Year 13 Computer Science Coursework  
2D platformer game in C#

11/07/2022

Contents

[Introduction 2](#_Toc127359706)

[Analysis 3](#_Toc127359707)

[1. Analysis of current systems 3](#_Toc127359708)

[2. Questionnaire Results 4](#_Toc127359709)

[3. Interviews (Blank) 8](#_Toc127359710)

[4. User analysis (Blank) 8](#_Toc127359711)

[5. Identified Issues in the current systems: 8](#_Toc127359712)

[6. Limitations (Blank) 8](#_Toc127359713)

[7. Final proposition (Blank) 8](#_Toc127359714)

[8. Testing: 9](#_Toc127359715)

[Player Movement Script V1: 9](#_Toc127359716)

[Camera Follow V1: 9](#_Toc127359717)

[Currency System V1: 10](#_Toc127359718)

[Currency System V2: 11](#_Toc127359719)

[Currency System V3: 11](#_Toc127359720)

[Currency System V4: 12](#_Toc127359721)

[Main Menu V1: 13](#_Toc127359722)

# Introduction

My project is going to be a non-linear 2D side-scrolling Metroidvania in which the player navigates a cave like structure, fighting enemies in search of different items and keys in order to help them progress through the level.

My inspiration for this game is the Moon Studios’ Ori series, in which you play as a small mythical animal wall climbing and wall jumping whilst fighting enemies along the way.

Another inspiration for this game is Spelunky. In Spelunky, you are a miner navigating a cave system full of enemies and booby traps. Collecting money and materials to buy better weapons to help you progress through the levels.

Personally, I find the Ori games too slow and that there is no clear way to go or end goal. I aim to fix this by making it clear what the player’s current objective is and I will also speed up the pacing of the game to match both mine and my target audience’s needs. I also do not how in Spelunky, you are required to blow up the terrain in order to progress, I feel as if this feature confuses the player and slows down the speed of what ought to be a fast-paced game.

# Analysis

## Analysis of current systems

‘

‘Ori and the Blind Forest’ (Xbox and PC, 2015)

Ori and the Blind Forest is a Metroidvania platform adventure video game developed by Moon Studios and published by Microsoft Studios. It was released in March 2015 on Xbox and PC and was later ported to the Nintendo Switch in September 2019.

In this game, the player controls Ori, a small white creature, navigating caves, moving between platforms, solving puzzles, and fighting enemies and bosses. The player can unlock weapons and abilities by progressing through the terrain, this helps the player to fight enemies and bosses easier and unlock paths to places they haven’t been yet.

Ori was made using the Unity engine because the developers liked Unity’s “flexibility and extensibility”. As I am going for a similar style of game, I could also use unity for my project.



‘Hollow Knight’ (All mainstream modern platforms, 2017)

Hollow Knight is an indie 2D side-scrolling Metroidvania game, developed and published by an independent developer Team Cherry in 2017. Hollow Knight was officially released for Windows in February 2017, with versions being made for Linux and Mac OS in April later that year. It was ported to the Nintendo Switch in June 2018 and eventually came out on Xbox One and PlayStation 4 in September 2018.

In Hollow Knight, the player controls an unnamed insect-like knight exploring the underground world. Throughout the game the player uses melee combat to fight hostile bugs and creatures to allow them to progress forwards through the game.

Hollow Knight was also made with the Unity engine for the same reasons that Ori was.

## Questionnaire Results

I have conducted market research for my game by creating a Microsoft Office Forms questionnaire in order to know if it would be beneficial for me to produce certain features within my game. I have specifically asked gamers important questions such as the desired platform of choice, genre of game, graphical style and other features that they may want to see or be added to my game. I have also investigated the demographics and psychographics of these gamers; this will make it straightforward to see which audience I will need to appeal to in order to make my game more successful.

Chart, pie chart

Description automatically generated

Which is your favourite platform to play on?

* PC
* Console
* Mobile

This pie chart makes it obvious that the overwhelming majority of gamers prefer to play on a PC opposed to a console or on their mobile device. This may be because PCs are generally a lot less limiting in what they can do, and they are more common than owning a console. This made it apparent that I will be developing my game for PC rather than console as they are also extremely difficult to get your game published onto, so it was likely out of the question anyway. PC is also a better option to mobile, not only because it was more popular but there is also a very competitive market for mobile games, and it is more difficult to develop and playtest.

Chart, pie chart

Description automatically generatedWhat is your preferred input device?

* Keyboard and mouse
* Gamepad controller
* Touchscreen
* Joystick

From the market research I have gathered, a keyboard and mouse is the most popular input device. This is due to its many inputs and versatility. A mouse gives you a much more accurate control than a gamepad or joystick as you can flick onto things quickly on the screen as there is a much larger surface area for the mouse to be moved on opposed to only a few centimeters of movement like an analogue stick on a gamepad controller. Keyboards also have much more customisability when it comes to different button inputs, this can make it a more comfortable end user experience.

Chart, pie chart

Description automatically generatedDo you prefer casual or competitive gaming?

* Casual
* Competitive

Chart, pie chart

Description automatically generatedThis chart represents how much of our target audience prefers casual gaming, compared to competitive gaming. From this, I have gathered that roughly ¾ of my audience prefer casual gaming although the amount that prefer competitive is definitely not insignificant. Plenty of casual games have competitive players, competing for the fastest time or highest score. I could implement a leaderboard or stopwatch into the game so people can compete to have the fastest times and “speedrun” the game.

Do you enjoy fast or slow-paced games?

* Fast
* Slow

From this I can take away that I will need to make my game fast paced so my end user does not get bored and lose interest. For this to happen, I will need to give the end-user enough things to do and to avoid long stretches of time with nothing going on. I will also try and avoid repetitive making repetitive or complicated tasks for the end user to go through.

Do you prefer top-down or side-view 2D games?

Chart, pie chart

Description automatically generated

* Top-down
* Side-view

The vast majority of my target audience prefer the side-view over style over the top-down style. From this, I will take away that I will do a side-view.

Chart, pie chart

Description automatically generatedWould you prefer to have an online, connected experience or to be completely offline?

* Online
* Offline

From this, I have learnt that the majority of players prefer for a game to be online. This may be difficult so I will create an offline, singleplayer experience first and if I am successful in completing that then I will work on an online element to my game. An score leaderboard could be a simple way to make my game online and connected.

Chart, pie chart

Description automatically generatedHow Important is immersive sound to you?

* Extremely important
* Somewhat important
* Neutral
* Somewhat not important
* Extremely not important

From this, I know that creating the game with immersive sound is important however as it’s 2D, I can make the sound stereo and it come from either the left or right speaker respectively.

What keeps you invested in a video game?

Chart, bar chart

Description automatically generated

* Multiplayer
* Visual Graphics
* A compelling story
* Consistent updates
* Amount of content

A large part of my target audience have said that the amount of content available to play is what keeps them invested. A lot of games have very repetitive tasks to make up for a lack of content, I will try and avoid this at all costs.

## Interviews (Blank)

## User analysis (Blank)

## Identified Issues in the current systems:

Ori and the Blind Forest

With the information I have gathered from the analysis on Ori I have found that the game’s pace is often extremely slow until the player finally progresses, then the speed and difficulty of the game changes drastically, leaving the user in a situation that they do not know how to deal with. It is also not clear what you’re supposed to do or what your next objective actually is. This further slows down the pace of the game and just confuses the player. These issues tend to frustrate the player and make them lose interest very quickly. Moon Studios does however seem to focus a lot on the graphics and visual aspects of Ori, this helps make up for the poor gameplay but as the visuals of my game aren’t assessed, there is a lot of room for improvement upon this game which I will aim to fix it mine.

Hollow Knight

After my research, I have concluded that Hollow Knight suffers from one of the same issues that Ori does, it is not clear where you’re supposed to go or what you have to do. Some “secrets” are also ridiculously difficult to find without searching up a tutorial on the internet of what to do. Finally, the gameplay is very difficult for new players. I aim to fix these problems by making sure navigating the level is simple

## Limitations (Blank)

## Final proposition (Blank)

## Testing:

System Hardware:

GPU GeForce RTX 3050Ti (Mobile)

CPU AMD Ryzen 5 5600 (Mobile)

RAM 8GB DDR4

Storage 500GB M.2 NVME SSD

### Player Movement Script V1:

public class PlayerMovement : MonoBehaviour

{

public CharacterController2D controller;

public float runSpeed = 40f;

float horizontalMove = 0f;

bool jump = false;

// Start is called before the first frame update

void Update()

{

horizontalMove = Input.GetAxisRaw("Horizontal") \* runSpeed;

if (Input.GetButtonDown("Jump"))

{

jump = true;

}

}

// Update is called once per frame

void FixedUpdate()

{

controller.Move(horizontalMove \* Time.fixedDeltaTime, false, jump);

jump = false;

}

}

##### No Error.

---------------------------------------------------------------------------------------------------------------------

### Camera Follow V1:

public class CameraFollow : MonoBehaviour

{

public Transform Target;

Vector3 tempVec3 = new Vector3();

void LateUpdate()

{

if (Target != null)

{

//z axis

tempVec3.x = Target.position.x;

tempVec3.y = this.transform.position.y;

tempVec3.z = this.transform.position.z;

this.transform.position = tempVec3;

//y axis

tempVec3.y = Target.position.y;

tempVec3.z = this.transform.position.z;

tempVec3.x = this.transform.position.x;

this.transform.position = tempVec3;

}

}

}

##### No Error.

---------------------------------------------------------------------------------------------------------------------

### Currency System V1:

public class CurrencySystem : MonoBehaviour

{

public GameObject currencyText;

public int Score;

void OnTriggerEnter(Collider other)

{

Score += 50;

Score.GetComponent <currencyText>().text = "SCORE: " + Score;

Destroy(gameObject);

}

}

##### Errors:

Nothing works.

Assets\Scripts\CurrencySystem.cs(14,29): error CS0246: The type or namespace name 'CurrencyCounter' could not be found (are you missing a using directive or an assembly reference?)

Assets\Scripts\CurrencySystem.cs(14,15): error CS1061: 'int' does not contain a definition for 'GetComponent' and no accessible extension method 'GetComponent' accepting a first argument of type 'int' could be found (are you missing a using directive or an assembly reference?)

---------------------------------------------------------------------------------------------------------------------

### Currency System V2:

public class CurrencySystem : MonoBehaviour

{

public GameObject currencyText;

public int Score;

void OnTriggerEnter(Collider other)

{

Score += 50;

currencyText.GetComponent<Text>().text = "SCORE: " + Score;

Destroy(gameObject);

}

}

#### Fixes:

Complete change of code.

##### Errors:

Object is not destroyed.

Counter does not display.

Counter does not increase.

Error Code

---------------------------------------------------------------------------------------------------------------------

### Currency System V3:

public class CurrencySystem : MonoBehaviour

{

public Text currencyText;

private int Score;

void Start()

{

Score += 50;

currencyText.text = "Currency: " + Score;

}

}

#### Fixes:

Counter now displays currency.

##### Errors:

Object not being destroyed.

Counter does not increase.

Error Code

---------------------------------------------------------------------------------------------------------------------

### Currency System V4:

public class CurrencySystem : MonoBehaviour

{

public Text currencyText;

private int Score;

void Start()

{

Score = 0;

currencyText.text = "Currency: " + Score;

}

private void OnTriggerEnter2D(Collider2D MyScrap)

{

if(MyScrap.tag == "MyScrap")

{

Score += 1;

Destroy(MyScrap.gameObject);

currencyText.text = "Currency" + Score;

}

}

}

#### Fixes:

No error codes.

##### Errors:

Object not being destroyed.

Counter does not increase.

Player does not collide with gameobject

---------------------------------------------------------------------------------------------------------------------

### Currency System V5:

public class CurrencySystemV5 : MonoBehaviour

{

public Text currencyText;

private int currencyScore;

void Start()

{

currencyScore = 0;

currencyText.text = "Currency: " + currencyScore;

}

private void OnTriggerEnter2D(Collider2D other)

{

if (other.CompareTag("MyScrap"))

{

Destroy(other.gameObject);

currencyScore += 5;

currencyText.text = "Currency: " + currencyScore;

}

}

}

#### Fixes:

Player now collides with gameobject

##### Errors:

Object not being destroyed.

Counter does not increase.

### Main Menu V1:

public class MainMenu : MonoBehaviour

{

public void PlayGame()

{

SceneManager.LoadScene("Main");

}

}

#### Fixes:

“Start Game” button loads the scene “Main”.

##### Errors:

Have not yet implemented “” Quit Game” button or any other scene buttons.