Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia - Social and Behavioral Sciences 56 (2012) 767 – 773

Procedia Social and Behavioral Sciences

International Conference on Teaching and Learning in Higher Education (ICTLHE2012) in conjunction with RCEE & RHED 2012

Java Programming Assessment Tool for Assignment Module in Moodle E-learning System

Norazah Yusof[a,*], Nur Ariffin Mohd Zin[b], Noor Shyahira Adnan[a]

[a]*Faculty of Computer Science and Information Systems, Universiti Teknologi Malaysia, UTM Skudai, 81310, Johor, Malaysia*
[b]*Faculty of Computer Science and Information Technology Universiti Tun Hussein Onn Malaysia, 86400 Parit Raja, Johor, Malaysia*

Abstract

This paper proposes an intermediate system called JAssess which is developed to provide a handy way to manage submission of students' Java programming exercises from Moodle™, as well as grading them semi-automatically. Details about the proposed system and the algorithm that lay behind it is explain. It presents the major methods used while evaluating the Java programming assignment and how to overcome the different environment used in JAssess and Moodle™. A few test samples are included. Results show that the proposed model is able to display the suggested mark along with the output for every successful compilation, and will display the error along with the suggested mark for every failed compilation. Some limitations of the system and suggestions for future works section was conclude in this paper.

Keywords: programming assessment; Moodle™; web-based; Java programming language; JAssess

1. Introduction

Learning programming languages, such as the Java object oriented programming language, is a great challenge especially when the students are still in the basic level of programming efficiency. A lot of practical exercises are needed to help the students to lead to better understanding on a particular topic. By giving appropriate programming exercises and assignments, the students' cognitive skill can be increased. Unfortunately, lecturers take a lot of time and effort to evaluate and assess the students' programming exercises and assignments. The

* Corresponding author. Tel.: +6-019-728-0505
E-mail address: norazah@utm.my

doi:10.1016/j.sbspro.2012.09.714

manual approaches of assessing programming assignment, such as the hardcopies and diskettes, are obviously costly, time consuming and inflexible.

Nowadays, learning management system (LMS) is widely used to provide learning supports to the lecturers and students, as well as administrative and technical support (Guido & Andreas, 2009). One of the most prominent LMS used today is Moodle™ (Cole & Foster, 2007). Assignment module is among the most useful facility in Moodle™ that allows lecturer to set assignment with a due date and a maximum grade. Within the specified date, the students are able to submit their assignments to the server. The system records the date of when the students upload their programming files and this function allows lecturer to view any late submission. The lecturer is then able to download the programming work and give grades and feedbacks. Students are able to view the grades and feedback at any time and location. However, Moodle™ lacks of support in compiling and running the programming language assignments, specifically the Java programming language. The lecturer needs to download each programming source files and save it in different locations in the lecturer's own personal computer. The lecturer will then need to compile and run the program separately and then come back to Moodle™ to give marks and feedbacks.

The functionality of Moodle™ as the learning management system needs to be extended so that it can provide a handy way to evaluate students' Java programming exercises. Therefore, this paper reports about the proposed system named JAssess which is developed in Java platform. It provides a seamless solution in providing a handy way to manage submission of students' Java programming exercises, including compiling and running the programs, as well as giving marks and feedbacks. Within this system, the lecturers and students interact with the Moodle™ system to upload and download Java programming assignments. The lecturers are able to evaluate the programs without the need to log out of the Moodle™ system.

2. Related Works on Programming Assessment Tools

The earliest automatic program grading tool was published by Uollingsworth (1960) which was used for a formal course in programming. At that moment, old style key punched programming was still widely used. Since then, the use of automatic grader has been expanded widely in several programming languages until today.

TRY system was developed in late 80's for Unix operating system which test student program with a set of hidden test data (Reek, 1989). By keeping the test data hidden, students are encouraged to design programs without any advance knowledge of the test cases.

Web-CAT, an open source automated grading system, grades the student codes and students' assignment for each submission by writing their own test code (Edwards and Pérez-Quifiones, 2008). This approach helps the students to understand better of their own work by having the sense of responsibility to prove the correctness and validity of their program. But as far as we concern, a fully automated grader that performs the entire task would be very much convenient without having a single effort.

3. Overview of the proposed system

Moodle™, an acronym for Modular Object-Oriented Dynamic Learning Environment, is an open source software written in PUP language, which is designed to help both lecturers and students the opportunity to interact each other through online course (Cole & Foster, 2007). Moodle™ provides many modules for various teaching and learning activities, such as the resource module, assignment module, discussion forums, and quiz module (Kumar et. al., 2011). One important characteristics of Moodle™ is that it allows modification

made to the original copy of the source code provided that the developer should not modify or remove the original license. This characteristic makes it possible to extend the functionalities of Moodle™ so that it can meet certain organization needs.

