Retro Stickman Runner - Developer Diary

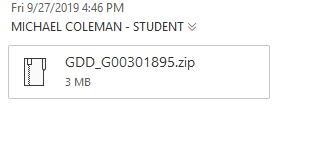
**Developer:** Michael Coleman

**GMIT Number**: G00347650

**Note:** Permission of the use of any of client’s emails has been requested and permission granted from client.

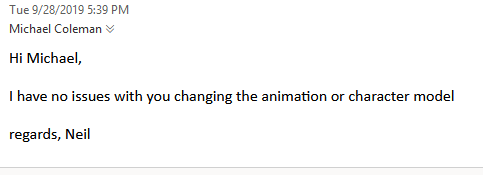
**September 27th –**

Received email from client with a folder containing assets needed for the creation of the game.



**September 28th –**

Created the project and added the supplied assets (backgrounds, music… etc) it to the repository. Attempted to get the animation for supplied character but supplied assets provided a clunky movement style. As a result, I contacted the client, who said it would be acceptable to change the character animation to provide a smoother feel to animation. I then implemented this animation into the unity game on the character. I then added the movement of the character via a script which allowed user to move using A and D keys and added gravity to the player using a rigidbody2D.



**September 29th –**

Fixed a small issue with player movement. Added the ability to jump the player when the space key is pressed. This resulted in an issue where the player would get stuck in platforms when jumping, so had to fix this issue using a physics material for the player, also improved object collision using a game object to detect when in contact with platforms. Mouse click to jump was also added to allow player option of both jumping with space and left mouse click. Changed Camera to follow player as they move, this is done via a script which takes in an object (player) and follows it. The background was also changed to move with the camera and player.

**October 1st –**

Added some scripts to the repo which had failed to upload due to a git ignore issue preventing them from uploading. Fixed an issue with jumping where the user could constantly jump if they keep pressing the jump button, so added a on the ground property to ensure that the user must be on the ground before jumping again. Also tested double jump but removed due to not being on the client’s requirements. I then changed the movement so that the user is constantly moving in the Y axis as this was listed as a requirement by the client.

**October 2nd –**

Added the ability to have platforms which will continue to spawn in front of the user, this is done via a script and a spawn point game object which spawns the platforms in. The spawn point is then moved in tandem with the player so that the platforms are spawned in ahead of the players view for immersion. This version is not fully optimised and was only a test version of the final implementation.

**October 3rd –**

Fixed a lot of issues I was having with github. When downloading and testing on another computer not all necessary game assets were being downloaded and resulted in the game not running correctly. This resulted in another change to the git ignore as well as removal of directories which should not have been uploaded to the github repository. This was a result of a mistake I made in original creation of the git ignore.

**October 4th –**

Added the destruction of the spawned platforms that were being created during the game. These are destroyed outside of the players view for immersion via a script. Similar to the spawn point for spawning in platforms, the destruction uses a destruction point, which follows the camera and destroys the platforms as it passes them. I also added a random element to the platform spawning by adding distances between the platforms, this is done in the platform spawn script. I then changed the way the spawner’s work, by switching the platform spawning to one spawner (previously implemented using multiple spawners at different X axis levels – very inefficient) which now spawns platforms at multiple heights and distances randomly for the user but in a progression where it’s possible for them to jump to e.g. platform wont spawn outside player jump ability range. I also improved on the player’s jump ability to coincide with this change by allowing them to jump with more force depending on how long they hold the jump button for.

**October 6th –**

Added a speed increase to the player as they progress through the levels. This is done by increasing the player move speed by a multiplier as the players x axis position increases. I also lowered the radius at which platform detection occurs at in order to fix an issue where the player could actually recover from hitting the side of a platform and jump again to save themselves. (mistake on my part making the radius too big on the detection game object to start with. Bug was caught during my testing of the game). I later added a death functionality where if the user hits a layer called a death area then they die. This layer moves with the player and camera so as not to create another spawner for it which would be an inefficient way of doing it. I then added a reset functionality when the user dies to reset them back to the start and let the game continue from the beginning. This is done via setting the player inactive upon collision with the death area and transforming their position to the start point when game was first started, then setting them to active again. This also required resetting the platform spawners to prevent platforms staying when the user was reset as death would cause platforms that were spawned from a previous life to remain and new platforms would spawn on top of them/around them. This was done via running through the loop of platform objects and setting them to be inactive when user death occurred.

