MiniJava Sprachbericht (v3)*†‡

Matthias Braun Jürgen Graf

29. April 2009

1 Einleitung

MiniJava ist eine Untermenge der Programmiersprache Java. Daher können Programme in MiniJava von jedem Java Übersetzer zu Bytecode übersetzt werden. Die Sprache enthält viele für den Übersetzerbau interessante Konzepte wie rekursive Methodenaufrufe. Trotzdem erlaubt ihre Kompaktheit eine Implementierung im Rahmen eines universitären Praktikums. In MiniJava wird auf viele Eigenschaften von Java, die das Laufzeit System und die Übersetzung unnötig erschweren, wie z.B. Zusnahmen und Mehrfädigkeit, verzichtet.

MiniJava ist, wie Java, eine Objektorientiert Sprache. Verfügt jedoch nur über ein Minimum an Anweisungen und Ausdrücken und benötigt nur sehr einfaches Laufzeitsystem.

2 Eigenschaften

2.1 Typsystem

MiniJava kennt den Basistyp **int** für Ganzzahlen und einen boolschen Typ für Wahrheitswerte. Der boolsche Typ tritt nur intern bei Ergebnissen von Vergleichsoperationen auf, es können keine Felder oder Variablen mit diesem Typ erzeugt werden. Benutzerdefinierte Typen sind Klassen und Methodentypen. Klassen enthalten Felder und Methoden. Methoden besitzen einen Methodentyp der die Anzahl und Typen der Parameter sowie den Typ des Rückgabewerts falls die Methode Werte zurückgibt. Methoden mit gleichem Bezeichner aber verschiedenen Parameter/Ergebnistypen sind nicht erlaubt. Beim Aufruf von Methoden müssen die Typen der Argumente denen der Parameter entsprechen.

*Die Sprache wurde um WhileStatement und BooleanExpression erweitert.
†Methodenaufrufe sind nun mit mehreren Parametern erlaubt (und auch ohne Parameter).
‡CompareExpression wurde erweitert und MultiplicativeExpression eingeführt.

2.2 Anweisungen und Ausdrücke

2.3 Die Hautpmethode

2.4 Laufzeitsystem

MiniJava besitzt ein minimalistisches Laufzeitsystem ohne große Standardbibliotheken. Die Ausführung eines Programms beginnt an der immer vorhandenen Hauptmethode. Es gibt keine automatische Speicherverwaltung, d.h. MiniJava Programme können zwar Speicher allozieren diesen aber nicht mehr zur weiteren Benutzung freigeben.

3 Wortschatz

MiniJava besteht aus den folgenden Wörtern:

- **Leerzeichen**: Dazu Zählen Leerzeichen, Zeilenumbruch, Wagenrücklauf und Tabulator.
- **Kommentar**: Die Zeichenkette `/*` gefolgt von beliebigen Zeichen bis zu einem abschliessenden `*/`. Kommentare können nicht geschachtelt werden, weitere `/*` innerhalb eines Kommentars werden ignoriert; [1] beim ersten `*/` ist der Kommentar stets zu Ende.
- **Zeilenkommentar**: Die Zeichenkette `//` gefolgt von beliebigen Zeichen bis zum nächsten Zeilenumbruch.
- **IDENT**: Ein Bezeichner, er beginnt mit einem Buchstaben oder Unterstrich gefolgt von beliebig vielen Buchstaben, Unterstrichen oder Zahlen in beliebiger Reihenfolge. Schlüsselwörter sind keine **IDENT**s.
- **INTEGER_LITERAL**: Eine Ziffernfolge.
- Schlüsselwörter und Operatoren: Alle anderen **fettgedruckten** Wörter in der Grammatikspezifikation sind Schlüsselwörter oder Operatoren.

Kommentare, Zeilenkommentare und Leerzeichen haben keine Bedeutung. Ihre Funktion ist die Trennung von Wörtern, Dokumentation des Quelltexts und Sie sorgen für ästhetische Programme. Sie werden für syntaktische Belange ignoriert.

