Logo

Description automatically generated

Icon

Description automatically generated

CTEC3451 Development Project

**“Creating a Control Mapping Program for Game Accessibility”**

Final Deliverable

Project Supervisor: Jethro Shell

Project Author: Ruya Kumru-Holroyd (P2512547)

Word Count: 0000

Contents

[Acknowledgements 3](#_Toc103801283)

[Introduction 3](#_Toc103801284)

[Background 3](#_Toc103801285)

[Main Body 3](#_Toc103801286)

[Development Cycle 3](#_Toc103801287)

[Basic Functionality 3](#_Toc103801288)

[System Design 3](#_Toc103801289)

[Underlying Data Structures & Algorithms 3](#_Toc103801290)

[User Interface 3](#_Toc103801291)

[Testing 4](#_Toc103801292)

[Critical Evaluation 4](#_Toc103801293)

[Project Evaluation 4](#_Toc103801294)

[Evaluation of your approach 4](#_Toc103801295)

[Evaluation of tools used: 4](#_Toc103801296)

[Final paragraph 4](#_Toc103801297)

# Acknowledgements

To begin this project report, I would like to give thanks to people who helped me along the development process and my time at De Montfort University, this includes all the lecturers and lab tutors who have provided me with a variety of knowledge and skills, throughout all the different modules. They have also been incredibly helpful when I have asked for their assistance.

I would also like to thank Dr Jethro Shell, my supervisor for this Development Project, who has guided me and helped me throughout since the start of this project, as well as helping to come up with the project itself. And generally, being supportive of my project and its themes.

Finally, I would like to thank my friends and family for their support over the years at De Montfort University. My family who has supported me and been there for me throughout, my housemates who provided encouragement and finally a massive thanks to the friends that I have met on the Computer Games Programming course, who have shared their knowledge and help in times of need, and also given fun and laughter through working together whether in person or on a Discord server, providing me the confidence and motivation in completing this project and other modules.

# Abstract

There are millions of video games available worldwide, but how many are actually accessible and well represented? The answer is less than you might expect, which is why this project was created to demonstrate that it is possible to do so and that it can be used and altered for future usage.

This report details the development of a control mapping program for game accessibility, as well as the two prototype games created to test the control mapping program, including the system design of the project, the development cycle, the functionalities, user interfaces and the testing of the project. Finally, providing a critical analysis of the project to conclude the Development Project.

# Introduction

## Background

The main product of this project is a Control Mapping Program for Game Accessibility, it maps the control settings from one game and loads it to another. Included with the project, are two Prototype Games in which I created to test and use the mapping program between; “Wheelchair Basketball Shoot”, a 2D basketball shooter, and “Bye-Bye Thoughts”, a 3D first person shooter.

The main functions of this project, is by loading one of the prototype games, navigate through its’ menu and to the ‘game settings’ scene where it allows the player to select and change their key bindings and what type of input device they are using, whether it be keyboard and mouse or a controller, and then save these changes which saves to a json file in an accessible file for both prototype games. Then, the player can load the other prototype game, go into its game settings after navigating through the menu and press load and it loads the changes in key bindings to this second prototype game. Unfortunately, the keyboard and its key bindings are the only input device that can be currently mapped and saved, but the use of a controller input in the second prototype is possible, the first one does not use a controller due to the simple controls.

Timeline

Description automatically generated with low confidence*Graphical user interface

Description automatically generated*Diagram

Description automatically generated

*Figure 1: Navigation of the first prototype*

*A screenshot of a video game

Description automatically generated with medium confidenceGraphical user interface, application

Description automatically generatedA screenshot of a video game

Description automatically generatedA screenshot of a video game

Description automatically generated*

*Figure 2: Navigation of the second prototype*

## The need for the project

Such a program and its prototypes are needed because both of these games include representation of disabilities within them to deliver the importance and the ability to include these in video games, as the Games Industry lacks this with only a handful of games that include appropriate representation. The same goes for accessibility in games, it is usually an after-thought in the development in video games, which is why a program such as this is necessary, so that the developers can include it into their games with ease, which allows them to improve on the accessibility their game provides and may even give them further incentives to include more accessibility options and disability representation. It was also a chance to develop personal skills and become more familiar and confident in creating games using Unity 3D.

