Test Code Generation Tool for Self Learning Programming Unity

1st Usman Nurhasan [1], 2nd Anugrah Nur Rahmantyo[2], 3rd Ariadi Retno Ririd [3] , 4th Alwy Abdullah [4] , 5th Yan Watequlis Syaifudin[5] , 6th Pramana Yoga Saputra [6] , 7th Satrio Binusa[7]
{usmannurhasan@polinema.ac.id[1] , anugrahnur@polinema.com[2] , faniri4education@gmail.com[3] }

Information Technology, State Polytechnics of Malang, Jalan Soekarno Hatta No.9 Malang 65141, Indonesia

Abstract. The curriculum for making game applications is commonly applied in vocational education institutions. Learning to make game applications generally uses Unity and is multiplatform. The concept of learning programming is writing code in an IDE. Students read the module and then do the coding. However, when an error occurs, students often do not understand the concept of improvement. Another problem is the length of the gradle synchronization process in the Unity framework and the difficulty of the teacher in checking the work of each student. The length of the synchronization process has the potential to hinder the learning process. Because game programming has a complex structure and is related to game supporting assets. The lower the specification of the device used, the longer the synchronization process. So it affects the learning process and correction. This system facilitates self-learning of game programming. The results of student work will be checked using the Test-driven Development (TDD) method with the test code provided. If the test code shows a successful result, students can upload their work results to the SeLPU (Self Learning Programming Unity) website and the assessment of the work results will be automatically displayed on the web page.

Keywords: auto grading, automation testing, programming, self-learning, unity, TDD

1 Introduction

With the increasing use of smartphone devices, the need for Android mobile applications is increasing. One of the significant increases in the need for mobile applications is in the gaming sector[1][2]. As a result the demand for game application developers has become one of the highest in the IT field. The curriculum for making game applications has been commonly applied in educational institutions[3], [4]. Learning to make game applications generally uses Unity and is multiplatform[5]. The concept of learning programming is to use an IDE. Students read the module and then do the coding. However, when an error occurs, students often do not understand the concept of improvement. Another problem is the length of the gradle synchronization process in the Unity framework and the difficulty of the teacher in checking the work of each student[6][7]. The length of the synchronization process has the potential to hinder the learning process. Because game programming has a complex structure and is related to game supporting assets. The lower the specifications of the device used, the longer the synchronization process. So it affects the learning process and correction.

To increase knowledge of the game programming learning environment, we have developed Self Learning Programming Unity (SeLPU) by adopting the concept of Test-Driven

ICoSTA 2022, November 01-02, Medan, Indonesia

DOI 10.4108/eai.1-11-2022.2326182

Development (TDD). SeLPU offers automatic tagging of student answers that can guide students to self-study and reduce the burden on teachers[8]. In SeLPU, a student will be given a set of tasks to create a game project according to the specifications given in the assignment guide written in the C# programming language[9]. Then, the results of the assignments or answers are verified by the appropriate test code generated by the teacher on the NUnit software API and Unity Test Framework. In this paper, we propose a learning model for SeLPU. In learning Android programming, there are many topics to be learned. Here, we develop steps for studying the topic of 2D game programming. This is very important considering to be able to become a game developer, one must understand the basic concepts of game programming and be able to realize the interaction of game supporting assets in an application design[10]. To evaluate the proposal, we applied a system to undergraduate students in Indonesia.

2 Related Works

In 2019, Umar et al conducted a performance comparison on several software testing framework automation [11]. With the aim of getting Quality and Speed, because it saves time, reduces costs, increases efficiency, and increases accuracy. Therefore, effective software testing. This can be achieved by using the right automation framework.

Currently the C# programming language is widely used for game-based application development. The framework used as the compiler is developed in .Net. So the unit testing framework that can be used is NUnit. In 2013, Kumar et al. have tested the Effective Unit Testing Framework for Automation of Windows Applications [12]. Testing is done in general C# programming. The results of his research state that by using a unit testing framework, programmers can save time to make code improvements and the quality of the resulting code is much better than if not using a unit testing framework.

In research conducted by Funabiki in 2017, a Test Code Generation Tool for java programming has been produced. In this study, students' test code corrections were carried out automatically and the learning patterns used Test Driven Development [13]. The results obtained after testing on a number of source code, the automated generation tool that was built succeeded in producing the test correctly.

