

Project Report

by Craig David Brannan

FILE	17731_CRAIG_DAVID_BRANNAN_PROJECT_REPORT_78196_1684095850.DOCX (2.58M)		
TIME SUBMITTED	20-APR-2016 11:58AM	WORD COUNT	15400
SUBMISSION ID	56114043	CHARACTER COUNT	89256



Investigating Cross Platform Software for Game Development + Test Techniques

By

Craig Brannan

Supervisor: Dr Kemi Ademoye

Second Marker: Mr Michael Dacey

Project submitted as part of the requirements for the award of BSc(Hons) Software
Engineering

April 2016

DECLARATION

I, Craig David Brannan declare that I am the sole author of this Project; that all references cited have been consulted; that I have conducted all work of which this is a record, and that the finished work lies within the prescribed word limits.

This work has not previously been accepted as part of any other degree submission.

Signed :

Date :

FORM OF CONSENT

I Craig David Brannan hereby consent that my Project, submitted in candidature for the B.Sc.(Hons) Software Engineering degree, if successful, may be made available for inter-library loan or photocopying (subject to the law of copyright), and that the title and abstract may be made available to outside organisations.

Signed :

Date :

Abstract

The following project will discuss the development of creating a cross platform application and using different types of test techniques to evaluate the app. This will include looking into different types of cross platform software available, look into different types of testing methods, creating the game and then evaluate the tests implemented into it.

In the project the method that will be undergoing is the positivism deductive approach as this will decide how the development will be based on the research produced in the document. This will give an overall idea what is expected when going through the development stage. There is also testing that will occur this will act like an experiment to determine what results we will get.

The development method will be focused on the prototype approach as this allows the developer to break the application down into separate parts, obtain a list of requirements, review them and then test them individually. This will make the application fully tested and more reliable when put all together at the end.

Some of the findings to help make this project a success is that Xamarin will be used to create the application as it offers many tools and has a C# syntax language that the others do not offer which makes this ideal.

This will continue with the conclusion and recommendation section of the project which will discuss the overall result and what complications were found. Also it will discuss what recommendations will be made for future preference.

Table of Contents

Abstract.....	4
Table of Contents.....	5
Table of Figures.....	8
Table of Tables.....	9
Aims & Objectives.....	10
Introduction	11
Literature Review.....	11
History of App Development	11
Apps	12
Native Apps.....	12
Web Apps.....	12
Hybrid Apps.....	13
How do they work.....	14
Android vs. IOS.....	15
IOS –Advantages	16
Disadvantages	16
Android-Advantages	16
Disadvantages	16
Identifying Users	17
Why are mobile applications important	17
Mobile User Experience	18
Most Popular Apps being used today	20
Cross Platform Software	20
PhoneGap.....	21
Appcelerator titanium.....	22
Xamarin	22
Developing Games With Xamarin	23
The game Loop.....	23
Math.....	24
Content Pipeline.....	24
Game Development Technologies.....	25
CocoSharp	25
Monogame.....	25

UrhoSharp	25
Testing.....	26
Code Based Testing.....	26
Object Oriented Testing.....	26
Component Based Testing	26
Specification Testing	26
Manual testing	26
Automatic testing.....	27
Black Box Testing	27
White Box Testing	27
Grey Box Testing	27
Alpha Testing.....	28
Acceptance Testing/Beta Testing.....	28
Usability Testing.....	28
System Testing	28
Seven Principles of software testing.....	29
Research and Development Methodology	31
Research Methodology	31
Development Methodology	31
Software	33
Hardware	34
Materials And Tools	34
Design.....	35
About The Game	35
Scope of the Game.....	35
Application Features	35
Title Screen	35
Instruction Screen.....	36
Leader Board screen	36
Objects in Game.....	36
Gameplay	37
Game over Screen.....	37
Overall Non-Functional Requirements for Application.....	38
Scenarios and Use Cases	38

Storyboard	38
Class Diagram	39
Gameplay Use case diagram	40
Collision Use Case Diagram	42
State Chart Diagram-Collision	43
Database - Leader Board	43
Update Score Use Case Diagram	45
Implementation	46
Problems and Successes	46
Code Snippets	47
Gravity	47
Player Collide With Box	47
Add Score	48
Increase Difficulty	48
Spawn Boxes	49
Animation	49
Jumping	51
Testing	51
System Testing	51
Evaluation	53
Usability Testing	53
Evaluation	55
Acceptance Testing	56
Evaluation	56
Evaluation	57
Conclusion and Recommendations	58
Reflection	60
References	61
Appendix	64

Table of Figures

Figure 1 Hybrid layout on android device	13
Figure 2 Hybrid Layout on ios device	13
Figure 3 Website before converted into hybrid app	14
Figure 4 Revenue taken from each platform	16
Figure 5 User experience elements	18
Figure 6 Example of screens from amazon app	19
Figure 7 Difference of appearance portrait to landscape	20
Figure 8 Different types of testing	28
Figure 9 Prototype Model	32
Figure 10 Storyboard of the application	38
Figure 11 Class Diagram	39
Figure 12 Gameplay Use Case Diagram	40
Figure 13 Collision Use Case Diagram	42
Figure 14 Collision-State Chart Diagram	43
Figure 15 Entity Relationship Diagram	44
Figure 16 Update Score Use Case Diagram	45

Table of Tables

Table 1 System Testing Results 52

Table 2 Usability Testing - User 1 Findings 55

Table 3 Usability Testing - User 2 Findings 55

Aims & Objectives

The aim of this project is to create a successful working and fully tested game application to be run cross platform. As a developer it is important to take into account of what knowledge and skills you have and how these can be used to complete your goals.

➤ **Objective One – Investigate Cross Platform App Development**

Research will be needed to look at how apps have become more popular over time, how they are created and to see whether software has been created and is available to use to make a game app and make it cross platform. Also it will compare different cross platform software and why I have chosen Xamarin over the others.

➤ **Objective Two – Investigate Testing Techniques**

Research will also need to be carried out to see what different kind of tests are available and can they be implemented through the use of Xamarin and how each can be beneficial throughout development.

➤ **Objective Three-Develop a game to demonstrate test Techniques**

A game will be developed through the use of Xamarin which then this can then be used to run tests that will help make the game run smoother and help find any bugs so that users doesn't have a bad experience when playing.

➤ **Objective Four-Evaluate Test Results**

Results will be taken from each of the test that will be carried then evaluated to see if they were successful or unsuccessful.

Introduction

A mobile app is a computer program that can be run on smartphone devices. Mobile applications are used to do virtually anything you like for example they can be used as a web browser (shortcuts to websites or quick way to look at information on a specific topic), listen to music, play games etc. The use of mobile apps only becomes more popular through the rise of use of smartphones that has become more and more popular over the years. Many users have switched their mobile phones/devices to smartphones because of the interactivity you can have whilst being in public whether you on public transport, at work or just simply out. This allows business to take advantage and to create an application to suit both them and also the consumer as they can now promote their business through applications [1].

Other contents within this report will discuss the research and literature review this will be so that you can witness what research has been carried out and what results were gathered this will be to see what information is currently produced and also to see the project is worthwhile. Research and development methodology will have an important role in this as it will explain in detail what ,how and why certain procedures was used in order to generate a suitable solution to the idea. An design and implementation stage will be required so you can see the requirements that needs to be met before it can be implemented .Here it will discuss what tools and techniques were used and also what difficulties occurred and how these were overcome.

Literature Review

History of App Development

The first applications that was used on mobiles was the use of the calendar, a few Java games and a calculator. The first smartphone was presented in 1993 by IBM which then introduced a modern way of including a touch screen instead of buttons. They decided to add additional features such as calendar, address book, world clock, calculator, notepad, email and games. This achievement allowed companies to progress and in 2002 RIM the blackberry smartphone introduced the first mass market smartphone optimised for wireless email. In 2007 Apple then released the first iPhone that had preloaded apps which included maps, photos, texts and weather. Only in 2008-2009 apps started to get really popular as apple and android then first launched their app stores allowing people to download apps straight to their phone.by 2010 there were over 1 billion downloads from both markets. The Key advantage of android and apple introducing stores was that app

developers could develop their own app and could pay a fee to those markets so that people can view and download their app through the use of android and apple store. By 2017 it is predicted that there will be over 200 billion apps downloaded [2].

Apps

There are many different types of apps there are native apps, web apps and hybrid apps.

Native Apps

Native apps are installed onto your device which has been downloaded from the store such as apple app store or android store. These are specifically created for one platform only. These can use all features that are already on the device such as the use of camera, list of contacts etc[3]. Native apps refers to applications written in a computer language for a particular smartphone/device. This will allow the app to look, feel and behave to the standards of that device. By allowing this users can grasp using them very quickly due to the conformity to the platform interface. An example of a native app are those that are already pre-installed on your device from new.

Building native apps requires unique code for each platform you decide to create an application for. To build an app for Apple mobile operating system(IOS) you will need to use objective-c, for android you will need java and for windows you will need C#. If you wanted to create the same app for two platforms then the development cost and time to create one will most likely double or increase if you are not skilled enough(don't know to use that particular computing language for another platform) as you will need to recode using another language[4].

Web Apps

These are not your typical application these are just websites that have been implemented using html so that web browsers can look and feel like native apps but don't have the necessary features that can be used via the browser [3]. The display of information is important as the content of many websites are detailed so by creating an application for it will look unprofessional and can be very hard to read and use for the user. The use of HTML 5 and JavaScript will allow developers to create applications that are much interactive and present the information more professionally. Mobile web apps won't look or behave like native apps but it will offer a good user experience.

The cost to develop a mobile web app is approximately the same for developing a single native app. The cost advantage of creating a web app is that a web app can cover all platforms which supports the use of HTML 5[4].

Hybrid Apps

Hybrid apps are part native and part web apps. Like explained above this is where html has been used to create a web app and has been configured through the use of PhoneGap or Sencha Touch to make it cross platform. These are accessed just like native apps through the use of apple or Google store. However when this has occurred although the website will be the same, on each device the website will appear differently for example [3].

This the app run on an android device



Figure 1 Hybrid layout on android device

This is app run on ios device

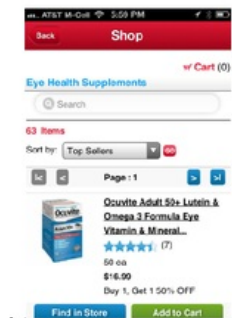


Figure 2 Hybrid Layout on Ios device

As you notice the top the page each device renders the website differently android has an arrow whereas apple will have a back button.

This is what the actual mobile website looks like this is before it has been converted from PhoneGap or Sencha Touch.



