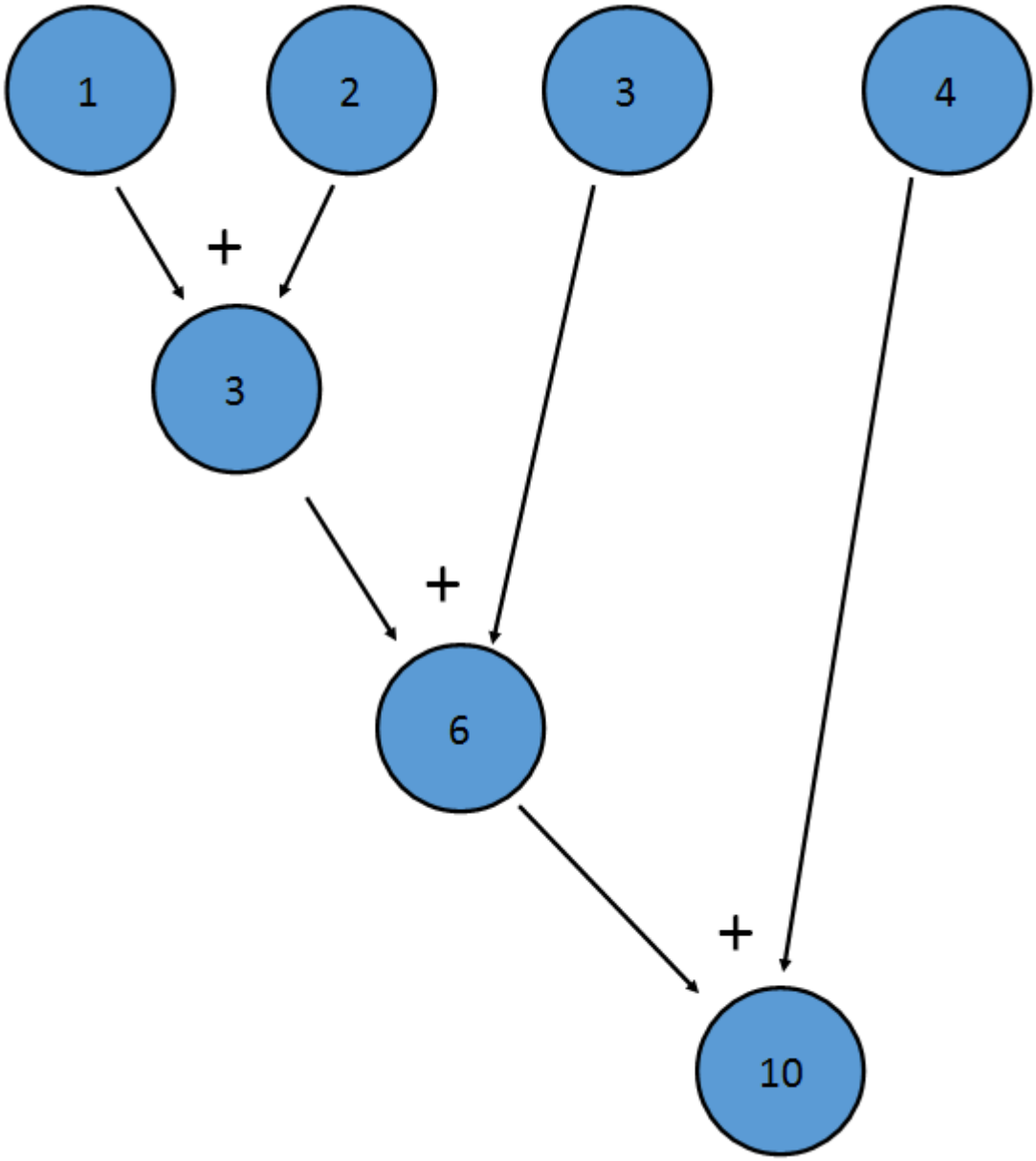
```
// Out of memory - stateful stream
IntStream.iterate(0, i -> i +
1).limit(BIG_ENOUGH_NUMBER).sorted().forEach(System.out::println);
```

Spliterators.spliterator() Spliterators.spliteratorUnknownSize() .

```
Iterator<String> iterator = Arrays.asList("A", "B", "C").iterator();
Spliterator<String> spliterator = Spliterators.spliteratorUnknownSize(iterator, 0);
Stream<String> stream = StreamSupport.stream(spliterator, false);
```

Reduction .

IntStream sum() . .



(((1+2)+3)+4)

reduce . reduce sum() .

```
IntStream istr;

//Initialize istr

OptionalInt istr.reduce((a,b)->a+b);
```

Optional .

`Stream<LinkedList<T>>  LinkedList<T>` .

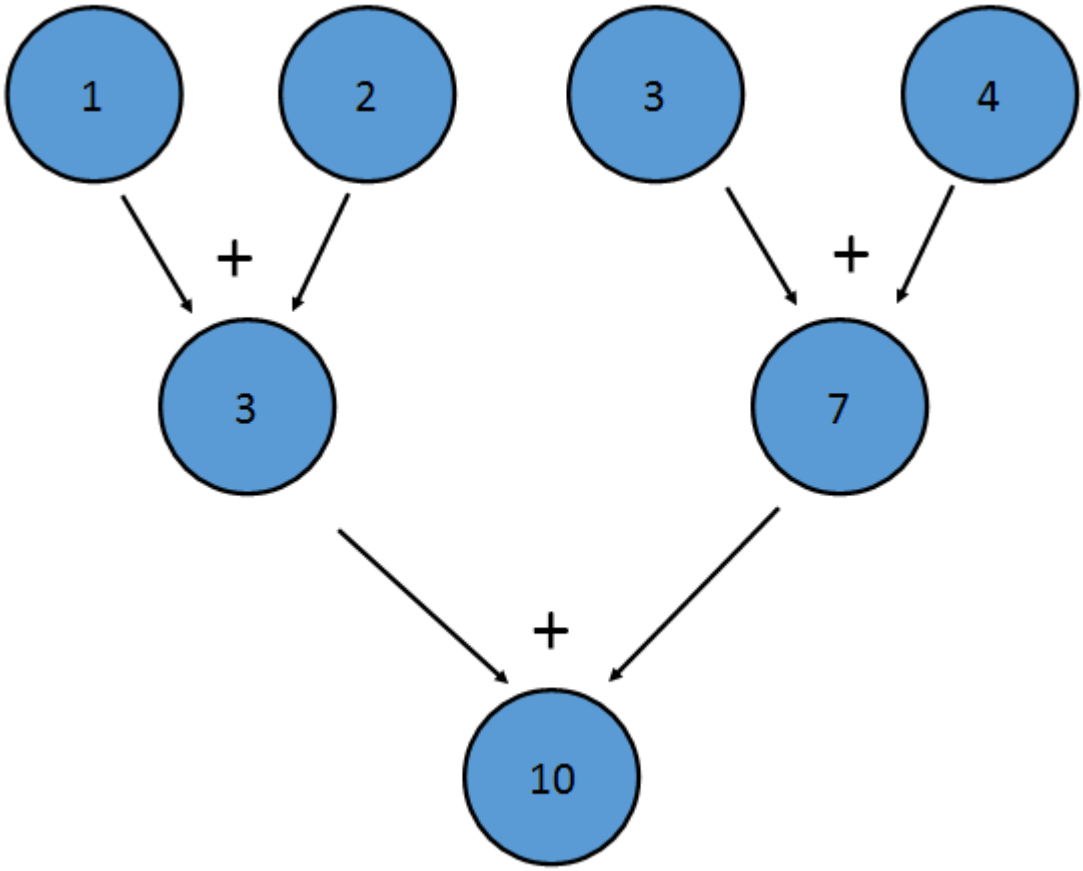
```
Stream<LinkedList<T>> listStream;

//Create a Stream<LinkedList<T>>

Optional<LinkedList<T>> bigList = listStream.reduce((LinkedList<T> list1, LinkedList<T>
list2)->{
    LinkedList<T> retList = new LinkedList<T>();
    retList.addAll(list1);
    retList.addAll(list2);
    return retList;
});
```

. , `x+0==x` `x+0==x` . , identity 1 `x*1==x` . , "" identity `LinkedList<T>` .

```
Stream<LinkedList<T>> listStream;

//Create a Stream<LinkedList<T>>

LinkedList<T> bigList = listStream.reduce(new LinkedList<T>(), (LinkedList<T> list1,
LinkedList<T> list2)->{
    LinkedList<T> retList = new LinkedList<T>();
    retList.addAll(list1);
    retList.addAll(list2);
    return retList;
});
```

identity `Optional` If . `reduce()` identity .

. , `(a+b)+c==a+(b+c)` . . , .



`((1+2)+(3+4))` . Java `Stream` . .

`String` . `Collectors.joining()` .

```
Stream<String> fruitStream = Stream.of("apple", "banana", "pear", "kiwi", "orange");

String result = fruitStream.filter(s -> s.contains("a"))
           .map(String::toUpperCase)
           .sorted()
           .collect(Collectors.joining(", "));

System.out.println(result);
```

:

APPLE, BANANA, ORANGE, PEAR

`Collectors.joining()` prefix postfix .

```
String result = fruitStream.filter(s -> s.contains("e"))
           .map(String::toUpperCase)
           .sorted()
           .collect(Collectors.joining(", ", "Fruits: ", "."));

System.out.println(result);
```

:

: APPLE, ORANGE, PEAR.

: https://riptutorial.com/ko/java/topic/88/

130:

. Singleton Design Patterns Singleton .

Examples

Java SE 5

```
public enum Singleton {
    INSTANCE;

    public void execute (String arg) {
        // Perform operation here
    }
}
```

Enum . .

JVM .

enum . .

Joshua Bloch Effective Java .

Singleton Singleton .

Java SE 5

```
public class MySingleton {

    // instance of class
    private static volatile MySingleton instance = null;

    // Private constructor
    private MySingleton() {
        // Some code for constructing object
    }

    public static MySingleton getInstance() {
        MySingleton result = instance;

        //If the instance already exists, no locking is necessary
        if(result == null) {
            //The singleton instance doesn't exist, lock and check again
            synchronized(MySingleton.class) {
                result = instance;
                if(result == null) {
                    instance = result = new MySingleton();
                }
            }
        }
        return result;
    }
}
```

SE 5 . Java 5 Java .

( )

```
public class Singleton {

    private static final Singleton INSTANCE = new Singleton();

    private Singleton() {}

    public static Singleton getInstance() {
        return INSTANCE;
    }
}
```

. Java 12.4.1,

T, .

- T T .
- T T .
- T .
- T .
- T T assert .

`getInstance()` `Singleton` .

thread-safe lazy | Bill Pugh Singleton

```
public class Singleton {
    private static class InstanceHolder {
        static final Singleton INSTANCE = new Singleton();
    }

    public static Singleton getInstance() {
        return InstanceHolder.INSTANCE;
    }

    private Singleton() {}
}
```

`Singleton.getInstance()` `INSTANCE` .

Bill Pugh . []

( )

`Singleton` `"Hello world!"` `getMessage()` `"Hello world!"` .

`UppercaseSingleton` `LowercaseSingleton` getMessage () .

```
//Yeah, we'll need reflection to pull this off.
```

```
import java.lang.reflect.*;

/*
Enumeration that represents possible classes of singleton instance.
If unknown, we'll go with base class - Singleton.
*/
enum SingletonKind {
    UNKNOWN,
    LOWERCASE,
    UPPERCASE
}

//Base class
class Singleton{

    /*
    Extended classes has to be private inner classes, to prevent extending them in
    uncontrolled manner.
     */
    private class UppercaseSingleton extends Singleton {

        private UppercaseSingleton(){
            super();
        }

        @Override
        public String getMessage() {
            return super.getMessage().toUpperCase();
        }
    }

    //Another extended class.
    private class LowercaseSingleton extends Singleton
    {
        private LowercaseSingleton(){
            super();
        }

        @Override
        public String getMessage() {
            return super.getMessage().toLowerCase();
        }
    }

    //Applying Singleton pattern
    private static SingletonKind kind = SingletonKind.UNKNOWN;

    private static Singleton instance;

    /*
    By using this method prior to getInstance() method, you effectively change the
    type of singleton instance to be created.
     */
    public static void setKind(SingletonKind kind) {
        Singleton.kind = kind;
    }

    /*
    If needed, getInstance() creates instance appropriate class, based on value of
    singletonKind field.
     */
```

```
    public static Singleton getInstance()
        throws  NoSuchMethodException,
                IllegalAccessException,
                InvocationTargetException,
                InstantiationException {

        if(instance==null){
            synchronized (Singleton.class){
                if(instance==null){
                    Singleton singleton = new Singleton();
                    switch (kind){
                        case UNKNOWN:

                            instance = singleton;
                            break;

                        case LOWERCASE:

                            /*
                             I can't use simple

                             instance = new LowercaseSingleton();

                             because java compiler won't allow me to use
                             constructor of inner class in static context,
                             so I use reflection API instead.

                             To be able to access inner class by reflection API,
                             I have to create instance of outer class first.
                             Therefore, in this implementation, Singleton cannot be
                             abstract class.
                             */

                            //Get the constructor of inner class.
                            Constructor<LowercaseSingleton> lcConstructor =

LowercaseSingleton.class.getDeclaredConstructor(Singleton.class);

                            //The constructor is private, so I have to make it accessible.
                            lcConstructor.setAccessible(true);

                            // Use the constructor to create instance.
                            instance = lcConstructor.newInstance(singleton);

                            break;

                        case UPPERCASE:

                            //Same goes here, just with different type
                            Constructor<UppercaseSingleton> ucConstructor =

UppercaseSingleton.class.getDeclaredConstructor(Singleton.class);
                            ucConstructor.setAccessible(true);
                            instance = ucConstructor.newInstance(singleton);
                    }
                }
            }
        }
        return instance;
    }
```

```
    //Singletons state that is to be used by subclasses
    protected String message;

    //Private constructor prevents external instantiation.
    private Singleton()
    {
        message = "Hello world!";
    }

    //Singleton's API. Implementation can be overwritten by subclasses.
    public String getMessage() {
        return message;
    }
}

//Just a small test program
public class ExtendingSingletonExample {

    public static void main(String args[]){

        //just uncomment one of following lines to change singleton class

        //Singleton.setKind(SingletonKind.UPPERCASE);
        //Singleton.setKind(SingletonKind.LOWERCASE);

        Singleton singleton = null;
        try {
            singleton = Singleton.getInstance();
        } catch (NoSuchMethodException e) {
            e.printStackTrace();
        } catch (IllegalAccessException e) {
            e.printStackTrace();
        } catch (InvocationTargetException e) {
            e.printStackTrace();
        } catch (InstantiationException e) {
            e.printStackTrace();
        }
        System.out.println(singleton.getMessage());
    }
}
```

: https://riptutorial.com/ko/java/topic/130/-

131:

Examples

equals ()

`equals` `EqualsBuilder` .

:

```
@Override
public boolean equals(Object obj) {

    if(!(obj instanceof MyClass)) {
        return false;
    }
    MyClass theOther = (MyClass) obj;

    EqualsBuilder builder = new EqualsBuilder();
    builder.append(field1, theOther.field1);
    builder.append(field2, theOther.field2);
    builder.append(field3, theOther.field3);

    return builder.isEquals();
}
```

:

```
@Override
public boolean equals(Object obj) {
    return EqualsBuilder.reflectionEquals(this, obj, false);
}
```

boolean , equals .

:

```
@Override
public boolean equals(Object obj) {
    return EqualsBuilder.reflectionEquals(this, obj, "field1", "field2");
}
```

hashCode ()

`hashCode` `HashCodeBuilder` .

:

```
@Override
public int hashCode() {
```

```
    HashCodeBuilder builder = new HashCodeBuilder();
    builder.append(field1);
    builder.append(field2);
    builder.append(field3);

    return builder.hashCode();
}
```

:

```
@Override
public int hashCode() {
    return HashCodeBuilder.reflectionHashCode(this, false);
}
```

.

:

```
@Override
public int hashCode() {
    return HashCodeBuilder.reflectionHashCode(this, "field1", "field2");
}
```

toString ()

toString ToStringBuilder .

:

```
@Override
public String toString() {

    ToStringBuilder builder = new ToStringBuilder(this);
    builder.append(field1);
    builder.append(field2);
    builder.append(field3);

    return builder.toString();
}
```

:

```
ar.com.jonat.lang.MyClass@dd7123[<null>,0,false]
```

:

```
@Override
public String toString() {

    ToStringBuilder builder = new ToStringBuilder(this);
    builder.append("field1",field1);
```

```
    builder.append("field2",field2);
    builder.append("field3",field3);

    return builder.toString();
}
```

:

```
ar.com.jonat.lang.MyClass@dd7404[field1=<null>,field2=0,field3=false]
```

.

```
@Override
public String toString() {

    ToStringBuilder builder = new ToStringBuilder(this,
            ToStringStyle.MULTI_LINE_STYLE);
    builder.append("field1", field1);
    builder.append("field2", field2);
    builder.append("field3", field3);

    return builder.toString();
}
```

:

```
ar.com.bna.lang.MyClass@ebbf5c[
  field1=<null>
  field2=0
  field3=false
]
```

JSON, Classname, short .

:

```
@Override
public String toString() {
    return ToStringBuilder.reflectionToString(this);
}
```

.

```
@Override
public String toString() {
    return ToStringBuilder.reflectionToString(this, ToStringStyle.JSON_STYLE);
}
```

: https://riptutorial.com/ko/java/topic/3338/--

132:

Java Java .

.

2016 . Plugin-Free Web .

API .

Java . . 2000 .

.

`init()`	.
`destroy()`	.
`start()`	.
`stop()`	.
`paint()`	`repaint()` .

Examples

.

```
public class MyApplet extends JApplet{

    private String str = "StackOverflow";

    @Override
    public void init() {
        setBackground(Color.gray);
    }
    @Override
    public void destroy() {}
    @Override
    public void start() {}
    @Override
    public void stop() {}
    @Override
    public void paint(Graphics g) {
        g.setColor(Color.yellow);
        g.fillRect(1,1,300,150);
        g.setColor(Color.red);
        g.setFont(new Font("TimesRoman", Font.PLAIN, 48));
        g.drawString(str, 10, 80);
    }
```

```
}
```

`javax.swing.JApplet` .

Java SE 1.2

Java 1.2 API `java.applet.Applet` .

main . . HTML . .

```
<html>
  <head></head>
  <body>
     <applet code="MyApplet.class" width="400" height="200"></applet>
  </body>
</html>
```

GUI

GUI . `Container` `awt` `swing` `add()` .

```
public class MyGUIApplet extends JApplet{

    private JPanel panel;
    private JButton button;
    private JComboBox<String> cmbBox;
    private JTextField textField;

    @Override
    public void init(){
        panel = new JPanel();
        button = new JButton("ClickMe!");
        button.addActionListener(new ActionListener(){
            @Override
            public void actionPerformed(ActionEvent ae) {
                if(((String)cmbBox.getSelectedItem()).equals("greet")) {
                    JOptionPane.showMessageDialog(null,"Hello " + textField.getText());
                } else {
                    JOptionPane.showMessageDialog(null,textField.getText() + " stinks!");
                }
            }
        });
        cmbBox = new JComboBox<>(new String[]{"greet", "offend"});
        textField = new JTextField("John Doe");
        panel.add(cmbBox);
        panel.add(textField);
        panel.add(button);
        add(panel);
    }
}
```

`getAppletContext()` `AppletContext` . `showDocument()` . `_blank` `_self` .

```
public class MyLinkApplet extends JApplet{
    @Override
```

```
    public void init(){
        JButton button = new JButton("ClickMe!");
        button.addActionListener(new ActionListener(){
            @Override
            public void actionPerformed(ActionEvent ae) {
                AppletContext a = getAppletContext();
                try {
                    URL url = new URL("http://stackoverflow.com/");
                    a.showDocument(url,"_blank");
                } catch (Exception e) { /* omitted for brevity */ }
            }
        });
        add(button);
    }
}
```

,

Java . . .

URL `getCodeBase()` URL . `getImage()` `getAudioClip()` .

```
public class MyImgApplet extends JApplet{

    private Image img;

    @Override
    public void init(){
        try {
            img = getImage(new URL("http://cdn.sstatic.net/stackexchange/img/logos/so/so-
logo.png"));
        } catch (MalformedURLException e) { /* omitted for brevity */ }
    }
    @Override
        public void paint(Graphics g) {
            g.drawImage(img, 0, 0, this);
        }
}
```

```
public class MyAudioApplet extends JApplet{

    private AudioClip audioClip;

    @Override
    public void init(){
        try {
            audioClip = getAudioClip(new URL("URL/TO/AN/AUDIO/FILE.WAV"));
        } catch (MalformedURLException e) { /* omitted for brevity */ }
    }
    @Override
    public void start() {
        audioClip.play();
    }
```

```
    @Override
    public void stop(){
        audioClip.stop();
    }
}
```

```
public class MyTextApplet extends JApplet{
    @Override
    public void init(){
        JTextArea textArea = new JTextArea();
        JScrollPane sp = new JScrollPane(textArea);
        add(sp);
        // load text
        try {
            URL url = new URL("http://www.textfiles.com/fun/quotes.txt");
            InputStream in = url.openStream();
            BufferedReader bf = new BufferedReader(new InputStreamReader(in));
            String line = "";
            while((line = bf.readLine()) != null) {
                textArea.append(line + "\n");
            }
        } catch(Exception e) { /* omitted for brevity */ }
    }
}
```

: https://riptutorial.com/ko/java/topic/5503/

133:

Examples

:-

- .
- .

: . GC .

Hashmap WeakHashMap : -

Java WeakHashMap .

:-

```
public class WeakHashMapTest {
    public static void main(String[] args) {
        Map hashMap= new HashMap();

        Map weakHashMap = new WeakHashMap();

        String keyHashMap = new String("keyHashMap");
        String keyWeakHashMap = new String("keyWeakHashMap");

        hashMap.put(keyHashMap, "Ankita");
        weakHashMap.put(keyWeakHashMap, "Atul");
        System.gc();
        System.out.println("Before: hash map value:"+hashMap.get("keyHashMap")+" and weak hash
map value:"+weakHashMap.get("keyWeakHashMap"));

        keyHashMap = null;
        keyWeakHashMap = null;

        System.gc();

        System.out.println("After: hash map value:"+hashMap.get("keyHashMap")+" and weak hash
map value:"+weakHashMap.get("keyWeakHashMap"));
    }
```

(HashMap WeakHashMap) :

HashMap size () - . size HashMap remove () .

() , WeakHashMap, thread . size . **WeakHashMap** .

: https://riptutorial.com/ko/java/topic/10749/-

134:

LinkedHashMap Map . HashMap Map .

Java LinkedHashMap . LinkedHashMap . . (null) . HashMap .

Examples

Java LinkedHashMap

:-

- Map .
- HashMap Map .
- .
- .
- (null) .
- HashMap .

: -

- void clear ().
- boolean containsKey (Object key)
- Object get (Object key).
- boolean removeEldestEntry (Map.Entry eldest)

:-

```
public static void main(String arg[])
    {
        LinkedHashMap<String, String> lhm = new LinkedHashMap<String, String>();
        lhm.put("Ramesh", "Intermediate");
        lhm.put("Shiva", "B-Tech");
        lhm.put("Santosh", "B-Com");
        lhm.put("Asha", "Msc");
        lhm.put("Raghu", "M-Tech");

        Set set = lhm.entrySet();
        Iterator i = set.iterator();
        while (i.hasNext()) {
            Map.Entry me = (Map.Entry) i.next();
            System.out.println(me.getKey() + " : " + me.getValue());
        }

        System.out.println("The Key Contains : " + lhm.containsKey("Shiva"));
        System.out.println("The value to the corresponding to key : " + lhm.get("Asha"));
```

```
    }
```

: https://riptutorial.com/ko/java/topic/10750/--

135:

Java , .

Java , (). ( ). .

Java 40 . .

- , ,
-
- ,

Examples

(+)

`+` Java .

- `+` Plus .
- . .
- `String` .

Concatenation . :

```
String s1 = "a String";
String s2 = "This is " + s1;    // s2 contains "This is a String"
```

`String` .

- boxed `toString()` .
- `toString()` . `null` `toString()` `null` `"null"` .

:

```
int one = 1;
String s3 = "One is "  + one;         // s3 contains "One is 1"
String s4 = null + " is null";        // s4 contains "null is null"
String s5 = "{1} is " + new int[]{1}; // s5 contains something like
                                      // "{} is [I@xxxxxxxx"
```

`s5` `toString()` `java.lang.Object` , ID .

Concatenation `String` . , ., .

```
String typing = "The quick brown fox " +
                "jumped over the " +
                "lazy dog";           // constant expression
```

String . .

```
public String stars(int count) {
    String res = "";
    for (int i = 0; i < count; i++) {
        res = res + "*";
    }
    return res;
}
```

String . String. stars(N) :

- N String .
- N * (N + 1) / 2
- O(N^2) .

N . . .

```
public String stars(int count) {
    // Create a string builder with capacity 'count'
    StringBuilder sb = new StringBuilder(count);
    for (int i = 0; i < count; i++) {
        sb.append("*");
    }
    return sb.toString();
}
```

StringBuilder . (: . O(N) O(N^2)).

. JLS Java . :

```
String s1 = ...;
String s2 = ...;
String test = "Hello " + s1 + ". Welcome to " + s2 + "\n";
```

.

```
StringBuilder tmp = new StringBuilder();
tmp.append("Hello ")
tmp.append(s1 == null ? "null" + s1);
tmp.append("Welcome to ");
tmp.append(s2 == null ? "null" + s2);
tmp.append("\n");
String test = tmp.toString();
```

(JIT s1 s2 null .) .

, :

- ( ) .
- .

(+, -, *, /, %)

Java 7 .

- `+` .
  - . (`+` .)
  - ( )
- `-` :
  - .
  - 0 .
- (*) .
- (/) .
- 2 [1] (%) .

"" . "Remainder" JLS . "" "" .

. `java.lang` (, `byte` , `short` , `char` , `int` , `long` , `float` `double`) . ie (`Byte` , `Character` , `Short` , `Integer` , `Long` , `Float` `Double` .

.

- `double` `Double` `double` .
- `float` `Float` `float` .
- , `long` `Long` `long` .
- `int` . `byte` , `short` , `char` `int.

- `double` `double` 64 ( ) IEE 754 .
- `float` `float` 32 (2) IEE 754 .
- `long` , `long` 64 2 2 .
- `int` `int` 32 2 2 .

.

- , *unboxed* .
- .
  - `int` `long` .
  - `double` `float` .
  - . IEE 768 " " .

/ `n` () `d` () `q` ().

Java 0 . JLS 15.17.2 Java .

`n d` `|d • q| ≤ |n|` `q . |d • q| ≤ |n| . , q |n| ≥ |d| n d` , `q |n| ≥ |d| n d` .

.

- `n` `MIN_VALUE` -1 `MIN_VALUE` . .
- `d` 0, ArithmeticException .

Java . `q d . q = n d . q = n .`

. , 0 INF NaN . .

C C ++ Java .

`, a % b r (a / b) * b + r , a / , * + . b` 0 `. , ArithmeticException .`

`a % b a a . , a % b b .`

`. a % b r r = a – (b • q) .` :

- `q ,`
- `a / b a / b`
- `a b .`

`b` 0 `INF NaN . . .`

:

`%` IEEE 754 . IEEE 754 `Math.IEEEremainder` .

Java 32 64 2 . , (32) `int` -2 ^{31} +2 ^{31} - 1 .

N (N == 32 64) , N . .

- 2 . N .
- 32 64 .

.

INF NAN

Java `float double` IEE 754 . .

- "" INF . `+INF` . `–INF` .
- ""/ " " NaN .

INF 0 .

NaN 0 0 0 .

INF NaN . :

- + INF + INF.
- + INF + INF + INF.
- + INF -INF NaN .
- INF +0.0 -0.0.
- NaN NaN .

JLS 15 . " " . `INF` `NaN` . .

(==,! =)

`==` `!=` `true` `false` . `==` `true` `false` . `!=` `false` `true` .

. JLS .

- `==` `!=` .
- `==` `!=` .
- `==` `!=` .

`==` `!=` `boolean` .

== !=

`==` `!=` () (`byte` , `short` , `char` , `int`, `long` , `float` `double`) . .

.

1. unbox.
2. `byte` , `short` `char` `int` .
3. " " " " .
4. :
    - `int` `long` .
    - `float` `double` :
        - 0 (`+0.0` `-0.0`)
        - `NaN` .
        - IEEE 754 .

: `==` `!=` .

== !=

`boolean` `boolean` `Boolean` `Boolean` `==` `!=` . .

1. `Boolean` unbox.
2.

		A == B	A! = B

`==` `!=` "".

- `==` `!=` `Boolean` . . Pitfall : == Integer

- == = . . boolean Boolean . Pitfall - '=='

== !=

== != . . .equals() .

```
String s1 = "We are equal";
String s2 = new String("We are equal");

s1.equals(s2); // true

// WARNING - don't use == or != with String values
s1 == s2;      // false
```

: == != String . http://www.ripting.com/java/example/16290/pitfall--using-to-compare-strings . . http://www.riptutorial.com/java/example/8996/pitfall--using-to-compare-primitive-wrappers-objects-such-as-integer .

NaN

JLS 15.21.1 :

NaN == false != true ., x != x true x NaN .

(:) . NaN "!". , == NaN .

Java "oddity" IEEE 754 . (http://stackoverflow.com/a/1573715/139985 ... " " !)

/ (++ / -)

++ -- 1 .

++ -- .

```
int a = 10;
a++; // a now equals 11
a--; // a now equals 10 again
```

++ -- .

```
int x = 10;
--x; // x now equals 9
++x; // x now equals 10
```

. .

```
int x=10;

System.out.println("x=" + x + " x=" + x++ + " x=" + x); // outputs x=10 x=10 x=11
System.out.println("x=" + x + " x=" + ++x + " x=" + x); // outputs x=11 x=12 x=12
```

```
System.out.println("x=" + x + " x=" + x-- + " x=" + x); // outputs x=12 x=12 x=11
System.out.println("x=" + x + " x=" + --x + " x=" + x); // outputs x=11 x=10 x=10
```

. in / decremented post-in / decrement . in / decrement . :

```
int x = 0;
x = x++ + 1 + x++;      // x = 0 + 1 + 1
                        // do not do this - the last increment has no effect (bug!)
System.out.println(x);  // prints 2 (not 3!)
```

:

```
int x = 0;
x = x++ + 1 + x;        // evaluates to x = 0 + 1 + 1
x++;                    // adds 1
System.out.println(x);  // prints 3
```

(? :)

{ - } **?** *{statement-executed-true}* **:** *{statement-executed-on-false}*

(Ternary Operator [1]) `?` () `:`) . `if-else` .

```
result = testCondition ? value1 : value2
```

```
if (testCondition) {
    result = value1;
} else {
    result = value2;
}
```

"testCondition true result value1 . result2 value2 . ".

:

```
// get absolute value using conditional operator
a = -10;
int absValue = a < 0 ? -a : a;
System.out.println("abs = " + absValue); // prints "abs = 10"
```

```
// get absolute value using if/else loop
a = -10;
int absValue;
if (a < 0) {
    absValue = -a;
} else {
    absValue = a;
}
System.out.println("abs = " + absValue); // prints "abs = 10"
```

((null check)) .

```
String x = y != null ? y.toString() : ""; //where y is an object
```

.

```
String x = "";

if (y != null) {
    x = y.toString();
}
```

.

```
// no parenthesis needed for expressions in the 3 parts
10 <= a && a < 19 ? b * 5 : b * 7

// parenthesis required
7 * (a > 0 ? 2 : 5)
```

. switch .

```
a ? "a is true" :
b ? "a is false, b is true" :
c ? "a and b are false, c is true" :
    "a, b, and c are false"

//Operator precedence can be illustrated with parenthesis:

a ? x : (b ? y : (c ? z : w))
```

:

1 - Java Java Tutorial (? : :) . Java () "".""C C ++ .

(~, &, |, ^)

Java 4 .

- (~) . JLS 15.15.5 . .
- AND (&) "and" . JLS 15.22.2 . .
- OR (|) 2 "" 2 . JLS 15.22.2 . .
- XOR (^) "" . JLS 15.22.2 . .

.

		~ A	A & B	A |	A ^ B
0	0	1	0	0	0
0	1	1	0	1	1

		~ A	A & B	A |	A ^ B
1	0	0	0	1	1
1	1	0	1	1	0

. 32 64 .

▪

.

~ .

& "" . :

```
int word = 0b00101010;
int mask = 0b00000011;   // Mask for masking out all but the bottom
                         // two bits of a word
int lowBits = word & mask;            // -> 0b00000010
int highBits = word & ~mask;          // -> 0b00101000
```

| . :

```
int word2 = 0b01011111;
// Combine the bottom 2 bits of word1 with the top 30 bits of word2
int combined = (word & mask) | (word2 & ~mask);   // -> 0b01011110
```

^ "" .

```
int word3 = 0b00101010;
int word4 = word3 ^ mask;             // -> 0b00101001
```

(Bit Manipulation) .

Instanceof

/ . **instanceof** .

```
( Object reference variable ) instanceof  (class/interface type)
```

:

```
public class Test {

   public static void main(String args[]){
      String name = "Buyya";
      // following will return true since name is type of String
      boolean result = name instanceof String;
```

```
      System.out.println( result );
   }
}
```

.

```
true
```

true .

:

```
class Vehicle {}

public class Car extends Vehicle {
   public static void main(String args[]){
      Vehicle a = new Car();
      boolean result =  a instanceof Car;
      System.out.println( result );
   }
}
```

.

```
true
```

(=, + =, - =, * =, / =, % =, << =, >> =, >>> =, & =, | = ^ =)

. . , , . , JLS 5.2 .

"operation and assign" JLS 15.26.2 .

```
E1 op= E2   E1 = (T) ((E1) op (E2)) . T E1    E1 .
```

.

1. =

: .

: `c = a + b` `a + b` `c` `c`

2. +=

"add and assign" : . `String` " ".

: `c += a` `c = c + a` `c = c + a`

3. -=

" ": .

: `c -= a  c = c - a  c = c - a`

4. `*=`

" ": . .

: `c *= a  c = c * a  c = c * a`

5. `/=`

" ": .

: `c /*= a  c = c / a  c = c / a`

6. `%=`

"modulus and assign" .

: `c %*= a  c = c % a  c = c % a`

7. `<<=`

"left shift and assign".

: `c <<= 2  c = c << 2  .`

`>>=`

" ".

: `c >>= 2  c = c >> 2  .`

9. `>>>=`

" ".

: `c >>>= 2  c = c >>> 2  .`

10. `&=`

" and ".

: `c &= 2  c = c & 2  .`

11. `|=`

" ".

: `c |= 2  c = c | 2  .c = c | 2`

12. `^=`

"bitwise exclusive or and assign".

: `c ^= 2` `c = c ^ 2` .

(&& ||)

Java `boolean` `boolean` . :

- `&&` - AND ,
- `||` - OR . `<left-expr> && <right-expr>` :

```
{
   boolean L = evaluate(<left-expr>);
   if (L) {
       return evaluate(<right-expr>);
   } else {
       // short-circuit the evaluation of the 2nd operand expression
       return false;
   }
}
```

`<left-expr> || <right-expr>` `<left-expr> || <right-expr>` .

```
{
   boolean L = evaluate(<left-expr>);
   if (!L) {
       return evaluate(<right-expr>);
   } else {
       // short-circuit the evaluation of the 2nd operand expression
       return true;
   }
}
```

`if` / `else if` .

- &&

`&&` . `Integer` 0 .

```
public boolean isZero(Integer value) {
    return value == 0;
}

public boolean isZero(Integer value) {
    return value != null && value == 0;
}
```

`value` `null` `NullPointerException` .

"" . `value != null && value == 0` `value != null` `value != null` `value != null` . `null` (, `true` `true`) `value == 0` . `null` `value == 0` (), `NullPointerException` .

- &&

&& .

```
public boolean verify(int value, boolean needPrime) {
    return !needPrime | isPrime(value);
}

public boolean verify(int value, boolean needPrime) {
    return !needPrime || isPrime(value);
}
```

| () isPrime . || . | | .

(<<, >> >>>)

Java 32 64 . .

- << . 0 .
- '>>' . . .
- '>>>' . 0 .

:

1. int long . (byte , short char .)
2. byte , char short int int .
3. . **mod** Modulus .
4. . Java "" .
5. 2 (2) .
6. a (2) 2 .

. (.)

1	2	<<	>>	>>>
0b0000000000001011	0	0b0000000000001011	0b0000000000001011	0b0000000000001011
0b0000000000001011	1	0b0000000000010110	0b0000000000000101	0b0000000000000101
0b0000000000001011	2	0b0000000000101100	0b0000000000000010	0b0000000000000010
0b0000000000001011	28	0b1011000000000000	0b0000000000000000	0b0000000000000000
0b0000000000001011	31	0b1000000000000000	0b0000000000000000	0b0000000000000000
0b0000000000001011	32	0b0000000000001011	0b0000000000001011	0b0000000000001011
...	...	...	...	...

1	2	<<	>>	>>>
0b1000000000001011	0	0b1000000000001011	0b1000000000001011	0b1000000000001011
0b1000000000001011	1	0b0000000000010110	0b1100000000000101	0b0100000000000101
0b1000000000001011	2	0b0000000000101100	0b1110000000000010	0b00100000000000100
0b1000000000001011	31	0b1000000000000000	0b1111111111111111	0b0000000000000001

.

(->)

Java 8 (`->`) . .

Java SE 8

```
  a -> a + 1               // a lambda that adds one to its argument
  a -> { return a + 1; }   // an equivalent lambda using a block.
```

, .

(. Lambda Expressions .)

(<, <=,>,> =)

`<` , `<=` , `>` `>=` . . , `a` `b` `byte` , `short` , `char` , `int` , `long` , `float` , `double` :

```
- `a < b` tests if the value of `a` is less than the value of `b`.
- `a <= b` tests if the value of `a` is less than or equal to the value of `b`.
- `a > b` tests if the value of `a` is greater than the value of `b`.
- `a >= b` tests if the value of `a` is greater than or equal to the value of `b`.
```

`boolean` .

. :

```
int i = 1;
long l = 2;
if (i < l) {
    System.out.println("i is smaller");
}
```

. :

```
Integer i = 1;   // 1 is autoboxed to an Integer
Integer j = 2;   // 2 is autoboxed to an Integer
if (i < j) {
    System.out.println("i is smaller");
}
```

.

1. unbox.
2. `byte` , `short` `char` `int` .
3. " " " " .
4. `int` , `long` , `float` `double` .

.

- .
- .

Java . , .

: https://riptutorial.com/ko/java/topic/176/

136:

Java EnumMap , enum Map . Enum AbstractMap .

java.util.EnumMap Parameters

K : . V : .

Examples

```
import java.util.*;
class Book {
int id;
String name,author,publisher;
int quantity;
public Book(int id, String name, String author, String publisher, int quantity) {
    this.id = id;
    this.name = name;
    this.author = author;
    this.publisher = publisher;
    this.quantity = quantity;
}
}
public class EnumMapExample {
// Creating enum
    public enum Key{
           One, Two, Three
           };
public static void main(String[] args) {
    EnumMap<Key, Book> map = new EnumMap<Key, Book>(Key.class);
    // Creating Books
    Book b1=new Book(101,"Let us C","Yashwant Kanetkar","BPB",8);
    Book b2=new Book(102,"Data Communications & Networking","Forouzan","Mc Graw Hill",4);
    Book b3=new Book(103,"Operating System","Galvin","Wiley",6);
    // Adding Books to Map
       map.put(Key.One, b1);
       map.put(Key.Two, b2);
       map.put(Key.Three, b3);
    // Traversing EnumMap
       for(Map.Entry<Key, Book> entry:map.entrySet()){
            Book b=entry.getValue();
            System.out.println(b.id+" "+b.name+" "+b.author+" "+b.publisher+" "+b.quantity);

        }
}
}
```

: https://riptutorial.com/ko/java/topic/10158/-

137:

enum (`enum`) `enum` .

- [public / protected / private] enum Enum_name {// enum .
- ENUM_CONSTANT_1 [, ENUM_CONSTANT_2 ...]; // enum . enum .
- ENUM_CONSTANT_1 (param) [, ENUM_CONSTANT_2 (param) ...]; // enum . .
- ENUM_CONSTANT_1 {...} [, ENUM_CONSTANT_2 {...} ...]; // enum . . .
- ENUM_CONSTANT.name () // enum String .
- ENUM_CONSTANT.ordinal () // . , 0 .
- Enum_name.values () // (Enum_name []) .
- Enum_name.valueOf ("ENUM_CONSTANT") // ENUM_CONSTANT.name () - enum .
- Enum.valueOf (Enum_name.class, "ENUM_CONSTANT") // : ENUM_CONSTANT.name () - enum .

`java.lang.Enum` enum . .

&

enum . .

Examples

enum

.

.

```
public enum Season {
    WINTER,
    SPRING,
    SUMMER,
    FALL
}
```

, .

Enum .

```
/**
 * This enum is declared in the Season.java file.
*/
public enum Season {
    WINTER,
    SPRING,
    SUMMER,
```

```
    FALL
}
```

.

```
 public class Day {

    private Season season;

    public String getSeason() {
        return season.name();
    }

    public void setSeason(String season) {
        this.season = Season.valueOf(season);
    }

    /**
     * This enum is declared inside the Day.java file and
     * cannot be accessed outside because it's declared as private.
     */
    private enum Season {
        WINTER,
        SPRING,
        SUMMER,
        FALL
    }

}
```

Enum .

```
public class Day {

    /**
     * Constructor
    */
    public Day() {
        // Illegal. Compilation error
        enum Season {
            WINTER,
            SPRING,
            SUMMER,
            FALL
        }
    }

    public void aSimpleMethod() {
        // Legal. You can declare a primitive (or an Object) inside a method. Compile!
        int primitiveInt = 42;

        // Illegal. Compilation error.
        enum Season {
            WINTER,
            SPRING,
            SUMMER,
            FALL
        }
```

```
        Season season = Season.SPRING;
    }

}
```

.

```
public enum Season {
    WINTER,
    WINTER, //Compile Time Error : Duplicate Constants
    SPRING,
    SUMMER,
    FALL
}
```

---

enum **public , static final .** static .

.

```
public static void display(Season s) {
    System.out.println(s.name());  // name() is a built-in method that gets the exact name of
the enum constant
}

display(Season.WINTER);  // Prints out "WINTER"
```

values() . .

```
Season[] seasons = Season.values();
```

:, .

.

```
public static void enumIterate() {
    for (Season s : Season.values()) {
        System.out.println(s.name());
    }
}
```

switch enum .

```
public static void enumSwitchExample(Season s) {
    switch(s) {
        case WINTER:
            System.out.println("It's pretty cold");
            break;
        case SPRING:
            System.out.println("It's warming up");
            break;
        case SUMMER:
```

```
            System.out.println("It's pretty hot");
            break;
        case FALL:
            System.out.println("It's cooling down");
            break;
    }
}
```

== .

```
Season.FALL == Season.WINTER    // false
Season.SPRING == Season.SPRING  // true
```

---

equals() .

```
Season.FALL.equals(Season.FALL); // true
Season.FALL.equals(Season.WINTER); // false
Season.FALL.equals("FALL"); // false and no compiler error
```

enum . enum .

```
public enum MutableExample {
    A,
    B;

    private int count = 0;

    public void increment() {
        count++;
    }

    public void print() {
        System.out.println("The count of " + name() + " is " + count);
    }
}

// Usage:
MutableExample.A.print();       // Outputs 0
MutableExample.A.increment();
MutableExample.A.print();       // Outputs 1 -- we've changed a field
MutableExample.B.print();       // Outputs 0 -- another instance remains unchanged
```

enum ., final . enum .

Enum Serializable Comparable .

```
public abstract class Enum<E extends Enum<E>>
extends Object
implements Comparable<E>, Serializable
```

enum public . private (enum package-private).

```
public enum Coin {
    PENNY(1), NICKEL(5), DIME(10), QUARTER(25); // usual names for US coins
    // note that the above parentheses and the constructor arguments match
    private int value;

    Coin(int value) {
        this.value = value;
    }

    public int getValue() {
        return value;
    }
}

int p = Coin.NICKEL.getValue(); // the int value will be 5
```

getter .

Enum class .

```
public class Coin<T extends Coin<T>> implements Comparable<T>, Serializable{
    public static final Coin PENNY = new Coin(1);
    public static final Coin NICKEL = new Coin(5);
    public static final Coin DIME = new Coin(10);
    public static final Coin QUARTER = new Coin(25);

    private int value;

    private Coin(int value){
        this.value = value;
    }

    public int getValue() {
        return value;
    }
}

int p = Coin.NICKEL.getValue(); // the int value will be 5
```

. .

final Enum .

```
public enum Coin {
    PENNY(1), NICKEL(5), DIME(10), QUARTER(25);

    private final int value;

    Coin(int value){
        this.value = value;
    }

    ...

}
```

. , enum :

```
public enum Coin {
    PENNY(1, true), NICKEL(5, false), DIME(10), QUARTER(25);

    private final int value;
    private final boolean isCopperColored;

    Coin(int value){
        this(value, false);
    }

    Coin(int value, boolean isCopperColored){
        this.value = value;
        this.isCopperColored = isCopperColored;
    }

    ...

}
```

: `Enum` () , enum `Serializable` .

. enum .

```
public enum Direction {
    NORTH, SOUTH, EAST, WEST;
}
```

.

```
public enum Direction {
    NORTH, SOUTH, EAST, WEST;

    public Direction getOpposite(){
        switch (this){
            case NORTH:
                return SOUTH;
            case SOUTH:
                return NORTH;
            case WEST:
                return EAST;
            case EAST:
                return WEST;
            default: //This will never happen
                return null;
        }
    }
}
```

.

```
public enum Direction {
    NORTH, SOUTH, EAST, WEST;
```

```
    private Direction opposite;

    public Direction getOpposite(){
        return opposite;
    }

    static {
        NORTH.opposite = SOUTH;
        SOUTH.opposite = NORTH;
        WEST.opposite = EAST;
        EAST.opposite = WEST;
    }
}
```

, opposite , Direction . (NORTH SOUTH), (NORTH(SOUTH), SOUTH(NORTH), EAST(WEST), WEST(EAST) opposite final).

enum String .

```
import java.util.function.Predicate;
import java.util.regex.Pattern;

enum RegEx implements Predicate<String> {
    UPPER("[A-Z]+"), LOWER("[a-z]+"), NUMERIC("[+-]?[0-9]+");

    private final Pattern pattern;

    private RegEx(final String pattern) {
        this.pattern = Pattern.compile(pattern);
    }

    @Override
    public boolean test(final String input) {
        return this.pattern.matcher(input).matches();
    }
}

public class Main {
    public static void main(String[] args) {
        System.out.println(RegEx.UPPER.test("ABC"));
        System.out.println(RegEx.LOWER.test("abc"));
        System.out.println(RegEx.NUMERIC.test("+111"));
    }
}
```

.

```
import java.util.function.Predicate;

enum Acceptor implements Predicate<String> {
    NULL {
        @Override
        public boolean test(String s) { return s == null; }
    },
    EMPTY {
        @Override
        public boolean test(String s) { return s.equals(""); }
    },
```

```
    NULL_OR_EMPTY {
        @Override
        public boolean test(String s) { return NULL.test(s) || EMPTY.test(s); }
    };
}

public class Main {
    public static void main(String[] args) {
        System.out.println(Acceptor.NULL.test(null));  // true
        System.out.println(Acceptor.EMPTY.test(""));   // true
        System.out.println(Acceptor.NULL_OR_EMPTY.test(" ")); // false
    }
}
```

"" `enum` , `class` Where anywere `enum` :

```
public interface ExtensibleEnum {
    String name();
}
```

, (`enum`) `enum` `enum` . ( ) `enum` API API `enum` "".

enum .

```
public enum DefaultValues implements ExtensibleEnum {
    VALUE_ONE, VALUE_TWO;
}
```

.

```
public enum ExtendedValues implements ExtensibleEnum {
    VALUE_THREE, VALUE_FOUR;
}
```

- `printEnum()` `enum` .

```
private void printEnum(ExtensibleEnum val) {
    System.out.println(val.name());
}

printEnum(DefaultValues.VALUE_ONE);    // VALUE_ONE
printEnum(DefaultValues.VALUE_TWO);    // VALUE_TWO
printEnum(ExtendedValues.VALUE_THREE); // VALUE_THREE
printEnum(ExtendedValues.VALUE_FOUR);  // VALUE_FOUR
```

: . enum . switch-on-enum `enum` .

`enum` .

```
enum Action {
    DODGE {
        public boolean execute(Player player) {
            return player.isAttacking();
        }
```

```
    },
    ATTACK {
        public boolean execute(Player player) {
            return player.hasWeapon();
        }
    },
    JUMP {
        public boolean execute(Player player) {
            return player.getCoordinates().equals(new Coordinates(0, 0));
        }
    };

    public abstract boolean execute(Player player);
}
```

.

/ .

enums

`enum` . `enum` javadoc .

```
/**
 * United States coins
 */
public enum Coins {

    /**
     * One-cent coin, commonly known as a penny,
     * is a unit of currency equaling one-hundredth
     * of a United States dollar
     */
    PENNY(1),

    /**
     * A nickel is a five-cent coin equaling
     * five-hundredth of a United States dollar
     */
    NICKEL(5),

    /**
     * The dime is a ten-cent coin refers to
     * one tenth of a United States dollar
     */
    DIME(10),

    /**
     * The quarter is a US coin worth 25 cents,
     * one-fourth of a United States dollar
     */
    QUARTER(25);

    private int value;

    Coins(int value){
        this.value = value;
    }
```

```
    public int getValue(){
        return value;
    }
}
```

enum `values()` . . . .

```
public enum Day {
    MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, SATURDAY, SUNDAY;

    /**
    * Print out all the values in this enum.
    */
    public static void printAllDays() {
        for(Day day : Day.values()) {
            System.out.println(day.name());
        }
    }
}
```

`Set EnumSet.allOf(Day.class)` .

generics java type enum . `Enum` .

```
public class Holder<T extends Enum<T>> {
    public final T value;

    public Holder(T init) {
        this.value = init;
    }
}
```

`T` .

▪

`DayOfWeek DayOfWeek` .

```
enum DayOfWeek {
    SUNDAY, MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, SATURDAY;
}
```

`valueOf()` . .

```
String dayName = DayOfWeek.SUNDAY.name();
assert dayName.equals("SUNDAY");

DayOfWeek day = DayOfWeek.valueOf(dayName);
assert day == DayOfWeek.SUNDAY;
```

enum .

```
Class<DayOfWeek> enumType = DayOfWeek.class;
DayOfWeek day = Enum.valueOf(enumType, "SUNDAY");
assert day == DayOfWeek.SUNDAY;
```

enum , `valueOf()` Methods `IllegalArgumentException` .

Guava Guava `Optional` `Enums.getIfPresent()` .

```
DayOfWeek defaultDay = DayOfWeek.SUNDAY;
DayOfWeek day = Enums.valueOf(DayOfWeek.class, "INVALID").or(defaultDay);
assert day == DayOfWeek.SUNDAY;
```

. Singleton .

```
public enum Attendant {

    INSTANCE;

    private Attendant() {
        // perform some initialization routine
    }

    public void sayHello() {
        System.out.println("Hello!");
    }
}


public class Main {

    public static void main(String... args) {
        Attendant.INSTANCE.sayHello();// instantiated at this point
    }
}
```

Joshua Bloch "Effective Java" . .