3 Test Driven Development Method

In this section we will review the Test-Driven Development Method (TDD) that utilized in SeLPU

3.1 TDD Method

The TDD method is a software development process that relies on repeating very short development cycles. Requirements are turned into very specific test cases. Then, the program code is upgraded to pass the new tests. In game programming applications there are three types of tests, namely unit testing, integration testing, and UI testing[14]. In this paper, we will focus on unit tests, because they can test certain objects from the application part of the application. The application of TDD in programming learning includes several aspects, including:

a. Do the right test, Developers need to create proper unit tests to verify functionality or features which are learning aspects. The test results that have been collected can be used as an analysis of success in learning. In most cases, the test will fail. However, the failure is meaningful because developers can analyze how the feature works.
b. Code correction, After the test fails, the developer can make the necessary changes to improve the previous code. Changes to the source code aim to improve the process and syntax in order to achieve the expected output.
c. Refactor code, The code refactor is part of the validation phase, where the development team analyzes the data collected during the test run. The team needs to compare the results with the standards set in the testing phase. If not met, the code must be rewritten to run the tests again.

3.2 Unit Testing

In general, unit testing is the process of testing the functions, steps, units and methods of the source code that we have written in the IDE. This is done as proof of whether the code complies with the desired specifications[15]. This stage is very important and is recommended to be done on every source code that has been built. At this stage unit testing will help software developers verify the functions that have been built in order to get results that are in accordance with the rules of writing the syntax of a source code.

For example, a developer has built a simple function, namely the authentication (login) function. In this function, the developer will first check whether the username or password has met the main requirements, for example a minimum character length of 6 characters. If this condition is not met, your function will return an error message. But if it has been fulfilled, then proceed to the next check, do the given username and password designate the user registered in the application? If yes, then the login is successful, otherwise it will display an error message that the user was not found.

For the authentication function above, we have to create a unit test that satisfies each stage in the function. Each stage of the function must be made into a unit test. For example, the stage of checking the validity of the entity on the username or password, will be used as a unit test. Then checking the application regarding the presence or absence of users is also a unit test. Now, if all the stages in the function have been made, it can be concluded that the existing functions have met the expected standards. Usually if all unit tests for this function are executed, it will produce a green color or 100% coverage. However, it can be different if the unit tests that have been carried out have not covered all the existing stages, then the coverage produced is usually below 100%.

The benefits of having unit tests will certainly help reduce errors in the application being built. This is because the developer has thought of alternatives for each function, both normal alternatives and alternatives that can produce errors, such as the authentication function above. Also, because of the alternatives that have been considered, you must have handled them well and successfully passed the unit tests. Of course this is what helps the application to be worthy of release to the market.

So it can be concluded, unit testing is one of the important things to do in automatic correction of a source code, but there are many other types of tests that can be developed further, for example integration testing, end-to-end testing, load testing and so on. When it comes to learning programming, especially game programming, unit testing is an indispensable concept. This is because building a game requires a lot of assets and logic to create a good game scenario. While the obstacles that exist in the field, students who learn to make games have difficulty in correcting the errors that occur. And teaching teachers also have difficulty in assessing student

projects because they have to carry out each student's project. This takes a lot of time and resources.

As previously explained, unit testing is software testing that is carried out manually or automatically using a special tool or tools. However, most of the developers nowadays prefer the automated method. The following are some tools that you can choose to do unit testing based on the programming language used.

- JUnit: unit testing tools for applications using the Java programming language
- NUnit: a unit testing framework for applications using the .NET programming language
- JMockit: unit testing tools for opensource applications
- EMMA: tools for analyzing and reporting code using the Java programming language
- PHPUnit: unit testing tools for PHP programmers

