



# GHOSTVILLE

Gesture based UI

## ABSTRACT

This is the PDF document for the Gesture Based UI project. This is a project made in unity which uses both player inputs and voice commands in order to control the game.

Daniel Gallagher – G00360986

Final Year – Software Development

## Github Repository

<https://github.com/DanielGallagher6499/Ghostville>

## Screencast

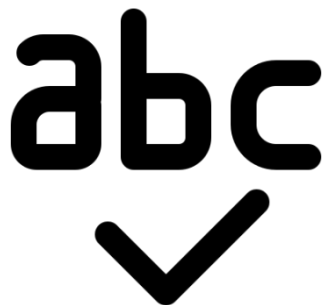
[https://www.youtube.com/watch?v=WgWD\\_9tDDS4&ab\\_channel=DanielG](https://www.youtube.com/watch?v=WgWD_9tDDS4&ab_channel=DanielG)



## **Ghostville.**

The game I decided to make is called “Ghostville”. It is a platform game like the very early day Mario games. The game offers the user a great, immersive fun game while also providing a cool retro feeling to the players experience. The game is made for all ages from new gamers who are young children to adults looking for an old nostalgic feeling game with a new modern twist.

The aim of the game is to collect as many coins as you can throughout your progression in the game. The maps are made to be challenging and include enemies who come towards the player and if hit they player will have to restart the level all over again! The game includes some cool animations and really feels well put together from a user point of view. The game was developed in Unity with some custom fonts added also to add to the old feel of the game.

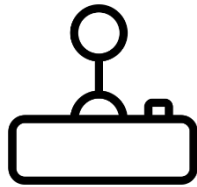


## **Grammar Recognition.**

The game uses your microphone to take in things the user might say. In the grammar recognisers the phrases and sentences recognised are defined in separate XML files. I made these files separate so I could use the users input voice to control the character but also to control pause menus and main menus also. I thought of as many phrases and sentences as I could to use in the game so that the recogniser would offer a better range of phrases and sentences for the user to say in order to do a certain action in the game.

These grammar recognisers can be very sensitive and slightly finicky which is one of the only downfalls I as a develops saw of them when developing my application [2]. I would have preferred to use an armband or some smart technology to complete this assignment but

with our current work schedule combined with my part time job and covid I was unable to find anything to use so resorted to using these gestures, which do not get me wrong are great to learn and to have knowledge on but for personal development in the unity area I would have liked to use smart technology with unity. [1]

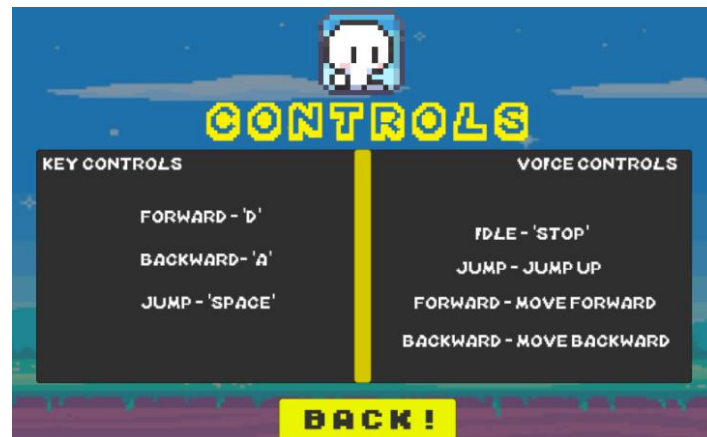


### **Purpose of the application.**

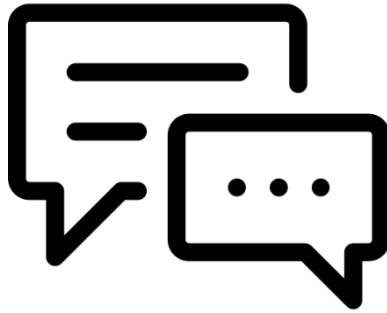
My application is like the old retro Mario games. These types of games for me personally are very interesting and enjoyable for many reasons. Firstly, they are a simple way to pass the time whether you are bored or just on a short bus ride playing simple old retro games is always a good way to pass the time. Secondly, I feel as though these types of games bring back childhood memories of playing on Gameboys and Nintendo Ds handheld consoles and I feel as though I am probably not the only one. Developing this game, I felt as though it would be cool for older people like me and even my parents to look back and play on a game from long ago rather than all the high-quality HD 3D games you see nowadays.