-
-
- out-of-the-box

.

()

`enum` , enum .

.

```
public enum Coin {
    PENNY(1), NICKEL(5), DIME(10), QUARTER(25);

    private final int value;
```

```
    Coin(int value){
        this.value = value;
    }

    public boolean isGreaterThan(Coin other){
        return this.value > other.value;
    }

}
```

Coin Enum .isGreaterThan enum .

```
Coin penny = Coin.PENNY;
Coin dime = Coin.DIME;

System.out.println(penny.isGreaterThan(dime)); // prints: false
System.out.println(dime.isGreaterThan(penny)); // prints: true
```

enum String .

String . .

.

```
public enum Fruit {
    APPLE, ORANGE, STRAWBERRY, BANANA, LEMON, GRAPE_FRUIT;
}
```

Fruit.APPLE ? "APPLE" Fruit.APPLE ?

---

name() name()

name() enum String enum String .

:

```
System.out.println(Fruit.BANANA.name());       // "BANANA"
System.out.println(Fruit.GRAPE_FRUIT.name()); // "GRAPE_FRUIT"
```

---

toString()

toString() name() .

, toString() (override toString() , String

`toString()` `.name()` . `toString()` .

:

```
System.out.println(Fruit.BANANA.toString());      // "BANANA"
System.out.println(Fruit.GRAPE_FRUIT.toString()); // "GRAPE_FRUIT"
```

```
System.out.println(Fruit.BANANA.toString());      // "Banana"
System.out.println(Fruit.GRAPE_FRUIT.toString()); // "Grape Fruit"
```

`enum` `enum` `enum` .

, John, Ben Luke `PianoClass` `enum` .

```
    enum PianoClass {
    JOHN, BEN, LUKE;
        public String getSex() {
            return "Male";
        }
        public String getLevel() {
            return "Beginner";
        }
    }
```

- - () () :

```
    enum PianoClass2 {
    JOHN, BEN, LUKE, RITA, TOM;
        public String getSex() {
            return "Male"; // issue, Rita is a female
        }
        public String getLevel() {
            return "Beginner"; // issue, Tom is an intermediate student
        }
    }
```

.

```
PianoClass2 tom = PianoClass2.TOM;
PianoClass2 rita = PianoClass2.RITA;
System.out.println(tom.getLevel()); // prints Beginner -> wrong Tom's not a beginner
System.out.println(rita.getSex()); // prints Male -> wrong Rita's not a male
```

`PianoClass2` Rita Tom .

```
enum PianoClass3 {
    JOHN, BEN, LUKE,
    RITA {
        @Override
        public String getSex() {
            return "Female";
        }
```

```
    },
    TOM {
        @Override
        public String getLevel() {
            return "Intermediate";
        }
    };
    public String getSex() {
        return "Male";
    }
    public String getLevel() {
        return "Beginner";
    }
}
```

Tom Rita .

```
PianoClass3 tom = PianoClass3.TOM;
PianoClass3 rita = PianoClass3.RITA;
System.out.println(tom.getLevel()); // prints Intermediate
System.out.println(rita.getSex()); // prints Female
```

. .

```
enum Friend {
    MAT("Male"),
    JOHN("Male"),
    JANE("Female");

    private String gender;

    Friend(String gender) {
        this.gender = gender;
    }

    public String getGender() {
        return this.gender;
    }
}
```

:

```
Friend mat = Friend.MAT;
Friend john = Friend.JOHN;
Friend jane = Friend.JANE;
System.out.println(mat.getGender());     // Male
System.out.println(john.getGender());    // Male
System.out.println(jane.getGender());    // Female
```

0 enum

```
enum Util {
    /* No instances */;

    public static int clamp(int min, int max, int i) {
```

```
        return Math.min(Math.max(i, min), max);
    }

    // other utility methods...
}
```

(1) `enum` (0) . () a ; .

enum private .

enum ., `NullPointerException` .

```
enum Example {
    ONE(1), TWO(2);

    static Map<String, Integer> integers = new HashMap<>();

    private Example(int value) {
        integers.put(this.name(), value);
    }
}
```

.

```
enum Example {
    ONE(1), TWO(2);

    static Map<String, Integer> integers;

    private Example(int value) {
        putValue(this.name(), value);
    }

    private static void putValue(String name, int value) {
        if (integers == null)
            integers = new HashMap<>();
        integers.put(name, value);
    }
}
```

.

```
enum Example {
    ONE(1), TWO(2);

    // after initialisisation integers is null!!
    static Map<String, Integer> integers = null;

    private Example(int value) {
        putValue(this.name(), value);
    }

    private static void putValue(String name, int value) {
        if (integers == null)
            integers = new HashMap<>();
        integers.put(name, value);
```

```
    }
    // !!this may lead to null poiner exception!!
    public int getValue(){
        return (Example.integers.get(this.name()));
    }
}
```

:

- enum .
  - putValue () .
- .
  - = null; // .

.

== ., .equals . .equals NullPointerException == check .equals .

```
enum Day {
    GOOD, AVERAGE, WORST;
}

public class Test {

    public static void main(String[] args) {
        Day day = null;

        if (day.equals(Day.GOOD)) {//NullPointerException!
            System.out.println("Good Day!");
        }

        if (day == Day.GOOD) {//Always use == to compare enum
            System.out.println("Good Day!");
        }

    }
}
```

EnumSet , , EnumSet .

- EnumSet#range :
- EnumSet#of : . of .
- EnumSet#complementOf : enum enum

```
enum Page {
   A1, A2, A3, A4, A5, A6, A7, A8, A9, A10
}

public class Test {

  public static void main(String[] args) {
      EnumSet<Page> range = EnumSet.range(Page.A1, Page.A5);
```

```
        if (range.contains(Page.A4)) {
            System.out.println("Range contains A4");
        }

        EnumSet<Page> of = EnumSet.of(Page.A1, Page.A5, Page.A3);

        if (of.contains(Page.A1)) {
            System.out.println("Of contains A1");
        }
    }
}
```

: https://riptutorial.com/ko/java/topic/155/-

138:

Throwable , throw try…catch try…catch try…catch .

- void someMethod () throws SomeException {} // . SomeException catch.
- {

```
someMethod(); //code that might throw an exception
```

}

- catch (SomeException e) {

```
 System.out.println("SomeException was thrown!"); //code that will run if certain
exception (SomeException) is thrown
```

}

- {

```
 //code that will always run, whether try block finishes or not
```

}

Examples

try-catch

try...catch . try try...catch...finally try-with-resources try-with-resources .

catch try-catch

.

```
try {
    doSomething();
} catch (SomeException e) {
    handle(e);
}
// next statement
```

try...catch .

- try .
- try throw try...catch .
- try Throw .
  -

SomeException .
- catch *catch* .
  - e .
  - catch .
  - throw throw .
  - try...catch .
- .

try-catch

try...catch catch . :

```
try {
    doSomething();
} catch (SomeException e) {
    handleOneWay(e)
} catch (SomeOtherException e) {
    handleAnotherWay(e);
}
// next statement
```

catch , . ( ) try...catch . *throw* catch .

catch "" . :

```
try {
    throw new RuntimeException("test");
} catch (Exception e) {
    System.out.println("Exception");
} catch (RuntimeException e) {
    System.out.println("RuntimeException");
}
```

"RuntimeException" "Exception" . RuntimeException Exception ( ) catch . ( ) catch .

catch catch . (Java catch .)

catch

Java SE 7

Java SE 7 catch . (|) . :

```
try {
    doSomething();
} catch (SomeException | SomeOtherException e) {
    handleSomeException(e);
}
```

. catch throw () .

. e . e , . catch e throw throw union . :

```
public void method() throws IOException, SQLException
    try {
        doSomething();
    } catch (IOException | SQLException e) {
        report(e);
        throw e;
    }
```

, IOException SQLException Exception . , report report(Exception) . throw IOException SQLException throw . method throws Exception throws IOException, SQLException . (: Pitfall - Throwable Throwable, Exception, Error RuntimeException .)

.

throw .

```
public void checkNumber(int number) throws IllegalArgumentException {
    if (number < 0) {
        throw new IllegalArgumentException("Number must be positive: " + number);
    }
}
```

3 . :

- new IllegalArgumentException(...) IllegalArgumentException .
- throw ... .

. .

. . . (Java .)

.

- checkNumber throws IllegalArgumentException . IllegalArgumentException . Java - . javadoc (good practice).
- throw . :

  ```
  throw new IllegalArgumentException("it is bad");
  return;
  ```

  return .

message cause cause . cause . .

. cause cause .

```
    public class AppErrorException extends RuntimeException {
```

```
        public AppErrorException() {
            super();
        }

        public AppErrorException(String message) {
            super(message);
        }

        public AppErrorException(String message, Throwable cause) {
            super(message, cause);
        }
    }
```

.

```
    public String readFirstLine(String file) throws AppErrorException {
        try (Reader r = new BufferedReader(new FileReader(file))) {
            String line = r.readLine();
            if (line != null) {
                return line;
            } else {
                throw new AppErrorException("File is empty: " + file);
            }
        } catch (IOException ex) {
            throw new AppErrorException("Cannot read file: " + file, ex);
        }
    }
```

`try` `throw` . `catch` `throw` `IOException` ( ) . . `cause` `IOException` .

`Exception` throw `Exception` . . , RuntimeException Exception . .

- UnsupportedOperationException -
- IllegalArgumentException -
- IllegalStateException - API API .

:

- API API API .
- throw .

`RuntimeException` *RuntimeException* `Exception` `Exception` . .

RuntimeException Exception .

```
public class StringTooLongException extends RuntimeException {
    // Exceptions can have methods and fields like other classes
    // those can be useful to communicate information to pieces of code catching
    // such an exception
    public final String value;
    public final int maximumLength;

    public StringTooLongException(String value, int maximumLength){
        super(String.format("String exceeds maximum Length of %s: %s", maximumLength, value));
        this.value = value;
```

```
        this.maximumLength = maximumLength;
    }
}
```

.

```
void validateString(String value){
    if (value.length() > 30){
        throw new StringTooLongException(value, 30);
    }
}
```

.

```
void anotherMethod(String value){
    try {
        validateString(value);
    } catch(StringTooLongException e){
        System.out.println("The string '" + e.value +
                "' was longer than the max of " + e.maximumLength );
    }
}
```

Oracle Java .

[...] . .

:

- RuntimeException ?

Java SE 7

try-catch-final `finally` " " . Java 7 *try-with-resources* .

?

Java 7 *try-with-resources* `java.lang.AutoCloseable` . `AutoCloseable` . .

`AutoCloseable` .

```
public void close() throws Exception
```

`close()` . . `Autocloseable` `close()` `Exception` throw throw .

Java `AutoCloseable` . .

- `InputStream` , `OutputStream`
- `Reader` , `Writer`
- `Socket` `ServerSocket`
- `Channel`

- JDBC `Connection`, `Statement` `ResultSet` .

3 .

try-with-resource

try-with-resources *try-catch*, *try-finally* *try-catch-finally* . "" . `catch` `finally`.

```
try (PrintStream stream = new PrintStream("hello.txt")) {
    stream.println("Hello world!");
}
```

`try` (...) . `stream` `PrintStream`.

`try` . `stream.close()` .`close()` .

try-with-resource

try-catch-finally *try-with-resources* `catch` `finally` . `PrintStream` throw `FileNotFoundException` `catch` .

```
try (PrintStream stream = new PrintStream("hello.txt")) {
    stream.println("Hello world!");
} catch (FileNotFoundException ex) {
    System.err.println("Cannot open the file");
} finally {
    System.err.println("All done");
}
```

try throw `catch` .`finally` *try-catch-finally* .

.

- `catch` `finally` .
- `catch` .
- throw `catch` `catch` .

., . :

```
try (InputStream is = new FileInputStream(file1);
     OutputStream os = new FileOutputStream(file2)) {
    // Copy 'is' to 'os'
}
```

.`try` `is` `os` . .

- .
- .
- .

, , is os is os .

try-with-resource classic try-catch-finally

Java *try-catch-finally* *try-with-resource* . (JLS .)

, *try-with-resource* :

```
try (PrintStream stream = new PrintStream("hello.txt")) {
    stream.println("Hello world!");
}
```

try-catch-finally .

```
// Note that the constructor is not part of the try-catch statement
PrintStream stream = new PrintStream("hello.txt");

// This variable is used to keep track of the primary exception thrown
// in the try statement. If an exception is thrown in the try block,
// any exception thrown by AutoCloseable.close() will be suppressed.
Throwable primaryException = null;

// The actual try block
try {
    stream.println("Hello world!");
} catch (Throwable t) {
    // If an exception is thrown, remember it for the finally block
    primaryException = t;
    throw t;
} finally {
    if (primaryException == null) {
        // If no exception was thrown so far, exceptions thrown in close() will
        // not be caught and therefore be passed on to the enclosing code.
        stream.close();
    } else {
        // If an exception has already been thrown, any exception thrown in
        // close() will be suppressed as it is likely to be related to the
        // previous exception. The suppressed exception can be retrieved
        // using primaryException.getSuppressed().
        try {
            stream.close();
        } catch (Throwable suppressedException) {
            primaryException.addSuppressed(suppressedException);
        }
    }
}
```

(JLS, `t primaryException` Java (`primaryException` .

try-with-resources . :

```
try (PrintStream stream = new PrintStream(fileName)) {
    stream.println("Hello world!");
} catch (NullPointerException ex) {
    System.err.println("Null filename");
} finally {
```

```
    System.err.println("All done");
}
```

.

```
try {
    try (PrintStream stream = new PrintStream(fileName)) {
        stream.println("Hello world!");
    }
} catch (NullPointerException ex) {
    System.err.println("Null filename");
} finally {
    System.err.println("All done");
}
```

(`new`) `Throwable` . . .

stacktrace `printStackTrace()` . :

```
try {
    int a = 0;
    int b = 0;
    int c = a / b;
} catch (ArithmeticException ex) {
    // This prints the stacktrace to standard output
    ex.printStackTrace();
}
```

`printStackTrace()` . `System.out` . `printStackTrace(PrintStream)` `printStackTrace(PrintWriter)` `Stream` `Writer` .

:

1. . `toString()` .

    ```
    // Print exception and stacktrace
    System.out.println(ex);
    ex.printStackTrace();
    ```

2. . Pitfall - . .

. (.)

```
File: "Main.java"
1   public class Main {
2       public static void main(String[] args) {
3           new Test().foo();
4       }
5   }

File: "Test.java"
1   class Test {
2       public void foo() {
3           bar();
```

```
4       }
5
6       public int bar() {
7           int a = 1;
8           int b = 0;
9           return a / b;
10      }
```

.

```
Exception in thread "main" java.lang.ArithmeticException: / by zero
        at Test.bar(Test.java:9)
        at Test.foo(Test.java:3)
        at Main.main(Main.java:3)
```

.

1 "main" . `java.lang.ArithmeticException` "/ by 0".

javadocs .

Throw. , "0 " .

, "/ by zero" 0 . ?

. ( ) .

- ,
- ,
-

. `Test.bar` Test.java 9 . :

```
    return a / b;
```

`b` , `b` 0 . .

, `bar()` `foo()` Test.java 3, `foo()` `Main.main()` .

: . .

- "OuterClass $ InnerClass" .
- "OuterClass $ 1", "OuterClass $ 2" .
- , "".
- "".

Java stacktrace .

Java SE 1.4

throw. .

```
File: Test,java
1   public class Test {
2      int foo() {
3           return 0 / 0;
4      }
5
6       public Test() {
7           try {
8               foo();
9           } catch (ArithmeticException ex) {
10              throw new RuntimeException("A bad thing happened", ex);
11          }
12      }
13
14      public static void main(String[] args) {
15          new Test();
16      }
17  }
```

.

```
Exception in thread "main" java.lang.RuntimeException: A bad thing happened
        at Test.<init>(Test.java:10)
        at Test.main(Test.java:15)
Caused by: java.lang.ArithmeticException: / by zero
        at Test.foo(Test.java:3)
        at Test.<init>(Test.java:8)
        ... 1 more
```

stacktrace ( ) , . `cause` " :". . N "... N more" .

" :" `cause` `null` . "" .

: `cause` Java 1.4.0 `Throwable` API . `printStackTrace` .

Java `String` . `printStackTrace(...)` `OutputStream` `Writer` .

Apache Commons Guava .

```
org.apache.commons.lang.exception.ExceptionUtils.getStackTrace(Throwable)

com.google.common.base.Throwables.getStackTraceAsString(Throwable)
```

.

```
   /**
     * Returns the string representation of the stack trace.
     *
     * @param throwable the throwable
     * @return the string.
     */
    public static String stackTraceToString(Throwable throwable) {
        StringWriter stringWriter = new StringWriter();
        throwable.printStackTrace(new PrintWriter(stringWriter));
        return stringWriter.toString();
    }
```

stacktrace `getStackTrace()` `getCause()` .

InterruptedException

`InterruptedException` . `Thread.sleep()` .

`InterruptedException` `Thread.interrupt()` ."! !" .

```
// Bad. Don't do this.
try {
  Thread.sleep(1000);
} catch (InterruptedException e) {
  // disregard
}
```

"" . `InterruptedException` .

"" .

```
// When nothing will interrupt your code
try {
  Thread.sleep(1000);
} catch (InterruptedException e) {
  Thread.currentThread().interrupt();
  throw new AssertionError(e);
}
```

. `InterruptedException` throw invariant `AssertionError` throw. `catch` .

Guava `Uninterruptibles` . `Uninterruptibles.sleepUninterruptibly()` `Uninterruptibles.sleepUninterruptibly()` , thread . , `InterruptedException` Throw . try-catch .

. , `Executor` . .

`InterruptedException` . `throws InterruptedException` . kludgy . .

```
// Let the caller determine how to handle the interrupt if you're unsure
public void myLongRunningMethod() throws InterruptedException {
  ...
}
```

(: `throw`) . . .

```
// Suppresses the exception but resets the interrupted state letting later code
// detect the interrupt and handle it properly.
try {
  Thread.sleep(1000);
} catch (InterruptedException e) {
  Thread.currentThread().interrupt();
  return ...; // your expectations are still broken at this point - try not to do more work.
}
```

Java Exception Hierarchy -

Java Exception . .

- `java.lang.Throwable` - . .
  - : `java.lang.Exception` - .
    - .
    - `java.lang.RuntimeException` - .
      -
  - `java.lang.Error` - " " .

:

1. .
2. `Throwable` , `Exception` `RuntimeException` `abstract` . Pitfall - Throwable Throwable, Exception, Error RuntimeException .
3. `Error` JVM .
4. `Throwable` . Java `Error` `Exception` `Throwable` .

. .

Java . Java .

- . , `IOException` I / O . .
- . .

("throw " `throw` throw , . "propagated" a throw . catch . .)

.

- , , `throws` . (`throws` .)
- .
- . .

, .

. throw .

. .

```
// This declares a custom checked exception.
public class MyException extends Exception {
    // constructors omitted.
}

// This declares a custom unchecked exception.
public class MyException2 extends RuntimeException {
    // constructors omitted.
}
```

throw "throw " .

```
// INCORRECT
public void methodThrowingCheckedException(boolean flag) {
    int i = 1 / 0;                // Compiles OK, throws ArithmeticException
    if (flag) {
        throw new MyException();  // Compilation error
    } else {
        throw new MyException2(); // Compiles OK
    }
}

// CORRECTED
public void methodThrowingCheckedException(boolean flag) throws MyException {
    int i = 1 / 0;                // Compiles OK, throws ArithmeticException
    if (flag) {
        throw new MyException();  // Compilation error
    } else {
        throw new MyException2(); // Compiles OK
    }
}
```

.

```
// INCORRECT
public void methodWithPropagatedCheckedException() {
    InputStream is = new FileInputStream("someFile.txt");  // Compilation error
    // FileInputStream throws IOException or a subclass if the file cannot
    // be opened.  IOException is a checked exception.
    ...
}

// CORRECTED (Version A)
public void methodWithPropagatedCheckedException() throws IOException {
    InputStream is = new FileInputStream("someFile.txt");
    ...
}

// CORRECTED (Version B)
public void methodWithPropagatedCheckedException() {
    try {
        InputStream is = new FileInputStream("someFile.txt");
        ...
    } catch (IOException ex) {
        System.out.println("Cannot open file: " + ex.getMessage());
    }
}
```

.

```
// INCORRECT
public class Test {
    private static final InputStream is =
            new FileInputStream("someFile.txt");  // Compilation error
}

// CORRECTED
public class Test {
```

```
    private static final InputStream is;
    static {
        InputStream tmp = null;
        try {
            tmp = new FileInputStream("someFile.txt");
        } catch (IOException ex) {
            System.out.println("Cannot open file: " + ex.getMessage());
        }
        is = tmp;
    }
}
```

, `is` , , .

. Java .

```
class Division {
    public static void main(String[] args) {

        int a, b, result;

        Scanner input = new Scanner(System.in);
        System.out.println("Input two integers");

        a = input.nextInt();
        b = input.nextInt();

        result = a / b;

        System.out.println("Result = " + result);
    }
}
```

0 .

```
Input two integers
7 0
Exception in thread "main" java.lang.ArithmeticException: / by zero
    at Division.main(Disivion.java:14)
```

0 . . *ArithmeticException* .

: .

. `b == 0` .

```
class Division {
    public static void main(String[] args) {

        int a, b, result;

        Scanner input = new Scanner(System.in);
        System.out.println("Input two integers");

        a = input.nextInt();
        b = input.nextInt();
```

```
        if (b == 0) {
            System.out.println("You cannot divide by zero.");
            return;
        }

        result = a / b;

        System.out.println("Result = " + result);
    }
}
```

`You cannot divide by zero.` `You cannot divide by zero.` 0 . `if` `try-catch` .

```
...

a = input.nextInt();
b = input.nextInt();

try {
    result = a / b;
}
catch (ArithmeticException e) {
    System.out.println("An ArithmeticException occurred. Perhaps you tried to divide by
zero.");
    return;
}

...
```

try catch .

1. `try` .
2. try `catch` (, `ArithmeticException`).
3. `e` `catch` .
4. `try` `catch` (, catch) `try-catch` .

. , `null` , . . *throw* .

Java Pitfalls - .

try catch

return .

```
public static int returnTest(int number){
   try{
       if(number%2 == 0) throw new Exception("Exception thrown");
       else return x;
   }
   catch(Exception e){
       return 3;
   }
   finally{
       return 7;
```

```
    }
}
```

try / catch finally 7 . , `return 7;` `return 7;` try / catch .

catch finally catch finally .

"1" "0" .

```
public class FinallyExample {

    public static void main(String[] args) {
        int n = returnTest(4);

        System.out.println(n);
    }

    public static int returnTest(int number) {

        int returnNumber = 0;

        try {
            if (number % 2 == 0)
                throw new Exception("Exception thrown");
            else
                return returnNumber;
        } catch (Exception e) {
            return returnNumber;
        } finally {
            returnNumber = 1;
        }
    }
}
```

.

Java SE 1.4

. . , `SecurityManager` .

.

```
    Exception ex = new Exception();   // this captures the call stack
    StackTraceElement[] frames = ex.getStackTrace();
    System.out.println("This method is " + frames[0].getMethodName());
    System.out.println("Called from method " + frames[1].getMethodName());
```

.

1. `StackTraceElement` . `printStackTrace` . .
2. `getStackTrace()` Javadoc JVM .

. , throwable , .

. . : . .

`Throwable Throwable.fillInStackTrace()` . `public`., (override) . `Throwable` .

```
  public class MyException extends Exception {
      // constructors

      @Override
      public void fillInStackTrace() {
          // do nothing
      }
  }
```

`fillInStackTrace()` .

Java SE 1.4

. `Throwable.setStackTrace` `StackTraceElement` .

.

```
 exception.setStackTrace(new StackTraceElement[0]);
```

Java SE 7

Java 7 *try-with-resources* . .

```
try (Writer w = new BufferedWriter(new FileWriter(someFilename))) {
    // do stuff
    int temp = 0 / 0;    // throws an ArithmeticException
}
```

`try w close()` `FileWriter` . `IOException` ?

throw . . *try-with-resources* . .

`try` . Throw .

`getSuppressedExceptions` .

try-finally try-catch-finally

`try...catch...finally` . `finally` . .

Try-finally

(`try...finally`) .

```
try {
    doSomething();
```

```
} finally {
    cleanUp();
}
```

`try...finally` .

- `try` .
- `try` :
    - `finally` .
    - `finally` throw .
    - `try...finally` .
- try throw :
    - `finally` .
    - `finally` throw .
    - .

`finally` . (`System.exit(int)` JVM . `finally` . .

try-catch-finally

`catch finally` . .

```
// This code snippet writes the first line of a file to a string
String result = null;
Reader reader = null;
try {
    reader = new BufferedReader(new FileReader(fileName));
    result = reader.readLine();
} catch (IOException ex) {
    Logger.getLogger.warn("Unexpected IO error", ex);  // logging the exception
} finally {
    if (reader != null) {
        try {
            reader.close();
        } catch (IOException ex) {
            // ignore / discard this exception
        }
    }
}
```

`try...catch...finally` () . `finally` .

.

- `try` "resource"(, `reader`) `finally` .
- `new FileReader(...)` `catch` throw `IOError` .
- `try` `catch` `finally` `reader.close()` .
- `reader` `null` .
- , `reader.close()` ( ) throw . .

Java SE 7

Java 7 try-with-resources .

'throws'

Java throw . `throws` . :

```
public class OddNumberException extends Exception { // a checked exception
}

public void checkEven(int number) throws OddNumberException {
    if (number % 2 != 0) {
        throw new OddNumberException();
    }
}
```

`throws OddNumberException checkEven OddNumberException` throw `OddNumberException.`

`throws` .

```
public void checkEven(Double number)
        throws OddNumberException, ArithmeticException {
    if (!Double.isFinite(number)) {
        throw new ArithmeticException("INF or NaN");
    } else if (number % 2 != 0) {
        throw new OddNumberException();
    }
}
```

?

`throws` .

1. catch ( ) .
2. . , `throws` .

: `throws` , API javadoc , IDE 「hover text」 .

Throw

`throws` . (override) , (override) . override . :

```
@Override
public void checkEven(int number) throws NullPointerException // OK—NullPointerException is an
unchecked exception
    ...

@Override
public void checkEven(Double number) throws OddNumberException // OK—identical to the
superclass
    ...
```

```
class PrimeNumberException extends OddNumberException {}
class NonEvenNumberException extends OddNumberException {}

@Override
public void checkEven(int number) throws PrimeNumberException, NonEvenNumberException //
OK—these are both subclasses

@Override
public void checkEven(Double number) throws IOExcepion         // ERROR
```

throw throw .

: https://riptutorial.com/ko/java/topic/89/---

139:

javax.sound.sampled `Clip` API `AudioClip` . `AudioClip` `Clip` .

Examples

:

```
import javax.sound.sampled.AudioSystem;
import javax.sound.sampled.Clip;
```

.

```
Clip clip = AudioSystem.getClip();
clip.open(AudioSystem.getAudioInputStream(new URL(filename)));
clip.start();
clip.loop(Clip.LOOP_CONTINUOUSLY);
```

:

```
AudioFileFormat.Type [] audioFileTypes = AudioSystem.getAudioFileTypes();
```

MIDI

MIDI `javax.sound.midi` . `Sequencer` , ,, .

MIDI :

```
import java.io.File;
import java.io.IOException;
import javax.sound.midi.InvalidMidiDataException;
import javax.sound.midi.MidiSystem;
import javax.sound.midi.MidiUnavailableException;
import javax.sound.midi.Sequence;
import javax.sound.midi.Sequencer;

public class MidiPlayback {
    public static void main(String[] args) {
        try {
            Sequencer sequencer = MidiSystem.getSequencer(); // Get the default Sequencer
            if (sequencer==null) {
                System.err.println("Sequencer device not supported");
                return;
            }
            sequencer.open(); // Open device
            // Create sequence, the File must contain MIDI file data.
            Sequence sequence = MidiSystem.getSequence(new File(args[0]));
            sequencer.setSequence(sequence); // load it into sequencer
            sequencer.start();  // start the playback
        } catch (MidiUnavailableException | InvalidMidiDataException | IOException ex) {
            ex.printStackTrace();
```

```
        }
    }
}
```

.

```
sequencer.stop(); // Stop the playback
```

. .

```
import javax.sound.midi.Track;
// ...

Track[] track = sequence.getTracks();
sequencer.setTrackMute(track[0]);
```

. .

```
sequencer.setLoopCount(3);
sequencer.setLoopCount(Sequencer.LOOP_CONTINUOUSLY);
```

. . .

```
sequencer.setLoopStartPoint(512);
sequencer.setLoopEndPoint(32768);
sequencer.setTickPosition(8192);
```

MIDI . (BPM) 4 (MPQ) . .

```
sequencer.setTempoInBPM(1250f);
sequencer.setTempoInMPQ(4750f);
sequencer.setTempoFactor(1.5f);
```

Sequencer .

```
sequencer.close();
```

Java . OS OS . . .

. .

.

```
public void rectangleWave(byte volume, int hertz, int msecs) {
    final SourceDataLine dataLine;
    // 24 kHz x 8bit, single-channel, signed little endian AudioFormat
    AudioFormat af = new AudioFormat(24_000, 8, 1, true, false);
    try {
        dataLine = AudioSystem.getSourceDataLine(af);
        dataLine.open(af, 10_000); // audio buffer size: 10k samples
    } catch (LineUnavailableException e) {
```

```
        throw new RuntimeException(e);
    }

    int waveHalf = 24_000 / hertz; // samples for half a period
    byte[] buffer = new byte[waveHalf * 20];
    int samples = msecs * (24_000 / 1000); // 24k (samples / sec) / 1000 (ms/sec) * time(ms)

    dataLine.start(); // starts playback
    int sign = 1;

    for (int i = 0; i < samples; i += buffer.length) {
        for (int j = 0; j < 20; j++) { // generate 10 waves into buffer
            sign *= -1;
            // fill from the jth wave-half to the j+1th wave-half with volume
            Arrays.fill(buffer, waveHalf * j, waveHalf * (j+1), (byte) (volume * sign));
        }
        dataLine.write(buffer, 0, buffer.length); //
    }
    dataLine.drain(); // forces buffer drain to hardware
    dataLine.stop();  // ends playback
}
```

. .

! Java . .wav mp3 .

```
import java.io.*;
import java.net.URL;
import javax.sound.sampled.*;

public class SoundClipTest {

   // Constructor
   public SoundClipTest() {
      try {
         // Open an audio input stream.
          File soundFile = new File("/usr/share/sounds/alsa/Front_Center.wav"); //you could
also get the sound file with an URL
          AudioInputStream audioIn = AudioSystem.getAudioInputStream(soundFile);
          AudioFormat format = audioIn.getFormat();
         // Get a sound clip resource.
         DataLine.Info info = new DataLine.Info(Clip.class, format);
         Clip clip = (Clip)AudioSystem.getLine(info);
         // Open audio clip and load samples from the audio input stream.
         clip.open(audioIn);
         clip.start();
      } catch (UnsupportedAudioFileException e) {
         e.printStackTrace();
      } catch (IOException e) {
         e.printStackTrace();
      } catch (LineUnavailableException e) {
         e.printStackTrace();
      }
   }

   public static void main(String[] args) {
      new SoundClipTest();
   }
}
```

: https://riptutorial.com/ko/java/topic/160/

140:

Java Java . , , , , , , , , .

- . .
- .
- , (Java) (Java).
- . . . . .

Examples

- .
- . / .
- . API `java.lang.annotation` `java.lang.annotations` .
- : `com.yourcompany.widget.button` , `com.yourcompany.core.api`

,

- .
- ... able .
- (: CamelCase) .
- `^[AZ][a-zA-Z0-9]*$` .
- .
- .
- : `ArrayList` , `BigInteger` , `ArrayIndexOutOfBoundsException` , `Iterable` .

.

- `^[az][a-zA-Z0-9]*$` .
- .
- : `toString` , `hashCode`

.

- `^[az][a-zA-Z0-9]*$` .
- :
- : `elements` , `currentIndex`

.

- `^[AZ][0-9]?$` .
- (: `K V` , `R`), `T` .
- (_) .
- : `T` , `V` , `SRC_VERTEX`

`static final` (_) .

- `^[AZ][A-Z0-9]*(_[A-Z0-9]+)*$` .
- : `BUFFER_SIZE` , `MAX_LEVEL`

- , .
- . , .
- public static .
- : ( )

:

Java

- CR CR + LF (LF, ASCII 10) .
- .
- `.java` . `package-info.java` .
- LF (ASCII 32) . (:) .
- (`\047`) (: `\u0027`) `\'` , `\"` , `\\` , `\t` , `\b` , `\r` , `\f` `\n` .
- , .

```
package com.example.my.package;
```

.

```
// First java/javax packages
import java.util.ArrayList;
import javax.tools.JavaCompiler;

// Then third party libraries
import com.fasterxml.jackson.annotation.JsonProperty;

// Then project imports
import com.example.my.package.ClassA;
import com.example.my.package.ClassB;

// Then static imports (in the same order as above)
```

```
import static java.util.stream.Collectors.toList;
```

- ...
    - ... / .
    - ...
        -
        - javax
        - (: org.xml)
        - (: com.sun)
    - ... 3
- .
- .

- .
- (: "") .
- .

.

1. (,)
2.
3.
4. (,)

.

.

```
class Example {

    private int i;

    Example(int i) {
        this.i = i;
    }

    static Example getExample(int i) {
        return new Example(i);
    }

    @Override
    public String toString() {
        return "An example [" + i + "]";
    }

}
```

- .
-

. .
- . .
- .
- .

```
class ExampleClass {
    // Access modifiers first (don't do for instance "static public")
    public static void main(String[] args) {
        System.out.println("Hello World");
    }
}

interface ExampleInterface {
    // Avoid 'public' and 'abstract' since they are implicit
    void sayHello();
}
```

- .
    - (`public` / `private` / `protected`)
    - `abstract`
    - `static`
    - `final`
    - `transient`
    - `volatile`
    - `default`
    - `synchronized`
    - `native`
    - `strictfp`
- . `public abstract` enum static .
- `final` .
- `final` .
- **4** .
- . **.**
- . (.)
- `case` 4 case 4 .

```
switch (var) {
    case TWO:
        setChoice("two");
        break;
    case THREE:
        setChoice("three");
        break;
    default:
        throw new IllegalArgumentException();
}
```

.

- 80 100 .

  . "" . . , " " 80 . .

- URL .

```
// Ok even though it might exceed max line width when indented.
Error e = isTypeParam
        ? Errors.InvalidRepeatableAnnotationNotApplicable(targetContainerType, on)
        : Errors.InvalidRepeatableAnnotationNotApplicableInContext(targetContainerType));

// Wrapping preferable
String pretty = Stream.of(args)
                      .map(Argument::prettyPrint)
                      .collectors(joining(", "));

// Too strict interpretation of max line width. Readability suffers.
Error e = isTypeParam
        ? Errors.InvalidRepeatableAnnotationNotApplicable(
                targetContainerType, on)
        : Errors.InvalidRepeatableAnnotationNotApplicableInContext(
                targetContainerType);

// Should be wrapped even though it fits within the character limit
String pretty = Stream.of(args).map(Argument::prettyPrint).collectors(joining(", "));
```

- .
- .
- .
  - **1** : 8 .
  - **2** : .
  - **3** : ( )
  - **4** : .

```
int someMethod(String aString,
               List<Integer> aList,
               Map<String, String> aMap,
               int anInt,
               long aLong,
               Set<Number> aSet,
               double aDouble) {
    …
}

int someMethod(String aString, List<Integer> aList,
        Map<String, String> aMap, int anInt, long aLong,
        double aDouble, long aLong) {
    …
}

int someMethod(String aString,
               List<Map<Integer, StringBuffer>> aListOfMaps,
               Map<String, String> aMap)
```

```
        throws IllegalArgumentException {
    …
}

int someMethod(String aString, List<Integer> aList,
        Map<String, String> aMap, int anInt)
                throws IllegalArgumentException {
    …
}
```

- +8
- throws throws +8 +8 .

- / .
- .
- . .

```
popupMsg("Inbox notification: You have "
        + newMsgs + " new messages");

// Don't! Looks like two arguments
popupMsg("Inbox notification: You have " +
         newMsgs + " new messages");
```

- ...
    -
    -
    -
    -
    -
    -
- ... .
    -
    -
    -
- .

- ...
    -
    - lambda for (:) 2 .
    - // .
    - for .
    - .
- .

- ( )
- (`String args[]`) (`String[] args`).
- .

.

```
@SuppressWarnings("unchecked")
public T[] toArray(T[] typeHolder) {
    ...
}
```

. .

```
@Nullable String getName() { return name; }
```

.

```
// Bad.
@Deprecated @SafeVarargs
@CustomAnnotation
public final Tuple<T> extend(T... elements) {
    ...
}

// Even worse.
@Deprecated @SafeVarargs
@CustomAnnotation public final Tuple<T> extend(T... elements) {
    ...
}

// Good.
@Deprecated
@SafeVarargs
@CustomAnnotation
public final Tuple<T> extend(T... elements) {
    ...
}

// Good.
@Deprecated @SafeVarargs @CustomAnnotation
public final Tuple<T> extend(T... elements) {
    ...
}
```

```
Runnable r = () -> System.out.println("Hello World");

Supplier<String> c = () -> "Hello World";

// Collection::contains is a simple unary method and its behavior is
// clear from the context. A method reference is preferred here.
appendFilter(goodStrings::contains);

// A lambda expression is easier to understand than just tempMap::put in this case
trackTemperature((time, temp) -> tempMap.put(time, temp));
```

-

.

- .
- 1 arity .
- .
- .

```
return flag ? "yes" : "no";

String cmp = (flag1 != flag2) ? "not equal" : "equal";

// Don't do this
return (flag ? "yes" : "no");
```

- (:) .
- . .
- return return .

```
long l = 5432L;
int i = 0x123 + 0xABC;
byte b = 0b1010;
float f1 = 1 / 5432f;
float f2 = 0.123e4f;
double d1 = 1 / 5432d;  // or 1 / 5432.0
double d2 = 0x1.3p2;
```

- long L .
- 16 A - F .
- , .

```
class Example {
    void method(boolean error) {
        if (error) {
            Log.error("Error occurred!");
            System.out.println("Error!");
        } else { // Use braces since the other block uses braces.
            System.out.println("No error");
        }
    }
}
```

- .
- ( ).
- if loop .
  - .
  - if / else if .
  - .
- do…while else , catch while .

```
enum Response { YES, NO, MAYBE }
public boolean isReference() { return true; }
```

(/). " " . enum .

: https://riptutorial.com/ko/java/topic/2697/---

141:

Autoboxing Java . , int -> Integer, double -> Double ... unboxing. boxing primitive Object .

Autoboxing .

- http://docs.oracle.com/javase/1.5.0/docs/guide/language/autoboxing.html
- ?

Examples

int Integer .

.

```
List<Integer> ints = new ArrayList<Integer>();
```

Java SE 7

```
List<Integer> ints = new ArrayList<>();
```

, Integer int .

```
for (int i = 0; i < 10; i++)
    ints.add(i);
```

ints.add(i); .

```
ints.add(Integer.valueOf(i));
```

JVM Integer Integer#valueOf .

:

- byte Byte
- short Short
- float Float
- double Double
- long Long
- char Character
- boolean Boolean

. .

```
List<Integer> ints = new ArrayList<Integer>();
```

```
ints.add(1);
ints.add(2);
ints.add(3);
ints.remove(1); // ints is now [1, 3]
```

java.util.List remove(int index) (List) remove(Object o) (java.util.Collection java.util.Collection) . remove(int index) .

autoboxing Java -128 127 :

```
Integer a = 127;
Integer b = 127;
Integer c = 128;
Integer d = 128;
System.out.println(a == b); // true
System.out.println(c <= d); // true
System.out.println(c >= d); // true
System.out.println(c == d); // false
```

>= intValue() int == int .

Java [-128, 127] == . Integers . -XX:AutoBoxCacheMax JVM . -XX:AutoBoxCacheMax=1000 true true .

```
Integer a = 1000;
Integer b = 1000;
System.out.println(a == b); // true
```

if

if Boolean .

```
Boolean a = Boolean.TRUE;
if (a) { // a gets converted to boolean
    System.out.println("It works!");
}
```

for do while while .

Boolean null NullPointerException .

NullPointerException .

.

```
Integer arg = null;
int x = arg;
```

java.lang.NullPointerException .

int null .

. NullPointerException .

autoboxing auto-unboxing unboxed null .

Autoboxing

Autoboxing . :

```
Map<Integer, Integer> square = new HashMap<Integer, Integer>();
for(int i = 256; i < 1024; i++) {
    square.put(i, i * i); // Autoboxing of large integers
}
```

(6k 60kb).

CPU . 5 , HashMap , 2. Entry[] table , 3. Entry , 4. entrys key (boxing), 5. entrys value (boxing the primitive value).

```
class Example {
  int primitive; // Stored directly in the class `Example`
  Integer boxed; // Reference to another memory location
}
```

boxed primitive .

```
int sumOfSquares = 0;
for(int i = 256; i < 1024; i++) {
    sumOfSquares += square.get(i);
}
```

.

```
int sumOfSquares = 0;
for(int i = 256; i < 1024; i++) {
    sumOfSquares += square.get(Integer.valueOf(i)).intValue();
}
```

Map#get(Integer) Integer . .

. 4 . Java .

Java 8 IntStream .

: Java valueOf autoboxing Integer . Integer -128 +127. JVM / JVM .

Integer int

1 : .

, .

:

```
int i;
Integer j;
void ex_method(Integer i)//Is a valid statement
void ex_method1(int j)//Is a valid statement
```

2 : :

.

:

```
int i;
Integer j;
int ex_method()
{...
return j;}//Is a valid statement
Integer ex_method1()
{...
return i;//Is a valid statement
}
```

3 : .

.

```
int i=5;
Integer j=new Integer(7);
int k=i+j;//Is a valid statement
Integer m=i+j;//Is also a valid statement
```

Pitfall : .

.

:

```
public class Test{
    Integer i;
    int j;
    public void met()
    {j=i;//Null pointer exception
    SOP(j);
    SOP(i);}
    public static void main(String[] args)
    {Test t=new Test();
    t.go();//Null pointer exception
    }
```

null . .

: https://riptutorial.com/ko/java/topic/138/-

142:

CompletableFuture Java SE 8 Future Java SE 8 . Future .

Examples

1-2 . . .

```
public static long blockingGetWebPageLength(String urlString) {
    try (BufferedReader br = new BufferedReader(new InputStreamReader(new
URL(urlString).openConnection().getInputStream()))) {
        StringBuilder sb = new StringBuilder();
        String line;
        while ((line = br.readLine()) != null) {
            sb.append(line);
        }
        return sb.toString().length();
    } catch (IOException ex) {
        throw new RuntimeException(ex);
    }
}
```

. supplyAsync . .

```
static private ExecutorService service = Executors.newCachedThreadPool();

static public CompletableFuture<Long> asyncGetWebPageLength(String url) {
    return CompletableFuture.supplyAsync(() -> blockingGetWebPageLength(url), service);
}
```

thenAccept lamda . .

```
public static void main(String[] args) {

    asyncGetWebPageLength("https://stackoverflow.com/")
            .thenAccept(l -> {
                System.out.println("Stack Overflow returned " + l);
            })
            .exceptionally((Throwable throwable) -> {
                Logger.getLogger("myclass").log(Level.SEVERE, "", throwable);
                return null;
            });

}
```

CompletableFuture

`calculateShippingPrice` . , .

`CompletableFuture` `CompletableFuture` (,).

```
public static void main(String[] args) {
    int price = 15; // Let's keep it simple and work with whole number prices here
    int weightInGrams = 900;

    calculateShippingPrice(weightInGrams) // Here, we get the future
        .thenAccept(shippingPrice -> { // And then immediately work on it!
            // This fluent style is very useful for keeping it concise
            System.out.println("Your total price is: " + (price + shippingPrice));
        });
    System.out.println("Please stand by. We are calculating your total price.");
}


public static CompletableFuture<Integer> calculateShippingPrice(int weightInGrams) {
    return CompletableFuture.supplyAsync(() -> {
        // supplyAsync is a factory method that turns a given
        // Supplier<U> into a CompletableFuture<U>

        // Let's just say each 200 grams is a new dollar on your shipping costs
        int shippingCosts = weightInGrams / 200;

        try {
            Thread.sleep(2000L); // Now let's simulate some waiting time...
        } catch(InterruptedException e) { /* We can safely ignore that */ }

        return shippingCosts; // And send the costs back!
    });
}
```

: https://riptutorial.com/ko/java/topic/10935/---

143: (RMI)

RMI , . - . . ("") .
. - - . "" . .

Remote . .

1. Remote UnicastRemoteObject , .
2. UnicastRemoteObject UnicastRemoteObject#exportObject UnicastRemoteObject#exportObject .
3. , Registry Naming . , RMI , .

. .

Examples

- : JVM

:

```
package remote;

import java.rmi.Remote;
import java.rmi.RemoteException;

public interface RemoteServer extends Remote {

    int stringToInt(String string) throws RemoteException;
}
```

:

```
package server;

import java.rmi.RemoteException;
import java.rmi.registry.LocateRegistry;
import java.rmi.registry.Registry;
import java.rmi.server.UnicastRemoteObject;

import remote.RemoteServer;

public class Server implements RemoteServer {

    @Override
    public int stringToInt(String string) throws RemoteException {

        System.out.println("Server received: \"" + string + "\"");
        return Integer.parseInt(string);
    }

    public static void main(String[] args) {

        try {
```

```
            Registry reg = LocateRegistry.createRegistry(Registry.REGISTRY_PORT);
            Server server = new Server();
            UnicastRemoteObject.exportObject(server, Registry.REGISTRY_PORT);
            reg.rebind("ServerName", server);
        } catch (RemoteException e) {
            e.printStackTrace();
        }
    }
}
```

() :

```
package client;

import java.rmi.NotBoundException;
import java.rmi.RemoteException;
import java.rmi.registry.LocateRegistry;
import java.rmi.registry.Registry;

import remote.RemoteServer;

public class Client {

    static RemoteServer server;

    public static void main(String[] args) {

        try {
            Registry reg = LocateRegistry.getRegistry();
            server = (RemoteServer) reg.lookup("ServerName");
        } catch (RemoteException | NotBoundException e) {
            e.printStackTrace();
        }

        Client client = new Client();
        client.callServer();
    }

    void callServer() {

        try {
            int i = server.stringToInt("120");
            System.out.println("Client received: " + i);
        } catch (RemoteException e) {
            e.printStackTrace();
        }
    }
}
```

:

: "120"
: 120

: ""

. . . . 10 . ( ) .

1. .
2. `login` .:
    - . .
    - ("connection" "session") . `logout` ( ) .
3. `int passInt` .
4. `int half` . ( ) .

:

```
package callbackRemote;

import java.rmi.Remote;
import java.rmi.RemoteException;

public interface RemoteLogin extends Remote {

    RemoteConnection login(String name, RemoteClient client) throws RemoteException;
}
```

:

```
package callbackRemote;

import java.rmi.Remote;
import java.rmi.RemoteException;

public interface RemoteConnection extends Remote {

    void logout() throws RemoteException;

    String passInt(String name, int i) throws RemoteException;
}
```

:

```
package callbackRemote;

import java.rmi.Remote;
import java.rmi.RemoteException;

public interface RemoteClient extends Remote {

    void half(int i) throws RemoteException;
}
```

:

```
package callbackServer;

import java.rmi.RemoteException;
import java.rmi.registry.LocateRegistry;
import java.rmi.registry.Registry;
import java.rmi.server.UnicastRemoteObject;
import java.util.HashMap;
```

```
import java.util.Map;

import callbackRemote.RemoteClient;
import callbackRemote.RemoteConnection;
import callbackRemote.RemoteLogin;

public class LoginServer implements RemoteLogin {

    static Map<String, RemoteClient> clients = new HashMap<>();

    @Override
    public RemoteConnection login(String name, RemoteClient client) {

        Connection connection = new Connection(name, client);
        clients.put(name, client);
        System.out.println(name + " logged in");
        return connection;
    }

    public static void main(String[] args) {

        try {
            Registry reg = LocateRegistry.createRegistry(Registry.REGISTRY_PORT);
            LoginServer server = new LoginServer();
            UnicastRemoteObject.exportObject(server, Registry.REGISTRY_PORT);
            reg.rebind("LoginServerName", server);
        } catch (RemoteException e) {
            e.printStackTrace();
        }
    }
}
```

:

```
package callbackServer;

import java.rmi.NoSuchObjectException;
import java.rmi.RemoteException;
import java.rmi.registry.Registry;
import java.rmi.server.UnicastRemoteObject;
import java.rmi.server.Unreferenced;

import callbackRemote.RemoteClient;
import callbackRemote.RemoteConnection;

public class Connection implements RemoteConnection, Unreferenced {

    RemoteClient client;
    String name;

    public Connection(String name, RemoteClient client) {

        this.client = client;
        this.name = name;
        try {
            UnicastRemoteObject.exportObject(this, Registry.REGISTRY_PORT);
        } catch (RemoteException e) {
            e.printStackTrace();
        }
    }
```

```
    @Override
    public void unreferenced() {

        try {
            UnicastRemoteObject.unexportObject(this, true);
        } catch (NoSuchObjectException e) {
            e.printStackTrace();
        }
    }

    @Override
    public void logout() {

        try {
            UnicastRemoteObject.unexportObject(this, true);
        } catch (NoSuchObjectException e) {
            e.printStackTrace();
        }
    }

    @Override
    public String passInt(String recipient, int i) {

        System.out.println("Server received from " + name + ":" + i);
        if (i < 10)
            return String.valueOf(i);
        RemoteClient client = LoginServer.clients.get(recipient);
        try {
            client.half(i);
        } catch (RemoteException e) {
            e.printStackTrace();
        }
        return String.valueOf(i);
    }
}
```

:

```
package callbackClient;

import java.rmi.NotBoundException;
import java.rmi.RemoteException;
import java.rmi.registry.LocateRegistry;
import java.rmi.registry.Registry;
import java.rmi.server.UnicastRemoteObject;

import callbackRemote.RemoteClient;
import callbackRemote.RemoteConnection;
import callbackRemote.RemoteLogin;

public class Client implements RemoteClient {

    RemoteConnection connection;
    String name, target;

    Client(String name, String target) {

        this.name = name;
        this.target = target;
```

```
    }

    public static void main(String[] args) {

        Client client = new Client(args[0], args[1]);
        try {
            Registry reg = LocateRegistry.getRegistry();
            RemoteLogin login = (RemoteLogin) reg.lookup("LoginServerName");
            UnicastRemoteObject.exportObject(client, Integer.parseInt(args[2]));
            client.connection = login.login(client.name, client);
        } catch (RemoteException | NotBoundException e) {
            e.printStackTrace();
        }

        if ("Client1".equals(client.name)) {
            try {
                client.connection.passInt(client.target, 120);
            } catch (RemoteException e) {
                e.printStackTrace();
            }
        }
    }

    @Override
    public void half(int i) throws RemoteException {

        String result = connection.passInt(target, i / 2);
        System.out.println(name + " received: \"" + result + "\"");
    }
}
```

:

1. .
2. `Client2 Client1 1097` .
3. `Client1 Client2 1098` .