3.3 Unity Aplication Testing Framework

NUnit is an open source framework used to test .NET Framework applications. Currently Nunit can be used for automatic checking of game programming using C#, although in fact nunit is also compatible with other programming languages. The game programming learning process uses the Unity IDE, so the IDE is compatible with the NUnit Framework[16]. Since the environment is ready for the framework, we need to write unit tests. First every developer has to find the source software docs and then start writing unit tests. Since we are writing unit tests for the automation of a windows application, we must access every control in the application and validate them. So unit test developers must be familiar with application code to access controls in unit tests. In the case of NUnit, after the developer writes the class, he has to create a Test project, or if the Test project already exists, he has to manually write a separate class to write unit tests because there are no integrated features built into the IDE. An example of a unit test is shown in **Figure** 1.

```
// A Test behaves as an ordinary method
[Test]
0 references | Run Test | Debug Test
public void TestHelloWorld()
{
    // Masukkan Hello World! pada string hello
    // string hello = "Hello World!";
    // Assert.AreSame("Hello World!", hello);

    GameObject testObject = new GameObject();
    HelloWorld halo = testObject.AddComponent<HelloWorld>();
    Assert.AreEqual("Hello World!", halo.hello);
}
```

Fig. 1. Sample Unit Test

4. Learning Model

Computer-assisted Learning (CAL) is one of the strategies for the teaching and learning process that is carried out by utilizing computers as a medium of information. CAL has also been known by several other terms such as technology-enhanced language learning, computer-assisted language instruction (Davies) and computer-aided language learning, but the field is the same. Methodologically, the term CAL is often referred to by various terminology, including Computer-Managed Learning/Instruction (CML), Computer Aided or Assisted Instruction

(CAI), Computer-Based Education/Learning (CBL) or others. Subject matter can be presented by the CAI program through various methods such as: tutorials, drill and practice, simulations, games, problem solving, discovery and investigation. Some of the benefits of Information Technology in supporting the teaching and learning process, among others:

- As a tool to improve the quality of learning, such as visualization tools and computational tools.
- Computers can be a tool to automate the student assessment process.
- Sources of materials and information as well as easy and inexpensive learning media.
- As a non-online source of information

As the initial stage of developing Self Learning Programming Unity (SeLPU), this study focuses on the User Interface and Basic Unity programming language logic for game programming learning.

4.1 Learning Process

The learning application prototype consists of a client-server system. It aims to integrate the roles of teachers and students. The platform used is a website and was developed to distribute learning materials to students, collect assignment answers from students, and automatically validate student data. To clarify, **Figure 2** shows the client server model in the application.

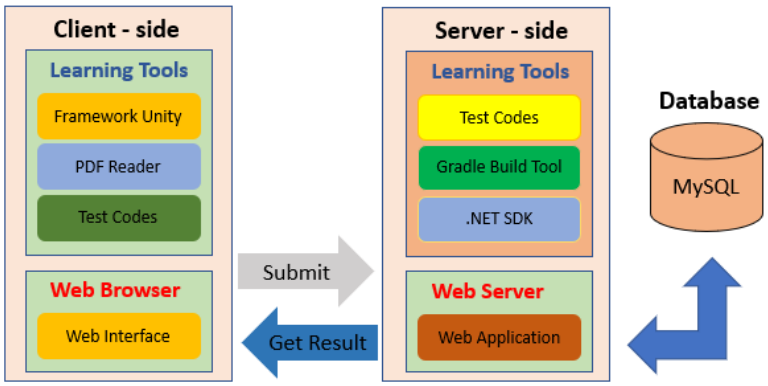

Fig. 2. Client Server Architecture Model.

- Client side: For the client side, SeLPU provides a web application for students that can be accessed using a browser. Students use this feature to download learning materials, send assignment answers to the server, and get the results of validating the submitted answers.
- Server side: Server side platforms provide web applications, validator programs and system databases. The web application has a function to distribute teaching materials and collect assignment answers from students. To confirm the correctness of the answer code, the validator program functions to validate the answers sent through the middleware automatically and in real time. Then the results can be accessed by students and teachers through a web application.

- validator: Details of the validator process will be shown in **Figure 3**. In the picture, the functions of the validator are explained, including receiving the unity project file sent by the web client. Then compare the source with the test code that is already available. The execution results are stored in the database and sent back to the client as information.

