The Cerium Reference Manual

Contents

1. Introduction

*** Note that Cerium is a work in progress. I still don't have the code generator built and I have language features to add. This means that this document is a combination of things that are already built and some things that are still yet to be developed. For example, when I say that this is a language that runs on the JVM, it doesn't yet run on the JVM, since I still have to build the code generator. ***

This manual describes the programming language Cerium. The purpose of this document is to describe the language features of Cerium and introduce the basic syntax and semantics of the language.

Cerium is an OOP, statically typed language that runs on the JVM.

Cerium programs consist of a set of .cerium files, where one or more classes are defined. Every class defines a type. Each class encapsulates the members and methods of that class.

Inheritance allows new types to extend the behavior of existing types. Cerium uses the colon operator : to specify the superclass, the same as in C++.

2. Getting Started

The best way to get a feel for the Cerium language is to look at an example .cerium file. The project contains a number of .cerium files in the source-code directory within the project. The language itself resembles C++ in its syntax. I'm sure it will evolve and diverge a bit from C++ over time. C++ just seemed like a good starting point.

Within the project, there is a Compiler.java file. This is the class that actually performs the lexing, parsing, semantic analysis, and (eventually) the code generation routines. This class contains a main method, that takes a .cerium file as input (I currently hard-code a reference to a single .cerium file) and performs these actions. Note, that it currently only outputs a description of the annotated AST. Instead of writing this information to the console, I'll perform code generation, once that part has been developed. Also, I'll need to compile multiple

.cerium files. Right now, I’m only doing one file at a time, as this project is still in its infancy.

3. Classes

TODO