STUDY OF AN APPLICATION DEVELOPMENT ENVIRONMENT BASED ON UNITY GAME ENGINE

Sagor Ahamed, Anomita Das, Shahnawaz Md Tanjib and
Ms. Qamrun Nahar Eity

Department of Computer Science and Engineering, Ahsanullah University of Science and Technology, Dhaka, Bangladesh

ABSTRACT

In the current era of smartphone, mobile games have become really popular. To the high growth rate of mobile media, online games are getting progressively well-known and have been generally played, particularly among teenage-aged citizens. Our paper is about learning the development process of an ordinary online multiplayer game model and analysis of the complexity for its mobile option by several types of testing using Unity game engine. The emphasis is put on utilizing the inherent Unity segments in a multiplayer game in particular, building up accessory practices to utilize Unity's Scripting API for C# and incorporating outsider segments like the networking engine, 2D models, and molecule frameworks made for use with Unity and accessible through the Unity Asset Store. We choose to lead a convenient testing on the implemented mobile game application. We also make remarks on Unity as a multiplayer game improvement condition and execution engine.

KEYWORDS

Unity, mobile game testing, Multiplayer, Android Mini Game, Analysis

1. INTRODUCTION

Games and particularly online multiplayer games have been a gigantic piece of regular day to day existence. Worldwide acknowledgment, increasingly aggressive occasions, and prevalence increment have been ascending for a few years which likewise bring about quick advancement in the gaming business and greater amount of specialized aptitudes utilized to individuals concerning game improvement.

A mobile game application is an interesting form of entertainment that can be widely accepted by the user. For entertainment purpose, mobile game is one of the most favourite categories, as it provides fun and challenge at the same time. Between different genres multiplayer option is preferable and mini game is a potential choice without having to learn too many deep aspects [1]. Game engine allows developers to build a mobile game rapidly as the software is already equipped with the building blocks for mobile game app, such as collision and physics resources, 2D game assets, game renderers, artificial intelligence, visual effects, sounds, and other sub systems. The game developer can use the available assets from the game engine, develop and can add specific resources to the games, and finally bind them by writing program codes. However, the developers need to make sure the game can be distributed, deployed in the user's gadgets and run smoothly without errors or bugs which requires a mobile game testing process.

Testing is a process to detect the difference of developed software and the features requirements for that software [2]. For this testing process two commonly used approaches are black-box

testing and white-box testing. However, the approach to test a mobile game can be different from the common software.

In this study, our aims to learn the development process of the online multiplayer game on multiple platforms using unity game engine. And analysing the complexity with the help of different types of testing and scripting enabled between any four devices. So, the users of same or different devices running on same or different platforms can play with each other.

This paper is organized as follows: Section II gives an introduction of the background studies such as, Android Studio, Unity, PUN and the Google Play Games Services. Section III presents details about the mobile game application. In Section IV, we discussed about the various tests and analysis of the application. Finally, section V provides a conclusion and some slight details of future work.

2. BACKGROUND STUDIES

We have studied many topics beyond what we need to analyse and test. For analysis and testing we have studied unity, android studio and other essential components, which are described below.

2.1. Android Studio

The official IDE for Google's android operating system is Android Studio. It is a product made by Google through which one can make android applications. One can survey how the interface of application looks like and what changes are required as it comprises of realistic design. It provides various benefits when building Android apps to provide with automation testing services such as:

- It is a versatile Gradle-based system.
- It is a feature-rich emulator.
- Android studio is a uniform environment where we can develop for all Android devices.
- It has broad testing instruments and structures.
- Easy to assimilate GCM.

Android studio has a lot of features & functions, but for this experiment we have just learned and explored a few of them. Those are given below.

2.1.1. Android Alert Dialog

It is used to display the dialog message with OK and Cancel buttons also to break and ask the user about his desire to retain or discontinue.

2.1.2. Countdown Timer

It is used to set a countdown based on interval and it will stop when the time has come in future.

2.1.3. Bundles

Bundles are typically used for passing data between different Android activities. Depending on the types of need, bundles can hold all types of values to pass them to another activity.

2.1.4. Shared Preference

One of the most interesting data storage option in Android Studio is Shared Preferences. It allows activities and applications to save and recover small amounts of primeval data that will preserve even if the user closes the app.

2.1.5. Toast

Toast is an expedient way to display information or prompt messages for a time span to user. After timeout, it also disappears automatically.

2.1.6. Handling Click events in Button

In Android Studio there are two ways to handle the click event after the button click.

- Onclick in xml layout: If any user clicks a button, button object receives an on click event.
- OnClickListener: The click event handler can also be declared programmatically. This is the most preferred way because it can be used in both Activities and Fragments.

2.1.7. Supporting Different Screen Sizes and Pixels Density

As android runs on a variety of devices that have different screen sizes and pixel densities. We have used different drawable for adopting different screen sizes.

2.1.8. Android Studio Emulator

Android Studio has a default emulator that works just like a real Android device. So those who do not have an Android device do not have to face any problem.

2.2. Unity

Unity is a cross-platform game engine with a built-in IDE. It is the most powerful, full featured, business multi-platform 2D and 3D game development device made by means of Unity Technologies [3]. Basically, it is used for broadening video games for web plugins, computer platforms, consoles and cellular devices.

- It has the best community support.
- The Asset Store of unity is rich.
- It is a scripting Language.
- This game engine has the ability to create Multiplayer game.
- It is very easy to study and execute unity.

Like Android studio, Unity also has a lot of features & functions, but for this experiment we just learned and explored a few of them. Those are given below.

2.2.1. Graphics

Stunning images are provided by unity game engine. It also enables to facilitate the production importing assets from one-of-a-kind platforms. So that users must not wait lots of time with every import.

