##### GAME DEVELOPMENT IN UNITY 3D

Using C#, JavaScript and Boo

7138

##### MINOR PROJECT REPORT



###### ***Submitted by***

##### Debashis Pradhan 201210268

##### Ankur Upadhyay 201216265

###### ***Under the guidance of***

###### Mr. Debananda Kanhar

IN

COMPUTER SCIENCE AND ENGINEERING

NATIONAL INSTITUTE OF SCIENCE AND TECHNOLOGY

PALUR HILLS, BERHAMPUR-761008

December **2015**

**ACKNOWLEDGMENT**

ABSTRACT

Unity is a flexible and powerful development platform for creating multiplatform 3D and 2D games and interactive experiences. It's a complete ecosystem for anyone who aims to build a business on creating high-end content and connecting to their most loyal and enthusiastic players and customers. Unity confidently targets the hottest platforms including newcomers such as WebGL and Oculus Rift and efficiently optimizes performance with cross-platform tools.

The only thing more interesting than playing games is designing games. The Project concerns with creation of games­ from concept to publishing. One of the games here deals with a rolling ball which collects coins on its way, while running from enemy cubes. Another game is in first person view, in a cubic world. The player sees a cross-hair and a number of colored boxes to shoot.

**TABLE OF CONTENTS**

**Title Page**

**Bonafide Certificate**

**Acknowledgement**

**Abstract**

**Table of Contents**

**List of Figures**

**Project Overview**

1.1. Game Development in Unity

1.2. Game Overview

1.3. Audience

1.4. Scope

**Ideation**

2.1. Player Actions

2.2. Game World

2.3. Game Look and Feel

2.4. Game Implementation

2.5. User Experience

**Asset Creation and Acquiring**

**Game Design Process**

4.1. Designing Game play, Levels and Balance

4.2. Prototyping and Play testing

4.3. Game GUI

4.4. Game Documentation

**Level Abstract**

1.1.Camera Setup

1.2. Player Setup

1.3. Physics Engine

1.4. Particles and SFX

1.5. Animation

**Unity Programming**

2.1. Using MonoDevelop

2.2. Scripting

2.2.1. Health and Pick-Ups

2.2.2. Spawners

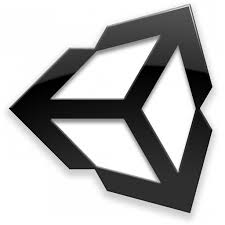
2.2.3. Chaser Script

2.3. Game Manager

**Build and Deploy**

**References**

**LIST OF FIGURES**

1. **GAME OVERVIEW**
   1. **GAME DEVELOPMENT IN UNITY** 

Unity is a [cross-platform](https://en.wikipedia.org/wiki/Cross-platform) [game engine](https://en.wikipedia.org/wiki/Game_engine) developed by [Unity Technologies](https://en.wikipedia.org/wiki/Unity_Technologies) and used to develop [video games](https://en.wikipedia.org/wiki/Video_game) for [PC](https://en.wikipedia.org/wiki/Personal_computer), [consoles](https://en.wikipedia.org/wiki/Video_game_console),[mobile devices](https://en.wikipedia.org/wiki/Mobile_device) and [websites](https://en.wikipedia.org/wiki/Website). First announced only for [OS X](https://en.wikipedia.org/wiki/OS_X), at Apple’s Worldwide Developers Conference in 2005, it has since been extended to target more than fifteen platforms. It is now the default [software development kit](https://en.wikipedia.org/wiki/Software_development_kit) (SDK) for the [Wii U](https://en.wikipedia.org/wiki/Wii_U).

Unity 3D is one of the most popular and powerful game development tools.The factors responsible for emerging unity as a unique tools is the union of

(1) a game engine, that allows game created to run (hence to be played) in different environments,

(2) an application where the “visible pieces” of a game can be put together (the IDE) with a graphical preview and using a controlled “play it” function, and

(3) a code editor one of the features of Unity is that it is multi-platform, that is more or less the same game can run similarly on an iPad or on Windows, say.

Videogames are complex structures. Unity provides a complete workflow and a lot of help along the line, and a large set of features come in Unity’s free version.

