



ANTLR Installation



Professor: Dr. Parsa

Ali Sedaghi
Amin Ghasvari



Install Java

ANTLR is written in Java, so you must have Java installed on your machine even if you are going to use ANTLR with, say, Python. ANTLR requires a Java version of 1.6 or higher.

1. Download Java Development Kit (JDK)

[Oracle java download page](#)










Or you can google for jdk and download it.

2. Install the JDK

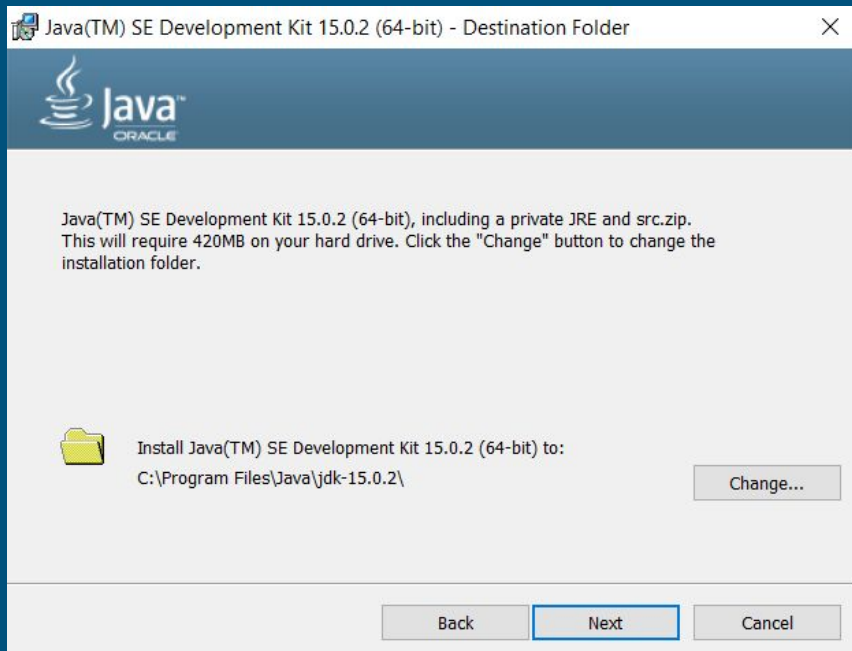
Download JDK

Java SE Development Kit 15.0.2

This software is licensed under the [Oracle Technology Network License Agreement](#) for Oracle Java SE

Product / File Description	File Size	Download
Linux ARM 64 RPM Package	141.82 MB	 jdk-15.0.2_linux-aarch64_bin.rpm
Linux ARM 64 Compressed Archive	157 MB	 jdk-15.0.2_linux-aarch64_bin.tar.gz
Linux x64 Debian Package	154.81 MB	 jdk-15.0.2_linux-x64_bin.deb
Linux x64 RPM Package	162.03 MB	 jdk-15.0.2_linux-x64_bin.rpm
Linux x64 Compressed Archive	179.35 MB	 jdk-15.0.2_linux-x64_bin.tar.gz
macOS Installer	175.93 MB	 jdk-15.0.2_osx-x64_bin.dmg
macOS Compressed Archive	176.51 MB	 jdk-15.0.2_osx-x64_bin.tar.gz
Windows x64 Installer	159.71 MB	 jdk-15.0.2_windows-x64_bin.exe
Windows x64 Compressed Archive	179.28 MB	 jdk-15.0.2_windows-x64_bin.zip

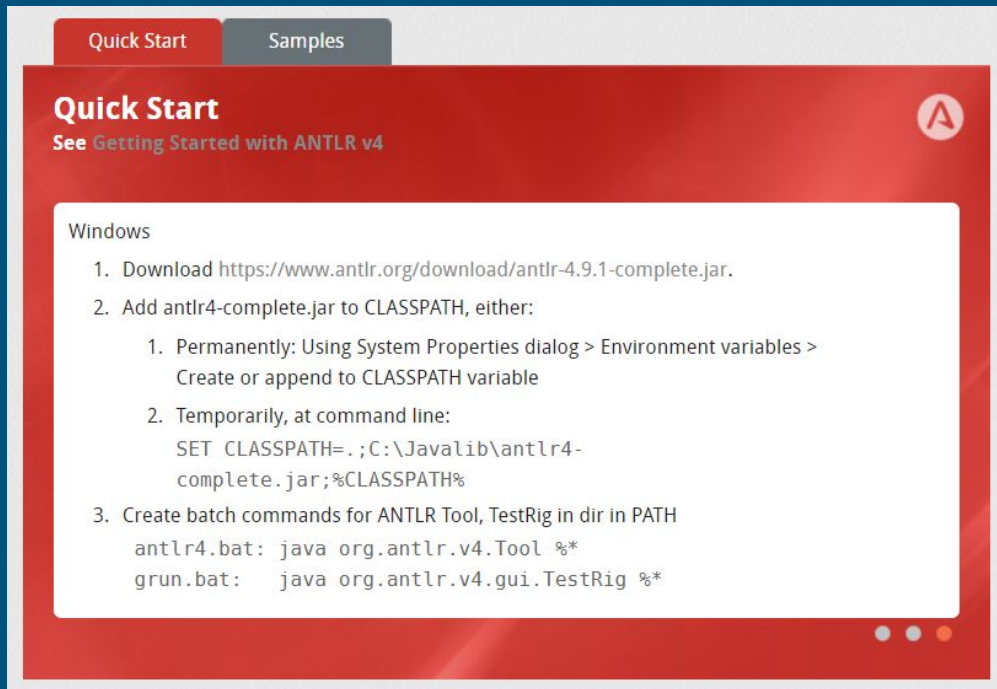
Install JDK



Install ANTLR

ANTLR official installation document:

<https://www.antlr.org>



The image shows a screenshot of the ANTLR website's 'Quick Start' page. The page has a red header with the title 'Quick Start' and a sub-link 'See Getting Started with ANTLR v4'. Below the header, there is a white box containing instructions for Windows installation. The instructions are numbered 1 through 3, detailing the download of the ANTLR-4.9.1-complete.jar file, how to add it to the CLASSPATH (either permanently via System Properties or temporarily via command line), and how to create batch commands for the ANTLR Tool and TestRig.

