Milestone 3 Report: 1/1/21 – 8/1/21

# Goals:

|  |  |  |
| --- | --- | --- |
| Goal | Description | Done |
| Refactoring: PlayerAttack | Over the last milestone my coding practices shifted as I moved from each script to the next. In this sprint, I must formalise my style so that each script follows the same style rules and logics to make them all equally readable and understood. Each script is also to be treated as its own object, and is as decoupled as much as it can be. | Done |
| Refactoring: PlayerBlock | Over the last milestone my coding practices shifted as I moved from each script to the next. In this sprint, I must formalise my style so that each script follows the same style rules and logics to make them all equally readable and understood. Each script is also to be treated as its own object, and is as decoupled as much as it can be. | Done |
| Refactoring: PlayerDash | Over the last milestone my coding practices shifted as I moved from each script to the next. In this sprint, I must formalise my style so that each script follows the same style rules and logics to make them all equally readable and understood. Each script is also to be treated as its own object, and is as decoupled as much as it can be. | Done |
| Refactoring:  PlayerController | Over the last milestone my coding practices shifted as I moved from each script to the next. In this sprint, I must formalise my style so that each script follows the same style rules and logics to make them all equally readable and understood. Each script is also to be treated as its own object, and is as decoupled as much as it can be. | Done |
| Refactoring:  PlayerAnimations | Over the last milestone my coding practices shifted as I moved from each script to the next. In this sprint, I must formalise my style so that each script follows the same style rules and logics to make them all equally readable and understood. Each script is also to be treated as its own object, and is as decoupled as much as it can be. | Done |
| Refactoring:  PlayerMovement | Over the last milestone my coding practices shifted as I moved from each script to the next. In this sprint, I must formalise my style so that each script follows the same style rules and logics to make them all equally readable and understood. Each script is also to be treated as its own object, and is as decoupled as much as it can be. | Done |
| Added Goals |  |  |
| Implement parrying | When making the previous milestone goals I had missed that I still needed to properly implement a parrying system.  When the player presses the block button within a tweakable amount of time before getting hit, the player will “parry” the attack. This means that the player will not take damage, not get knocked back, and will play a parrying animation. | Done |

Discussion:

Code was tidied up during this milestone and a key feature of the players combat-capabilities was added. One last feature must be achieved before the player can be considered “complete” – attacks deal damage – but that will be dealt with next milestone. Now the player is mostly done, work can begin on the Hunter.

Milestone 4: 9/1/21 – 22/1/21

This milestone will be a bit longer to accommodate for the extra work that I want to get done within this timeframe.

Now that the initial Hunter meeting has occurred, work can properly begin on implementing them.

|  |  |  |
| --- | --- | --- |
| Goal | Description | To Deliver |
| Player can deal damage with their attack | The players attack should deal a single instance of damage to enemies. | Player Attack Object will have a script that gives its attack value. |
| Hunter can take damage | **Important, this will determine how all enemies take damage**: The Hunter should have health which can be depleted by taking hits from the player’s attack. The Hunter will take 1 damage per hit.  On hit, Hunter will flash and be knocked backwards. | Hunter will have a script that contains all “getting hurt” information and programming. |
| Hunter – Idle behaviour | When not attacking, the Hunter will have an Idle state. During this state the hunter will play the Idle animation.  The Hunter will not be moving during this time.  The Hunter can turn to face the player during this time. | Hunter’s Idle state will work as intended. |
| Hunter – Power Swipe behaviour | At a certain point in the animation the Hunter will instantiate a Power Swipe Prefab, facing the same way the Hunter is.  The Hunter will slide forwards during this attack.  When doing this attack, the Hunter will not be able to turn to face the player if the player gets behind them. | Hunter’s Power Swipe state will work as intended. |
| Hunter – Power Swipe Animation | The Hunter will play an animation with three parts: tell, attack, end-of-attack. These phases will last for a tweakable amount of time. | Hunter’s Power Swipe animation will play as intended. |
| Hunter – Power Swipe Prefab | The Power Swipe Prefab will appear a set distance from the Hunter. It will travel horizontally upon creation in the appropriate direction. It will start moving slow and gradually gain speed. | Hunter’s Power Swipe Attack Prefab will behave as intended. |
| Meeting:  Discuss art direction and design for the Hunter enemy | On Wednesday we will have another meeting to discuss the Hunter’s production. | Meeting will occur. |
| Art assets implemented | All art assets that have been made for the Player are in the game, and those that map to functions worked on in this sprint have been applied. | Art assets will be implemented as they are made for the game. |