Figure 3 Website before converted into hybrid app

Many apps are created through the use of html which is a web based language and are usually used for web browsers and is the mostly used today to create apps because of the popularity of the internet .HTML has components which allows it to be reused on different mobile operating systems .All websites today use html, so tools have been used such as PhoneGap or SenchaTouch to make these websites across platform. This is why it is known apps are created via html. It is important today to have cross platform apps because of the competition on the market. The more devices that can use your application the bigger your audience become and therefore attract more people to use your app and also increase your revenue. Each device has its own programming language for apps to work on that device android uses java and ios uses objective c this is why PhoneGap and SenchaTouch has been used to allow people to design and code across platforms through the use html as this language is compatible with these software's. Although the use html can be used to create there are now many other methods/tools that can be used to produce a cross platform app using different types of programming languages.

How do they work

Apps can be used on smartphones or devices that have connectivity or access to the internet. Although some apps do not require internet, as most devices today have pre-installed apps these of which are accessible on the phone at all times. Apps today requires internet because of the Apple, Android and Microsoft operating systems which these offers stores on the device which allows users to browse, download and install apps onto their device[5]. Through the store you can buy and download apps for free. By Offering your app on the store will mean charges will apply as a certain percentage of money that you make with the app will go to the distribution provider and the rest will go to the producer. The

price is different depending on the platform you want to choose. Some of the app store fees can range from \$99 a year for apple app store and google play(Android) has a one-time fee of \$25. The standard industry apple and android take is 30% of the income that the developer gets from the app. Also you don't get your money immediately when you make a sale android and apple holds onto your money for a period of time[6].

Android vs. IOS

Building an application for both iPhone and android can be challenging. Both platforms have many advantages and disadvantages these of which you got to account for when building an app. Even though there are other platforms such as windows and blackberry. Android and IOS account for 90% of the smartphone market share alone. This is why it is most likely business will chose either or both to create their app for. Currently Android own the current market share at over 80% while apple is around 15%[7]. Even though android dominates the market it is not the most profitable ,IOS apps are more profitable. This is because generally Apple (IOS) products are more expensive and therefore attract people who are willing to spend more money on apps or in app purchases. Also another reason is that IOS doesn't market share in developing countries. This is because most people in these countries do not have a credit card which means they aren't able to purchase any apps as when people first use their device it is forced that the user enters their credit card details so this is another factor on why more purchases are occurred with apple. Another reason that fluctuate the market is also due to android being slower than IOS to adopt a carrier billing system so apple had the advantage when making money. Since IOS is a closed system this means that there is a lot harder to pirate apps whereas android is an open system and people can steal the app and can get away without paying for them. Here is a picture of the revenue and apps download of each platform [8].

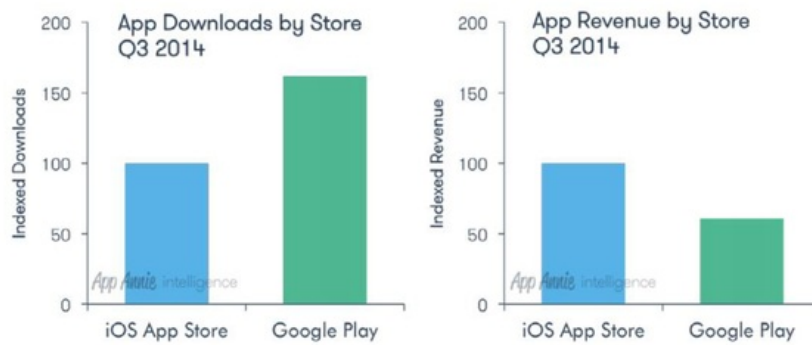


Figure 4 Revenue taken from each platform

The breakdown of IOS and Android are as follows.

IOS –Advantages

- Increased revenue which will mean it will be more profitable.
- More advanced tools and support
- More secure as it's a closed system

Disadvantages

- A approval of an app can take longer than two weeks
- Fees are much greater
- More restrictive on data sharing,advertising,and iteration methods
- Can only be on apple products/devices

Android-Advantages

- More open platform which means more users and therefore downloads.
- No approval is needed
- Broader reach , in terms of devices android can be on a number of different devices and not just one device.

Disadvantages

- Harder to maintain due to wider variety of form factors and operating system version.
- Higher chance of errors and piracy [9].

To choose either of these platforms depends on the type of app you want to create as both offer good value points. If you looking to make a profitable app then IOS can be your choice, But if creating an app for general use i.e simple effective game or application to do something quickly then android. But by doing cross platform you getting both advantages then this can potentially mean your app can be successful on a greater scale.

Identifying Users

When building an application it is always good to know your potential users. As this can play a key part in your design and functionality of the app knowing your audience can give you a range of ideas that can make your app successful. Today nearly all the people that use smart phones/devices will know how to operate a app but this doesn't mean that yours can be created with complex functionality that only some users knows how exactly how it works you still need it to be very user friendly so that all users can use and also that it can attract them to use it more. The main purpose of an application is to be simple to use , gain interest and get a name for itself. It will need to be created so that users will use it more than on a few occasions as if created badly they could look for another similar type of app that is currently on the market. Then this will make the app unsuccessful.

Why are mobile applications important

People today are always on the move and are always using their mobile phones/devices. Most mobile phones today are smart phones which mean they have the ability to use applications on their device. Most devices today whether it's a phone or tablet can have access to a lot of information. Therefore business can take advantage of this and therefore apps are so important in today's market.

For business this is a great way to attract, gain and also retain your customers. As today most people first look for a product or look for a service or search any necessary information is online. By creating a application for a business allowing users to download on their device will give a really good impression and also allow them to see and open your app on device and make any purchases they want [10].

Some of the main reasons business can benefit from making a application for their business are:-

- Build loyalty with their customers
- Increase your visibility which could potentially mean increasing your customers

- Increase your accessibility allowing customers to gain quicker access to your services you are providing
- Increase exposure across mobile devices
- Customers can get notifications if offering special discounts or services/events
- Get directions to your location from wherever they are
- Increase Revenue
- Have one touch access to your information [10].

Mobile User Experience

The mobile user experience is important when creating an application. It is important to consider what complications you will face when creating your app because of the constraints of the small screens , variations of in device features, connectivity and also the ever changing mobile context.By breaking these elements up will give you a conceptual framework to work with which will help build and evaluate a good mobile app. Here an image to show some of the elements.



Figure 5 User experience elements

All of those elements can change depending on the type of platform you chose. You must ensure the functionality of the app achieve its goals. These can be done by setting tasks and ensure they can be completed. Some features include navigation, search labelling etc.

The main features and content on the application must be prioritized to the user needs. Just add the features that users are mostly going to use. Also can the content be represented if they use the device vertical or horizontal? Also they must be able to access the important content within the app within a few taps or key presses as possible. Another aspect to think about is that address the navigation so that users can use the app via both touchscreen and non-touch screen. When designing for touch, you need to make sure the

navigation link or item must be 30 pixels wide or tall. Could also think about adding keypad navigations that act like shortcuts for non-touch users. The app must have clear, consistent and descriptive labels for navigation items or links.

When designing the app it is important not to add too much information i.e. content. If you are re designing a website that has a lot of content just consider the information you want to add as having the same amount of data displayed on a small screen will mean that everything can be miniaturised and look unprofessional and may put off many user from using the app. You want to design the app for glance ability and quick scanning, you want the visual design to be simple so that it is very user friendly. You also need to maintain visual consistency throughout the app for example amazon is known to be associated with the colour orange, white and blue. By only using those colours you know the app is consistent and will look more professional [11].



Figure 6 Example of screens from amazon app

Here an example here you see different screens that taken from amazon app.

Also when designing it is a good idea to outlook the design by having a portrait and landscape display. This is too ensuring maintain the user's location on the page that they are currently on and what they are doing. You can also change the functionality of the app for example



Figure 7 Difference of appearance portrait to landscape

This is the same page one portrait and the other landscape both showing different functionality of the app [11].

Most Popular Apps being used today

When it comes to the mobile phone markets today it always good to have a look to see which are most grossing and why. As of march the most apps downloaded was game type apps which made up of 22% of apple market. The purpose of this is to take into consideration when creating an app for the first and you are unsure on what to create many developers have the knowledge how to create but not what users are looking for. Game apps are at top and are most profitable because they are usually free to download but consist of in game purchases which allow players to progress quicker or be stronger than other players. If a successful game is created then add on can be added making the game better and better. Most popular games today are those which allows players play with friends or other players around the world .The objective of the game can be many things but the most used are those which are competitive for e.g. flappy bird a simple and effective 2D game which was a competitive game that made players want to beat each other score. Other games may consist of building a base which players can attack players or defend themselves other players. These games are usually most profitable because to build buildings it will take hours to build so an in purchase feature could be items to speed them up which gives the player an advantage over others an example of this clash of clans, the most popular game at the moment[12].

Cross Platform Software

Over the years the market for cross platform apps has increased dramatically. This is because large and medium sized companies are looking to adapt to the mobile industry. When choosing a suitable software you must consider why is that a suitable software to use, as some focus on gaming and others specialize in a specific programming languages

meaning that it can be more useful if creating a app within a team or if you specialize in that language. You must be familiar with the programming language if you are going to use it. Some of the top cross platform tools that are available are phoneGap , Appcelerator titanium and Xamarin .All of which offers many different features and use different programming languages[13]. A problem building applications for each device i.e iPhone, Android , windows all require different frameworks and languages, by using different cross platform software we can solve this problem.

PhoneGap

Phone gap is an open source framework for quickly building cross platform apps using HTML 5 , JavaScript and CSS. This software allows users to create apps by using open web standards. They will compile the code for you and also maintain the code for you as it stored using cloud storage. All apps created will have the most up to date SDK. It allows you to add team members so that you can work together to create the app so this will speed up the ime to create the app.it uses the hydration feature so you can debug your code faster. This will be an advantage as you can then test your code faster [14].

Pros

- This particular tool supports HTML , JavaScript and CSS.so those who have experience with this language would be able to use this tool.
- These apps install just like native apps and gain store discoverability
- It is open source and free so there are no licensing costs
- It is guided by the use of plugins enabling developers to focus on web based skills they already have.

Cons

- by this being free doesn't mean guarantee of success as some plug in may be out of date or the custom ones may not support the target platform you need so it's not reliable.
- The plugin architecture works well if you can find the plugins you need.
- The performance can be not that great if you building a app that is game related that uses a lot of graphics [15].