3 JVM 3 . .

Client2
Client1
1 : 120
Client2 : 60
Client1 : 30
Client2 : 15
Client1 : 7
Client1 : "7"
Client2 : "15"
Client1 : "30"
Client2 : "60"

RMI

5 2 *server client* RMI .

PersonListInterface.java

```
public interface PersonListInterface extends Remote
{
    /**
     * This interface is used by both client and server
     * @return List of Persons
     * @throws RemoteException
     */
    ArrayList<String> getPersonList() throws RemoteException;
}
```

PersonListImplementation.java

```
public class PersonListImplementation
extends UnicastRemoteObject
implements PersonListInterface
{

    private static final long serialVersionUID = 1L;

    // standard constructor needs to be available
    public PersonListImplementation() throws RemoteException
    {}

    /**
     * Implementation of "PersonListInterface"
     * @throws RemoteException
     */
    @Override
    public ArrayList<String> getPersonList() throws RemoteException
    {
        ArrayList<String> personList = new ArrayList<String>();

        personList.add("Peter Pan");
        personList.add("Pippi Langstrumpf");
        // add your name here :)

        return personList;
    }
}
```

Server.java

```
public class Server {

    /**
     * Register servicer to the known public methods
     */
    private static void createServer() {
        try {
            // Register registry with standard port 1099
            LocateRegistry.createRegistry(Registry.REGISTRY_PORT);
            System.out.println("Server : Registry created.");

            // Register PersonList to registry
            Naming.rebind("PersonList", new PersonListImplementation());
            System.out.println("Server : PersonList registered");
```

```
        } catch (final IOException e) {
            e.printStackTrace();
        }
    }

    public static void main(final String[] args) {
        createServer();
    }
}
```

PersonListLocal.java

```
public class PersonListLocal {
    private static PersonListLocal instance;
    private PersonListInterface personList;

    /**
     * Create a singleton instance
     */
    private PersonListLocal() {
        try {
            // Lookup to the local running server with port 1099
            final Registry registry = LocateRegistry.getRegistry("localhost",
                    Registry.REGISTRY_PORT);

            // Lookup to the registered "PersonList"
            personList = (PersonListInterface) registry.lookup("PersonList");
        } catch (final RemoteException e) {
            e.printStackTrace();
        } catch (final NotBoundException e) {
            e.printStackTrace();
        }
    }

    public static PersonListLocal getInstance() {
        if (instance == null) {
            instance = new PersonListLocal();
        }

        return instance;
    }

    /**
     * Returns the servers PersonList
     */
    public ArrayList<String> getPersonList() {
        if (instance != null) {
            try {
                return personList.getPersonList();
            } catch (final RemoteException e) {
                e.printStackTrace();
            }
        }

        return new ArrayList<>();
    }
       }
```

PersonTest.java

```
public class PersonTest
{
    public static void main(String[] args)
    {
        // get (local) PersonList
        ArrayList<String> personList = PersonListLocal.getInstance().getPersonList();

        // print all persons
        for(String person : personList)
        {
            System.out.println(person);
        }
    }
}
```

- Server.java main . :

```
Server : Registry created.
Server : PersonList registered
```

- PersonTest.java main . :

```
Peter Pan
Pippi Langstrumpf
```

(RMI) : https://riptutorial.com/ko/java/topic/171/----rmi-

144:

`byte` , `short` , `int` , `long` , `char` , `boolean` , `float` `double` 8 Java .

- int aInt = 8; // int () .
- int hexInt = 0x1a; // = 26; 16 *0x* .
- int binInt = 0b11010; // = 26; . *0b* .
- long goodLong = 10000000000L; // int . L . " " .
- double aDouble = 3.14; // double .
- float aFloat = 3.14F; // double " " . F float .

Java `boolean` , `byte` , `short` , `char` , `int` , `long` , `float` `double` 8 . (. `String` , `Class` , `Throwable` `Class` / .)

(, ,) `int` `short` (A) `short` `int` , `byte` A `byte` `char` A `char` . , .

```
byte a = 1;
byte b = 2;
byte c = (byte) (a + b);
```

.

Java §2.11.1 .

> `int` Java Virtual Machine `byte` `short` . `boolean` `char` `int` . [..]. `boolean` , `byte` , `char` `short` `int` .

.

> **1** , . Java `byte` . [...] .

Examples

int

`int` `Integer` .

Java API : "Integer int . Integer int ."

`int` 32 . 1 - -2 (31) 2 (31) .

```
int example = -42;
int myInt = 284;
int anotherInt = 73;
```

```
int addedInts = myInt + anotherInt; // 284 + 73 = 357
int subtractedInts = myInt - anotherInt; // 284 - 73 = 211
```

`long` . `int` (,). `((value - MIN_VALUE) % RANGE) + MIN_VALUE` `((value + 2147483648) % 4294967296) - 2147483648`

```
int demo = 2147483647; //maximum positive integer
System.out.println(demo); //prints 2147483647
demo = demo + 1; //leads to an integer overflow
System.out.println(demo); // prints -2147483648
```

`int` .

```
int high = Integer.MAX_VALUE;    // high == 2147483647
int low = Integer.MIN_VALUE;     // low == -2147483648
```

`int` *0.*

```
int defaultInt;    // defaultInt == 0
```

`short` 16 . -2^{15} (-32,768) $2^{15}-1$ (32,767) .

```
short example = -48;
short myShort = 987;
short anotherShort = 17;

short addedShorts = (short) (myShort + anotherShort); // 1,004
short subtractedShorts = (short) (myShort - anotherShort); // 970
```

`short` .

```
short high = Short.MAX_VALUE;        // high == 32767
short low = Short.MIN_VALUE;         // low == -32768
```

`short` *0.*

```
short defaultShort;    // defaultShort == 0
```

`long` 64 (Java 8). , $-2^{(63)}$ $2^{(63)}-1$ 0 2^{64} (2) 1 -

```
long example = -42;
long myLong = 284;
long anotherLong = 73;

//an "L" must be appended to the end of the number, because by default,
//numbers are assumed to be the int type. Appending an "L" makes it a long
//as 549755813888 (2 ^ 39) is larger than the maximum value of an int (2^31 - 1),
//"L" must be appended
long bigNumber = 549755813888L;

long addedLongs = myLong + anotherLong; // 284 + 73 = 357
```

```
long subtractedLongs = myLong - anotherLong; // 284 - 73 = 211
```

`long` .

```
long high = Long.MAX_VALUE;    // high == 9223372036854775807L
long low = Long.MIN_VALUE;     // low == -9223372036854775808L
```

a `long` *0L*

```
long defaultLong;    // defaultLong == 0L
```

: `long` "L" . .

```
2L == 2l;            // true
```

: Java Integer -128 127 .
https://blogs.oracle.com/darcy/entry/boxing_and_caches_integer_valueof .

.

```
Long val1 = 127L;
Long val2 = 127L;

System.out.println(val1 == val2); // true

Long val3 = 128L;
Long val4 = 128L;

System.out.println(val3 == val4); // false
```

2 Object Long (Java 1.7).

```
Long val3 = 128L;
Long val4 = 128L;

System.out.println(Objects.equal(val3, val4)); // true
```

long Object long (), 2 == does false negative .

`boolean true false` .

```
boolean foo = true;
System.out.println("foo = " + foo);                // foo = true

boolean bar = false;
System.out.println("bar = " + bar);                // bar = false

boolean notFoo = !foo;
System.out.println("notFoo = " + notFoo);          // notFoo = false

boolean fooAndBar = foo && bar;
System.out.println("fooAndBar = " + fooAndBar);    // fooAndBar = false
```

```
boolean fooOrBar = foo || bar;
System.out.println("fooOrBar = " + fooOrBar);      // fooOrBar = true

boolean fooXorBar = foo ^ bar;
System.out.println("fooXorBar = " + fooXorBar);    // fooXorBar = true
```

`boolean` *false.*

```
boolean defaultBoolean;    // defaultBoolean == false
```

byte

`byte` 8 . (127) 1 - 7 -2 (-128) , (2) (7)

```
byte example = -36;
byte myByte = 96;
byte anotherByte = 7;

byte addedBytes = (byte) (myByte + anotherByte); // 103
byte subtractedBytes = (byte) (myBytes - anotherByte); // 89
```

`byte` .

```
byte high = Byte.MAX_VALUE;        // high == 127
byte low = Byte.MIN_VALUE;         // low == -128
```

`byte` *0.*

```
byte defaultByte;    // defaultByte == 0
```

float

`float` 32 IEEE 754 . double . `float` `f` .

```
double doubleExample = 0.5;      // without 'f' after digits = double
float floatExample = 0.5f;       // with 'f' after digits    = float

float myFloat = 92.7f;           // this is a float...
float positiveFloat = 89.3f;     // it can be positive,
float negativeFloat = -89.3f;    // or negative
float integerFloat = 43.0f;      // it can be a whole number (not an int)
float underZeroFloat = 0.0549f;  // it can be a fractional value less than 0
```

5 , , , .

: . (, 34.600002).

```
// addition
float result = 37.2f + -2.6f;  // result: 34.6
```

```
// subtraction
float result = 45.1f - 10.3f;    // result: 34.8

// multiplication
float result = 26.3f * 1.7f;   // result: 44.71

// division
float result = 37.1f / 4.8f;   // result: 7.729166

// modulus
float result = 37.1f % 4.8f;   // result: 3.4999971
```

(:) .

```
float notExact = 3.1415926f;
System.out.println(notExact); // 3.1415925
```

`float` `float` ( ) `float` `double` . BigDecimal .

`float` *0.0f* .

```
float defaultFloat;    // defaultFloat == 0.0f
```

`float` 1 1 .

: `Float.POSITIVE_INFINITY` , `Float.NEGATIVE_INFINITY` , `Float.NaN` `float` . NaN 2 . `0f` `-0f` == true.

```
float f1 = 0f;
float f2 = -0f;
System.out.println(f1 == f2); // true
System.out.println(1f / f1); // Infinity
System.out.println(1f / f2); // -Infinity
System.out.println(Float.POSITIVE_INFINITY / Float.POSITIVE_INFINITY); // NaN
```

`double` 64 IEEE 754 .

```
double example = -7162.37;
double myDouble = 974.21;
double anotherDouble = 658.7;

double addedDoubles = myDouble + anotherDouble; // 315.51
double subtractedDoubles = myDouble - anotherDouble; // 1632.91

double scientificNotationDouble = 1.2e-3;    // 0.0012
```

.

```
double notExact = 1.32 - 0.42; // result should be 0.9
System.out.println(notExact); // 0.9000000000000001
```

`double` `float` `double` . BigDecimal .

`double` *0.0d.*

```
public double defaultDouble;    // defaultDouble == 0.0
```

: `Double.POSITIVE_INFINITY` , `Double.NEGATIVE_INFINITY` , `Double.NaN` `double` . `NaN` 2 . `0d` `-0d` == true.

```
double d1 = 0d;
double d2 = -0d;
System.out.println(d1 == d2); // true
System.out.println(1d / d1); // Infinity
System.out.println(1d / d2); // -Infinity
System.out.println(Double.POSITIVE_INFINITY / Double.POSITIVE_INFINITY); // NaN
```

char

`char` 16 . .

```
char myChar = 'u';
char myChar2 = '5';
char myChar3 = 65; // myChar3 == 'A'
```

`\u0000` (10 0, *NULL*) `\uffff` (65,535) `\uffff` .

`char` `\u0000` .

```
char defaultChar;    // defaultChar == \u0000
```

' ( ) .

```
char singleQuote = '\'';
```

.

```
char tab = '\t';
char backspace = '\b';
char newline = '\n';
char carriageReturn = '\r';
char formfeed = '\f';
char singleQuote = '\'';
char doubleQuote = '\"'; // escaping redundant here; '"' would be the same; however still
allowed
char backslash = '\\';
char unicodeChar = '\uXXXX' // XXXX represents the Unicode-value of the character you want to
display
```

`char` .

```
char heart = '\u2764';
System.out.println(Character.toString(heart)); // Prints a line containing "❤".
```

char . .

```
for (int i = 0; i <= 26; i++) {
    char letter = (char) ('a' + i);
    System.out.println(letter);
}
```

Java *2* .

n .

n-1 x . VITH s . repesented

x-s * 2n-1

, 1 ($2^{n-2} + 2^{n-3} + \ldots + 2^{1} + 2^{0} = 2^{n-1} - 1$ 1 $2^{n-2} + 2^{n-3} + \ldots + 2^{1} + 2^{0} = 2^{n-1} - 1$) - (2) $^{N-1}$ (S = 1, X = 0) $^{N - 1}$ 2 1 - ((S) = 0, X = $^{2, N-1}$ - 1).

.

$$v1 = x1 - s1 * 2^{n-1}$$
$$v2 = x2 - s2 * 2^{n-1}$$

s1	s2	x1 + x2	
0	0		x1 + x2 = v1 + v2
0	0		()
0	1		$x1 + x2 - 2^{n-1} = x1 + x2 - s2 * 2^{n-1} = v1 + v2$
0	1		$(x1 + x2) \bmod 2^{n-1} = x1 + x2 - 2^{n-1} = v1 + v2$
1	0	*	(summand swap)
1	1		(X1 + X2 - 2 ^{N} <-2 $^{N-1]}$)
1	1		$(x1 + x2) \bmod 2^{n-1} - 2^{n-1} = (x1 + x2 - 2^{n-1}) - 2^{n-1} = (x1 - s1 * 2^{n-1}) + (x2 - s2 * 2^{n-1}) = v1 + v2$

(,) .

1. 1 0 (0) .

`i` (`int`)

```
(~i) + 1
```

: 0 (`byte`) `byte` :

`0` `11111111` . 1 `100000000` (9) . `byte` 8 `00000000`

0 (00000000)		-0 (11111111)
11111111	1 .	100000000
100000000	8 .	00000000 (-0 0)

(primitives) (boxed primitives)

		/
		1 / 16
		1 / 16
		2 / 16
		2 / 16
int		4 / 16
		8 / 16
		4 / 16
		8 / 16

8 8 *16* . , JVM JVM 4 8 .

. . `float[5]` 32 . 5 null `Float[5]` 112 (64 152).

. . , `Integer` `-128` `+127` . .

autoboxing static `valueOf(primitive)` . . , "" `new` `valueOf` . `new` . `new` .

Java . `char` . `boolean` .

.

. .

. .

Java . ( ), Java `cast` .

:

```
int a = 1;
double d = a;    // valid conversion to double, no cast needed (widening)
```

:

```
double d = 18.96
int b = d;       // invalid conversion to int, will throw a compile-time error
int b = (int) d; // valid conversion to int, but result is truncated (gets rounded down)
                 // This is type-casting
                 // Now, b = 18
```

Cheatsheet

:

	8	-2 7 ~ 2 7 - 1	0
		-128 ~ +127	
	16	-2 15 ~ 2 15 -1	0
		-32,768 ~ + 32,767	
int	32	-2 31 2 31 -1	0
		-2,147,483,648 +2,147,483,647	
	64	-2 63 2 63 -1	0L
		-9,223,372,036,854,775,808 ~ 9,223,372,036,854,775,807	
	32	1.401298464e-45 ~ 3.402823466e + 38 ( )	0.0F
	64	4.94065645841246544e-324d ~ 1.79769313486231570e + 308d ( )	0.0D
	16	0 2 16 - 1	0
		0 ~ 65,535	

:

1. (Java `byte long`) IEE 754 .

2. Java 8 `int` `long` . .
3. . , . 1.175494351e-38 2.2250738585072014e-308.
4. `char` / UTF-16 .
5. `boolean` 1 Java Virtual Machine ( ).

: https://riptutorial.com/ko/java/topic/148/--

145:

Java Atomic Types . .

compareAndSet	expeed .
getAndSet	

CAS . . : AtomicInteger , Unsafe.getAndSet

Examples

. , Livelock .

. .

`AtomicInteger` :

```
AtomicInteger aInt = new AtomicInteger() // Create with default value 0

AtomicInteger aInt = new AtomicInteger(1) // Create with initial value 1
```

.

```
AtomicIntegerArray aIntArray = new AtomicIntegerArray(10) // Create array of specific length
AtomicIntegerArray aIntArray = new AtomicIntegerArray(new int[] {1, 2, 3}) // Initialize array
with another array
```

.

`float double` . `Float.floatToIntBits(float) Float.intBitsToFloat(int) float Double.doubleToLongBits(double) Double.longBitsToDouble(long)` .

`sun.misc.Unsafe` `sun.misc.Unsafe` . int longs . sun.misc.Unsafe .

Java . : `synchronized` . `synchronized` .

```
public class Counters {
    private final int[] counters;

    public Counters(int nosCounters) {
        counters = new int[nosCounters];
    }

    /**
```

```
     * Increments the integer at the given index
     */
    public synchronized void count(int number) {
        if (number >= 0 && number < counters.length) {
            counters[number]++;
        }
    }

    /**
     * Obtains the current count of the number at the given index,
     * or if there is no number at that index, returns 0.
     */
    public synchronized int getCount(int number) {
        return (number >= 0 && number < counters.length) ? counters[number] : 0;
    }
}
```

. Counters . :

1. synchronized Counters .
2. number .
3. .

1 (). .

.

- (,), .
- . .
- thread , "" (, thread ()) . .

?

AtomicInteger AtomicInteger .

```
public class Counters {
    private final AtomicInteger[] counters;

    public Counters(int nosCounters) {
        counters = new AtomicInteger[nosCounters];
        for (int i = 0; i < nosCounters; i++) {
            counters[i] = new AtomicInteger();
        }
    }

    /**
     * Increments the integer at the given index
     */
    public void count(int number) {
        if (number >= 0 && number < counters.length) {
            counters[number].incrementAndGet();
        }
    }
```

```
    /**
     * Obtains the current count of the object at the given index,
     * or if there is no number at that index, returns 0.
     */
    public int getCount(int number) {
        return (number >= 0 && number < counters.length) ?
                counters[number].get() : 0;
    }
}
```

int[] AtomicInteger[] . int incrementAndGet() get() get() .

synchronized . incrementAndGet() get() . .

- "" ("") "" .
- T , T "" .

, 2 thread AtomicInteger (), , 1 (increment) . .

?

. , Intel CAS (Compare and Swap) .

AtomicXxx API . (C) :

```
private volatile num;

int increment() {
  while (TRUE) {
    int old = num;
    int new = old + 1;
    if (old == compare_and_swap(&num, old, new)) {
      return new;
    }
  }
}
```

AtomicXxxx if . if if "". , (CAS) .

, CAS JVM . JVM CAS . CAS , . HotSpot .

: https://riptutorial.com/ko/java/topic/5963/-

146:

.

. .

```
A -> B
  -> C -> D -> Done
       -> E -> Done
       -> F -> Done.
       -> G -> H -> I -> Done.
```

. , **A -> B** **A -> C** .

Examples

-

" " http://google.github.io/truth/

```
String string = "awesome";
assertThat(string).startsWith("awe");
assertWithMessage("Without me, it's just aweso").that(string).contains("me");

Iterable<Color> googleColors = googleLogo.getColors();
assertThat(googleColors)
    .containsExactly(BLUE, RED, YELLOW, BLUE, GREEN, RED)
    .inOrder();
```

`this` () .

.

.

```
public class Person {
  private  String firstName;
  private String lastName;

  public String getFirstName() {
    return firstName;
  }

  public void setFirstName(String firstName) {
    this.firstName = firstName;
  }

  public String getLastName() {
    return lastName;
  }

  public void setLastName(String lastName) {
```

```
    this.lastName = lastName;
  }

  public String whoAreYou() {
    return "I am " + firstName + " " + lastName;
  }

  public static void main(String[] args) {
    Person person = new Person();
    person.setFirstName("John");
    person.setLastName("Doe");
    System.out.println(person.whoAreYou());
  }
}
```

, 4 main Person . .

```
public class Person {
  private  String firstName;
  private String lastName;

  public String getFirstName() {
    return firstName;
  }

  public Person withFirstName(String firstName) {
    this.firstName = firstName;
    return this;
  }

  public String getLastName() {
    return lastName;
  }

  public Person withLastName(String lastName) {
    this.lastName = lastName;
    return this;
  }

  public String whoAreYou() {
    return "I am " + firstName + " " + lastName;
  }

  public static void main(String[] args) {
    System.out.println(new Person().withFirstName("John")
      .withLastName("Doe").whoAreYou());
  }
}
```

. .

: https://riptutorial.com/ko/java/topic/5090/-

147:

- TargetType target = (SourceType) source;

Examples

`boolean` .

`char` / . `char` 16 (2) `byte` (1) 8 (ASCII). `Character` `int` (4) / `short` (2) .

```
int badInt   = (int)  true; // Compiler error: incompatible types

char char1   = (char)   65; // A
byte byte1   = (byte)  'A'; // 65
short short1 = (short) 'A'; // 65
int int1     = (int)   'A'; // 65

char char2   = (char) 8253; // ?
byte byte2   = (byte)  '?'; // 61 (truncated code-point into the ASCII range)
short short2 = (short) '?'; // 8253
int int2     = (int)   '?'; // 8253
```

. .

```
//Implicit casting
byte byteVar = 42;
short shortVar = byteVar;
int intVar = shortVar;
long longVar = intvar;
float floatVar = longVar;
double doubleVar = floatVar;
```

.

```
//Explicit casting
double doubleVar = 42.0d;
float floatVar = (float) doubleVar;
long longVar = (long) floatVar;
int intVar = (int) longVar;
short shortVar = (short) intVar;
byte byteVar = (byte) shortVar;
```

(`float` , `double`) .

.

(3) 3 .

( ) . ( ) (`ClassCastException`) .

```
Float floatVar = new Float(42.0f);
```

```
Number n = floatVar;                //Implicit (Float implements Number)
Float floatVar2 = (Float) n;        //Explicit
Double doubleVar = (Double) n;      //Throws exception (the object is not Double)
```

```
 static void testNumericPromotion() {

    char char1 = 1, char2 = 2;
    short short1 = 1, short2 = 2;
    int int1 = 1, int2 = 2;
    float float1 = 1.0f, float2 = 2.0f;

    // char1 = char1 + char2;       // Error: Cannot convert from int to char;
    // short1 = short1 + short2;    // Error: Cannot convert from int to short;
    int1 = char1 + char2;           // char is promoted to int.
    int1 = short1 + short2;         // short is promoted to int.
    int1 = char1 + short2;          // both char and short promoted to int.
    float1 = short1 + float2;       // short is promoted to float.
    int1 = int1 + int2;             // int is unchanged.
}
```

instanceof

Java `instanceof` . .

```
Object obj = Calendar.getInstance();
long time = 0;

if(obj instanceof Calendar)
{
    time = ((Calendar)obj).getTime();
}
if(obj instanceof Date)
{
    time = ((Date)obj).getTime(); // This line will never be reached, obj is not a Date type.
}
```

: https://riptutorial.com/ko/java/topic/1392/-

148:

`interface` . , , , ... Java .

- Foo {void foo (); / * * /}
- Foo1 Foo {void bar (); / * * /}
- Foo2 Foo, Foo1 . {/ * Foo Foo1 * /}

Examples

`interface` `interface` :

```
public interface Animal {
    String getSound(); // Interface methods are public by default
}
```

```
@Override
public String getSound() {
    // Code goes here...
}
```

.

implements `implements` .

```
public class Cat implements Animal {

    @Override
    public String getSound() {
        return "meow";
    }
}

public class Dog implements Animal {

    @Override
    public String getSound() {
        return "woof";
    }
}
```

`Cat` `Dog` `getSound()` . ( ).

```
Animal cat = new Cat();
Animal dog = new Dog();

System.out.println(cat.getSound()); // prints "meow"
System.out.println(dog.getSound()); // prints "woof"
```

Java .

```
public interface NoiseMaker {
    String noise = "Making Noise"; // interface variables are public static final by default

    String makeNoise(); //interface methods are public abstract by default
}

public interface FoodEater {
    void eat(Food food);
}

public class Cat implements NoiseMaker, FoodEater {
    @Override
    public String makeNoise() {
        return "meow";
    }

    @Override
    public void eat(Food food) {
        System.out.println("meows appreciatively");
    }
}
```

`Cat` `abstract` . , (JVM **65,535**).

```
NoiseMaker noiseMaker = new Cat(); // Valid
FoodEater foodEater = new Cat(); // Valid
Cat cat = new Cat(); // valid

Cat invalid1 = new NoiseMaker(); // Invalid
Cat invalid2 = new FoodEater(); // Invalid
```

:

1. `public static final`
2. `public abstract` (Java 7 . Java 8 , .)
3. `final` .
4.
5. **InterfaceName.class** .

`extends` .

```
public interface BasicResourceService {
    Resource getResource();
}

public interface ExtendedResourceService extends BasicResourceService {
    void updateResource(Resource resource);
}
```

`ExtendedResourceService` `getResource()` `updateResource()` .

`extends` ( )

```
public interface BasicResourceService {
    Resource getResource();
```

```
}

public interface AlternateResourceService {
    Resource getAlternateResource();
}

public interface ExtendedResourceService extends BasicResourceService,
AlternateResourceService {
    Resource updateResource(Resource resource);
}
```

`ExtendedResourceService` `getResource()`, `getAlternateResource()` `updateResource()` .

Generics

(: AMQP, JMS) / .

IO .

```
public interface IO<IncomingType, OutgoingType> {

    void publish(OutgoingType data);
    IncomingType consume();
    IncomingType RPCSubmit(OutgoingType data);

}
```

, .

```
    IO<String, String> mockIO = new IO<String, String>() {

        private String channel = "somechannel";

        @Override
        public void publish(String data) {
            System.out.println("Publishing " + data + " to " + channel);
        }

        @Override
        public String consume() {
            System.out.println("Consuming from " + channel);
            return "some useful data";
        }

        @Override
        public String RPCSubmit(String data) {
            return "received " + data + " just now ";
        }

    };

    mockIO.consume(); // prints: Consuming from somechannel
    mockIO.publish("TestData"); // Publishing TestData to somechannel
    System.out.println(mockIO.RPCSubmit("TestData")); // received TestData just now
```

, RabbitMQ :

```
public class RabbitMQ implements IO<String, String> {

    private String exchange;
    private String queue;

    public RabbitMQ(String exchange, String queue){
        this.exchange = exchange;
        this.queue = queue;
    }

    @Override
    public void publish(String data) {
        rabbit.basicPublish(exchange, queue, data.getBytes());
    }

    @Override
    public String consume() {
        return rabbit.basicConsume(exchange, queue);
    }

    @Override
    public String RPCSubmit(String data) {
        return rabbit.rpcPublish(exchange, queue, data);
    }

}
```

IO . .

```
import java.util.concurrent.atomic.AtomicLong;

public class VisitCounter implements IO<Long, Integer> {

    private static AtomicLong websiteCounter = new AtomicLong(0);

    @Override
    public void publish(Integer count) {
        websiteCounter.addAndGet(count);
    }

    @Override
    public Long consume() {
        return websiteCounter.get();
    }

    @Override
    public Long RPCSubmit(Integer count) {
        return websiteCounter.addAndGet(count);
    }

}
```

VisitCounter .

```
    VisitCounter counter = new VisitCounter();

    // just had 4 visits, yay
    counter.publish(4);
    // just had another visit, yay
```

```
    counter.publish(1);

    // get data for stats counter
    System.out.println(counter.consume()); // prints 5

    // show data for stats counter page, but include that as a page view
    System.out.println(counter.RPCSubmit(1)); // prints 6
```

. . .

```
interface Printer<T> {
    void print(T value);
}

// Invalid!
class SystemPrinter implements Printer<Double>, Printer<Integer> {
    @Override public void print(Double d){ System.out.println("Decimal: " + d); }
    @Override public void print(Integer i){ System.out.println("Discrete: " + i); }
}
```

. , .

```
{cat, dog, bird}
```

. .

```
public interface Animal {
    public String getSound();
}
```

implements Animal implements Animal getSound() , .

```
public class Dog implements Animal {
    public String getSound() {
        return "Woof";
    }
}

public class Cat implements Animal {
    public String getSound() {
        return "Meow";
    }
}

public class Bird implements Animal{
    public String getSound() {
        return "Chirp";
    }
}
```

, getSound() . implement Animal , getSound() Animal getSound()

```
Animal dog = new Dog();
Animal cat = new Cat();
```

```
Animal bird = new Bird();

dog.getSound(); // "Woof"
cat.getSound(); // "Meow"
bird.getSound(); // "Chirp"
```

Animal

```
Animal[] animals = { new Dog(), new Cat(), new Bird() };
for (Animal animal : animals) {
    System.out.println(animal.getSound());
}
```

Dog , Cat , Bird *"Woof Meow Chirp"* .

. , *"dog"* Dog , *"cat"* Cat , *"bird"* Bird .

```
public Animal getAnimalByName(String name) {
    switch(name.toLowerCase()) {
        case "dog":
            return new Dog();
        case "cat":
            return new Cat();
        case "bird":
            return new Bird();
        default:
            return null;
    }
}

public String getAnimalSoundByName(String name){
    Animal animal = getAnimalByName(name);
    if (animal == null) {
        return null;
    } else {
        return animal.getSound();
    }
}

String dogSound = getAnimalSoundByName("dog"); // "Woof"
String catSound = getAnimalSoundByName("cat"); // "Meow"
String birdSound = getAnimalSoundByName("bird"); // "Chirp"
String lightbulbSound = getAnimalSoundByName("lightbulb"); // null
```

. Animal .

interface public abstract . abstract class interface , interface abstract class . abstract class . / abstract class abstract .

```
public interface NoiseMaker {
    void makeNoise();
}

public abstract class Animal implements NoiseMaker {
    //Does not need to declare or implement makeNoise()
    public abstract void eat();
```

```
}

//Because Dog is concrete, it must define both makeNoise() and eat()
public class Dog extends Animal {
    @Override
    public void makeNoise() {
        System.out.println("Borf borf");
    }

    @Override
    public void eat() {
        System.out.println("Dog eats some kibble.");
    }
}
```

Java 8 `interface default .`, `abstract default .`

Java 8 . "Base" "Abstract" .

Observer-Listener .

```
interface Observer {
    void onAction(String a);
}

interface Observable{
    public abstract List<Observer> getObservers();

    public default void addObserver(Observer o){
        getObservers().add(o);
    }

    public default void notify(String something ){
        for( Observer l : getObservers() ){
            l.onAction(something);
        }
    }
}
```

Observable "Observable" .

```
abstract class Worker{
    public abstract void work();
}

public class MyWorker extends Worker implements Observable {

    private List<Observer> myObservers = new ArrayList<Observer>();

    @Override
    public List<Observer> getObservers() {
        return myObservers;
    }

    @Override
    public void work(){
```

```
        notify("Started work");

        // Code goes here...

        notify("Completed work");
    }

    public static void main(String[] args) {
        MyWorker w = new MyWorker();

        w.addListener(new Observer() {
            @Override
            public void onAction(String a) {
                System.out.println(a + " (" + new Date() + ")");
            }
        });

        w.work();
    }
}
```

Java 8 .

.

```
interface InterfaceA {
    public default String getName(){
        return "a";
    }
}

interface InterfaceB {
    public default String getName(){
        return "b";
    }
}

public class ImpClass implements InterfaceA, InterfaceB {

    @Override
    public String getName() {
        //Must provide its own implementation
        return InterfaceA.super.getName() + InterfaceB.super.getName();
    }

    public static void main(String[] args) {
        ImpClass c = new ImpClass();

        System.out.println( c.getName() );                    // Prints "ab"
        System.out.println( ((InterfaceA)c).getName() );      // Prints "ab"
        System.out.println( ((InterfaceB)c).getName() );      // Prints "ab"
    }
}
```

.

.

. .

.

Oracle Java Style Guide .

.

(Oracle Oracle Official Code Standard .)

. .

```
interface I {
    public static final int VARIABLE = 0;

    public abstract void method();

    public static void staticMethod() { ... }
    public default void defaultMethod() { ... }
}
```

`public` ( ), `static` ( ) `final` ( ) .

```
~~public static final~~ int VARIABLE = 0;
```

---

1. `public abstract` .

```
~~public abstract~~ void method();
```

Java SE 8

2. `static` `default` `public` .

```
~~public~~ static void staticMethod() { ... }
```

.

```
interface I {
    int VARIABLE = 0;

    void method();

    static void staticMethod() { ... }
    default void defaultMethod() { ... }
}
```

.

```
class SomeClass {

}

class Demo<T extends SomeClass> {

}
```

.

. & .

```
interface SomeInterface {

}

class GenericClass<T extends SomeClass & SomeInterface> {

}
```

.

.

```
class Demo<T extends SomeClass & FirstInterface & SecondInterface> {

}
```

. , .

```
interface NewInterface extends FirstInterface, SecondInterface {

}

class Demo<T extends SomeClass & NewInterface> {

}
```

: https://riptutorial.com/ko/java/topic/102/

149:

Java .

Java .

API API . . Java . .

Examples

Pitfall : == Integer

(Integer int .)

Integer == , int . :

```
Integer int1_1 = Integer.valueOf("1");
Integer int1_2 = Integer.valueOf(1);

System.out.println("int1_1 == int1_2: " + (int1_1 == int1_2));          // true
System.out.println("int1_1 equals int1_2: " + int1_1.equals(int1_2));   // true
```

1 Integer (String int .) . (== equals) true .

.

```
Integer int2_1 = Integer.valueOf("1000");
Integer int2_2 = Integer.valueOf(1000);

System.out.println("int2_1 == int2_2: " + (int2_1 == int2_2));          // false
System.out.println("int2_1 equals int2_2: " + int2_1.equals(int2_2));   // true
```

equals .

JVM -128 127 Integer . ("java.lang.Integer.IntegerCache.high" JVM "-XX : AutoBoxCacheMax = size"). Integer.valueOf() .

Integer.valueOf(1) Integer.valueOf("1") Integer . Integer.valueOf(1000) Integer.valueOf("1000") Integer .

== (,) . int1_1 == int1_2 true . int2_1 == int2_2 false.

:

. . FileInputStream close() finalizer . . try-catch finally .

Java SE 7

```
private static void printFileJava6() throws IOException {
```

```
    FileInputStream input;
    try {
        input = new FileInputStream("file.txt");
        int data = input.read();
        while (data != -1){
            System.out.print((char) data);
            data = input.read();
        }
    } finally {
        if (input != null) {
            input.close();
        }
    }
}
```

Java 7 Java 7 try-with-resources .

Java SE 7

```
private static void printFileJava7() throws IOException {
    try (FileInputStream input = new FileInputStream("file.txt")) {
        int data = input.read();
        while (data != -1){
            System.out.print((char) data);
            data = input.read();
        }
    }
}
```

try-with `Closeable` `AutoCloseable` . . . `Closeable` `close()` `Closeable` `IOException` .

try-with-resources

Java SE 7

```
private static void printFileJava7(InputStream extResource) throws IOException {
    try (InputStream input = extResource) {
        ... //access resource
    }
}
```

try-with-resources .

:

Java . . .

. .

Java .

`static` `static null` . `static` GC . JNI .

. . . . . .

.

```
final ScheduledExecutorService scheduledExecutorService = Executors.newScheduledThreadPool(1);
final Deque<BigDecimal> numbers = new LinkedBlockingDeque<>();
final BigDecimal divisor = new BigDecimal(51);

scheduledExecutorService.scheduleAtFixedRate(() -> {
    BigDecimal number = numbers.peekLast();
    if (number != null && number.remainder(divisor).byteValue() == 0) {
        System.out.println("Number: " + number);
        System.out.println("Deque size: " + numbers.size());
    }
}, 10, 10, TimeUnit.MILLISECONDS);

scheduledExecutorService.scheduleAtFixedRate(() -> {
    numbers.add(new BigDecimal(System.currentTimeMillis()));
}, 10, 10, TimeUnit.MILLISECONDS);

try {
    scheduledExecutorService.awaitTermination(1, TimeUnit.DAYS);
} catch (InterruptedException e) {
    e.printStackTrace();
}
```

. `numbers` deque . 51 . . 10ms .

deque . deque .

, `pollLast` . `peekLast` , `pollLast` `peekLast` .

: ==

Java `==` . :

```
public class Hello {
    public static void main(String[] args) {
        if (args.length > 0) {
            if (args[0] == "hello") {
                System.out.println("Hello back to you");
            } else {
                System.out.println("Are you feeling grumpy today?");
            }
        }
    }
}
```

"hello" . . " ?" . .

`String` "hello" `String` args [0] . , . `==` . .

`==` `String` Java . , . `equals(Object)` . .

```
public class Hello2 {
    public static void main(String[] args) {
        if (args.length > 0) {
```

```
            if (args[0].equals("hello")) {
                System.out.println("Hello back to you");
            } else {
                System.out.println("Are you feeling grumpy today?");
            }
        }
    }
}
```

. == .

```
public class Test1 {
    public static void main(String[] args) {
        String s1 = "hello";
        String s2 = "hello";
        if (s1 == s2) {
            System.out.println("same");
        } else {
            System.out.println("different");
        }
    }
}
```

, "same" . ? (3.10.5 :) >> literal << Java . == test true . ("interned" " " .)

Java , 2 string , . :

```
    public class Test1 {
    public static void main(String[] args) {
        String s1 = "hello";
        String s2 = "hel" + "lo";
        String s3 = " mum";
        if (s1 == s2) {
            System.out.println("1. same");
        } else {
            System.out.println("1. different");
        }
        if (s1 + s3 == "hello mum") {
            System.out.println("2. same");
        } else {
            System.out.println("2. different");
        }
    }
}
```

"1. same" "2. different" . + String . String

== Java .

Pitfall : .

. , path .

```
public static File getValidatedFile(String path) throws IOException {
    File f = new File(path);
```

```
    if (!f.exists()) throw new IOException("Error: not found: " + path);
    if (!f.isFile()) throw new IOException("Error: Is a directory: " + path);
    if (!f.canRead()) throw new IOException("Error: cannot read file: " + path);
    return f;
}
```

.

```
File f = null;
try {
    f = getValidatedFile("somefile");
} catch (IOException ex) {
    System.err.println(ex.getMessage());
    return;
}
try (InputStream is = new FileInputStream(file)) {
    // Read data etc.
}
```

`FileInputStream(File)` . `IOException` .

`getValidatedFile` `FileInputStream` .

- : `getValidatedFile` . "" `IOException` .
- . , SELinux "" `canRead()` `true` .

. , `exists` , `isFile` `canRead` . , .

, `getValidatedFile` `getValidatedFile` . .

```
try (InputStream is = new FileInputStream("somefile")) {
    // Read data etc.
} catch (IOException ex) {
    System.err.println("IO Error processing 'somefile': " + ex.getMessage());
    return;
}
```

try / catch . `exists` , `isFile` `canRead` `exists` .

Pitfall :

Java .

```
String foo;   // NOT AN OBJECT
```

Java .

```
String bar[] = new String[100];  // No member is an object.
```

Java .

-

(: `int` `float`) Java . . `int` . .

- ( ) Java . . , `null` . .

. .

2D .

```
public final class MutableLocation {
   public int x;
   public int y;

   public MutableLocation(int x, int y) {
       this.x = x;
       this.y = y;
   }

   public boolean equals(Object other) {
       if (!(other instanceof MutableLocation) {
           return false;
       }
       MutableLocation that = (MutableLocation) other;
       return this.x == that.x && this.y == that.y;
   }
}
```

, `int` 2 `x` `y` .

`MutableLocation` . 2D . , `x` `y` . .

```
 MutableLocation here = new MutableLocation(1, 2);
 MutableLocation there = here;
 MutableLocation elsewhere = new MutableLocation(1, 2);
```

`here` , `there` `elsewhere` `MutableLocation` .

() , :

1. "[1, 2]" `here` `here`
2. "[1, 2]" `there`
3. "[1, 2]" `elsewhere` .

. . `here` `there` .

. .

```
System.out.println("BEFORE: here.x is " + here.x + ", there.x is " + there.x +
                   "elsewhere.x is " + elsewhere.x);
here.x = 42;
System.out.println("AFTER: here.x is " + here.x + ", there.x is " + there.x +
                   "elsewhere.x is " + elsewhere.x);
```

.

```
BEFORE: here.x is 1, there.x is 1, elsewhere.x is 1
AFTER: here.x is 42, there.x is 42, elsewhere.x is 1
```

here.x , there.x . there.x . . elsewhere.x . elsewhere .

here.x = 42 . there.x .

▪

(==) . () "" .

```
 MutableLocation here = new MutableLocation(1, 2);
 MutableLocation there = here;
 MutableLocation elsewhere = new MutableLocation(1, 2);

 if (here == there) {
     System.out.println("here is there");
 }
 if (here == elsewhere) {
     System.out.println("here is elsewhere");
 }
```

"here is there" "here is other is" . (here elsewhere .)

equals(Object) MutableLocation .

```
 if (here.equals(there)) {
     System.out.println("here equals there");
 }
 if (here.equals(elsewhere)) {
     System.out.println("here equals elsewhere");
 }
```

. , here.equals(elsewhere) MutableLocation MutableLocation true true .

▪

Java [1] .

. .

, . .

. [2] . .

```
void f(MutableLocation foo) {
    foo = new MutableLocation(3, 4);   // Point local foo at a different object.
}

void g() {
    MutableLocation foo = MutableLocation(1, 2);
    f(foo);
```

```
    System.out.println("foo.x is " + foo.x); // Prints "foo.x is 1".
}
```

.

```
void f(MutableLocation foo) {
    foo.x = 42;
}

void g() {
    MutableLocation foo = new MutableLocation(0, 0);
    f(foo);
    System.out.println("foo.x is " + foo.x); // Prints "foo.x is 42"
}
```

---

1 - ," " / " ".

2 - """" . . Java . https://en.wikipedia.org/wiki/Evaluation_strategy .

:

StackOverflow Java (C C ++) .

```
i += a[i++] + b[i--];
```

`i , a b` .

:

- Java [1] .
- C C ++ .

Java . " " .

, (StackOverflow). .

```
int a = 1;
a = a++;
System.out.println(a);    // What does this print.
```

(Java) `2` . `1` . .

, . .

---

1 - Java .

: String .

Java Java `String` . .

```
public class Shout {
    public static void main(String[] args) {
        for (String s : args) {
            s.toUpperCase();
            System.out.print(s);
            System.out.print(" ");
        }
        System.out.println();
    }
}
```

., , . .

```
s.toUpperCase();
```

toUpperCase() s . . ! String . .

toUpperCase() String String . String . s .

. :

```
s = s.toUpperCase();
```

" " String . .

: https://riptutorial.com/ko/java/topic/4388/--

150: (JVM)

Examples

.

JVM RAM . Java . (.)

Java JVM JVM . JDK .

(*Byte code* *Machine code* Windows Linux .)

: -

- - .class RAM.
- - .
- - .
- JIT (Just in time) - JIT JVM JVM . Java .

( )

(JVM) : https://riptutorial.com/ko/java/topic/8110/----jvm-

151:

JNIEnv	JNI
	`static` `native`
jclass	`static` `native`

JNI Java . IDE OS . Eclipse . .

Windows Java-C ++ .

- `javac` Java (`.java`) (`.class`) `javac` .
- `javah` `native` Java (`.h`) `javah` . "".
- `native` C ++ (`.cpp`) (`#include`) .
- C ++ (`.dll`) . .
- (`-Djava.library.path`) Java (`System.loadLibrary(...)`).

(Java) . . `javap -s` .

Examples

C ++

Java . JVM ( ) .

Java

```
/*** com/example/jni/JNIJava.java **/

package com.example.jni;

public class JNIJava {
    static {
        System.loadLibrary("libJNI_CPP");
    }

    // Obviously, native methods may not have a body defined in Java
    public native void printString(String name);
    public static native double average(int[] nums);

    public static void main(final String[] args) {
        JNIJava jniJava = new JNIJava();
        jniJava.printString("Invoked C++ 'printString' from Java");

        double d = average(new int[]{1, 2, 3, 4, 7});
        System.out.println("Got result from C++ 'average': " + d);
    }
```

```
}
```

C ++

javah . .

```
javah -o com_example_jni_JNIJava.hpp com.example.jni.JNIJava
```

... ( ).

```
// com_example_jni_JNIJava.hpp

/* DO NOT EDIT THIS FILE - it is machine generated */
#include <jni.h> // The JNI API declarations

#ifndef _Included_com_example_jni_JNIJava
#define _Included_com_example_jni_JNIJava
#ifdef __cplusplus
extern "C" { // This is absolutely required if using a C++ compiler
#endif

JNIEXPORT void JNICALL Java_com_example_jni_JNIJava_printString
  (JNIEnv *, jobject, jstring);

JNIEXPORT jdouble JNICALL Java_com_example_jni_JNIJava_average
  (JNIEnv *, jclass, jintArray);

#ifdef __cplusplus
}
#endif
#endif
```

.

```
// com_example_jni_JNIJava.cpp

#include <iostream>
#include "com_example_jni_JNIJava.hpp"

using namespace std;

JNIEXPORT void JNICALL Java_com_example_jni_JNIJava_printString(JNIEnv *env, jobject jthis,
jstring string) {
    const char *stringInC = env->GetStringUTFChars(string, NULL);
    if (NULL == stringInC)
        return;
    cout << stringInC << endl;
    env->ReleaseStringUTFChars(string, stringInC);
}

JNIEXPORT jdouble JNICALL Java_com_example_jni_JNIJava_average(JNIEnv *env, jclass jthis,
jintArray intArray) {
    jint *intArrayInC = env->GetIntArrayElements(intArray, NULL);
    if (NULL == intArrayInC)
        return -1;
    jsize length = env->GetArrayLength(intArray);
```

```
    int sum = 0;
    for (int i = 0; i < length; i++) {
        sum += intArrayInC[i];
    }
    env->ReleaseIntArrayElements(intArray, intArrayInC, 0);
    return (double) sum / length;
}
```

.

Java C ++ 'printString'
C ++ 'average': 3.4

C ++ () Java

Java .

1. , `GetMethodID` JNI .
2. `Call*Method` `Call*Method` .

Java

```
/*** com.example.jni.JNIJavaCallback.java ***/

package com.example.jni;

public class JNIJavaCallback {
    static {
        System.loadLibrary("libJNI_CPP");
    }

    public static void main(String[] args) {
        new JNIJavaCallback().callback();
    }

    public native void callback();

    public static void printNum(int i) {
        System.out.println("Got int from C++: " + i);
    }

    public void printFloat(float i) {
        System.out.println("Got float from C++: " + i);
    }
}
```

C ++

```
// com_example_jni_JNICppCallback.cpp

#include <iostream>
#include "com_example_jni_JNIJavaCallback.h"
```

```
using namespace std;

JNIEXPORT void JNICALL Java_com_example_jni_JNIJavaCallback_callback(JNIEnv *env, jobject
jthis) {
    jclass thisClass = env->GetObjectClass(jthis);

    jmethodID printFloat = env->GetMethodID(thisClass, "printFloat", "(F)V");
    if (NULL == printFloat)
        return;
    env->CallVoidMethod(jthis, printFloat, 5.221);

    jmethodID staticPrintInt = env->GetStaticMethodID(thisClass, "printNum", "(I)V");
    if (NULL == staticPrintInt)
        return;
    env->CallVoidMethod(jthis, staticPrintInt, 17);
}
```

C ++ : 5.221
C ++ int : 17

( ) `.class` **javap** . `javap -p -s com.example.jni.JNIJavaCallback` .

```
Compiled from "JNIJavaCallback.java"
public class com.example.jni.JNIJavaCallback {
  static {};
    descriptor: ()V

  public com.example.jni.JNIJavaCallback();
    descriptor: ()V

  public static void main(java.lang.String[]);
    descriptor: ([Ljava/lang/String;)V

  public native void callback();
    descriptor: ()V

  public static void printNum(int);
    descriptor: (I)V  // <---- Needed

  public void printFloat(float);
    descriptor: (F)V  // <---- Needed
}
```

Java .

```
public class ClassWithNativeMethods {
    static {
        System.loadLibrary("Example");
    }

    public native void someNativeMethod(String arg);
    ...
```

`System.loadLibrary` . .

```
public class ClassWithNativeMethods {
```

```
    // Call this before using any native method
    public static void prepareNativeMethods() {
        System.loadLibrary("Example");
    }

    ...
```

`java.lang.UnsatisfiedLinkError` .

`-Djava.library.path=` JVM `java.library.path` .

```
java -Djava.library.path=path/to/lib/:path/to/other/lib MainClassWithNativeMethods
```

(: Windows `;` `:` .

`System.loadLibrary` . Linux `libExample.so` Windows `Example.dll` .

`System.loadLibrary` `System.load(String)` . `java.library.path` .

```
public class ClassWithNativeMethods {
    static {
        System.load("/path/to/lib/libExample.so");
    }

    ...
```

: https://riptutorial.com/ko/java/topic/168/--

152:

Java , "" . `JAR` , `WAR` `EAR` , .

(: ".class") Java . Java .

- JAR JAR ; : `javac` .
- WAR, EAR " " " " .
- . JRE .
- JAR Java WebStart .

JAR, WAR EAR f} .

Java " " "EXE " . Java ( ) . "Java " .

Examples

JAR

jar . .

.

```
import javax.swing.*;
import java.awt.Container;

public class HelloWorld {

    public static void main(String[] args) {
        JFrame f = new JFrame("Hello, World");
        JLabel label = new JLabel("Hello, World");
        Container cont = f.getContentPane();
        cont.add(label);
        f.setSize(400,100);
        f.setVisible(true);
        f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

}
```

HelloWorld.java .

.

. Java .

HelloWorld.class .

manifest.txt .

```
Main-Class: HelloWorld
Class-Path: HelloWorld.jar
```

HelloWorld.class .
(`cd C:\Your\Folder\Path\Here` Windows) .

(`cd /Users/user/Documents/Java/jarfolder` Mac).

, `jar -cvfm HelloWorld.jar manifest.txt HelloWorld.class` Enter . .class HelloWorld.jar jar (HelloWorld.class) . (: -m -v).
HelloWorld.jar .
`Hello, World` .

JAR, WAR EAR

JAR, WAR EAR "" ZIP WAR EAR / .

"" Java . IDE "" .

(*:* . . *!*)

Maven JAR WAR

Maven JAR WAR POM `<packaging>` . e, g,

```
<packaging>jar</packaging>
```

```
<packaging>war</packaging>
```

. Maven Maven jar Plugin " "JAR . JAR "uberJAR" .

Maven (http://www.riptutorial.com/topic/898) .

Ant JAR, WAR EAR

Ant JAR, WAR EAR "" . Ant (http://www.riptutorial.com/topic/4223) .

IDE JAR, WAR EAR

Java IDE . "" .

- Eclipse - http://www.riptutorial.com/topic/1143
- NetBeans - http://www.riptinar.com/topic/5438
- Intellij-IDEA -

`jar` JAR, WAR EAR .

`jar` "" . , , JAR `jar` .

`jar` Topic (JAR) .

Java Web Start

Oracle Java Tutorials Web Start .

Java Web Start . .

Java Web Start .

Java Web Start JavaWS JAWS. .

- - : Java Web Start
- Java Web Start Guide
- Java Web Start FAQ
- JNLP
- `javax.jnlp` API
- Java Web Start

Web Start JAR Java . .

- Java (JRE JDK). Java 1.2.2 .
  - Java 5.0 Web Start JRE / JDK .
  - Web Start .
  - Web Start JavaScript .
- .
- Web Start .
  - .
  - () Java .

