

Mobile Games Programming

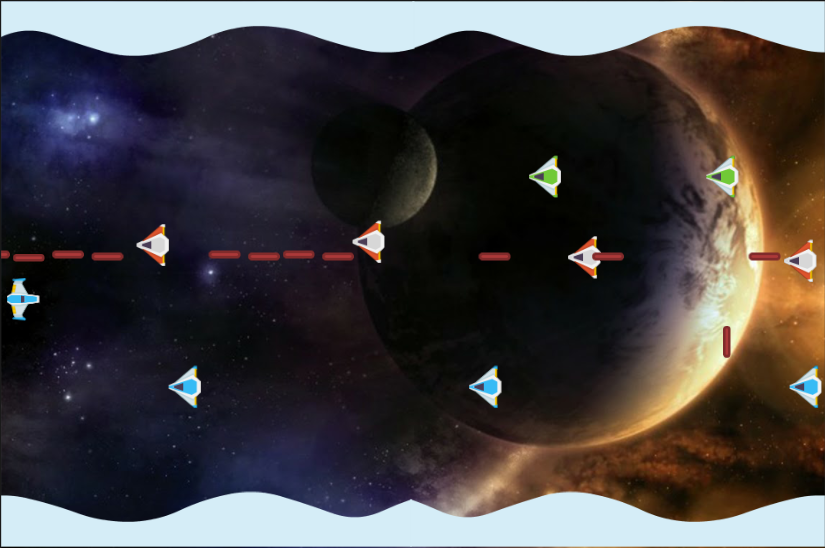
Shmup

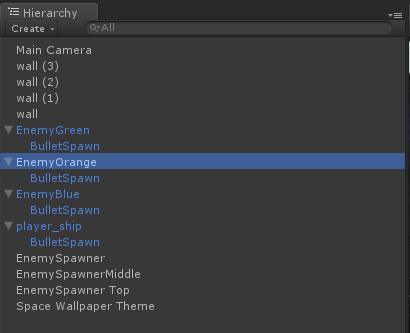


December 4, 2015

Steven Reynolds

In this assignment, you will find the basic barebones of a Mobile Shmup game that should have been a completely finished game but due to Unity being buggy in P225, Unity crashed and my Scripts all became NULL and the key code “¬¬¬” and all of my gamePrefabs attached to codes were destroyed. No screenshots or code snippets were recorded before my Unity crashed but after many attempts by students around me to revive the project, I restarted the project and got as far as possible to create the skeleton of a basic mobile game.

The game itself included Input and worked perfectly with the Unity Remote 4 controller which was hooked up to an iPad device, on the left hand side of the game there was a slider that allowed the user to move the character up or down and then the right side was the for the user to press down with their finger to start shooting. The reason I chose to do this in this way is because it was easier for the player to add another finger to the screen to drop a bomb, then having to press The reason I chose to do it this way in comparison to getting the user to tap to shoot is because I wanted the user

to have two fingers free to activate a Nuke which cleared the game screen. This, cleared all bullets and NPC’s on screen and had a 0.5 second delay before they could start spawning again so the user could regain their pace within the game. Not only this, but the game did include a Health and Score system in which the player initially started with 3 lives, and whenever they were hit by the NPC or one of their bullets they would lose a life. (There was plans to include a shield system in which a power up would give you 5-10 seconds of invincibility allowing you to clear the game screen quickly.) The user would level up by killing the Enemy AI’s, you would get 50 points for killing a Single Line Shooter “Straight Line AI”, 100 points for killing a ZigZag Shooter, and 150 points for killing a kamikaze pilot that would when spawned into the game they would lock on to the targets position and then dive towards the character.

In the little time left, you can see that the game contains 4 walls which have collision attached to them which stops the Ai and the Player going out of bounds, and we also have a Player\_Ship with a bulletSpawn on it. The bulletSpawn is an empty game object that is attached to the front of the ship and this is where the bullets will be firing from. The game also used a function called “OnBecameInvisible” which enables us to destroy the object when the renderer is not visible by the main camera,

Figure Hierarchy of the Unity Project

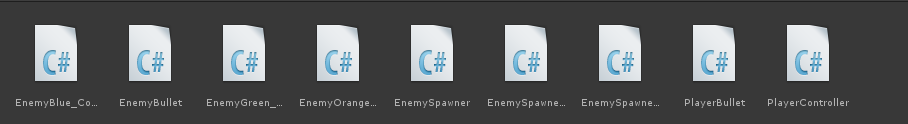
The game also contains 3 unique enemy spawns which have bullet spawners attached to them also, but this allows for the three enemy types (Although not obvious at first, they do have slightly different code) to spawn into the game either firing, or going straight towards the player in a horizontal motion. There is also code to randomize the EnemySpawner Middle to fluctuate up and down on the enemy spawn point so it randomly generates however, this became very buggy when the spawners decided to fly off when the game was being run.

Figure Assets\Scripts made in the project

Above are the scripts that were re-written in the time left which contain 3 individual NPC classes, Player Controller and Player Bullet Scripts and finally an NPC spawner that should randomly generate where the enemy ship should spawn and then follow it’s path, However this became very buggy because the Spawner was going off the map and flooding the world with ships so I used 3 different spawners located at different points on the map and allowed the Enemy ship to flow freely through the playable level. On the other hand, when this was enabled the blue enemy ship and the green enemy ship decided to start shooting downwards and the Orange ship continuously shot in front creating a massive bullet train, again when this was turned to face one direction the OnBecomeInvisible function ceased to work and this is something that needs to be debugged to fully understand why this does not work.

To play the game, just load the project – Use W and S to move up and down (If you’re playing on Keyboard) and on the Mobile Device (The code is included in PlayerController and should work) use the left hand side of the screen to move your character up and down, and the right hand side to shoot, there are no bombs.

In conclusion to the events of my project, I believe I would have created a nice Mobile Game to go onto my portfolio (As this is where I want to go to in the future) but I will be working on this to get it back up to scratch again. I have learnt a lot from the tutorial sessions that Pete runs on a Wednesday and this is how I have exceeded my own skills in making this mobile game, I just wish I could of given you a more professional project to mark on and I hope you take into consideration the problems I have had and the time I had to basically rebuild a game.

**References**

Unity. (2015). *Unity3D Documentation.* Available: http://docs.unity3d.com/Manual/index.html. Last accessed 4/12/15.

BroTechGames. (2015). *Unity3D tutorial - Random Enemy spawning.* Available: https://www.youtube.com/watch?v=-s6jobwtuRk. Last accessed 3/12/2015.