

 免費電子書

學習

ANTLR

Free unaffiliated eBook created from
Stack Overflow contributors.

#antlr

.....	1
1: ANTLR	2
.....	2
.....	2
Examples.....	3
.....	3
2: ANTLR v3	4
Examples.....	4
.....	4
EclipseANTLR.....	4
3: ANTLR v4	6
.....	6
Examples.....	6
.....	6
Build Automation.....	7
EclipseHello World.....	7
Visual Studio 2015ANTLRNuget.....	8
.....	10
4: ANTLR/	13
Examples.....	13
.....	13
Python.....	13
5: Lexer4	15
Examples.....	15
.....	15
.....	15
.....	15
.....	16
.....	16
Lexer.....	17
.....	17
6: TestRig / grun	18

Examples.....	18
TestRig.....	18
TestRig.....	18
Visual Parse Tree.....	18
7:	22
Examples.....	22
.....	22
8:	23
.....	23
Examples.....	23
.....	23
.....	25

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

It is an unofficial and free ANTLR 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 ANTLR.

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

1: ANTLR

ANTLR。 。 ANTLR。

- [antlr](#)

Antlr

Antlr。 antlrV1.V2.V3

- V1V1
- V2V2
- V3

AntlrJava。 antlr。

- Java
- C
- Python23
- JavaScript

2.0	1997-05-01
3.0	2011-01-19
4	2013121
4.1	2013-07-01
4.2	2014-02-05
4.2.1	2014325
4.2.2	201447
4.3	2014-06-19
4.4	2014716
4.5	2015123
4.5.1	2016716
4.5.2	2016130

4.5.3	2016331
4.6	○
4.7	2017330

Examples

hello world

```
// define a grammar called Hello
grammar Hello;
r   : 'hello' ID;
ID  : [a-z]+ ;
WS  : [ \t\r\n]+ -> skip ;
```

.g4/

```
Java -jar antlr-4.5.3-complete.jar Hello.g4

//OR if you have setup an alias or use the recommended batch file

antlr4 Hello.g4
```

Hello.g4

1. Hello.tokens
2. HelloBaseListener.java
3. HelloLexer.java
4. HelloLexer.tokens
5. HelloListener.java
6. HelloParser.java

ANTLR jar。 Java

```
javac *.java
```

ANTLR <https://riptutorial.com/zh-TW/antlr/topic/4453/antlr>


```

@header { //parser
    package pkgName; //optional
    import java.<whatever you need>.*;
}

@members { //parser
    // java code here
}

@lexer::header { //lexer
    package pkgName; //optional
    import java.<whatever you need>.*;
}

@lexer::members {
    // java code here
}

/*-----
 *  PARSER RULES (convention is all lowercase)
 *-----*/
parserule: LEXRULE;

/*-----
 *  LEXER RULES (convention is all uppercase)
 *-----*/
LEXRULE: 'a'..'z';

```

ANTLR v3 <https://riptutorial.com/zh-TW/antlr/topic/6629/antlr-v3>

3: ANTLR v4

ANTLR v4/。 ANTLRAST。 。 ANTLR v4JavaCJavaScriptPython2Python3。 C ++。 GUI IDE
Visual StudioIntelliJNetBeansEclipse。

[ANTLR](#)。 [ANTLRTerrence ParrANTLR The Definitive ANTLR 4 Reference](#)。

- 4.501/22/15 - JavaScriptC。 [4.5](#)
- 4.407/16/14 - Python2Python3。 [4.4](#)
- 4.306/18/14 - ;。 [4.3](#)
- 4.202/04/14 - /。 [4.2](#)
- 4.106/30/13 - ;ASTPNG。 [4.1](#)
- 4.001/21/13 - 。

Examples

ANTLRJava Jar。 ANTLRjarJava。

ANTLR JARANTLRJAR

```
Java -jar antlr-4.5.3-complete.jar
```

antlr-4.5.3-complete.jar。

```
ANTLR Parser Generator Version 4.5.3
-o ____          specify output directory where all output is generated
-lib ____        specify location of grammars, tokens files
-atn             generate rule augmented transition network diagrams
-encoding ____   specify grammar file encoding; e.g., euc-jp
-message-format ____ specify output style for messages in antlr, gnu, vs2005
-long-messages   show exception details when available for errors and warnings
-listener        generate parse tree listener (default)
-no-listener     don't generate parse tree listener
-visitor         generate parse tree visitor
-no-visitor      don't generate parse tree visitor (default)
-package ____    specify a package/namespace for the generated code
-depend          generate file dependencies
-D<option>=value set/override a grammar-level option
-Werror          treat warnings as errors
-XdbgST          launch StringTemplate visualizer on generated code
-XdbgSTWait      wait for STViz to close before continuing
-Xforce-atn      use the ATN simulator for all predictions
-Xlog            dump lots of logging info to antlr-timestamp.log
```