Quick Start

See [Getting Started with ANTLR v4](#)

Windows

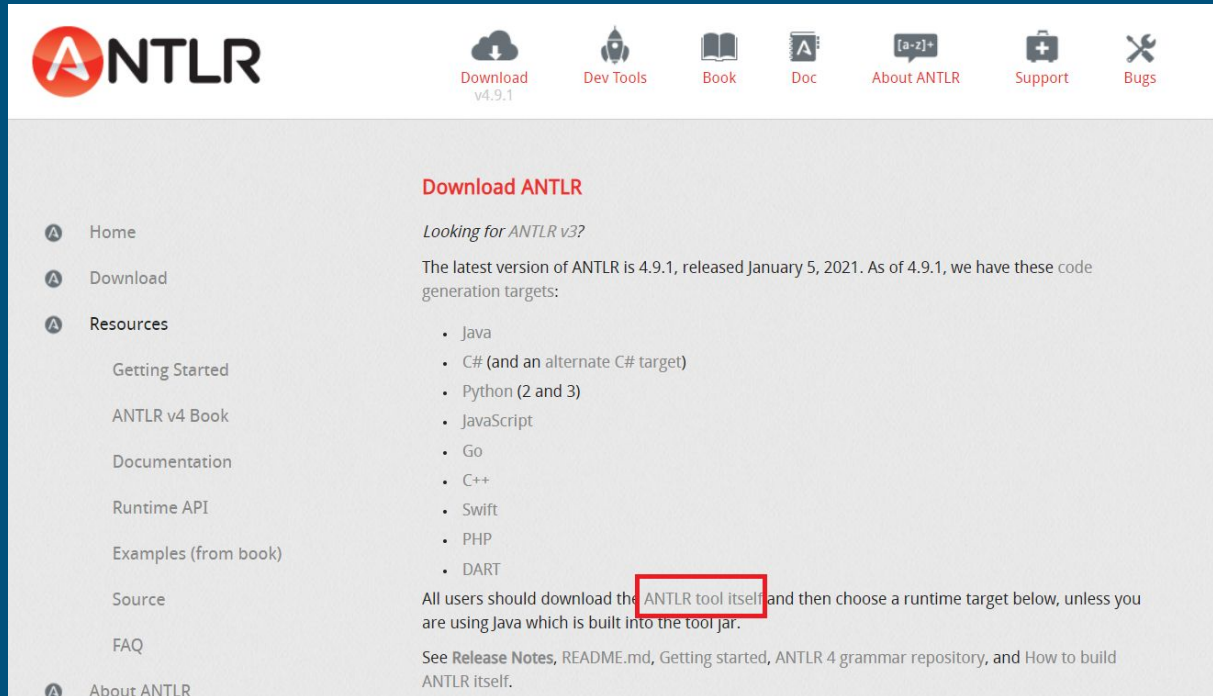
1. Download <https://www.antlr.org/download/antlr-4.9.1-complete.jar>.
2. Add antlr4-complete.jar to CLASSPATH, either:
 1. Permanently: Using System Properties dialog > Environment variables > Create or append to CLASSPATH variable
 2. Temporarily, at command line:

```
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 %*
```

1. Download ANTLR jar file (version 4.9.1)

[ANTLR download page](#)



The screenshot shows the ANTLR website's download page. The header features the ANTLR logo and navigation links: Download v4.9.1, Dev Tools, Book, Doc, About ANTLR, Support, and Bugs. The main content area is titled "Download ANTLR" and includes a sidebar with links to Home, Download, Resources, Getting Started, ANTLR v4 Book, Documentation, Runtime API, Examples (from book), Source, FAQ, and About ANTLR. The main text states that the latest version is 4.9.1, released on January 5, 2021, and lists supported code generation targets: Java, C# (and an alternate C# target), Python (2 and 3), JavaScript, Go, C++, Swift, PHP, and DART. A red box highlights the text "ANTLR tool itself" in the instruction: "All users should download the ANTLR tool itself and then choose a runtime target below, unless you are using Java which is built into the tool jar." Below this, it says "See Release Notes, README.md, Getting started, ANTLR 4 grammar repository, and How to build ANTLR itself."

ANTLR

Download v4.9.1 Dev Tools Book Doc About ANTLR Support Bugs

Download ANTLR

Looking for ANTLR v3?

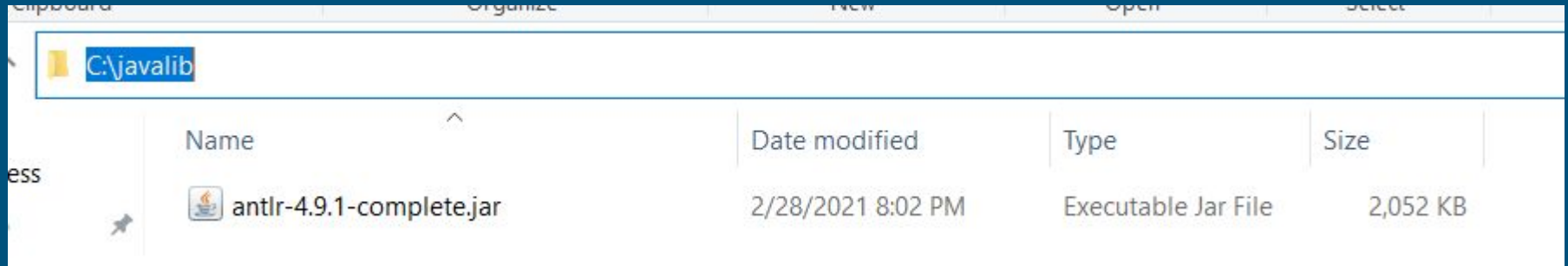
The latest version of ANTLR is 4.9.1, released January 5, 2021. As of 4.9.1, we have these code generation targets:

- Java
- C# (and an alternate C# target)
- Python (2 and 3)
- JavaScript
- Go
- C++
- Swift
- PHP
- DART

All users should download the ANTLR tool itself and then choose a runtime target below, unless you are using Java which is built into the tool jar.

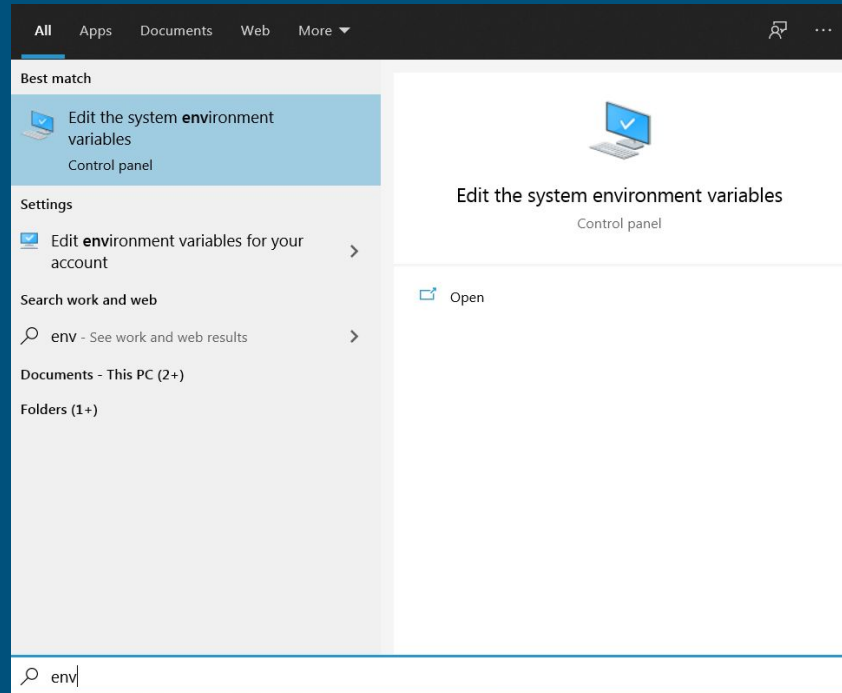
See [Release Notes](#), [README.md](#), [Getting started](#), [ANTLR 4 grammar repository](#), and [How to build ANTLR itself](#).

