FlowScript Documentation

By Kenny Zhang (ID: 48618419)

Rules for Lexical Parsing

The code is tokenized into several types of tokens, with the tokenizer splitting the string using this regex: “\s*(->|\w+\b|\W|\n)\s*”. This will basically make tokens that are words, single character symbols, and the -> operator. The tokens are then categorized using the regex key below:

Token Type	Regex
SEMICOLON	;
ARROW	->
LBRACE	\{
RBRACE	\}
LBRACKET	\[
RBRACKET	\]
EQUALS	=
QUOTE	“
WORD	\w+
INVALID	.*

The gist of the tokens is that a token will qualify as a word if they are made up of the characters a-z, A-Z, 0-9, and/or underscore. A token qualifies as invalid if it fails all previous token capturing groups and will cause an error and end the parsing early. The rest of the tokens are self explanatory. These are then fed to the syntax parser for processing.

Rules for Syntactic Parsing

The syntactic parsing will create the dependency map and detect syntax errors. The ends of statements are defined by “ ; “, “ { ‘, and “ } “. This means that whitespace does not matter for the syntactic parsing of flowscript.

Graph creation must follow this format:

<WORD> <WORD> <LBRACKET> [Node Assignment Statements] <RBRACKET>

Ex: digraph flowscript { }

Node assignments can use the following formats:

<WORD> <SEMICOLON>

Ex: A;

<WORD> <ARROW> <WORD> <SEMICOLON>

Ex: A -> B;

<WORD> <ARROW> <WORD> <LBRACKET> <WORD> <EQUALS> <QUOTE> <WORD> <QUOTE> <SEMICOLON>

Ex: A -> B [condition = “status”];

Note: The third word in the third statement can use any node attribute. Currently the only valid attribute is “condition” but that can be easily changed by adding to the graphTypes vector.

Error Codes and What They Mean

Only error code one is lexical. The rest are syntactical.

Code	Description:
1	Invalid token processed. The only valid tokens are ' { ', ' } ', ' [', '] ', '= ', ' “ ‘, and words. Only words can only contain characters a-z, A-Z, 0-9, and/or underscores
2	Duplicate ‘{‘ used. Nested brackets are not allowed.
3	Right brace has no matching left brace.
4	Semicolons cannot be used outside of braces.
5	Left brace has no matching right brace.
6	Too many/too few arguments for graph definition.
7	Graph type is not a word.
8	Graph name is invalid. Names can only contain characters a-z, A-Z, 0-9, and/or underscores.
9	Node definition does not include a valid name for the source node.

10	Node definition does not include a valid arrow operator.
11	Node definition does not include a valid name for the end node.
12	Node definition has too many tokens.
13	Node attribute is not a word.
14	Node attribute is invalid.
15	Invalid attribute operator.
16	Invalid attribute assignment value. Value must be enclosed with quotation marks.
17	Attribute assignment value is not a word.
18	Attribute assignment value is missing end quotes. Value must be fully enclosed with quotation marks.
19	Bracket pair is not matched.
20	Node assignment has too many tokens.
21	Graph type does not exist.
22	Statement does not end properly. Use brackets, braces, or semicolons to end statements.