# IIC2323 — Construcción de Compiladores
# Especificación para un subconjunto de Sather 1.1

Primer semestre 2014

## Índice



## 1. Notación

El lenguaje es especificado utilizando gramáticas libres de contexto, donde al lado derecho de las producciones se utilizan expresiones regulares con la siguiente notación:

1. $[\varphi]$ es equivalente a $\varphi + \varepsilon$, es decir, que $\varphi$ es opcional.
2. $\{\varphi\}$ corresponde a $\varphi^*$, es decir que $\varphi$ se puede repetir una cantidad arbitraria de veces.
3. $\varphi|\psi$ es equivalente a $\varphi + \psi$, es decir que se escoge o bien $\varphi$ o $\psi$.

Por ejemplo, la gramática:

$$\begin{aligned} identifier &\rightarrow [\$]letter\{letter|digit\} \\ letter &\rightarrow a|b|c \\ digit &\rightarrow 0|1 \end{aligned}$$

indica que un identificador puede comenzar opcionalmente con \$, luego sigue con una letra, y termina con una cantidad arbitraria de letras y dígitos. Algunos identificadores válidos según esta son: \$a00b10a, b, y \$c0, mientras que 10, y a\$ no lo son.

# 2. Estructura léxica

## 2.1. Identificadores

Los identificadores dan nombre a clases, métodos, atributos y variables.

$$
\begin{aligned}
identifier &\rightarrow letter\{letter|decimal\_digit|\_\} \\
uppercase\_identifier &\rightarrow uppercase\_letter\{uppercase\_letter|decimal\_digit|\_\} \\
iter\_name &\rightarrow identifier\textbf{!} \\
letter &\rightarrow lowercase\_letter|uppercase\_letter \\
lowercase\_letter &\rightarrow \textbf{a}|\textbf{b}|\textbf{c}|\textbf{d}|\textbf{e}|\textbf{f}|\textbf{g}|\textbf{h}|\textbf{i}|\textbf{j}|\textbf{k}|\textbf{l}|\textbf{m}|\textbf{n}|\textbf{o}|\textbf{p}|\textbf{q}|\textbf{r}|\textbf{s}|\textbf{t}|\textbf{u}|\textbf{v}|\textbf{w}|\textbf{x}|\textbf{y}|\textbf{z} \\
uppercase\_letter &\rightarrow \textbf{A}|\textbf{B}|\textbf{C}|\textbf{D}|\textbf{E}|\textbf{F}|\textbf{G}|\textbf{H}|\textbf{I}|\textbf{J}|\textbf{K}|\textbf{L}|\textbf{M}|\textbf{N}|\textbf{O}|\textbf{P}|\textbf{Q}|\textbf{R}|\textbf{S}|\textbf{T}|\textbf{U}|\textbf{V}|\textbf{W}|\textbf{X}|\textbf{Y}|\textbf{Z} \\
decimal\_digit &\rightarrow \textbf{0}|\textbf{1}|\textbf{2}|\textbf{3}|\textbf{4}|\textbf{5}|\textbf{6}|\textbf{7}|\textbf{8}|\textbf{9}
\end{aligned}
$$

## 2.2. Keywords

Los keywords identifican elementos sintácticos fundamentales, y no pueden ser utilizados como identificadores.

$$
\begin{aligned}
keyword \rightarrow\ & \textbf{and}|\textbf{attr}|\textbf{break!}|\textbf{class}|\textbf{else}|\textbf{elsif}|\textbf{end}|\textbf{false}|\textbf{if}|\textbf{include}|\textbf{is}|\textbf{loop}|\textbf{new}|\textbf{or}|\textbf{once} \\
& \textbf{private}|\textbf{quit}|\textbf{return}|\textbf{SAME}|\textbf{self}|\textbf{then}|\textbf{true}|\textbf{typecase}|\textbf{void}|\textbf{when}|\textbf{while!}|\textbf{yield}
\end{aligned}
$$

## 2.3. Símbolos especiales

Otros símbolos utilizados:

$$special\_symbol \rightarrow (\ |\ )\ |\{|\}|,|.|;|:|+|-|*|/|\%|<|<=|=|/=|>|>=|\tilde{}|:=|::|\#$$