2. Save antlr-4.9.1-complete.jar to your directory for 3rd party Java libraries.
C:\javalib

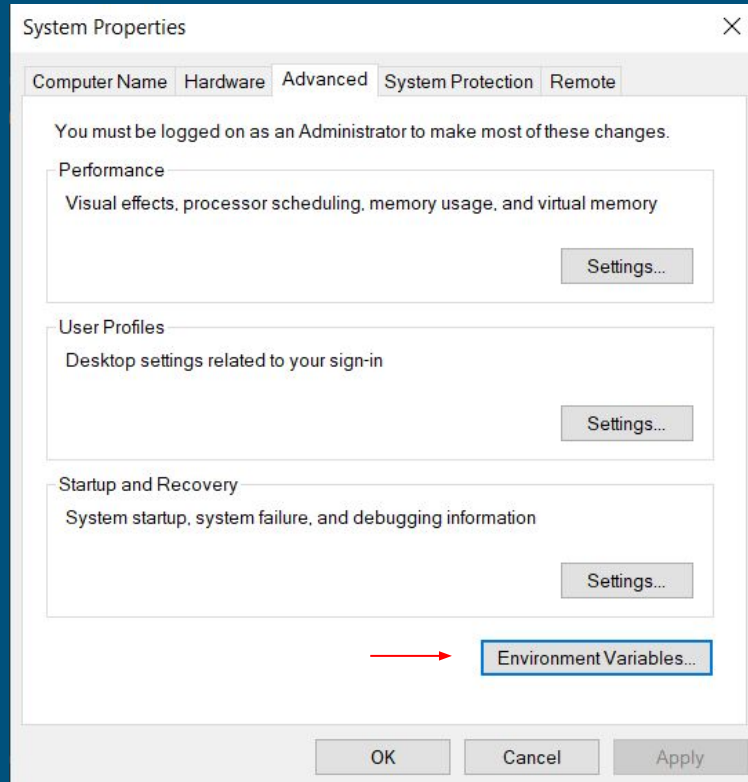


3. Add JDK bin files into Windows PATH variable.

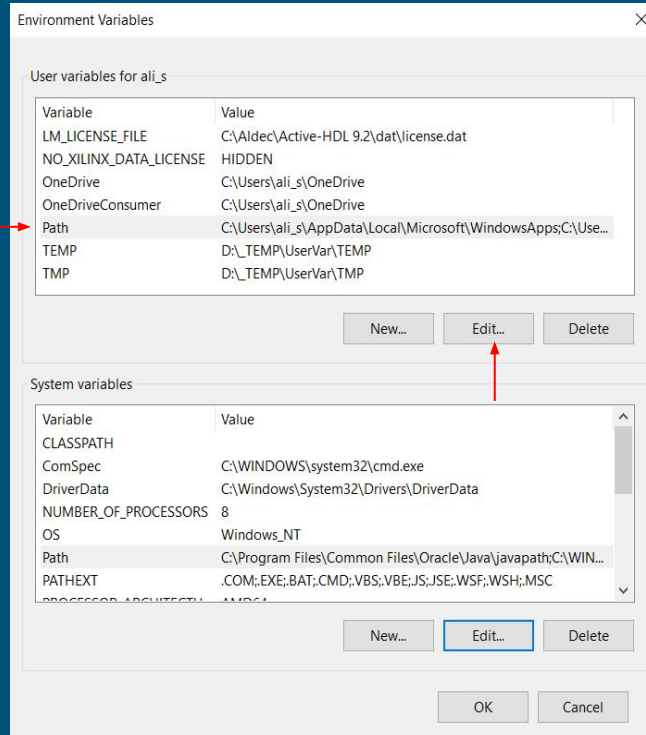
3.1. Use the windows search, to look for “env”.



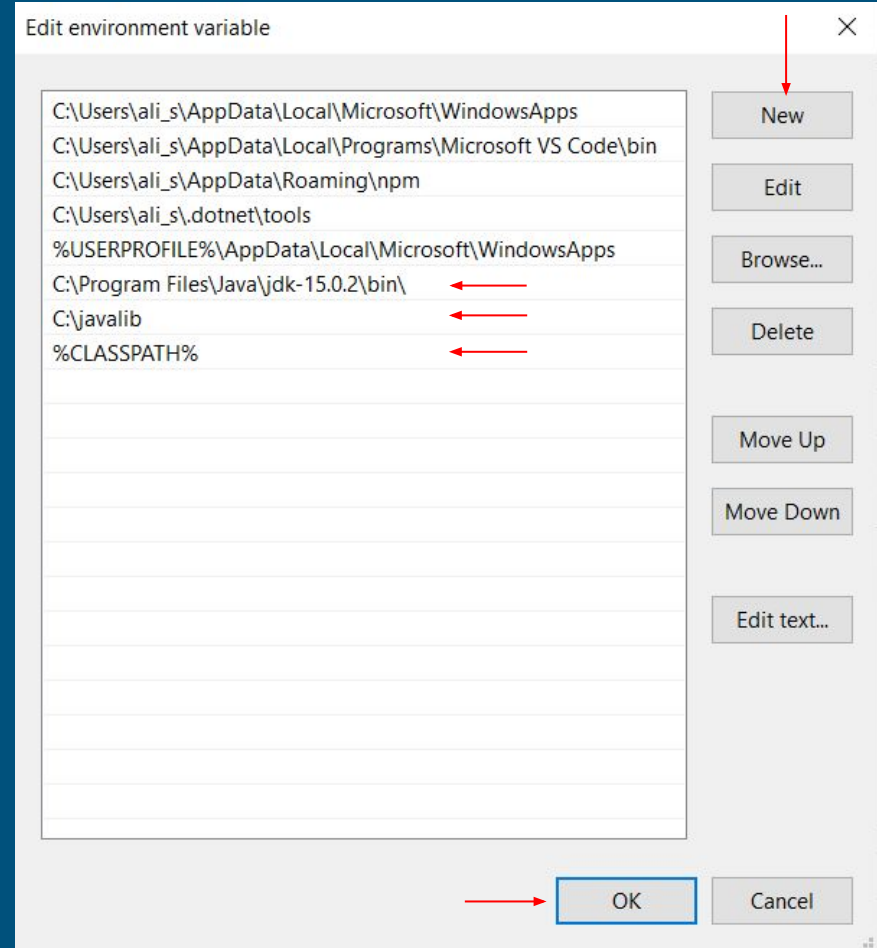
1.2. Click on “Environment Variables”



- 1.3. Inside User Variables look for a variable named “Path” or “PATH”
And double click on it OR select it and click on Edit



