Java Code Engine for Online Code Compilation

Research · March 2018

DOI: 10.13140/RG.2.2.27241.19046

Java Code Engine for Online Code Compilation

Nalin Adhikari

Kathmandu, Nepal

nalinadhikariofficial@gmail.com

Abstract— **Java is a general-purpose, object-oriented, concurrent, class-based, computer-programming language that runs on Java Virtual Machine (JVM) regardless of computer architecture. As this is the age of internet, everything digital are online and need to work fast and efficiently. To work fast and efficient, plenty of high configured resources like cpu, memory, storage are required. So I created a research project called "Java Code Engine Online". The main aim of this project is to compile and run java code at the server side as the code is given from the client interface. The client machine does not have to install or configure Java Development Kit. This paper aims to describe an online compiler which helps to reduce the problems of portability, storage space and dependencies for Java Runtime Environment (JRE) by making use of concept of cloud computing. The ability to use different version of java compilers allows the programmer to write code faster and in convenient way hence compiling the code and removing the errors. Furthermore, the programmer can test his/her code in different version of compiler for backward compatibility check. Additionally, a web-based system can be used remotely through any network connection and is accessible worldwide. Ultimately, the trouble of installing the compiler on each client machine is avoided.**

Keywords— **java, compiler, online compiler, code engine**

I. Introduction

Java is one of the most supreme and popular language basically since its creation in the mid-90's [6]. Java is everywhere because once code is written in Java, it can be run everywhere because of its portability, thanks to the platform-agnostic Java Virtual Machine (JVM). However, JVM comes with dedicated system resources requirements such as memory, disk space, processor. Hence, in this era of cloud computing, it will be the revolutionary if Java source code can be compiled and run in the cloud. This obviously solves the system resources issue at the client side and a truly succours to portability features. To address the above mentioned problems, I have designed and developed a code engine that takes input as java source code from the client interface, compiles and run on the server. The system can detect the syntax error and runtime exception and returns them to client in proper format. As it is well known that compiling and running Java source code is done easily by saving source file in the disk, but my code engine can do it saving files in the memory which makes it much faster than other system. In this paper, I will present an overview of this code engine with it's in-memory compile and run feature.

II. Related Works

Various research papers have been published on online compiler and similar topics.

A survey had been done comparing three compiler; C, C++ and Java using Cloud Computing to address the problem of portability and storage space. [1]

Hiroki and Ushio proposes a web-based system that automatically scores programming assignments for students. The system receives a Java application program submitted by a student and returns the test result immediately. The test consists of compiler check, JUnit test, and result test. [2]

Students from BSIOTR had worked in browser based online IDE for coding, compiling and running Java source code which allows real time collaboration with other programmer. [3]

Similar work had been done by students from Pune University where they researched on online C/C++ compiler using cloud computing which helps to reduce the problems of portability and storage space. Furthermore, they talked about advantages of using web based application that uses multiple compilers remotely throughout any network connection making an ideal for conducting examinations online. [3]

III. Compiler

A compiler [4] is a software program that transforms high-level source code that is written by a developer in a high-level programming language into a low level object code (binary code) in machine language, which can be understood by the processor. The process of converting high-level programming into machine language is known as compilation.

A. Types of Compiler

1. Native Code Compiler

A native compiler is a compiler that works on compilation for the same technology on which it runs. It uses the same operating system or platform as the software for which it is assembling machine language.

2. Cross Compiler

A compiler that runs on one computer but produces object code for a different type of computer. Cross compilers are used to generate software that can run on computers with a new architecture or on special-purpose devices that cannot host their own compilers.

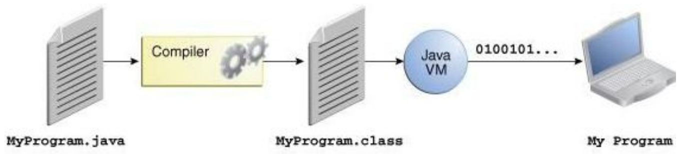
3. Source to source Compiler

As the name suggests such compilers convert the source code written in one programming language into another programming language that operates at almost same level of abstraction.

