CROSS-PLATFORM MOBILE APPLICATION TESTING

Anna Latonina

CROSS-PLATFORM MOBILE APPLICATION TESTING

Anna Latonina
Bachelor's thesis
Autumn 2014
Business Information Technology
Oulu University of Applied Sciences

ABSTRACT

Oulu University of Applied Sciences
Degree in Business Information Technology

Author: Anna Latonina
Title of Bachelor's thesis: Cross –platform Mobile Application Testing
Supervisor: Matti Viitala
Term and Year of Completion: Autumn 2014
Number of pages: 37

This bachelor thesis describes multi platform mobile application testing process. Marmalade Software Development Kit was used as a primer environment for development and testing process.

The thesis is divided into three major parts. The first describes development environment and target platforms. The second part introduces testing basics. The third part describes actual process of testing of the application. Arcade type game Glow Hockey 3D was a part of research part. This application was tested from the beginning of the development process and is still monitored.

The application was successfully released on Google Play market and received mostly positive reviews from the critics. Post release problems appeared, however, their number was not great. As a result, multiplatform testing and development saved a lot of time and money since there was no need to double the application on another platform with different environment.

Keywords: testing, mobile application, game, cross-platform, Marmalade SDK

CONTENTS

1 INTRODUCTION

Mobile phone began playing an essential part of life by performing all kinds of services. Mobile applications increase the interest towards smartphones and simplify the everyday activities, for example such as entertainment, sport actions and time management. The incredible growth of mobile devices and wide range of operating systems lead to the need of establishment of cross-platform development tool which will simplify the delivery of applications for different mobile platforms at one point of time.

The range of products as well as high competition has also contributed to the need for good quality applications. The vital part of application development is testing, which leads to the reliable and robust products. Quality assurance takes almost half of the time in development process on Android, while the major time in development process on iOS takes programming itself. By using multi-platform development tools it is possible to reduce the time spent on coding and constrict possible errors in code as well decrease the cost of application development.

1.1 The objective and purpose

The main objective of this thesis is to describe testing process of cross -platform application and the theoretical part, which provides the overview of the methodology. A variety of operating systems and hardware differences make this part of process very wide. While the testing strategies and types are different, the focus is on the cross- platform application quality assurance in general and on the Android devices in particular. Different types of testing processes are also discussed and described as a part of quality assurance of the case.

1.2 The structure and content

The thesis will be divided into three main parts. The first part introduces cross- platform development tool concept and the Marmalade SDK particularly. The chapter will introduce capabilities and goals of this kind of platforms, as well the advantages and disadvantages from the quality assurance point of view. Furthermore, it includes a description of its target operating systems. The emphasis is put on Android and iOS as widely used operating systems for

smartphones and tablets. Description of testing environments of Android and iOS is also included in the theoretical part of thesis.

The second part contains some main aspects and defenition of testing. Quality assurance is an indispensable part of development of application, which improves altogether the usability, control and quality of the mobile application. It describes testing types that could be used during multi-platform development.

The third part briefly introduces testing methods were used and the reason they were selected during development part, chosen software environment and bug management. The main idea of quality assurance is to reduce time spent on developing the application, improve the user interface, and decrease the post-publishing costs. The arcade game was chosen as a model for testing and developing, because this kind of game is one of the most frequently downloaded application types.

2 CROSS PLATFORM SOFTWARE

Marmalade SDK is multi-platform tool to develop applications on different operating systems. It provides a possibility to use one code to deploy an application on numerous devices. C++ programming language is used mostly, and in the model application particularly. However, it is also possible to program in Lua, HTML and Objective –C. Marmalade SDK is conformed to an integrated development environment, for example Xcode on Mac's or Visual Studio on Windows. (Marmalade Technologies Ltd 2014, cited 2.12.2014).

Smartphone manufacturers use mostly ARM processors in their devices. It is possible to compile the application on ARM, which will be perfectly launched on smartphone devices as well on x86 so the simulator could be run on the computer. (Scaplehorn 2012, 23.) Simulator is very important built-in tool in testing activities.

2.1 Marmalade Target platforms

Marmalade SDK has few target platforms on smartphones and tablets, namely Android, IOS, Blackberry, Tizen, Windows Phone, LG. It also supports Windows and Mac on desktops. The smart TV is rapidly developing market and with Marmalade SDK it is possible to create application and programmes that can run on large screens of LG Smart TV and Roku. (Marmalade Technologies Ltd 2014, cited 2.12.14.)

The most popular operating systems are Android and IOS on smartphones and tablets. According to International Data Corporation, in third quarter of 2014 around 327.6 million smartphones were sold globally, and especially popular were Android and iOS. The biggest vendor is Samsung with 78.1 million sold devices based on Android operating system. The second one is Apple with 39.3 million sold devices with the release of a new Iphone 6 and Iphone 6 plus. The two most popular operating systems on smartphones and tablets are among target platforms of Marmalade SDK. (International Data Corporation 2014, cited 30.11.2014.)

In the third quarter of 2014 the downloads from Google Play where 60% higher than from App Store. However, the revenue of App store is still much higher than Google Play's one. The

biggest market of App store is The U.S.A. and the second biggest purchasing country is Japan. Applications from Google play are also downloaded mostly from the U.S.A., but the top purchasing country is Japan. In both stores the most downloaded applications are games, this kind of applications also bring the most income among possible categories. (App Annie 2014, cited 2.12.2014.)

2.1.1 Android

Android is an operating system, which is based on Linux kernel and is meant for mobile devices, like phones, smartphones, tablets, smart watches, smart TVs and so one. It is developed by the Open Handset Alliance in cooperation with Google. It is the most popular mobile platform and powers millions of devices. (Android Open Source Project, cited 28.11.2014.) It is open- source software and is meant for a product for all (Android Open Source Project, cited 28.11.2014). Android operating system is run on HTC, Samsung, Nexus, Acer, LG, Sony, Sony Ericsson, Motorola and smaller manufacturers (Android Open Source Project, cited 28.11.2014).

Due to its rapid development, Android launches high-quality versions of operating systems every year. On October 15 2014 Android released new version of operating system 5.0 Lollipop.

Android testing tools, unit testing and AVD

Android software development kit (SDK) is application development software. It provides application programming interface (API) libraries and developer tools for Linux, Windows and Mac users. (Android Open Source Project, cited 28.11.2014.)

Android has also developed a nice environment for application testing. Android SDK supports JUnit framework that is meant for automating unit testing. (Torres Milano 2011,13). Unit testing is meant for testing a single class (low level part) isolated from project`s other classes (low level parts) (Android Open Source Project, cited 29.11.2014).

Instrumentation is a public class that contains sets of methods, which force the application to enter needed state of Android life cycle. It makes possible to monitor application behaviour during any state. It also controls the application loading on Android. (Android Open Source Project, cited 29.11.2014.)

Testing on real devices is impossible due to variety of devices. Android SDK provides an opportunity to use Android Virtual Device, which simulates devices with different features and configurations. The developer chooses properties for device. It can support camera, mark cache partition size, include GPS, implement touch screen and other properties. It helps to test physical characteristic of devices. (Torres Milano 2011, 129-132.)

The Monkey is also one tool that is used for stress testing of applications. It produces random click, gestures, taps on a device. It produces random events that can be repeated for better overview. (Torres Milano 2011, 142.)

Logcat provides an exact view of the running processes (Torres Milano 2011, 38). While the program runs on emulator, platform provides text based logs that help developer to monitor the process of execution (Android Open Source Project, cited 29.11.2014). LogCat logs are saved every time the application is run (Torres Milano 2011, 230).

Android NDK

Android Native Development Kit (NDK) is a toolset that makes possible to develop some parts of the application in native-code languages such as C and C++. Due to NDK developer can use C and C++ libraries. It also gives a huge resource of already existing code. Those languages had been used for decades and accumulated knowledge helps easily to find a solution. NDK is preferred to SDK when a developer needs to get quick access to hardware and to develop performance-critical parts of application; these are applications that contain a lot of graphics and physics. For example this kind of parts are signal processing and physics simulation. Android NDK is meant only for partial development of application. (Android Open Source Project, cited 29.11.2014.)

2.1.2 iOS

iOS is an operating system used on Apple mobile devices, such as iPhones, iPads ad iPods. iOS was introduced only several years ago in 2007, however, iPhones are among most popular smartphones. Apple produces hardware and software for its devices. It uses also ARM based

processors from the Apple Ax series in their smartphones. Last launched chip was A8 and its target device is iPhone 6 (Smith, 2014, cited 30.11.2014).

