Antlr

--Antlr takes a grammar file as its input (rules for lexer and parser)

--Here is an example Grammar Hello.g4 (g4 is the extension Antlr expects)

```
// Define a grammar called Hello

grammar Hello;

r  : 'hello' ID ;          // match keyword hello followed by an identifier

ID : [a-z]+ ;              // match lower-case identifiers

WS : [ \t\r\n]+ -> skip ; // skip spaces, tabs, newlines
```

Antlr

-- Once you have downloaded Antlr and set up your environment variables properly (remember to add your current directory to your classpath) you can open a command prompt, navigate to the folder with Hello.g4 and type

```
C:\> Antlr Hello.g4
```

--Your output will be some java classes and some text documents

-- Then compile

```
C:\> javac Hello*.java
```

Antlr

--Antlr is all set up to parse your code

--I set up some commands using a tutorial to simply the run commands,

```
c:\> grun Hello r -tree
```

(r is the starting rule of the grammar, -tree is the output type)

--We could have passed a file but by default it will wait for user input.

```
c:> Hello World
^z
```

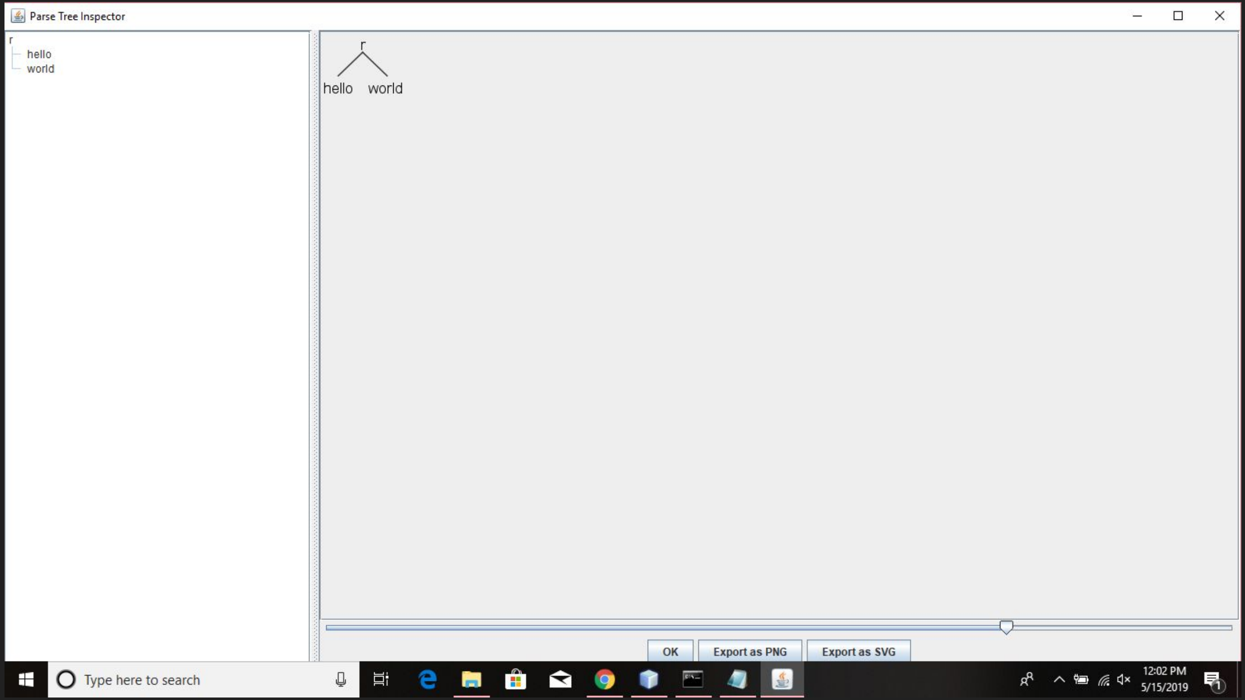
Antlr

--The output will look like this

```
(r hello world)
```

-- Antlr aslo has a visual representation

```
c:\> grun Hello r -gui
```


Parse Tree Inspector
r
hello
world
r
hello
world
OK
Export as PNG
Export as SVG
Type here to search
12:02 PM
5/15/2019