**October 7th –**

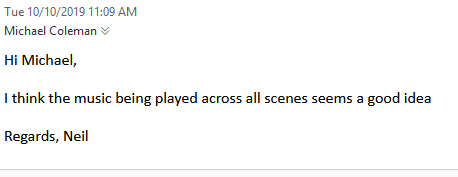
Set up a scoring system in the game which increase as the player progresses through the game. This also has a high score value which is stored in the player preferences within unity. Both of these are placed in the score manager object which calculates the scores via a score manager script. The score is reset upon player death. I also setup a basic menu scene to practice/learn how the UI side of unity works.

**October 8th –**

Added traps on platforms which when stood on by the user will cause them to die and to reset them. These were set up so that the traps spawn in different areas (either left or right of a platform). I set up the main menu layout with all the necessary buttons, backgrounds etc… for the required options. I then set up the Quit game functionality which is used to exit the game (using Application.Quit()). I also added the play game functionality which sends the user to the main game scene.

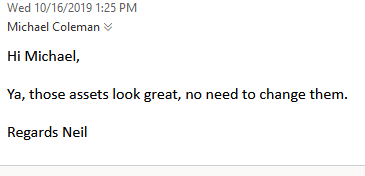
**October 9th –**

Added menu scene which contains the options for turning on/off sound effects and in game music. I also added the provided soundtrack to the main menu that was given to me. At this point I wasn’t sure whether the game music should be played across all scenes or just when the gameplay starts. So, I contacted the client and waited for a response. The music added doesn’t play across scenes as the object is destroyed upon change of scene.



**October 15th-**

I emailed the client about the mute and unmuted button pictures as the assets weren’t provided for these, so I created them myself and emailed the client, to ensure that my created assets suited them.



**October 17th –**

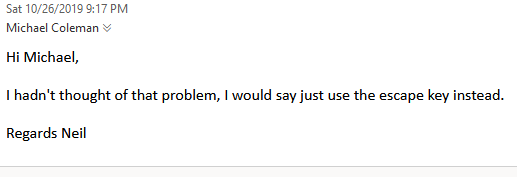
After the response from the client that the music would be best played across all scenes. I implemented this by added a script to the background music object which prevents it being destroyed upon switching of scenes. I also added the muting of music functionality which is done via using the player preferences by passing true or false when the user presses the mute/unmute button depending on the current setting. When muted the picture will change to the mute logo and vice versa.

**October 25th –**

Added death screen so that when the user dies, they are presented with a screen which gives them the restart and quit options. This is done by created the UI menu and setting it inactive while the player is alive and when they die, setting it to active. If restart is pressed reset player and set inactive again.

**October 26th –**

I implemented a pause menu which can be accessed using the escape key. The menu gives the user the option to restart, resume or quit to main menu. In the requirements, the client says that the game should have a pause button on screen, but upon implementation, I realised that the player would jump upon clicking of the pause menu which would affect the players gameplay experience. This problem even if resolved (which is possible after researching) would require the user to move the mouse to the button, which in such a fast-paced game often caused a death for the user before they can get to the pause button. So, I contacted the client over this issue of conflict. Which they said to leave out the pause button on screen. So, pause screen is just implemented using escape key. I also updated the fonts of the score/ high score on the screen.

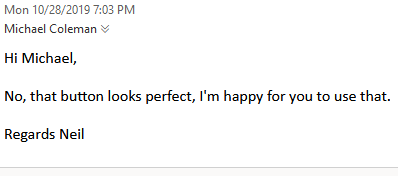


**October 27th –**

Added the provided sound effects for the jump and death functionalities. So, they play when either user dies or when the user jumps respectively. Upon adding these, I also added the ability to mute the sound effects in the options menu. I also fixed a bug with the jump sound effect. This was as a result of the power up jump functionality, resulting in the audio playing multiple times upon powering up. To resolve this, I added a delay to the playing of jump audio that was unnoticeable to the user of 0.03 seconds which prevented it.