- 1.1.1. Click on “NEW” to create a new value.
- 1.1.2. Enter jdk bin directory path:
C:\Program Files\Java\jdk-15.0.2\bin\
- 1.1.3. Click on “NEW” to create a new value.
- 1.1.4. Enter java lib folder:
C:\javalib\
- 1.1.5. Click on “NEW” to create a new value.
- 1.1.6. Enter java class path:
%CLASSPATH%

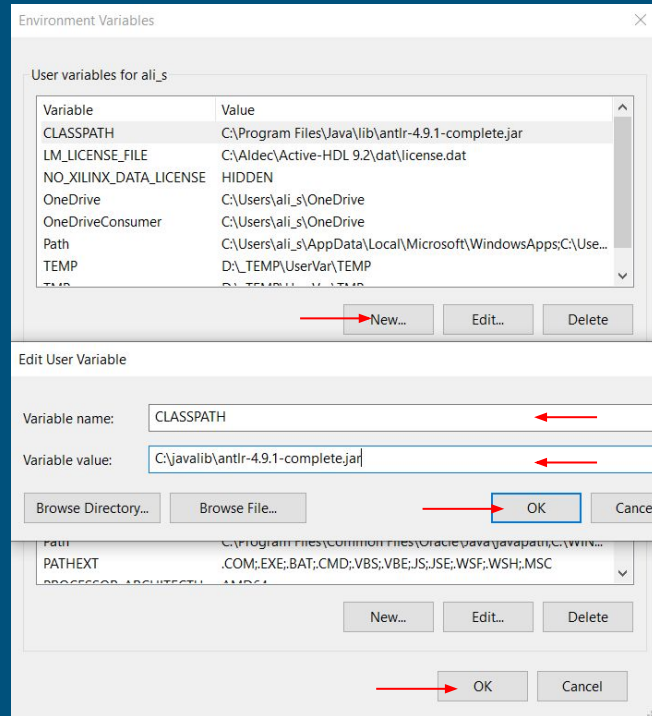


4. Add ANTLR jar file into Windows CLASSPATH variable.

4.1. Inside User Variables click on “New...” button and enter following fields as:

Variable name: CLASSPATH

Variable value: C:\javalib\antlr-4.9.1-complete.jar



5. Create batch commands for ANTLR Tool, TestRig in lib directory

5.1. Inside C:\javalib\ create 2 new files:

antlr4.bat

grun.bat

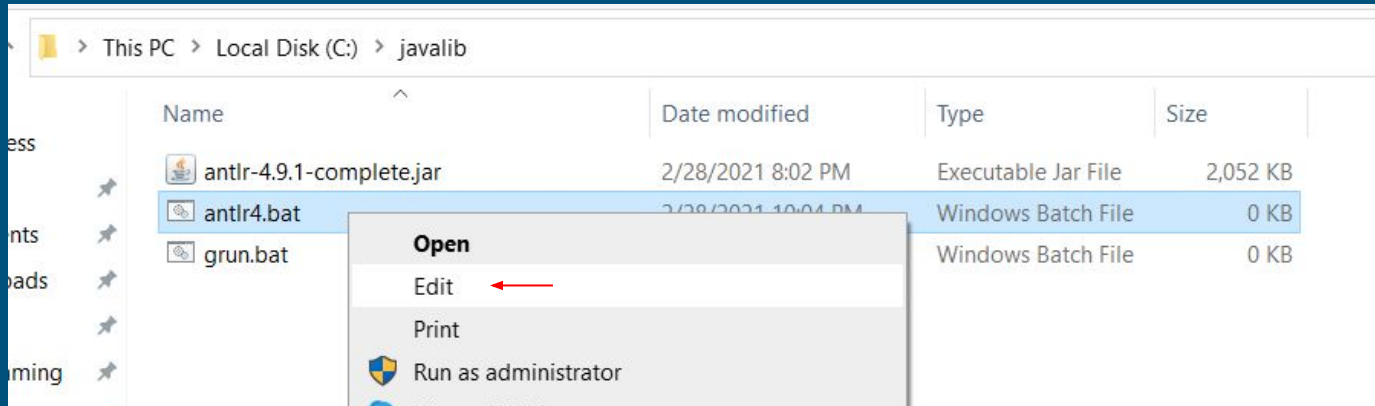
5.2. Open these bat files with notepad and copy and paste these lines.

5.2.1. Inside antlr4.bat:

```
java org.antlr.v4.Tool %*
```

5.2.2. Inside grun.bat:

```
java org.antlr.v4.gui.TestRig %*
```



antlr4.bat - Notepad

File Edit Format View Help

```
java org.antlr.v4.Tool %*
```

Ln 1, Col 26 100% Windows (CRLF) UTF-8

grun.bat - Notepad

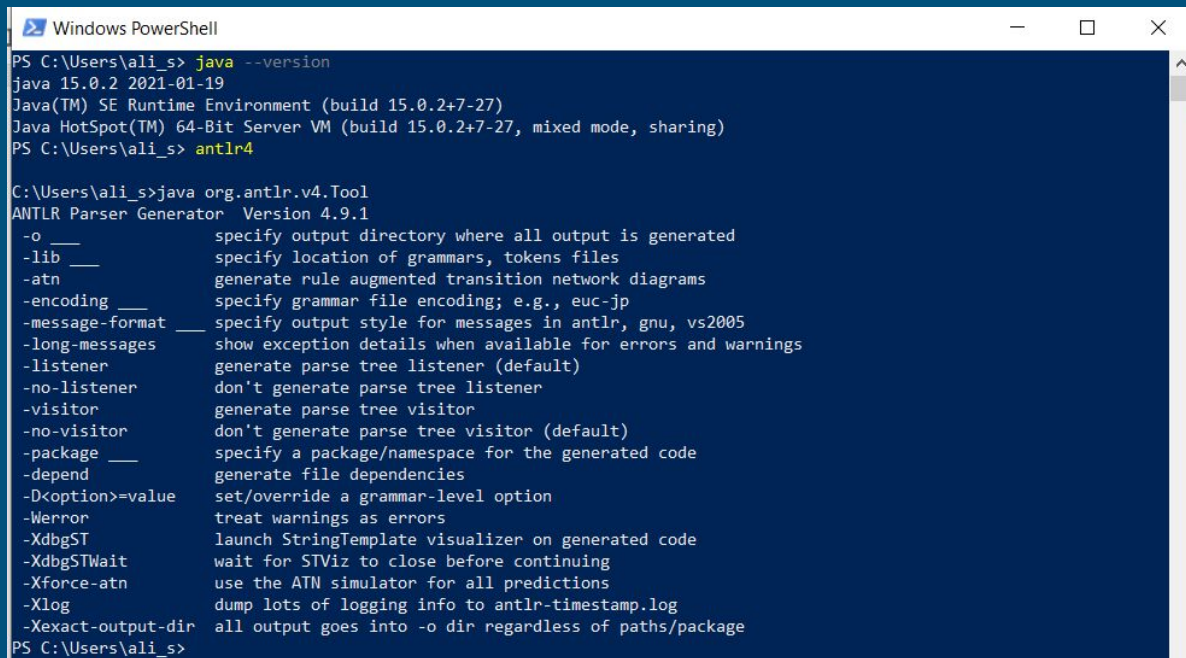
File Edit Format View Help

```
java org.antlr.v4.gui.TestRig %*
```

