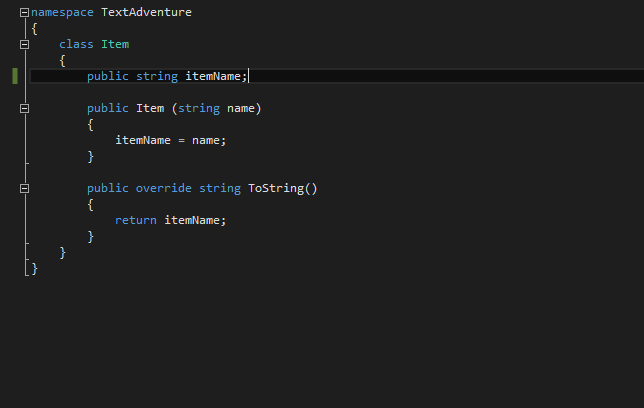
Research and Code Documentation

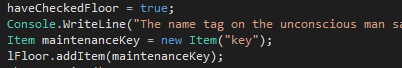
# Item Class

To create the item class, I used what is known as a constructor, specifically and Instance Constructor. This is essentially an empty class that provides the barebones tools to create an instance of the class in other parts of the code, allowing any variables to be handled outside the class code block (akin to instantiating a game object in Unity). Given that a game like this can have many items, it stands to reason that it can have many instances of the Item class.



*Above: A screenshot of the Item class, in its entirety*

The entire item class is made up of these two blocks. Because the instance of the item, and all code handling that instance is declared in the Game class, the Item class can get away with having a single variable (“itemName”). This variable is assigned to during instance declaration, using the parameter “name” to parse the name of the item into the itemName variable.



*Above: An example of Item declaration in the Game class, showing the use of a parameter to parse in a string*

The Item class also contains an example of an override. This is a way of modifying the behaviour of an existing method. In this example, when Item.ToString() is called, the program returns the name of the instance, rather than the class name (e.g. “key” rather than “TextAdventure.Item”).

# References

<http://www.yoda.arachsys.com/csharp/constructors.html>

<https://msdn.microsoft.com/en-GB/library/0f66670z.aspx>

<https://www.codeproject.com/articles/18734/method-overriding-in-c>

<http://stackoverflow.com/questions/4718725/best-practice-to-avoid-invalidoperationexception-collection-was-modified>