## 2.4. Expresiones literales

Valores literales para tipos básicos.

$$
\begin{aligned}
bool\_literal\_expression &\rightarrow\ \textbf{true}|\ \textbf{false} \\
char\_literal\_expression &\rightarrow' (ISO\_character|\backslash escape\_seq)' \\
escape\_seq &\rightarrow a|b|f|n|r|t|v|'|"|octal\_digit\{octal\_digit\}|\backslash \\
str\_literal\_expression &\rightarrow "\{(ISO\_character|\backslash escape\_seq)\}"\ \{"\{(ISO\_character|\backslash escape\_seq)\}"\} \\
int\_literal\_expression &\rightarrow [\textbf{-}]decimal\_digit\{decimal\_digit\}
\end{aligned}
$$

## 2.5. Comentarios

Los comentarios comienzan con `--` y continúan hasta el siguiente salto de línea.

## 3. Estructura sintáctica

$$
\begin{aligned}
\mathit{source\_file} &\rightarrow [\mathit{class}]\{;[\mathit{class}]\}\\
\mathit{type\_specifier} &\rightarrow \mathit{uppercase\_identifier} \mid \textbf{SAME}\\
\mathit{routine\_argument} &\rightarrow \mathit{identifier\_list}\ \textbf{:}\ \mathit{type\_specifier}\\
\mathit{identifier\_list} &\rightarrow \mathit{identifier}\{,\mathit{identifier}\}\\
\mathit{subtyping\_clause} &\rightarrow < \mathit{type\_specifier\_list}\\
\mathit{type\_specifier\_list} &\rightarrow \mathit{type\_specifier}\{,\mathit{type\_specifier}\}\\
\mathit{class} &\rightarrow \textbf{class}\ \mathit{uppercase\_identifier}[\mathit{subtyping\_clause}]\ \textbf{is}\\
&\quad [\mathit{class\_element}]\{;[\mathit{class\_element}]\}\ \textbf{end}\\
\mathit{class\_element} &\rightarrow \mathit{attr\_definition}|\mathit{routine\_definition}|\mathit{iter\_definition}|\mathit{include\_clause}\\
\mathit{attr\_definition} &\rightarrow [\textbf{private}]\ \textbf{attr}\ \mathit{identifier\_list}\ \textbf{:}\ \mathit{type\_specifier}\\
\mathit{routine\_definition} &\rightarrow [\textbf{private}]\mathit{identifier}[(\mathit{routine\_argument}\{,\mathit{routine\_argument}\})][\ \textbf{:}\ \mathit{type\_specifier}]\\
&\quad \textbf{is}\ \mathit{statement\_list}\ \textbf{end}\\
\mathit{iter\_definition} &\rightarrow [\textbf{private}]\mathit{iter\_name}[([\textbf{once}]\mathit{routine\_argument}\{,[\textbf{once}]\mathit{routine\_argument}\})]\\
&\quad [\ \textbf{:}\ \mathit{type\_specifier}]\ \textbf{is}\ \mathit{statement\_list}\ \textbf{end}\\
\mathit{include\_clause} &\rightarrow\ \textbf{include}\ \mathit{type\_specifier}[\mathit{feature\_modifier}\{,\mathit{feature\_modifier}\}]\\
\mathit{feature\_modifier} &\rightarrow \mathit{identifier}\ \textbf{-}\ \mathit{identifier}|\mathit{iter\_name}\ \textbf{-}\ \mathit{iter\_name}\\
\mathit{statement\_list} &\rightarrow [\mathit{statement}]\{;[\mathit{statement}]\}\\
\mathit{statement} &\rightarrow \mathit{declaration\_statement}|\mathit{assign\_statement}|\mathit{if\_statement}|\mathit{return\_statement}\\
\mathit{statement} &\rightarrow \mathit{typecase\_statement}|\mathit{expression\_statement}|\mathit{loop\_statement}\\
\mathit{statement} &\rightarrow \mathit{yield\_statement}|\mathit{quit\_statement}\\
\mathit{declaration\_statement} &\rightarrow \mathit{identifier\_list}\ \textbf{:}\ \mathit{type\_specifier}\\
\mathit{assign\_statement} &\rightarrow (\mathit{expression}|\mathit{identifier}\ \textbf{:}\ [\mathit{type\_specifier}])\ \textbf{:=}\ \mathit{expression}\\
\mathit{if\_statement} &\rightarrow \textbf{if}\ \mathit{expression}\ \textbf{then}\ \mathit{statement\_list}\{\textbf{elsif}\ \mathit{expression}\ \textbf{then}\ \mathit{statement\_list}\}\\
&\quad [\textbf{else}\ \mathit{statement\_list}]\ \textbf{end}\\
\mathit{return\_statement} &\rightarrow \textbf{return}\ [\mathit{expression}]\\
\mathit{typecase\_statement} &\rightarrow \textbf{typecase}\ \mathit{identifier}\ \textbf{when}\ \mathit{type\_specifier}\ \textbf{then}\ \mathit{statement\_list}\\
&\quad \{\textbf{when}\ \mathit{type\_specifier}\ \textbf{then}\ \mathit{statement\_list}\}[\textbf{else}\ \mathit{statement\_list}]\ \textbf{end}\\
\mathit{expression\_statement} &\rightarrow \mathit{expression}\\
\mathit{expression} &\rightarrow \mathit{bool\_literal\_expression}|\mathit{char\_literal\_expression}\\
\mathit{expression} &\rightarrow \mathit{str\_literal\_expression}|\mathit{int\_literal\_expression}\\
\mathit{expression} &\rightarrow \mathit{self\_expression}|\mathit{local\_expression}|\mathit{call\_expression}|\mathit{void\_expression}|\mathit{or\_expression}\\
\mathit{expression} &\rightarrow \mathit{void\_test\_expression}|\mathit{new\_expression}|\mathit{create\_expression}|\mathit{and\_expression}\\
\mathit{expression} &\rightarrow \mathit{sugar\_expression}|\mathit{while}!\mathit{\_expression}|\mathit{break}!\mathit{\_expression}\\
\mathit{self\_expression} &\rightarrow \textbf{self}\\
\mathit{local\_expression} &\rightarrow \mathit{identifier}\\
\mathit{call\_expression} &\rightarrow \mathit{expression}.(\mathit{identifier}|\mathit{iter\_name})[(\mathit{expression}\{,\mathit{expression}\})]\\
\mathit{void\_expression} &\rightarrow \textbf{void}\\
\mathit{void\_test\_expression} &\rightarrow \textbf{void}(\mathit{expression})
\end{aligned}
$$