Ln 1, Col 33 100% Windows (CRLF) UTF-8

Now you need to check if ANTLER and Java are installed successfully.
Inside powershell enter these commands:

- `java --version`
- `antlr4`



```
Windows PowerShell
PS C:\Users\ali_s> java --version
java 15.0.2 2021-01-19
Java(TM) SE Runtime Environment (build 15.0.2+7-27)
Java HotSpot(TM) 64-Bit Server VM (build 15.0.2+7-27, mixed mode, sharing)
PS C:\Users\ali_s> antlr4

C:\Users\ali_s>java org.antlr.v4.Tool
ANTLR Parser Generator Version 4.9.1
-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
-Xexact-output-dir _____ all output goes into -o dir regardless of paths/package
PS C:\Users\ali_s>
```


Run ANTLR to generate
Parser and Lexer

- ★ ANTLR grammar files have **.g4** extension.
- ★ You may generate lexer and parser for every languages such as C# , Java, and Python.

For testing purposes download cpp14.g4 grammar file.

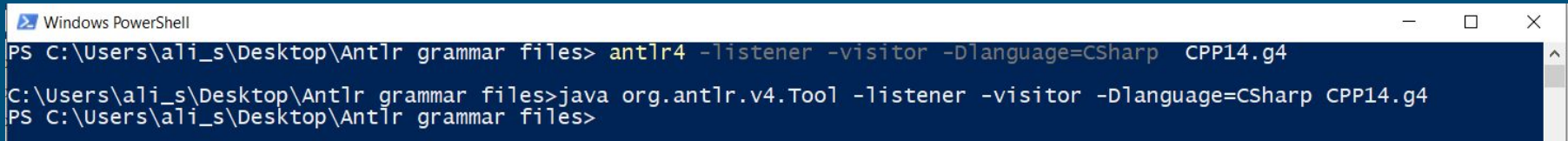
<https://github.com/antlr/grammars-v4/blob/master/antlr/antlr4/examples/CPP14.g4>

- ★ Inside CPP14.g4 directory open a powershell and enter below command:

`antlr4 -listener -visitor -Dlanguage=CSharp CPP14.g4`












- ★ Which antlr4.bat will translate it into:

`java org.antlr.v4.Tool -listener -visitor -Dlanguage=CSharp CPP14.g4`



```
Windows PowerShell
PS C:\Users\ali_s\Desktop\Antlr grammar files> antlr4 -listener -visitor -Dlanguage=CSharp CPP14.g4
C:\Users\ali_s\Desktop\Antlr grammar files> java org.antlr.v4.Tool -listener -visitor -Dlanguage=CSharp CPP14.g4
PS C:\Users\ali_s\Desktop\Antlr grammar files>
```

- ★ That command will generate Lexer, Parser, Listener, Visitor files.

Name	Date modified	Type	Size
 CPP14.g4	2/28/2021 10:40 PM	G4 File	28 KB
 CPP14.interp	2/28/2021 10:40 PM	INTERP File	95 KB
 CPP14.tokens	2/28/2021 10:40 PM	TOKENS File	4 KB
 CPP14BaseListener.cs	2/28/2021 10:40 PM	CS File	130 KB
 CPP14BaseVisitor.cs	2/28/2021 10:40 PM	CS File	108 KB
 CPP14Lexer.cs	2/28/2021 10:40 PM	CS File	103 KB
 CPP14Lexer.interp	2/28/2021 10:40 PM	INTERP File	51 KB
 CPP14Lexer.tokens	2/28/2021 10:40 PM	TOKENS File	4 KB
 CPP14Listener.cs	2/28/2021 10:40 PM	CS File	99 KB
 CPP14Parser.cs	2/28/2021 10:40 PM	CS File	817 KB
 CPP14Visitor.cs	2/28/2021 10:40 PM	CS File	59 KB












★ You can generate Lexer, Parser, Listener, Visitor files for any programming language by setting:

-Dlanguage=???

??? can be: Java, Python3, Python2, ...

For example:

`antlr4 -listener -visitor -Dlanguage=Java CPP14.g4`

Name	Date modified	Type	Size
 CPP14.g4	2/28/2021 10:40 PM	G4 File	28 KB
 CPP14.interp	2/28/2021 11:01 PM	INTERP File	95 KB
 CPP14.tokens	2/28/2021 11:01 PM	TOKENS File	4 KB
 CPP14BaseListener.java	2/28/2021 11:01 PM	IntelliJ IDEA	71 KB
 CPP14BaseVisitor.java	2/28/2021 11:01 PM	IntelliJ IDEA	52 KB
 CPP14Lexer.interp	2/28/2021 11:01 PM	INTERP File	51 KB
 CPP14Lexer.java	2/28/2021 11:01 PM	IntelliJ IDEA	51 KB
 CPP14Lexer.tokens	2/28/2021 11:01 PM	TOKENS File	4 KB
 CPP14Listener.java	2/28/2021 11:01 PM	IntelliJ IDEA	73 KB
 CPP14Parser.java	2/28/2021 11:01 PM	IntelliJ IDEA	649 KB
 CPP14Visitor.java	2/28/2021 11:01 PM	IntelliJ IDEA	43 KB

Install Antlr Plugin

File Edit View Navigate Code Refactor Run Tools VCS Window Help Compiler [C:\Users\Amin\MAG_term_6\Compiler] - ...CPP14.g4 - PyCharm

Compiler > CPP14.g4 >

Project > 1. Open settings from file menu

generate_tree.py x btree.cpp x CPP14.g4 x

24 grammar CPP14;

2. Search plugin

Settings

plugin

Appearance & Behavior

System Settings

Data Sharing

Notifications

Material Theme

Keymap

Editor

Inspections

Plugins

Build, Execution, Deployment

Required Plugins

3. Search Antlr

Plugins

Marketplace

Installed

antlr

Search Results (2)

Sort By: Relevance

ANTLR v4

338.9K 4.6 ANTLR Project

INSTALLED

Verilog support

3.2K 3.5 MrTsepa

INSTALL

4. Install Antlr and restart IDE.

ANTLR v4

338.9K 4.6 ANTLR Project

Tool integration

1.16-2019 Jan 23, 2021

Plugin homepage

This plugin is for ANTLR v4 grammars and includes ANTLR 4.9.1. It works with 2016.2-2020.3. It should work in other JetBrains IDEs.