Xcode is the integrated development environment for creating applications for iOS and OS X. Objective-C is used as a programming language. Xcode now includes frameworks such as Cocoa and Cocoa Touch, where Swift programming language could be used for iOS application development. (Apple Inc 2014, cited 30.11.2014.)

Testflight

App store contains a lot of supportive applications that could be helpful in quality assurance. One of them is Testflight beta testing which gives users access to the application even before application's release. However, testers are not usual users with the iOS device, but those who are invited via e-mail by developer himself. Even though the program was tested on devices, it still has to pass the app review before it appears in App Store. It has strict rules about design, user interface, content, functionality, advertising and other important aspects. If application has some discrepancy with guidelines, it would not appear on App Store . Testflight testes are sending feedback that also comes to the developers e-mail. Also crush reports are available for the overview. (Apple Inc 2014, cited 30.11.2014.)

Crush reports are available for a developer as information why application aborted on user's device. It contains data about the program activity before it crashed, namely it has full stack trace. It means developer can review all application action and trace the error that lead to application crash. It describes method by method that was applied and took part in application run before it terminated. (Apple Inc 2014, cited 30.11.2014.)

iOS simulator

iOS simulator is integrated part of Xcode and it can simulate Apple devices for example iPhones. It is easy to use in a programming process without integrating app on a device every time it is necessary to review it. Fast access to check how friendly user interface is, do new graphical parts suit each other and so on. It provides a possibility to simulate every Apple device that is meant to run this kind of application in the future. Even if only one actual device is available for further testing, it is still possible to review the application at least on simulator. (Apple Inc 2014, cited 1.12.2014.)

The simulator has its limitations, like memory use. While running application on a smartphone, the tool uses computers memory, which has better capabilities than any phone's memory. The performance of application run on the computer might have more advantages than on device. As well there might be difference in colours or user interface or graphic applying. Games that are using accelerometer input are also hard to test on simulator, since the screen cannot be turned to different angles and cannot be viewed as normally on device. If app is using camera on microphone, than the simulator has once again a limitation of performing different picture qualities due to camera differences. Every new generation of devices has usually better camera features. Same is with sound performance using computers microphone. This leads to necessity of quality assurance on real devices. It allows viewing an application, as it would run on users smartphone or tablet. It might be more difficult to manage interface, buttons are harder to tap than trying to perform the same action on simulator by clicking This is advantage over Android platforms, since the Android device number is enormous, on a counterweight iOS runs only on Apple products, which quantity is much smaller. (Apple Inc 2014, cited 1.12.2014.)

Testing on actual device

Applications are mostly available for iPhone and iPads. However, still they have different resolutions, some of them have retina display, older devices have less memory capabilities and some application need the access to device's hardware. This kind on actions requires a test on real iPhone or iPad. Before the application can be run on a smartphone, the developer must purchase the certificate, register his devices and create an ID for the application (Lee 2011, 511). Developers Program certificate costs 99 dollars per year (Apple Inc 2014, cited 1.12.2014).

Before the tester can launch the application on at least one device, it is supposed that developer has at least one Mac and one iPhone or iPad. They should be connected and smartphone or tablet is required to be added as Provisioning Portal. This Provisioning Portal provides information about the certificates, mac and testing devices. For application testing it is required to have a development certificate. Every application as well is supposed to have own ID as well, it could be obtained through Provisioning Portal. If application is good enough to become a part of the market in App store, the developer is also required to have a distribution certificate. (Lee 2011, 512-515.)

2.2 Marmalade simulator

Marmalade simulator was the basic used tool while testing the case. Marmalade simulator is available tool with Marmalade SDK. It is possible to run the application on development environment whenever it is Visual Studio or some else. The developer or tester can choose by himself what kind of configuration he wants to deploy. Since the desktop runs x86 and smartphones which are target devices run on ARM processors it is possible to simulate both of them. Computer uses ARM emulator to perform smartphone processors architecture. This is advantage since with the help of simulator it is possible to pretend the running application with all kind of possible configurations. (Hopwood 2013, cited 2.12.2014.)

Configurations

Marmalade simulator has different configurations, which can be changed in order to simulate different devices and different conditions that affect application work. Simulator controls hardware options. If an application requires access to the accelerometer, simulator can pretend using it. In real life user controls the device's angle by his hands, in simulator it is possible to control the rotation and angle by mouse click. Accelerometer is a sensor that indicates the acceleration. Application needs an access to this hardware in order to react to rotation. (Dimension Engineering LLC. Cited 2.12.2014.)

In audio configuration it is possible to choose the coder. Usually it depends, what kind of this software the target device has. In this model application PCM audio file format is used. Also it is possible to choose a volume that will be performed when opening application. Compass is presented as a detail in devices hardware. Nowadays it is almost default part of the device. The configuration simulates the device orientation changing. Contacts configuration mimics the access to the Phone Book. Device configurations allow arranging the settings. It has 17 facilities that could be changed. One of them is reported operating system that allows choosing an operating system that would be reported as a current simulator's operating system. Among values it is possible to select Android, OS X, Windows and Linux. Another option is device language that simulates the language that is used as the key language. It has no effect on applications if the application has only one language in its configuration. Some applications use phone numbers for identifying and the software can simulate the number use. File configurations allow setting the storage size. If application needs some space to store the files that were configured in applications the capacity might be set in this configuration. GL configuration allows

choosing the version on graphic library, for example EGL or Open GL ES. Keyboard configuration allows use PC keyboard – numbers, arrows and letters. License configuration allows changing the license status. If changing reported license status, than it is possible to choose among uncommercial, expired, purchase and other variations. Also it is possible to configure expiration date. Location configuration allows changing the latitude and longitude, horizontal and vertical accuracy. Pointer configuration allows changing pointer reported type. It means that if you still use mouse on the desktop, application receives pointer type as specified, for example stylus. SMS configuration is meant for SMS sending and receiving simulation. It is possible to insert a message that will be sent. Sound configuration is meant to control the volume, for example the application might be absolutely soundless by unchecking Enable Sound Device setting option. Surface configuration allows change the screen size and its settings, resolution. Predefined resolution setting option has a drop down menu with possible resolutions. They can be chosen or the width and height could be inserted manually. From the drop down menu it is possible to choose for example 960x640 iPhone4 landscape or 640x960 iPhone 4 portrait. It the most often used configuration due to variety of devices and their differences in screen resolutions. One of options is also to choose pixel format. Timer configuration allows choosing the time zone for the application receiving. Device reports to the application given time. Vibra configuration makes it possible to use vibration in application. Video configuration allows choosing the video format that is allowed to be performed by an application and also configure the volume. (Doc Editor 2013, cited 2.12.2014.)

3 QUALITY ASSURANCE

This chapter describes the testing as a part of quality assurance in general, aspects of cross-platform application testing and the description of the case. The case in this thesis is a cross-platform- ready-made application`s testing.

3.1 Principles of testing

Testing has many definitions and meanings. "Testing is generally described as a group of procedures carried out to evaluate some aspect of a piece of software" (Burnstein 2002, 7). However, Burnstein (2002, 7) gives another definition of testing as " testing can be described as a process used for revealing defects in software..." However, testing also has an economical approach. It decreases costs, lowers time spent on development (Burstein 2002, 8).

Testing is a process that supports development. It runs through projects life cycle and is applied after every step of development process. Before the project is finally released, it is tested again. (Hass 2008, 1-2.) This is not common in all development processes.

However, after publishing the application still some bugs would appear on users devices. The work of testers cannot be underestimated. The money spent on testing during development process reduces costs that will occur in case of failure of software. People have faced a lot of failures of software caused by minor errors and bugs. The pleasant for customers bug caused Commonwealth bank in Australia the loss of round sum of money. About 40 cash machines gave to customers large sums of money even though they did not have enough cash on their accounts. (BBC News Asia-Pacific 2011, cited 20.11.2011.) Another story which caused a loss of billions dollars was occasioned by a single error. The software was not able to fill 64-bit number into 16-bit space. European Space agency spent 10 years and $7 billion to manufacture Ariane 5 just to see how the rocket explodes at 39 seconds after launch. (Gleick 1996, cited 20.11.2014.)

Documentation

Bringing the list of bugs, errors or some remarks or comments to the programmers is also a challenge. The bug should be properly documented, so not only the tester understands how the

error acts and how is caused. After each correction it is easier to check over once again the problematic unit, so no further errors will occur. Testing documents are more technical specific, they rarely contain a business goals. (Burnstein, 2002, 197.) It is not enough just to find a bug. A tester should be able to produce it once again and then again.

