Bogus Reference Manual

1 Introduction

Bogus (BOard Game Specification) is a specialized language for defining "board games" that can be played by one or more users at a terminal. The language includes constructs that are particularly suited to the implementation of a board game, as well as control flow, input/output, comparison and other operations common to most high-level languages. The overall goal of Bogus is to create a language that makes it straightforward to implement the basic components of many type of boardgames, while also allowing the programmer to include more complex game rules and interations between players.

2 Lexical Conventions

A Bogus program consists of a single text file containing characters in the ASCII character set. This file is compiled into an intermediate format which is then excuted by the Bogus interpreter to actually play the game. This section describes the various token types that make up a bogus program.

2.1 Conventions

All ASCII values in this manual are given in decimal.

The terms "letter" and "digit" in this manual follow the common meaning of the terms in most common programming languages that use ASCII:

A *letter* is an uppercase or lower case character between A and Z (ASCII values 65 - 122)

A *digit* is a character between '0' and '9' (ASCII values 48 - 57)

All keywords in this manual are printed in `fixed-width` type.

Bogus is case-sensitive, so for example:

`if` is a keyword, but If is an identifier (though one that should probably be avoided).

thing1, Thing1 and THING1 are three separate identifiers.

2.2 Whitespace

The following ASCII characters are all considered whitespace. Since Bogus is a freeform language, whitespace (outside of string literals) is generally ignored. The exception is identifier separation: a sequence of one or more whitespace characters serve to separate adjacent tokens.

Whitespace characters:

Character	**ASCII Value (decimal)**
Horizontal tab	9

Newline/line feed	10
Carriage return	13
Space	32

2.3 Comments

A comment begins with a hash - '#' - character, and extends until a newline (ASCII 10) character. All contents of a comment are ignored by the compiler, and the comment is semantically equivalent to a whitespace.

2.4 Identifiers

An identifier is a sequence of one or more letters and digits and underscores (ASCII 95), where the first character is a letter. There is no predefined limit on identifier length, though the programmer is reminded that he is the one who has to decipher short, cryptic identifiers and retype long, verbose ones.

2.5 Keywords

All words listed below are reserved and may not be used as identifiers:

as

bool

contains

else

false

float

for

has

if

int

list

of

print

read

true

while

Card

Die

None

Player

Rule

Space

2.6 Constants

There are four types of constants in Bogus: integer, floating point and string.

2.6.1 Integer Constant

An integer constant is a sequence of 1 or more digits, which is taken to be a decimal value. The sequence may be preceded by a minus sign ('-', ASCII 45) to indicate that the integer is negative.

2.6.2 Floating Point Constant

A floating point constant consists of a sequence of 1 or more digits, followed by a decimal point ('.', ASCII 46) follow by a sequence of 1 or more digits. The decimal point is required, and one may omit the sequence of digits before or after the point, but not both. The floating point may be preceded by a minus sign ('-') to indicate that the value is negative

Some examples of valid floating point constants are:

1.0, .1, -2.99999, -.4

2.6.3 Boolean Constant

As you might expect, there are two boolean constants - `true` and `false` - which correspond to the boolean values you would expect.

2.6.4 String constant

A string constant is enclosed by double quotes ("", ASCII 34) and consists of characters and valid escape sequences. Valid characters are ASCII 32 through 254, excluding the double-quote character (ASCII 34), the backslash (ASCII 92) and the DEL character (ASCII 127).

Escape sequences are two character combinations that can be used to represent some special characters, as defined below:

Escape sequence	Represents
\\	\ (backslash)
\"	" (double quote)

\n	(line feed)
\r	(carriage return)
\t	(horizontal tab)

Any two character sequence begining with backslash and not listed in the table above is invalid in a string constant.

3 Types

3.1 Primitives

Bogus contains four primitive types: integers, floating point numbers (precision TBD), booleans and character strings.

Variables containing primitive types are not explicitly declared. Instead, the variable springs into existance when it is first assigned to, and its type is inferred from the assignment value. As long as a variable is in scope, it will always have the same type as when it was first create (see the scoping rules for more detail).

3.1.1 Integers

Integers are integral values in the range -TBD to TBD

3.1.2 Floating point values

Floating point values can be positive or negative and have precision TBD.

3.1.3 Booleans

Booleans can store only the Boolean values true or false.

3.1.4 Character strings

Character strings have a size limited only by memory constraints on the system where the compiler and interpreter are running. They may contain all ASCII characters from 32 through 254 (excluding ASCII 127, the DEL character) as well as the following white space characters:

Character	**ASCII Value**
Horizontal tab	9
Newline/line feed	10
Carriage return	13

4 Expressions

4.1 References

4.2 Expression Values

4.3 Assignment Operator

=

4.4 Concatenation Operator

+ (overloaded from binary addition operator)

4.5 Arithmetic Operators

+, -, *, /, % (modulo)

4.6 Comparison Operators

<, >, <=, =>, ==, !=

== and != can also do string comparison (are strings' contents identical).

4.7 Logical Operators

&&, ||, unary !

5 Statements

6 Scoping rules

Global scope

Lexical scope