# Defeat them All Developer diary

**Timeline:**

**October 2018:**Began working on core gameplay mechanics such as player controls, weapon systems and enemy behaviour.

**October week 1:**I began by using the simple shape sprites generated in unity as the player and enemies. I set up multiple spawn points in the game and the had the enemies spawn in at an interval collision detection was needed for the game to progress. I chose to do wave spawning because the inspiration in the design document taken from “EverWing” which has waves spawning in intervals and gets progressively more difficult as the game progresses.

**October week 3:**

I created enemy prefab with falling behaviour attached to the objects along with rigid body a circle collider and a score value attached to them which I will use later. I also created a player prefab. While doing this I added a simple background objects to set the scene which I will change later to suit the theme of the game. In order to spawn the enemies from the prefabs I created a parent utilities class which controlled all of the possible spawns for now it will just be enemies, later it will spawn coins and tokens.

**October week 4:**

This week I had to recreate the repository because of a github desktop error corrupting the files. I implemented a sound controller object to the game scene but did not select any sound effects yet. I implemented and tested bullet collision with enemy objects and fixed some bugs with the spawning. I conferred with my customer and decided that it was best to build the game to a mobile device. I used android because of unity’s in-built build and run functions and the .NET4 library to generate random numbers in a range which I will use later to spawn collectables from the enemies.

Once I had collision detection working, I implemented wave spawning for the enemies the enemies in a row and fall in increasing speed based on time. I decided to base the progression on time instead of the score because of the competitive aspect of the game with one’s self. If the game were to progress by score it wouldn’t incentivize the player to earn more points while the game was in the early and easier portion of gameplay.

**November 2018:**

I implemented touch and key controllers for the player. I chose to implement key controller for testing on a machine rather than device in order to save time and the key controls would not effect on the gameplay on a mobile device. I added menu and GUI to the game along with game ending condition to the gameplay. I also added event handlers along with utility classes to manage scene selection through menus.

**November Week 1:**

Using the sound controller, I added sound effects to the player when firing a bullet and colliding with an enemy, I also attempted to fix the spawn interval issue with the wave spawning. The enemies were falling at a constant speed without increasing although the waves were becoming more frequent. One problem encountered was the fire rate of the bullets shooting at an exponential rate due to the invoke repeating method being dependent on touch count. I created two different Shoot methods in a weapon controller attached to the player object, 1 for using key and 1 for using touch. Once I had added different scenes to the game, I used a utilities class to set the scene names using names as apposed to index which could cause issues.

Using a game controller, I handled the events in the gameplay e.g. when an enemy or player is destroyed. The game over condition is determined in the player behaviour script when collision is detected it notifies the game controller to set the timescale to 0 and loads the game over panel the quit button in the pause menu also ends the game but the collectables earned will not be added to the total to incentive the player to finish play throughs. I chose to do this because of the relatively short time it takes to do 1 round of gameplay. A problem I encountered was when the play again was clicked the game was frozen this due to the timescale not being reset so in the start method in the game controller it is set 1

**November Week 2:**

Through the game controller I tracked the player score which I will track later using play prefs in unity. I found a fix to the touch fire rate using a Boolean to track whether the player was shooting then using an invoke repeating method it would fire every 0.2 seconds. I had run into issues with menu buttons not being interactable, I had to recreate the menus.

The navigation through the menus was done by loading different scenes by name. I created scenes for each, but the character selection had multiple tabs to do this I multiple game objects on the scene with a controller, when a tab is clicked the respective game object and child elements are set to active and all other game objects and child objects contained within each tab are deactivated. I asked my customer how he would like each of the tabs laid out as he had only specified one, he gave me free reign over the design of these tabs.

Using the game controller, I set the text fields to display the current score to the player along with coins. I done this by serializing the fields in which score is recorded and updated using ToString methods in the update method to have constant update to the screen.

**November Week 3:**

The game loads in the splash screen first in which there is an audio controller which controls the background music for the game this game object is set to not destroy on load meaning the music is playing the background constantly. While creating the menus and options scene I added a mute toggle button which sets the volume to either 1 or 0 muting the entire game or allowing sound effects and background music. I conferred with my customer and presented my choice for the background music “You say run” by Friedrich Habetler and he approved.

I created a coin through paint.net and added it to the sprites and created a prefab after adding rigid body and a collider. Through the game controller event handlers, I set it so that when an enemy is destroyed by a bullet a coin is set to spawn. A problem I encountered with this was that the bullets would collide with the coins. I tried several solutions even looking into the physic engine in unity to ignore collisions with certain object with tags but to no avail. Later looking into OnTriggerEnter I realised the could be used to only detect certain collisions rather than ignoring them, using the tags I had already set up it was a simple fix and improved the overall gameplay greatly.

**November Week 4:**

The wave spawning, I had set up was not working correctly, after speaking with my fellow students I had realised that I was tracking the acceleration in the falling behaviour script attached to the enemy object so when each on was spawned a new script would be instantiated and the starting value would be at it initial value. Using the game controller object I set it to update the speed using time.deltaTime and an acceleration value I had set the set the enemy prefabs to take the speed from this method in the game controller.

