**Pattern Report**

Joseph O’Keeffe – C00258019

**Intro**

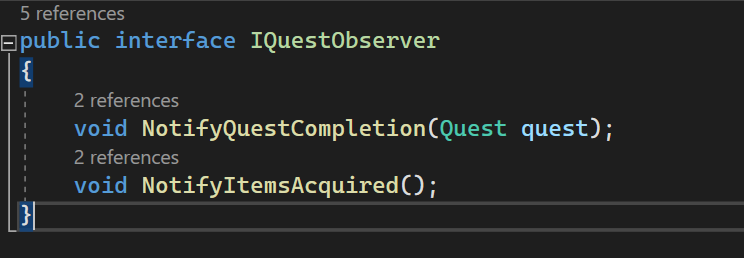
The patterns I used during my project were:

* **The observer pattern**
* **The singleton pattern**

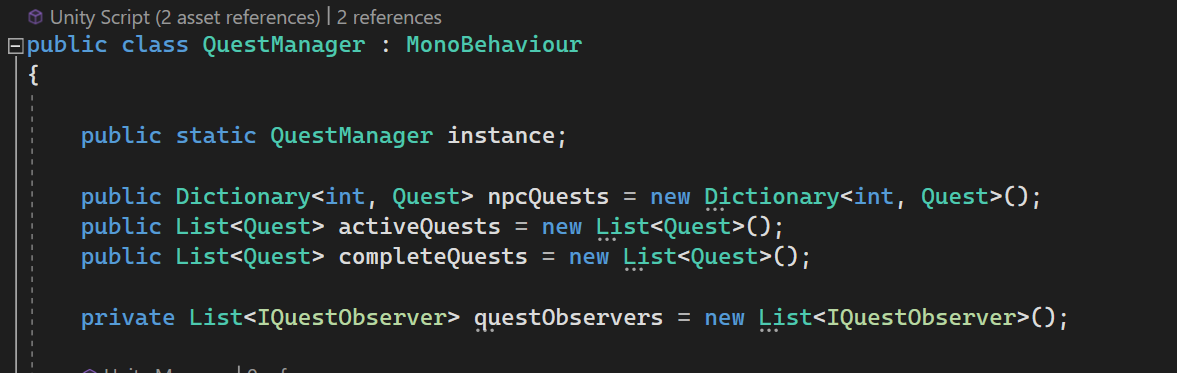
I used the singleton pattern quite a few times during this project as its always useful to be able to access classes that only need to exist once. Then I used the observer pattern to notify me when I complete a quest and when I have an active quest and have all the items I need to complete it in my inventory.

**The observer pattern**

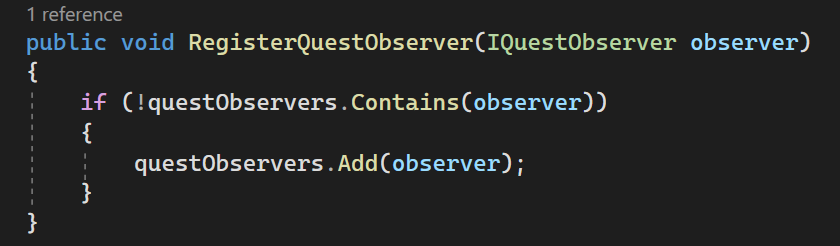
This is the interface for the pattern. It has two functions. One takes a quest and notifies you when you complete a quest. The other one notifies you when you collect all the items you need for your current quest



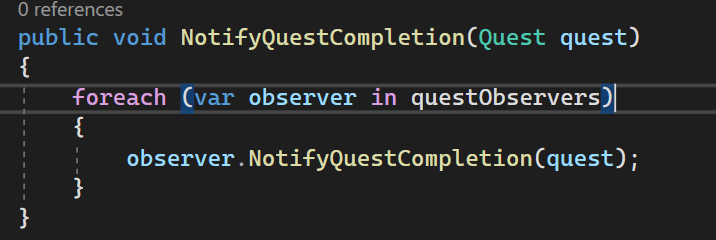
Then in my Quest Manager class, I’m making a list of quest observers



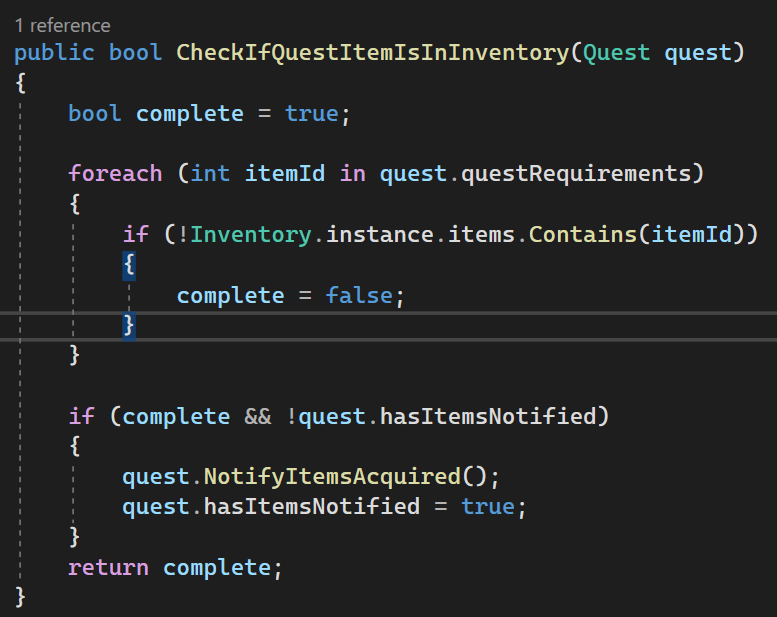
This adds an observer to the questObservers list if it doesn’t already exist



