

 免費電子書

學習

# Java Language

Free unaffiliated eBook created from  
**Stack Overflow contributors.**

#java

.....	1
<b>1: Java</b> .....	<b>2</b>
.....	2
<b>Java</b> .....	<b>2</b>
<b>Java</b> .....	<b>2</b>
<b>Java</b> .....	<b>2</b>
.....	2
.....	3
.....	3
.....	3
.....	3
Examples .....	3
Java .....	3
<b>Hello World</b> .....	<b>5</b>
<b>2: Apache Commons Lang</b> .....	<b>9</b>
Examples .....	9
equals .....	9
hashCode .....	9
toString .....	10
<b>3: AppDynamicsTIBCO BusinessWorks Instrumentation</b> .....	<b>12</b>
.....	12
Examples .....	12
AppdynamicsBW .....	12
<b>*** **</b> .....	<b>12</b>
<b>4: BigDecimal</b> .....	<b>13</b>
.....	13
Examples .....	13
BigDecimal .....	13
BigDecimals .....	13
BigDecimal .....	13
<b>1.Addition</b> .....	<b>13</b>

2.Subtraction.....	13
3.Multiplication.....	14
4.Division.....	14
5.....	15
6.Power.....	15
7.Max.....	15
8.Min.....	15
9.Move Point to Left.....	16
10.Move Point to Right.....	16
BigDecimalfloat.....	16
BigDecimal.valueOf.....	17
BigDecimals.....	17
<b>5: C ++.....</b>	<b>19</b>
.....	19
.....	19
.....	19
.....	19
C ++.....	19
Java.....	19
.....	19
C ++.....	19
Java.....	19
.....	20
C ++.....	20
Java.....	20
.....	20
.....	20
/.....	20
.....	21
.....	21

.....	21
C ++ .....	22
.....	22
java.lang.Object .....	22
Java CollectionsC ++ .....	22
Java .....	22
C ++ .....	22
.....	22
Examples .....	23
.....	23
C ++ .....	23
Java .....	23
.....	23
.....	23
C ++ .....	23
Java .....	24
.....	24
C ++ .....	24
Java .....	24
.....	24
C ++ .....	24
Java .....	25
.....	25
C ++ .....	25
Java .....	25
.....	26
.....	26
C ++ .....	26
Java .....	26
.....	26

.....	26
C ++ .....	26
Java .....	26
.....	26
C ++ .....	26
Java .....	27
.....	27
C ++ .....	27
Java .....	27
<b>6: Classpath .....</b>	<b>28</b>
.....	28
.....	28
Examples .....	28
.....	28
JAR .....	28
.....	29
.....	29
.....	29
.....	29
.....	30
.....	30
.....	30
<b>7: CompletableFuture .....</b>	<b>32</b>
.....	32
Examples .....	32
.....	32
CompletableFuture .....	32
<b>8: EnumSet .....</b>	<b>34</b>
.....	34
Examples .....	34
.....	34
<b>9: ExecutorExecutorServiceThread .....</b>	<b>35</b>
.....	

- ..... 35
- Examples..... 35
  - ..... 35
  - ThreadPoolExecutor..... 35
  - ..... 36
  - ..... 37
  - ..... 37
  - ..... 37
  - ..... 37
  - ..... 38
  - submitvs execute..... 38
  - ..... 40
  - ExecutorService..... 41
  - ExecutorService..... 42
  - ..... 44
- 10: FileUploadAWS..... 45**
  - ..... 45
  - Examples..... 45
    - s3..... 45
- 11: FTP..... 48**
  - ..... 48
  - ..... 48
  - Examples..... 48
    - FTP..... 48
- 12: HttpURLConnection..... 53**
  - ..... 53
  - Examples..... 53
    - URL..... 53
    - ..... 54
    - ..... 54
    - ..... 54

.....	55
.....	55
.....	55
.....	55
<b>13: InputStreamsOutputStreams</b> .....	<b>57</b>
.....	57
.....	57
Examples.....	57
InputStreamString.....	57
OutputStream.....	57
.....	58
.....	58
/.....	59
.....	59
/.....	59
DataInputStream.....	59
<b>14: Java Native Access</b> .....	<b>61</b>
Examples.....	61
JNA.....	61
<b>JNA</b> .....	<b>61</b>
.....	61
.....	61
<b>15: Java Native Interface</b> .....	<b>63</b>
.....	63
.....	63
Examples.....	63
JavaC ++.....	63
Java.....	63
C ++.....	64
.....	65
C ++Java.....	65

Java .....	65
C ++ .....	65
.....	66
.....	66
.....	66
.....	67
<b>16: Java SE 7 .....</b>	<b>68</b>
.....	68
.....	68
Examples .....	68
Java SE 7 .....	68
.....	68
try-with-resources .....	68
.....	68
.....	69
.....	69
<b>17: Java SE 8 .....</b>	<b>70</b>
.....	70
.....	70
Examples .....	70
Java SE 8 .....	70
<b>18: java.util.Objects .....</b>	<b>71</b>
Examples .....	71
null .....	71
<b>null check in .....</b>	<b>71</b>
<b>null .....</b>	<b>71</b>
apiObjects.nonNull .....	71
<b>19: JavaBean .....</b>	<b>72</b>
.....	72
.....	72
.....	72



Examples.....	72
Java Bean.....	72
<b>20: Java2D.....</b>	<b>74</b>
.....	74
Examples.....	74
1Java.....	74
2.....	76
<b>21: JavaJSON.....</b>	<b>77</b>
.....	77
.....	77
Examples.....	77
JSON.....	77
JSON.....	77
optXXX vs getXXX.....	78
JSONGson.....	78
JSONGson.....	79
JSON.....	79
Jackson Object Mapper.....	79
.....	<b>80</b>
ObjectMapper.....	80
.....	80
.....	80
JSON.....	80
JSON Builder - .....	81
JSONObject.NULL.....	81
JsonArrayJava ListGson Library.....	81
JacksonJSON.....	82
<b>JSON.....</b>	<b>82</b>
TypeFactory.....	83
TypeReference.....	83
<b>JSON.....</b>	<b>83</b>
TypeFactory.....	83



C ++ - .....	94
Java - .....	94
.....	94
.....	95
PermGen.....	95
Java.....	96
.....	96
.....	97
<b>25: Java - 'java'javaw'</b> .....	<b>98</b>
.....	98
.....	98
Examples.....	98
JAR.....	98
“”Java.....	98
HelloWorld.....	98
.....	99
.....	99
JavaFX.....	99
'java'.....	99
“”.....	99
“”.....	100
“<name>”.....	100
.....	101
Java.....	101
.....	101
POSIX shell.....	102
Windows.....	102
Java.....	103
-D.....	103
.....	103
.....	103

VM.....	104
<b>26: Java</b> .....	<b>105</b>
.....	105
.....	105
Examples.....	105
TCP.....	105
<b>27: Java</b> .....	<b>109</b>
Examples.....	109
.....	109
.....	109
Java.....	109
<b>28: Java</b> .....	<b>111</b>
.....	111
Examples.....	111
.....	111
.....	111
.....	112
.....	112
.....	112
.....	113
<b>PrintJobEvent pje</b> .....	<b>114</b>
.....	114
<b>29: Java</b> .....	<b>116</b>
.....	116
Examples.....	116
URLClassLoader.....	116
<b>30: Java</b> .....	<b>120</b>
.....	120
Examples.....	120
.....	120
OverFlowUnderFlow.....	122

.....	122
IEEE.....	123
<b>31: Java.....</b>	<b>124</b>
Examples.....	124
Java SE JREJava SE JDK.....	124
Java.....	124
Java.....	124
Oracle HotspotOpenJDK.....	125
Java EEJava SEJava MEJavaFX.....	125
<b>Java.....</b>	<b>125</b>
<b>Java SE.....</b>	<b>125</b>
<b>Java EE.....</b>	<b>125</b>
<b>Java ME.....</b>	<b>125</b>
<b>Java FX.....</b>	<b>126</b>
Java SE.....	126
Java SE.....	126
Java SE.....	127
<b>32: Java - 'javac'.....</b>	<b>128</b>
.....	128
Examples.....	128
'javac' - .....	128
.....	128
.....	128
'javac'.....	129
'javac'.....	130
.....	130
Java.....	130
Java.....	130
.....	130
<b>33: JavaJVM.....</b>	<b>132</b>
Examples.....	132
.....	.....

<b>34: Java</b> .....	<b>133</b>
.....	133
.....	133
Examples .....	133
JAR .....	133
JARWAREAR .....	134
MavenJARWAR .....	134
AntJARWAREAR .....	134
IDEJARWAREAR .....	134
jarJARWAREAR .....	134
Java Web Start .....	135
.....	135
JNLP .....	135
Web .....	136
.....	136
Web Start .....	136
UberJAR .....	136
<b>"jar"UberJAR</b> .....	<b>137</b>
<b>MavenUberJAR</b> .....	<b>137</b>
<b>UberJAR</b> .....	<b>137</b>
<b>35: Java - NullsNullPointerException</b> .....	<b>139</b>
.....	139
Examples .....	139
- NullPointerExceptions .....	139
- null .....	140
- "" .....	141
"a""b" .....	141
null .....	141
null"" .....	141
"" .....	141

/.....	141
.....	141
- null.....	142
- I / O.....	142
- "Yoda"NullPointerException.....	143
<b>36: Java - .....</b>	<b>144</b>
.....	144
.....	144
Examples.....	144
- .....	144
.....	144
- .....	144
- 'new'.....	145
- 'new StringString'.....	145
- System.gc.....	146
- .....	146
- Map.....	147
- size.....	147
- .....	148
PatternMatcher.....	148
findmatch.....	148
.....	149
.....	149
- ==.....	150
.....	150
'intern'.....	150
.....	150
.....	151
.....	151
- /.....	151
.....	152
.....	152

.....	152
Java .....	152
<b>37: Java - .....</b>	<b>153</b>
.....	153
Examples .....	153
- .....	153
- ThrowableExceptionErrorRuntimeException .....	153
- ThrowableExceptionErrorRuntimeException .....	154
“”ThrowableException .....	155
- InterruptedException .....	155
- .....	157
- .....	158
- `Throwable` .....	158
<b>38: Java - .....</b>	<b>159</b>
Examples .....	159
wait/ notify .....	159
“” .....	159
“” .....	159
/ .....	159
- 'java.lang.Thread' .....	159
- .....	161
- .....	161
.....	162
.....	163
.....	163
.....	164
.....	164
.....	164
<b>39: Java - .....</b>	<b>165</b>
.....	165
.....	165
Examples .....	165



- .....	165
- "" .....	165
- .....	165
- "" .....	167
- .....	168
- .....	169
- .....	169
- '==' .....	170
- .....	171
'assert' .....	171
.....	172
<b>40: JAXB .....</b>	<b>173</b>
.....	173
.....	173
.....	173
.....	173
Examples .....	173
XML .....	173
XML .....	174
XmlAdapterxml .....	174
/XML@XmlAccessorType .....	176
/XML .....	177
XmlAdapter .....	178
.....	178
.....	178
.....	179
XML .....	180
.....	180
XMLJava .....	180
XmlAdapter .....	181
<b>41: JAX-WS .....</b>	<b>182</b>
Examples .....	182
.....	

<b>42: JMX</b> .....	<b>183</b>
.....	183
Examples.....	183
Platform MBean Server.....	183
<b>43: JNDI</b> .....	<b>187</b>
Examples.....	187
RMIJNDI.....	187
<b>44: JShell</b> .....	<b>191</b>
.....	191
.....	191
.....	191
.....	191
Examples.....	191
JShell.....	191
JShell.....	191
JShell.....	191
.....	192
.....	192
.....	192
.....	192
<b>45: Just in TimeJIT</b> .....	<b>194</b>
.....	194
.....	194
Examples.....	194
.....	194
<b>46: JVM</b> .....	<b>196</b>
.....	196
Examples.....	196
1.0.....	196
JVMTI.....	198

Agent_OnLoad .....	199
<b>47: JVM</b> .....	<b>200</b>
.....	200
Examples .....	200
-XXaggressive .....	200
-XXallocClearChunks .....	200
-XXallocClearChunkSize .....	200
-XXcallProfiling .....	200
-XXdisableFatSpin .....	201
-XXdisableGCHeuristics .....	201
-XXdumpSize .....	201
-XXexitOnOutOfMemory .....	201
<b>48: Lambda</b> .....	<b>202</b>
.....	202
.....	202
Examples .....	202
Lambda .....	202
.....	202
.....	203
Java lambdas .....	203
.....	203
<b>Lambda</b> .....	<b>204</b>
.....	205
.....	205
<b>Lambda</b> .....	<b>205</b>
<b>Lambda</b> .....	<b>206</b>
.....	206
.....	206
.....	206
.....	207
.....	207
.....	

.....	207
Lambdas.....	208
lambda.....	208
`return`lambda.....	209
lambdaJava.....	210
Lambda - .....	212
Lambda.....	212
Lambda.....	213
lambda.....	213
<b>49: LinkedHashMap.....</b>	<b>215</b>
.....	215
Examples.....	215
Java LinkedHashMap.....	215
<b>50: log4j / log4j2.....</b>	<b>217</b>
.....	217
.....	217
.....	217
<b>Log4j 1.....</b>	<b>217</b>
Examples.....	217
Log4j.....	217
JavaLog4j.....	218
.....	219
log4j2.xml.....	219
log4j 1.x2.x.....	220
- DB.....	221
Logoutputlog4j 1.x.....	221
<b>51: Nashorn JavaScript.....</b>	<b>223</b>
.....	223
.....	223
.....	223
Examples.....	223

.....	223
Nashorn.....	223
JavaScript.....	224
.....	224
.....	224
NashornJavaScriptJava.....	225
.....	225
.....	226
<b>52: NIO - .....</b>	<b>227</b>
.....	227
Examples.....	227
SelectorOP_CONNECT.....	227
<b>53: Oracle.....</b>	<b>229</b>
.....	229
.....	229
Examples.....	229
.....	229
.....	229
.....	229
.....	229
.....	229
.....	230
.....	230
Java.....	230
.....	230
.....	230
.....	230
.....	231
.....	231
.....	231
.....	231

.....	231
.....	232
.....	232
.....	233
.....	234
.....	234
.....	<b>234</b>
.....	<b>234</b>
.....	234
.....	234
Lambda.....	235
.....	236
.....	236
.....	236
.....	<b>236</b>
<b>54: RSA.....</b>	<b>238</b>
Examples.....	238
OAEPGCM.....	238
<b>55: Stack-Walking API.....</b>	<b>243</b>
.....	243
Examples.....	243
.....	243
.....	243
.....	244
<b>56: StringBuilder.....</b>	<b>246</b>
.....	246
.....	246
.....	246
Examples.....	246
n.....	246

StringBufferStringBuilderFormatterStringJoiner.....	247
<b>57: sun.misc.Unsafe.....</b>	<b>249</b>
.....	249
Examples.....	249
sun.misc.Unsafe.....	249
bootclasspathsun.misc.Unsafe.....	249
.....	249
.....	250
<b>58: ThreadLocal.....</b>	<b>251</b>
.....	251
Examples.....	251
ThreadLocal Java 8.....	251
ThreadLocal.....	251
.....	253
<b>59: TreeMapTreeSet.....</b>	<b>255</b>
.....	255
Examples.....	255
JavaTreeMap.....	255
JavaTreeSet.....	255
JavaTreeMap / TreeSet.....	256
TreeMapTreeSet.....	257
<b>60: Varargs.....</b>	<b>259</b>
.....	259
Examples.....	259
varargs.....	259
Varargs.....	259
<b>61: WeakHashMap.....</b>	<b>261</b>
.....	261
Examples.....	261
WeakHashmap.....	261
<b>62: XJC.....</b>	<b>262</b>
.....	262

.....	262
.....	262
.....	262
Examples.....	262
XSDJava.....	262
<b>XSDschema.xsd.....</b>	<b>262</b>
<b>xjc.....</b>	<b>262</b>
.....	263
package-info.java.....	264
<b>63: XML XPath.....</b>	<b>265</b>
.....	265
Examples.....	265
XMLNodeList.....	265
XMLXPath.....	265
XMLXPath.....	266
<b>64: XOM - XML.....</b>	<b>268</b>
Examples.....	268
XML.....	268
XML.....	270
<b>65:.....</b>	<b>274</b>
.....	274
.....	274
Examples.....	274
.....	274
.....	274
.....	274
.....	275
<b>66:.....</b>	<b>276</b>
.....	276
Examples.....	276
.....	276



.....	276
.....	277
<b>67:</b> .....	<b>281</b>
.....	281
.....	281
Examples .....	281
.....	281
- .....	282
ThreadLocal .....	283
CountDownLatch .....	283
.....	285
.....	286
.....	287
.....	288
synchronized / volatile/ .....	289
java.lang.Thread .....	290
/ .....	292
/ .....	293
/ .....	295
.....	296
.....	296
ThreadPool`int` .....	297
.....	298
.....	299
.....	300
<b>68:</b> .....	<b>302</b>
.....	302
Examples .....	302
.....	302
.....	302
.....	<b>303</b>
ConcurrentHashMap .....	303

<b>69:</b>	<b>305</b>
.....	305
.....	305
Examples	305
.....	305
.....	306
.....	306
.....	306
BufferedImage	308
BufferedImage	309
BufferedImage	310
BufferedImage	310
<b>70: BufferedWriter</b>	<b>312</b>
.....	312
.....	312
Examples	312
.....	312
<b>71:</b>	<b>313</b>
.....	313
Examples	313
/	313
.....	313
2	314
2	315
java.util.BitSet	316
.....	317
<b>72: Fork / Join</b>	<b>318</b>
Examples	318
JavaFork / Join	318
<b>73: JAXP APIXML</b>	<b>320</b>
.....	320
<b>DOM</b>	<b>320</b>
<b>SAX</b>	<b>320</b>

<b>StAX</b> .....	<b>320</b>
Examples.....	320
DOM API.....	320
StAX API.....	321
<b>74: static</b> .....	<b>324</b>
.....	324
Examples.....	324
static.....	324
.....	324
.....	325
<b>75:</b> .....	<b>326</b>
.....	326
.....	326
Examples.....	326
try-catch.....	326
.....	326
.....	327
.....	327
.....	328
.....	328
.....	329
try-with-resources.....	330
.....	330
try-with-resource.....	330
try-with-resource.....	330
.....	331
try-with-resourcetry-catch-finally.....	331
.....	332
.....	332
.....	333
.....	334

stacktraceString.....	335
InterruptedException.....	335
Java - .....	336
.....	336
.....	337
.....	338
try catch.....	340
.....	340
callstack.....	341
.....	341
.....	341
.....	341
try-finallytry-catch-finally.....	342
.....	342
try-catch-.....	342
'throws'.....	343
.....	343
.....	343
<b>76:</b> .....	<b>345</b>
.....	345
Examples.....	345
SealedObjectjavax.crypto.SealedObject.....	345
SignedObjectjava.security.SignedObject.....	345
<b>77:</b> .....	<b>347</b>
.....	347
.....	347
Examples.....	347
Deque.....	347
Deque.....	347
.....	347
Deque.....	348
<b>78: SET</b> .....	<b>349</b>

.....	349
Examples.....	349
.....	349
<b>79:</b> .....	<b>350</b>
.....	350
Examples.....	350
Java.....	350
<b>80:</b> .....	<b>352</b>
.....	352
.....	352
Examples.....	352
- .....	352
<b>81:</b> .....	<b>355</b>
.....	355
.....	355
Examples.....	355
Packages.....	355
.....	355
<b>82:</b> .....	<b>357</b>
.....	357
.....	357
.....	357
Examples.....	357
int.....	357
.....	358
.....	358
.....	359
.....	360
.....	360
.....	361
char.....	362
.....	362

.....	363
.....	364
.....	364
Cheatsheet.....	364
<b>83:</b> .....	<b>366</b>
.....	366
.....	366
.....	366
Examples.....	366
.....	366
.....	366
.....	367
.....	368
<b>84:</b> .....	<b>369</b>
Examples.....	369
.....	369
.....	369
<b>85:</b> .....	<b>370</b>
Examples.....	370
.....	370
<b>86: API</b> .....	<b>372</b>
.....	372
.....	372
.....	<b>372</b>
Examples.....	372
.....	372
.....	373
.....	374
.....	375
.....	375
.....	375
.....	375

Class.....	376
.....	376
Reflection API.....	377
.....	378
.....	378
Java.....	379
<b>87:</b> .....	<b>382</b>
.....	382
.....	382
Examples.....	382
Comparable.....	382
<b>Lambda</b> .....	<b>385</b>
.....	385
.....	385
compareTo.....	386
.....	386
.....	387
.....	388
<b>88:</b> .....	<b>389</b>
.....	389
.....	389
Examples.....	389
.....	389
.....	390
.....	390
.....	390
.....	391
.....	391
<b>89:</b> .....	<b>392</b>
.....	392
.....	392

Examples.....	392
Optional.....	392
.....	392
.....	393
.....	393
.....	394
.....	394
.....	394
FlatMap.....	394
<b>90:</b> .....	<b>396</b>
.....	396
Examples.....	396
GettersSetter.....	396
settergetter.....	396
.....	397
<b>91:</b> .....	<b>399</b>
.....	399
.....	399
.....	399
Examples.....	399
GWT ToolBase.....	399
.....	400
.....	400
.....	400
“flag”.....	400
<b>92:</b> .....	<b>402</b>
.....	402
Examples.....	402
.....	402
<b>93:</b> .....	<b>403</b>
Examples.....	403
.....	403



PreferenceChangeEvent .....	403
NodeChangeEvent .....	403
.....	403
.....	404
.....	405
.....	406
.....	407
.....	407
.....	408
.....	408
<b>94:</b> .....	<b>409</b>
.....	409
.....	409
.....	<b>409</b>
.....	<b>409</b>
Examples .....	409
.....	409
.....	<b>410</b>
.....	<b>410</b>
.....	<b>411</b>
<b>95:</b> .....	<b>412</b>
.....	412
Examples .....	412
Enum Singleton .....	412
Singleton .....	412
SingletonEnum .....	413
holderBill Pugh Singleton .....	413
.....	413
<b>96: Java</b> .....	<b>417</b>
.....	417
.....	417
Examples .....	417

nashorn-scriptingjavascript.....	417
<b>97: MultiThreadedThreadPoolExecutor.....</b>	<b>420</b>
.....	420
Examples.....	420
Runnable.....	420
.....	421
Lambdas.....	424
<b>98: .....</b>	<b>426</b>
.....	426
.....	426
Examples.....	426
.....	426
.....	426
Java 8.....	427
.....	429
Map.....	430
.....	430
Map <XY>Map <YZ>Map <XZ>.....	431
.....	431
.....	<b>432</b>
.....	432
.....	435
HashMap.....	435
.....	436
.....	<b>436</b>
<b>99: .....</b>	<b>438</b>
.....	438
Examples.....	438
If / Else If / Else Control.....	438
.....	438
.....	439
.....	439

.....	440
.....	441
.....	441
.....	442
.....	443
.....	443
/.....	444
Java.....	444
<b>100:</b> .....	<b>445</b>
.....	445
Examples.....	445
JMH.....	445
<b>101:</b> .....	<b>448</b>
.....	448
.....	448
Examples.....	448
.....	448
.....	449
.....	450
.....	451
.....	452
<b>102: JAR</b> .....	<b>456</b>
.....	456
Examples.....	456
Jar.....	456
jarJar.....	456
JarURL.....	457
<b>103:</b> .....	<b>459</b>
.....	459
Examples.....	459
.....	459
<b>104:</b> .....	<b>460</b>

.....	460
.....	460
Examples.....	460
.....	460
<b>==</b> .....	<b>461</b>
<b>switch</b> .....	<b>461</b>
.....	462
.....	462
.....	462
String.....	463
.....	464
.....	465
.....	465
n.....	465
.....	466
toString.....	466
.....	467
.....	469
.....	469
.....	470
StringBuilders.....	470
.....	472
.....	472
.....	472
.....	472
.....	472
.....	472
.....	472
.....	473
.....	473
.....	473
.....	475

<b>105:</b>	<b>476</b>
.....	476
Examples	476
StringTokenizer	476
StringTokenizer"	476
<b>106:</b>	<b>477</b>
Examples	477
UTF-8	477
UTF-8	477
UTF-8	478
<b>107:</b>	<b>479</b>
Examples	479
.....	479
.....	479
.....	479
/	479
.....	479
ASMjar	479
ClassNode	482
jar	483
Javassist Basic	483
<b>108:</b>	<b>485</b>
.....	485
.....	485
Examples	485
- ByteBuffer	485
-	485
- DirectByteBuffer	486
<b>109:</b>	<b>487</b>
.....	487
Examples	487
.....	

.....	487
.....	488
Unicode.....	488
.....	489
.....	489
.....	489
.....	489
.....	489
.....	490
.....	490
.....	490
Null.....	490
.....	491
.....	491
.....	491
.....	491
.....	491
.....	492
.....	492
.....	492
<b>110:</b> .....	<b>493</b>
Examples.....	493
.....	493
.....	493
/.....	494
.....	494
/.....	495
<b>111:</b> .....	<b>496</b>
.....	496
.....	496
Examples.....	496
JCE.....	496

.....	496
Java.....	496
.....	496
.....	496
.....	496
<b>112:</b> .....	<b>498</b>
Examples.....	498
SecurityManager.....	498
ClassLoader.....	498
.....	499
DeniedPermission.....	499
DenyingPolicy.....	504
.....	506
<b>113: Java</b> .....	<b>507</b>
.....	507
Examples.....	507
WindowsPATHJAVA_HOME.....	507
.....	507
.....	507
.....	508
Java SE.....	508
Java.....	508
Java.....	509
LinuxJava JDK.....	509
.....	509
Oracle Java RPM.....	510
WindowsJava JDKJRE.....	510
macOSJava JDK.....	511
LinuxJava.....	512
.....	<b>512</b>
<b>Arch</b> .....	<b>513</b>
.....	513
.....	

513	
Linux.....	513
tarLinuxoracle java.....	514
.....	515
<b>114:</b> .....	<b>516</b>
.....	516
.....	516
Examples.....	516
.....	516
.....	517
<b>115:</b> .....	<b>519</b>
.....	519
Examples.....	519
.....	519
.....	519
<b>116:</b> .....	<b>520</b>
.....	520
.....	520
Examples.....	520
toString.....	520
equals.....	521
.....	522
hashCode.....	523
Arrays.hashCode.....	524
.....	524
waitnotify.....	525
getClass.....	526
clone.....	527
finalize.....	528
.....	529
<b>117:</b> .....	<b>531</b>
.....	531



Examples.....	531
.....	531
Clonable.....	531
.....	532
.....	533
.....	534
<b>118:</b> .....	<b>535</b>
.....	535
Examples.....	535
.....	535
<b>119:</b> .....	<b>538</b>
.....	538
.....	538
Examples.....	538
.....	538
GUI.....	539
applet.....	539
.....	540
.....	540
.....	540
.....	541
<b>120:</b> .....	<b>542</b>
.....	542
.....	542
.....	542
Examples.....	542
.....	542
.....	543
XML.....	545
<b>121:</b> .....	<b>547</b>
.....	547
.....	547

.....	547
.....	547
.....	547
Examples.....	547
.....	547
.....	548
.....	550
.....	551
.....	551
.....	552
.....	552
.....	553
<b>122:</b> .....	<b>554</b>
.....	554
.....	554
.....	554
.....	<b>554</b>
.....	<b>554</b>
.....	<b>554</b>
.....	<b>554</b>
Examples.....	554
.....	554
Pattern.....	555
.....	556
.....	556
.....	557
.....	557
<b>123: Java</b> .....	<b>559</b>
.....	559
Examples.....	559
==Integer.....	559
.....	559

.....	560
==.....	561
.....	562
.....	563
.....	563
.....	564
.....	564
.....	565
.....	565
String.....	566
<b>124:</b> .....	<b>567</b>
.....	567
Examples.....	567
Java.....	567
Gson.....	568
2.....	569
.....	570
serialVersionUID.....	572
.....	573
.....	573
JacksonJSON.....	573
<b>125:</b> .....	<b>576</b>
.....	576
.....	576
Examples.....	576
javap.....	576
<b>126:</b> .....	<b>584</b>
.....	584
.....	584
.....	584
Examples.....	584
Scanner.....	584

.....	584
ScannerString.....	585
.....	585
.....	585
int.....	587
.....	587
<b>127:</b> .....	<b>589</b>
.....	589
.....	589
Examples.....	589
.....	589
.....	590
.....	590
Generics.....	591
.....	593
.....	594
.....	595
.....	<b>595</b>
.....	<b>596</b>
.....	<b>596</b>
.....	597
.....	597
.....	597
.....	598
<b>128: I / O.....</b>	<b>599</b>
Examples.....	599
.....	599
BufferedReader .....	599
Scanner .....	599
System.console .....	600
.....	600
.....	601

.....	602
<b>129:</b> .....	<b>603</b>
.....	603
.....	603
.....	603
<b>Examples</b> .....	<b>603</b>
.....	603
.....	603
.....	604
<b>130:</b> .....	<b>605</b>
.....	605
.....	605
.....	605
<b>Examples</b> .....	<b>605</b>
.....	605
.....	<b>605</b>
.....	<b>606</b>
.....	<b>606</b>
.....	<b>607</b>
.....	<b>608</b>
<b>Java</b> .....	<b>608</b>
.....	<b>608</b>
.....	<b>609</b>
.....	<b>609</b>
.....	<b>610</b>
.....	<b>610</b>
.....	610
.....	611
.....	612
<b>Arrays.asList</b> .....	<b>613</b>
.....	613

<b>Java</b> .....	<b>614</b>
ArrayIndexOutOfBoundsException.....	615
.....	616
.....	616
.....	617
.....	617
.....	619
<b>for</b> .....	<b>619</b>
<b>Object.clone</b> .....	<b>620</b>
<b>Arrays.copyOf</b> .....	<b>620</b>
<b>System.arraycopy</b> .....	<b>620</b>
<b>Arrays.copyOfRange</b> .....	<b>620</b>
.....	620
.....	621
ArrayList.....	621
System.arraycopy.....	621
Apache Commons Lang.....	621
.....	622
.....	622
.....	623
.....	623
Arrays.binarySearch.....	623
Arrays.asList.....	623
Stream.....	623
.....	624
org.apache.commons.....	624
.....	624
.....	624
.....	626
<b>131: I/O</b> .....	<b>627</b>
.....	627

Examples.....	627
byte [].....	627
.....	627
byte [].....	627
Stream vs Writer / Reader API.....	628
.....	629
.....	629
.....	630
java.io.FileJava 7 NIOjava.nio.file.Path.....	630
.....	630
.....	630
/Path.....	630
.....	631
OutputStream.....	631
.....	631
.....	632
FileInputStream / FileOutputStream/.....	632
.....	634
.....	634
InputStreamOutputStream.....	635
ChannelBuffer.....	635
Channel.....	636
BufferedInputStream.....	636
ChannelBuffer.....	637
PrintStream.....	637
.....	638
.....	638
/.....	638
ZIP.....	639
.....	639
.....	639

<b>132: I / O.....</b>	<b>641</b>
.....	641
Examples.....	641
.....	641
.....	641
.....	642
.....	<b>642</b>
.....	<b>642</b>
.....	642
.....	<b>642</b>
.....	<b>642</b>
.....	<b>643</b>
<b>MIME.....</b>	<b>643</b>
.....	643
.....	643
<b>133: .....</b>	<b>644</b>
.....	644
.....	644
.....	644
Examples.....	644
.....	644
<b>134: .....</b>	<b>645</b>
.....	645
Examples.....	645
.....	645
/.....	645
/.....	645
.....	645
<b>135: java.time*.....</b>	<b>647</b>
Examples.....	647
.....	647
.....	.....



647	648
.....	648
.....	648
Date Time API.....	648
.....	650
2LocalDates.....	651
<b>136:</b> .....	<b>652</b>
.....	652
.....	652
.....	652
Examples.....	652
Date.....	652
Date.....	653
<b>LocalDate</b> .....	<b>653</b>
<b>beforeaftercompareToequals</b> .....	<b>653</b>
<b>isBeforeisAftercompareToequals</b> .....	<b>654</b>
<b>Java 8</b> .....	<b>655</b>
<b>Java 8</b> .....	<b>655</b>
.....	656
.....	656
.....	656
Date.....	657
.....	657
Java 8 LocalDateLocalDateTime.....	658
java.util.Date.....	659
java.util.Datejava.sql.Date.....	659
.....	660
<b>137:</b> .....	<b>661</b>
.....	661
Examples.....	661
Apache HashBagGuava HashMultisetEclipse HashBag.....	661

1. ApacheSynchronizedSortedBag .....	661
2.EclipseTreeBagGC .....	662
3. GuavaLinkedHashMultiset .....	662
.....	663
GuavaApacheEclipseMultimap.....	663
Nore .....	665
- .....	666
- .....	666
<b>138:</b> .....	<b>670</b>
.....	670
Examples.....	670
.....	670
<b>139:</b> .....	<b>671</b>
.....	671
.....	671
Java.....	671
Examples.....	671
"locale".....	671
<b>Java</b> .....	<b>671</b>
.....	671
.....	672
.....	672
.....	672
<b>Java ResourceBundle</b> .....	<b>672</b>
.....	673
<b>140:</b> .....	<b>674</b>
.....	674
.....	674
.....	674
.....	674
.....	674

Examples.....	674
.....	674
.....	677
.....	679
.....	680
Enum.....	681
.....	681
.....	682
.....	682
.....	683
.....	683
Singleton.....	683
.....	684
String.....	685
<b>name()</b> .....	<b>685</b>
<b>toString()</b> .....	<b>685</b>
.....	685
.....	685
.....	685
.....	687
.....	687
.....	688
<b>141:</b> .....	<b>690</b>
.....	690
Examples.....	690
.....	690
<b>142:</b> .....	<b>691</b>
.....	691
Examples.....	691
.....	691
<b>143:</b> .....	<b>692</b>
.....	692

.....	692
Examples.....	692
.....	692
.....	693
.....	693
<b>144:</b> .....	<b>695</b>
.....	695
.....	695
Examples.....	695
.....	695
<b>145:</b> .....	<b>696</b>
.....	696
.....	696
.....	696
Examples.....	696
.....	696
.....	<b>697</b>
.....	<b>698</b>
.....	698
.....	699
“AB”.....	700
.....	700
‘T’‘T’‘T’.....	701
.....	703
.....	<b>703</b>
.....	<b>703</b>
.....	<b>703</b>
.....	703
.....	704
.....	704
.....	704
.....	704

instanceof.....	705
.....	707
.....	708
.....	708
<b>146:</b> .....	<b>710</b>
.....	710
.....	710
Examples.....	710
Streams.....	710
.....	711
.....	711
.....	712
.....	712
toList()toSet().....	712
ListSet.....	712
.....	714
.....	714
.....	715
h21.....	716
.....	716
.....	716
.....	717
.....	717
.....	717
.....	718
.....	718
.....	719
IntStreamString.....	719
.....	720
.....	720
.....	720

721	
IntStream	721
flatMap	721
	722
Streams	723
Streams	724
	724
Map.Entry	725
	725
	<b>725</b>
	<b>726</b>
	<b>726</b>
	<b>726</b>
	726
Stresh	726
String	729
<b>147:</b>	<b>730</b>
	730
Examples	730
-	730
	730
<b>148:</b>	<b>732</b>
	732
	732
	732
Examples	732
	732
	734
	735
	736
BA	737
2	737

.....	737
ArrayList.....	738
List.....	738
.....	739
.....	739
List - .....	740
List.....	740
.....	740
.....	740
AttributeList.....	741
CopyOnWriteArrayList.....	741
.....	741
RoleList.....	741
RoleUnresolvedList.....	741
.....	741
.....	741
<b>149: SortedMap.....</b>	<b>742</b>
.....	742
Examples.....	742
.....	742
<b>150: Java.....</b>	<b>743</b>
Examples.....	743
JSONPOJO.....	743
<b>151: .....</b>	<b>744</b>
.....	744
.....	744
.....	744
Examples.....	744
.....	744
.....	744
LocalTime.....	745
.....	746

<b>152: BigInteger</b> .....	<b>747</b>
.....	747
.....	747
.....	747
Examples.....	747
.....	747
BigInteger.....	748
BigInteger.....	749
BigInteger.....	751
BigInteger.....	752
<b>153: NumberFormat</b> .....	<b>753</b>
Examples.....	753
NumberFormat.....	753
<b>154: ServiceLoader</b> .....	<b>754</b>
.....	754
Examples.....	754
.....	754
.....	<b>754</b>
.....	754
<b>META-INF // servicetest.Logger</b> .....	<b>755</b>
.....	<b>755</b>
ServiceLoader.....	755
<b>155: StringBuffer</b> .....	<b>757</b>
.....	757
Examples.....	757
String Buffer.....	757
<b>156:</b> .....	<b>759</b>
.....	759
Examples.....	759
.....	759
.....	759
.....	.....



.....	759
.....	760
- .....	760
InputStreamTrustStoreKeyStore.....	761
- .....	762
UDP/.....	763
.....	764
SSL.....	765
Channel.....	766
.....	766
<b>157:</b> .....	<b>768</b>
.....	768
.....	768
Examples.....	768
intInteger.....	768
ifBoolean.....	769
NullPointerException.....	769
.....	769
Integerint.....	770
<b>158:</b> .....	<b>772</b>
.....	772
Examples.....	772
Java<1.5.....	772
ProcessBuilder.....	772
.....	773
ch.vorburger.exec.....	773
Runtime.execProcessProcessBuildershell.....	773
.....	774
shell.....	774
Shell.....	775
<b>159:</b> .....	<b>776</b>

.....	776
.....	776
Examples.....	776
.....	776
.....	777
.....	777
.....	778
.....	778
.....	778
.....	779
.....	779
.....	780
.....	780
<b>160: java.util.logging.....</b>	<b>781</b>
Examples.....	781
.....	781
.....	781
.....	782
<b>161: Java.....</b>	<b>784</b>
.....	784
.....	784
.....	784
Examples.....	784
.....	784
.....	785
.....	786
.....	786
.....	786
Javadocs.....	787
.....	788
.....	788
<b>162: .....</b>	<b>790</b>

.....	790
.....	790
.....	790
.....	<b>790</b>
<b>Examples</b> .....	790
.....	790
.....	793
.....	793
.....	793
.....	<b>793</b>
<b>@</b> .....	794
.....	794
<b>@</b> .....	795
.....	795
<b>@Documented</b> .....	795
<b>@</b> .....	795
<b>@Repeatable</b> .....	795
.....	796
.....	797
.....	798
.....	798
.....	799
.....	<b>799</b>
.....	<b>799</b>
.....	<b>800</b>
.....	<b>800</b>
<b>javac</b> .....	<b>801</b>
<b>IDE</b> .....	<b>801</b>
Netbeans.....	801
.....	801
.....	802

'this' .....	802
.....	803
<b>163:</b> .....	<b>804</b>
.....	804
Examples .....	804
BufferedReader .....	804
.....	804
<b>BufferedReader</b> .....	<b>804</b>
<b>BufferedReader</b> .....	<b>804</b>
<b>BufferedReader.readLine</b> .....	<b>804</b>
<b>List</b> .....	<b>804</b>
StringWriter .....	805
<b>164:</b> .....	<b>806</b>
Examples .....	806
.....	806
<b>165:</b> .....	<b>807</b>
.....	807
.....	807
Examples .....	808
.....	808
.....	808
JAR .....	808
.....	808
.....	809
.....	809
get .....	809
<b>166:</b> .....	<b>810</b>
Examples .....	810
.....	810
.....	810
.....	810
.....	810

**167:** ..... **812**

Examples ..... 812

String ..... 812

/ ..... 812

Base64/ ..... 813

..... 814

`InputStream` `String` ..... 814

String ..... 815

**168: Iterable** ..... **817**

..... 817

..... 817

Examples ..... 817

forIterable ..... 817

..... 817

Iterable ..... 817

..... 818

**169:** ..... **820**

..... 820

..... 820

Examples ..... 820

+ ..... 820

..... 820

+ - \*/ ..... 821

..... 822

..... 822

..... 822

..... 823

INFNAN ..... 823

=== ..... 823

Numeric == != ..... 824

== != ..... 824

Reference ==!=	824
NaN	825
/++ / -	825
:)	825
.....	<b>825</b>
.....	<b>826</b>
^	827
.....	827
.....	828
=+ = - =* =/ ==<< ==>> ==>> ==  =^ =	828
&&	830
- &&	830
- &&	831
<<>>>>	831
Lambda - >	832
<<=> =	832
<b>170:</b>	<b>834</b>
Examples	834
.....	834
<b>171:</b>	<b>835</b>
.....	835
.....	835
.....	835
.....	<b>835</b>
JavaTail-call	835
Examples	835
.....	835
N	836
1N	836
N	836
Recursion	837
.....	837

.....	838
StackOverflowError.....	838
.....	838
.....	<b>838</b>
.....	838
Java.....	840
Java.....	841
<b>172: RMI.....</b>	<b>842</b>
.....	842
Examples.....	842
Client-ServerJVMJVM.....	842
“”.....	843
.....	843
.....	844
.....	844
RMI.....	847
.....	847
.....	849
.....	850
<b>173: .....</b>	<b>851</b>
.....	851
Examples.....	851
Java.....	851
<b>174: .....</b>	<b>852</b>
.....	852
.....	852
.....	852
Examples.....	852
.....	852
.....	853
'final'.....	854
.....	854

.....	855
.....	855
.....	855
.....	856
.....	857
.....	858
.....	858
.....	859
“Is-a”vs“Has-a” .....	861
.....	864
<b>175: Deques</b> .....	<b>865</b>
Examples.....	865
PriorityQueue.....	865
LinkedListFIFO.....	865
.....	866
.....	<b>866</b>
<b>API</b> .....	<b>866</b>
.....	<b>866</b>
BlockingQueue.....	867
.....	868
.....	868
.....	<b>869</b>
.....	<b>869</b>
<b>176:</b> .....	<b>870</b>
.....	870
Examples.....	870
.....	870
.....	870
.....	871
.....	871
.....	872
.....	872
apache-common lang3.....	872



<b>177:</b> .....	<b>874</b>
Examples .....	874
HashSet .....	874
.....	874
<b>HashSet -</b> .....	<b>874</b>
<b>LinkedHashSet -</b> .....	<b>874</b>
<b>TreeSet - compareTo()Comparator</b> .....	<b>874</b>
.....	875
.....	875
Set .....	876
Set .....	877
<b>178:</b> .....	<b>878</b>
.....	878
.....	878
Examples .....	878
ArrayList .....	879
.....	879
.....	<b>879</b>
Java Collections .....	879
Google Guava Collections .....	879
.....	<b>880</b>
Java Collections .....	880
Apache Commons Collections .....	880
Google Guava Collections .....	880
.....	880
List .....	881
.....	<b>881</b>
forSkips“Banana” .....	881
for .....	881
.....	<b>882</b>
Iteratorwhile .....	882

.....	882
.....	883
.....	883
.....	883
.....	883
removelf.....	883
.....	883
.....	884
.....	<b>884</b>
<b>Set</b> .....	<b>885</b>
.....	<b>885</b>
.....	886
.....	886
Iterator.....	886
IterableIteratorfor-each.....	887
.....	889
.....	889
subListint fromIndexint toIndex.....	<b>889</b>
subSetfromIndextoIndex.....	<b>890</b>
subMapfromKeytoKey.....	<b>890</b>
<b>179:</b> .....	<b>891</b>
.....	891
Examples.....	891
.....	891
.....	892
.....	892
.....	894
.....	894
.....	895
strictfp.....	895
<b>180:</b> .....	<b>896</b>
.....	896

Examples.....	896
.....	896
MIDI.....	896
.....	897
.....	898
<b>181: - Java.....</b>	<b>899</b>
.....	899
Examples.....	899
ObjectgetClass.....	899
<b>182:.....</b>	<b>900</b>
.....	900
Examples.....	900
.....	900
classLoader.....	900
.class.....	901
<b>183:.....</b>	<b>902</b>
.....	902
.....	902
Examples.....	902
.....	902
.....	902
.....	903
.....	903
.....	906
.....	906
.....	907
<b>184:.....</b>	<b>910</b>
.....	910
Examples.....	910
.....	910
.....	910
.....	910

.....	.911
instanceof.....	.911
<b>185:</b> .....	<b>912</b>
.....	.912
.....	.912
.....	.912
.....	.912
.....	.912
.....	.912
.....	.912
Examples.....	.912
.....	.913
.....	.913
.....	.914
.....	.914
.....	.915
.....	.916
.....	<b>918</b>

---

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: [java-language](#)

It is an unofficial and free Java Language ebook created for educational purposes. All the content is extracted from [Stack Overflow Documentation](#), which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official Java Language.

The content is released under Creative Commons BY-SA, and the list of contributors to each chapter are provided in the credits section at the end of this book. Images may be copyright of their respective owners unless otherwise specified. All trademarks and registered trademarks are the property of their respective company owners.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to [info@zzzprojects.com](mailto:info@zzzprojects.com)

---

# 1: Java

Java.....

- ◦
- ◦ ◦ JavaScript◦
- ◦ String◦
- Java◦
- javacjavac JVM◦

Java“WORAJavaJava◦

Java .classJavaJVM◦ JavaJVM◦ JVM◦ “JIT”◦

---

# Java

Sun / OracleJava

- JavaSE◦
- Java Enterprise EditionEEJava“”◦ Java EE ◦
- Java Micro EditionMEJava SE◦

Java SE / EE / ME◦

- Java SE◦

---

# Java

Java◦

---

# Java

- Java
- JavaJAR
- Java
- Classpath

---

Java◦ Java◦

- [Java](#)
- [Java](#)
- [Java](#)
- [Java](#)
- [Java](#)
- [Java](#)
- [Java](#)

---

Java。

- [JUnit](#)
- [TestNG](#)

- 
- Java 。
  - [Android](#)。
  - [Java EE](#) Java Enterprise Edition。
  - [JavaFX](#)。

1.“ ” OracleJava SE。 Java SE[Oracle Java SE](#) 。

Java SE	1	
<a href="#">Java SE 9</a>		2017727
<a href="#">Java SE 8</a>		2014318
<a href="#">Java SE 7</a>	2015414	2011-07-28
<a href="#">Java SE 6</a>	2013416	2006-12-23
<a href="#">Java SE 5</a>	2009-11-04	2004-10-04
<a href="#">Java SE 1.4</a>	2009-11-04	2002-02-06
<a href="#">Java SE 1.3</a>	2009-11-04	200058
<a href="#">Java SE 1.2</a>	2009-11-04	1998-12-08
<a href="#">Java SE 1.1</a>	2009-11-04	1997219
<a href="#">Java SE 1.0</a>	2009-11-04	1996121

## Examples

Java

HelloWorld.java [IDE](#)

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

## Ideone

Java `public class 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.  
IDE **ex** eclipse

**windows** `javac.exeC:\Program Files\Java\jdk(version number)\bin`

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

`javac OSjavac PATH`

Windows XP / Vista / `PATHPATH`

- `=>=>`
- `""=>`
- `""""=>`

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

**""IN FRONT ;**

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

.

Linux



`javac` **Java**

HelloWorld.class **JavaJVM** • **Java** `javac` **Java** `bytecode` • Pluggable Annotation Processing API • **Java** Compiler API.

`javamainHelloWorld` • `.class`

```
$ java HelloWorld
```

`java` **Java**

**Java**

**Java** `java` `javac`

- **JDK** [Oracle](#) [OpenJDK](#)
- 

**JVM** `javac` `java` • `java -version` `javac -version` • 1.8.0\_73 •

---

# Hello World

“Hello World” `HelloWorldmainmain`

```
public class HelloWorld {
```

`classHelloWorld` • **Java** •

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

`public static void main(String[])` **JVM** • **Java** •

- `public` •
- `static` •
- `void` • **CC ++int** **Java** `System.exit()` •
- `String args` **main** •

**Java** •

- `argsargs` •
- `String[] args` **Varargs** `String... args` **varargs** •

`main` • `java` •

**main**

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

System	java.langSystem°
.	"" 1 ;° Systemout°
out	SystemPrintStream°
.	° outprintln°
println	PrintStream° °
(	println°
"Hello, World!"	printlnString° String°
)	println°
;	°

Java ; °

°

```
    } // end of main function 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 °

## gettermain

```
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

ideone

Testmain◦ mainJavaJVM◦

---

1 - HelloWorldSystempublic◦

Java <https://riptutorial.com/zh-TW/java/topic/84/java>

---

## 2: Apache Commons Lang

### Examples

#### equals

equalsEqualsBuilder°

```
@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);
}
```

#### booleanequals°

```
@Override
public boolean equals(Object obj) {
    return EqualsBuilder.reflectionEquals(this, obj, "field1", "field2");
}
```

#### hashCode

hashCodeHashCodeBuilder°

```
@Override
public int hashCode() {

    hashCodeHashCodeBuilder builder = new hashCodeHashCodeBuilder();
    builder.append(field1);
    builder.append(field2);
    builder.append(field3);

    return builder.hashCode();
}
```

```
@Override
public int hashCode() {
    return HashCodeBuilder.reflectionHashCode(this, false);
}
```

## boolean

```
@Override
public int hashCode() {
    return HashCodeBuilder.reflectionHashCode(this, "field1", "field2");
}
```

## toString

### toStringToStringBuilder

```
@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
]
```

## JSONClassnameshort.....

```
@Override
public String toString() {
    return ToStringBuilder.reflectionToString(this);
}
```

```
@Override
public String toString() {
    return ToStringBuilder.reflectionToString(this, ToStringStyle.JSON_STYLE);
}
```

Apache Commons Lang <https://riptutorial.com/zh-TW/java/topic/3338/apache-commons-lang>

---

# 3: AppDynamicsTIBCO BusinessWorks Instrumentation

AppDynamicsDevOps◦ AppDTIBCO BW◦ BW◦

## Examples

### AppdynamicsBW

1. TIBCO\_HOME / bw / 5.12 / bin / bwengine.traLinuxTIBCO 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. ◦ ◦

AppDynamicsTIBCO BusinessWorks Instrumentation <https://riptutorial.com/zh-TW/java/topic/10602/appdynamicstibco-businessworks-instrumentation>



---

# 4: BigDecimal

BigDecimal。 BigDecimal。 。

## Examples

### BigDecimal

#### BigDecimalBigDecimal

```
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
```

### BigDecimals

compareToBigDecimals

```
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

BigDecimals。

---

## 1.Addition

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

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

## 2.Subtraction

```
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.Multiplication

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

multiplyMathContext - ◦ 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;**◦

```
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.

```
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.Max

```
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.Min

```
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

---

## 9. Move Point to Left

```
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. Move Point to Right

```
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.

### BigDecimalfloat

- . . . Java 7BigDecimal.

```
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

```

10000.001.9910008010.00。 8009.77。 BigDecimal。

## BigDecimal.valueOf

BigDecimal010. BigDecimal.valueOfab。

```

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

```

## BigDecimals

BigDecimal0,110。

- `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/zh-TW/java/topic/1667/bigdecimal>

---

## 5: C ++

JavaC ++。 JavaC ++。

---

---

## C ++

[\[ref\]](#)

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

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

## Java

[]

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

---

## C ++

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

## Java

[]

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

```
    ...  
  }  
}
```

---

## C ++

[]

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

## Lambda

## Java

[]

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

---

## Overriding vs OverloadingC ++Java

- ◦
- ◦
- ◦ [“Java”Stackoverflow - 1 ; 2](#)

---

◦

- 
- SquareCircle
- Squarearea

C ++◦ Java◦

---





## C++

Java finalize C++ GC - "close".

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

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

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

C++	Java
-	
-	
-	
class defaultprivate; struct defaultpublic	
-	

# C ++

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

BCADBC。 ABCDDBC

C ++JavaJava 8。 JavaJava“”。 。

## java.lang.Object

JavaObject。 JavaObject。

C ++“”。

## Java CollectionsC ++

Java CollectionsC ++ Containers。

## Java

## C ++

			C ++ LLP64LP64	Java
8	-28-1= -128	28-1-1 = 127		
8	0	28-1 = 255		-
16	-216-1= - 32,768	216-1-1 = 32,767		
16	0\ u0000	216-1 = 65,535\ uFFFF		char
32	-232-1= -2.147	232-1-1 = 21.47	INT	INT
32	0	232-1 = 42.95	unsigned int	-
64	-264-1	216-1-1	*	

			C ++ LLP64LP64	Java
64	0	216-1	* unsigned long long	-

\* Win64 API32

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;
};
};
```

## Java

[]

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

---

## C ++

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

## Java

[]

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

---

## C ++

[]

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

# Java



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

Java. C ++Java pass-by-value aka copy pass-by-reference aka alias.

---

## 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

"" - . .

## 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;

```

## Java

```

abstract void draw();

```

## C ++

;

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

## Java

;

```
abstract class GraphicObject {}
```

---

## C ++

*Java*

## Java

1;2

```
interface TestInterface {}
```

**C ++** <https://riptutorial.com/zh-TW/java/topic/10849/c-plusplus>

# 6: Classpath

Java◦ Java◦

## Java

JVMJava◦ -

1. JavaJava◦
2. jre/lib/ext/
- 3.

java.lang.ClassLoader◦ ◦

JVM◦ CLASSPATH

## Examples

◦

1. CLASSPATH

```
set CLASSPATH=...          # Windows and csh
export CLASSPATH=...       # Unix ksh/bash
```

2. 

```
java -classpath ...
javac -classpath ...
```

-classpath -cp CLASSPATH◦

3. MANIFEST.MFClass-PathJARClass-Path

```
Class-Path: jar1-name jar2-name directory-name/jar3-name
```

## JAR

```
java -jar some.jar ...
```

JARClass-Path-classpathCLASSPATH◦

java -jarJAR◦

- <https://docs.oracle.com/javase/tutorial/deployment/jar/downman.html>
- <http://docs.oracle.com/javase/7/docs/technotes/tools/windows/classpath.html>

## JAR



## JARclasspath;

```
someFolder/*
```

JVMsomeFolderJARZIP。 -cp CLASSPATHJARClass-Path。 ◦

1. ClasspathJava 6。 Java“\*”。
2. “ ”;“someFolder/.jar”。
3. “.jar”“JAR”。 ZIPJAR。
4. JARJAR。
5. JAR。

## JARZIPJAR / ZIP。

- -classpath ;;Windows:LinuxUNIXMacOSX。
- JARMANIFEST.MFClass-Path。
- shell。

```
export CLASSPATH="/home/user/My JAR Files/foo.jar:second.jar"
```

shell。

- JAR“MANIFEST.MF”URL。

```
Class-Path: /home/user/My%20JAR%20Files/foo.jar second.jar
```

JARJAR。 main()。 main()。 ◦

SWTJARJARmain() [Java SWT](#)

JARKeyStore。 ClassClassLoader。 ◦

```
program.jar
|
\ -com
  \ -project
    |
    | -file.txt
    \ -Test.class
```

Testfile.txt。

```
InputStream is = Test.class.getClassLoader().getResourceAsStream("com/project/file.txt");
```

◦

## Test

```
InputStream is = Test.class.getResourceAsStream("file.txt");
```

◦ Test.class.com.projectfile.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 anon	SomeClass.java	SomeClass\$1.class SomeClass\$2.class

## JVM

◦ JARZIP

1. RP

2. E

- ◦ ERPAP ◦
  - AP◦
  -
- JARZIP
  - JAR / ZIPLookup RP ◦
  - JAR / ZIP◦

◦ ◦ Class.getResourceClass.getResourceAsStream◦

## Java

Java “rt.jar” JREJAR. Java SE

◦ java javac◦

Java “show stopper”. JARJVM

java-X

- -Xbootclasspath:<path>◦
- -Xbootclasspath/a:<path>◦
- -Xbootclasspath/p:<path>◦

bootclasspathJavaJava◦ ◦ Java.....◦

**Classpath** <https://riptutorial.com/zh-TW/java/topic/3720/classpath>

# 7: CompletableFuture

CompletableFutureJava SE 8Java SE 5Future。 Future。

## Examples

◦ ◦ ◦

```
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);
}
```

lamdathenAccept◦ exceptionallyhandle◦

```
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◦

```

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!
    });
}

```

**CompletableFuture** <https://riptutorial.com/zh-TW/java/topic/10935/completablefuture>

---

## 8: EnumSet

Java EnumSetSet。 AbstractSetSet。

### 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/zh-TW/java/topic/10159/enumset>

# 9: ExecutorService Thread

Java Executor. Executors.

- ScheduledExecutorService. [Mother F \\*\\* k ScheduledExecutorService](#)

## Examples

-

java.lang.Runnable Thread.

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

o

Java 8 lambda.

Java SE 8

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

## ThreadPoolExecutor

Executor ThreadPoolExecutor. Pool.