- syntax highlighting
- syntax error checking
- semantic error checking
- navigation window
- goto-declaration
- find usages
- rename tokens
- rename rules
- comment grammar rule lines with meta-/ (1.7)
- save parse trees as svg/jpg/png; right click in parse tree view (1.9)
- grammar/comment folding (1.7)
- generates parser code; shortcut (ctrl-shift-G / meta-shift-G) but it's in Tools menu and popups.
- code completion for tokens, rule names;
- finds tokenVocab option for code gen if there is a tokenVocab option, don't warn about implicit tokens.
- handles separate parsers and lectures like TParser.g4 and TLexer.g4 (1.7)
- Parse tree nodes show the alternative number the parser chose to match that node. (1.7)
- has live grammar interpreter for grammar preview. Right click on rule and say "Test ANTLR Rule".
- view parse trees in hierarchy (sideways tree) view. (1.8)
- can view parse trees for input matched in more than one way

OK CANCEL APPLY

Terminal: Local x +

(venv) C:\Users\Amin\MAG_term_6\Compiler>pip ins
Requirement already satisfied: antlr4-python3-run

(venv) C:\Users\Amin\MAG_term_6\Compiler>python

(venv) C:\Users\Amin\MAG_term_6\Compiler>antlr4 -Dlanguage=Python3 CPP14.g4 -visitor

(venv) C:\Users\Amin\MAG_term_6\Compiler>java org.antlr.v4.Tool -Dlanguage=Python3 CPP14.g4 -visitor

(venv) C:\Users\Amin\MAG_term_6\Compiler>antlr4 -Dlanguage=Python3 CPP14.g4 -visitor

ANTLR Preview Tool Output Terminal Python Console 4: Run 4: Debug 4: TODO

Material Oceanic 15 chars 28:16 CRLF UTF-8 4 spaces Python 3.8 (Compiler)

File Edit View Navigate Code Refactor Run Tools VCS Window Help Compiler [C:\Users\Amin\MAG_term_6\Compiler] - ... \CPP14.g4 - PyCharm

Compiler > CPP14.g4 >

Project > Compiler C:\Users\Amin\MAG_term_6\Compiler

- cpp_test
- venv library root
- btree.cpp
- CPP14.g4
- CPP14.int
- CPP14.tok
- CPP14Lex
- CPP14Lex
- CPP14Lex
- CPP14List
- CPP14Par
- CPP14Visi
- generate_1

External Libra

Scratches and

Terminal: Local

Local History

(venv) C:\Us

Requirement

(venv) C:\Us

(venv) C:\Us

(venv) C:\Us

(venv) C:\Us

(venv) C:\Users\Amin\MAG_term_6\Compiler>antlr4 -Dlanguage=Python3 CPP14.g4 -visitor

ANTLR Preview Tool Output Terminal Python Console Run Debug TODO

Material Oceanic 15 chars 28:16 CRLF UTF-8 4 spaces Python 3.8 (Compiler)

```
1 ...
24 grammar CPP14;
25 /*Basic concepts*/
26
27
28 translationunit
29 : declarationseq? EOF
30 ;
31 /*Expressions*/
32
33
34 primaryexpression
35 : literal+
36 | This
37 | '(' expression ')'
38 | idexpression
39 | lambdaexpression
40 ;
41
42 idexpression
43 : unqualifiedid
```

Click On for more configuration

PC File Edit View Navigate Code Refactor Run Tools VCS Window Help compiler - CPP14.g4

compiler homeworks CPP14.g4

Project

- compiler D:\jst\term 6\compiler
 - ANTLR
 - homeworks
 - ANTLR.pptx
 - Assignment 1.pptx
 - CPP14.g4
 - CPP14.interp
 - CPP14.tokens
 - CPP14Lexer.interp
 - CPP14Lexer.py
 - CPP14Lexer.tokens
 - CPP14Listener.py
 - CPP14Parser.py
 - CPP14Visitor.py
 - hw1.py

جزوات فارسی

External Libraries

Scratches and Consoles

Configure ANTLR Tool 4.9.1 for CPP14.g4

- ☒ Auto-generate parsers upon save
- Output directory where all output is generated D:\jst\term 6\compiler\ANTLR
- Location of imported grammars
- Grammar file encoding; e.g., euc-jp
- Package/namespace for the generated code
- Language (e.g., Java, Python2, CSharp, ...) Python3
- Case transformation in the Preview window Leave as-is
- ☒ generate parse tree listener (default)
- ☒ generate parse tree visitor

OK Cancel

ANTLR Preview

CPP14.g4 start rule: translationunit

Input File

```
5 int main() {  
6     std::cout << "Hello World!";  
7     return 0;  
8 }
```

Parse tree Hierarchy Profiler

translationunit

- declarationseq:1 <EOF>
 - declaration:2
 - functiondefinition

Tool Output ANTLR Preview Run Problems Debug TODO Terminal Python Console

Event Log

27:1 CRLF UTF-8 4 spaces Python 3.6

Compiler > CPP14.g4 >

GENERATE_TREE

Project

C:\Users\Amin\MAG_term_6\Compiler

cpp_test

gen

venv library root

G+ btree.cpp

CPP14.g4

CPP14.interp

CPP14.tokens

CPP14Lexer.interp

CPP14Lexer.py

CPP14Lexer.tokens

CPP14Listener.py

CPP14Parser.py

CPP14Visitor.py

generate_tree.py

External Libraries

Scratches and Consoles

```
1 ...
24 grammar CPP14;
25 /*Basic concepts*/
26
27
28 translationunit
29 : declarationseq? EOF
30 ;
31 /*Expressions*/
32
33
34 primaryexpression
35 : literal+
36 | This
37 | '(' expression ')'
38 | idexpression
39 | lambdaexpression
```

1.Right click on the rule and click on test rule

2.Enter your sample cpp code here

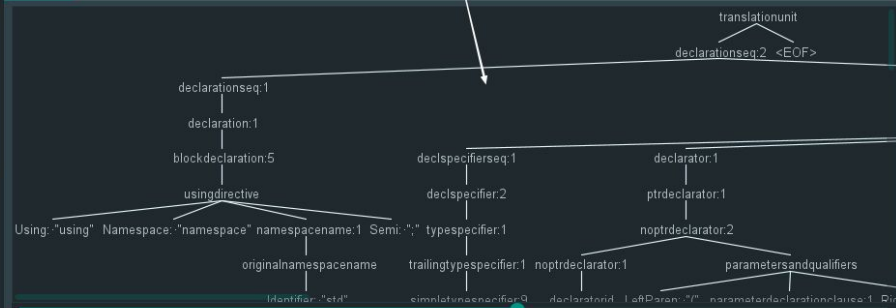
3.Then u will see the parse tree on the right side

ANTLR Preview

CPP14.g4 start rule: translationunit Input File

```
1 #include <iostream>
2 using namespace std;
3
4 int main() {
5     for (int i = 0; i <= 10; i = i + 2) {
6         cout << i << "\n";
7     }
8     return 0;
9 }
10
```