3.2 Multi platform application features

The fact that the cross platform mobile applications are meant not only for one type of smart phones makes the quality assurance more challenging.

In the process of testing cross platform mobile applications it is important to remember the fact that the application is not meant only for one type of smart phones, but for a huge range of them. Even though the number of operating systems the software supports is quite small, the number of the operating system releases is high, as well the number of different sized and shaped, containing different hardware, devices is continuously increasing. It is essential to test am application on a real device for each platform, for example to test on one iPhone that performs iOS and a few Android devices. Android is running on different manufactured devices, for example Samsung, HTC.

Cross –platform application must run on all supported platforms without any big difference. The difference might occur when the code is changed under the pressure of market requirements. For example, one reason is hardware buttons and their meaning. IPhone has one hardware button that exits from all applications and opens desktop. The Android phone buttons vary from the model to model. Samsung Galaxy S3 has 3 hardware buttons. The first is “back button”, the second is “home button” by taping it the main display is performed, the third button allows configuring settings. Also phones have a lock button and sound buttons that still vary on different modes.

Different platforms possess more extended and profound research on market requirements of different companies. The application will not be accepted even with a minor deviation.

Platform differences

The programming code is one for all of them during development time. The application might act differently on devices. Even if a programmer is writing a code for native Android only and using Java the application might act differently. The reasons are simple –divers hardware. Some smart phones do not have enough power to display it, not enough memory to keep it playing. In cross platform applications the difference is higher.

Case application was developed in Marmalade SDK platform and written in C++. The application was meant mostly for Android and iOS. Both have own ways to support testing, which is quite difficult to follow without using native platforms and libraries.

3.3 Testing techniques

General testing techniques include white and black box testing (Watkins 2001, 15). Different testing types were applied while model application testing.

White box testing

White box testing techniques is meant for those who have programming skills. It allows checking on the internal structure on the application. (Watkins 2001, 17.) It tests application logic. The inputs should give expected outputs. However, if the logic of the application is somehow improper, then given unexpected outputs could be and this particular part of code could be modified. One of disadvantages of white box testing is its scale, since the number of paths in code that needed to be tested is too big. And it would take too much time to test all of them. Also the technique might not be sensitive enough to detect all logical errors. Unit tests are done to test the part of the application's code. (Copeland 2003, 144-146.)

Black box testing

It is an opposite of white box testing. It does not require the knowledge of application structure nor programming skills nor understanding the internal logic of the game. The functionality of the application is tested without knowing what happens to this input inside the application. The output is analysed and compared to supposed output. In order to get the all kind of misbehaviour of the application all kind of input should be tested. (Copeland 2003, 19-21.)

User interface testing

Smartphones and tablets have a touchscreen and though absolutely different interface requirements. There is no need in mouse anymore, any point at the screen can perform as a special button, but it is necessary to distinguish it from the other area of the screen. Only few hardware buttons are represented on devices. Usually hardware keyboard is missing on device. However, the software keyboard is performed. A finger or a stylus replaces mouse and arrows. Touchscreen is meant for easy and fast control of the device. (Jason 2011, 2.)

User interface testing is needed since many devices have different sizes, different scale of the touchscreen. That is why the application's button is also different sized and sometimes shaped on different smartphones or tablets (Jason 2011, 2). For example Apple's iPad Air's screen size is 9,7 inches, but Samsung Galaxy Fame lite screen size is only 3,5 inches. However, the same kind of application should run and should be easily accessible on both devices.

User interface should be user friendly. Morris Jason (2011, 3) names consistency as one of the most important aspects. This means that button should be recognizable. The pattern should be the same through all screens of application. Simple design is better, because the limitation of the screen size and resolution and user understanding as well. Complex menu with complex access doesn't allow user to get to application core information as fast as may be he would like to. Android has a hardware button with the menu setting, so basically the access to the settings of the application. The interface of application must have some logic in order to achieve an audience. (Jason 2011, 2-4.)

Break point

In Visual Studio integrated development environment it is possible to use break points. Break point is set to some code line and program will run till this point and pause. At this moment when the program is paused, it is possible to check its execution, its variables and memory use. Code can be changed during break point pause. It helps to determine bugs faster and change it while execution. (Microsoft 2014, cited 5.12.2014.)

4 THE CASE

The role application for this thesis is an arcade-type game. Visual Studio and Marmalade SDK is a development environment for this application. Marmalade simulator was one of the most used tools in testing process. Available devices for testing were Samsung Galaxy S3, Kindle Fire HDX and Google Nexus 1. Since there was no possibility to test an application on Apple devices, the game was developed only for Android operating system. However, as an advantage of multi-platform the code might be deployed to other platforms at any point of time. The game's name is Glow Hockey 3D. It is available now at Google Play store.

The game is arcade type game. At this point of time 40 levels are available to pass as adventure. Another possibility to play this game is multiplayer. Two players can use one device at once. The point of the game is to score the ball to the competitive area.

4.1 Starting Testing

Testing was performed mainly on Marmalade SDK Simulator and available devices such as Galaxy S3, Kindle Fire HDX, Sony Xperia, Nexus 1. The selection of devices was based on the availability. Android builds were supposed to be installed manually on devices. Bitbucket is a git repository for sharing a project with all its components, such as graphics and code. For tester it was possible to run latest committed state of application. As it is seen from the Figure 1 git pull command updated local repository and tester received latest state of the game on Windows 7 PC.

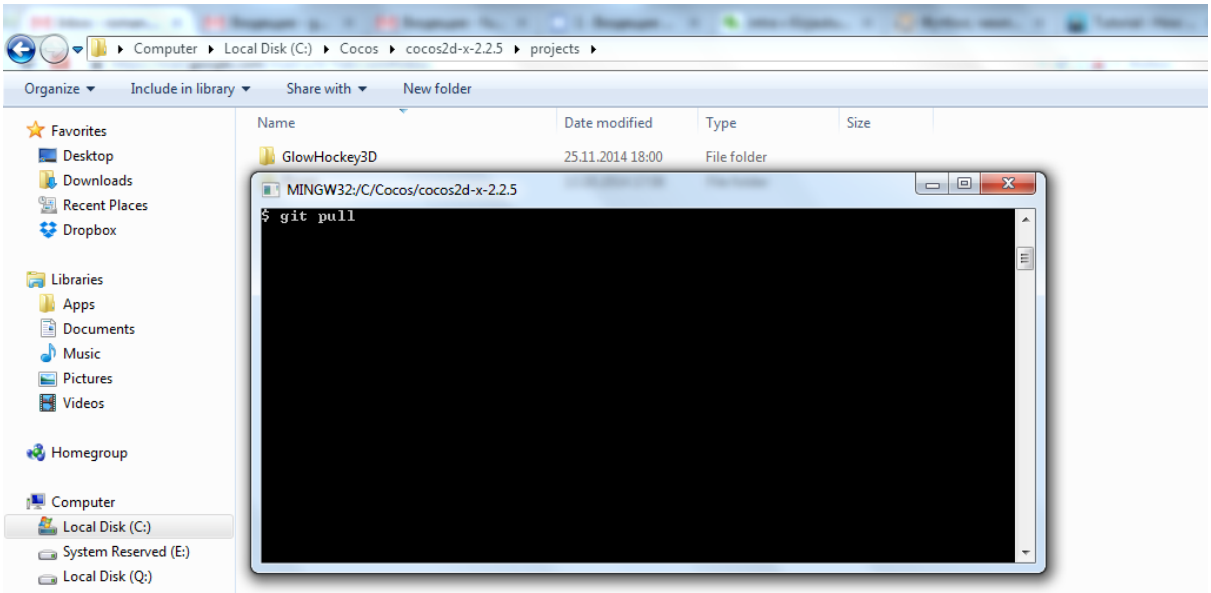


FIGURE 1. $ git pull command in Bash GUI.

Microsoft Visual Studio Integrated development environment was used for project development. Glow Hockey 3D could be opened by launching .mkb file as a Figure 2 shows. After the project started, it was possible to run Marmalade SDK simulator.