Working with Unity can greatly facilitate making people with different skills work together and at different times, without interfering with each other’s work.

**Why Unity? And Other GameEngines.**

The POWER OF SCRIPTING

There are a great many game engines in existence that have a visual editor. Many of these engines lack the power required to build significant or complex games due to their canned approach to behaviours. With Unity, object behaviours aren’t limited to built-in modules that come packaged with the engine. Instead, Unity allows for powerful behaviours written in any of three robust languages: JavaScript, C#, and Boo. Furthermore, all three languages can be used at the same time within a project to allow people of different technology backgrounds to contribute to a project at the same time. The fact that the languages are used as scripts allows for fast compilation times, quick iterations, and flexibility of design. This common language approach to game development ensures that you can begin making games quickly using knowledge you already have (or is easily attained).

ONE SOURCE TO RULE THEM ALL

Probably the most impressive feature of the Unity engine is the ability to build your projects for multiple platforms with incredible ease.With just the simple selection of a drop down menu, Unity can build for Windows PC, Linux (new with version 4), iOS (with plugin), Mac, Android (with plugin), Web Browser, Flash (with plugin), PS3, Xbox, and Wii U.

Awards and Accolades

Develop Awards 2014 - Winner - Best Engine

Unity Technologies Named Winner of Best in Entertainment and Gaming in ‘The San Francisco Business Times’ Tech and Innovation Awards

Stevie Awards: Tech Innovator of the Year Finalist (David Helgason, CEO)

Stevie Awards: Innovative Tech Company of the Year (Company) - Finalist

2014 Gamesindustry.biz Innovation Awards - Finalist - Innovation in Technology

**1.4. Audience**

Popularity of videos games is increasing exponentially these days .Gaming industry is attracting people of all age group .Some play it for fun and some are seriously involved in it, they are known as professional gamers. As video games become more social with multiplayer and online capability, gamers find themselves in growing social networks. Gaming can both be entertainment as well as competition, as a new trend known as electronic sports is becoming more widely accepted. Today, the impact of computer and video games can be seen in social media, television, films.

**1.5. Scope**

Everyone knows that games are designed to help develop practical skills, serve as a form of exercise, or otherwise perform an educational, simulation, or psychological role. We can say that, in this fast growing world, people have very less time for refreshment, and these games will certainly will be the one of the best option. Plus, games add to the knowledge about a domain. Students have often admitted to have learnt history more from playing World War II games than from any book. A game developer who is working on a game on bees, will find that after completing his game, he has much more knowledge on bees than he could ever imagine. There are many things which are to be learnt and after applying all those things a game can be converted in to a lot more exciting, better looking, and more playable game.

**2.IDEATION**

**2.1. Player Action**

Player can use the keys such as W,A,S,D for movements even the arrow keys can also be used for the same. Different actions can be crouch, sprint, walk, run, jump ,dive etc. Scripts can be used to control speed of the actions and it must be suitable enough for the user to control it efficiently.

**2.2. Game World**

A **ga**me world is an artificial universe, an imaginary place in which the events of the **game** occur .In unity in the scene view, we need to use different assets and objects to create a game world.

The most important phase of designing game world is to have a clear idea about the game and the characters to be designed.