```
ThreadPoolExecutor pool = new ThreadPoolExecutor(
    1, // keep at least one thread ready,
    5, // even if no Runnables are executed
    5, // at most five Runnables/Threads
    1, TimeUnit.MINUTES, // executed in parallel
    // 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 maximumPoolSize ◦

1. BlockingQueue ◦ ◦

2. ◦

1. ThreadPoolExecutor.AbortPolicy RejectedExecutionException ◦

2. ThreadPoolExecutor.CallerRunsPolicy **execute** ◦ ◦

3. ThreadPoolExecutor.DiscardPolicy ◦

4. ThreadPoolExecutor.DiscardOldestPolicy ◦

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()

• ◦

```
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 **API4** ScheduledFuture ◦ ◦

◦

```

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

```

◦

```

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

```

pool future ◦

scheduledAtFixedRate ◦ ◦

◦

```
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) ◦

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)
```

**submitvs execute**

executefireforgetssubmitFuture◦

◦

submit◦

**1executeRunnable◦**

```

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)
```

**2submitexecute** service.submit(new Runnable(){run◦

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

**3newFixedThreadPoolExtendedExecutor**

```
//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
```

ThreadPoolExecutorThreadPoolExecutorExeption◦

ExecutorServicesubmitFutureFutureFuturegetAPI◦ afterExecute◦ ThreadPoolExecutor - ThreadPoolExecutor◦

### 1. ExecutorService

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

◦ ThreadPoolExecutor◦

Callable/Runnable◦ CPU/MemoryThreadPoolExecutorRejectedExecutionHandler◦

### 2. CountdownLatch

CountDownLatch◦ countDown()◦ await()◦ await()◦ *java*◦

1.

2. N

3. ◦

### 3. ThreadPoolExecutor◦ Runnable / Callable◦ RejectionHandler◦ Java

RejectedExecutionHandler ◦

1. ThreadPoolExecutor.AbortPolicy RejectedExecutionException◦

2. ThreadPoolExecutor.CallerRunsPolicy`execute◦ ◦

3. `ThreadPoolExecutor.DiscardPolicy` ◦

4. `ThreadPoolExecutor.DiscardOldestPolicy` ◦

`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** ◦

◦ ◦

## ExecutorService

### Executor

1. `ExecutorService` `invokeAll()`

**Futures** ◦

```
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** ◦ `countDown()` `await()` ◦ - ◦ **CyclicBarrier** ◦

## 3. Executors ForkJoinPool `newWorkStealingPool()`

## 4. ExecutorServiceFuture

## 5. ExecutorService oracle

```

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() : ◦`

```

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

```

```

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

```

}

## ExecutorService

## Executors Thread Pools

1. `public static ExecutorService newSingleThreadExecutor()`

### Executor

`newFixedThreadPool(1) newSingleThreadExecutor()` [java doc](#)

### newFixedThreadPool1

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

1. ◦

2.

1.

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

◦ **nThreads**

1. ◦ `nThreadsRuntime.getRuntime().availableProcessors()`

2.

1. ◦

3. `public static ExecutorService newCachedThreadPool()`

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. CustomThreadFactory (public Thread newThread(Runnable r)

ExecutorService

submit	<b>future</b>
execute	
invokeAll	<b>Futures</b>
invokeAny	

shutdown() ◦ ◦ awaitTerminationisShutdown() ◦

**ExecutorServiceThread** <https://riptutorial.com/zh-TW/java/topic/143/executor-executorservicethread>



# 10: FileUploadAWS

Spring rest APIAWS s3.

## Examples

s3

APIjava rest APIS3.

- 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);  
});
```

**FileUploadAWS** <https://riptutorial.com/zh-TW/java/topic/10589/fileuploadaws>

# 11: FTP

- FTPClientInetAddressInt
- FTPClientString usernameString password
- FTPClient disconnect
- FTPReply getReplyStrings
- boolean storeFileString remoteInputStream local
- OutputStream storeFileStreamString remote
- boolean setFileTypeint fileType
- boolean completePendingCommand

	FTPIP
	FTP
	FTP
	FTP

## Examples

### FTP

JavaFTP 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.println("Operation failed. Server reply code: " + replyCode)
    return;
}
ftp.login(user, pass);
} catch {
}
}
```

◦

```
showServerReply(ftp);
```

◦

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

/◦

```
if (!FTPReply.isPositiveCompletion(replyCode)) {
    System.out.println("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");
}
```

;"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();
}
```

“。 ”。 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();
    }
}
}

```

FTPJava。

FTP <https://riptutorial.com/zh-TW/java/topic/5228/ftp-->



# 12: HttpURLConnection

- [Android HttpURLConnectionInternet](#) `AndroidManifest.xml` ◦
- [Java HTTPSquareOkHttp](#) ◦

## Examples

### URL

```
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();
}
```

### URLString◦

- `new URL(url).openConnection()` **URL** `HttpURLConnection` ◦ `URLConnection` `HttpURLConnection` ◦ ◦
- `InputStream`
- `BufferedReader` `InputStream`◦
- `StringBuilder` ◦
- `InputStream` **String**◦
- `IOException` **URL** `MalformedURLException` ◦
-

## URLHTMLJSONXMLREST API。

- [JavaStringURL](#) 。

```
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();  
    }  
}
```

## POSTURL。

- URLHttpURLConnection 。
- 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。

---

- URLURLConnection。
- setRequestProperty(application/x-www-form-urlencoded)
- setDoInput(true) URL。
- setRequestMethod("DELETE") HTTP 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;
}

```

## HEADGET。。

200 true。 3XX。

```

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

```

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

[HttpURLConnection](https://riptutorial.com/zh-TW/java/topic/156/httpURLConnection) <https://riptutorial.com/zh-TW/java/topic/156/httpURLConnection>

# 13: InputStreamsOutputStreams

- int readbyte [] bIOException

InputStream BufferedStream◦ read InputStream◦ CPU◦

## Examples

### InputStreamString

◦ bytechar "Java"UTF-16◦ [InputStreamReader](#)◦

""Input◦

### 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

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
}
```

◦

## 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);
```

/

OutputStreamInputStream ◦ ◦

◦

```
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)));
```

/

<b>BufferedOutputStream / BufferedInputStream</b>	OutputStreamBufferedOutputStream ◦ ◦
<b>DeflaterOutputStream / DeflaterInputStream</b>	◦
<b>InflaterOutputStream / InflaterInputStream</b>	◦
<b>CipherOutputStream / CipherInputStream</b>	/◦
<b>DigestOutputStream / DigestInputStream</b>	◦
<b>CheckedOutputStream / CheckedInputStream</b>	Checksum ◦ CheckSumMessage Digest ◦
<b>DataOutputStream / DataInputStream</b>	◦ ◦ ◦
<b>PrintStream</b>	◦ ◦ ◦
<b>OutputStreamWriter</b>	OutputStreamWriter ◦ OutputStreamWriters
<b>PrintWriter</b>	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+"-");
        }
    }
}
```

[InputStreamsOutputStreams https://riptutorial.com/zh-TW/java/topic/1110/inputstreams-outputstreams](https://riptutorial.com/zh-TW/java/topic/1110/inputstreams-outputstreams)



---

# 14: Java Native Access

## Examples

### JNA

---

## JNA

Java Native Access JNA Java Windows.dll Unix.so.....

- [JNA CLASSPATH jna.jar](#)
- Java

*Windows* "C" "msvcrt"

JavaC 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 Native Access <https://riptutorial.com/zh-TW/java/topic/5244/java-native-access>

# 15: Java Native Interface

JNIEnv	JNI
jobject	static native
JCLASS	static native

JNIJava IDEOS Eclipse

WindowsJava-C ++

- javacJava .java .class
- javahnativeJava .h
- nativeC ++ .cpp #include
- C ++ .dll
- -Djava.library.path Java System.loadLibrary(...)

Java javap -s

## Examples

JavaC ++

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 **build**

```
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;
}

```

JavaC ++'printString'

C ++'average'3.4

## C ++Java

### Java

1. GetMethodID JNI;
2. 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 ++int17**

.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 ◦
```

```
java.library.path-Djava.library.path= JVM
```

```
java -Djava.library.path=path/to/lib/:path/to/other/lib MainClassWithNativeMethods
```

**Windows; : ◦**

```
System.loadLibraryLinuxlibExample.soWindowsExample.dll ◦
```

```
System.loadLibrarySystem.load(String) java.library.path
```

```
public class ClassWithNativeMethods {  
    static {  
        System.load("/path/to/lib/libExample.so");  
    }  
  
    ...  
}
```

**Java Native Interface** <https://riptutorial.com/zh-TW/java/topic/168/java-native-interface>

# 16: Java SE 7

Java SE 7 Java JDBC Java JVM

Java SE 7

## Examples

### Java SE 7

- [byte short int long](#) 0b0B
- [switch](#) [switchString](#)
- [try-with-resources](#) [try-with-resource](#)try [try-with-resources](#) [java.lang.AutoCloseable](#) [java.io.Closeable](#)
- [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;
```

### try-with-resources

◦ [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](#) ◦

[BufferedReader](#) [try-with-resource](#) [try](#) [BufferedReader](#).readLine() [IOException](#) ◦

```
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;
```

;

- 
- 
- FL
- 

```
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 getDayOfWeekType(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/zh-TW/java/topic/8272/java-se-7>

---

# 17: Java SE 8

Java SE 8 [Java](#) [JDBC](#) [Java JVM](#)

[Java SE 8](#)

## Examples

### Java SE 8

- [Lambda Expressions](#)
  - [Lambda](#)
    - [lambda](#)
    - [Java SE 8 Lambda API](#)
- - [Java](#) [Java](#) [Java SE 7](#) [Java SE 8](#)
  - [Lambda](#)
  - [Date-time-api](#)
- `java.lang.reflect.Executable.getParameters` [MethodConstructorExecutable](#)  
`Executable.getParameters` `.class` `.class` [Reflection API](#) `javac-parameters`
- [Date-time-api](#) - `java.time` [api](#)

[Java SE 8](#) <https://riptutorial.com/zh-TW/java/topic/8267/java-se-8>

---

# 18: java.util.Objects

## Examples

null

---

## null check in

```
Object nullableObject = methodReturnObject();
if (Objects.isNull(nullableObject)) {
    return;
}
```

---

## null

```
Object nullableObject = methodReturnObject();
if (Objects.nonNull(nullableObject)) {
    return;
}
```

## apiObjects.nonNull

```
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/zh-TW/java/topic/5768/java-util-objects) <https://riptutorial.com/zh-TW/java/topic/5768/java-util-objects>

# 19: JavaBean

JavaBeans™ Java API bean Java。 getter setter API。

- **JavaBean**
- getter get。 getSize “size” JavaBeans getter。 size。 getter setter。 getSize。
- getter get is。 get Stopped is Stopped JavaBeans。
- setter。 set Size size Java Bean。
- getter setter get is set。
- Setter public void。
- Getter public setter。
- **JavaBean**
- “” add。 addAction Listener 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;  
    }  
}
```

JavaBean <https://riptutorial.com/zh-TW/java/topic/8157/javabean>

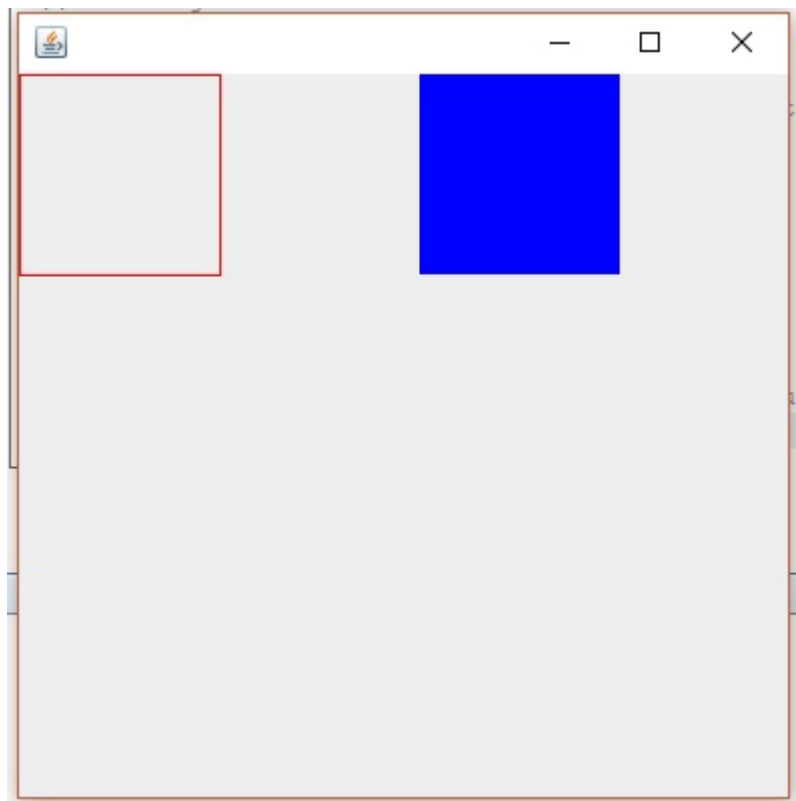
# 20: Java2D

◦ ◦ ◦

Java 2D API ◦ Java2D ◦

## Examples

### 1Java



◦ <https://i.stack.imgur.com/dlC5v.jpg>

### Graphics

- 1.
- 2.

◦ MyPanelTest ◦ MyPanel ◦ drawRect ◦ fillRect ◦ methods ◦ Color ◦ setColor ◦ Color.blue ◦ Test Class ◦ JFrame  
MyPanel p = new MyPanel ◦ Test Class ◦ RectangleBlue ◦ 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
}
}

```

```

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

- 
- 
- 

## Graphic2DGraphic2D Class

### 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 xint, int yint, int height, int width);**

xy。

**g2.fillOval(int xint, int yint, int height, int width);**xy。

**Java2D** <https://riptutorial.com/zh-TW/java/topic/10127/java2d>



---

# 21: JavaJSON

JSONJavaScript Object Notation。 JSON。 JSONAjaxRESTful Web。 [JSON ProcessingJava API](#)  
APIJSON。

[Google Gson](#)Jackson Object MapperJavaJSON。

[JavaJSON](#)

## 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

```
String json =
"{\"foo\": \"bar\", \"temperature\": 21.5, \"year\": 2016, \"message\": {\"Hello\": \"world\"}, \"months\": [\"J

// 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 vs getXXX

JSONObjectJSONArray°

```
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"
```

JSONArraygetXXX / optXXX°

## JSONGson

Personname

```
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 jarclasspath](#)◦

## JSONGson

Personname

```
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

Pojo

```
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
```

```
String jsonString = objectMapper.writeValueAsString(inputObject);
//result: {"firstName":"John","lastName":"Doe","age":23}
```

```
import com.fasterxml.jackson.databind.ObjectMapper;
```

## Mavenjackson-databind

### ObjectMapper

```
//creating one
ObjectMapper objectMapper = new ObjectMapper();
```

- ObjectMapper
- 

```
<T> T readValue(String content, Class<T> valueType)
```

- valueType -
  - IOException - I/O
  - JsonParseException -
  - JsonMappingException - JSON

### jsonString

```
Model fromJson = objectMapper.readValue(jsonString, Model.class);
```

### String writeValueAsStringObject value

- - JsonProcessingException
  - 2.1throwsIOException; 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 -

JSONObjectJSONArray ◦

### 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 **final** JSONObject.NULL **Java** null ◦

JSONObject.NULL **sentinel** ◦

```
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\(\)](#)

xx.equalsnullfalse

## JSONArrayJava ListGson Library

JSONArrayJava ArrayList

```
{
  "list": [
    "Test_String_1",
```

```
        "Test_String_2"
    ]
}
```

JSONArray "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;
}
```

mavenPOM.xml

```
<!-- https://mvnrepository.com/artifact/com.google.code.gson/gson -->
<dependency>
    <groupId>com.google.code.gson</groupId>
    <artifactId>gson</artifactId>
    <version>2.7</version>
</dependency>
```

jar com.google.code.gson:gson:jar:<version> ◦

## JacksonJSON

pojoPerson

```
public class Person {
    public String name;

    public Person(String name) {
        this.name = name;
    }
}
```

JSONPerson ◦ List<Person>Map<String, Person> JSON ◦ jacksonTypeFactoryTypeReference ◦

### TypeFactory

◦ ◦

### TypeReference

◦ TypeReferenceList<Person> ◦ TypeReference ◦

JSONJava ◦ JSONList ◦ Map ◦ ◦

---

# JSON

```
String jsonString = "[{\"name\": \"Alice\"}, {\"name\": \"Bob\"}]"
```

## TypeFactory

```
CollectionType listType =  
    factory.constructCollectionType(List.class, Person.class);  
List<Person> list = mapper.readValue(jsonString, listType);
```

## TypeReference

```
TypeReference<Person> listType = new TypeReference<List<Person>>() {};  
List<Person> list = mapper.readValue(jsonString, listType);
```

---

# JSON

```
String jsonString = "{\"0\": {\"name\": \"Alice\"}, \"1\": {\"name\": \"Bob\"}}"
```

## TypeFactory

```
CollectionType mapType =  
    factory.constructMapLikeType(Map.class, String.class, Person.class);  
List<Person> list = mapper.readValue(jsonString, mapType);
```

## TypeReference

```
TypeReference<Person> mapType = new TypeReference<Map<String, Person>>() {};  
Map<String, Person> list = mapper.readValue(jsonString, mapType);
```

---

```
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.

o





# 22: Java

## Examples

### Manifest.mf

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

JavaInstrumentation。 Instrumentation*addTransformerClassFileTransformer*。 ClassFileTransformers。 ClassLoaderjava.lang.ClassProtectionDomain。

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

。 。

### ASMJavassistBCEL。

```
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。 /。 ASMTree API。

### ClassNode*transform*

```
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.attachString 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);
}
}

```

*premainString agentArgsInstrumentation inst agentmainString agentArgsInstrumentation inst*。  
**Manifest.mfAgent-Class**。

Premain“*premainString agentArgs Instrumentation inst*”

```

import java.lang.instrument.Instrumentation;

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

```

**jarManifestPremain-Class**。

```
Premain-Class: PremainExample
```

Java“myProgram”JVM

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

Java <https://riptutorial.com/zh-TW/java/topic/1265/java>

# 23: Java

JavaJLS. “JLS 17.4” [Java 8](#) [Java 7](#) [Java 6](#)

Java 5 `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().doIt()` `JavaJLS`.

```
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, NN, N + 1
- N + 1, N
- 0, 0<sup>1</sup>

1 - println

JIT doIt() JLS doIt() JIT

o o o o JLS

o o o o

o o JLSJava JIT

JLSJIT

Java

- synchronized
- volatile
- ;java.util.concurrent

o Java Memory Model

JavaJLS o "" JIT

Java

Java

- o JLS [JLS 17.4.5](#)

“ o ”

[JLS 17.4.2](#) 5

- o
- o
- o o
  - o volatile
  - o o
  - o o o

◦ ◦

◦ ◦

• ◦ ◦

• ◦ ◦

### JLS 17.4.3 JLS 17.4.4 Java

◦

• ◦

• volatile ◦

• Thread.start() run() ◦

• ◦ JLS ◦

• ◦ join() true isTerminated() ◦

• ◦

### JLS 17.4.5 ◦

write(v) read(v) vvv ◦ read(v) ◦

• **1** - xyxyx y ◦

• **2** - ◦

• **3** - x y x y ◦

• **4** - x - yy - 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)
    }
}
```

- 1

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

- 4

4. write(a) - write(b) write(b)write(b) - read(a) read(a) **IMPLIES** write(a) read(a) ◦
5. write(b) - read(a) read(a)read(a) - read(b) read(b) **IMPLIES** write(b) - read(b) ◦
6. write(a) read(a) a + ba ◦
7. write(b) - read(b) a + bb ◦

## 2“volatile”

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() ◦

- 1

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

- 2

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

- 4

4. write(b) read(b) ◦

b ◦ updateobserve()ba ◦ volatile-read(a)volatile-write(a) ◦

observe()b ◦

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

◦ ◦

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) ◦ b ◦ b3 ◦

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) *before-before* write(b, 1) ◦ read(b) ◦

volatile ◦

◦ ◦

Java ◦

- ◦ ◦
- “” ◦
- Lock<sup>1</sup> ◦
- Executor / ExecutorService fork join ◦
- `java.util.concurrentwait / notify / notifyAll ◦
- java.util.concurrentmapsetslistsqueuesdeques ◦

Java“” ◦ ◦

1 - . .

Java <https://riptutorial.com/zh-TW/java/topic/6829/java>



# 24: Java

Java◦ Java◦

Java◦ ◦

## Examples

Javafinalize◦ Java◦

```
public class MyClass {  
  
    //Methods for the class  
  
    @Override  
    protected void finalize() throws Throwable {  
        // Cleanup code  
    }  
}
```

Java◦

- Javafinalize()◦
- Javafinalize()◦
- ...◦

finalize◦ ◦

◦

◦ ◦ 1

```
public class CaptainJack {  
    public static CaptainJack notDeadYet = null;  
  
    protected void finalize() {  
        // Resurrection!  
        notDeadYet = this;  
    }  
}
```

CaptainJack finalize()notDeadYet◦ ◦

◦

JVM◦ notDeadYetnull notDeadYetfinalize() ◦

1 - [https://en.wikipedia.org/wiki/Jack\\_Harkness](https://en.wikipedia.org/wiki/Jack_Harkness) ◦

## GC

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

Java. ""JVMJava.

◦ -XX:+DisableExplicitGCJVMSystem.gc()◦ System.gc()JVM/◦

## C ++ -

C ++◦ newC ++delete

- delete""◦ ◦
- delete

C ++newdelete◦ ◦

## Java -

Java◦ Javadelete◦ Java◦ GC◦

Java<sup>1</sup>

- JLS

◦

static◦

- ◦

◦ ◦

- 1◦

- ◦

JavaJVM◦ JVM◦

JLS◦

◦ ◦ GC◦

GC◦

1. Reference◦

2. ◦ ◦

### 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.println("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 null n2 ◦ Node2 Node2 ◦

T6 null n3 ◦ Node3 Node3 ◦ Node1 Node2 n1 ◦

test() n1 n2 n3 ◦ Node1 Node2 Node ◦

---

1 - finalization Reference ◦ 2 - Java ◦

## PermGen

Java ◦ java ◦ Java 8 Java PermGen ◦

Java 8 PermGen PermGen ◦

JVM ◦ ◦ ◦

## Heap PermGen

JVM

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

<size>k mg ° °

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

-XX:+printFlagsFinalJVM °

- LinuxUnixSolarisMac OSX

```
$ java -XX+ PrintFlagsFinal -version | grep -iE'HapSize | 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

GarbageJava ° ° 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;
    }
}

```

pushpopstack◦

API. ""◦ 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◦ ◦ ◦

◦ ◦ WeakHashMapWeakHashMapJVM◦

Java <https://riptutorial.com/zh-TW/java/topic/2804/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`

LinuxMac OSXUNIX `java`

`<opt>` `java` [“Java”](#) [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`

[“Java](#)

`JAR` `java`

## HelloWorld

`Java` [“HelloWorld”](#) `HelloWorld`

[“HelloWorld.class”](#)

```
java HelloWorld
```

- “.class”“.java”。
- **Java**java° com.exampleSomeClass com.example.SomeClass °

java -jarjava;Classpath ° ° -cp°

```
java -cp . HelloWorld
```

“。 ”。

-cpjava° java**classname**° **Java**mainString[]°

-cpjavaCLASSPATH° java“。 ”。

**Java**main

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

```
(String args[])
```

javamain ° **Java** String° main nullnull°

- main
- publicstatic
- void
- String[]° °
- °
- 

public° **Java 5** mainString varargs° mainargs °

## JavaFX

Java 8 javaJavaFX° JavaFXJavaFXJavaFX

- javafx.application.Application
- publicabstract
- 
- public no-args

'java'

'java'。

“”

```
java: command not found
```

javashelljava°

- Java JREJDK
- shellPATH
- “”shell°

“Java”°

““”

java/°

- °
- °
- Java°

1. °

- ° “Main”“com.example.myapplication”“com.example.myapplication.Main”°
- javap°
- °
- JARZIPJARZIP°

2. java° java°

- °
- “.java”“.class”°
- Java<sup>1</sup>°
- °

3.

- ;
- ;
- JARZIP°
- °

-Xdiag-XshowSettings° JVM°

- Main-ClassJAR°
- Class-PathJAR°
- 2<sub>java</sub>°
- Java°
- °

“<name>”

java°



- JARJAR“Main-Class”。
- java。
- ;。
- “”
- <http://docs.oracle.com/javase/tutorial/getStarted/problems/index.html>

1 - Java 8 java“/”“ ”。 。

2 - “”。

## Java

““JAR”。

Java。 JAR。

Java。 JavaJVM。

- JAR。
- JVMJAR。

JAR“Class-Path”。 `jar`。 `-cpCLASSPATH`。

“myApp.jar”Java`com.example.MyApp`。 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;:setexportset CLASSPATH。

Java“” shellWindows。 shell“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 ◦ Javashell ◦

```
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
```

shell/home/steve/Test File.txt ◦ **Java2** ◦

## POSIX shell

POSIX shellshbashksh ◦ shell ◦

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

shell ◦ ◦

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

◦

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

◦

**Bash** ◦

## Windows

Windows ◦ ◦ ◦

Java

- java ◦
- java

- SET°
- cmd.exe shell;^°

°

## Java

java

- - GNU / Linux--“long”°
- <classname>-jar <jarfile>° Java°
- -X-XX° Java<sup>1</sup>°
- -XJavaJava°
- -XX°

-D

-D<property>=<value>PropertiesProperties ° °

## PermGen ° °

-ea-daJava assert

- °
- -ea
- -ea:<packagename>...°
- -ea:<classname>...-ea:<classname>...°
- -da
- -da:<packagename>...°
- -da:<classname>...-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 “” JIT◦

JVM64◦ -d32-d64◦

---

1 - java◦ “”◦

Java - 'java'javaw' <https://riptutorial.com/zh-TW/java/topic/5791/java----java--javaw->

# 26: Java

- 
- 

Internet -

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 basically 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 connction
            // 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);
    }
}
}

```

#### telnet◦

#### IP◦

```
Welcome!
```

Connection to host lost.

◦ TCP ◦

◦

```
public class CAPECHOCClient extends Thread{

Socket server;
Scanner key; // Scanner for input

    public CAPECHOCClient(String ip, int port){
        try {
            server = new Socket(ip, port);
            key = new Scanner(System.in);
        } catch (IOException ex) {
            Logger.getLogger(CAPECHOCClient.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(CAPECHOCClient.class.getName()).log(Level.SEVERE, null, ex);
        } finally {
            try {
                istream.close();
                ostream.close();
                server.close();
            } catch (IOException ex) {
                Logger.getLogger(CAPECHOCClient.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/zh-TW/java/topic/9923/java>



# 27: Java

## Examples

Java

- 
- 
- 

JVM VisualVM JDK

Java String GC

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

“ 97 3 “

sage

1.

2. ◦

3. ◦ ◦

4. ◦

5. ◦ “”◦

6. ◦ ◦ ◦

7. ◦

8. “”;◦

9. ◦

10. ◦

11. ◦
- 4◦
  - 9◦

◦ ◦ 1500.5◦

◦ ◦

- ◦
- ◦
- CPU◦
- ◦

◦

Java <https://riptutorial.com/zh-TW/java/topic/4160/java>

# 28: Java

Java Print Service API。

IETF RFC 2911 Internet 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();
```

DocPrintJobprint

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

docDoc ◦

prasPrintRequestAttributeSetPrintRequestAttribute ◦

- 1,2
- 
- 
- 
- 
- ...

printPrintException ◦

DocJava Print Service APISimpleDoc◦

Doc

- 
- DocFlavor **MIME+**◦

Doc◦

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

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 pje

PrintJobEvent pje° °

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

PrintJobAttributeSet°

---

PrintJobAdapterPrintJobListener° ° °

```
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**print**

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

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

Java <https://riptutorial.com/zh-TW/java/topic/10178/java>

# 29: Java

IDE/。 APIAPI。 - 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);
    }
}

```

## *MainApplicationHashSet*

```

static HashSet<Plugin> plugins=new HashSet<>();

```

## *URLClassLoader*

```

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() //overridden to show user's home directory
    {
        System.out.println("User home directory: "+System.getProperty("user.home"));
    }
}
```

Jars - IDE◦

Jars"" *MainApplication*◦

Java <https://riptutorial.com/zh-TW/java/topic/7160/java>

# 30: 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
    }
}
```

Java float double ◦ float 238 ◦ double 5215 ◦ ◦ delta ◦

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

## 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, 0.000000001, 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 == 1.00000001 ? true
1.00000001 == 1.0000000001 ? true
1.0000000001 == 1.00000000000001 ? true

```

```

delta: 1.0E-5
1.0 == 1.0001 ? false
1.0001 == 1.00000001 ? false
1.00000001 == 1.0000000001 ? true
1.0000000001 == 1.00000000000001 ? true

```

```

delta: 1.0E-7
1.0 == 1.0001 ? false
1.0001 == 1.00000001 ? false
1.00000001 == 1.0000000001 ? true
1.0000000001 == 1.00000000000001 ? true

```

```

delta: 1.0E-10
1.0 == 1.0001 ? false
1.0001 == 1.00000001 ? false
1.00000001 == 1.0000000001 ? false
1.0000000001 == 1.00000000000001 ? false

```

```

delta: 0.0
1.0 == 1.0001 ? false
1.0001 == 1.00000001 ? false
1.00000001 == 1.0000000001 ? false
1.0000000001 == 1.00000000000001 ? false

```

doublefloatcompare°

```

double a = 1.0;
double b = 1.0001;

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

```

- **delta** ◦ ◦

## OverflowUnderFlow

### 32IEEE 754

Float

3.4028235e+38 Infinity

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

Float **UnderFlow**

### 1.4e-45f0.0

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

### 64-bit IEEE 754

Double **Overflow**

1.7976931348623157e+308 Infinity

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

Double **UnderFlow**

### 4.9e-3240.0

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

'f'String.format

```
//Two digits in fractional 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

```
// 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

floatdoublefloat IEEE 754. ;°

strictfp ° strictfp floatdouble. 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 reproducible.
    }

    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/zh-TW/java/topic/6167/java>

---

# 31: Java

## Examples

Java SE JREJava SE JDK

Java SESun / OracleJREJDK。 JREJavaJDKJava。

## Java

Java Runtime EnvironmentJREJava。 JRE

- JVMJavaJava<sub>java</sub>
- Nashorn Javascript<sub>js</sub>。
- Java<sub>keytool</sub>。
- <sub>policytool</sub>。
- <sub>pack200unpack200unpack200</sub>“pack200”Web。
- <sub>orbd rmid rmiregistrynameserv</sub>Java CORBARMl。

“JRE”WebJava。 “Server JRE”installers.linux

Java 76JREJavaFX2.2。

## Java

Java Development KitJDKJREJava。

- <sub>javac</sub>Java“。 java”“.class”。
- <sub>jarjarsigner</sub>JAR
- - <sub>applet</sub>appletviewer
  - <sub>idlj</sub> CORBA IDLJava
  - <sub>javah</sub> JNI
  - <sub>native2ascii</sub>Java
  - <sub>schemagen</sub> JavaXMLJAXB
  - <sub>serialver</sub>Java Object Serialization。
  - <sub>JAX-WS</sub><sub>wsgenwsimport</sub>
- - <sub>jdb</sub>Java
  - <sub>Java</sub><sub>jmapjhat</sub>。
  - <sub>jstack</sub>。
  - “.class”<sub>javap</sub>。
- - <sub>jconsole</sub>
  - <sub>jstat jstatd jinfojps</sub>

Sun / Oracle JDKZIPJava。 Java 6Java。



Java 6 OpenJDK OpenJDK。 LinuxJDK。

## Oracle Hotspot OpenJDK

JRE JDK Java

- Oracle Hotspot Oracle。
- OpenJDK OpenJDK。

Hotspot OpenJDK。 Hotspot“”Oracle Java Hotspot OpenJDK。

Hotspot OpenJDK Hotspot。 OpenJDK; Linux QA OpenJDK。

Linux Hotspot。 Hotspot Linux Java。

## Java EE Java SE Java ME Java FX

Java。 Java。 Java Java。

Java。 Java。

---

# Java

Java

- Java Java SE
- Java Java EE
- Java Micro Edition Java ME
- Java FX

Java Java VM API。 Java 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 API Java SE API Java EE

---

# Java FX

Java FX Script Internet Java FX Script Java VM Java EE

---

- [Oracle](#)

## Java SE

## Java SE

Java SE

Java SE <sup>1</sup>		2
<a href="#">Java SE 9</a>		2017-07-27
<a href="#">Java SE 8</a>		2014318
<a href="#">Java SE 7</a>	2015414	2011-07-28
<a href="#">Java SE 6</a>	2013416	2006-12-23
<a href="#">Java SE 5</a>	2009-11-04	2004-10-04
<a href="#">Java SE 1.4.2</a>	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
<a href="#">Java SE 1.3</a>	2009-11-04	200058
Java SE 1.2	2009-11-04	1998-12-08
Java SE 1.1	2009-11-04	1997219
Java SE 1.0	2009-11-04	1996121

1. Oracle. Oracle Java Archives.

2. Java SE™。JavaOracle。。

- [JDK](#) Canadian Mind Products Roedy Green

## Java SE

Java SE	
Java SE 8	Lambda MapReduce Streams。 Nashorn Javascript。 。 。 API。 JNI。 JavaFX。 PermGen。
Java SE 7	<i>try-with-resource</i> <> /。 。 。 Timsort。 ECC。 2DGPU。 。
Java SE 6	JVM Swing。 API Mozilla Rhino Javascript。 JDBC 4.0。 API。 JAXB 2.0。 Web JAX-WS
Java SE 5	enum varargs for。 Java。 Swing RMI。 java.util.concurrent.*Scanner 。
Java SE 1.4	assert。 。 。 NIO API - I / O BufferChannel。 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。 Java JNDI。 JPDA。 JavaSound API。 API。
Java SE 1.2	strictfp。 Swing API。 JavaWeb。 CORBA。 。
Java SE 1.1	。 。 JDBC。 RMI。 Unicode /。 。 AWT。 JavaBeans。

- [Java](#)

Java <https://riptutorial.com/zh-TW/java/topic/8973/java-->

## 32: Java - 'javac'

`javac` `Java` ◦ ◦ `Java` ◦

`javac` `Java Development Kit` `JDK` ◦

`Java` `Java`

- “.java”
- “.class”
- ◦

`javac` `JIT` ◦

## 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” `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
```

shell....

```
$ javac *.java
$ javac com/example/*.java
$ javac */**/*.java #Only works on Zsh or with globstar enabled on your shell
```

“com / example”Java◦ ◦

```
$ javac @sourcefiles
```

sourcefiles

```
Foo.java  
Bar.java  
com/example/HelloWorld.java
```

◦ Java◦ Java IDE [NetBeans](#) [eclipse](#) [IntelliJ IDEA](#) “”◦

## 'javac'

javac

- -d“.class”◦
- -sourcepath◦
- -cp-classpath◦ ◦
- -version◦

◦

javac [Oracle javac](#) ◦

## Java

Java◦

- Java
- JavaAPI◦
- Java◦

enum “”◦

- JavaJavaJava◦
- JavaJavaJava◦

## Java

JavaJava◦ enum-source◦

```
public class OldSyntax {  
    private static int enum; // invalid in Java 5 or later  
}
```

Java 5

```
$ javac -source 1.4 OldSyntax.java
```

JavaJavaJDKJDK◦

## Java

- JavaJava
- Java
- 

-target

```
$ javac -target 1.4 SomeClass.java
```

Java 1.4JVM -source-target javac -source 1.4 ... - --source-targetOracle

-target-source JDK API -bootclasspath

```
$ javac -target 1.4 --bootclasspath path/to/jav1.4/rt.jar SomeClass.java
```

• •

Java - 'javac' <https://riptutorial.com/zh-TW/java/topic/4478/java----javac->

---

# 33: JavaJVM

## Examples

◦

JVMRAM ◦ Java ◦ JavacJavaJavaBytecode

JavaJVM ◦ JDK ◦

*Byte code Machine code*WindowsLinux;◦

-

- - .classRAM ◦
- - ◦
- - ◦
- JIT - JITJVMJVM ◦ java ◦

JavaJVM <https://riptutorial.com/zh-TW/java/topic/8110/java-jvm->



# 34: Java

“JavaWeb” JAR WAREAR

“.class”Java Java

- JARJAR; javac
- WAREAR“servlet”
- JRE
- JARWebJava WebStart

JARWAREAR

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 cd C:\Your\Folder\Path\Here on windows

**Mac** cd /Users/user/Documents/Java/jarfolder

```
jar -cvfm HelloWorld.jar manifest.txt HelloWorld.class Enter .classHelloWorld.jarjar
HelloWorld.class -m-v
HelloWorld.jar
Hello, World
```

## JARWAREAR

JARWAREAR ZIPWAREAR/

Java IDE export

◦ ◦

## MavenJARWAR

MavenJARWAR <packaging>POM;

```
<packaging>jar</packaging>
```

```
<packaging>war</packaging>
```

◦ Mavenmaven jar JAR. “uberJAR”JAR.

Maven <http://www.riptutorial.com/topic/898>

## AntJARWAREAR

AntJARWAREAR Ant <http://www.riptutorial.com/topic/4223>

## IDEJARWAREAR

Java IDE

- Eclipse - <http://www.riptutorial.com/topic/1143>
- NetBeans - <http://www.riptutorial.com/topic/5438>
- IntelliJ-IDEA -

jar

# JARWAREAR。

jar“”。

jar JAR

## Java Web Start

Oracle JavaWeb Start

Java Web Start。 Internet。

Java Web Start。

Java Web StartJavaWSJAWS。

- [Java - Java Web Start](#)
- [Java Web Start](#)
- [Java Web Start](#)
- [JNLP](#)
- [javax.jnlp API](#)
- [Java Web Start Developers Site](#)

Web StartWebJARJava。

- JavaJREJDK。 Java 1.2.2
  - Java 5.0Web StartJRE / JDK。
  - Web Start。
  - Web StartJavascript。
- Web。
- Web Start
  - 
  - Java.....Web。

## 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>

```

## JNLPXML<jnlp>◦

- specJNPL◦
- codebasehref URLURL◦
- hrefJNLPURL◦
- <information>◦
- <resources>JavaOSJAR◦
- <application-desc> <applet-desc> ◦

## Web

Webapplication/x-java-jnlp-file.jnlpMIME .jnlp ◦

WebJNLPJARJNLPURL◦

WebWeb◦

- Java Web Start◦ ◦

```
<a href="https://www.example.com/demo_webstart.jnlp">Launch the application</a>
```

- Java◦

JavaWebJavaJNLP◦

## Web Start

Web Start◦ Java 5.0 JREJDK

```
$ javaws <url>
```

<url>JNLPURL◦

## UberJAR

Java。 Java SEJAR。

。 JARJARJava。 JAR。 。

JARUberJAR。

---

## “jar”UberJAR

UberJAR。 Linux。 Mac OSWindows。

1. 。

```
$ mkdir tempDir  
$ cd tempDir
```

2. JAR jarJAR。

```
$ jar -xf <path/to/file.jar>
```

JAR JAR。

3. \$ cp -r path/to/classes .

4. UberJAR

```
$ jar -cf ../myApplication.jar
```

JARMANIFEST.MF。

---

## MavenUberJAR

Maven“maven-assembly”“maven-shade”UberJAR。 [Maven AssemblyMaven](#)。

---

## UberJAR

UberJAR

- UberJAR。
- UberJAR。

UberJARJAR。 。

## UberJARs

- UberJARUberJAR。
- <sup>1</sup>UberJAR。

---

1 - 。 UberJARs。

Java <https://riptutorial.com/zh-TW/java/topic/6840/java>

# 35: Java - NullsNullPointerException

null◦

NullPointerException **NPE** null◦

- null
- null
- null
- nullsynchronized
- null
- null
- null◦

**NPE**

- 
- 
- nullAPI◦

null

- Map **API**get (key) null ◦
- ClassLoaderClass **API**getResource (path) getResourceAsStream (path) null ◦
- Reference **API**get () null ◦
- Java EE servlet **API**getXxxnull ◦

NPE null “Yoda”◦

## Examples

### - 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<sup>1</sup>◦ new abcdnull ◦

intInteger◦ d nullNullPointerException◦

- cdnull ◦ null◦

- ◦ ◦

◦

---

1 - ◦ ◦ ◦

## - 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 ◦ nullNullPointerException◦ null;◦

◦

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



◦

- “”

## StackOverflowAnswers

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

null “” NullPointerException ◦

◦

joinStrings

“a”“b”

String ◦ null "" ◦

# null

null ◦ ◦ ◦

# null “” “”

null; “” ◦ Java 8 Optional ◦

“”

“” null ◦ NullPointerException bug

“” ◦

# /

“make good” “” null ◦ ◦ “” ◦

null null case ◦ null null null javadoc ◦

null NullPointerException ◦

## - null

### Java

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

}

getReadernullReader ◦ null ◦ NullPointerException ◦

1. IOException◦
2. ◦
3. null "" Reader◦

null

1. NullReader;API◦
2. Java 8getReaderOptional<Reader> ◦

## - I/O

◦ catchtry - catch - finally

```
void writeNullBytesToFile(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.close()outnull NullPointerException

null ◦

```
void writeNullBytesToFile(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();
    }
}
```

try finally NPE.

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

## - “Yoda” NullPointerException

### StackOverflow

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

someStringnull “” NullPointerException.

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

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

◦ ◦

### YodaNullPointerException.

"A".equals(someString) "A".equals(someString) "someStringnull ◦ - “” null.

“” 1 ◦ null NullPointerException /null.

Yoda null API API null ◦ null.

---

1 - “ ”. Yoda. ◦

[Java - Nulls NullPointerException](https://riptutorial.com/zh-TW/java/topic/5680/java----nulls-exception) <https://riptutorial.com/zh-TW/java/topic/5680/java----nulls-exception>

# 36: Java -

Java

Java

## Examples

-

TRACEDEBUG

```
// Processing a request of some kind, logging the parameters
LOG.debug("Request coming from " + myInetAddress.toString()
    + " parameters: " + Arrays.toString(veryLongParamArray));
```

INFO debug

- StringString
- InetAddressDNS
- veryLongParamArray - veryLongParamArrayString

o

```
// No toString() evaluation, no string concatenation if debug is disabled
LOG.debug("Request coming from {} parameters: {}", myInetAddress, parameters);
```

### String.valueOfObject

```
if (LOG.isDebugEnabled()) {
    // Argument expression evaluated only when DEBUG is enabled
    LOG.debug("Request coming from {}, parameters: {}", myInetAddress,
        Arrays.toString(veryLongParamArray));
}
```

LOG.debug(Arrays.toString(Object[])) DEBUG

-

```
public String joinWords(List<String> words) {
    String message = "";
    for (String word : words) {
        message = message + " " + word;
    }
    return message;
}
```

words

```
message = message + " " + word;
```

message message ◦

joinWords **NMONOMN** <sup>2</sup> ◦ **N** <sup>2</sup> ◦

<sup>1</sup>StringBuilder

```
public String joinWords2(List<String> words) {
    StringBuilder message = new StringBuilder();
    for (String word : words) {
        message.append(" ").append(word);
    }
    return message.toString();
}
```

joinWords2 **""**StringBuilder ◦ **OlogNOMN** ◦ toString() ◦

StringBuilder ◦ ◦

joinWords **Java**

```
StringBuilder tmp = new StringBuilder();
tmp.append(message).append(" ").append(word);
message = tmp.toString();
```

**Java**StringBuilder **""**joinWords2joinWords2 ◦

- **"Java'+"**

---

<sup>1</sup> - **Java 8** Joiner ◦ ◦

- **'new'**

**Java**newInteger Boolean ◦ **Java 5**valueOf ◦

```
Integer i1 = new Integer(1); // BAD
Integer i2 = 2; // BEST (autoboxing)
Integer i3 = Integer.valueOf(3); // OK
```

new Integer(int) **JIT** ◦ valueOf **Java**Integer ◦ **""** ◦ **GC** ◦

1. **Java**valueOfBoolean Byte Short Integer LongCharacter ◦
2. **Java** ◦

- **'new StringString'**

new String(String) ◦

-

- 
- `Java String` ◦ `Java 7 String` ◦

`new String(String)`

- CPU ◦
- memory/GC ◦
- `String.equals(Object)` ◦ `hashCode()` ◦

## - System.gc

`System.gc()` ◦

`gc()` javadoc

“`gc` Java ◦ Java ◦ “

1. “” JVM ◦ JVM JVM-XX:+DisableExplicitGC ◦
2. “” `gc` “” ◦

`System.gc()`

◦ `GC` “” ; ◦ `GC` ◦

“” ◦ ◦ ◦ ; ◦ ◦ ◦

◦ ◦

JVMGC ◦ JVMGC ◦

“.....” ◦

1. `//System.gc()` ◦

2. ◦ “” ; ◦

-

```
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
```

intInteger ◦ valueOfInteger ◦ ◦

valueOf ◦ Integer **16** ◦ ab int ◦

---

◦

int [] Integer [] int [] int ◦ Integer [] Integer Integer int ◦

◦ int [] **CPU** ◦ Integer [] **CPU** ◦

---

**CPU** ◦ ◦

## - Map

```
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 ◦ ◦

“ ”

- size ◦

Java Collections Framework **Collection**

- `size()` **Collection**
- `Collection.isEmpty()` **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 ◦ ◦

## PatternMatcher

```
/**
 * 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 ◦ ◦ Matchers ◦ 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](#)

◦ Matcher ◦

## findmatch



s°

```

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()allAlphanumeric

```

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());

```

printlntrue ° false ° ° CA °

° °

(A+)+B° “AB”A°

- 'AB"AB'

- 'AAB"AAB"AAB`
- 'AAAB"AAAB"AAAB or '(A)(AA)B'AAAB`
- 

$2^N N_A$

$K \circ (2^N) \circ (N^K) \circ$

- ◦
- ◦
- ◦
- regex ◦ ◦

API ◦ "" ◦

- [http://www.riptutorial.com/regex/topic/259/getting-started-with-regular-expressions/977/backtracking#t=201610010339131361163http://www.riptutorial.com/// 259 / -/ 4527 /-T = 201610010339593564913](http://www.riptutorial.com/regex/topic/259/getting-started-with-regular-expressions/977/backtracking#t=201610010339131361163http://www.riptutorial.com///259/-/4527/-T=201610010339593564913)
- "" ◦
- Andreas Haufler "Java" ◦

- ==

"=="

== ◦ ==String.equals(...)String.equals(...) ◦

String== ◦ JLS ◦ String.intern(String)Java SE API ◦ String ◦

## 'intern'

String ◦ HashMap ◦ Hashtable ◦ internCPU ◦

==equals ◦ "" "" ◦

◦

CPU ◦

Java 7Java "PermGen" ◦ PermGen ◦ PermGenJVM ◦

Java 7 "PermGen" ◦ ◦ Eden ◦

◦ JVM ◦

◦ ◦

◦ intern ◦

Java 6. Java 6. -XX:StringTableSize Java 6. Java 7 100960013 .

intern Java. intern.

◦ DoS.  $O(N)$  intern ...N.

DoS/. DoSDoS.

-/

```
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.

◦ ◦

```
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(). ◦ ◦

• os. os.write(int). os.

## Java I / O API。

- InputStreamOutputStream I / O API
- ReaderWriter I / O API。

I / O BufferedReaderBufferedReaderBufferedReaderInputStreamBufferedReaderOutputStream。

- JavaJavaJVM。
- JVM。

1. syscall。
2. SYSENTER。
3. 。
4. 。
5. read
  - 1.
  - 2.
  - 3. JVM
  - 4. thstream
6. VM。

。

read() read()byte。

I / O。

byte[]char[]。

## Java

JavaAPI NIO ByteBufferChannel API。

Java - <https://riptutorial.com/zh-TW/java/topic/5455/java--->

# 37: Java -

Java

## Examples

-

“”

“”

IDE

```
try {
    inputStream = new FileInputStream("someFile");
} catch (IOException e) {
    /* add exception handling code here */
}
```

IDE “”;inputStreamNullPointerException

o o o

```
try {
    selfie.show();
} catch (InterruptedException e) {
    // It doesn't matter if showing the selfie is interrupted.
}
```

```
try {
    selfie.show();
} catch (InterruptedException ignored) { }
```

ignored IDE IntelliJ IDE catch

## - ThrowableExceptionErrorRuntimeException

Java “”1

```
....
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 ◦

```
Could not open file null
```

- " " I/O ◦

Exception ◦

- Exception ◦
- RuntimeException ◦
- Error JVM ◦ ◦
- Throwable ◦

- 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 - [Pokemon](#) ◦

## - ThrowableExceptionErrorRuntimeException

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 ◦ somethingBadtrueException “NullPointerException ◦

- /◦
- ExceptionThrowableException throws ◦ ◦

◦ “”◦ ◦

## “”ThrowableException◦

throwsExceptionThrowable◦

1. Exception ◦
2. ◦
3. Exception◦ ◦
4. Throwable◦

◦

```

try {
    doSomething();
} catch (Exception ex) {
    report(ex);
    throw ex;
}

```

◦ **Java 7** throw ex;Exception ◦ throws Exception ◦ **Java 7**◦

## - InterruptedException

```

try {
    // Some code
} catch (Exception) {
    // Some error handling
}

```

◦ ◦

◦

```

Thread t = new Thread(new Runnable() {
    public void run() {
        while (true) {
            //Do something indefinitely
        }
    }
});

t.start();

//Do something else

// The thread should be canceled if it is still active.
// A Better way to solve this is with a shared variable that is tested
// regularly 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** ◦ **ThreadInterruptedException** ◦

```

Thread t = new Thread(new Runnable() {
    public void run() {
        try {
            while (true) {
                //Do something indefinitely
            }
        } 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-allInterruptedException** ◦ **thead**`t.join()` ◦

```

Thread t = new Thread(new Runnable() {
    public void run() {
        try {
            while (true) {
                try {
                    //Do something indefinitely
                }
                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();
    }
}

```

## Exceptions catch-all InterruptedException.

```

Thread t = new Thread(new Runnable() {
    public void run() {
        try {
            while (true) {
                try {
                    //Do something indefinitely
                } 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();
        }
    }
}

```

## Java

“ ”

[http //programmers.stackexchange.com/questions/184654](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 "";
    }
}

```

wordnull. *if ... elsetry ... catch* null.

◦ ◦ Math.minString.substringNullPointerException ◦

◦ **Java 8** ◦

“” “”;

-

printStackTrace()

printStackTrace()printStackTrace()

- **Java**

- ◦

printStackTrace

◦ ◦

```
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

◦ ◦

## - `Throwable`

Throwable ExceptionError ◦ ThrowableExceptionError ◦

Throwable ◦ **Subclassing** Exception ◦

**Java** - <https://riptutorial.com/zh-TW/java/topic/5381/java--->

# 38: Java -

## Examples

### 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.util.concurrent` ◦

- 'java.lang.Thread'

## Thread javadoc

```
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. **ForkJoin** ◦ `PrimeThread` ◦ `PrimeThreadRunnable` `ThreadRunnable`.....◦
2. ◦ `PrimeThread.start()` `Thread.start()` “”◦

Runnable ◦ **Java 1.1** ◦ `Runnable` ◦

```
final long minPrime = ...
new Thread(new Runnable() {
    public void run() {
        // compute primes larger than minPrime
        . . .
    }
}).start();
```

## lambda Java 8

```
final long minPrime = ...
new Thread(() -> {
    // compute primes larger than minPrime
```

```
    . . .
}).start();
```

- ◦

◦ ◦

- ◦
- Java◦
- /Java◦

Java◦

- ◦ 512K1M◦ ◦
- ◦ ◦
- ◦ OS◦
- waitnotify/ notifyAll◦

“”◦ ◦

◦

◦ ◦ ExecutorService◦

-

◦ 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
```

## Linux 64bit Java 8 u101PC33.633.9。

ExecutorServiceFuture。

```
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
```

## 5.35.6。

。。

```
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.
}
}

```

1000stopstop ◦

◦

main◦ ◦ ◦ tt.stoptrue◦

◦ truestop◦ ◦

Java◦ ◦ ◦ stopstoptrue◦

Java◦

stop

1. stopvolatile;

```
private volatile boolean stop = false;
```

volatileJLS◦

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

◦ ◦

## Java ◦ Java

◦

◦ ◦ ◦ ◦

◦ ◦ ◦ /◦

◦ ◦ ◦ “”◦

◦

## Java ◦ Java ◦ volatile ◦ Java

◦ ◦

1. ◦ ◦

2. ◦ ◦

3. ◦ I/O ◦ ◦

4. JIT ◦ ◦ ◦

◦

Java - <https://riptutorial.com/zh-TW/java/topic/5567/java--->



# 39: Java -

Java ◦ ◦

Java ◦

## Examples

-

JavaJava ◦ ◦

◦ API ◦ SOLID ◦ ◦

public ◦

- ◦ ◦ API ◦ API ◦

- ""

Java ◦ switchfallthrough; "break" ◦ "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 "" ◦ ◦

```
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");
```

## Javaelse Java

```
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");
```

o

```
for (int i = 0; i < 5; i++);
    System.out.println("Hello");
```

“” for println

```
for (int i = 0; i < 5; i++);
    System.out.println("The number is " + i);
```

i“” println

o

1. Java“” . . .

2. Java

3. IDEJava

4. Java“then”“else”body. {

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 if “then” “else” “” “”

```

if (a) {           // <- open brace
    doSomething();
    doSomeMore();
}                 // <- close brace

```

## Java if “then”

```

if (a)
    doSomething();

```

```

if (a) doSomething();

```

## Java if “then” “else”

“”

◦

```

if (a)
    doSomething();
    doSomeMore();

```

at true doSomething doSomeMore ◦ **Java** doSomeMore() if ◦

```

if (a)
    doSomething();
doSomeMore();

```

“”

else ◦

```

if (a)
    if (b)
        doX();
    else if (c)
        doY();
else
    doZ();

```

afalsedoZ ° °

```
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 (b) {
    doX();
  } else if (c) {
    doY();
  } else {
    doZ();
  }
}

if (a) {
  if (b) {
    doX();
  } else if (c) {
    doY();
  } else {
    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.equalsHashCodeObject`

- `equals` ◦ `equals(Object)` ◦ `equals(String)` ◦
- `hashCode` ◦ `hashCode()` **C** ◦

`Person` ◦

**Java 5** ◦ `@Override`

**Java SE 5**

```

public final class Person {
    ...

    @Override
    public boolean equals(String other) {
        ....
    }

    @Override
    public hashCode() {
        ....
    }
}

```

`@Override` ◦

-

```

// Print the sum of the numbers 1 to 10
int count = 0;
for (int i = 1; i < 10; i++) {    // Mistake here ....
    count = count + i;
}
System.out.println("The sum of 1 to 10 is " + count);

```

**Java** ◦ **18** ◦

**Java** ◦ **0** ◦ **10** ◦ **108** ◦

-

**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
```

Testmainjava° java°

TestStringjava.lang.String° Test.main

```
void main(com.example.String[] args)
```

```
void main(java.lang.String[] args)
```

java°

java.langJava SE° °

- '==='

## Java

```
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 = ...bug x = true true x true check ""

== 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
    ...
}
```

1.0 somelib 1.0 otherlib otherlib 2.0 1.0 2.0 otherlib Context

Test Context

- ◦
- ◦
- Java
- ◦ IDE
- IDE

## 'assert'

StackOverflow assert

- 
- IllegalArgumentException
- Google Guava Preconditions
- Apache Commons Lang3 Validate

Java JLS 14.10 Java 8

- 
- ◦ ◦ ◦

◦ ◦

IllegalArgumentException ArrayIndexOutOfBoundsException NullPointerException ◦ ◦ ◦  
AssertionError **trytrythrow**◦

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

        // example:
        Boolean ignore = null;
        if (ignore == false) {
            System.out.println("Do not ignore!");
        }
    }
}
```

nullfalse◦ booleanBoolean Boolean Object◦ null NullPointerException ◦

**Java**nullNullPointerException ◦ false == null;incomparable types: int and <null> ◦

**Java** - <https://riptutorial.com/zh-TW/java/topic/5382/java--->



# 40: JAXB

XML JAXBJava JAXBJavaJavaXML。 JAXBJavaxml。

- JAXB.marshallObjectFileObjOfXML;
- Object obj = JAXB.unmarshallFileObjOfXMLclassName;

fileObjOfXML	XMLFile
	.class

JDKXJCxml .xsdxmljavaXJC。

## 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。 userFilemarshal()。

UserDetails.xmlXML。

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<user>
  <name>Jon Skeet</name>
  <userID>8884321</userID>
</user>
```

## XML

UserDetails.xmlXML

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<user>
  <name>Jon Skeet</name>
  <userID>8884321</userID>
</user>
```

User.javaPOJO

```
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。

## XmlAdapterxml

XMLJavaXmlAdapterxml。。

```

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

@XmlAccessorType/XML ◦ @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](#)

```

@XmlAccessorType(XmlAccessType.NONE) // we want no automatic field/property marshalling
public class ManualXmlElementExample {

```

```

    @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 ManualXmlElementExample(), System.out);
        } catch (Exception e) {
            System.err.println("Exception occurred while writing in XML!");
        }
    }
}

```

## XmlAdapter

- static◦

UnmarshallerXmlAdapterXmlAdapter/◦

- 

```

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);
}
}

```

o

## Java SE 7

### Java 7getImage

```

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

### 2xmlsJon Skeet2

```

<?xml version="1.0" encoding="UTF-8"?>

<user name="Jon Skeet"
image="https://www.gravatar.com/avatar/6d8ebb117e8d83d74ea95fbdd0f87e13?s=328&d=identicon&r=PG" />

```

```

<?xml version="1.0" encoding="UTF-8"?>

<user name="Jon Skeet (Earth 2)"
image="https://www.gravatar.com/avatar/6d8ebb117e8d83d74ea95fbdd0f87e13?s=328&d=identicon&r=PG" />

```

```

ImageCacheAdapter adapter = new ImageCacheAdapter();

JAXBContext context = JAXBContext.newInstance(User.class);

Unmarshaller unmarshaller = context.createUnmarshaller();

// specify 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());

```

### XMLJava

package-info.java XMLJava 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; // Package where you intend to apply trimming filter

import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

```

JAXB <https://riptutorial.com/zh-TW/java/topic/147/jaxb>

---

# 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/zh-TW/java/topic/4105/jax-ws>

# 42: JMX

JMXWeb ◦ ◦

## Examples

### Platform MBean Server

◦ ◦

```
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

```

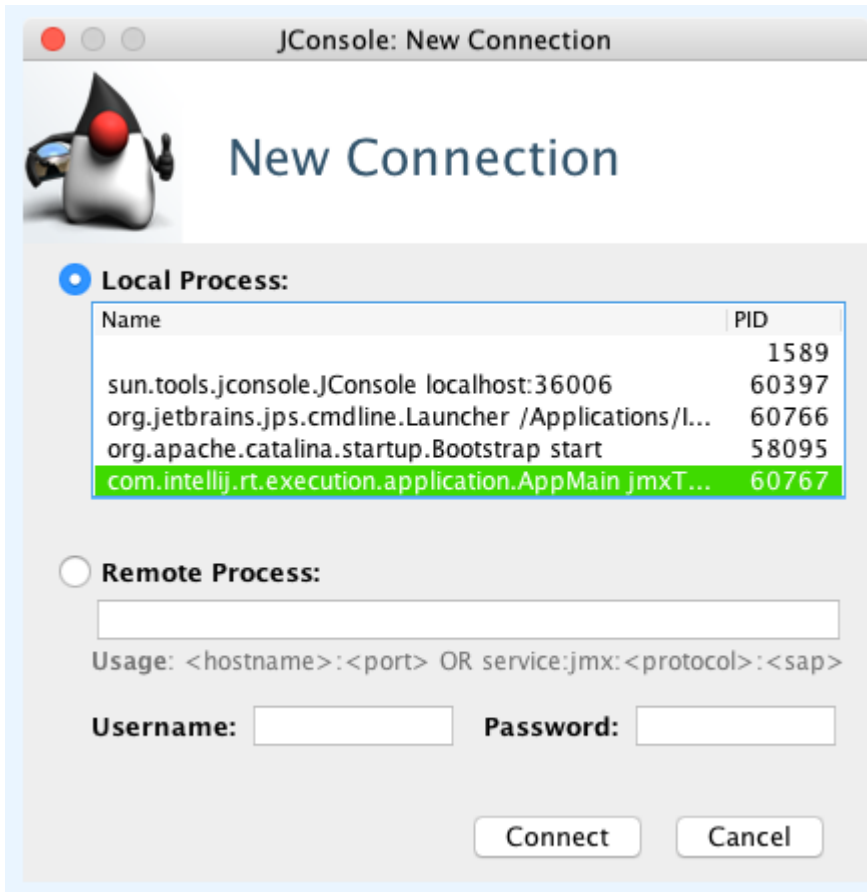
import javax.management.InstanceAlreadyExistsException;
import javax.management.MBeanRegistrationException;
import javax.management.MBeanServer;
import javax.management.MalformedObjectNameException;
import javax.management.NotCompliantMBeanException;
import javax.management.ObjectName;
import java.lang.management.ManagementFactory;

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);

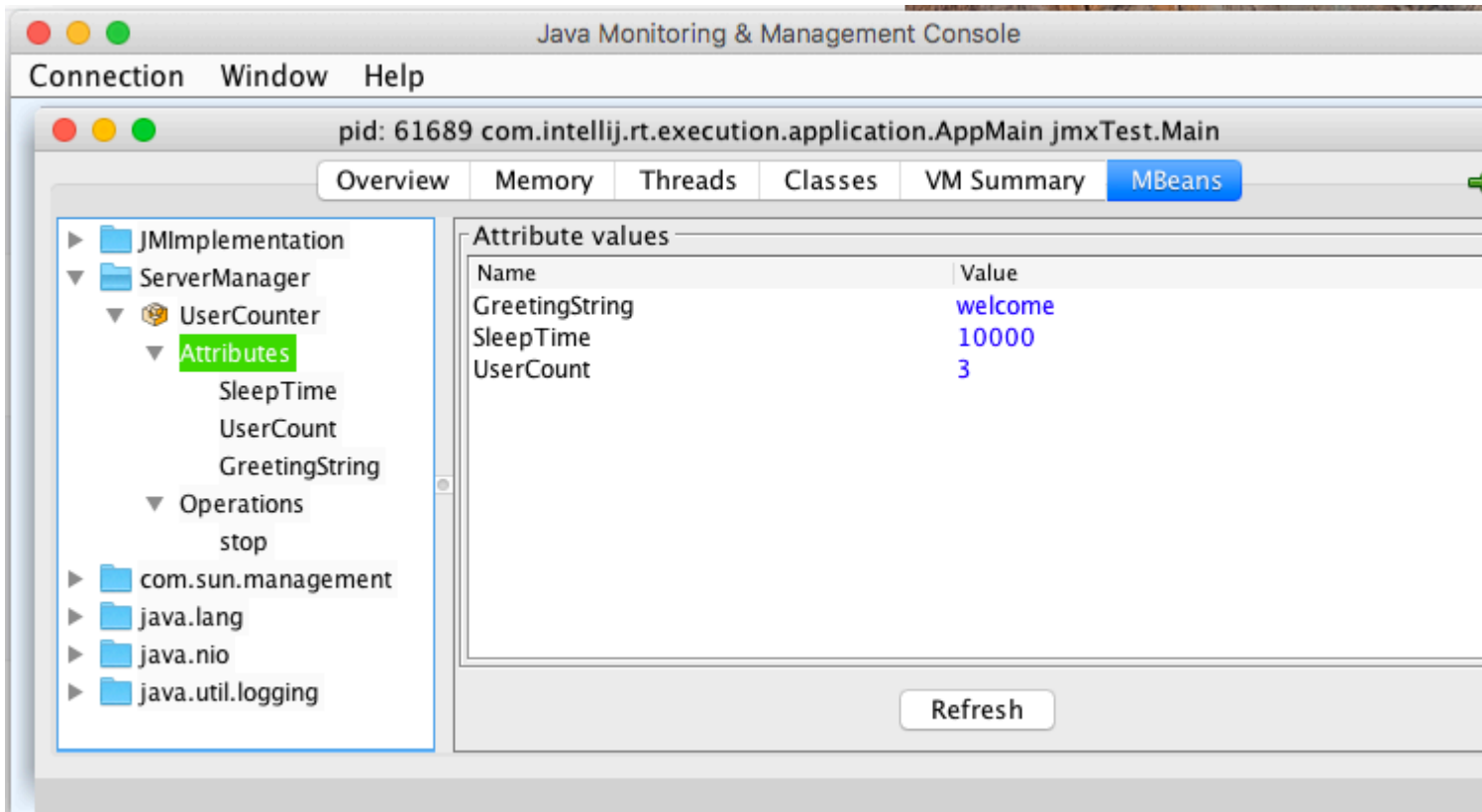
        final Thread thread = new Thread(userCounter);
        thread.start();
        thread.join();
    }
}

```

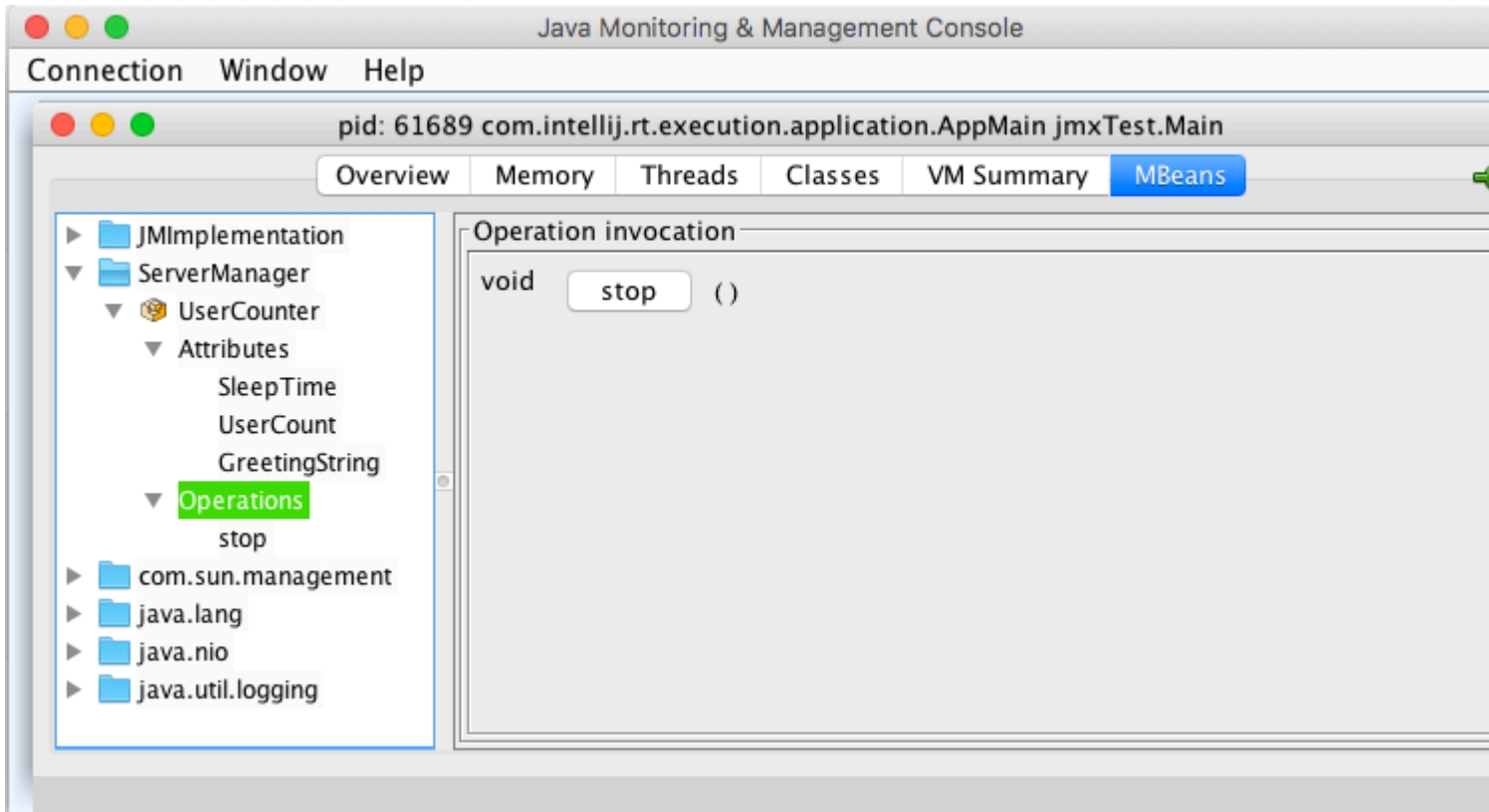
jConsole\$JAVA\_HOME/bin java



MBeansMainObjectName MBeanServerManager ◦ Attributes◦ get◦ getset◦



“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
```

**JMX2** ◦ `jconsole host:port` **jConsole GUI** `host:port`  
service: `jmx:rmi:///jndi/rmi://hostName:portNum/jmxrmi` ◦

- [JMX](#)
- [JMX](#)

**JMX** <https://riptutorial.com/zh-TW/java/topic/9278/jmx>

# 43: JNDI

## Examples

### RMIJNDI

#### JNDIRMI

- //APIRMI
- RMI/API

#### RMI RegistryRMIJNDI

java.rmi.registry.CreateRegistry() java.rmi.registry.CreateRegistry() RMI

#### 1. Server.javaJNDI

```
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.javaJNDI

```

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.javaRMI

```

package com.neohope.jndi.test;

import java.io.Serializable;
import java.rmi.Remote;

/**

```



```

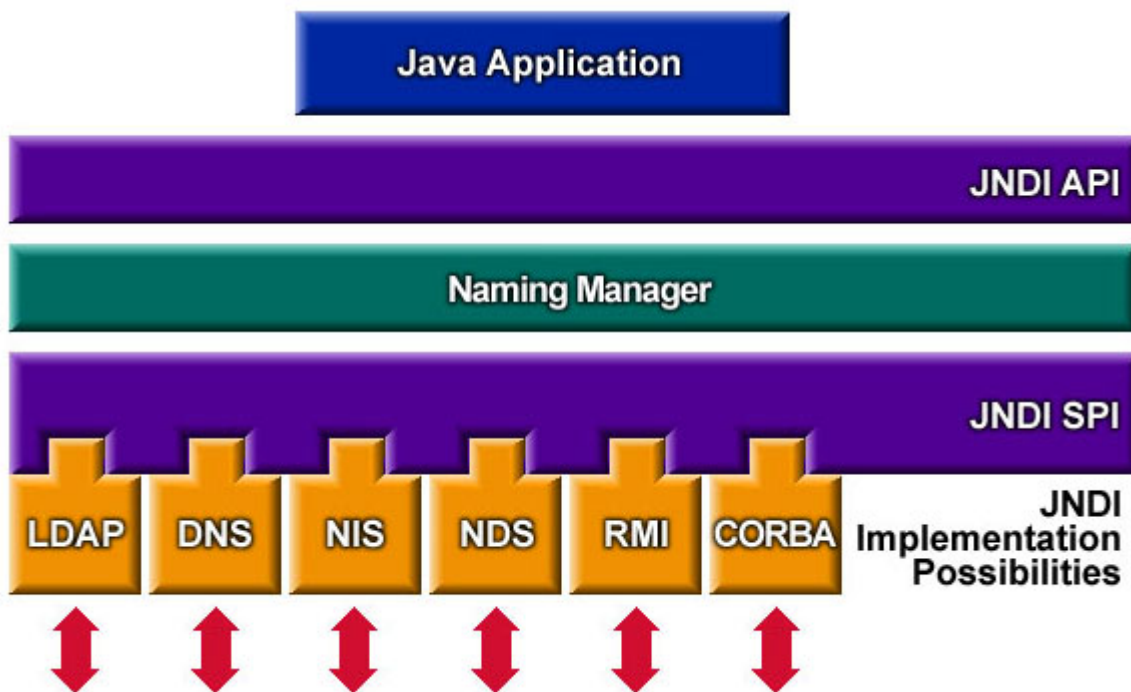
* NMessage
* RMI server class
* must implements Remote and Serializable
*/
public class NMessage implements Remote, Serializable {
    public String message = "";

    public NMessage(String message)
    {
        this.message = message;
    }
}

```

example

- 1.
- 2.



Java JNDI Java API Java

JNDI API SPI 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. Context◦ Context◦ Context◦

```
//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 url 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/zh-TW/java/topic/5720/jndi>

# 44: JShell

JShellJDK 9JavaREPL。 Java。 jdk 9 [http //jdk.java.net/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> - shell
- jshell> / edit <identifier> -
- jshell> / set editor <command> - / edit
- jshell> / drop <identifier> -
- jshell> / reset - JVM

JShellJava 9 JDK20173[jdk9.java.net](http://jdk9.java.net)。 jshellUnable to locate an executableJAVA\_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

JShellJAVA\_HOMEJDK 9。 JShell

```
$ jshell
```

```
jshell>◦
```

### JShell

## JShellJShell

```
jshell> /exit
```

## JShellJava

```
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** ◦ / **method / class** ◦ /edit ◦ barFoo

```
jshell> class Foo {  
  ...> void bar() {  
  ...> }  
  ...> }
```

◦

```
jshell> /edit Foo
```

swing ◦ **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)
|
```

\reset **JVM**\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/zh-TW/java/topic/9511/jshell>

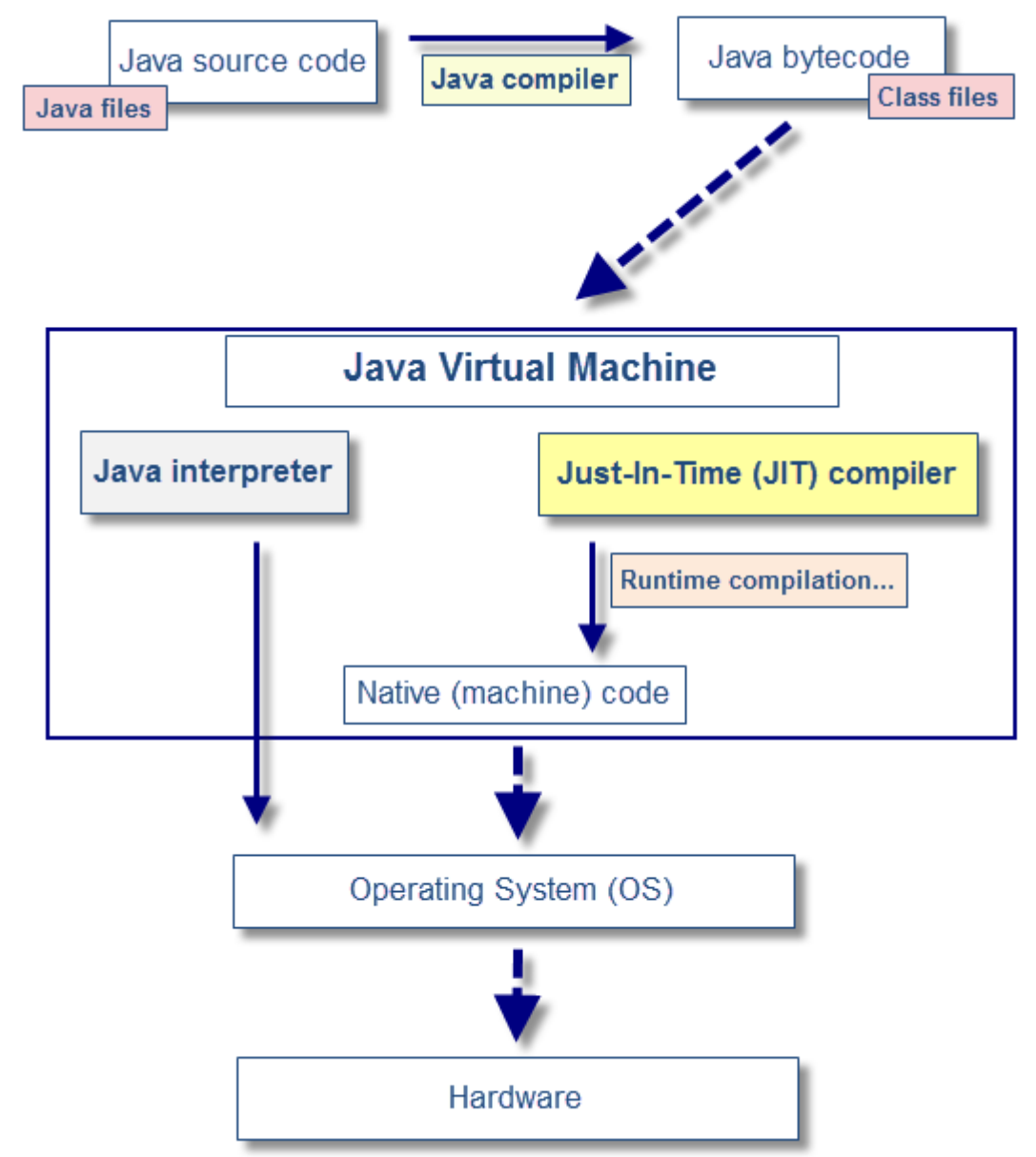
# 45: Just in Time JIT

Symantec JIT 1.1.5 Sun Java

Hotspot JIT 1.2.2 Sun Java Java 1.3 JIT

Java JIT

## Examples



Just-In-Time JIT Java™ Java

- Java JVM
- JVM

Java。

JITJava。

JITJava。 JIT "just in time"。

JVM。 Java。

JIT。 JVM。 。

---

- ◦ JVMcall count call countcall count。
- JVMJIT。
- JVM。
- JITJVM。
- ◦
- ◦
- JITJIT。
- ◦

JavaJIT。

JIToperational data at run timeoperational data at run time 。

JITJava。 JITJIT。

Just in TimeJIT <https://riptutorial.com/zh-TW/java/topic/5152/just-in-time-jit->

# 46: JVM

## JVM TM

### 1.2

<http://docs.oracle.com/javase/7/docs/platform/jvmti/jvmti.html>

## Examples

### 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 heapObjectReferencesCallback(
    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, &heapObjectReferencesCallback,
    (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);
        }
    }
}
}
}

```

## JVMTI

### 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/zh-TW/java/topic/3316/jvm>

# 47: JVM

- 
- 

Java◦

## Examples

### -XXaggressive

-XXaggressiveJVM◦ JVM;◦ ◦

```
-XXaggressive:<param>
```

<PARAM>	
opt	◦
memory	◦ JRockit JVM◦

### -XXallocClearChunks

TLATLA◦ Onull◦ Java◦ TLA◦

```
-XXallocClearChunks
```

```
-XXallocClearChunks=<true | false>
```

IA64;◦ -XXallocClearChunkSize◦ true◦

### -XXallocClearChunkSize

-XXallocClearChunkSize◦ 512◦

```
-XXallocClearChunks -XXallocClearChunkSize=<size> [k|K] [m|M] [g|G]
```

### -XXcallProfiling

◦ JVM◦

JRockit JVM R27.3.0◦ ◦

```
java -XXcallProfiling myApp
```

◦ ◦

## -XXdisableFatSpin

Java◦

Java◦ ◦ “”◦ “”◦ ◦ ◦ -XXdisableFatSpin-XXdisableFatSpin◦

```
-XXdisableFatSpin
```

## -XXdisableGCHeuristics

◦ ◦ ◦

```
-XXdisableFatSpin
```

## -XXdumpSize

◦

```
-XXdumpsize:<size>
```

none	◦
small	WindowsLinux◦ ◦ Service Pack 12JRockit JVM 8.1Service Pack 37.0◦
normal	◦ Java◦ JRockit JVM 1.4.2◦
large	Java◦ -XXdumpSize-XXdumpFullState◦

## -XXexitOnOutOfMemory

JRockit JVM◦ JRockit JVM◦ JRockit JVM◦

```
-XXexitOnOutOfMemory
```

JVM <https://riptutorial.com/zh-TW/java/topic/2500/jvm>

# 48: Lambda

Lambda ◦ ◦ ◦ ◦

◦ lambda ◦

- - > {return expression; } // ◦
- - > //; ◦
- - > {function-body} // lambda ◦
- parameterName - > expression // lambda ◦ lambda ◦
- parameterNames secondParameterName... - > // lambda
- parameterNames secondParameterName... - > expression // ◦ ◦

## Examples

### Lambda

Java 8<sup>1</sup> [java.util.Comparator](#)

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 lambda ◦ p1p2

```
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◦ SetCollection API◦

---

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`◦ SAM◦

```
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](#) [java.util.function](#) ◦ [Predicate<T>](#) [Tboolean](#) ◦

# Lambda

## Lambda

Lambda Operator

```
fiFunctionalInterface{ System.out.println("Hello"); } ◦
```

```
FunctionalInterface fi = new FunctionalInterface() {
    @Override
    public void theOneMethod() {
        System.out.println("Hello");
    }
};
```

lambda""lambda this supertoString() ◦



## lambda - Java

### lambdalambda

```
Object fooHolder = (Foo1) () -> System.out.println("Hello");
System.out.println(fooHolder instanceof Foo1); // returns true
```

### lambda

```
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
```

---

### lambdaJava

```
IntUnaryOperator addOneShort = (x) -> (x + 1);
IntUnaryOperator addOneLong = (x) -> { return (x + 1); };
```

---

### lambdas;finallambda

```
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
}
```

### lambdaJava Closures

---

# Lambda

### lambdalambdalambda

```
public void passMeALambda(Foo1 f) {
    f.bar();
}
```

```
}  
passMeALambda(() -> System.out.println("Lambda called"));
```

# Lambda

lambda ◦ ◦ lambda

- myPredicate = s -> s.isEmpty()
- stream.filter(s -> s.isEmpty())
- return s -> s.isEmpty()
- (Predicate<String>) s -> s.isEmpty()

lambda ◦ lambda ◦ -> o.isEmpty() ◦ lambda

```
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();
```

lambda ◦

lambda ◦

```
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)
```

lambda

```
people.stream().map(person -> person.getName())
```

Person::getName ◦ ◦

```
people.forEach(System.out::println);
```

System.outPrintStream◦

## lambda

```
people.forEach(person -> System.out.println(person));
```

```
List<Integer> numbers = Arrays.asList(1, 2, 3, 4, 5, 6);  
numbers.stream().map(String::valueOf)
```

StringvalueOf()◦ valueOf()◦

## lambda

```
numbers.stream().map(num -> String.valueOf(num))
```

```
List<String> strings = Arrays.asList("1", "2", "3");  
strings.stream().map(Integer::new)
```

◦

IntegerString◦ ◦ lambda

```
strings.stream().map(s -> new Integer(s));
```

		Lambda
	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]

\* instancegetInstance()::method this::method

\*\* TypeName

lambda◦ [java.io.Serializable](#)◦

ComparatorTreeSet◦

```
TreeSet<Long> ts = new TreeSet<>((x, y) -> Long.compare(y, x));
```

## lambdaSerializable ◦ lambda

```
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

### lambdasFunctionalInterface ◦ lambdasExecute-Around ◦ /◦ ioio / 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

## Lambdas ◦ ◦

## RunnableThread

```
//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);
}
```

## lambdaRunnable.

```
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` lambda

return lambda.

## Scala Kotlin

```
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!
    });
}
```

forreturn

```
void demo2() {
    for (int i = 0; i < 3; i++) {
        System.out.println(i);
        return; // Return from 'demo2' entirely
    }
}
```

```
void demo3() {
    threeTimes(new MyIntConsumer());
}

class MyIntConsumer implements IntConsumer {
    public void accept(int i) {
        System.out.println(i);
        return;
    }
}
```

## Java returnforwhile◦

## 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);
```

## Javatry-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()◦ ScalaKotlintry-with-resourcesbefore accept()◦

## lambdaJava◦

## lambdalambda◦ ◦

## lambdafinal◦ Java 8lambdas final◦

```
int n = 0; // With Java 8 there is no need to explicit final
```

```
Runnable r = () -> { // Using lambda
    int i = n;
    // do something
};
```

n° lambda

“lambda”。

```
int n = 0;
Runnable r = () -> { // Using lambda
    int i = n;
    // do something
};
n++; // Will generate an error.
```

lambda<sub>final</sub>。

```
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
```

lambda。

Java。 Java lambda。 。

```
// 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。 。

```
// Correct ...
```

```
public class Accumulator {
    private int value = 0;

    public int accumulate(int x) {
        value += x;
        return value;
    }
}
```

## Lambda -

Java 8 JButtonclick ◦ btn.addActionListenerbtn.addActionListener ◦

```
JButton btn = new JButton("My Button");
btn.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        System.out.println("Button was pressed");
    }
});
```

## Lambda

ActionListeneractionPerformed() Lambda ◦ Lambda

```
JButton btn = new JButton("My Button");
btn.addActionListener(e -> {
    System.out.println("Button was pressed");
});
```

## Lambda

```
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));
    }
}
```

## Lambda

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.

5.

```
MathOperation isEven = num -> num%2 == 0;
```

## Lambda

Java lambda<sup>1</sup>. lambda - s -> s.length() lambdas s -> s.length() - lambdas. lambda () -> j  
lambda

```
public static void main(String[] args) throws Exception {
    for (int i = 0; i < 1000000000; i++) {
        int j = i;
        doSomethingWithLambda(() -> j);
    }
}
```

new 1,000,000,000 () -> j lambda. Java <sup>1</sup>lambda.

1 - Java 9Java<sup>1</sup>.

## lambda

Java 8lambda.

lambda. 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** `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);` **lambdaPredicate** ◦ `checker.test(person)` ◦

`print(personList, p -> p.getName().startsWith("J"));` ◦ **“J”** ◦

**Lambda** <https://riptutorial.com/zh-TW/java/topic/91/lambda>

# 49: LinkedHashMap

LinkedHashMapMapHashLinked list。 HashMapMap。

Java LinkedHashMapLinkedHashMap。 。 。 HashMap。

## Examples

### Java LinkedHashMap

-

- MapHashLinked list。
- HashMapMap。
- 。
- 。
- 。
- HashMap。

-

- void clear。
- boolean containsKeyObject key。
- get。
- protected boolean removeEldestEntryMap.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"));  
}
```

**LinkedHashMap** <https://riptutorial.com/zh-TW/java/topic/10750/linkedhashmap>

---

# 50: log4j / log4j2

Apache Log4j Java Java。 JavaLog4j。

- `Logger.debug“”; //`
- `Logger.info“”; //`
- `Logger.error“”; //`
- `Logger.warn“”; //`
- `Logger.trace“”; //`
- `Logger.fatal“”; //`
- `Log4j2`
- `Logger.debug“Debug params { } { }”param1param2param3; //`
- `Logger.info“Info params { } { }”param1param2param3; //`
- `Logger.error“Error params { } { }”param1param2param3; //`
- `Logger.warn“Warn params { } { }”param1param2param3; //`
- `Logger.trace“Trace params { } { }”param1param2param3; //`
- `Logger.fatal“Fatal params { } { }”param1param2param3; //`
- `Logger.error“Caught Exception”ex; //`

---

## Log4j 1

201585Log4j 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>
```

```
<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.

---

## Maven

POM.xml

```
<dependency>
  <groupId>log4j</groupId>
  <artifactId>log4j</artifactId>
  <version>1.2.17</version>
</dependency>
```

```
<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

### JavaLog4j

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<sup>o</sup> 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>

```

log4j2.xmlappenderroot logger<sup>o</sup> .

log4j2.xmlstatus = <WARN | DEBUG | ERROR | FATAL | TRACE | INFO>log4j2.xmlstatus = <WARN | DEBUG

| ERROR | FATAL | TRACE | INFO> ◦

◦ monitorInterval = 30 ◦ **30**◦

## log4j 1.x2.x.

log4j 1.xlog4j 2.xlog4j 1.x

### Log4j 1.x API

#### Maven Build

```
<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 Bridgelog4j 1.xApache Commons Logginglog4j 2.x

#### Maven Build

```
<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 commons

<https://logging.apache.org/log4j/2.x/maven-artifacts.html>

- DB

JDBC。 IBM System iDB2 [JT400](#)

DB2JDBC 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
```

## Logoutputlog4j 1.x

“”ERROR。 **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/zh-TW/java/topic/2472/log4j---log4j2>

# 51: Nashorn JavaScript

Nashorn Oracle Java JavaScript Java 8. Nashorn JSR-223 Java Javascript Javascript ECMA Javascript

- ScriptEngineManager //ScriptEngine;SPI
- ScriptEngineManager.ScriptEngineManager//
- ScriptEngine //
- ScriptEngine ScriptEngineManager.getEngineByNameString shortName//
- Object ScriptEngine.evalString script//
- Object ScriptEngine.evalReader reader//
- ScriptContext ScriptEngine.getContext//readerwriter
- void ScriptContext.setWriterWriter writer//

Nashorn Java JavaScript Java 8. javax.script.

ScriptEngineManager APINashorn JavaScript.

## 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.

```

## NashornJavaScriptJava

JavaNashornJava。 JavaScriptNashornjava。

JavaScriptJava。

1. if。 JSifJava。 nullundefined0false
2. Nashorn - - JSJava。
3. 。 JS。

				。
Java null				
Java				0
Java				
Java/	= 0			
Java ArrayList				
Java HashMap				
Java HashSet				

## Recommendatons

- if (some\_string)
- for eachnull
- nullundefinedJava

```

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 JavaScript <https://riptutorial.com/zh-TW/java/topic/166/nashorn-javascript>

# 52: NIO -

[SelectionKey](#) [Selector](#) [Channel](#) ◦ ◦

OP\_READ ◦ OP\_WRITE ◦ OP\_WRITE ◦

OP\_CONNECT ◦ SO ◦ finishConnect () OP\_CONNECT ◦

## Examples

### Selector OP\_CONNECT

NIOJava 1.4“Channels”I/O ◦ [SelectableChannel](#) [Channel](#) ◦ C select ()

- OP\_ACCEPT
- OP\_CONNECT
- FIFO OP\_READ
- FIFO OP\_WRITE

I/O// ... I/O// ... ◦ I/OI/O ◦ ◦

1. [Selector](#)
2. [SocketChannel](#)
3. [SocketChannelSelector](#)
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 attachment
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: empty 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/zh-TW/java/topic/5513/nio---->



---

# 53: Oracle

JavaOracleOracleJava◦ ◦

- Oracle◦ ◦
- ◦
- Brian GoetzJavaMark ReinholdJava ◦
- ;◦ ◦

## Examples

---

- ◦
- ◦ /◦
- ◦ APIAPIjava.lang.annotationjava.lang.annotations◦
- com.yourcompany.widget.button com.yourcompany.core.api

- 
- ◦
  - ...◦
  - 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]?\$
- KVRT
- \_◦
-

T V SRC\_VERTEX

---

static final \_ ◦

- `^[AZ][A-Z0-9]*(_[A-Z0-9]+)*$`
- `BUFFER_SIZE MAX_LEVEL`

- 
- `/◦`
  - `◦ ◦`
  - `◦`
  - **Java**

Oracle[Java](#)

## Java

- `LFASCII10CRCR + LF◦`
- `◦`
- `.java◦ package-info.java ◦`
- `LFSpaceASCII32◦ ◦`
- `\' \" \\ \t \b \r \f\n\047 Unicode\u0027◦`
- `◦`

```
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;
```

- .....

- .../°
- ...
  - java
  - javax
  - org.xml
  - com.sun
- ...

- ○

- ○

- ○

- “”°

- ○

- 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 + "]";
    }
}
```

- ○

- ;°

- **arity**° ○

- ○

- ○

```
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 **static** ◦
- final ◦
- final ◦
- ◦
- ◦ ◦
- ◦ ◦
- case **case** ◦

```

switch (var) {
    case TWO:
        setChoice("two");
        break;
    case THREE:
        setChoice("three");
        break;
    default:
        throw new IllegalArgumentException();
}

```

- 
- 80100 ◦
  - “” ; “” 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◦
- ◦ **18**◦
  - **28**◦
  - **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
-

throwstrows+8+8。

- /。
- 
- 。

```
popupMsg("Inbox notification: You have "
         + newMsgs + " new messages");

// Don't! Looks like two arguments
popupMsg("Inbox notification: You have " +
         newMsgs + " new messages");
```

- .....
- 
- 
- 
- 
- 
- 

- ...
- 
- 
- 

- 。

- .....
- 
- lambdafor
- //。
- 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) {
    ...
}
```

## Lambda

```
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));
```

- `lambdablock` `lambdas`◦
- `lambda`◦
- `arity1lambda`◦ ◦
- ◦
- `lambda`◦

```
return flag ? "yes" : "no";

String cmp = (flag1 != flag2) ? "not equal" : "equal";

// Don't do this
return (flag ? "yes" : "no");
```

- ◦
- ◦ ◦
- 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 **literals**L◦
- 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**◦
  - ◦
  - if / else◦
  - ◦
- do...whileelse catchwhiledo...while◦

---

```
enum Response { YES, NO, MAYBE }
public boolean isReference() { return true; }
```

/◦ “”◦ **lambda**◦





# 54: RSA

## Examples

### OAEPGCM

AES GCM OAEP 128 AES。

PKCS1 v1.5 OAEP Oracle。 GCM Oracle。

。 RSA。 AES / GCM。

1. `RSAPrivateKey.getKeySize()`;
2. `/RSA`;
3. `GCM128Java`。

- 2048 RSA;
- AES-256;
- CMS / PKCS7 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 since 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
    GCMPParameterSpec staticParameterSpec = new GCMPParameterSpec(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();
}

```

o o

```

/**
 * 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
    GCMPParameterSpec staticParameterSpec = new GCMPParameterSpec(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/zh-TW/java/topic/1889/rsa>

# 55: Stack-Walking 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,
12     IllegalAccessException, IllegalArgumentException, InvocationTargetException {
13         Method fooMethod = FooHelper.class.getDeclaredMethod("foo", (Class<?>[])null);
14         fooMethod.invoke(null, (Object[]) null);
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() + " " +
29             frame.getMethodName());
30         }
31 }
```

```
test.BarHelper 26 bar
test.FooHelper 19 foo
test.StackWalkerExample 13 main
```

◦ [RETAIN\\_CLASS\\_REFERENCE StackWalker](#) 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
```

/o o [SHOW\\_REFLECT\\_FRAMESStackWalker](#)

```

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,
IllegalAccessOperationException, 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()` ◦

**Stack-Walking API** <https://riptutorial.com/zh-TW/java/topic/9868/stack-walking-api>

# 56: StringBuilder

Java StringBuilder。 Java StringBuilderStringBuffer。 JDK 1.5。

- StringBuilder
- StringBuilderint capacity
- StringBuilderCharSequence seq
- StringBuilderStringBuilder
- new StringBuilderString string
- StringJoinerCharSequence
- new StringJoinerCharSequenceCharSequenceCharSequence

charStringBuilderint capacityString 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

StringnString s。

String

```
final int n = ...  
final String s = ...  
String result = "";  
  
for (int i = 0; i < n; i++) {  
    result += s;  
}
```

$n \times O(s.length() * n^2) = O(s.length() * (1+2+\dots+(n-1)+n))$ 。

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();

```

## StringBufferStringBuilderFormatterStringJoiner

StringBuffer StringBuilder FormatterStringJoinerJava SE

- StringBufferJava 1.0™。
- Java 5StringBuilderStringBuffer。 API。 StringBufferStringBuilder。

StringBuilder

```

int one = 1;
String color = "red";
StringBuilder sb = new StringBuilder();
sb.append("One=").append(one).append(", Colour=").append(color).append('\n');
System.out.print(sb);
// Prints "One=1, Colour=red" followed by an ASCII newline.

```

StringBufferStringBuilderStringBuffer

StringBufferStringBuilder;。 。

- FormatterJava 5Cprintf。 。 。
- StringJoinerJava 8。 。 APIJava 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`◦
- `StringBuilder` `StringBuffer` ◦
- `FormatterStringBuilder` ◦ `Formatter.format(...)`
  - `format`
  - ***varargs***
  - ◦
- `StringJoiner`◦

**StringBuilder** <https://riptutorial.com/zh-TW/java/topic/1037/stringbuilder>

# 57: sun.misc.Unsafe

UnsafeJava ◦ Unsafe ◦

1. Unsafe APIJVM/◦
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.UnsafePrivategetUnsafe()◦ ◦

### bootclasspathsun.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.MyAppUnsafeLoader.loadUnsafe() ◦

◦ 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)
<b>CAS</b>	compareAndSwap*
	setMemory
	get*Volatile put*Volatile putOrdered*

getput. null.

```

// Putting a value to a field
protected static long fieldOffset = UNSAFE.objectFieldOffset(getClass().getField("theField"));
UNSAFE.putLong(this, fieldOffset , newValue);

// Putting an absolute value
UNSAFE.putLong(null, address, newValue);
UNSAFE.putLong(address, newValue);

```

intlong. floatToRawIntBits intBitsToFloat, doubleToRawLongBits , longBitsToDouble.

[sun.misc.Unsafe](https://riptutorial.com/zh-TW/java/topic/6771/sun-misc-unsafe) <https://riptutorial.com/zh-TW/java/topic/6771/sun-misc-unsafe>

# 58: ThreadLocal

SimpleDateFormat Marshaller

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 Web SimpleDateFormat SimpleDateFormat ThreadLocal SimpleDateFormat

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-0formatterthread-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。 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/zh-TW/java/topic/2001/threadlocal>

# 59: TreeMapTreeSet

TreeMapTreeSet Java 1.2 Java ◦ TreeMap Map ◦ TreeSet Set ◦

TreeMapO(log n) ◦ TreeMapTreeSet ◦

◦

## 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 elements 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 map ◦

#### 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 elements in the set  
System.out.println(treeSet.contains(12)); // Prints true  
System.out.println(treeSet.contains(15)); // Prints false
```

## Iteratorforeachmap。

## 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  
}
```

## JavaTreeMap / TreeSet

TreeMapTreeSet/◦ TreeMapTreeSet◦

Person

```
public class Person {  
  
    private int id;  
    private String firstName, lastName;  
    private Date birthday;  
  
    //... Constructors, getters, setters and various methods  
}
```

TreeSet TreeMapKey

```
TreeSet<Person2> set = ...  
set.add(new Person(1, "first", "last", Date.from(Instant.now())));
```

```
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;

    //... Constructors, getters, setters and various methods

    @Override
    public int compareTo(Person o) {
        return Integer.compare(this.id, o.id); //Compare by id
    }
}
```

## 2. TreeSet Comparator

### 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**  $\text{sgn}(x.\text{compareTo}(y)) == -\text{sgn}(y.\text{compareTo}(x))$  ◦  $y.\text{compareTo}(x) < x.\text{compareTo}(y)$  ◦

$(x.\text{compareTo}(y) > 0 \ \&\& \ y.\text{compareTo}(z) > 0) \implies x.\text{compareTo}(z) > 0$  ◦

**Z**  $x.\text{compareTo}(y) == 0 \implies \text{sgn}(x.\text{compareTo}(z)) == \text{sgn}(y.\text{compareTo}(z))$  ◦

## TreeMapTreeSet

TreeMapTreeSet ◦

TreeMapTreeSet ◦ ◦

## ConcurrentModificationException ◦ /

## 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 ◦

[TreeMapTreeSet](https://riptutorial.com/zh-TW/java/topic/9905/treemaptreeset) <https://riptutorial.com/zh-TW/java/topic/9905/treemaptreeset>

---

# 60: Varargs

“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
```

o

```
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
```

**Varargs** <https://riptutorial.com/zh-TW/java/topic/1948/varargs-->



# 61: WeakHashMap

HashMap

## Examples

### WeakHashMap

-

- ◦
- ◦

;GC◦

### HashMapWeakHashMap -

JavaWeakHashMap◦

-

```
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"));
    }
}
```

### *HashMapWeakHashMap*

HashMap.size◦ HashMap.removeSize◦

WeakHashMap◦ size◦ **WeakHashMap**◦

[WeakHashMap](https://riptutorial.com/zh-TW/java/topic/10749/weakhashmap) <https://riptutorial.com/zh-TW/java/topic/10749/weakhashmap>

---

# 62: XJC

XJCJava SEXMLJava。

JDK/bin/xjc。

- xjc [options]/ URL / dir / jar ... [-b bindinfo] ...

The logo for javaxsd, featuring the text "javaxsd" in a white sans-serif font on a dark grey rectangular background.

XJCJDK。 unJAXBJava。

## Examples

XSDJava

---

# XSDschema.xsd

xmlxsdnamereputation。

```
<?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>
```



xjcJDKOS。

```
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/zh-TW/java/topic/4538/xjc>

# 63: XML XPath

XPathXML。

W3C。

## Examples

### XMLNodeList

#### 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>
```

JavaexampleXMLXPath。 XMLXPath。

```
XPathFactory xPathFactory = XPathFactory.newInstance();
XPath xPath = xPathFactory.newXPath(); //Make new XPath
InputStream inputSource = new InputStream("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
...
```

### XMLXPath

#### XMLNodeListXPath

## 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

...
```

## XMLXPath

evaluatecompile

```
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/zh-TW/java/topic/4148/xml-xpath>

# 64: XOM - XML

## Examples

### XML

[XOMXMLBuilder](#) Document ◦

```
Builder builder = new Builder();
Document doc = builder.build(file);
```

[xmlDocument](#) getRootElement() ◦

```
Element root = doc.getRootElement();
```

### Elementxml

- getChildElements(String name) - 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



```

1  <example>
2      <person>
3          <name>
4              <first>Dan</first>
5              <last>Smith</last>
6          </name>
7          <age unit="years">23</age>
8          <fav_color>green</fav_color>
9      </person>
10     <person>
11         <name>
12             <first>Bob</first>
13             <last>Autry</last>
14         </name>
15         <age unit="months">3</age>
16         <fav_color>N/A</fav_color>
17     </person>
18 </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

### XOMXML

Element(String name) ◦ Document ◦

```
Element root = new Element("root");
```

Element ◦

- appendChild(String name) - **name** ◦
- appendChild(Node node) - **nodenode** ◦ ◦
- addAttribute(Attribute attribute) - ◦

Attribute ◦ Attribute(String name, String value) ◦

Document ◦ DocumentElement ◦

SerializerXML ◦ 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("Auntry", "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”

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <example>
3   <person>
4     <name>
5       <first>Dan</first>
6       <last>Smith</last>
7     </name>
8     <age unit="years">23</age>
9     <fav_color>green</fav_color>
10  </person>
11  <person>
12    <name>
13      <first>Bob</first>
14      <last>Autry</last>
15    </name>
16    <age unit="months">3</age>
17    <fav_color>N/A</fav_color>
18  </person>
19 </example>
20
```

XOM - XML <https://riptutorial.com/zh-TW/java/topic/5091/xom-----xml>

# 65:

- String◦

## Java

1. java.lang.String
2. java.lang.Integerjava.lang.Bytejava.lang.Characterjava.lang.Shortjava.lang.Boolean  
java.lang.Longjava.lang.Double java.lang.Float
3. ◦
4. java.math.BigIntegerjava.math.BigDecimal
5. java.io.File◦ VM◦ File◦

## Examples

◦

1. “setter” - ◦
2. ◦
3. ◦ final◦ ◦
- 4.
5. ◦
6. ◦ ;◦ ◦

```
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);
    }
}
```

## 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/zh-TW/java/topic/10561/>

# 66:

◦

final ◦ **Java Concurrency in Practice** 3.4.1 ◦

◦

## Examples

◦

Java ◦ `java.util.Date` ◦

◦

```
public class ImmutableIntArray {
    private final int[] array;

    public ImmutableIntArray(int[] array) {
        this.array = array.clone();
    }

    public int[] getValue() {
        return this.clone();
    }
}
```

`int[]`

- `clone()` ◦ `ImmutableIntArray` ◦
- `getValue()` ◦ `clone()` ◦ `ImmutableIntArray` ◦

`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;
    }
}

```

privatepublic static

---

- ◦
- ◦
- **setter**◦
- ◦ privatefinal ◦
  - **getter**◦
- final◦
- ;nullString◦
- final ◦ ◦

## setter

```

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()` `List` `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

```

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;}
}

```

PersonPerson

```

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**

```
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 **S**private ◦

<https://riptutorial.com/zh-TW/java/topic/2807/>

# 67:

- Java ◦ ;◦

## StackOverflow

- [ExecutorServiceThread](#)
- [ThreadRunnable](#)

## 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 AtomicBooleanBlockingQueue](#) ◦ [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 needed 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 needed 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 Concurrency ThreadLocal - . . .

servlet.

```

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
    }
}

```

MyServlet.getContext () MyUserContext . . .

contexts . servletservlet .

## CountDownLatch

## CountDownLatch

◦

1. CountDownLatch ◦
2. countDown() **await** ◦
3. - ◦ CyclicBarrier ◦

```
public void await() throws InterruptedException
```

◦

```
public void countDown()
```

◦

```
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`
2. `await()` `await()` ◦
3. `DoSomethingInAThread` ◦ `countDown()` ◦
- 4.

## Java `synchronized` `JavaJava` ◦

- `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` ◦ `postt` ◦ 1100 ◦

Java ◦ ◦ ◦ ◦

Java ◦ ◦

```
public void bar(){  
    synchronized(this){  
        ...  
    }  
}  
public void foo(){  
    synchronized(this){  
        bar();  
    }  
}
```

synchronizedsynchronized°

## 1. synchronizedthis

```
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 - t10t。 t11t。

t. t.

`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` ◦ `printlnAtomicInteger` ◦

◦ `java` ◦ `synchronized` ◦ `synchronized` ◦

12FirstSecond2R1R2. R1R2SecondR2R1.

t = 0

R1R2. R2R1. ◦

```
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)
```

```
Thread.sleep(1000);
```

◦ ◦

**try / catch** Thread.sleepInterruptedException ◦

## **synchronized / volatile/**

synchronized ◦ synchronizedvolatile / ◦ /

```
class Counter {  
  
    private Integer count = 10;  
  
    public synchronized void incrementCount() {  
        count++;  
    }  
  
    public Integer getCount() {  
        return count;  
    }  
}
```

**A**incrementCount() **B**getCount() ◦ **B**count ◦ count10 count◦

**Java** ◦ **Java** ◦ ◦ ◦ / ◦ synchronized volatile synchronized ◦ / ◦

**B**count count **A** ◦ **B**countgetCount() ◦

```
public synchronized Integer getCount() {  
    return count;  
}
```

**A**countCounter ◦ **B**Countercount ◦

**Thread A**

Acquire lock

Increment 'count'

Release lock

Flush everything to  
main memory

Updates its local copy  
with main memory

Acq

Re

Re

volatile/◦ volatilevolatile◦

## java.lang.Thread

Java◦ ◦ ◦

Java - [java.lang.Thread](#)◦ *run*◦

*Thread*[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.Runnablerun◦ Thread◦ Runnable◦

```
Thread t = new Thread(aRunnable);
```

## lambdasJava 8

```
Thread t = new Thread(operator::hardWork);
```

◦

```
Thread t = new Thread(operator::hardWork, "Pi operator");
```

◦ ◦

---

## java.lang.ThreadGroup◦

```
ThreadGroup tg = new ThreadGroup("Operators");
Thread t = new Thread(tg, operator::hardWork, "PI operator");
```

## ThreadGroup◦ ThreadThreadThreadGroup◦ ThreadGroupThreadThreadThreadGroup◦

## sumarizeThread

```
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](#) ◦ *newThreadRunnable* ◦

```
class WorkerFactory implements ThreadFactory {
    private int id = 0;

    @Override
    public Thread newThread(Runnable r) {
        return new Thread(r, "Worker " + id++);
    }
}
```

/

Javafalse ◦ true ◦ ◦ ◦ Thread ◦ ◦ ◦ WAITINGTIMED\_WAITING ◦ InterruptedExceptiontrue  
 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。 ProducerConsumer。

```

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. sharedQueue LinkedBlockingQueue
2. 200sharedQueue
3. ConsumerssharedQueue
4. synchronizedLock [BlockingQueue](#)

BlockingQueue -

BlockingQueue

/

“”

ReadWriteLockReentrantReadWriteLock

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();
    }
}
}

```

1 AtomicInteger Atomic

2

1. JVMdata646432

2. *Happen Before JVM*

## Java SE 8

StampedLock ReadWriteLock

Runnablerun()

RunnableThread Threadstart()

```

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 vs Thread

RunnableRunnableThread

ThreadThread

RunnableAPI

o o 0 o

Semaphore semaphore = new Semaphore(1); // The int value being the number of permits

Semaphore false true

```
Semaphore semaphore = new Semaphore(1, true);
```

**javadocsSemaphore**◦ `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
    }
}
```

## ThreadPool`int`

ThreadPool◦

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 lambdas

```

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◦ Threads Thread Group Thread set◦

## ThreadGroup

```
Reference Handler
Signal Dispatcher
Attach Listener
Finalizer
```

Runnable◦ RunnableRunnable◦

CallableJava 5 Runnable◦ Callablecallrun ◦ call◦

## Future

FutureCallable◦ 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();
}
```

## CallableFuture

```
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
```

## Future

get() **Future**

- get(long timeout, TimeUnit unit);
- cancel(mayInterruptIfRunning) ◦ mayInterrupt;
- isDone();
- 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/zh-TW/java/topic/121/-->

# 68:

[1] [1] http

//stackoverflow.com/documentation/java/90/collections#t=201612221936497298484

## Examples

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.collectionsCollections.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.CopyOnWriteArrayListList javadoc

“” ConcurrentModificationException

```
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` ◦

`javadoc ◦ 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/zh-TW/java/topic/8363/>

# 69:

[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);

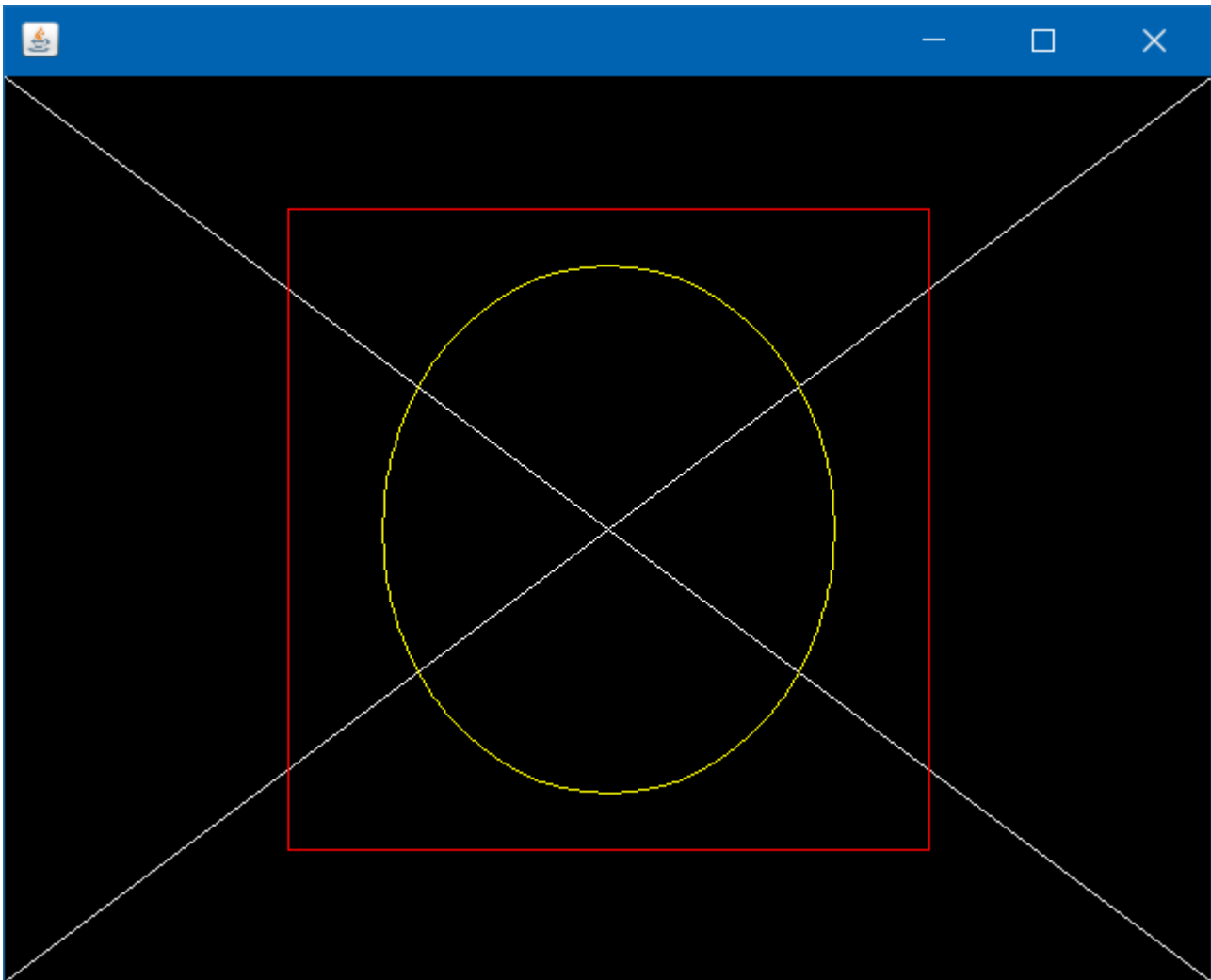
        return img;
    }

    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);
    }
}
```

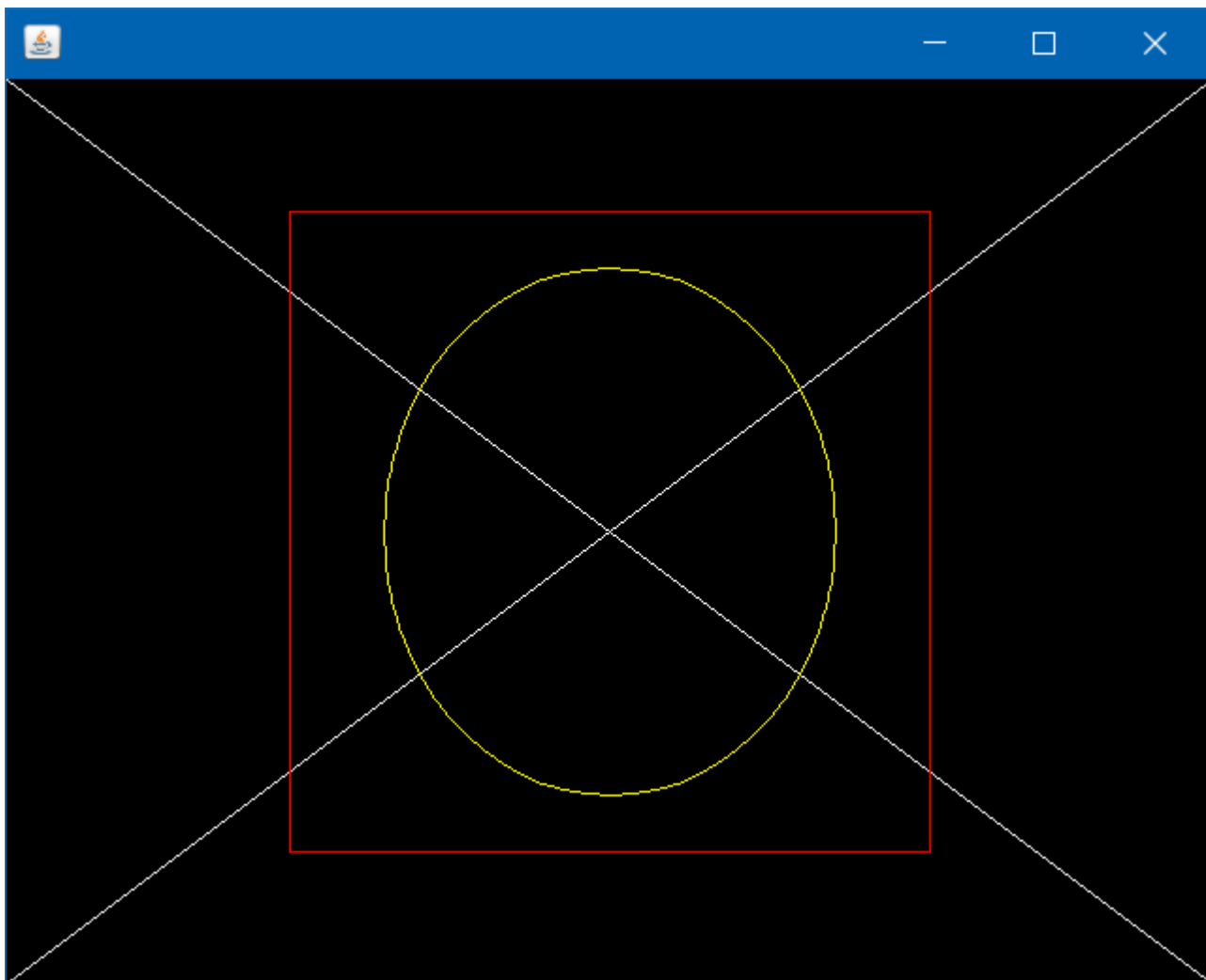


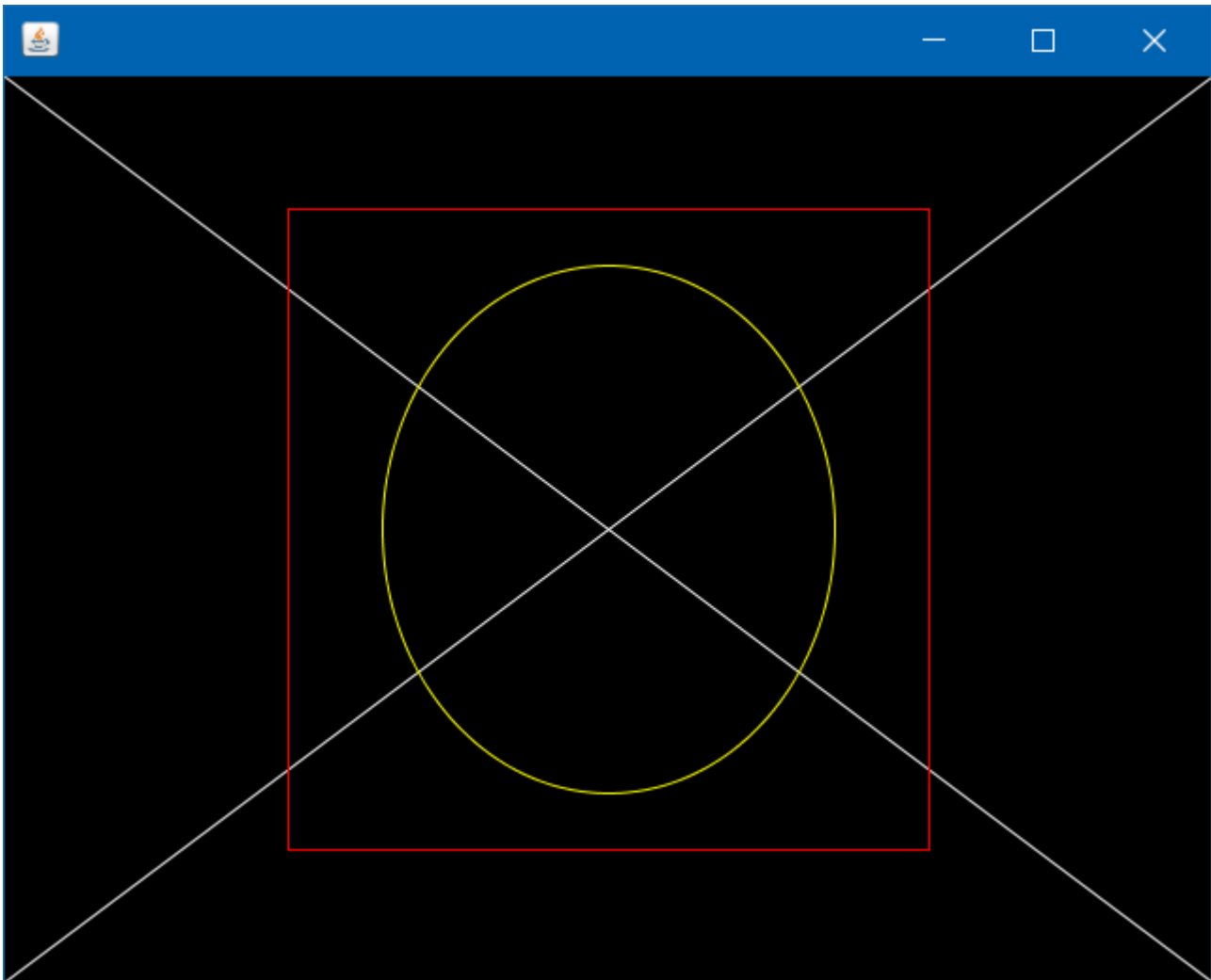
```
public static void saveImage(String destination) throws IOException {
    // method implemented in "Creating a simple image Programmatically and displaying it"
    example
    BufferedImage img = createSampleImage();