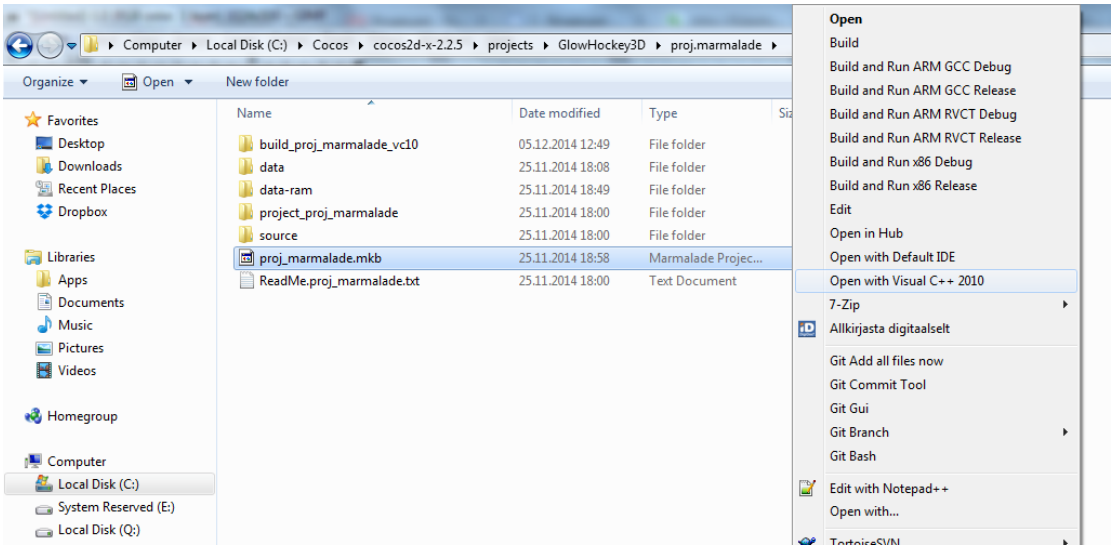


FIGURE 2. launching .mkb file with Visual Studio.

4.2 Gameplay

Gameplay testing's aim is to check over the application logic and detect major bugs during actual application use. It is highly necessary to make sure that the gameplay is smooth before correcting it visually and build for different devices.

The game logic should look like it was supposed to be and like it was documented in requirements. For example during the gameplay testing it was controlled how physics works. In Glow Hockey 3D all balls should be repulsed from the wall as it would happen in real life according to the angle and speed. If some bonus was chosen, the bonus should work instantly and contribute to the part that won the bonus. The purpose of the game is to score the ball to the competitive area. Every part has some number of lives, for example a player can miss 8 balls before he is terminated from the round. Every missed ball equals to one lost life. If the adventure mode is chosen, then after every completed level, new level should be accessible. As new bonus was added it had to be tested how it really works in the game. One of the bugs that was detected during game play testing is represented on Figure 3. Namely, when the spawning bonus was chosen, it is supposed that one ball transforms into 7 and they were supposed to move into different direction with same speed the ball reached the bonus. However, the new 7 balls where moving very slowly and had distorted movements.

FIGURE 3. Spawning balls.

Some tests were run to control the in-game currency. The player has own balance. If player wins, his balance should increase by defined amount of coins. If player purchased in-game item, there should be enough balance to complete the operation and the balance cannot be negative.

4.3 User interface

Simulator has many advantaged over the limited set of available devices. It is possible to select any resolution and control major specifications. It is also possible read logs in Visual Studio IDE and track the exceptions roots.

Before the application was run on actual devices, simulator was used to test different resolutions. It is possible to change the parameters according to most popular resolutions. For example iPhone 3 and 4 resolution is 960x640, iPhone 5 and 6 has 1136x640 resolution, iPad has 1024x768, some Android running devices have 1920x1080. Marmalade SDK simulator has a surface configuration that allows to set any height and width. Figure 4 shows possible setting changes in Marmalade SDK simulator. In a drop down menu most common resolutions are available. However, it is possible also to test some extreme screen ratios. IPhone 5 ratio is 1.775 of width/height ratio. This screen is quite wide. Some Blackberry devices have square-shaped screens, when their resolution is 720x720 pixels. This kind of devices has 1.0 screen ratio. Kindle Fire HDX is full HD screen and graphics should be adjusted to current resolution without and defects and artefacts. Kindle Fire HDX is Android based device and was available as actual device for testing the application.

White box testing technique is used at this point, because tester can read logs and check upon some special methods behaviour using break points during application execution in Visual Studio.

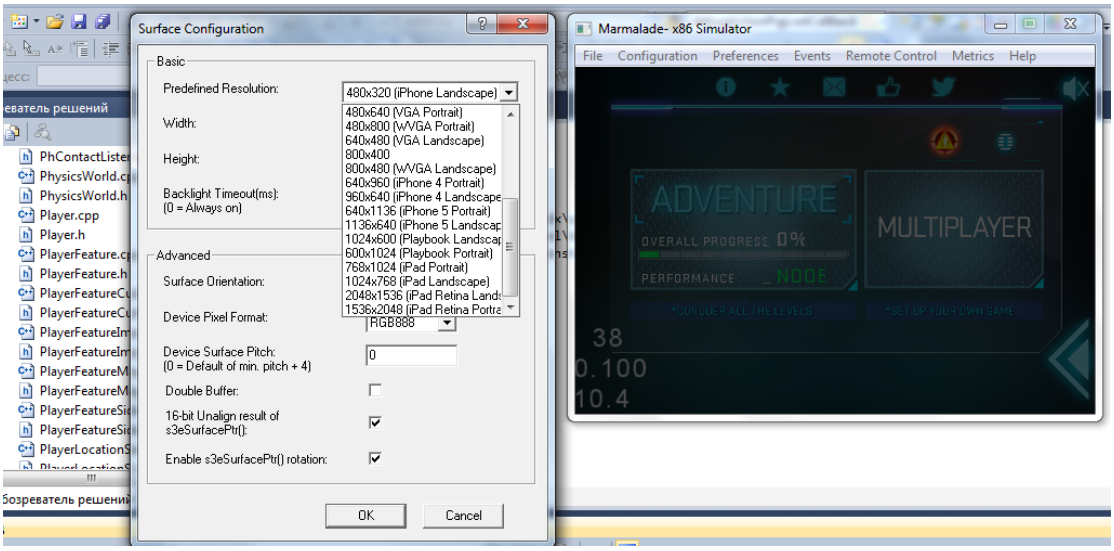


FIGURE 4. Marmalade SDK Simulator surface configuration set.

Testing user interface aim is to make sure all elements are visible and they are located on proper positions. All buttons respond, details can be taped on even most smallest and weirdest screens and layout is consistent through all displays and all resolutions. Gameplay should fully fit the screen. Glow Hockey 3D doesn't have this issue on any tested device, but on the Figure 5 it is shown that even very popular games can have this kind of issue. This is a screenshot from Nokia Lumia 1520. As it can be seen the application's canvas does not fully fit the screen of the device.

FIGURE 5. Nokia Lumia 1520 screenshot.

User interface testing part also makes sure all buttons are clear, easy accessible and work on device. Same test is run on actual devices to check buttons are fine. In this application two players are able to play at the same time. The screen is split into two parts and multitouch option should be available on device so both players can tap their part of the screen at the same time. It is important that application should recognize if device has a multitouch option. Some old devices that have only 1 available touch at the same time should pop up the message containing information about multitouch requirements in the split screen mode. Figure 6 represents the screenshot of split screen mode of the application. Both players should be able control their paddles on corresponding sides. However, on this kind of old devices it is still possible to play the standalone adventure mode.

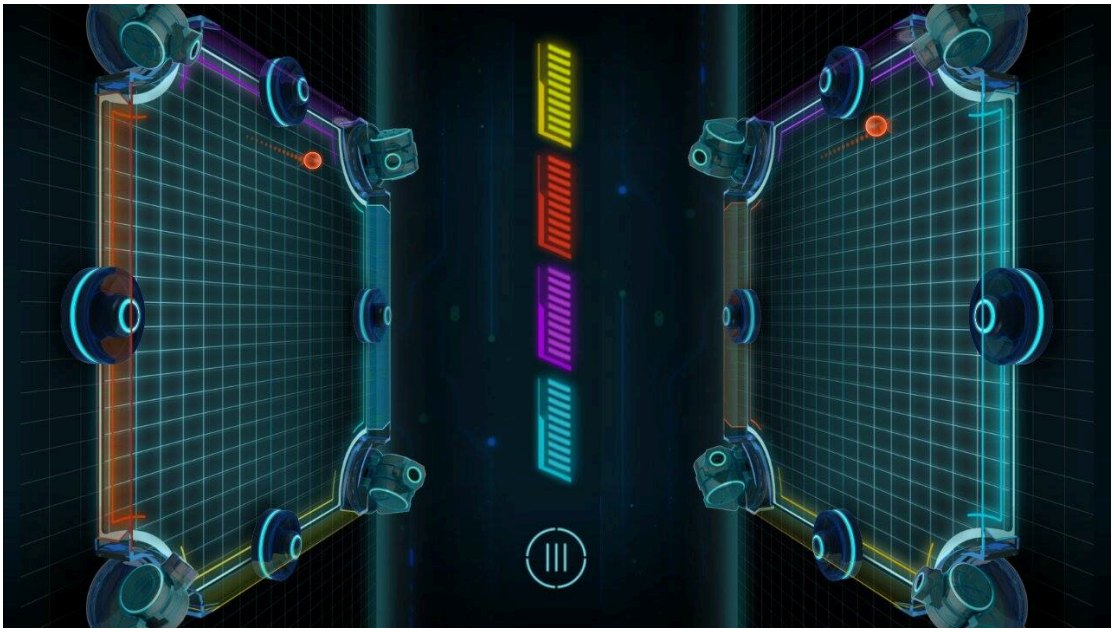


FIGURE 6. Split screen mode in landscape view.