****

**2.3. Game Look and Feel**

Three important things which contribute to look and feel of game are

Sound

Vision and

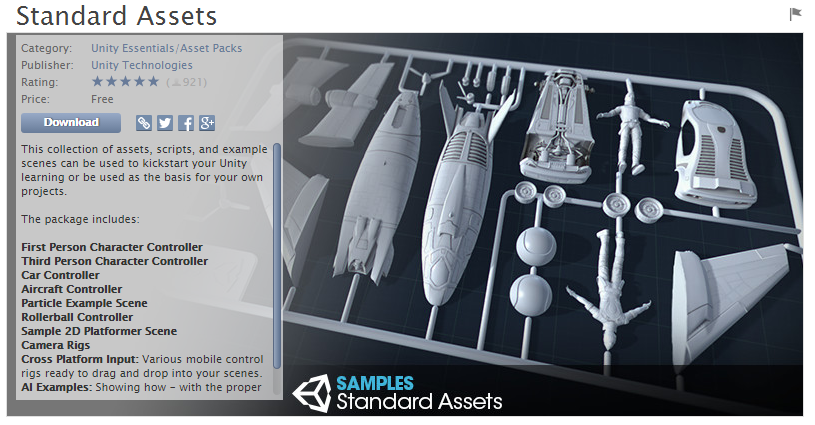
Touch

Sound effects can also be used to foreshadow action as well as reward player actions. Concept art is used to set the general style of the game’s appearance. Touch refers to the game developers’ handling of the game .It is important to convey the physical feeling of the game environment from the player’s perspective.

**2.4. Game Implementation**

Game is implemented using a player object enemy and some targets .Generally physics is involved in implementing different events. Different events like collision, moving, jumping involves material of the objects as well as the texture .Arrow keys can be used to control the player or game object .Several scripts are attached for different events to occur keeping in mind the health of player and enemy object .Every object is of different color so that a player can identify them.

1. **ASSET CREATION AND ACQUIRING**



Unity provides several standard assets to start a new Unity Project. Unity also provides an Asset Store.

Following are several asset creation tools categorized by asset type.

Bitmap Graphics Vector Graphics

|  |
| --- |
| Adobe Photoshop |
| Corel Painter |
| GIMP (free) |
| Pixlr (free) |

|  |
| --- |
| Adobe Illustrator |
| Adobe Flash |
| Corel Draw |

3D Graphics Digital Sound Editors

|  |
| --- |
| Autodesk Maya |
| Autodesk 3dsmax |
| Blender (free) |

|  |
| --- |
| Adobe Audition |
| Avid Pro Tools |
| Audacity (free) |

Music Creation

|  |
| --- |
| Avid Pro Tools |
| Apple GarageBand (free) |
| Sony Acid |
| Ableton Live |
| Mixxx (free) |

Several asset acquisition websites categorized by asset type

2D Bitmap and Vector Graphics

|  |
| --- |
| Unity Asset Store |
| opengameart.org |
| <www.pexels.com> |
| <www.shutterstock.com> |
| <www.istockphoto.com> |

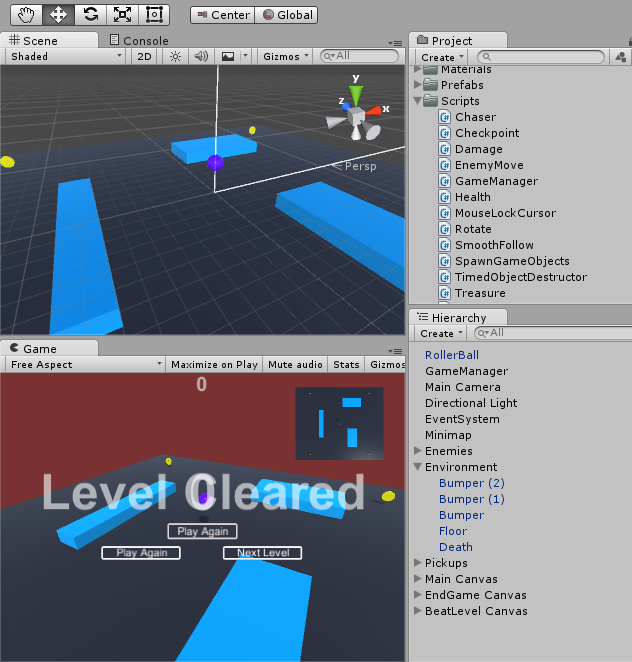
3D Graphics Digital Sound and Music

|  |
| --- |
| Unity Asset Store |
| <www.turbosquid.com> |
| shop.bitgem3d.com |
| <www.cgtrader.com/free-3d-models> |
| <www.creativecrash.com> |
| sketchfab.com |
| opengameart.org |

|  |
| --- |
| Unity Asset Store |
| <www.freesound.org> |
| <www.newgrounds.com> |
| incompetech.com |
| <www.sounddogs.com> |

**4. GAME DESIGN PROCESS**

**4.1. Designing game play, Levels and Balance**



Designing the First level of a game leads to the creation of prefabs. These prefabs can be used to generate more levels. The first level in the Roller ball game consists of a floor, death floor and bumpers. The second level also consists of the same prefabs. But the location and size of floor and bumpers has been changed.