Appcelerator titanium

Appcelerator titanium uses JavaScript to develop and support apps for IOS and android only. This particular framework compiles the code by combining the java code with the titanium API which is the native device language, and evaluates your code at runtime with a JavaScript interpreter that runs on the device's operating system. When the app is launched JavaScript execution is created in the native code and the JavaScript is injected as an inline object paired with native objects within the runtime environment. The titanium API works like a bridge which allows direct access to the native features and API [16].

Pros

- With the use of rapid prototyping it allows you to create a prototype very quickly allowing you to evaluate the user's interaction.
- It is written in JavaScript so developers with knowledge of this programming language can be able to use it.

Cons

- It has increased complexity this means that the more complex your app is the more chance you will get technical issues such as random crashes, weird behaviours, annoying bugs etc.
- It has no freemium in app purchases is buggy and can be a pain to download.
- Apps created can be laggy when running [17].

Xamarin

Xamarin is a library that uses a set of APIs to access common mobile device functionality that are used through Android, IOS and windows platforms[18]. The apps are built with standard, native user interface controls, so that they will look and behave like the user expects and run and feel like a native app too. They have the full functionality to the platform device as they have native API access. The performance will be very good as the code compiled will be for native performance [19].

Pros

- Xamarin strives to be the only framework that covers all of the major platforms such as IOS, android and windows
- The quality and performance is of a high standard
- C# is used to code for all platforms

- It has a simple API which allow developers to build native UI for all the platforms[20]

Cons

- It uses numerous open source libraries for IOS and android, so if you wanted to create a cross platform app for these platforms then it will require unique UI code for each platform.
- If you using this tool/software to create an application for a specific platform then tool will not necessary be useful. As other tools/software will offer more UI controls for particular platform [21].

Although Xamarin allows you write apps in c# and shares the code across multiple platforms, the actual implementation is entirely different. For each device the code is compiled differently ios is compiled to ARM assembly language and Android is compiled to IL. Xamarin makes the features provided by the platform specific SDK. Xamarin exposes each device SDK by using namespaces so you can reference any part of the supported SDK with a using statement. For IOS it using MonoTouch.UIKit: and android using Android.Views;

Developing IOS applications on Xamarin still requires a Mac computer as it's needed for build and licensing purposes. Android applications requires the java and android SDK to be installed. These allow the compiler, emulator and other tools that is required for building. These of which will run on windows 7 or higher and has either the Xamarin studio IDE or with visual studio 2012 , 2013 or 2015[22].

Developing Games With Xamarin

When developing games you can be confronted by new concepts and development patterns. These concepts concerns the game loop, Maths and also content pipeline.

The game Loop

A typical game requires constant movement this can be any changes on screen such as user interaction or the automatic gameplay. This is done by the game loop. The game loop is a statement which describes the steps of how the gameplay works. A simple game loop starts when the game is initialized. The first step in the loop is input, then game will perform the necessary movement, check for collision, check for any game state change and perform the rendering then repeat this process.

When coding the performance of each function could take milliseconds to execute and this could potentially effect the performance of a game. Depending on your frame rate per second each time a function is run especially if called more than once then this could create a lag which users will find annoying and may not like the game because of this. When building a simple game, a function may not be noticeable if it only executes in response to on click or a button [23].

There are different types of logics that could be performed every frame and which functions are used to perform the necessary actions that needs to be carried out. There are:-

Reading input – This is where the game checks for any user input from such as touch screen, keyboard, and mouse or game controller.

Movement – Any objects that are moving or spawning.

Collision – This is where you check two objects are overlapping or intersecting

Checking for game specific condition – This can be the state of how the game is controlled by certain condition for example player earned enough points to unlock new items, make game harder etc.

AI Behaviour- This is the behaviour of objects which are not controlled by the player

Rendering – This is where the game updates itself from frame to frame. Any changes within the game then this will update to show changes that the user or game has made.

Math

Many games require math that's uses X and Y coordinates to position objects within a game. The level of maths depends on the complexity of the game you are going to create. In 3D games usually requires more maths than 2D. Maths is used so that you can calculate speed, positions, player movement, enemy movement etc. Without any maths the game wouldn't be very pleasing to play [23].

Content Pipeline

Content pipeline refers to the process that a file takes to get from its format such as a .png image file to a final format so that it can be used in a game. Some content pipelines are very fast and require no manual effort. This depends on the technology that is being used and also if you are using complicated formats such as 3D models then these are much larger and are more complex than using images [23].

Game Development Technologies

Developing games using Xamarin has a wide list of technology to help assist in game development some of the most popular ones that are used are CocoSharp, Monogame and UrhoSharp.

CocoSharp

CocoSharp is an open source cross platform version of Cocos 2d game engine. The engine allows the developer to have access to Android, IOS, Mac OS X, Windows and Windows Phone.

CocoSharp uses a simple programmer API which is used for 2D game development. It provides graphics, audio, game state management, input and a content pipeline for importing assets. Cocosharp provides a mix ease of use and performance which allows developers to get started quickly on creating their game without having any worries of the size and complexity of their game.

The growth in 2D games on mobile devices has risen considerably making this the most popular technology for creating 2D games. The only disadvantage of using this is that it doesn't give you a visual editor so any interaction with this is all code based so you must have a good understanding of programming [23].

Monogame

Monogame is an open source cross platform version of Microsoft XNA API. The engine allows the developer to have access to Android, IOS, Mac OS X, Windows and Windows Phone.

Monogame is a game development API not a game engine. This means if using this technology then you will need to manage game objects, manually draw objects and implement objects such as camera and scene graphs. If using this then you will be building a game from scratch purely based on knowledge of programming. This can be difficult for developers with no knowledge of game development [23].

UrhoSharp

UrhoSharp is a high level 3D and 2D engine that can be used to create animated 3D and 2D scenes [23].

Testing

The general aim of testing is test the quality of the software. This insures that the system is bug free and results gathered are as expected. This is used to check the validation and verification of the software/application. By introducing tests in the early stages this will mean that you will reduce the testing time overall, also improve the quality and the time for it to be produced. There are different testing techniques to target different types of faults. These are a few:-

Code Based Testing

Code based testing (also known as white box testing or structural testing) .This allows developers to test parts of the software that is rarely tested and ensures the most important functions have been tested [24].

Object Oriented Testing

OOT (object oriented testing) focus on integration and system testing. All methods within the class are tested with unit testing, events between classes are tested with integration testing [24].

Component Based Testing

Component based testing is an assembly of reusable components designed to meet the quality attributes identified during the architecting phase. Different components can be either white box or back box testing [24].

Specification Testing

Specification testing also known as black box testing) is a testing method which will allow the tester to get an idea on how the internal structure, design or implementation will work. This usually tests the functionality of the application. This is where the tester inputs data and get vs. the output then it will be checked the output with the results specified in the test case. This is usually required due to code unavailability, scalability, accuracy and effectiveness [24].

Manual testing

This is where the user tests the software to identify any unexpected behaviour or bugs .There are different stages used and use test plans, test cases or test scenarios to identify errors in the software[25].

Automatic testing

Known as Test Automation, this is where scripts are written to test the product. Also another software can be used in the testing phase. This type of testing can save time and money and also improve accuracy [25].

Black Box Testing

The tester in this method has no knowledge or access to any of the software source code, he/she does not have any knowledge on how the software execute commands within the software interface. The tester will input data into the system and will observe the outcome. This method is really suited for software that has large code segments. Some disadvantages of this is that it has limited coverage of testing as not every scenario within the system will be fully tested [25].

White Box Testing

White box testing is the actual programmer who designs and creates the software. He/she needs to know the internal structure of the software and how it should work and performs. He/she will need to look at the source code and decide which part of the code is not working to its function and act upon it to test and fix this. This help to optimise the code and this could remove extra lines of code which will reduce any defects. Disadvantage of this is that you will need a skilled programmer to perform this method [25].

Grey Box Testing

This type of method is similar to white box testing but as limited knowledge of the internal workings within the application [25].

Black-Box Testing	Grey-Box Testing	White-Box Testing
The internal workings of an application need not be known.	The tester has limited knowledge of the internal workings of the application.	Tester has full knowledge of the internal workings of the application.
Also known as closed-box testing, data-driven testing, or functional testing.	Also known as translucent testing, in the tester has limited knowledge of the insides of the application.	Also known as clear-box testing, structural testing, or code-based testing.
Performed by end-users and also by testers and developers.	Performed by end-users and also by testers and developers.	Normally done by testers and developers.
Testing is based on external expectations - Internal behavior of the application is unknown.	Testing is done on the basis of high-level database diagrams and data flow diagrams.	Internal workings are fully known and the tester can design test data accordingly.
It is exhaustive and the least time-consuming.	Partly time-consuming and exhaustive.	The most exhaustive and time-consuming type of testing.
Not suited for algorithm testing.	Not suited for algorithm testing.	Suited for algorithm testing.
This can only be done by trial-and-error method.	Data domains and internal boundaries can be tested, if known.	Data domains and internal boundaries can be better tested.

Figure 8 Different types of testing

The image above shows the comparison between the black box white box and grey box testing methods [25]

The two main types of testing are:-

Alpha Testing

This is the first stage of testing and it is performed by the team who develops the application, it is made up of the developer and quality assurance team. Unit testing, integration testing and system testing is all combined and this is known as alpha testing [25].

Acceptance Testing/Beta Testing

This is performed after the alpha testing has been successful. This is where the actual software/application is at its final stage and will be available to a group of people who it's intended for. This is to get feedback on any issues that they have come across within the software and report back to the developers with the issue so it can get resolved. The purpose of this test is so that people will try different actions within the software that programmers haven't tried or found out that that's possible within the system which will cause an error or cause the application to crash.

After this stage any errors found if any will be fixed and then the software will meet its specification and then it will need to be verified and validated. Quality assurance and quality control will then determine whether it will meet the customers' requirements and standards [25].

Usability Testing

This is used to test how easy your application is to use by testing it with real users. Tasks will be set that will need to be completed while they are being observed to see if they encounter any problems or if they get confused on how they do it. This will then get documented and if the same problem occurs then changes will need to be made to overcome these issues [26].

System Testing

This is used to test the behaviour of the whole system defined in the requirements. The purpose of this test is to verify the customer requirements to see whether they are met or

not. This confirms the completeness in performing the designed functions to insure they are accurate and that they carry out the function as they should [27].

Seven Principles of software testing

There are seven principles of software testing they are:-

Testing shows that defects are present in the software

This is where testing a program to make it fail. Using the verification method, this method tests it to assess the quality and to trigger failures. This tests a program only and finds the faults but does not correct them [28].

Testing an application alone does not show it is error free. It is Important to design test cases which finds as many defects as possible [29].

Exhaustive testing is not possible

When testing the software there is no exhaustive test that considers all possible parameters within the software to be tested. Testers will need optimum time to test an application. To resolve this, risks and priorities are considered when performing tests [29].

