Gesture Based UI Development Project

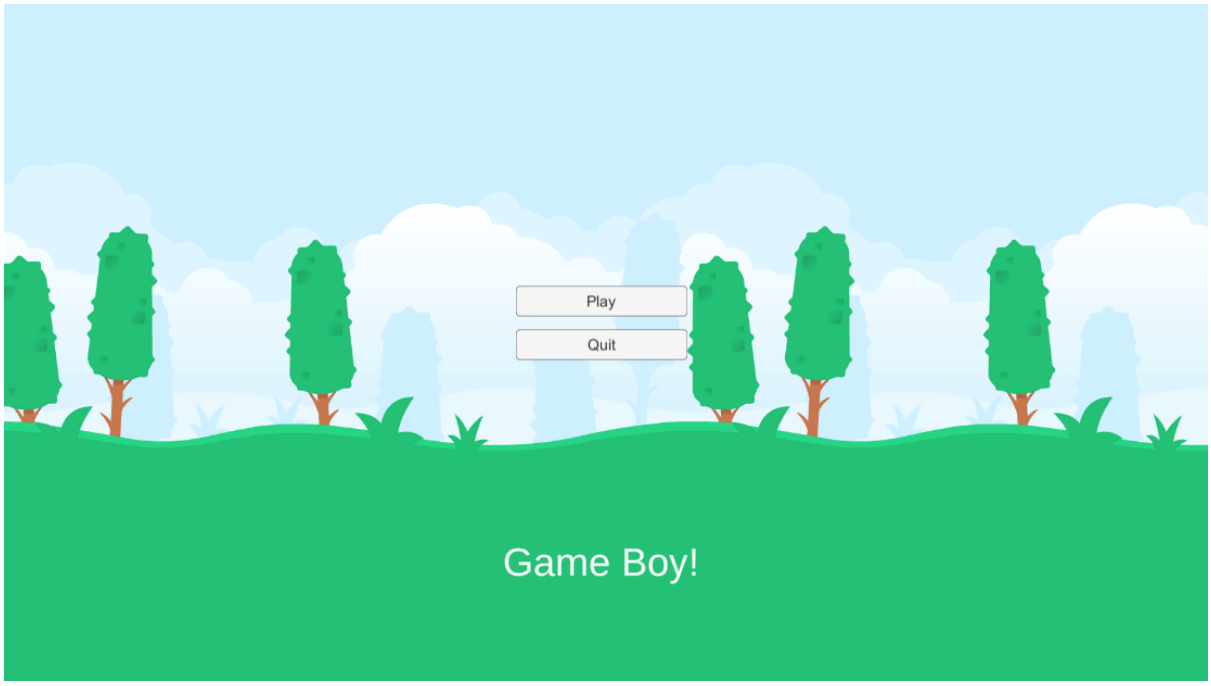
Darren Butler

G00299944

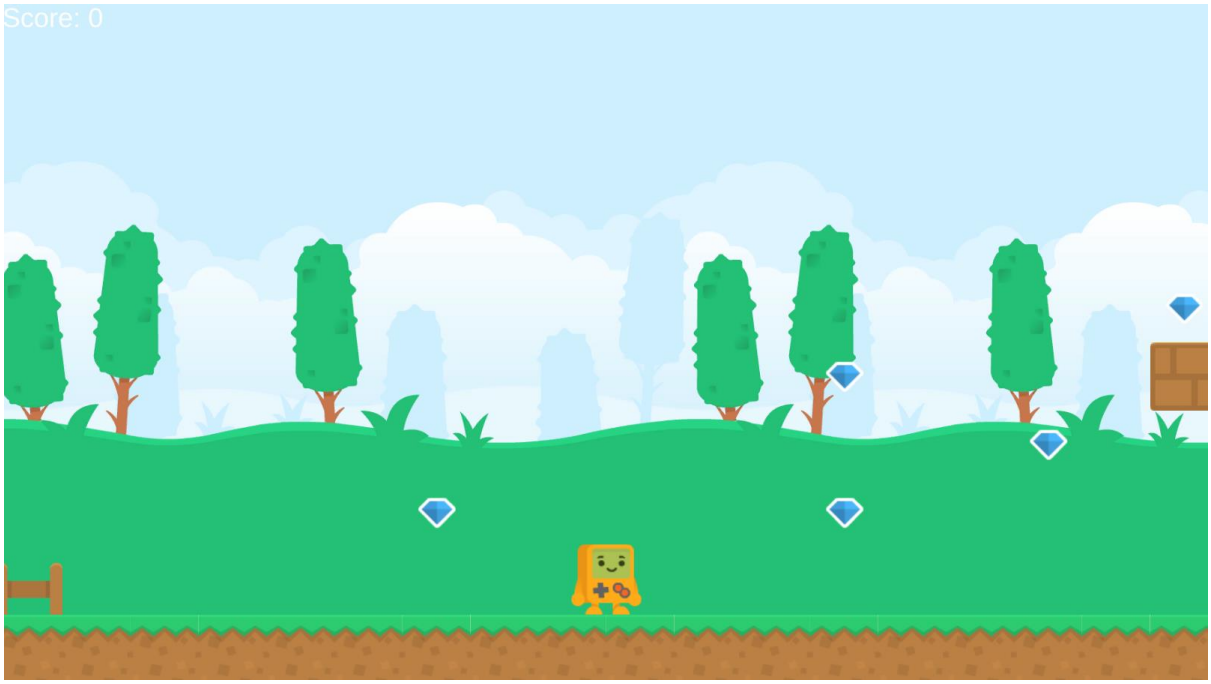
Project Source Code: https://github.com/dnbutler64/GBUID-Project