    // 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);
}
```





## 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"));
```





## 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/zh-TW/java/topic/5482/>

# 70: BufferedWriter

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

- `BufferedWriter BufferedWriter.write() BufferedWriter BufferedWriter.close() IOException` ◦
- `BufferedWriter(Writer) IOException` ◦ `FileWriter(File)FileNotFoundException IOException` ◦  
`IOExceptionFileNotFoundException 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](https://docs.oracle.com/javase/7/docs/api/java/nio/file/)

```
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/zh-TW/java/topic/3063/bufferedwriter>

# 71:

- C / C ++Java。 ;。
- byte-128+127。 0xFF (b & 0xFF) 。

## Examples

/

。 。

### 3RGBInt。

```
// 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)
};
```

。

### n i byteshortcharintlong

```
(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 FIRST_BIT | THIRD_BIT | FIFTH_BIT | BIT_55 ◦
```

## 2

$2^2 \wedge n_n \circ$

```
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
```

$2 \circ$

```
int twoExp4 = 0x10; //hexadecimal
int twoExp5 = 0x20; //hexadecimal
int twoExp6 = 64; //decimal
...
int twoExp31 = -2147483648; //is that a power of 2?
```

## 2int

```
int pow2(int exp){
    return 1<<exp;
}
```

## 2

$x \cdot 2^{x-1} = 41003011 = 2$

```
boolean isPowerOfTwo(int x)
{
    return (x != 0) && ((x & (x - 1)) == 0);
}
```

## READ WRITEEXECUTE ◦ 07.4

RESOURCE = READ WRITE EXECUTE12

RESOURCE = 0100 0110 0101 = 4 6 512

1212

0100 0110 0101

0000 0000 0111

0000 0000 0101 = 5

## RESOURCEEXECUTE ◦ RESOURCE READ

0100 0110 0101

0111 0000 0000

0100 0000 0000 = 1024

1024.READ ◦ ◦ READ8READ

0100 0000 0000 >> 8 => 0000 0000 01000

4READ ◦

## RESOURCE READ WRITE EXECUTERESOURCE

◦ 4

READ = 0001

WRITE = 0100

= 0110

READ | WRITE | EXECUTE ◦ READ ||

0001 | 0100 | 0110 => 0111

0001 0100 0110

**READ8 WRITE4 PERMISSIONS. RESOURCE12. ◦**

<< 8| WRITE << 4|

0000 0000 0001 << 8

0001 0000 00008

0000 0000 0100 << 4

0000 0100 00004

0000 0000 0001

;

0001 0000 0000

0000 0100 0000

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
```

BitSetCloneableSerializable 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。  $\text{int}[-2^{31} - 1, 2^{31}] - 10$ 。

>><<。 ◦

◦  $\text{int}[0, 2^{32} - 1]$  signed signed。

◦ 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 (	11111111111111111111111111111110)

<<<

◦ ◦ 2。

◦

<https://riptutorial.com/zh-TW/java/topic/1177/>

---

# 72: Fork / Join

## Examples

### JavaFork / Join

Javafork / join。 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();
```

Fork / Join <https://riptutorial.com/zh-TW/java/topic/4245/fork---join>

# 73: JAXP API XML

XMLXML " "。

JavaXMLAPIJAXPXMLJava API。Java 1.4JAXP v1.1JAXPJava。Java 8JAXP 1.6。

APIXML

- DOM
- SAXXMLAPI
- StAXXMLStreaming API

## DOM

DOMW3C DOMXML。JAXPDOM3。

XML"。APINodeNodeNode NodeNode NodeNode Text。XMLElement Node。

DOMXML".....。DOMXML。XMLXML。

## SAX

SAX APIAPIXML。XML " "。 ..

SAX API"SAX ParserXML ContentHandler XML。XML。

SAXDOMContentHandler。 "/ XML"DOMNodeSAX。

## StAX

StAX APISAX APIXMLStAXSAX。 SAX ParserContentHandler。 Stax/XML"。

APIXMLStreamReader 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>
```

## StringDOM

```
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("",
""));
                        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, "evey", ATTRIBUTES)` throws `IllegalStateException`.

2. `xmlStreamReader.getTextContent()` returns `START_ELEMENT<book>`.

...`"BookParserBookParser"` returns `START_ELEMENTXMLEND_ELEMENT`.

Stack

**JAXP API XML** <https://riptutorial.com/zh-TW/java/topic/3943/jaxp-apixml>

---

# 74: static

- `public static int myVariable; //`
- `public static myMethod{} //`
- `public static final double MY_CONSTANT; //`
- `MY_CONSTANT; //`

## Examples

### static

`static` ◦ `staticALL_CAPS` ◦

```
static E STATIC_VARIABLE_NAME
```

`staticfinal`

### 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
```

### static

`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
    }

}

```

static <https://riptutorial.com/zh-TW/java/topic/2253/static>

# 75:

Throwable throw try...catch try...catch ◦

- void someMethodSomeException {} //SomeException

- {

```
someMethod(); //code that might throw an exception
```

```
}
```

- catchSomeException 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 ◦ trytry...catch...finallytry-with-resources ◦

```
try {  
    doSomething();  
} catch (SomeException e) {  
    handle(e);  
}  
// next statement
```

try...catch

- try ◦
- trytry...catch ◦
- try ◦
  - SomeException ◦
  - catch
    - e ◦
    - catch ◦
    -

- 
- try...catch◦
- ◦

try...catchcatch◦

```
try {
    doSomething();
} catch (SomeException e) {
    handleOneWay(e)
} catch (SomeOtherException e) {
    handleAnotherWay(e);
}
// next statement
```

catch◦ try...catch◦ catchcatch◦

“”catch◦

```
try {
    throw new RuntimeException("test");
} catch (Exception e) {
    System.out.println("Exception");
} catch (RuntimeException e) {
    System.out.println("RuntimeException");
}
```

“Exception”“RuntimeException”◦ RuntimeExceptionException catch ◦ catch◦

catchcatchlast◦ catchJava◦

## Java SE 7

Java SE 7catch◦ | ◦

```
try {
    doSomething();
} catch (SomeException | SomeOtherException e) {
    handleSomeException(e);
}
```

◦ catch◦

◦ e ◦ eunion◦ catche union◦

```
public void method() throws IOException, SQLException
{
    try {
        doSomething();
    } catch (IOException | SQLException e) {
        report(e);
        throw e;
    }
}
```

IOExceptionSQLExceptionException ◦ reportreport (Exception) ◦ throwIOExceptionSQLException ◦  
methodthrows IOException, SQLExceptionthrows Exception ◦ - [ThrowableExceptionError](#)  
[RuntimeException](#) ◦

```
public void checkNumber(int number) throws IllegalArgumentException {  
    if (number < 0) {  
        throw new IllegalArgumentException("Number must be positive: " + number);  
    }  
}
```

3◦

- new IllegalArgumentException(...)IllegalArgumentException◦
- throw ...◦

◦ ◦

◦ ◦ ◦ **Java**◦

- checkNumberthrows IllegalArgumentException ◦ IllegalArgumentException;[Java](#) - ◦ [javadoc](#)◦
- throw◦

```
throw new IllegalArgumentException("it is bad");  
return;
```

return◦

messagecause◦ 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);
    }
}

```

trythrow◦ catchthrowIOException◦ ◦ IOExceptioncause **stacktrace**◦

Exception◦ ◦ [RuntimeException](#) Exception◦

- [UnsupportedOperationException](#) -
- [IllegalArgumentException](#) -
- [IllegalStateException](#) - APIAPI
- APIAPIAPI◦
- ◦

[RuntimeException](#)*RuntimeException* Exception

## RuntimeExceptionException

```

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 );
    }
}

```

[...]° °

- [RuntimeException](#)

## try-with-resources

Java SE 7

[try-catch-final](#)finally""° [Java 7 try-with-resources](#)°

[Java 7 java.lang.AutoCloseable try-with-resources](#)° [AutoCloseable](#) ° °

AutoCloseable

```
public void close() throws Exception
```

close()° ° [Autocloseableclose\(\)](#)ExceptionException °

[Java](#)AutoCloseable °

- [InputStream](#) [OutputStream](#)
- [Reader](#) [Writer](#)
- [SocketServerSocket](#)
- [Channel](#)
- [JDBC](#)Connection StatementResultSet°

°

## try-with-resource

[try-with-resource](#)[try-catch](#) [try-finally](#)[try-catch-finally](#)° ""°;catchfinally°

```
try (PrintStream stream = new PrintStream("hello.txt")) {
    stream.println("Hello world!");
}
```

try(...)° [streamPrintStream](#) °

try° [stream.close\(\)](#)° [close\(\)](#)°

## try-with-resource

[catchfinally](#)[try-with-resources](#)[Java 7 try-catch-finally](#)° [catchPrintStreamFileNotFoundException](#)

```
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` catch<sup>o</sup> finally *try-catch-finally*<sup>o</sup>.

- catchfinally<sup>o</sup>
- catch<sup>o</sup>
- catch<sup>o</sup>

◦ *try-with-resources*<sup>o</sup>.

```

try (InputStream is = new FileInputStream(file1);
     OutputStream os = new FileOutputStream(file2)) {
    // Copy 'is' to 'os'
}

```

◦ try isos<sup>o</sup>

- ◦
- ◦
- ◦

isisos<sup>o</sup>

## try-with-resource try-catch-finally

Java *try-catch-finally try-with-resource*<sup>o</sup> JLS<sup>o</sup>.

```

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**<sub>try</sub>**primaryException**Java<sup>o</sup>

*try-with-resources*<sup>o</sup>

```

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<sup>o</sup> <sup>o</sup>

**stacktrace**<sub>printStackTrace()</sub><sup>o</sup>

```

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 <sup>o</sup> printStackTrace(PrintStream)printStackTrace(PrintWriter)Stream



Writer ◦

1. ◦ toString();

```
// Print exception and stacktrace
System.out.println(ex);
ex.printStackTrace();
```

2. Stacktrace; - ◦ ◦

◦ ◦

```
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 zero”◦

javadoc

◦ “”◦

“/ by zero”◦

3◦

- 
- 
-

◦ Test.bar Test.java9◦

```
return a / b;
```

bb◦ ◦

stacktrace:bar() Test.java3foo() foo()Main.main()◦

◦

- “OuterClass \$ InnerClass”◦
- “OuterClass \$ 1”“OuterClass \$ 2”◦
- “”◦
- “”◦

Java◦

Java SE 1.4

◦

```
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 }
```

stacktrace

```
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◦ cause◦ “..... N more”N◦

causenum “”◦ “”◦

cause [Java 1.4.0 Throwable API](#) ◦ [printStackTrace](#) ◦

## stacktraceString

[Java String](#) ◦ [OutputStreamWriter printStackTrace\(...\)](#) ◦

### [Apache Commons Guava String](#)

```
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();
}
```

[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](#) [AssertionError](#) ◦ [catch](#) ◦

Guava `Uninterruptibles`; `Uninterruptibles.sleepUninterruptibly()` `InterruptedException` ◦ **try-catch**  
**sleep** ◦

◦ `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 -

**JavaException** ◦

- `java.lang.Throwable` - ◦ ◦
  - `java.lang.Exception` - ◦
    - ◦
    - `java.lang.RuntimeException` - ◦
      - ◦
  - `java.lang.Error` - “” ◦

1. ◦

2. `Throwable` `Exception` `RuntimeException` `abstract` ; - `Throwable` `Exception` `Error` `RuntimeException`

◦

3. **JVM** `Error` ◦

4. `Throwable` ◦ **Java** `Error` `Exception` `Throwable` ◦

◦ ◦

**Java** ◦ **Java**

• ◦ `IOException` | / `O` ◦ ◦

• ◦ ◦

“thrown” `throw` `throw` ◦ “propagated” ◦ ◦ ◦

- throws◦ throws◦
- ◦
- ◦ ◦

◦  
◦ ◦  
◦ ◦

```
// 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.
}
```

“◦

```
// 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);
    }
}

```

Input two integers

7 0

```
Exception in thread "main" java.lang.ArithmeticException: / by zero
    at Division.main(Division.java:14)
```

- **Java. *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. ◦ 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 **Exception** `ArithmeticException` ◦
3. e catch ◦
4. try catch try-catch

◦

◦ null ◦ ◦

Java -

## 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;
    }
}
```

7try / catchfinally◦ return 7; try / catch◦

catchfinallycatchfinally◦

“0”“1”◦

```
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;
        }
    }
}
```

◦



# callstack

## 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()◦ stacktraces◦ public◦ Throwable◦ 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◦ close()◦ FileWriter◦ IOException

- ◦ try-with-resources◦ ◦

try° °

getSuppressedExceptions°

## try-finallytry-catch-finally

try...catch...finally° finally° °

try...finally

```
try {
    doSomething();
} finally {
    cleanUp();
}
```

try...finallytry...finally

- try°
- try
  - finally°
  - finally°
  - try...finally°
- try
  - finally°
  - finally°
  - °

finally° System.exit(int) JVM° finally;°

## try-catch-

catchfinally° °

```
// 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” readerfinally◦
- new FileReader(...) catchIOException◦
- finallyreader.close() trycatch◦
- readernull◦
- reader.close()◦ ◦

## Java SE 7

Java 7try-with-resources ◦

## 'throws'

Java ◦ throws◦

```
public class OddNumberException extends Exception { // a checked exception
}

public void checkEven(int number) throws OddNumberException {
    if (number % 2 != 0) {
        throw new OddNumberException();
    }
}
```

throws OddNumberExceptioncheckEvencheckEvenOddNumberException ◦

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. ◦
2. ◦ throws◦

APIjavadocthrowsIDE“”◦

throws◦ ◦ ◦

```
@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 IOException // ERROR
```

overridden.

<https://riptutorial.com/zh-TW/java/topic/89/>

## 76:

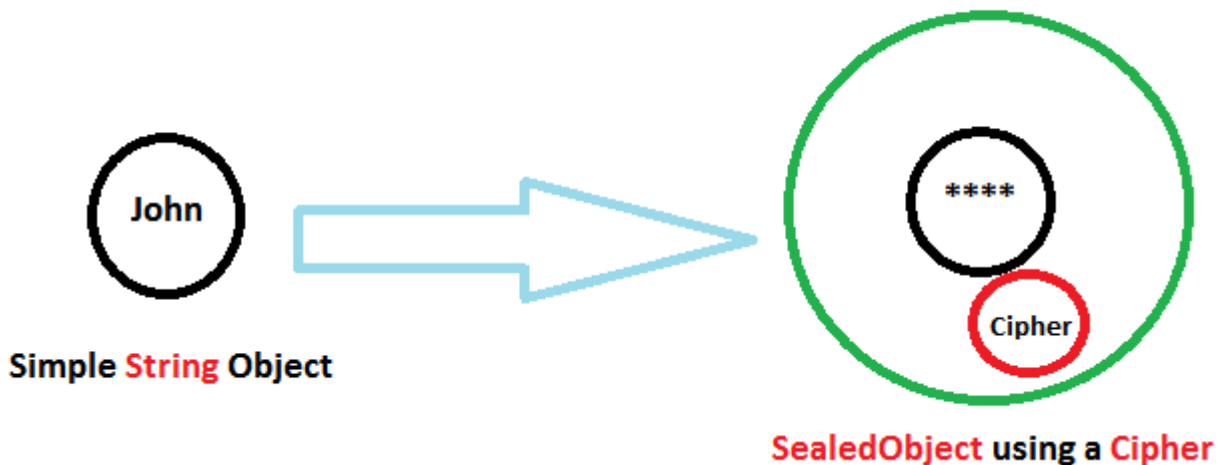
- `SealedObject sealedObject = new SealedObject(objcipher);`
- `SignedObject signedObject = new SignedObject(objsigningKeysigningEngine);`

## Examples

### `SealedObject` `javax.crypto.SealedObject`

◦

`SerializableSealedObject` `""AESDES` ◦ ◦ ◦

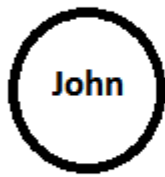


```
Serializable obj = new String("John");
// Generate key
KeyGenerator kgen = KeyGenerator.getInstance("AES");
kgen.init(128);
SecretKey aesKey = kgen.generateKey();
Cipher cipher = Cipher.getInstance("AES");
cipher.init(Cipher.ENCRYPT_MODE, aesKey);
SealedObject sealedObject = new SealedObject(obj, cipher);
System.out.println("sealedObject-" + sealedObject);
System.out.println("sealedObject Data-" + sealedObject.getObject(aesKey));
```

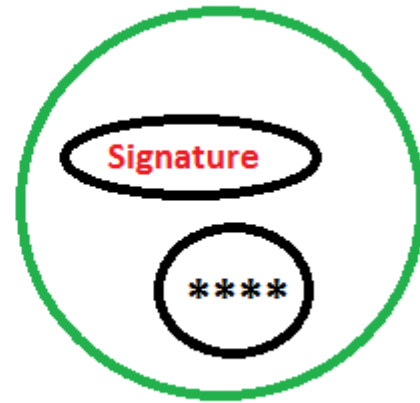
### `SignedObject` `java.security.SignedObject`

`SignedObject` ◦

`SignedObjectSerializable` ◦



Simple **String** Object



**SignedObject** using **Signature**

```
//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/zh-TW/java/topic/5528/>

# 77:

Deque。

deque“”“deck”。

DequeDequeDeque。

DequeStackQueue

Deque。

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

dequeFIFO。

DequeLIFO。

。

## 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;  
}
```

<https://riptutorial.com/zh-TW/java/topic/10156/>



# 78: SET

ListSetjavaListSet in Java

## Examples

```
import java.util.ArrayList;
```

```
import java.util.HashSet; import java.util.List; import java.util.Set;
```

```
public class SetAndListExample {public static void mainString [] args{System.out.println“List  
example .....”; List list = new 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/zh-TW/java/topic/10125/set>

# 79:

Java 8+ Object. [JLS§9.8](#).

## Examples

### Java

	ř	
		<a href="#">BooleanSupplier</a>
	INT	<a href="#">IntSupplier</a>
		<a href="#">LongSupplier</a>
		<a href="#">DoubleSupplier</a>
T		<a href="#">&lt;T&gt;</a>
T	ř	<a href="#">UnaryOperator &lt;T&gt;</a>
T	[R	<a href="#">&lt;TR&gt;</a>
T		<a href="#">&lt;T&gt;</a>
T	INT	<a href="#">ToIntFunction &lt;T&gt;</a>
T		<a href="#">ToLongFunction &lt;T&gt;</a>
T		<a href="#">ToDoubleFunction &lt;T&gt;</a>
TT	ř	<a href="#">BinaryOperator &lt;T&gt;</a>
TU		<a href="#">BiConsumer &lt;TU&gt;</a>
TU	[R	<a href="#">&lt;TUR&gt;</a>
TU		<a href="#">BiPredicate &lt;TU&gt;</a>
TU	INT	<a href="#">ToIntBiFunction &lt;TU&gt;</a>
TU		<a href="#">ToLongBiFunction &lt;TU&gt;</a>
TU		<a href="#">ToDoubleBiFunction &lt;TU&gt;</a>

Tint		ObjIntConsumer <T>
T		ObjLongConsumer <T>
T		ObjDoubleConsumer <T>
INT		IntConsumer
INT	[R	IntFunction <R>
INT		IntPredicate
INT	INT	IntUnaryOperator
INT		IntToLongFunction
INT		IntToDoubleFunction
intint	INT	IntBinaryOperator
		LongConsumer
	[R	LongFunction <R>
		LongPredicate
	INT	LongToIntFunction
		LongUnaryOperator
		LongToDoubleFunction
		LongBinaryOperator
		DoubleConsumer
	[R	DoubleFunction <R>
		DoublePredicate
	INT	DoubleToIntFunction
		DoubleToLongFunction
		DoubleUnaryOperator
		DoubleBinaryOperator

<https://riptutorial.com/zh-TW/java/topic/10001/>

# 80:

- Java◦ ◦
- 
- =
- 
- 
- 
- ◦ abstract
- ◦ java◦
- ◦
- =

## 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/zh-TW/java/topic/9204/>

# 81:

javapackage ◦ ◦ /◦ ◦ java > 1.java.util> 2.java.lang> 3.java.io ◦

◦

package ◦ ◦

◦

## Examples

### Packages

#### First 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/zh-TW/java/topic/8273/>



# 82:

byte short int long char boolean floatdoubleJava

- `int aInt = 8; //int`
- `int hexInt = 0x1a; // = 26;0x`
- `int binInt = 0b11010; // = 26;;0b`
- `long goodLong = 10000000000L; //int L. ""`
- `double aDouble = 3.14; //double`
- `float aFloat = 3.14F; //double""F`

Java8 boolean byte short char int long floatdouble ◦ ◦ Java;/String ClassThrowable◦

int shortshortint bytebyte charchar ◦ ◦

```
byte a = 1;
byte b = 2;
byte c = (byte) (a + b);
```

◦

## Java§2.11.1

byte shortJava int◦ boolean char int◦ [...]◦ boolean byte char short int◦

Java ◦ Java byte ◦ [...]◦

## Examples

### int

int Integer◦

java API "Integerint.Integerint. "

int32◦  $-2^{31}$   $2^{31} - 1$ ◦

```
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  $2^{15}$  -132,767

```
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 ◦  $-2^{63}$   $2^{63}$  -  $10^{24}$  - 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
```

long *0L*

```
long defaultLong;    // defaultLong == 0L
```

long *"L"*

```
2L == 2l;           // true
```

Java-128127Integer。 [https://blogs.oracle.com/darcy/entry/boxing\\_and\\_caches\\_integer\\_valueof](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
```

## 2Object LongJava 1.7

```
Long val3 = 128L;
Long val4 = 128L;

System.out.println(Objects.equal(val3, val4)); // true
```

**2==。**

boolean truefalse

```
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<sup>8</sup> -2<sup>7</sup> -128<sup>7</sup> - 1127

```
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<sup>0</sup>

```
byte defaultByte;    // defaultByte == 0
```

float<sup>32</sup> IEEE 754<sup>0</sup> ◦ float f<sup>0</sup>

```
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
```

◦

◦ 34.600002<sup>0</sup>

```
// 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
```

floatfloatdouble ◦ BigDecimal ◦

float **0.0f** ◦

```
float defaultFloat; // defaultFloat == 0.0f
```

float **1** ◦

Float.POSITIVE\_INFINITY Float.NEGATIVE\_INFINITY Float.NaNfloat ◦ NaN**2** ◦ 0f-0f==

```
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 **64IEEE 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
```

doublefloatdouble ◦ BigDecimal

double **0.0d**

```
public double defaultDouble; // defaultDouble == 0.0
```

Double.POSITIVE\_INFINITY Double.NEGATIVE\_INFINITY Double.NaNdouble ◦ 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

char16Unicode◦

```
char myChar = 'u';
char myChar2 = '5';
char myChar3 = 65; // myChar3 == 'A'
```

\u0000 0 \uffff 65,535◦

char\u0000 ◦

```
char defaultChar; // defaultChar == \u0000
```

' value 'char

```
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
```

Unicodechar ◦

```
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);
}
```

Java2◦

n

$n-1x \circ s \circ$

$x - s * 2^{n-1}$

$11 \ 2^{n-2} + 2^{n-3} + \dots + 2^1 + 2^0 = 2^{n-1} - 12 \ 2^{n-2} + 2^{n-3} + \dots + 2^1 + 2^0 = 2^{n-1} - 1 - 2^{n-1} \ s =$

$$1; x = 02^{n-1} - 1s = 0; x = 2^{n-1} - 1.$$

$$v1 = x1 - s1 * 2^{n-1}$$

$$v2 = x2 - s2 * 2^{n-1}$$

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	*	
1	1		$x1 + x2 - 2^n < -2^{n-1};$
1	1		$(x1 + x2) \bmod 2^{n-1} - 2^{n-1} = (x1 + x2 - 2^{n-1}) - 2^{n-1}$ $= (x1 - s1 * 2^{n-1}) + (x2 - s2 * 2^{n-1})$ $= v1 + v2$

1.100.

iint

$$(\sim i) + 1$$

0 byte

011111111 ◦ 110000000 9◦ byte 800000000

00000000	-011111111
11111111	1
8	00000000-00

			/
			1 / 16
			1 / 16
			2 / 16
			2 / 16
INT			4 / 16
			8 / 16
			4 / 16
			8 / 16

8816。 48JVMJVM。

◦ float[5]32;5Float[5]11264152。

◦ ◦ Integer-128+127◦ ◦

valueOf(primitive)◦ ◦ ""valueOfnew ◦ new◦ new◦

Java◦ Unicodechar◦ 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

N / A			



	8	$-2^7 - 1$	0
		-128+127	
	16	$-2^{15} - 1$	0
		-32,768+32,767	
INT	32	$-2^{31} - 1$	0
		-2,147,483,648+2,147,483,647	
	64	$-2^{63} - 1$	0L
		-9,223,372,036,854,775,808+9,223,372,036,854,775,807	
	32	1.401298464e-453.402823466e + 38	0.0F
	64	4.94065645841246544e-324d1.79769313486231570e + 308d	0.0D
	16	$2^{16} - 1$	0
		065,535	

1. Java `byte` long IEEE 754。
2. Java `short` long 。
3. ; 。
4. `char` Unicode / UTF-16 。
5. `boolean` Java 。

<https://riptutorial.com/zh-TW/java/topic/148/>

# 83:

Java ◦

lazySet	
compareAndSet	
getAndSet	

CAS ◦ [AtomicInteger Unsafe.getAndSet](#)

## Examples

◦ ◦

◦

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
```

◦

floatdouble ◦ Float.floatToIntBits(float) Float.intBitsToFloat(int) float  
Double.doubleToLongBits(double) Double.longBitsToDouble(long) ◦

sun.misc.Unsafe sun.misc.Unsafe ◦ intlong ◦ [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. synchronizedCounters◦
2. number◦
3. ◦

1. ◦

- ◦
- ◦ ◦
- “”◦ ◦

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;
    }
}

```

AtomicInteger[] counters = new AtomicInteger[10];

synchronized { incrementAndGet().get(); }

- “”。

- T“”T。

AtomicInteger。

- IntelCAS。

AtomicXxxAPI。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“”。CAS。

JVMCAS。JVMCAS。CAS。HotSpot。

<https://riptutorial.com/zh-TW/java/topic/5963/>

# 84:

## Examples

```
Object obj = new Object(); // Note the 'new' keyword
```

- Object◦
- obj◦
- Object()Object◦
- ◦
- Object()◦
- obj ◦

```
int i = 10;
```

10i ◦

.

```
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/zh-TW/java/topic/1046/>

# 85:

## Examples

`java.lang.ref`

Java

- 
- 
- 
- 

### 1.

◦

```
MyObject myObject = new MyObject();
```

holder ◦ 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

“◦◦ Java◦ “ - Oracle [PhantomJavadoc](#) ◦

#### Phantom Reference

```
PhantomReference myObjectRef = new PhantomReference(MyObject);
```

<https://riptutorial.com/zh-TW/java/topic/4017/>

# 86: API

JVM。 [Java Reflection API](#)。 ◦

Java [The Reflection API](#)

Java。 ◦

## Examples

Reflection API。 ◦

```
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

```

o

Stringo

```

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!"

```

## ClassgetField

```
// 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);
```

## ◦ Fieldget

```
// 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);
```

## ClassgetDeclaredField

```
class HackMe extends Hacked {
    public String iAmDeclared;
}

class Hacked {
    public String someState;
}
```

HackMe#iAmDeclaredfield◦ HackMe#someStateHacked◦

ClassConstructor

```
Class myClass = ... // get a class object
Constructor[] constructors = myClass.getConstructors();
```

constructorsConstructor◦

◦ Integer

```
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◦ Reflection API◦

```
Compass.class.isEnum();
```

true.

```
Object[] values = Compass.class.getEnumConstants();
```

Compass.values.

```
for(Field f : Compass.class.getDeclaredFields()){
    if(f.isEnumConstant())
        System.out.println(f.getName());
}
```

.

## Class

String Class.forNameClass

```
Class clazz = null;
try {
    clazz = Class.forName("java.lang.Integer");
} catch (ClassNotFoundException ex) {
    throw new IllegalStateException(ex);
}
```

## Java SE 1.2

forNameClassLoader

```
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. 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) ◦

## Class.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"});
    }
}

```

◦

## ReflectionAPI ◦ ◦ ◦

◦

```

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!

```

## Java

JDKReflection API ◦ ◦

Reflection ◦ Java SecurityManager ◦ [java.lang.System](#)

```
private static volatile SecurityManager security = null;
```

## System

```
for(Field f : System.class.getDeclaredFields())  
    System.out.println(f);
```

fieldFilterMapsun.reflect.ReflectionSystem.class SecurityManager

Java StringJVMString JVM String.intern() "hello"

" valueString

```
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**

## Integer

```
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/zh-TW/java/topic/629/api>

## 87:

- `MyClassComparable <MyClass >`
- `MyComparatorComparator <SomeOtherClass >`
- `public int compareToMyClass other`
- `public int compareSomeOtherClass o1SomeOtherClass o2`

`doublecompareTo(..)`

```
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
}
```

### Java 1.2 `Comparable<T>Comparable ° Comparable<T>°`

```
public class A implements Comparable<A>
```

°

`Comparable<T> Comparator<T> Comparable<T> ° Comparator; Comparable°`

## Examples

### Comparable

`Person° equalshashCode°`

```
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, 3, 5, 4, 21, 2, 3, 4, 5 ◦ **IntegersJava** ◦ **Person** 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 Comparator<T>Person ◦ circle, square, rectangle, triangle, hexagon ◦ **Java** ◦

```

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
}

```

/

```

//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

## Lambda

Java 8lambda

```

//Lambda
Collections.sort(people, (p1, p2) -> { //Legal
    //Method code....
});

```

**Comparator** `lastNamefirstName`

```

Collections.sort(people, Comparator.comparing(Person::getLastName)
                .thenComparing(Person::getFirstName));

```

reversedMethod reversedMethod°

## compareTo

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 thisother comparecomparet1t2 comparethis °

◦ **compareTo** ◦ ◦ ab

- 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)
- ab0 °

Collections.sort()

- List<T>T**Comparable**compareTo()°
- List**Comparator**°

## ComparablePerson

```
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();
    }
}

```

## List<Person> 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);

```

## ComparableList

```

//-- 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.Entry

### Java SE 8

```

Map<String, Integer> numberOfEmployees = new HashMap<>();
numberOfEmployees.put("executives", 10);
numberOfEmployees.put("human resources", 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.comparing(Person::getName)
```

Person ◦ longintdouble ◦

```
Comparator.comparingInt(Person::getAge)
```

reversed()

```
Comparator.comparing(Person::getName).reversed()
```

```
Comparator.comparing(Person::getLastName).thenComparing(Person::getFirstName)
```

◦ ◦

<https://riptutorial.com/zh-TW/java/topic/3137/>



# 88:

- [=];
- [=];
- [=];
- [=];
- {
- {

## Java

- 
- - public *package-private* ◦
- - public private protected *package-private* ◦
- public ◦ ◦
- public *no package-private* ◦ private protected ◦ private ◦ protected *package-private* ◦
- 

public	ÿ	ÿ	ÿ	ÿ
protected	ÿ	ÿ	ÿ	ñ
	ÿ	ÿ	ñ	ñ
private	ÿ	ñ	ñ	ñ

## 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 XY ◦ - MyInterfaceMyInterface ◦

◦

## Test◦

```
public class Test{
    public int number = 2;

    public Test(){

    }
}
```

◦ numberpublic ◦

```
public class Other{

    public static void main(String[] args){
        Test t = new Test();
        System.out.println(t.number);
    }

}
```

private◦ public◦

```
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** “[]”◦ `javax.swing`

```
package javax.swing;
public abstract class JComponent extends Container ... {
```

```

...
static boolean DEBUG_GRAPHICS_LOADED;
...
}

```

DebugGraphicsDEBUG\_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◦

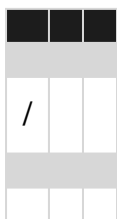
```

package some.other.pack;
import com.stackexchange.docs.MyClass;
public class SubClass extends MyClass{
    public SubClass(){
        super();
        System.out.println(super.variable);
    }
}

```

protectedprotected◦

◦



private protected ◦ [Java 1.0](#) ◦

<https://riptutorial.com/zh-TW/java/topic/134/-->

# 89:

`Optional``null` ◦ `isPresent()``true` `get()` ◦

`orElse()` `ifPresent()` ◦

- `Optional.empty()`//`Optional` ◦
- `Optional.of(value)`//`nullOptional` ◦ `nullPointerException` ◦
- `Optional.ofNullable(value)`//`Optional``null` ◦

## Examples

### Optional

`Optional.get()` `NoSuchElementException` ◦ `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` `Optional` `Optional` ◦ `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"
```

## Optional.mapnullmap。 Null-safe 。

```
String value = foo.getBar().getBaz().toString();
```

getBar getBaztoStringNullPointerException 。

OptionaltoString()

```
String value = Optional.ofNullable(foo)
    .map(Foo::getBar)
    .map(Bar::getBaz)
    .map(Baz::toString)
    .orElse("");
```

null。

◦ null。

```
Optional.ofNullable(foo)
    .map(Foo::getBar)
    .map(Bar::getBaz)
    .map(Baz::toString)
    .ifPresent(System.out::println);
```

orElseThrow() Optional 。

```
Optional optional = Optional.of("something");

return optional.orElseThrow(IllegalArgumentException::new);
// returns "something" string
```

```
Optional optional = Optional.empty();

return optional.orElseThrow(IllegalArgumentException::new);
// throws IllegalArgumentException
```

lambda

```
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"
```

[Optional](#)[getValueThatIsHardToCalculate](#)()◦

## FlatMap

## [flatMap](#) ◦ javadoc

`map(Function) Optional.flatMapOptional` ◦

`OptionalOptional.flatMapOptionals` ◦

```
public class Foo {
    Optional<Bar> getBar() {
        return Optional.of(new Bar());
    }
}

public class Bar {
}
```

`Optional.map Optional ;Optional<Optional<Bar>>` ◦

```
Optional<Optional<Bar>> nestedOptionalBar =
    Optional.of(new Foo())
        .map(Foo::getBar);
```

`Optional.flatMap Optional ;Optional<Bar>` ◦

```
Optional<Bar> optionalBar =
    Optional.of(new Foo())
        .flatMap(Foo::getBar);
```

<https://riptutorial.com/zh-TW/java/topic/152/>

# 90:

gettersetter;Java°

## Examples

### GettersSetter

OOP° ° private GettersSetters/°

```
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;
    }
}
```

° **GettersSetter**°

getXxx() xxxsetXxx() xxx°

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 GettersSetterJava Bean [Property](#)°

### settergetter



## SettersGetters

```
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) 2null ◦ setter\_name public\_name ◦ getter

```
public String getName(){  
    if(name.length()>16)  
        return "Name is too large!";  
    else  
        return name;  
}
```

getName() name 16 "Name is too large" ◦ ◦

## Javagettersetter

```
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;  
}
```

◦ ◦

- “intint” ◦
- “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.outout ◦ ◦

1. ◦ **getter / setter** ◦ // API ◦

2. ◦ ◦ **99.9** ◦

◦ **getter / setter** ◦

<https://riptutorial.com/zh-TW/java/topic/3560/>

# 91:

- public static void mainString [] args

```
ARGS ◦ Java main args null null ◦
```

java Java main args ◦

Java SE ◦

- Java ◦
  - ◦
- StackOverflow “Java” ◦ ◦

## Examples

### GWT ToolBase

GWT ◦

<https://gwt.google.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);
}
}

```

ArgHandlerisRequired() false◦

◦

◦ System.exit(1) shell◦ “myapp”Java◦

◦ 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 ◦

## “flag”

◦

```

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/zh-TW/java/topic/4775/>

---

# 92:

## HashtableJavaMapDictionary

### Examples

```
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);
    }
}
```

<https://riptutorial.com/zh-TW/java/topic/10709/>

# 93:

## Examples

[Preferences](#) [PreferenceChangeEvent](#) [NodeChangeEvent](#) ◦

### PreferenceChangeEvent

[PreferenceChangeEventProperties](#) ◦ [PreferenceChangeEvent](#) **S**[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](#)





```

// 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□□.

□□□□□□□□□□□□□□□□ □

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

```







## 94: 测试

测试

测试是软件开发中不可或缺的一部分。Java 测试框架可以帮助开发者验证代码的正确性。测试可以防止代码中的错误，提高代码的质量和稳定性。

测试

## 测试框架

Java 测试框架包括 JUnit、JUnit4、JUnit5、TestNG 等。

JUnit

JUnit4 - JUnit4 测试框架; 测试。

测试框架

TestNG

## 测试库

测试库

Mockito - 测试; 测试。测试框架/测试。

JBehave - BDD 测试。测试框架/测试。测试; 测试。

## Examples

测试

测试。测试框架。测试框架。

测试。测试框架。

测试

```

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;
    }
}
}
```

Consumer 的 getVal() 方法调用了 DataReader 的 readVal() 方法。DataReader 的 readVal() 方法调用了 File 的 readLines() 方法。

Capitalizer 的 getVal() 方法调用了 DataReader 的 readVal() 方法。DataReader 的 readVal() 方法调用了 File 的 readLines() 方法。

```
// Another team's module.
class Capitalizer {
    ...

    public String getVal() {
        return dr.readVal().toLowerCase();
    }
}
}
```

Consumer 的 getVal() 方法调用了 DataReader 的 readVal() 方法。DataReader 的 readVal() 方法调用了 File 的 readLines() 方法。

Capitalizer 的 getVal() 方法调用了 DataReader 的 readVal() 方法。DataReader 的 readVal() 方法调用了 File 的 readLines() 方法。

Consumer 的 getVal() 方法调用了 DataReader 的 readVal() 方法。DataReader 的 readVal() 方法调用了 File 的 readLines() 方法。



Consumer 的 getVal() 方法调用了 DataReader 的 readVal() 方法。DataReader 的 readVal() 方法调用了 File 的 readLines() 方法。

Capitalizer 的 getVal() 方法调用了 DataReader 的 readVal() 方法。DataReader 的 readVal() 方法调用了 File 的 readLines() 方法。

Consumer 的 getVal() 方法调用了 DataReader 的 readVal() 方法。DataReader 的 readVal() 方法调用了 File 的 readLines() 方法。



## 問題

この問題は、`Capitalizer` クラスの `getVal()` メソッドが、`toUpperCase()` と `toLowerCase()` の両方を呼び出すことに起因するバグです。

この問題を修正するには、`Capitalizer` クラスの `getVal()` メソッドを、`toUpperCase()` と `toLowerCase()` の両方を呼び出すのではなく、`toUpperCase()` のみまたは `toLowerCase()` のみを呼び出すように修正する必要があります。

この問題を修正するには、`Capitalizer` クラスの `getVal()` メソッドを、`toUpperCase()` のみまたは `toLowerCase()` のみを呼び出すように修正する必要があります。

## 解決

この問題を修正するには、`Capitalizer` クラスの `getVal()` メソッドを、`toUpperCase()` のみまたは `toLowerCase()` のみを呼び出すように修正する必要があります。

この問題を修正するには、`Capitalizer` クラスの `getVal()` メソッドを、`toUpperCase()` のみまたは `toLowerCase()` のみを呼び出すように修正する必要があります。

この問題を修正するには、<https://riptutorial.com/zh-TW/java/topic/8155> を参照してください。

95:

Singleton Design Patterns Singleton

### Examples

#### Enum Singleton

Java SE 5

```
public enum Singleton {
    INSTANCE;

    public void execute (String arg) {
        // Perform operation here
    }
}
```

JVM Singleton Joshua Bloch Effective Java

#### Singleton

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;
    }
}
```



```
}
```

Java SE 5 开始引入 Enum 枚举。Java 5 引入了 Enum 枚举。

### Singleton 和 Enum

```
public class Singleton {
    private static final Singleton INSTANCE = new Singleton();

    private Singleton() {}

    public static Singleton getInstance() {
        return INSTANCE;
    }
}
```

Enum 是 Java 12.4.10 开始引入。

Enum 是 T 的枚举。

- T 的枚举。
- T 的枚举。
- Enum T
- Enum T
- T 的枚举。

getInstance() 返回 Singleton。

### holder 和 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() 返回 INSTANCE。

Bill Pugh 枚举。 [ ]

枚举。

Singleton 枚举 "Hello world!" getMessage() 返回 "Hello world!"。

UppercaseSingleton 和 LowercaseSingleton 返回 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/zh-TW/java/topic/130>

## 96: Java

Java JSR223 Java Specification Request 223

Java Scripting API Java

Scripting API java. .

nashorn JavaScript ECMAScript. . . . .  
Writer

Jython JRuby. eval

. . . . .  
EngineFactory

### Examples

nashorn -scripting javascript

```
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 bufferedReader  
            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 unnecessary
    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();

```

□□□□□□□□□□

```

public static void main(String[] args) {

```

```
JSEngine jse = new JSEngine();
// Create a new batch probably unnecessary
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/zh-TW/java/topic/9926/> java

## 97: MultiThreaded ThreadPoolExecutor

ThreadPoolExecutor

### Examples

```
Runnable
"Fire Forget"
Runnable main
```

### 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>Tmain
```

### 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 AsyncValueTypeCompleter call
INFO: Task 7 is sleeping
Dec 28, 2016 3:07:15 PM AsyncValueTypeCompleter call
INFO: Task 8 is sleeping
Dec 28, 2016 3:07:15 PM AsyncValueTypeCompleter call
INFO: Task 2 is sleeping
Dec 28, 2016 3:07:15 PM AsyncValueTypeCompleter call
INFO: Task 1 is sleeping
Dec 28, 2016 3:07:15 PM AsyncValueTypeCompleter call
INFO: Task 4 is sleeping
Dec 28, 2016 3:07:15 PM AsyncValueTypeCompleter call
INFO: Task 9 is sleeping
Dec 28, 2016 3:07:15 PM AsyncValueTypeCompleter call
INFO: Task 0 is sleeping
Dec 28, 2016 3:07:15 PM AsyncValueTypeCompleter call
INFO: Task 6 is sleeping
Dec 28, 2016 3:07:15 PM AsyncValueTypeCompleter call
INFO: Task 5 is sleeping
Dec 28, 2016 3:07:15 PM AsyncValueTypeCompleter call
INFO: Task 3 is sleeping
Dec 28, 2016 3:07:16 PM AsyncValueTypeCompleter 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 AsyncValueTypeCompleter 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 AsyncValueTypeCompleter 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 AsyncValueTypeCompleter 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 AsyncValueTypeCompleter 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 AsyncValueTypeCompleter 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 AsyncValueTypeCompleter 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 AsyncValueTypeCompleter 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 AsyncValueTypeCompleter call

```



```

        });
        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. Lambda#####lambda#####。
2. #####Lambda#####Callable#####Runnable <T>#####。

#####MultiThreaded#####ThreadPoolExecutor。 <https://riptutorial.com/zh-TW/java/topic/8646/multithreaded#####threadpoolexecutor->

## 98:

`java.util.Map` 接口定义了 `Map` 接口。

`Map` 接口定义了 `Map` 接口；`HashMap` 和 `TreeMap`

`Map` 接口定义了 `Map` 接口。 `Map` 接口定义了 `Map` 接口。

- `Map` 接口定义了 `Map` 接口。
- `Map` 接口定义了 `Map` 接口。 `Map` 接口定义了 `Map` 接口。
- `Map` 接口定义了 `Map` 接口。

`HashMap` 接口定义了 `HashMap` 接口。

`HashMap` 接口定义了 `HashMap` 接口。 `HashMap` 接口定义了 `HashMap` 接口。

- `TreeMap` 接口定义了 `TreeMap` 接口。
- `LinkedHashMap` 接口定义了 `LinkedHashMap` 接口。

### 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` 接口定义了 `Map` 接口。

`put(K key, V value)`

`put` 方法用于向 `Map` 中添加元素。

```
String currentVal;
Map<Integer, String> map = new TreeMap<>();
currentVal = map.put(1, "First element.");
```



## 2. `forEach`

Map Entry 'action'

```
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 Entry

```
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`

Map Entry 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. `remove`

Map Entry

```
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. `replace`

Map Entry new-value

```
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 Entry

```
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 的 `computeIfPresent` 方法。

```

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. `compute`

Map 的 `compute` 方法。

```

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. `merge`

Map 的 `merge` 方法。如果键不存在或值为 null，则添加键值对。如果键存在且值不为 null，则替换值。如果键存在且值为 null，则从 Map 中移除键。

```

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 的 `clear` 方法。

```

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

Map

```
Map<String, Integer> repMap = new HashMap<>();
repMap.put("Jon Skeet", 927_654);
repMap.put("BalusC", 708_826);
repMap.put("Darin Dimitrov", 715_567);
```

Map

```
for (String key : repMap.keySet()) {
    System.out.println(key);
}
```

Map

```
Jon Skeet
BalusC
```

keySet() Set. Set. HashMap.

Map

```
for (Integer value : repMap.values()) {
    System.out.println(value);
}
```

Map

```
715567
927654
708826
```

values() Collection. Collection.

Map

```
for (Map.Entry<String, Integer> entry : repMap.entrySet()) {
    System.out.printf("%s = %d\n", entry.getKey(), entry.getValue());
}
```

Map

```
715567 = 715567
Jon Skeet = 927654
BalusC = 708826
```

entrySet() Map.Entry. Map.Entry.

Map

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###

```

"One" -> 1
"Two" -> 2
"Three" -> 4 //old value 3 was overwritten by new value 4

```

#####Java 8#####Map.merge #####BiFunction#####. 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#####

```

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
}

```



HashMap 的 null 值

```

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.containsKey(key) <=> map.get(key) != null

Map 的迭代器

Map<Integer, Integer> 的迭代器。 @n 的迭代器。

### 1. Iterator 遍历 Map.Entry

```

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. for 遍历 Map.Entry

```

for (Map.Entry<Integer, Integer> pair : map.entrySet()) {
    sum += pair.getKey() + pair.getValue();
}

```

### 3. Map.forEach 在 Java 8 中

```

map.forEach((k, v) -> sum[0] += k + v);

```

### 4. Map.keySet 遍历 for

```

for (Integer key : map.keySet()) {
    sum += key + map.get(key);
}

```

### 5. Map.keySet 遍历 Iterator

```

Iterator<Integer> it = map.keySet().iterator();
while (it.hasNext()) {
    Integer key = it.next();
    sum += key + map.get(key);
}

```

### 6. Iterator 遍历 Map.Entry 遍历 for

```
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 Collections `IterableMap`

```
MapIterator<Integer, Integer> mit = iterableMap.mapIterator();
while (mit.hasNext()) {
    sum += mit.next() + it.getValue();
}
```

10. Eclipse Collections `MutableMap`

```
mutableMap.forEachKeyValue((key, value) -> {
    sum += key + value;
});
```

Source: [Github](#)

Environment: Windows 8.1 64-bit Intel i7-4790 3.60GHz 16 GB

1. Benchmark score: 308 ± 21 ns / 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. Benchmark score: 37.606 ± 0.790 μs / op

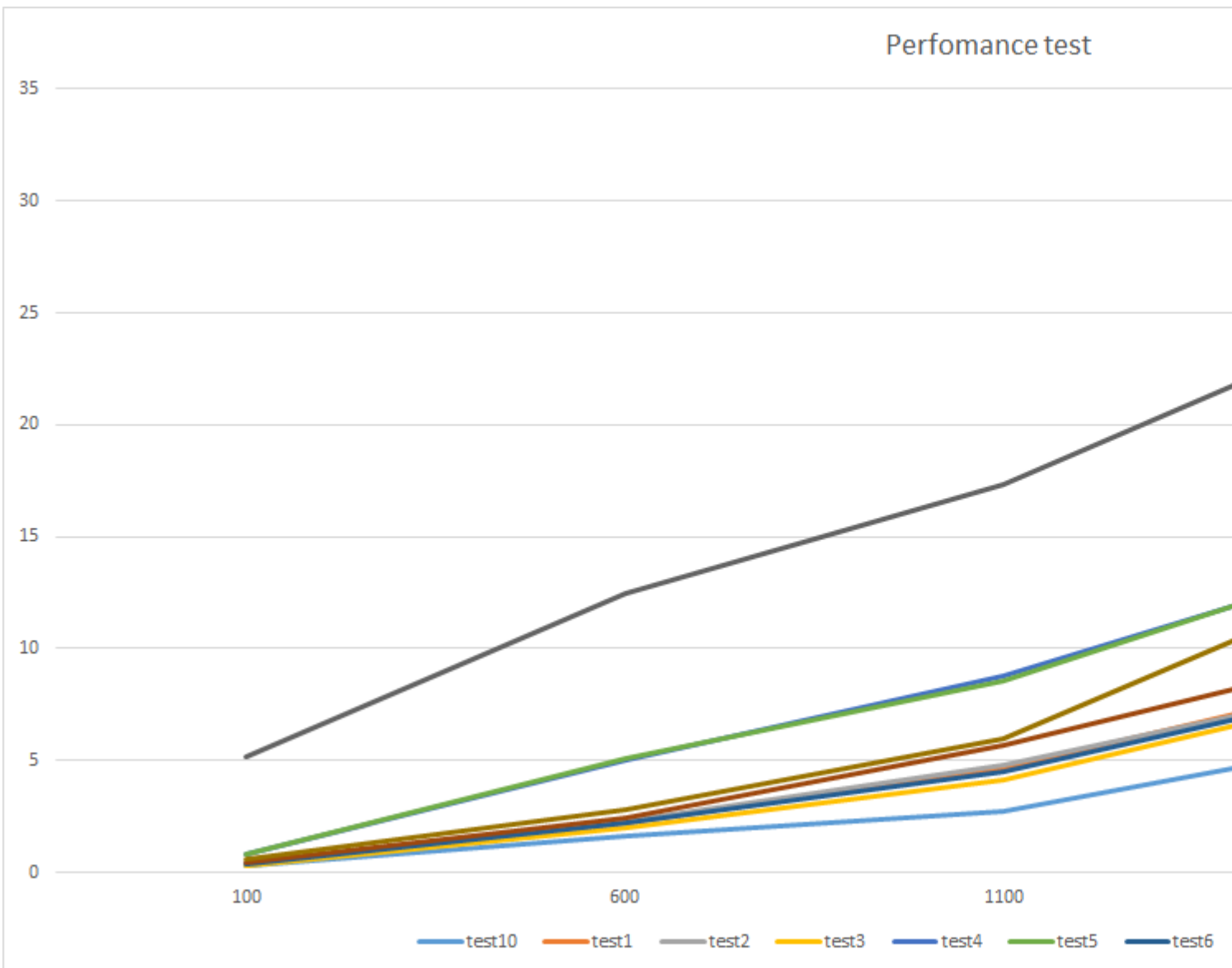
Benchmark	Score	Error	Units
test10_UsingEclipseMutableMap	37606	± 790	ns/op
test3_UsingForEachAndJava8	50368	± 887	ns/op
test6_UsingForAndIterator	50332	± 507	ns/op
test2_UsingForEachAndMapEntry	51406	± 1032	ns/op
test1_UsingWhileAndMapEntry	52538	± 2431	ns/op
test7_UsingJava8StreamAPI	54464	± 712	ns/op
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. 100000000001000000000000001184.767±332.968µs/ 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. 0000000000000000



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

0000000000

0000000000000000000000000000hashCode000equals000000.

0000000000000000000000

```
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();
    }
}
```

hashCode000000000000000000000000 equals000000000000000000.

00.

HashMap000

HashMap0Map00000000000000000000000000Key-Value0000000000.

### 1.00HashMap

```
Map<KeyType, ValueType> myMap = new HashMap<KeyType, ValueType>();
```

KeyType0ValueType000Java000000000000 - String0Integer0Float000000000000Employee0Student0.

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

### 2.0000HashMap0.

00HashMap000000000000000000000000Key0Value0000000000HashMap00000put000.