2.2.2. GUI

Creators with tools are provided by unity game engine to build its own graphics user interfaces, such as buttons and drop-down menus, sliders and manner of combining different interactable elements.

2.2.3. Scripting

Additional game-unique behaviours are required in video games. These behaviours may be implemented programmatically in unique components called Scripts by the use of Unity's

Scripting API (C#, JavaScript or Boo). The most common methods observed in Scripts are Awake(), Start() and Update().

2.2.4. Coroutine

A coroutine is like a function. An execution can be paused and control can be returned to Unity by this tool.

2.2.5. Canvas Scaler

This component is useful to control the overall scale and pixel density of UI. It also fits the smallest resolution of computer image. Though pixel size depends on user's screen resolution but pixel lighting is calculated at every screen pixel.

2.2.6. PlayerPrefs

It is a useful utility which is accessed like a dictionary (key/value pair) to persist certain types of data between scenes. It isn't encrypted, so sensitive things shouldn't be stored using this. It is very useful for storing user preferences or configuration.

2.2.7. Static Instance

This helps to access any function and public variable of this manager class from any other script.

2.2.8. Platforms

Unity is a cross-platform game engine and used to develop games for PC, consoles, mobile devices and websites, as it supports deployment to multiple platforms [5].

2.3. Photon Unity Networking

PUN (Photon Unity Networking) is a Unity package for online Multiplayer games which always connects to a dedicated server. It provides rooms, flexible matchmaking and in-room communication for players. It is very easy to use & export to basically all platforms supported by unity. It is available in two options, Pun free (20 CCUs) and the Pun Plus (100 CCUs). Our game uses the free version of PUN v2 [6].

Rooms and lobbies are two main concepts of PUN. Here connecting to a lobby means connecting to a session server, while connecting to a room corresponds to connecting a game server. Which means that a player must have to connect with a lobby first before connect to a room. The player joins a lobby after successful authentication. Then they can create or join a room. Photon server provides a means of sending data from one client to the other available clients and uses PUN to synchronize the game state between themselves. The Following synchronization mechanisms is used to achieve this communication: PUN RPCs (Remote Procedure Calls), Custom room properties and etc.

2.3.1. PUN RPCs

This provides an easy way to synchronize events between clients by allowing calling a script method on another client. This method calls on remote clients in the same room.

2.3.2. Custom Room Properties

It is used as key-value pairs to synchronize values that do not belong to any objects. It shows a way to share values across room that player is currently in without having to have any reference to a specific item. Player's icon can be set in a custom property and other players could grab that custom property without having any reference to this player.

2.3.3. RaiseEvent

RPCs (Remote Procedure Calls) needs a PhotonView and a method to be called as RPCs aren't exactly what is required for game events. So, it is used to make up own events and send them without any relation to some networked objects.

2.4. Play Games Services

Some elements such as, high scores, cloud save, leader boards, quests, and other supports are required for the implementation of developing a game for the Android platform. Adding these features to the game will help keep players more engaged. Instead of doing it manually, it can be done with Google Play games services APIs.

Google Play Games Services [7], Provides a range of ready to use features that can be added to a game by using a simple API. In addition, the APIs and tools will reduce the burden of testing, managing, and publishing a game. The comprehensive reporting and statistics will track and improve games performance.

2.4.1. Sign in

In order to access Google Play Game Service functionality, a game desires to offer the signed in player's account. If the player isn't authenticated, the game may encounter errors when making calls to the APIs.

2.4.2. Unlocking Achievements

With achievements, players may be retained by way of adding rewards for engaging in set goals in the game. Players can earn experience points for conducting achievements. It can be a brilliant manner to grow the user's engagement with game.

2.4.3. Posting score to leaderboard

Players get a scope to evaluate their scores with friends and compete with eminent players with leader board.

2.4.4. Saved Games service

By using this the game can retrieve the stored game records to permit returning players to continue a game with the last gained games data such as scores, levels, etc. from any device. It helps to concur a player's game data via multiple devices and ensures that players can continue game play from where it left off.

3. EXPERIMENTAL DEVELOPMENT OF AN APPLICATION

Description of features and functionalities of the target system that can be hidden or obvious, expected or unexpected, known or unknown from client's point of view is needed. So, after the background study we understand that Unity is specially developed to be good at handling graphic

assets, animations, sound assets etc[4]. Which also allow to run the game inside the game editor. Developer can also change the properties of the game assets directly and view the results immediately, even when running the game. But, for building a game on a tool like Android Studio, we need to develop most of such possibilities by ourselves. As Unity provides more flexible & better performance for android game development than Android Studio. That's why we decide to keep our testing and analysis progress only on Unity. So, we've implemented & improved our game by adding different kinds of features in unity which helped us in the analysis and testing part. The detailed description of our game is given below.

3.1. Summary of the Experimental Development

- **Game Name:** Catch the Culprit.
- **Game Genre:** Card.
- **Platform:** Android.

3.2. Game Characteristics

Catch the Culprit is a popular local game of Bangladesh which is played by every people of the country in their childhood. It is typically known as "Thief, Robber, Master and Police". The game consists of four players and there is a card for each player from where each player has to pick one card. The interesting fact here is that in the first round who gets the police card will have to predict the thief. For the next round the police would have to assume who is the robber among the players. If a police card player makes the right prediction, he will get points, otherwise the other player will get points. In this way the game will continue in phases. The players can take three types of challenges in the game: Time challenge, Score challenge and Level challenge.

3.3. Graphic Concept Art:

Some of the graphic concept art used in the game are discussed in (e.g. Table 1).

Table 1. Game Character Designing and Modelling.