JAssess, an acronym for Java Assessment tool, is the interface to compile and run Java programming source files to produce appropriate output. This tool is written in Java language. The lecturer may adjust the marks suggested by the system and give feedback to the students appropriately. Figure 1 shows the Moodle (in PUP platform) and JAssess (in Java platform) in which both are web-based systems and supported by the MySql database. Although PUP and Java are not similar in their syntax and semantics, we are able to connect them together through the intermediate controller (named JAssessMoodle) that can interact between the two technologies.

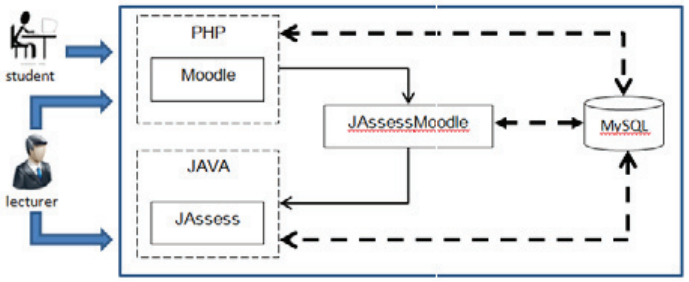

Fig. 1. JAssess and Moodle framework

Figure 2 illustrates the flow of the assessment process. The process starts by the lecturer defining the assignment's description and tasks in Moodle environment.

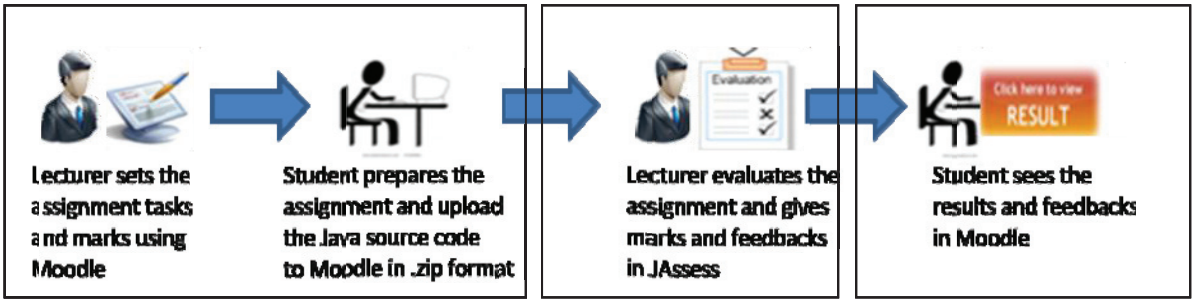

Fig. 2. Flow of the assessment process

Lecturer need to specify the assignment name, the due dates and more importantly the marks that students should get during successful and failed compilation as shown in Figure 3. After posting the task to the assignment module, the students are able to view the assignment tasks and start preparing the answer by writing Java program based on the question given. A text editor such as "Textpad" is used to write Java source file. Students are encouraged to test their Java programs to make sure that there is no syntax and runtime error.