The below screenshots will show snippets of my game and the menus used inside of the game. As from the screenshots you can see it is very straight forward and easy to navigate which is intentionally done as I did not want to add a large learning curve or make the user mess around in the menus etc for long periods of time as this can lead to users becoming un-interested in the actual content of the game.





As I said above, I would have really liked to experiment with other smart technologies but that is certainly not off the cards as I am very much taking this project as a personal development of mine and I will be adding to the game in the future. Adding the voice recognition to the game really helped in a way as some gamers might find the game very easy and blast through the levels using the key commands and using the voice commands helps to add a level of difficulty to my game giving the veteran users a challenge. As I found in this article not all gamers will like hard games such as Dark Souls for example and therefore games like Call of Duty and Grand Theft Auto sell much better than hard games but that does not mean the hard games are bad it just means people are challenged by them. [3]



### **Gestures appropriate for this project.**

In this game I decided to use the built-in speech recognition with Unity. This allows developers such as me to define words and phrases that when said by the user and recognised via their microphone, does an action based on what they said. In my opinion, the user of the grammar recogniser fitted very well into the theme and feel of my game. The game itself is quite a retro feeling game which when combined with something very technical like a smart armband etc I feel might ruin some of the retro feeling made by the game. The grammar recogniser is quite subtle in comparison and does not even have to be used technically unless the user wants to as the game has the functionality of a normal game but can also be used with the grammar recogniser. As I also outlined above, using the phrases throughout the game as you try to complete it adds a new level of difficulty to the game especially for those who might be very good at games and blast through the levels. Using the phrases requires the user to get used to how fast the game paces and when and what to say to the game for the character to avoid dying. Overall, I would say the grammar recogniser or maybe a smart armband would have been the best fit for a game as these are subtle changes to the game but also provide the user with a new modern way to interact with a game.

### **Hardware used in creating the application.**

In this project I used my own personal gaming PC to develop the application as for one, Unity does require good power and a strong PC to run properly without constantly crashing which unfortunately, I have had loads of experience with using my older outdated laptop I got in first year for this course. Developing this game, I used a very good microphone to test and develop my grammar recogniser which is the "Blue Snowball" [4] as I felt this was much more accurate and far superior to the old built-in microphone I had used in my laptop for some of the labs as we experimented and began to develop with the grammar recogniser.

I also used my mouse and keyboard a lot obviously for developing the game but also the mouse is used to navigate the menus if a user does not want to use the voice commands and the keyboard is used for player movements in the game again if the user does not want to use the voice commands.

## Architecture for the solution

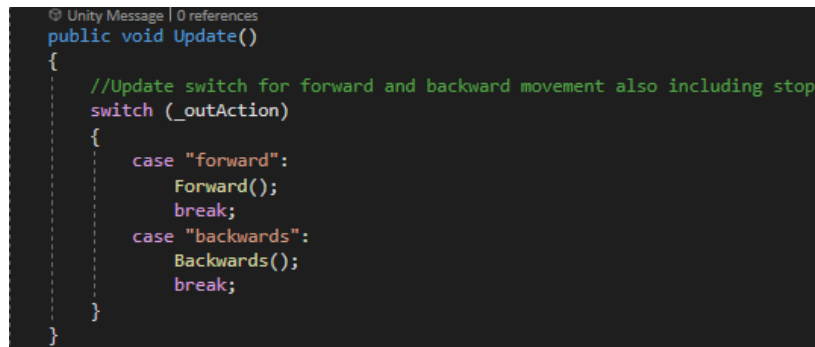
In this project I first started off by developing an environment like we have been taught through the unity section of the course. I first set up my scene with all the folders I would need and started to develop a scene with a character and some simple movements which I tested worked before moving on at all. I then made the XML files needed for the movement of the character and which contain the recognised phrases and words for the character movement actions. Shown below is one example of the XML file and the parts needed for the forward movement of the character.

```
<!--User Controls-->
<rule id="controls">
  <one-of>
    <!--Forward-->
    <item>
      <ruleref uri="#forward"/>
    </item>
```

```
<!--Forward recognized phrases-->
<rule id="forward">
  <tag>out.action = "forward";</tag>
  <one-of>
    <item>Go forward</item>
    <item>Forward</item>
    <item>Straight</item>
    <item>Go straight</item>
    <item>I want to go forward</item>
    <item>Straight ahead</item>
  </one-of>
</rule>
```

As you can see from this the top image is the top of our XML file where the controls rules are located. Then depending on the uri which in this case is forward it goes to the associated part in the bottom part of our file which contains the recognised words and phrases which the forward rule will accept.