Pesticide Paradox

When the same tests are repeated continuously and these tests do not find any new bugs then this is known as a pesticide paradox. A review procedure of different software tests needs to be implemented to find new error/bugs within the software [29]. Anytime a fault is fixed or new function has been added, a regression test is required to make sure that the new changed software has not broken any other part of the software [29]. This checks that what has been corrected, the software still works and what has been corrected does not re-appear in the software [26].

Early Testing

Early testing can start as soon as the requirements and design documents have been created and are available to the software testing team within a working environment. By preparing tests these will help create test for each level of the development cycle. Also it will help reduce costs as changing a requirement are much easier and cheaper to fix rather than changing the whole functionality of the software [29].

Also applying oracles can benefit in early testing. An oracle is a criteria of a test run that determines if it has passed. Determining success or failures of tests must be an automatic process and should be part of the program text. This can help in the early testing phase of the software [28].

Testing is context Dependant

Different applications will require different test methodologies. For example a software application for a medical device will be more complex than a game software. The game software will require minimal testing whereas the medical software will need to be tested extensively as this type of software is more risk dependant and needs to be more stable and reliable[29].

Defect Clustering

During tests it can be observed that most of the reported defects are related to small number of modules within a system. Following the Pareto principle in software testing it is known that 80% of the problems found are contained in the 20% of modules within the software [29].

Absence of errors fallacy

If testing was carried out and did not find or found/fix defects then this does not mean the software is ready to be used for its purpose. Other factors need to be taken into consideration before a decision is made for its completion. Other principles can be noted examples are:-[29].

- Keep Software Static during test
- Provide expected test results if possible
- Testing must be done by an independent party
- Test for invalid and unexpected input conditions as well as valid conditions.

Research and Development Methodology

Research Methodology

The research onion contains different philosophies, approaches, strategies, time horizons and techniques and procedures. The philosophy of this project will be positivism with a deductive approach. Following this approach the strategies are software development and document review. The document review is completed before the software development stage as the development stage will be using a development methodology which will explain how the project will be successful. By following these strategies it gives the project to see if it's worthwhile for the development of the project.

The Advantage is that the document review will contain is it necessarily to make an app cross platform and discuss what software is available and can be used within the developer capability .Also it will contain what tests can be used throughout developing the game application.

The disadvantages is that when users play the game it could be buggy and could make the game really unlikable , by performing tests throughout this will minimise the bugs so that the game can run more smoothly.

Development Methodology

The development methodology used is the prototype method .This methodology allows opportunities to revisit each aspect of the software development life cycle, development, requirements, design etc. The prototype is a working model of the application with limited functionality. By breaking down the application it allows users to evaluate and gain good feedback this will help make improvements and if changes are needed then these can be changed before working on the next phase of the application the purpose of this is to make sure each phase is in full working order before moving on the next phase. By doing this it will help to fully understand the requirements and also give you more ideas what can be added in that phase to make it better. The step taken to produce a prototype is rather simple.

1. You will need basic requirements
This step will help you understand the basics of the application.
2. Develop the initial phase

This is where the requirements have been implemented to produce an application with a user interface with basic functionality. This is so that the user can use it and get the look and feel of the app and get a better understanding on how it works.

3. Review

Once each phase is developed it is then presented to the customer. From here feedback is collected and is used to make improvements if needed.

4. Revise and enhance

The feedback collected is discussed by the customer and the developer to see if it's worth doing as in all projects there is a time limit and also a budget which as to be taking into account. When changes have been made then the cycle process will repeat until the customer expectations have been met [30].

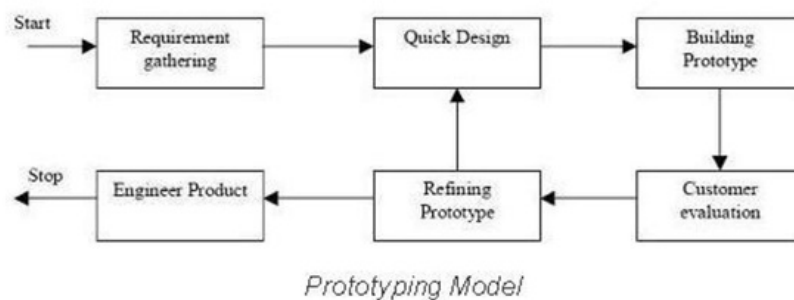


Figure 9 Prototype Model

The main advantages of this is that you can breakdown the overall application into phases this is suitable for this project as on each individual phase tests can be implemented and can gather results resulting on how each the test performs these will be discussed in the results.

The only disadvantage of this is that it may increase the complexity of the overall application as newer requirements may be added. This may be time costly and potentially make the project unsuccessful. To manage this it will need to be monitored carefully what added functionality there will be and only add those which will be less time costly.

Prototypes will be created to breakdown the overall system this will make the project much more successful as each breakdown can be reviewed and make any changes when required before moving on to the next. This will also allow testing to be made on each stage so that it will ensure there are no or very minor bugs before moving on to the next. This will make the application much more professional when the final product is created as it will be fully tested so no more tests will be carried out by the producer. There are six stages to this project these are:-

- Title Screen
- Instruction Screen
- Leader boards Screen
- Game over Screen
- Objects in game
- Gameplay

UML and sequence diagrams will also be created to help design how the game app will be implemented. This will also show how the game will run by explaining certain actions that the user will make when playing the game app. Also storyboards will be created to show in general how the user interface will look. This is a really good way to see the first visual representation of the all screens within the application and also help usability

Software

The software that will be used to create the application will be Xamarin. Xamarin has a library that uses a set of APIs to access common mobile device functionality that are used through Android, IOS and windows platforms. This allows apps to be built with standard, native user interface controls, so that they will look and behave like the user expects and run and feel like a native app too. Xamarin is mostly used today by companies to create cross platform applications. The goal of the project is to create a cross platform app and therefore this is most important software used and widely suitable for this project.

Other software that needs to be used is Photoshop this will be for any graphics that will need to be included in the application. Any graphics will be created using this software and is mostly reliable.

Visual studio 2012 or higher will be need for the Xamarin software to be used. It is needed as Xamarin will need this for it to work.

Hardware

A computer, with an operating system of windows 7 will be needed. This is important as without this Xamarin cannot be installed.

Materials And Tools

Xamarin uses a tool that requires the c# syntax. C# is considered as powerful and flexible programming language. it can be used to create a variety of different application. It doesn't have any constraints that stops you on what you can do with it. It is an object orientated language which allows techniques such as inheritance and polymorphism which makes your code easier for you. It is also modular which means code can be written by using classes which makes your code easier to use and is more understandable for other users if used appropriately [31]. This is suitable as C# is one of the required skills that is needed to complete this project. Throughout studying at Swansea Trinity St David c# is one of the main skills that has been learnt and by using this a solution can be developed to build a suitable cross platform app and make this project successful.

Design

About The Game

The game created is a simple 2D graphic based game. The game involves a character which the user will control by simply pressing their screen on their portable device which will perform an action that will make the character to jump. The object of the game is to jump over many obstacles as possible as each one successfully jumped will add one to the player score. The addictive side to this is that the game will save the player high score and add it to a list of leader boards which they can compare it with their friends to see who is the better player.

Scope of the Game

This report will describe all the requirements for the project. The purpose of this is to provide a suitable solution to show in a structured way how the game works by providing numerous diagrams to show the functionality of the game and talk about each in detail.

This will allow the client to get a better understanding of the project.

Application Features

- Title Screen
- Instruction Screen
- Leader boards Screen
- Game over Screen
- Objects in game
- Gameplay

Title Screen

Description

The title screen is the screen which will display every time when the game is launched .On this screen the player can choose to start a new game, view the instructions or view the leader boards. Because of the interactivity on this page the priority of including this is high.

Functional Requirements

- The title screen must appear every time the game is launched
- If the player quits the game in mid play then they will be redirected to this screen
- Once its game over the user will be redirected here if they decide not to try again.

- Render three button “start game” , “instructions” and “Leader boards”
- Display background image

Instruction Screen

Description

The instruction screen will show a description on how to play the game. Also they can go back to the title screen from here. Since this is involved with user and that it needs to be included to explain how the game is played the priority of this is high.

Functional Requirements

- Render a button “Go Back”
- Display background image
- On click of the instruction button from the title screen , this screen will need to be displayed
- Display a label which will display some text

Leader Board screen

Description

This screen will show a table with all the player and players’ friends that plays the game and display their scores. This is also one of the main activities the player can do but can be still able to play the game without this function so this is set to a medium priority.

Functional Requirements

- Render a button to “Go Back”
- Display background image
- On click of the leader board button from the title screen , this screen will need to be displayed
- Display a table

Objects in Game

Description

These are the objects in game such as the character , boxes and the platform(street).These object are important as these will be displayed and used in game. These particular are used as these determines how the game is played. The character is what the player controls , the street and boxes will have functionality to make the game work as these will be moving to create the scene of the game(i.e. make the character look like he running with the

surrounding changing). This is one of the major factors of the game so this is set to high priority.

Functional Requirements

- The character will need to look like he is moving by using the animated sprite feature
- The box and platform will need to move simultaneously
- Random boxes will need to display
- Speed of the box and platform will need to increase

Gameplay

Description

The game itself contains the main functionality of the game as this is where the player will have most interactivity. This is where the player controls the character by touching the screen on their device. This will allow the character to jump over boxes and to gain and beat their high score. Here they can also pause the game and when ready can continue simply by touching the screen. Also they can exit the game. Once the character has hit a box then it will be game over. This is set at high priority due to the functionality involved with the user/player

Functional Requirements

- Render all objects in game
- Display Background image
- Render a pause button
- On click "pause" Button the game will freeze until the user click back on screen or if click exit redirect them to title screen
- On click of touch screen the character will need to jump
- Every time the character successfully jumps over object the score will increase by 1
- If character hits a box then it will direct the player to game over screen.
- Display some text when game is paused

Game over Screen

Description

The game over screen will display once the game has ended. Here the player can see their score and has a choice to play again or exit the game.

Functional Requirements

- Display background image
- Render two buttons “play again” and “exit game”
- If user clicks on play again then the game will start again
- If user click exit game then they will be redirected to title screen
- Update their higher score in leader boards if they have beaten it.