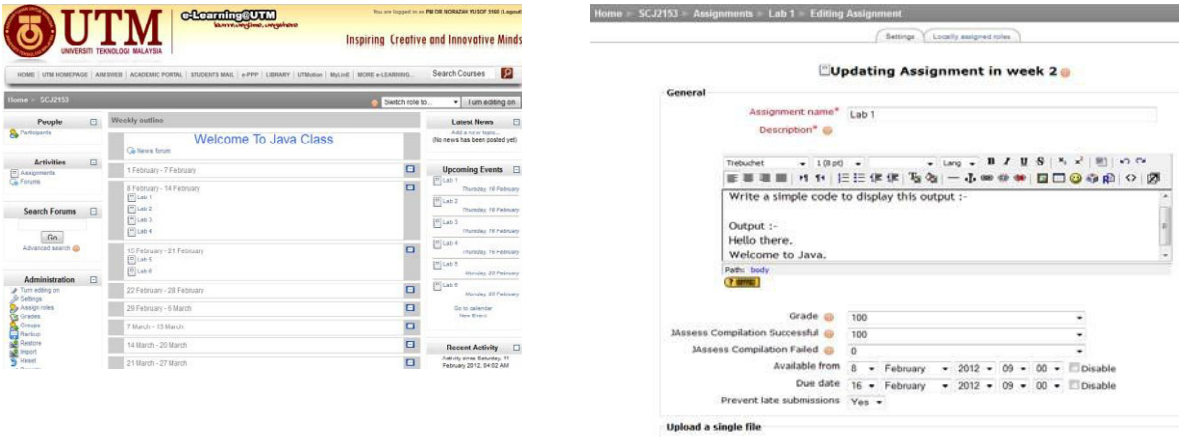

Fig. 3. Lecturer setting the assignment tasks

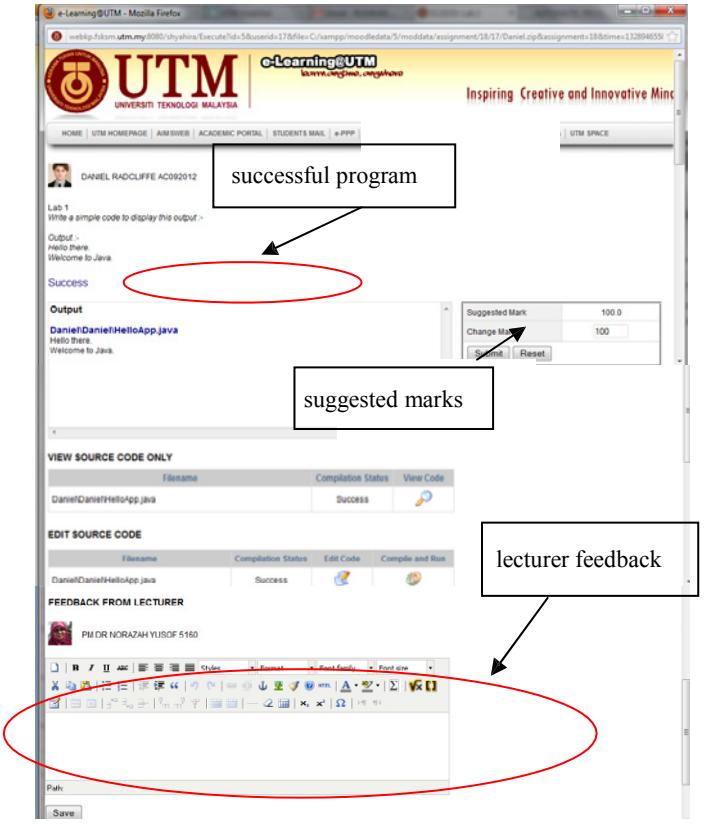

Fig. 4. Lecturer setting the assignment tasks

Submission of the assignments is done through uploading the Java source files through Moodle environment. However, before submitting the source files, students are required to compress their java programs into .zip format. The name of the zip file should be the same as the name of the class that consists of the main method. This is to follow the syntax of compiling Java program using java statement. Once uploaded the compressed .zip file will be stored in the database enclosed with their particulars, submission date and assignment's description.

Next, is the evaluation step in which the lecturer is able to view all submitted assignments in the Moodle environment, including the student's name, the name of the source file, and time of submission. To assess the assignment, the lecturer may click on the "Grade" button. This action directs the system to the JAssess system via JAssess Moodle. The result of the execution of the student's source files will be displayed as shown in Figure 4. For the successful compilation, lecturer able to see the output of the program and the source code of the original program. For the unsuccessful compilation, the lecturer may see the code to identify the possible reason and may edit the code. The system also displays the suggested marks and the lecturer may change the marks at the 'Change Mark' area, as well as provide feedback at the feedback text area.

4. JAssess Assessment Algorithm

The predefined Java File, Java Unzip, Java Runtime, Java Compiler dan Java Reflection plays a big role in making the assessment successful (Truong et. al., 2004). The first step in the assessment process is to create a folder at the server with the name of the student's matric number. Inside this folder, a subfolder with the assignment's name is created. This subfolder contains the compressed zip file. These processes require Java File class and therefore the File class from package java.io.File should be imported to ensure these processes possible.

Before any process could proceed, the class should be initiated as example below.

```
File f = new File("C:\\student");
```

Based on the example, an instance "f" is declared which assigned to the location and the name of the file/folder. At this point, we can perform several tasks by using predefined methods. To indicate whether the location assigned is a directory or a file, we can use is Directory() or is File() method, respectively. To determine whether a file exist or not, the method exists() should be used. The method returns a true if the file/folder is exists or false if the file is not exists. If the folder location defined is not exists, we can use method mkdirs() to create the folder. On certain condition, we need to list out all the files in the student's folder, and that's when we invoke the method listFiles(). Method delete() erases a folder by returning a true value if the folder successfully deleted and false value, otherwise. One of the requirements that the student should follow prior to the submission of the Java source files is to compress all source files into a single file using the .zip format. Thus, during the second assessment process, the decompression process of the zipped files needs to be performed before the compilation process. The java.util.zip package provides features for data compression and decompression in ZIP and GZIP formats. This package provides a ZiplnputStream class for reading the ZIP files. Once a ZIP input stream is opened, we read the zip entries using the getNextEntry() method which returns a Zip Entryobject. The set up of the decompressed output stream is specified as follows.

```
int BUFFER = 2048;
FileOutputStream fos = new FileOutputStream(entry.getName()); BufferedOutputStream dest = new
BufferedOutputStream(fos, BUFFER);
```