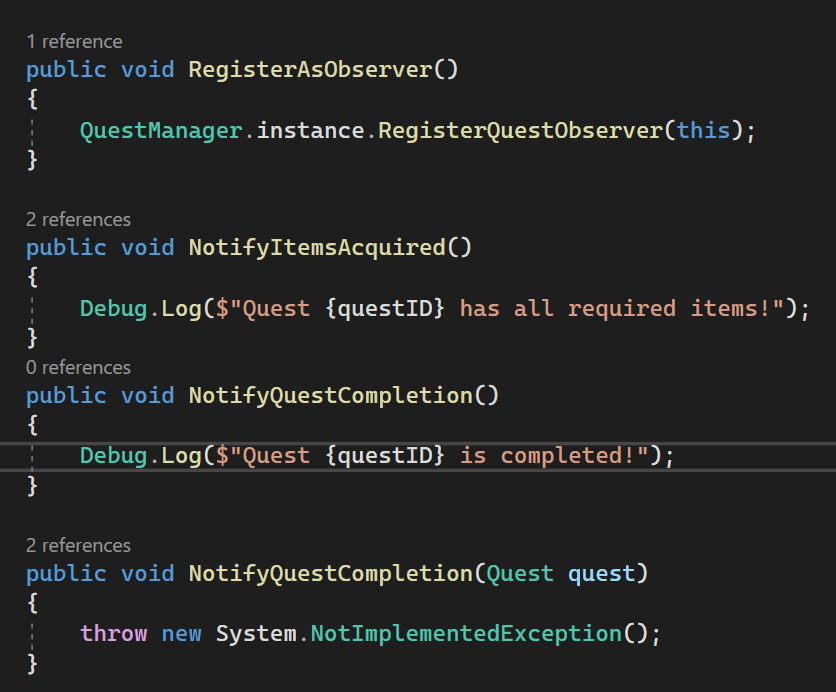
This is the function that checks if the quest is complete and then calls the notify function



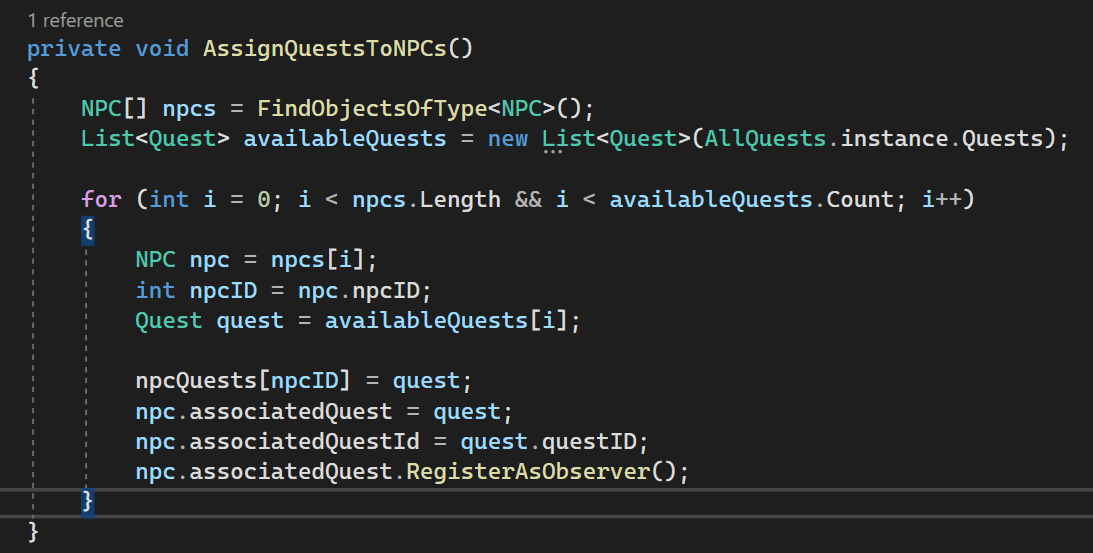
This is where I am calling the function once you have all the items for a certain quest and then it notifies you

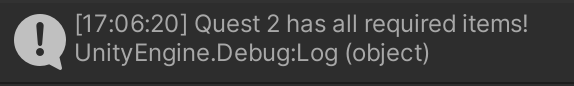


This is just where the functions are being defined and the logic of when they get called



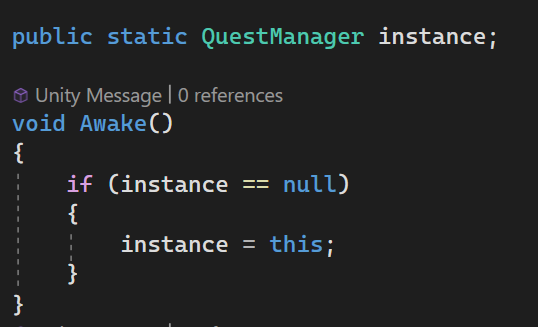
When you get a quest, that’s when you register the observer to a quest



This is what it looks like in the game when you have all the items  


**The singleton pattern**

This is how you make the singleton pattern. Make an instance of the class and make sure it’s the only instance that can exist. It is much simpler than the observer pattern. What makes this so useful is that you can call this class from anywhere



This is is how you use the singleton in a different class, by using the class name followed by instance and then the function or variable you want to use.