Overall Non-Functional Requirements for Application

- The game must be easy to use
- The game should take less than 3 seconds to load
- The game should have consistent colours that are
- Game should have a user friendly interface
- The game will be self-explanatory

Scenarios and Use Cases

Storyboard

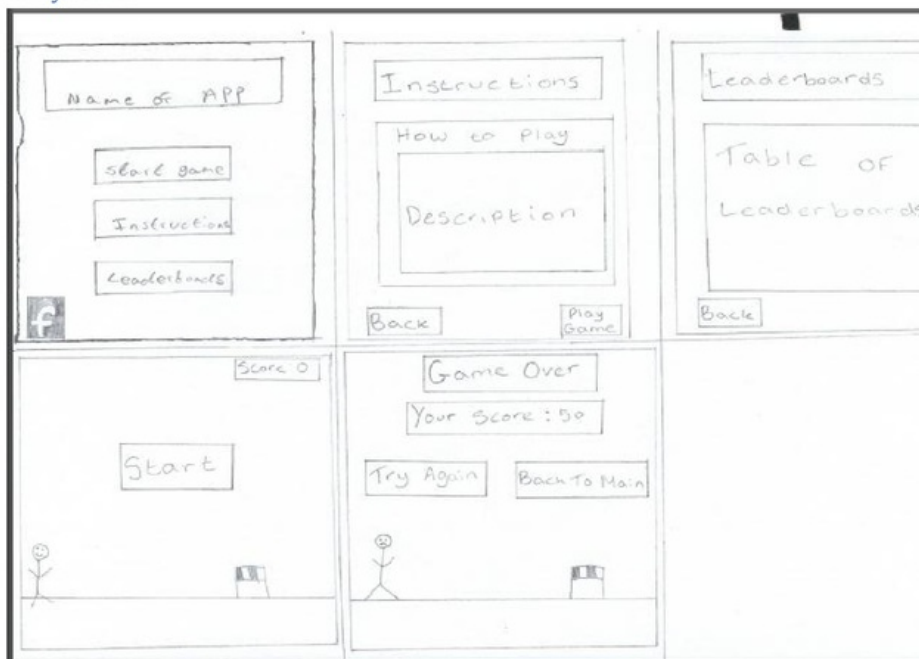


Figure 10 Storyboard of the application

A story board was created to show the overall first view of the game , it is a simple diagram just so that the user wil have a basic idea on how the game will look and how it will work.This has been showed to a number of people whilst asking them questions can they

interact with the game by performing actions ive asked.all users had no issues directing through which means it had a very user friendly interface.

Class Diagram

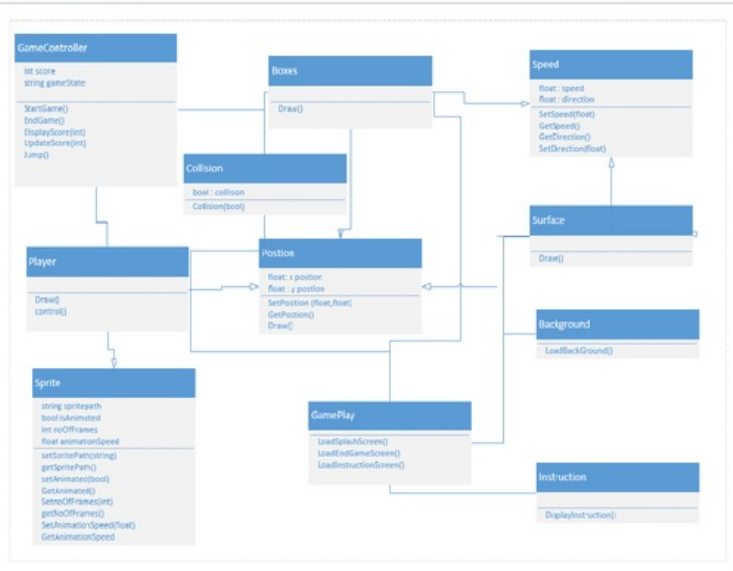


Figure 11 Class Diagram

To represent the whole system as a whole, a class diagram has been created to show the back end of the application. This representation shows a number of classes with their attributes and methods that will be needed for system to work. It also describes how each class works with other classes and how such techniques as inheritance will be used throughout. This is a general idea how the overall system will communicate and how it works.

Gameplay Use case diagram

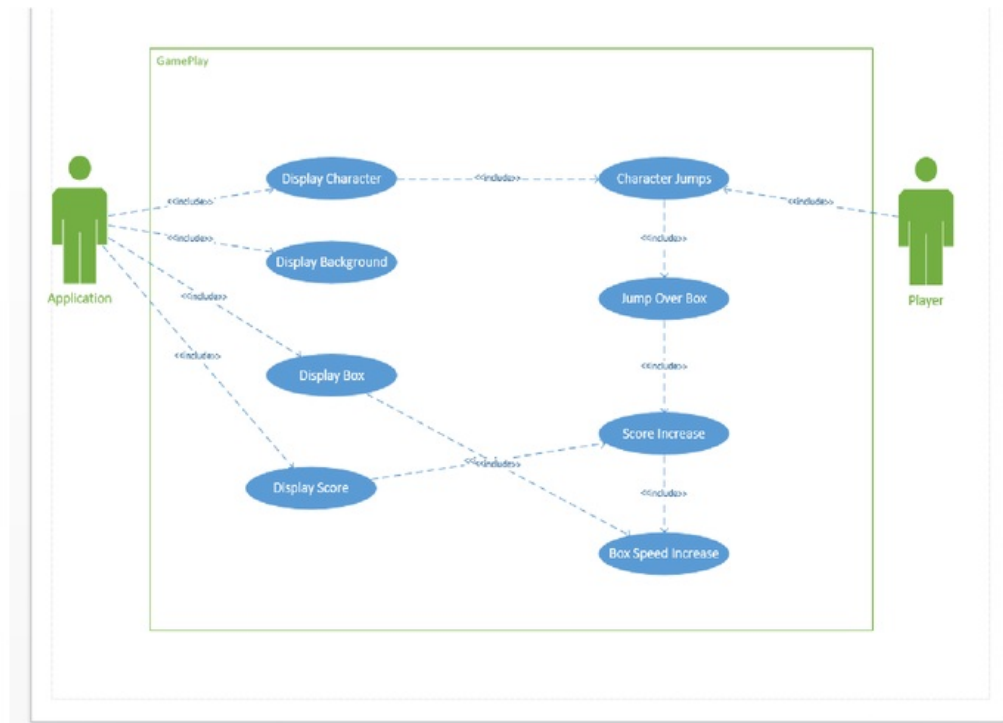


Figure 12 Gameplay Use Case Diagram

Use Case Documentation

Goal: Game display all objects will display and all controls works as expected

Description: The user will start the game and all controls and functionality will work accordingly. User can jump and each successful jump the score will increase by 1 and for every 5 points the box speed will increase.

Frequency of Execution: This should happen every time a new game will start

Scalability: each instance of the application one user only

Criticality: If game doesn't load properly or any errors were to occur whilst playing the game will need to be reset. The user will simply need to close the app and reload it.

Primary path: These steps are carried out by the users each time they start a new game.

1. User starts new game
2. The game screen will display
3. Application display all objects

4. User each touch of the screen character will jump
5. Each successful jump add 1 to score
6. For every 5 points box speed increase

Collision Use Case Diagram

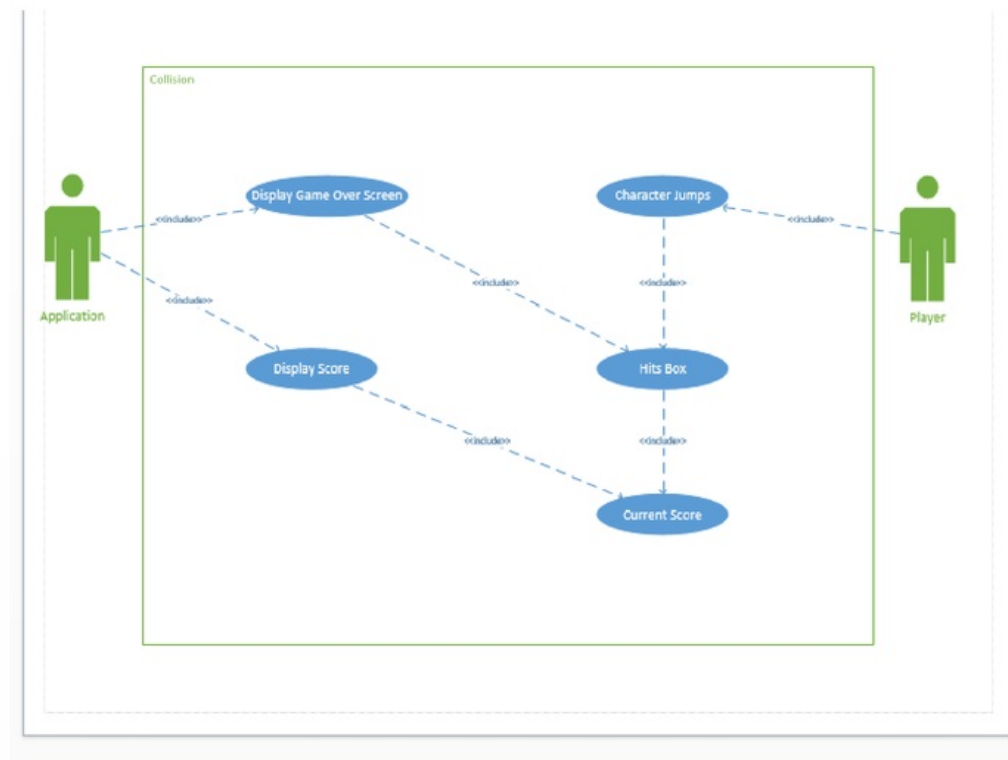


Figure 13 Collision Use Case Diagram

Use Case Documentation

Goal: When collision is made display their overall score and display game over screen

Description: The user controls the character by jumping over boxes. When the character hits the box then this will be game over.

Frequency of Execution: This should happen every time the character hits a box.

Scalability: each instance of the application one user only

Criticality: If the character looks like it has collided with the box but game still continues then this will mean the collision set is not correctly positioned and this will need to be amended.

Primary path: These steps are carried out by the users each time the character has collided.