```
myMap.put("key1", 1);
myMap.put("key2", 2);
```

HashMap.put() method

### 3. HashMap.get()

HashMap.get() method

```
myMap.get("key1"); //return 1 (class Integer)
```

HashMap.get() returns null

### 4. HashMap.containsKey()

```
myMap.containsKey(varKey);
```

### 5. HashMap.containsValue()

```
myMap.containsValue(varValue);
```

containsValue() returns boolean true or false

Example

1

Maps are used to store key-value pairs.

Maps are implemented as `HashMap` in Java.

- 10 -

```
/*J2SE < 5.0*/
Map map = new HashMap();
map.put("name", "A");
map.put("address", "Malviya-Nagar");
map.put("city", "Jaipur");
System.out.println(map);
```

- 20 -

```
/*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);
```

- 30 -

```
Map<String, Object> map = new HashMap<String, Object>(){
    put("name", "A");
    put("address", "Malviya-Nagar");
    put("city", "Jaipur");
};
System.out.println(map);
```

- 40 -



```
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);
```

- 50 -

```
//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);
```

- 60 -

```
//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");
}
```

- 70 - 。

```
//Immutable single key-value map
Map<String, String> singletonMap = Collections.singletonMap("key", "value");
```

。

UnsupportedOperationException。

```
//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/zh-TW/java/topic/105/>

99:

{}.

0

```

// 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");
}

```

i

else if if false else if i > 2 else if

true else if else

if else if true else

```

for (int i = 0; i < 100; i++) {
    System.out.println(i);
}

```

for; / int i = 0 i < 100 i++ . { ' . true false }

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(;;){
    //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);
    }
}
```

。

```
int i = 0;
while (i < 100) { // condition gets checked BEFORE the loop body executes
    System.out.println(i);
    i++;
}
```

true 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 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。

。

Iterable<T> T[] for s

- T extends S
- T s
- s T s。
- T s

Java

```
T elements = ...
for (S s : elements) {
}
```

T		
INT []		
[]	INT	
Iterable<Byte>		
Iterable<String>	CharSequence	
Iterable<CharSequence>		
INT []		
INT []		

if

```
int i = 2;
if (i < 2) {
    System.out.println("i is less than 2");
} else {
    System.out.println("i is greater than 2");
}
```

if 语句的语法。if 语句的语法如下所示：`if (布尔表达式) { 语句 }`。if 语句的语法如下所示。

else 语句的语法。if 语句的语法如下所示：`if (布尔表达式) { 语句 }`。if 语句的语法如下所示。

else if

switch

switch 语句是 Java 中的一种选择语句。它类似于 if - else if - else 语句。switch 语句的语法如下所示；

switch 语句的语法如下所示

- case 语句 switch 语句的语法如下所示。
- default 语句 case 语句的语法如下所示 true 语句的语法如下所示 catch-all 语句。
- case 语句 ; break 语句的语法如下所示 case 语句的语法如下所示。

continue 语句的语法如下所示。continue 语句的语法如下所示。

- break
  - return
  - throw

switch 语句的语法如下所示 default 语句的语法如下所示。

```
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 语句的语法如下所示。break 语句的语法如下所示。

```
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 == 10000000000

```
I'm equal or greater than one
I'm one, two, or three
```

foo == 30000000000

```
I'm one, two, or three
```

Java SE 5

switch与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与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 ( ... ) { ... }#####。

```
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!");
    }
}
```





100:

java System.currentTimeMillis() .

### Examples

JMH

JMH . HashSet TreeSet .

JMH maven shade . JMH 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>
</dependencies>
```

```
<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 -DskipTests -Dmaven.test.skip=true -Dmaven.compiler.source=8 -Dmaven.compiler.target=8 -Dmaven.compiler.release=8 -Dmaven.javadoc.skip=true -Dmaven.wagon.provider=org.apache.maven.plugins:maven-wagon-provider-api:3.0.0

java -cp target/benchmarks.jar benchmark.BenchmarkMain

mvn package -DskipTests -Dmaven.test.skip=true -Dmaven.compiler.source=8 -Dmaven.compiler.target=8 -Dmaven.compiler.release=8 -Dmaven.javadoc.skip=true -Dmaven.wagon.provider=org.apache.maven.plugins:maven-wagon-provider-api:3.0.0

# 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()。 Shipilëv。 Jenkov。 Java。 1。 2。

Shipilëv。 Jenkov。 Java。 1。 2。

<https://riptutorial.com/zh-TW/java/topic/9514/>

**101:**

OOPL. "poly" "morphs". Poly " " " " .

. .

`Interfaces` Java API. `implement interface`.

**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
```

编译选项: javac Polymorph.java

```
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
    }
}
```

编译选项: javac Polymorph.java

```
public class Polymorph {

    void methodOverloaded(){
        //No argument and No return type
    }

    int methodOverloaded(){
        //No argument and int return type
        return 0;
    }
}
```

编译选项: javac Polymorph.java

编译选项: javac Polymorph.java

编译选项: javac Polymorph.java

编译选项: javac Polymorph.java

```
class SuperType {
    public void sayHello(){
        System.out.println("Hello from SuperType");
    }

    public void sayBye(){

```





```

    }
}

public class B extends A{
    public void hello(){
        System.out.println("No");
    }

    public void boo(){
        System.out.println("Say haha");
    }
}

```

當執行 B 的 hello 或 boo 方法時，會輸出 "No" 或 "Say haha"。這與 B 繼承自 A 的行為不同。在 A 中，hello 方法會輸出 "Hello"。

在 Java 中，abstract 方法與 Virtual 方法不同。abstract 方法必須在子類中實現，而 Virtual 方法則不一定。在 B 中，boo 方法重寫了 A 的 abstract 方法。

```

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");
    }
}

```

當執行 B 的 boo 方法時，會輸出 "Say haha"。這與 B 繼承自 A 的行為不同。在 A 中，boo 方法是一個抽象方法，必須在子類中實現。

在 Java 中，

Java 的

在 Java 中，

Java 的

在 Java 中，

java 的

在 Java 中，super.methodName() 方法用於訪問父類的方法。這與 Virtual 方法不同。在 A 中，boo 方法是一個抽象方法，必須在子類中實現。

在 Java 中，

1. 在 Java 中，super.methodName() 方法用於訪問父類的方法。
2. 在 Java 中，super.methodName() 方法用於訪問父類的方法。
3. 在 Java 中，super.methodName() 方法用於訪問父類的方法。



#### 4. 00000000000000000000000000000000

0000

```
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/zh-TW/java/topic/980/□□□>

## 102: JAR

Java 9 引入了 MRJAR 和 Jar 的更新。请参考 [JEP 238](#)。

### Examples

Jar

MANIFEST.MF 中的 `Multi-Release: true` Jar 的 Java 9 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 的 `root/META-INF/versions/9` 目录。
- JDK 9 的 `root/META-INF/versions/9/demo` 目录。
- JDK 10 的 `root/META-INF/versions/10` 目录。

jar Jar

Jar 的 Java 8 和 Java 9 的更新。

```
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 jar sampleproject-9 sampleproject-9 sampleproject-9 demo  
root/META-INF/versions/9。

```
jar root
- demo
  - SampleClass.class
- META-INF
  - versions
    - 9
      - demo
        - SampleClass.class
```

Main SampleClass URL Java 9

```
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 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/zh-TW/java/topic/9866/jar>

103:

。

### 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/zh-TW/java/topic/9918>

104:

java.lang.String 在 Java 中是不可变的。

在 Java 中，字符串是不可变的。

在 Java 中，字符串是不可变的。String 类。在 C/C++ 中，字符串是不可变的。

String 和 StringBuilder 类。String 和 StringBuilder 类。

StringBuffer 和 StringBuilder 类。StringBuffer 和 StringBuilder 类。

在 Java 中，字符串是不可变的。Java 中的 String.concat(...) 和 StringBuilder 的 concatenations 方法。

在 Java 中，字符串是不可变的。String 类。在 C/C++ 中，字符串是不可变的。char[] 数组。

在 Java 中，字符串是不可变的。JVM 中的 String 类。在 Java 中，字符串是不可变的。String 类。

在 Java 中，字符串是不可变的。

Java SE 7

在 Java 7 中，Java 虚拟机中的“PermGen”空间。

Java SE 7

在 Java 7 中，Java 虚拟机中的“PermGen”空间。

在 Java 中，字符串是不可变的。String 类。

在 Java 8 中，String 类使用 UTF-16 char 数组。在 Java 9 中，String 类使用 UTF-16 的 LATIN-1 字符。

### Examples

Strings 类中的 equals 和 equalsIgnoreCase 方法。

在 Java 中，字符串是不可变的。String 类。





```

case "B":
    System.out.println("B");
    break;
default:
    break;
}

```

=====

```

String s = "baz";
String foo = null;

```

s.equals(foo) returns false because foo is null, which would cause a NullPointerException.

```

"baz".equals(foo)

```

foo == null || foo.equals("baz") returns false because foo is null, which would cause a NullPointerException. "baz".equals(foo) returns false.

Java SE 7

Objects.equals() returns false because Objects.equals(foo, "baz") is false.

NullPointerException: Cannot invoke "String.equals(Object)" because the return value of "foo" is null.

```

String s = "baz";
String foo = null;

```

String.compareTo() returns 1 because "baz" is greater than "baz". String.compareTo("baz", "baz") returns 0. String.compareTo("baz", "baz", CASE\_INSENSITIVE\_ORDER) returns 0.

```

String s = "baz";
String foo = null;

```

Java SE 7 JLS 3.10.6 String.intern()

```

String s = "Hello!";
String interned = s.intern();
interned == s

```

s.intern() returns the same reference as s because "Hello!" is already interned.

====

```

String strObj = new String("Hello!");
String str = "Hello!";

// The two string references point to 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() returns the same reference as the literal.

- 字符串常量池。
- 字符串池。

interning == 字符串池。 ; - 字符串池 == 字符串池。 。

String

String

- toUpperCase
- toLowerCase

String String String Java。 Java

```
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"
```

Unicode

HTML

"TITLE".toLowerCase() " title " (\\u0131) LATIN SMALL LETTER DOTLESS I Locale.ROOT toLowerCase(Locale.ROOT) toUpperCase(Locale.ROOT)

Locale.ENGLISH Locale.ROOT

Unicode Consortium Unicode

ASCII

ASCII

1. 字符串池。
2. 字符串池。
3. 字符串池。
4. 字符串池。
5. 字符串池。
6. 字符串池。
  - 32 ASCII
  - ASCII 32
7. Character
8. 字符串池。

```
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);
```

XXXXXXXXXXXX

XXXXXXXXaXXXXXXXXb XXXXXXXString.contains()XXXXXXXX

```
b.contains(a); // Return true if a is contained in b, false otherwise
```

[String.contains\(\)](#) XXXXXXXXXXXXXXXXXXXString CharSequence。 XXXXXXXXXXXaString bXXXXXXXXXXXX。

```
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](#)XXXXXXXX

XXXString XXXStringXXXXXXXXXXXXXXXXString.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](#)XXXXXXXX

[String.indexOf\(\)](#) XXXXXXXXXXXString charStringXXXXXXXX。 XXXXXXXXXXXXXXX-1。

XX [] [String.indexOf\(\)](#)XXXXXXXX。

XXXXXXXXXXXX

```
String str1 = "Hello World";
String str2 = "wOr";
str1.indexOf(str2); // -1
str1.toLowerCase().contains(str2.toLowerCase()); // true
str1.toLowerCase().indexOf(str2.toLowerCase()); // 6
```

[Ideone](#)XXXXXXXX

String.length()

String.length() UTF-16

```
String str = "Hello, World!";
System.out.println(str.length()); // Prints out 13
```

Ideone

String char UTF-16. ≥0x1000 Unicode char. String Unicode UTF-16 char codePointCount

```
int length = str.codePointCount(0, str.length());
```

Java 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 datestring = "2015-11-17"

```
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. String char[] offset count start length new String(s.substring(...)) char[]

Java SE 7

JDK <7u6 substring String String String char[] offset count start length new String(s.substring(...)) char[]

Java SE 7

JDK 7u6 substring char[]

String.charAt(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"
```

String.charAt(n) n

System.out.println("Line separator: " + lineSeparator);

System.out.println("Line separator: " + lineSeparator);

System.out.println("Line separator: " + lineSeparator);

```
System.getProperty("line.separator")
```

Java SE 7

System.out.println("Line separator: " + lineSeparator);

```
System.lineSeparator()
```

System.out.println("Line separator: " + lineSeparator);

System.out.println("Line separator: " + lineSeparator);

```
System.out.println(String.format('line 1: %s.%nline 2: %s%n', lines[0], lines[1]));
```

System.out.println("Line separator: " + lineSeparator);

System.out.println("Line separator: " + lineSeparator);

```
public class Person {
    String name;
    int age;

    public Person (int age, String name) {
        this.age = age;
        this.name = name;
    }
}
```

System.out.println("Line separator: " + lineSeparator);

```
Person person = new Person(25, "John");
```

System.out.println("Line separator: " + lineSeparator);

```
System.out.println(person.toString());
```

Ideone

System.out.println("Line separator: " + lineSeparator);

```
Person@7ab89d
```

System.out.println("Line separator: " + lineSeparator);

```
Object.toString() returns a string representation of the object. Object.toString() returns the string representation of the object.
```

```
getClass().getName() + '@' + Integer.toHexString(hashCode())
```

System.out.println("Line separator: " + lineSeparator);



```
String s = "a|b|c";
String regex = Pattern.quote("|");
String[] arr = s.split(regex);
```

- 限制分割次数

```
String s = "a|b|c";
String[] arr = s.split("\\|", 2);
```

限制分割次数

`split(delimiter)` 返回包含子字符串的数组。 `split(delimiter, limit)` 返回

```
String[] split = data.split("\\|", -1);
```

`split(regex)` 等同于 `split(regex, 0)` 或 `split(regex, 0)`。

`limit` 限制返回数组中的子字符串数量。

如果 `n` 为正数，则返回最多 `n - 1` 个子字符串，最多 `n` 个子字符串。

如果 `n` 为 0，则返回所有子字符串。

如果 `n` 为负数，则返回所有子字符串。

**StringTokenizer**

`split()` 使用 `StringTokenizer`。

`StringTokenizer` 与 `String.split()` 不同。 `StringTokenizer` 返回 `String` 数组。 `StringTokenizer` 与 `String.split()` 不同。

默认分隔符为 `\t\n\r\f`。 `StringTokenizer`。

```
String str = "the lazy fox jumped over the brown fence";
StringTokenizer tokenizer = new StringTokenizer(str);
while (tokenizer.hasMoreTokens()) {
    System.out.println(tokenizer.nextToken());
}
```

输出结果

```
the
lazy
fox
jumped
over
the
brown
fence
```

限制分割次数

```
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());
}
```

输出结果



```
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 s □□□□□□□ `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` □

```
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. 字符串

```
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));
```

字符串反转

[org.apache.commons.lang3.StringUtils](#) countMatches 字符串 countMatches 字符串

```
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
```

字符串 Java API 字符串

```
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;
}
```

字符串 **StringBuilders**

字符串+字符串。 字符串

```
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 s1 = "a";
String s2 = "b";
String s = s1 + s2 + 2; // ab2
```

append

```
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 result = new StringBuilder();
for(int i = 0; i < array.length; i++) {
    result.append(extractElement(array[i]));
}
return result.toString();
```

StringBuilder

```
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

concat

```
String string1 = "Hello ";
```

```
String string2 = "world";
String string3 = string1.concat(string2); // "Hello world"
```

String string1 + string2。 concat

```
"My name is ".concat("Buyya");
```

String String

```
String replace(char oldChar, char newChar)
```

newChar oldChar。

```
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.
```

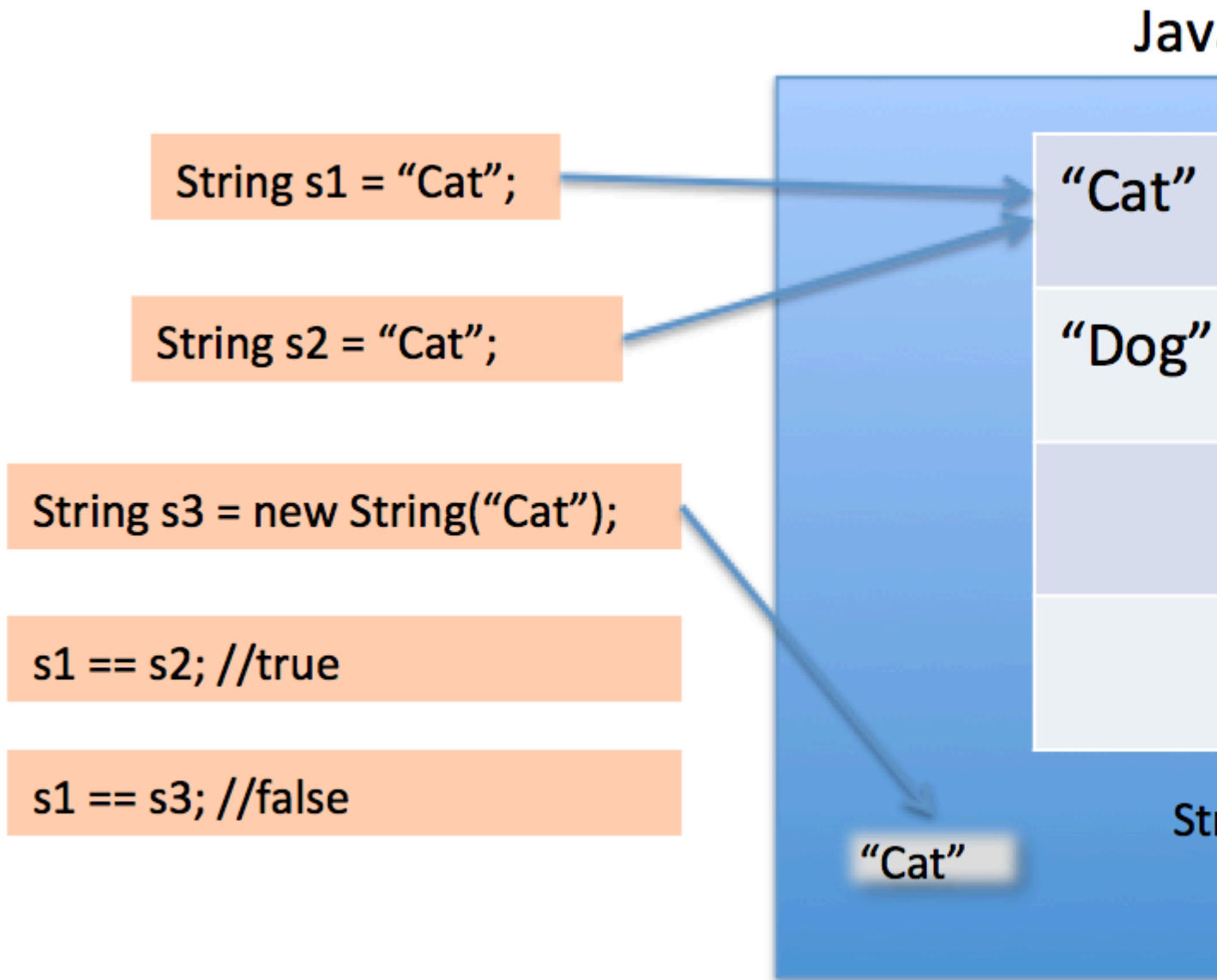
\$1。

```
String replaceAll(String regex, String replacement)
```

```
String s = "spiral metal petal et al.";
System.out.println(s.replaceAll("(\\w*etal)", "$1lica"));
```



```
true  
true  
true  
true
```



String String String String String

new String String intern String

Java SE 7

Java 7 String PermGen

PermGen

Java SE 7

RFC 6962931

JDK 7 Java Java  
Java

String.intern().....

.....

Java SE 7

switch.....toLowerCase().....toUpperCase.....

```
switch (myString.toLowerCase()) {
    case "case1" :
        ...
        break;
    case "case2" :
        ...
        break;
}
```

..

- Locale.....
- ..... - .....

..... <https://riptutorial.com/zh-TW/java/topic/109/>.....

105:

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());
    }
}
```

<https://riptutorial.com/zh-TW/java/topic/10563/>



## 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 Ъ");
        }
        /* 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 Ъ");
        }
    }
}
```

```
}
```

UTF-8

```
import java.nio.charset.StandardCharsets;
import java.util.Arrays;

public class GetUtf8BytesFromString {

    public static void main(String[] args) {
        String str = "Cyrillic symbol Ъ";
        //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/zh-TW/java/topic/2735>

107:

### 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*"Out" of *System*.

**ldc** - "Hello World"

**invokevirtual** - .

255. Java.

- 
- 

- - [ASM](#)
  - [Javassist](#)
  - [BCEL](#) - *Java 8+*
- - 
  - [JBytedit](#)
  - [reJ](#) - *Java 8+*
  - [JBE](#) - *Java 8+*

ASM jar

jar。 ClassNodes。 Tree API。

- loadClasses
- readJar JarFile JarEntry Map
- getNode []

```
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。 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

- ClassNode
- jarManifest / jar
- jar

- processNodes Map <String ClassNode>
- loadNonClasses File jarFile
- saveAsJar Map <String byte []> outBytes String 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()) {
            // Append 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。。

## Javassist Basic

Javassist Java Java Javassist /。

"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 {
                // retrieve 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 bytecode of the class
        byteCode = cc.toBytecode();
        // remove the CtClass from the ClassPool
        cc.detach();
    } catch (Exception ex) {
        ex.printStackTrace();
    }
}
return byteCode;
}
}

```

XXXXXXXXXXXXXXXXXXXX JVMXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

```

import java.lang.instrument.Instrumentation;

public class EasyAgent {

    public static void premain(String agentArgs, Instrumentation inst) {

        // registers the transformer
        inst.addTransformer(new DynamicTransformer());
    }
}

```

XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX JVMXXXXX。 XXXXXXXXXXXXXXXXXXXXXXX shellXXXXXXXXXXXX

```
java -javaagent:myAgent.jar MyJavaApplication
```

```
XXXXXXXXXXXXXX/XXXXXXXXXXXX jarXXXXXXXXXXXXMyJavaApplicationXXXXXXXXXXXXXX
"com.my.to.be.instrumented.MyClass"XXXXXXXXXXXXXX。

```

XXXXXXXXXXXX <https://riptutorial.com/zh-TW/java/topic/3747/>XXXXXXXXXXXX



108:

Java 1.4 ByteBuffer 1024 byte[]

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

Examples

ByteBuffer

ByteBuffer

byte[] reqBuffer 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

ByteBuffer put put put put

ByteBuffer

```
buffer.putInt(0xCAFEFEBABE).putChar('c').putFloat(0.25).putLong(0xDEADBEEFCAFEFEBABE);
```

```
buffer.putInt(0xCAFEFEBABE);
buffer.putChar('c');
buffer.putFloat(0.25);
buffer.putLong(0xDEADBEEFCAFEFEBABE);
```

ByteBuffer.put(byte s) 方法。 ByteBuffer.put(byte[]) 方法。 ByteBuffer.put -methods。  
ByteBuffer.put\* 方法。 ByteBuffer.put(byte s)。

### ByteBuffer - DirectByteBuffer

DirectByteBuffer 是 ByteBuffer 的子类，它直接操作 byte[]。

ByteBuffer 是 ByteBuffer 的父类。

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

ByteBuffer.allocateDirect(16) 方法。 ByteBuffer.allocateDirect 方法。

ByteBuffer 是 ByteBuffer 的父类。

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

DirectByteBuffer 是 JVM 直接操作 byte[] 的 ByteBuffer。 ByteBuffer 是 ByteBuffer 的父类。

ByteBuffer 是 IO 的 DirectByteBuffer。 ByteBuffer 是 ByteBuffer 的父类。

ByteBuffer.array() 方法。 UnsupportedOperationException。 ByteBuffer 是 ByteBuffer 的父类。

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

ByteBuffer 是 JNI 的。 ByteBuffer 是 byte[] 的。 ByteBuffer 是 Java 的。 ByteBuffer 是 ByteBuffer 的父类。

ByteBuffer 是 JNI 的。 ByteBuffer 是 NIO 的。

ByteBuffer 是 <https://riptutorial.com/zh-TW/java/topic/702> 的。

```
109: 
```

```
JavaJava. 1 0.333F false 'X'"Hello world\n"
```

### Examples

hexadecimal base-16 0-9AF 10-16.

octal base-8 0-7.

binary base-2 01.

```
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
```

```
'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 socialSecurityNumber = 999_99_9999;
long creditCardNumber = 1234_5678_9012_3456L;
float piFourDecimals = 3.14_15F;
double piTenDecimals = 3.14_15_92_65_35;
```

```
Java SE 7
```



00000000。

0000000000

TBD

00000000

0000000000000000byte 0 short 0 int 0 long0char00000000。 000000000000000000。 00000000000000000000000000  
0000000000。 0

00000000

00000000000000000000000000。 000

```
0 // The decimal number zero (type 'int')
1 // The decimal number one (type 'int')
42 // The decimal number forty two (type 'int')
```

00000000。 000000000000000000000000。

```
077 // This literal actually means 7 x 8 + 7 ... or 63 decimal!
```

0000000000。 00000000-10+10 000000000000000000 -000+000。

000000000000int000000000000002<sup>31</sup>02,147,483,6480000。

00002<sup>31</sup>Integer.MAX\_VALUE 1。 000000002147483647000000000000000000000000002147483648000000-000。 000000  
00000000Integer.MIN\_VALUE00。 0

```
int max = 2147483647; // OK
int min = -2147483648; // OK
int tooBig = 2147483648; // ERROR
```

000000

long00000000000L000000。 000

```
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
```

0000 int0long000000000000000000。 00

```
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.
```

000 JLS 3.10.1 - 0000

0000

Java 的 `boolean` 类型只有 `true` 和 `false` 两个值。

```
boolean flag = true;    // using the 'true' literal
flag = false;         // using the 'false' literal
```

字符串

Java 的字符串类型是 `String`。

- 字符串是 `String` 类型。
- 字符串是 `String` 类型，而不是 `String` 类型；`String` 是 `String` 类型的子类型。
- 字符串是不可变的。

字符串

```
"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
```

字符串的转义字符。字符串的转义字符。字符串的转义字符。

```
"Jello world // Compilation error (at the end of the line!)
```

字符串

字符串的转义字符。字符串的转义字符。字符串的转义字符。

```
String typingPractice = "The quick brown fox " +
                        "jumped over " +
                        "the lazy dog"
```

字符串的转义字符。字符串的转义字符。字符串的转义字符。

字符串

字符串的转义字符。字符串的转义字符。字符串的转义字符。

字符串的转义字符。字符串的转义字符。字符串的转义字符。

Null

Null 是 `null` 类型的值。Null 是 `null` 类型的值。

```
MyClass object = null;
MyClass[] objects = new MyClass[]{new MyClass(), null, new MyClass()};

myMethod(null);

if (objects != null) {
    // Do something
}
```

Null 是 `null` 类型的值。Null 是 `null` 类型的值。

Null 是 `null` 类型的值。Null 是 `null` 类型的值。

`null instanceof <SomeReferenceType>` `false`.

`float` `double`.

- 
- 
- 

JLS.

`float` `double` literals.

`f` `F` `d` `D` `f` `F` `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
```

`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` `float`.

Java 6.

1. `0` `x` `X`
2. `a` `f`
3. `p` `P` `e` `E` `20` `10`.

```
0x0.0p0f // this is zero expressed in hexadecimal form (`float`)
0xff.0p19 // this is 255.0 x 2^19 (`double`)
```

Java 的浮点数据类型。

Java

Java 7 引入的浮点数据类型。包括 `NaN`、`Infinity` 和 `Double`。

Java

IEEE 754 浮点数据类型。包括 `NaN`、`Infinity` 和 `Double`。

IEEE 754 浮点数据类型。包括 `NaN`、`Infinity` 和 `Double`。

Java

Java 的 `char` 数据类型。

- 数据类型 `' '`。
- 数据类型 `'\'`。
- 数据类型 `'\n'`。

Java

```
char a = 'a';
char doubleQuote = '"';
char singleQuote = '\'';
```

Java

```
char newline = '
// Compilation error in previous line
char newLine = '\n'; // Correct
```

更多请参考 <https://riptutorial.com/zh-TW/java/topic/8250>



`00 110: 000000`

**Examples**

`000000`

`000000000000000000000000`

```

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)));

```

`000000`

```

MD5    hash: E99E768582F6DD5A3BA2D9C849DF736E
SHA1   hash: 0135FAA6323685BA8A8FF8D3F955F0C36949D8FB
SHA256 hash: 8D35C97BCD902B96D1B551741BBE8A7F50BB5A690B4D0225482EAA63DBFB9DED

```

`0000 Java000000000000000000000000。`

`00000000`

`00000000000000`

```

final byte[] sample = new byte[16];

new SecureRandom().nextBytes(sample);

System.out.println("Sample: " + DatatypeConverter.printHexBinary(sample));

```

`0000000000`

```

Sample: E4F14CEA2384F70B706B53A6DF8C5EFE

```

`00000 nextBytes()000000000000000000000000。`

`000000000000`

```

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));

```

`0000000000`

```

Provider: SUN version 1.8
Algorithm: SHA1PRNG

```

Sample: C80C44BAEB352FD29FBBE20489E4C0B9

0000/000

00000000000000000000

```
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();
```

0000Java00000000000000000000。

000000000000000000

```
final KeyPairGenerator generator = KeyPairGenerator.getInstance("RSA");

generator.initialize(2048, SecureRandom.getInstance("SHA1PRNG", "SUN"));

final KeyPair pair = generator.generateKeyPair();
```

000000000

000000

```
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();
```

00000000000000000000000000000000。

00000

```
final PublicKey publicKey = keyPair.getPublic();
final Signature verifier = Signature.getInstance("SHA1withRSA");

verifier.initVerify(publicKey);
verifier.update(data);

System.out.println("Signature: " + verifier.verify(signature));
```

000000

Signature: true

0000/0000000000

0000000000

```
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));
```

0000000000

```
Message: Hello  
Encrypted: 5641FBB9558ECFA9ED...
```

00000000 Cipher00000000000000000000000000000000。 0000000000000000 JCA0000000。 0。 00RSA00000 message.getBytes()00  
0000000000。 0000 so00000000。

0000000

```
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);
```

00000000

```
Decrypted: Hello
```

000000000000 <https://riptutorial.com/zh-TW/java/topic/7529/0000000>

**111:**

Java  
Java  
Java  
Java

1. JCE
- 2.

### Examples

JCE  
Java Cryptography Extension

byte  
Java  
byte

Java  
Random  
SecureRandom  
X = Y  
f0 f0 x f0 y



### Examples

#### SecurityManager

SecurityManager Java JVM。 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;
    }
}
```

findClass loadClass 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

ProtectionDomain Permission ProtectionDomain Permission Policy

•

- 
- 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;

/**

```





```

*         refer to a concrete Permission descendant, or refers to
*         DeniedPermission.class or
<code>UnresolvedPermission.class</code>.</li>
```

- \* `<code>targetName</code>`
- \* `<code>targetClassName</code>`

caller's fault;

e.g., because `<code>targetName</code>`

not adhere

to the naming constraints of the target class; or due to the target class

not

exposing a `<code>(String name)</code>`

actions)</code>

constructor, depending on whether `<code>targetActions</code>`

`<code>null</code>`

or

not.</li>

```

*/
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 DeniedPermission 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 Permission descendant, or refers to
 *         DeniedPermission.class 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 DeniedPermission that encapsulates the given target
     permission.
     *
     * @throws IllegalArgumentException
     *         if target is null, a
     DeniedPermission, or an
     *         UnresolvedPermission.
     */
    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 implies(domain, permission)</code> invocation on the system-
default
    *      <code>Policy</code> yields true</code>, and</li>
    *      <li><code>permission</code> is not</em> implied by any
<code>DeniedPermission</code>s
    *      having potentially been assigned to 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□□□□□□□□□□ "foo"□□□□ DeniedPermission □□□□grant □□PropertyPermission □。 □□□□  
 setProperty("foo.baz", "xyz")□□□□□□□□□□□□□□□□ "foo"□□□□□□□□ "foo.bar"□□。

□□□□□□□□□□ <https://riptutorial.com/zh-TW/java/topic/5712/□□□□□□>



Path。 %JAVA\_HOME%\bin; Path。

Oracle Path Java - Oracle。

### ：

1. "cmdEnter。
2. javac -version。 JDK。

Windows Path。

### Java SE

1995 Java 1.0 Java。 Java。 Oracle / ;。

Java SE。 Java 8。 Java 9 2017。 Java 7 2015 4。 Java 7。

Java Java。 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 <sup>1</sup>	Java SE 9

1 - Oracle Java "。。

"SE"。 PC Android Java。

Java "Java ME" Micro Edition "Java EE"。 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)

Oracle - Oracle Java 8 Java 1.8 ; Java 8 Java 1.8

Java

JDK Java Java

- Java IDE Eclipse IntelliJ IDEA NetBeans
- Java Ant Gradle Maven
- 
- CI

Linux Java JDK

Linux OpenJDK Oracle JDK / JRE

"root" sudo

Java

apt-get Debian Linux Ubuntu

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 Linux

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

#### yum RedHat CentOS

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

#### dnf Fedora

Fedora yum dnf。

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

Fedora Java 7。

#### pacman Arch Linux

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

root sudo。

#### Gentoo Linux

Gentoo Java Wiki portage USE。

#### Redhat CentOS Fedora Oracle JDK

Oracle JDK JRE tar.gz JDK。

- Oracle Java Oracle "tar.gz"。
- ;
- ;

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

#### Oracle Java RPM

- Oracle Java RPM。
- rpm。

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

#### Windows Java JDK JRE

Oracle JDK JRE Windows。

- Oracle Java
- JDK JRE Server JRE。 Java JDK JRE
- 。
- ""
- Windows x86 32 Windows x64 64
- Windows....

Windows Java Choclately

- <https://chocolatey.org/>

2. 在cmd中按Win + R 打开“运行”对话框，输入“cmd”。

3. 在cmd中安装Java 8 JDK。

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

在cmd中运行

在cmd中运行以下命令以安装JDK / JRE。对于64位系统，请运行以下命令。对于32位系统，请运行以下命令。

1. 在cmd中运行以下命令。
2. 在cmd中运行以下命令。
3. 在cmd中运行以下命令。
4. 在cmd中运行以下命令。
5. 在cmd中运行以下命令。

```
cd .rsrc\JAVA_CAB10
extrac32 111
```

6. 在cmd中运行以下命令。

7. 在cmd中运行以下命令。

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

8. 在cmd中运行以下命令。

在cmd中运行以下命令。

请访问 <http://stackoverflow.com/a/6571736/1448252>

### 在macOS上安装Java JDK

#### Oracle Java 7和Java 8

在macOS上安装Java 7和Java 8。Oracle提供了Java for Mac。Apple提供了7u25。Java 7。

Oracle Java 7和Java 8在macOS 10.7.3和Intel Mac上可用。

#### Oracle Java

在macOS上安装Java 7和Java 8。Oracle提供了JDK和JRE。

- Java 8 - [Java SE](#)
- Java 7 - [Oracle Java Archive](#)

在cmd中运行以下命令。

```
/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 shell

```
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. shell \$ JDK; ;

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

2. alternatives --install Java SDK

```
$ 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

## Arch

Arch Linux

```
$ 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
```

Arch Linux Wiki

Linux

Java SDK

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

shell JRE JDK

- " " JRE JDK ; **PATH** .
- " " ;
- ; **JAVA\_HOME**

**PATH**

java javac

Java ; **Java**

bash shell ~/.bash\_profile ~/.bash\_profile ~/.bashrc Bash shell

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

export JAVA_HOME
export PATH
```

...Java<installation directory>. bin bin java javac.  
shell.

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

java javac. which java which javac.  
ptopagate shell.

Java

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

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

Java

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

alternatives. PATH.

- Linux Java
- "PATH".

### Java

Java.

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

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 privileges

```
sudo su
```

3. jdk install

```
mkdir /opt/jdk
```

#### 4. 解压缩tar包

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

#### 5. 列出目录内容

```
ls /opt/jdk
```

#### 6. 配置Oracle JDK环境变量

```
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. 验证Java安装

```
java -version
```

```
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)
```

更多Java教程 <https://riptutorial.com/zh-TW/java/topic/4754>





```

        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(){}
}

```

Angle类包含两个私有成员变量：angleInDegrees和angleInRadians。它们分别表示角度的度数值和弧度值。

此外，类中还包含以下方法：

1. setDegrees(double degrees)：用于设置角度的度数值。该方法会根据度数值计算出对应的弧度值。

2. setRadians(double radians)：用于设置角度的弧度值。该方法会根据弧度值计算出对应的度数值。

```

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(){}
}

```

```
}
```

。

。 `angleInRadians` `Angle` 。

<https://riptutorial.com/zh-TW/java/topic/1295/>

**115:**

```
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**

```
String[] parts = str.split("(?<=\\G.{8})");
```

```
int length = 5;
String[] parts = str.split("(?<=\\G.{ " + length + "})");
```

<https://riptutorial.com/zh-TW/java/topic/5613/>

116:

java.lang.Object.

- public final native Class <> getClass()
- public final native void notify()
- public final native void notifyAll()
- public final native void wait(long timeout) throws InterruptedException
- public final void wait() throws InterruptedException
- public final void wait(long timeout, int nanos) throws InterruptedException
- public native int hashCode()
- public boolean equals(Object obj)
- public String toString()
- protected native clone clone() throws CloneNotSupportedException
- protected void finalize() throws Throwable

### Examples

toString()

toString() returns a 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

Object.toString() User.

equals

TL; DR

==

.equals()

Object.equals() 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()

```

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
}
}
```

Object equals() method. hashCode() method. hashCode() equals(). Object obj Foo obj equals(). equals() hashCode() IDE. String equals() Java API. String String. Java SE 7. Java 1.7 java.util.Objects equals null 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);
}
}
```

Object equals() null.

```
@Override
public boolean equals(Object obj) {
    //...check for null
    if (getClass() != obj.getClass()) {
        return false;
    }
    //...compare fields
}
}
```

EntityManager.merge() 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. 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().

### hashCode

Java equals hashCode.

- Java hashCode equals.
- equals(Object) hashCode.
- equals(Object) hashCode.

HashMap HashTable HashSet. 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 java.util.Arrays#hashCode(Object primitives)

```

@Override
public int hashCode() {
    return Arrays.hashCode(new Object[] {field1, field2, field3});
}

```

Java SE 7

Java 1.7 java.util.Objects.hash(Object... 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() Object.hashCode()。

ImmutableArray

ImmutableArray

```

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;
}
}
}

```

Object.equals()方法使用Object.hashCode()方法。Object.hashCode()方法使用Object.hashCode()方法。

Object.hashCode()方法使用Object.hashCode()方法。Object.hashCode()方法使用Object.hashCode()方法。

Object.hashCode()方法使用Object.hashCode()方法。Object.hashCode()方法使用Object.hashCode()方法。

wait()和notify()方法

wait()和notify()方法 - wait()和notify()方法。wait()和notify()方法。wait()和notify()方法。

wait()和notify()方法

```

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()anotherSpecificUsergetClass()class SpecificUser  
User。

getClass()

```
public final native Class<?> getClass();
```

getClass()Class<? extends T>TgetClass()。

```
Class<? extends String> cls = "".getClass();
```

clone()

clone()。 clone()。

Cloneable。

Cloneablenative clone()Cloneable。 CloneNotSupportedException。

ObjectCloneNotSupportedException Object。

。 "。 。

```
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。。

。

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。 finalize()WeakReference。

Joshua Bloch的“Effective Java”中的finalize()。

。

Java的Object。super()。。

super()。

。

super()。

```
public class MyClass {
    public MyClass() {
        super();
    }
}
```

super()。

```
public class MyClass {
    public MyClass() {
        // empty
    }
}
```

。

```
public class MyClass {
}
```

### Constructor-Chaining

。

```
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-argument  
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/zh-TW/java/topic/145/>

117:

### 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

```

```

public class Sheep implements Cloneable {

    private String name;

    private int weight;

    public Sheep(String name, int weight) {
        this.name = name;
        this.weight = weight;
    }

}

```





```

// 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/zh-TW/java/topic/2830/□□□□>



输出。

输出结果

```
private static void getAnotherObjectNot(Person person) {
    person = new Person();
    person.setName("George");
}
```

main方法调用setAnotherName方法并打印输出

```
getAnotherObjectNot(person);
System.out.println(person.getName());
```

输出结果

```
Linda 5
Linda
```

输出结果与main方法调用getAnotherObjectNot方法并打印输出Linda 5。Linda  
main方法调用getAnotherObjectNot方法并打印输出Linda 5。Linda  
main方法调用getAnotherObjectNot方法并打印输出Linda 5。

输出结果

```
private static Person getAnotherObject() {
    Person person = new Person();
    person.setName("Mary");
    return person;
}
```

输出结果

```
Person mary;
mary = getAnotherObject();
System.out.println(mary.getName());
```

输出结果

```
Linda 5
Linda
Mary
```

输出结果

```
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/zh-TW/java/topic/5454>

119: 119

119

Applet 是 Java 的 一个 类。

Applet 是 Java 的 一个 类。

2016 年 11 月 19 日 11:19 Web

API

119

Applet 是 Java 的 一个 类。 2000 年 11 月 19 日 11:19。

Applet 是 Java 的 一个 类。

init()	applet
destroy()	applet
start()	applet
stop()	applet
paint()	repaint()

### Examples

119

Applet 是 Java 的 一个 类。

```
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);
    }
}
```

```
}  
}
```

applet 运行在 javax.swing.JApplet 类中。

Java SE 1.2

Java 1.2 的 swing API 使用 java.applet.Applet 类。

Applet 类是 javax.swing.JApplet 类的子类。Applet 类是 HTML 文档的一部分。

```
<html>  
  <head></head>  
  <body>  
    <applet code="MyApplet.class" width="400" height="200"></applet>  
  </body>  
</html>
```

## GUI

Applet 类是 javax.swing.JApplet 类的子类。Applet 类是 HTML 文档的一部分。

```
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);  
    }  
}
```

## Applet 类

Applet 类是 javax.swing.JApplet 类的子类。Applet 类是 HTML 文档的一部分。Applet 类是 HTML 文档的一部分。Applet 类是 HTML 文档的一部分。

```
public class MyLinkApplet extends JApplet{
```





```
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/zh-TW/java/topic/5503/□□□>

120: 000

00

properties 是 java.util.Properties 的 Hashtable 。

Properties 是 Hashtable 的子类。 Properties 是 Hashtable 的包装类。

Properties

Properties 是 Hashtable 的包装类。

00

- Properties
- 是
- 是

00

Properties 是 Map 的子类。 Properties 是 Map 的子类。 Properties 是 Map 的子类。 `getProperty` 和 `setProperty` 是 `stringPropertyNames` 。

Properties 是 Java 的类。 Properties 是 `Properties.load` 。

- `load` 方法从流中加载属性。
- `load` 方法从流中加载属性。
- `load` 方法从流中加载属性。
- `load` 方法从流中加载属性。
- `load` 方法从流中加载属性。

Properties 是 Java SE 的类。 Properties 是 `java.util.ResourceBundle` 的子类。 Properties 是 `InputStream` 的子类。 Properties 是 `ISO 8859-1` 的子类。 Properties 是 `native2ascii` 的子类。 Properties 是 `JDK` 的子类。

Properties 是 `Charset` 的子类。 Properties 是 `Reader` 的子类。 Properties 是 `InputStreamReader` 的子类。 Properties 是 `load` 的子类。 Properties 是 `InputStream` 的子类。

Properties 是 XML 的子类。 Properties 是 `loadFromXML` 的子类。 Properties 是 `XML` 的子类。 Properties 是 `DTD` 的子类。 <http://java.sun.com/dtd/properties.dtd>。

### Examples

0000

Properties

```
public class Defaults {
    public static Properties loadDefaults() {
        try (InputStream bundledResource =
            Defaults.class.getResourceAsStream("defaults.properties")) {
            Properties defaults = new Properties();
            defaults.load(bundledResource);
            return defaults;
        }
    }
}
```



```

* <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.properties.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");
}

```

.

```

1  <?xml version="1.0" encoding="UTF-8" standalone="no"?>
2  <!DOCTYPE properties SYSTEM "http://java.sun.com/dtd/properties.dtd">
3  <properties>
4  <comment>testing properties with xml</comment>
5  <entry key="age">23</entry>
6  <entry key="color">green</entry>
7  <entry key="name">Steve</entry>
8  </properties>
9

```

XML

properties.loadFromXML().properties.load()。

```

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/zh-TW/java/topic/576>

**121:**

Java... . . . Java...

- public class OuterClass {public class InnerClass {}}
- public class OuterClass {public static class StaticNestedClass {}}
- public void method(){private class LocalClass {}}
- SomeClass anonymousClassInstance = new SomeClass{};
- SomeInterface anonymousClassInstance = new SomeInterface{};

Java... JLS... Java...

- .
- .
- .
- .

... " " . . .

... lambdas... lambdas.

- " " . . .
  - .
    - 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;
}
}

```

编译并运行。

```

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() + ", ");
        }
    }
}

```

编译并运行。

编译并运行。注意，静态类。

```

public class OuterClass1 {

    private static class StaticNestedClass {

    }

}

```

编译并运行。

```

public class OuterClass2 {

    private class NestedClass {

    }

}

```



```
}  
  
}
```

000000000000000000000000 000000000000. 00. 0000000000

```
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  
    }  
  
}
```

public class OuterClass {

public class InnerClass {

public int x = 5;

}  
public InnerClass createInner() {

return new InnerClass();  
}  
public class SomeOtherClass {

```
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 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 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;
    }
}
```

private class InnerClass {}  
public InnerClass makeInnerClass() {  
 return new InnerClass();  
}  
}

```
public class OuterClass {
    private class InnerClass {}

    public InnerClass makeInnerClass() {
        return new InnerClass();
    }
}
```







## 122: 正则表达式

正则表达式

正则表达式是用于匹配字符串中模式的强大工具。Java 提供了 `java.util.regex` 包来实现正则表达式。正则表达式在 Java 中非常有用。

正则表达式

- `Pattern patternName = Pattern.compile(regex);`
- `Matcher matcherName = patternName.matcher(textToSearch);`
- `matcherName.matches()` // 是否 textToSearch 完全匹配正则表达式 true
- `matcherName.find()` // 在 textToSearch 中查找下一个匹配项。返回 `String` 类型的匹配项。
- `matcherName.group(groupNum)` // 返回正则表达式中的第 groupNum 个捕获组
- `matcherName.groupName(groupNum)` // 返回正则表达式中的第 groupNum 个捕获组的名称 (Java 7+)

正则表达式

## 正则表达式

正则表达式

```
import java.util.regex.Matcher
import java.util.regex.Pattern
```

## 正则表达式

正则表达式是用于匹配字符串中模式的强大工具。正则表达式在 Java 中非常有用。

## 正则表达式

符号	描述
*	匹配前面的子表达式零次或多次
+	匹配前面的子表达式一次或多次
?	匹配前面的子表达式零次或一次

## 正则表达式

正则表达式

### Examples

正则表达式

正则表达式

正则表达式

```
\d{3}-\d{3}-\d{4}
```

Regex groups and capturing groups

```
(\d{3})-(\d{3})-(\d{4})
^-----^ ^-----^ ^-----^
Group 1 Group 2 Group 3
```

Java Pattern and Matcher classes

```
"(\d{3})-(\d{3})-(\d{4})"
```

Pattern and Matcher classes

```
Pattern phonePattern = Pattern.compile("(\\d{3})-(\\d{3})-(\\d{4})");
Matcher phoneMatcher = phonePattern.matcher("abcd800-555-1234wxyz");
```

Matcher.find() method

```
phoneMatcher.find();
```

Matcher.group() method

```
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() vs Matcher.group(0)

Java SE 7

Java 7 introduces named capturing groups

Named capturing groups

```
(?<AreaCode>\d{3})-(\d{3})-(\d{4})
^-----^ ^-----^ ^-----^
AreaCode      Group 2 Group 3
```

Named capturing groups example

```
String aCode = phoneMatcher.group("AreaCode"); // "800"
```

Pattern.compile() flags

Pattern.compile() flags example

```
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.
```







```
}
```

🔗 <https://riptutorial.com/zh-TW/java/topic/135>

## 123: Java

Java 的 API 文档中 API 的注释。 通常，Java 的 API 文档中 API 的注释。 通常，Java 的 API 文档中 API 的注释。

### Examples

Integer 的 equals 方法。 Integer 的 equals 方法。 Integer 的 equals 方法。

```
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
```

Integer 的 equals 方法。 Integer 的 equals 方法。 Integer 的 equals 方法。

```
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
```

Integer 的 equals 方法。 Integer 的 equals 方法。 Integer 的 equals 方法。

Integer 的 equals 方法。 Integer 的 equals 方法。 Integer 的 equals 方法。

Integer 的 equals 方法。 Integer 的 equals 方法。 Integer 的 equals 方法。

Integer 的 equals 方法。

Integer 的 equals 方法。 Integer 的 equals 方法。 Integer 的 equals 方法。

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 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-resources Closeable AutoCloseable Closeable close() 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

Java

Java

Java

static static null static GC JNI

**ClassLoader**

ClassLoader



```

        System.out.println("Hello back to you");
    } else {
        System.out.println("Are you feeling grumpy today?");
    }
}
}
}
}

```

String s1 = "hello"; String s2 = "hello"; s1 == s2 返回 true

```

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");
        }
    }
}

```

String s1 = "hello"; String s2 = "hello"; s1 == s2 返回 true。这是因为 Java 3.10.5 之前，字符串字面量是存储在堆内存中的。从 Java 7 开始，字符串字面量是存储在字符串常量池中的。因此，相同的字符串字面量会指向相同的内存地址。

String s1 = "hello"; String s2 = "hel" + "lo"; s1 == s2 返回 false

```

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");
        }
    }
}

```

String s1 = "hello"; String s2 = "hel" + "lo"; s1 == s2 返回 false。这是因为 s2 是由两个字符串拼接而成的，因此它不会与 s1 指向相同的内存地址。

String s1 = "hello"; String s2 = "hel" + "lo"; s1 == s2 返回 false

String s1 = "hello"; String s2 = "hel" + "lo"; s1 == s2 返回 false

String s1 = "hello"; String s2 = "hel" + "lo"; s1 == s2 返回 false

```

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;
}

```



```
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;
}
}
```

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;
}
}

```
MutableLocation here = new MutableLocation(1, 2);
MutableLocation there = here;
MutableLocation elsewhere = new MutableLocation(1, 2);
```

MutableLocation here = new MutableLocation(1, 2);
MutableLocation there = here;
MutableLocation elsewhere = new MutableLocation(1, 2);
1. "[1,2]" here
2. "[1,2]" there
3. "[1,2]" elsewhere

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);

```
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

```
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 == elsewhere.x == true
here.x = 42 here.x = 42 there.x = 42

here.x = 42 here.x = 42 there.x = 42

here.x == there.x == elsewhere.x == true

here.x == there.x == elsewhere.x == true

```
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 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) true MutableLocation "===="。

====

Java pass by value <sup>1</sup>。

。

。

<sup>2</sup>。 。

```

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 - Python Ruby "====" "====/====" "===="。

2 - "====" "====" "===="。 Java "====" "====" [https://en.wikipedia.org/wiki/Evaluation\\_strategy](https://en.wikipedia.org/wiki/Evaluation_strategy)。

====

StackOverflow Java C++

```
i += a[i++] + b[i--];
```

i a b

- Java<sup>1</sup>
- C++

Java " "

StackOverflow

```
int a = 1;
a = a++;
System.out.println(a); // What does this print.
```

Java 2 1

Java

1 - Java Memory Model

String

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 toUpperCase() String String s

```
s = s.toUpperCase();
```

" " String

Java [https://riptutorial.com/zh-TW/java/topic/4388/000 java](https://riptutorial.com/zh-TW/java/topic/4388/000-java)









```
}
}
```

Serializable Thread 。 readObject

```
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();
}
}
```

Main

```
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();
}
}
}

```

main RangeRandom Thread.start readObject

### serialVersionUID

java.io.Serializable serialVersionUID long static final. serialVersionUID Sun. serialVersionUID.

serialVersionUID serialVersionUID InvalidClassException.

long Serializable 'serialVersionUID' InvalidClassException.

```

public class Example implements Serializable {
    static final long serialVersionUID = 1L /*or some other value*/;
    //...
}

```

serialVersionUID Java Serialization.



## 序列化

- 序列化是指将对象转换为字节流的过程。反序列化是指将字节流转换回对象的过程。
- 序列化通常用于在网络上传输对象或在持久化存储中保存对象。
- 序列化时，对象的状态（非静态成员变量）被转换为字节流。反序列化时，字节流被还原为对象的状态。
- 序列化接口 `writeObject` / `readObject` 用于自定义对象的序列化。默认情况下，使用 `defaultReadObject` 和 `defaultWriteObject` 方法进行序列化。
- 实现 `java.io.Serializable` 接口。如果不实现该接口，序列化时会抛出 `InvalidClassException`。
- 序列化方法可以声明为 `public`、`package-private`、`protected` 或 `private`。
- 序列化时，对象的类名、包名、版本号等信息会被写入字节流。反序列化时，这些信息用于验证对象的兼容性。

## JSON

- JSON (JavaScript Object Notation) 是一种轻量级的数据交换格式。
- JSON 数据由键值对组成，键用双引号括起来，值可以是字符串、数字、布尔值、数组或对象。
- JSON 对象是键值对的集合，键用双引号括起来，值可以是字符串、数字、布尔值、数组或对象。
- JSON 数组是值的有序列表，值可以是字符串、数字、布尔值、数组或对象。
- 序列化时，JSON 对象被转换为字符串。反序列化时，字符串被解析为 JSON 对象。
- 序列化时，JSON 对象中的键名、键值、数组元素等都会被序列化。
- 序列化时，JSON 对象中的键名、键值、数组元素等都会被序列化。
- 序列化时，JSON 对象中的键名、键值、数组元素等都会被序列化。
- 序列化时，JSON 对象中的键名、键值、数组元素等都会被序列化。

## Jackson 与 JSON

在 REST API 中，JSON 是一种常见的数据格式。使用 `org.codehaus.jackson.map.ObjectMapper` 类可以将 POJO 对象转换为 JSON 字符串，也可以将 JSON 字符串转换为 POJO 对象。

序列化示例：

```
public class User {
    private Long id;
    private String name;
    private String email;

    //getter setter are omitted for clarity
}
```

反序列化示例：

```
public class Program {
    private Long id;
    private String name;
    private User createdBy;
    private String contents;

    //getter setter are omitted for clarity
}
```

序列化与反序列化：

```
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");

```

```

{
  "id": 1,
  "name": "Program @# 1",
  "createdBy": {
    "id": 1,
    "name": "Bazlur Rahman",
    "email": "example@example.com"
  },
  "contents": "Some contents"
}

```

ObjectMapper.writeValueAsString() method converts the object into JSON string. In this example, we have a Program object and we are converting it into JSON string.

```

{
  "id": 1,
  "name": "Program @# 1",
  "ownerId": 1
  "contents": "Some contents"
}

```

ObjectMapper.readValue() method converts the JSON string into object. In this example, we have a JSON string and we are converting it into Program object.

```

@JsonIgnoreProperties(ignoreUnknown = true)
public class Program {}

```

ObjectMapper.readValue() method converts the JSON string into object. In this example, we have a JSON string and we are converting it into Program object. We are using ObjectMapper.readValue() method to convert the JSON string into Program object.

```

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/zh-TW/java/topic/767>□□□

125:

- 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;
```



```

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/zh-TW/java/topic/2318/🔗>

126:

- 
- 

## SourceStringFileInputStream

Scanner Java 5 reset() Java 6 Path

### Examples

Scanner

```
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);
```

Reading from keyboard

```
Reading
from
keyboard
//space
```

```
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();
}
```

File Scanner File scanner.hasNext() while Names.txt nextLine() nextInt()

nextBoolean()。 scanner.nextLine()。 nextLine() scanner。 close()。

try with resources Java 7

```
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);
```

IOException Scanner

Scanner.useDelimiter(",") String.split(...) Scanner String

```
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

java.util.Scanner System.in stdin 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);
}
}

```

Scanner.nextInt()

```

import java.util.Scanner;

Scanner s = new Scanner(System.in);
int number = s.nextInt();

```

Scanner.nextInt() returns an int. Scanner.nextInt() returns an int. Scanner.nextInt() returns an int.

Scanner.nextInt()

```

Scanner s = new Scanner(System.in);
try {
    int number = s.nextInt();
} catch (NoSuchElementException | java.lang.IllegalStateException e) {
}

```

00

```
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();
```

00000000 <https://riptutorial.com/zh-TW/java/topic/551/000>



127:

interface. - Java

- Foo {void foo(); /\* \*/ }
- Foo1Foo {void bar(); /\* \*/ }
- Foo2FooFoo { /\* 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();
```



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();
```



```
Animal dog = new Dog();
Animal cat = new Cat();
Animal bird = new Bird();

dog.getSound(); // "Woof"
cat.getSound(); // "Meow"
bird.getSound(); // "Chirp"
```

Animal[] animals = { new Dog(), new Cat(), new Bird() };

```
for (Animal animal : animals) {
    System.out.println(animal.getSound());
}
```

Animal[] animals = { new Dog(), new Cat(), new Bird() };  
 System.out.println(animals[0].getSound());  
 System.out.println(animals[1].getSound());  
 System.out.println(animals[2].getSound());

Animal[] animals = { new Dog(), new Cat(), new Bird() };  
 System.out.println(animals[0].getSound());  
 System.out.println(animals[1].getSound());  
 System.out.println(animals[2].getSound());

```
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[] animals = { new Dog(), new Cat(), new Bird() };  
 System.out.println(animals[0].getSound());  
 System.out.println(animals[1].getSound());  
 System.out.println(animals[2].getSound());

Animal[] animals = { new Dog(), new Cat(), new Bird() };  
 System.out.println(animals[0].getSound());  
 System.out.println(animals[1].getSound());  
 System.out.println(animals[2].getSound());

interface NoiseMaker {  
 void makeNoise();  
}

public abstract class Animal implements NoiseMaker {

```
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 default

Java 8

Java 8 "Base" "Abstract"

Observer-Listener

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.....

.....

.....Oracle.....Oracle.....

.....

