# The Language $\text{XLE}_F EATURES$

BNF-converter

October 25, 2016

This document was automatically generated by the *BNF-Converter*. It was generated together with the lexer, the parser, and the abstract syntax module, which guarantees that the document matches with the implementation of the language (provided no hand-hacking has taken place).

## The lexical structure of $\text{XLE}_F EATURES$

### Literals

MyIdent literals are recognized by the regular expression $(\langle letter \rangle \mid \langle digit \rangle \mid$ '_')$(\langle letter \rangle \mid \langle digit \rangle \mid$ '_' $\mid$ '$-$')$*$

### Reserved words and symbols

The set of reserved words is the set of terminals appearing in the grammar. Those reserved words that consist of non-letter characters are called symbols, and they are treated in a different way from those that are similar to identifiers. The lexer follows rules familiar from languages like Haskell, C, and Java, including longest match and spacing conventions.

The reserved words used in $\text{XLE}_F EATURES are the following:$

```
o
```

The symbols used in $\text{XLE}_F EATURES are the following:$

```
:     &    !
{     }    .
->    $    %any
<<    [    ]
|     +    -
```

1

## Comments

There are no single-line comments in the grammar.
Multiple-line comments are enclosed with " and ".

## The syntactic structure of $\mathbf{XLE}_F EATURES$

Non-terminals are enclosed between $\langle$ and $\rangle$. The symbols ::= (production),
| (union) and $\epsilon$ (empty rule) belong to the BNF notation. All other symbols
are terminals.

$$\langle FEATURE \rangle \quad ::= \quad \langle ListRULE \rangle$$

$$
\begin{aligned}
\langle RULE \rangle \quad ::= \quad & \langle MyIdent \rangle : \langle EXP \rangle \\
| \quad & \text{\&} \ \langle MyIdent \rangle : \langle EXP \rangle \\
| \quad & \text{!} \ \langle MyIdent \rangle : \langle EXP \rangle \\
| \quad & \langle MyIdent \rangle : \{ \ \langle ListEXDISJ \rangle \ \} \\
| \quad & \langle MyIdent \rangle \\
| \quad & \text{o : :}
\end{aligned}
$$

$$
\begin{aligned}
\langle ListRULE \rangle \quad ::= \quad & \epsilon \\
| \quad & \langle RULE \rangle \ . \ \langle ListRULE \rangle
\end{aligned}
$$

$$
\begin{aligned}
\langle EXP \rangle \quad ::= \quad & \text{->} \ \$ \ \{ \ \text{\%any} \ \} \\
| \quad & \text{->} \ \$ \ \{ \ \langle ListCATS \rangle \ \} \\
| \quad & \text{->} \ \$ \ \{ \ \langle ListSYMS \rangle \ \} \\
| \quad & \text{->} \ \text{<<} \ [ \ \text{\%any} \ ] \\
| \quad & \text{->} \ \text{<<} \ [ \ \langle ListCATS \rangle \ ]
\end{aligned}
$$

$$\langle EXDISJ \rangle \quad ::= \quad \langle EXP \rangle$$

$$
\begin{aligned}
\langle ListEXDISJ \rangle \quad ::= \quad & \epsilon \\
| \quad & \langle EXDISJ \rangle \\
| \quad & \langle EXDISJ \rangle \ | \ \langle ListEXDISJ \rangle
\end{aligned}
$$

$$\langle CATS \rangle \quad ::= \quad \langle MyIdent \rangle$$

$$
\begin{aligned}
\langle ListCATS \rangle \quad ::= \quad & \epsilon \\
| \quad & \langle CATS \rangle \ \langle ListCATS \rangle
\end{aligned}
$$

$$
\begin{aligned}
\langle SYMS \rangle \quad ::= \quad & + \\
| \quad & - \\
| \quad & + \ \langle MyIdent \rangle \\
| \quad & - \ \langle MyIdent \rangle
\end{aligned}
$$

$\langle ListSYMS \rangle$    ::=    $\epsilon$
                              |     $\langle SYMS \rangle$ $\langle ListSYMS \rangle$