4.4 Performance testing

First time the game is running on device the performance is slow. Game has many draw calls and physics is not optimized. In the left bottom corner of the screen the application has a debug output of current draw calls, time step and frames per second (FPS). For example in the Figure 7 the draw call number is 37, time step is 0.012 seconds and FPS is 50.0. At the beginning of the development process this numbers are much higher, draw calls might be around 1000 or even more. This makes the application work very slow. However, only about dozen draw calls remain

before application‘s release. Draw call is time consuming operation, its optimization must come first.

FIGURE 7. Draw calls, time step and FPS.

FPS is a rate that represents how many times game was updated during 1 second. This parameter should be close to 60 FPS so the application will run smooth and without lags. Tracking the FPS value drops helps to determine the weak spots of the game. If the FPS drops, the tester could detect the moment when it happened. For example, some expensive calculations are done and game is not smooth at that time. Box2d was used for physics simulation. When some collision of objects was improper due to code or some other relation, FPS decreased as well.

Device testing of performance depends on hardware and software of the smartphone or tablet. It brings out specific issues. For example SonyEricsson Xperia had a very bad performance during application run. However, after turning off the sounds and music the performance improved. Was it software or hardware issue, the exact problem did not concern the project.

In general, all devices usually are close to perform 60 FPS during the game. When the game optimization is done, devices can have different time of loading the game, but they don't have differences in game performance. For example, loading time depends on memory allocation for the objects.

4.5 Platform native testing

Every platform has own specifications. Android has own hardware buttons that differ from iOS. Platforms have its own way in which device falls asleep or how the game loses its focus, how device switches between different applications. Android has 3 hardware buttons, namely home, menu and back buttons. Menu and back buttons are handled to navigate in the application. Blackberry PlayBook tablet is sensitive to the swipe gestures. Turning device off also should be controlled. It means if the game was on, when the turn off device button was pressed, the game should enter the pause mode. The game should save its state to the persistent memory storage. All in game settings should be stored as well (level, score, user settings) and retrieved without losing or changing any data.

During native testing, problem was detected with Blackberry Z10 smartphone. Figure 8 represents the view of the game. It was corrupted, had some artefacts. Blackberry device had a specific graphical chip. The problem might be in hardware or in Marmalade SDK.

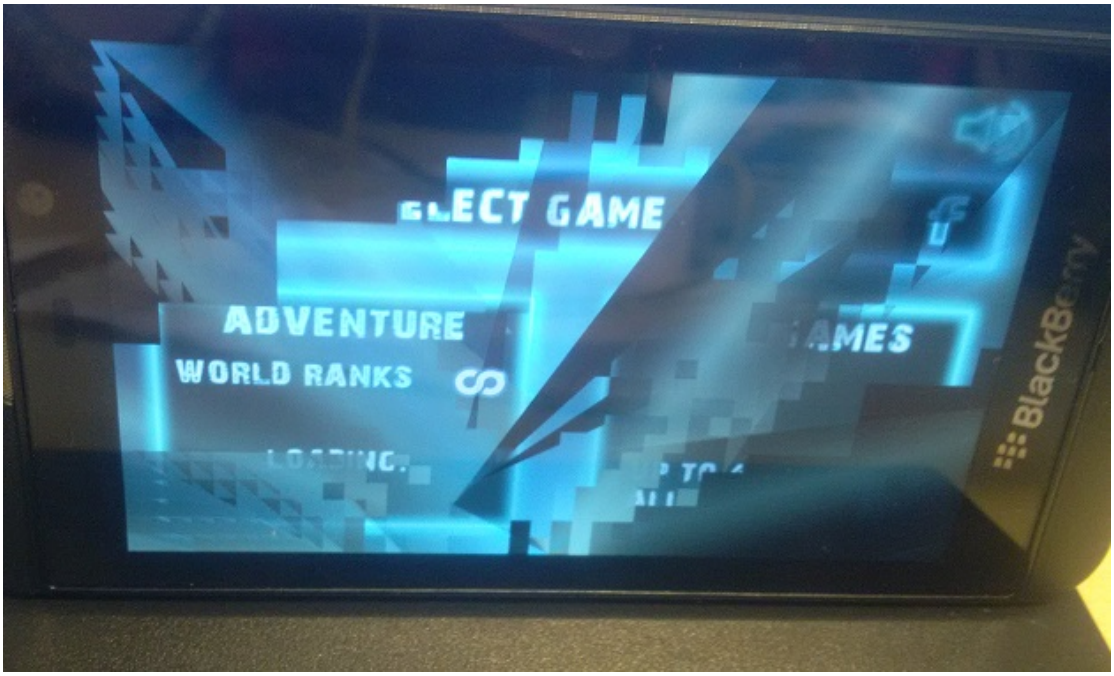


FIGURE 8. Blackberry graphical appearance due to graphical chip.

During Android-operating-system-based devices native testing it was possible to check the LogCat. If some issue appears on Android devices, the log could be read. For example the application crushed when the tester pressed one specific button. The phone was connected to the PC and once again the button issue was reproduced. Android SDK contains LogCat tool that shows game logs. Figure 9 shows a screenshot of the log that states the problem. In this case the advertisement unit was not initialized. It must be initialized before the ad unit would be used. The line from log "at .org.iddqd.gh3d.glowhockey3d$4.run(glowhockey3d.java:178)" shows that the exception is caused by line 178.

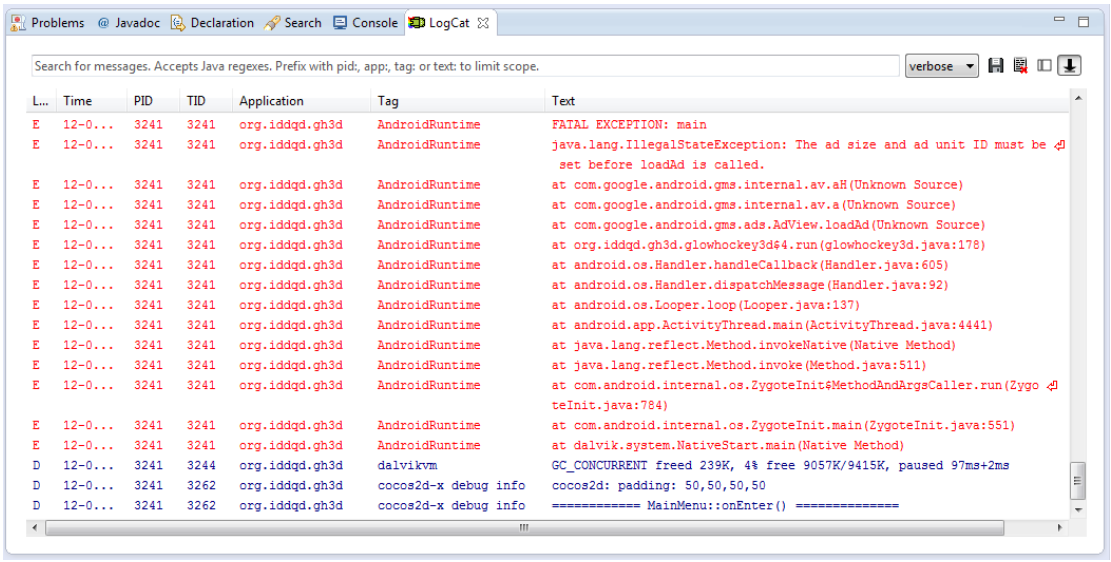


FIGURE 9. LogCat screenshot.