## Objectives

The aim of this project is to create a program that will map the specific control settings and its key bindings, off a prototype game and transfer them onto a different prototype game, to allow for accessible gaming. And to develop two prototype games and to include disability representation within them to ensure the possibilities and ability to do. As well as being accessible with the ability to set key bindings and save the changes in the settings using json and load the saved changes. The software that is created will be well constructed and work as intended. Furthermore, another objective is User Testing by gaining primary data from a group of individuals in testing the two prototype games and mapping their selected input decisions and key bindings and output it. Finally, to document the process in the form of a project development report.

# Main Body

## Development Methodology

The development lifecycle of this project, started with planning on what the project would be

The methodology for development applied to this project was the Agile Methodology. This allowed the components of the projects development to be broken down into a number of stages, completing each stage before moving onto the next, making sure that all the requirements had been met and there were no bugs or issues in the functionality of that stage. These components would be broken down based on the size of each task, and these become the focus of each sprint. Choosing Agile Methodology was beneficial for this project, as there was already a clear split of the tasks, with the control mapping program itself and the requirements of two prototype games needed for testing, these tasks were already large components and were focuses of each sprint. Agile methodology also allows for breaking up the tasks within each component further, such as within making one prototype game there is many tasks that one would focus on such as designing and implementation, before completing, testing, and moving onto the next stage.

A Gantt chart was created at the start of the project to split up the tasks and plan the development time for each one. The Gantt chart which can be seen in Figure 3, can show that the timing of some tasks took place later than expected or took longer than originally had planned. Although, it was accurate in the planning of the order in which tasks would be executed. The implementation of the control mapping program took a lot longer and later than expected, this was due to there still being small fixes to the prototype games that had to be completed before moving onto that stage, and also due to outstanding circumstances such as illnesses, jobs, and other university module commitments. The time management could have been a lot better, and if it had been then some of the tasks that were implemented later than planned, could have been on track with the Gantt chart.

## Basic Functionality

In the first report for this project, the functional requirements were set out and identified, this was done by constructing a Use Case Diagram (UCD) for the main sections that required their own functional requirements and use cases. There were two made, one for the main menus for each of the prototype games, with its own requirements and path, and the second being the game prototypes functional requirements themselves. Figures 4 and 5 show these.

Diagram

Description automatically generatedDiagram, schematic

Description automatically generatedFigure 4: Use Case Diagram for the Main Menu scenes

Figure 5: Use Case Diagram for the Game Prototypes

These diagrams show each functional requirements in the two game prototypes, the control mapping program itself is included in the main menu as that is where it is saved and loaded, within the game settings which is only accessed through the main menus.

In description, the way that the Player can navigate through the games and use the control mapping program, is by running one of the prototype games, for example Wheelchair Basketball Shoot, the player opens up first to the main menu which includes three buttons; play game, game settings, and exit game. Pressing play game takes the user through the Wheelchair Basketball Shoot game play, there is a back button that will take the user back to the main menu once they are finished with the gameplay. Pressing game settings brings the user to a settings page which includes the buttons and key bindings for the game. There is also save and load buttons which allows the user to save their key binding preferences, which is done by pressing one of the key binding’s buttons, pressing again with the new key that the player would like to be bound to the action, which is described next to the button, and that sets the new key bind and by pressing save this will save it to a json file that can then be loaded either in this prototype or the other prototype Bye-Bye Thoughts.

## System Design

The Control Mapping program is designed to help with game accessibility, capturing the mapping of keyboard controls, how these mappings can move from two different playable prototype games and help benefits disabled players gaming experiences, making games more accessible and easier to play. The player is supplied with two playable prototype games that include their own menu and game settings sections where they can save their own key bindings or load already previously set ones from another game, which are saved by using json and a created a json file that is accessible for both games using the same directory path, as the scripts create a folder called ‘SaveData’ with a json file inside, which is in the users Documents file on their own computer.

The project was created using Unity3D version 2020.3.12f1.

Controls, per game prototype, are relatively simple and easy to understand which is important due to the fact “if they are too difficult to learn or if the game experience becomes uninteresting then you’re creating barriers that only skilled and patient players will be able to overcome,” (Carrera, 2016). This is especially necessary when considering accessibility in games, as players won’t be able to play the games if they are difficult for them or unable to control.

### Control Mapping Program

To keep the control mapping program simple, easy to understand and accessible for all, the code itself is within the back-end of the project, and the front-end where it is accessed is the two prototypes’ games menu/game settings UI which can change the key bindings and save and load the keycodes from the json file.