4 Syntax

$$\begin{aligned} \textit{TranslationUnit} &\rightarrow \textit{ClassDeclaration}^* \\ \textit{ClassDeclaration} &\rightarrow \textbf{public}?\ \textbf{class}\ \textbf{IDENT}\ \{\ \textit{ClassMember}^*\ \} \end{aligned}$$

[1] Die eleganteste Lösung ist bei `/*` innerhalb des Kommentars eine Warnung auszugeben

$$
\begin{aligned}
\textit{ClassMember} &\rightarrow \textit{Field} \mid \textit{Method} \mid \textit{MainMethod} \\
\textit{Field} &\rightarrow \textbf{public}\ \textit{Type}\ \textbf{IDENT}\ ; \\
\textit{MainMethod} &\rightarrow \textbf{public static void main}\ (\ \textbf{String}\ [\ ]\ \textbf{IDENT}\ )\ \textit{CompoundStatement} \\
\textit{Method} &\rightarrow \textbf{public}\ \textit{Type}\ \textbf{IDENT}\ (\ \textit{Parameters?}\ )\ \textit{MethodBody} \\
\textit{Parameters} &\rightarrow \textit{Parameter} \mid \textit{Parameter}\ ,\ \textit{Parameters} \\
\textit{Parameter} &\rightarrow \textit{Type}\ \textbf{IDENT} \\
\textit{Type} &\rightarrow \textbf{boolean} \mid \textbf{int} \mid \textbf{void} \mid \textbf{IDENT} \\
\textit{MethodBody} &\rightarrow \{\ \textit{LocalVarDeclaration}^*\ \textit{Statement}^*\ \} \\
\textit{LocalVarDeclaration} &\rightarrow \textit{Type}\ \textbf{IDENT}\ ; \\
\textit{Statement} &\rightarrow \textit{CompoundStatement} \\
&\quad \mid \textit{IfStatement} \\
&\quad \mid \textit{PrintStatement} \\
&\quad \mid \textit{ExpressionStatement} \\
&\quad \mid \textit{WhileStatement} \\
&\quad \mid \textit{ReturnStatement} \\
\textit{CompoundStatement} &\rightarrow \{\ \textit{Statement}^*\ \} \\
\textit{WhileStatement} &\rightarrow \textbf{while}\ (\ \textit{Expression}\ )\ \textit{CompoundStatement} \\
\textit{IfStatement} &\rightarrow \textbf{if}\ (\ \textit{Expression}\ )\ \textit{Statement}\ \textbf{else}\ \textit{Statement} \\
\textit{PrintStatement} &\rightarrow \textbf{System}\ .\ \textbf{out}\ .\ \textbf{println}\ (\ \textit{Expression}\ )\ ; \\
\textit{ExpressionStatement} &\rightarrow \textit{Expression}\ ; \\
\textit{ReturnStatement} &\rightarrow \textbf{return}\ \textit{Expression?}\ ; \\
\textit{Expression} &\rightarrow \textit{AssignmentExpression} \\
\textit{AssignmentExpression} &\rightarrow \textit{BooleanExpression}(= \textit{AssignmentExpression})? \\
\textit{BooleanExpression} &\rightarrow \textit{CompareExpression}((\&\& \mid ||)\ \textit{BooleanExpression})? \\
&\quad \mid !\ \textit{BooleanExpression} \\
\textit{CompareExpression} &\rightarrow \textit{AdditiveExpression}((< \mid <= \mid > \mid >= \mid == \mid !=)\ \textit{AdditiveExpression})^* \\
\textit{AdditiveExpression} &\rightarrow \textit{MultiplicativeExpression}((+ \mid -)\ \textit{MultiplicativeExpression})^* \\
\textit{MultiplicativeExpression} &\rightarrow \textit{PostfixExpression}(* \ \textit{PostfixExpression})^* \\
\textit{PostfixExpression} &\rightarrow \textit{PrimaryExpression} \\
&\quad \mid \textit{CallExpression} \\
&\quad \mid \textit{FieldAccessExpression} \\
\textit{CallExpression} &\rightarrow \textit{PostfixExpression}\ (\ \textit{ExpressionList?}\ ) \\
\textit{ExpressionList} &\rightarrow \textit{Expression}(,\ \textit{Expression})^* \\
\textit{FieldAccessExpression} &\rightarrow \textit{PostfixExpression}\ .\ \textbf{IDENT} \\
\textit{PrimaryExpression} &\rightarrow \textbf{true} \\
&\quad \mid \textbf{false}
\end{aligned}
$$

| **INTEGER_LITERAL**
| **IDENT**
| **this**
| (*Expression*)
| *NewExpression*

NewExpression → **new IDENT** ( )

5 Semantik

Die Semantik von MiniJava entspricht der Semantik der Sprache Java. Diese ist in [GJSB00] beschrieben.

Literatur

[GJSB00] James Gosling, Bill Joy, Guy Steele, and Gilad Bracha. *Java Language Specification, Second Edition: The Java Series*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2000.