Testing on actual devices is important during every testing part. iOS devices require developer program certificate. It is much easier to check the application on Android devices. For Android devices .apk executable file is transferred on smart phone or tablet, the application can be installed using wire or via Wi-Fi. If device is connected with PC, the logs from the device could be checked.

4.6 Third party components testing.

This part also depends on a target platform such as Android or iOS, target distribution channels such as Google Play, Amazon app store, Samsung apps or iTunes. The application contains ads banners and full screen advertisement. During this testing ads are controlled. They should appear when they are required. As well ads should not disturb the game play. However, tester should not

click on the ad banner since every click brings some income and store follows that developers will not click by themselves on the banners inside their application. The device should be added as a test device and then the click on the ad banner will not be proceeded as a customers click. As seen in the figure 10, the ad banner is located at the bottom aligned to the centre. It does not cover any buttons, does not interrupt the gameplay, because it vanishes when the game starts. Also click should open an external link to the store or a webpage that contained the ad banner.

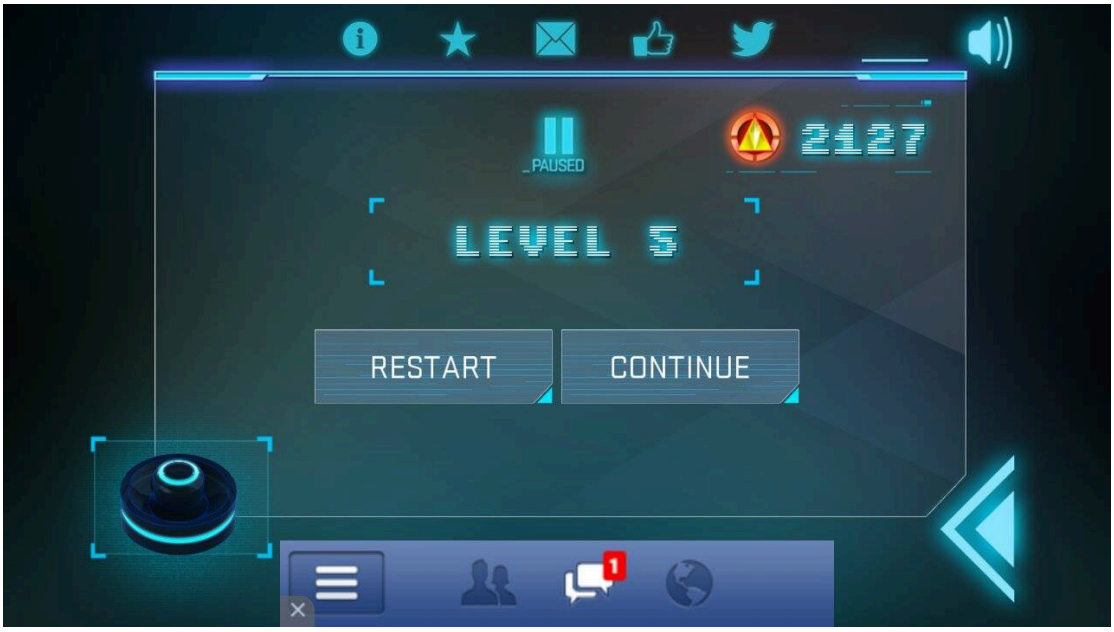


FIGURE 10. Ad banner in a game.

Android Google Play and iOS iTunes have an implementation in its deployment settings that allows placing in app purchases. This was tested on Android devise only. Test user was added to Google Play service and purchasing test was conducted. After purchase the new item should be unlocked, ad banners must be removed. No banners can appear after test purchase for the test user.

Game also contains external links to Facebook, twitter, e-mail contact, link to the game in store. As it can be seen in Figure 10 this external links are located at the top of the screen. Game rate proposal is an external link to the game in store. By clicking on any of this links customer launches browser or application (for example Facebook application). When customer finishes browser or external application use, the game application should appear. Clicking on symbol should open corresponding application or web page. Mail message collects the information about device. However, customer can delete this information before sending e-mail.

Analytics system's aim is to track users activity. Google Analytics is service that was used in this project. This is very helpful tool for quality assurance, however, during this part Google Analytics information is controlled if it receives last data of the game played. It is enough to complete one level and check if the data was updated.

4.7 Alpha testing and after release debugging

Before final release, users could test application. Alpha testing possibility is available on Google Play and TestFlight on iOS. Application can be distributed among users who want to test the games, for example same company developers, friends, forum members. Test user activity can be tracked, crash reports can be checked, as well overview of users' progress. Google play allows distributing an app in 3 states as it can be seen in the Figure 11. Production state allows every customer of Google Play downloading the application to their Android device. When the application is in testing state, only when the user is invited to. Google Analytics also collects data while application is in Alpha or Beta testing state. Beta testing is also for user and is meant when application is in a pre releasing state. Sometimes this kind of testing is shared to see the public reaction. However, in this particular case, the alpha testing was not used.

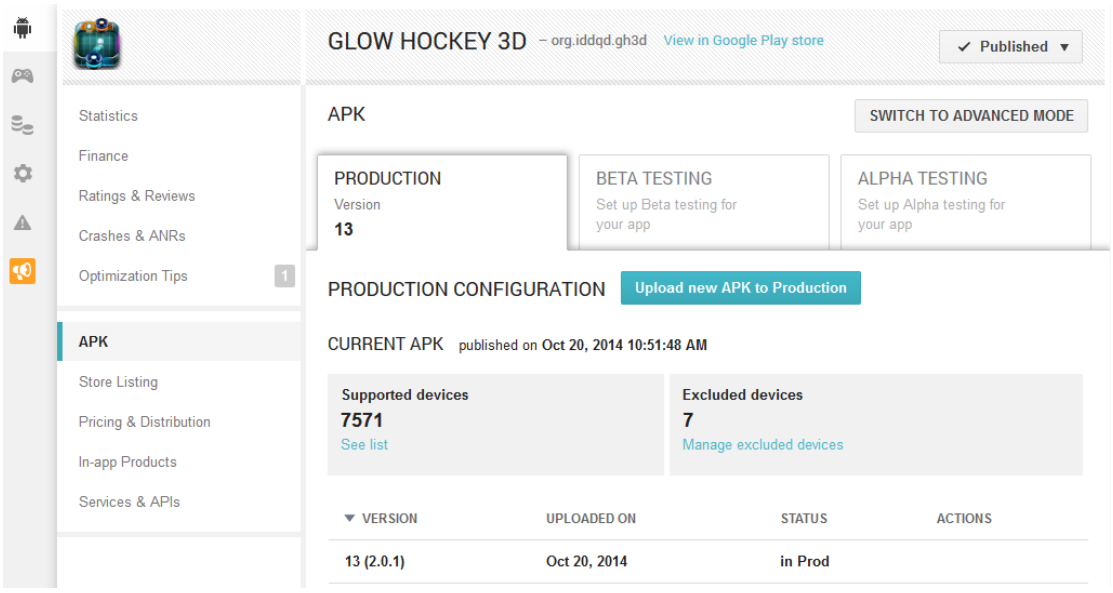


FIGURE 11. Google play options of distribution.

After application release some cases started to appear that were missed from quality assurance. There are few ways to check the incorrect application behaviour on users devices.

Some users write comments to the application on Google Play. For example one user stated that the on screen buttons on Android did not disappear when the application was launched. Figure 12 shows one message from the user that stated this problem with visible buttons and a reply on his message. Developers solved this problem soon.

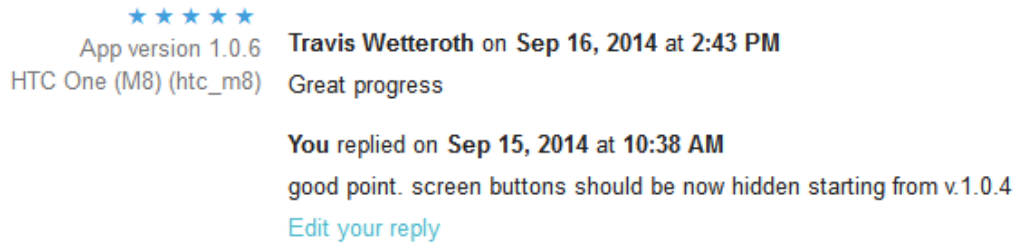


FIGURE 12. Messages from users

As well was pointed out by users a problem with game restores, it takes about 5 seconds to load textures. This problem was solved partly, sometimes application after it restore may come with black screen. It is possible for users to write personal e-mails. E-mail contains also information about device and operating system version, which helps to determine the problem faster. Crash reports are available for analysing and debugging. Figure 13 shows the crash reports that were received from users of Glow Hockey 3D. Tester determines which problem is the most popular and it comes as most important issue for developers. As seen from the Figure 13, run time exception appears most frequently among crashes.