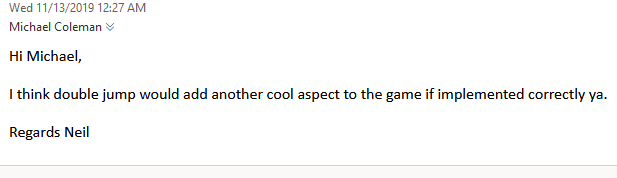
**October 28th–**

Added the help section to the main menu telling the user how to play the game. This is added by setting it active/inactive when the user presses the help button. As no asset was supplied, I created my own for the button and emailed the client to see if they were happy with it and if they weren’t to supply one which they were happy with, they replied that they were satisfied with my design.



**November 13th–**

Added some game objects as prefabs as I had forgot to do this during development. I also contacted the client about the possibility of adding a double jump, as this would add another element to the game, as well as give the user another chance to save themselves after making a mistake, the client agreed to this idea’s implementation.



**November 15th–**

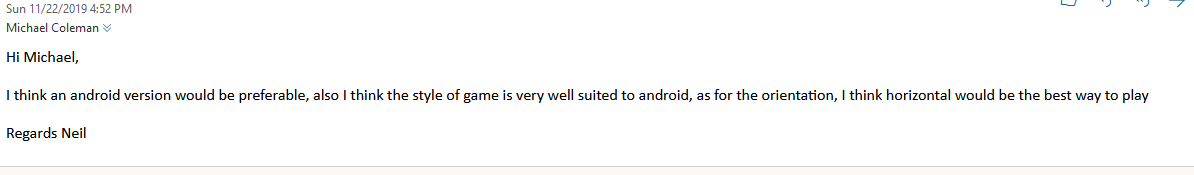
Fixed an issue with indefinite falling, this was caused by a problem where the death platform was spawning too far away and if the player hit a platform too close to the start they would fall indefinitely, this was an issue I discovered in testing. I also fixed a bug with the traps as the platforms with traps had been set to death areas by mistake, this then led to me discovering another issue with the traps, as the traps themselves had no box colliders and therefore couldn’t kill the player upon contact with them. I also fixed another issue with the death platform below the user, as they sometimes would spawn with a gap between them, again allowing the user to indefinitely fall and also fixed an issue where the death platforms were hitting the destruction point before the user got there causing the same problem. These indefinite falling issues are ones which are important to fix as upon infinitely falling the player is still moving forward and gaining score, which would allow for cheating. I also optimized the platform spawning for the user so that its easier to progress throughout the game. This was done by increasing and decreasing distances and heights between spawn points.

**November 17th–**

Cleaned some code and also fixed a time issue where somehow the time setting in the game had been set to 0 in the project settings. This caused the player to freeze in position when the game would begin. Fixed an issue where the player could create a double jump effect for themselves by walking off a platform they hadn’t jumped from and they would be allowed to jump. I then added the double jump functionality. With this came a bug where the users second double jump was only a tiny one, as the jump counter wasn’t being reset, which I then fixed.

**November 22nd–**

Fixed another issue with indefinite falling, which was caused by the death area platform spawner transform not being reset when the user restarted the game or died. This resulted in no death area being created after a user death/restart and could infinitely fall. I also contacted the client over whether they wanted the game to have an android version and if so, what layout they wished it to be in?

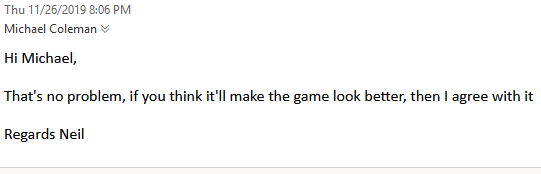


**November 23rd–**

Set up the project layout to allow change over to android, building to android caused a lot of issues with layout, as I had to change all the scene resolutions to suit android. I also had to change canvas scaling to change with the screen size to allow for the game to look good on different size devices. I also had to move some components as a result of the android build. I also had to force the camera to build the game in horizontal layout so that the user couldn’t play vertically at the client’s request. Building to android caused the issue of when the user tapped on the screen to jump the audio for jump would play multiple times, due to a fingers size if not pressed very slightly on screen, this was an issue that would cause a big problem for user experience. I removed this by adding a further delay (from 0.03f to 0.05f) on the sound audio which prevents this. The change is minuscule and is unnoticeable on the PC version. The android build, due to having a wide screen in horizontal also caused me to have to change the layout of my camera. I had to zoom in more to the game, as the player could see the platform spawning and destroying in front of them and behind them respectively on android because of the width (my phone for testing is a OnePlus6t which has a very large screen). I had to find a happy medium between the PC and android versions where the camera doesn’t look bad for both e.g. The more zoomed on android the better it looks, but the more zoomed in on PC the worse it would look.