Third, in order to compile the Java program, a file named Compile.class need to be copied to the student's subfolder. This file contains algorithm that applies the Java Compiler as a mean to compile student's Java program. The command prompt application needs to be executed in order to compile the student's source code and run them. The tricky part is to run the command prompt within JAssess, which means it has to be triggered automatically by a command in the source code. This is done by employing Runtime class from package and calling exec()method as shown below.

```
Runtime rt = Runtime.getRuntime(); Process pr = rt.exec("cmd /c dir");
```

The fourth step is the Java Compiler from javax.tools package as shown below. At this stage, the compilation is done within the source code and without employing the javac command. Once successful, a byte code file is being generated that is known as a class file.

```
JavaCompiler compiler = ToolProvider.getSystemJavaCompiler();
```

If unsuccessful, error message is displayed. Java Reflection provides a dynamic way to manipulate object at runtime. It has the ability to examine or modify the runtime behavior of applications running in the Java virtual machine. By narrowing the usage of this class, it is used to assess the main method of the student's program. This task could be done by importing the java.lang.reflect.Method package. Inside the Method class, resides a function named invoke() which is used to run any method in a class.

The example below shows the use of Java Reflection to run a Java program called Hello.java.

```
Class a = Class.forName("Hello");

Method m = a. getDeclaredMethod("main", new Class[] {String[].class });
```

In the above code, the Java program is declared by the forname() method and the main() method accessed by the getDeclaredMethod()method. Once the main()method is accessed, the invoke() method is called as follows.

```
m.invoke(null, new Object[] { null });
```

5. Results and Discussion

We have conducted several tests by preparing a set of test cases. The test cases represent sample programs that are based on the assignment questions. Each case purposely prepared for any of these two conditions: successful compilation and unsuccessful compilation.

With the proposed system, the lecturer found out that the students' programs can be graded in ease. Lecturers just need to click on the 'Grade' button under the JAssess column and the system will display the compilation results. For every successful compilation, it displays the output with the suggested high marks. However, if a compilation failed, it will display the error along with the suggested low marks. Lecturers are able to view each of the source code submitted by the students. Hence, for unsuccessful compilation cases, they can track for errors and manually correct it until the programs are error free and ready to be run.

6. Conclusions and Future Work

JAssess is a web based system that grades student's Java program based on successful or failed compilation. This system implements the predefined packages of Java File, Java Unzip, Java Runtime, Java Compiler dan Java Reflection to complete a full process of assessment. Each packages assigned their own roles and together, it helps to reduce the workloads faced by educators each semester. Moodle is a learning management system that creates opportunities for educators and students for interaction for teaching and learning activities. Regardless of features of an e-learning platform, we did not find any grading tools that fit to assess a Java program. As a result, integrating between Moodle and JAssess would be a perfect match as it will complements each other.

Although the integration is a successful attempt, it still has drawbacks. One limitation is that the marks is given based on successful compilation and are not aware of the logic errors. Students may receive high marks if they submit any error free Java program even though the programs do not meet the criteria of the question. Therefore, the lecturers still need to view the output and verify the correctness of the program, which makes this system obviously semi-automated. For future research, the assessment process may be improved; from compilation based to logical based.

Acknowledgements

We are grateful to the Center of Teaching and Learning, Universiti Teknologi Malaysia for supporting the work in this field with the Research Grant, Vote No. Q.J130000.2528.01H82.

References

Cole, J. and Foster, H. (2007). Using Moodle: Teaching with the Popular Open Source Course Management System, O'Reilly Media, Inc.
Edwards, S. H. and Pérez-Quiñones, M. A. (2008). Web-CAT: Automatically Grading Programming Assignments. ITiCSE'08, June 30—July 2, Madrid, Spain
Guido, R. and Andreas, K. (2009). Extending Moodle To Better Support Computing Education. Proceedings of the 14th Annual ACM SIGCSE Conference on Innovation and Technology in Computer Science Education, Paris, France, p. 146-150, ACM Press, New York.
Hollingsworth, J. (1960). Automatic Graders for Programming Classes. Communication of ACM Vol 3, Issue 10, pp. 528-529.
Kumar S., Gankotiya A. K. and Dutta, K. (2011). A comparative study of Moodle with other e-learning systems, 3rd International Conference on Electronics Computer Technology (ICECT)
Reek, K.A. (1989). The TRY System – or – How to Avoid Testing Student Programs. Proceedings SIGCSE Bulletin vol.21, 1 (February 1989), pp. 112-116.
Truong, N., Roe, P. and Bancroft, P. (2004). Static Analysis of Students' Java Programs. Sixth Australasian Computing Education Conference (ACE2004), Dunedin, New Zealand.