SOCKit Reference Manual

Quentin Autry	`System Architect`
Hakim El Ghazi	`Float Goat`
Ryan Najac	`Manager`
Misha Smirnov	`Language Guru`
Stacey Yao	`Tester`

COMS W4115 Programming Languages and Translators
March 22, 2024

Contents

1 Overview

Sections and subsections will be automatically added to the table of contents.

2 Lexical conventions

There are six kinds of tokens: identifiers, keywords, constants, strings, expression operators, and other separators.

In general, blanks, tabs, newlines, and comments as described below are ignored except as they serve to separate tokens. At least one of these characters is required to separate otherwise adjacent identifiers, constants, and certain operator-pairs.

2.1 Comments

The characters ‘/*‘ introduce a comment, which terminates with the characters ‘*/‘. Single-line comments prefixed with ‘//‘ are also supported.

2.2 Identifiers (Names)

An identifier is a sequence of letters and digits; the first character must be a letter. The underscore character _ is considered a letter. Upper and lower case letters are distinct.

3 Keywords

The following identifiers are reserved for use as keywords and may not be used otherwise:

3.1 Constants

There are several kinds of constants, as follows:

3.1.1 Integer Constants

In keeping with the spirit of minimalism, we only support nonnegative integer constants. The value of an integer constant is the sequence of digits. We do, however, support different bases for integer constants. The base is indicated by a prefix that is either ‘0x‘ or ‘0X‘ for hexadecimal, ‘0b‘ for binary, and nothing for decimal.

3.1.2 data

The keyword `data` is used to declare a variable. It works similar to how Unix treats everything as files (i.e., everything data). It’s basically a void pointer, letting the programmer know that this data exists somewhere.

3.1.3 null

The keyword `null` is used to declare a null value. This is the absence of a value. Uninitialized variables are implicitly null.

3.2 Primitive Types

SOCKit has primitive types: `int`, `float`, `bool`, and `char`.

4 Syntax Notation

The syntax notation used in this manual follows the syntax notation in the "C Reference Manual" (Ritchie, 1975):

- `italic` is used for non-terminal symbols.
- `bold` is used for terminal symbols.
- `[ ... ]` is used for optional items.
- `{ ... }` is used for repeated items.
- `|` is used to separate alternatives.
- `...` is used for a syntactic category that can be repeated.

5 Data Types

No data types, everything is void pointers.

6 Expressions

The precedence of expression operators is the same as the order of the major subsections of this section (highest precedence first).

APPENDIX

Syntax Summary

1. Expressions.

```
expression:
    primary
    * expression
```

2. Declarations...