JNLP

, JNLP .

```
<?xml version="1.0" encoding="UTF-8"?>
<jnlp spec="1.0+" codebase="https://www.example.com/demo"
    href="demo_webstart.jnlp">
    <information>
        <title>Demo</title>
        <vendor>The Example.com Team</vendor>
    </information>
    <resources>
```

```
        <!-- Application Resources -->
        <j2se version="1.7+" href="http://java.sun.com/products/autodl/j2se"/>
        <jar href="Demo.jar" main="true"/>
    </resources>
    <application-desc
         name="Demo Application"
         main-class="com.example.jwsdemo.Main"
         width="300"
         height="300">
     </application-desc>
     <update check="background"/>
</jnlp>
```

, JNLP XML `<jnlp>` .

- `spec` JNPL .
- `codebase` `href` URL URL .
- `href` JNLP URL .
- `<information>` , , .
- `<resources>` Java , OS JAR .
- `<application-desc>` (`<applet-desc>`) .

`.jnlp` MIME `application/x-java-jnlp-file` `application/x-java-jnlp-file` .

JNLP JAR JNLP URL .

.

- Java Web Start . .

```
<a href="https://www.example.com/demo_webstart.jnlp">Launch the application</a>
```

- Java .

: Java Java JNLP .

Web Start

Web Start . Java 5.0 JRE JDK .

```
$ javaws <url>
```

`<url>` JNLP URL.

UberJAR

Java . Java SE , ( ) JAR .

. JAR JAR Java . JAR . .

JAR UberJAR.

"jar" UberJAR

UberJAR . (. Mac OS Windows .)

1. .

```
$ mkdir tempDir
$ cd tempDir
```

2. JAR `jar` JAR .

```
$ jar -xf <path/to/file.jar>
```

JAR JAR .

3. .

```
$ cp -r path/to/classes .
```

4. UberJAR .

```
$ jar -cf ../myApplication.jar
```

JAR MANIFEST.MF .

Maven UberJAR

Maven "maven-assembly" "maven-shade" UberJAR . ( ).

UberJAR

UberJAR .

- UberJAR .
- UberJAR .

UberJAR JAR . . , .

UberJAR .

- UberJAR .
- UberJAR [1] .

1 - . UberJAR .

: https://riptutorial.com/ko/java/topic/6840/-

153:

JavaBeans (TM) Java (bean) Java API . getter setter API .

- **JavaBean**
- getter get. getSize () "size" JavaBeans getter . size . getter setter . getSize () .
- getter get is. , getStopped () isStopped () JavaBeans .
- setter . setSize () size JavaBean .
- getter setter (get, is set) .
- Setter public void .
- Getter public setter .
- **JavaBean**
- "" add . addActionListener () Action .
- (" ") remove add ( ).
- .
- "Listener" .

Java Bean . .

- getter setter .
- .
- `java.io.Serializable` .

Examples

Java Bean

```
public class BasicJavaBean implements java.io.Serializable{

    private int value1;
    private String value2;
    private boolean value3;

    public BasicJavaBean(){}

    public void setValue1(int value1){
        this.value1 = value1;
    }

    public int getValue1(){
        return value1;
    }

    public void setValue2(String value2){
        this.value2 = value2;
    }

    public String getValue2(){
        return value2;
    }
```

```
    public void setValue3(boolean value3){
        this.value3 = value3;
    }

    public boolean isValue3(){
        return value3;
    }
}
```

: https://riptutorial.com/ko/java/topic/8157/-

154: - 'javac'

`javac` Java . . , Java .

`javac` Java Development Kit (JDK) .

Java Java .

- ".java"
- ".class" .
- .

: `javac` JIT (Just in Time) .

Examples

'javac' -

"HelloWorld.java" Java .

```
public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello world!");
    }
}
```

(Java .)

.

```
$ javac HelloWorld.java
```

"HelloWorld.class" .

```
$ java HelloWorld
Hello world!
```

.

1. "HelloWorld.java" ... `HelloWorld` . .
2. "HelloWorld.class" . "HelloWorld.class" .
3. `java` Java .

Java .

`com.example` `HelloWorld` , "HelloWorld.java" :

```
package com.example;
```

```
public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello world!");
    }
}
```

.

```
.    # the current directory (for this example)
|
 ----com
     |
      ----example
          |
           ----HelloWorld.java
```

.

```
$ javac com/example/HelloWorld.java
```

"com / example / HelloWorld.class" . , .

```
.    # the current directory (for this example)
|
 ----com
     |
      ----example
          |
           ----HelloWorld.java
           ----HelloWorld.class
```

.

```
$ java com.example.HelloWorld
Hello world!
```

.

1. .
2. . "com.example.HelloWorld" "HelloWorld" .
3. Java . .

'javac' .

! . .

```
$ javac Foo.java Bar.java
```

....

```
$ javac *.java
$ javac com/example/*.java
$ javac */**/*.java #Only works on Zsh or with globstar enabled on your shell
```

"com / example" Java Java . ( ) . :

```
$ javac @sourcefiles
```

`sourcefiles` .

```
Foo.java
Bar.java
com/example/HelloWorld.java
```

: 1 . Java . "" Java IDE (: NetBeans , Eclipse , IntelliJ IDEA) .

'javac'

`javac` .

- `-d` ".class" .
- `-sourcepath` .
- `-cp -classpath` . .
- `-version` .

.

`javac` `javac` .

Java

Java ( ) . .

- Java
- Java API .
- Java ( ) .

(: `enum` , "") .

- Java Java Java .
- Java Java Java .

Java

Java Java () . (: `enum`) `-source` . .

```
public class OldSyntax {
    private static int enum;  // invalid in Java 5 or later
```

```
}
```

Java 5 ( ) .

```
$ javac -source 1.4 OldSyntax.java
```

Java Java JDK JDK .

Java . .

- Java Java .
- Java , , .
- .

`-target` . ,

```
$ javac -target 1.4 SomeClass.java
```

Java 1.4 JVM . (, `-source` `-target` `javac -source 1.4 ...` . `-source` `-target` Oracle .)

`-target` `-source` JDK . API . `-bootclasspath` . :

```
$ javac -target 1.4 --bootclasspath path/to/java1.4/rt.jar SomeClass.java
```

. ( ) .

- 'javac' : https://riptutorial.com/ko/java/topic/4478/-----javac-

155:

IDE / . API API . , . jar 'plugins' .

Examples

URLClassLoader

Java . *URLClassLoader* . JavaFX .

. 'plugins' Jars . :

```
package main;

public class MainApplication extends Application
{
    @Override
    public void start(Stage primaryStage) throws Exception
    {
        File pluginDirectory=new File("plugins"); //arbitrary directory
        if(!pluginDirectory.exists())pluginDirectory.mkdir();
        VBox loadedPlugins=new VBox(6); //a container to show the visual info later
        Rectangle2D screenbounds=Screen.getPrimary().getVisualBounds();
        Scene scene=new
Scene(loadedPlugins,screenbounds.getWidth()/2,screenbounds.getHeight()/2);
        primaryStage.setScene(scene);
        primaryStage.show();
    }
    public static void main(String[] a)
    {
            launch(a);
    }
}
```

.

```
package main;

public interface Plugin
{
    default void initialize()
    {
        System.out.println("Initialized "+this.getClass().getName());
    }
    default String name(){return getClass().getSimpleName();}
}
```

'.jar' .

```
File[] files=pluginDirectory.listFiles((dir, name) -> name.endsWith(".jar"));
```

URL .

```
    if(files!=null && files.length>0)
    {
        ArrayList<String> classes=new ArrayList<>();
        ArrayList<URL> urls=new ArrayList<>(files.length);
        for(File file:files)
        {
            JarFile jar=new JarFile(file);
            jar.stream().forEach(jarEntry -> {
                if(jarEntry.getName().endsWith(".class"))
                {
                    classes.add(jarEntry.getName());
                }
            });
            URL url=file.toURI().toURL();
            urls.add(url);
        }

    }
```

MainApplication HashSet .

```
static HashSet<Plugin> plugins=new HashSet<>();
```

URLClassLoader *Plugin* .

```
URLClassLoader urlClassLoader=new URLClassLoader(urls.toArray(new URL[urls.size()]));
classes.forEach(className->{
    try
    {
        Class
cls=urlClassLoader.loadClass(className.replaceAll("/",".").replace(".class",""));
//transforming to binary name
        Class[] interfaces=cls.getInterfaces();
        for(Class intface:interfaces)
        {
            if(intface.equals(Plugin.class)) //checking presence of Plugin interface
            {
                Plugin plugin=(Plugin) cls.newInstance(); //instantiating the Plugin
                plugins.add(plugin);
                break;
            }
        }
    }
    catch (Exception e){e.printStackTrace();}
});
```

.

```
if(!plugins.isEmpty())loadedPlugins.getChildren().add(new Label("Loaded plugins:"));
    plugins.forEach(plugin -> {
        plugin.initialize();
        loadedPlugins.getChildren().add(new Label(plugin.name()));
});
```

MainApplication :

```
package main;
public class MainApplication extends Application
{
    static HashSet<Plugin> plugins=new HashSet<>();
    @Override
    public void start(Stage primaryStage) throws Exception
    {
        File pluginDirectory=new File("plugins");
        if(!pluginDirectory.exists())pluginDirectory.mkdir();
        File[] files=pluginDirectory.listFiles((dir, name) -> name.endsWith(".jar"));
        VBox loadedPlugins=new VBox(6);
        loadedPlugins.setAlignment(Pos.CENTER);
        if(files!=null && files.length>0)
        {
            ArrayList<String> classes=new ArrayList<>();
            ArrayList<URL> urls=new ArrayList<>(files.length);
            for(File file:files)
            {
                JarFile jar=new JarFile(file);
                jar.stream().forEach(jarEntry -> {
                    if(jarEntry.getName().endsWith(".class"))
                    {
                        classes.add(jarEntry.getName());
                    }
                });
                URL url=file.toURI().toURL();
                urls.add(url);
            }
            URLClassLoader urlClassLoader=new URLClassLoader(urls.toArray(new
URL[urls.size()]));
            classes.forEach(className->{
                try
                {
                    Class
cls=urlClassLoader.loadClass(className.replaceAll("/",".").replace(".class",""));
                    Class[] interfaces=cls.getInterfaces();
                    for(Class intface:interfaces)
                    {
                        if(intface.equals(Plugin.class))
                        {
                            Plugin plugin=(Plugin) cls.newInstance();
                            plugins.add(plugin);
                            break;
                        }
                    }
                }
                catch (Exception e){e.printStackTrace();}
            });
            if(!plugins.isEmpty())loadedPlugins.getChildren().add(new Label("Loaded
plugins:"));
            plugins.forEach(plugin -> {
                plugin.initialize();
                loadedPlugins.getChildren().add(new Label(plugin.name()));
            });
        }
        Rectangle2D screenbounds=Screen.getPrimary().getVisualBounds();
        Scene scene=new
Scene(loadedPlugins,screenbounds.getWidth()/2,screenbounds.getHeight()/2);
        primaryStage.setScene(scene);
        primaryStage.show();
    }
```

```
    public static void main(String[] a)
    {
            launch(a);
    }
}
```

. , .

```
package plugins;

import main.Plugin;

public class FirstPlugin implements Plugin
{
    //this plugin has default behaviour
}
```

:

```
package plugins;

import main.Plugin;

public class AnotherPlugin implements Plugin
{
    @Override
    public void initialize() //overrided to show user's home directory
    {
        System.out.println("User home directory: "+System.getProperty("user.home"));
    }
}
```

Jars . IDE .

Jars " *MainApplication* .

: https://riptutorial.com/ko/java/topic/7160/---

156: -

Examples

Pitfall : wait () / notify () .

`object.wait()` , `object.notify()` `object.notifyAll()` . (http://stackoverflow.com/documentation/java/5409/wait-notify#t=20160811161648303307)

" "

`object.wait()` .

```
private final Object lock = new Object();

public void myConsumer() {
    synchronized (lock) {
        lock.wait();     // DON'T DO THIS!!
    }
    doSomething();
}
```

`lock.notify()` `lock.notifyAll()` `lock.wait()` .

`lock.notify()` `lock.notifyAll()` . `myConsumer()` .

" "

`wait()` `notify()` JVM `IllegalMonitorStateException` .

```
public void myConsumer() {
    lock.wait();      // throws exception
    consume();
}

public void myProducer() {
    produce();
    lock.notify();    // throws exception
}
```

(`wait()` / `notify()` . `wait()` `notify()` : .)

/ .

`wait()` `notify()` . `java.utils.concurrent` (, ,) .

- java.lang.Thread

`Thread` javadoc `Thread` .

:

```
class PrimeThread extends Thread {
    long minPrime;
    PrimeThread(long minPrime) {
        this.minPrime = minPrime;
    }

    public void run() {
        // compute primes larger than minPrime
         . . .
    }
}

PrimeThread p = new PrimeThread(143);
p.start();
```

`Runnable` :

```
class PrimeRun implements Runnable {
    long minPrime;
    PrimeRun(long minPrime) {
        this.minPrime = minPrime;
    }

    public void run() {
        // compute primes larger than minPrime
         . . .
    }
}

PrimeRun p = new PrimeRun(143);
new Thread(p).start();
```

(: `java.lang.Thread` javadoc .)

:

1. , executor ForkJoin `PrimeThread` `PrimeThread` . (`PrimeThread` `Runnable` `Thread` `Runnable` .)
2. . , `Thread.start()` `PrimeThread.start()` "" .

`Runnable` . `Runnable` (Java 1.1) .

```
final long minPrime = ...
new Thread(new Runnable() {
    public void run() {
        // compute primes larger than minPrime
         . . .
    }
}).start();
```

(Java 8) .

```
final long minPrime = ...
new Thread(() -> {
   // compute primes larger than minPrime
    . . .
}).start();
```

Pitfall - .

. , . .

- ( ) .
- Java (hyperthread) .
- ( ) / Java , .

Java . :

- . () 512KB 1M . .
- . .
- . OS .
- (: wait (), notify () / notifyAll) .

" " . .

(:) .

( ) . . `ExecutorService` .

- .

.

, . thread `Runnable` .

```
public class ThreadTest {
    public static void main(String[] args) throws Exception {
        while (true) {
            long start = System.nanoTime();
            for (int i = 0; i < 100_000; i++) {
                Thread t = new Thread(new Runnable() {
                        public void run() {
                });
                t.start();
                t.join();
            }
            long end = System.nanoTime();
            System.out.println((end - start) / 100_000.0);
        }
    }
```

```
}

$ java ThreadTest
34627.91355
33596.66021
33661.19084
33699.44895
33603.097
33759.3928
33671.5719
33619.46809
33679.92508
33500.32862
33409.70188
33475.70541
33925.87848
33672.89529
^C
```

64 Java 8 u101 Linux PC 33.6 - 33.9 , .

`ExecutorService` `Future` .

```
import java.util.concurrent.*;

public class ExecutorTest {
    public static void main(String[] args) throws Exception {
        ExecutorService exec = Executors.newCachedThreadPool();
        while (true) {
            long start = System.nanoTime();
            for (int i = 0; i < 100_000; i++) {
                Future<?> future = exec.submit(new Runnable() {
                    public void run() {
                    }
                });
                future.get();
            }
            long end = System.nanoTime();
            System.out.println((end - start) / 100_000.0);
        }
    }
}

$ java ExecutorTest
6714.66053
5418.24901
5571.65213
5307.83651
5294.44132
5370.69978
5291.83493
5386.23932
5384.06842
5293.14126
5445.17405
5389.70685
^C
```

, 5.3 ~ 5.6 .

. .

: .

.

```
public class ThreadTest implements Runnable {

    private boolean stop = false;

    public void run() {
        long counter = 0;
        while (!stop) {
            counter = counter + 1;
        }
        System.out.println("Counted " + counter);
    }

    public static void main(String[] args) {
        ThreadTest tt = new ThreadTest();
        new Thread(tt).start();    // Create and start child thread
        Thread.sleep(1000);
        tt.stop = true;            // Tell child thread to stop.
    }
}
```

1000 `stop` `stop` .

?

, .

`main` . , . . `tt.stop` `true` `true` .

., `true` `stop` `stop` `true` . ? , .

Java , . , , () . . `stop` `stop` `true` .

(: .)

?

`stop` .

1. `volatile` `stop` .

   ```
   private volatile boolean stop = false;
   ```

   `volatile` , JLS, 1 thread write 2 thread read , *happen-before* .

2. .

```
public class ThreadTest implements Runnable {

    private boolean stop = false;

    public void run() {
        long counter = 0;
        while (true) {
            synchronize (this) {
                if (stop) {
                    break;
                }
            }
            counter = counter + 1;
        }
        System.out.println("Counted " + counter);
    }

    public static void main(String[] args) {
        ThreadTest tt = new ThreadTest();
        new Thread(tt).start();    // Create and start child thread
        Thread.sleep(1000);
        synchronize (tt) {
            tt.stop = true;        // Tell child thread to stop.
        }
    }
}
```

JLS, , 1 thread 2 thread .

?

, !

. .

?

Java . . . Java ?

().

() . . . .

. . . / .

. . , . (" ") .

(cache write-through) .

Java . Java (write-through) . (: , `volatile` ,) . Java ( ) .

. .

?

.

1. , . .

2. . .

3. . , I / O . traceprints .

4. JIT . . .

.

- : https://riptutorial.com/ko/java/topic/5567/-----

157:

. . . .

Java .

.

- . . .

```
if (n <= 1) {
    return 1;
}
```

- . . .

```
else {
    return n * factorial(n - 1);
}
```

n . .

0! = 1

1! = 1

2! = 1 × 2 = 2

! = 1 x 2 x 3 = 6

4! = 1 x 2 x 3 x 4 = 24

...

Java Tail-call

Java (Java 9) . . `StackOverflowError` . Java .

Examples

?

. :

```
// This method calls itself "infinitely"
public void useless() {
```

```
    useless();  // method calls itself (directly)
}
```

:

.

1. . ( ) .

2. . . .

Java . . Java .

. `factorial` . `n` 1 . `n` 1 ( ) .

```
public int factorial(int n) {
    if (n <= 1) { // the base condition
        return 1;
    } else {
        return n * factorial(n - 1);
    }
}
```

`n` (: `StackOverflowError`) Java .

N

N .

```
public int fib(final int n) {
    if (n > 2) {
        return fib(n - 2) + fib(n - 1);
    }
    return 1;
}
```

(n <= 2) (n> 2) . .

. N . O (2 N) O (N) . O (N) " " .

1 N

0 N .

```
public int sum(final int n) {
    if (n > 0) {
        return n + sum(n - 1);
    } else {
        return n;
    }
}
```

O (N) . `O(1)` .

N

`exp` `exp` `num` .

```
public long power(final int num, final int exp) {
    if (exp == 0) {
        return 1;
    }
    if (exp == 1) {
        return num;
    }
    return num * power(num, exp - 1);
}
```

. (n = 0 n = 1) . O (N) .

.

```
/**
 * Just a snippet to explain the idea of recursion
 *
 **/

public class Reverse {
    public static void main (String args[]) {
        String string = "hello world";
        System.out.println(reverse(string)); //prints dlrow olleh
    }

    public static String reverse(String s) {
        if (s.length() == 1) {
            return s;
        }

        return reverse(s.substring(1)) + s.charAt(0);
    }
}
```

3 , , Node .

```
public class Node {
    public int data;
    public Node left;
    public Node right;

    public Node(int data){
        this.data = data;
    }
}
```

Node . .

```
public static void inOrderTraversal(Node root) {
```

```
        if (root != null) {
            inOrderTraversal(root.left); // traverse left sub tree
            System.out.print(root.data + " "); // traverse current node
            inOrderTraversal(root.right); // traverse right sub tree
        }
    }
```

.

Head Recursion Tail Recursion .

( ) .

. . .

head recursion . . . .

```
public void tail(int n)
{
   if(n == 1)
      return;
   else
      System.out.println(n);

   tail(n-1);
}
```

```
public void head(int n)
   {
     if(n == 0)
        return;
     else
        head(n-1);

      System.out.println(n);
    }
```

tail recursion . similar to a loop . method executes all the statements before jumping into the next recursive call .

beginning of a method, it is called a head recursion beginning of a method, it is called a head recursion beginning of a method, it is called a head recursion . method saves the state before jumping into the next recursive call .

:

StackOverflowError

StackOverflowError . Java . . (SO) .

```
public static void recursion(int depth) {
    if (depth > 0) {
        recursion(depth-1);
    }
}
```

(: recursion(50000) . , , -Xss JVM

. .

while "" . .

- ( )
-
-
-

.

```
public class Node {

    public int data;
    public Node left;
    public Node right;

    public Node(int data) {
        this(data, null, null);
    }

    public Node(int data, Node left, Node right) {
        this.data = data;
        this.left = left;
        this.right = right;
    }

    public void print(final int maxDepth) {
        if (maxDepth <= 0) {
            System.out.print("(...)");
        } else {
            System.out.print("(");
            if (left != null) {
                left.print(maxDepth-1);
            }
            System.out.print(data);
            if (right != null) {
                right.print(maxDepth-1);
            }
            System.out.print(")");
        }
    }

}
```

```
Node n = new Node(10, new Node(20, new Node(50), new Node(1)), new Node(30, new Node(42),
null));
n.print(2);
System.out.println();
```

```
(((...)20(...))10((...)30))
```

:

```
public class Frame {

    public final Node node;

    // 0: before printing anything
    // 1: before printing data
    // 2: before printing ")"
```

```
    public int state = 0;
    public final int maxDepth;

    public Frame(Node node, int maxDepth) {
        this.node = node;
        this.maxDepth = maxDepth;
    }

}
```

```
List<Frame> stack = new ArrayList<>();
stack.add(new Frame(n, 2)); // first frame = initial call

while (!stack.isEmpty()) {
    // get topmost stack element
    int index = stack.size() - 1;
    Frame frame = stack.get(index); // get topmost frame
    if (frame.maxDepth <= 0) {
        // termial case (too deep)
        System.out.print("(...)");
        stack.remove(index); // drop frame
    } else {
        switch (frame.state) {
            case 0:
                frame.state++;

                // do everything done before the first recursive call
                System.out.print("(");
                if (frame.node.left != null) {
                    // add new frame (recursive call to left and stop)
                    stack.add(new Frame(frame.node.left, frame.maxDepth - 1));
                    break;
                }
            case 1:
                frame.state++;

                // do everything done before the second recursive call
                System.out.print(frame.node.data);
                if (frame.node.right != null) {
                    // add new frame (recursive call to right and stop)
                    stack.add(new Frame(frame.node.right, frame.maxDepth - 1));
                    break;
                }
            case 2:
                // do everything after the second recursive call & drop frame
                System.out.print(")");
                stack.remove(index);
        }
    }
}
System.out.println();
```

: . / .

Java .

.

```
public static int add(int a, int b) {
    if (a == 0) {
        return b;
    } else {
        return add(a - 1, b + 1);  // TAIL CALL
    }
}
```

. `a add StackOverflowError` , Java 9 Java .

( ) . `add` ( ) . (tail-call elimination).

Java . (, .) `add` (), thread . `add(1000, 1) 1000 1001` .

Java . ("" .) . JVM `StackOverflowError` .

. JVM `Thread` . "". `StackOverflowError` .

- . `add` .

```
public static int add(int a, int b) {
    while (a != 0) {
       a = a - 1;
       b = b + 1;
    }
    return b;
}
```

(. .)

Java ()

Java . :

- `StackOverflowError` ( ) .
- .

John Rose "VM Tail "

" *API* . . *callee* . "

, (tail-call elimination) API .

: https://riptutorial.com/ko/java/topic/914/

158:

. Java `java.util.regex` . Java .

- patternName = Pattern.compile ();
- Matcher matcherName = patternName.matcher (textToSearch);
- matcherName.matches () // textToSearch true .
- matcherName.find () // textToSearch . .
- matcherName.group (groupNum) // .
- matcherName.group (groupName) // (Java 7) .

Regex .

```
import java.util.regex.Matcher
import java.util.regex.Pattern
```

. . .

*	0 .
+	1 .
?	0 1 .

regex .

Examples

regex .

.

```
\d{3}-\d{3}-\d{4}
```

. .

```
(\d{3})-(\d{3})-(\d{4})
^-----^ ^-----^ ^-----^
```

```
Group 1 Group 2 Group 3
```

Java .

```
"(\\d{3})-(\\d{3})-(\\d{4})"
```

Pattern Matcher :

```
Pattern phonePattern = Pattern.compile("(\\d{3})-(\\d{3})-(\\d{4})");
Matcher phoneMatcher = phonePattern.matcher("abcd800-555-1234wxyz");
```

, Matcher .

```
phoneMatcher.find();
```

.

```
String number = phoneMatcher.group(0); //"800-555-1234" (Group 0 is everything the regex
matched)
String aCode = phoneMatcher.group(1); //"800"
String threeDigit = phoneMatcher.group(2); //"555"
String fourDigit = phoneMatcher.group(3); //"1234"
```

: Matcher.group() Matcher.group(0) Matcher.group() .

Java SE 7

Java 7 . ( ) . .

.

```
(?<AreaCode>\d{3})-(\d{3})-(\d{4})
^----------------^ ^-----^ ^-----^
AreaCode           Group 2 Group 3
```

"AreaCode" .

```
String aCode = phoneMatcher.group("AreaCode"); //"800"
```

Pattern . String .

```
Pattern pattern = Pattern.compile("foo.", Pattern.CASE_INSENSITIVE | Pattern.DOTALL);
pattern.matcher("FOO\n").matches(); // Is true.

/* Had the regex not been compiled case insensitively and singlelined,
 * it would fail because FOO does not match /foo/ and \n (newline)
 * does not match /./.
 */

Pattern anotherPattern = Pattern.compile("(?si)foo");
anotherPattern.matcher("FOO\n").matches(); // Is true.
```

```
"foOt".replaceAll("(?si)foo", "ca"); // Returns "cat".
```

(`?+|`) . `\` . Java Strings `\\` .

.

```
"???".replaceAll ("?", "!"); //java.util.regex.PatternSyntaxException
"???".replaceAll ("\?", "!"); //Invalid escape sequence
```

.

```
"???".replaceAll ("\\?", "!"); //"!!!"
```

.

```
"a|b".split ("|"); // [a, |, b]
```

.

```
"a|b".split ("\\|"); // [a, b]
```

****

.

```
"\\".matches("\\"); // PatternSyntaxException
"\\".matches("\\\"); // Syntax Error
```

:

```
"\\".matches("\\\\"); // true
```

.

.

`\Q` . `\E` .

```
// the following throws a PatternSyntaxException because of the un-closed bracket
"[123".matches("[123");

// wrapping the bracket in \Q and \E allows the pattern to match as you would expect.
"[123".matches("\\Q[\\E123"); // returns true
```

`\Q \E` `Pattern.quote()` .

```
"[123".matches(Pattern.quote("[") + "123"); // returns true
```

lookahead :

: `(?!string-to-not-match)`

:

```
//not matching "popcorn"
String regexString = "^(?!popcorn).*$";
System.out.println("[popcorn] " + ("popcorn".matches(regexString) ? "matched!" : "nope!"));
System.out.println("[unicorn] " + ("unicorn".matches(regexString) ? "matched!" : "nope!"));
```

:

```
[popcorn] nope!
[unicorn] matched!
```

.

. '\\' .

Java . , . '\\\\' '\\' '\\' .

"C : \ dir \ myfile.txt" . `([A-Za-z]):\\(.*)` . .

.

```
    String path = "C:\\dir\\myfile.txt";
    System.out.println( "Local path: " + path ); // "C:\dir\myfile.txt"

    String regex = "([A-Za-z]):\\\\.*"; // Four to match one
    System.out.println("Regex:      " + regex ); // "([A-Za-z]):\\(.*)"

    Pattern pattern = Pattern.compile( regex );
    Matcher matcher = pattern.matcher( path );
    if ( matcher.matches()) {
        System.out.println( "This path is on drive " + matcher.group( 1 ) + ":.");
        // This path is on drive C:.
    }
```

8 4 *2* .

```
    String path = "\\\\myhost\\share\\myfile.txt";
    System.out.println( "UNC path: " + path ); // \\myhost\share\myfile.txt"

    String regex = "\\\\\\\\(.+?)\\\\(.*)"; // Eight to match two
    System.out.println("Regex:    " + regex ); // \\\\(.+?)\\(.*)

    Pattern pattern = Pattern.compile( regex );
    Matcher matcher = pattern.matcher( path );

    if ( matcher.matches()) {
        System.out.println( "This path is on host '" + matcher.group( 1 ) + "'.");
        // This path is on host 'myhost'.
    }
```

: https://riptutorial.com/ko/java/topic/135/-

159:

- int myVariable; //
- public static myMethod () {} //
- public static final double MY_CONSTANT; //
- public final double MY_CONSTANT; // ( )

Examples

static

static . . static ALL_CAPS .:

```
 static E STATIC_VARIABLE_NAME
```

final static .

, pi .

```
public class MathUtilities {

    static final double PI = 3.14159265358

}
```

(:

```
public class MathCalculations {

    //Calculates the circumference of a circle
    public double calculateCircumference(double radius) {
        return (2 * radius * MathUtilities.PI);
    }

}
```

Static . . , .

```
public class Apple {
    public static int test;
    public int test2;
}

Apple a = new Apple();
a.test = 1; // Warning
Apple.test = 1; // OK
Apple.test2 = 1; // Illegal: test2 is not static
a.test2 = 1; // OK
```

.

`this` !

```
public class Pineapple {

    private static int numberOfSpikes;
    private int age;

    public static getNumberOfSpikes() {
        return this.numberOfSpikes; // This doesn't compile
    }


    public static getNumberOfSpikes() {
        return numberOfSpikes; // This compiles
    }

}
```

(:) `static` `equals()` . Java `main` ., `this` `main()` .

. .

. . .

. . .

, . .

. () .

```
public class Week {
    static int daysOfTheWeek = 7; // static variable
    int dayOfTheWeek; // instance variable

    public static int getDaysLeftInWeek(){
        return Week.daysOfTheWeek-dayOfTheWeek; // this will cause errors
    }

    public int getDaysLeftInWeek(){
        return Week.daysOfTheWeek-dayOfTheWeek; // this is valid
    }

    public static int getDaysLeftInTheWeek(int today){
        return Week.daysOfTheWeek-today; // this is valid
    }

}
```

: https://riptutorial.com/ko/java/topic/2253/--

160:

Generics Java . , Java .

- class ArrayList <E> {} // E
- class HashMap <K, V> {} // 2 K V .
- <E> void print (E element) {} // E
- ArrayList <String> ; //
- ArrayList <?> ; //
- new ArrayList <String> () //
- ArrayList <> () // "diamond"(Java 7)

Java . , `List<String> names = new ArrayList<>();` `String` . `List<String>` `List<String>` / Java Reflection API

generic (: `(List<String>) list`) . `<String>` JVM `List<?>` `List<String>` . JVM `List` to `List` .

Examples

, .

`Param` (`<>`) `T` .

```
public class Param<T> {
    private T value;

    public T getValue() {
        return value;
    }

    public void setValue(T value) {
        this.value = value;
    }
}
```

, `T` **type** . : `Integer` :

```
Param<Integer> integerParam = new Param<Integer>();
```

type .

```
Param<String[]> stringArrayParam;
Param<int[][]> int2dArrayParam;
Param<Param<Object>> objectNestedParam;
```

Java SE 7 type *diamond* (`<>`) .

Java SE 7

```
Param<Integer> integerParam = new Param<>();
```

. . (JavaDocs .)
"type" `T` , "element" `E` , "key"/ "value" `K` / `V` .

```
public abstract class AbstractParam<T> {
    private T value;

    public T getValue() {
        return value;
    }

    public void setValue(T value) {
        this.value = value;
    }
}
```

`AbstractParam` type `T` . `<>` . `String` `Integer` type . type . `Object` . `AbstractParam` `ObjectParam` . (" ") .

```
public class Email extends AbstractParam<String> {
    // ...
}

public class Age extends AbstractParam<Integer> {
    // ...
}

public class Height<T> extends AbstractParam<T> {
    // ...
}

public class ObjectParam extends AbstractParam {
    // ...
}

public class MultiParam<T, E> extends AbstractParam<E> {
    // ...
}
```

.

```
Email email = new Email();
email.setValue("test@example.com");
String retrievedEmail = email.getValue();

Age age = new Age();
age.setValue(25);
Integer retrievedAge = age.getValue();
int autounboxedAge = age.getValue();

Height<Integer> heightInInt = new Height<>();
heightInInt.setValue(125);
```

```
Height<Float> heightInFloat = new Height<>();
heightInFloat.setValue(120.3f);

MultiParam<String, Double> multiParam = new MultiParam<>();
multiParam.setValue(3.3);
```

Email T getValue() String getValue() void setValue(T) void setValue(String) .

({}) .

```
AbstractParam<Double> height = new AbstractParam<Double>(){};
height.setValue(198.6);
```

.

Java . . :

```
public class MultiGenericParam<T, S> {
    private T firstParam;
    private S secondParam;

    public MultiGenericParam(T firstParam, S secondParam) {
        this.firstParam = firstParam;
        this.secondParam = secondParam;
    }

    public T getFirstParam() {
        return firstParam;
    }

    public void setFirstParam(T firstParam) {
        this.firstParam = firstParam;
    }

    public S getSecondParam() {
        return secondParam;
    }

    public void setSecondParam(S secondParam) {
        this.secondParam = secondParam;
    }
}
```

:

```
MultiGenericParam<String, String> aParam = new MultiGenericParam<String, String>("value1",
"value2");
MultiGenericParam<Integer, Double> dayOfWeekDegrees = new MultiGenericParam<Integer,
Double>(1, 2.6);
```

.

```
public class Example {

    // The type parameter T is scoped to the method
    // and is independent of type parameters of other methods.
    public <T> List<T> makeList(T t1, T t2) {
        List<T> result = new ArrayList<T>();
        result.add(t1);
        result.add(t2);
        return result;
    }

    public void usage() {
        List<String> listString = makeList("Jeff", "Atwood");
        List<Integer> listInteger = makeList(1, 2);
    }
}
```

. (:) . Y .

, , :

```
void usage() {
    consumeObjects(this.<Object>makeList("Jeff", "Atwood").stream());
}

void consumeObjects(Stream<Object> stream) { ... }
```

`stream()` `T Object makeList String` "" . (`this`).

Java SE 7

Java 7 [1] . Java 7 .

```
List<String> list = new LinkedList<>();
```

.

```
List<String> list = new LinkedList<String>();
```

(Anonymous Classes) type .

```
// This will compile:

Comparator<String> caseInsensitiveComparator = new Comparator<String>() {
    @Override
    public int compare(String s1, String s2) {
        return s1.compareToIgnoreCase(s2);
    }
};

// But this will not:

Comparator<String> caseInsensitiveComparator = new Comparator<>() {
    @Override
    public int compare(String s1, String s2) {
```

```
        return s1.compareToIgnoreCase(s2);
    }
};
```

Java SE 8

Java 7 8 **Java 9** .

:

1 - `<>` " ". . JLS () Java . `<>` Java . `<a>`, ( ) . JLS `<>` "" .

("extends A & B")

.

: `Number` `Comparable` .

```
public <T extends Number & Comparable<T>> void sortNumbers( List<T> n ) {
  Collections.sort( n );
}
```

`T` `Number` `Integer` `BigDecimal` "" `Striped64` `Comparable<T>` .

. , `<T extends Comparable<T> & Number>` Comparable .

. .

```
public abstract class Animal {
    public abstract String getSound();
}

public class Cat extends Animal {
    public String getSound() {
        return "Meow";
    }
}

public class Dog extends Animal {
    public String getSound() {
        return "Woof";
    }
}
```

.

```
public class AnimalContainer<T> {

    private Collection<T> col;

    public AnimalContainer() {
        col = new ArrayList<T>();
    }
```

```
    public void add(T t) {
        col.add(t);
    }

    public void printAllSounds() {
        for (T t : col) {
            // Illegal, type T doesn't have makeSound()
            // it is used as an java.lang.Object here
            System.out.println(t.makeSound());
        }
    }
}
```

.

```
public class BoundedAnimalContainer<T extends Animal> { // Note bound here.

    private Collection<T> col;

    public BoundedAnimalContainer() {
        col = new ArrayList<T>();
    }

    public void add(T t) {
        col.add(t);
    }

    public void printAllSounds() {
        for (T t : col) {
            // Now works because T is extending Animal
            System.out.println(t.makeSound());
        }
    }
}
```

generic .

```
// Legal
AnimalContainer<Cat> a = new AnimalContainer<Cat>();

// Legal
AnimalContainer<String> a = new AnimalContainer<String>();
```

```
// Legal because Cat extends Animal
BoundedAnimalContainer<Cat> b = new BoundedAnimalContainer<Cat>();

// Illegal because String doesn't extends Animal
BoundedAnimalContainer<String> b = new BoundedAnimalContainer<String>();
```

`T`,`? T`,`? T`

Java generics `?` :

- `? extends T` . T T .

- `? super T` . T 4 T .

,

- `? extends T` "" ("") `? extends T`
- `? super T` "" ("") `? super T`
- `T` `T` ("")

`extends` `super` (:) .

```
class Shoe {}
class IPhone {}
interface Fruit {}
class Apple implements Fruit {}
class Banana implements Fruit {}
class GrannySmith extends Apple {}

   public class FruitHelper {

        public void eatAll(Collection<? extends Fruit> fruits) {}

        public void addApple(Collection<? super Apple> apples) {}
}
```

:

```
 public class GenericsTest {
      public static void main(String[] args){
  FruitHelper fruitHelper = new FruitHelper() ;
    List<Fruit> fruits = new ArrayList<Fruit>();
    fruits.add(new Apple()); // Allowed, as Apple is a Fruit
    fruits.add(new Banana()); // Allowed, as Banana is a Fruit
    fruitHelper.addApple(fruits); // Allowed, as "Fruit super Apple"
    fruitHelper.eatAll(fruits); // Allowed

    Collection<Banana> bananas = new ArrayList<>();
    bananas.add(new Banana()); // Allowed
    //fruitHelper.addApple(bananas); // Compile error: may only contain Bananas!
    fruitHelper.eatAll(bananas); // Allowed, as all Bananas are Fruits

    Collection<Apple> apples = new ArrayList<>();
    fruitHelper.addApple(apples); // Allowed
    apples.add(new GrannySmith()); // Allowed, as this is an Apple
    fruitHelper.eatAll(apples); // Allowed, as all Apples are Fruits.

    Collection<GrannySmith> grannySmithApples = new ArrayList<>();
    fruitHelper.addApple(grannySmithApples); //Compile error: Not allowed.
                                  // GrannySmith is not a supertype of Apple
    apples.add(new GrannySmith()); //Still allowed, GrannySmith is an Apple
    fruitHelper.eatAll(grannySmithApples);//Still allowed, GrannySmith is a Fruit

    Collection<Object> objects = new ArrayList<>();
    fruitHelper.addApple(objects); // Allowed, as Object super Apple
    objects.add(new Shoe()); // Not a fruit
    objects.add(new IPhone()); // Not a fruit
    //fruitHelper.eatAll(objects); // Compile error: may contain a Shoe, too!
}
```

`T ? super T ? extends T` . . .

PECS .

P roducer " **E** xtends" **C** onsumer " **S** uper" .

( )

. .

Java . . .

.

```
List list = new ArrayList();
list.add("hello");
String s = (String) list.get(0);
```

.

```
List<String> list = new ArrayList<>();
list.add("hello");
String s = list.get(0);   // no cast
```

, , .

`T extends Type1 & Type2 & ...` .

`Flushable Closeable` Generic .

```
class ExampleClass<T extends Flushable & Closeable> {
}
```

`ExampleClass` `Flushable` `Closeable` .

```
ExampleClass<BufferedWriter> arg1; // Works because BufferedWriter implements both Flushable
and Closeable

ExampleClass<Console> arg4; // Does NOT work because Console only implements Flushable
ExampleClass<ZipFile> arg5; // Does NOT work because ZipFile only implements Closeable

ExampleClass<Flushable> arg2; // Does NOT work because Closeable bound is not satisfied.
```

```
ExampleClass<Closeable> arg3; // Does NOT work because Flushable bound is not satisfied.
```

Closeable Flushable .

```
class ExampleClass<T extends Flushable & Closeable> {
    /* Assign it to a valid type as you want. */
    public void test (T param) {
        Flushable arg1 = param; // Works
        Closeable arg2 = param; // Works too.
    }

    /* You can even invoke the methods of any valid type directly. */
    public void test2 (T param) {
        param.flush(); // Method of Flushable called on T and works fine.
        param.close(); // Method of Closeable called on T and works fine too.
    }
}
```

:

OR (|) . *AND* (&) . . .

.

```
public <T> void genericMethod() {
    T t = new T(); // Can not instantiate the type T.
}
```

T . JVM T .

---

1. genericMethod T .

```
public <T> void genericMethod(Class<T> cls) {
    try {
        T t = cls.newInstance();
    } catch (InstantiationException | IllegalAccessException e) {
         System.err.println("Could not instantiate: " + cls.getName());
    }
}
```

```
genericMethod(String.class);
```

throw.

Java SE 8

2. T :

```
public <T> void genericMethod(Supplier<T> cons) {
    T t = cons.get();
}
```

```
genericMethod(String::new);
```

.

? .

`DataSeries<T>` ( ) `T` . double `DoubleSeries extends DataSeries<Double>` `DoubleSeries extends DataSeries<Double>` . `DataSeries<T>` `add(values)` . `DataSeries<Double>` `DoubleSeries values` ?

( ).

```
public interface DataSeries<T, DS extends DataSeries<T, DS>> {
    DS add(DS values);
    List<T> data();
}
```

`T` (: `Double` `DS`). `Double` .

```
public interface DoubleSeries extends DataSeries<Double, DoubleSeries> {
    static DoubleSeries instance(Collection<Double> data) {
        return new DoubleSeriesImpl(data);
    }
}
```

IDE . .

```
class DoubleSeriesImpl implements DoubleSeries {
    private final List<Double> data;

    DoubleSeriesImpl(Collection<Double> data) {
        this.data = new ArrayList<>(data);
    }

    @Override
    public DoubleSeries add(DoubleSeries values) {
        List<Double> incoming = values != null ? values.data() : null;
        if (incoming == null || incoming.size() != data.size()) {
            throw new IllegalArgumentException("bad series");
        }
        List<Double> newdata = new ArrayList<>(data.size());
        for (int i = 0; i < data.size(); i++) {
            newdata.add(this.data.get(i) + incoming.get(i)); // beware autoboxing
        }
        return DoubleSeries.instance(newdata);
    }

    @Override
    public List<Double> data() {
        return Collections.unmodifiableList(data);
    }
}
```

`add` `DoubleSeries add(DoubleSeries values)` `DoubleSeries add(DoubleSeries values)` .

.

instanceof

instanceof

`<T>` `Example` .

```
class Example<T> {
    public boolean isTypeAString(String s) {
        return s instanceof T; // Compilation error, cannot use T as class type here
    }
}
```

Java , *(erasure)* T . `instanceof` ., .

`Example` , `Example<String>` `Example<Number>` .

```
class Example { // formal parameter is gone
    public boolean isTypeAString(String s) {
        return s instanceof Object; // Both <String> and <Number> are now Object
    }
}
```

JVM `T` .

(?) `instanceof` .

```
    public boolean isAList(Object obj) {
        return obj instanceof List<?>;
    }
```

`obj` `List` .

```
System.out.println(isAList("foo")); // prints false
System.out.println(isAList(new ArrayList<String>()); // prints true
System.out.println(isAList(new ArrayList<Float>()); // prints true
```

.

---

instanceof

`t` `T` `instanceof` :

```
class Example<T> {
    public boolean isTypeAString(T t) {
        return t instanceof String; // No compilation error this time
    }
}
```

.

```
class Example { // formal parameter is gone
    public boolean isTypeAString(Object t) {
        return t instanceof String; // No compilation error this time
    }
}
```

JVM (`Object`) .

```
Object obj1 = new String("foo"); // reference type Object, object type String
Object obj2 = new Integer(11); // reference type Object, object type Integer
System.out.println(obj1 instanceof String); // true
System.out.println(obj2 instanceof String); // false, it's an Integer, not a String
```

( )

.

```
public interface MyGenericInterface<T> {
    public void foo(T t);
}
```

.

`MyGenericClass` `<T>` .

```
public class NonGenericClass implements MyGenericInterface<String> {
    public void foo(String t) { } // type T has been replaced by String
}
```

`String` `MyGenericInterface` , `Integer` , `Object` `MyGenericInterface` . .

```
NonGenericClass myClass = new NonGenericClass();
myClass.foo("foo_string"); // OK, legal
myClass.foo(11); // NOT OK, does not compile
myClass.foo(new Object()); // NOT OK, does not compile
```

`MyGenericInterface` `<T>` .

```
public class MyGenericSubclass<T> implements MyGenericInterface<T> {
    public void foo(T t) { } // type T is still the same
    // other methods...
}
```

.

```
public class MyGenericSubclass<U> implements MyGenericInterface<U> { // equivalent to the
previous declaration
```

```
    public void foo(U t) { }
    // other methods...
}
```

`MyGenericInteface` ( ).

```
public class MyGenericSubclass implements MyGenericInterface {
    public void foo(Object t) { } // type T has been replaced by Object
    // other possible methods
}
```

( ) () *generics* 100 % . Java JVM (1.4) .

.

generics .

```
private Map<String, Object> data;
public <T> T get(String key) {
    return (T) data.get(key);
}
```

. Java .

```
Bar bar = foo.get("bar");
```

.

```
List<Bar> bars = foo.get("bars");
```

`List` (, `List<String>` () `ClassCastException`), `ClassCastException` . .

API .

```
public final static Key<List<Bar>> BARS = new Key<>("BARS");
```

`put()` :

```
public <T> T put(Key<T> key, T value);
```

.

:

-

( ). . .

. . Neal Gafter .

Java .

-
- Spring ParameterizedTypeReference
-

```
public class DataService<MODEL_TYPE> {
     private final DataDao dataDao = new DataDao();
     private final Class<MODEL_TYPE> type = (Class<MODEL_TYPE>) new TypeToken<MODEL_TYPE>
                                                                (getClass()){}.getRawType();
     public List<MODEL_TYPE> getAll() {
         return dataDao.getAllOfType(type);
    }
}

// the subclass definitively binds the parameterization to User
// for all instances of this class, so that information can be
// recovered at runtime
public class UserService extends DataService<User> {}

public class Main {
    public static void main(String[] args) {
          UserService service = new UserService();
          List<User> users = service.getAll();
    }
}
```

: https://riptutorial.com/ko/java/topic/92/

161:

- ;
- *expression1* : *expression2* ;

1	`false` assertion `AssertionError` .
2	. `AssertionError` Throw `AssertionError` .

.

`-ea` java .

```
java -ea com.example.AssertionExample
```

`false` . . .

Examples

assert

```
a = 1 - Math.abs(1 - a % 2);

// This will throw an error if my arithmetic above is wrong.
assert a >= 0 && a <= 1 : "Calculated value of " + a + " is outside of expected bounds";

return a;
```

: https://riptutorial.com/ko/java/topic/407/

162:

Java . . . . Java Nested Inner Class .

- OuterClass {public class InnerClass {}} // private .
- OuterClass {public static class StaticNestedClass {}} // private .
- public void method () {private class LocalClass {}} // .
- SomeClass anonymousClassInstance = SomeClass () {}; // . 'SomeClass ()' abstract ,
  .
- SomeInterface anonymousClassInstance = SomeInterface () {}; // .

Java (JLS) Java .

.

.

.

, . .

" " . " " ( ) .

.

- " ". . , .
- .
  - `private` . .
  - `private` .

  " " . " " .
- .
  - .
  - (`final` . (Java 8 .)
  - .

  . , .

Examples

```
public class IntStack {

    private IntStackNode head;

    // IntStackNode is the inner class of the class IntStack
```

```
    // Each instance of this inner class functions as one link in the
    // Overall stack that it helps to represent
    private static class IntStackNode {

        private int val;
        private IntStackNode next;

        private IntStackNode(int v, IntStackNode n) {
            val = v;
            next = n;
        }
    }

    public IntStack push(int v) {
        head = new IntStackNode(v, head);
        return this;
    }

    public int pop() {
        int x = head.val;
        head = head.next;
        return x;
    }
}
```

() .

```
public class Main {
    public static void main(String[] args) {

        IntStack s = new IntStack();
        s.push(4).push(3).push(2).push(1).push(0);

        //prints: 0, 1, 2, 3, 4,
        for(int i = 0; i < 5; i++) {
            System.out.print(s.pop() + ", ");
        }
    }
}
```

.

```
public class OuterClass1 {

    private static class StaticNestedClass {

    }

}
```

:

```
public class OuterClass2 {

    private class NestedClass {

    }
```

```
}
```

. . :

```
public class OuterClass1 {

    private int aField;
    public void aMethod(){}

    private static class StaticNestedClass {
        private int innerField;

        private StaticNestedClass() {
             innerField = aField; //Illegal, can't access aField from static context
             aMethod();           //Illegal, can't call aMethod from static context
        }

        private StaticNestedClass(OuterClass1 instance) {
             innerField = instance.aField; //Legal
        }

    }

    public static void aStaticMethod() {
        StaticNestedClass s = new StaticNestedClass(); //Legal, able to construct in static
context
        //Do stuff involving s...
    }

}

public class OuterClass2 {

    private int aField;

    public void aMethod() {}

    private class NestedClass {
        private int innerField;

        private NestedClass() {
             innerField = aField; //Legal
             aMethod(); //Legal
        }
    }

    public void aNonStaticMethod() {
        NestedClass s = new NestedClass(); //Legal
    }

    public static void aStaticMethod() {
        NestedClass s = new NestedClass(); //Illegal. Can't construct without surrounding
OuterClass2 instance.
                                           //As this is a static context, there is no
surrounding OuterClass2 instance
    }
}
```

.

. ( ) .

Java . Inner ?

public .

```
public class OuterClass {

    public class InnerClass {

        public int x = 5;

    }

    public InnerClass createInner() {
        return new InnerClass();
    }
}

public class SomeOtherClass {

    public static void main(String[] args) {
        int x = new OuterClass().createInner().x; //Direct field access is legal
    }
}
```

protected default ( ) .

private , . .java . , Outer Inner , private .

```
public class OuterClass {

    public class InnerClass {

        private int x;
        private void anInnerMethod() {}
    }

    public InnerClass aMethod() {
        InnerClass a = new InnerClass();
        a.x = 5; //Legal
        a.anInnerMethod(); //Legal
        return a;
    }
}
```

public . private () . .

```
public class OuterClass {

    private class InnerClass{}

    public InnerClass makeInnerClass() {
        return new InnerClass();
    }
```

```
}

public class AnotherClass {

    public static void main(String[] args) {
        OuterClass o = new OuterClass();

        InnerClass x = o.makeInnerClass(); //Illegal, can't find type
        OuterClass.InnerClass x = o.makeInnerClass(); //Illegal, InnerClass has visibility
private
        Object x = o.makeInnerClass(); //Legal
    }
}
```

. . .

. . :

```
public static Comparator<String> CASE_INSENSITIVE =
        new Comparator<String>() {
            @Override
            public int compare(String string1, String string2) {
                return string1.toUpperCase().compareTo(string2.toUpperCase());
            }
        };
```

`Comparator<String>` (`CASE_INSENSITIVE`) .