1. Character hits box

2. The game over screen will display
3. Display their Score

State Chart Diagram-Collision

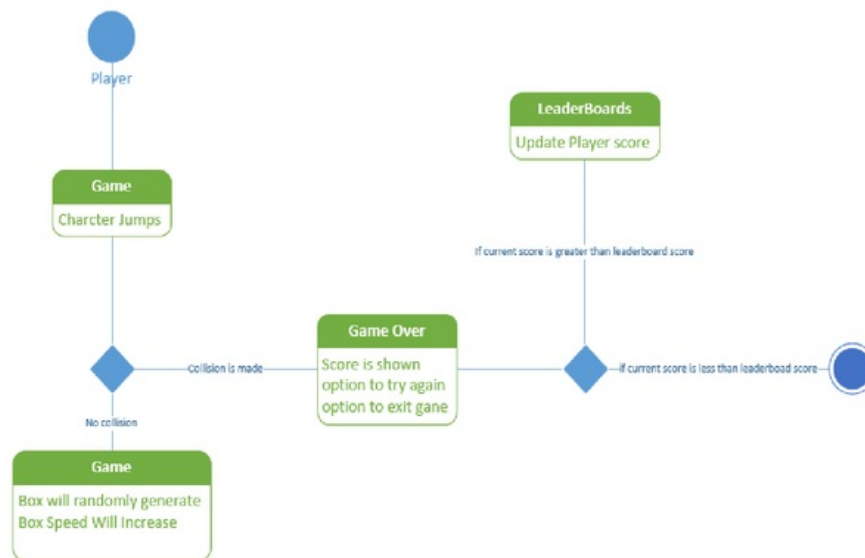


Figure 14 Collision-State Chart Diagram

The state chart diagram represents what happens if a collision is made in a much more detailed manner. Here it shows what happens when no collision has occurred and also a collision. It then shows how the system will handle when a collision has been made. Here you see that no collision will allow the game to continue and more boxes will generate and also that their speed will increase. A collision however will display the game over screen which will allow the user options to try again, exit the game and also display their score. This is not the final part of the functionality behind the game from here the system will check to see their score is better than their previous score and if so then this will update their score in the leader boards.

Database - Leader Board

A database will be added so that scores made from the game can be saved and updated every time a new game is played. This is the key part of the game that makes it have an

competitive edge. The purpose of adding a database is so that users can see their high score and also view who the current best out of their friends.

Entity Relationship Diagram

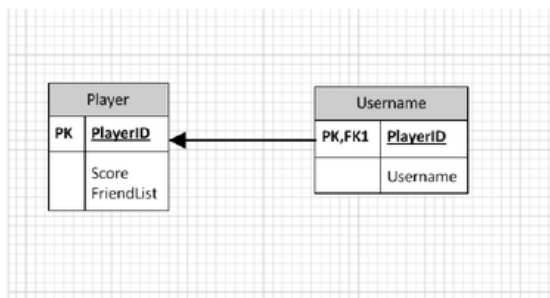


Figure 15 Entity Relationship Diagram

Here an entity relationship diagram showing the relationship between the two tables .The purpose of creating this is so that when programming to display the player and friends score you will need to grab both the user score data and also the friends this relationship will allow you to do this. It works by when a friend is added it will look for that specific players id when identified it will grab the username from the username table and also the score from the player table the friend list is used to show what users are friends with who for example:-

Username Table	
PlayerID	Username
1	Craig
2	Andrew
3	Aaron

Player Table		
PlayerID	Score	FriendList
1	18	2,3
2	15	1,3
3	16	1,2

In the player table it will show that Andrew is friends with Craig and Aaron so on his game the leader board's screen will show the score for himself and the two he is friends with.

Update Score Use Case Diagram

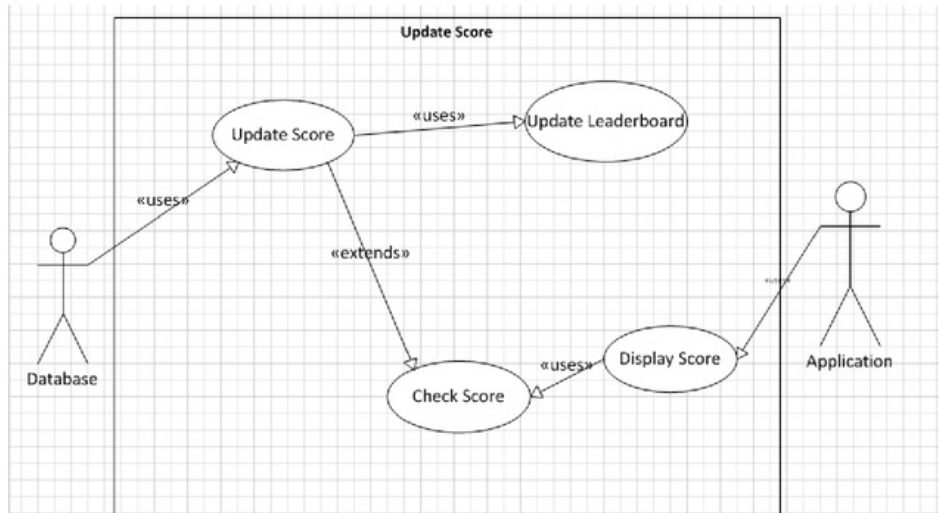


Figure 16 Update Score Use Case Diagram

Use Case Documentation

Goal: The user gets a score, the application check the score to see if its higher than its previous if so then it will update the database. The database will then update the leader board. If score is less than or equal to previous score then no changes will be made.

Description: Once the player gets a score then automatically the application and database will communicate to make any necessary changes.

Frequency of Execution: This should happen every time the player gets a higher score than previous.

Scalability: each instance of the application one user only

Criticality: score may not update to leader boards if score is higher than previous.

Primary path: These steps are carried out by the users each time the character has collided.

1. User gets a high score
2. Application check score to previous
3. If less than or equal to previous then no changes is made
4. If score is higher than previous then the database will update
5. The leader board will be updated.

Implementation

Problems and Successes

Throughout the development process of the creation of the application, is that I had to use the free trial version of Xamarin. By using this it only allowed the application to run for 24 hours only. You had to continuously re build the solution to be able to run the application. Although you could have used the free version however this had a complication that would have not made this project a success. This is because the free version only allowed you to use so much data therefore if this was chosen then this would have been a gamble as you would not know what size the application would be. If the size would be bigger than the limit they gave then the solution would not be built and therefore the free trial will need to be downloaded, So it would be a waste of time .Although the free trial is annoying as you have to re build your solution it does however allow you to create the application without any other applications.

Another complication was that importing CocoSharp was a problem, on the Xamarin website it showed an example of how to install its library and its components. However this did not work and had to look into the problem and luckily found a forum that other people had the same problem. This then found a solution to the problem. This would have been a massive set back if this problem couldn't be fixed.

When coding the application the biggest issue was that building the solution to see the progress was that you had to run it in a simulator which Xamarin provided. These were not very good as they took too long to load can was very buggy which caused the pc to sometimes crash. So another solution was found to fix this, luckily there was another emulator called bluestacks which allowed android applications to be run. This saved a lot of time and effort as this application needed a lot of trial and errors to fix any functions that needed to be included.

Considering all the above issues have caused some concern however they were still able to overcome and create a successful working application. To make this successful Xamarin did provide some source code for a previous game that was created called gonebananas using cocosharp. This helped to understand some of the code. By using this and having knowledge of C# the project was successful.

Code Snippets

Code snippets has been taken to show the main functionality within the application.

Gravity

This is the method which was used to create the gravity pull down on the character.

Comments has been used to explain how each code works.

```
// If Player is not on the floor, then increase their Y velocity by negative gravity value
    if (isOnFloor == false) {
        playerYVelocity += frameTimeInSeconds * -gravity;

        player.PositionY += playerYVelocity * frameTimeInSeconds;
    }
    // Checks if the player has landed on the floor, it does this by checking if the player is colliding with the floor object
    bool doesPlayercollide = player.BoundingBoxTransformedToParent.IntersectsRect(
        floor.BoundingBoxTransformedToParent);

    //If the playeryvelocity is < 0 they are not moving down
    bool isMovingDownward = playerYVelocity < 0;

    // Modify gravity depending on if the player is going down, by doing this gives the player the ability to jump smoothly.
    if (isMovingDownward) {
        gravity = 500f;
    } else {
        gravity = 200f;
    }
    //If the player collides with the floor and aren't on the upward part of a jump then they are on the floor
    if (doesPlayercollide && isMovingDownward)
    {
        isOnFloor = true;
    }
```

Player Collide With Box

This method shows how it determine the collision is made. A simple check was created to see whether the player has collided with the box. This is done by determining the scaling of both objects if the position is less than 52 then we determine it's a hit and they be taken to the game over screen

```

        // Check if the player collides with the box sprite.
        bool doesPlayerHitBox = player.BoundingBoxTransformedToParent.IntersectsRect(
            box.BoundingBoxTransformedToParent);

        if (doesPlayerHitBox)
        {
            // Taking into account scaling and of sprite (5
            // 2 seems to work with the current sprite sizes)
            if(CCVector2.Distance(player.Position, box.
            Position) < 52)
            {
                //If collided then send to game over screen
                CCScene scene = GameOverLayer.GameOverScreen (G
                ameView, playerScore);
                GameView.RunWithScene (scene);
            }
        }
    }
}

```

Add Score

This is how a score is added , it checks if a box position is less than 0 then this will mean the box is off screen and therefor the character has jumped over a box and a score should be added.

```

        //Simply check the X Position of the box
        if (box.PositionX < 0)
        {
            //If it's less than 0 then it has made it past the pla
            //yer and should be off the screen
            //Now it can be deleted and score increased.
            box.RemoveFromParent();
            DeletedBoxes.AddLast (box);
            playerScore++;
            scoreLabel.Text = "Score: " + playerScore;
            scorePassed++;
        }
    }
}

```

Increase Difficulty

This determine how the difficulty increases for every 5 boxes that has passed the speed will get 5 seconds quicker and the counter will reset .For another 5 boxes that will pass it will go another 5 seconds quicker again.

```

        //Increase Difficulty every 5 boxes
        if (scorePassed >= 5)
        {
            // Reset the counter
            scorePassed = 0;
            // Increase Box Speed
            boxSpeed = boxSpeed - 5000;

            if (maxSpawnTime > 2)
            {

```

```

        // Increase Spawn rate
        maxSpawnTime = maxSpawnTime - 1;
    }
}

```

Spawn Boxes

This method shows how each box is spawned into the game. If it's time to spawn a box then it will spawn them in the position as set below. If a box is not ready then it will calculate the next time for it to be spawned.

```

void SpawnBoxes(float frameTimeInSeconds)
{
    // If there is not a box waiting to spawn then calculate the next time to spawn a box.
    if (boxWaitingToSpawn == false) {
        Random rnd = new Random ();
        DateTime now = DateTime.Now;
        TimeSpan nextSpawnTime = new TimeSpan (0, 0, 0, rnd.Next (
minSpawntime, maxSpawnTime), 0);
        spawnTime = now.Add (nextSpawnTime);
        boxWaitingToSpawn = true;
    }

    //If it's time to spawn a box then do so at the box starting position.
    if (DateTime.Now > spawnTime)
    {
        var newBox = new CCSprite ("box");
        newBox.PositionX = 950;
        newBox.PositionY = 200;
        newBox.Scale = 2;
        AddChild (newBox);

        boxes.AddLast (newBox);
        boxWaitingToSpawn = false;
    }
}

