Obdulio Benitez Garcia GDEV-267-01

Space SHMUP Questions

1) The Hero script has a public method named “shieldLevel”. This acts as a get and setter for the shield level property. In it’s set code block it has a Mathf.Min(value, 4) which sets any value over 4 to 4. This would have to be changed to 6 if the maximum shield strength would want to be increased to 6.

2) The shields use a sprite texture that currently only holds 4 levels of shield and a sprite for no shield. This would now have to include 2 more sprites if we were to increase the shield level to 6 instead of 4 as it currently is.

3) Quaternion is how we rotate the ship in place to give the impression that the ship is boosting around space.

4) We get to work on smaller batches of the project at a time. We can stop before moving on if we encounter any problems and work within that block without having to go down the rabbit hole if we were to do it all at once.

5) The boundscheck script is run in the LateUpdate() method so it is one of the last blocks of code to be read after everything else is done moving. Two important bools are KeepOnScreen and isOnScreen. Based on the keeponscreen bool determines what happens to the object when near or out of bounds of the camera. The onscreen checks the position of the object and compares it to the camera edges, if the position is out of the boundary of the camera then this bool is changed to false.

6) isOnScreen = !(offRight || offLeft || offTop || offBottom) works is by checking each bool. || means or. Each bool is set to false by default so having them all in the parentheses it is checking if any of those bools turn true. Because of the or comparison, the first instance any of those variables turn true satisfies the condition. That true then becomes a false because of the ! exclamation point as this returns the opposite. So then isOnScreen is turned to false.

7) transform.root and then from there get the root gameobject by transform.gameObject. The first part gives you the transform of the root gameobject and then from that you get thew gameobject itself.

8) Property are the getters and setters while method are functions that either return something or perform an action.

9) private variables will have an underscore in front of the name.

10) so the gameobject is not moved by forces or gravity. We set kinematic to true so we want the object to be moved by code. If we add force to the object, it would want to roll over as a ball would. We don’t want that to happen so we freeze the rotations so it doesn’t roll when moving. Freezing the z axis is because we are making a 2d game in 3d space. We don’t want the third axis in a 2d game.

11) One way to do this by first defining the list and then using Random.Range to randomly pick a number between 0 and the length of the list. That number can then be used as the list index to pick a randomly assigned index from a list.

12) MonoBehavior is a base class. When we create a new script containing : MonoBehavior, it means that the script is deriving from that class. It inherits methods from that class. Another name to describe it would be GameObject.

13) No, not everything needs to extend MonoBehavior. Things like abstract classes or something that does not have to exist within the game space do not need to extend MonoBehavior.

14) I would classify these objects as tightly coupled as they need references to each other. For example the shield and hero classes.

15) Delegates can hold multiple methods, like a variable holds a data type. It is used in this game to have all weapons look at the same method in order to check if it can fire.