Next

Language Reference Manual

Ernesto Arreguin (eja2124)
Danny Park (dsp2120)
Morgan Ulinski (mu2189)
Xiaowei Zhang (xz2242)

Contents

1 Introduction

2 Lexicon

The Next programming language uses a standard grammar and character set. Characters in the source code are grouped into tokens, which can be punctuators, operators, identifiers, keywords, or string literals. The compiler forms the longest possible token from a given string of characters; tokens end when white space is encountered, or when it would not be possible for the next character to be part of the token. White space is defined as space characters, tab characters, return characters, and newline characters.

The compiler processes the source code and identifies tokens and locates error conditions. There are three types of errors:

- Lexical errors occur when the compiler cannot form a legal token from the character stream.
- Syntax errors occur when a legal token can be formed, but the compiler cannot make a legal statement from the tokens.
- Semantic errors?

2.1 Character Set

The Next programming languages accepts standard ASCII characters.

2.2 Identifiers

An identifier is a sequence of characters that represents a name for a:

- Variable
- Location
- Character

- Item
- Action

Rules for identifiers:

- Identifiers consist of a sequence of one or more uppercase or lowercase characters, the digits 0 to 9, and the underscore character (_).
- Identifier names are case sensitive.
- Identifiers cannot begin with a digit or an underscore.
- Keywords are not identifiers.

2.3 Comments

Comments are introduced by /* and ended by */, except within a string literal. Comments cannot be nested. If a comment is started by /*, the next occurrence of */ ends the comment.

2.4 Keywords

Keywords identify statement constructs and specify basic types. Keywords cannot be used as identifiers. The keywords are listed in Table **??**.

Table 1: Keywords

`if`	`then`	`else`	`and`	`or`
`start`	`end`	`when`	`choose`	`kill`
`grab`	`hide`	`exists`	`drop`	`output`
`character`	`location`	`action`	`item`	`int`
`string`	`next`			

Keywords are used:

- To qualify a data type (`character, location, action, item, int, string`)
- As part of a statement (`if, then, else, and, or, start, end, when, choose, kill, grab, hide, exists, drop, output`)

2.5 Operators

Operators are tokens that specify an operation on at least on operand and yield a result (a value, side effect, or combination). Operands are expressions. Operators with one operand are unary operators, and operators with two operands are binary operators.

Operators are ranked by precedence, which determines which operators are evaluated before others in a statement.

Some operators are composed of more than one character, and others are single characters.

The single character operators are shown in Table **??**.

Table 2: Single-character operators

+	-	*	/	<
>	=			

The multiple-character operators are shown in Table **??**.

Table 3: Multiple-character operators

>=	<=	==	!=	and
or				

2.6 Punctuators

2.7 String literals

3 Basic Concepts

3.1 Blocks

3.2 Scope

3.3 Visibility

3.4 Side Effects and Sequence Points

4 Data Types

5 Declarations

5.1 Declaration Syntax Rules

6 Expressions and Operators

6.1 Primary Expressions

6.2 Overview of the Next Operators

6.3 Unary Operators

logical negation?

6.4 Binary Operators

7 Statements

7.1 Labeled Statements

7.2 Compound Statements

7.3 Expression Statements

7.4 Selection Statements

7.5 Iteration Statements

8 Examples