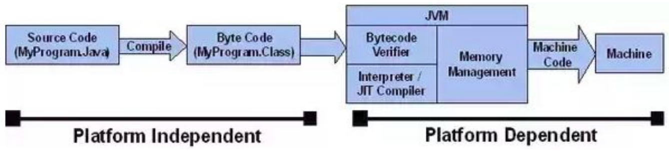
The [5] Java programming language introduced the possibility of compiling output (called bytecode) that can run on any computer system platform for which a Java virtual machine or bytecode interpreter is provided to convert the bytecode into instructions that can be executed by the actual hardware processor. Using this virtual machine, the bytecode can optionally be recompiled at the execution platform by a just-in-time compiler.

B. How Java Compiler works

Java [7] has compiler name as javac which convert source code to intermediate code which is know as java bytecode. This java bytecode is not dependent on any platform that is if you compile your source code in windows platform by using javac compiler so you can run this code in any other platform like linux , Mac.



This bytecode is actually not native code it is intermediate code. In java there is a virtual machine is available to convert intermediate code into the native code which is known as JVM (Java Virtual Machine). The JVM is actually work like Interpreter and it’s functionality is first verified bytecode and JVM has compiler which is known as JIT (Just In Time) Compiler. So JVM take bytecode and give it to the JIT and JIT compiler actually compile the bytecode and convert to machine dependent native code.



IV. Cloud Computing

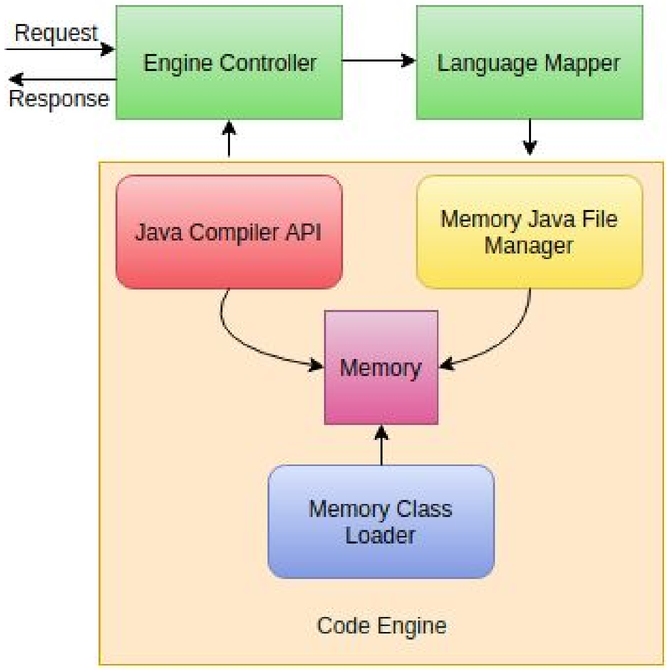
Cloud [8] computing is a type of computing that relies on shared computing resources rather than having local servers or personal devices to handle applications. In a cloud computing system, there's a significant workload shift. Local computers no longer have to do all the heavy lifting when it comes to running applications. The network of computers that make up the cloud handles them instead. Hardware and software demands on the user's side decrease. The only thing the user's computer needs to be able to run is the cloud computing system's interface software, which can be as simple as a Web browser, and the cloud's network takes care of the rest.

V. Code Engine Meets Cloud

A. System Architecture

The core system architecture of code engine includes: engine controller, language mapper and code engine. Furthermore, code engine includes: java compiler api, memory java file manager, memory class loader and memory (RAM).

Engine controller, handles request from the client app and transfer it to language mapper. It also receives response from code engine and pass it to the client app in predefined format. Language mapper maps the request from engine controller to the code engine service. For this system, it only maps java source code. Inside code engine, the source code received from the language mapper is handle by memory java file manager, which saves the source code in the memory with file extension as java. Then the java compiler api kicks in and compile that source file in the memory without saving on the disk which yields java bytecode (class file). The class file also resides in the memory. After that memory class loader loads the class file from the same location and again java compiler api execute the java bytecode for result. If any compilation error and runtime error or any errors occurs then proper error message is thrown to the client app.



B. Features

The key features of the system are:

- Takes java source code as request and gives out the result after running the source code.
- Saves source file and class file in the memory and removes after the task completes.
- Handles syntax error and logical error with proper error format.
- Faster than traditional way of compiling program source code because of highly configured and resourceful cloud server.
- Multiple version of compiler can be used and switched easily.