1. Add antlr4-complete.jar to CLASSPATH, either: Permanently:
Using System Properties dialog > Environment variables > Create or append to CLASSPATH variable Temporarily, at command line: SET CLASSPATH=.;C:\Javalib\antlr4-complete.jar;%CLASSPATH%
- 3.Create batch commands for ANTLR Tool, TestRig in dir in PATH

```
antlr4.bat: java org.antlr.v4.Tool %*
grun.bat:   java org.antlr.v4.gui.TestRig %*
```

.g4

```
Java -jar antlr-4.5.3-complete.jar yourGrammar.g4
```

-Dlanguage=C

```
java -jar antlr-4.5.3-complete.jar yourGrammar.g4 -Dlanguage=CSharp
```

o

Build Automation

[ANTLR](#) o

MavenGradle org.antlr:antlr4-runtime o

- - maven [Maven](#) org.antlr:antlr4 o

EclipseHello World

ANTLR 4.5.3Eclipse NeonANTLR 4 IDE 0.3.5Java 1.8

1. [ANTLR](#) o ANTLR Javajar o Java o o

2. EclipseANTLR IDE o

- Eclipse“Eclipse Marketplace” o
- “antlr” o
- “ANTLR 4 IDE” o
- Confirm Selected FeaturesFinish o
- “” o
- Eclipse o

3. “...” o

- EclipseANTLR 4HOMEcom.github.jknack.antlr-4ide.Antlr4com.github.jknack.antlr-4ide.Antlr4 o o
- HOME o o
- [Xtext 2.7.3](#)antlr-nnn-complete.jar o
- Eclipse“” o
- “...” o
- xtext 2.7.3Archive ...Xtext 2.7.3OK o
- “”> o o

- Eclipse。

4. Eclipse / Java ANTLR。

- EclipseWindowPreferences。
- JavaBuild PathClasspath Variables。
- New ...NameFile ...antlr-nnn-complete.jar。 “”””。
- 。

5. ANTLR IDE。

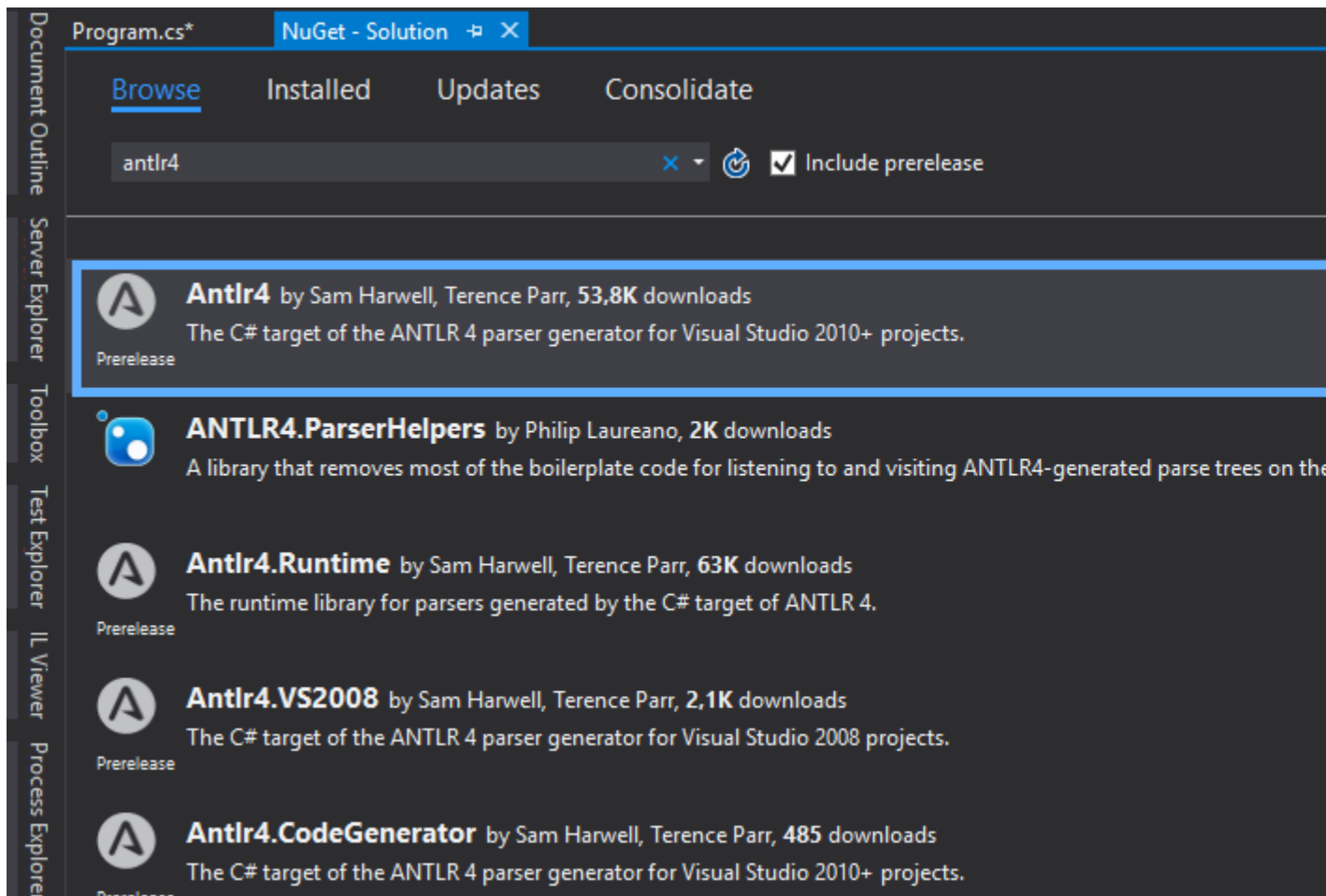
- EclipseWindowPreferences。
- ANTLR 4Tool。
- “””” java./antlr-java。
- “”””。