```

Animation

To create the animation you simply use the current player position and replace images over each other, because there two types of animation used we need to determine how each will be used. Firstly you needed to get the player position, see if they on floor or not and work out when to swap images by creating frames to get a working animation. Comments are used to show how this works.

```

void DoAnimation(float frameTimeInSeconds)
{
    // Get the current Player Position, this will be used to create a new sprite at the same location
}

```

```

float playPosX = player.PositionX;
float playPosY = player.PositionY;

// If The Player is on the floor we should do the running anim
ation
if (isOnFloor) {

    if (shouldDoAnim == false) {
        // We need to calculate the next time we should swap f
        rame, (100 ms from now)
        DateTime now = DateTime.Now;
        TimeSpan nextAnim = new TimeSpan (0, 0, 0, 0, 100);
        animTime = now.Add (nextAnim);
        shouldDoAnim = true;
    }

    //If it's time to swap frame then we will swap to the next
    frame in the list.
    if (DateTime.Now > animTime) {

        if (animType == 0) {

            RemoveChild (player);
            animType = 1;
            player = new CCSprite ("frame1");
            player.PositionX = playPosX;
            player.PositionY = playPosY;
            player.Scale = 2;
            AddChild (player);

        } else if (animType == 1) {
            RemoveChild (player);
            animType = 2;
            player = new CCSprite ("frame2");
            player.PositionX = playPosX;
            player.PositionY = playPosY;
            player.Scale = 2;
            AddChild (player);

        } else if (animType == 2) {
            RemoveChild (player);
            animType = 3;
            player = new CCSprite ("frame3");
            player.PositionX = playPosX;
            player.PositionY = playPosY;
            player.Scale = 2;
            AddChild (player);

        } else if (animType == 3) {
            RemoveChild (player);
            animType = 0; //This is the last frame of the anim
            ation, we set it back to the first frame this time around.
            player = new CCSprite ("frame4");
            player.PositionX = playPosX;
            player.PositionY = playPosY;
            player.Scale = 2;
            AddChild (player);

        }
    }
}

```

```

        shouldDoAnim = false;
    }
} else if (isOnFloor == false) // We are not on the floor so we must be somewhere in a jump.
{
    //Check if we are moving up or down
    bool isMovingDownward = playerYVelocity < 0;

    // If we are going down then show the jumpdown animation frame
    if (isMovingDownward) {
        RemoveChild (player);
        player = new CCSprite ("jumpdown");
        player.PositionX = playPosx;
        player.PositionY = playPosy;
        player.Scale = 2;
        AddChild (player);
        //Otherwise show the jump up animation frame.
    } else if(isMovingDownward == false)
    {
        RemoveChild (player);
        player = new CCSprite ("jumpup");
        player.PositionX = playPosx;
        player.PositionY = playPosy;
        player.Scale = 2;
        AddChild (player);
    }
}
}
}

```

Jumping

This determine when and how the player can jump. This will specify the height the character can jump.

```

void OnTouchesEnded (List<CCTouch> touches, CCEvent touchEvent)
{
    //If the player is on the floor, we can jump
    if (isOnFloor) {
        isOnFloor = false;
        playerYVelocity = playerYVelocity + 250;
    }
}

```

Testing

The types of testing used to evaluate the game were system testing, usability testing and acceptance testing. These were chosen from reviewing the different types of test researched and are performed as they should. The results are shown below:-

System Testing

Task ID	Task Name	Days expected	Actual Days	Planned Results	Actual Results	Problems	Comments
1	Character Jumps	1 days	1 days	Function added so that the player jumps to a specific height and also once it reaches the the max height, The gravity speed will need to look like he landing at a considerable pace and not too quickly	As Expected	N/A	N/A
2	Jumping Over Box Detection	1 days	1 days	Function created so that the game can detect when the character has jumped over a box without collision.	As Expected	N/A	N/A
3	Score Increase	0.5 days	0.5 days	The score will increase if the character jumps successfully over each box.	As Expected	N/A	N/A
4	Box Speed Increase	0.5 days	0.5 days	The box speed will increase for every 5 points	As Expected	N/A	N/A
5	Displaying In game objects	1.5 days	2 days	When the game starts the application must display all objects necessary	As Expected	N/A	N/A
6	Collision of the box	1 days	2 days	The character hits the box and cause game over	As Expected	N/A	N/A
7	Display their score	0.5 days	0.5 days	Score will display on game over screen	As Expected	N/A	N/A
8	Display Game Over Screen	0.5 days	0.5 days	Game over screen will display	As Expected	N/A	N/A
9	Animation	1 days	2 days	Character looks like he is running and jumping	As expected	N/A	N/A
10	Background Moving	0.5 days	1 days	Make background continuously moving to make game flow look professional	Failed	Slight glitch which makes image look like its lagging	Made amendments to make the image overlap to stop lag. Still error occurs

Table 1 System Testing Results

Evaluation

A system test was performed to show the main functionality within the game, this explains how long each function would take and how long it actually took. From results gathered main functionality that had problems was the displaying in game objects, collision of the box, Animation and Background Moving. The displaying in game objects was a trial and error as when coding it the box spawn should start outside the screen then as the player jumps over 1 then another should display. The issue that occurred was that the spawn were sometimes spawning out of place which made the game impossible. So to fix this it was just changing values until the box was correctly positioned from start to when the box disappears off screen.

The collision was another trial and error as to determine the collision the way the image as scaled in game, there was initially an issue where the box would count as being collided with before the actual box reaches the player to overcome this we simply added a check on the boxes to see the position was close enough to the player to count as a collision.

The animation was a bit of a problem as the actual images created to use for the animation didn't seem to fit together when placed. The animation looked really unprofessional so to overcome this I looked at free art online that was made for animations for in game. Then this was used instead of the original art that was created.

The background moving was a tricky problem that seem to be laggy for unknown reasons, amendments was made to overlap the image so that it will look like the image is just being repeated like a wallpaper but this did not change anything and therefore it is a continuing problem but doesn't affect the gameplay what so ever.

Usability Testing

Now that the application is made it good to get an understanding to see users can use the game without any instruction. To perform these tests users will be giving a set list of tasks to perform to see if they can understand the game without any assistance. The tasks given will have to be completed in an allocated time limit so that they cannot necessary try and perform different interactions to reach the goal. It is best the task is completed first time as this will show the application has been a success. Here is the results:-

User 1

Task/s	Goal	The User Process	Findings/comments
Start Game	Load the app and start game.	Located the app without no issues, started game with a simple touch press.	No problems here user performed task as expected.
Make character jump	Once game loads, user will need to control the character by a touch press.	The game loaded without no issues, the user looked a little confused how the game worked. After the initial thought he/she touch press then realised how it worked.	Although there were no problems, however when the user looked confused how to control the character this had some concern. I think adding some text explain how to make the character jump will be added.
View their high score/game over screen	Once the game over screen has appeared the user will need to simply touch the screen to start a new game.	Cause the user got a little trigger happy when trying not to collide with the box they accidentally missed out the game over screen to view their high score and automatically started a new game.	This was a little unexpected and does need fixing. A simple solution to this is to add a button so that a new game will only load once this is clicked rather than a touch press.
Collide with box	The character will collide with box causing game over	No actions were made they waited for a box to collide with character	This was expected no need for any changes
Reach 15 points	Character will need to jump over 15 boxes in total successfully.	The user did struggle to jump over 15 boxes on first attempt.	Although the user struggled which made me think maybe I should slow the speed of the boxes. However I did allow a few extra goes so that the user can get use to the game. After a few tries they successfully reached the goal. So no changes is needed as this what makes the game enjoyable and addictive
Start A new game once game over	Start a new game by simply a touch press when game over screen is displayed	Once game over screen was shown the user simply touch the screen	This was expected no issues here so no changes will need to be made.

		and game re started with no issues	
--	--	------------------------------------	--

Table 2 Usability Testing - User 1 Findings

User 2

Task/s	Goal	The User Process	Findings/comments
Start Game	Load the app and start game.	Located the app without no issues, started game with a simple touch press.	No problems here user performed task as expected.
Make character jump	Once game loads, user will need to control the character by a touch press.	The game loaded without no issues, the user had no problem working out how the game worked	No problems here
View their high score/game over screen	Once the game over screen has appeared the user will need to simply touch the screen to start a new game.	Once the user collided with the box they stopped any interaction with the game and viewed their high score	This was as expected so no problems here
Collide with box	The character will collide with box causing game over	The user made the character jump onto the box.	Collision was made so no errors here
Reach 15 points	Character will need to jump over 15 boxes in total successfully.	User managed to gain over 15 points.	The user had no issues getting to 15 points.
Start A new game once game over	Start a new game by simply a touch press when game over screen is displayed	Once game over screen was shown the user simply touch the screen and game re started with no issues	This was expected no issues here so no changes will need to be made.

Table 3 Usability Testing - User 2 Findings

Evaluation

After looking at the usability testing I did notice some issues did arise and therefore this could be due to the fact of the age group that was selected to participate in this test. Cause the game created is for all ages I wanted to get feedback from a wide range of people. The number of people that participated in this test was 10 with an age group ranging from 14-52. Although most results from people were the same however I did notice that the older range were trigger happy whereas the younger were taking their time whilst playing the game. This meant that some changes had to be made because of the touch press in some

places would cause some actions to occur that mean the game over screen will be skipped. So some necessary amendments will be made to help stop such problems.

Acceptance Testing

This type of testing is to evaluate the product to its potential end users. This will test the product to see if the requirements has been met. To perform this test friends and family was given the game to play in their own time. This was done so that I can gain positive and any negative feedback about the game. The game was tested between 14 different users each of which give me feedback after 10-15 minutes after playing the game.

All comments was positive at first saying that the game has a huge potential if it were to be released on market. However I was more concerned on the negative feedback the main problems was the background needed fixing which was found out during the system test. The other issues is that the game over screen needed a button so that they cannot skip screens on accident. However apart from those two major errors it was a good working game as they described.

The main thing about this sort of test is that a lot of feedback was what improvements could be added to make the game stand out more and make it more enjoyable. Some suggested that to make the character shoot, and add taller objects which is impossible to jump over and by adding the shooting function only this would destroy this object and stop the player from losing. Other suggested different character unlocks, add new levels etc.

Evaluation

From the feedback my overall interpretation is in future possibly those features could be added to keep the interest in the app. But overall it's relatively promising and very positive except for a few minor error which was found in the other tests. Once these are fixed this will mean that it is a worthwhile game to be put on the market.

