Grammar For Project – Super Simple Language

I use EBNF for my grammar

It is unambiguous since there is only one parsing tree for one string.

This can prevent unknown syntax error

Goal = MainClass, { ClassDeclaration }, EOF;

MainClass = "class", Identifier, "{", "function", "static", "void", "main", "(", "String", "[", "]", Identifier, ")", "{", Statement, "}", "}";

ClassDeclaration = "class", Identifier, ["extends", Identifier], "{", { VarDeclaration }, { MethodDeclaration } "}";

VarDeclaration = Type, Identifier, ";";

MethodDeclaration = "function", Type, Identifier, "(", [Type, Identifier, { ",", Type, Identifier },], ")", "{", { VarDeclaration }, { Statement }, "return", Expression, ";", "}";

Type = "int", "[", "]"

| "boolean"

| "int"

| "double"

| "string"

| Identifier

;

Statement = "{", { Statement }, "}"

| "if", "(", Expression, ")", Statement, "else", Statement

| "while", "(", Expression, ")", Statement

| "System.out.println", "(" , Expression, ")", ";"

| Identifier, "=", Expression, ";"

| Identifier, "[", Expression, "]", "=", Expression, ";"

;

Expression = Expression , ("&&" | "<" | "+" | "-" | "*") , Expression

| Expression, "[", Expression, "]"

| Expression, ".", "length"

| Expression, ".", Identifier, "(", [Expression { ",", Expression }], ")"

| IntegerLiteral

| DoubleLiteral

| "true"

| "false"

| Identifier

| "this"

| "new", "int", "[", Expression, "]"

| "new", Identifier ,"(" ,")"

| "!", Expression

| "(", Expression, ")"

;

Identifier is one or more letters, digits, and underscores, starting with a letter

IntegerLiteral is one or more decimal digits

DoubleLiteral is one or more decimal digits contains "."

EOF is a distinguished token returned by the scanner at end-of-file

EBNF

ISO/IEC 14977: 1996(E)

Terminal symbols are quoted

[and] indicate optional symbols

{ and } indicate repetition

(and) group items together; the other brackets do too

Super simple Characterset

A Super simple program is a text file conisisting of US-ASCII characters.

Comments

For example,

%One commment%

No test for primitive equality, or equality of references.

You may assume 5 or fewer formal arguments in methods.

You may assume no methods are overloaded.

You may assume no methods are overriden.

Allow fields and methods to be inherited.