Parse tree Hierarchy Profiler



2 favorites

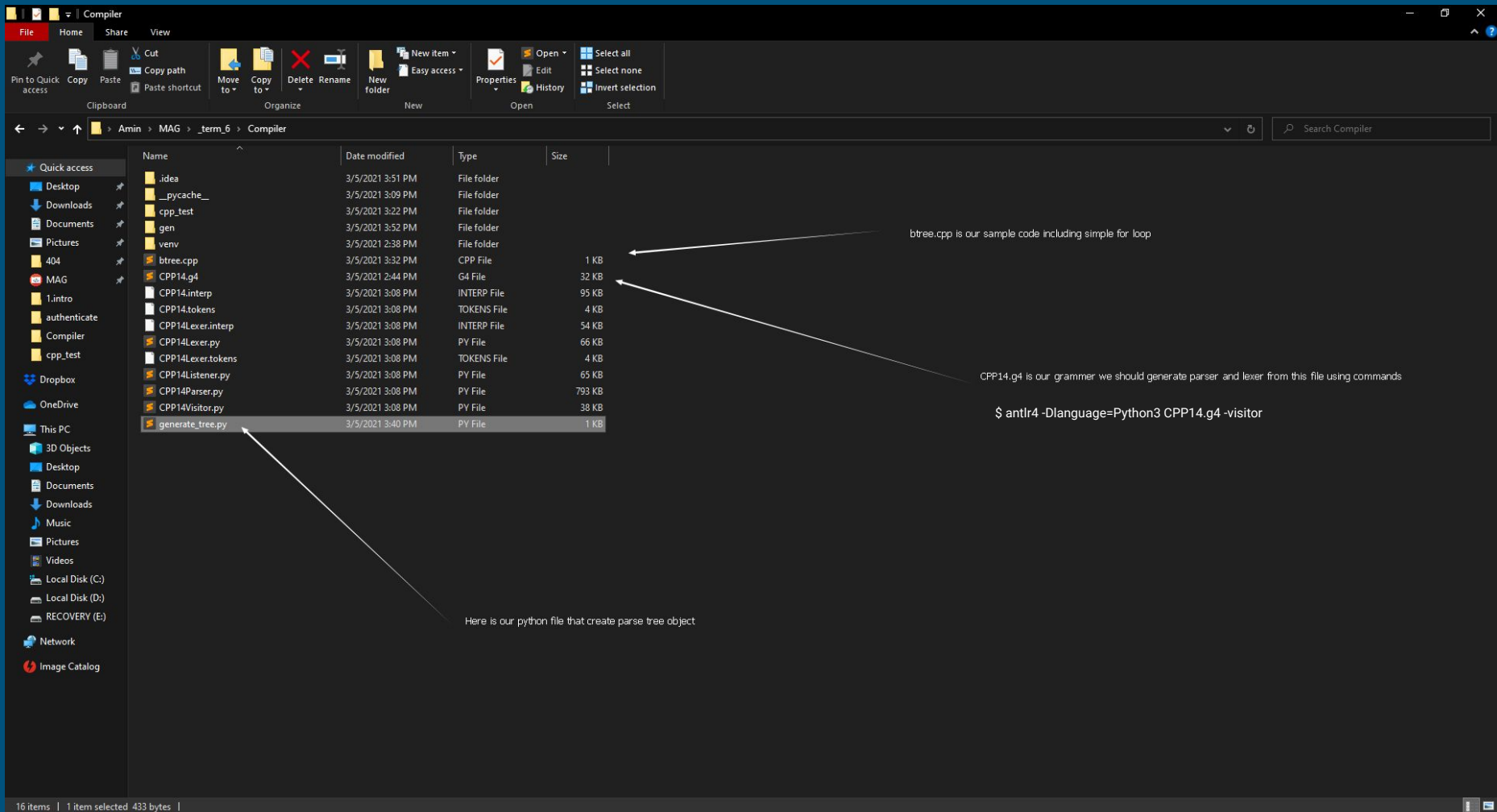
Structure

ANTLR Preview Tool Output Terminal Python Console 4: Run 5: Debug 6: TODO

parser for CPP14.g4 generated: to C:\Users\Amin\MAG_term_6\Compiler\gen (4 minutes ago)

Material Oceanic 28.4 CRLF UTF-8 4 spaces Python 3.8 (Compiler) Event Log

Generate Parse Tree in Python



Compiler > generate_tree.py >

GENERATE_TREE

Project generate_tree.py x G+ btree.cpp x CPP14.g4 x

Project C:\Users\Amin\MAG_term_6\Compiler

cpp_test

gen

venv library root

G+ btree.cpp

CPP14.g4

CPP14.interp

CPP14.tokens

CPP14Lexer.interp

CPP14Lexer.py

CPP14Lexer.tokens

CPP14Listener.py

CPP14Parser.py

CPP14Visitor.py

generate_tree.py

External Libraries

Scratches and Consoles

Import the cpp file here

We are able to create
parse tree here

```
1 import sys
2 from antlr4 import *
3 from CPP14Lexer import CPP14Lexer
4 from CPP14Parser import CPP14Parser
5
6
7 def main(argv):
8     input_stream = FileStream(argv[1])
9     lexer = CPP14Lexer(input_stream)
10    stream = CommonTokenStream(lexer)
11    parser = CPP14Parser(stream)
12    tree = parser.translationunit()
13
14
15 if __name__ == '__main__':
16     main(sys.argv)
17
```

Run/Debug Configurations

Python

generate_tree

Templates

1. set "btree.cpp" as parameter

Name: generate_tree

Share through

Configuration Logs

Script path: C:\Users\Amin\MAG_term_6\Compiler\generate_tree.py

Parameters: btree.cpp

Environment

Environment variables: PYTHONUNBUFFERED=1

Python interpreter: Project Default (Python 3.8 (Compiler)) C:\Users\Amin\MAG_term_6\G

Interpreter options:

Working directory: C:\Users\Amin\MAG_term_6\Compiler

☒ Add content roots to PYTHONPATH☒ Add source roots to PYTHONPATH

Execution

☐ Emulate terminal in output console☐ Run with Python Console☐ Redirect input from:

Before launch: Activate tool window

There are no tasks to run before launch

OK

File Edit View Navigate Code Refactor Run Tools VCS Window Help Compiler [C:\Users\Amin\MAG\term_6\Compiler] - ...generate_tree.py - PyCharm

Compiler > generate_tree.py

Project C:\Users\Amin\MAG\term_6\Compiler

- cpp_test
- gen
- venv library root
- btrees.cpp
- CPP14.g4
- CPP14.interp
- CPP14.tokens
- CPP14Lexer.interp
- CPP14Lexer.py
- CPP14Lexer.tokens
- CPP14Listener.py
- CPP14Parser.py

Debug: generate_tree.py

Debugger Console

Frames

- MainThread
- main_generate_tree.py:13
- <module>, generate_tree.py:17

Variables

```
> {} EMPTY = (ParserRuleContext) [None]
> children = (list 2) [<CPP14Parser.CPP14Parser.DeclarationseqContext object at 0x00002B5CECF27A0D>, <CPP14Parser.CPP14Parser.DeclarationContext object at 0x00002B5CECF86820>]
> {} 0 = (DeclarationseqContext) [122 400]
> {} 1 = (DeclarationContext) [1176 400]
> {} EMPTY = (ParserRuleContext) [None]
> children = (list 1) [<CPP14Parser.CPP14Parser.FunctiondefinitionContext object at 0x00002B5CECF85280>]
> {} 0 = (FunctiondefinitionContext) [1183 1176 400]
> {} EMPTY = (ParserRuleContext) [None]
> children = (list 3) [<CPP14Parser.CPP14Parser.DeclspecifierseqContext object at 0x00002B5CECF859A0>, <CPP14Parser.CPP14Parser.DeclaratorContext object at 0x00002B5CF1C5580>, <CPP14Parser.CPP14Parser.FunctionbodyContext object at 0x00002B5CECF859A0>]
> {} 0 = (DeclspecifierseqContext) [1898 1183 1176 400]
> {} 1 = (DeclaratorContext) [1901 1183 1176 400]
> {} 2 = (FunctionbodyContext) [1905 1183 1176 400]
len_ = (int) 3
```

exception = (NoneType) None

parser for CPP14.g4 generated: to C:\Users\Amin\MAG\term_6\Compiler\gen (18 minutes ago)

Material Oceanic 12:10 CRLF UTF-8 4 spaces Python 3.8 (Compiler)

def main(argv): argv: ['C:/Users/Amin/MAG/_term_6/Compiler/generate_tree.py', 'btrees.cpp']

input_stream = FileStream(argv[1]) input_stream: #include <iostream>\r\nusing namespace std;\r\n\r\n\r\nint main() {\r\n for (int i = 0; i <= 10; i

lexer = CPP14Lexer(input_stream) lexer: <CPP14Lexer.CPP14Lexer object at 0x000002B5CEA12C40>

stream = CommonTokenStream(lexer) stream: <antlr4.CommonTokenStream.CommonTokenStream object at 0x000002B5CED6DDC0>

parser = CPP14Parser(stream) parser: <CPP14Parser.CPP14Parser object at 0x000002B5CEC0C4C0>

tree = parser.translationunit() tree:

a = 32

if __name__ == '__main__':

main(argv)

main()

Debug the python app and consider tree variable

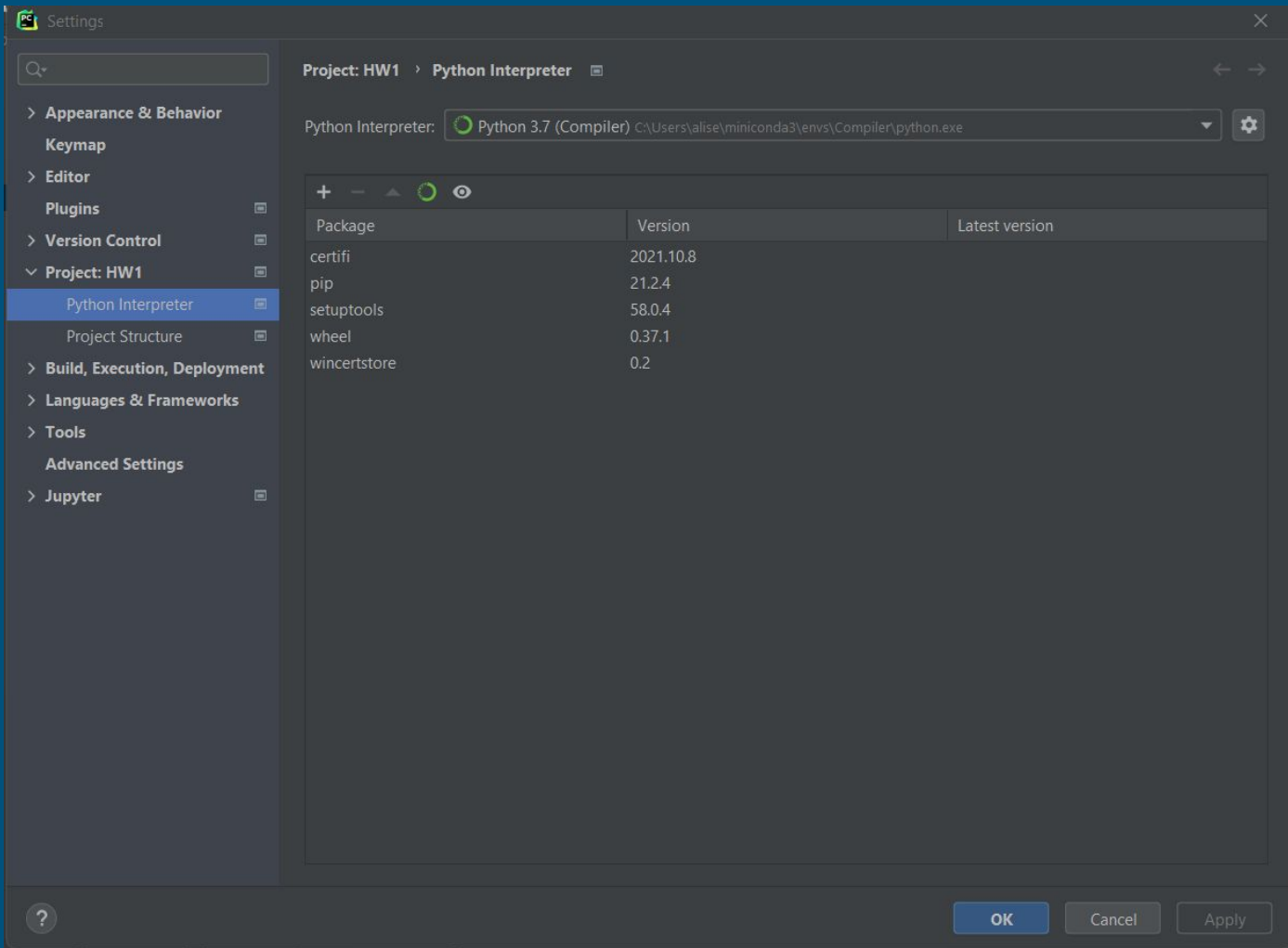
open tree object from Debug tools

As u see this tree is just like the previous one!

ANTLR4 Python Package

- ★ There is a python package called “antlr4-python3-runtime”
- ★ You can install it globally via:
 - `pip install antlr4-python3-runtime`

- ★ Or you can install it on PyCharm
 - Go to PyCharm Settings
 - Project: [Project_Name]
 - Python Interpreter
 - Install
 - Search for “antlr4-python3-runtime”
 - Install Package



Available Packages

antlr4-python3-runtime

antlr4-python3-runtime

Description

ANTLR 4.9.3 runtime for Python 3.7

Version

4.9.3

Author

Eric Verinaud, Terence Parr, Sam Harwell

<mailto:eric.verinaud@wanadoo.fr>

<http://www.antlr.org>

☐ Specify version

4.9.3

☐ Options

☐ Install to user's site packages directory (C:\Users\alise\AppData\Roaming\Python)

Install Package

Manage Repositories

Thank You!