Student learns how to code a C# programming language by solving the given question on C# specifications using Unity from data type, if else condition, looping, operator etc. Then, with the test code given from the teacher, then the student runs the answer given the question. **Figure. 4** shows the process

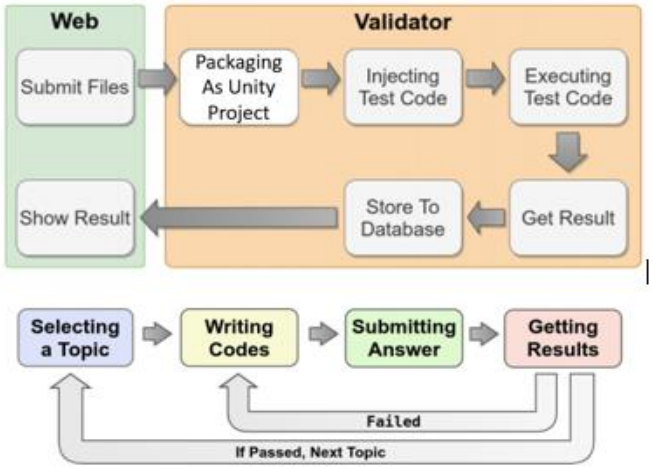

Fig. 3. Program Walidation Process.

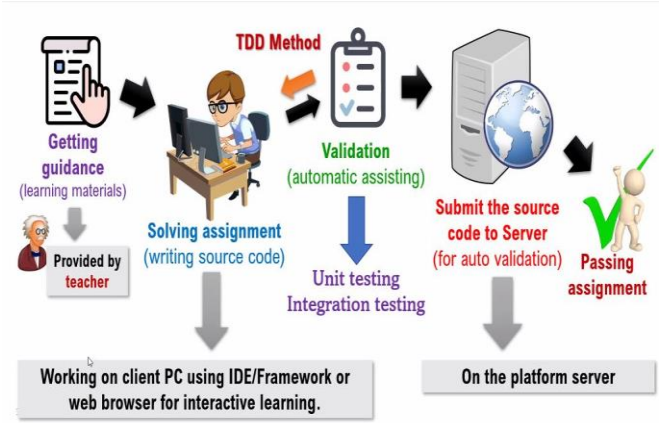

Fig. 4. Self Learning Process[17]

4.2 Unit Testing

There are three aspects that must be understood to start an Unity Game Engine, including 1) Start Unity 2D Project, 2) Configure unity test framework (Installing), 3) Design the UI. There

will be thirteen tasks in Table 1 that contain: one task for pretest, seven tasks focus on basic C# and basic Unity UI, five tasks that must be completed sequentially to build a simple 2D flappy bird game. When a student completes the entire task, student test the code using unity test runner. After that, all the test data will be recorded to database which contain 1) class name, 2) total test, 3) test passed, 4) test failed, 5) test date

Table 1. Learning Topics

Task No.	Title	Aspect
01	Introduction Unity + Hello World	Scripting
02	Pretest	Scripting, UI
03	Data Type + Arithmetical operation	Scripting
04	If Condition, Operator	Scripting
05	Button, Text Field	Scripting, UI
06	Button Toggle, Slider	Scripting, UI
07	Moving Object	Scripting, UI
FLAPPY BIRD		
08	Asset + Object	UI
09	Obstacle (Pipe)	Scripting, UI
10	Player Movement, sound management	Scripting, UI
11	Score manager	Scripting, UI
12	Spawning obstacle	Scripting, UI
13	Posttest	Scripting, UI

From the table of teaching materials above, the following will explain case examples regarding buttons and sliders. **Figure 5** shows the student code made by the students. While in **Figure 6** is a test code that has been made as a reference for the automation of assessment and checking. In the student code the variable used to display the slider is mySlider with the asset name being Slider which is public. The use of public is intended so that variables can be accessed by other classes so that further processing can be carried out.

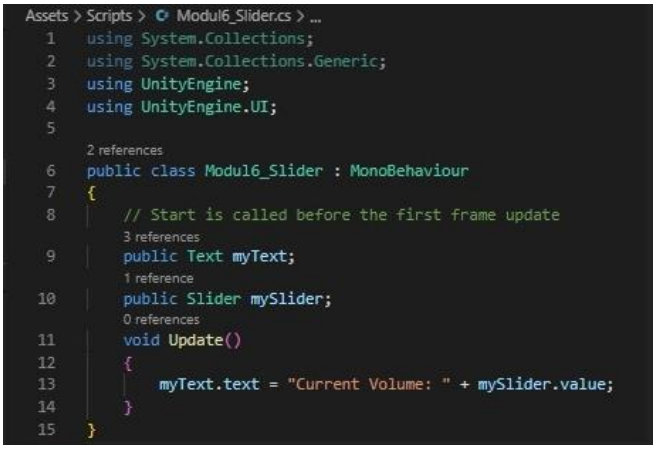

Fig. 6. Shows the test code that has been created.