Next is the grammar recogniser which I will show in the next images below. The grammar recogniser detects words said by the user and depending on what they say it calls a method which does a certain action or movement in the game. As you will see from the first image, we are still looking at the forward method as our example to keep things fluid in explaining them. The first thing we see from the first image is a switch statement which based on what is said is called. In our example if as we saw above forward has many phrases so let's presume the user says, "Go forward", saying this now enters us into this switch statement in the update method and calls our forward method which is located at the bottom of our file and is shown in the second image below.

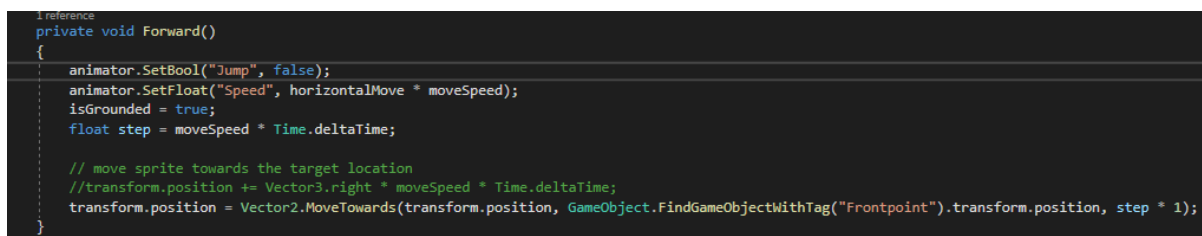


```

Unity Message | 0 references
public void Update()
{
    //Update switch for forward and backward movement also including stop
    switch (_outAction)
    {
        case "forward":
            Forward();
            break;
        case "backwards":
            Backwards();
            break;
    }
}

```

As you can see in this second image, we have our forward method. Once the grammar recogniser knows that the forward words have been said and we enter the switch statement this method is called and moves our player forward.



```

1 reference
private void Forward()
{
    animator.SetBool("Jump", false);
    animator.SetFloat("Speed", horizontalMove * moveSpeed);
    isGrounded = true;
    float step = moveSpeed * Time.deltaTime;

    // move sprite towards the target location
    //transform.position += Vector3.right * moveSpeed * Time.deltaTime;
    transform.position = Vector2.MoveTowards(transform.position, GameObject.FindWithTag("Frontpoint").transform.position, step * 1);
}

```

Not only does the grammar recognition work for the player movement but the keys also work using a separate script called “playermovement” which has the exact same method as the grammar recognition script for movement.

## Conclusions & Recommendations

Overall, I would say throughout my time in college I have enjoyed developing in Unity the most as it in my opinion is the best aspect of the course and I would love a career developing in Unity in the future. I feel as though it is more interesting than other modules as you can see what your code does in a game environment. Overall, I feel my game came out very well and very professional looking, but I also feel as though I could have developed more, and I genuinely would love to develop it more maybe including a coin store for other characters or items etc but with the time I had and the experience I have as of right now I feel as though my game came out very well.

If I was to do the project again, I would try and add a store if possible and also, I would have loved to develop it with a smart armband just to change up what I have learned and broaden my horizons in regard to the capabilities of developing in unity.

As a student throughout our time learning unity in this course I feel as though we learned so much which has great relevancy in today’s world and overall, I have really enjoyed my time learning unity and I really wouldn’t have any suggestions on what to change as I feel it’s all perfect but maybe for future years learning some more 3D unity might be beneficial but overall I think the unity aspect of this course is very good.

**Thank you!**

## **Project Links**

**GITHUB** - <https://github.com/DanielGallagher6499/Ghostville>

**SCREENCAST** - [https://www.youtube.com/watch?v=WgWD\\_9tDDS4&ab\\_channel=DanielG](https://www.youtube.com/watch?v=WgWD_9tDDS4&ab_channel=DanielG)

## **References**

[1] [https://www.youtube.com/watch?v=RteaV5tv85c&ab\\_channel=KlaudijusMiseckas](https://www.youtube.com/watch?v=RteaV5tv85c&ab_channel=KlaudijusMiseckas)

[2] <https://docs.microsoft.com/en-us/windows/mixed-reality/develop/unity/voice-input-in-unity>

[3] [https://www.gameinformer.com/b/features/archive/2014/09/25/why-we-like-hard-games.aspx#:~:text=But%20the%20vast%20majority%20of,they%20feel%20it%20was%20fun\).](https://www.gameinformer.com/b/features/archive/2014/09/25/why-we-like-hard-games.aspx#:~:text=But%20the%20vast%20majority%20of,they%20feel%20it%20was%20fun).)

[4] <https://www.pcmag.com/reviews/blue-snowball-ice>