Description	Image
These icons appear according to the challenge type when a player starts playing a single challenge. It indicates the time, level, score that has flowed.	 Figure 2. Indicators.
These are the different methods for coin collection which is required for multiplayer game play.	 Figure 3. Different ways for collecting coins.
Different types of cards which will be revealed in the game during game play. The first card appears in the row represents the back of all the cards. The second card is the unknown card which appears when a player gets thief or robber card. Then comes respectively the master card, the police card, the robber card and the thief card.	 Figure 4. Different types of cards.

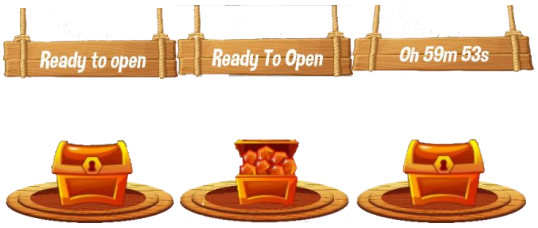
This wheel indicates the number of player's turn. The wheel of serials helps to decide which player will play first.	 Figure 5. The wheel of serials.
While playing each player can send emoticons to others. Also, some conventional words are included here with the emoticons which are commonly used during play.	 Figure 6. Emoticons box.
Individual game details will be displayed here when the game starts. Such as, Name, Serial, turn time, Emoji's send button.	 Figure 7. Profile indicators.
The treasure box is for collecting coins. When the box is ready to be opened, the player can collect the coins by clicking on the box. Then a random number of coins from the box will be deposited with the total coins. After collecting the coins, the box will be closed again for one hour.	 Figure 8. Phase of coin collection

Table 2. Animation.

We have applied animation to the treasure box, coins and cards using JavaScript's C\# and sprite sheet. We can use transition animation, assign keys to open the treasure box, movement of the coins or card flipping.

Table 2. Interface.

The menu was designed in Adobe Photoshop and animation. There is a background sound for the game which can be turned on or off by the user

3.4. Core Idea of the Game

A player can play the game in three ways.

- Multiplayer: A real-time turn-based play on the server with random / selected opponents on their own respective devices.
- Single: An individual play with computers able to pick random moves and decisions.
- Four Player: Four players involved in one game playing in the same device.

The flow chart of the working game is presented in Figure 1 below.

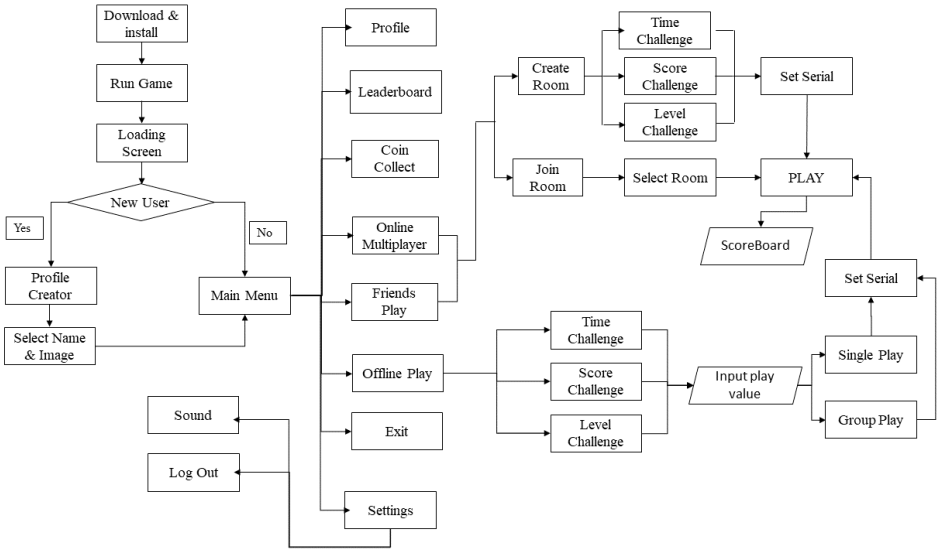


Figure 1. Game flow chart

3.5. Game Development

We tried to create a native card & board game as our experimental development. Detailed description of our Game UI and Gameplay is given below.

3.5.1. Game UI Screen

Start Menu

This is the first interface when the game is executed. There are about fifteen characters from which a player can select his desired profile picture. He has to input what nick name he/she wants to use for playing. After saving information's another screen appears which is called main menu.

Figure 9. Game Menu.

Main Menu

This is the main menu interface of the game (Figure 9). Player can update his profile name and image, collect coins, rate the app and can proceed to play the game in different modes from here.

Total earned coin by the player will always be appeared on the right side of the screen. The star option which is placed above it, is for storing multiplayer game scores only. None of the offline play scores will add here.

Profile

From the information of Start menu, the name and character image of the player appears on the top right. If the player wants to change the name or image given earlier, he has to click on the image where he can update the name and image. Above the local name, another name appears which is given by the Google Play games services.

Online Multiplayer

In online multiplayer game, a player can create a room or can join in an existing room. The master client needs to select a playable mode and the game's play range. After joining 4 members in the room, the master client can start the game (Figure 10).

Offline play

In offline play, the player has to select if he wants to play with CPU or four friends. If he selects the CPU, he will start playing as player1 where the remaining three competitors will be computer controlled and will do random actions. If he wants to play with his friends using same device, he can select 4 players.

Settings

There are two options here. Using the first option, the player can turn on/off the game sound. And another is for Logout from the GPGs sign in.