$$
\begin{aligned}
\textit{new\_expression} &\rightarrow \textbf{new} \\
\textit{create\_expression} &\rightarrow \#[\textit{type\_specifier}][(\textit{expression}\{,\textit{expression}\})] \\
\textit{and\_expression} &\rightarrow \textit{expression}\ \textbf{and}\ \textit{expression} \\
\textit{or\_expression} &\rightarrow \textit{expression}\ \textbf{or}\ \textit{expression} \\
\textit{sugar\_expression} &\rightarrow \textit{expression}\ \textit{binary\_op}\ \textit{expression}|\textbf{-}\textit{expression} \\
\textit{sugar\_expression} &\rightarrow (\textit{expression})|\tilde{}\textit{expression} \\
\textit{binary\_op} &\rightarrow +|-|*|/|\%|\ |<|<=|=|/=|>|>= \\
\textit{loop\_statement} &\rightarrow \textbf{loop}\ \textit{statement\_list}\ \textbf{end} \\
\textit{yield\_statement} &\rightarrow \textbf{yield}\ [\textit{expression}] \\
\textit{quit\_statement} &\rightarrow \textbf{quit} \\
\textit{while!\_expression} &\rightarrow \textbf{while!}(\textit{expression}) \\
\textit{break!\_expression} &\rightarrow \textbf{break!}
\end{aligned}
$$

# Referencias


[1] Stoutamire, David, and Stephen Omohundro. "The sather 1.1 specification." INTERNATIONAL COMPUTER SCIENCE INSTITUTE-PUBLICATIONS-TR (1996).