**4.2. Prototyping and Play testing**

The Game development process consists of a Design – Play – Experience (DPE) Process. Every element of the game is first thought of, then implemented in a prototype, tested and back to design process.

**4.3. Game GUI**

As gamers and game developers we know that immersion is everything. When you're immersed you lose track of time and become involved in what the game is presenting. A major factor in what makes or breaks immersion is how easy it is for your player to convert an idea into an in-game action -- that is, how fluid your game's User Experience (UX) is and how well-designed its User Interface (UI) is. A game hurts itself by providing too little information or too much, requiring too many inputs, confusing the player with unhelpful prompts or making it hard for a new player to interact. Poor UI design can even break your game completely.

UI, or User Interface, refers to the methods (keyboard control, mouse control) and interfaces (inventory screen, map screen) through which a user interacts with your game. UX, or User Experience, refers to how intuitive and enjoyable those interactions are. To look at it another way: the UI of a car is its steering wheel, its pedals, the dials and controls on the dashboard; the UX of the car comes from intangibles like the brake pedal being responsive, the engine smoothly accelerating when you step on the gas, the gear stick having just the right amount of resistance - those things that make the car enjoyable to drive.

UI design is a largely logical process, one that's perfectly approachable to a developer.

In fact, we believe that the programmer (or creative director, or whoever in your team has the most direct role in shaping the playing experience of your game) should be responsible for making the UI, as they are the ones who know the game inside and out, and who know what information is important and what is incidental.

It’s better to focus on the functional aspects of the UI: How big is it, does it (or should it) scroll, what information is displayed and where, and how the player navigates through it. Mock it up with two or three colours, and never rely on color changes alone to convey information. At the very least, run it through ColorOracle or Coblis to ensure that color-blind people aren't missing out.

**4.4. Game Documentation**

Why Document? First of all, it helps flush out the vision. From the time of having the first idea to the time of documenting it, the amount of detail will increase. It is also a way to coordinate with the team members. Design documentation is therefore, very important. It helps keep track of the design process and define the scope of the project.

**5. LEVEL ABSTRACT**

**4.1.** Camera Setup

Camera can be controlled as per the user requirement whether he wants to follow the player or respond to mouse action. To make camera follow the player in First Person game generally we make camera as the child of the gameobject or otherwise we can add followscript from the available standard assets, in Rollerball game followscript is used .Similarly in order to respond to mouse option we can add script such as MouseLooker , in shootergame mouselooker script is used .Apart from that we can also apply rendering and image effects including skybox, contrast and bloom effects .We can use image effects on camera by using Effects package from Standard assets which will enhance the visual .Using Audio Listener feature of main camera we can add music to our game.

**4.2.** Player Setup

Most important thing in player setup is creating gameobject because it is through gameobject , player or user can interact with game world , we can choose any 3D object as our gameobject . Then we need to go to character controller component through which we can control, modify and change player actions .For implementing these actions script is to be attached to the character controller .In our project, RollerBall is used as the game object which rolls across the map. Scripting section contains the necessary classes and functions to control movements of the RollerBall as per the user requirement .We can include movements like rotate ,translation etc. along global axis .Arrow and W,A,S,D keys are used to control movement.

### 4.3. Physics Engine

Physics engine from the game point of view refers to adding different components to the character of the game .It varies from game to game .In our game, object which is Rollerball (user),camera is a child object of rollerball which follows the rollerball throughout the game. Rigid texture is added to Game object which signifies that other parameters can also be added to the object such as gravity and force which provides real time experience to the user.

Game can be made more exciting if enemy element keep following character of the game ,these features are available in unity. Any collision with the enemy element lead to game termination.

### 4.5. Animation

Unity’s Animation features include Retargetable animations, Full control of animation weights at runtime, Event calling from within the animation playback, Sophisticated State Machine hierarchies and transitions, Blend shapes for facial animations, and more.