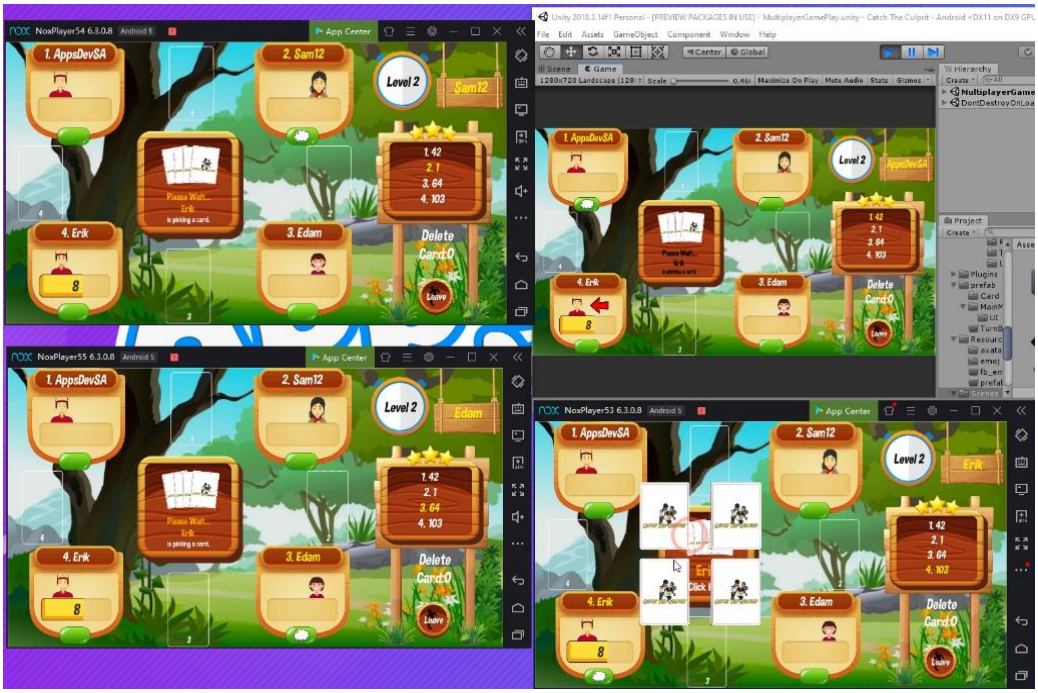


Figure 10. Online multiplayer game play.

Score Board

The score board which appears after finishing the game. Here all the players are placed in ascending order according to their score. The interesting fact is, every player can see their score highlighted in their own screen (Figure 11).

Figure 11. Final Score Board.

Leader Boards

The button that placed before the treasure box shows how many times the player has taken the time challenge, score challenge and level challenge. There is another button named leader board where the player's highest earned scores can be seen. By clicking the achievement button, player can see the unlocked achievements.

3.5.2. Gameplay Description

Detailed description about gameplay is given below.

3.5.2.1 Multi Player

For playing multiplayer games, the player who creates a room or starts the game is called master client. Clicking on Online Multiplayer button will bring up two options, Create Room and Join Room. If the player clicks on the option named Create Room, the next appeared screen will ask him to select the game challenge type. The players can take three different challenges.

- **Time challenge:** In this challenge, a certain period of time will be determined by the players. Whoever gets the highest number after the end of certain time will win.
- **Score challenge:** In this challenge, the highest achievable points are assigned by the players. The player who first earns the assigned points wins.
- **Level challenge:** In this challenge, the player has to determine how much level he is willing to play. The player who gets the highest number at the end of the specified level will win.

After selection, a master client has to fill up the text box with his expected time, points or level. Then the interface of room creation will appear where the player has to enter the room name and thus the room will be created. Then, the room name will be displayed to other ready players [9]. Other players can join the room in three ways. One can join room by choosing from the room which is available in the rooms list or by manually type the name of the created room. The other way is by clicking on the "Join random" button. Then if the master player gets four players who

have joined his room then he can start the game. After that, master client spins the wheel of serials which will determines the serial number of the players. Then the game will begin.

After that, the gaming interface will appear. Each player has their turn highlighted on the interface so they can easily understand when their turn comes. Every time an arrow indicator comes up to the screen to show whose turn it is. The game will automatically start after few secs. There is a card for each player. The cards will be randomly arranged in the game and will be shuffled again after completion of a round. When the game starts, 4 cards appear in the middle. Players can choose any of them. Once a player selects his card, the card will be displayed on the screen to every player if the player gets the Master or the police card. Thus, the rest will select their card. One card reduces each time. The other two cards except Master and Police will be invisible to all but the player himself can see his card. If a player does not play his moves within 20 seconds then auto move will be activated. A card will be randomly selected for that player.

In the first round who gets the police card will have to predict the thief. If the prediction is correct, police gets 80 points and the thief will get zero. If the prediction is incorrect vice versa would happen. For the next round the police would have to assume who is robber among the players. The player with the master card will get 100 points directly. The points fixed for the thief and robber is respectively 40 and 60. If the player who got the card of police doesn't select anyone on time, he will get minus 50 points. In this way the game will continue in phases. The highest scoring player in the first level will get the chance to play his move first in next level.
The final score board (Figure 12) will come up when the game ends, where the highest scorer will be on the top and the rest of the players will be there in ascending order according to their scores.

3.5.2.2 Single Player

When this option is selected, a screen appears with three options named time, score and level challenges. The player has to input the expected time, score or level to be played. Then the game will start with three opponents controlled by the computer and they will perform random actions.

3.5.2.3 Four Player

Three options like Single player appears for this option also. But the difference is in this mode player can play with real players in same device.

4. EVALUATION OF APPLICATION

Nowadays smartphones have become a part of our everyday activities, so testing mobile apps has become important. Testing is a process to verify the features of a software and is done to show that the system is working or not. Although it is expensive and time-consuming, it ensures our customers have a positive experience when using our mobile app. In general, software testing is a set of activity conducted to check whether the actual results match with the expected results and confirming that the software component is defect free. The software which is currently under testing must provide all the defined functionalities and match the required workflows. Between different testing approaches, two commonly used process are black box testing and white box testing [8]. However, mobile game testing can be different from the software testing.

