Practical Gaming 2022

# Name of Student **Edvinas Tkacenka**

# T Number **T00216714**

# Name of Project **ETPG**

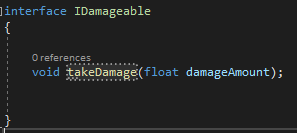
# Gameplay

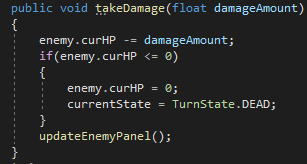
Walk in front of any of the 2 enemies in the start zone to initiate a battle. Use left mouse click to walk around and left mouse click to select abilities/targets.

# Coding

Under each of the following headings, please describe the concept, why is it or isn’t it useful/needed, where do you implement in you project, you may provide screenshots or cut and past code segments etc..

* Frame Rate Independence
  + Time.deltaTime was only used to update the progress bar. Once the progress bar was updated the enemy or the player could choose an action and a target.
  + It is useful because if the game loses frames, the enemy wouldn’t be at an advantage when attacking. Both the player and enemy are on the same playing field.
  + A screenshot of a computer

    Description automatically generated with medium confidence
* Interfaces
  + I had an interface for doing damage to each character in a fight.
  + 



* + I did use an IEnumerator for when the player or enemy attack each other. It moves character A to B, animates, does damage, removes them from the attacking list and sets their state back to Waiting.

Text

Description automatically generated

* Inheritance
  + I used inheritance for the enemy and player classes. I also used it for all the attacks they use.
  + I used inheritance as it was much easier and simpler for the characters to have a base class that included all their health, mana, attack, defense and inherit from that and change the values respective of the character. Same goes for the attacks.
  + Text

    Description automatically generated

Text

Description automatically generated

* + Text

    Description automatically generated
  + Text

    Description automatically generated
* Case pattern
  + Cases were most predominantly used in my statemachines. It was pretty much a requirement to use them as a turn based game relies heavily upon them. The cases were used to update the progress bar, choose an action, start the action, etc.
  + They were also used by the manager to see who’s turn it was and to make sure that the turns go in order of first come first serve.
  + Text

    Description automatically generatedText

    Description automatically generated

Text

Description automatically generated

You get the idea. 80% of the game revolves around these statemachines.

* Observer Pattern
  + There are three main lists that are used to store actions, how many players there are in battle and enemies. This is useful so each character in battle takes their turns in order, and they can select a character individually.
  + A screenshot of a computer

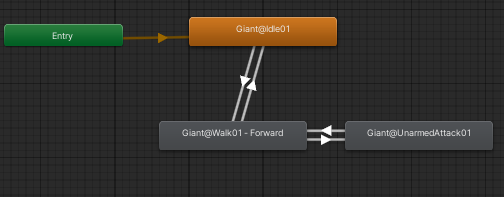
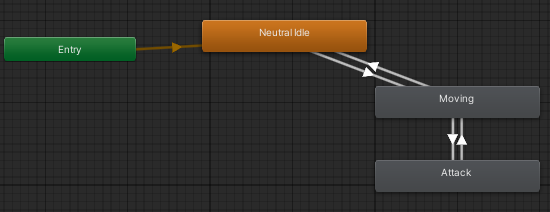
    Description automatically generated with medium confidence
  + 
* Polymorphism
  + I did not use polymorphism as I relied mostly on inheritance
* Communication between scripts/game objects
  + Communication between scripts and objects is vital here, mostly for getting scripts from other game objects so I can their methods.
  + 





* + 
* Instantiation and Prefabs
  + There’s Instantiation for select buttons for every enemy in battle,
  + 

Other than that, there’s not much for instantiation.

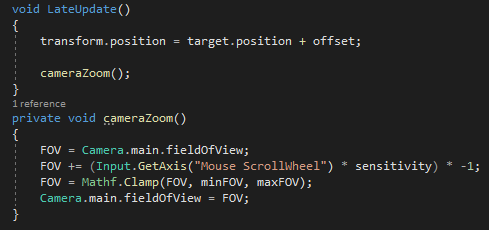
* Magic Numbers
  + There’s only one place I can find a magic number in my code that is sort of necessary in getting the enemy position so the player can play an animation in front of them. This could be done better.
  + 
* Model Animation
  + Importing unity model and animations is something I never want to do again. My player model inverted itself 3 weeks after I first imported it and I still don’t know why to this day. I have used models and animations from Mixamo and Kevin Iglesias.
  + 
  + 
* Self made models and or animations
  + The model that I made is a snake and it has one animation where it does a moving/idle animation.
* Interactions between objects/scripts
  + There’s a few examples of interactions between objects and scripts
  + Text

    Description automatically generated
  + Text

    Description automatically generated
  + Text

    Description automatically generated
* Proper code placement
  + My Player script contains an animate and move\_forward method that can only be accessed by itself and its game object.



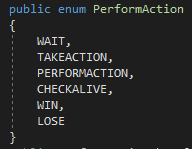
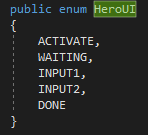
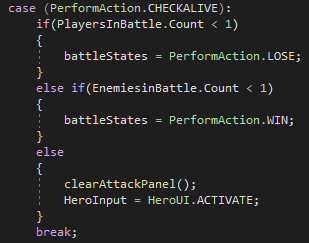
* + Camera zoom is another example of this 

* Code repetition
  + There are a few instances of code repetition, mostly between the HeroStateMachine and EnemyStateMachine.
  + Text

    Description automatically generated
  + 
  + TimeForAction is used in both scripts
  + Text

    Description automatically generated
* Feature 1
  + GUI is very important for a turn based game and it was one the more difficult subjects to learn. The code itself is simple enough, however adding it and using it in Unity is very fiddly.
  + Text

    Description automatically generated
  + Text

    Description automatically generated
* Feature 2
  + Enums were an essential part of my game. They were mainly used to control the GUI and to make sure the game goes to a specific state once the conditions were reached. For example CHECKALIVE checks if there is less than 1 player or less than 1 enemy and switches the state to WIN or LOSE depending on the situation.
  + 
  + 
  + 
* Feature 3