6. ANTLR 4。

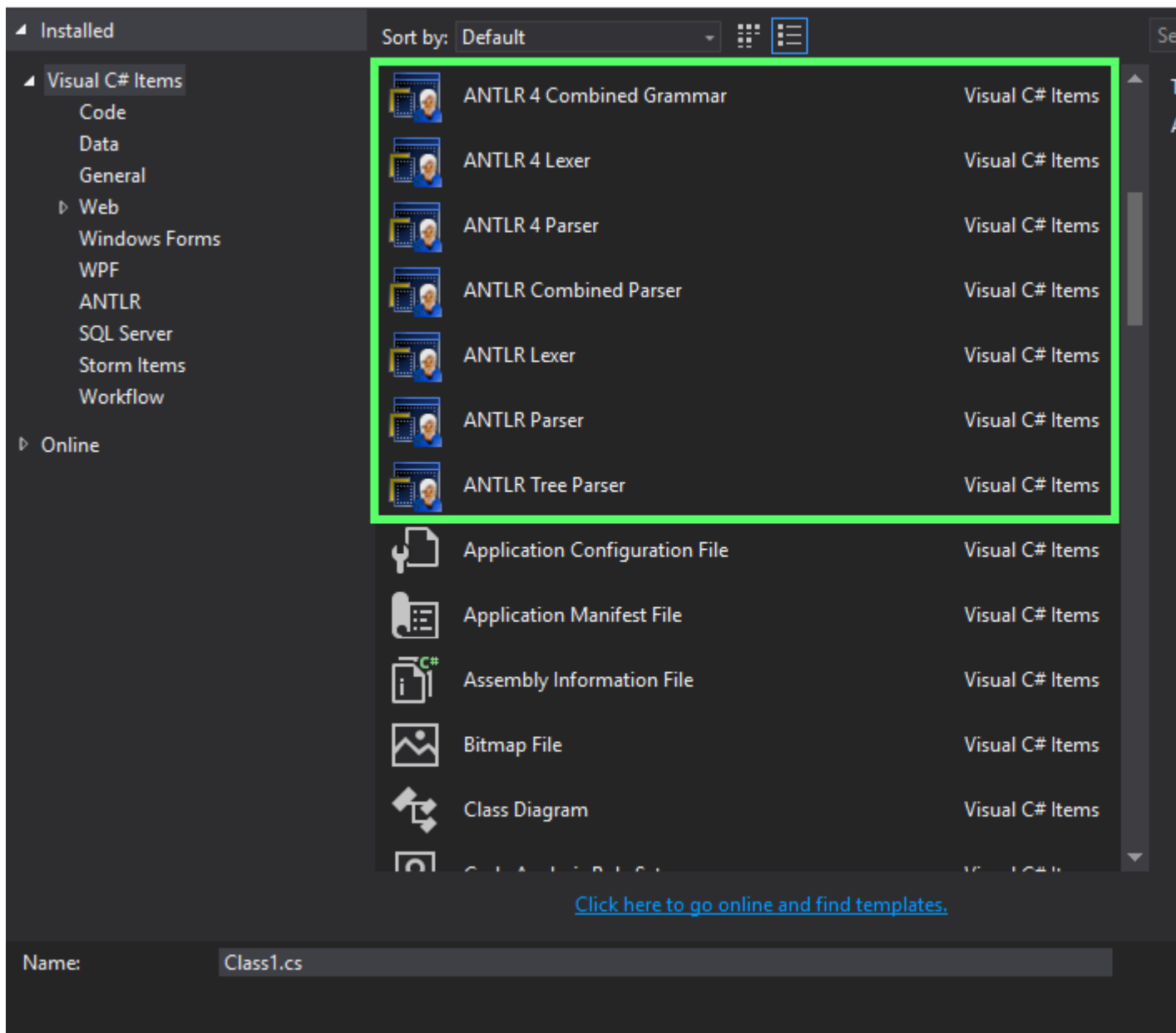
- EclipseFileNewProject。
- New ProjectGeneralANTLR 4 Project。
- NextFinish。
- Hello.g4“Hello World”。
- g4target5。