Mobile game usually builds of interacting sub systems and continuous game loop. Mobile application testing is procedure to test functionality, usability and consistency glitches of an application software developed for handheld mobile devices. In this paper, we conduct a series of activities for mobile game testing. As we develop the game on unity game engine, the study utilizes the testing "Unity Test Runner" available in the engine. Unity testing will identify if there

exist errors in the structural level of the code. Our primary goal for this testing efforts was to find the existing errors in the game.

Factors of Mobile App Testing

- Emulators
- Real Mobile devices.

4.1. Types of Mobile Apps Testing

For game testing several techniques can be applied. But, as we are specifically focusing on mobile games testing, the types of testing methods are shown in.

4.1.1. Unit Testing

We design a unit test to test a single unit of code. And we also determine that, whether it behaves exactly as same as our expectation for small, logical, snippet of codes. In unity, Unity Test is performed through Unity Test Runner.

4.1.1.1 Unit Test Runner

The component provides test runner to execute test case and report the results (Figure 12). This tool can be used to test code in both Edit mode and Play mode. This page [10] helped to us get started and understand the basics Unit Testing concept.

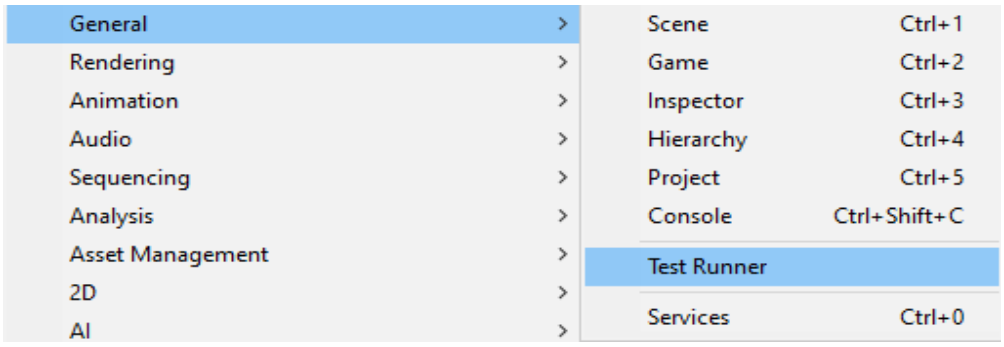


Figure 12. Unity Test Runner.

4.1.1.2 The Experiments

The tool exploration has been performed on the game source code and makes it as the test subject. Normally test are divided into some parts. The general structure of this test is given below.

- Arrange: In the arrange part we prepare the tests. And setup the prime configuration there so that we can create dependent object and defined the preconditions and inputs.
- Act: In the act part, we called the methods of the test class.
- Assert: In the Assert phase we ensure that our expectations are met. We assert that the expected results have occurred in the range part of our test.

The source code that has been modified using the above-mentioned rules in order to generate test case.

After applying the rules to the source code, the next steps are:

- Instantiate the tested class.
- Instantiate the Test Suite (set of test cases) class.
- Initialize the variable value.
- Call the tested method, which is Start method.
- Use Assert related functions to check if the expected result equals the real result.

The environment for testing process is as follows:

Hardware specification:

- Processor: Intel(R) core (TM) i5-5200U CPU @ 2.20 GHz.
- Memory: 8.00 GB RAM.
- SSD: 256 GB.

Software specification:

- OS: Windows 10 Education.
- Tools: Unity, Unity Test Runner, Visual Studio, Nox player.
- Programming Language: C#.

4.1.1.3 Results & Discussions

We reviewed each class and methods and prepared some methods for the test cases. The criteria for selecting the methods is on the fact that other methods interact with game engine sub systems. As because the dependency with other game resources and results domain is very large these methods can't be easily tested using the Test tool. The process of testing the test cases: The authors can conclude that the methods are bugs free as marked by the green check mark in the testing results.

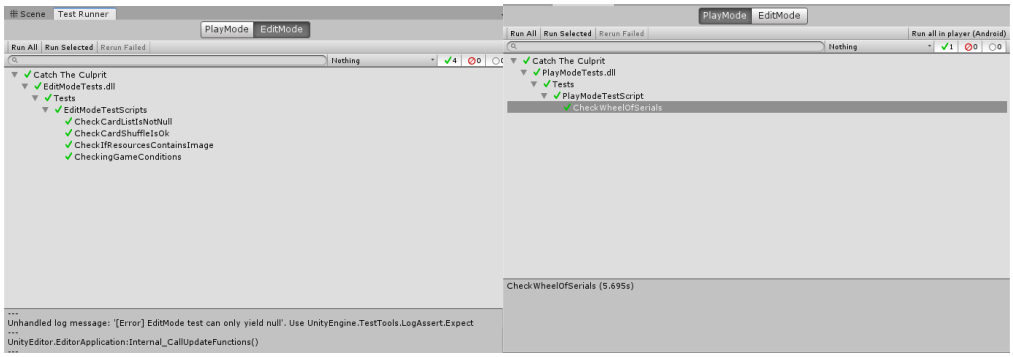


Figure 13. Test cases run on Unity Test Runner.

The authors also tried to test whether the test cases are reliable or not by entering inputs that do not match for the required condition during the testing process. And Test runner shows that the tested methods was not bugs free.

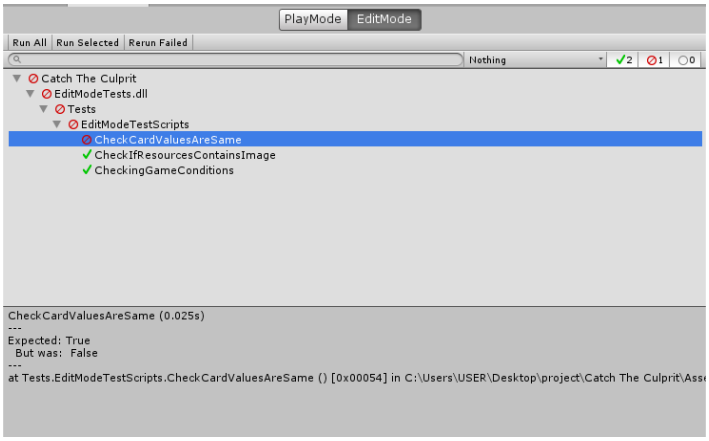


Figure 14. Test cases run on Unity Test Runner with errors condition.