In the test code, the Assert tool is used to call object components in other classes. In addition, assert also functions to check whether the syntax of the student code is in accordance with the rules of writing code or not. After checking by the test code, records in the form of processing time, error or correct results will be automatically stored in the database in real time. Meanwhile, in the student IDE, a form like **Figure 7** will appear as the output of the compiled source code earlier

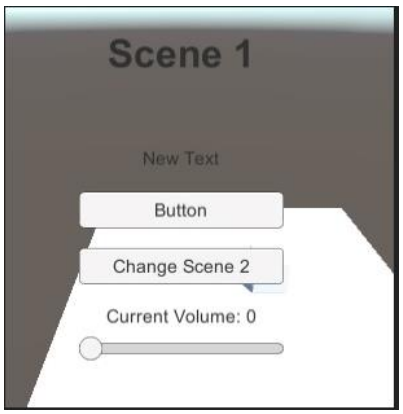

Fig. 7. Output Compile Task 06

4.3 Validation Process

A student who has completed the task based on Table 1 must validate it using the unit testing method in **Figure 8**:

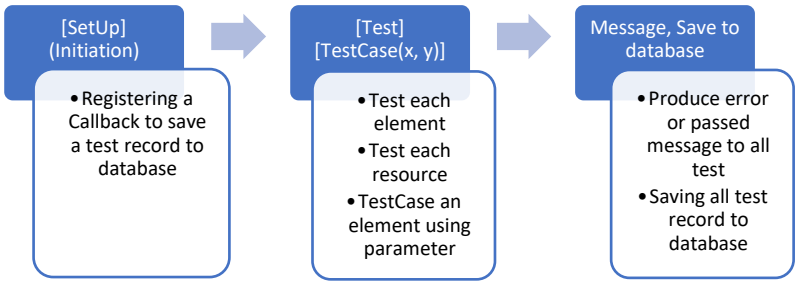

Fig. 8. Validation process

The process including: 1) Check the completeness of all the UI and scripting based on the given module for all topics. 2) Test script on each topic and some of the test using testcase. 3) If there is a failed test, a message will appear and all the tests will be saved to database. Figure 9 shows a graph of the results of testing the auto grading application in the independent programming learning process

nim	Total Test	Test Berhasil	Test Gagal	Nama Class	Tanggal Test	Score
1841720021	1	1	0	CubeTests MathTests2	2022-08-08 05:24:04	100%
1841720021	13	12	1	CubeTests MathTests2	2022-08-08 05:24:10	92.31%
1841720021	12	11	1	CubeTests	2022-08-08 05:33:09	91.67%
1841720021	1	1	0	CubeTests MathTests2	2022-08-08 05:33:24	100%
1841720021	12	11	1	CubeTests	2022-08-08 05:39:10	91.67%
1841720021	1	1	0	CubeTests MathTests2	2022-08-08 05:39:20	100%
1841720021	13	12	1	MathTests2 CubeTests	2022-08-08 05:40:52	92.31%
1841720021	13	12	1	CubeTests MathTests2	2022-08-08 05:40:52	92.31%
1841720021	14	12	2	CubeTests MathTests2	2022-08-08 06:03:28	85.71%

Fig. 9. Result Auto Grading Self Learning Programming

5. Conclusion

This paper presents a learning model that uses the Test Driven Development method to mark student answers automatically. Experiments confirmed the effectiveness of this learning model. Finally we were able to implement unit testing for self-learning authentication of Unity programming in real time. Student learning outcomes can be processed and analyzed further to obtain conclusions regarding the impact of using the application. As for future exploration, it can be aimed at learning 3D game programming. This is a challenge because in this case it requires a lot of game support assets and more complex source code.

References

[1] I. Otaduy and O. Diaz, "User acceptance testing for Agile-developed web-based applications: Empowering customers through wikis and mind maps," *J. Syst. Softw.*, vol. 133, pp. 212–229, 2017, doi: 10.1016/j.jss.2017.01.002.

