JFlat Language Reference Manual

Project Manager:	David Watkins	djw2146
System Architect:	Emily Chen	ec2805
Language Guru:	Phillip Schiffrin	pjs2186
Tester & Verifier:	Khaled Atef	kaa2168

Contents

1. Introduction

JFlat is a general purpose, class-based, object-oriented programming language. The principal is simplicity, pulling many themes of the language from Java. JFlat is a high level language utilizes LLVM IR to abstract away hardware implementation of code. Utilizing the LLVM as a backend allows for automatic garbage collection of variables as well.

JFlat is a strongly typed programming language, meaning that at compile time the language will be type-checked, thus preventing runtime errors of type.

This language reference manual is organized as follows:

- Chapter 2 Describes types, values, and variables, subdivided into primitive types and reference types
- Chapter 3 Describes the lexical structure of JFlat, based on Java. The language is written in the ASCII character set
- Chapter 4 Describes syntactic grammar for the language
- Chapter 5 Describes classes, how they are defined, fields of classes or their variables, and their methods
- Chapter 6 discusses the different library classes provided by the compiler and their definitions

2. Types

Primitive Data Types:

- Integer
- Double
- Void
- Character
- Boolean

Integer

The integer type stores the given value in 32 bits. You should use integer types for storing whole number values (and the char data type for storing characters). The integer type can hold values ranging from -2,147,483,648 to 2,147,483,647.

Double

The double type stores the given value in 64 bits. You should use double types to store fractional number values or whole number values that do not fit into the range provided by the integer type. The double type can hold values ranging from 1e-37 to 1e37. Since all values are represented in binary, certain floating point values must be approximated. It is therefore recommended that the programmer compare doubles within a given range rather than with the equivalence operator ==.

Void

The void type is used to indicate an empty return value from a method call. As it is assumed that every method will return a value, and that value must have a type, the type of a return which has no value is null. An example would look like:
public void inc(a) a++

Char

A character constant is a single character enclosed with single quotation marks, such as p. The size of the char data type is 8 bits. Some characters cannot be represented using only one character. These extra characters are represented with an escape sequence, which consists of a backslash and another character. Some examples are:
Newline character T̑ab character Single quotation mark   Double quotation mark

Bool

The bool type is a binary indicator which can be set to either True or False. The bool type is stored as one byte (??). A bool must be given a value at the time of declaration:
bool x = True; // Valid declaration bool y; // Invalid declaration, value not declared

3. Lexical Conventions

4. Expressions and Operators

5. Program Structure and Scope

Program Structure

A Dice program may exist either within one source file or spread among multiple files which can be linked at compile-time. An example of such a linked file is the standard library, which is required to be linked at the top of each program.

6. Statements

7. Classes

8. Standard Library Classes