4.1.2. Functionality Testing

It's a type of testing to ensure that mobile applications are working as per the requirements. So we did it by inserting the input and examining the output. Some examples are:

- We have ensured easy login using GPGs.
- Test to make menu options easy to find.
- We test it to see if the app works well with a smartphone screen keyboard.
- Manage data handling, storing and avoid data deletion. And many more.

4.1.3. Regression Testing

This test is basically, re-testing unchanged parts of the app. During the entire testing process, we make sure of that a recent code change has not adversely affected existing features. Such as, we have ensured that new code changes do not cause the misrepresentation / erroneous behaviour of the existing code.

4.1.4. Performance Testing

It is performed to uncover what needs to be improved before releasing to the market and determine how a system performs in terms of stability and responsiveness. Such as,

- We make sure that while rotating the screen, only landscape mode works, as the game's UI is preferable for that mode.
- We also make sure that; the game's menu was working properly.
- Multiplayer game was also properly working.

4.1.5. Usability Testing

According to reviews from Play Console we have found that our UI provides users with a satisfying experience. Also, we can say that the game is easy to play, because the game has one tutorial round that will show the user how to play the game.

4.1.6. Compatibility Testing

It is a testing of the application in different mobiles devices and checking if it supports different size, resolution, screen, version according to the requirements. Some of the following basic scenarios are considered when performing this test in the game:

- No text is cutting off and text is readable.
- Sound can be on/off.
- All types of timer used in the game can be suspend/resume/start.
- Check if app network is not available during multiplayer game starting.
- Behaviour of app if is running for longer period of time.

4.1.7. Localization Testing

We check the game to assure that it behaves according to the settings. Which ensures that the apps are capable enough for using in that particular country and the major area affected by it includes content and UI.

4.2. Black Box Testing

The paper referred at [1] conducts some testing by the author to see which bug or error appears. Similarly, we also tested the entire game with a group of 4 members. We performed 10 tests in total of playing out a full game play. Some of the tests were also recorded, one of which was uploaded to YouTube [11]. We made sure that application is working as per requirements and verify all the functionality. We also check if it supports different mobile screen sizes or not and menu, buttons, settings. Several issues were discovered during this testing process:

- Starting Multiple Coroutine was creating problems during Multiplayer game play.
- GPGs Leaderboard / Achievement was not working.
- During Time Mode game play, time was working only on master client Device.
- After leaving a room, after creating the new one, it was not visible to other players.
- The name of the room shows up after leaving the room but players can't re-join.
- During multiplayer game play problem was encountered in case of adding scores.
- During the multiplayer game play sometimes, it auto leaves the room and then re-joins.
- The animation stopped working suddenly due to a new plugin being inserted.
- Sometimes rooms is not shown to other player but random manual join perfectly works
- The wheel of serials (Roulette) was showing 0 in all positions.
- Player profile pic was showing in incorrect order.
- Score wasn't in sorted order in game score board.
- Turn was not working perfectly according to highest score.

Although in some places, we faced problems in understanding that how the function works. But by searching online and after several iterations we have been able to solve the problem successfully.

4.3. Problems Encountered During Testing

During testing some bug or error appeared which are shown below.

4.3.1. Problem 1: APKs or App Bundle are available to 64-bit devices, but it only have 32 bit native code: 16

What is 64-bit requirement? According to Google's announcement starting from August 1, 2019 every published apps on Google Play will need to support 64-bit architectures (Figure 13). Some of the reasons mentioned by them are:

- CPU's deliver faster.
- User's richer experiences.
- Improvements of performance.

And full details are mentioned in the following URL: [12]
Solutions: After researching a lot we found some preferable solution which are described as follows.

Going to "Other Settings" from "Player Settings" which is inside "Build Settings". Change "Scripting Backend" to "IL2CPP", and we make sure that all boxes under "Target Architectures" are checked (Figure 15).

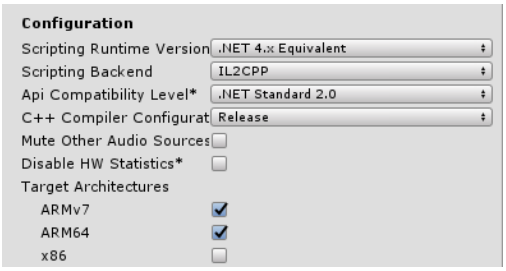


Figure 15. Unity's others settings for android.

Now, as of Unity 2018.3.14f1 (version which we are using for this game), there's an option in Build Settings to export to an App Bundle, this is the method we have followed and this is the most recommended suggestion for moving forward (Figure 16).

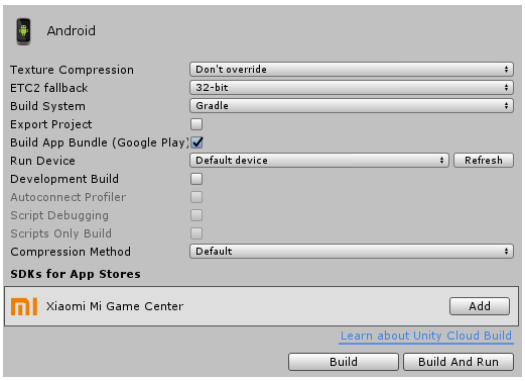


Figure 16. Unity's build settings window.