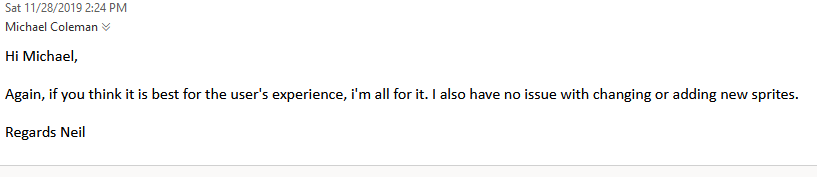
**November 26th–**

Went doing the animation for the jump, created the animation and implemented it so that when the user jumps it transitions using the Animator.setTrigger() property. But the transition didn’t look very clean, as the prolonged periods of almost constant jumping, led to a non-aesthetically pleasing view for the user. So, I contacted the client over this and asked them should I remove the jumping animation and explained the issue to them. They agreed and also said that it may make more sense as it has a space feeling to it anyway.



**November 28th–**

Fixed the orientation for android as I had it set to horizontal right, I changed this to horizontal left as I felt it is a more natural user preference to turn their device to the left for games. I then verified this by checking some games on the app store. I also fixed an issue where the music would not loop on the non-destructible object through changing the audio source variable for looping. I then implemented the random coloured platforms using GetComponent<SpriteRenderer>().color, assigning a random colour to the platforms as they were created by creating a new colour within a range of values. The random colours looked out of place in correlation with the theme of the game and as a result, I narrowed it to variable colours of purple and pink only, but again it didn’t look very user friendly. I contacted the developer over this and asked what they wanted to do, they then said that one colour platforms would be good. While contacting the developer over this issue, I also enquired about changed the sprites for traps due to them lacking symmetry with the platforms and also enquired about adding new traps styles both to which they allowed me to do.



**November 30th–**

Added the new sprites to replace the old fire sprites, also added new spike traps which work in a similar way but are placed in the middle of a platform to give the user a new type of challenge to deal with. Also fixed up some text from the help section to inform user double jump is an option.

**December 2nd–**

Cleaned up the code from testing the animator as I forgot to remove the code I had commented. I also changed the spike trap frequency and size, due to it being a very difficult obstacle to get passed and requires an almost perfectly timed jump to get passed them. I also played around with the jump and speed mechanics, fine tuning them to get a nice balance in the game. I also changed the platform spawn distances to make it a small bit easier for the player. A boost to which the speed increase was applied at was also changed, I did this due to my changes overtime which have increased the difficulty.

**December 10th–**

After a discussion with my lecturer, I followed his advice, I changed the colour of the character due to colour clashing with the background in the android build.

**Topics of Development**

Though these areas are touched on throughout the diary, I thought I’d just delve into the mentioned topics on the spec sheet to expand more on my decisions throughout the building process.

**Rate of increasing difficulty:** The rate at which the difficulty increases is due to the increased rate of speed of the character, the further the user progresses, the faster the character will get, making it hard for the user to time jumps correctly and progress. Note: I have made this game intentionally very challenging for users, as if you look at some of the most successful endless level style games, most of them are quite difficult and require a lot of skill to get far. E.g. Flappy Bird. This difficulty can get the user addicted as they want to better their previous score or their friends scores.

**Speed of Gameplay:** I originally had a very faster game speed starting off but it made it very difficult for progression, also the speed along with the jumping force meant that the user could jump over large areas of the map, leading to a very quick of a progression system. So, I slowed it down a small bit, though it is still played at a very high pace, due to the quick drop in and out nature of the game genre.

**Win/loss conditions:** There aren’t many win/loss conditions in the game other than keep progressing to gain a high score for winning conditions, losing also has a basic pretence of don’t die via falling or standing on/colliding with a trap.

**Control Placement:** Luckily the game doesn’t require any manual movement to move the character on the user’s end, so therefore no cluttering of the screen occurs via user controls i.e. directional pad on screen. The user can simply tap the screen to jump and press the back button to access the pause menu. This simplicity allows for a great user experience.