Visual Studio 2015ANTLRNugget

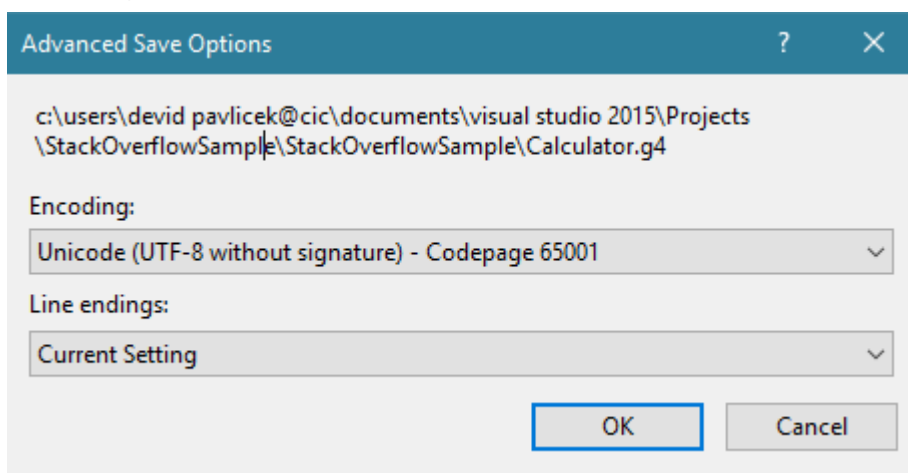
1. Visual Studio 2015→→Antlr。 ANTLRSam HarwellVisual Studio。
2. 。