The work is not done here because next we installed Android NDK (it wasn't installed before). Most of the time, we found solution in online to do it through Android Studio, but when we installed a version, Unity apparently didn't like it, maybe that package wasn't compatible with our machine. So, we downloaded the Ndk and setup the NDK path's as below (Figure 17).

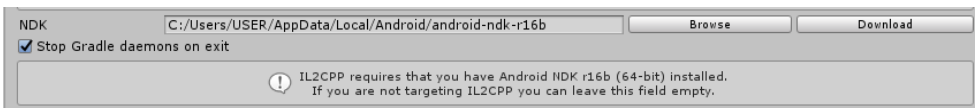


Figure 17. Unity's external tools window.

After that we have successfully built the apk which took almost 2-3 times longer to build than usual. But doing all this thing we were still facing problem to upload it. Lastly after Removing x86 from the Player Settings allowed us to upload it to Google Play. Because of we have selected "Build App Bundle (Google Play)" in the Build Settings, we got an 'aab' file for the Google Play Console which worked the same as an APK

4.3.2. Problem 2: Google Play Services "Login" not working

As for our game we have decided to implement some features of Google Play Games services. When the login was called it showed the GPGs screen trying to connect and then closed and failed. But the internet status was fine and plugin importing was also successfully done.

Solutions: Going to the path: Google Developer Console > Game Services > Our Application > Game Details > Api Console Project. Where our game 'Catch the culprit' (which is basically our game name) is linked (Figure 18). By clicking on app name this take us to Google developers Api console.

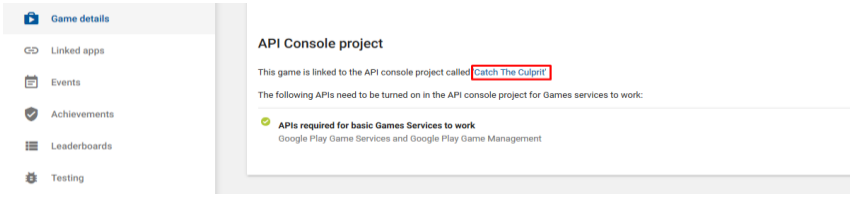


Figure 18. API console project.

Then, clicking on Credentials > Edit (Figure 19) we found an SHA1 key which is wrong for our project.

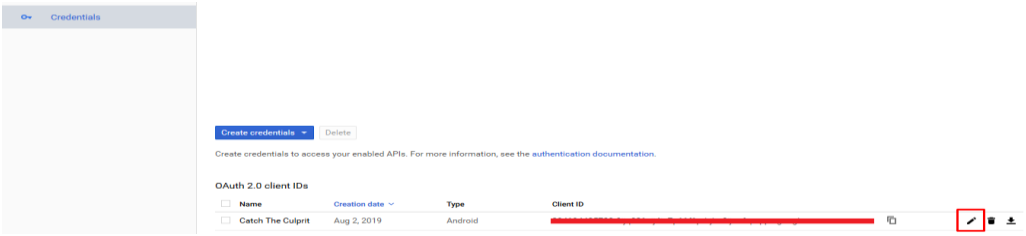


Figure 19. Credentials Window.

Then we create the correct SHA1 key using the CMD code mentioned there in upper position for generation of SHA1 key. And after that we changed our SHA1 key in Api console.

This login is working perfectly only after building directly on android. When we published it on playstore it was creating the same issue again, when we try to test it downloading from playstore. After researching we get to know that from online, when an apk is uploaded, the SHA1 is used of the Uploaded Version. But Google gives a different SHA1 after uploading an apk for playstore. So need to replace this "Upload certificate" SHA1 with the "App signing certificate" SHA1 from credentials window.

How App Signing by Google Play works? We get to learn more about from the following URL: [13] about how it works.

Here is the explanation of how we solve it, Going to this path again: Google Developer Console> All Applications > Our Application > Release Management > App Signing. There we find two certificates, and we copy the SHA1 from the APP SIGNING CERTIFICATE. Again going to the same path: Google Developer Console> Game Services > Our Application > Game Details > Api Console Project (Figure 20) we replace the "Upload certificate" SHA1 certificate with the "App signing certificate" SHA1 and make sure that our package name is correct.

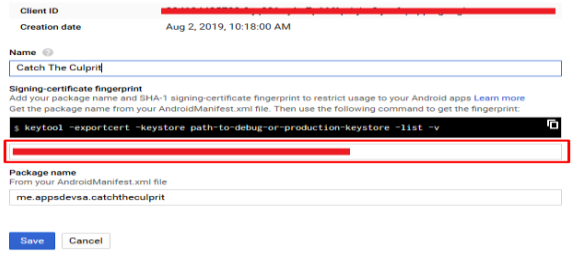


Figure 20. Credentials Window.

After a couple of minutes, it started to work fine.

4.3.3. Problem 3: Particles effects are not showing in Editor Window

This problem was small but annoying while, we are trying to implement particles effect for animation purpose and the particles are not showing in the canvas.

Solutions: After a lot of searching we found a preferable solution, which is our main camera's render mode was “Screen Space-Overlay”. So, we select "Screen Space - Camera" from the dropdown menu and assign a check mark to the “Pixel Perfect” option and drag the Main Camera to Render Camera as shown in the screenshot below (Figure 21).

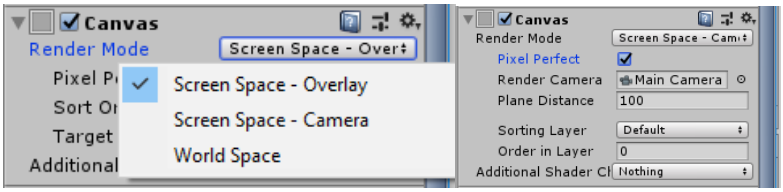


Figure 21. Unity Inspector Panel.