Read this section to find out how to import and work with imported animation and how to animate objects, colours, and any other parameters within Unity itself.

The default animation that will be played when Play Automatically is enabled.Curves and events give us a fine level of detail and control during the execution of an animation.

**4.1.** Camera Setup

Camera can be controlled as per the user requirement whether he wants to follow the player or respond to mouse action. To make camera follow the player in First Person game generally we make camera as the child of the gameobject or otherwise we can add followscript from the available standard assets, in Rollerball game followscript is used .Similarly in order to respond to mouse option we can add script such as MouseLooker , in shootergame mouselooker script is used .Apart from that we can also apply rendering and image effects including skybox, contrast and bloom effects .We can use image effects on camera by using Effects package from Standard assets which will enhance the visual .Using Audio Listener feature of main camera we can add music to our game.

**4.2.** Player Setup

Most important thing in player setup is creating gameObject because it is through gameObject , player or user can interact with game world , we can choose any 3D object as our gameObject . Then we need to go to character controller component through which we can control, modify and change player actions .For implementing these actions script is to be attached to the character controller .In our project, RollerBall is used as the game object which rolls across the map. Scripting section contains the necessary classes and functions to control movements of the RollerBall as per the user requirement .We can include movements like rotate ,translation etc. along global axis .Arrow and W,A,S,D keys are used to control movement.

### 4.3. Physics Engine

Physics engine from the game point of view refers to adding different components to the character of the game .It varies from game to game .In our game, object which is Rollerball (user),camera is a child object of rollerball which follows the rollerball throughout the game. Rigid texture is added to Game object which signifies that other parameters can also be added to the object such as gravity and force which provides real time experience to the user.

Game can be made more exciting if enemy element keep following character of the game, these features are available in unity. Any collision with the enemy element lead to game termination.

### 4.5. Animation

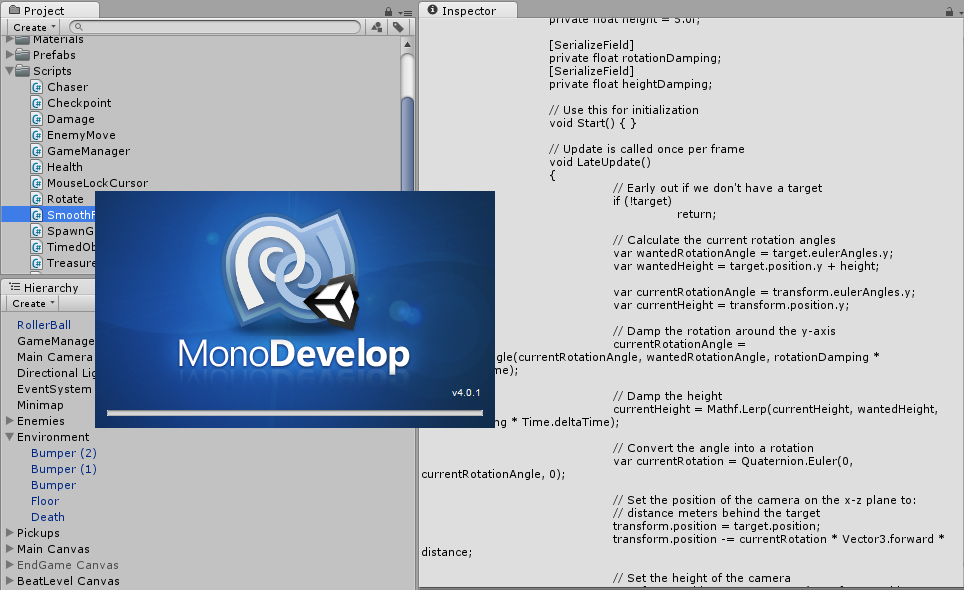
Unity’s Animation features include Retarget able animations, Full control of animation weights at runtime, Event calling from within the animation playback, Sophisticated State Machine hierarchies and transitions, Blend shapes for facial animations, and more.

Read this section to find out how to import and work with imported animation and how to animate objects, colours, and any other parameters within Unity itself.

The default animation that will be played when Play Automatically is enabled. Curves and events give us a fine level of detail and control during the execution of an animation.

**6. UNITY PROGRAMMING**

# 6.1. Using Monodevelop



# Monodevelop is a cross platform IDE for C#, F# and Much more. It is integrated with Unity 3D and can be used for scripting in C#, JavaScript(UnityScript) and Boo. Some of the highlights of Monodevelopare :

**Multi-platform** Supports Linux, Windows and Mac OS X.

**Advanced Text Editing**Code completion support for C#, code templates, code folding.

**Configurable workbench**Fully customizable window layouts, user defined key bindings, external tools

**Multiple language support**C#, F#, Visual Basic .NET, C/C++, Vala

**Integrated Debugger**For debugging Mono and native applications

Although Unity3D is preconfigured to use MonoDevelop, It can be configured to use Visual Studio for development. It provides a more sophisticated C# development environment, think smart auto-completion, computer-assisted changes to source files, smart syntax highlighting and more.

**6.2 Scripting**

Scripts are components that can be attached to gameObjects. There are various ways of attaching components to gameObjects. It can be grabbed from the project view to the scene view or the hierarchy view on to the gameObject. We can also use the add component option on the gameObject.

**6.2.1. Health and Pickups**

Falling off from the floor or game environment in unity will lead to falling forever. It will be ideal for the game to restart or end when objects fall of the floor or hit enemy targets. This can be done using a health and damage script to the gameObjects. The health script attaches a hit point to the player and the damage script attaches a damage point to the enemy units or death floor. The damage script affects the hit-point and when hit-point of the player is zero or less, the game will end or restart.

Health Script

|  |
| --- |
| Health Points (float) |
| Re-spawn Health Points (float) |
| Number of Lives (int) |
| Is Alive? (bool) |

Damage Script

|  |
| --- |
| Damage on Trigger (float) |
| Damage on collision (bool) |
| Continuous Damage (bool) |
| Continuous Damage between hits (float) |
| Destroy self on impact (bool) |
| Delay Before Destroy (float) |
| Explosion Prefab (gameObject) |

**6.2.2. Spawners**

Spawners are added for both coins and enemies. The objective is to add more coins to collect with time but at the same time, more enemies to increase the difficulty of the level. Spawners are implemented as empty game objects, which are attached with a script to spawn prefabs.

Enemy/Coin Spawner

|  |
| --- |
| Spawn Prefab (gameObject) |
| Min seconds bet spawning (int) |
| Max seconds bet spawning (int) |
| Chase Target (gameObject) |

Box Spawner

|  |
| --- |
| Spawn Objects (list) |
| Seconds bet spawning (float) |
| X min (int) |
| X max (int) |
| Y min (int) |
| Y max (int) |
| Z min (int) |
| Z max (int) |

**6.2.3. Chase Script**

public class Chaser : MonoBehaviour {

public float speed = 20.0f;

public float minDist = 1f;

public Transform target;

void start(){

target = GameObject.FindWithTag ("Player").GetComponent<Transform>();

}

void Update () {

// face the target

transform.LookAt(target);

//get the distance between the chaser and the target

float distance = Vector3.Distance(transform.position,target.position);

// So long as the chaser is farther away than the minimum distance, move towards it at rate speed.

if(distance > minDist)

transform.position += transform.forward \* speed \* Time.deltaTime;

}

}

**6.3. Game Manager**

With Unity doing the magic for us behind the scenes, it becomes difficult to know how it works. It creates uncertainty on how to control the flow of the program. This is the purpose of the game Manager. A Game Manager is a something that keeps track of what state the game is in, manages the menu/pause systems, records and stores information for various purposes (audio/video settings, control bindings, game save data). It needs to exist over multiple levels.

# 7. BUILD AND DEPLOY

### C:\Users\Ankur\Desktop\Project\New folder\BUild and Deploy.png

# 8. REFERENCES

1. Unity 3D Manual

http://docs.unity3d.com/Manual/index.html

[2] Unity Script Reference

<http://docs.unity3d.com/ScriptReference/index.html>

[3] MSUs Game Development Program

<http://gamedev.msu.edu/>