[2] W. Novayani, "Game Genre untuk Permainan Pembelajaran Sejarah Berdasarkan Kebutuhan Pedagogi dan Learning Content," vol. 5, no. 2, pp. 54–63, 2019.

[3] R. Hidayat, "Game-Based Learning: Academic Games sebagai Metode Penunjang Pembelajaran Kewirausahaan," *Bul. Psikol.*, vol. 26, no. 2, p. 71, 2018, doi: 10.22146/buletinpsikologi.30988.

[4] U. Nurhasan, "Pembelajaran game menggunakan unity."

[5] E. Kucera, O. Haffner, and R. Leskovsky, "Multimedia application for object-oriented programming education developed by unity engine," *Proc. 30th Int. Conf. Cybern. Informatics, K I 2020*, no. January, 2020, doi: 10.1109/KI48306.2020.9039853.

[6] D. Winkler, R. Hametner, and S. Biffl, “Automation component aspects for efficient unit testing,” *ETFA 2009 - 2009 IEEE Conf. Emerg. Technol. Fact. Autom.*, no. October, 2009, doi: 10.1109/ETFA.2009.5347022.
[7] D. Toll and T. Olsson, “Why is unit-testing in computer games difficult?,” *Proc. Eur. Conf. Softw. Maint. Reengineering, CSMR*, pp. 373–378, 2012, doi: 10.1109/CSMR.2012.46.
[8] Y. W. Syaifudin, P. N. Malang, N. Funabiki, M. Kuribayashi, and S. E. Monitoring, “Learning Model for Android Programming Learning Assistant System,” no. December, 2019.
[9] D. A. B. Weikle, M. O. Lam, and M. S. Kirkpatrick, “Automating systems course unit and integration testing experience report,” *SIGCSE 2019 - Proc. 50th ACM Tech. Symp. Comput. Sci. Educ.*, pp. 565–570, 2019, doi: 10.1145/3287324.3287502.
[10] L. Végh and O. Takáč, “Teaching and Learning Computer Programming By Creating 2D Games in Unity,” *ICERI2021 Proc.*, vol. 1, no. November, pp. 5696–5700, 2021, doi: 10.21125/iceri.2021.1285.
[11] M. A. Umar and C. Zhanfang, “A Study of Automated Software Testing: Automation Tools and Frameworks,” *Int. J. Comput. Sci. Eng.*, 2019.
[12] A. N. Seshu Kumar and S. Vasavi, “Effective unit testing framework for automation of windows applications,” *Adv. Intell. Syst. Comput.*, vol. 174 AISC, pp. 813–822, 2013, doi: 10.1007/978-81-322-0740-5_97.
[13] N. Funabiki, R. Kusaka, N. Ishihara, and W. C. Kao, “A proposal of test code generation tool for Java programming learning assistant system,” *Proc. - Int. Conf. Adv. Inf. Netw. Appl. AINA*, pp. 51–56, 2017, doi: 10.1109/AINA.2017.60.
[14] Y. W. Syaifudin, S. Rohani, N. Funabiki, and P. Y. Saputra, “Blending Android Programming Learning Assistance System into Online Android Programming Course,” *2021 9th Int. Conf. Inf. Educ. Technol. ICIET 2021*, pp. 26–33, 2021, doi: 10.1109/ICIET51873.2021.9419650.
[15] Y. W. Syaifudin, N. Funabiki, M. Kuribayashi, and W. chung Kao, “A Proposal of Advanced Widgets Learning Topic for Interactive Application in Android Programming Learning Assistance System,” *SN Comput. Sci.*, vol. 2, no. 3, pp. 1–13, 2021, doi: 10.1007/s42979-021-00580-1.
[16] R. Bandara and I. Perera, “Unit Test Code Generation Tool Support for Lower Level Programming Languages,” *MERCon 2020 - 6th Int. Multidiscip. Moratuwa Eng. Res. Conf. Proc.*, pp. 1–6, 2020, doi: 10.1109/MERCon50084.2020.9185378.
[17] Y. W. Syaifudin, P. N. Malang, N. Funabiki, and M. Kuribayashi, “Learning Model for Android Programming Learning Assistant System,” no. December, 2019.