3. ◦ ANTLR4◦

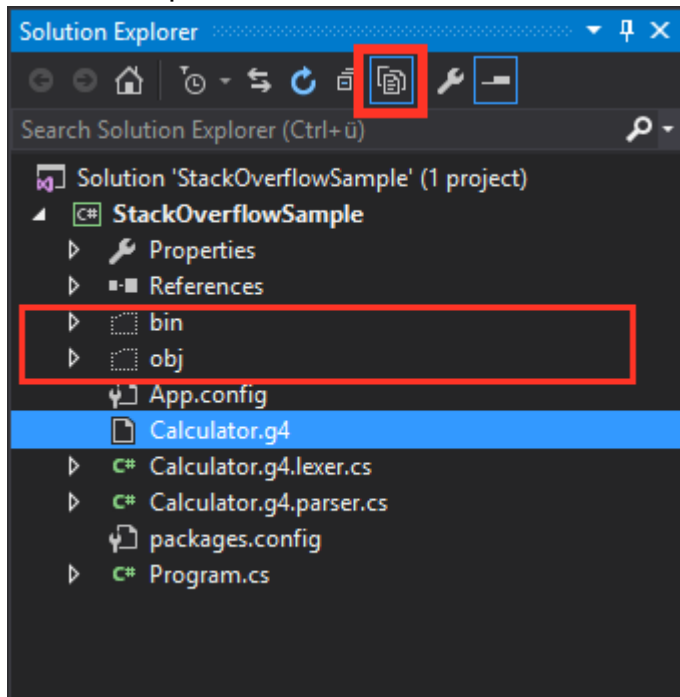


4. ANTLR.g4→Unicode UTF-8 - 65001。。

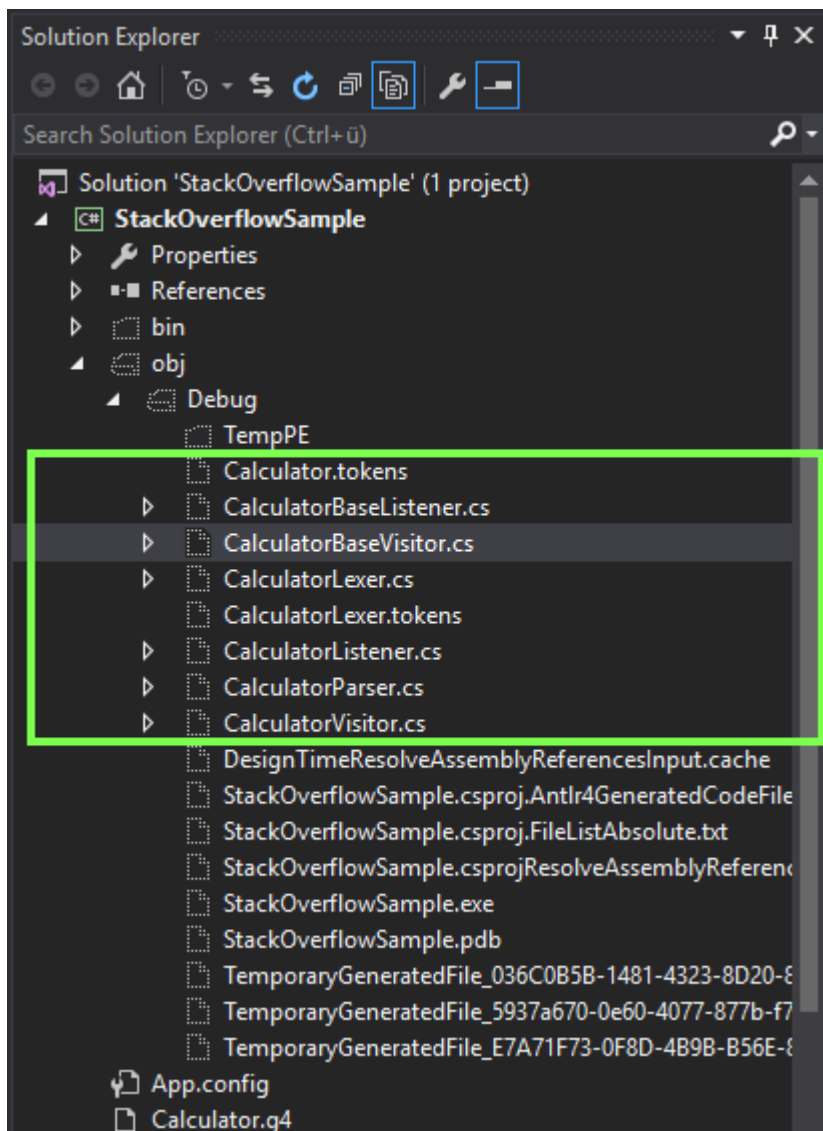


- ANTLR 4 Combined GrammarCalculator.g4
- Github [Tom Everett](#)
-
-

Solution Explorer → ""。



-
- objSolution ExplorerrcsVisitorListener。 ◦ Visual Studio 2015ANTLR。



4: ANTLR/

Examples

ANTLR

1. C
2. Python
3. JavaScript
4. Java

ANTLRJava

```
Java -jar antlr-4.5.3-complete.jar yourGrammar.g4 //Will output a
java parser
```

OS/

```
antlr4 -Dlanguage=Python3 yourGrammar.g4
//with alias
java -jar antlr-4.5.3-complete.jar -Dlanguage=Python3 yourGrammar.g4
//without alias
```

'-Dlanguage'.g4

```
options {
    language = "CSharp";
}
//or
options {
    language="Python";
}
```

ANTLR

1. CSharp
2. Python 2
3. python 3

ANTLR

Python

ANTLR.jar.g4

```
1.yourGrammarNameListener.py
2.yourGrammarNameParser.py
3.yourGrammarName.tokens
...
```


pythonPythonANTLR。 IDE。

```
#main.py
import yourGrammarNameParser
import sys

#main method and entry point of application

def main(argv):
    """Main method calling a single debugger for an input script"""
    parser = yourGrammarNameParser
    parser.parse(argv)

if __name__ == '__main__':
    main(sys.argv)
```