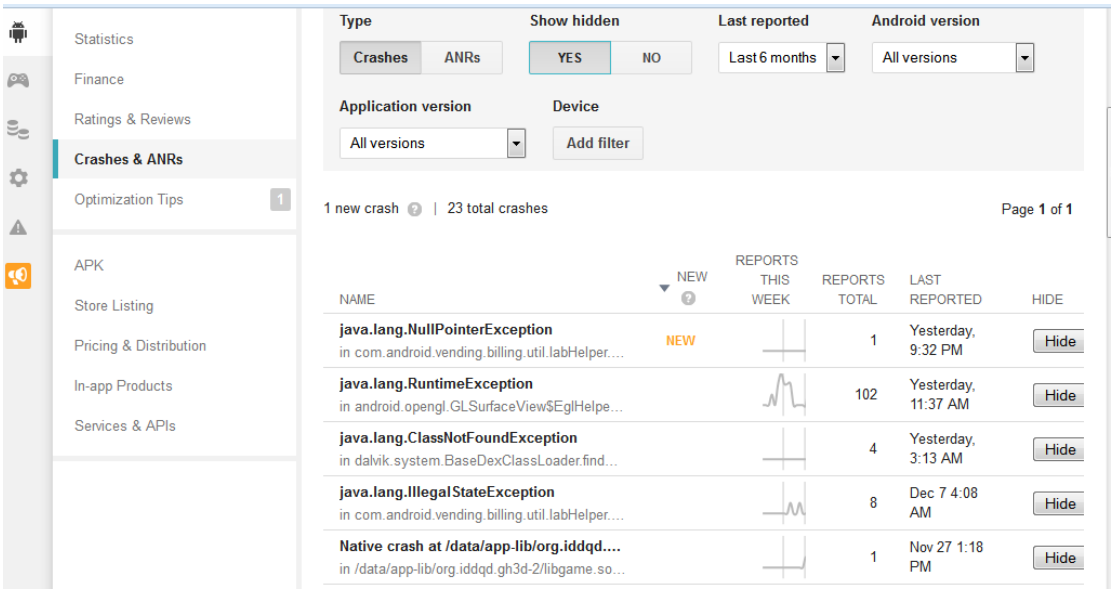


FIGURE 13. Crash reports in Google Play.

Exception can be chosen and more data about the error appears. Figure 14 shows run time exception details. Last time it was reported the previous day and totally it appeared 11 times on users devices. It also can be marked out, that the error most frequently appears on ZTE Light Tab 2 V9A. This is not popular product that is why this error was not prioritized.

It is also possible to use soft launch, which means that this application will be released in specific region, for example only in Finland. After collecting crashes and improving the game, the application could be released globally.

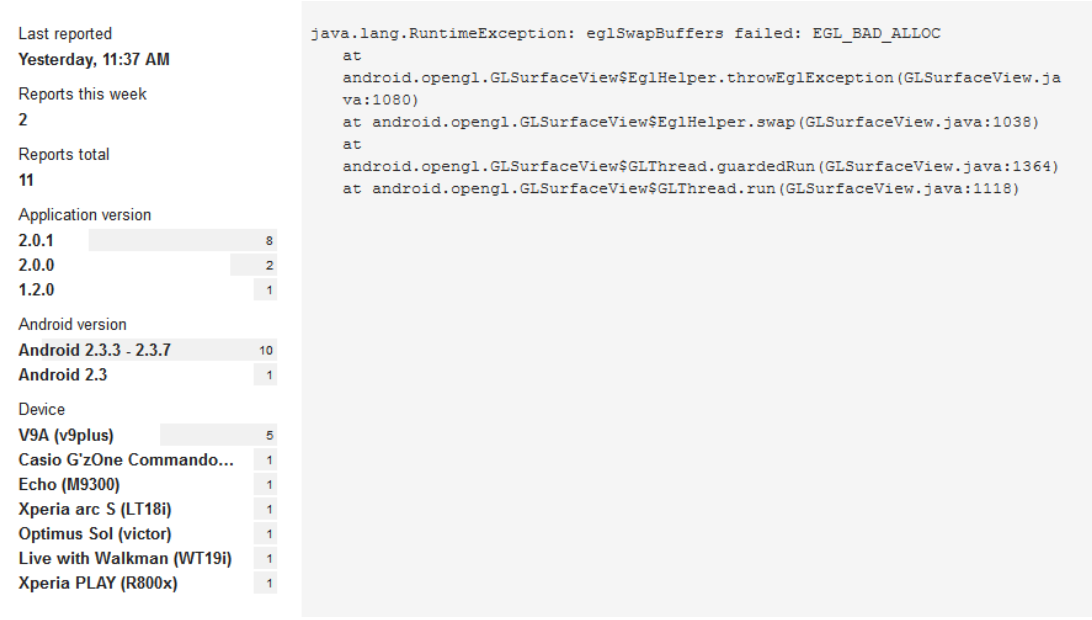
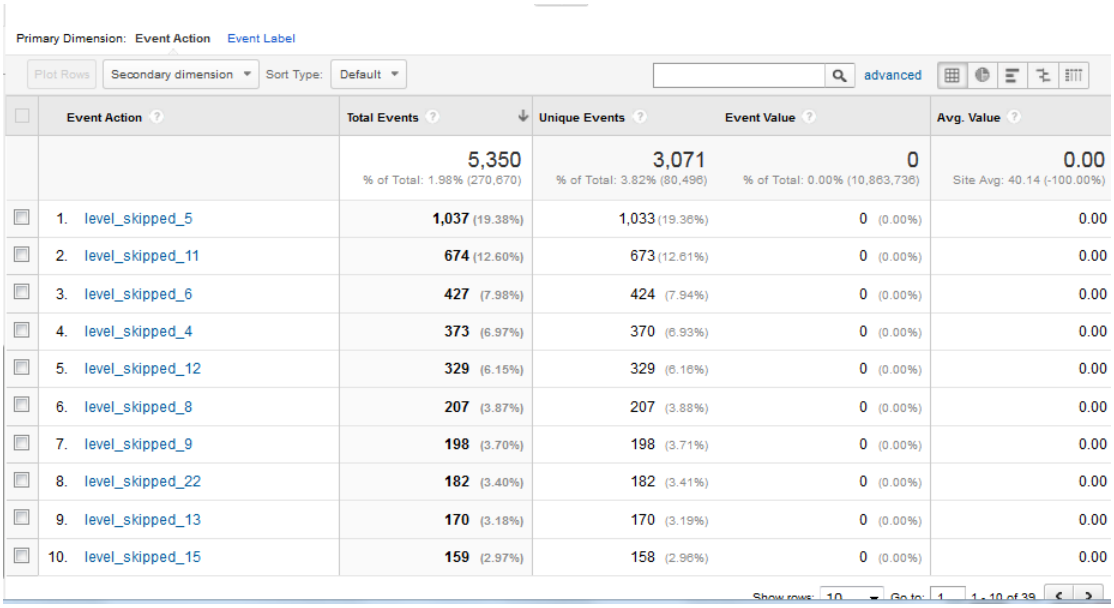


FIGURE 14. Crash report detailed view.

Google Analytics tool is very helpful. It shows users behaviour in the games, could be checked how many users are online at the point of time, how many new users added, what they do, what progress did they have. In game it is possible to skip the level that seems to be hard for user. After analysing this information some levels can be improved. Figure 15 shows most skipped levels. These levels were difficult to pass and after that analyse, the most difficult level was simplified, for example, few seconds for passing the level were added.