After asking my customer how he would like the game to get progressively difficult he decided to only increase the speed at which the enemies fall and the interval at which they spawn rather than increasing their health.

In the enemy script I added a health value and along with damage to take when a collision with a bullet is detected. After play testing this I determined an appropriate value for the health and damage.

**December:**

I implemented the currency, upgrade, character select systems and altered the images and sprites to better suit the theme of the game

**December week 1:**

Using a random range set in the enemy script I set tokens to spawn in the same way I had set up the coins, but the token would only have a probability of spawning of 1 in 5 for each enemy. Because the design document had little mention of tokens, I asked my customer what their function was he told me that the tokens fill a meter and once the meter is filled the rate of fire of the player increases.

Because of how I had implemented the shoot method in the weapon controller script I had issues. The issue was that that interval at which an invoke repeating method repeats cannot be changed while it’s repeating. I had to revise the implementation. Instead I had the player shooting constantly by calling the invoke method in the start method. Once the meter had been filled the invoke would be cancelled, the firing rate would be increased then invoked again. Using an IEnumerator method it would wait for 10 seconds real time cancel the invoke reset the firing rate and invoke the shoot method again. There is a latency of 0.2 seconds between each invoke but this isn’t noticeable because the firing rate is once every 0.5 seconds.

Once I had the core gameplay majorly completed, I took 8bit sprites from the internet and used them as the players and enemies to suit the theme of the game.

Using PlayerPrefs I tracked the coins earned by the player when the game is ended the coins earned is added to a variable and then this is added to the current balance.

Because PlayerPrefs doesn’t contain a Boolean I used an int value of 0 and 1 to control whether the player had paid for upgrades and additional characters. The for the upgrades the could only be purchased once and the damage taken by the enemy would be increased for each grade.

For purchasing character I followed a similar approach I set up if statement to check if the player had paid for the character if they hadn’t the cost would be deducted providing the had enough coins, if it had been purchased the value 1 would be assigned to the paid value in the player allowing them to select different characters without paying again.

In the game I added image objects to the canvas which I had drawn and divided into 5 parts. These images represent the progress bar for the increased firing rate. For each token collected a temporary value in the game controller is increased this bar is filled depending on however many tokens out of 5 the player has, once 5 is reached the meter is reset and the rate of fire increases.

**Design decisions/Rational:**

**Gameplay:**After conferring with my client I determined that the game would be best suited towards a mobile device, so I built it for android. Because of the quick reflexes and controls needed to progress far into the gameplay touch controls were needed.

As I previously mentioned determining the difficulty progression by time rather than score was to encourage the player to play again to get points while the level was in the early stages.

By having the enemies spawning in waves it increases the difficulty, but the enemies can be moved by bullets allowing the player to create hole in the waves. The wave eventually become very fast and the screen becomes somewhat chaotic to progress further the player will have to collect tokens as they fall to keep up with the speed of the enemies.

I had the tokens fall slower than the coins due to their immediate importance to the player by changing the gravity scale.

After speaking with my customer and determining how many characters could be selected and what each one could do, I took some other well known 8bit sprites that were suited to the theme and used them. For the default character the movement speed and lives were standard, the second unlockable character would have 3 lives and the third would have high movement speed.

After researching PlayerPrefs I discovered that I could permanently store data to the local device allowing for a high score, currency, upgrade and stat tracking system. Using PlayerPrefs I set the high score if the new score was higher by comparing using the GetInt method and setting them using the SetInt method.

I did something similar with the stat tracking system but when a game was ended by collision and not quit the variables in stat controller would be increased by the values earned in the run then the set method was called to set the new values to the sum of the previous total and values from the new run.

In the project settings under edit->player in the android tab I locked the screen orientation to portrait to avoid needless flipping and interruption of gameplay.

I chose the song “You say Run” because of the fast and catchy nature of the song and it is suited to many things

**Menus/GUI:**

There weren’t many decisions to be made in the GUI other than the sprites to use and the layout of the character select along with its tabs. I tried to keep the general shapes and positions of the buttons and displays as specified on the design document while adding some extras like displaying high score on the main page.

After being given permission to design the character selection tabs myself I decided to keep the layout roughly the same for each tab to ensure smooth transitions when the game objects are deactivated and activated.

For the splash screen I found a text converter online where you can create png in the classic pokemon font. I used this as the title images for each screen.

In the splash screen a sound controller is started and never destroyed and set to loop over and over the user can turn of the volume in the options menu by toggling the mute button.

In players settings under edit I found I can add an icon for the game by using a sprite within the game, so I added one of the more well-known pokemon sprites (Lugia).

I chose to have the first upgrade to increase the dame by 2, the second to increase the damage by 3 and the third to increase by 5 so when all are purchased it only takes 2 shots to destroy an enemy.

**Theme/style:**

For this game came across an old background image from a pokemon game and decided to use soft blue colours for buttons and display as they make the player and enemy sprites standout in comparison.

**Known Bugs / issues:**

There is one bug in the stat tracker when the game is shutdown and rebooted the play prefs for the total stats are reset to 0. This does affect the achievements, but the functionality of the achievements does work otherwise.

Although the game compiles and runs perfectly the classes aren’t encapsulated properly as with the gameController class being a example.