Evaluation

To evaluate the project was of a success, a game was created through the use of Xamarin a cross platform software and was tested to a good standard. The data collected has backed up the research in the literature review has it give a good understanding what apps was all about in great detail, what current software is out on the market to produce cross platform apps and also what testing techniques could been used.

The objectives was first set so that you can have a breakdown on how the project can be successful. Overall all the objectives of the project was met, the third objective did however include that unit testing was going to be performed when investigating different techniques but because some issues occurred whilst performing these tests this was unsuccessful. Although this test was not used it didn't affect the overall project.

The main aim of the project was too create a successful working and fully tested game application to be run cross platform. Even though a cross platform app was created and could be used on different devices. An issue with this is that when played on a device the game would not fit to screen it would shrink. On pc and tablets the game was visually ok and playable. When played on mobile devices the game was playable but wasn't very pleasing on the eye. This was because a trial version was used so that everything needed to be coded for the size of the device you was testing it on for.

Some of the answers that was collected to help create the application was left unanswered. During the research everything seem to be self-explanatory when using the software to create the app. It was well explained how to install and how easy it was to use. When installing the software and programming with the use of CocoSharp development tools. More research had to be carried out for me to progress further and help create the app successfully.

In the design phase it showed that a database was going to be added so that a leader boards section could be added and that users could see their high scores and compare them against their friends. However due to the knowledge of importing a database into Xamarin this was unsuccessful.

If the project was done again there would be a few changes if you wanted to create a game application then I would suggest using a better game engine such as unity as this allowed

you to create games using much better tools and also allow you to make it cross platform just like Xamarin. If you were to create an application then Xamarin would be of good use but I would run the tool through visual studio as this would have been suitable as I was more familiar with it.

Conclusion and Recommendations

To summarise the project each section was created to give an overview how the project will be created. The sections were literature review, research and development methodology and design and implementation.

The literature review covered areas such as the progress of apps and how they work etc. Other study showed why an app is important today and how they should be created for good user experience. This also explained what apps are being used mostly today .The other reasons for this was to see what cross platform are available and the chosen software that will be used is it going to be of use to create a game and what tools does that offer. Also it described what testing methods are out there and which are going to be used for this project.

The research methodology explained the research onion and explained the approach this project was taken. It described the advantages and disadvantages of this. The development methodology explained the prototype methodology which was used throughout the project. In this section it explained in detail what is the prototype method, how it worked and its advantages and its disadvantages.

The design and implementation section detailed the design of the project by creating a list of requirements and diagrams that described the system as a whole. The implementation explained the problems and success that was encountered throughout the creation stage, explained code of the main functionality in the game and also explained the different types of tests performed and how each was evaluated.

This project has many areas that can be improved and could be added to help make this app grow and get a lot better. As described in the evaluation that the database was left out because of the knowledge of importing it into Xamarin. This would need be looked into to see if it is even possible to include a database into Xamarin and also how to implement it. Also when evaluating Acceptance testing many encouraged to add many other features to make the game more effective and more interesting. These could be looked over and see if it worth introducing some if this project were to be picked up and progressed further. For a future student/developer to continue this project they will require enthusiasm, interest and ready to encounter and solve any difficulties they encounter. Also a large time scale will be needed so that they can learn new things and also in case they encounter problems that needs fixing.

Reflection

Throughout the whole project it's been very interesting on what has been found out. The main interest of this project is that creating a cross platform app could be created without the knowledge of using java to create android apps and objective c to create IOS apps. Also that by using one programming language an application could have been created for both devices.

In the course of learning it was important that a software chosen used the C# syntax as this was a key part when programming the application. As this particular language was learnt throughout the studies at Swansea Trinity St David University. If any software available did not use the C# language then the project would have been unsuccessful.

The process of implementing the app was interesting as it taught and allowed to gain new skills as creating a game was new and this was self-taught. The skills gained was to how to use Xamarin, how a game engine become of use and how the syntax was different to perform functions that was never used before such as animation.

Xamarin would be a good choice for any plans on creating non related game applications for the future as there is other software for game development but to create applications this would be ideal for C# developers.

References

1. "History mobile development"[Online]Available:<http://manifesto.co.uk/history-mobile-application-development/> [Accessed 09/11/2015]
2. "History of mobile application development"[Online]Available:<http://www.appschopper.com/blog/history-mystery-of-mobile-application-development-revealed-here/> [Accessed 17/11/2015]
3. "Apps:Native"[Online]Available:<http://www.nngroup.com/articles/mobile-native-apps/> [Accessed 09/11/2015]
4. "Implementing a Mobile App"<http://www.craigslist.com/article/20130718/BLOG105/130719803/implementing-a-mobile-app-strategy> [Accessed 17/11/2015]
5. "Understanding Mobile Apps"[Online]Available:<https://www.onguardonline.gov/articles/0018-understanding-mobile-apps> [Accessed 25/11/2015]
6. "App Store Fees"[Online]Available:<http://www.techrepublic.com/blog/software-engineer/app-store-fees-percentages-and-payouts-what-developers-need-to-know/>[Accessed 25/11/2015]
7. "Ios App or Android App"[Online]Available:<https://crew.co/how-to-build-an-online-business/build-ios-app-or-android-app/> [Accessed 15/12/2015]
8. "Developing On Ios Or Android"[Online]Available:<http://developers.magmic.com/developing-on-ios-vs-android-there-is-one-clear-winner/> [Accessed 15/12/2015]
9. "Things to consider when producing an app"[Online]Available:<http://info.localytics.com/blog/9-things-to-consider-before-during-and-after-launching-your-app> [Accessed 19/12/2015]
10. "Importance of Mobile Apps "[Online]Available:<http://www.ceraintmobileapps.com/blog/mobile-apps-importance> [Accessed 10/11/2015]
11. "Elements mobile user experience"[Online]Available:<http://www.smashingmagazine.com/2012/07/elements-mobile-user-experience/> [Accessed 18/11/2015]

12. "Most popular type of apps"[Online Available]
<http://www.statista.com/statistics/270291/popular-categories-in-the-app-store/>
[Accessed 15/04/2016]
13. "Top Ten Best Cross Platform"[Online]Available:<http://appindex.com/blog/ten-best-cross-platform-development-mobile-enterprises/> [Accessed 09/11/2015]
14. "PhoneGap"[Online]Available:<https://build.phonegap.com/> [Accessed 18/11/2015]
15. "Pros And Cons Top 5 Platform tools"[Online]Available:<http://www.developereconomics.com/pros-cons-top-5-cross-platform-tools/> [Accessed 09/11/2015]
16. "Titanium appcelerator how it works"[Online]Available:<http://forumone.com/blog/what-titanium-appcelerator-really-and-how-it-works/> [Accessed 18/11/2015]
17. "History mobile developmnt"[Online]Available:<http://enricoangelini.com/2012/5-pros-and-cons-of-appcelerators-titanium/> [Accessed 09/11/2015]
18. "Xamarin Plugins"[Online]Available:<https://xamarin.com/mobileapi> [Accessed 18/11/2015]
19. "Xamarin Platforms"[Online]Available:<https://xamarin.com/platform> [Accessed 18/11/2015]
20. "Pros And Cons of Xamarin"[Online]Available:<https://www.quora.com/What-are-the-pros-and-cons-of-using-Xamarin-for-Android-development> [Accessed 10/11/2015]
21. "Xamarin pros and cons cross platform development"[Online]Available:<http://skelia.com/articles/xamarin-pros-and-cons-for-cross-platform-development/> [Accessed 10/11/2015]
22. "Building Cross platform on Xamarin"[Online]Available:https://developer.xamarin.com/guides/cross-platform/application_fundamentals/building_cross_platform_applications/sharing_code_options/ [Accessed 19/12/2015]
23. "Game Development With Xamrin"[Online Available]
https://developer.xamarin.com/guides/cross-platform/game_development/
[Accessed 15-04-2016]

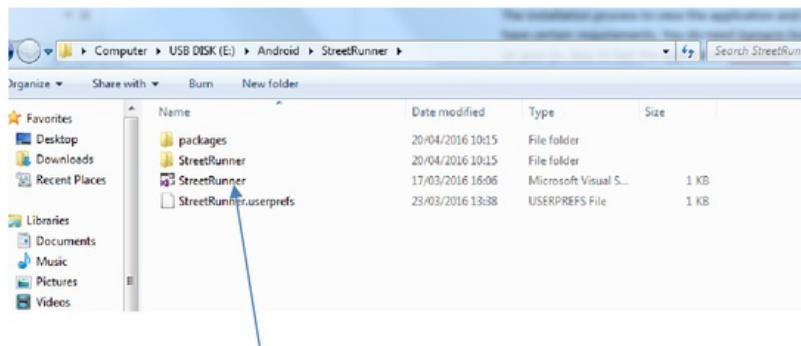
24. "Overview of game testing techniques" Available:
http://www.idt.mdh.se/kurser/ct3340/ht11/MINICONFERENCE/FinalPapers/ircse1_1_submission_15.pdf [Accessed 02/02/2016]
25. "Types of testing" [Online Available] http://www.tutorialspoint.com/software_testing/software_testing_types.htm [Accessed 08-02-2016]
26. "Usability Testing" [Online Available] <http://www.experienceux.co.uk/faqs/what-is-usability-testing/> [Accessed 08-02-2016]
27. "System Testing" [Online Available] <http://www.softwaretestingmentor.com/test-levels/system-testing/> [Accessed 08-02-2016]
28. "Seven Principles of software testing" [Online-Journal] Bertrand Meyer, ETH Zurich and Eiffel software <http://se.ethz.ch/~meyer/publications/testing/principles.pdf> [Accessed 08-03-2016]
29. "Seven principles of software testing" [Online Available]
<http://www.testingexcellence.com/seven-principles-of-software-testing/> [Accessed 08-03-2016]
30. "Software Prototype" [Online] Available:
http://www.tutorialspoint.com/sdlc/sdlc_software_prototyping.htm [Accessed 03/02/2016]
31. "C#"
[Online] Available: <http://www.developer.com/net/asp/article.php/922211/What-is-C.htm> [Accessed 06/01/2016]

Appendix

The installation process to view the application and its source code is fairly simple but does have certain requirements. You do need Xamarin business edition or higher to be installed on your pc. Also to test the application, CocoaSharp is also required.

Following the below steps will ensure that the application will work on your pc:

1. Copy the file somewhere on your pc
2. Click on the android file
3. Open the StreetRunner file
4. Then run the StreetRunner application



This is what is needed to be open to view the source code and test it.

