Best Coding Practices.

Please Keep Close by while Coding ☺

To ensure that your code is easy to understand and that it does not generate countless headaches when trying to modify it, there are a few good practices that you can start applying as your begin with coding; these should save you some time along the line.

Variable naming

- Use meaningful names that you can understand, especially after leaving your code for two weeks.
- Capitalize all words in the name, except for the first word.

Functions

- Use unique (i.e., different) names (for global and local variables).
- Check that all opening brackets have a corresponding closing bracket.
- Indent your code.
- Comment your code as much as possible to explain how it works.
- Use the **Start** function if something just needs to be done once at the start of the game.
- If something needs to be done repeatedly, then the function **Update** might be a better option.

Linking Your Script

- Sometimes, although the syntax of your code is correct and does not yield any error in the **Console** window, it looks like nothing is happening; in other words, it looks like the code, and especially the functions that you have created do not work. This is bound to happen as you create your first scripts. It can be quite frustrating (and I have been there :-)) because, in this case, Unity will not let us know where the error is. However, there is a succession of checks that you can perform to ensure that this does not happen; so you could check the following:
- The script that you have written has been saved.
- The script has no errors.
- The script is attached to an object.
- If the script is indeed attached to an object and you are using a built-in function that depends on the type of object it is attached to, make sure that the script is linked to the correct object. For example, if your script is using the built-in function **OnControllerColliderHit**, which is used to detect collision between the **FPSController** and other objects, but you don't drag and drop the script on the **FPSController** object, the

script, while being error-free, will not be used, and the function **OnControllerColliderHit** will not be called if you collide with an object.

- If the script is indeed attached to the right object and is using a built function such as **Start**, or **Update**, make sure that these functions are spelt properly (i.e., exact spelling and case). For example for the function **Update**, what happens here is that the system will call the function **Update** every frame, and no other function. So if you write a function spelt **update**, the system will look for the **Update** function, and since it has not been defined (or overwritten), nothing will happen, unless you specifically call this function. The same would happen for the function **Start**. In both cases, the system will assume that you have created two new functions **update** and **start**.