`Runnable` `Callable` . :

```
// An anonymous Runnable class is used to provide an instance that the Thread
// will run when started.
Thread t = new Thread(new Runnable() {
        @Override
        public void run() {
              System.out.println("Hello world");
        }
    });
t.start();  // Prints "Hello world"
```

. `extends` `extends` . abstract . .

. `super(...)` . :

```
SomeClass anon = new SomeClass(1, "happiness") {
            @Override
            public int someMethod(int arg) {
                // do something
            }
        };
```

`SomeClass` `SomeClass(int, String)` `SomeClass` . . throw throw.

. no args `java.lang.Object` .

. .

.

:

```
public class OuterClass {
    private void outerMethod() {
       final int outerInt = 1;
        // Method Local Inner Class
        class MethodLocalInnerClass {
            private void print() {
                System.out.println("Method local inner class " + outerInt);
            }
        }
        // Accessing the inner class
        MethodLocalInnerClass inner = new MethodLocalInnerClass();
        inner.print();
    }

    public static void main(String args[]) {
        OuterClass outer = new OuterClass();
        outer.outerMethod();
    }
}
```

. Method local inner class 1 .

, this

```
public class OuterClass {
    public class InnerClass {
        public void method() {
            System.out.println("I can access my enclosing class: " + OuterClass.this);
        }
    }
}
```

.

```
public class OuterClass {
    private int counter;

    public class InnerClass {
        public void method() {
            System.out.println("I can access " + counter);
        }
    }
}
```

.

```
public class OuterClass {
    private int counter;

    public class InnerClass {
```

```
        private int counter;

        public void method() {
            System.out.println("My counter: " + counter);
            System.out.println("Outer counter: " + OuterClass.this.counter);

            // updating my counter
            counter = OuterClass.this.counter;
        }
    }
}
```

.

. new .

```
class OuterClass {

    class InnerClass {
    }
}

class OutsideClass {

    OuterClass outer = new OuterClass();

    OuterClass.InnerClass createInner() {
        return outer.new InnerClass();
    }
}
```

outer.new outer.new .

: https://riptutorial.com/ko/java/topic/3317/----

163:

java.util.Map . . .

Map . Map , java.util.HashMap java.util.TreeMap .

. / . .

- / .
- . ., .
- / .

HashMap . .

HashMap . / . .

- TreeMap (Comparator) . , .
- LinkedHashMap .

Examples

1.

```
Map<Integer, String> map = new HashMap<>();
map.put(1, "First element.");
System.out.println(map.get(1));
```

: First element.

2.

```
Map<Integer, String> map = new HashMap<>();
map.put(1, "First element.");
map.put(1, "New element.");
System.out.println(map.get(1));
```

: New element.

HashMap . Map .

V put(K key,V value) .

(). .

```
String currentVal;
Map<Integer, String> map = new TreeMap<>();
currentVal = map.put(1, "First element.");
System.out.println(currentVal);// Will print null
currentVal = map.put(2, "Second element.");
```

```
System.out.println(currentVal); // Will print null yet again
currentVal = map.put(2, "This will replace 'Second element'");
System.out.println(currentVal); // will print Second element.
System.out.println(map.size()); // Will print 2 as key having
// value 2 was replaced.

Map<Integer, String> map2 = new HashMap<>();
map2.put(2, "Element 2");
map2.put(3, "Element 3");

map.putAll(map2);

System.out.println(map.size());
```

:

```
3
```

.

```
Map<Integer, String> map = new HashMap<>() {{
    // This is now an anonymous inner class with an unnamed instance constructor
    put(5, "high");
    put(4, "low");
    put(1, "too slow");
}};
```

.

```
static Map<Integer, String> map = new HashMap<>();

static {
    // Now no inner classes are created so we can avoid memory leaks
    put(5, "high");
    put(4, "low");
    put(1, "too slow");
}
```

. static .

putAll . putAll .

```
another.putAll(one);
```

Java 8 Map

Java 8

1. **getOrDefault**

.

```
Map<Integer, String> map = new HashMap<>();
map.put(1, "First element");
```

```
map.get(1);                                 // => First element
map.get(2);                                 // => null
map.getOrDefault(2, "Default element");     // => Default element
```

2. **forEach**

" .

```
  Map<Integer, String> map = new HashMap<Integer, String>();
  map.put(1, "one");
  map.put(2, "two");
  map.put(3, "three");
  map.forEach((key, value) -> System.out.println("Key: "+key+ " :: Value: "+value));

   // Key: 1 :: Value: one
   // Key: 2 :: Value: two
   // Key: 3 :: Value: three
```

3. **replaceAll**

.

```
 Map<String, Integer> map = new HashMap<String, Integer>();
 map.put("john", 20);
 map.put("paul", 30);
 map.put("peter", 40);
 map.replaceAll((key,value)->value+10);   //{john=30, paul=40, peter=50}
```

4. **putIfAbsent**

null - .

```
    Map<String, Integer> map = new HashMap<String, Integer>();
    map.put("john", 20);
    map.put("paul", 30);
    map.put("peter", 40);
    map.putIfAbsent("kelly", 50);     //{john=20, paul=30, peter=40, kelly=50}
```

5. .

```
Map<String, Integer> map = new HashMap<String, Integer>();
map.put("john", 20);
map.put("paul", 30);
map.put("peter", 40);
map.remove("peter",40); //{john=30, paul=40}
```

6. . .

```
Map<String, Integer> map = new HashMap<String, Integer>();
map.put("john", 20);
map.put("paul", 30);
map.put("peter", 40);
map.replace("peter",50); //{john=20, paul=30, peter=50}
map.replace("jack",60); //{john=20, paul=30, peter=50}
```

7. **computeIfAbsent**

.

```
    Map<String, Integer> map = new HashMap<String, Integer>();
    map.put("john", 20);
    map.put("paul", 30);
    map.put("peter", 40);
    map.computeIfAbsent("kelly", k->map.get("john")+10); //{john=20, paul=30, peter=40,
kelly=30}
    map.computeIfAbsent("peter", k->map.get("john")+10); //{john=20, paul=30, peter=40,
kelly=30} //peter already present
```

8. **computeIfPresent**

. .

```
    Map<String, Integer> map = new HashMap<String, Integer>();
    map.put("john", 20);
    map.put("paul", 30);
    map.put("peter", 40);
    map.computeIfPresent("kelly", (k,v)->v+10); //{john=20, paul=30, peter=40} //kelly not
present
    map.computeIfPresent("peter", (k,v)->v+10); //{john=20, paul=30, peter=50} // peter
present, so increase the value
```

9.

.

```
    Map<String, Integer> map = new HashMap<String, Integer>();
    map.put("john", 20);
    map.put("paul", 30);
    map.put("peter", 40);
    map.compute("peter", (k,v)->v+50); //{john=20, paul=30, peter=90} //Increase the value
```

10.

, null , . , . null , .

```
    Map<String, Integer> map = new HashMap<String, Integer>();
    map.put("john", 20);
    map.put("paul", 30);
    map.put("peter", 40);

    //Adds the key-value pair to the map, if key is not present or value for the key is null
    map.merge("kelly", 50 , (k,v)->map.get("john")+10); // {john=20, paul=30, peter=40,
kelly=50}

    //Replaces the value with the newly computed value, if the key is present
    map.merge("peter", 50 , (k,v)->map.get("john")+10); //{john=20, paul=30, peter=30,
kelly=50}

    //Key is removed from the map , if new value computed is null
    map.merge("peter", 30 , (k,v)->map.get("nancy")); //{john=20, paul=30, kelly=50}
```

```
Map<Integer, String> map = new HashMap<>();

map.put(1, "First element.");
map.put(2, "Second element.");
map.put(3, "Third element.");

map.clear();

System.out.println(map.size());   // => 0
```

, - . . :

```
Map<String, Integer> repMap = new HashMap<>();
repMap.put("Jon Skeet", 927_654);
repMap.put("BalusC", 708_826);
repMap.put("Darin Dimitrov", 715_567);
```

:

```
for (String key : repMap.keySet()) {
    System.out.println(key);
}
```

:

C

`keySet()` `Set` . `Set` . . HashMaps , .

:

```
for (Integer value : repMap.values()) {
    System.out.println(value);
}
```

:

715567
927654
708826

`values()` `Collection` . ., .

```
for (Map.Entry<String, Integer> entry : repMap.entrySet()) {
    System.out.printf("%s = %d\n", entry.getKey(), entry.getValue());
}
```

:

= 715567
Jon Skeet = 927654
BalusC = 708826

`entrySet()` `Map.Entry` . Map.Entry .

,

`putAll` . .

```
Map<String, Integer> numbers = new HashMap<>();
numbers.put("One", 1)
numbers.put("Three", 3)
Map<String, Integer> other_numbers = new HashMap<>();
other_numbers.put("Two", 2)
other_numbers.put("Three", 4)

numbers.putAll(other_numbers)
```

`numbers` `numbers` .

```
"One" -> 1
"Two" -> 2
"Three" -> 4 //old value 3 was overwritten by new value 4
```

`BiFunction` Java 8 `Map.merge` . `merge` `Map.forEach` . .

```
for (Map.Entry<String, Integer> e : other_numbers.entrySet())
    numbers.merge(e.getKey(), e.getValue(), Integer::sum);
//or instead of the above loop
other_numbers.forEach((k, v) -> numbers.merge(k, v, Integer::sum));
```

`AssertionError` throw .

```
mapA.forEach((k, v) ->
    mapB.merge(k, v, (v1, v2) ->
        {throw new AssertionError("duplicate values for key: "+k);}));
```

Map <X, Y> Map <Y, Z> Map <X, Z>

.

```
    Map<String, Integer> map1 = new HashMap<String, Integer>();
    map1.put("key1", 1);
    map1.put("key2", 2);
    map1.put("key3", 3);

    Map<Integer, Double> map2 = new HashMap<Integer, Double>();
    map2.put(1, 1.0);
    map2.put(2, 2.0);
    map2.put(3, 3.0);
```

```
    Map<String, Double> map3 = new new HashMap<String, Double>();
    map1.forEach((key,value)->map3.put(key,map2.get(value)));
```

.

```
    "key1" -> 1.0
    "key2" -> 2.0
    "key3" -> 3.0
```

.

```
Map<String, String> num = new HashMap<>();
num.put("one", "first");

if (num.containsKey("one")) {
    System.out.println(num.get("one")); // => first
}
```

null .

" " "" . , HashMap null . , .

```
Map<String, String> map = new HashMap<>();
map.put("one", null);
if (map.containsKey("one")) {
    System.out.println("This prints !"); // This line is reached
}
if (map.get("one") != null) {
    System.out.println("This is never reached !"); // This line is never reached
}
```

map.contains(key) <=> map.get(key)!=null map.contains(key) <=> map.get(key)!=null

Map<Integer, Integer> Integer 10 . O(n) "" .

1. **Map.Entry Iterator**

```
    Iterator<Map.Entry<Integer, Integer>> it = map.entrySet().iterator();
    while (it.hasNext()) {
        Map.Entry<Integer, Integer> pair = it.next();
        sum += pair.getKey() + pair.getValue();
    }
```

2. **Map.Entry for**

```
    for (Map.Entry<Integer, Integer> pair : map.entrySet()) {
        sum += pair.getKey() + pair.getValue();
    }
```

3. `Map.forEach` (Java 8)

```
map.forEach((k, v) -> sum[0] += k + v);
```

4. `for` `Map.keySet`

```
for (Integer key : map.keySet()) {
    sum += key + map.get(key);
}
```

5. `Iterator` `Map.keySet`

```
Iterator<Integer> it = map.keySet().iterator();
while (it.hasNext()) {
    Integer key = it.next();
    sum += key + map.get(key);
}
```

6. `for` `Iterator` `Map.Entry`

```
for (Iterator<Map.Entry<Integer, Integer>> entries =
         map.entrySet().iterator(); entries.hasNext(); ) {
    Map.Entry<Integer, Integer> entry = entries.next();
    sum += entry.getKey() + entry.getValue();
}
```

7. `Stream.forEach` (Java 8)

```
map.entrySet().stream().forEach(e -> sum += e.getKey() + e.getValue());
```

8. `Stream.forEach` `Stream.parallel` (Java 8)

```
map.entrySet()
   .stream()
   .parallel()
   .forEach(e -> sum += e.getKey() + e.getValue());
```

9. Apache `IterableMap`

```
MapIterator<Integer, Integer> mit = iterableMap.mapIterator();
while (mit.hasNext()) {
    sum += mit.next() + it.getValue();
}
```

10. `MutableMap`

```
 mutableMap.forEachKeyValue((key, value) -> {
     sum += key + value;
 });
```

(*Github*)

: Windows 8.1 64 , Intel i7-4790 3.60GHz, 16GB

1. 10 (100) : 308 ± 21ns / op

```
Benchmark                              Score     Error  Units
test3_UsingForEachAndJava8               308  ±     21  ns/op
test10_UsingEclipseMutableMap            309  ±      9  ns/op
test1_UsingWhileAndMapEntry              380  ±     14  ns/op
test6_UsingForAndIterator                387  ±     16  ns/op
test2_UsingForEachAndMapEntry            391  ±     23  ns/op
test7_UsingJava8StreamAPI                510  ±     14  ns/op
test9_UsingApacheIterableMap             524  ±      8  ns/op
test4_UsingKeySetAndForEach              816  ±     26  ns/op
test5_UsingKeySetAndIterator             863  ±     25  ns/op
test8_UsingJava8StreamAPIParallel       5552  ±    185  ns/op
```

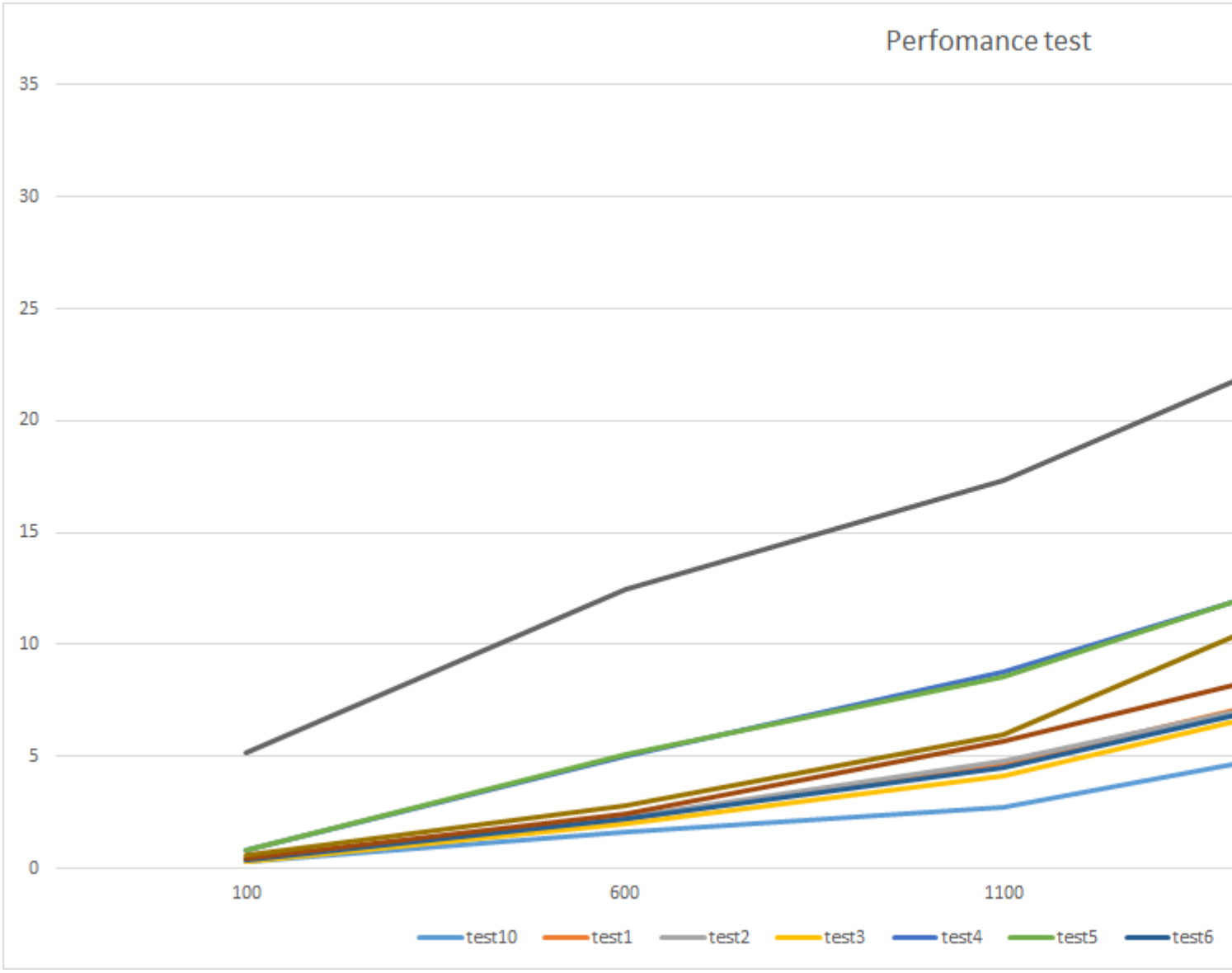
2. 10 (10000) : 37.606 ± 0.790µs / op

```
Benchmark                              Score     Error  Units
test10_UsingEclipseMutableMap          37606  ±    790  ns/op
test3_UsingForEachAndJava8             50368  ±    887  ns/op
test6_UsingForAndIterator              50332  ±    507  ns/op
test2_UsingForEachAndMapEntry          51406  ±   1032  ns/op
test1_UsingWhileAndMapEntry            52538  ±   2431  ns/op
test7_UsingJava8StreamAPI              54464  ±    712  ns/op
test4_UsingKeySetAndForEach            79016  ±  25345  ns/op
test5_UsingKeySetAndIterator           91105  ±  10220  ns/op
test8_UsingJava8StreamAPIParallel     112511  ±    365  ns/op
test9_UsingApacheIterableMap          125714  ±   1935  ns/op
```

3. 10 (100000) : 1184.767 ± 332.968 µs / op

```
Benchmark                                Score       Error  Units
test1_UsingWhileAndMapEntry           1184.767  ±  332.968  µs/op
test10_UsingEclipseMutableMap         1191.735  ±  304.273  µs/op
test2_UsingForEachAndMapEntry         1205.815  ±  366.043  µs/op
test6_UsingForAndIterator             1206.873  ±  367.272  µs/op
test8_UsingJava8StreamAPIParallel     1485.895  ±  233.143  µs/op
test5_UsingKeySetAndIterator          1540.281  ±  357.497  µs/op
test4_UsingKeySetAndForEach           1593.342  ±  294.417  µs/op
test3_UsingForEachAndJava8            1666.296  ±  126.443  µs/op
test7_UsingJava8StreamAPI             1706.676  ±  436.867  µs/op
test9_UsingApacheIterableMap          3289.866  ± 1445.564  µs/op
```

4.




```
       x: Size of Map
    f(x): Benchmark Score (µs/op)

                 100      600     1100     1600     2100
        ---------------------------------------------------
        10   |   0.333    1.631    2.752    5.937    8.024
         3   |   0.309    1.971    4.147    8.147   10.473
         6   |   0.372    2.190    4.470    8.322   10.531
         1   |   0.405    2.237    4.616    8.645   10.707
Tests    2   |   0.376    2.267    4.809    8.403   10.910
 f(x)    7   |   0.473    2.448    5.668    9.790   12.125
         9   |   0.565    2.830    5.952    13.22   16.965
         4   |   0.808    5.012    8.813   13.939   17.407
         5   |   0.81     5.104    8.533   14.064   17.422
         8   |   5.173   12.499   17.351    24.671  30.403
```

hashCode () equals () .

.

```
class MyKey {
    private String name;
```

```
    MyKey(String name) {
        this.name = name;
    }

    @Override
    public boolean equals(Object obj) {
        if(obj instanceof MyKey) {
            return this.name.equals(((MyKey)obj).name);
        }
        return false;
    }

    @Override
    public int hashCode() {
        return this.name.hashCode();
    }
}
```

`hashCode` `equals` .

, .

HashMap

HashMap - Map .

1. HashMap

```
Map<KeyType, ValueType> myMap = new HashMap<KeyType, ValueType>();
```

KeyType ValueType String, Integer, Float Employee, Student Java .

: `Map<String,Integer> myMap = new HashMap<String,Integer>();`

2. HashMap .

HashMap Key Value HashMap `put` .

```
myMap.put("key1", 1);
myMap.put("key2", 2);
```

Map Key put .

3. HashMap .

HashMap `get` `get` .

```
myMap.get("key1");    //return 1 (class Integer)
```

HashMap , `null` `null`

4. .

```
myMap.containsKey(varKey);
```

5. .

```
myMap.containsValue(varValue);
```

`boolean` , true false .

`Maps` / . .

`Maps` . Java .

- 1 : -

```
/*J2SE < 5.0*/
Map map = new HashMap();
map.put("name", "A");
map.put("address", "Malviya-Nagar");
map.put("city", "Jaipur");
System.out.println(map);
```

- 2 : -

```
/*J2SE 5.0+ style (use of generics):*/
Map<String, Object> map = new HashMap<>();
map.put("name", "A");
map.put("address", "Malviya-Nagar");
map.put("city", "Jaipur");
System.out.println(map);
```

- 3 : -

```
Map<String, Object> map = new HashMap<String, Object>(){{
    put("name", "A");
    put("address", "Malviya-Nagar");
    put("city", "Jaipur");
}};
System.out.println(map);
```

- 4 : -

```
Map<String, Object> map = new TreeMap<String, Object>();
    map.put("name", "A");
    map.put("address", "Malviya-Nagar");
    map.put("city", "Jaipur");
System.out.println(map);
```

- 5 : -

```
//Java 8
```

```
  final Map<String, String> map =
      Arrays.stream(new String[][] {
          { "name", "A" },
          { "address", "Malviya-Nagar" },
          { "city", "jaipur" },
      }).collect(Collectors.toMap(m -> m[0], m -> m[1]));
  System.out.println(map);
```

- 6 : -

```
  //This way for initial a map in outside the function
  final static Map<String, String> map;
  static
  {
      map = new HashMap<String, String>();
      map.put("a", "b");
      map.put("c", "d");
  }
```

- Way 7 : - - .

```
    //Immutable single key-value map
    Map<String, String> singletonMap = Collections.singletonMap("key", "value");
```

.

(), UnsupportedOperationException Throw.

```
    //Immutable single key-value pair
    Map<String, String> singletonMap = Collections.singletonMap("key", "value");
    singletonMap.put("newKey", "newValue"); //will throw UnsupportedOperationException
    singletonMap.putAll(new HashMap<>()); //will throw UnsupportedOperationException
    singletonMap.remove("key"); //will throw UnsupportedOperationException
    singletonMap.replace("key", "value", "newValue"); //will throw
UnsupportedOperationException
    //and etc
```

: https://riptutorial.com/ko/java/topic/105/

164:

Java . .

.

Examples

Java

?

( ) . . , Java RMI, , , JVM () . JVM .

`java.io.Serializable` ( ). Java "" .

`serialVersionUID` . . `serialVersionUID` deserialization `InvalidClassException` . `static, final, long serialVersionUID` `serialVersionUID` .

```
ANY-ACCESS-MODIFIER static final long serialVersionUID = 1L;
```

`java.io.Serializable` .

```
import java.io.Serializable;

public class SerialClass implements Serializable {

    private static final long serialVersionUID = 1L;
    private Date currentTime;

    public SerialClass() {
        currentTime = Calendar.getInstance().getTime();
    }

    public Date getCurrentTime() {
        return currentTime;
    }
}
```

. `java.io.ObjectOutputStream` .

```
import java.io.FileOutputStream;
import java.io.ObjectOutputStream;
import java.io.IOException;

public class PersistSerialClass {

    public static void main(String [] args) {
        String filename = "time.ser";
        SerialClass time = new SerialClass(); //We will write this object to file system.
        try {
            ObjectOutputStream out = new ObjectOutputStream(new FileOutputStream(filename));
            out.writeObject(time); //Write byte stream to file system.
```

```
            out.close();
        } catch(IOException ex){
            ex.printStackTrace();
        }
     }
 }
```

java.io.ObjectInputStream .

```
import java.io.FileInputStream;
import java.io.ObjectInputStream;
import java.io.IOException;
import java.io.java.lang.ClassNotFoundException;

 public class ReadSerialClass {

    public static void main(String [] args) {
        String filename = "time.ser";
        SerialClass time = null;

        try {
            ObjectInputStream in = new ObjectInputStream(new FileInputStream(filename));
            time = (SerialClass)in.readObject();
            in.close();
        } catch(IOException ex){
            ex.printStackTrace();
        } catch(ClassNotFoundException cnfe){
            cnfe.printStackTrace();
        }
        // print out restored time
        System.out.println("Restored time: " + time.getTime());
     }
 }
```

2 . . Java Serialization Specification .

. reset() ObjectOutputStream .

-

Gson

Gson JSON .

```
public class Employe {

    private String firstName;
    private String lastName;
    private int age;
    private BigDecimal salary;
    private List<String> skills;

    //getters and setters
}
```

()

```
//Skills
List<String> skills = new LinkedList<String>();
skills.add("leadership");
skills.add("Java Experience");

//Employe
Employe obj = new Employe();
obj.setFirstName("Christian");
obj.setLastName("Lusardi");
obj.setAge(25);
obj.setSalary(new BigDecimal("10000"));
obj.setSkills(skills);

//Serialization process
Gson gson = new Gson();
String json = gson.toJson(obj);
//{"firstName":"Christian","lastName":"Lusardi","age":25,"salary":10000,"skills":["leadership","Java
Experience"]}
```

.

( )

```
//it's very simple...
//Assuming that json is the previous String object....

Employe obj2 = gson.fromJson(json, Employe.class); // obj2 is just like obj
```

Jackson 2

JSON .

```
class Test {

    private int idx;
    private String name;

    public int getIdx() {
        return idx;
    }

    public void setIdx(int idx) {
        this.idx = idx;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }
}
```

:

```
Test test = new Test();
test.setIdx(1);
test.setName("abc");

ObjectMapper mapper = new ObjectMapper();

String jsonString;
try {
    jsonString = mapper.writerWithDefaultPrettyPrinter().writeValueAsString(test);
    System.out.println(jsonString);
} catch (JsonProcessingException ex) {
    // Handle Exception
}
```

:

```
{
  "idx" : 1,
  "name" : "abc"
}
```

Default Pretty Printer .

.

```
<dependency>
    <groupId>com.fasterxml.jackson.core</groupId>
    <artifactId>jackson-databind</artifactId>
    <version>2.6.3</version>
</dependency>
```

.

```
    public class SimpleRangeRandom implements Runnable {
    private int min;
    private int max;

    private Thread thread;

    public SimpleRangeRandom(int min, int max){
        this.min = min;
        this.max = max;
        thread = new Thread(this);
        thread.start();
    }

    @Override
 private void WriteObject(ObjectOutputStreamout) throws IO Exception;
    private void ReadObject(ObjectInputStream in) throws IOException, ClassNotFoundException;
    public void run() {
        while(true) {
            Random rand = new Random();
            System.out.println("Thread: " + thread.getId() + " Random:" + rand.nextInt(max -
min));
            try {
                Thread.sleep(10000);
            } catch (InterruptedException e) {
```

```
                e.printStackTrace();
            }
        }
    }
    }
```

Serializable . . . . . ***readObject ()*** . **serialization** . .

```
private void writeObject(ObjectOutputStream out) throws IOException;
private void readObject(ObjectInputStream in) throws IOException, ClassNotFoundException;
```

readObject () :

```
class RangeRandom implements Serializable, Runnable {

private int min;
private int max;

private transient Thread thread;
//transient should be any field that either cannot be serialized e.g Thread or any field you
do not want serialized

public RangeRandom(int min, int max){
    this.min = min;
    this.max = max;
    thread = new Thread(this);
    thread.start();
}

@Override
public void run() {
    while(true) {
        Random rand = new Random();
        System.out.println("Thread: " + thread.getId() + " Random:" + rand.nextInt(max -
min));
        try {
            Thread.sleep(10000);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }
}

private void writeObject(ObjectOutputStream oos) throws IOException {
    oos.defaultWriteObject();
}

private void readObject(ObjectInputStream in) throws IOException, ClassNotFoundException {
    in.defaultReadObject();
    thread = new Thread(this);
    thread.start();
}
}
```

.

```
public class Main {
```

```
public static void main(String[] args) {
    System.out.println("Hello");
    RangeRandom rangeRandom = new RangeRandom(1,10);

    FileOutputStream fos = null;
    ObjectOutputStream out = null;
    try
    {
        fos = new FileOutputStream("test");
        out = new ObjectOutputStream(fos);
        out.writeObject(rangeRandom);
        out.close();
    }
    catch(IOException ex)
    {
        ex.printStackTrace();
    }


    RangeRandom rangeRandom2 = null;
       FileInputStream fis = null;
       ObjectInputStream in = null;
       try
       {
             fis = new FileInputStream("test");
             in = new ObjectInputStream(fis);
           rangeRandom2 = (RangeRandom)in.readObject();
             in.close();
           }
       catch(IOException ex)
       {
             ex.printStackTrace();
           }
       catch(ClassNotFoundException ex)
       {
             ex.printStackTrace();
           }

}
}
```

() , RangeRandom 2 thread . ***Thread.start () readObject ()*** .

serialVersionUID

`java.io.Serializable` `long serialVersionUID static final .` `serialVersionUID` .
Sun . .

. `serialVersionUID` `serialVersionUID` . `InvalidClassException` .

serialize long final, finalVersionUID . . **'serialVersionUID'**
`InvalidClassException` .

```
public class Example implements Serializable {
    static final long serialVersionUID = 1L /*or some other value*/;
    //...
}
```

`serialVersionUID` Java . .

- **:** , . , readObject .
- **:** . . .
- **:** . . , . .
- **writeObject / readObject :** , readObject, , . defaultReadObject . writeObject defaultWriteObject .
- **java.io.Serializable :** . . . InvalidClassException throw.
- **:** public, package, protected private .
- **static nonstatic transient nontransient :** . .

- **:** . . .
- **:** .
- **transient :** . . .
- **:** . .
- writeObject readObject . .
- Serializable Externalizable .
- enum enum .
- Serializable Externalizable . .
- writeReplace readResolve .

Jackson JSON

rest API JSON POJO . Jackson org.codehaus.jackson.map.ObjectMapper " " . deserializer deserializer .

.

```
public class User {
    private Long id;
    private String name;
    private String email;

    //getter setter are omitted for clarity
}
```

```
public class Program {
    private Long id;
    private String name;
    private User createdBy;
    private String contents;

    //getter setter are omitted for clarity
}
```

/ .

```
User user = new User();
user.setId(1L);
user.setEmail("example@example.com");
```

```
user.setName("Bazlur Rahman");

Program program = new Program();
program.setId(1L);
program.setName("Program @# 1");
program.setCreatedBy(user);
program.setContents("Some contents");

ObjectMapper objectMapper = new ObjectMapper();
```

final String json = objectMapper.writeValueAsString (program); System.out.println (json);

JSON-

```
{
    "id": 1,
    "name": "Program @# 1",
    "createdBy": {
        "id": 1,
        "name": "Bazlur Rahman",
        "email": "example@example.com"
    },
    "contents": "Some contents"
}
```

. JSON ObjectMapper .

. `Program` API JSON .

```
{
"id": 1,
"name": "Program @# 1",
"ownerId": 1
"contents": "Some contents"
}
```

JSON owenerId .

JSON .

.

.

`onwerId` . Program .

```
@JsonIgnoreProperties(ignoreUnknown = true)
public class Program {}
```

`owerId` . `User` id .

.

, `JsonNode` `JonsParser` . `get()` `JsonNode` . . .

`ObjectMapper` ProgramDeserializer .

```
ObjectMapper mapper = new ObjectMapper();
SimpleModule module = new SimpleModule();
module.addDeserializer(Program.class, new ProgramDeserializer());

mapper.registerModule(module);

String newJsonString = "{\"id\":1,\"name\":\"Program @# 1\",\"ownerId\":1,\"contents\":\"Some
contents\"}";
final Program program2 = mapper.readValue(newJsonString, Program.class);
```

.

```
@JsonDeserialize(using = ProgramDeserializer.class)
public class Program {
}
```

: https://riptutorial.com/ko/java/topic/767/

165:

Examples

```
Object obj = new Object(); // Note the 'new' keyword
```

:

- Object .
- obj .
- Object() Object .

:

- .
- Object() .
- obj .

.

```
int i = 10;
```

10 i .

. :

```
Object obj = new Object();
String text = obj.toString(); // 'obj' is dereferenced.
```

. , (toString).

---

null , NullPointerException .

```
Object obj = null;
obj.toString(); // Throws a NullpointerException when this statement is executed.
```

null , . .

: https://riptutorial.com/ko/java/topic/1046/--

166:

Examples

`java.lang.ref` .

Java . :

-
-
-
-

1.

.

```
MyObject myObject = new MyObject();
```

. live `MyObject` `MyObject` .

2.

.

```
WeakReference myObjectRef = new WeakReference(MyObject);
```

, . .

`.get()` .

```
myObjectRef.get();
```

.

```
WeakReference myObjectRef = new WeakReference(MyObject);
System.out.println(myObjectRef.get()); // This will print the object reference address
System.gc();
System.out.println(myObjectRef.get()); // This will print 'null' if the GC cleaned up the
object
```

3.

. .

```
SoftReference myObjectRef = new SoftReference(MyObject);
```

. / .

. . . .

```
SoftReference myObjectRef = new SoftReference(MyObject);
System.out.println(myObjectRef.get()); // This will print the reference address of the Object
System.gc();
System.out.println(myObjectRef.get()); // This may or may not print the reference address of
the Object
```

4.

. Phantom Reference `get()` null !

" . . ." - Phantom Reference Javadoc .

Phantom Reference :

```
PhantomReference myObjectRef = new PhantomReference(MyObject);
```

: https://riptutorial.com/ko/java/topic/4017/-

167:

. . , OOP ( ) . .

`private` `public` .

. "" ( ).

```
public class DumbData {
    public String name;
    public int timeStamp;
    public int value;
}
```

, .

`final` `public` .

Examples

.

. public .

. .

. .

, .

```
public class Angle {

    private double angleInDegrees;
    private double angleInRadians;

    public static Angle angleFromDegrees(double degrees){
        Angle a = new Angle();
        a.angleInDegrees = degrees;
        a.angleInRadians = Math.PI*degrees/180;
        return a;
    }

    public static Angle angleFromRadians(double radians){
        Angle a = new Angle();
        a.angleInRadians = radians;
        a.angleInDegrees = radians*180/Math.PI;
        return a;
    }

    public double getDegrees(){
        return angleInDegrees;
    }
```

```
    public double getRadians(){
        return angleInRadians;
    }

    public void setDegrees(double degrees){
        this.angleInDegrees = degrees;
        this.angleInRadians = Math.PI*degrees/180;
    }

    public void setRadians(double radians){
        this.angleInRadians = radians;
        this.angleInDegrees = radians*180/Math.PI;
    }
    private Angle(){}
}
```

( ) . **angleInDegrees angleInRadians** . .

. .

, Angle .

```
public class Angle {

    private double angleInDegrees;

    public static Angle angleFromDegrees(double degrees){
        Angle a = new Angle();
        a.angleInDegrees = degrees;
        return a;
    }

    public static Angle angleFromRadians(double radians){
        Angle a = new Angle();
        a.angleInDegrees = radians*180/Math.PI;
        return a;
    }

    public double getDegrees(){
        return angleInDegrees;
    }

    public double getRadians(){
        return angleInDegrees*Math.PI / 180;
    }

    public void setDegrees(double degrees){
        this.angleInDegrees = degrees;
    }

    public void setRadians(double radians){
        this.angleInDegrees = radians*180/Math.PI;
    }

    private Angle(){}
}
```

, 1 , () .

. angleInRadians `Angle` . (), .

: https://riptutorial.com/ko/java/topic/1295/

168:

java.util .

List<O> Set<O> Map<K,V> Collection<O> . .

. Generics .

java.util java.util.concurrent .

Java SE 1.4

Java 1.4.2 . . .

Collection<E> .

- List<E> . . .
- Set<E> .
  - SortedSet<E> Set<E> .
- Map<K,V> / .
  - SortedMap<K,V> Map<K,V> .

Java SE 5

Java 5 .

- Queue<E> . FIFO LIFO . Stack .

Java SE 6

Java 6 .

- NavigableSet<E> , Set<E> .
- NavigableMap<K,V> Map<K,V> .
- Deque<E> Queue<E> .

. ArrayList , HashSet , HashMap PriorityQueue .

.

Liskov Substitution Principle . , SortedSet<E> Set<E> . Generics Bounded Parameters .

, , AbstractList .

Java SE 1.2

1.2 / .

- ArrayList Vector

- `Map` `Dictionary` . Dictionary .
- `HashMap` `Hashtable`

.

Examples

ArrayList

`List` `ArrayList` .

```
List aListOfFruits = new ArrayList();
```

Java SE 5

```
List<String> aListOfFruits = new ArrayList<String>();
```

Java SE 7

```
List<String> aListOfFruits = new ArrayList<>();
```

`add` `String` .

```
aListOfFruits.add("Melon");
aListOfFruits.add("Strawberry");
```

`ArrayList` 0 `String` "Melon" 1 `String` "Strawberry" .

`addAll(Collection<? extends E> c)` .

```
List<String> aListOfFruitsAndVeggies = new ArrayList<String>();
aListOfFruitsAndVeggies.add("Onion");
aListOfFruitsAndVeggies.addAll(aListOfFruits);
```

"Onion" `aListOfFruitsAndVeggies` 0 "Melon" 1 "Strawberry" 2 .

`List` `java.utils.Arrays` . `Arrays.asList` :

```
List<String> data = Arrays.asList("ab", "bc", "cd", "ab", "bc", "cd");
```

.

```
List<String> list = new ArrayList<>(data); // will add data as is
Set<String> set1 = new HashSet<>(data); // will add data keeping only unique values
SortedSet<String> set2 = new TreeSet<>(data); // will add data keeping unique values and
sorting
Set<String> set3 = new LinkedHashSet<>(data); // will add data keeping only unique values and
preserving the original order
```

Google Guava Collections

( ) `Google Guava . Lists Sets` :

```
import com.google.common.collect.Lists;
import com.google.common.collect.Sets;
...
List<String> list1 = Lists.newArrayList("ab", "bc", "cd");
List<String> list2 = Lists.newArrayList(data);
Set<String> set4 = Sets.newHashSet(data);
SortedSet<String> set5 = Sets.newTreeSet("bc", "cd", "ab", "bc", "cd");
```

Map , `Map<String, Object> map` (), .

```
Map<String, Object> map1 = new HashMap<>(map);
SortedMap<String, Object> map2 = new TreeMap<>(map);
```

Apache Commons Collections

`Apache Commons ArrayUtils.toMap MapUtils.toMap` Map .

```
import org.apache.commons.lang3.ArrayUtils;
...
// Taken from org.apache.commons.lang.ArrayUtils#toMap JavaDoc

// Create a Map mapping colors.
Map colorMap = MapUtils.toMap(new String[][] {{
    {"RED", "#FF0000"},
    {"GREEN", "#00FF00"},
    {"BLUE", "#0000FF"}});
```

Map.Entry Array , . .

Google Guava Collections

`Google Guava Maps .`

```
import com.google.common.collect.Maps;
...
void howToCreateMapsMethod(Function<? super K,V> valueFunction,
          Iterable<K> keys1,
          Set<K> keys2,
          SortedSet<K> keys3) {
    ImmutableMap<K, V> map1 = toMap(keys1, valueFunction); // Immutable copy
    Map<K, V> map2 = asMap(keys2, valueFunction); // Live Map view
    SortedMap<K, V> map3 = toMap(keys3, valueFunction); // Live Map view
}
```

Java SE 8

Stream ,

```
Stream.of("xyz", "abc").collect(Collectors.toList());
```

```
Arrays.stream("xyz", "abc").collect(Collectors.toList());
```

.

. .

```
List<String> newList = new ArrayList<String>();
newList.addAll(listOne);
newList.addAll(listTwo);
```

. .

```
List<String> newList = new ArrayList<String>(listOne);
newList.addAll(listTwo);
```

. Apache commons-collections .

```
ListUtils.union(listOne,listTwo);
```

Java SE 8

```
List<String> newList = Stream.concat(listOne.stream(),
listTwo.stream()).collect(Collectors.toList());
```

.

. .

.

```
List<String> fruits = new ArrayList<String>();
fruits.add("Apple");
fruits.add("Banana");
fruits.add("Strawberry");
```

---

for "" :

Apple Strawberry . Banana Apple 0 . i 1 .

```
for (int i = 0; i < fruits.size(); i++) {
    System.out.println (fruits.get(i));
    if ("Apple".equals(fruits.get(i))) {
         fruits.remove(i);
    }
```

```
}
```

for .

.

: java.util.ConcurrentModificationException

```
for (String fruit : fruits) {
    System.out.println(fruit);
    if ("Apple".equals(fruit)) {
        fruits.remove(fruit);
    }
}
```

Iterator while

```
Iterator<String> fruitIterator = fruits.iterator();
while(fruitIterator.hasNext()) {
    String fruit = fruitIterator.next();
    System.out.println(fruit);
    if ("Apple".equals(fruit)) {
        fruitIterator.remove();
    }
}
```

`Iterator` `remove()` ., 「」 `UnsupportedOperationException` .

UnsupportedOperationException - remove

(, 3 `Collections.unmodifiable...()` ,).

`Iterator` , `Iterator` `List` `modCount` , `ConcurrentModificationException` Throw. .

`modCount` `int` . `Collection` `add()` `remove()` (`Iterator`). `Iterator` , `modCount` `modCount` `Iterator` `modCount` `List` . `modCount` (), `ConcurrentModificationException` Throw.

.

```
Iterator<String> fruitIterator = fruits.iterator();
fruits.set(0, "Watermelon");
while(fruitIterator.hasNext()){
    System.out.println(fruitIterator.next());
}
```

, `Iterator` `List` (), `ConcurrentModificationException` throw.

```
Iterator<String> fruitIterator = fruits.iterator();
```

```
fruits.add("Watermelon");
while(fruitIterator.hasNext()){
    System.out.println(fruitIterator.next());    //ConcurrentModificationException here
}
```

```
for (int i = (fruits.size() - 1); i >=0; i--) {
    System.out.println (fruits.get(i));
    if ("Apple".equals(fruits.get(i))) {
         fruits.remove(i);
    }
}
```

. . . LinkedList .

,

```
for (int i = 0; i < fruits.size(); i++) {
    System.out.println (fruits.get(i));
    if ("Apple".equals(fruits.get(i))) {
         fruits.remove(i);
         i--;
    }
}
```

.i List i+1 i . i .

" "

```
ArrayList shouldBeRemoved = new ArrayList();
for (String str : currentArrayList) {
    if (condition) {
        shouldBeRemoved.add(str);
    }
}
currentArrayList.removeAll(shouldBeRemoved);
```

.

Java SE 8

Java 8 . .

List . .

```
List<String> filteredList =
    fruits.stream().filter(p -> !"Apple".equals(p)).collect(Collectors.toList());
```

List List .

removeIf

removeIf

.

```
fruits.removeIf(p -> "Apple".equals(p));
```

. "" Java .

. UnsupportedOperationException .

.

```
import java.util.ArrayList;
import java.util.Collections;
import java.util.List;

public class MyPojoClass {
    private List<Integer> intList = new ArrayList<>();

    public void addValueToIntList(Integer value){
        intList.add(value);
    }

    public List<Integer> getIntList() {
        return Collections.unmodifiableList(intList);
    }
}
```

throw.

```
import java.util.List;

public class App {

    public static void main(String[] args) {
        MyPojoClass pojo = new MyPojoClass();
        pojo.addValueToIntList(42);

        List<Integer> list = pojo.getIntList();
        list.add(69);
    }
}
```

:

```
Exception in thread "main" java.lang.UnsupportedOperationException
    at java.util.Collections$UnmodifiableCollection.add(Collections.java:1055)
    at App.main(App.java:12)
```

```
List<String> names  = new ArrayList<>(Arrays.asList("Clementine", "Duran", "Mike"));
```

Java SE 8

```
names.forEach(System.out::println);
```

```
names.parallelStream().forEach(System.out::println);
```

Java SE 5

```
for (String name : names) {
    System.out.println(name);
}
```

Java SE 5

```
for (int i = 0; i < names.size(); i++) {
    System.out.println(names.get(i));
}
```

Java SE 1.2

```
//Creates ListIterator which supports both forward as well as backward traversel
ListIterator<String> listIterator = names.listIterator();

//Iterates list in forward direction
while(listIterator.hasNext()){
    System.out.println(listIterator.next());
}

//Iterates list in backward direction once reaches the last element from above iterator in
forward direction
while(listIterator.hasPrevious()){
    System.out.println(listIterator.previous());
}
```

```
Set<String> names = new HashSet<>(Arrays.asList("Clementine", "Duran", "Mike"));
```

Java SE 8

```
names.forEach(System.out::println);
```

Java SE 5

```
for (Iterator<String> iterator = names.iterator(); iterator.hasNext(); ) {
    System.out.println(iterator.next());
}

for (String name : names) {
    System.out.println(name);
}
```

Java SE 5

```
Iterator iterator = names.iterator();
while (iterator.hasNext()) {
    System.out.println(iterator.next());
```

```
}
```

```
Map<Integer, String> names = new HashMap<>();
names.put(1, "Clementine");
names.put(2, "Duran");
names.put(3, "Mike");
```

Java SE 8

```
names.forEach((key, value) -> System.out.println("Key: " + key + " Value: " + value));
```

Java SE 5

```
for (Map.Entry<Integer, String> entry : names.entrySet()) {
    System.out.println(entry.getKey());
    System.out.println(entry.getValue());
}

// Iterating over only keys
for (Integer key : names.keySet()) {
    System.out.println(key);
}
// Iterating over only values
for (String value : names.values()) {
    System.out.println(value);
}
```

Java SE 5

```
Iterator entries = names.entrySet().iterator();
while (entries.hasNext()) {
    Map.Entry entry = (Map.Entry) entries.next();
    System.out.println(entry.getKey());
    System.out.println(entry.getValue());
}
```

. Collections .

```
List<String> anEmptyList = Collections.emptyList();
Map<Integer, Date> anEmptyMap = Collections.emptyMap();
Set<Number> anEmptySet = Collections.emptySet();
```

. , emptyList() List emptySet() emptyMap() .

put , (add , put) (), UnsupportedOperationException Throw . null new .

Java . Map<int, int> Java Map<int, int> . , Map<Integer, Integer> . Java auto-boxing .

```
Map<Integer, Integer> map = new HashMap<>();
map.put(1, 17); // Automatic boxing of int to Integer objects
int a = map.get(1); // Automatic unboxing.
```

. `HashMap<Integer, Integer>` 72 (: 64 JVM , 256 50 %). 8 . (Map -> Entry -> Value)
.

(50 % ~ 16 , 4 1) , Java .

Iterator .

List `Iterator` .
.

: .

.

```
    String[] names = {"James","Smith","Sonny","Huckle","Berry","Finn","Allan"};
    List<String> nameList = new ArrayList<>();

    //Create a List from an Array
    nameList.addAll(Arrays.asList(names));

    String[] removeNames = {"Sonny","Huckle","Berry"};
    List<String> removeNameList = new ArrayList<>();

    //Create a List from an Array
    removeNameList.addAll(Arrays.asList(removeNames));
```

Collection `nameList` `removeNameList` .

```
private static void removeNames(Collection<String> collection1, Collection<String>
collection2) {
      //get Iterator.
    Iterator<String> iterator = collection1.iterator();

    //Loop while collection has items
    while(iterator.hasNext()){
        if (collection2.contains(iterator.next()))
            iterator.remove(); //remove the current Name or Item
    }
}
```

`nameList` `removeNameList` . `removeNames(nameList,removeNameList);`
.

: **James Smith Sonny Huckle Berry**
: **James Smith Finn Allan**

.

Iterator for-each Iterable .

for-each .

1. `Iterable` `iterator()` .

2. `hasNext()` , `next()` `remove()` `hasNext()` override `hasNext()` , `java.util.Iterator` .

.

```
package org.algorithms.linkedlist;

import java.util.Iterator;
import java.util.NoSuchElementException;


public class LinkedList<T> implements Iterable<T> {

    Node<T> head, current;

    private static class Node<T> {
        T data;
        Node<T> next;

        Node(T data) {
            this.data = data;
        }
    }

    public LinkedList(T data) {
        head = new Node<>(data);
    }

    public Iterator<T> iterator() {
        return new LinkedListIterator();
    }

    private class LinkedListIterator implements Iterator<T> {

        Node<T> node = head;

        @Override
        public boolean hasNext() {
            return node != null;
        }

        @Override
        public T next() {
            if (!hasNext())
                throw new NoSuchElementException();
            Node<T> prevNode = node;
            node = node.next;
            return prevNode.data;
        }

        @Override
        public void remove() {
            throw new UnsupportedOperationException("Removal logic not implemented.");
        }
    }

    public void add(T data) {
        Node current = head;
        while (current.next != null)
            current = current.next;
        current.next = new Node<>(data);
```

```
    }

}

class App {
    public static void main(String[] args) {

        LinkedList<Integer> list = new LinkedList<>(1);
        list.add(2);
        list.add(4);
        list.add(3);

        //Test #1
        System.out.println("using Iterator:");
        Iterator<Integer> itr = list.iterator();
        while (itr.hasNext()) {
            Integer i = itr.next();
            System.out.print(i + " ");
        }

        //Test #2
        System.out.println("\n\nusing for-each:");
        for (Integer data : list) {
            System.out.print(data + " ");
        }
    }
}
```

```
using Iterator:
1 2 4 3
using for-each:
1 2 4 3
```

Java 7 . Java 5 Java 6 .

```
LinkedList<Integer> list = new LinkedList<>(1);
```

```
LinkedList<Integer> list = new LinkedList<Integer>(1);
```

.

Pitfall :

iterator . , .

```
List<IHat> hats = new ArrayList<>();
hats.add(new Ushanka()); // that one has ear flaps
hats.add(new Fedora());
hats.add(new Sombrero());
for (IHat hat : hats) {
    if (hat.hasEarFlaps()) {
        hats.remove(hat);
    }
}
```

ConcurrentModificationException . . . ,

subList (int fromIndex, int toIndex)

fromIndex toIndex .

```
List list = new ArrayList();
List list1 = list.subList(fromIndex,toIndex);
```

1. , IndexOutofBoundException Throw.
2. 1 . .
3. fromIndex toIndex (fromIndex> toIndex) , IllegalArgumentException Throw.

:

```
List<String> list = new ArrayList<String>();
List<String> list = new ArrayList<String>();
list.add("Hello1");
list.add("Hello2");
System.out.println("Before Sublist "+list);
List<String> list2 = list.subList(0, 1);
list2.add("Hello3");
System.out.println("After sublist changes "+list);
```

:
[Hello1, Hello2]
[Hello1, Hello3, Hello2]

subSet (fromIndex, toIndex) .

fromIndex toIndex .

```
Set set = new TreeSet();
Set set1 = set.subSet(fromIndex,toIndex);
```

, () IllegalArgumentException throw.