```
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() { ... }
}
```



### Examples

Example 1

#### 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());
}
```

Example 2

```
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 + "!");
```

Example 3

```
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 + "!");
```

Scanner methods: next(), nextLine(), nextInt(), nextDouble(), nextBoolean(), nextShort(), nextByte(), nextLong(), nextFloat(), nextLine()

Scanner methods: next(), nextLine(), nextInt(), nextDouble(), nextBoolean(), nextShort(), nextByte(), nextLong(), nextFloat(), nextLine()

```
scanner.nextByte();
scanner.nextShort();
scanner.nextInt();
```

```

scanner.nextLong();
scanner.nextFloat();
scanner.nextDouble();
scanner.nextBigInteger();
scanner.nextBigDecimal();

```

```

Scanner scanner = new Scanner(System.in);
scanner.hasNextLine() // true
scanner.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();

```

•

- 
- 

IDE Eclipse 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");
}
}
```

□□□□□□□□□□

`PrintWriter.format` □□□□ `System.out.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□□□;□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□
- %-5s □□□□□□□□ 5□□□□□□□□;□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□
- %5.10s □□□□□□□□ □□ 5□□□□ □□ 10□□□□;□□□□□□□□ 5□□□□□□□□ 5□□□□□□□□;□□□□□□□□ 10□□□□□□□□ 10□□□□□□□□
- %-5.5s □□□□□□□□□□ 5□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□;□□□□□□□□ 5□□□□□□□□□□ 5□□□□□□□□□□;□□□□□□□□□□ 5□□□□□□□□□□ 5□□□□□□□□□□

□□□□□□□□I / O.□ <https://riptutorial.com/zh-TW/java/topic/126/□□□□i---o->

## 129: List, Set, Map

00

Java 9 的 Java Collections API 增加了 `List.of()`, `Set.of()` 和 `Map.of()` 方法。

这些方法返回不可变的 `List`, `Set` 和 `Map`。

00

- `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)`

00

w /	
<code>List.of(E e)</code>	◦
<code>Set.of(E e)</code>	◦
<code>Map.of(K k, V v)</code>	◦
<code>Map.of(Map.Entry&lt;? extends K, ? extends V&gt; entry)</code>	<code>Map.EntryKV</code> ◦

### Examples

00000000

- `List<Integer> immutableEmptyList = List.of();`
  - `immutableEmptyList instanceof List<Integer>`
- `List<Integer> immutableList = List.of(1, 2, 3, 4, 5);`
  - `immutableList instanceof List<Integer>`
- `List<Integer> mutableList = new ArrayList<>(immutableList);`
  - `mutableList instanceof List<Integer>` `mutableList instanceof List<Integer>`

00000000

- `Set<Integer> immutableEmptySet = Set.of();`
  - `immutableEmptySet instanceof Set<Integer>`
- `Set<Integer> immutableSet = Set.of(1, 2, 3, 4, 5);`
  - `immutableSet instanceof Set<Integer>`

- `Set<Integer> mutableSet = new HashSet<>(immutableSet);`
  - `Set<Integer>` `Set<Integer>` `Set<Integer>`。

Map

- `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>` `Map<Integer, Integer>`。

<https://riptutorial.com/zh-TW/java/topic/9783>



130:

Object

- `ArrayType[] myArray;` //
- `ArrayType myArray[];` //
- `ArrayType[][] myArray;` // `s`
- `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 [] extends Object anyType []`

ArrayType	◦ int long byte Objects String MyObject◦
	Object◦
	◦ new int[3] {1, 2, 3}◦

### Examples

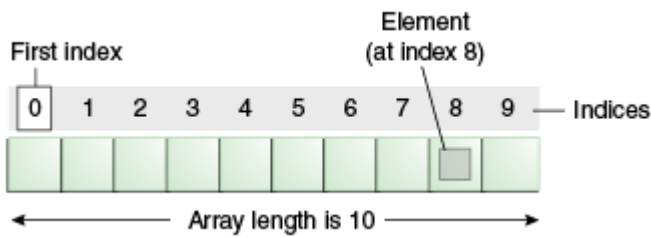
```
int[] numbers1 = new int[3]; // Array for 3 int values, default value is 0
int[] numbers2 = { 1, 2, 3 }; // Array literal of 3 int values
int[] numbers3 = new int[] { 1, 2, 3 }; // Array of 3 int values initialized
int[][] numbers4 = { { 1, 2 }, { 3, 4, 5 } }; // Jagged array literal
int[][] numbers5 = 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).
```



length - 100.

0 1 2 3 4 5 6 7 8 9 10



0 1 2 3 4 5 6 7 8 9 10

Java new Type[length]

- 0 byte 1 short 2 int 4 long 8 float 8 double.
- char '\u0000'.
- false boolean.
- null.

int[] array1 = new int[] { 1, 2, 3 };

```
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 */
```

Java

Java

```
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



```
String[] array8 = new String[3]; // { null, null, null }
String[] array9 = null; // null
```

Ideone

String[] array10 = new Object[2];

```
Object[] array10 = { new Object(), new Object() };
```

String[] array11 = new String[3];

```
Object[] array11 = new String[] { "foo", "bar", "baz" };
array11[1] = "qux"; // fine
array11[1] = new StringBuilder(); // throws ArrayStoreException
```

Object[] emptyArray = new Object[0];

String[] emptyArray = new String[0]; Collection Array.

String[] array8 = new String[3];



String[] array8 = new String[3];

```
public class MyGenericClass<T> {
    private T[] a;

    public MyGenericClass() {
        a = new T[5]; // Compile time error: generic array creation
    }
}
```

String[] array8 = new String[3];

- 1. Object[] array8 = new Object[3];

```
a = (T[]) new Object[5];
```

Object[] array8 = new Object[3];

- 2. Array.newInstance()

```
public MyGenericClass(Class<T> clazz) {
    a = (T[]) Array.newInstance(clazz, 5);
}
```

T[] array8 = (T[]) Array.newInstance(Object.class, 3);



Java SE 1.2

Arrays.fill()

```
Arrays.fill(array8, "abc"); // { "abc", "abc", "abc" }
```

Ideone

fill()

```
Arrays.fill(array8, 1, 2, "aaa"); // Placing "aaa" from index 1 to 2.
```

Ideone

Java SE 8

Java 8 Concurrent parallelSetAll

```
int[] array = new int[5];
Arrays.setAll(array, i -> i); // The array becomes { 0, 1, 2, 3, 4 }.
```

Ideone

ArrayIndexOutOfBoundsException

0,1,2,3,4... ArrayIndexOutOfBoundsException

```
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 }
```

ArrayIndexOutOfBoundsException

ArrayIndexOutOfBoundsException

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.
```

Ideone

java.util.Collection 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 T[] toArray(T[] a) arraycopy

- Object[] toArray() arraycopy T[] toArray(T[] a) arraycopy
- T[] toArray(new T[non-zero-size]) T[] toArray(new T[0])

Java SE 8

Java SE 8+ Stream Stream Stream.toArray() Array

```
String[] strings = list.stream().toArray(String[]::new);
```

1 2 ArrayList Java

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()` 调用 `Object.toString()` 返回 `String` 数组。 `String` 数组的每个元素都是 `Object.toString()` 的返回值。

```
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!]
```

`Arrays.toString()` 调用 `Object.toString()`。 `Object.toString()` 返回 `String`。

```
public class Dog {
    /* implicitly extends Object */
}

Dog[] arr = { new Dog() };
System.out.println(Arrays.toString(arr)); // [Dog@17ed40e0]
```

要使用 `Arrays.asList()`，需要导入 `java.util.Arrays`。

`Arrays.asList()` 返回 `List`。 `List` 是 `Collection` 的子接口。

```
String[] stringArray = {"foo", "bar", "baz"};
List<String> stringList = Arrays.asList(stringArray);
```

在 Java SE 5 中，`Arrays.asList()` 返回 `java.util.Arrays$ArrayList`，它是 `Collection` 的子接口。

在 Java SE 7 中，`Arrays.asList()` 返回 `java.util.ArrayList`。

Java SE 5

```
String[] stringArray = {"foo", "bar", "baz"};
List<String> stringList = new ArrayList<String>(Arrays.asList(stringArray));
```

Java SE 7

在 Java SE 7 中，`Arrays.asList()` 返回 `java.util.ArrayList`。这允许使用 `Diamond`。这允许使用 `Type Inference` 和 `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<>();
```







int[][] table = {

```
int[][] table = {
    {1, 2, 3},
    {4, 5, 6}
};
```

int[][] table = {

```
int[][] table = {
    null,
    {},
    {1},
    {1,2}
};
```

int[][][] arr = new int[3][3][3];

```
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 ≤ i < array.length - 1

0 ≤ i < array.length

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

```
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 2
at your.package.path.method(YourClass.java:15)
```

2 ≤ index < people.length

int index = 2;

```
int index = 2;
if (index >= 0 && index < people.length) {
    System.out.println(people[index]);
}
```



```
Object[] bObject = { b }; // bObject contains one element

System.out.println(Arrays.equals(aObject, bObject)); // false
System.out.println(Arrays.deepEquals(aObject, bObject)); // true
```

Object[] equals() hashCode() Set[] Map[] helper[] equals() hashCode() 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 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});
```

Java SE 5

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);
}
```

Iterable

Arrays.asList []

```
Integer[] boxed = { 1, 2, 3 };
Iterable<Integer> boxedIt = Arrays.asList(boxed); // list-backed iterable
```

```
Iterator<Integer> fromBoxed1 = boxedIt.iterator();
```

##### java 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 [guava](#)##

```
Iterable<Integer> fromPrimitive2 = Ints.asList(primitives);
```

#####.

##

```
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);
    }
}
```

#####Array#####.

##do-while#####while##do-while##.

#####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;
    }
}
```

#####.

##



```
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

```
int[] a = { 4, 1, 3, 2 };
int[] b = Arrays.copyOfRange(a, 0, a.length); // [4, 1, 3, 2]
```

A[]T[] A[]A

```
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\(original, newLength, newType\)](#)#####[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](#)##### [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
```



### 3. 使用 System.arraycopy

在 Java 6 之前，我们使用 System.arraycopy 来复制数组。

```
String[] newArray = new String[listOfCities.length + 1];
System.arraycopy(listOfCities, 0, newArray, 0, listOfCities.length);
newArray[listOfCities.length] = "Sydney";
```

在 Java 6 之后，我们可以使用 Arrays.copyOf 和 Arrays.copyOfRange。

```
String[] newArray = Arrays.copyOf(listOfCities, listOfCities.length + 1);
newArray[listOfCities.length] = "Sydney";
```

在 Java 8 之前，我们使用 System.arraycopy 来复制数组。

- 使用 System.arraycopy

在 Java 6 之前，我们使用 System.arraycopy 来复制数组。

在 Java 6 之后，我们可以使用 Arrays.copyOf 和 Arrays.copyOfRange。

- 使用 System.arraycopy
- 使用 Arrays.copyOf 和 Arrays.copyOfRange

在 Java 8 之前，我们使用 System.arraycopy 来复制数组。

在 Java SE 8 之前，我们使用 System.arraycopy 来复制数组。在 Java SE 8 中，我们使用 List 和 Set/Map API。ArrayList 使用 O(N) 的时间复杂度来复制数组。

ArrayList 使用 O(N) 的时间复杂度来复制数组。在 Java SE 8 中，我们使用 List 的 copyOf 方法。它使用 O(1) 的时间复杂度来复制数组。

在 Java 8 之前，我们使用 System.arraycopy 来复制数组。

在 Java 8 之后，我们可以使用 Arrays.copyOf 和 Arrays.copyOfRange。

```
String[] strings = new String[] { "A", "B", "C" };
int[] ints = new int[] { 1, 2, 3, 4 };
```

在 Java 8 之前，我们使用 System.arraycopy 来复制数组。

在 Java 8 之后，我们可以使用 Arrays.binarySearch。

```
int index = Arrays.binarySearch(strings, "A");
int index2 = Arrays.binarySearch(ints, 1);
```

在 Java 8 之后，我们可以使用 Arrays.asList。

```
int index = Arrays.asList(strings).indexOf("A");
int index2 = Arrays.asList(ints).indexOf(1); // compilation error
```

在 Java 8 中，我们可以使用 Stream。

在 Java SE 8 中，我们可以使用 Stream。

```
int index = IntStream.range(0, strings.length)
    .filter(i -> "A".equals(strings[i]))
    .findFirst()
    .orElse(-1); // If not present, gives us -1.
```



```
String[] sort() String[] sort() Comparable
```

Code

```
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));
```

Code

```
String array before sorting : [John, Steve, Shane, Adam, Ben]
String array after sorting in ascending order : [Adam, Ben, John, Shane, Steve]
```

Code

```
Arrays.sort(names, 0, names.length, Collections.reverseOrder());
System.out.println("String array after sorting in descending order : " +
Arrays.toString(names));
```

Code

```
String array after sorting in descending order : [Steve, Shane, John, Ben, Adam]
```

### Object

Comparable Comparator

sort (Object []) Comparable

e1.compareTo(e2) e1<e2 ClassCastException sort (T[], Comparator) Object sort (T[], Comparator)

```
// 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));
```

Code

```
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 ]
```

oooooooooooooooo

oooooo000000000000。

ooooooooooooooooJava 8oooooooooo

Java SE 8

```
int[] primitiveArray = {1, 2, 3, 4};
Integer[] boxedArray =
    Arrays.stream(primitiveArray).boxed().toArray(Integer[]::new);
```

oooooooooooooooooooooooooooooooooooo

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
}
```

oooooooooooooooooooooooooooooooo

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
}
```

oooooooo <https://riptutorial.com/zh-TW/java/topic/99>oo

131: I / O.

Java I / O API Java I / O java.io API Java I / O

Examples

byte []

Java 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();
}
```

byte []

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

Streams `InputStream` / `OutputStream` `int` byte s.

```

// 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` `Unicode` `Java` `UTF-8` `FileWriter` `FileReader`



00000000

```

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();

```

0000Reader0BufferedReader 000

```

// 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);
}

```

00000000

```

File f = new File(path);
String content = new Scanner(f).useDelimiter("\\Z").next();

```

\ zEOF00000000. 00EOF

00000000

000000

```

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();
        }
    }
}

```

000

```

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.useDelimiter

```

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

Java 7 NIO FileSystem ZIP JAR

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");

```

## Path

```
// -> IO to NIO
Path pathFromFile = new File("io.txt").toPath();

// -> NIO to IO
File fileFromPath = Paths.get("nio.txt").toFile();
```

## Files

```
// -> IO
if (file.exists()) {
    boolean deleted = file.delete();
    if (!deleted) {
        throw new IOException("Unable to delete file");
    }
}

// -> NIO
Files.deleteIfExists(path);
```

## OutputStream

FileChannel 与 Files.write(Path path, byte[] bytes, OpenOption... options) 与 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));
    }
}
```

## Files

```
// -> 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();
}
```

0000000000

000000000000Java000000000000000000

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();
```

00000000Java 70000000000nio API000000000000

Java SE 7

```
Path path = Paths.get("path_to_the_file");
byte [] data = Files.readAllBytes(path);
```

00

00000000000streams0readers000FileChannel API00000

streams00

```
//000000FileInputStream ios = new FileInputStream0filename0;
```

```
// 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和OutputStream

InputStream和OutputStream。 InputStream和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和Buffer

Channel和Buffer。 Channel和I / O。

Channel -

1. FileInputStream。 FileInputStream.getChannel() Channel。
2. FileInputStream.getChannel() Channel。
3. ByteBuffer。 ByteBuffer。
4. Channel read ByteBuffer read。 ByteBuffer - flip()。 flip()。 flip()。
5. Channel read。
6. ByteBuffer。
7. read() -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 exist.");
            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

Channel 的 transferTo() 方法。

```

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 sinkChannel = new FileOutputStream(destFile).getChannel()) {

            srcChannel.transferTo(0, srcChannel.size(), sinkChannel);
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}

```

## BufferedInputStream

BufferedInputStream 的构造函数。



```

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 Buffer

Channel

1. FileOutputStream
2. FileOutputStream.getChannel() FileChannel
3. ByteBuffer
4. ByteBuffer flip() 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



System.out System.err。 。

。

。

UNIX Linux MacOSX shell。

```
$ java -jar app.jar arg1 arg2 > /dev/null 2>&1
$ java -jar app.jar arg1 arg2 > out.log 2> error.log
```

"/ dev / null" "out.log" "error.log"。

shell。 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"))));
```

" / 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();
}
```

<https://riptutorial.com/zh-TW/java/topic/93>

- Paths.get(String first String ... more //String Path
- Paths.get(URI uri //URI Path

### Examples

Path programmatically

Paths 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()

```
Path p1 = Paths.get("/var/www"); // p1.toString() returns "/var/www"
```

- 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() 方法。 用于将路径解析为绝对路径。

```
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"
```

██████ □

normalize() 方法。 用于将路径规范化。

规范化路径。 用于将路径规范化为绝对路径。

Paths API 的 normalize() 方法

```
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 类。 用于检查文件是否存在。

████

exists() 方法。 用于检查文件是否存在。

```
Files.exists(Path path)
```

□

```
Files.notExists(Path path)
```

Files.exists(path) 与 Files.notExists(path) 方法。 用于检查文件是否存在。

- exists() 返回 true 且 notExists() 返回 false。
- exists() 返回 false 且 notExists() 返回 true。
- exists() 与 notExists() 方法。 用于检查文件是否存在。

□□□□ □□□□ □□□□ □

Files.isDirectory(Path path) 与 Files.isRegularFile(Path path) 方法。

```
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
```



00000000000000

```
Files.isReadable(Path path)
Files.isWritable(Path path)
Files.isExecutable(Path path)

Files.isHidden(Path path)
Files.isSymbolicLink(Path path)
```

## 00 MIME 00

```
Files.probeContentType(Path path)
```

0000000000MIME00。 000MIME00 String00000000

- text/plain00000000
- HTML0000text/html
- application/pdf for PDF files
- 00PNG0000image/png

000

0000Files000000000000Files。

```
Path p2 = Paths.get(URI.create("file:///home/testuser/File.txt"));
byte[] content = Files.readAllBytes(p2);
List<String> linesOfContent = Files.readAllLines(p2);
```

Files.readAllLines()0000charset0000000000StandardCharsets.UTF\_8 00

```
List<String> linesOfContent = Files.readAllLines(p2, StandardCharsets.ISO_8859_1);
```

000

0000Files000000000000Files

```
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)
```

00000000000000000000。

00000000I / O.0 <https://riptutorial.com/zh-TW/java/topic/5519/000i---o->

133:

- `assert 1 ;`
- `assert 1 < 2 ;`

	<code>falseAssertionError</code>
2	◦ <code>assert</code> <code>AssertionError</code>

`-ea` `java`

```
java -ea com.example.AssertionExample
```

`false` `;`

### Examples

```
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/zh-TW/java/topic/407/>



134:

Java 8 Calendar java.time API

### Examples

getInstance() GregorianCalendar Calendar

Calendar JANUARY int 000. 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);
```

2015 9 30

roll()

```
calendar.roll(Calendar.MONTH, -6);
```

/

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");
```

Calendar 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/zh-TW/java/topic/165>

135: 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 "with" LocalDate

```
LocalDate.now().withMonth(6);
```

June java.util.Date setMonth 6 6 50

LocalDate LocalDate

```
LocalDate ld = LocalDate.now().plusDays(1).plusYears(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

```



```

    DateTimeFormatter dtF3 = DateTimeFormatter.ISO_LOCAL_DATE_TIME;

    LocalDateTime ldtf1 = LocalDateTime.now();

    System.out.println(ldtf1.format(dtF2) + "\n"+ldtf1.format(dtF3));
}
}

```

ISO8061

LocalDates

LocalDate ChronoUnit

```

    LocalDate d1 = LocalDate.of(2017, 5, 1);
    LocalDate d2 = LocalDate.of(2017, 5, 18);

```

ChronoUnit between Temporal s LocalDate

```

    long days = ChronoUnit.DAYS.between(d1, d2);
    System.out.println( days );

```

java.time \* <https://riptutorial.com/zh-TW/java/topic/4813>-java-time---

## 136: 日期

日期

- `Date object = new Date();`
  - `Date object = new Date(long date);`

日期

Date	
Date	"the epoch"1970 11:00:00 GMT

日期

日期

Java Date 类使用 long 值表示日期。Java Date 类与 Calendar 类。

Calendar 类使用 Date 类。Calendar 类使用 Date 类。Calendar 类使用 Date 类。

Java Date 类使用 long 值表示日期。Java Date 类使用 Calendar 类。

日期

```
import java.util.Date;
```

Date 类使用 java.util 包。

日期

Date 类使用 getDate() 方法获取日期。

```
public final class Transaction {
    private final Date date;

    public Date getTransactionDate() {
        return date;
    }
}
```

日期 date 类使用 Java 8 的 java.time 包 API。

Date 类使用 Calendar 类。

### Java 8

Java 8 的 java.time 包 API 使用 LOCALDATE 类。java.time 包 API 使用 Java 8 的 API。使用 java.time 类。

### Examples

Date 类





```

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 milliseconds!%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`
- `compareTo`
- `compareTo` `Date` `anotherDate`
- `equals` `Object` `obj`

after `before` `compareTo` `equals` `getTime`。

`compareTo`。

- `0` `Date` `Date`
- `0` `Date` `Date`
- `0` `Date` `Date`

`equals`。

## Java 8

Java 8 `java.time.LocalDate` `Date` `Object`。 `LocalDate` `ChronoLocalDate` `Chronology`。

`Object` `java.time.LocalDateTime`。 `LocalDate` `LocalDateTime`。

`LocalDate` `ChronoLocalDate`。

`LocalDate` `ChronoLocalDate`。 `LocalDate`。

`LocalDate` `[ChronoLocalDate]`。

`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() compareTo()。

ChronoLocalDate Epoch Day。 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
```

dateFormat.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。

```
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
```

4 SHORT MEDIUM LONG FULL。

	Locale.US	Locale.France
	09 6 30	30/06/09
	2009 6 30	2009 7 30
	2009 6 30	2009 7 30
	2009 6 30	mardi 30 juin 2009

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()));
```

Java Date

Java 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

Java Date。 Date。 long。 Date。

Calendar。 Calendar。

```
Calendar c = Calendar.getInstance();
```

Calendar。 Calendar。 Calendar。

```
c.set(1974, 6, 2, 8, 0, 0);
Date d = c.getTime();
```

getTime。 Date。 Calendar set。 Calendar。

[]

=====  
/====.

- (1974, 6, 2, 8, 0)====
- Calendar=====.

### Java 8 LocalDate与LocalDateTime

Date与LocalDate、LocalDateTime、Date与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 to Date

```
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()

java.text.SimpleDateFormat

```
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

Date java.util.Date java.sql.Date

java.sql.Date JDBC SQL DATE

java.util.Date() Date convert(java.util.Date utilDate) java.sql.Date

```
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("dateFormatted 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;
    }

}
```

00

```
java.util.Date is : Fri Jul 22 14:40:35 IST 2016
java.sql.Date is : 2016-07-22
dateFormatted date is : 22/07/2016 - 02:40:35
```

java.util.Date java.sql.Date

0000

Date LocalTime LocalTime

1. LocalTime time = LocalTime.now();
  - time = LocalTime.MIDNIGHT;
  - time = LocalTime.NOON;
  - 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/zh-TW/java/topic/164>



137:

guava apache eclipse Java 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	TreeMap
LinkedHashMultiset	-	-	LinkedHashMap
ConcurrentHashMultiset	SynchronizedBag	SynchronizedBag	Collections.synchronized
-	SynchronizedSortedBag	SynchronizedSortedBag	Collections.synchronized
ImmutableMultiset	UnmodifiableBag	UnmodifiableBag	Collections.unmodifiable
ImmutableSortedMultiset	UnmodifiableSortedBag	UnmodifiableSortedBag	Collections.unmodifiable

#### 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` `TreeBag` `GC`

```

// 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. `Guava` `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

1. `HashBag` - □□□□□
2. `SynchronizedBag` - □□□□□□□□□□
3. `SynchronizedSortedBag` - - □□□□□□□□
4. `TreeBag` - □□□□□

II. GS / Eclipse

5. `MutableBag` - □□□□□□
6. `MutableSortedBag` - □□□□□

III. □□□

7. `HashMultiset` - □□□□□□
8. `TreeMultiset` - □□□□□
9. `LinkedHashMultiset` - □□□□□
10. `ConcurrentHashMultiset` - □□□□□□□□□□

Guava □ Apache □ Eclipse □□□□ Multimap

□□□□□□□□□□. JDK □□□ HashMap <K □ List> □ HashMap <K □ Set> □□.

				Eclipse
HashMap		ArrayListMultimap		MultiValueMap FastLi
HashMap	HashSet	HashMultimap	MultiValueMap. multiValueMap( new HashMap<K, Set>(), HashSet.class);	Unifie
HashMap	TreeSet	Multimaps. newMultimap( HashMap, Supplier <TreeSet>)	MultiValueMap.multiValueMap( new HashMap<K, Set>(), TreeSet.class)	TreeSo Multim
LinkedHashMap		LinkedListMultimap	MultiValueMap. multiValueMapnew LinkedHashMap <KList> ArrayList.class;	
LinkedHashMap	LinkedHashSet	LinkedHashMultimap	MultiValueMap. multiValueMap( new LinkedHashMap<K, Set>(), LinkedHashSet.class)	
TreeMap	TreeSet	TreeMultimap	MultiValueMap. multiValueMap( new TreeMap<K, Set>(), TreeSet.class)	

□□ Multimap □□□

```
□□ □□□ "Hello World"□□□□□□□□□□ 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 Multimaps
MultiMap<String, Integer> multiMap = new MultiValueMap<String, Integer>();

// Fill Multimaps
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. Guava 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 Multimaps
Multimap<String, Integer> multiMap = HashMultimap.create();

// Fill Multimaps
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
```

## Notes

### I. Apache

1. [MultiValueMap](#)
2. [MultiValueMapLinked](#)
3. [MultiValueMapTree](#)

### II. GS / Eclipse

1. [FastListMultimap](#)
2. [HashBagMultimap](#)
3. [TreeSortedSetMultimap](#)
4. [UnifiedSetMultimap](#)

### III. Guava

1. [HashMultiMap](#)
2. [LinkedHashMultimap](#)
3. [LinkedListMultimap](#)
4. [TreeMultimap](#)
5. [ArrayListMultimap](#)

00000000 - 0000

00000000 - 0000

1.0000

00	JDK	000	GS-00
000000	new ArrayList<> ()	Lists.newArrayList()	FastList.newList()
00000000	Arrays.asList("1", "2", "3")	Lists.newArrayList("1", "2", "3")	FastList.newListWith("1", "2", "3")
0000= 100 000	new ArrayList<>(100)	Lists.newArrayListWithCapacity(100)	FastList.newList(100)
000 collectin 0000	new ArrayList<>(collection)	Lists.newArrayList(collection)	FastList.newList(collection)
000 Iterable[] 000	-	Lists.newArrayList(iterable)	FastList.newList(iterable)
[] Iterator 0000	-	Lists.newArrayList(iterator)	-
00000000	Arrays.asList(array)	Lists.newArrayList(array)	FastList.newListWith(array)
0000000000	-	-	FastList.newWithNValues(10, ( ) -> "1")

000

```

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");

```

2000

☐☐	JDK	☐☐☐	GS-☐☐
☐☐☐☐	new HashSet<>()	Sets.newHashSet()	UnifiedSet.newSet()
Creatre☐ ☐☐☐☐	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)
☐☐☐ Iterable ☐☐☐☐	-	Sets.newHashSet(iterable)	UnifiedSet.newSet(iterable)
☐☐☐☐☐☐☐☐☐ ☐☐	-	Sets.newHashSet(iterator)	-
☐Array☐☐ set	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");

```

30000

	JDK		GS-00
000 00	new HashMap<>()	Maps.newHashMap()	UnifiedMap.newMap()
000 0=	new HashMap<>(130)	Maps.newHashMapWithExpectedSize(100)	UnifiedMap.newMap(130)



API	JDK	Guava	GS-API
130 API			
API API API	new HashMap<> (map)	Maps.newHashMap (map)	UnifiedMap.newMap (map)
API API	-	-	UnifiedMap.newWithKeyValues ("1", "a", "2", "b")

API

```

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.newWithKeyValues("1", "a", "2", "b");
System.out.println(withKeys);

System.out.println("createHashMap end");

```

API CreateCollectionTest

1. CollectionCompare
2. CollectionSearch
3. JavaTransform

API <https://riptutorial.com/zh-TW/java/topic/2958>

138:

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/zh-TW/java/topic/10160/>

**139: 本地化日期**

本地化

Java 本地化日期和时间的格式。

本地化日期和时间的格式。使用“locale”属性。

20/3/16	1.234,56	\$ 1,000.50	1,000.50		
20/3/16	1,234.56	£1,000.50			100
16320	1,234.56	\$ 1,000.50	1,000.50 MXN		60

本地化日期

- 本地化日期 [本地化日期](#)

**Java 本地化**

- Java 本地化日期 [本地化日期](#)
- Oracle 本地化日期 [Java 本地化日期](#)
- JavaDoc [Locale](#)

**Examples**

本地化“locale”属性

SimpleDateFormat 本地化日期。

本地化日期格式“MM/dd/yyyy”。

**Java 本地化日期**

本地化日期 `DateFormat` 静态方法。本地化日期 `Locale` 属性。

```
String localizedDate = DateFormat.getDateInstance(style).format(date);
```

本地化日期 `DateFormat` 属性 `FULL` `LONG` `MEDIUM` `SHORT`。

本地化日期 `DateFormat` 方法 `getDateInstance()`。

```
String localizedDate =
    DateFormat.getDateInstance(style, request.getLocale()).format(date);
```

本地化日期

本地化日期

```
"School".equalsIgnoreCase("school"); // true
```



This is locale



setDefault() 方法

```
setDefault(Locale.JAPANESE); //Set Japanese
```

<https://riptutorial.com/zh-TW/java/topic/4086/>

## 140: Enum

Enum

Java Enum 是 一种 特殊 的 类。

Enum

- [public / protected / private] enum Enum\_name { // 枚举类型名。
- ENUM\_CONSTANT\_1 [ENUM\_CONSTANT\_2 ...]; // 枚举常量。 枚举常量名。 枚举常量名。
- ENUM\_CONSTANT\_1 [ENUM\_CONSTANT\_2 ...]; // 枚举常量名。 枚举常量名。
- ENUM\_CONSTANT\_1 {...} [ENUM\_CONSTANT\_2 {...} ...]; // 枚举常量名。 枚举常量名; 枚举常量名。
- ENUM\_CONSTANT.name // 枚举常量名 String。
- ENUM\_CONSTANT.ordinal // 枚举常量名。
- Enum\_name.values // 枚举常量名 Enum\_name []。
- Enum\_name.valueOf "ENUM\_CONSTANT" // ENUM\_CONSTANT.name - 枚举常量名。
- Enum.valueOf Enum\_name.class "ENUM\_CONSTANT" // ENUM\_CONSTANT.name - 枚举常量名。

Enum

Enum

Enum 是 java.lang.Enum 的子类。

Enum

Enum 是 一种 特殊 的 类。

### Examples

Enum

Enum

Enum

```
public enum Season {
    WINTER,
    SPRING,
    SUMMER,
    FALL
}
```

Enum 是 一种 特殊 的 类。

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
    }

}
```

□□□□□□□□□□□□□□□□

```
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;
    }
}
```

```
}  
  
}
```

enum 枚举

```
public enum Season {  
    WINTER,  
    WINTER, //Compile Time Error : Duplicate Constants  
    SPRING,  
    SUMMER,  
    FALL  
}
```

enum 枚举 public 公共 static 静态 final 最终。enum 枚举 static 静态。enum 枚举。

enum 枚举

```
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"
```

enum 枚举 values() 枚举 枚举。enum 枚举。

```
Season[] seasons = Season.values();
```

enum 枚举。

enum 枚举

```
public static void enumIterate() {  
    for (Season s : Season.values()) {  
        System.out.println(s.name());  
    }  
}
```

enum 枚举 switch 枚举

```
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

```
Season.FALL == Season.WINTER // false
Season.SPRING == Season.SPRING // true
```

Season.FALL.equals(Season.FALL)

```
Season.FALL.equals(Season.FALL); // true
Season.FALL.equals(Season.WINTER); // false
Season.FALL.equals("FALL"); // false and no compiler error
```

enum MutableExample

```
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 MutableExample final

Serializable Comparable Enum

```
public abstract class Enum<E extends Enum<E>>
    extends Object
    implements Comparable<E>, Serializable
```

enum Coin

enum Coin; 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
```

setter

Enum final

```
public enum Coin {
    PENNY(1), NICKEL(5), DIME(10), QUARTER(25);

    private final int value;

    Coin(int value){
        this.value = value;
    }

    ...
}
```

Enum final boolean

```
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;
}

...

}

```

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;
    }
}

```

enum opposite Direction { NORTH, SOUTH, EAST, WEST }  
enum opposite { NORTH(SOUTH), SOUTH(NORTH), EAST(WEST), WEST(EAST) }  
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

enum 是 Java 5 引入的，它提供了一种新的数据类型，用于表示一组有限的、互斥的常量值。enum 类是类，因此可以继承、实现接口、重载方法、使用 final 修饰符等。enum 类是 final 的，不能被继承。

```
public interface ExtensibleEnum {
    String name();
}
```

enum 类可以继承接口，也可以实现接口。enum 类 API 和类 API 类似，enum 类 API 如下：

enum 类 API

```
public enum DefaultValues implements ExtensibleEnum {
    VALUE_ONE, VALUE_TWO;
}
```

enum 类 API

```
public enum ExtendedValues implements ExtensibleEnum {
    VALUE_THREE, VALUE_FOUR;
}
```

enum 类 API - printEnum() 方法

```
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 类 API - switch-on-enum 方法

enum 类 API

enum 类 API

```
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);
}
```

enum values() returns an array of enum constants.

enum constants are of type Enum.

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

enum values() returns an array of enum constants. enum constants are of type Enum.

```
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());
    }
}
}

```

enumSet()返回EnumSet.allOf(Day.class)。

enumSet()

enumSet()返回EnumSet.allOf(Day.class)。

```

public class Holder<T extends Enum<T>> {
    public final T value;

    public Holder(T init) {
        this.value = init;
    }
}

```

enumSet()返回EnumSet.allOf(Day.class)。

enumSet()

enumSet()返回EnumSet.allOf(Day.class)。

```

enum DayOfWeek {
    SUNDAY, MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, SATURDAY;
}

```

enumSet()返回EnumSet.allOf(Day.class)。

```

String dayName = DayOfWeek.SUNDAY.name();
assert dayName.equals("SUNDAY");

DayOfWeek day = DayOfWeek.valueOf(dayName);
assert day == DayOfWeek.SUNDAY;

```

enumSet()返回EnumSet.allOf(Day.class)。

```

Class<DayOfWeek> enumType = DayOfWeek.class;
DayOfWeek day = Enum.valueOf(enumType, "SUNDAY");
assert day == DayOfWeek.SUNDAY;

```

enumSet()返回EnumSet.allOf(Day.class)。

enumSet()返回EnumSet.allOf(Day.class)。

```

DayOfWeek defaultDay = DayOfWeek.SUNDAY;
DayOfWeek day = Enums.valueOf(DayOfWeek.class, "INVALID").or(defaultDay);
assert day == DayOfWeek.SUNDAY;

```

enumSet()返回EnumSet.allOf(Day.class)。

enum 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"

- enum
- enum
- enum

enum

enum

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
```



String

String

```
public enum Fruit {
    APPLE, ORANGE, STRAWBERRY, BANANA, LEMON, GRAPE_FRUIT;
}
```

Fruit.APPLE "APPLE"

### name()

name() enum enum String String

```
System.out.println(Fruit.BANANA.name()); // "BANANA"
System.out.println(Fruit.GRAPE_FRUIT.name()); // "GRAPE_FRUIT"
```

### toString()

toString() name()

toString() String

toString() name() logs stdout 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

PianoClass enum

```
enum PianoClass {
    JOHN, BEN, LUKE;
    public String getSex() {
        return "Male";
    }
    public String getLevel() {
        return "Beginner";
    }
}
```

```
}
```

enum PianoClass2 - 000000 - 000000000000000000000000

```
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
    }
}
```

enum PianoClass2 - 00000000000000000000000000000000

```
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
```

enum PianoClass2 - 00000000000000000000000000000000  
enum PianoClass2 - 00000000000000000000000000000000

```
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";
    }
}
```

enum PianoClass3 - 000000000000000000000000

```
PianoClass3 tom = PianoClass3.TOM;
PianoClass3 rita = PianoClass3.RITA;
System.out.println(tom.getLevel()); // prints Intermediate
System.out.println(rita.getSex()); // prints Female
```

enum PianoClass3 - 00000000000000000000000000000000

```
enum Friend {
    MAT("Male"),
    JOHN("Male"),
    JANE("Female");
}
```





```
    }  
  }  
}
```

EnumSet 的常用方法。

- EnumSet#range 返回一个 EnumSet，包含从指定枚举类型的第一个元素到最后一个元素的所有元素。
- EnumSet#of 返回一个 EnumSet，包含指定的枚举类型的所有元素。
- EnumSet#complementOf 返回一个 EnumSet，包含指定枚举类型的所有元素，但不包含指定 EnumSet 中的元素。

```
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");  
        }  
    }  
}
```

更多请参考 <https://riptutorial.com/zh-TW/java/topic/155/>

141: 0000

00

Java EnumMap 是 Map 的子类，它继承自 EnumMap。

java.util.EnumMap 是。

它继承自 EnumMap。

## Examples

00000000

```
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);
        }
    }
}
```

00000000 <https://riptutorial.com/zh-TW/java/topic/10158/0000>

142:

Java 10A 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/zh-TW/java/topic/10719/

143:

Java

Java [JLS8.8](#)

### Examples

```
public class TestClass {
    private String test;
}
```

```
public class TestClass {
    private String test;
    public TestClass() {

    }
}
```

```
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"
    }
}
```







144:

- java.xml;
- public java.xml;
- export com.example.foo; #dependents
- 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;
}

```

DNS

Java java.base

requires java.httpclient

exports

exports com.example.foo com.example.foo export

exports com.example.bar

<https://riptutorial.com/zh-TW/java/topic/5286/>

145:

Java... Java...

- class ArrayList <E> {} //E
- class HashMap <K,V> {} //KV
- <E> void print(E element) {} //E
- ArrayList <String>; //
- ArrayList <>; //
- new ArrayList <String>; //
- new ArrayList <>; // "diamond" Java 7

Type erasure Java... List<String> names = new ArrayList<>();

(List<String>) list ... <String> JVM List<?> List<String>;

### Examples

Param <> T

```
public class Param<T> {
    private T value;

    public T getValue() {
        return value;
    }

    public void setValue(T value) {
        this.value = value;
    }
}
```

T Integer

```
Param<Integer> integerParam = new Param<Integer>();
```

type

```
Param<String[]> stringArrayParam;
Param<int[][]> int2dArrayParam;
Param<Param<Object>> objectNestedParam;
```

Java SE 7 type diamond <>

```
Param<Integer> integerParam = new Param<>();
```

Annotations are used to provide metadata for the Java compiler and other tools. Annotations are used to provide metadata for the Java compiler and other tools. Annotations are used to provide metadata for the Java compiler and other tools.



```
public abstract class AbstractParam<T> {
    private T value;

    public T getValue() {
        return value;
    }

    public void setValue(T value) {
        this.value = value;
    }
}
```

AbstractParam is a generic class that defines a contract for its subclasses. It has a private field 'value' of type T, and two methods: 'getValue()' which returns the value, and 'setValue(T value)' which sets the value. Subclasses like Email, Age, Height, and MultiParam inherit from AbstractParam and implement these methods.

```
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> {
    // ...
}
```

Example:

```
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);
```

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);
    }
}

```

编译选项: javac Example.java  
 编译选项: javac Example.java

```

void usage() {
    consumeObjects(this.<Object>makeList("Jeff", "Atwood").stream());
}

void consumeObjects(Stream<Object> stream) { ... }

```

编译选项: javac Example.java  
 编译选项: javac Example.java

编译选项: javac Example.java

Java SE 7

Java 7 [Diamond](#) <sup>1</sup> 编译选项: javac Example.java

```
List<String> list = new LinkedList<>();
```

编译选项: javac Example.java

```
List<String> list = new LinkedList<String>();
```

编译选项: javac Example.java

```

// 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 9

1 -

<> " " . JLS Java <> Java token <> JLS Tutorials <> " " .

"A B"

.

Number Comparable .

```
public <T extends Number & Comparable<T>> void sortNumbers( List<T> n ) {  
    Collections.sort( n );  
}
```

T Number Comparable<T> Integer BigDecimal Striped64 Striped64 .

<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());
        }
    }
}

```

□□□□□□□□□□□□□□□□

```

// 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□□□□□□□□□□□□□□□□ ?□□

- ? extends T□□□□□□□□□□。 □□□□□□□□T□□□□□□□□T□□□□□□。
- ? super T□□□□□□□□□□。 □□□□□□□□□T□□□□□□□□T□□□□□□。

□□□□□□□□□□

- ? extends T
- ? super 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

Producer extends Consumer super





public interface `DoubleSeries` extends `DataSet` {  
 static `DoubleSeries` instance(Collection<Double> data) {  
 return new `DoubleSeriesImpl`(data);  
 }  
}

public interface `DataSet` {  
 List<T> data();  
}

```
public interface DataSet {  
    DS add(DS values);  
    List<T> data();  
}
```

public interface `DoubleSeries` extends `DataSet` {  
 static `DoubleSeries` instance(Collection<Double> data) {  
 return new `DoubleSeriesImpl`(data);  
 }  
}

```
public interface DoubleSeries extends DataSet {  
    static DoubleSeries instance(Collection<Double> data) {  
        return new DoubleSeriesImpl(data);  
    }  
}
```

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);  
 }  
}

```
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 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 虚拟机 (JVM) 在编译时检查 `instanceof` 操作符。如果 `T` 不是一个类类型，编译器会报错。

我们可以在 `Example` 类中定义 `isTypeAString` 方法，使其返回 `Object` 类型的布尔值。

```

class Example { // formal parameter is gone
    public boolean isTypeAString(String s) {
        return s instanceof Object; // Both <String> and <Number> are now Object
    }
}

```

JVM 在编译时检查 `T`。

编译时

编译时 `instanceof` 操作符

```

public boolean isAList(Object obj) {
    return obj instanceof List<?>;
}

```

编译时 `obj instanceof 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

```

编译时。

**instanceof** 操作符

编译时 `t instanceof` 操作符

```

class Example<T> {
    public boolean isTypeAString(T t) {
        return t instanceof String; // No compilation error this time
    }
}

```

编译时

```

class Example { // formal parameter is gone
    public boolean isTypeAString(Object t) {
        return t instanceof String; // No compilation error this time
    }
}

```

JVM 在编译时检查 `Object` 类型。

```

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

```



同义词

同义词

```
private Map<String, Object> data;  
public <T> T get(String key) {  
    return (T) data.get(key);  
}
```

同义词。同义词。Java同义词。

```
Bar bar = foo.get("bar");
```

同义词

```
List<Bar> bars = foo.get("bars");
```

同义词。List同义词。List<String>同义词。ClassCastException；同义词。

同义词API

```
public final static Key<List<Bar>> BARS = new Key<>("BARS");
```

同义词put()

```
public <T> T put(Key<T> key, T value);
```

同义词。同义词。同义词。同义词。同义词。

同义词

- [同义词](#)

同义词

同义词。同义词。同义词。同义词。同义词。同义词。同义词。同义词。同义词。

同义词

同义词。同义词。同义词。同义词 [Neal Gafter](#)同义词。

同义词

Java同义词

- [同义词TypeToken](#)
- [Spring同义词ParameterizedTypeReference](#)
- [同义词TypeReference](#)

同义词

```
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/zh-TW/java/topic/92>

146:

Stream Java 8 Collection Stream stream() parallelStream(). Stream Stream Stream.

- collection.stream
- Arrays.stream
- Stream.iterate firstValue currentValue -> nextValue
- Stream.generate -> value
- Stream.of elementOfT [elementOfT...]
- Stream.empty
- StreamSupport.stream iterable.spliterator false

### Examples

#### Streams

Stream Stream Stream lambda

:( Ideone )

```
Stream<String> fruitStream = Stream.of("apple", "banana", "pear", "kiwi", "orange");

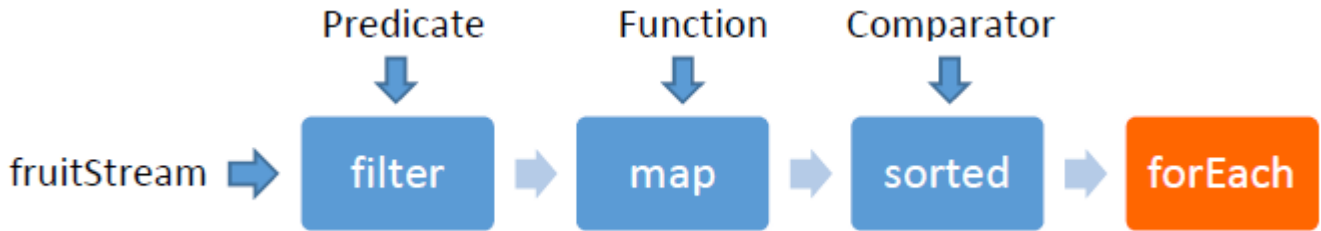
fruitStream.filter(s -> s.contains("a"))
    .map(String::toUpperCase)
    .sorted()
    .forEach(System.out::println);
```

1. Stream.of(values) String Stream Stream<String>.
2. filter() true. "a". lambda.
3. map() mapper. String String::toUpperCase String.
 

```
map() Stream<String>
.map(String::isEmpty) Stream<Boolean>
```
4. sorted() String Stream.
5. forEach(action) Stream Consumer.
 

```
Stream Consumer
```

Stream Stream.



Stream API Overview

Streams

Stream is an interface. It extends Iterable and Iterator. It is a sequence of elements.

Stream is `AutoCloseable`. It has a `close()` method. It is recommended to use `try-with-resources` for Streams.

Stream is a sequence of elements.

```
try (Stream<String> lines = Files.lines(Paths.get("somePath"))) {
    lines.forEach(System.out::println);
}
```

Stream has a `Stream.onClose()` method. It takes a `Runnable` as an argument. It is used to perform an action when the stream is closed.

```
public Stream<String> streamAndDelete(Path path) throws IOException {
    return Files.lines(path).onClose(() -> someClass.deletePath(path));
}
```

`try-with-resources` automatically calls `close()` when the resource is no longer needed.

Streams

Stream is a sequence of elements.

Stream is a sequence of elements. It can be converted to a `SortedMap` or a `List`. It can also be converted to a `Java HashMap keySet()`.

Streams

```
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
```

Ideone

Streams

Stream is a sequence of elements. It can be converted to a `Stream` or a `Stream`. It can also be converted to a `Stream`.

Streams

```
// parallel
long howManyOddNumbersParallel = integerList.parallelStream()
```

```
.filter(e -> (e % 2) == 1)
    .count();
```

```
System.out.println(howManyOddNumbersParallel); // Output: 2
```

[Ideone](#)

Container Streams

Container Streams. StreamCollection

StreamCollection

`toList()` `toSet()`

`Stream.collect()` `Stream`

```
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 HashSet<>()))
);
```

`toMap()`

`Map` `key` `Student Id` `Value` `Student 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}
```

```

Collectors.toMap Collector<T, ?, Map<K,U>> toMap(Function<? super T, ? extends K>
keyMapper, Function<? super T, ? extends U> valueMapper, BinaryOperator<U> mergeFunction)。
Map mergeFunction。

```

```
mergeFunction (s1, s2) -> s1 (s1, s2) -> s2。
```

```
ArrayList<String> List<>>
```

```
ArrayList<String> List<>>
```

```
ArrayList<String> List<>>。 ArrayList<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);

```

```
ArrayList<String> List<>>
```

```

{ Robert=[LITERATURE],
Sascha=[ENGLISH, MATH, SCIENCE, LITERATURE],
Davis=[MATH, SCIENCE, GEOGRAPHY] }

```

```
ArrayList<String> List<>>
```

```

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□□□□□□□□ MyObjects	Collectors.groupingBy(MyObject::getter)

□□□

□□□□□□□□Stream ◦ □□□Stream□□□□□□□□□□Stream□□□□□□◦ Stream□limit□□□□□□□□□□Java□□□Stream□□□□□◦  
□□□□□□□□□□Stream □□□□1□□◦ Stream□□□□□□□□□□□□◦ □□□□□□Stream□limit□□□□□□□□□□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
- 3
- 4

0

Stream.generate(). Supplier lambda .

```
// 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);
```

000

Stream count() collect() forEach() . Stream .

Stream filter() peek() .

```
IntStream.range(1, 10).filter(a -> a % 2 == 0).peek(System.out::println);
```

Ideone

Stream .

forEach peek

```
IntStream.range(1, 10).filter(a -> a % 2 == 0).forEach(System.out::println);
```

Ideone

000

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);
}
```

000

FOO  
00  
FOO  
00

Iterable Iterator<T> iterator() . lambda .

Stream .