The ‘program’ itself is a collection of scripts made using C# language and Json. There is a save manager that includes the write and read functions to and from the json file, as well as creating the directory in which it is held and there is a save object script, which is slightly different between the two prototypes as it holds the key codes needed in each of its game settings scene. Those two scripts bring the essence of the control mapping program, but the game settings and game manager scripts were also essential in putting it together. Both prototypes have all 4 scripts so theoretically by taking these scripts and including them in another game, just by editing the game settings and game manager for that game, if a basis script was created it could be more executable as a program. However, that is not something that was created in this project, but it is something that would be developed in the future.

When first developing the control mapping program, the first step was to create key bindings in a prototype game and make it able to switch and save so that the player could choose any key they would like to play with. So, when first saving these key bindings so that they would be set and playable, PlayerPrefs was used instead of json. This was because when researching and testing ways to change the keys a tutorial included PlayerPrefs which saves key codes in the computer’s registry. This meant, that when the prototype game was closed and then opened again, any changes in the key binding had been automatically set. This did work very well, however it was not ideal when it came to mapping the key changes from one prototype to another, because as mentioned earlier, PlayerPrefs saves the key codes to the registry which is not an easily found or accessible directory. Although it is possible, it was not suitable for the project at the time, and json allowed for an easy creation of a directory in the users Documents folder which is more accessible.

Graphical user interface, application

Description automatically generatedGraphical user interface, application

Description automatically generated  
Figure 6: Game settings menu in second prototype, with save and load and changeable key buttons

### Prototype 1: Wheelchair Basketball Shoot

The first prototype, titled “Wheelchair Basketball Shoot” is a simple as the name suggests, it is a 2D basketball shooter game where the player aims and releases a ball to a basketball net, gaining points for each shot. There are 2 levels currently built in the prototype, the second more difficult than the first, but there would be multiple levels if this game were to be built to completion. The default keys set to control the player are the keys A and D on a keyboard, to move left and right respectively on the screen, the ball follows as the player moves. The ball can be aimed and released by using the mouse, holding down the left mouse button and dragging down on the screen to aim the ball into the basketball net. The strength at which the ball is fired at depends on the speed and duration of the mouse button being held down. It is possible to invert the y position of the ball drag with a mouse, instead of top to bottom you can change it to bottom to top in the game’s settings. There was a planned line renderer to be added to the game prototype but due to lack of time and needing to create the mapping program, this was not added.

The action needed to aim and shoot the ball using the mouse, by dragging on the screen and aiming at the basketball net, is accessible because the player can start and end the drag position at any point on the screen in their preferred way, instead of a specific control area. There is also an option to use a key and hold it down and it will projectile in the direction of the basketball hoop the strength depending on how long the key is held down for. Although, this option does not currently work within the prototype. It has been tested that if it is selected and a key has been selected, that key is saved in the json files and can be loaded and will translate in the game but currently if it were to be selected, nothing would happen as it has not been implemented. This was added mainly to test json files and also to be a future functionality of the game for a possible future completed version of the game, past the prototype.

There are two levels included the first being a simple throw and shoot, the second the basketball hoop moves adding some difficulty, as these games are prototypes, only two levels were necessary as it was enough to test the robustness of the game and its controls.

The representation of disability in games, for this prototype, as explored within the Literature review in the first report, has come from (Brody, 2020) AbleGamers article about the need for more disabilities in the games we play, where it was suggested that sports games can be adapted to include disabilities such as basketball games can have a wheelchair basketball player. So, it was decided to create this prototype from this idea and showcasing the possibility of including representation this way.

Diagram

Description automatically generatedDiagram

Description automatically generated  
Figure 7: Screenshots of Prototype 1 game, Level 1

Diagram

Description automatically generatedDiagram

Description automatically generated  
Figure 8: Screenshot of Prototype 1 game, Level 2

The Game Settings menu for this prototype, only has a few changeable key options and game options due to the very simple game mechanics. As can be seen in Figure 9, there is only two options for moving the character which is moving it left and right. The option for choosing ball drag/throw with a key press and the player can set which key that is, or using the mouse to drag the ball, and finally the option of having the y position of the ball drag inverted or not.

Graphical user interface

Description automatically generated  
Figure 9: Game Settings scene in Prototype 1

All the art assets within this prototype were all created by using a sprite draw website called Piskel, (Piskel - Free online sprite editor, n.d.), the animations were also created in the Unity editor.

Logo

Description automatically generated Logo

Description automatically generated  
Figure 10: Character art and animation stages of staionary and ball thrown in Prototype 1

Bye-Bye Thoughts:

Looks very prototype-y tried to make it look somewhat nice but was just testing out and made a simple terrain, and very simple shootable objects that simply have the text ‘bad thoughts’ on them. For the fully developed game these would be cloud shaped bubbles that would have texts of actual bad thoughts that a person with depression could have and they would dissipate when the player shot them.