(fromKey, toKey)

fromKey toKey .

```
Map map = new TreeMap();
Map map1 = map.get(fromKey,toKey);
```

fromKey toKey , fromKey toKey IllegalArgumentException Throw.

. .

: https://riptutorial.com/ko/java/topic/90/

169:

Java . . .

Examples

Java Collections

.

[http://i.stack.imgur.com/aSDsG.png] .

: https://riptutorial.com/ko/java/topic/10846/-

170: I / O

Examples

BufferedReader :

```
System.out.println("Please type your name and press Enter.");

BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));
try {
    String name = reader.readLine();
    System.out.println("Hello, " + name + "!");
} catch(IOException e) {
    System.out.println("An error occurred: " + e.getMessage());
}
```

.

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
```

Scanner :

Java SE 5

```
System.out.println("Please type your name and press Enter");

Scanner scanner = new Scanner(System.in);
String name = scanner.nextLine();

System.out.println("Hello, " + name + "!");
```

.

```
import java.util.Scanner;
```

`scanner.nextLine()` .

```
System.out.println("Please enter your first and your last name, on separate lines.");

Scanner scanner = new Scanner(System.in);
String firstName = scanner.nextLine();
String lastName = scanner.nextLine();

System.out.println("Hello, " + firstName + " " + lastName + "!");
```

`Strings` , `next()` `nextLine()` `.next()` (`""` `nextLine()` `nextLine()` .

Scanner String . .

```
scanner.nextByte();
scanner.nextShort();
scanner.nextInt();
scanner.nextLong();
scanner.nextFloat();
scanner.nextDouble();
scanner.nextBigInteger();
scanner.nextBigDecimal();
```

hasNextLine() , hasNextInt()) has hasNextInt() true true . : (: nextInt() "a") .try {} catch() {} ( ).

```
Scanner scanner = new Scanner(System.in); //Create the scanner
scanner.useLocale(Locale.US); //Set number format excepted
System.out.println("Please input a float, decimal separator is .");
if (scanner.hasNextFloat()){ //Check if it is a float
    float fValue = scanner.nextFloat(); //retrive the value directly as float
    System.out.println(fValue + " is a float");
}else{
    String sValue = scanner.next(); //We can not retrive as float
    System.out.println(sValue + " is not a float");
}
```

System.console :

Java SE 6

```
String name = System.console().readLine("Please type your name and press Enter%n");

System.out.printf("Hello, %s!", name);

//To read passwords (without echoing as in unix terminal)
char[] password = System.console().readPassword();
```

:

- .
- .

: . Eclipse IDE . IDE .

.

```
public class ExampleCli {

    private static final String CLI_LINE   = "example-cli>"; //console like string

    private static final String CMD_QUIT   = "quit";    //string for exiting the program
    private static final String CMD_HELLO  = "hello";    //string for printing "Hello World!"
on the screen
    private static final String CMD_ANSWER = "answer";    //string for printing 42 on the
```

```
screen

    public static void main(String[] args) {
        ExampleCli claimCli = new ExampleCli();    // creates an object of this class

        try {
            claimCli.start();    //calls the start function to do the work like console
        }
        catch (IOException e) {
            e.printStackTrace();    //prints the exception log if it is failed to do get the
user input or something like that
        }
    }

    private void start() throws IOException {
        String cmd = "";

        BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));
        while (!cmd.equals(CMD_QUIT)) {    // terminates console if user input is "quit"
            System.out.print(CLI_LINE);    //prints the console-like string

            cmd = reader.readLine();    //takes input from user. user input should be started
with "hello",  "answer" or "quit"
            String[] cmdArr = cmd.split(" ");

            if (cmdArr[0].equals(CMD_HELLO)) {    //executes when user input starts with
"hello"
                hello(cmdArr);
            }
            else if (cmdArr[0].equals(CMD_ANSWER)) {    //executes when user input starts with
"answer"
                answer(cmdArr);
            }
        }
    }

    // prints "Hello World!" on the screen if user input starts with "hello"
    private void hello(String[] cmdArr) {
        System.out.println("Hello World!");
    }

    // prints "42" on the screen if user input starts with "answer"
    private void answer(String[] cmdArr) {
        System.out.println("42");
    }
}
```

System.out.format PrintWriter.format . String .

```
String rowsStrings[] = new String[] {"1",
                                     "1234",
                                     "1234567",
                                     "123456789"};

String column1Format = "%-3s";    // min 3 characters, left aligned
String column2Format = "%-5.8s";  // min 5 and max 8 characters, left aligned
String column3Format = "%6.6s";   // fixed size 6 characters, right aligned
String formatInfo = column1Format + " " + column2Format + " " + column3Format;

for(int i = 0; i < rowsStrings.length; i++) {
```

```
    System.out.format(formatInfo, rowsStrings[i], rowsStrings[i], rowsStrings[i]);
    System.out.println();
}
```

:

```
1   1          1
1234 1234    1234
1234567 1234567 123456
123456789 12345678 123456
```

.

```
String rowsStrings[] = new String[] {"1",
                                     "1234",
                                     "1234567",
                                     "123456789"};

String column1Format = "%-3.3s";  // fixed size 3 characters, left aligned
String column2Format = "%-8.8s";  // fixed size 8 characters, left aligned
String column3Format = "%6.6s";   // fixed size 6 characters, right aligned
String formatInfo = column1Format + " " + column2Format + " " + column3Format;

for(int i = 0; i < rowsStrings.length; i++) {
    System.out.format(formatInfo, rowsStrings[i], rowsStrings[i], rowsStrings[i]);
    System.out.println();
}
```

:

```
1   1             1
123 1234       1234
123 1234567  123456
123 12345678 123456
```

- `%s` :
- `%5s` : 5 . 5 .
- `%-5s` : 5 . 5 .
- `%5.10s` : 5 10 . 5 5 . 10 10 .
- `%-5.5s` : 5 ( ). 5 5 . 5 5 .

I / O : https://riptutorial.com/ko/java/topic/126/-i---o

171:

Java 9 Java Collections API . . .

`List<E>` , `Set<E>` `Map<K, V>` .

- `static <E> List<E> of()`
- `static <E> List<E> of(E e1)`
- `static <E> List<E> of(E e1, E e2)`
- `static <E> List<E> of(E e1, E e2, ..., E e9, E e10)`
- `static <E> List<E> of(E... elements)`
- `static <E> Set<E> of()`
- `static <E> Set<E> of(E e1)`
- `static <E> Set<E> of(E e1, E e2)`
- `static <E> Set<E> of(E e1, E e2, ..., E e9, E e10)`
- `static <E> Set<E> of(E... elements)`
- `static <K,V> Map<K,V> of()`
- `static <K,V> Map<K,V> of(K k1, V v1)`
- `static <K,V> Map<K,V> of(K k1, V v1, K k2, V v2)`
- `static <K,V> Map<K,V> of(K k1, V v1, K k2, V v2, ..., K k9, V v9, K k10, V v10)`
- `static <K,V> Map<K,V> ofEntries(Map.Entry<? extends K,? extends V>... entries)`

w /	
`List.of(E e)`	.
`Set.of(E e)`	.
`Map.of(K k, V v)`	- .
`Map.of(Map.Entry<? extends K, ? extends V> entry)`	`K` 1 `Map.Entry` `Map.Entry` , `V` .

Examples

- `List<Integer> immutableEmptyList = List.of();`
  - (empty) `List<Integer>` .
- `List<Integer> immutableList = List.of(1, 2, 3, 4, 5);`
  - 5 `List<Integer>` .
- `List<Integer> mutableList = new ArrayList<>(immutableList);`
  - `List<Integer>` `List<Integer>` .

- `Set<Integer> immutableEmptySet = Set.of();`
  - `Set<Integer>` .
- `Set<Integer> immutableSet = Set.of(1, 2, 3, 4, 5);`
  - 5 `Set<Integer>` .
- `Set<Integer> mutableSet = new HashSet<>(immutableSet);`
  - `Set<Integer>` `Set<Integer>` .

- `Map<Integer, Integer> immutableEmptyMap = Map.of();`
  - `Map<Integer, Integer>` .

- `Map<Integer, Integer> immutableMap = Map.of(1, 2, 3, 4);`
    - - `Map<Integer, Integer>` .
- `Map<Integer, Integer> immutableMap = Map.ofEntries(Map.entry(1, 2), Map.entry(3, 4));`
    - - `Map<Integer, Integer>` .
- `Map<Integer, Integer> mutableMap = new HashMap<>(immutableMap);`
    - `Map<Integer, Integer>` `Map<Integer, Integer>` .

: https://riptutorial.com/ko/java/topic/9783/--

172: - Java Reflection

java.lang.Class .

java.lang java.lang.reflect Java .

Reflection API .

IDE (Integrated Development Environment) (: Eclipse, MyEclipse, NetBeans)

Examples

Object getClass ()

```
class Simple { }

class Test {
     void printName(Object obj){
         Class c = obj.getClass();
         System.out.println(c.getName());
     }
     public static void main(String args[]){
         Simple s = new Simple();

         Test t = new Test();
         t.printName(s);
    }
}
```

- Java Reflection : https://riptutorial.com/ko/java/topic/10151/---java-reflection

173:

. .

JAR ZIP . JAR ZIP .

JVM . . .

Examples

.

```
URL[] urls = new URL[] {new URL("file:/home/me/extras.jar")};
Classloader loader = new URLClassLoader(urls);
Class<?> myObjectClass = loader.findClass("com.example.MyObject");
```

"extra.jar" . `findClass` .

`findClass` . :

- `ClassNotFoundException` throw .
- `NoClassDefFoundError` throw.

classLoader

`java.lang.ClassLoader` . .

- `findClass(String)` - .
- `loadClass(String, boolean)` - .
- `findResource findResources` - .

`defineClass` `final` . `defineClass` .

.

```
public class ByteArrayClassLoader extends ClassLoader {
    private String classname;
    private byte[] classfile;

    public ByteArrayClassLoader(String classname, byte[] classfile) {
        this.classname = classname;
        this.classfile = classfile.clone();
    }

    @Override
    protected Class findClass(String classname) throws ClassNotFoundException {
        if (classname.equals(this.classname)) {
            return defineClass(classname, classfile, 0, classfile.length);
        } else {
            throw new ClassNotFoundException(classname);
        }
```

```
    }
}
```

findClass loadClass .

1. loadClass findLoadedClass . , Class .
2. loadClass loadClass . Class , Requestor .
3. findClass findClass .
4. this.classname this.classname defineClass this.classfile . Class .
5. ClassNotFoundException .

.class

. ClassLoader . . ClassLoader . ClassLoader private defineClass() .

ByteClassLoader ClassLoader. ClassLoader ClassLoader private . defineClass() private defineClass() . , name , classBytes , 0 classBytes ' classBytes[0] , (offset) , classBytes.lenght .

```
public class ByteClassLoader extends ClassLoader {

    public Class<?> defineClass(String name, byte[] classBytes) {
        return defineClass(name, classBytes, 0, classBytes.length);
    }

}
```

public defineClass() . .

stackoverflow MyClass .

Paths.get() Path . Files.readAllBytes(path) . ByteClassLoader defineClass(). (stackoverflow.MyClass) () .

```
Path path = Paths.get("MyClass.class");

ByteClassLoader loader = new ByteClassLoader();
loader.defineClass("stackoverflow.MyClass", Files.readAllBytes(path);
```

: defineClass() Class<?> . .

loadClass() . ClassNotFoundException try cath .

```
try{
    loader.loadClass("stackoverflow.MyClass");
} catch(ClassNotFoundException e){
    e.printStackTrace();
}
```

: https://riptutorial.com/ko/java/topic/5443/-

174:

Java . Java .

JVM (Java Virtual Machine) ((lazy-loading)). .

1. Java Java .
2. (, `jre/lib/ext/`)
3. .

`java.lang.ClassLoader` . : .

classpath JVM . (`CLASSPATH`) .

Examples

.

1. `CLASSPATH` .

```
set CLASSPATH=...          # Windows and csh
export CLASSPATH=...       # Unix ksh/bash
```

2. .

```
java -classpath ...
javac -classpath ...
```

`-classpath` (`-cp`) `CLASSPATH` .

3. JAR `Class-Path` `MANIFEST.MF` `Class-Path` .

```
Class-Path: jar1-name jar2-name directory-name/jar3-name
```

JAR .

```
java -jar some.jar ...
```

JAR `Class-Path` `-classpath` CLASSPATH .

classpath , classpath, `java -jar` JAR , .

:

- https://docs.oracle.com/javase/tutorial/deployment/jar/downman.html
- http://docs.oracle.com/javase/7/docs/technotes/tools/windows/classpath.html

JAR

JAR classpath classpath . :

```
someFolder/*
```

`someFolder` JAR ZIP JVM . JAR `-cp` , `CLASSPATH` `Class-Path` . : .

:

1. Classpath Java 6 . Java "*" .
2. " " . : "*someFolder* / .jar" .
3. ".jar" ".JAR" . ZIP JAR .
4. JAR .
5. JAR .

, JAR ZIP JAR / ZIP .

- (: `-classpath`) , ; ; () (Linux, UNIX, MacOSX) :) .
- JAR `MANIFEST.MF` `Class-Path` .

.

- , . :

  ```
  export CLASSPATH="/home/user/My JAR Files/foo.jar:second.jar"
  ```

  (.)

- JAR "MANIFEST.MF" URL .

  ```
  Class-Path: /home/user/My%20JAR%20Files/foo.jar second.jar
  ```

JAR . , JAR . `main()` . `main()` .

SWT JAR JAR `main()` . Java SWT

JAR (, , , KeyStore, ...) . `Class` `ClassLoader` .

.

```
program.jar
|
\-com
  \-project
    |
    |-file.txt
    \-Test.class
```

Test file.txt . .

```
InputStream is = Test.class.getClassLoader().getResourceAsStream("com/project/file.txt");
```

( ) .

Test .

```
InputStream is = Test.class.getResourceAsStream("file.txt");
```

. Test.class com.project , file.txt .

.

```
 is = Test.class.getResourceAsStream("/com/project/file.txt");
```

Java ( ) .

.

- .
- .
- $.
- .

.

SomeClass	SomeClass.java	SomeClass.class
com.example.SomeClass	com/example/SomeClass.java	com/example/SomeClass.class
SomeClass.Inner	(SomeClass.java)	SomeClass$Inner.class
SomeClass	(SomeClass.java)	SomeClass$1.class , SomeClass$2.class

:

JVM . .

. , JAR ZIP . .

.

1. RP . .
2. E :
    - :
        - E RP AP .

        - AP .
        - .
    - JAR ZIP :
        - JAR / ZIP RP .
        - JAR / ZIP .

. . Class.getResource Class.getResourceAsStream , .

Java .

Java . "rt.jar" JRE JAR . "" Java SE .

. java , javac .

, Java . " " . , JAR JVM .

java -X .

- -Xbootclasspath:<path> .
- -Xbootclasspath/a:<path> .
- -Xbootclasspath/p:<path> .

bootclasspath Java () Java . . (Java Binary License .)

: https://riptutorial.com/ko/java/topic/3720/-

175:

. : , , , , . , , . .

- / / .

- class Example {} // , ,

Examples

```
class TrivialClass {}
```

class , .

new .

```
TrivialClass tc = new TrivialClass();
```

vs

:

```
class ObjectMemberVsStaticMember {

    static int staticCounter = 0;
    int memberCounter = 0;

    void increment() {
        staticCounter ++;
        memberCounter++;
    }
}
```

:

```
final ObjectMemberVsStaticMember o1 = new ObjectMemberVsStaticMember();
final ObjectMemberVsStaticMember o2 = new ObjectMemberVsStaticMember();

o1.increment();

o2.increment();
o2.increment();

System.out.println("o1 static counter " + o1.staticCounter);
System.out.println("o1 member counter " + o1.memberCounter);
System.out.println();

System.out.println("o2 static counter " + o2.staticCounter);
System.out.println("o2 member counter " + o2.memberCounter);
System.out.println();
```

```
System.out.println("ObjectMemberVsStaticMember.staticCounter = " +
ObjectMemberVsStaticMember.staticCounter);

// the following line does not compile. You need an object
// to access its members
//System.out.println("ObjectMemberVsStaticMember.staticCounter = " +
ObjectMemberVsStaticMember.memberCounter);
```

.

```
o1 static counter 3
o1 member counter 1

o2 static counter 3
o2 member counter 2

ObjectMemberVsStaticMember.staticCounter = 3
```

: `static` . JVM , .

`static` . `static` . .

. . . .

```
public class Displayer {

    public void displayName(String firstName) {
        System.out.println("Name is: " + firstName);
    }

    public void displayName(String firstName, String lastName) {
        System.out.println("Name is: " + firstName + " " + lastName);
    }

    public static void main(String[] args) {
        Displayer displayer = new Displayer();
        displayer.displayName("Ram");          //prints "Name is: Ram"
        displayer.displayName("Jon", "Skeet"); //prints "Name is: Jon Skeet"
    }
}
```

. ( ) .

.

1. .

   : `method(String s)` `method(String s1, String s2)` .

2. .

   : `method(int i, float f)` `method(float f, int i))` .

: (*`int method()` `String method()` `RuntimeException`* ***).*** .

( ).

```
public class Car {

    //Variables describing the characteristics of an individual car, varies per  object
   private int milesPerGallon;
   private String name;
   private String color;
   public int numGallonsInTank;

    public Car(){
        milesPerGallon = 0;
        name = "";
        color = "";
        numGallonsInTank = 0;
    }

    //this is where an individual object is created
    public Car(int mpg, int, gallonsInTank, String carName, String carColor){
        milesPerGallon = mpg;
        name = carName;
        color = carColor;
        numGallonsInTank = gallonsInTank;
    }

    //methods to make the object more usable

    //Cars need to drive
    public void drive(int distanceInMiles){
        //get miles left in car
        int miles = numGallonsInTank * milesPerGallon;

        //check that car has enough gas to drive distanceInMiles
        if (miles <= distanceInMiles){
            numGallonsInTank = numGallonsInTank - (distanceInMiles / milesPerGallon)
            System.out.println("Drove " + numGallonsInTank + " miles!");
        } else {
            System.out.println("Could not drive!");
        }
    }

    public void paintCar(String newColor){
        color = newColor;
    }
        //set new Miles Per Gallon
    public void setMPG(int newMPG){
        milesPerGallon = newMPG;
    }

       //set new number of Gallon In Tank
    public void setGallonsInTank(int numGallons){
        numGallonsInTank = numGallons;
    }

    public void nameCar(String newName){
        name = newName;
    }

    //Get the Car color
    public String getColor(){
        return color;
```

```
    }

    //Get the Car name
    public String getName(){
        return name;
    }

    //Get the number of Gallons
    public String getGallons(){
        return numGallonsInTank;
    }

}
```

. (Java Android onCreate) Car .

1

```
`Car newCar = new Car(30, 10, "Ferrari", "Red");
```

1 Car . `repaintCar` . :

```
 newCar.repaintCar("Blue");
```

: Y . `repaintCar` .

. (`void`) Car . :

```
String myCarName = newCar.getName();  //returns string "Ferrari"
```

1 .

2

```
`Car newCar = new Car();
```

2 . Car Constructor .

```
public void Car(){
        milesPerGallon = 0;
        name = "";
        color = "";
        numGallonsInTank = 0;
    }
```

. . .

: . ,

```
Car myCar = new Car();
String color = Car.getColor(); //returns empty string
```

. Car (`public Car(){}`) myCar . :

```
Car myCar = new Car();
myCar.nameCar("Ferrari");
myCar.paintCar("Purple");
myCar.setGallonsInTank(10);
myCar.setMPG(30);
```

, . :

```
String myCarName = myCar.getName(); //returns string "Ferrari"
```

. . . .

```
public class Hello{
    // constructor
    public Hello(String wordToPrint){
        printHello(wordToPrint);
    }
    public void printHello(String word){
        System.out.println(word);
    }
}
// instantiates the object during creating and prints out the content
// of wordToPrint
```

.

1. `public` , `private` `protected` `abstract` , `final` , `static` `synchronized` .
2. .
3. (MUST). `Hello` `Hello` .
4. `this` . `this.method(...)` , `this(...)` .

`super` .

```
public class SuperManClass{

    public SuperManClass(){
        // some implementation
    }

    // ... methods
}


public class BatmanClass extends SupermanClass{
    public BatmanClass(){
        super();
    }
    //... methods...
}
```

Java # 8.8 # 15.9 .

static final static . String .

```
public class MyClass {

    public static final Set<String> WORDS;

    static {
        Set<String> set = new HashSet<>();
        set.add("Hello");
        set.add("World");
        set.add("foo");
        set.add("bar");
        set.add("42");
        WORDS = Collections.unmodifiableSet(set);
    }
}
```

.

Java .

( ) () . , .

```
public class Shape{
    //It could be a circle or rectangle or square
    private String type;

    //To calculate area of rectangle
    public Double area(Long length, Long breadth){
        return (Double) length * breadth;
    }

     //To calculate area of a circle
     public Double area(Long radius){
        return (Double) 3.14 * r * r;
    }
}
```

.

.

Well Java (area() .

( ) .

: public Double area(Long side) .

. .

.

.

throw .

?

.

. `String.valueOf()` ? :

```
static String valueOf(boolean b)
static String valueOf(char c)
static String valueOf(char[] data)
static String valueOf(char[] data, int offset, int count)
static String valueOf(double d)
static String valueOf(float f)
static String valueOf(int i)
static String valueOf(long l)
static String valueOf(Object obj)
```

, (, .) .

. ? ? .

```
public abstract class Shape{

    public abstract Double area(){
        return 0.0;
    }
}
```

Shape , area .

.

```
public class Circle extends Shape {
    private Double radius = 5.0;

    // See this annotation @Override, it is telling that this method is from parent
    // class Shape and is overridden here
    @Override
    public Double area(){
        return 3.14 * radius * radius;
    }
}
```

:

```
 public class Rectangle extends Shape {
    private Double length = 5.0;
    private Double breadth= 10.0;


    // See this annotation @Override, it is telling that this method is from parent
    // class Shape and is overridden here
    @Override
    public Double area(){
```

```
        return length * breadth;
    }
}
```

(Shape) . ? psvm .

```
public class AreaFinder{

    public static void main(String[] args){

        //This will create an object of circle class
        Shape circle = new Circle();
        //This will create an object of Rectangle class
        Shape rectangle = new Rectangle();

        // Drumbeats ......
        //This should print 78.5
        System.out.println("Shape of circle : "+circle.area());

        //This should print 50.0
        System.out.println("Shape of rectangle: "+rectangle.area());

    }
}
```

! ? . , .

.

.	.
.	IS-A () .
.	.
.	.
. . .	.

: https://riptutorial.com/ko/java/topic/114/-

176:

Java Java . . . , , , .

- @AnnotationName // ' '( )
- @AnnotationName (someValue) // 'value' .
- @AnnotationName (param1 = value1) //
- @AnnotationName (param1 = value1, param2 = value2) //
- @AnnotationName (param1 = {1, 2, 3}) //
- @AnnotationName ({value1}) // 'value'

.

- `String`
- `Class`
-
- Enum
-

Examples

Standard Edition Java . . , .

@

. .

```
public class Vehicle {
   public void drive() {
        System.out.println("I am driving");
   }
}

class Car extends Vehicle {
    // Fine
    @Override
    public void drive() {
        System.out.prinln("Brrrm, brrm");
    }
}
```

```
abstract class Animal  {
   public abstract void makeNoise();
}

class Dog extends Animal {
    // Fine
    @Override
    public void makeNoise() {
        System.out.prinln("Woof");
```

```
    }
}
```

.

```
class Logger1 {
    public void log(String logString) {
        System.out.prinln(logString);
    }
}

class Logger2 {
    // This will throw compile-time error. Logger2 is not a subclass of Logger1.
    // log method is not overriding anything
    @Override
    public void log(String logString) {
        System.out.println("Log 2" + logString);
    }
}
```

.

```
class Vehicle {
   public void drive() {
        System.out.println("I am driving");
   }
}

class Car extends Vehicle {
    // Compiler error. "dirve" is not the correct method name to override.
    @Override
    public void dirve() {
        System.out.prinln("Brrrm, brrm");
    }
}
```

`@Override` .

- Java 5 .
- Java 6 / .

(Java 5 .)

@Deprecated

. :

- API .
- API .
- API API ,
- API .

- API .
- .

API .

. IDE .

```
class ComplexAlgorithm {
    @Deprecated
    public void oldSlowUnthreadSafeMethod() {
        // stuff here
    }

    public void quickThreadSafeMethod() {
        // client code should use this instead
    }
}
```

@SuppressWarnings

. , .

, . .

```
@SuppressWarnings("deprecation")
public class RiddledWithWarnings {
    // several methods calling deprecated code here
}

@SuppressWarning("finally")
public boolean checkData() {
    // method calling return from within finally block
}
```

. , :

```
ComplexAlgorithm algorithm = new ComplexAlgorithm();
@SuppressWarnings("deprecation") algoritm.slowUnthreadSafeMethod();
// we marked this method deprecated in an example above

@SuppressWarnings("unsafe") List<Integer> list = getUntypeSafeList();
// old library returns, non-generic List containing only integers
```

. JLS `unchecked deprecation` . .

@SafeVarargs

`void method(T... t)` `void method(Object[] t)` varargs . :

```
private static <T> void generatesVarargsWarning(T... lists) {
```

. `SafeVarargs` . .

@

FunctionalInterface . FunctionalInterface ( ) .

```
@FunctionalInterface
public interface ITrade {
  public boolean check(Trade t);
}

@FunctionalInterface
public interface Predicate<T> {
  boolean test(T t);
}
```

Java Reflection API , . `RetentionPolicy RUNTIME` .

```
@interface MyDefaultAnnotation {

}

@Retention(RetentionPolicy.RUNTIME)
@interface MyRuntimeVisibleAnnotation {

}

public class AnnotationAtRuntimeTest {

    @MyDefaultAnnotation
    static class RuntimeCheck1 {
    }

    @MyRuntimeVisibleAnnotation
    static class RuntimeCheck2 {
    }

    public static void main(String[] args) {
        Annotation[] annotationsByType = RuntimeCheck1.class.getAnnotations();
        Annotation[] annotationsByType2 = RuntimeCheck2.class.getAnnotations();

        System.out.println("default retention: " + Arrays.toString(annotationsByType));
        System.out.println("runtime retention: " + Arrays.toString(annotationsByType2));
    }
}
```

`@interface` . .

```
@interface MyAnnotation {
    String param1();
    boolean param2();
    int[] param3();  // array parameter
}
```

```
@interface MyAnnotation {
    String param1() default "someValue";
    boolean param2() default true;
    int[] param3() default {};
```

```
}
```

. - .

@

`@Target` - .

```
@Target(ElementType.METHOD)
@interface MyAnnotation {
    // this annotation can only be applied to methods
}
```

`@Target({ElementType.FIELD, ElementType.TYPE})` : `@Target({ElementType.FIELD, ElementType.TYPE})`

ANNOTATION_TYPE		`@Retention(RetentionPolicy.RUNTIME)` `@interface MyAnnotation`
		`@MyAnnotation` `public MyClass() {}`
	,	`@XmlAttribute` `private int count;`

LOCAL_VARIABLE		`for (@LoopVariable int i = 0; i < 100; i++)`
		`{`
		`  @Unused`
		`  String resultVariable;`
		`}`

	(`package-info.java`)	`@Deprecated` `package very.old;`
		`@XmlElement` `public int getCount() {...}`
	/	`public Rectangle(`

		`@NamedArg("width") double width, @NamedArg("height") double height) { ... }`
	`''`	`@XmlRootElement public class Report {}`

Java SE 8

TYPE_PARAMETER		`public <@MyAnnotation T> void f(T t) {}`
TYPE_USE		`Object o = "42"; String s = (@MyAnnotation String) o;`

@

`@Retention` . `.class` . `RetentionPolicy.RUNTIME` .

```
@Retention(RetentionPolicy.RUNTIME)
@interface MyAnnotation {
    // this annotation can be accessed with reflections at runtime
}
```

RetentionPolicy	
	`.class` .
	.
	`.class` . .

@Documented @Documented

`@Documented` javadoc API . . `@Documented` . `@Documented` .

@Inherited

`@Inherited` meta-annotation . . `@Inherited` .

-

.

- , ().

. .

@

`@Repeatable` Java 8 . . .

Annotation

Reflection Annotation Method Field Class Annotation .

```
@Retention(RetentionPolicy.RUNTIME)
@interface MyAnnotation {
    String key() default "foo";
    String value() default "bar";
}


class AnnotationExample {
    // Put the Annotation on the method, but leave the defaults
    @MyAnnotation
    public void testDefaults() throws Exception {
        // Using reflection, get the public method "testDefaults", which is this method with
no args
        Method method = AnnotationExample.class.getMethod("testDefaults", null);

        // Fetch the Annotation that is of type MyAnnotation from the Method
        MyAnnotation annotation = (MyAnnotation)method.getAnnotation(MyAnnotation.class);

        // Print out the settings of the Annotation
        print(annotation);
    }

    //Put the Annotation on the method, but override the settings
    @MyAnnotation(key="baz", value="buzz")
    public void testValues() throws Exception {
        // Using reflection, get the public method "testValues", which is this method with no
args
        Method method = AnnotationExample.class.getMethod("testValues", null);

        // Fetch the Annotation that is of type MyAnnotation from the Method
        MyAnnotation annotation = (MyAnnotation)method.getAnnotation(MyAnnotation.class);

        // Print out the settings of the Annotation
        print(annotation);
    }

    public void print(MyAnnotation annotation) {
        // Fetch the MyAnnotation 'key' & 'value' properties, and print them out
        System.out.println(annotation.key() + " = " + annotation.value());
    }

    public static void main(String[] args) {
        AnnotationExample example = new AnnotationExample();
        try {
```

```
            example.testDefaults();
            example.testValues();
        } catch( Exception e ) {
            // Shouldn't throw any Exceptions
            System.err.println("Exception [" + e.getClass().getName() + "] - " +
e.getMessage());
            e.printStackTrace(System.err);
        }
    }
}
```

```
foo = bar
baz = buzz
```

Java 8 . .

```
// Author.java
@Retention(RetentionPolicy.RUNTIME)
public @interface Author {
    String value();
}

// Authors.java
@Retention(RetentionPolicy.RUNTIME)
public @interface Authors {
    Author[] value();
}

// Test.java
@Authors({
    @Author("Mary"),
    @Author("Sam")
})
public class Test {
    public static void main(String[] args) {
        Author[] authors = Test.class.getAnnotation(Authors.class).value();
        for (Author author : authors) {
            System.out.println(author.value());
            // Output:
            // Mary
            // Sam
        }
    }
}
```

Java SE 8

Java 8 `@Repeatable` . `Author` .

```
@Repeatable(Authors.class)
```

`@Author` `@Author` `@Authors` . `Class.getAnnotationsByType()` `@Author` .

```
@Author("Mary")
@Author("Sam")
public class Test {
```

```
    public static void main(String[] args) {
        Author[] authors = Test.class.getAnnotationsByType(Author.class);
        for (Author author : authors) {
            System.out.println(author.value());
            // Output:
            // Mary
            // Sam
        }
    }
}
```

. @Inherited .

2 .

```
@Inherited
@Target(ElementType.TYPE)
@Retention(RetentionPolicy.RUNTIME)
public @interface InheritedAnnotationType {
}
```

```
@Target(ElementType.TYPE)
@Retention(RetentionPolicy.RUNTIME)
public @interface UninheritedAnnotationType {
}
```

3 :

```
@UninheritedAnnotationType
class A {
}

@InheritedAnnotationType
class B extends A {
}

class C extends B {
}
```

```
System.out.println(new A().getClass().getAnnotation(InheritedAnnotationType.class));
System.out.println(new B().getClass().getAnnotation(InheritedAnnotationType.class));
System.out.println(new C().getClass().getAnnotation(InheritedAnnotationType.class));
System.out.println("_____");
System.out.println(new A().getClass().getAnnotation(UninheritedAnnotationType.class));
System.out.println(new B().getClass().getAnnotation(UninheritedAnnotationType.class));
System.out.println(new C().getClass().getAnnotation(UninheritedAnnotationType.class));
```

.

```
null
@InheritedAnnotationType()
@InheritedAnnotationType()
_____
@UninheritedAnnotationType()
null
```

```
null
```

.

.

`@Setter` . .

```
package annotation;

import java.lang.annotation.ElementType;
import java.lang.annotation.Retention;
import java.lang.annotation.RetentionPolicy;
import java.lang.annotation.Target;

@Retention(RetentionPolicy.SOURCE)
@Target(ElementType.METHOD)
public @interface Setter {
}
```

`SetterProcessor` . `@Setter` `public`, `static` `set`, 4 . `Messager`. stderr . , NetBeans IDE .

```
package annotation.processor;

import annotation.Setter;
import java.util.Set;
import javax.annotation.processing.AbstractProcessor;
import javax.annotation.processing.Messager;
import javax.annotation.processing.ProcessingEnvironment;
import javax.annotation.processing.RoundEnvironment;
import javax.annotation.processing.SupportedAnnotationTypes;
import javax.annotation.processing.SupportedSourceVersion;
import javax.lang.model.SourceVersion;
import javax.lang.model.element.Element;
import javax.lang.model.element.ElementKind;
import javax.lang.model.element.ExecutableElement;
import javax.lang.model.element.Modifier;
import javax.lang.model.element.TypeElement;
import javax.tools.Diagnostic;

@SupportedAnnotationTypes({"annotation.Setter"})
@SupportedSourceVersion(SourceVersion.RELEASE_8)
public class SetterProcessor extends AbstractProcessor {

    private Messager messager;

    @Override
    public boolean process(Set<? extends TypeElement> annotations, RoundEnvironment roundEnv)
{
        // get elements annotated with the @Setter annotation
        Set<? extends Element> annotatedElements =
roundEnv.getElementsAnnotatedWith(Setter.class);
```

```
        for (Element element : annotatedElements) {
            if (element.getKind() == ElementKind.METHOD) {
                // only handle methods as targets
                checkMethod((ExecutableElement) element);
            }
        }

        // don't claim annotations to allow other processors to process them
        return false;
    }

    private void checkMethod(ExecutableElement method) {
        // check for valid name
        String name = method.getSimpleName().toString();
        if (!name.startsWith("set")) {
            printError(method, "setter name must start with \"set\"");
        } else if (name.length() == 3) {
            printError(method, "the method name must contain more than just \"set\"");
        } else if (Character.isLowerCase(name.charAt(3))) {
            if (method.getParameters().size() != 1) {
                printError(method, "character following \"set\" must be upper case");
            }
        }

        // check, if setter is public
        if (!method.getModifiers().contains(Modifier.PUBLIC)) {
            printError(method, "setter must be public");
        }

        // check, if method is static
        if (method.getModifiers().contains(Modifier.STATIC)) {
            printError(method, "setter must not be static");
        }
    }

    private void printError(Element element, String message) {
        messager.printMessage(Diagnostic.Kind.ERROR, message, element);
    }

    @Override
    public void init(ProcessingEnvironment processingEnvironment) {
        super.init(processingEnvironment);

        // get messager for printing errors
        messager = processingEnvironment.getMessager();
    }

}
```

SPI (ServiceLoader).

`META-INF/services/javax.annotation.processing.Processor` jar . . , .

```
annotation.processor.SetterProcessor
```

jar **`AnnotationProcessor.jar`** .

. .

```
import annotation.Setter;

public class AnnotationProcessorTest {

    @Setter
    private void setValue(String value) {}

    @Setter
    public void setString(String value) {}

    @Setter
    public static void main(String[] args) {}

}
```

javac

SPI . : AnnotationProcessorTest

```
javac -cp AnnotationProcessor.jar AnnotationProcessorTest.java
```

.

```
AnnotationProcessorTest.java:6: error: setter must be public
    private void setValue(String value) {}
                 ^
AnnotationProcessorTest.java:12: error: setter name must start with "set"
    public static void main(String[] args) {}
                       ^
2 errors
```

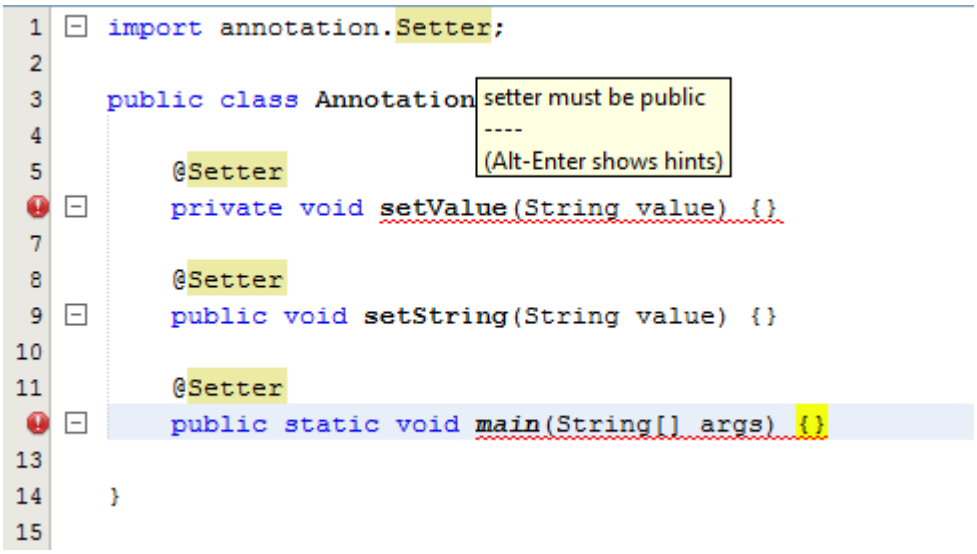
.class .

javac -proc:none . -proc:only -proc:only .

IDE

NetBeans . .

1. Project Properties > Build > Compiling Build
2. Enable Annotation Processing Enable Annotation Processing in Editor Enable Annotation Processing in Editor .
3. Add Add
4. Ok .



Java .

.

. .
" " `@Target` . " " .

. Spring Spring-MVC .

. JPA Java (SQL) .

.

- ?
- ?
- ?

'this'

Java . Java 8 . JLS 8.4.1 .

receiver . , receiver . receiver . , . receiver .
(4.12.3) .

.

```
public class Outer {
    public class Inner {
        public Inner (Outer this) {
           // ...
        }
        public void doIt(Inner this) {
           // ...
        }
    }
```

```
}
```

. , Closeable @IsOpen @IsOpen . :

```
public class MyResource extends Closeable {
    public void update(@IsOpen MyResource this, int value) {
        // ...
    }

    public void close() {
        // ...
    }
}
```

,@IsOpen this . . :

- update this .
- update this .

Annotation . @SuppressWarnings .

```
public @interface SuppressWarnings {
    String[] value();
}
```

value . .

```
@SuppressWarnings({"unused"})
@SuppressWarnings({"unused", "javadoc"})
```

.

```
@SuppressWarnings("unused")
```

: https://riptutorial.com/ko/java/topic/157/-

177: I / O

Java I / O ( ) . Java I / O . java.io . Java I / O API Java .

Examples

byte [] .

Java 7 Files .

Java SE 7

```
import java.nio.file.Files;
import java.nio.file.Paths;
import java.nio.file.Path;

Path path = Paths.get("path/to/file");

try {
    byte[] data = Files.readAllBytes(path);
} catch(IOException e) {
    e.printStackTrace();
}
```

```
import java.awt.Image;
import javax.imageio.ImageIO;

...

try {
    Image img = ImageIO.read(new File("~/Desktop/cat.png"));
} catch (IOException e) {
    e.printStackTrace();
}
```

[]

Java SE 7

```
byte[] bytes = { 0x48, 0x65, 0x6c, 0x6c, 0x6f };

try(FileOutputStream stream = new FileOutputStream("Hello world.txt")) {
    stream.write(bytes);
} catch (IOException ioe) {
    // Handle I/O Exception
    ioe.printStackTrace();
}
```

Java SE 7

```
byte[] bytes = { 0x48, 0x65, 0x6c, 0x6c, 0x6f };
```

```
FileOutputStream stream = null;
try {
    stream = new FileOutputStream("Hello world.txt");
    stream.write(bytes);
} catch (IOException ioe) {
    // Handle I/O Exception
    ioe.printStackTrace();
} finally {
    if (stream != null) {
        try {
            stream.close();
        } catch (IOException ignored) {}
    }
}
```

java.io API `String` `File`

```
File file = new File("Hello world.txt");
FileOutputStream stream = new FileOutputStream(file);
```

Stream vs Writer / Reader API

`InputStream` / `OutputStream` `int` `byte` .

```
// Read a single byte from the stream
int b = inputStream.read();
if (b >= 0) { // A negative value represents the end of the stream, normal values are in the
range 0 - 255
    // Write the byte to another stream
    outputStream.write(b);
}

// Read a chunk
byte[] data = new byte[1024];
int nBytesRead = inputStream.read(data);
if (nBytesRead >= 0) { // A negative value represents end of stream
    // Write the chunk to another stream
    outputStream.write(data, 0, nBytesRead);
}
```

. `PrintStream` " " . `System.out` `InputStream` `System.out.println()` .

, Java ( ) (: `DataOutputStream` / `DataInputStream`) .

`Writer` `Reader` Java API . API .

```
// This example uses the platform's default charset, see below
// for a better implementation.

Writer writer = new OutputStreamWriter(System.out);
writer.write("Hello world!");

Reader reader = new InputStreamReader(System.in);
char singleCharacter = reader.read();
```

(: `InputStreamWriter` / `OutputStreamWriter`) charset charset . Java UTF-8 . `FileWriter` `FileReader` `FileReader` . .

```
Charset myCharset = StandardCharsets.UTF_8;

Writer writer = new OutputStreamWriter( new FileOutputStream("test.txt"), myCharset );
writer.write('Ä');
writer.flush();
writer.close();

Reader reader = new InputStreamReader( new FileInputStream("test.txt"), myCharset );
char someUnicodeCharacter = reader.read();
reader.close();
```

`Reader` `BufferedReader` .

```
// Read from baseReader, one line at a time
BufferedReader reader = new BufferedReader( baseReader );
String line;
while((line = reader.readLine()) != null) {
  // Remember: System.out is a stream, not a writer!
  System.out.println(line);
}
```

```
File f = new File(path);
String content = new Scanner(f).useDelimiter("\\Z").next();
```

\ Z EOF ( ) . EOF .

```
public class Main {

    public static void main(String[] args) {
        try {
            Scanner scanner = new Scanner(new File("example.txt"));
            while(scanner.hasNextLine())
            {
                String line = scanner.nextLine();
                //do stuff
            }
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }
    }
}
```

```
public class Main {

    public static void main(String[] args) {
        try {
            Scanner scanner = new Scanner(new File("example.txt"));
            while(scanner.hasNext())
            {
                String line = scanner.next();
                //do stuff
            }
        } catch (FileNotFoundException e) {
```

```
            e.printStackTrace();
        }
    }
}
```

scanner.useDelimeter () delimeter

```
    public void iterateAndFilter() throws IOException {
        Path dir = Paths.get("C:/foo/bar");
        PathMatcher imageFileMatcher =
            FileSystems.getDefault().getPathMatcher(
                "regex:.*(?i:jpg|jpeg|png|gif|bmp|jpe|jfif)");

        try (DirectoryStream<Path> stream = Files.newDirectoryStream(dir,
                entry -> imageFileMatcher.matches(entry.getFileName()))) {

            for (Path path : stream) {
                System.out.println(path.getFileName());
            }
        }
    }
```

java.io.File Java 7 NIO (java.nio.file.Path)

Java 7 NIO `java.io.File` . NIO .

FileSystem ZIP JAR Java 7 NIO . .

`java.io.File` / `java.nio.file.Files` .

.

```
// -> IO
File file = new File("io.txt");

// -> NIO
Path path = Paths.get("nio.txt");
```

```
// Forward slashes can be used in place of backslashes even on a Windows operating system
// -> IO
File folder = new File("C:/");
File fileInFolder = new File(folder, "io.txt");

// -> NIO
Path directory = Paths.get("C:/");
Path pathInDirectory = directory.resolve("nio.txt");
```

/

```
// -> IO to NIO
Path pathFromFile = new File("io.txt").toPath();

// -> NIO to IO
File fileFromPath = Paths.get("nio.txt").toFile();
```

-

```
// -> IO
if (file.exists()) {
    boolean deleted = file.delete();
    if (!deleted) {
        throw new IOException("Unable to delete file");
    }
}

// -> NIO
Files.deleteIfExists(path);
```

OutputStream

`FileChannel` , `Files.write(Path path, byte\[\] bytes, OpenOption... options)` , NIO `Files.write(Path path, byte\[\] bytes, OpenOption... options)` .. OutputStream java.nio.file.Files .

```
List<String> lines = Arrays.asList(
        String.valueOf(Calendar.getInstance().getTimeInMillis()),
        "line one",
        "line two");

// -> IO
if (file.exists()) {
    // Note: Not atomic
    throw new IOException("File already exists");
}
try (FileOutputStream outputStream = new FileOutputStream(file)) {
    for (String line : lines) {
        outputStream.write((line + System.lineSeparator()).getBytes(StandardCharsets.UTF_8));
    }
}

// -> NIO
try (OutputStream outputStream = Files.newOutputStream(path, StandardOpenOption.CREATE_NEW)) {
    for (String line : lines) {
        outputStream.write((line + System.lineSeparator()).getBytes(StandardCharsets.UTF_8));
    }
}
```

```
// -> IO
for (File selectedFile : folder.listFiles()) {
    // Note: Depending on the number of files in the directory folder.listFiles() may take a
```

```
long time to return
    System.out.println((selectedFile.isDirectory() ? "d" : "f") + " " +
selectedFile.getAbsolutePath());
}

// -> NIO
Files.walkFileTree(directory, EnumSet.noneOf(FileVisitOption.class), 1, new
SimpleFileVisitor<Path>() {
    @Override
    public FileVisitResult preVisitDirectory(Path selectedPath, BasicFileAttributes attrs)
throws IOException {
        System.out.println("d " + selectedPath.toAbsolutePath());
        return FileVisitResult.CONTINUE;
    }

    @Override
    public FileVisitResult visitFile(Path selectedPath, BasicFileAttributes attrs) throws
IOException {
        System.out.println("f " + selectedPath.toAbsolutePath());
        return FileVisitResult.CONTINUE;
    }
});
```

```
// -> IO
recurseFolder(folder);

// -> NIO
// Note: Symbolic links are NOT followed unless explicitly passed as an argument to
Files.walkFileTree
Files.walkFileTree(directory, new SimpleFileVisitor<Path>() {
    @Override
    public FileVisitResult preVisitDirectory(Path dir, BasicFileAttributes attrs) throws
IOException {
        System.out.println("d " + selectedPath.toAbsolutePath());
        return FileVisitResult.CONTINUE;
    }

    @Override
    public FileVisitResult visitFile(Path selectedPath, BasicFileAttributes attrs) throws
IOException {
        System.out.println("f " + selectedPath.toAbsolutePath());
        return FileVisitResult.CONTINUE;
    }
});


private static void recurseFolder(File folder) {
    for (File selectedFile : folder.listFiles()) {
        System.out.println((selectedFile.isDirectory() ? "d" : "f") + " " +
selectedFile.getAbsolutePath());
        if (selectedFile.isDirectory()) {
            // Note: Symbolic links are followed
            recurseFolder(selectedFile);
        }
    }
}
```

FileInputStream / FileOutputStream /

test.txt :

```
String filepath ="C:\\test.txt";
FileOutputStream fos = null;
try {
      fos = new FileOutputStream(filepath);
      byte[] buffer = "This will be written in test.txt".getBytes();
      fos.write(buffer, 0, buffer.length);
      fos.close();
} catch (FileNotFoundException e) {
      e.printStackTrace();
} catch (IOException e) {
      e.printStackTrace();
} finally{
      if(fos != null)
        fos.close();
}
```

test.txt :

```
String filepath ="C:\\test.txt";
FileInputStream fis = null;
try {
   fis = new FileInputStream(filepath);
   int length = (int) new File(filepath).length();
   byte[] buffer = new byte[length];
   fis.read(buffer, 0, length);
} catch (FileNotFoundException e) {
     e.printStackTrace();
} catch (IOException e) {
     e.printStackTrace();
} finally{
   if(fis != null)
     fis.close();
}
```

Java 1.7 try-with-resources / .

test.txt :

```
String filepath ="C:\\test.txt";
try (FileOutputStream fos = new FileOutputStream(filepath)){
    byte[] buffer = "This will be written in test.txt".getBytes();
    fos.write(buffer, 0, buffer.length);
} catch (FileNotFoundException e) {
    e.printStackTrace();
} catch (IOException e) {
    e.printStackTrace();
}
```

test.txt :

```
String filepath ="C:\\test.txt";
```

```
try (FileInputStream fis = new FileInputStream(filepath)){
    int length = (int) new File(filepath).length();
    byte[] buffer = new byte[length];
    fis.read(buffer, 0, length);
} catch (FileNotFoundException e) {
    e.printStackTrace();
} catch (IOException e) {
    e.printStackTrace();
}
```

Java .

Java SE 1.4

```
File file = new File("path_to_the_file");
byte[] data = new byte[(int) file.length()];
DataInputStream stream = new DataInputStream(new FileInputStream(file));
stream.readFully(data);
stream.close();
```

Java 7 `nio API` .

Java SE 7

```
Path path = Paths.get("path_to_the_file");
byte [] data = Files.readAllBytes(path);
```

`streams readers FileChannel` API .

`streams`

// FileInputStream ios = new FileInputStream (filename);

```
    // get underlying channel
    FileChannel channel = ios.getChannel();

    /*
     * try to lock the file. true means whether the lock is shared or not i.e. multiple
processes can acquire a
     * shared lock (for reading only) Using false with readable channel only will generate an
exception. You should
     * use a writable channel (taken from FileOutputStream) when using false. tryLock will
always return immediately
     */
    FileLock lock = channel.tryLock(0, Long.MAX_VALUE, true);

    if (lock == null) {
        System.out.println("Unable to acquire lock");
    } else {
        System.out.println("Lock acquired successfully");
    }

    // you can also use blocking call which will block until a lock is acquired.
    channel.lock();

    // Once you have completed desired operations of file. release the lock
```

```
    if (lock != null) {
        lock.release();
    }

    // close the file stream afterwards
    // Example with reader
    RandomAccessFile randomAccessFile = new RandomAccessFile(filename,  "rw");
    FileChannel channel = randomAccessFile.getChannel();
    //repeat the same steps as above but now you can use shared as true or false as the
channel is in read write mode
```

InputStream OutputStream

. InputStream OutputStream . .

```
public void copy(InputStream source, OutputStream destination) throws IOException {
    try {
        int c;
        while ((c = source.read()) != -1) {
            destination.write(c);
        }
    } finally {
        if (source != null) {
            source.close();
        }
        if (destination != null) {
            destination.close();
        }
    }
}
```

`Channel` `Buffer` . . `Channel` I/O .

`Channel` .

1. `FileInputStream` . `FileInputStream` `getChannel()` .
2. FileInputStream `getChannel()` Channel .
3. ByteBuffer . ByteBuffer .
4. read , read ByteBuffer . ByteBuffer . `flip()` . , . 0 . . . 0 . flip () .
5. `Channel` read .
6. `ByteBuffer` .
7. `read()` 0 -1 .

```
import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import java.nio.ByteBuffer;
import java.nio.channels.FileChannel;

public class FileChannelRead {

public static void main(String[] args) {

   File inputFile = new File("hello.txt");
```

```
   if (!inputFile.exists()) {
    System.out.println("The input file doesn't exit.");
    return;
   }

  try {
   FileInputStream fis = new FileInputStream(inputFile);
   FileChannel fileChannel = fis.getChannel();
   ByteBuffer buffer = ByteBuffer.allocate(1024);

   while (fileChannel.read(buffer) > 0) {
    buffer.flip();
    while (buffer.hasRemaining()) {
     byte b = buffer.get();
     System.out.print((char) b);
    }
    buffer.clear();
   }

   fileChannel.close();
  } catch (IOException e) {
   e.printStackTrace();
  }
 }
}
```

Channel . FileChannel transferTo() .

```
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.nio.channels.FileChannel;

public class FileCopier {

    public static void main(String[] args) {
        File sourceFile = new File("hello.txt");
        File sinkFile = new File("hello2.txt");
        copy(sourceFile, sinkFile);
    }

    public static void copy(File sourceFile, File destFile) {
        if (!sourceFile.exists() || !destFile.exists()) {
            System.out.println("Source or destination file doesn't exist");
            return;
        }


        try (FileChannel srcChannel = new FileInputStream(sourceFile).getChannel();
             FileChannel sinkChanel = new FileOutputStream(destFile).getChannel()) {

            srcChannel.transferTo(0, srcChannel.size(), sinkChanel);
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

BufferedInputStream

FileInputStream BufferedInputStream .

```
import java.io.BufferedInputStream;
import java.io.FileInputStream;
import java.io.IOException;


public class FileReadingDemo {

    public static void main(String[] args) {
        String source = "hello.txt";

        try (BufferedInputStream bis = new BufferedInputStream(new FileInputStream(source))) {
            byte data;
            while ((data = (byte) bis.read()) != -1) {
                System.out.println((char) data);
            }
        } catch (IOException e) {
            e.printStackTrace();
        }

    }
}
```

Channel .

1. FileOutputStream FileOutputStream
2. FileOutputStream getChannel() FileChannel
3. ByteBuffer .
4. ByteBuffer flip() FileChannel FileChannel write() .
5. .

```
import java.io.*;
import java.nio.*;
public class FileChannelWrite {

 public static void main(String[] args) {

  File outputFile = new File("hello.txt");
  String text = "I love Bangladesh.";

  try {
   FileOutputStream fos = new FileOutputStream(outputFile);
   FileChannel fileChannel = fos.getChannel();
   byte[] bytes = text.getBytes();
   ByteBuffer buffer = ByteBuffer.wrap(bytes);
   fileChannel.write(buffer);
   fileChannel.close();
  } catch (java.io.IOException e) {
   e.printStackTrace();
  }
 }
}
```

PrintStream

PrintStream . .println() . PrintStream PrintStream .

```
import java.io.FileNotFoundException;
import java.io.PrintStream;
import java.time.LocalDate;

public class FileWritingDemo {
    public static void main(String[] args) {
        String destination = "file1.txt";

        try(PrintStream ps = new PrintStream(destination)){
            ps.println("Stackoverflow documentation seems fun.");
            ps.println();
            ps.println("I love Java!");
            ps.printf("Today is: %1$tm/%1$td/%1$tY", LocalDate.now());

            ps.flush();
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }

    }
}
```

.

```
  public void iterate(final String dirPath) throws IOException {
    final DirectoryStream<Path> paths = Files.newDirectoryStream(Paths.get(dirPath));
    for (final Path path : paths) {
      if (Files.isDirectory(path)) {
        System.out.println(path.getFileName());
      }
    }
  }
```

File mkdirs() mkdir() .

- mkdir() - .( )
- mkdirs() - . .( )

: createNewFile() .

```
File singleDir = new File("C:/Users/SomeUser/Desktop/A New Folder/");

File multiDir = new File("C:/Users/SomeUser/Desktop/A New Folder 2/Another Folder/");

// assume that neither "A New Folder" or "A New Folder 2" exist

singleDir.createNewFile(); // will make a new file called "A New Folder.file"
singleDir.mkdir(); // will make the directory
singleDir.mkdirs(); // will make the directory

multiDir.createNewFile(); // will throw a IOException
```

```
multiDir.mkdir(); // will not work
multiDir.mkdirs(); // will make the directory
```

/

`System.out` `System.err` . ( ) .

.

, MacOSX `>` . :

```
$ java -jar app.jar arg1 arg2 > /dev/null 2>&1
$ java -jar app.jar arg1 arg2 > out.log 2> error.log
```

"/ dev / null" . "/ dev / null" . "out.log", "error.log" .

. Windows .

Java .

Java

`System.setOut()` `System.setErr()` Java . 2 .

```
System.setOut(new PrintStream(new FileOutputStream(new File("out.log"))));
System.setErr(new PrintStream(new FileOutputStream(new File("err.log"))));
```

"" . UNIX "/ dev / null" .

```
System.setOut(new PrintStream(new FileOutputStream(new FileDescriptor())));
System.setErr(new PrintStream(new FileOutputStream(new FileDescriptor())));
```

: `setOut` `setErr` .

1. JVM .
2. .

ZIP

Java 7 FileSystem API Java NIO API Zip .

FileSystem try-with-resources .

```
Path pathToZip = Paths.get("path/to/file.zip");
try(FileSystem zipFs = FileSystems.newFileSystem(pathToZip, null)) {
  Path root = zipFs.getPath("/");
  ... //access the content of the zip file same as ordinary files
} catch(IOException ex) {
  ex.printStackTrace();
}
```

```
Map<String, String> env = new HashMap<>();
env.put("create", "true"); //required for creating a new zip file
env.put("encoding", "UTF-8"); //optional: default is UTF-8
URI uri = URI.create("jar:file:/path/to/file.zip");
try (FileSystem zipfs = FileSystems.newFileSystem(uri, env)) {
  Path newFile = zipFs.getPath("/newFile.txt");
  //writing to file
  Files.write(newFile, "Hello world".getBytes());
} catch(IOException ex) {
  ex.printStackTrace();
}
```

I / O : https://riptutorial.com/ko/java/topic/93/-i---o

178:

java . . / . . > 1.java.util> 2.java.lang> 3.java.io .

.

package . .

.

Examples

Test.class :

```
package foo.bar

public class Test {

}
```

Test.class

```
package foo.bar.baz

public class Test {

}
```

.

Java . , .

```
package foo.bar

public class ExampleClass {
    double exampleNumber;
    String exampleString;

    public ExampleClass() {
        exampleNumber = 3;
        exampleString = "Test String";
    }
    //No getters or setters
}

package foo.bar

public class AnotherClass {
    ExampleClass clazz = new ExampleClass();

    System.out.println("Example Number: " + clazz.exampleNumber);
    //Prints Example Number: 3
```

```
    System.out.println("Example String: " + clazz.exampleString);
    //Prints Example String: Test String
}
```

.

```
package baz.foo

public class ThisShouldNotWork {
    ExampleClass clazz = new ExampleClass();

    System.out.println("Example Number: " + clazz.exampleNumber);
    //Throws an exception
    System.out.println("Example String: " + clazz.exampleString);
    //Throws an exception
}
```

: https://riptutorial.com/ko/java/topic/8273/

179: /

Examples

Java /

Java fork / join . / .

- .
- .
- .

ForkJoinTask . (RecursiveTask) .

10 .

```
import java.util.List;
import java.util.concurrent.RecursiveTask;

public class SummingTask extends RecursiveTask<Integer> {
    private static final int MAX_BATCH_SIZE = 10;

    private final List<Integer> numbers;
    private final int minInclusive, maxExclusive;

    public SummingTask(List<Integer> numbers) {
        this(numbers, 0, numbers.size());
    }

    // This constructor is only used internally as part of the dividing process
    private SummingTask(List<Integer> numbers, int minInclusive, int maxExclusive) {
        this.numbers = numbers;
        this.minInclusive = minInclusive;
        this.maxExclusive = maxExclusive;
    }

    @Override
    public Integer compute() {
        if (maxExclusive - minInclusive > MAX_BATCH_SIZE) {
            // This is too big for a single batch, so we shall divide into two tasks
            int mid = (minInclusive + maxExclusive) / 2;
            SummingTask leftTask = new SummingTask(numbers, minInclusive, mid);
            SummingTask rightTask = new SummingTask(numbers, mid, maxExclusive);

            // Submit the left hand task as a new task to the same ForkJoinPool
            leftTask.fork();

            // Run the right hand task on the same thread and get the result
            int rightResult = rightTask.compute();

            // Wait for the left hand task to complete and get its result
            int leftResult = leftTask.join();

            // And combine the result
            return leftResult + rightResult;
```

```
        } else {
            // This is fine for a single batch, so we will run it here and now
            int sum = 0;
            for (int i = minInclusive; i < maxExclusive; i++) {
                sum += numbers.get(i);
            }
            return sum;
        }
    }
}
```

ForkJoinPool .

```
// Because I am not specifying the number of threads
// it will create a thread for each available processor
ForkJoinPool pool = new ForkJoinPool();

// Submit the task to the pool, and get what is effectively the Future
ForkJoinTask<Integer> task = pool.submit(new SummingTask(numbers));

// Wait for the result
int result = task.join();
```

/ : https://riptutorial.com/ko/java/topic/4245/--------

180:

Java .

, .

Examples

. , `1 + 2 x 3` .

1. `1 2 3 . 9 . ( 1 + 2 ) x 3 .`
2. `2 3 1 . 7 . 1 + ( 2 x 3 ) .`

. . ""(!) . .

Java .

. .

:

```
1 + 2 * 3
```

`+` `*` 9 7.

	/ ()		
	. (expr) new . [expr] (expr, ...) ::	15	
	expr ++ , expr --	14	-
1	++ expr, -- expr, + EXPR, - EXPR, ~ EXPR, ! expr, ( ) expr	13	-
Multiplicative	* / %	12	
	+ -	11	
	<< >> >>>	10	
	<> <=> = instanceof	9	

	/ ()		
	==! =	8	
AND	&	7	
	^	6	
OR	|	5	
AND	&&	4	
OR	||		
1	? :	2	
1	= = = = = = = = = = = << = >> = >>> = & = ^ = = = ->	1	

1 . .

. .

- .
- `String` .
- : `+` , `-` , `~` `!` .
- `*` , `/` , `%` , `+` , `-` , `<<` `>>` , `>>>` , `<` , `<=` , `>` , `>=` , `==` `!=` , `&` , `^` , `|` , `&&` `||` .
- `?` `:` .
- .
- . (`final` .)
- `<TypeName> . <Identifier>` `<TypeName> . <Identifier>` .

`++` `--` , , `class` `instanceof` , .

`String` "interned" `String` FP .

. .

switch case . :

```
switch (someValue) {
case 1 + 1:            // OK
case Math.min(2, 3):   // Error - not a constant expression
    doSomething();
}
```

. . (JLS 5.1.3 5.2) :

```
byte b1 = 1 + 1;              // OK - primitive narrowing conversion.
byte b2 = 127 + 1;            // Error - out of range
byte b3 = b1 + 1;             // Error - not a constant expession
byte b4 = (byte) (b1 + 1);    // OK
```

do , while for . :

```
while (false) {
    doSomething();            // Error - statenent not reachable
}
boolean flag = false;
while (flag) {
    doSomething();            // OK
}
```

(if . then else if . C C ++ .)

, static final . .

JLS 15.28 . .

Java .

- .
- .
- .
- .

:

```
int i = method1() + method2();
```

.

1. = i .
2. + (method1()) .
3. + (method2()) .
4. + .
5. = i .

method2 method1 .

:

```
int i = 1;
intArray[i] = ++i + 1;
```

.

1. = . intArray[1] .
2. . 1 i 2 .
3. + .
4. + 2 + 1 -> 3 .
5. = 3 intArray[1] .

= ++i .

:

- JLS 15.7 -

Java . :

```
1                // A simple literal is an expression
1 + 2            // A simple expression that adds two numbers
(i + j) / k      // An expression with multiple operations
(flag) ? c : d   // An expression using the "conditional" operator
(String) s       // A type-cast is an expression
obj.test()       // A method call is an expression
new Object()     // Creation of an object is an expression
new int[]        // Creation of an object is an expression
```

.

- :
  - . : someIdentifier
  - ; : MyClass.someField
- :
  - ; : 1 , 1.0 , 'X' , "hello" , false null
  - ; : MyClass.class
  - this <TypeName> . this
  - ; : (a + b)
  - ; : new MyClass(1, 2, 3)
  - ; : new int[3]
  - ; : obj.someField this.someField
  - ; : vector[21]
  - ; : obj.doIt(1, 2, 3)
  - (Java 8). : MyClass::doIt
- ; : !a i++
- ; a + b obj == null
- ; (obj == null) ? 1 : obj.getCount()
- (Java 8); : obj -> obj.getCount()

.

- , 2 3 .
- .
- .
- .

- Literals .

, . . .

(Java 8) .

-
-
-
-
-
-

, . .

. *heap* . " " () .

Java . :

```
public void compute(int i, int j) {
    i + j;   // ERROR
}
```

Java .

. . .

- .
- .
- (`void` `void`).
- .

: https://riptutorial.com/ko/java/topic/8167/

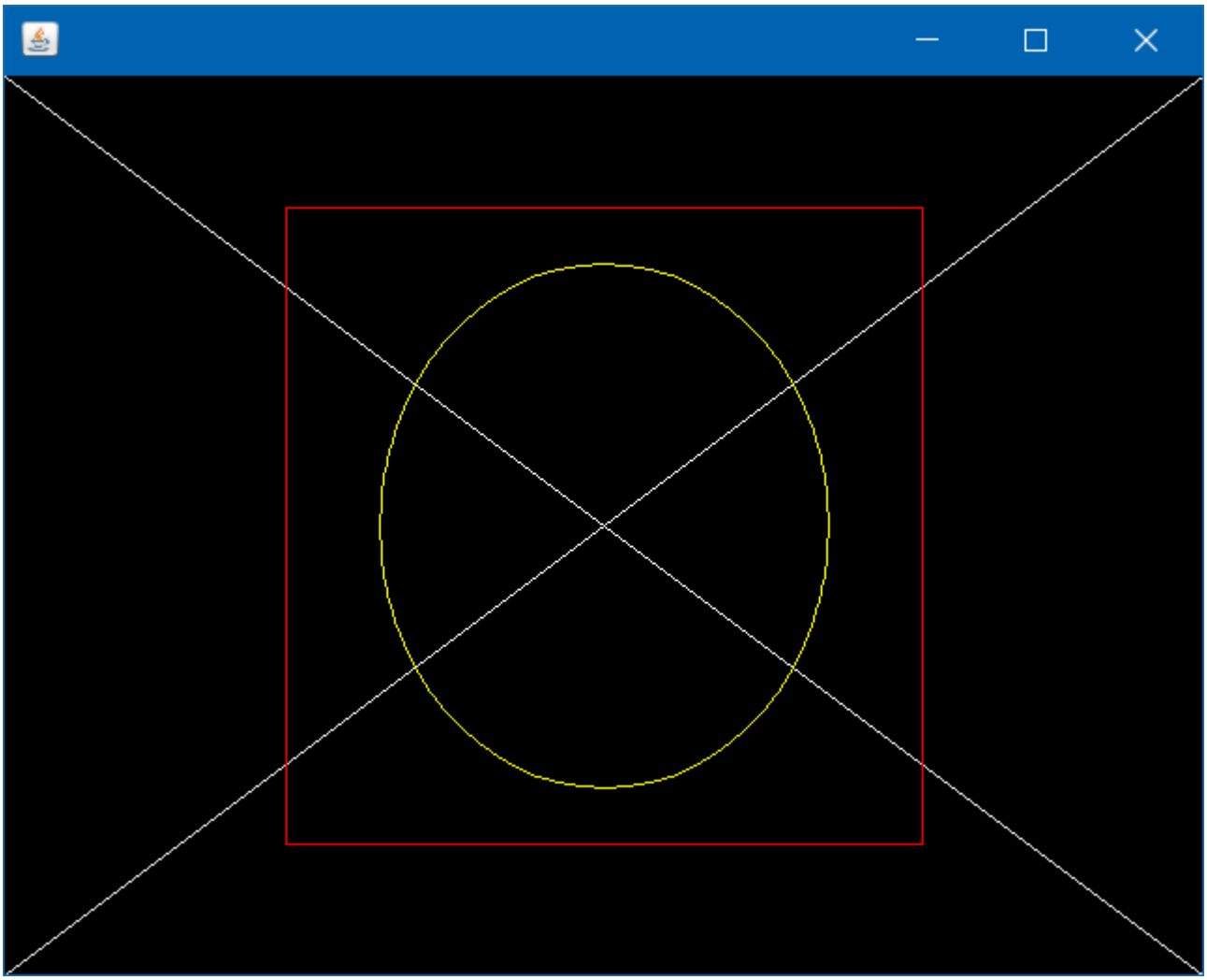
181:

`BufferedImage.getGraphics()` `Graphics2D` .

`VolatileImage` (), , . () , .

Examples

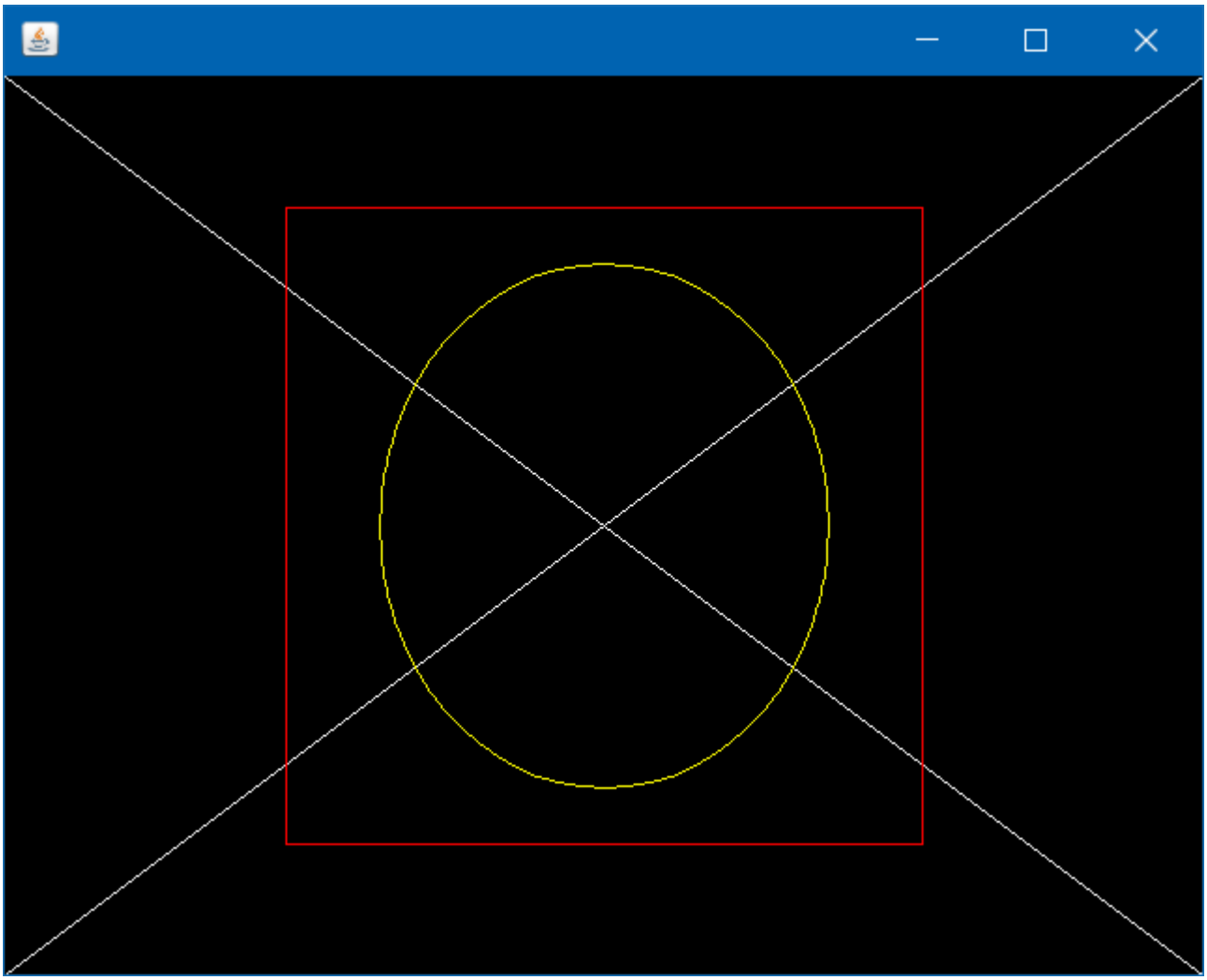
```
class ImageCreationExample {

    static Image createSampleImage() {
        // instantiate a new BufferedImage (subclass of Image) instance
        BufferedImage img = new BufferedImage(640, 480, BufferedImage.TYPE_INT_ARGB);

        //draw something on the image
        paintOnImage(img);

        return img;
    }

    static void paintOnImage(BufferedImage img) {
        // get a drawable Graphics2D (subclass of Graphics) object
        Graphics2D g2d = (Graphics2D) img.getGraphics();

        // some sample drawing
        g2d.setColor(Color.BLACK);
        g2d.fillRect(0, 0, 640, 480);
        g2d.setColor(Color.WHITE);
        g2d.drawLine(0, 0, 640, 480);
        g2d.drawLine(0, 480, 640, 0);
        g2d.setColor(Color.YELLOW);
        g2d.drawOval(200, 100, 240, 280);
        g2d.setColor(Color.RED);
        g2d.drawRect(150, 70, 340, 340);

        // drawing on images can be very memory-consuming
        // so it's better to free resources early
        // it's not necessary, though
        g2d.dispose();
    }

    public static void main(String[] args) {
        JFrame frame = new JFrame();
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        Image img = createSampleImage();
        ImageIcon icon = new ImageIcon(img);
        frame.add(new JLabel(icon));
        frame.pack();
        frame.setVisible(true);
    }
}
```

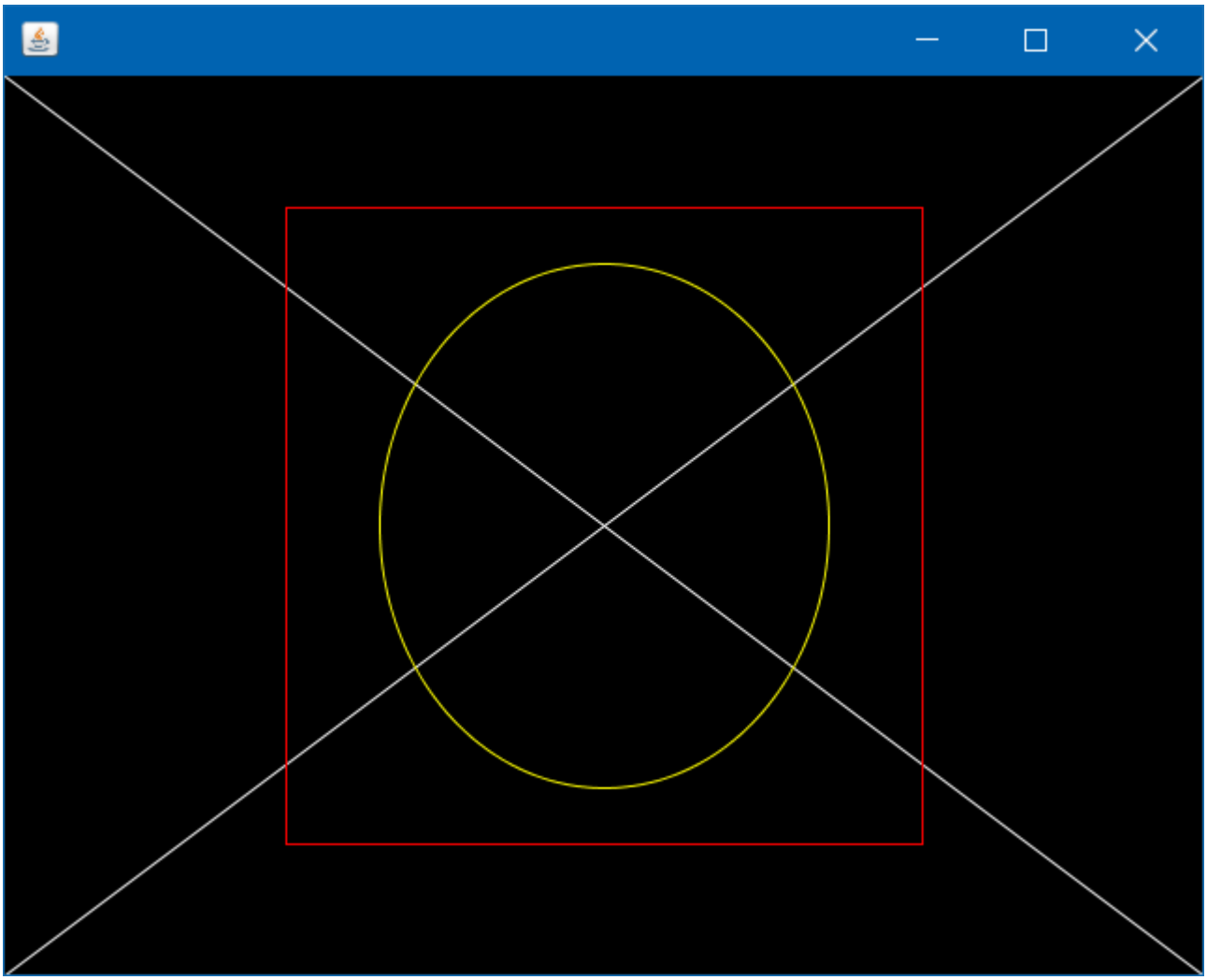


```
public static void saveImage(String destination) throws IOException {
    // method implemented in "Creating a simple image Programmatically and displaying it"
example
    BufferedImage img = createSampleImage();

    // ImageIO provides several write methods with different outputs
    ImageIO.write(img, "png", new File(destination));
}
```

```
static void setupQualityHigh(Graphics2D g2d) {
    g2d.setRenderingHint(RenderingHints.KEY_ANTIALIASING, RenderingHints.VALUE_ANTIALIAS_ON);
    g2d.setRenderingHint(RenderingHints.KEY_RENDERING, RenderingHints.VALUE_RENDER_QUALITY);
    // many other RenderingHints KEY/VALUE pairs to specify
}


static void setupQualityLow(Graphics2D g2d) {
    g2d.setRenderingHint(RenderingHints.KEY_ANTIALIASING, RenderingHints.VALUE_ANTIALIAS_OFF);
    g2d.setRenderingHint(RenderingHints.KEY_RENDERING, RenderingHints.VALUE_RENDER_SPEED);
}
```

QUALITY SPEED :





BufferedImage

```
int width = 256; //in pixels
int height = 256; //in pixels
BufferedImage image = new BufferedImage(width, height, BufferedImage.TYPE_4BYTE_ABGR);
//BufferedImage.TYPE_4BYTE_ABGR - store RGB color and visibility (alpha), see javadoc for more
info

Graphics g = image.createGraphics();

//draw whatever you like, like you would in a drawComponent(Graphics g) method in an UI
application
g.setColor(Color.RED);
g.fillRect(20, 30, 50, 50);

g.setColor(Color.BLUE);
g.drawOval(120, 120, 80, 40);

g.dispose(); //dispose graphics objects when they are no longer needed

//now image has programmatically generated content, you can use it in graphics.drawImage() to
draw it somewhere else
//or just simply save it to a file
ImageIO.write(image, "png", new File("myimage.png"));
```

:

BufferedImage

```
BufferedImage cat = ImageIO.read(new File("cat.jpg")); //read existing file

//modify it
Graphics g = cat.createGraphics();
g.setColor(Color.RED);
g.drawString("Cat", 10, 10);
g.dispose();

//now create a new image
BufferedImage cats = new BufferedImage(256, 256, BufferedImage.TYPE_4BYTE_ABGR);

//and draw the old one on it, 16 times
g = cats.createGraphics();
for (int i = 0; i < 4; i++) {
    for (int j = 0; j < 4; j++) {
        g.drawImage(cat, i * 64, j * 64, null);
    }
}

g.setColor(Color.BLUE);
g.drawRect(0, 0, 255, 255); //add some nice border
g.dispose(); //and done

ImageIO.write(cats, "png", new File("cats.png"));
```

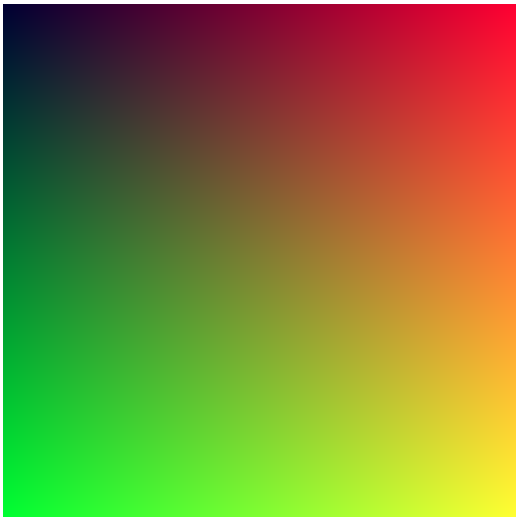
cat :

:

BufferedImage

```
BufferedImage image = new BufferedImage(256, 256, BufferedImage.TYPE_INT_ARGB);

//you don't have to use the Graphics object, you can read and set pixel color individually
for (int i = 0; i < 256; i++) {
    for (int j = 0; j < 256; j++) {
        int alpha = 255; //don't forget this, or use BufferedImage.TYPE_INT_RGB instead
        int red = i; //or any formula you like
        int green = j; //or any formula you like
        int blue = 50; //or any formula you like
        int color = (alpha << 24) | (red << 16) | (green << 8) | blue;
        image.setRGB(i, j, color);
    }
}

ImageIO.write(image, "png", new File("computed.png"));
```

:



BufferedImage

```
/**
* Resizes an image using a Graphics2D object backed by a BufferedImage.
```

```
 * @param srcImg - source image to scale
 * @param w - desired width
 * @param h - desired height
 * @return - the new resized image
 */
private BufferedImage getScaledImage(Image srcImg, int w, int h){

    //Create a new image with good size that contains or might contain arbitrary alpha values
between and including 0.0 and 1.0.
    BufferedImage resizedImg = new BufferedImage(w, h, BufferedImage.TRANSLUCENT);

    //Create a device-independant object to draw the resized image
    Graphics2D g2 = resizedImg.createGraphics();

    //This could be changed, Cf. http://stackoverflow.com/documentation/java/5482/creating-
images-programmatically/19498/specifying-image-rendering-quality
    g2.setRenderingHint(RenderingHints.KEY_INTERPOLATION,
RenderingHints.VALUE_INTERPOLATION_BILINEAR);

    //Finally draw the source image in the Graphics2D with the desired size.
    g2.drawImage(srcImg, 0, 0, w, h, null);

    //Disposes of this graphics context and releases any system resources that it is using
    g2.dispose();

    //Return the image used to create the Graphics2D
    return resizedImg;
}
```

: https://riptutorial.com/ko/java/topic/5482/---

182:

- LocalTime time = LocalTime.now (); // .
- LocalTime time = LocalTime.MIDNIGHT; // 00:00
- LocalTime time = LocalTime.NOON; // 12:00
- LocalTime = LocalTime.of (12, 12, 45); // 12:12:45

`LocalTime.of(13, 12, 11)`	13:12:11
`LocalTime.MIDNIGHT`	00:00
`LocalTime.NOON`	12:00
`LocalTime.now()`	
`LocalTime.MAX`	23 : 59 : 59.999999999
`LocalTime.MIN`	00:00
`LocalTime.ofSecondOfDay(84399)`	23:59:59, 2
`LocalTime.ofNanoOfDay(2000000000)`	00:00:02,

`LocalTime  .  .  .`

`equals .`

java.time .

Examples

`, ,  .`

```
LocalTime time = LocalTime.now();
LocalTime addHours = time.plusHours(5); // Add 5 hours
LocaLTime addMinutes = time.plusMinutes(15) // Add 15 minutes
LocalTime addSeconds = time.plusSeconds(30) // Add 30 seconds
LocalTime addNanoseconds = time.plusNanos(150_000_000) // Add 150.000.000ns (150ms)
```

```
import java.time.LocalTime;
import java.time.ZoneId;
import java.time.temporal.ChronoUnit;

public class Test {
    public static void main(String[] args)
    {
            ZoneId zone1 = ZoneId.of("Europe/Berlin");
            ZoneId zone2 = ZoneId.of("Brazil/East");
```

```
            LocalTime now = LocalTime.now();
            LocalTime now1 = LocalTime.now(zone1);
            LocalTime now2 = LocalTime.now(zone2);

            System.out.println("Current Time : " + now);
            System.out.println("Berlin Time : " + now1);
            System.out.println("Brazil Time : " + now2);

            long minutesBetween = ChronoUnit.MINUTES.between(now2, now1);
            System.out.println("Minutes Between Berlin and Brazil : " + minutesBetween
+"mins");
    }
}
```

LocalTime

`LocalTime` . (1) ~ `until(Temporal, TemporalUnit)` (2) `TemporalUnit.between(Temporal, Temporal)` .

```
import java.time.LocalTime;
import java.time.temporal.ChronoUnit;

public class AmountOfTime {

    public static void main(String[] args) {

        LocalTime start = LocalTime.of(1, 0, 0); // hour, minute, second
        LocalTime end = LocalTime.of(2, 10, 20); // hour, minute, second

        long halfDays1 = start.until(end, ChronoUnit.HALF_DAYS); // 0
        long halfDays2 = ChronoUnit.HALF_DAYS.between(start, end); // 0

        long hours1 = start.until(end, ChronoUnit.HOURS); // 1
        long hours2 = ChronoUnit.HOURS.between(start, end); // 1

        long minutes1 = start.until(end, ChronoUnit.MINUTES); // 70
        long minutes2 = ChronoUnit.MINUTES.between(start, end); // 70

        long seconds1 = start.until(end, ChronoUnit.SECONDS); // 4220
        long seconds2 = ChronoUnit.SECONDS.between(start, end); // 4220

        long millisecs1 = start.until(end, ChronoUnit.MILLIS); // 4220000
        long millisecs2 = ChronoUnit.MILLIS.between(start, end); // 4220000

        long microsecs1 = start.until(end, ChronoUnit.MICROS); // 4220000000
        long microsecs2 = ChronoUnit.MICROS.between(start, end); // 4220000000

        long nanosecs1 = start.until(end, ChronoUnit.NANOS); // 4220000000000
        long nanosecs2 = ChronoUnit.NANOS.between(start, end); // 4220000000000

        // Using others ChronoUnit will be thrown UnsupportedTemporalTypeException.
        // The following methods are examples thereof.
        long days1 = start.until(end, ChronoUnit.DAYS);
        long days2 = ChronoUnit.DAYS.between(start, end);
    }
}
```

LocalTime . hour-min-sec . . , "13 : 45.30.123456789" LocalTime .

. . . equals .

MAX - LocalTime, '23 : 59 : 59.999999999 '. , ,

now (), now ( ), now (zoneId zone), parse (CharSequence text)

isAfter (LocalTime other), isBefore (LocalTime other), (TemporalAmount amountToSubtract), (long amountToSubtract, TemporalUnit), (TemporalAmount amountToAdd), (long amountToAdd, TemporalUnit)

```
ZoneId zone = ZoneId.of("Asia/Kolkata");
LocalTime now = LocalTime.now();
LocalTime now1 = LocalTime.now(zone);
LocalTime then = LocalTime.parse("04:16:40");
```

.

```
long timeDiff = Duration.between(now, now1).toMinutes();
long timeDiff1 = java.time.temporal.ChronoUnit.MINUTES.between(now2, now1);
```

LocalTime , .

minusNanos (long nanosToSubtract), minusSeconds (long secondsToSubtract), plusHours (long hoursToSubtract), plusMinutes (long hoursToMinutes), PlusNanos (long nanosToSubtract), PlusSeconds (long secondsToSubtract)

```
now.plusHours(1L);
now1.minusMinutes(20L);
```

: https://riptutorial.com/ko/java/topic/3065/-

183:

Java , .

. "" .

	20/3/16	1.234, 56	$ 1,000.50	1,000.50 USD	
	20/3/16	1,234.56	£ 1,000.50		100km
	3/20/16	1,234.56	$ 1,000.50	1,000.50 MXN	60

- Wikipedia :

Java

- Java :
- Oracle : : Java
- JavaDoc : `Locale`

Examples

"""

Pinch `SimpleDateFormatter` , .

`"MM/dd/yyyy"` .

`DateFormat` `static` . ( ) .

```
String localizedDate = DateFormat.getDateInstance(style).format(date);
```

`style` `DateFormat` (`FULL` , `LONG` , `MEDIUM` , `SHORT`) .

`, getDateInstance()` .

```
String localizedDate =
    DateFormat.getDateInstance(style, request.getLocale()).format(date);
```

:

```
"School".equalsIgnoreCase("school"); // true
```

.

```
text1.toLowerCase().equals(text2.toLowerCase());
```

. " 'i' . " 'I'. `toLowerCase()` `Locale` : `String.toLowerCase(Locale)` .

:

```
Collator collator = Collator.getInstance(Locale.GERMAN);
collator.setStrength(Collator.PRIMARY);
collator.equals("Gärten", "gaerten"); // returns true
```

(:

```
String[] texts = new String[] {"Birne", "äther", "Apfel"};
Collator collator = Collator.getInstance(Locale.GERMAN);
collator.setStrength(Collator.SECONDARY); // ignore case
Arrays.sort(texts, collator::compare); // will return {"Apfel", "äther", "Birne"}
```

`java.util.Locale` , , ", " . Locale , , . .

Locale , . Locale .

ISO 639 2 3 (8) . 2 3 2 . IANA .

Locale

`java.util.Locale` 4 .

```
Locale constants
Locale constructors
Locale.Builder class
Locale.forLanguageTag factory method
```

Java ResourceBundle

ResourceBundle .

```
Locale locale = new Locale("en", "US");
ResourceBundle labels = ResourceBundle.getBundle("i18n.properties");
System.out.println(labels.getString("message"));
```

`i18n.properties` .

```
message=This is locale
```

:

```
This is locale
```

---

setDefault() . :

```
 setDefault(Locale.JAPANESE); //Set Japanese
```

: https://riptutorial.com/ko/java/topic/4086/--

184:

Examples

classpath JAR :

- javax.money:money-api:1.0 (JSR354 API)
- org.javamoney : moneta : 1.0 ( )
- javax : annotation-api : 1.2. ( )

```
// Let's create non-ISO currency, such as bitcoin

// At first, this will throw UnknownCurrencyException
MonetaryAmount moneys = Money.of(new BigDecimal("0.1"), "BTC");

// This happens because bitcoin is unknown to default currency
// providers
System.out.println(Monetary.isCurrencyAvailable("BTC")); // false

// We will build new currency using CurrencyUnitBuilder provided by org.javamoney.moneta
CurrencyUnit bitcoin = CurrencyUnitBuilder
    .of("BTC", "BtcCurrencyProvider") // Set currency code and currency provider name
    .setDefaultFractionDigits(2)      // Set default fraction digits
    .build(true);                     // Build new currency unit. Here 'true' means
                                      // currency unit is to be registered and
                                      // accessible within default monetary context

// Now BTC is available
System.out.println(Monetary.isCurrencyAvailable("BTC")); // True
```

: https://riptutorial.com/ko/java/topic/8359/-

185:

Examples

Preferences NodeChangeEvent , PreferenceChangeEvent NodeChangeEvent 2 .

PreferenceChangeEvent

PreferenceChangeEvent - Properties . PreferenceChangeEvent , PreferenceChangeListener .

Java SE 8

```
preferences.addPreferenceChangeListener(evt -> {
    String newValue = evt.getNewValue();
    String changedPreferenceKey = evt.getKey();
    Preferences changedNode = evt.getNode();
});
```

Java SE 8

```
preferences.addPreferenceChangeListener(new PreferenceChangeListener() {
    @Override
    public void preferenceChange(PreferenceChangeEvent evt) {
        String newValue = evt.getNewValue();
        String changedPreferenceKey = evt.getKey();
        Preferences changedNode = evt.getNode();
    }
});
```

- .

NodeChangeEvent

Properties .

```
preferences.addNodeChangeListener(new NodeChangeListener() {
    @Override
    public void childAdded(NodeChangeEvent evt) {
        Preferences addedChild = evt.getChild();
        Preferences parentOfAddedChild = evt.getParent();
    }

    @Override
    public void childRemoved(NodeChangeEvent evt) {
        Preferences removedChild = evt.getChild();
        Preferences parentOfRemovedChild = evt.getParent();
    }
});
```

Preferences Preferences .

```
/userRoot
├── com
│   └── mycompany
│       └── myapp
│           ├── darkApplicationMode=true
│           ├── showExitConfirmation=false
│           └── windowMaximized=true
└── org
    └── myorganization
        └── anotherapp
            ├── defaultFont=Helvetica
            ├── defaultSavePath=/home/matt/Documents
            └── exporting
                ├── defaultFormat=pdf
                └── openInBrowserAfterExport=false
```

/com/mycompany/myapp □□□ □□□□□ □□□ □□□□□□.

1. □□□ □□ □□ □□□□ □□□□□□□.

```
package com.mycompany.myapp;

// ...

// Because this class is in the com.mycompany.myapp package, the node
// /com/mycompany/myapp will be returned.
Preferences myApp = Preferences.userNodeForPackage(getClass());
```

2. □□ □□ □□ :

```
Preferences myApp = Preferences.userRoot().node("com/mycompany/myapp");
```

□□ □□ (/ □□□□ □□ □□)□ □□□□ □□ □ □□ □□□ □□□□ □□□□ □□□ □□□□□. □□ □□ □□ □□□□□
/one/two/three/com/mycompany/myapp □□□ □□□ □□□□□.

```
Preferences prefix = Preferences.userRoot().node("one/two/three");
Preferences myAppWithPrefix = prefix.node("com/mycompany/myapp");
// prefix          is /one/two/three
// myAppWithPrefix is /one/two/three/com/mycompany/myapp
```

3. □□ □□ □□ :

```
Preferences myApp = Preferences.userRoot().node("/com/mycompany/myapp");
```

□□ □□□□ □□ □□□ □□□□ □□ □□ □□□ □□□□ □□ □□□ □□□□. □□□□ □□ □□□□ □□ □ □□ □□□ □□ □□□ □□□□ □□□□□
□□□□.

```
Preferences prefix = Preferences.userRoot().node("one/two/three");
Preferences myAppWitoutPrefix = prefix.node("/com/mycompany/myapp");
// prefix            is /one/two/three
// myAppWitoutPrefix is /com/mycompany/myapp
```

□□ □□ □□□□ □□□□□ □□ □□ □□ □□□ □□

Preferences □□ □□□□□ □□ JVM (Java Virtual Machine)□ □□□□□ □□ □□□□□□ □□□□□. Preferences □ □□ JVM□
□ □□ □ □ □□□□ □□ □□□□□ □□ □□□ □□□□□ □□□ □□□ □□□□.

□□ □□□□ □□□ □□□□□□ □□ □□□□□□□ □□ □□ □□□ □ □□ □□ □□□□.

□□ □□□□□□ □□ □□□□□ □□ □□□□□ □□ □□□□□□ □□□□ □□□□□ Preferences □□□□□ □□□□□ JVM□□ □□□□□□□□ □□ Preferences □□ □ **sync()** □□□□ □ □□□□□ Preferences □□□□ □□ □□□□ □□□□ □ □□□□□. □□□□□ □□ JVM□ □□ :

```
// Warning: don't use this if your application is intended
// to only run a single instance on a machine once
// (this is probably the case for most desktop applications)
try {
    preferences.sync();
} catch (BackingStoreException e) {
    // Deal with any errors while saving the preferences to the backing storage
    e.printStackTrace();
}
```

□□ □□ □□□□

Preferences □□□□ □□ □□□□ □□□□□ XML □□□□ □□□□ □ □□□□□. □□ XML □□□□ □□ □□□□ □ □□□□□. □□ XML.□□ □□□□ □□ □□□□ Preferences □□ □ □□ □ □□□□ □ o□□□.

□□ □□ □ □□□□□ □□ □□□□ □□□□□□ □□□□ □□□□□□□.

Java SE 7

```
try (OutputStream os = ...) {
    preferences.exportNode(os);
} catch (IOException ioe) {
    // Exception whilst writing data to the OutputStream
    ioe.printStackTrace();
} catch (BackingStoreException bse) {
    // Exception whilst reading from the backing preferences store
    bse.printStackTrace();
}
```

Java SE 7

```
OutputStream os = null;
try {
    os = ...;
    preferences.exportSubtree(os);
} catch (IOException ioe) {
    // Exception whilst writing data to the OutputStream
    ioe.printStackTrace();
} catch (BackingStoreException bse) {
    // Exception whilst reading from the backing preferences store
    bse.printStackTrace();
} finally {
    if (os != null) {
        try {
            os.close();
        } catch (IOException ignored) {}
    }
}
```

□□ □□□□ □□ □□ □□□□ □□□□□□

Java SE 7

```
try (OutputStream os = ...) {
    preferences.exportNode(os);
} catch (IOException ioe) {
```

```
    // Exception whilst writing data to the OutputStream
    ioe.printStackTrace();
} catch (BackingStoreException bse) {
    // Exception whilst reading from the backing preferences store
    bse.printStackTrace();
}
```

Java SE 7

```
OutputStream os = null;
try {
    os = ...;
    preferences.exportSubtree(os);
} catch (IOException ioe) {
    // Exception whilst writing data to the OutputStream
    ioe.printStackTrace();
} catch (BackingStoreException bse) {
    // Exception whilst reading from the backing preferences store
    bse.printStackTrace();
} finally {
    if (os != null) {
        try {
            os.close();
        } catch (IOException ignored) {}
    }
}
```

[illegible]

Preferences [illegible] XML [illegible] [illegible] [illegible]. [illegible] [illegible] [illegible] XML [illegible] [illegible] [illegible] Preferences [illegible] [illegible] [illegible] [illegible] [illegible] [illegible] [illegible].

XML [illegible] [illegible] [illegible] [illegible] Preferences [illegible] [illegible] [illegible] [illegible] [illegible]. [illegible] [illegible] [illegible] [illegible] [illegible] [illegible] [illegible] Preferences [illegible] [illegible] [illegible] [illegible] [illegible]. [illegible] [illegible] [illegible] [illegible] [illegible] Preferences [illegible] XML [illegible] [illegible] [illegible] [illegible] [illegible] [illegible] [illegible] [illegible] [illegible] [illegible] [illegible].

Java SE 7

```
try (InputStream is = ...) {
    // This is a static call on the Preferences class
    Preferences.importPreferences(is);
} catch (IOException ioe) {
    // Exception whilst reading data from the InputStream
    ioe.printStackTrace();
} catch (InvalidPreferencesFormatException ipfe) {
    // Exception whilst parsing the XML document tree
    ipfe.printStackTrace();
}
```

Java SE 7

```
InputStream is = null;
try {
    is = ...;
    // This is a static call on the Preferences class
    Preferences.importPreferences(is);
} catch (IOException ioe) {
    // Exception whilst reading data from the InputStream
    ioe.printStackTrace();
} catch (InvalidPreferencesFormatException ipfe) {
```

```
    // Exception whilst parsing the XML document tree
    ipfe.printStackTrace();
} finally {
    if (is != null) {
        try {
            is.close();
        } catch (IOException ignored) {}
    }
}
```

□□□ □□□ □□

□□□ □□□□ □□ Properties □□□□ □□ □□ □ □ □□□ □□□□ □□□□□□□ □□□□□□□.

Java SE 8

```
Preferences preferences = Preferences.userNodeForPackage(getClass());

PreferenceChangeListener listener = evt -> {
    System.out.println(evt.getKey() + " got new value " + evt.getNewValue());
};
preferences.addPreferenceChangeListener(listener);

//
// later...
//

preferences.removePreferenceChangeListener(listener);
```

Java SE 8

```
Preferences preferences = Preferences.userNodeForPackage(getClass());

PreferenceChangeListener listener = new PreferenceChangeListener() {
    @Override
    public void preferenceChange(PreferenceChangeEvent evt) {
        System.out.println(evt.getKey() + " got new value " + evt.getNewValue());
    }
};
preferences.addPreferenceChangeListener(listener);

//
// later...
//

preferences.removePreferenceChangeListener(listener);
```

NodeChangeListener □□□□□□□□.

□□ □□ □ □□ □□

Preferences □□□ □□ String , boolean , byte[] , double , float , int □□ long □□□ □ □ □□□□. □□□ □□ Preferences □□□ □□□□ □□ □□□ □□□□ □□□□□□□□.

```
Preferences preferences = Preferences.userNodeForPackage(getClass());

String someString = preferences.get("someKey", "this is the default value");
boolean someBoolean = preferences.getBoolean("someKey", true);
byte[] someByteArray = preferences.getByteArray("someKey", new byte[0]);
```

```
double someDouble = preferences.getDouble("someKey", 887284.4d);
float someFloat = preferences.getFloat("someKey", 38723.3f);
int someInt = preferences.getInt("someKey", 13232);
long someLong = preferences.getLong("someKey", 2827637868234L);
```

□□ □□ □ □□□□

Preferences □□□ □□ □□□□□□ putXXX() □□□ □ □□□ □□□□□□. Preferences □□□ □□ String , boolean , byte[] , double , float , int □□ long □□□ □ □ □□□□.

```
Preferences preferences = Preferences.userNodeForPackage(getClass());

preferences.put("someKey", "some String value");
preferences.putBoolean("someKey", false);
preferences.putByteArray("someKey", new byte[0]);
preferences.putDouble("someKey", 187398123.4454d);
preferences.putFloat("someKey", 298321.445f);
preferences.putInt("someKey", 77637);
preferences.putLong("someKey", 2873984729834L);
```

□□ □□ □□□□

Preferences □ □□□□□ □□ □□ □□□□□ □□□ □□□□□ □□□ □□ (□ : □□□ □□, □□ □□□□□□ □□ □□ □□□ □□□□□ □□, "□□ □□□□□ □□"□□□ □ □□ □□□ □□□□□□ □□)□ □□□ □ □□□ □ □□□□□ □□ □□□.

```
public class ExitConfirmer {
    private static boolean confirmExit() {
        Preferences preferences = Preferences.userNodeForPackage(ExitConfirmer.class);
        boolean doShowDialog = preferences.getBoolean("showExitConfirmation", true); // true
is default value

        if (!doShowDialog) {
            return true;
        }

        //
        // Show a dialog here...
        //
        boolean exitWasConfirmed = ...; // whether the user clicked OK or Cancel
        boolean doNotShowAgain = ...; // get value from "Do not show again" checkbox

        if (exitWasConfirmed && doNotShowAgain) {
            // Exit was confirmed and the user chose that the dialog should not be shown again
            // Save these settings to the Preferences object so the dialog will not show again
next time
            preferences.putBoolean("showExitConfirmation", false);
        }

        return exitWasConfirmed;
    }

    public static void exit() {
        if (confirmExit()) {
            System.exit(0);
        }
    }
}
```

□□□ □□ □□ □□ : https://riptutorial.com/ko/java/topic/582/□□-□□

□□□

S. No	□□	Contributors
1	□□ □□ □□□□	aa_oo, Aaqib Akhtar, abhinav, Abhishek Jain, Abob, acdcjunior, Adeel Ansari, adsalpha, AER, akhilsk, Akshit Soota, Alex A, alphaloop, altomnr, Amani Kilumanga, AndroidMechanic, Ani Menon, ankit dassor, Ankur Anand, antonio, Arkadiy, Ashish Ahuja, Ben Page, Blachshma, bpoiss, Burkhard, Carlton, Charlie H, Coffeehouse Coder, c□ LDS□ EED, Community, Confiqure, CraftedCart, dabansal, Daksh Gupta, Dan Hulme, Dan Morenus, DarkV1, David G., David Grinberg, David Newcomb, DeepCoder, Do Nhu Vy, Draken, Durgpal Singh, Dushko Jovanovski, E_net4, Edvin Tenovimas, Emil Sierżęga, Emre Bolat, enrico.bacis, Eran, explv, fgb, Francesco Menzani, Functino, garg10may, Gautam Jose, GingerHead, Grzegorz Górkiewicz, iliketocode, ıɯɐƃoʇ əızuəʞ, intboolstring, ipsi, J F , James Taylor, Jason, JavaHopper, Javant, javydreamercsw, Jean Vitor, Jean-François Savard, Jeffrey Brett Coleman, Jeffrey Lin, Jens Schauder, John Fergus, John Riddick, John Slegers, Jojodmo, JonasCz, Jonathan, Jonny Henly, Jorn Vernee, kaartic, Lambda Ninja, LostAvatar, madx, Magisch, Makoto, manetsus, Marc, Mark Adelsberger, Maroun Maroun, Matt, Matt, mayojava, Mitch Talmadge , mnoronha, Mrunal Pagnis, Mukund B, Mureinik, NageN, Nathan Arthur, nevster, Nithanim, Nuri Tasdemir, nyarasha, ochi, OldMcDonald, Onur, Ortomala Lokni, OverCoder, P.J.Meisch, Pavneet_Singh, Petter Friberg, philnate, Phrancis, Pops, ppeterka, Přemysl Šťastný, Pritam Banerjee, Radek Postołowicz, Radouane ROUFID, Rafael Mello, Rakitić, Ram, RamenChef, rekire, René Link, Reut Sharabani, Richard Hamilton, Ronnie Wang, ronnyfm, Ross Drew, RotemDev, Ryan Hilbert, SachinSarawgi, Sanandrea, Sandeep Chatterjee, Sayakiss, ShivBuyya, Shoe, Siguza , solidcell, stackptr, Stephen C, Stephen Leppik, sudo, Sumurai8 , Snađomfa□, tbodt, The Coder, ThePhantomGamer, Thisaru Guruge, Thomas Gerot, ThomasThiebaud, ThunderStruct, tonirush, Tushar Mudgal, Unihedron, user1133275, user124993, uzaif, Vaibhav Jain, Vakerrian, vasili111, Victor Stafusa, Vin, VinayVeluri, Vogel612 , vorburger, Wilson, worker_bee, Yash Jain, Yury Fedorov, Zachary David Saunders, Ze Rubeus
2	BigDecimal	alain.janinm, Christian, Dth, Enigo, ggolding, Harish Gyanani, John Nash, Loris Securo, Łukasz Piaszczyk, Manish Kothari, mszymborski, RamenChef, sudo, xwoker
3	BigInteger	Alek Mieczkowski, Alex Shesterov, Amani Kilumanga, Andrii Abramov, azurefrog, Byte1518, dimo414, dorukayhan, Emil Sierżęga , fabian, GPI, Ha., hd84335, janos, Kaushal28, Maarten Bodewes, Makoto, matt freake, Md. Nasir Uddin Bhuiyan, Nufail, Pritam Banerjee, Ruslan Bes, ShivBuyya, Stendika, Vogel612
4	BufferedWriter	Andrii Abramov, fabian, Jorn Vernee, Robin, VatsalSura
5	ByteBuffer	Community, Jon Ericson, Jorn Vernee, Tarık Yılmaz, Tomasz Bawor, victorantunes, Vogel612
6	C ++ □□	John DiFini
7	Dequeue □□□□□	Suketu Patel
8	EnumSet □□□	KIRAN KUMAR MATAM

9	Executor, ExecutorService □ Thread □	Andrii Abramov, Cache Staheli, Fildor, hd84335, Jens Schauder, JonasCz, noscreenname, Olivier Grégoire, philnate, Ravindra babu , Shettyh, Stephen C, Suminda Sirinath S. Dharmasena, sv3k, tones, user1121883, Vlad-HC, Vogel612
10	FileUpload to AWS	Amit Gujarathi
11	FTP (□□ □□ □□□□)	Kelvin Kellner
12	Hashtable	KIRAN KUMAR MATAM
13	HttpURLConnection	Community, Datagrammar, EJP, Inzimam Tariq IT, JonasCz, kiedysktos, Mureinik, NageN, Stephen C, still_learning
14	InputStream □ OutputStream	akgren_soar, EJP, Gubbel, J Atkin, Jens Schauder, John Nash, Kip , KIRAN KUMAR MATAM, Matt Clark, Michael, RamenChef, Stephen C, Vogel612
15	Iterator □ Iterable	Abubakkar, Comic Sans, Dariusz, Hulk, Lukas Knuth, RamenChef, Stephen C, user1121883, WillShackleford
16	Java (Standard Edition) □□	4444, Adeel Ansari, ajablonski, akhilsk, Alex A, altomnr, Ani Menon, Anthony Raymond, anuvab1911, Confiqure, CraftedCart, Emil Sierżęga, Gautam Jose, hd84335, ipsi, Jeffrey Brett Coleman, Lambda Ninja, Nithanim, Radouane ROUFID, Rakitić, ronnyfm, Sanandrea, Sandeep Chatterjee, sohnryang, Stephen C, Sнаđомfa□, tonirush, Walery Strauch, Ze Rubeus
17	Java 2D □□□	17slim, ABDUL KHALIQ
18	Java Editions, □□, □ □□ □ □□□	Gal Dreiman, screab, Stephen C
19	Java Pitfalls - Null □ NullPointerException	17slim, Andrii Abramov, Daniel Nugent, dorukayhan, fabian, François Cassin, Miles, Stephen C, Zircon
20	Java SE 7□ □□	compuhosny, RamenChef
21	Java SE 8□ □□	compuhosny, RamenChef, sun-solar-arrow
22	Java □□□□ □□□	Ezekiel Baniaga, Stephan, Stephen C
23	Java □□□ □□	Daniel M., engineercoding, fgb, John Nash, jwd630, mnoronha, OverCoder, padippist, RamenChef, Squidward, Stephen C
24	Java □□□ □□	Shree, Stephen C, Suminda Sirinath S. Dharmasena
25	Java □□ - 'java'□ 'javaw'	4444, Ben, mnoronha, Stephen C, Vogel612
26	Java □□ □□□ □□	Dariusz, hd84335, HTNW, Ilya, Mr. P, Petter Friberg, ravthiru, Stephen C, Stephen Leppik, Vogel612
27	Java □□ □□	Gene Marin, jatanp, Stephen C, Vogel612
28	Java □□	Nikhil R
29	Java □□□□	Display Name, mnoronha
30	Java □□ □□□	Danilo Guimaraes, Leonardo Pina

31	Java □□ □□□	Blubberguy22, Burkhard, Caleb Brinkman, Carter Brainerd, Community, Do Nhu Vy, Emil Sierżęga, George Bailey, Gerald Mücke, hd84335, ipsi, Kevin Thorne, Martijn Woudstra, Mitch Talmadge, Nagesh Lakinepally, PizzaFrog, Radouane ROUFID, RamenChef, sargue, Stephan, Stephen C, Trevor Sears, Universal Electricity
32	Java □□ □□	Tony
33	Java □□ - □□ □□	Dorian, GPI, John Starich, Jorn Vernee, Michał Rybak, mnoronha, ppeterka, Sharon Rozinsky, steffen, Stephen C, xTrollxDudex
34	Java □□ - □□ □□	Alex T., Cody Gray, Enwired, Friederike, Gal Dreiman, hd84335, Hiren, Peter Rader, piyush_baderia, RamenChef, Ravindra HV, RudolphEst, Stephen C, Todd Sewell, user3105453
35	Java □□ - □□ □□	Bhoomika, bruno, dimo414, Gal Dreiman, hd84335, SachinSarawgi, scorpp, Stephen C, Stephen Leppik, user3105453
36	java.util.Objects □□ □	mnoronha, RamenChef, Stephen C
37	Java□□ □□ □□□□ □□ □□	Nikhil R
38	Java□ JSON	Asaph, Bogdan Korinnyi, Burkhard, Cache Staheli, hd84335, ipsi, Jared Hooper, Kurzalead, MikaelF, Mrunal Pagnis, Nicholas J Panella, Nikita Kurtin, ppeterka, Prem Singh Bist, RamenChef, Ray Kiddy, SirKometa, still_learning, Stoyan Dekov, systemfreund, Tim, Vikas Gupta, vsminkov, Yury Fedorov
39	JAXB	Dariusz, Drunix, fabian, hd84335, Jabir, ppeterka, Ram, Stephan, Thomas Fritsch, vallismortis, Walery Strauch
40	JAXP API□ □□□ XML □□ □□	GPI
41	JAX-WS	ext1812, Jonathan Barbero, Stephen Leppik
42	JIT (Just in Time) □ □□□	Liju Thomas, Stephen C
43	JMX	esin88
44	JNDI	EJP, neohope, RamenChef
45	JShell	ostrichofevil, Sudip Bhandari
46	JVM □□ □□□□□	desilijic
47	JVM □□□	Confiqure, RamenChef
48	log4j / log4j2	Daniel Wild, Fildor, HCarrasko, hd84335, Mrunal Pagnis, Rens van der Heijden
49	MultiThreaded □□ □□□ □□□ ThreadPoolExecutor □ □.	Brendon Dugan
50	Nashorn □□ □□□□ □□	ben75, ekaerovets, Francesco Menzani, hd84335, Ilya, InitializeSahib, kasperjj, VatsalSura

51	NIO - □□□□	Matthieu, mnoronha
52	NumberFormat	arpit pandey, John Nash, RamenChef, ΦXocę □ Пepeúpa ヅ
53	RSA □□□	Dennis Kriechel, Drunix, iqbal_cs, Maarten Bodewes, Nicktar, Shog9
54	ServiceLoader	fabian, Florian Genser, Gerald Mücke
55	SortedMap	Amit Gujarathi
56	String Tokenizer	M M
57	StringBuffer	Amit Gujarathi
58	StringBuilder	Andrii Abramov, Cache Staheli, David Soroko, Enigo, fabian, fgb, JudgingNotJudging, KIRAN KUMAR MATAM, Nicktar, P.J.Meisch, Stephen C
59	sun.misc.Unsafe	4444, Daniel Nugent, Grexis, Stephen C, Suminda Sirinath S. Dharmasena
60	ThreadLocal	Dariusz, Liju Thomas, Manish Kothari, Nithanim, taer
61	TreeMap □ TreeSet	Malt, Stephen C
62	XJC	Danilo Guimaraes, fabian
63	XML XPath □□	17slim, manouti
64	XOM - XML □□ □□	Arthur, Makoto
65	□□□ (□□□ □□□ □□ □□□ □□)	Aasmund Eldhuset, Abhishek Balaji R, Catalina Island, Daniel M., intboolstring, Jonathan, Mark Yisri, Mureinik, NageN, ParkerHalo , Stephen C, Vogel612
66	□□ □□	Ayush Bansal, Christophe Weis, Jonathan
67	□□ □□	Andrii Abramov, arcy, Vasiliy Vlasov
68	□□ □□□ □□□□ □□□	A Boschman, Ad Infinitum, Andrii Abramov, Ani Menon, anuvab1911, Arthur Noseda, augray, Brett Kail, Burkhard, CaffeineToCode, Chris Midgley, cricket_007, Dariusz, Elazar, Emil Sierżęga, Enigo, fabian, fgb, Floern, fzzfzzfzz, hd84335, intboolstring, james large, JamesENL, Jens Schauder, John Slegers, Jorn Vernee, kstandell, Lahiru Ashan, Laurel, Miljen Mikic, mnoronha, mykey, NageN, Nayuki, Nicktar, Pace, Petter Friberg, Radouane ROUFID, Ram, Robert Columbia, Ronnie Wang, shmosel, Stephen C, TNT
69	□□□ □□	Fildor, Ironcache, Krös, martin, Petter Friberg, Stephen C, Sujith Niraikulathan, Thisaru Guruge, uzaif
70	□□	Ad Infinitum, Adam, Adrian Krebs, agoeb, Ali Dehghani, Andrii Abramov, ar4ers, Arkadiy, Blubberguy22, Bohemian, Brad Larson, Burkhard, CodeCore, coder-croc, Dariusz, David Grinberg, devnull69, DonyorM, DVarga, Emre Bolat, explv, fabian, gattsbr, geniushkg, GhostCat, Gubbel, hirosht, HON95, J Atkin, Jason V, JavaHopper, Jeffrey Bosboom, Jens Schauder, Jonathan, Jorn Vernee, Kai, Kevin DiTraglia, kiuby_88, Lahiru Ashan, Luan Nico, maheshkumar, Mshnik, Muhammed Refaat, OldMcDonald, Oleg Sklyar, Ortomala Lokni, PM 77-1, Prateek Agarwal, QoP, Radouane ROUFID,

		RamenChef, Ravindra babu, Shog9, Simulant, SjB, Slava Babin, Stephen C, Stephen Leppik, still_learning, Sudhir Singh, Theo, ToTheMaximum, uhrm, Unihedron, Vasiliy Vlasov, Vucko
71	□□ □□□□□	Andreas
72	□□ □□□	ar4ers, hd84335, intboolstring, javac, Jeffrey Bosboom, Jens Schauder, Kai, matt freake, o_nix, philnate, Ravindra HV, richersoon, Ruslan Bes, Stephen C, Stephen Leppik, Vasiliy Vlasov
73	□□ □□ □□	Adrian Krebs, AJNeufeld, Andrew Brooke, AshanPerera, Buddy, Caleb Brinkman, Cas Eliëns, Coffeehouse Coder, CraftedCart, dedmass, ebo, fabian, intboolstring, Inzimam Tariq IT, Jens Schauder, JonasCz, Jorn Vernee, juergen d, Makoto, Matt Champion , philnate, Ram, Santhosh Ramanan, sevenforce, Stephen C, teek, Unihedron, Uri Agassi, xwoker
74	□□□	17slim, A Boschman, Arthur, Avinash Kumar Yadav, Blubberguy22, ced-b, Daniel Nugent, granmirupa, Ilya, Jan Vladimir Mostert, janos, JD9999, jopasserat, Karthikeyan Vaithilingam, Kenster, Krzysztof Krasoń, Oleg Sklyar, RamenChef, Sheshnath, Stephen C, sudo, Thisaru Guruge, Vasilis Vasilatos, ΦXocę □ Пepeúpa ツ
75	□□ □□	Arthur, David Grant, David Soroko, dorukayhan, F. Stephen Q, Kichiin, MasterBlaster, michaelbahr, rokonoid, Stephen C, Thodgnir
76	□□ □□□	A_Arnold, alain.janinm, arcy, Bob Rivers, Christian Wilkie, explv, Jabir, Jean-Baptiste Yunès, John Smith, Matt Clark, Miles , NamshubWriter, Nicktar, Nishant123, Ph0bi4, ppeterka, Ralf Kleberhoff, Ram, skia.heliou, Squidward, Stephen C, Vinod Kumar Kashyap
77	□□□ □□ (java.time.*)	Bilbo Baggins, bowmore, Michael Piefel, Miles, mnoronha, Simon, Squidward, Tarun Maganti, Vogel612, ΦXocę □ Пepeúpa ツ
78	□□□□	Arthur, Burkhard, devnull69, DonyorM, glee8e, Grayson Croom, Ilya, Malt, Matej Kormuth, Matthieu, Mine_Stone, ppeterka, RamenChef, Stephen C, Tot Zam, vsav
79	□□□□ □□□ □□□□	Jeet
80	□□ □□□ JAR □□	manouti
81	□□□	Adrian Krebs, Amani Kilumanga, Daniel LIn, Dushman, Kakarot, Lernkurve, Markus L, NageN, Pawan, Ravindra babu, Saiful Azad, Stephen C
82	□□ □□□	Ironcache
83	□□ □ □ □□ □□□	Bob Rivers, cdm, kann, Makoto, mnoronha, ppeterka, Ram, VGR
84	□□□ □ □□□	Ad Infinitum, Alek Mieczkowski, Androbin, DimaSan, engineercoding, ppeterka, RamenChef, rd22, Samk, Stephen C
85	□□ □□□	mnoronha, ppeterka, Viacheslav Vedenin
86	□□□ □□	JD9999, KIRAN KUMAR MATAM, Mureinik, Stephen C, VatsalSura
87	□□ □□	GPI, Kenster, Powerlord, user2296600

88	□□ □□□□□□ (□□□)	adino, Alex, assylias, bfd, Bhagyashree Jog, bowmore, Burkhard, Chetya, corsiKa, Dariusz, Diane Chastain, DimaSan, dimo414, Fildor, Freddie Coleman, GPI, Grzegorz Górkiewicz, hd84335, hellrocker, hexafraction, Ilya, james large, Jens Schauder, Johannes, Jorn Vernee, Kakarot, Lance Clark, Malt, Matěj Kripner, Md. Nasir Uddin Bhuiyan, Michael Piefel, michaelbahr, Mitchell Tracy, MSB, Murat K., Mureinik, mvd, NatNgs, nickguletskii, Olivier Durin, OlivierTheOlive, Panda, parakmiakos, Paweł Albecki, ppeterka, RamenChef, Ravindra babu, rd22, RudolphEst, snowe2010, Squidward, Stephen C, Sudhir Singh, Tobias Friedinger, Unihedron, Vasiliy Vlasov, Vlad-HC, Vogel612, wolfcastle, xTrollxDudex, YCF_L, Yury Fedorov, ZX9
89	□□ □	Abhishek Jain, Ad Infinitum, Adam, aioobe, Amit Gupta, Andrei Maieras, Andrew Tobilko, Andrii Abramov, Ankit Katiyar, Anony-Mousse, assylias, Brian Goetz, Burkhard, Conrad.Dean, cringe, Daniel M., David Soroko, dimitrisli, Draken, DVarga, Emre Bolat, enrico.bacis, fabian, fgb, Gal Dreiman, gar, GPI, Hank D, hexafraction, Ivan Vergiliev, J Atkin, Jean-François Savard, Jeroen Vandevelde, John Slegers, JonasCz, Jorn Vernee, Jude Niroshan, JudgingNotJudging, Kevin Raoofi, Malt, Mark Green, Matt, Matthew Trout, Matthias Braun, ncmathsadist, nobeh, Ortomala Lokni, Paŭlo Ebermann, Paweł Albecki, Petter Friberg, philnate, Pujan Srivastava, Radouane ROUFID, RamenChef, rolve, Saclyr Barlonium, Sergii Bishyr, Skylar Sutton, solomonope, Stephen C, Stephen Leppik, timbooo, Tunaki, Unihedron, vincentvanjoe, Vlasec, Vogel612, webo80, William Ritson, Wolfgang, Xaerxess, xploreraj, Yogi, Ze Rubeus
90	□□□ □□	RamenChef
91	□□ (java.util.logging)	bn., Christophe Weis, Emil Sierżęga, P.J.Meisch, vallismortis
92	□□ □□ □□□	KIRAN KUMAR MATAM
93	□□□ (□□□ □□□ □□)	Androbin, Christian, Emily Mabrey, Enwired, fabian, Gerald Mücke, Jesse van Bekkum, Kenster, Stephen C, timbooo, VGR, vorburger
94	□□□	1d0m3n30, EJP, ParkerHalo, Stephen C, ThePhantomGamer
95	□□□□ API	Ali786, ArcticLord, Aurasphere, Blubberguy22, Bohemian, Christophe Weis, Drizzt321, fabian, hd84335, Joeri Hendrickx, Luan Nico, madx, Michael Myers, Onur, Petter Friberg, RamenChef, Ravindra babu, Squidward, Stephen C, Tony BenBrahim, Universal Electricity, ΦXocę Пepeúpa ツ
96	□□ □ □□ □□	Burkhard, Michael von Wenckstern, Stephen C
97	□□	Jonathan, user140547
98	□□ □ SET	KIRAN KUMAR MATAM
99	□□ □□□	Ilya
100	□□□	17slim, A.J. Brown, A_Arnold, Abhishek Jain, Abubakkar, Adam Ratzman, Adrian Krebs, agilob, Aiden Deom, Alex Meiburg, Alex Shesterov, altomnr, Amani Kilumanga, Andrew Tobilko, Andrii Abramov, Andy Thomas, Anony-Mousse, Asaph, Ataeraxia, Austin, Austin Day, ben75, bfd, Bob Brinks, bpoiss, Burkhard, Cache Staheli, Caner Balım, Chris Midgley, Christian, Christophe Weis, coder-croc, Community, cyberscientist, Daniel Käfer, Daniel

		Stradowski, DarkV1, dedmass, DeepCoder, dnup1092, dorukayhan, drov, DVarga, ekeith, Emil Sierżęga, emotionlessbananas, enrico.bacis, Enwired, fabian, FlyingPiMonster, Gabriele Mariotti, Gal Dreiman, Gergely Toth, Gihan Chathuranga, GingerHead, giucal, Gray, GreenGiant, hamena314, Harish Gyanani, HON95, iliketocode, Ilya, Infuzed guy, intboolstring, J Atkin, Jabir, javac, JavaHopper, Jeffrey Lin, Jens Schauder, Jérémie Bolduc, John Slegers, Jojodmo, Jon Ericson, JonasCz, Jordi Castilla, Jorn Vernee, JSON C11, Jude Niroshan, Kamil Akhuseyinoglu, Kapep, Kaushal28, Kaushik NP, Kehinde Adedamola Shittu, Kenster, kstandell, Lachlan Dowding, Lahiru Ashan, Laurel, Leo Aso, Liju Thomas, LisaMM, M.Sianaki, Maarten Bodewes , Makoto, Malav, Malt, Manoj, Manuel Spigolon, Mark Stewart, Marvin, Matej Kormuth, Matt Clark, Matthias Braun, maxdev, Maxim Plevako, mayha, Michael, MikeW, Miles, Miljen Mikic, Misa Lazovic, mr5, Myridium, NikolaB, Nufail, Nuri Tasdemir, OldMcDonald, OliPro007, Onur, Optimiser, ozOli, P.J.Meisch, Paolo Forgia, Paweł Albecki, Petter Friberg, phant0m, piyush_baderia, ppeterka, Přemysl Šťastný, PSo, QoP, Radouane ROUFID, Raj, RamenChef, RAnders00, Rocherlee, Ronnie Wang, Ryan Hilbert, ryanyuyu, Sayakiss, SeeuD1, sevenforce, Shaan, ShivBuyya, Shoe, Sky, SmS, solidcell, Squidward, Stefan Isele - prefabware.com, stefanobaghino, Stephen C, Stephen Leppik, Steven Benitez, still_learning, Sudhir Singh, Swanand Pathak, S нађомƒа□, TDG, TheLostMind, ThePhantomGamer, Tony BenBrahim, Unihedron, VGR, Vishal Biyani, Vogel612, vsminkov, vvtx, Wilson, winseybash, xwoker, yuku, Yury Fedorov, Zachary David Saunders, Zack Teater, Ze Rubeus, ΦΧοcę □ Перeúра ツ
101	□□□□ □□ □ □□	Chirag Parmar, DarkV1, Gihan Chathuranga, Jabir, JonasCz, Kaushal28, Lachlan Dowding, Laurel, Maarten Bodewes, Matt Clark, PSo, RamenChef, Shaan, Stephen C, still_learning
102	□□□□ □□ □□ □□□ □□	Bohemian
103	□□□ □□ □□	bloo, Display Name, rakwaht, Squidward
104	□□	Andy Thomas, Bob Rivers, ppeterka, vorburger, yitzih
105	□□	3442, 416E64726577, A Boschman, A.M.K, A_Arnold, Abhishek Jain, Abubakkar, acdcjunior, Ad Infinitum, Addis, Adrian Krebs, AER, afzalex, agilob, Alan, Alex Shesterov, Alexandru, altomnr, Amani Kilumanga, Andrew Tobilko, Andrii Abramov, AndroidMechanic, Anil , ankidaemon, ankit dassor, anotherGatsby, antonio, Ares, Arthur , Ashish Ahuja, assylias, AstroCB, baao, Beggs, Berzerk, Big Fan , BitNinja, bjb568, Blubberguy22, Bob Rivers, bpoiss, Bryan, BudsNanKis, Burkhard, bwegs, c1phr, Cache Staheli, Cerbrus, Charitha, Charlie H, Chris Midgley, Christophe Weis, Christopher Schneider, Codebender, coder-croc, Cold Fire, Colin Pickard, Community, Confiqure, CptEric, Daniel Käfer, Daniel Stradowski, Dariusz, DarkV1, David G., DeepCoder, Devid Farinelli, Dhrubajyoti Gogoi, Dmitry Ginzburg, dorukayhan, Duh-Wayne-101, Durgpal Singh, DVarga, Ed Cottrell, Edvin Tenovimas, Eilit, eisbehr, Elad, Emil Sierżęga, Emre Bolat, Eng.Fouad, enrico.bacis, Eran, Erik Minarini, Etki, explv, fabian, fedorqui , Filip Haglund, Forest White, fracz, Franck Dernoncourt, Functino, futureelite7, Gal Dreiman, gar, Gene Marin, GingerHead , granmirupa, Grexis, Grzegorz Sancewicz, Gubbel, Guilherme Torres Castro, Gustavo Coelho, hhj8i, Hiren, Idos, ihatecsv, iliketocode, Ilya, Ilyas Mimouni, intboolstring, Irfan, J Atkin, jabbathehutt1234, JakeD, James Taylor, Jamie, Jamie Rees, Janez Kuhar, Jared Rummler, Jargonius, Jason Sturges, JavaHopper,

		Javant, Jeeter, Jeffrey Bosboom, Jens Schauder, Jérémie Bolduc, Jeutnarg, jhnance, Jim Garrison, jitendra varshney, jmattheis, Joffrey, Johannes, johannes_preiser, John Slegers, JohnB, Jojodmo, Jonathan, Jordi Castilla, Jorn, Jorn Vernee, Josh, JStef, JudgingNotJudging, Justin, Kapep, KartikKannapur, Kayathiri, Kaz Wolfe, Kenster, Kevin Thorne, Lambda Ninja, Liju Thomas, llamositopia, Loris Securo, Luan Nico, Lucas Paolillo, maciek, Magisch, Makoto, Makyen, Malt, Marc, Markus, Marvin, MasterBlaster, Matas Vaitkevicius, matsve, Matt, Matt, Matthias Braun, Maxim Kreschishin, Maxim Plevako, Maximillian Laumeister, MC Emperor, Menasheh, Michael Piefel, michaelbahr, Miljen Mikic, Minhas Kamal, Mitch Talmadge, Mohamed Fadhl, Muhammed Refaat, Muntasir, Mureinik, Mzzzzzz, NageN, Nathaniel Ford, Nayuki, nicael, Nigel Nop, niyasc, noʊʇʎdʎzɐɹƆ, Nuri Tasdemir, Ocracoke, OldMcDonald, Onur, orccrusher99, Ortomala Lokni, Panda, Paolo Forgia, Paul Bellora, Paweł Albecki, PeerNet, Peter Gordon, phatfingers, Pimgd, Piyush, ppeterka, Přemysl Šťastný, PSN, Pujan Srivastava, QoP, Radiodef, Radouane ROUFID, Raidri, Rajesh , Rakitić, Ram, RamenChef, Ravi Chandra, René Link, Reut Sharabani, Richard Hamilton, Robert Columbia, rolfedh, rolve, Roman Cherepanov, roottraveller, Ross, Ryan Hilbert, Sam Hazleton, sandbo00, Saurabh, Sayakiss, sebkur, Sergii Bishyr, sevenforce, shmosel, Shoe, Siguza, Simulant, Slayther, Smi, solidcell, Spencer Wieczorek, Squidward, stackptr, stark, Stephen C, Stephen Leppik, Sualeh Fatehi, sudo, Sumurai8, Sunnyok, syb0rg, tbodt, tdelev, tharkay, Thomas, ThunderStruct, Tol182, ʇolɐəz əɥʇ qoq, tpunt, Travis J, Tunaki, Un3qual, Unihedron, user6653173, uzaif, vasili111, VedX, Ven, Victor G., Vikas Gupta, vincentvanjoe, Vogel612, Wilson, Winter, X.lophix, YCF_L, Yohanes Khosiawan □□□, yuku, Yury Fedorov, zamonier, Φ Xocę □ Перейра ツ
106	□□ □□	esin88
107	□□ □□□□ □□	1d0m3n30, Bohemian, Holger, Idcmp, Jon Ericson, kristyna, Michael Piefel, Stephen C, Vogel612
108	□□ □□□□ □□□	Mykola Yashchenko
109	□□ (□□)	Daniel Nugent, Dushman, Omar Ayala, Rafael Pacheco, RamenChef, VGR, xsami
110	□□ □□	Ankit Katiyar
111	□□ □□□	alphaloop, hexafraction, Uux
112	□□ □ □□□	John Nash, shibli049
113	□□ □ □ □□□	ipsi, mnoronha
114	□ □□ □□ □	Ankit Katiyar, Arash, fabian, Florian Weimer, FlyingPiMonster, Grzegorz Górkiewicz, J-Alex, JavaHopper, Ken Y-N, KIRAN KUMAR MATAM, Miljen Mikic, NageN, Nuri Tasdemir, Onur, ppeterka, Prateek Agarwal
115	□□ □□ □ □□□	Andrii Abramov, Conrad.Dean, Daniel Nugent, fabian, GPI, Hazem Farahat, JAVAC, Mshnik, Nolequen, Petter Friberg, Prateek Agarwal, sebkur, Stephen C
116	□□ □□	Aimee Borda, Blubberguy22, dosdebug, esin88, Gerald Mücke, Jorn Vernee, Kineolyan, mnoronha, Nayuki, Rednivrug, Ryan Hilbert, Stephen C, thatguy

117	□ □□ I / O	dorukayhan, niheno, TuringTux
118	□□□	Andrii Abramov, Asiat, BrunoDM, ced-b, Codebender, Dylan, George Bailey, Jeremy, Ralf Kleberhoff, RamenChef, Thomas Gerot, tynn, Vogel612
119	□□ □□	A Boschman, Abubakkar, Andrey Rubtsov, Andrii Abramov, assylias, bowmore, Charlie H, Chris H., Christophe Weis, compuhosny, Dair, Emil Sierżęga, enrico.bacis, fikovnik, Grzegorz Górkiewicz, gwintrob, Hadson, hd84335, hzpz, J Atkin, Jean-François Savard, John Slegers, Jude Niroshan, Maroun Maroun, Michael Wiles, OldMcDonald, shmosel, Squidward, Stefan Dollase, Stephen C, ultimate_guy, Unihedron, user140547, Vince, vsminkov, xwoker
120	□□	A_Arnold, atom, ced-b, Chirag Parmar, Daniel Stradowski, demongolem, DimaSan, fabian, Kaushal28, Kenster
121	□□	Ordiel
122	□□ □□□	17slim, Arthur, J Atkin, Jabir, KIRAN KUMAR MATAM, Marvin, peterh, Stephen C, VGR, vorburger
123	□□□ □□□□□ AppDynamics □ TIBCO BusinessWorks □□	Alexandre Grimaud
124	□□□ □□□□ □□ □	Sugan
125	□□ □□□	Abhijeet
126	□□□	Alek Mieczkowski, Chirag Parmar, Community, Jon Ericson, JonasCz, Ram, RamenChef, Redterd, Stephen C, sun-solar-arrow, ΦXocę □ Π epeúpa ツ
127	□□ □□ API	manouti
128	□□□	4castle, Abubakkar, acdcjunior, Aimee Borda, Akshit Soota, Amitay Stern, Andrew Tobilko, Andrii Abramov, ArsenArsen, Bart Kummel, berko, Blubberguy22, bpoiss, Brendan B, Burkhard, Cerbrus, Charlie H, Claudio, Community, Conrad.Dean, Constantine, Daniel Käfer, Daniel M., Daniel Stradowski, Dariusz, David G., DonyorM, Dth, Durgpal Singh, Dushko Jovanovski, DVarga, dwursteisen, Eirik Lygre, enrico.bacis, Eran, explv, Fildor, Gal Dreiman, gontard, GreenGiant, Grzegorz Oledzki, Hank D, Hulk, iliketocode, ItachiUchiha, izikovic, J Atkin, Jamie Rees, JavaHopper, Jean-François Savard, John Slegers, Jon Erickson, Jonathan, Jorn Vernee, Jude Niroshan, JudgingNotJudging, Justin, Kapep, Kip, LisaMM, Makoto, Malt, malteo, Marc, MasterBlaster, Matt, Matt, Matt S., Matthieu, Michael Piefel, MikeW, Mitch Talmadge, Mureinik, Muto, Naresh Kumar, Nathaniel Ford, Nuri Tasdemir, OldMcDonald, Oleg L., omiel, Ortomala Lokni, Pawan, Paweł Albecki, Petter Friberg, Philipp Wendler, philnate, Pirate_Jack, ppeterka, Radnyx, Radouane ROUFID, Rajesh Kumar, Rakitić, RamenChef, Ranadip Dutta, ravthiru, reto, Reut Sharabani, RobAu, Robin, Roland Illig, Ronnie Wang, rrampage, RudolphEst, sargue, Sergii Bishyr, sevenforce, Shailesh Kumar Dayananda, shmosel, Shoe, solidcell, Spina, Squidward, SRJ, stackptr, stark, Stefan Dollase, Stephen C, Stephen Leppik, Steve K, Sugan, suj1th, thiagogcm, tpunt, Tunaki, Unihedron, user1133275, user1803551, Valentino, vincentvanjoe, vsnyc, Wilson, Ze Rubeus, zwl

129	□□ □	aasu, Andrew Antipov, Daniel Käfer, Dave Ranjan, David Soroko, Emil Sierżęga, Enigo, fabian, Filip Smola, GreenGiant, Gubbel, Hulk, Jabir, Jens Schauder, JonasCz, Jonathan, JonK, Malt, Matsemann, Michael Lloyd Lee mlk, Mifeet, Miroslav Bradic, NamshubWriter, Pablo, Peter Rader, RamenChef, riyaz-ali, sanastasiadis, shmosel, Stefan Dollase, stefanobaghino, Stephen C, Stephen Leppik, still_learning, Uri Agassi, user3105453, Vasiliy Vlasov, Vlad-HC, Vogel612, xploreraj
130	□□□ □□□ □	Jonathan Barbero
131	□□□	ArcticLord, Enigo, MadProgrammer, ppeterka
132	□□□□ □	Amit Gujarathi, KIRAN KUMAR MATAM
133	□□□ □□ □	Amit Gujarathi, KIRAN KUMAR MATAM
134	□□□	17slim, 1d0m3n30, A Boschman, acdcjunior, afuc func, AJ Jwair, Amani Kilumanga, Andreas, Andrew, Andrii Abramov, Blake Yarbrough, Blubberguy22, Bobas_Pett, c.uent, Cache Staheli, Chris Midgley, Claudia, clinomaniac, Dariusz, Darth Shadow, Davis, EJP, Emil Sierżęga, Eran, fabian, FedeWar, FlyingPiMonster, futureelite7, Harsh Vakharia, hd84335, J Atkin, JavaHopper, Jérémie Bolduc, jimrm, Jojodmo, Jorn Vernee, kanhaiya agarwal, Kevin Thorne, Li357, Loris Securo, Lynx Brutal , Maarten Bodewes, Mac70, Makoto, Marvin, Michael Anderson, Mshnik, NageN, Nuri Tasdemir, Ortomala Lokni, OverCoder, ParkerHalo, Peter Gordon, ppeterka, qxz, rahul tyagi, RamenChef, Ravan, Reut Sharabani, Rubén, sargue, Sean Owen, ShivBuyya, shmosel, SnoringFrog, Stephen C, tonirush, user3105453, Vogel612 , Winter
135	□□ □	KIRAN KUMAR MATAM
136	□□ □	1d0m3n30, A Boschman, aioobe, Amani Kilumanga, Andreas Fester, Andrew Sklyarevsky, Andrew Tobilko, Andrii Abramov, Anony-Mousse , bcosynot, Bob Rivers, coder-croc, Community, Constantine, Daniel Käfer, Daniel M., Danilo Guimaraes, DVarga, Emil Sierżęga , enrico.bacis, f_puras, fabian, Gal Dreiman, Gene Marin, Grexis , Grzegorz Oledzki, ipsi, J Atkin, Jared Hooper, javac, Jérémie Bolduc, Johannes, Jon Ericson, k3b, Kenster, Lahiru Ashan, Maarten Bodewes, madx, Mark, Michael Myers, Mick Mnemonic, NageN , Nef10, Nolequen, OldCurmudgeon, OliPro007, OverCoder, P.J.Meisch, Panther, Paweł Albecki, Petter Friberg, Punika, Radouane ROUFID, RamenChef, rd22, Ronon Dex, Ryan Hilbert, S.K. Venkat, Samk, shmosel, Spina, Stephen Leppik, Tarun Maganti, Tim , Torsten, VGR, Victor G., Vinay , Wolf, Yury Fedorov, Zefick, Φ Xocę □ Пepeúpa ツ
137	□□ □ □□ □□	Adrian Krebs, agilob, akhilsk, Andrii Abramov, Bhavik Patel, Burkhard, Cache Staheli, Codebender, Dariusz, DarkV1, dimo414, Draken, EAX, Emil Sierżęga, enrico.bacis, fabian, FMC, Gal Dreiman, GreenGiant, Hernanibus, hexafraction, Ilya, intboolstring, Jabir, James Jensen, JavaHopper, Jens Schauder, John Nash, John Slegers, JonasCz, Kai, Kevin Thorne, Malt, Manish Kothari, Md. Nasir Uddin Bhuiyan, michaelbahr, Miljen Mikic, Mitch Talmadge, Mrunal Pagnis, Myridium, mzc, Nikita Kurtin, Oleg Sklyar, P.J.Meisch, Paweł Albecki, Peter Gordon, Petter Friberg, ppeterka, Radek Postołowicz, Radouane ROUFID, Raj, RamenChef, rdonuk, Renukaradhya, RobAu, sandbo00, Saša Šijak, sharif.io, Stephen C, Stephen Leppik, still_learning, Sudhir Singh, sv3k, tatoalo, Thomas Fritsch, Tripta Kiroula,

		vic-3, Vogel612, Wilson, yiwei
138	□□□	Dac Saunders, Petter Friberg, RamenChef, TNT, tonirush, Tot Zam, Vogel612
139	□□□ □□ □□ □□	Ahmed Ashour, aioobe, akhilsk, alex s, Andrii Abramov, Cassio Mazzochi Molin, Dan Whitehouse, Enigo, erickson, f_puras, fabian, giucal, hd84335, J.D. Sandifer, Lahiru Ashan, Mac70, NamshubWriter, Nicktar, Petter Friberg, Pradatta, Pritam Banerjee, RamenChef, sanjaykumar81, Santa Claus, Santhosh Ramanan, VGR
140	□□ □□	17slim, Anony-Mousse, Bob Rivers, Chuck Daniels, cshubhamrao, fabian, hd84335, J Atkin, janos, kaartic, Kirill Sokolov, Luan Nico, Nayuki, piyush_baderia, Ram, RamenChef, Saagar Jha, Stephen C, Unihedron, Vladimir Vagaytsev
141	□□ □ □□□ □□	Adowrath, Kishore Tulsiani, WillShackleford
142	□□ □□□ □□ (RMI)	RamenChef, smichel, Stephen C, user1803551, Vasiliy Vlasov
143	□□ □□□ □□	17slim, 1d0m3n30, Amani Kilumanga, Ani Menon, Anony-Mousse, Bilesh Ganguly, Bob Rivers, Burkhard, Conrad.Dean, Daniel, Dariusz, DimaSan, dnup1092, Do Nhu Vy, enrico.bacis, fabian, Francesco Menzani, Francisco Guimaraes, gar, Ilya, IncrediApp, ipsi, J Atkin, JakeD, javac, Jean-François Savard, Jojodmo, Kapep, KdgDev, Lahiru Ashan, Master Azazel, Matt, mayojava, MBorsch, nimrod, Pang, Panther, ParkerHalo, Petter Friberg, Radek Postołowicz, Radouane ROUFID, RAnders00, RobAu, Robert Columbia, Simulant, Squidward, Stephen C, Stephen Leppik, Sundeep, SuperStormer, ThePhantomGamer, TMN, user1803551, user2314737, Veedrac, Vogel612
144	□□ □□	Daniel Nugent, Stephen C, Suminda Sirinath S. Dharmasena, xTrollxDudex
145	□□□ □□□□□	bn., noscreenname, P.J.Meisch, RamenChef, TuringTux
146	□□ □□	4castle, Filip Smola, Joshua Carmody, Nick Donnelly, RamenChef, Squidward
147	□□□□□	100rabh, A Boschman, Abhishek Jain, Adowrath, Alex Shesterov, Andrew Tobilko, Andrii Abramov, Cà phê đen, Chirag Parmar, Conrad.Dean, Daniel Käfer, devguy, DVarga, Hilikus, inovaovao, intboolstring, James Oswald, Jan Vladimir Mostert, JavaHopper, Johannes, Jojodmo, Jonathan, Jorn Vernee, Kai, kstandell, Laurel, Marvin, MikeW, Paul Nelson Baker, Peter Rader, ppovoski, Prateek Agarwal, Radouane ROUFID, RamenChef, Robin, Simulant, someoneigna, Stephen C, Stephen Leppik, Sujith Niraikulathan, Thomas Gerot, user187470, Vasiliy Vlasov, Vince Emigh, xwoker, Zircon
148	□□□□ □□ □□	akvyalkov, Anand Vaidya, Andy Thomas, Anton Hlinisty, anuvab1911, Conrad.Dean, Daniel Nugent, Dushko Jovanovski, Enwired, Gal Dreiman, Gerald Mücke, HTNW, james large, Jenny T-Type, John Starich, Lahiru Ashan, Makoto, Morgan Zhang, NamshubWriter, P.J.Meisch, Pirate_Jack, ppeterka, RamenChef, screab, Siva Sankar Rajendran, Squidward, Stephen C, Stephen Leppik, Steve Harris, tonirush, TuringTux, user3105453
149	□□ □□ □□ (JVM)	Dushman, RamenChef, Rory McCrossan, Stephen Leppik

150	□□ □□□□□ □□□□□□	Coffee Ninja, Fjoni Yzeiri, Jorn Vernee, RamenChef, Stephen C, user1803551
151	□□ □□	garg10may, nishizawa23, Pseudonym Patel, RamenChef, Smit, Stephen C
152	□□ □□	foxt7ot, J. Pichardo, James Fry, SaWo, Stephen C
153	□□ □□□□□ - 'javac'	CraftedCart, Jatin Balodhi, Mark Stewart, nishizawa23, Stephen C, Sнађошfa□, Tom Gijselinck
154	□□ □□□□□ □□□□ □□	Alexiy
155	□□ □□ - □□□□□ □□□□	dorukayhan, james large, Stephen C
156	□□	Andy Thomas, atom, Bobas_Pett, Ce7, charlesreid1, Confiqure, David Soroko, fabian, hamena314, hd84335, JavaHopper, Javant, Matej Kormuth, mayojava, Nicktar, Peter Gordon, RamenChef, Raviteja, Ruslan Bes, Stephen C, sumit
157	□□ □□□□	Amani Kilumanga, Andy Thomas, Asaph, ced-b, Daniel M., fabian, hd84335, intboolstring, kaotikmynd, Laurel, Makoto, nhahtdh, ppeterka, Ram, RamenChef, Saif, Tot Zam, Unihedron, Vogel612
158	□□ □□□□ □□	17slim, Amir Rachum, Andrew Brooke, Arthur, ben75, CarManuel, Daniel Nugent, EJP, Hi I'm Frogatto, Mark Yisri, Sadiq Ali, Skepter, Squidward
159	□□□	1d0m3n30, 4444, Aaron Digulla, Abhishek Jain, Alex Meiburg, alex s, Andrei Maieras, Andrii Abramov, Anony-Mousse, Bart Enkelaar, bitek, Blubberguy22, Bob Brinks, Burkhard, Cache Staheli, Cannon, Ce7, Chriss, code11, Codebender, Daniel Figueroa, daphshez, DVarga, Emil Sierżęga, enrico.bacis, Eran, faraa, hd84335, hexafraction, Jan Vladimir Mostert, Jens Schauder, Jorn Vernee, Jude Niroshan, kcoppock, Kevin Montrose, Lahiru Ashan, Lii, manfcas, Mani Muthusamy, Marc, Matt, Mistalis, Mshnik, mvd, Mzzzzzz, NatNgs, nishizawa23, Oleg Sklyar, Onur, Ortomala Lokni, paisanco, Paul Bellora, Paweł Albecki, PcAF, Petter Friberg, phant0m, philnate, Radouane ROUFID, RamenChef, rap-2-h, rd22, Rogério, rolve, RutledgePaulV, S.K. Venkat, Siguza, Stephen C, Stephen Leppik, suj1th, tainy, ThePhantomGamer, Thomas, TNT, ʇolɐəz əɥʇ qoq, Unihedron, Vlad-HC, Wesley, Wilson, yiwei, Yury Fedorov
160	□□	Jonathan, Makoto, rajah9, RamenChef, The Guy with The Hat, Uri Agassi
161	□□ □□□ □ □□ □□□	ChemicalFlash, DimaSan, fgb, hd84335, Mshnik, RamenChef, Sandesh, sargue, Slava Babin, Stephen C, tynn
162	□□	17slim, agilob, alain.janinm, ata, Binary Nerd, Burkhard, coobird, Dmitriy Kotov, Durgpal Singh, Emil Sierżęga, Emily Mabrey, Enigo, fabian, GPI, hd84335, J Atkin, Jabir, Javant, Javier Diaz, Jeffrey Bosboom, johnnyaug, Jonathan, Kakarot, KartikKannapur, Kenster, michaelbahr, Mo.Ashfaq, Nathaniel Ford, phatfingers, Ram, RamenChef, ravthiru, sebkur, Stephen C, Stephen Leppik, Viacheslav Vedenin, VISHWANATH N P, Vogel612
163	□□□	akhilsk, Batty, Bilesh Ganguly, Burkhard, EJP, emotionlessbananas, faraa, GradAsso, KIRAN KUMAR MATAM, noscreenname, Onur, rokonoid, Siva Sainath Reddy Bandi, Vasilis Vasilatos, Vasiliy Vlasov

164	□□ □□□ □□	Do Nhu Vy, giucal, Jorn Vernee, Lord Farquaad, Yohanes Khosiawan □□□
165	□□ □□	EJP, NageN, Thisaru Guruge
166	□□□	Adam Ratzman, Adil, Daniel M., Drayke, VISHWANATH N P
167	□□□	4castle, A_Arnold, Ad Infinitum, Alek Mieczkowski, alex s, altomnr, Andy Thomas, Anony-Mousse, Ashok Felix, Aurasphere, Bob Rivers, ced-b, ChandrasekarG, Chirag Parmar, clinomaniac, Codebender, Craig Gidney, Daniel Stradowski, dcod, DimaSan, Dušan Rychnovský, Enigo, Eran, fabian, fgb, GPI, Grzegorz Górkiewicz, ionyx, Jabir, Jan Vladimir Mostert, KartikKannapur, Kenster, KIRAN KUMAR MATAM, koder23, KudzieChase, Makoto, Maroun Maroun, Martin Frank, Matsemann, Mike H, Mo.Ashfaq, Mrunal Pagnis, mystarrocks, Oleg Sklyar, Pablo, Paweł Albecki, Petter Friberg, philnate, Polostor, Poonam, Powerlord, ppeterka, Prasad Reddy, Radiodef, rajadilipkolli, rd22, rdonuk, Ruslan Bes, Samk, SjB, Squidward, Stephen C, Stephen Leppik, Unihedron, user2296600, user3105453, Vasiliy Vlasov, Vasily Kabunov, VatsalSura, vsminkov, webo80, xploreraj
168	□□□ □□	John DiFini
169	□□ I / O	Aaron Franke, Ani Menon, Erkan Haspulat, Francesco Menzani, jayantS, Lankymart, Loris Securo, manetsus, Olivier Grégoire, Petter Friberg, rolve, Saagar Jha, Stephen C
170	□□□ □□□ □□□	Jacob G.
171	□□□ - Java Reflection	gobes, KIRAN KUMAR MATAM
172	□□□ □□	FFY00, Flow, Holger, Makoto, Stephen C
173	□□□ □□	Aaron Digulla, GPI, K'', Kenster, Ruslan Ulanov, Stephen C, trashgod
174	□□□□ □□	Community, Configure, Daniel LIn, Dave Ranjan, EJP, eveysky, fabian, Jens Schauder, Kevin Johnson, KIRAN KUMAR MATAM, MasterBlaster, Mureinik, Rakitić, Ram, RamenChef, Ryan Cocuzzo, Salman Kazmi, Tyler Zika
175	□□ □□	Ad Infinitum, Alon .G., Andrei Maieras, Andrii Abramov, bruno, Conrad.Dean, Dariusz, Demon Coldmist, Drizzt321, Dushko Jovanovski, fabian, faraa, GhostCat, hd84335, Hendrik Ebbers, J Atkin, Jorn Vernee, Kapep, Malt, MasterBlaster, matt freake, Nolequen, Ortomala Lokni, Ram, shmosel, Stephen C, Umberto Raimondi, Vogel612, ФХосę □ Перепра ヅ
176	□□ I / O	Alper Fırat Kaya, Arthur, assylias, ata, Aurasphere, Burkhard, Conrad.Dean, Daniel M., Enigo, FlyingPiMonster, Gerald Mücke, Gubbel, Hay, hd84335, Jabir, James Jensen, Jason Sturges, Jordy Baylac, leaqui, mateuscb, MikaelF, Moshiour, Myridium, Nicktar, Peter Gordon, Petter Friberg, ppeterka, RAnders00, RobAu, rokonoid, Sampada, sebkur, ShivBuyya, Squidward, Stephen C, still_learning, Tilo, Tobias Friedinger, TuringTux, Will Hardwick-Smith
177	□□□	JamesENL, KIRAN KUMAR MATAM
178	□□ / □□ □□□ □□□ □□□	Community, Joe C

	□□ □□□□□□	
179	□□□	1d0m3n30, Andreas, EJP, Li357, RamenChef, shmosel, Stephen C, Stephen Leppik
180	□□□□□□ □□□□□ □□□ □□□□□	alain.janinm, Dariusz, kajacx, Kenster, mnoronha
181	□□ □□	100rabh, A_Arnold, Alex, Andrii Abramov, Bob Rivers, Cache Staheli, DimaSan, Jasper, Kakarot, Kuroda, Manuel Vieda, Michael Piefel, phatfingers, RamenChef, Skylar Sutton, Vivek Anoop
182	□□□ □ □□□	Code.IT, dimo414, Eduard Wirch, emotionlessbananas, Squidward, sun-solar-arrow
183	□□□ □	Alexey Lagunov
184	□□ □□	RAnders00