

May 7, 2014

Jani Viherväs
jani.vihervas@cs.helsinki.fi

MINI-JAVA COMPILER

582648 CODE GENERATION

http://www.cs.helsinki.fi/u/vihavain/k12/compiler_project/project/compiler_project_2012.html

1 Overview

2 Grammar

<i>< program ></i>	→ <i>< main class ></i> <i>< class declaration ></i> *
<i>< main class ></i>	→ class <i>< new identifier ></i> { public static void main() { <i>< statement ></i> * }}
<i>< class declaration ></i>	→ class <i>< new identifier ></i> [extends <i>< identifier ></i>] { <i>< declaration ></i> * }
<i>< declaration ></i>	→ <i>< variable declaration ></i> <i>< method declaration ></i>
<i>< method declaration ></i>	→ public <i>< type ></i> <i>< new identifier ></i> ([<i>< formals ></i>]) { <i>< statement ></i> * }
<i>< variable declaration ></i>	→ <i>< type ></i> <i>< new identifier ></i> <i>< variable assignment ></i> ;
<i>< variable assignment ></i>	→ ϵ = <i>< expr ></i>
<i>< formals ></i>	→ <i>< type ></i> <i>< new identifier ></i> (, <i>< type ></i> <i>< new identifier ></i>) *
<i>< type ></i>	→ <i>< simple type ></i> <i>< array type ></i>
<i>< simple type ></i>	→ int boolean void <i>< type identifier ></i>
<i>< array type ></i>	→ ϵ []
<i>< type identifier ></i>	→ <i>< identifier ></i>
<i>< statement ></i>	→ assert (<i>< expr ></i>) <i>< local variable declaration ></i> { <i>< statement ></i> * } if (<i>< expr ></i>) <i>< statement ></i> <i>< else ></i> while (<i>< expr ></i>) <i>< statement ></i> System.out.println (<i>< expr ></i>); <i>< identifier ></i> = <i>< expr ></i> ; return <i>< expr ></i> ; <i>< method invocation ></i> ; <i>< else ></i> → ϵ else <i>< statement ></i>
<i>< local variable declaration ></i>	→ <i>< variable declaration ></i>
<i>< method invocation ></i>	→ <i>< expr ></i> . <i>< identifier ></i> ([<i>< expr ></i> (, <i>< expr ></i>) *])
<i>< expr ></i>	→ <i>< expr1 ></i> <i>< expr2 ></i>
<i>< expr1 ></i>	→ new <i>< new ></i> ! <i>< expr ></i> (<i>< expr ></i>) <i>< identifier ></i> <i>< integer literal ></i> this true false <i>< method invocation ></i>
<i>< expr2 ></i>	→ ϵ [<i>< expr ></i>] .length <i>< binary operator ></i> <i>< expr ></i>
<i>< new ></i>	→ <i>< simple type ></i> [<i>< expr ></i>] <i>< type identifier ></i> ()
<i>< binary operator ></i>	→ && < > == + - * / %