## Underlying Data Structures & Algorithms

-original key bindings

-Json

-Show player prefs stuff

-Physics for prototype 1 game

-Gameplay for prototype 2

## User Interface

Due to the two games being prototypes, the User Interface is relatively simple in its design and functionality.

## Known Issues

(Adding here, to be changed and adapted later)

Found out thar the base Unity Input Manager does not allow for re-bindings of any kind, controller, or keyboard, whilst it is on runtime, this messed up the way that I originally planned to do Inputs and mapping the controls, so I had to rethink and change my plans.   
I was able to fix key bindings for the first prototype, Wheelchair Basketball Shoot, relatively easy because when I was first researching ways of changing controls, I set up a custom input manager with changeable key-bindings only, but at the time I could only change these key-bindings in the inspector and not in the game itself during runtime. But despite this, it gave me a foundation to build on after having to change my original plans.

(Could have used Input System but it was too new and complicated, knew nothing about it and would mean have to re-build a lot of things from the start which was not viable for the point I was at in the development process)

Second Prototype, Bye-Bye Thoughts, included standard assets from the Unity Asset store which had its own First-Person Controller which I included into the game. I chose to use this for the prototype so I could build it more easily and spend more time on the mapping program itself.   
This sort of backfired as the scripts in the folders for the First-Person Controller used the Input Manager that is built into Unity which I decided I was going to use and re-bind controls using that, but I found out too late in the development process that it was not possible to rebind Input Manager controls during runtime.   
Therefore, I had to change controls in the assets scripts to do key-bindings and keyboard controls, but I did include the use of the Input Manager to allow for controller use which the player can switch to in settings but cannot change the control button bindings.

Another issue from using the First-Person Controller was that because it was in the Standard Assets, I had to download all of them and could not delete all of the files that I didn’t need because some were connected to the scripts that were being used for the controller, that being said I did manage to lose a few. Also, because those scripts were in their own folder, for some of my own scripts I had to find a work-around to get the scripts to work with each other. A way I did this was just by creating some of the new scripts in that folder as it was able to be accessed in the game either way, and any I didn’t need to connect to the controller I kept in my own Assets Scripts folder. This was not ideal and definitely not robust, but it was the only option at the time.

## Testing

# Critical Evaluation

## Project Evaluation

-How much does it do, how good is it?

-Are there any bits you are particularly proud of?

-What isn’t implemented? How would you extend it given more time? : mapping program as a separate unity project / executable.

Can use first person active voice for this section

## Evaluation of your approach

-In terms of project management approach and in terms of development methods or research methods used

-How successful was the approach you adopted?

-What techniques did you use that worked well? Why?

-What did you learn by doing the project? – did you meet your academic objectives

-How would you do it better if you did the project again?

## Evaluation of tools used:

-What languages, libraries, environments etc did you use for the development? Or, what libraries, methods, techniques did you use for the research?

-Were they suitable? What were their main plus and negative points?

## Final paragraph

-end on a positive note

# References

Brody, C. (2020). *The Need For More Disabilities In The Games We Play*. [online] ablegamers.org. Available at: <<https://ablegamers.org/the-need-for-more-disabilities-in-the-games-we-play/>> [Accessed 6 Jan. 2022].

Carrera, S. (2016). *Accessibility in Games: Including People with Disabilities*. Germany: LAP LAMBERT Academic Publishing.

Piskelapp.com. n.d. *Piskel - Free online sprite editor*. [online] Available at: <https://www.piskelapp.com/> [Accessed 27 December 2021].

# Appendix

## Project Availability

This project can be accessed through a public GitHub repository under the username RuyaKH, the link for this repository is; <https://github.com/RuyaKH/Final-Year-Project>

## Figure Table

|  |  |  |
| --- | --- | --- |
| Figure No. | Image | Description |
| 1 |  | The basic functional process of the first prototype game, starting from left to right |
| 2 |  | The basic functional process of the second prototype game, starting from top left to the bottom right. |
| 3 |  | The projects Gantt Chart |
| 4 | Diagram, schematic  Description automatically generated | Use Case Diagram for the Main Menu’s for each game prototype |
| 5 | Diagram  Description automatically generated | Use Case Diagram for the game prototypes |
| 6 |  |  |
| 7 |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |