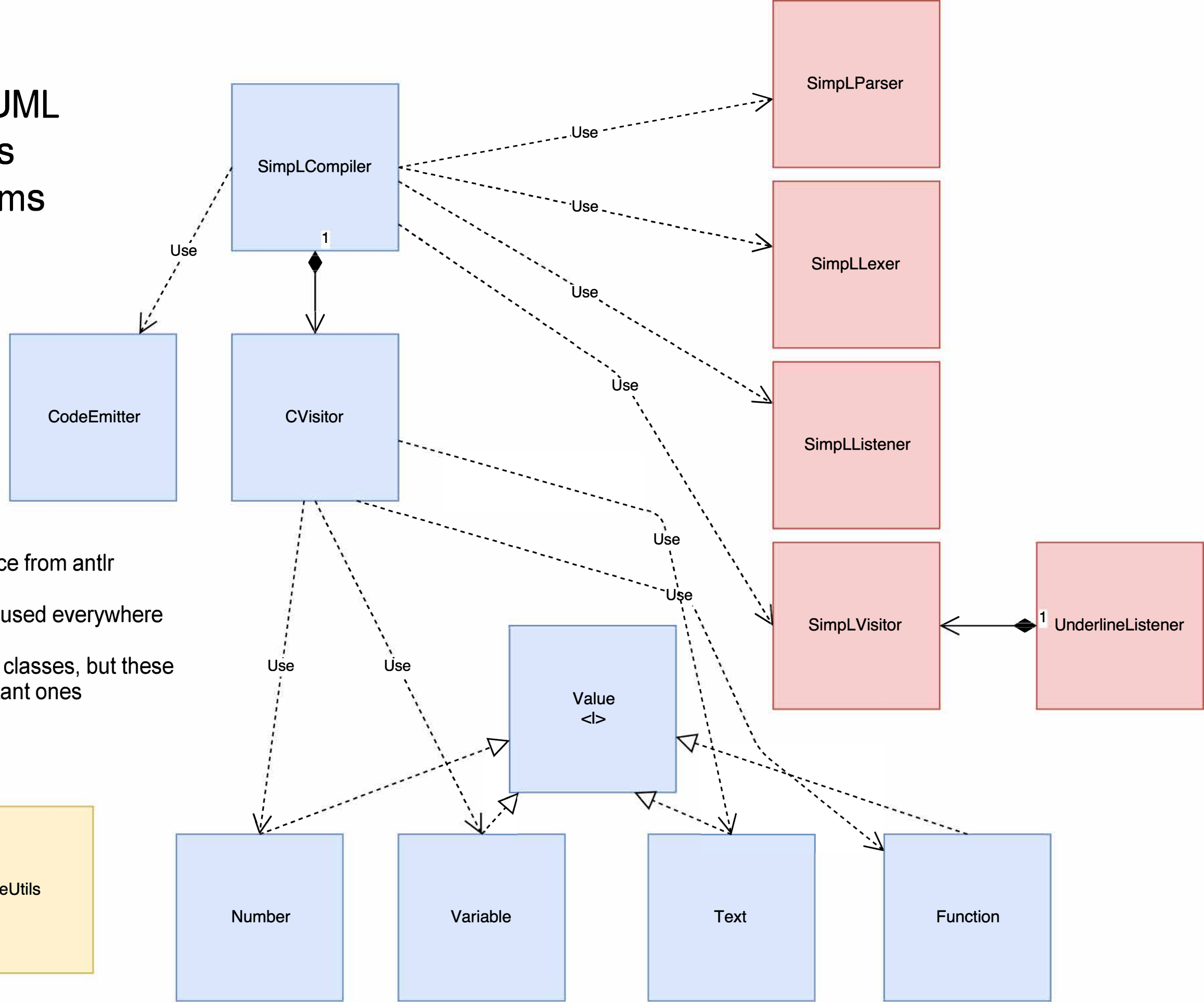


SimpL UML Class Diagrams



Red: generated source from antlr

Blue: core class

Yellow: Utils classes used everywhere

Note: there are other classes, but these were the most important ones

Simpl

Rules

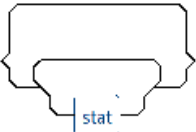
- program
- stat
- declaration
- assignment
- while_loop
- conditional
- func_def
- block
- expr
- func_call
- TYPE
- LITERAL
- NONE
- TEXT
- NUMBER
- BOOLEAN
- TEXT_VALUE
- NUMBER_VALUE
- BOOLEAN_VALUE
- SEPARATOR
- LPAREN
- RPAREN
- LCURL
- RCURL
- LSQUARE
- RSQUARE
- POW
- MUL
- DIV
- ADD
- SUB
- LT
- GT
- LTE
- GTE
- EQ
- NEQ
- NOT
- AND
- OR
- ASSIGN
- EOL
- NAME
- WHITESPACE
- LINE_COMMENT
- BLOCK_COMMENT
- QUOTE

programTop

Text notation:

program : stat* ;

Visual notation:

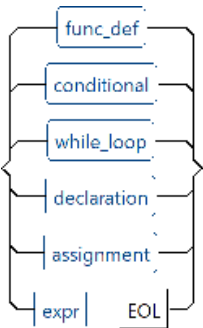


statTop

Text notation:

stat : func_def | conditional | while_loop | declaration | assignment | expr EOL ;

Visual notation:

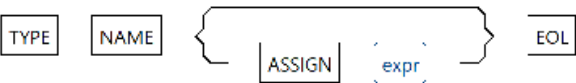


declarationTop

Text notation:

declaration : TYPE NAME (ASSIGN expr)? EOL ;

Visual notation:



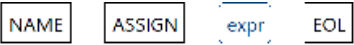
- DIGIT
- LETTER
- NEWLINE
- BACKSLASH

assignment Top

Text notation:

assignment : NAME ASSIGN expr EOL ;

Visual notation:



while_loop Top

Text notation:

while_loop : 'while' expr block ;

Visual notation:



conditional Top

Text notation:

conditional : ('if' expr block) ('elif' expr block)* ('else' block)? ;

Visual notation:

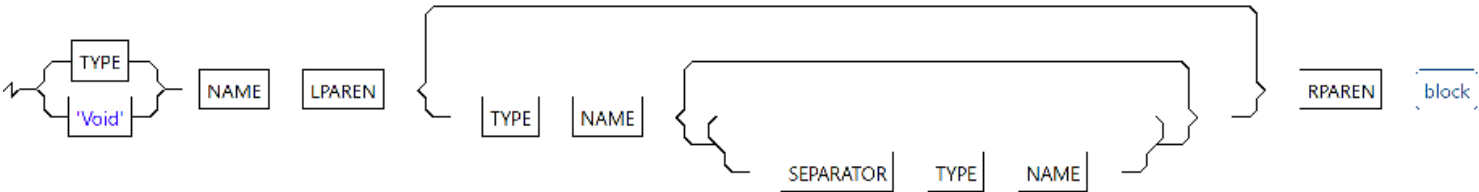


func_def Top

Text notation:

func_def : (TYPE | 'Void') NAME LPAREN (TYPE NAME (SEPARATOR TYPE NAME)*)? RPAREN block ;

Visual notation:

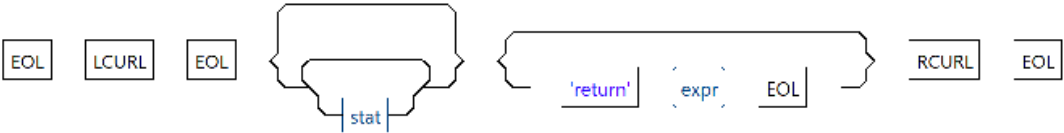


block Top

Text notation:

block : EOL LCURL EOL stat* ('return' expr EOL)? RCURL EOL ;

Visual notation:

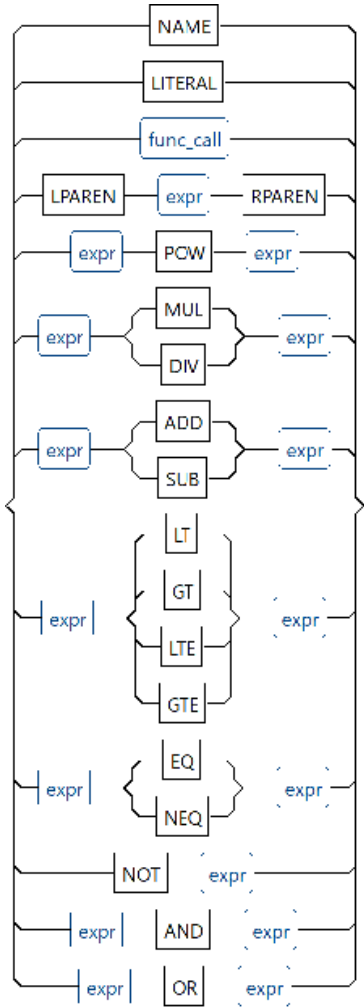


expr Top

Text notation:

expr : NAME | LITERAL | func_call | LPAREN expr RPAREN | expr POW expr | expr (MUL | DIV) expr | expr (ADD | SUB) expr | expr (LT | GT | LTE | GTE) expr | expr (EQ | NEQ) expr | NOT expr | expr AND expr | expr OR expr ;

Visual notation:

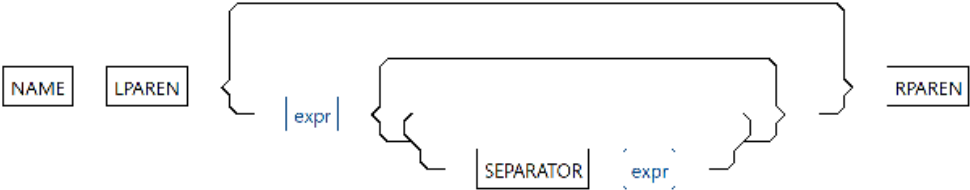


func_call [Top](#)

Text notation:

```
func_call : NAME LPAREN (expr (SEPARATOR expr)*)? RPAREN ;
```

Visual notation:

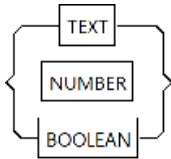


TYPE Top

Text notation:

TYPE : TEXT | NUMBER | BOOLEAN ;

Visual notation:

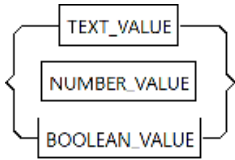


LITERAL Top

Text notation:

LITERAL : TEXT_VALUE | NUMBER_VALUE | BOOLEAN_VALUE ;

Visual notation:



NONE Top

Text notation:

NONE : 'None' ;

Visual notation:



TEXT Top

Text notation:

TEXT : 'Text' ;

Visual notation:



NUMBER

Text notation:

NUMBER : 'Number' ;

Visual notation:



BOOLEAN

Top

Text notation:

BOOLEAN : 'Boolean' ;

Visual notation:



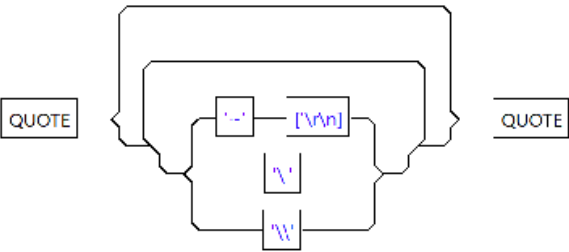
TEXT_VALUE

Top

Text notation:

TEXT_VALUE : QUOTE (~['\r\n' | '\\\'' | '\\\\']* QUOTE ;

Visual notation:



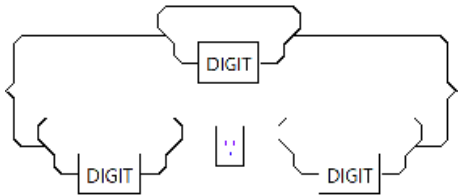
NUMBER_VALUE

Top

Text notation:

NUMBER_VALUE : DIGIT+ | DIGIT+ . DIGIT+ ;

Visual notation:

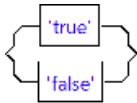


BOOLEAN_VALUE Top

Text notation:

```
BOOLEAN_VALUE : 'true' | 'false' ;
```

Visual notation:



SEPARATOR Top

Text notation:

```
SEPARATOR : ',' ;
```

Visual notation:



LPAREN Top

Text notation:

```
LPAREN : '(' ;
```

Visual notation:



RPAREN Top

Text notation:

```
RPAREN : ')' ;
```

Visual notation:



LCURL

Top

Text notation:

LCURL : ' { ' ;

Visual notation:



RCURL

Top

Text notation:

RCURL : ' } ' ;

Visual notation:



LSQUARE

Top

Text notation:

LSQUARE : ' [' ;

Visual notation:



RSQUARE

Top

Text notation:

RSQUARE : '] ' ;

Visual notation:



POW

Top

Text notation:

POW : ' ^ ' ;

Visual notation:



MUL Top

Text notation:

MUL : ' * ' ;

Visual notation:



DIV Top

Text notation:

DIV : ' / ' ;

Visual notation:



ADD Top

Text notation:

ADD : ' + ' ;

Visual notation:



SUB Top

Text notation:

SUB : ' - ' ;

Visual notation:



LT Top

Text notation:

LT : '<' ;

Visual notation:



GT Top

Text notation:

GT : '>' ;

Visual notation:



LTE Top

Text notation:

LTE : '<=' ;

Visual notation:



GTE Top

Text notation:

GTE : '>=' ;

Visual notation:



EQ Top

Text notation:

EQ : '==' ;

Visual notation:



NEQ Top

Text notation:

```
NEQ : '!=' ;
```

Visual notation:

NOT Top

Text notation:

```
NOT : 'not' ;
```

Visual notation:

AND Top

Text notation:

```
AND : 'and' ;
```

Visual notation:

OR Top

Text notation:

```
OR : 'or' ;
```

Visual notation:

ASSIGN Top

Text notation:

```
ASSIGN : '=' ;
```

Visual notation:

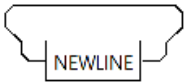
EOL

Top

Text notation:

EOL : NEWLINE+ ;

Visual notation:



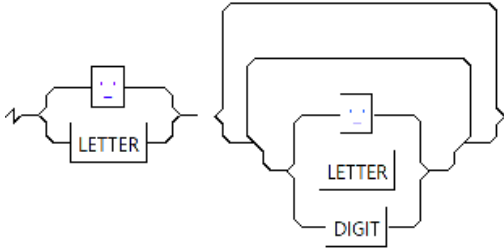
NAME

Top

Text notation:

NAME : (' _ ' | LETTER) (' _ ' | LETTER | DIGIT) * ;

Visual notation:



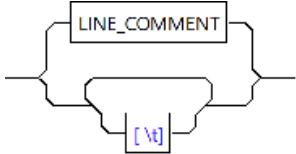
WHITESPACE

Top

Text notation:

WHITESPACE : (LINE_COMMENT | [\t] +) -> skip ;

Visual notation:



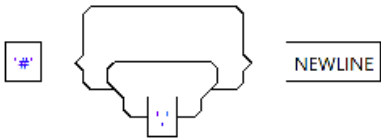
LINE_COMMENT

Top

Text notation:

LINE_COMMENT : ('#' .*? NEWLINE) -> skip ;

Visual notation:

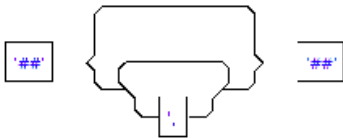


BLOCK_COMMENT [Top](#)

Text notation:

```
BLOCK_COMMENT : ( '##' .*? '##' ) -> skip ;
```

Visual notation:



QUOTE [Top](#)

Text notation:

```
QUOTE : '\\' ;
```

Visual notation:



DIGIT [Top](#)

Text notation:

```
DIGIT : '0'..'9' ;
```

Visual notation:

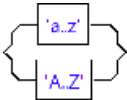


LETTER [Top](#)

Text notation:

```
LETTER : 'a'..'z' | 'A'..'Z' ;
```

Visual notation:

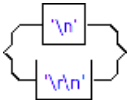


NEWLINE Top

Text notation:

```
NEWLINE : '\n' | '\r\n' ;
```

Visual notation:



BACKSLASH Top

Text notation:

```
BACKSLASH : '\\ ' ;
```

Visual notation:

