LANGUAGE SPECIFICATION

for

BNF Meta Language (BNF-ML)

Version 1.0 approved

Prepared by Andre Tabourian

June 4, 2021

Contents

Revision History

Name	Date	Reason For Changes	Version
21	22	23	24
31	32	33	34

1 ABOUT

BNFML is a programming language based on the Backus Naur Form (BNF) used to lex and parse the BNF language itself.

2 Language Specification

2.1 Grammar

The following, is the BNF specification of BNFML.

⟨*BNF*⟩ ::= ⟨*List of Bindings*⟩ | ⟨*empty*⟩

⟨*List of Bindings*⟩ ::= ⟨*List of Bindings*⟩ ⟨*Binding*⟩ | ⟨*Binding*⟩

⟨*Binding*⟩ ::= ⟨*NonTerminal*⟩ ‘`::=`’ ⟨*OrExpression*⟩

⟨*OrExpression*⟩ ::= ⟨*OrExpression*⟩ ‘`|`’ ⟨*AndExpression*⟩ | ⟨*AndExpression*⟩

⟨*AndExpression*⟩ ::= ⟨*AndExpression*⟩ ⟨*Expression*⟩ | ⟨*Expression*⟩

⟨*Expression*⟩ ::= ⟨*NonTerminal*⟩ | ⟨*Terminal*⟩

⟨*NonTerminal*⟩ ::= ‘`<.+>`’

⟨*Terminal*⟩ ::= ‘`.+`’

⟨*empty*⟩ ::= ‘’

Please note that BNFML is self-descriptive, as such the only specification needed to describe BNFML is BNFML itself. What this means is that the specification you see above is BNFML, and it describes itself.

2.2 Explanation of the Spec

The following is an explanation of the above specification:

- A *BNF* consists of a list of *Bindings* or nothing (‘’).
- A *Binding* consists of a *NonTerminal* followed by *::=* and an *OrExpression*
- An *OrExpression* consists of an *AndExpression* or another *OrExpression* followed by a pipe | and an *AndExpression*
- An *AndExpression* consists of an *Expression* or another *AndExpression* followed by an *Expression*

- An *Expression* is nothing more than a *NonTerminal* or a *Terminal*
- A *NonTerminal* is denoted by angle brackets <> containing any character.
- A *Terminal* is denoted by single quotes containing any regular expressions. Note that that to insert single quotes in a terminal, they must be escaped with \

2.3 Whitespace

This section deals with whitespace characters:

New Lines:

- Two Bindings are separated by at least one new line.
- New lines only separate bindings and nothing else.

Spaces and Tabs:

- Whitespace does not matter except within a Terminal.
- ‘1’ ‘2’ and ‘1’‘2’ are interpreted in the same way.
- $< a >$ ::= $< b >$ and $< a >$::$< b >$ are interpreted in the same way.

3 Implementation

3.1 Lexer Implementation

3.2 Parser Implementation