。

```
#yourGrammarNameParser.py
from yourGrammarNameLexer import yourGrammarNameLexer
from yourGrammarNameListener import yourGrammarNameListener
from yourGrammarNameParser import yourGrammarNameParser
from antlr4 import *
import sys

class yourGrammarNameParser(object):
    """
    Debugger class - accepts a single input script and processes
    all subsequent requirements
    """
    def __init__(self): # this method creates the class object.
        pass

#function used to parse an input file
def parse(argv):
    if len(sys.argv) > 1:
        input = FileStream(argv[1]) #read the first argument as a filestream
        lexer = yourGrammarNameLexer(input) #call your lexer
        stream = CommonTokenStream(lexer)
        parser = yourGrammarNameParser(stream)
        tree = parser.program() #start from the parser rule, however should be changed to your
entry rule for your specific grammar.
        printer = yourGrammarNameListener(tree,input)
        walker = ParseTreeWalker()
        walker.walk(printer, tree)
    else:
        print('Error : Expected a valid file')
```

ANTLR。

。

ANTLR/ <https://riptutorial.com/zh-TW/antlr/topic/3414/antlr->

5: Lexer4

Examples

Lexer ◦ ◦

```
INTEGER: [0-9]+;
IDENTIFIER: [a-zA-Z_] [a-zA-Z_0-9]*;

OPEN_PAREN: '(';
CLOSE_PAREN: ')';
```

A	A
AB	AB
(A B)	AB
'text'	"""
A?	A
A*	A
A+	A
[A-Z0-9]	AZ0-9
'a'..'z'	
~[AZ]	-
.	

- ◦

```
INTEGER: DIGIT+
        | '0' [Xx] HEX_DIGIT+
        ;

fragment DIGIT: [0-9];
fragment HEX_DIGIT: [0-9A-Fa-f];
```

'{' ◦

```
OPEN_BRACE: '{';
```

```
parserRule: '{';
parserRule: OPEN_BRACE;
```

OPEN_BRACE ◦ ◦

◦

-
- '{'
-

```
grammar LexerPriorityRulesExample;

// Parser rules

randomParserRule: 'foo'; // Implicitly declared token type

// Lexer rules

BAR: 'bar';
IDENTIFIER: [A-Za-z]+;
BAZ: 'baz';

WS: [ \t\r\n]+ -> skip;
```

```
aaa foo bar baz barz
```

```
IDENTIFIER 'foo' BAR IDENTIFIER IDENTIFIER
```

- aaaIDENTIFIER

```
IDENTIFIER◦
```

- foo'foo'

```
randomParserRule'foo'IDENTIFIER◦
```

- barBAR

```
BARIDENTIFIER◦
```

- bazIDENTIFIER

```
BAZIDENTIFIER◦ BAR ◦
```

```
BAZ IDENTIFIERBAZ◦
```

- barzIDENTIFIER

```
BAR3 bar IDENTIFIER4◦ IDENTIFIERBAR ◦
```

◦ ◦

'foo' ◦

Lexer

```
WHITESPACE: [ \r\n] -> skip;
```

->◦

- skip
- channel(n)
- type(n)
- mode(n) pushMode(n) popMode more

{ ... }

```
IDENTIFIER: [A-Z]+ { log("matched rule"); };
```

{ ... }? ◦ **false**◦

```
IDENTIFIER: [A-Z]+ { identifierIsValid() }?;
```

◦

Lexer4 <https://riptutorial.com/zh-TW/antlr/topic/3271/lexer4>

6: TestRig / grun

Examples

TestRig

ANTLR。

ANTLR jarANTLR

```
export CLASSPATH="./usr/local/lib/antlr-4.5.3-complete.jar:$CLASSPATH"
```

Dotjava。

AlisesLinux / MAC / Unix

```
alias antlr4='java -jar /usr/local/lib/antlr-4.5.3-complete.jar'  
//or any directory where your jar is located
```

。

TestRig

TestRig

```
alias grun='java org.antlr.v4.runtime.misc.TestRig'
```

WindowsANTLR jarTestRig

```
java -cp .;antlr.4.5.3-complete.jar org.antlr.v4.runtime.misc.TestRig  
//or  
java -cp .;antlr.4.5.3-complete.jar org.antlr.v4.gui.TestRig
```

TestRig

```
grun yourGrammar yourRule -tree //using the setup alias  
java -cp .;antlr.4.5.3-complete.jar org.antlr.v4.gui.TestRig yourGrammar YourRule -tree //on  
windows with no alias  
java -cp .;antlr.4.5.3-complete.jar org.antlr.v4.gui.TestRig yourGrammar Hello r -tree  
//Windows with the grammar Hello.g4 starting from the rule 'r'.
```

Visual Parse Tree

ANTLR-gui。

JSON.g4

```

/** Taken from "The Definitive ANTLR 4 Reference" by Terence Parr */

// Derived from http://json.org
grammar JSON;

json
  : value
  ;

object
  : '{' pair (',' pair)* '}'
  | '{' '}'
  ;

pair
  : STRING ':' value
  ;

array
  : '[' value (',' value)* ']'
  | '[' ']'
  ;

value
  : STRING
  | NUMBER
  | object
  | array
  | 'true'
  | 'false'
  | 'null'
  ;

STRING
  : '"' (ESC | ~ ["\\])* '"'
  ;

fragment ESC
  : '\\\' ([\\"/bfnrt] | UNICODE)
  ;

fragment UNICODE
  : 'u' HEX HEX HEX HEX
  ;

fragment HEX
  : [0-9a-fA-F]
  ;

NUMBER
  : '-'? INT '.' [0-9] + EXP? | '-'? INT EXP | '-'? INT
  ;

fragment INT
  : '0' | [1-9] [0-9]*
  ;

// no leading zeros
fragment EXP
  : [Ee] [+\\-]? INT
  ;

// \\- since - means "range" inside [...]
WS
  : [ \\t\\n\\r] + -> skip
  ;

```

JSON

example.json

```
{
  "name": "John Doe",
  "age": 25,
  "address": {
    "streetAddress": "21 2nd Street",
    "city": "New York",
    "state": "NY",
    "postalCode": "10021-3100"
  },
  "phoneNumbers": [
    {
      "type": "home",
      "number": "212 555-1234"
    },
    {
      "type": "mobile",
      "number": "123 456-7890"
    }
  ],
  "children": [],
  "spouse": null
}
```

```
export CLASSPATH="./usr/local/lib/antlr-4.0-complete.jar:$CLASSPATH"

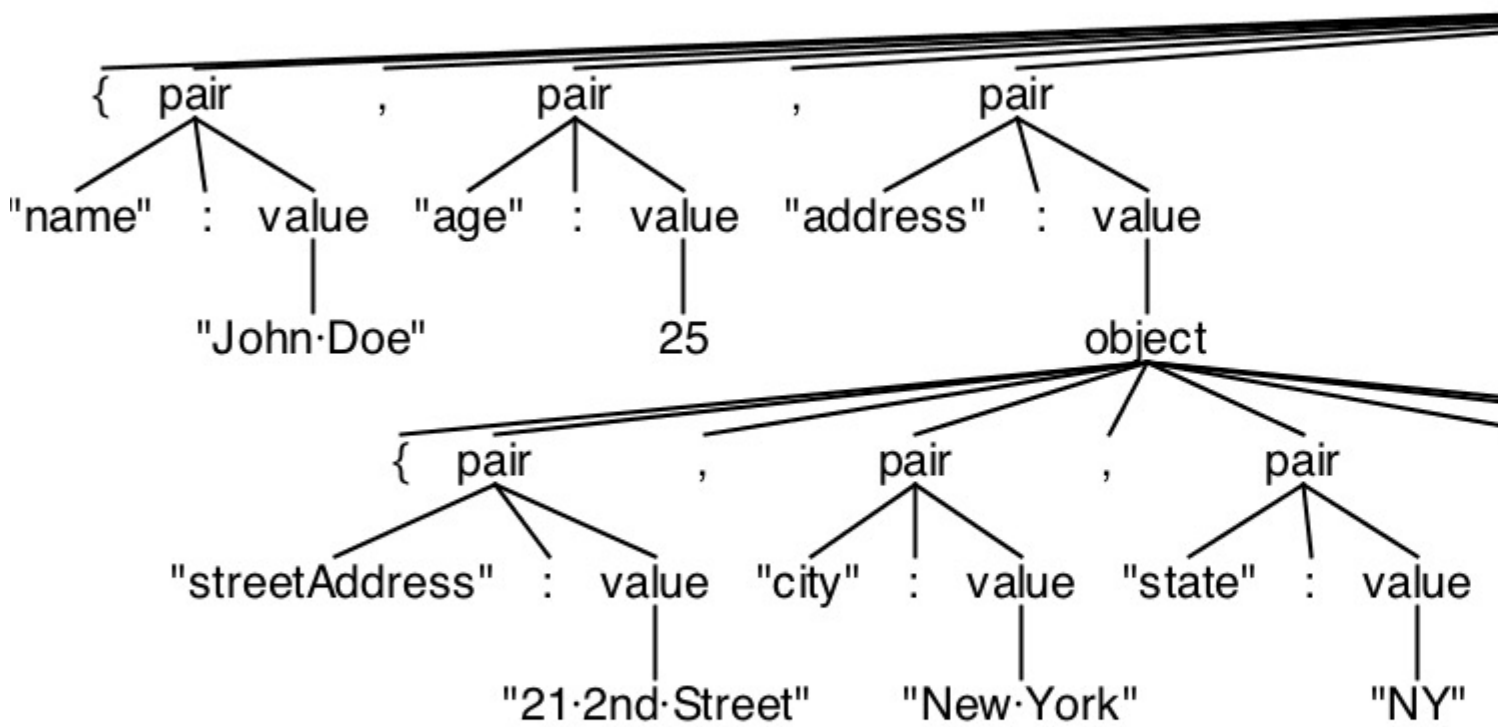
alias antlr4='java -jar /usr/local/lib/antlr-4.0-complete.jar'

alias grun='java org.antlr.v4.runtime.misc.TestRig'

antlr4 -o . -lib . -no-listener -no-visitor JSON.g4; javac *.java; grun JSON json -gui
example.json
```