C. Future Scope

With reference to this architecture or engine, various system can be build according to the requirements. For example, to build a java online IDE to take online exam on java programming language for recruitment employees (programmer) as well as to take test of students in universities or colleges. Similarly, by adding new languages like python and javascript at the language mapper service, one can build multi-language online compiler.

VI. IMPLEMENTATION

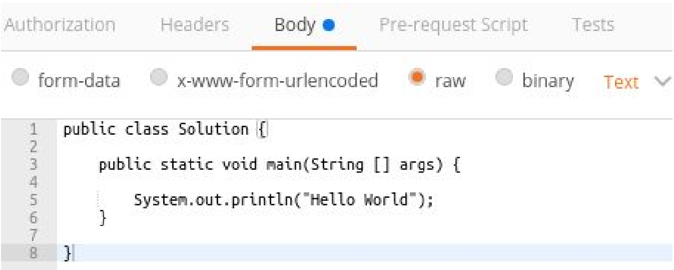
A. Tools Used

a. Java Version : 1.8.0_161 (64-Bit)
b. Eclipse IDE : Oxygen.2 Release (4.7.2)
c. Spring Boot, Version 1.5.4.RELEASE

B. Request & response

I have just included simple sample of request and response.

Request:



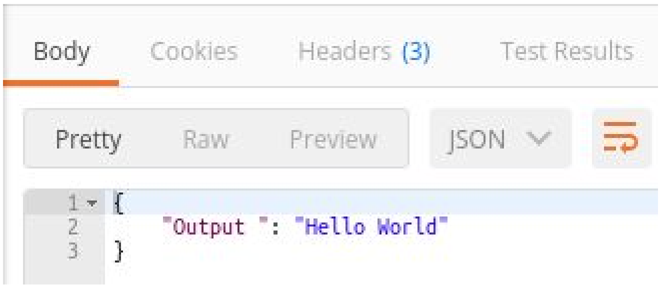
```
public class Solution {

    public static void main(String [] args) {

        System.out.println("Hello World");
    }

}
```

Response:



```
{
    "Output ": "Hello World"
}
```

C. Java Classes Used

- *MemoryClassLoader* that loads .class bytes from memory which extends *URLClassLoader*

- *MemoryJavaCompiler* which is an interface to Java compiler using JSR 199 Compiler API.
- *MemoryJavaFileManager* that keeps compiled .class bytes in memory and which extends *ForwardingJavaFileManager*.

VI. CONCLUSION

This architecture has obviously added extra layer of convenient and efficient way for building online compiler. Furthermore, it also opens the way of compiling java source code in the memory without saving file in the disk, which ultimately improves the compilation time of the compiler making the system faster. Using the advantages of cloud computing, hosting code engine on the cloud helps to eliminate the overhead of setting-up java runtime environment and other dependencies on each client side. In addition to this, compiling and running java source code in multiple version of java compiler can be done easily.

References

[1] Priyadarashani D., Surabhi S., Sneha K., Anumeha S., Heena Y., "ONLINE C, C++, JAVA COMPILER USING CLOUD COMPUTING - A SURVEY", Pune University, India, ISSN: 2319-1120 /V2N3: 318-323 IJAEST.

[2] Hiroki K. and Ushio I., "An Online Automated Scoring System for Java Programming Assignments", International Journal of Information and Education Technology, Vol. 6, No. 4, April 2016.

[3] Aamir N. A., Siddharth P., Arundhati N., Aditya P., Venkatesh B., "Online C/C++ compiler using cloud computing", IEEE Multimedia Technology (ICMT), 2011 International Conference on 26-28 July 2011.

[4] Techopedia, Compiler. Retrieved from https://www.techopedia.com/definition/3912/compiler.

[5] WhatIs.com (September 2016), Compiler. Retrieved from http://whatis.techtarget.com/definition/compiler.

[6] stackify.com (December 18, 2017) Most Popular and Influential Programming Languages of 2018, Retrieved from https://stackify.com/popular-programming-languages-2018.

[7] quora.com (Aug 18, 2016), How does the java compiler works?, Retrieved from https://www.quora.com/How-does-the-java-compiler-works.

[8] webopedia.com, cloud computing, Retrieved from https://www.webopedia.com/TERM/C/cloud_computing.html