As a result, particles effect starts to showing in the canvas.

4.3.4. Problem 4: Cannot authenticate with .EnableSavedGames

We were unable to authenticate when we were using the PlayGamesClientConfiguration with ".EnableSavedGames" feature, because it was causing the app to hang or stuck during playing. It also seemed to start authenticating and displays the Google Play Game Notification logo on the top when we were trying to run the game.

What is Enable Saved Games in GPGs? We learn it from GPGs [14]. The saved games service gives a convenient way to save player's game progression to Google's servers. This service is great for retrieving players' stored game data from any device.

Solutions: The prerequisite for using the Saved Games service is to have it enabled in Google Play Console. So, for that we browse to the Google Play Console and select our game from the list from the Games Services tab and turn on the "Saved Games" service. Then we click save and publish the game changes. After all the above steps are done, the Google Play Games service's Saved game feature takes almost twenty-four hours to activate for our game's.

But we had to do something further too, to test the Saved Games feature immediately. We opened the Google Play Services app on all testing devices and manually cleared the data from the app. To clear the cached data on Android the step is as follows,

Open Settings > Apps > Google Play services> Manage Space > Clear All Data.

We encountered these problems during the user testing and were able to solve them all successfully. No further significant issues were discovered, except only minor performance related issues.

5. CONCLUSIONS

Our aim was to learn about the platform in which online multiplayer game making is relatively easy and convenient as it makes all possible resolutions from both developer and user side's data. We tried to learn about the various features of Unity, how it works and how it differs from Android Studio. We also tried to find out all possible differences among them with analysis and testing. In this paper, our main focus was to learning the process of developing turn based multiplayer games on Unity Game Engine using PUN, GPGs and finding out all possible complexities with analysis. Though there are few limitations in our game, in future we will renovate our game remarkably. As of now this game is played randomly by creating rooms. But in future we will focus on attempts to further extend the function of the game lobby, such as adding chat-rooms into every Room and add Facebook login, invitation, share and some other features. Adding this, in future we have a plan to release this game in iOS, WebGL and Xbox version. We have uploaded it in play store and are collecting user data like review and downloads. We are promoting it to social media and collecting data. In this way we can find the problems if any as well as the opinions of users that will be more helpful to fixing minor bugs.

ACKNOWLEDGEMENTS

We would like to pay homage to our supervisor, Ms. Qamrun Nahar Eity, Assistant Professor, Department of CSE, AUST, for giving us the opportunity to work in our area of interest. We have felt privileged to have her throughout the journey. We would be happy to express our deep gratitude for her generous support and guidance. Conclusively, we are also appreciative to our respective parents, for supporting us in our journey during the course of preparation.

REFERENCES

[1] Y. Juan, T. A. Geumpana, and J. J. Martinez, "Developing a game application to encourage face-to-face local gaming experience," in 2016 1st International Conferenceon Game, Game Art, and Gamification (ICGGAG), pp. 1–6, IEEE, 2016.

[2] A. C. Barus, R. D. H. Tobing, D. N. Pratiwi, S. A. Damanik, and J. Pasaribu, "Mobile game testing: Case study of a puzzle game genre," in 2015 International Conference on Automation, Cognitive Science, Optics, Micro Electro-Mechanical System, and Information Technology (ICACOMIT), pp. 145–149, IEEE, 2015.

[3] "Unity user manual." https://docs.unity3d.com/Manual/ UnityManual.html. Accessed: 2019-12-10.

[4] T. Finnegan, Unity Android Game Development by Example Beginner's Guide. Packt Publishing Ltd, 2013.

[5] "Unity supported platforms." https://unity3d.com/unity/features/ multiplatform. Accessed: 2019-12-10.

[6] "Photon unity networking." https://doc.photonengine.com/en-us/ pun/current/getting-started/pun-intro. Accessed: 2019-12-10.

[7] "Play games services." https://developers.google.com/games/ services. Accessed: 2019-12-06.

[8] I.Kuswardayan and R.R.Hariadi, "Design of mini synchronous game food fest in social game food merchant saga on android devices," in 2017 International Conference on Advanced Mechatronics, Intelligent Manufacture, and Industrial Automation (ICAMIMIA), pp. 342–345, IEEE, 2017.

[9] M. Wang and L. Zhu, "Designing and implementing an online card game based on android 2d graphics," in 2014 International Conference on Audio, Language and Image Processing, pp. 817–821, IEEE, 2014.

[10] "Unity test runner." https://docs.unity3d.com/2018.1/ Documentation/Manual/testing-editortestsrunner.html. Accessed: 2019-12-28.

[11] "Online multiplayer game play testing." https://youtu.be/kGqKNGxPGHw. Accessed: 2020-01-29.

[12] "Android supporting 64-bit architectures." https://developer.android. com/distribute/best-practices/develop/64-bit. Accessed: 2019-12-15.

[13] "Use app signing by google play." https://support.google.com/ googleplay/android-developer/answer/7384423. Accessed: 2019-12-15.

[14] "Google play games plugin for unity." https://support.google.com/ googleplay/android-developer/answer/7384423. Accessed: 201912-01.

AUTHORS

Sagor Ahamed is a student at Ahsanullah University of Science and Technology, Dhaka, Bangladesh where he is currently an undergraduate student and studying in B.Sc. Engg.(CSE).

Anomita Das is a student at Ahsanullah University of Science and Technology, Dhaka, Bangladesh where she is currently an undergraduate student and studying in B.Sc. Engg.(CSE).

Shahnawaz Md Tanjib is a student at Ahsanullah University of Science and Technology,Dhaka, Bangladesh where he is currently an undergraduate student and studying in B.Sc. Engg.(CSE).

Ms. Qamrun Nahar Eity is an assistant professor of CSE at Ahsanullah University of Science and Technology, Dhaka, Bangladesh.