```
aParallelStream.forEach(System.out::println);
```

Stream 的并行流

```
[]
[]
[]
[]
[]
```

Stream 的并行流。使用 `parallelStream` 方法。

Stream

Stream 的并行流。使用 `parallelStream` 方法。fork-join 策略。

Stream 的并行流。使用 `parallelStream` 方法。CPU 核心数。

Stream

Stream emit Optional Stream Optional。使用 `null` 或 `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]
```

Stream

Stream 的并行流。使用 `parallelStream` 方法。

```
Collection<String> stringList = new ArrayList<>();
Stream<String> stringStream = stringList.parallelStream();
```

Stream 的并行流。使用 `parallelStream` 方法。

```
String[] values = { "aaa", "bbbb", "ddd", "cccc" };
Stream<String> stringStream = Arrays.stream(values);
Stream<String> stringStreamAlternative = Stream.of(values);
```

Stream 的并行流。使用 `parallelStream` 方法。

```
Stream<Integer> integerStream = Stream.of(1, 2, 3);
```

Stream 的并行流。使用 `parallelStream` 方法。

```
IntStream intStream = IntStream.of(1, 2, 3);
DoubleStream doubleStream = DoubleStream.of(1.0, 2.0, 3.0);
```

Stream 的并行流。使用 `parallelStream` 方法。

```
IntStream intStream = Arrays.stream(new int[]{ 1, 2, 3 });
```

Stream。

```
int[] values= new int[]{1, 2, 3, 4, 5};
IntStream intStream = Arrays.stream(values, 1, 3);
```

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
```

List<Integer>

```
int[] ints = {1,2,3};
List<Integer> list = IntStream.of(ints).boxed().collect(Collectors.toList());
```

Java 8 IntSummaryStatistics DoubleSummaryStatistics LongSummaryStatistics  
count min max sum average。

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}
```

30 Stream 21005000000。

```
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 < 0` throws `IllegalArgumentException`
- `skip(long) limit(long)` is supported
- `skip(long)` is supported
- `skip(long) limit(long)` is supported

\*\*\*

\*\*\*\*\*

```
Collection<String> abc = Arrays.asList("a", "b", "c");
Collection<String> digits = Arrays.asList("1", "2", "3");
Collection<String> greekAbc = Arrays.asList("alpha", "beta", "gamma");
```

### 001 - `Stream`

```
final Stream<String> concat1 = Stream.concat(abc.stream(), digits.stream());

concat1.forEach(System.out::print);
// prints: abc123
```

### 002 - `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` `Stream` `StackOverflowException` \*\*\*\*\*

### `IntStream` `String`

Java `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#####。

#####lambda#####

```
List<String> sortedData2 = data.stream().sorted((s1,s2) ->
    s2.compareTo(s1)).collect(Collectors.toList());
```

#### [Sydney, New York, Mumbai, London, California, Amsterdam]

#####Comparator.reverseOrder()#####reverse。

```
List<String> reverseSortedData =
    data.stream().sorted(Comparator.reverseOrder()).collect(Collectors.toList());
```

###

Java#####IntStream int s#### LongStream #### long s#### DoubleStream #### double s#####Stream。 ####  
#####。

```
IntStream is = IntStream.of(10, 20, 30);
double average = is.average().getAsDouble(); // average is 20.0
```

#####

####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) IntStream mapToObj(mapper) String

```

1 Jon
2
3 Bauke
4
5

```

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 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\(\)](#) ◦ □□□□□□Function.identity()□□□□□□□□□□lambda  
element -> element ◦

□□□□□□□□□□□□

Collectors.toMap [javadoc](#)□□□□

□□□□□□□□□□□□□□□□Object.equals(Object) □□□□□□□□□□□□□□□□IllegalStateException ◦ □□□□□□□□□□□□□□□□  
□□□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]}
```

Ideone

Streams

Strings Web Session-ID Stream

String SecureRandom

SecureRandom setSeed()

```
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 boolean boolean Stream

```
//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);
}
```

useThisCharacter

```
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;
}
```

Streams

Stream s IntStream

Madhava Pi

$$\pi = \sqrt{12} \sum_{k=0}^{\infty} \frac{(-3)^{-k}}{2k+1} = \sqrt{12} \sum_{k=0}^{\infty} \frac{(-\frac{1}{3})^k}{2k+1} = \sqrt{12} \left( \frac{1}{1 \cdot 3^0} - \frac{1}{3 \cdot 3^1} + \frac{1}{5 \cdot 3^2} - \frac{1}{7 \cdot 3^3} + \dots \right)$$

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 s

```
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.collectOrdered()

Stream.collectOrdered() Stream.forEachOrdered() Stream.collectOrdered()

Stream.mapOrdered() Stream.filterOrdered()

Stream.sorted() Stream.sortedBy() Stream.sortedByComparing()

```
// 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);
```

Stream

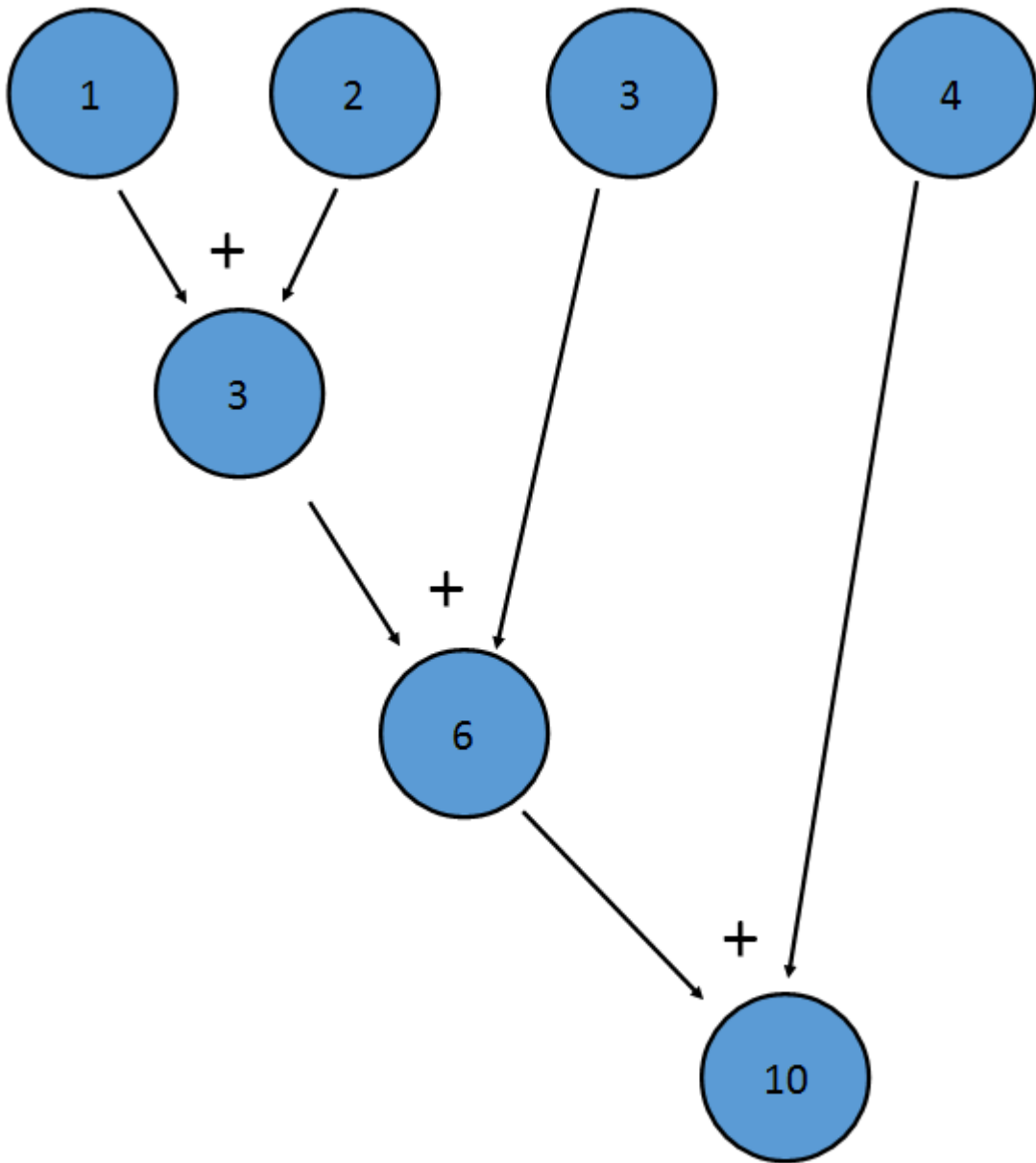
Stream spliterators.spliterator() Spliterators.spliteratorUnknownSize()

```
Iterator<String> iterator = Arrays.asList("A", "B", "C").iterator();
Spliterator<String> spliterator = Spliterators.spliteratorUnknownSize(iterator, 0);
Stream<String> stream = StreamSupport.stream(spliterator, false);
```

Stream

Stream

IntStream sum() Stream



```
int sum = ((1+2)+3)+4;
```

```
Stream<Integer> reduce = ... . reduce(sum, sum())
```

```

IntStream istr;

//Initialize istr

Optional<Integer> istr.reduce((a,b)->a+b);

```

```
Optional<Integer> Streams.
```

```
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;
});
```

LinkedList<T> 的 reduce 方法。identity 参数为 1。x+0==x。x\*1==x。identity 参数为 1。LinkedList<T> 的 reduce 方法。identity 参数为 1。x+0==x。x\*1==x。identity 参数为 1。

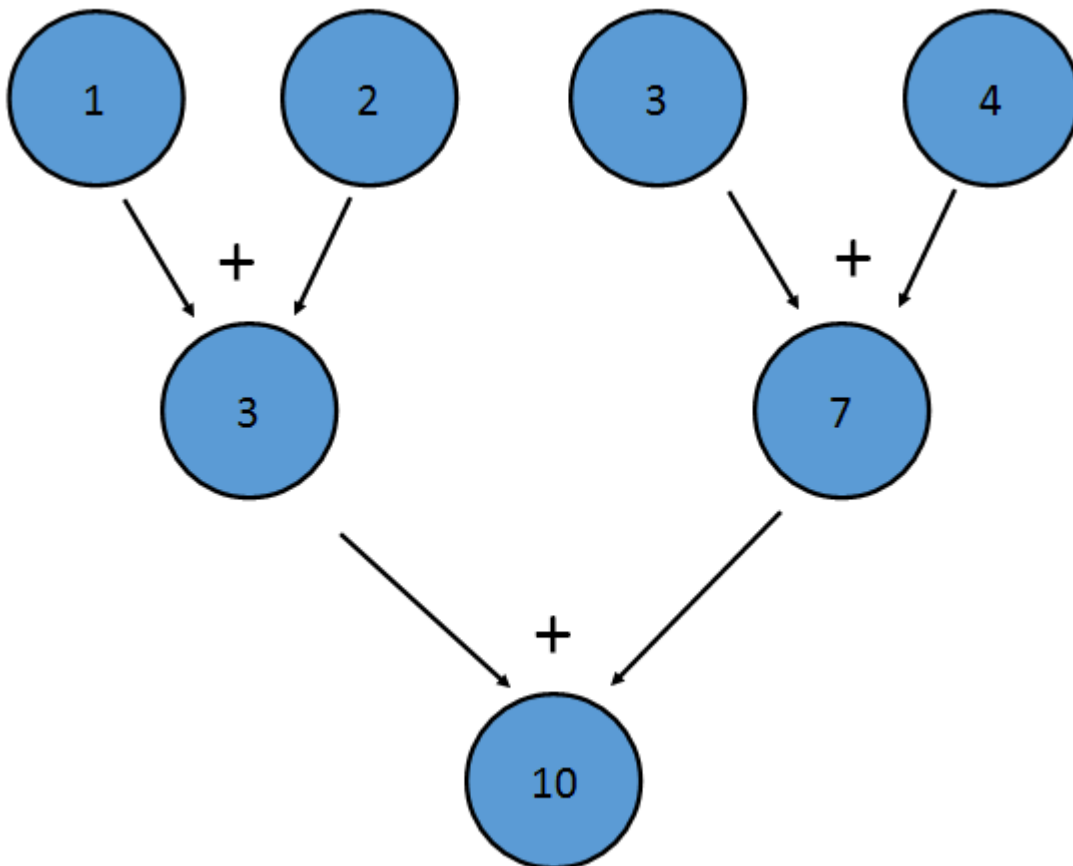
```
Stream<LinkedList<T>> listStream;

//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;
});
```

Optional 的 reduce 方法。identity 参数为 1。x+0==x。x\*1==x。identity 参数为 1。

(a+b)+c==a+(b+c) 的加法结合律。Optional 的 reduce 方法。identity 参数为 1。x+0==x。x\*1==x。identity 参数为 1。



Java Stream 的 reduce 方法。identity 参数为 1。x+0==x。x\*1==x。identity 参数为 1。

String

String stream-items。 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);
```

Collectors.joining()

```
String result = fruitStream.filter(s -> s.contains("e"))
    .map(String::toUpperCase)
    .sorted()
    .collect(Collectors.joining(", ", "Fruits: ", "."));

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

Ideone

<https://riptutorial.com/zh-TW/java/topic/88/>



```

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());
}
}

```

setter main 4 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/zh-TW/java/topic/5090/>

148:

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。

- 。
- 。
- 。
- 。
- 。
- 。

“” “” “”。 n n n + 1 。

```
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) 需要实现 Comparable 接口。

```

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 实现了 Comparable 接口。compareTo 方法。

User 对象的 id 属性。

```

List<User> users = Lists.newArrayList(
    new User(33L, "A"),
    new User(25L, "B"),
    new User(28L, "C"));
Collections.sort(users);

System.out.print(users);
// [B:25, C:28, A:33]

```

User 实现 Comparable 接口。

Comparator 接口。

```

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 `lambda` 表达式。

```
Collections.sort(users, (l, r) -> l.username.compareTo(r.username));
```

Java 8 `List` 的 `sort` 方法。

```
users.sort((l, r) -> l.username.compareTo(r.username))
```

输出

输出

输出 `String` 列表。 `List` 列表。

```
List<String> strings;
```

输出 "string1" 包含 "hello world!" 和 "goodbye"。9.2 输出

```
List<Double> doubles;
```

输出 9.2 包含 "hello world!"。

输出

输出 `NullPointerException` 包含 `strings` 和 `doubles` 为 `null`。

输出

输出 `List`

`List` 列表。 `ArrayList` 列表 `LinkedList` 列表。

```
List<String> strings = new ArrayList<String>();
```

输出

```
List<String> strings = new LinkedList<String>();
```

Java SE 7

Java SE 7 `Collection` 接口。

```
List<String> strings = new ArrayList<>();
```

输出

```
List<String> strings = new LinkedList<>();
```

输出 `Collections`

`Collections` 列表 `List` 列表。

- `emptyList()` 返回空列表。
- `singletonList(T)` 返回包含一个元素的列表。

返回空列表

- `addAll(L, T...)` 将指定集合中的所有元素添加到列表中。

示例

```
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);
```

返回空列表

List API

- `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)`

返回空列表

```
List<String> strings = new ArrayList<String>();
```

返回空列表

```
strings.add("Hello world!");
strings.add("Goodbye world!");
```

返回空列表。返回空列表

```
strings.add(0, "Program starting!");
```

返回空列表。

返回空列表

```
strings.remove("Goodbye world!");
```

返回空列表

```
strings.remove(0);
```

返回

1. 返回空列表
2. 返回空列表

返回空列表

```
strings.get(0);
```

000000000000。

00000set(int index, E element);000000000000set(int index, E element);。0000

```
strings.set(0,"This is a replacement");
```

00000000“This is a replacement”000000000000。

000set0000000000000。000000000000String0000000000001。

int indexOf(Object o);000000000000000000000000。000000000000000000-10。0000000000000000

```
strings.indexOf("This is a replacement")
```

000000000000000000“This is a replacement”0000000000。000int indexOf(Object o);000000000000int  
indexOf(Object o);0000000000000000000000000000。0000int lastIndexOf(Object o)000000000000000000000000。000000000000  
00“000000”0

```
strings.add("This is a replacement");  
strings.lastIndexOf("This is a replacement");
```

001000000000;

0000000000

000000000000String0000List000000000000“hello”0“how”0“are”0“you”0

0000000000000000for-each0000

```
public void printEachElement(List<String> list){  
    for(String s : list){  
        System.out.println(s);  
    }  
}
```

00000000

```
hello,  
how  
are  
you?
```

00000000000000000000StringBuilder0

```
public void printAsLine(List<String> list){  
    StringBuilder builder = new StringBuilder();  
    for(String s : list){  
        builder.append(s);  
    }  
    System.out.println(builder.toString());  
}
```

00000

```
hello, how are you?
```

ArrayList 的 removeAll() 方法。 移除集合中的所有元素。

ArrayList A 和 B

ArrayList A 和 B 的交集。 移除集合中的所有元素。

```
List.removeAll(Collection c);
```

运行

```
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);
}
```

输出

A [1, 3, 4, 7, 5, 2]

B [13, 32, 533, 3, 4, 2]

B 清除后 [13, 32, 533]

ArrayList 的 retainAll() 方法

ArrayList A 和 B 的交集。 移除集合中的所有元素。

ArrayList 的 retainAll() 方法。 移除集合中的所有元素。

运行

```
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);
}
```

输出

```
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()[]String[]List

ArrayList

ArrayList Java. .

AbstractList List. ArrayList. ArrayList ArrayList ArrayList. ArrayList. ArrayList. ArrayList.

ArrayList

```
List<T> myArrayList = new ArrayList<>();
```

T Generics ArrayList.

ArrayList Object. .

ArrayList add()

```
myArrayList.add(element);
```

```
myArrayList.add(index, element); //index of the element should be an int (starting from 0)
```

ArrayList 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 type

Program starting "Hello world" ArrayList. " "

```
List<String> strings = new ArrayList<String>();
strings.add("Program starting!");
strings.add("Hello world!");
strings.add("Goodbye world!");
```

strings.set(1, "Hi world");

```
strings.set(1, "Hi world");
```

strings.indexOf("Goodbye world!");

```
int pos = strings.indexOf("Goodbye world!");
if (pos >= 0) {
    strings.set(pos, "Goodbye cruel world!");
}
```

strings

1. set throws ConcurrentModificationException.
2. ArrayList set O(1) LinkedList set O(N).
3. ArrayList LinkedList indexOf O(N).

Collections

Collections.unmodifiableList

```
List<String> ls = new ArrayList<String>();
List<String> unmodifiableList = Collections.unmodifiableList(ls);
```

Collections.singletonList

```
List<String> unmodifiableList = Collections.singletonList("Only string in the list");
```

Collections

Collections.reverse

Collections

```
Collections.reverse(ls);
```

Collections

Collections.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

Collections

Collections

```
Collections.shuffle(ls);
```

java.util.Random

```
Random random = new Random(12);
Collections.shuffle(ls, random);
```

List -

List.

List

Java SE 8 java.util.List public

1.

- ArrayList
- AbstractSequentialList

2.

- 
- AttributeList
- CopyOnWriteArrayList
- 
- RoleList
- RoleUnresolvedList
- 
- 

```
public class ArrayList<E>
extends AbstractList<E>
implements List<E>, RandomAccess, Cloneable, Serializable
```

ArrayList List. ArrayList List.

ArrayList 100

```
List<Integer> myList = new ArrayList<Integer>(100); // Constructs an empty list with the
specified initial capacity.
```

- PROS

size isEmpty get set iterator listIterator List

```
int e1 = myList.get(0); // \
int e2 = myList.get(10); // | => All the same constant cost => O(1)
myList.set(2,10); // /
```

-

n n

n





## 149: SortedMap

### Examples

- SortedMap 是 Map 的子类。
- SortedMap 是有序的。

- SortedMap 是有序的。
- SortedMap 的 firstKey() 方法。
- SortedMap 的 headMap() 方法。
- SortedMap 的 lastKey() 方法。
- SortedMap 的 subMap(Object start, Object end) 方法。
- SortedMap 的 tailMap(Object start) 方法。

```
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();

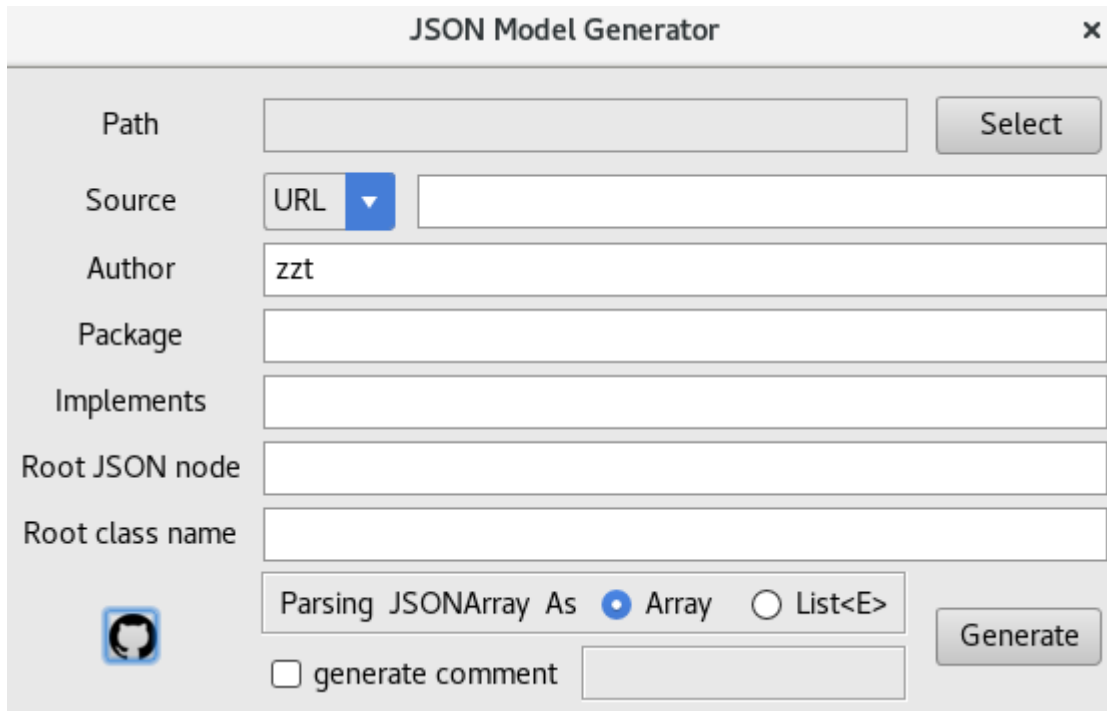
    // Deposit 1000 into Zara's account
    double balance = ((Double)tm.get("Zara")).doubleValue();
    tm.put("Zara", new Double(balance + 1000));
    System.out.println("Zara's new balance: " + tm.get("Zara"));
}
```

SortedMap <https://riptutorial.com/zh-TW/java/topic/10748/sortedmap>

### Examples

#### JSON POJO

- IntelliJ IntelliJ JSON Model Generator
- 
- UI 'Path' 'Source' 'Package'



- 

Java <https://riptutorial.com/zh-TW/java/topic/9400/java>

00 151: 0000

00

- `LocalTime time = LocalTime.now(); //000000000000`
- `LocalTime time = LocalTime.MIDNIGHT; // 00:00`
- `LocalTime time = LocalTime.NOON; // 12:00`
- `LocalTime time = LocalTime.of(12,12,45); // 12:12:45`

00

<code>LocalTime.of(13, 12, 11)</code>	<code>1312</code>
<code>LocalTime.MIDNIGHT</code>	<code>00:00</code>
<code>LocalTime.NOON</code>	<code>12:00</code>
<code>LocalTime.now()</code>	
<code>LocalTime.MAX</code>	<code>235959.999999999</code>
<code>LocalTime.MIN</code>	<code>00:00</code>
<code>LocalTime.ofSecondOfDay(84399)</code>	<code>23:59:59</code>
<code>LocalTime.ofNanoOfDay(2000000000)</code>	<code>00:00:02</code>

00

0000000 LocalTime0000000000。 0000000。 0000000000000。

00000000000000000000equals00。

00000 java.time0。

### Examples

0000

000000000000000000

```
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)
```

00000000

```
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");

        LocalDateTime now = LocalDateTime.now();
        LocalDateTime now1 = LocalDateTime.now(zone1);
        LocalDateTime now2 = LocalDateTime.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.LocalDateTime;
import java.time.temporal.ChronoUnit;

public class AmountOfTime {

    public static void main(String[] args) {

        LocalDateTime start = LocalDateTime.of(1, 0, 0); // hour, minute, second
        LocalDateTime end = LocalDateTime.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 UnsupportedOperationException.
        // The following methods are examples thereof.
    }
}

```



## 152: BigInteger

00

`BigInteger` 类是 Java 的任意精度有符号整数。它扩展了 `java.lang.Math` 类。它包含用于操作大整数的方法。

00

- `BigInteger variable_name = new BigInteger("12345678901234567890");` // 十进制字符串
- `BigInteger variable_name = new BigInteger("10101011010101001010100110011101011000111110000101011010010")` // 二进制字符串
- `BigInteger variable_name = new BigInteger("ab54a98ceb1f0800", 16);` // 十六进制字符串
- `BigInteger variable_name = new BigInteger(64, new Random());` // 64 位的随机数
- `BigInteger variable_name = new BigInteger(new byte[] {0, -85, 84, -87, -116, -21, 31, 10, -46});` // 字节数组
- `BigInteger variable_name = new BigInteger(1, new byte[] { -85, 84, -87, -116, -21, 31, 10, -46 });` // 字节数组

00

`BigInteger` 类包含 `sum` 属性。

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

00 00 00 sum 00 00 00 00。

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

---

Java SE 8

`BigInteger` 类包含 `BigInteger` 的 `-22147483647` 到 `22147483647` 的任意精度有符号整数。它还包含 `BigInteger` 的 `20` 个方法。

### Examples

000

`java.math.BigInteger` 类包含 `java.lang.Math` 类。它还包含 `java.math.BigInteger` 类。

00 `long` 到 `int` 和 `BigInteger` 的转换。

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

00 00 00 00 00

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

000 `intValue` 到 `long` 的转换。

---

String和BigInteger

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

String和BigInteger

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

Java的BigInteger。 的。

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

-128的BigInteger。

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

128的BigInteger。 的。

的。

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

BigInteger.TWO 的"2"的private。

## BigIntegers

BigIntegers和Java的String。

的。

```
BigInteger one = BigInteger.valueOf(1);  
BigInteger two = BigInteger.valueOf(2);  
  
if(one.equals(two)){  
    System.out.println("Equal");  
}  
else{  
    System.out.println("Not Equal");  
}
```

的。

Not Equal

的。

的BigIntegers

- ==的BigIntegers。
- equals()的BigIntegers。





000 10/5 = 2

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

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

0002

000 17/4 = 4

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

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

0004

000 10 \* 5 = 50

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

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

00050

000 10 ^ 3 = 1000

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

0001000

000 1006 = 4

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

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

0004

**GCD** 12 18 00000000GCD 6。

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

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

0006

0000BigIntegers

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



0002000010 << 1

00000000

```
BigInteger val1 = new BigInteger("10");
val1.not();
```

0005

NAND

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

val1.andNot(val2);
```

0007

#### BigIntegers

BigInteger 的构造函数之一，使用 `java.util.Random` 生成指定位数的 `BigInteger`。

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

生成指定位数的 `BigInteger`。

```
new BigInteger(2147483647, sourceOfRandomness)
```

---

生成指定位数的 `BigInteger`。

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

生成指定位数的 `BigInteger`。

```
import java.security.SecureRandom;

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

生成指定位数的 `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;
    }
});
```

生成指定位数的 `BigInteger`

## 153: NumberFormat

### Examples

#### NumberFormat

Example 1: java.util.Locale

```
Locale locale = new Locale("en", "IN");
NumberFormat numberFormat = NumberFormat.getInstance(locale);
```

Example 2:

1. Example

```
numberFormat.format(10000000.99);
```

2. Example

```
NumberFormat currencyFormat = NumberFormat.getCurrencyInstance(locale);
currencyFormat.format(10340.999);
```

3. Example

```
NumberFormat percentageFormat = NumberFormat.getPercentInstance(locale);
percentageFormat.format(10929.999);
```

4. Example

```
numberFormat.setMinimumIntegerDigits(int digits)
numberFormat.setMaximumIntegerDigits(int digits)
numberFormat.setMinimumFractionDigits(int digits)
numberFormat.setMaximumFractionDigits(int digits)
```

Example 3: NumberFormat <https://riptutorial.com/zh-TW/java/topic/7399/numberformat>

## 154: ServiceLoader

ServiceLoader= .jar . / .

/ServiceLoader .

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 / / servicetest.Logger

META-INF/services/servicetest.LoggerLoggerLogger。

```

servicetest.logger.ConsoleLogger
servicetest.logger.FileLogger

```



mainServiceLoader。

```

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
-

accounting-api.jar

```
package example;

public interface AccountingService {

    long getBalance();
}
```

accounting-impl.jar

```
package example.impl;
import example.AccountingService;

public interface DefaultAccountingService implements AccountingService {

    public long getBalance() {
        return balanceFromDB();
    }

    private long balanceFromDB(){
        ...
    }
}
```

accounting-impl.jar AccountingService。META-INF/services/

- META-INF/services/example.AccountingService

```
example.impl.DefaultAccountingService
```

jar 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 iterator() for-each。

ServiceLoader <https://riptutorial.com/zh-TW/java/topic/5433/service-loader>



## 155: StringBuffer

**Java StringBuffer**

### Examples

#### String Buffer

- 
- 
- 

- `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)`
- `public char charAt(int index)`
- `public int length()`
- `public String substring(int beginIndex)`
- `public String substring(int beginIndex, int endIndex)`

`String s = new StringBuffer("study");`

```
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/zh-TW/java/topic/10757/stringbuffer>

## 156: 156

- new Socket("localhost", 1234); // 创建 Socket 对象
- SocketServer("localhost", 1234); // 创建 SocketServer 对象
- socketServer.accept(); // 接受 Socket 连接

### Examples

Example 1

Example 2

```
// 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.  
}
```

Example 3

Example 4

```
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();
```

Example 5

```
// 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 outputStream = socket.getOutputStream();  
PrintWriter writer = new PrintWriter(  
    new OutputStreamWriter(outputStream, StandardCharsets.UTF_8));  
writer.println("Hello world!");
```



```

        } 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 outputStream = socket.getOutputStream();
            PrintWriter writer = new PrintWriter(
                new OutputStreamWriter(outputStream, 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 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();
    }
}
}
}

```

KeyTrustKeyStore。 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 outputStream = socket.getOutputStream();
        InputStream inputStream = socket.getInputStream();
        BufferedReader reader = new BufferedReader(new InputStreamReader(inputStream));
        PrintWriter writer = new PrintWriter(new BufferedOutputStream(outputStream));

        //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/1.1 200 OK HTTP HTTP HTML

readFully() EOF. readLine() Apache commons-io IOUtils

Java HTTP Apache HTTP Google 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);
    }
}

```







```

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();
}

```

Channel

```

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 */ }

```

- 
- 
-

🔗 <https://riptutorial.com/zh-TW/java/topic/149>

157:

Java int -> Integer, double -> Double ...  
Objects

- <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<>();
```

int Integer

```
for (int i = 0; i < 10; i++)
  ints.add(i);
```

ints.add(i);

```
ints.add(Integer.valueOf(i));
```

Integer#valueOf JVM Integer

- 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 remove(int index)

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] operator ==Integers. -XX:AutoBoxCacheMax JVM-XX:AutoBoxCacheMax. -XX:AutoBoxCacheMax=1000-XX:AutoBoxCacheMax=1000 true

```
Integer a = 1000;
Integer b = 1000;
System.out.println(a == b); // true
```

### if Boolean

if Boolean

```
Boolean a = Boolean.TRUE;
if (a) { // a gets converted to boolean
    System.out.println("It works!");
}
```

while do while for

Boolean null NullPointerException

### NullPointerException

```
Integer arg = null;
int x = arg;
```

java.lang.NullPointerException

int null

NullPointerException

null

```
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

HashMap 的 Entry 数组 table 的 Entry 的 key 和 value 的引用。

```
class Example {
    int primitive; // Stored directly in the class `Example`
    Integer boxed; // Reference to another memory location
}
```

boxed 的引用指向 primitive 的引用。

HashMap 的 Entry 数组 table 的 Entry 的 key 和 value 的引用。

```
int sumOfSquares = 0;
for(int i = 256; i < 1024; i++) {
    sumOfSquares += square.get(i);
}
```

HashMap 的 Entry 数组 table 的 Entry 的 key 和 value 的引用。

```
int sumOfSquares = 0;
for(int i = 256; i < 1024; i++) {
    sumOfSquares += square.get(Integer.valueOf(i)).intValue();
}
```

HashMap 的 Entry 数组 table 的 Entry 的 key 和 value 的引用。

HashMap 的 Entry 数组 table 的 Entry 的 key 和 value 的引用。

Java 8 的 IntStream。

Java 的 Integer 的 valueOf 的 autoboxing。Integer 的 -128 到 127。

Integer 的 int。

1。

HashMap 的 Entry 数组 table 的 Entry 的 key 和 value 的引用。

HashMap 的 Entry 数组 table 的 Entry 的 key 和 value 的引用。

```
int i;
Integer j;
void ex_method(Integer i)//Is a valid statement
void ex_method1(int j)//Is a valid statement
```

2。

HashMap 的 Entry 数组 table 的 Entry 的 key 和 value 的引用。

HashMap 的 Entry 数组 table 的 Entry 的 key 和 value 的引用。

```
int i;
Integer j;
int ex_method()
{...
return j;}//Is a valid statement
Integer ex_method1()
{...
return j;}//Is a valid statement
```

```
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
```

```
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
    }
}
```

<https://riptutorial.com/zh-TW/java/topic/138>

158:

API 1.5 Process ProcessBuilder.start()。

waitFor() /。 calc.exe notepad.exe。

### Examples

Java <1.5

windows。 /。

```
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();
}

```

Parameters

- `String[]`
- `String[].exe`
- `File` or `String`

Example

Example 1.

Example 2.

Example 3.

Example

```

//code setting up the commands omitted for brevity...

```

```

ProcessBuilder pb = new ProcessBuilder(cmds);

```

```

try {
    Process p = pb.start();
    p.waitFor();
} catch (IOException e) {
    e.printStackTrace();
} catch (InterruptedException e) {
    e.printStackTrace();
}

```

```

//more lines of code here...

```

### ch.vorburger.exec

Example 1. [Apache Commons Exec](#) Example 1.

```

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();

```

Example 2. `Runtime.exec` Example 2.

Runtime.exec(String ...) 与 Runtime.exec(String) 的区别。前者可以执行任何命令，后者只能执行 Java 命令。

exec(String) 方法

```
Process p = Runtime.exec("mkdir /tmp/testDir");
p.waitFor();
if (p.exitValue() == 0) {
    System.out.println("created the directory");
}
```

问题

如何执行系统命令

```
Process p = Runtime.exec("mkdir " + dirPath);
// ...
```

dirPath 为 "/home/user/My Documents"。exec(String) 方法只能执行 Java 命令。

```
"mkdir /home/user/My Documents"
```

问题

```
"mkdir", "/home/user/My", "Documents"
```

exec("mkdir") 只能执行 Java 命令。

如何执行系统命令。解决方案

```
"mkdir \"/home/user/My Documents\""
```

问题

```
"mkdir", "\"/home/user/My", "Documents\""
```

exec("mkdir") 只能执行 Java 命令。解决方案

exec(String ...) 方法

```
Process p = Runtime.exec("mkdir", dirPath);
// ...
```

dirPath 为 "/home/user/My Documents"。exec 方法可以执行任何命令。

如何执行系统命令 **shell**

如何执行系统命令。解决方案

```
Process p = Runtime.exec("find / -name *.java -print 2>/dev/null");
```

问题

```
Process p = Runtime.exec("find source -name *.java | xargs grep package");
```

如何执行系统命令。解决方案。exec 方法可以执行任何命令。

Process p = Runtime.exec("find / -name \*.java -print 2>/dev/null");

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");

```
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");

```
Process p = Runtime.exec("bash", "-c",
    "find source -name \\*.java | xargs grep package");
```

Process p = Runtime.exec("cd", "/tmp"); // Change java app's home directory

### Shell

Process p = Runtime.exec("export", "NAME=value"); // Export NAME to the java app's environment

```
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

```
Process p = Runtime.exec("export", "NAME=value"); // Export NAME to the java app's environment
```

1. "cd" "export" shell

2. shell "cd" "java" exec exec

- 1. "cd" "export" shell
- 2. shell "cd" "java" exec exec

Process p = Runtime.exec("cd", "/tmp");

1 - ProcessBuilder

2 - ...

<https://riptutorial.com/zh-TW/java/topic/4682>

159:

Java

### Examples

1 + 2 x 3

- 1. 1020000003. 0000009. 000000000000000000 ( 1 + 2 ) x 3.
- 2. 100000000203. 00000007. 000000000000000000 1 + ( 2 x 3 ).

Java

```
1 + 2 * 3
```

+000000 \*00000000000000007000009.

Operator	Precedence	Associativity	Direction
. ( expr ) new . [ expr ] primary ( expr... ) primary :: name	15		
expr ++   expr --	14		-
++ expr   -- expr + expr   - expr   ~ expr , ! EXPR ( ) expr	13		-
* /	12		
+ -	11		
<< >> >>>	10		
<> <=> = instanceof	9		
== !=	8		

Operator	Precedence	Associativity	Direction
AND	7		Left to Right
^	6		Left to Right
OR	5		Left to Right
&&	4		Left to Right
	3		Left to Right
1	2		Left to Right
Lambda <sup>1</sup>	1		Left to Right

<sup>1</sup> Lambda

String

String

- String
- String
- + - ~!
- \* / % + - << >> >>> < <= > >= == != & ^ | && ||
- ? :
- 
- final
- <TypeName> . <Identifier>

class instanceof

String "interned" String FP-strict

String

String

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 expression
byte b4 = (byte) (b1 + 1); // OK
```

do while for while

```
while (false) {
    doSomething();        // Error - statement not reachable
}
boolean flag = false;
while (flag) {
    doSomething();        // OK
}
```

if Java then else if C++ Java

static final

JLS 15.28

Java

- 
- 
- 
- 

```
int i = method1() + method2();
```

1. =i
2. + method1()
3. + method2()
4. +
5. =i

method1 method2

```
int i = 1;
intArray[i] = ++i + 1;
```

1. =intArray[1]
2. i 1 2
3. +
4. + 2 + 1 - > 3
5. =intArray[1] 3

0000000000=000000000000++i000000000000。

000

- [JLS 15.7 - 0000](#)

000000

Java0000000000000000。 00000000

```
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
```

0000000000000000

- 00000000
  - 00000;00someIdentifier
  - 000000;00MyClass.someField
- 000000
  - 00;001 0 1.0 0 'X' "hello" 0 false0null
  - 00000;00MyClass.class
  - this0<TypeName> . this
  - 00000;00( a + b )
  - 00000000;00 new MyClass(1, 2, 3)
  - 000000000;00new int[3]
  - 00000000;00obj.someField0this.someField
  - 00000000;00vector[21]
  - 0000;00obj.doIt(1, 2, 3)
  - 00000Java 8000000;00MyClass::doIt
- 00000000;00!a0i++
- 00000000;00a + b0obj == null
- 00000000;00(obj == null) ? 1 : obj.getCount()
- Lambda0000Java 8000000;00obj -> obj.getCount()

00000000000000000000000000。

- [Operators](#)000000000000000000000000。
- [Lambda](#)000000000000000000000000000000。
- “ 0000 ”000000000000000000000000。
- [Arrays](#)000000000000000000000000000000。
- [Literals](#)000000000000000000000000000000。

00000000

000。 000000000000。

00000Java 80000000000000000000000000000 0

- 00000000
- 000000000
- 00000000
- 00000000
- 000000
- Lambda0000

00000poly00000000000000000000000000 ;00000000。





## 160: java.util.logging

### Examples

Example 1

Example 2 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()});
    }
}
```

Example 3

Java Logging Api 7.0.0

- SEVERE
- WARNING
  - INFO
  - CONFIG
  - FINE
  - FINER
  - FINEST

Example 4 INFO

Example 5 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()###Java###.

javap -c LoggingComplex.class#####



Javadoc is a Java tool. Javadoc is Sun Microsystems' Java API HTML API. HTML API.

- / \*\* - JavaDoc
- @author //
- @version //
- @param // @param
- @return // void
- @exception // ArrayIndexOutOfBoundsException @exception
- @throws // @exception
- @see // package.Class
- @since // JDK-8 java.util.Optional <T>
- @serial @ serialField @ serialData // serialVersionUID
- @deprecated // java.io.StringBufferInputStream
- {@link} // {@link #setDefaultCloseOperation(int closeOperation) JFrame #setDefaultCloseOperation}
- {@linkplain} // {@link}
- {@code} // HTML { @ code <html> </ html> }
- {@literal} // {@code}
- {@value} // JFrame.EXIT\_ON\_CLOSE {@value} AppConstants.APP\_NAME
- {@docRoot} // JavaDoc HTML <a href="{@docRoot}/credits.html"></a>
- HTML <code>"Hi cookies".substring(3)</ code>
- \* / - JavaDoc

Javadoc is JDK's HTML API. 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

```



```
/**
 * 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 @param @return @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 further

#### 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] HTML [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/zh-TW/java/topic/140/###java###>

## 162:

Java 的 注解 是 一种 特殊 的 标识 符。 它们 可以 被 用在 类、 方法、 成员 变量、 局部 变量、 参数 和 代码 块 上。

- `@AnnotationName // 'value'`
- `@AnnotationName someValue // 'value'`
- `@AnnotationName param1 = value1 // 'value'`
- `@AnnotationName param1 = value1 param2 = value2 // 'value'`
- `@AnnotationName param1 = {1, 2, 3} // 'value'`
- `@AnnotationName {value1} // 'value'`

## 163:

Java 的 注解 是 一种 特殊 的 标识 符。 它们 可以 被 用在 类、 方法、 成员 变量、 局部 变量、 参数 和 代码 块 上。

- String
- Class
- 成员 变量
- 局部 变量
- 参数

### Examples

Java 的 注解 是 一种 特殊 的 标识 符。 它们 可以 被 用在 类、 方法、 成员 变量、 局部 变量、 参数 和 代码 块 上。

@

Java 的 注解 是 一种 特殊 的 标识 符。 它们 可以 被 用在 类、 方法、 成员 变量、 局部 变量、 参数 和 代码 块 上。

```
public class Vehicle {
    public void drive() {
        System.out.println("I am driving");
    }
}

class Car extends Vehicle {
    // Fine
    @Override
    public void drive() {
        System.out.println("Brrrm, brm");
    }
}
```

```
abstract class Animal {
    public abstract void makeNoise();
}
```



- 。

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
}

@SuppressWarnings("finally")
public boolean checkData() {
    // method calling return from within finally block
}
```

。

```
ComplexAlgorithm algorithm = new ComplexAlgorithm();
@SuppressWarnings("deprecation") algorithm.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。

```
@FunctionalInterface
public interface ITrade {
    public boolean check(Trade t);
}
```





Annotations are used to provide metadata for the class, fields, methods, and parameters.

### @Target

@Target is used to specify the elements to which the annotation can be applied.

```
@Target(ElementType.METHOD)
@interface MyAnnotation {
    // this annotation can only be applied to methods
}
```

@Target({ElementType.FIELD, ElementType.TYPE})

Examples

ElementType	Annotation	Example
ANNOTATION_TYPE	Annotation	<pre>@Retention(RetentionPolicy.RUNTIME) @interface MyAnnotation</pre>
CONSTRUCTOR	Annotation	<pre>@MyAnnotation public MyClass() {}</pre>
FIELD	Annotation	<pre>@XmlAttribute private int count;</pre>
LOCAL_VARIABLE	Annotation	<pre>for (@LoopVariable int i = 0; i &lt; 100; i++) {     @Unused     String resultVariable; }</pre>
PACKAGE	Annotation	<pre>@Deprecated package-info.java</pre>
METHOD	Annotation	<pre>@XmlElement public int getCount() {...}</pre>
PARAMETER	Annotation	<pre>public Rectangle(     @NamedArg("width") double width,     @NamedArg("height") double height) {     ... }</pre>

ElementType	Annotation	Example
		}
		@XmlElement public class Report {}

Java SE 8

ElementType	Annotation	Example
TYPE_PARAMETER		public <@MyAnnotation T> void f(T t) {}
TYPE_USE		Object o = "42"; String s = (@MyAnnotation String) o;

## @MyAnnotation

@Retention(RetentionPolicy.RUNTIME). @MyAnnotation.class. @MyAnnotation

```
@Retention(RetentionPolicy.RUNTIME)
@interface MyAnnotation {
    // this annotation can be accessed with reflections at runtime
}
```

Example

RetentionPolicy	Annotation
	.class
	.class. .

## @Documented

@Documented API javadoc. @Documented . @Documented

## @Inherited

@Inherited. @Inherited

- 
- 

## @Repeatable

Java 8 @Repeatable 注解。 使用 @Repeatable 注解。 使用 @Repeatable 注解。

使用 @Repeatable 注解。

使用 @Repeatable 注解。 使用 @Repeatable 注解。 使用 @Repeatable 注解。

```
@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);
        }
    }
}
```





```
}
```

□□□□

□□□□□□□□□□□□□□□□□□□□. □□□□@Inherited□□□□□□□□□□□□□□□□

□

□□□□2□□□□

```
@Inherited
@Target (ElementType.TYPE)
@Retention (RetentionPolicy.RUNTIME)
public @interface InheritedAnnotationType {
}
```

□

```
@Target (ElementType.TYPE)
@Retention (RetentionPolicy.RUNTIME)
public @interface UninheritedAnnotationType {
}
```

□□□□□□□□□□

```
@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
```

□□□□□□□□□□□□□□□□□□□□.



```

        // 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) {
    messenger.printMessage(Diagnostic.Kind.ERROR, message, element);
}

@Override
public void init(ProcessingEnvironment processingEnvironment) {
    super.init(processingEnvironment);

    // get messenger for printing errors
    messenger = processingEnvironment.getMessager();
}
}
}

```



ServiceLoader

META-INF/services/javax.annotation.processing.Processor jar

annotation.processor.SetterProcessor

AnnotationProcessor.jar



编译并运行。编译并运行。

```
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) {}

}
```

## 编译 javac

编译并运行 AnnotationProcessorTest

```
javac -cp AnnotationProcessor.jar AnnotationProcessorTest.java
```

编译并运行

```
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
```

编译并运行。编译并运行。

编译并运行 javac -proc:none。编译并运行 -proc:only -proc:only。

## IDE

### Netbeans

编译并运行 NetBeans。编译并运行。

1. Project Properties > Build > Compiling
2. Enable Annotation Processing in Editor " Enable Annotation Processing" Enable Annotation Processing in Editor
3. Add
4. Ok。

编译并运行



。 。@IsOpen。Closeable。 。

```
public class MyResource extends Closeable {
    public void update(@IsOpen MyResource this, int value) {
        // ...
    }

    public void close() {
        // ...
    }
}
```

。@IsOpenthis。 。

- updatethis。
- thisupdate。

。

Annotation。@SuppressWarnings。

```
public @interface SuppressWarnings {
    String[] value();
}
```

value。Array。

```
@SuppressWarnings({"unused"})
@SuppressWarnings({"unused", "javadoc"})
```

。

```
@SuppressWarnings("unused")
```

<https://riptutorial.com/zh-TW/java/topic/157/>

163:

/ / O.

### Examples

#### BufferedReader

BufferedReaderReaderReader

1. BufferedReaderReader. / O.
2. BufferedReader.

#### BufferedReader

BufferedReaderReader. Reader. ReaderBufferedReader. . . . .  
BufferedReader`Reader. . . . .

```
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. . . . . finally  
BufferedReader.
2. tryFileReader. . . . . BufferedReaderReader. . . . .

## BufferedReader

### BufferedReader.readLine

#### List

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();
    }
}

```

StringWriter BufferedReader

<https://riptutorial.com/zh-TW/java/topic/10618/>

164:

## Examples

JAR

- javax.money:money-api:1.0(JSR354)api
- org.javamoney:moneta:1.0
- javax-money-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/zh-TW/java/topic/8359/>

## 165: getResource

00

Java 的 `ClassLoader` 和 `Class` 类提供了 `getResource` 和 `getResourceAsStream` 方法。

00

这些方法用于从类加载器或类对象中获取资源。它们返回 `URL` 或 `InputStream`。

`ClassLoader.getResource` 和 `ClassLoader.getResourceAsStream` 方法接受一个资源名称作为参数。资源名称可以是 `package / name` 的形式，也可以是 `name` 的形式。

`getResource` 和 `getResourceAsStream` 方法返回 `URL` 或 `InputStream`。

`ClassLoader` 和 `Class` 类提供了 `getResource` 和 `getResourceAsStream` 方法。

- `ClassLoader.getResource` 和 `ClassLoader.getResourceAsStream` 方法。
- `Class.getResource` 和 `Class.getResourceAsStream` 方法。

资源名称可以是 `package / name` 的形式，也可以是 `name` 的形式。

00

- `ClassLoader.getResource` 和 `ClassLoader.getResourceAsStream` 方法。
- `Class.getResource` 和 `Class.getResourceAsStream` 方法。资源名称可以是 `package / name` 的形式，也可以是 `name` 的形式。

00

```
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);
    }
}
```

这些方法用于从类加载器或类对象中获取资源。它们返回 `URL` 或 `InputStream`。资源名称可以是 `package / name` 的形式，也可以是 `name` 的形式。

`getResource` 和 `getResourceAsStream` 方法返回 `URL` 或 `InputStream`。如果资源不存在，则返回 `null`。

资源名称可以是 `package / name` 的形式，也可以是 `name` 的形式。在 `URL` 上使用 `getFile` 方法可以获取资源的文件路径。

这些方法用于从类加载器或类对象中获取资源。它们返回 `URL` 或 `InputStream`。



4. getResource()。

5. getResourceAsStream()。

getResource() 返回 URL。getResourceAsStream() 返回 InputStream。

getResourceAsStream()

getResourceAsStream() 返回 InputStream。getResourceAsStream() 返回 InputStream。

getResourceAsStream() 返回 InputStream。getResourceAsStream() 返回 InputStream。

getResourceAsStream()

getResourceAsStream() 返回 InputStream。getResourceAsStream() 返回 InputStream。

- getResourceAsStream() 返回 InputStream。getResourceAsStream() 返回 InputStream。
- Object.getClass() 返回 Class。Object.getClass() 返回 Class。
- Class.forName(String) 返回 Class。Class.forName("java.lang.String") 返回 Class。

getResourceAsStream() 返回 InputStream。getResourceAsStream() 返回 InputStream。

getResourceAsStream()

getResourceAsStream() 返回 InputStream。getResourceAsStream() 返回 InputStream。

ClassLoader.getResource(path) ClassLoader.getResources(path)	URL。
ClassLoader.getResources(path) Class.getResources(path)	Enumeration<URL>foo.barURL;。
ClassLoader.getResourceAsStream(path) Class.getResourceStream(path)	InputStream foo.bar。

getResourceAsStream()

- ClassLoader 返回 Class。ClassLoader 返回 Class。
- ClassLoader 返回 Class。ClassLoader 返回 Class。
- getResource() 和 getResourceAsStream() 返回 null，而 getResources() 返回一个空的 Enumeration。
- URL 返回 URL.toStream()。URL 返回 URL。URL 返回 URL。URL 返回 URL。
- getResourceAsStream() 返回 URL.toStream() 返回 InputStream。getResourceAsStream() 返回 InputStream。

getResourceAsStream() 返回 InputStream。getResourceAsStream() 返回 InputStream。

### Examples

super

- 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 的 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 的 super 。

。

super 的 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 的 super 。

<https://riptutorial.com/zh-TW/java/topic/5764/>

167: 000000

### Examples

String

- String.valueOf() method

Code

```
int i = 42;
String string = String.valueOf(i);
//string now equals "42".
```

String.valueOf() for float, double, boolean, Object

- toString() method of Object, Date, Java classes

Code

```
Foo foo = new Foo(); //Any class.
String stringifiedFoo = foo.toString();
```

stringifiedFoo == foo.toString()

String concatenation

```
int i = 10;
String str = i + "";
```

Code

```
String str = 10 + "";
```

String.getBytes()

String.getBytes() method of Java

```
byte[] bytes = "test".getBytes(StandardCharsets.UTF_8);
```

Code

```
String testString = new String(bytes, StandardCharsets.UTF_8);
```

String.getBytes() example

```
import static java.nio.charset.StandardCharsets.UTF_8;
...
byte[] bytes = "test".getBytes(UTF_8);
```

String.getBytes() example

```
byte[] bytes = "test".getBytes("UTF-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);
```

parseXxx Integer.parseInt String int

```
String string = "59";
int primitive = Integer.parseInt(string);
```

String valueOf

```
String string = "59";
Integer wrapper = Integer.valueOf(string);
```

Java 5

```
String string = "59";
Integer wrapper = Integer.parseInt(string); // 'int' result is autoboxed
```

byte short int long float double Byte Short Integer Long Float Double

```
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) 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);
```



```
boolean tryParseInt (String value) {
    try {
        Integer.parseInt(value);
        return true;
    } catch (NumberFormatException e) {
        return false;
    }
}
```

tryParse... parse...

<https://riptutorial.com/zh-TW/java/topic/6678/>

## 168: Iterable

java.util.Iterator extends java.lang.Iterable  
java.util.Iterator implements java.lang.Iterable  
java.util.Iterator 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"

```
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 的 iterator() 方法返回一个 Iterator 对象。该对象实现了 Iterator 接口。该接口定义了 hasNext() 和 next() 方法。

```
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");
            }
        };
    }
}
```

测试

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

Iterator 的 next() 方法返回下一个元素。hasNext() 方法返回 true 表示还有下一个元素。remove() 方法删除下一个元素。

测试

Iterator.remove() 方法删除下一个元素。Iterator.next() 方法返回下一个元素。

```
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());
```



## 169:

Java

Java

Java

- 
- 
- 

### Examples

Java

- 
- 
- 

Concatenation

```
String s1 = "a String";
String s2 = "This is " + s1; // s2 contains "This is a String"
```

String

- 
- 

```
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@xxxxxxxxx"
```

s5

Concatenation

```
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)

- NString
- 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 JIT。

- 
- 

+ - \*/

Java 7

- +
  - 
  -
- -
  - 
  -

- 0 \* 0 = 0。
- 0 / 0 = 0。
- 0<sup>0</sup> = 1。

0 = 0。 “0” JLS。 “0” “0”。

0

byte [] short [] char [] int [] long [] float [] double [] java.lang.  
Byte [] Character [] Short [] Integer [] Long [] Float [] Double。

0

- double [] Double [] double。
- float [] Float [] float。
- long [] Long [] long。
- int。 [] byte [] short [] char [] int。

0

- double [] double 64 IEE 754。
- float [] float 32 IEE 754。
- long [] long 64。
- int [] int 32。

0

- 0。
- 0
  - int [] long。
  - float [] double。
  - 0 IEE 768 “”。

0

/ n 0 d 0 q 0 0。

Java。 [JLS Section 15.17.2](#) Java。

$n \cdot d \cdot q$   $|d \cdot q| \leq |n|$ 。  $|n| \geq |d| \cdot |q|$   $|n| \geq |d| \cdot n \cdot d$   
 $|n| \geq |d| \cdot |q|$   $|n| \geq |d| \cdot n \cdot d$ 。

0

- n MIN\_VALUE -1 MIN\_VALUE。
- d 0 ArithmeticException。

Java。  $q \cdot d \cdot n = n \cdot d \cdot q = n$ 。

0 INF NaN; 0。

0

C ++ Java。

a % b r (a / b) \* b + r a / \* + Java。 b。

ArithmeticException。

`a % b` 的符号与 `a` 和 `b` 的符号相同。 `a % b` 的符号与 `a` 的符号相同。

`a % b` 的符号与 `a` 和 `b` 的符号相同。 `a % b` 的符号与 `a` 的符号相同。

- `q`
- `a / b` 的符号与 `a` 和 `b` 的符号相同。
- `a` 除以 `b` 的余数。

`Infinity` 和 `NaN` 的符号与 `b` 的符号相同。

`Math`

`Math.IEEEremainder`

`int`

Java `int` 的符号与 `-231` 到 `231 - 1` 的符号相同。

`NaN` 的符号与 `NaN` 的符号相同。

- `NaN` 的符号与 `NaN` 的符号相同。
- `NaN` 的符号与 `NaN` 的符号相同。

`NaN` 的符号与 `NaN` 的符号相同。

`Infinity` 和 `NaN`

Java `float` 和 `double` 的符号与 `IEEE 754` 的符号相同。

- `Infinity` 和 `NaN` 的符号与 `NaN` 的符号相同。
- `NaN` 的符号与 `NaN` 的符号相同。

`Infinity` 和 `NaN` 的符号与 `NaN` 的符号相同。

`NaN` 的符号与 `NaN` 的符号相同。

`Infinity` 和 `NaN` 的符号与 `NaN` 的符号相同。

- `Infinity` 和 `NaN` 的符号与 `NaN` 的符号相同。
- `Infinity` 和 `NaN` 的符号与 `NaN` 的符号相同。
- `Infinity` 和 `NaN` 的符号与 `NaN` 的符号相同。
- `Infinity` 和 `NaN` 的符号与 `NaN` 的符号相同。
- `Infinity` 和 `NaN` 的符号与 `NaN` 的符号相同。

Java `String` 的符号与 `NaN` 的符号相同。

`String` 的符号与 `NaN` 的符号相同。

`String` 的符号与 `NaN` 的符号相同。

Java `String` 的符号与 `NaN` 的符号相同。

- `String` 的符号与 `NaN` 的符号相同。
- `Numeric` 的符号与 `NaN` 的符号相同。
- `Reference` 的符号与 `NaN` 的符号相同。

boolean 的 == 和 != 运算符。

### Numeric == 和 !=

byte, short, char, int, long, float, double 的 == 和 != 运算符。

注意：

1. 对于 byte, short, char, int, long, float, double 类型。
2. 对于 byte, short, char, int 类型。
3. 对于 float, double 类型，NaN 不等于 NaN。
4. 对于 IEEE 754 浮点格式。

boolean 的 == 和 != 运算符。

### boolean == 和 !=

boolean 的 == 和 != 运算符。

1. Boolean 的 == 和 != 运算符。
2. Boolean 的 == 和 != 运算符。

	A == B.	A = B.
boolean	true	true
byte	true	true
short	true	true
char	true	true
int	true	true
long	true	true
float	true	false
double	true	false

对于 Integer, Long, Short, Character 等包装类。

- Boolean 的 == 和 != 运算符。Boolean 的 == 和 != 运算符。
- Integer 的 == 和 != 运算符。Integer 的 == 和 != 运算符。

### Reference == 和 !=

String, Integer, Boolean 的 == 和 != 运算符。

```
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
```

对于 String, Integer, Boolean 等包装类。请参考 <http://www.riptutorial.com/java/example/16290/pitfall--using----to-compare-strings> 和 <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 NaN x != x true.

NaN "NaN". == NaN.

Java "IEEE 754". <http://stackoverflow.com/a/1573715/139985> ... "NaN"

++ / -

++ -- 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
```

. . .

```
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
```

;) )

{condition-to-evaluation} {statement-executed-on-true} {statement-executed-on-false}

<sup>1</sup> ? : if-else .



```

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 Tutorial ? : :) . "Java" . "C++" .

Java

Java 4

- ~ JLS 15.15.5 .
- AND & JLS 15.22.2 .
- OR | JLS 15.22.2 .
- XOR ^ JLS 15.22.2 .

Java

		AB	A	A ^ B
0	0	1	0	0
0	1	1	0	1
1	0	0	0	1
1	1	0	1	0

Java 32 64

Java

Java

Java

& Java

```

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

```

Java

```

int word2 = 0b01011111;

```





$E_1 \text{ op} = E_2$  的语义为  $E_1 = (T) ((E_1) \text{ op} (E_2))$  的语义为  $T[E_1] \text{ op} E_2$ 。

的语义为  $E_1 = (T) ((E_1) \text{ op} (E_2))$ 。

### 1. =

的语义为  $E_1 = (T) ((E_1) \text{ op} (E_2))$ 。

$c = a + b$  的语义为  $a + b$  的语义为  $c$ 。

### 2. +=

"add and assign" 的语义为  $c = c + a$ 。String 的语义为 " "。

$c += a$  的语义为  $c = c + a$ 。

### 3. -=

"subtract and assign" 的语义为  $c = c - a$ 。

$c -= a$  的语义为  $c = c - a$ 。

### 4. \*=

"multiply and assign" 的语义为  $c = c * a$ 。

$c *= a$  的语义为  $c = c * a$ 。

### 5. /=

"divide and assign" 的语义为  $c = c / a$ 。

$c /= a$  的语义为  $c = c / a$ 。

### 6. %=

"modulo and assign" 的语义为  $c = c \% a$ 。

$c \% = a$  的语义为  $c = c \% a$ 。

### 7. <<=

"left shift and assign" 的语义为  $c = c << n$ 。

$c << = 2$  的语义为  $c = c << 2$ 。

### 8. >>=

"right shift and assign" 的语义为  $c = c >> n$ 。

$c >> = 2$  的语义为  $c = c >> 2$ 。

### 9. >>>=

"right shift and assign" 的语义为  $c = c >>> n$ 。

$c >>> = 2$  的语义为  $c = c >>> 2$ 。

### 10. &=

"bitwise AND and assign" 的语义为  $c = c \& a$ 。

`c &= 2`  $c = c \& 2$

### 11. |=

“位或赋值”。

`c |= 2`  $c = c | 2$

### 12. ^=

“位异或赋值”。

`c ^= 2`  $c = c \wedge 2$

“逻辑与” | “逻辑或”

Java 的 `boolean` 类型。 逻辑与 | 逻辑或

- `&&` - 逻辑 AND
- `||` - 逻辑 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>`

```
{
    boolean L = evaluate(<left-expr>);
    if (!L) {
        return evaluate(<right-expr>);
    } else {
        // short-circuit the evaluation of the 2nd operand expression
        return true;
    }
}
```

“逻辑与” | “逻辑或”

“逻辑与” `&&`

“逻辑与” `&&`。 `Integer`

```
public boolean isZero(Integer value) {
    return value == 0;
}

public boolean isZero(Integer value) {
    return value != null && value == 0;
}
```

“逻辑与” `value != null` `NullPointerException`

value != null test value != null && value == 0。 null 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。

- <<>>。
- '>>'。
- '>>>'。

1.

- int long shift byte short char。
- byte char short int int。
- mod。
- Java ""。
- 2。
- 2。

。

OPERAND1		<<	>>	>>>
0b0000000000001011	0	0b0000000000001011	0b0000000000001011	0b0000000000001011
0b0000000000001011	1	0b00000000000010110	0b000000000000101	0b000000000000101
0b0000000000001011	2	0b000000000000101100	0b000000000000010	0b000000000000010
0b0000000000001011	28	0b1011000000000000	0b0000000000000000	0b0000000000000000
0b0000000000001011	31	0b1000000000000000	0b0000000000000000	0b0000000000000000
0b0000000000001011	32	0b0000000000001011	0b0000000000001011	0b0000000000001011
...	...	...	...	...

OPERAND1		<<	>>	>>>
0b1000000000001011	0	0b1000000000001011	0b1000000000001011	0b1000000000001011
0b1000000000001011	1	0b0000000000010110	0b110000000000101	0b010000000000101
0b1000000000001011	2	0b0000000000101100	0b111000000000010	0b0010000000000100
0b1000000000001011	31	0b1000000000000000	0b1111111111111111	0b0000000000000001

Bit

Lambda

Java 8 Lambda

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

Lambda

Operator

Operator: 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

Code Example

```
int i = 1;
long l = 2;
if (i < l) {
    System.out.println("i is smaller");
}
```

Code Example

```
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");
}
```

Code Example

- 1.
- 2.
- 3.