**Appropriateness of control mechanics:** The control mechanics of the game are mainly just jumping, this although sounding very simple is implemented in such a way which makes it difficult for the user, timing is the most critical part of this game and not alone are the traps detrimental for the user, the lack of ability to time jumps correctly will result in falling to their death also.

**Colour usage in UI:** For the User Interface, I kept with a purple, pink and yellow them. For the man menus I kept the colour trend to be just purple and pink, keeping the colour scheme in line with the backgrounds provided. This theme I feel further compliments the space feel/theme of the game. I kept this theme for the options menu, though making the symbols for the mute/unmute buttons white so they stand out from the background, so that the user can clearly see if their game is muted or not. For the score system in the main game, I used a yellow colour as I feel it too blends well to the spacey style, but also stands out from the background, so that the user can see their score without having to directly glance at it, taking their attention off the game, which may result in a death. I also used yellow for the pause/death menu options, as It stands out from the background so that the user can see their options clearly, also when hovering over the buttons/options it is a different colour so they can see the option they are selecting.

Critical Analysis and Conclusions

In conclusion, I felt my game turned out very well, I implemented and expanded upon the design document that was given to me by the client, I was very fortunate that my client was very responsive which allowed me to quickly implement the changes I wanted or when I had questions they were answered very quickly. I kept on top of my game consistently which kept made the development process easier. Through the making of this game, I feel I have greatly expanded my knowledge of Unity and game development and the challenges associated with it.

If I’m to be critical about my game, although very happy with the result, it surely could be improved in many areas. If we look at user customization for example, maybe the ability to add different colour character for the user to choose from would be better. Another aspect I could be critical of is perhaps the difficulty. The way I have created the game although easy in controls due to client spec, requires the user to be very skilled in their timing of jumps. One wrongly timed jump may result in death. But as I mentioned above in my diary, my rational was to look at popular games of a similar genres e.g. Flappy Bird. I suppose you could argue also maybe there’s too much of an element of luck, due to random platform generation, if say for example the game was put on the app store and a world ranking table was introduced, As the platforms having traps or not is random chance during generation. Obviously, most of these issues could be addressed with more development time for a developer, that is that none of the issues I’ve found with my game are outside of the realm of possibilities for myself or the Unity editor to achieve.

**References:**

Game object doesn’t destroy (for music across all scenes): <https://docs.unity3d.com/ScriptReference/Object.DontDestroyOnLoad.html>

Play audio: <https://docs.unity3d.com/ScriptReference/AudioSource.Play.html>

Destroy gameobject: <https://docs.unity3d.com/ScriptReference/Object.Destroy.html>

Load scene: <https://answers.unity.com/questions/44806/unity-script-to-open-a-new-scene.html>

Quit game: <https://docs.unity3d.com/ScriptReference/Application.Quit.html>

Pausing game <https://answers.unity.com/questions/1230216/a-proper-way-to-pause-a-game.html>

+ <https://docs.unity3d.com/ScriptReference/Time-timeScale.html>

Reference to font used in menus: <https://www.dafont.com/starcruiser.font>

Create a random Boolean variable: <https://gamedev.stackexchange.com/questions/110332/is-there-a-random-command-for-boolean-variables-in-unity-c>

Set the high score in player preferences: <https://unity3d.com/de/learn/tutorials/topics/scripting/high-score-playerprefs>

Find a game object: <https://docs.unity3d.com/ScriptReference/GameObject.Find.html>

Change camera zoom: <https://docs.unity3d.com/ScriptReference/Camera.html>

+ <https://docs.unity3d.com/ScriptReference/Screen.html>

Death Area collision detection:

<https://gamedev.stackexchange.com/questions/82119/simple-collision-detection-in-unity-2d>

Get high score from player preferences: <https://unity3d.com/de/learn/tutorials/topics/scripting/high-score-playerprefs>

Increase Score per frame: <https://answers.unity.com/questions/1357802/increase-score-per-frame.html>

Random colour for platforms: <https://forum.unity.com/threads/c-instantiate-prefab-with-random-color.268719/>

Animator using triggers: <https://docs.unity3d.com/ScriptReference/Animator.SetTrigger.html>