Purpose

The application is an attempt to reproduce a standard 2D platformer game with entirely voice controlled gameplay and interface. Super Mario Bros, the quintessential 2D platformer, was released in 1985 [1], 36 years ago. It seemed fitting to merge the timeless movement abilities of Mario with modern voice and grammar recognition technology.



The main menu of the game is very simple, with two “buttons”. These buttons don’t do anything, as the application is intended to work entirely with voice commands, but it is still important to signal to the player what is expected of them in menus. Both commands can be activated with a few alternative words to make the interface as natural as possible to use. It is undesirable for example, to have a user interface where the user must read the exact phrase to activate a command.



The above figure is a screenshot of starting gameplay. It is from here the player can move their character to the left, right or jump, all with the aim of collecting gems and reaching the end of the level.

Gestures

In main gameplay, the user can control this character with 4 basic commands. Jump, move left, move right, and stop. The player can also pause gameplay. Unlike in the menus, I felt it would be more appropriate to let players figure out how to control this character on their own. Part of the fun in gaming can be the trial and error experimentation to learn. The game is best approached with a mindset of trying to dictate actions to the character.

Hardware

The only hardware required for this application Is a microphone as well as a windows 10 PC (with US English installed). In my view, the main reason we have not seen gesture-based peripherals take the gaming world by storm, is because they are often so gimmicky and expensive. Of the top 20 selling games in 2020 [2], there is not a single game developed with gesture-based gameplay in mind, they all stick to the tried and tested handheld controller. The ubiquity of microphones in phones and laptops means that games that make use of voice controls might be the best way to reach a mass market. It is difficult enough to convince a consumer to buy your game in the first place, and much more difficult to convince them to buy an expensive and complicated peripheral, like a Microsoft Kinect or oculus rift as well.

Architecture

The voice controls in this application are implemented using the Unity/Windows GrammarRecognizer[3]. This library uses XML and the W3C "Speech Recognition Grammar Specification" to describe keywords and phrases of importance. In the context of Unity, there exists a GameObject with a script that implements this library to parse specific commands and act accordingly. So for example when a "move {direction}" command is recognized, the direction (left or right) is extracted and passed as an argument to a function which then sets the player character moving in that direction. As is good programming practice in general, the GameObjects that make up the system were kept as separate as possible. This is evident in the Player GameObject, it knows nothing about voice controls or grammar recognition, it only exposes a few public methods which other scripts can call to control the players' movement.

Conclusions

To be totally candid, this project deserved a great deal more time and attention than I gave it. I enjoy game development and I am very fond of Unity/C#. I think perhaps if I had strived to build something using more niche hardware like a VR headset, or motion capture tech I could have learned much. Hindsight is 20/20.

References

[1] https://en.wikipedia.org/wiki/Super_Mario_Bros. ; Super Mario Bros. ; Wikipedia

[2] https://venturebeat.com/2021/01/15/npd-reveals-the-best-selling-games-of-2020-in-the-u-s/ ; best-selling games of 2020 ; venturebeat.com

[3] https://docs.unity3d.com/ScriptReference/Windows.Speech.GrammarRecognizer.html; GrammarRecognizer; docs.unity3d.com