4. int long float double

- 
- 

Java

<https://riptutorial.com/zh-TW/java/topic/176>

170:

## Examples

addShutdownHook

```
Runtime.getRuntime().addShutdownHook(new Thread() -> {
    ImportantStuff.someImportantIOStream.close();
});
```

<https://riptutorial.com/zh-TW/java/topic/7304/>









```
}
```

尾递归的递归调用。

尾递归

尾递归的递归调用。

尾递归的递归调用。

尾递归的递归调用 - 尾递归的递归调用。

尾递归的递归调用。尾递归的递归调用。尾递归的递归调用。尾递归的递归调用。

```

public void tail(int n)                public void head(int n)
{
    if(n == 1)                          {
        return;                            if(n == 0)
    else                                  return;
        System.out.println(n);            else
    tail(n-1);                            head(n-1);
}                                          System.out.println(n);
}                                          }

```

尾递归的递归调用。尾递归的递归调用。尾递归的递归调用。尾递归的递归调用。

尾递归的递归调用。尾递归的递归调用。尾递归的递归调用。尾递归的递归调用。

尾递归的递归调用

### StackOverflowError

尾递归的递归调用。尾递归的递归调用。尾递归的递归调用。尾递归的递归调用。

尾递归的递归调用

```

public static void recursion(int depth) {
    if (depth > 0) {
        recursion(depth-1);
    }
}

```

尾递归的递归调用。尾递归的递归调用。尾递归的递归调用。尾递归的递归调用。



尾递归的递归调用。尾递归的递归调用。尾递归的递归调用。尾递归的递归调用。

尾递归的递归调用。尾递归的递归调用。尾递归的递归调用。尾递归的递归调用。

- 尾递归的递归调用
- 尾递归的递归调用
- 尾递归的递归调用
- 尾递归的递归调用

尾递归的递归调用











2. 实现 login 方法。
  - 实现 login 方法。
  - 实现 login 方法时，需要调用 logout 方法。
3. 实现 int 方法 passInt。
4. 实现 int 方法 half。

实现

实现

```
package callbackRemote;

import java.rmi.Remote;
import java.rmi.RemoteException;

public interface RemoteLogin extends Remote {

    RemoteConnection login(String name, RemoteClient client) throws RemoteException;
}
```

实现

```
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;
}
```

实现

```
package callbackRemote;

import java.rmi.Remote;
import java.rmi.RemoteException;

public interface RemoteClient extends Remote {

    void half(int i) throws RemoteException;
}
```

实现

实现

```
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□□□□
□Client1□□□□□□□□120
□Client2□□□□□□□□60
□Client1□30□□□□□□□□
□Client2□□□□□□□□15
□Client1□□□□□□□□7
□□1□□□"7"
Client2□□□□"15"
□□1□□□"30"
Client2□□□□"60"

```

□□□□□□□□□□□□**RMI**□□

□□□□□□□□**RMI**□□□□□□□□Java□□□□□□□□ □□□□□□□□。

□□□□

**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/zh-TW/java/topic/171/□□□□□□-rmi->

173:

Java Collection

### Examples

Java

[ <http://i.stack.imgur.com/aSDsG.png> ]

<https://riptutorial.com/zh-TW/java/topic/10846/>

174:

extends Interfaces implements

- classB ClassA {...}
- classB InterfaceA {...}
- InterfaceB InterfaceA {...}
- classB ClassA InterfaceC InterfaceD {...}
- AbstractClassB ClassA {...}
- AbstractClassB AbstractClassA {...}
- AbstractClassB 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
    }
}

```

编译并运行以下代码。

```

Component myButton = new Button();
Component myTextBox = new TextBox();

myButton.render(); //renders a button
myTextBox.render(); //renders a text box

```

编译并运行

以下代码以编译并运行。

编译并运行

- 编译并运行以下代码。
- 编译并运行以下代码 static 编译并运行 static。

Java SE 8

编译并运行以下代码。

Java SE 8

Java 8 编译并运行 编译并运行 编译并运行。

编译并运行

编译并运行 java 编译并运行。编译并运行

```

Component myAnonymousComponent = new Component() {
    @Override
    public void render() {
        // render a quick 1-time use component
    }
}

```

编译并运行

编译并运行以下代码 " " 编译并运行。编译并运行

```

public class BaseClass {

    public static int num = 5;

    public static void sayHello() {
        System.out.println("Hello");
    }

    public static void main(String[] args) {
        BaseClass.sayHello();
    }
}

```







```

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

```

Java

Object

Java

```

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");
    }
}

```

编译并运行。编译并运行 staticMethod() 方法。编译并运行 Parent 的 staticMethod() 方法。编译并运行 Child 的 staticMethod() 方法。

编译

编译 SHADOWED 和 OVERRIDDEN。编译并运行。编译并运行。

```

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
    }
}

```

编译

编译并运行 b = (B) a; 编译并运行。

编译并运行 A a = b; 编译并运行。

编译

```

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(); Car Vehicle.

Car c = vehicle; vehicle Vehicle Car Truck, Motorcycle, or any other current or future subclass of Vehicle, or any other current or future subclass of. (Or indeed, an instance of Vehicle itself, since we did not declare it as an. (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 car referring to a Truck`.

```
Car c = (Car) vehicle;
```

vehicle Car Car. ClassCastException.

```
String s = (String) vehicle; // not valid
```

Java Vehicle String. JLS.

Java.

```

public <T> Set<T> toSet(Collection<T> collection) {
    return Sets.newHashSet(collection);
}

```

```

public <T> HashSet<T> toSet(ArrayList<T> collection) {
    return Sets.newHashSet(collection);
}

```

- ;
- ;

- `CompletableFuture<RS> invoke(RQ request, Class<RS> responseClass);`
- `CompletableFuture<RS> invoke(RQ request, Class<RS> responseClass);`

CompletableFuture<RS> invoke(RQ request, Class<RS> responseClass);

- `CompletableFuture<RS> invoke(RQ request, Class<RS> responseClass);`
- `CompletableFuture<RS> invoke(RQ request, Class<RS> responseClass);`
- `CompletableFuture<RS> invoke(RQ request, Class<RS> responseClass);`

CompletableFuture<RS> invoke(RQ request, Class<RS> responseClass);

```
public interface RemoteInvoker {  
    <RQ, RS> CompletableFuture<RS> invoke(RQ request, Class<RS> responseClass);  
}
```

CompletableFuture<RS> invoke(RQ request, Class<RS> responseClass);

```
public interface AmqpInvoker extends RemoteInvoker {  
    static AmqpInvokerBuilder with(String instanceId, ConnectionFactory factory) {  
        return new AmqpInvokerBuilder(instanceId, factory);  
    }  
}
```

CompletableFuture<RS> invoke(RQ request, Class<RS> responseClass);

```
RemoteInvoker invoker = AmqpInvoker.with(instanceId, factory)  
    .requestRouter(router)  
    .build();
```

CompletableFuture<RS> invoke(RQ request, Class<RS> responseClass);

```
Response res = invoker.invoke(new Request(data), Response.class).get();
```

Java 8 CompletableFuture<RS> invoke(RQ request, Class<RS> responseClass);

```
public interface AmqpInvoker extends RemoteInvoker {  
    class Factory {  
        public static AmqpInvokerBuilder with(String instanceId, ConnectionFactory factory) {  
            return new AmqpInvokerBuilder(instanceId, factory);  
        }  
    }  
}
```

CompletableFuture<RS> invoke(RQ request, Class<RS> responseClass);

```
RemoteInvoker invoker = AmqpInvoker.Factory.with(instanceId, factory)  
    .requestRouter(router)  
    .build();
```

CompletableFuture<RS> invoke(RQ request, Class<RS> responseClass);

```
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();
}

```



7. Animal

TL; DR

Java

.....

- 1.
2. protected private
- 3.

.....

1. Serializable
- 2.
- 3.

ClassB ClassA

```

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/zh-TW/java/topic/87>

## 175: Deques

### Examples

#### PriorityQueue

PriorityQueue implements SortedSet. PriorityQueue implements Comparable. PriorityQueue implements Comparable with 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: []
```

#### LinkedList FIFO

java.util.LinkedList implements java.util.List java.util.Queue implements FIFO.

LinkedList offers offer() method. enqueue() method. while() method. FIFO Queue. dequeue() method.

```
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() );
}
```

Output

```
first element
second element
third element
fourth element
fifth element
```

Stacks are implemented using arrays.

Stacks

Stacks

Stacks are implemented using arrays.

Stack API

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");
        }
    }
}
```

000

```

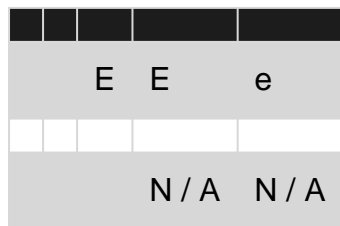
stack: []
10 was pushed to the stack
stack: [10]
15 was pushed to the stack
stack: [10, 15]
80 was pushed to the stack
stack: [10, 15, 80]
80 was popped from the stack
stack: [10, 15]
15 was popped from the stack
stack: [10]
10 was popped from the stack
stack: []
empty stack

```

### BlockingQueue

BlockingQueue 是一個阻塞的資料結構。當隊列為空時，put 方法會阻塞，直到隊列中有元素為止。

BlockingQueue 的构造函数可以指定初始容量，默認為 null，false。



BlockingQueue 是一個阻塞的資料結構。當隊列為空時，put 方法會阻塞，直到隊列中有元素為止。

```
BlockingQueue<String> bQueue = new ArrayBlockingQueue<String>(2);
```

當隊列為空時，put 方法會阻塞。

當隊列中的元素數量達到 Integer.MAX\_VALUE 時，put 方法會阻塞。

BlockingQueue 的實現包括：

1. ArrayBlockingQueue
2. LinkedBlockingQueue
3. PriorityBlockingQueue

以下代碼展示了 ArrayBlockingQueue 的使用：

```

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");

```

0000

```
Entry one done
Entry two done
```

Queue 的 add 方法。

Queue

Queue

Queue 的 offer 方法。 offer 方法返回 true 表示成功，返回 false 表示失败。

Queue 的 remove 方法。 remove 方法返回 Queue 中的元素，如果没有元素，则返回 null。

Queue

```
public interface Queue<E> extends Collection<E> {
    boolean add(E e);

    boolean offer(E e);

    E remove();

    E poll();

    E element();

    E peek();
}
```

Queue 的 add 方法。

- add 方法返回 true 表示成功，返回 false 表示失败。
- add 方法返回 false 表示失败，其他方法返回 null 表示 false。

方法	返回	说明
add(e)	boolean	add(e)
remove()	E	remove()
offer(e)	boolean	offer(e)
poll()	E	poll()
element()	E	element()
peek()	E	peek()

Queue

Deque 是 Queue 的子接口，它提供了双端队列的功能。Deque 的 add 方法。

Deque 的 add 方法。 add 方法返回 true 表示成功，返回 false 表示失败。 add 方法返回 false 表示失败，其他方法返回 null 表示 false。

方法	返回	说明
getFirst()	E	getFirst()
getLast()	E	getLast()
addFirst(E e)	boolean	addFirst(E e)



	;
addLast (E e)	
removeFirst ()	
removeLast ()	

offer()、poll()、peek()、peekFirst()、peekLast()、peekFirstElement()、peekLastElement()。

### Deque

Deque 是 List 的子接口，它提供了在列表的两端添加和删除元素的方法。

```
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
```

Deque 提供了 element()、elementAt(int index)、getFirst()、getLast()、getFirstElement()、getLastElement() 等方法。

```
String firstElement0 = dequeA.element();
String firstElement1 = dequeA.getFirst();
String lastElement = dequeA.getLast();
```

### Deque 操作

remove()、removeFirst()、removeLast()、removeFirstElement()、removeLastElement()。

```
String firstElement = dequeA.remove();
String firstElement = dequeA.removeFirst();
String lastElement = dequeA.removeLast();
```

Deque 的更多用法请参考 <https://riptutorial.com/zh-TW/java/topic/7196/deques>

**176:**

javadoc

### Examples

utils Java 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 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);
```

nextInt(int bound)  $2^{30} + 1$

$n = 2^{30} + 1$

nextInt(max int)

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 2^160 Random 2^48 Mersenne Twister Xorshift

- SecureRandom SUN sun.security.provider.SecureRandom
- Unix /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;
                    }
                }
            }
        }
    }
}

```



```
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/zh-TW/java/topic/890>

177:

## Examples

HashSet

HashSet

```
Set<String> h = new HashSet<String>() {{
    add("a");
    add("b");
}};
```

HashSet

```
Set<String> h = new HashSet<String>(Arrays.asList("a", "b"));
```

HashSet

```
Sets.newHashSet("a", "b", "c")
```

Streams

```
Set<String> set3 = Stream.of("a", "b", "c").collect(toSet());
```

HashSet

HashSet。 equals() hashCode()。

HashSet。

## 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

Set 是 Collection 的子接口。

Set 包含 HashSet、TreeSet 和 LinkedHashSet。

Set

### HashSet

```
Set<T> set = new HashSet<T>();
```

HashSet 是 Set 的子接口。HashSet 使用哈希表来存储元素。

### TreeSet

TreeSet 是 Set 的子接口。TreeSet 使用红黑树来存储元素。

```
TreeSet<T> sortedSet = new TreeSet<T>();
```

### LinkedHashSet

LinkedHashSet 是 Set 的子接口。LinkedHashSet 使用哈希表来存储元素，并且保持元素的插入顺序。

```
LinkedHashSet<T> linkedhashset = new LinkedHashSet<T>();
```

Set

Set

Set 是 Collection 的子接口。

Set

## 1. HashSet 与 HashMap

2. **Linked HashSet** HashSet Set

3. **TreeSet** TreeMap NavigableSet

Set

```
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()

add()

```
set.add(12); // - Adds element 12 to the set  
set.add(13); // - Adds element 13 to the set
```

Set

```
set = [12,13]
```

Set

```
set.clear(); //Removes all objects from the collection.
```

Set

```
set = []
```

Set

contains()

```
set.contains(0); //Returns true if a specified object is an element within the set.
```

False

Set

isEmpty() Set

```
set.isEmpty(); //Returns true if the set has no elements
```

Set

Set

```
set.remove(0); // Removes first occurrence of a specified object from the collection
```

Set

```
set.size(); //Returns the number of elements in the collection
```

0

Set



#####

```
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 Stream API

```
List<String> list = set.stream().collect(Collectors.toList());
```

##Set#####

#####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/zh-TW/java/topic/3102/>

178:

java.util

Collection<O> List<O> Set<O> Map<K, V> .

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 SortedSet<E> Set<E>

AbstractList

Java SE 1.2

1.2

- Vector ArrayList
- Dictionary Map Dictionary
- Hashtable HashMap

Examples

ArrayList

ArrayList List

```
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 String "Melon" 1 String "Strawberry".

addAll(Collection<? extends E> c)

```
List<String> aListOfFruitsAndVeggies = new ArrayList<String>();
aListOfFruitsAndVeggies.add("Onion");
aListOfFruitsAndVeggies.addAll(aListOfFruits);
```

"Onion" aListOfFruitsAndVeggies aListOfFruitsAndVeggies "Melon" 1 "Strawberry" 2.

### Java Collections

List java.util.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");
```



## Java Collections

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

## 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

Streams

```
List<String> newList = Stream.concat(listOne.stream(),
listTwo.stream()).collect(Collectors.toList());
```

#####

##### **List** #####

#####

#####

```
List<String> fruits = new ArrayList<String>();
fruits.add("Apple");
fruits.add("Banana");
fruits.add("Strawberry");
```



**for** *Skips "Banana"*

Apple Strawberry. Banana Apple 0 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() 抛出 UnsupportedOperationException。

UnsupportedOperationException - remove

Collections.unmodifiable...()

Iterator Iterator List modCount ConcurrentModificationException。

modCount int Collection add() remove() Iterator。 Iterator modCount List modCount Iterator。 ConcurrentModificationException。

```

Iterator<String> fruitIterator = fruits.iterator();
fruits.set(0, "Watermelon");
while(fruitIterator.hasNext()){
    System.out.println(fruitIterator.next());
}

```

Iterator List ConcurrentModificationException

```

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

ArrayList

```
for (int i = 0; i < fruits.size(); i++) {
    System.out.println (fruits.get(i));
    if ("Apple".equals(fruits.get(i))) {
        fruits.remove(i);
        i--;
    }
}
```

ArrayList. remove(i) method. remove(i) method. remove(i) method.

ArrayList

```
ArrayList shouldBeRemoved = new ArrayList();
for (String str : currentArrayList) {
    if (condition) {
        shouldBeRemoved.add(str);
    }
}
currentArrayList.removeAll(shouldBeRemoved);
```

ArrayList

Java SE 8

Java 8 ArrayList

ArrayList

ArrayList

```
List<String> filteredList =
    fruits.stream().filter(p -> !"Apple".equals(p)).collect(Collectors.toList());
```

ArrayList

removeIf

ArrayList

```
fruits.removeIf(p -> "Apple".equals(p));
```

ArrayList

ArrayList

ArrayList

ArrayList

```
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);
    }
}
```

□□□□□□□□□□□□□□□□□□

```
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 traversal
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

```
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

```
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());
}
```



```

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));

```

removeNameList nameList

```

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
    }
}

```

removeNameList removeNames(nameList, removeNameList);

```

James Smith Sonny Huckle Berry Finn Allan
James Smith Finn Allan

```

Iterable Iterator for-each

for-each

1. Iterable iterator()
2. hasNext() next() remove() 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) {

```



```
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。

## subMap fromKey toKey

fromKey toKey。

```
Map map = new TreeMap();
Map map1 = map.get(fromKey,toKey);
```

fromKey toKey fromKey toKey IllegalArgumentException。

。

<https://riptutorial.com/zh-TW/java/topic/90>

179:

### Examples

Java final

- 
- 
- 

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 final

Java SE 8

### final static

final foo static static

```
class TestFinal {
    private final static List foo;

    public Test() {
        foo = new ArrayList();
    }
}
```

final 关键字。TestFinal 类。TestFinal 类。foo 列表。

```
class TestFinal {  
    private static final List foo = new ArrayList();  
    //..  
}
```

final 关键字。

```
class TestFinal {  
    private static final List foo;  
    static {  
        foo = new ArrayList();  
    }  
    //..  
}
```

final 关键字。final 关键字。

Java 类 Integer 和 Long。Integer 和 Integer 类。final 关键字。

volatile 关键字。

volatile 关键字。volatile 关键字。volatile long double 数据类型。

```
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;  
    }  
}
```

volatile 关键字。

static 关键字。

- static 关键字。
- static 关键字。
- static 关键字。

```
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();

```





180:

applet API AudioClip javax.sound.sampled Clip。ClipAudioClip。

### 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

javax.sound.midiMIDI。SequencerMIDI。

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();
        }
    }
}
```



```

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
}

```

XX. XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX.

XXXXXX

Hello AudioXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXJavaXXXX. XXXXXXXX.wavXXXXXXXXXXXXmp3XXXXXXXX.

```

import java.io.*;
import java.net.URL;
import javax.sound.sampled.*;

public class AudioClipTest {

    // Constructor
    public AudioClipTest() {
        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 AudioClipTest();
    }
}

```

XXXXXXXX <https://riptutorial.com/zh-TW/java/topic/160>

181: - Java

java.lang.Class

java.lang java.lang.reflect java

Reflection API

IDE 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);
    }
}
```

181: - Java <https://riptutorial.com/zh-TW/java/topic/10151> - java

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");
```

findClass

findClass

- throw ClassNotFoundException
- NoClassDefFoundError

ClassLoader

java.lang.ClassLoader

- findClass(String) -
- loadClass(String, boolean) -
- findResource findResources -

defineClass final

```
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
3. findClass() findClass()
4. this.classname() defineClass() this.classfile() Class
5. ClassNotFoundException

.class

ClassLoader Oracle ClassLoader defineClass() ClassLoader

ByteClassLoader ClassLoader ClassLoader ClassLoader() defineClass() defineClass() private name classBytes classBytes[0] classBytes.length

```
public class ByteClassLoader extends ClassLoader {
    public Class<?> defineClass(String name, byte[] classBytes) {
        return defineClass(name, classBytes, 0, classBytes.length);
    }
}
```

defineClass() stackoverflow MyClass

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 catch

```
try{
    loader.loadClass("stackoverflow.MyClass");
} catch(ClassNotFoundException e){
    e.printStackTrace();
}
```

https://riptutorial.com/zh-TW/java/topic/5443

183: 0000

00

0000000000。 0000000000 - 000000000000 - 000000000000。 0000000000。

0 - 0000000000000000000000/000000/00。

00

- `class Example {} // class keyword name body`

### Examples

00000000

```
class TrivialClass {}
```

000000class000000000000000000000000。

00new000000000000。

```
TrivialClass tc = new TrivialClass();
```

0000000000

00000000

```
class ObjectMemberVsStaticMember {
    static int staticCounter = 0;
    int memberCounter = 0;

    void increment() {
        staticCounter++;
        memberCounter++;
    }
}
```

000000

```
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();
```



XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

```
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;
}
}

```

onCreate() 方法中，我们可以在 AndroidManifest.xml 文件中注册 Car 类。

## 1

```

Car newCar = new Car(30, 10, "Ferrari", "Red");

```

我们可以在 onCreate() 方法中，调用 Car 类的 repaintCar() 方法。

```

newCar.repaintCar("Blue");

```

我们可以在 onCreate() 方法中，调用 Car 类的 repaintCar() 方法。

我们可以在 onCreate() 方法中，调用 Car 类的 getName() 方法。

```

String myCarName = newCar.getName(); //returns string "Ferrari"

```

我们可以在 onCreate() 方法中，调用 Car 类的 getName() 方法。

## 2

```

Car newCar = new Car();

```

我们可以在 onCreate() 方法中，调用 Car 类的 Car() 方法。

```

public void Car(){
    milesPerGallon = 0;
    name = "";
    color = "";
    numGallonsInTank = 0;
}

```

我们可以在 onCreate() 方法中，调用 Car 类的 Car() 方法。

我们可以在 onCreate() 方法中，调用 Car 类的 Car() 方法。

```

Car myCar = new Car();
String color = Car.getColor(); //returns empty string

```

我们可以在 onCreate() 方法中，调用 Car 类的 Car() 方法。

```

Car myCar = new Car();
myCar.nameCar("Ferrari");
myCar.paintCar("Purple");

```









```

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());

}
}

```

编译并运行以下代码。

编译并运行以下代码 -

编译	编译
编译并运行以下代码。	编译并运行以下代码。
编译并运行以下代码。	编译并运行以下代码 IS-A 编译并运行以下代码。
编译并运行以下代码。	编译并运行以下代码。
编译并运行以下代码。	编译并运行以下代码。
编译并运行以下代码。	编译并运行以下代码。
编译并运行以下代码。	编译并运行以下代码。

编译并运行以下代码 <https://riptutorial.com/zh-TW/java/topic/114>

184:

- `TargetType target = SourceType source;`

### Examples

`boolean`

`Unicode` `char` `char` `byte` `ASCII` `Character` `int` `short` `Unicode`

```
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`

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/zh-TW/java/topic/1392/>

## 185: 默认方法

简介

Java 8 引入了默认方法。默认方法是在接口中定义的，但可以在实现类中重写。

简介

- `public default void methodName() { /* method body */ }`

简介

默认方法

- 默认方法是在接口中定义的，但可以在实现类中重写。
- 默认方法是在接口中定义的，但可以在实现类中重写。
- 默认方法是在接口中定义的，但可以在实现类中重写。
- 默认方法是在接口中定义的，但可以在实现类中重写。
- 默认方法是在接口中定义的，但可以在实现类中重写。

默认方法

- 默认方法是在接口中定义的，但可以在实现类中重写。
- 默认方法是在接口中定义的，但可以在实现类中重写。
- 默认方法是在接口中定义的，但可以在实现类中重写。

默认方法

-	SUPER_CLASS-INSTANCE	SUPER_CLASS
SUB_CLASS-INSTANCE	是	是-否
SUB_CLASS	是-否	否

默认方法

-	接口-DEFAULT	实现类
IMPL_CLASS-INSTANCE	是	否
IMPL_CLASS	是-否	否

默认方法

- <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;
}
}
```

编译并运行 3

```
System.out.println(new Sum().calculateSum());
```

编译并运行

```
public interface Summable {
    static int getA() {
        return 1;
    }

    static int getB() {
        return 2;
    }

    default int calculateSum() {
        return getA() + getB();
    }
}

public class Sum implements Summable {}
```

编译并运行 3

```
System.out.println(new Sum().calculateSum());
```

编译并运行

编译并运行 super.foo()。编译并运行 super。编译并运行 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
    }
}
```

编译并运行

编译并运行。

编译并运行 20 编译并运行 Swim。

```
public interface Swim {
    void 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");
    }
}

```

编译

```
new FooSwimmer().backStroke();
```

输出

```
FooSwimmer.backStroke
```

编译选项

编译选项

```

public interface A {
    default void foo() { System.out.println("A.foo"); }
}

public interface B {
    default void foo() { System.out.println("B.foo"); }
}

```

编译选项 default foo。

编译选项 extend 编译选项 Java 编译选项。

编译选项 abstract 编译选项 foo 编译选项 A B。

```

public interface ABExtendsAbstract extends A, B {
    @Override
    void foo();
}

```

编译选项 class implement ABExtendsAbstract 编译选项 foo

```

public class ABExtendsAbstractImpl implements ABExtendsAbstract {
    @Override
    public void foo() { System.out.println("ABImpl.foo"); }
}

```

编译选项 编译选项 default。编译选项 编译选项 A B foo。

```

public interface ABExtends extends A, B {
    @Override

```



```
default void foo() { System.out.println("ABExtends.foo"); }  
}
```

class implement ABExtends not foo

```
public class ABExtendsImpl implements ABExtends {}
```

<https://riptutorial.com/zh-TW/java/topic/113>



S. No		Contributors
1	Java	<a href="#">aa_oo</a> , <a href="#">Aaqib Akhtar</a> , <a href="#">abhinav</a> , <a href="#">Abhishek Jain</a> , <a href="#">Abob</a> , <a href="#">acdcjunior</a> , <a href="#">Adeel Ansari</a> , <a href="#">adsalpha</a> , <a href="#">AER</a> , <a href="#">akhilsk</a> , <a href="#">Akshit Soota</a> , <a href="#">Alex A</a> , <a href="#">alphaloop</a> , <a href="#">altomnr</a> , <a href="#">Amani Kilumanga</a> , <a href="#">AndroidMechanic</a> , <a href="#">Ani Menon</a> , <a href="#">ankit dessor</a> , <a href="#">Ankur Anand</a> , <a href="#">antonio</a> , <a href="#">Arkadiy</a> , <a href="#">Ashish Ahuja</a> , <a href="#">Ben Page</a> , <a href="#">Blachshma</a> , <a href="#">bpoiss</a> , <a href="#">Burkhard</a> , <a href="#">Carlton</a> , <a href="#">Charlie H</a> , <a href="#">Coffeehouse Coder</a> , <a href="#">cldsEED</a> , <a href="#">Community</a> , <a href="#">Configure</a> , <a href="#">CraftedCart</a> , <a href="#">dabansal</a> , <a href="#">Daksh Gupta</a> , <a href="#">Dan Hulme</a> , <a href="#">Dan Morenus</a> , <a href="#">DarkVl</a> , <a href="#">David G.</a> , <a href="#">David Grinberg</a> , <a href="#">David Newcomb</a> , <a href="#">DeepCoder</a> , <a href="#">Do Nhu Vy</a> , <a href="#">Draken</a> , <a href="#">Durgpal Singh</a> , <a href="#">Dushko Jovanovski</a> , <a href="#">E_net4</a> , <a href="#">Edvin Tenovimas</a> , <a href="#">Emil Sierżęga</a> , <a href="#">Emre Bolat</a> , <a href="#">enrico.bacis</a> , <a href="#">Eran</a> , <a href="#">explv</a> , <a href="#">fgb</a> , <a href="#">Francesco Menzani</a> , <a href="#">Functino</a> , <a href="#">garg10may</a> , <a href="#">Gautam Jose</a> , <a href="#">GingerHead</a> , <a href="#">Grzegorz Górkiewicz</a> , <a href="#">iliketocode</a> , <a href="#">иәБoт әizuәә</a> , <a href="#">intboolstring</a> , <a href="#">ipsi</a> , <a href="#">J F</a> , <a href="#">James Taylor</a> , <a href="#">Jason</a> , <a href="#">JavaHopper</a> , <a href="#">Javant</a> , <a href="#">javydreamercsw</a> , <a href="#">Jean Vitor</a> , <a href="#">Jean-François Savard</a> , <a href="#">Jeffrey Brett Coleman</a> , <a href="#">Jeffrey Lin</a> , <a href="#">Jens Schauder</a> , <a href="#">John Fergus</a> , <a href="#">John Riddick</a> , <a href="#">John Slegers</a> , <a href="#">Jojodmo</a> , <a href="#">JonasCz</a> , <a href="#">Jonathan</a> , <a href="#">Jonny Henly</a> , <a href="#">Jorn Vernee</a> , <a href="#">kaartic</a> , <a href="#">Lambda Ninja</a> , <a href="#">LostAvatar</a> , <a href="#">madx</a> , <a href="#">Magisch</a> , <a href="#">Makoto</a> , <a href="#">manetsus</a> , <a href="#">Marc</a> , <a href="#">Mark Adelsberger</a> , <a href="#">Maroun Maroun</a> , <a href="#">Matt</a> , <a href="#">Matt</a> , <a href="#">mayojava</a> , <a href="#">Mitch Talmadge</a> , <a href="#">mnoronha</a> , <a href="#">Mrunal Pagnis</a> , <a href="#">Mukund B</a> , <a href="#">Mureinik</a> , <a href="#">NageN</a> , <a href="#">Nathan Arthur</a> , <a href="#">nevster</a> , <a href="#">Nithanim</a> , <a href="#">Nuri Tasdemir</a> , <a href="#">nyarasha</a> , <a href="#">ochi</a> , <a href="#">OldMcDonald</a> , <a href="#">Onur</a> , <a href="#">Ortomala Lokni</a> , <a href="#">OverCoder</a> , <a href="#">P.J.Meisch</a> , <a href="#">Pavneet_Singh</a> , <a href="#">Petter Friberg</a> , <a href="#">philnate</a> , <a href="#">Phrancis</a> , <a href="#">Pops</a> , <a href="#">ppeterka</a> , <a href="#">Přemysl Šťastný</a> , <a href="#">Pritam Banerjee</a> , <a href="#">Radek Postołowicz</a> , <a href="#">Radouane ROUFID</a> , <a href="#">Rafael Mello</a> , <a href="#">Rakitić</a> , <a href="#">Ram</a> , <a href="#">RamenChef</a> , <a href="#">rekire</a> , <a href="#">René Link</a> , <a href="#">Reut Sharabani</a> , <a href="#">Richard Hamilton</a> , <a href="#">Ronnie Wang</a> , <a href="#">ronnyfm</a> , <a href="#">Ross Drew</a> , <a href="#">RotemDev</a> , <a href="#">Ryan Hilbert</a> , <a href="#">SachinSarawgi</a> , <a href="#">Sanandrea</a> , <a href="#">Sandeep Chatterjee</a> , <a href="#">Sayakiss</a> , <a href="#">ShivBuyya</a> , <a href="#">Shoe</a> , <a href="#">Siguza</a> , <a href="#">solidcell</a> , <a href="#">stackptr</a> , <a href="#">Stephen C</a> , <a href="#">Stephen Leppik</a> , <a href="#">sudo</a> , <a href="#">Sumurai8</a> , <a href="#">Shadowfa</a> , <a href="#">tbodt</a> , <a href="#">The Coder</a> , <a href="#">ThePhantomGamer</a> , <a href="#">Thisaru Guruge</a> , <a href="#">Thomas Gerot</a> , <a href="#">ThomasThiebaud</a> , <a href="#">ThunderStruct</a> , <a href="#">tonirush</a> , <a href="#">Tushar Mudgal</a> , <a href="#">Unihedron</a> , <a href="#">user1133275</a> , <a href="#">user124993</a> , <a href="#">uzaif</a> , <a href="#">Vaibhav Jain</a> , <a href="#">Vakerrian</a> , <a href="#">vasili111</a> , <a href="#">Victor Stafusa</a> , <a href="#">Vin</a> , <a href="#">VinayVeluri</a> , <a href="#">Vogel612</a> , <a href="#">vorburger</a> , <a href="#">Wilson</a> , <a href="#">worker_bee</a> , <a href="#">Yash Jain</a> , <a href="#">Yury Fedorov</a> , <a href="#">Zachary David Saunders</a> , <a href="#">Ze Rubeus</a>
2	Apache Commons Lang	<a href="#">Jonathan Barbero</a>
3	AppDynamics TIBCO BusinessWorks Instrumentation	<a href="#">Alexandre Grimaud</a>
4	BigDecimal	<a href="#">alain.janinm</a> , <a href="#">Christian</a> , <a href="#">Dth</a> , <a href="#">Enigo</a> , <a href="#">ggolding</a> , <a href="#">Harish Gyanani</a> , <a href="#">John Nash</a> , <a href="#">Loris Securo</a> , <a href="#">Łukasz Piaszczyk</a> , <a href="#">Manish Kothari</a> , <a href="#">mszymborski</a> , <a href="#">RamenChef</a> , <a href="#">sudo</a> , <a href="#">xwoker</a>
5	C ++	<a href="#">John DiFinì</a>
6	Classpath	<a href="#">Aaron Digulla</a> , <a href="#">GPI</a> , <a href="#">K'</a> , <a href="#">Kenster</a> , <a href="#">Ruslan Ulanov</a> , <a href="#">Stephen C</a> , <a href="#">trashgod</a>
7	CompletableFuture	<a href="#">Adowrath</a> , <a href="#">Kishore Tulsiani</a> , <a href="#">WillShackleford</a>
8	EnumSet	<a href="#">KIRAN KUMAR MATAM</a>
9	Executor ExecutorService	<a href="#">Andrii Abramov</a> , <a href="#">Cache Staheli</a> , <a href="#">Fildor</a> , <a href="#">hd84335</a> , <a href="#">Jens Schauder</a> , <a href="#">JonasCz</a> , <a href="#">noscreenname</a> , <a href="#">Olivier Grégoire</a> , <a href="#">philnate</a> , <a href="#">Ravindra babu</a>

	Thread	, Shettyh, Stephen C, Suminda Sirinath S. Dharmasena, sv3k, tones, user1121883, Vlad-HC, Vogel612
10	FileUploadAWS	Amit Gujarathi
11	FTP	Kelvin Kellner
12	URLConnection	Community, Datagrammar, EJP, Inzizam Tariq IT, JonasCz, kiedysktos, Mureinik, NageN, Stephen C, still_learning
13	InputStreams OutputStreams	akgren_soar, EJP, Gubbel, J Atkin, Jens Schauder, John Nash, Kip, KIRAN KUMAR MATAM, Matt Clark, Michael, RamenChef, Stephen C, Vogel612
14	Java Native Access	Ezekiel Baniaga, Stephan, Stephen C
15	Java Native Interface	Coffee Ninja, Fjoni Yzeiri, Jorn Vernee, RamenChef, Stephen C, user1803551
16	Java SE 7	compuhosny, RamenChef
17	Java SE 8	compuhosny, RamenChef, sun-solar-arrow
18	java.util.Objects	mnoronha, RamenChef, Stephen C
19	JavaBean	foxt7ot, J. Pichardo, James Fry, SaWo, Stephen C
20	Java2D	17slim, ABDUL KHALIQ
21	JavaJSON	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
22	Java	Display Name, mnoronha
23	Java	Shree, Stephen C, Suminda Sirinath S. Dharmasena
24	Java	Daniel M., engineerocoding, fgb, John Nash, jwd630, mnoronha, OverCoder, padippist, RamenChef, Squidward, Stephen C
25	Java - 'java' 'javaw'	4444, Ben, mnoronha, Stephen C, Vogel612
26	Java	Nikhil R
27	Java	Gene Marin, jatanp, Stephen C, Vogel612
28	Java	Danilo Guimaraes, Leonardo Pina
29	Java	Alexiy
30	Java	Dariusz, hd84335, HTNW, Ilya, Mr. P, Petter Friberg, ravthiru, Stephen C, Stephen Leppik, Vogel612
31	Java	Gal Dreiman, screab, Stephen C
32	Java - 'javac'	CraftedCart, Jatin Balodhi, Mark Stewart, nishizawa23, Stephen C, Snaoшfa, Tom Gijselinck

33	Java虚拟机JVM	Dushman, RamenChef, Rory McCrossan, Stephen Leppik
34	Java	garg10may, nishizawa23, Pseudonym Patel, RamenChef, Smit, Stephen C
35	Java - Nulls NullPointerException	17slim, Andrii Abramov, Daniel Nugent, dorukayhan, fabian, François Cassin, Miles, Stephen C, Zircon
36	Java -	Dorian, GPI, John Starich, Jorn Vernee, Michał Rybak, mnoronha, ppeterka, Sharon Rozinsky, steffen, Stephen C, xTrollxDudex
37	Java -	Bhoomika, bruno, dimo414, Gal Dreiman, hd84335, SachinSarawgi, scorpp, Stephen C, Stephen Leppik, user3105453
38	Java -	dorukayhan, james large, Stephen C
39	Java -	Alex T., Cody Gray, Enwired, Friederike, Gal Dreiman, hd84335, Hiren, Peter Rader, piyush_baderia, RamenChef, Ravindra HV, RudolphEst, Stephen C, Todd Sewell, user3105453
40	JAXB	Dariusz, Drunix, fabian, hd84335, Jabir, ppeterka, Ram, Stephan, Thomas Fritsch, vallismortis, Walery Strauch
41	JAX-WS	ext1812, Jonathan Barbero, Stephen Leppik
42	JMX	esin88
43	JNDI	EJP, neohope, RamenChef
44	JShell	ostrichofevil, Sudip Bhandari
45	Just in Time JIT	Liju Thomas, Stephen C
46	JVM	desilijic
47	JVM	Configure, RamenChef
48	Lambda	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 Vandavelde, John Slegers, JonasCz, Jorn Vernee, Jude Niroshan, JudgingNotJudging, Kevin Raoofi, Malt, Mark Green, Matt, Matthew Trout, Matthias Braun, ncmathsadist, nobeh, Ortomala Lokni, Paulo 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
49	LinkedHashMap	Amit Gujarathi, KIRAN KUMAR MATAM
50	log4j / log4j2	Daniel Wild, Fildor, HCarrasko, hd84335, Mrunal Pagnis, Rens van der Heijden
51	Nashorn JavaScript	ben75, ekaerovets, Francesco Menzani, hd84335, Ilya, InitializeSahib, kasperjj, VatsalSura

52	NIO - <code>ByteBuffer</code>	<a href="#">Matthieu</a> , <a href="#">mnoronha</a>
53	Oracle <code>ByteBuffer</code>	<a href="#">Ahmed Ashour</a> , <a href="#">aioobe</a> , <a href="#">akhilsk</a> , <a href="#">alex s</a> , <a href="#">Andrii Abramov</a> , <a href="#">Cassio Mazzochi Molin</a> , <a href="#">Dan Whitehouse</a> , <a href="#">Enigo</a> , <a href="#">erickson</a> , <a href="#">f_puras</a> , <a href="#">fabian</a> , <a href="#">giucal</a> , <a href="#">hd84335</a> , <a href="#">J.D. Sandifer</a> , <a href="#">Lahiru Ashan</a> , <a href="#">Mac70</a> , <a href="#">NamshubWriter</a> , <a href="#">Nicktar</a> , <a href="#">Petter Friberg</a> , <a href="#">Pradatta</a> , <a href="#">Pritam Banerjee</a> , <a href="#">RamenChef</a> , <a href="#">sanjaykumar81</a> , <a href="#">Santa Claus</a> , <a href="#">Santhosh Ramanan</a> , <a href="#">VGR</a>
54	RSA <code>KeyPairGenerator</code>	<a href="#">Dennis Kriechel</a> , <a href="#">Drunix</a> , <a href="#">iqbal_cs</a> , <a href="#">Maarten Bodewes</a> , <a href="#">Nicktar</a> , <a href="#">Shog9</a>
55	Stack-Walking API	<a href="#">manouti</a>
56	<code>StringBuilder</code>	<a href="#">Andrii Abramov</a> , <a href="#">Cache Staheli</a> , <a href="#">David Soroko</a> , <a href="#">Enigo</a> , <a href="#">fabian</a> , <a href="#">fgb</a> , <a href="#">JudgingNotJudging</a> , <a href="#">KIRAN KUMAR MATAM</a> , <a href="#">Nicktar</a> , <a href="#">P.J.Meisch</a> , <a href="#">Stephen C</a>
57	<code>sun.misc.Unsafe</code>	<a href="#">4444</a> , <a href="#">Daniel Nugent</a> , <a href="#">Grexis</a> , <a href="#">Stephen C</a> , <a href="#">Suminda Sirinath S. Dharmasena</a>
58	<code>ThreadLocal</code>	<a href="#">Dariusz</a> , <a href="#">Liju Thomas</a> , <a href="#">Manish Kothari</a> , <a href="#">Nithanim</a> , <a href="#">taer</a>
59	<code>TreeMap</code> <code>TreeSet</code>	<a href="#">Malt</a> , <a href="#">Stephen C</a>
60	<code>Varargs</code>	<a href="#">Daniel Nugent</a> , <a href="#">Dushman</a> , <a href="#">Omar Ayala</a> , <a href="#">Rafael Pacheco</a> , <a href="#">RamenChef</a> , <a href="#">VGR</a> , <a href="#">xsami</a>
61	<code>WeakHashMap</code>	<a href="#">Amit Gujarathi</a> , <a href="#">KIRAN KUMAR MATAM</a>
62	XJC	<a href="#">Danilo Guimaraes</a> , <a href="#">fabian</a>
63	XML XPath <code>Navigator</code>	<a href="#">17slim</a> , <a href="#">manouti</a>
64	XOM - XML <code>DOM</code>	<a href="#">Arthur</a> , <a href="#">Makoto</a>
65	<code>ByteBuffer</code>	<a href="#">Mykola Yashchenko</a>
66	<code>ByteBuffer</code>	<a href="#">1d0m3n30</a> , <a href="#">Bohemian</a> , <a href="#">Holger</a> , <a href="#">Idcmp</a> , <a href="#">Jon Ericson</a> , <a href="#">kristyna</a> , <a href="#">Michael Piefel</a> , <a href="#">Stephen C</a> , <a href="#">Vogel612</a>
67	<code>ByteBuffer</code>	<a href="#">adino</a> , <a href="#">Alex</a> , <a href="#">assylias</a> , <a href="#">bfd</a> , <a href="#">Bhagyashree Jog</a> , <a href="#">bowmore</a> , <a href="#">Burkhard</a> , <a href="#">Chetya</a> , <a href="#">corsiKa</a> , <a href="#">Dariusz</a> , <a href="#">Diane Chastain</a> , <a href="#">DimaSan</a> , <a href="#">dimo414</a> , <a href="#">Fildor</a> , <a href="#">Freddie Coleman</a> , <a href="#">GPI</a> , <a href="#">Grzegorz Górkiewicz</a> , <a href="#">hd84335</a> , <a href="#">hellrocker</a> , <a href="#">hexafraction</a> , <a href="#">Ilya</a> , <a href="#">james large</a> , <a href="#">Jens Schauder</a> , <a href="#">Johannes</a> , <a href="#">Jorn Vernee</a> , <a href="#">Kakarot</a> , <a href="#">Lance Clark</a> , <a href="#">Malt</a> , <a href="#">Matěj Kripner</a> , <a href="#">Md. Nasir Uddin Bhuiyan</a> , <a href="#">Michael Piefel</a> , <a href="#">michaelbahr</a> , <a href="#">Mitchell Tracy</a> , <a href="#">MSB</a> , <a href="#">Murat K.</a> , <a href="#">Mureinik</a> , <a href="#">mvd</a> , <a href="#">NatNgs</a> , <a href="#">nickguletskii</a> , <a href="#">Olivier Durin</a> , <a href="#">OlivierTheOlive</a> , <a href="#">Panda</a> , <a href="#">parakmiakos</a> , <a href="#">Paweł Albecki</a> , <a href="#">ppetverka</a> , <a href="#">RamenChef</a> , <a href="#">Ravindra babu</a> , <a href="#">rd22</a> , <a href="#">RudolphEst</a> , <a href="#">snowe2010</a> , <a href="#">Squidward</a> , <a href="#">Stephen C</a> , <a href="#">Sudhir Singh</a> , <a href="#">Tobias Friedinger</a> , <a href="#">Unihedron</a> , <a href="#">Vasilii Vlasov</a> , <a href="#">Vlad-HC</a> , <a href="#">Vogel612</a> , <a href="#">wolfcastle</a> , <a href="#">xTrollxDudex</a> , <a href="#">YCF_L</a> , <a href="#">Yury Fedorov</a> , <a href="#">ZX9</a>
68	<code>ByteBuffer</code>	<a href="#">GPI</a> , <a href="#">Kenster</a> , <a href="#">Powerlord</a> , <a href="#">user2296600</a>
69	<code>ByteBuffer</code>	<a href="#">alain.janinm</a> , <a href="#">Dariusz</a> , <a href="#">kajacx</a> , <a href="#">Kenster</a> , <a href="#">mnoronha</a>
70	<code>BufferedWriter</code>	<a href="#">Andrii Abramov</a> , <a href="#">fabian</a> , <a href="#">Jorn Vernee</a> , <a href="#">Robin</a> , <a href="#">VatsalSura</a>
71	<code>ByteBuffer</code>	<a href="#">Aimee Borda</a> , <a href="#">Blubberguy22</a> , <a href="#">dosdebug</a> , <a href="#">esin88</a> , <a href="#">Gerald Mücke</a> , <a href="#">Jorn</a>

		Vernee, Kineolyan, mnoronha, Nayuki, Rednivrug, Ryan Hilbert, Stephen C, thatguy
72	Fork / Join	Community, Joe C
73	JAXP API XML	GPI
74	static	17slim, Amir Rachum, Andrew Brooke, Arthur, ben75, CarManuel, Daniel Nugent, EJP, Hi I'm Frogatto, Mark Yisri, Sadiq Ali, Skepter, Squidward
75		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
76		Ankit Katiyar
77		Suketu Patel
78	SET	KIRAN KUMAR MATAM
79		Andreas
80		Jeet
81		JamesENL, KIRAN KUMAR MATAM
82		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
83		Daniel Nugent, Stephen C, Suminda Sirinath S. Dharmasena, xTrollxDudex
84		Do Nhu Vy, giucal, Jorn Vernee, Lord Farquaad, Yohanes Khosiawan
85		EJP, NageN, Thisaru Guruge
86	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 □ Pepeúpa ヽ
87	□□□□□□	Andrii Abramov, Conrad.Dean, Daniel Nugent, fabian, GPI, Hazem Farahat, JAVAC, Mshnik, Nolequen, Petter Friberg, Prateek Agarwal, sebkur, Stephen C
88	□□□□□□□□□□□□□□□□	Aasmund Eldhuset, Abhishek Balaji R, Catalina Island, Daniel M., intboolstring, Jonathan, Mark Yisri, Mureinik, NageN, ParkerHalo, Stephen C, Vogel612
89	□□□	A Boschman, Abubakkar, Andrey Rubtsov, Andrii Abramov, assylia, 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
90	□□□□□□□□	Fildor, Ironcache, Kröw, martin, Petter Friberg, Stephen C, Sujith Niraikulathan, Thisaru Guruge, uzaif
91	□□□□□□□□	Burkhard, Michael von Wenckstern, Stephen C
92	□□□	KIRAN KUMAR MATAM
93	□□	RAnders00
94	□□□□	Ironcache
95	□□	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
96	□ Java□□□□□□□□□□	Nikhil R
97	□ MultiThreaded□□□□□□□□ □ ThreadPoolExecutor °	Brendon Dugan
98	□□	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
99	□□□□□□□□	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
100	□□	esin88

101	000	Adrian Krebs, Amani Kilumanga, Daniel LIn, Dushman, Kakarot, Lernkurve, Markus L, NageN, Pawan, Ravindra babu, Saiful Azad, Stephen C
102	0000JAR00	manouti
103	000	Ordriel
104	000	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 Balim, 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 Štastný, 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 hadomfa, 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, ☼Xocę ☐ Pepeúpa ッ
105	0000000	M M
106	0000	Ilya
107	00000	bloo, Display Name, rakwaht, Squidward
108	000000	Community, Jon Ericson, Jorn Vernee, Tarik Yilmaz, Tomasz Bawor, victorantunes, Vogel612
109	00	1d0m3n30, EJP, ParkerHalo, Stephen C, ThePhantomGamer
110	0000000	John Nash, shibli049
111	00000	alphaloop, hexafraction, Uux
112	00Java000000	4444, Adeel Ansari, ajablonski, akhilsk, Alex A, altomnr, Ani Menon, Anthony Raymond, anuvab1911, Configure, CraftedCart, Emil Sierżęga, Gautam Jose, hd84335, ipsi, Jeffrey Brett Coleman,



		Lambda Ninja, Nithanim, Radouane ROUFID, Rakitić, ronnyfm, Sanandrea, Sandeep Chatterjee, sohnryang, Stephen C, Snaðowfal, tonirush, Walery Strauch, Ze Rubeus
113	00	Adam Ratzman, Adil, Daniel M., Drayke, VISHWANATH N P
114	00000000000000000000	Bohemian
115	000000000000	A Boschman, Ad Infinitum, Andrii Abramov, Ani Menon, anuvab1911, Arthur Nosedá, 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
116	0000	Ayush Bansal, Christophe Weis, Jonathan
117	0000	Andrii Abramov, arcy, Vasiliy Vlasov
118	000	ArcticLord, Enigo, MadProgrammer, ppeterka
119	000	17slim, Arthur, J Atkin, Jabir, KIRAN KUMAR MATAM, Marvin, peterh, Stephen C, VGR, vorburger
120	000000	ChemicalFlash, DimaSan, fgb, hd84335, Mshnik, RamenChef, Sandesh, sargue, Slava Babin, Stephen C, tynn
121	0000	Amani Kilumanga, Andy Thomas, Asaph, ced-b, Daniel M., fabian, hd84335, intboolstring, kaotikmynd, Laurel, Makoto, nhahtdh, ppeterka, Ram, RamenChef, Saif, Tot Zam, Unihedron, Vogel612
122	000Java00	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
123	000	akhilsk, Batty, Bilesh Ganguly, Burkhard, EJP, emotionlessbananas, faraa, GradAsso, KIRAN KUMAR MATAM, noscreenname, Onur, rokonoid, Siva Sainath Reddy Bandi, Vasilis Vasilatos, Vasiliy Vlasov
124	000000	ipsi, mnoronha
125	000	Alek Mieczkowski, Chirag Parmar, Community, Jon Ericson, JonasCz, Ram, RamenChef, Redterd, Stephen C, sun-solar-arrow, ☕Xocę ☐ Π epeúpa ヽ
126	00	100rabh, A Boschman, Abhishek Jain, Adowrath, Alex Shestеров, Andrew Tobilko, Andrii Abramov, Cà phê đên, 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 Niraiikulathan, Thomas Gerot, user187470, Vasiliy Vlasov, Vince Emigh, xwoker, Zircon

127	□□□I / O.	Aaron Franke, Ani Menon, Erkan Haspulat, Francesco Menzani, jayants, Lankymart, Loris Securo, manetsus, Olivier Grégoire, Petter Friberg, rolve, Saagar Jha, Stephen C
128	□□□□□□	Jacob G.
129	□□	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, clphr, Cache Staheli, Cerbrus, Charitha, Charlie H, Chris Midgley, Christophe Weis, Christopher Schneider, Codebender, coder-croc, Cold Fire, Colin Pickard, Community, Configure, CptEric, Daniel Käfer, Daniel Stradowski, Dariusz, DarkV1, David G., DeepCoder, Devid Farinelli, Dhruvajyoti Gogoi, Dmitry Ginzburg, dorukayhan, Duh-Wayne-101, Durgpal Singh, DVarga, Ed Cottrell, Edwin 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 Rummier, 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, lllamositopia, 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 Fadhil, Muhammed Refaat, Muntasir, Mureinik, Mzzzzzz, NageN, Nathaniel Ford, Nayuki, nicael, Nigel Nop, niyasc, nouAdKzerD, 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 Wiczorek, Squidward, stackptr, stark, Stephen C, Stephen Leppik, Sualeh Fatehi, sudo, Sumurai8, Sunnyok, syb0rg, tbdot, tdelev, tharkay, Thomas, ThunderStruct, Tol182, ɔoleɛz əɥɔ qoq, tpunt, Travis J, Tunaki, Un3qual, Unihedron, user6653173, uzaif, vasilil111, VedX, Ven, Victor G., Vikas Gupta, vincentvanjoe, Vogel612, Wilson, Winter, X.lophix, YCF_L, Yohanes Khosiawan □□□, yuku, Yury Fedorov, zamonier, ☺ Xocę □ Переура ㄣ
130	□□I / O.	Alper Firat 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, Moshior, 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
131	0000I / O.	dorukayhan, niheno, TuringTux
132	00	Jonathan, Makoto, rajah9, RamenChef, The Guy with The Hat, Uri Agassi
133	0000000	Bob Rivers, cdm, kann, Makoto, mnoronha, ppeterka, Ram, VGR
134	0000000java.time.*0	Bilbo Baggins, bowmore, Michael Piefel, Miles, mnoronha, Simon, Squidward, Tarun Maganti, Vogel612, Xocę 0 Pepeúpa ˘
135	000	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
136	000000	mnoronha, ppeterka, Viacheslav Vedenin
137	0000000	KIRAN KUMAR MATAM
138	00000000	Code.IT, dimo414, Eduard Wirch, emotionlessbananas, Squidward, sun-solar-arrow
139	00	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, Xocę 0 Pepeúpa ˘
140	0000	KIRAN KUMAR MATAM
141	00000000	Sugan
142	0000	Andrii Abramov, Asiat, BrunoDM, ced-b, Codebender, Dylan, George Bailey, Jeremy, Ralf Kleberhoff, RamenChef, Thomas Gerot, tynn, Vogel612
143	00	Jonathan, user140547
144	00	1d0m3n30, 4444, Aaron Digulla, Abhishek Jain, Alex Meiburg, alexs, 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, 7oleæz æq1 qoq, Unihedron, Vlad-HC, Wesley, Wilson, yiwei, Yury Fedorov
145	□	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
146	□□□□□	bn., noscreenname, P.J.Meisch, RamenChef, TuringTux
147	□□	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, ☿Xocę □ Πεπεύπα Ψ
148	□ SortedMap	Amit Gujarathi
149	□□ Java□□	Tony
150	□□□□	100rabh, A_Arnold, Alex, Andrii Abramov, Bob Rivers, Cache Staheli, DimaSan, Jasper, Kakarot, Kuroda, Manuel Vieda, Michael Piefel, phatfingers, RamenChef, Skylar Sutton, Vivek Anoop
151	□ 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
152	□ NumberFormat	arpit pandey, John Nash, RamenChef, ☿Xocę □ Πεπεύπα Ψ
153	□ ServiceLoader	fabian, Florian Genser, Gerald Mücke

154	StringBuffer	Amit Gujarathi
155		Arthur, Burkhard, devnull169, DonyorM, glee8e, Grayson Croom, Ilya, Malt, Matej Kormuth, Matthieu, Mine_Stone, ppeterka, RamenChef, Stephen C, Tot Zam, vsav
156		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
157		Andy Thomas, Bob Rivers, ppeterka, vorburger, yitzih
158		1d0m3n30, Andreas, EJP, Li357, RamenChef, shmosel, Stephen C, Stephen Leppik
159	java.util.logging	bn., Christophe Weis, Emil Sierżęga, P.J.Meisch, vallismortis
160	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
161		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, Vogel1612, ☕Xocę ☐ Pepeúpa ☺
162		JD9999, KIRAN KUMAR MATAM, Mureinik, Stephen C, VatsalSura
163		Alexey Lagunov
164		Androbin, Christian, Emily Mabrey, Enwired, fabian, Gerald Mücke, Jesse van Bekkum, Kenster, Stephen C, timbooo, VGR, vorburger
165		Abhijeet
166		Chirag Parmar, DarkV1, Gihan Chathuranga, Jabir, JonasCz, Kaushal28, Lachlan Dowding, Laurel, Maarten Bodewes, Matt Clark, PSo, RamenChef, Shaan, Stephen C, still_learning
167	Iterable	Abubakkar, Comic Sans, Dariusz, Hulk, Lukas Knuth, RamenChef, Stephen C, user1121883, WillShackleford
168		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, Vogel1612, Winter

169	□□□□□	RamenChef
170	□□	Andy Thomas, atom, Bobas_Pett, Ce7, charlesreid1, Configure, David Soroko, fabian, hamena314, hd84335, JavaHopper, Javant, Matej Kormuth, mayojava, Nicktar, Peter Gordon, RamenChef, Raviteja, Ruslan Bes, Stephen C, sumit
171	□□□□□□□□RMI□	RamenChef, smichel, Stephen C, user1803551, Vasiliy Vlasov
172	□□□□	John DiFini
173	□□	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, hirosh, 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
174	□□□Deques	Ad Infinitum, Alek Mieczkowski, Androbin, DimaSan, engineercoding, ppeterka, RamenChef, rd22, Samk, Stephen C
175	□□□□□	Arthur, David Grant, David Soroko, dorukayhan, F. Stephen Q, Kichiin, MasterBlaster, michaelbahr, rokonoid, Stephen C, Thodgnir
176	□	A_Arnold, atom, ced-b, Chirag Parmar, Daniel Stradowski, demongolem, DimaSan, fabian, Kaushal28, Kenster
177	□□	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, mystarocks, 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
178	□□□□□□	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
179	□□	Dac Saunders, Petter Friberg, RamenChef, TNT, tonirush, Tot Zam, Vogel612
180	□ - Java□□	gobes, KIRAN KUMAR MATAM
181	□□□□	FFY00, Flow, Holger, Makoto, Stephen C
182	□□□□	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

183 0000

4castle, Filip Smola, Joshua Carmody, Nick Donnelly, RamenChef, Squidward

184 0000

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