.java.tokens.class

JSON.g4	JSONLexer.class	JSONListener.java
JSONParser\$PairContext.class	JSON.tokens	JSONLexer.java
JSONParser\$ArrayContext.class	JSONParser\$ValueContext.class	JSONBaseListener.class
JSONLexer.tokens	JSONParser\$JsonContext.class	JSONParser.class
JSONBaseListener.java	JSONListener.class	
JSONParser\$ObjectContext.class	JSONParser.java	



TestRig / grun <https://riptutorial.com/zh-TW/antlr/topic/3270/testrig---grun>

7:

Examples

#ANTLR。

```
// Rule
type : int      #typeInt
     | short    #typeShort
     | long     #typeLong
     | string   #typeString
     ;

// Tokens
int : 'int' ;
short : 'short' ;
long : 'long' ;
string : 'string' ;
```

[ParseTreeListener](#)

```
public void enterTypeInt (TypeShortContext ctx);
public void enterTypeShort (TypeIntContext ctx);
public void enterTypeLong (TypeLongContext ctx);
public void enterTypeString (TypeStringContext ctx);
```

<https://riptutorial.com/zh-TW/antlr/topic/6717/>

8:

ANTLRwalker。 visit。 。

Examples

Expr.g4

```
grammar Expr;
prog:      (expr NEWLINE)* ;
expr:     expr ('*' | '/') expr
        |   expr ('+' | '-') expr
        |   INT
        |   '(' expr ')'
        ;
NEWLINE  :  [\r\n]+ ;
INT      :  [0-9]+ ;
```

```
-visitor          generate parse tree visitor
-no-visitor       don't generate parse tree visitor (default)
```

commandline / terminal

```
java - jar antlr-4.5.3-complete.jar Expr.g4 -visitor
java - jar antlr-4.5.3-complete.jar Expr.g4 -no-visitor
```

/。

ExprBaseVisitor.java ExprVisitor.java 。

```
// Generated from Expr.g4 by ANTLR 4.5.3
import org.antlr.v4.runtime.tree.AbstractParseTreeVisitor;

/**
 * This class provides an empty implementation of {@link ExprVisitor},
 * which can be extended to create a visitor which only needs to handle a subset
 * of the available methods.
 *
 * @param <T> The return type of the visit operation. Use {@link Void} for
 * operations with no return type.
 */
public class ExprBaseVisitor<T> extends AbstractParseTreeVisitor<T> implements ExprVisitor<T>
{
    /**
     * {@inheritDoc}
     *
     * <p>The default implementation returns the result of calling
     * {@link #visitChildren} on {@code ctx}.</p>
     */
    @Override public T visitProg(ExprParser.ProgContext ctx) { return visitChildren(ctx); }
    /**
```

```
* {@inheritDoc}
*
* <p>The default implementation returns the result of calling
* {@link #visitChildren} on {@code ctx}.</p>
*/
@Override public T visitExpr(ExprParser.ExprContext ctx) { return visitChildren(ctx); }
}
```

<https://riptutorial.com/zh-TW/antlr/topic/8211/>

S. No		Contributors
1	ANTLR	Athafoud , cb4 , Community , D3181 , Gábor Bakos , KvanTTT
2	ANTLR v3	Athafoud , cb4
3	ANTLR v4	Athafoud , cb4 , Community , D3181 , Devid , Gábor Bakos , GRosenberg , Lucas Trzesniewski
4	ANTLR/	D3181
5	Lexer4	Athafoud , bn. , Loxley , Lucas Trzesniewski
6	TestRig / grun	bn. , D3181 , Lucas Trzesniewski , Pascal Le Merrer
7		bn. , Lucas Trzesniewski
8		D3181