Primary Dimension: Event Action Event Label

Plot Rows | Secondary dimension | Sort Type: Default | advanced

	Event Action	Total Events	Unique Events	Event Value	Avg. Value
		5,350 % of Total: 1.98% (270,670)	3,071 % of Total: 3.82% (80,496)	0 % of Total: 0.00% (10,863,736)	0.00 Site Avg: 40.14 (-100.00%)
1.	level_skipped_5	**1,037** (19.38%)	1,033 (19.36%)	0 (0.00%)	0.00
2.	level_skipped_11	**674** (12.60%)	673 (12.61%)	0 (0.00%)	0.00
3.	level_skipped_6	**427** (7.98%)	424 (7.94%)	0 (0.00%)	0.00
4.	level_skipped_4	**373** (6.97%)	370 (6.93%)	0 (0.00%)	0.00
5.	level_skipped_12	**329** (6.15%)	329 (6.16%)	0 (0.00%)	0.00
6.	level_skipped_8	**207** (3.87%)	207 (3.88%)	0 (0.00%)	0.00
7.	level_skipped_9	**198** (3.70%)	198 (3.71%)	0 (0.00%)	0.00
8.	level_skipped_22	**182** (3.40%)	182 (3.41%)	0 (0.00%)	0.00
9.	level_skipped_13	**170** (3.18%)	170 (3.19%)	0 (0.00%)	0.00
10.	level_skipped_15	**159** (2.97%)	158 (2.96%)	0 (0.00%)	0.00

Show rows: 10 Go to: 1 1 - 10 of 39

FIGURE 15. Most skipped levels in the game.

Also it is possible to track the bonuses use or how many coins were gained. May be some functionality stayed unclear and users ignore its possibility.

The average duty after release is to check the rates and read the comments, analyse reports, check e-mail and improve the game by fixing bugs. As soon as new version of improved version is ready, it is released to the market. The application on users devices will update. The new version could be tested with alpha and beta testing types. If everything seems correct, it will be released on the market. As well new version could be available for some region only. And if no major problems appear, it can be released globally.

5 CONCLUSIONS

The aim of this report was to show the process of testing of application. This application was developed on Marmalade SDK, which has a few target platforms. At the same point of time due to cross-platform tool it is possible to implement an application on different mobile operating platforms including Android and iOS.

The arcade type game Glow Hockey 3D was tested through the game development cycle. It's main target platform was Android, however, it is possible to release a game on iOS market by simply buying iOS developer certificate and implementing the game on new platform. There is no need to program it on Xcode using Objective-C. If it is possible to test most resolutions on simulator, it is still necessary to test on actual devices that represent every target platform, because they behave differently than simulator. In future projects more alpha testing could be held to check the usability and playability of the game.

During the testing period no major problems appeared. Google Play and iTunes provide different manuals and guidelines that help during testing process. The development process took about 6 month from the beginning of production till the release on Google Play. The game is still monitored and if crucial errors appear, new improved version is released.

6 DISCUSSION

The main idea of this thesis was to show the testing process of the application that was meant more than for one platform. Due to lack of resources and time it was decided to use multi platform tool for the application development.

Before this project I was taking part in game development process as a part of practical studies. However, during this project there was no college who was going to show step by step what I am supposed to do when I face the error I have not seen before. Marmalade SDK is a very developer friendly tool with clear instructions and number of guidelines. The developer forum was also available for finding the solution to the issue.

Overall the application was released in 6 months after the start. The application received good reviews and no major bugs were detected after the game release. The main objective was to save time and resources and it was successfully achieved.

REFERENCES

Android Open Source Project. Android NDK. Cited 29.11.2014.
http://developer.android.com/tools/sdk/ndk/index.html

Android Open Source Project. Android, the world's most popular mobile platform. Cited 28.11.2014. http://developer.android.com/about/index.html

Android Open Source Project. Instrumentation. Cited 29.11.2014
http://developer.android.com/reference/android/app/Instrumentation.html

Android Open Source Project. Intelligent Code Editor. Cited 28.11.2014.
developer.android.com/tools/index.html

Android Open Source Project. Logcat. Cited 29.11.2014
http://developer.android.com/tools/help/logcat.html

Android Open Source Project. Testing fundamentals. Cited 29.11.2014
developer.android.com/tools/testing/testing_android.html

Android Open Source Project. The Android Source Code. Cited 28.11.2014.
https://source.android.com/source/

Android Open Source Project. Using Hardware Devices. Cited 28.11.2014.
developer.android.com/tools/device.html

App Annie 2014. App Annie Index: Market Q3 2014. Cited 2.12.2014,
www.appannie.com/intelligence/

Apple Inc 2014. About iOS Simulator. Cited 1.12.2014.
https://developer.apple.com/library/ios/documentation/IDEs/Conceptual/iOS_Simulator_Guide/Introduction/Introduction.html

Apple Inc 2014. Analyzing Crash Reports. Cited 30.11.2014. https://developer.apple.com/library/ios/documentation/IDEs/Conceptual/AppDistributionGuide/AnalyzingCrashReports/AnalyzingCrashReports.html

Apple Inc 2014. App Store Review Guidelines. Cited 30.11.2014. https://developer.apple.com/app-store/review/guidelines/

Apple Inc 2014. iOS Developer Program. Cited 1.12.2014. https://developer.apple.com/programs/ios/

Apple Inc 2014. TestFlight Beta Testing (Optional). Cited 30.11.2014. https://developer.apple.com/library/ios/documentation/LanguagesUtilities/Conceptual/iTunesConnect_Guide/Chapters/BetaTestingTheApp.html#//apple_ref/doc/uid/TP40011225-CH35-SW2

Apple Inc 2014. Testing and Debugging in iOS Simulator. Cited 1.12.2014 https://developer.apple.com/library/ios/documentation/IDEs/Conceptual/iOS_Simulator_Guide/TestingontheiOSSimulator/TestingontheiOSSimulator.html

Apple Inc 2014. Understanding and Analysing iOS Application Crash Reports. Cited 30.11.2014. https://developer.apple.com/library/ios/technotes/tn2151/_index.html#//apple_ref/doc/uid/DTS40008184

Apple Inc 2014. Xcode. Cited 30.11.2014 https://itunes.apple.com/us/app/xcode/id497799835?ls=1&mt=12

BBC News Asia-Pacific. 2011. Sydney cash machine glitch gives customers extra money. Cited 20.11.2014. http://www.bbc.co.uk/news/world-asia-pacific-12606735

Burnstein, I. 2002. Practical Software Testing: A Process Oriented Approach. New York: Springer.

Copeland, L. 2003. A Practitioner's Guide to Software Test Design. Artech House

Dimension Engineering LLC. A beginner's guide to Accelerometer. Cited 2.12.2014. http://www.dimensionengineering.com/info/accelerometers

Doc Editor 2013. Simulator Windows PC Configuration Options. Cited 2.12.2014. http://docs.madewithmarmalade.com/display/MD/Simulator+Windows+PC+configuration+options#SimulatorWindowsPCconfigurationoptions-contactsondesktop

Gleick, J. 1996. A Bug And A Crash. New York Times Magazine. Cited 20.11.2014. http://www.around.com/ariane.html

Hass, A.M.J. 2008. Guide to Advanced Software Testing. Norwood: Artech House.

Hopwood, M. 2013. The Marmalade Simulator. Cited 1.12.2014. http://docs.madewithmarmalade.com/display/MD/The+Marmalade+Simulator

International Data Corporation 2014. Worldwide Smartphone Shipments Increase 25.2% in the Third Quarter with Heightened Competition and Growth Beyond Samsung and Apple, Says IDC. Cited 30.11.2014, http://www.idc.com/getdoc.jsp?containerId=prUS25224914

Jason, M. 2011. Android User Interface Development: Beginner's Guide. Birmingham: Packt Publishing.

Lee, W. M. 2011. Beginning iOS 5 Application Development. Indianapolis : John Wiley & Sons.

Marmalade Technologies Ltd 2014. Supported Platforms. Cited 2.12.2014 https://www.madewithmarmalade.com/marmalade/supported-platforms

Marmalade Technologies Ltd 2014. The Marmalade Platform. Cited 2.12.2014 https://www.madewithmarmalade.com/marmalade/marmalade-platform

Microsoft 2014. Debugging Basics: Breakpoints. Cited 5.12.2014. http://msdn.microsoft.com/en-us/library/vstudio/4607yxb0(v=vs.100).aspx

Scaplehorn, S. 2012. Marmalade SDK Mobile Game Development Essentials, Birmingham: Packt Publishing.

Smith, R. 2014. Analyzing Apple's A8 SoC: PowerVr Gx6450 & More. Cited 30.11.2014. http://www.anandtech.com/show/8514/analyzing-apples-a8-soc-gx6650-more

Torres Milano, D. 2011. Android Application Testing Guide, Birmingham: Packt Publishing.

Watkins, J. 2001. Testing IT : An Off-the-Shelf Software Testing Handbook. Cambridge: Cambridge University Press