**List of script changes, file by file:**

**Ballcontroller.cs**

* Removed the GetComponent<Rigidbody2D>() call, and put a public variable exposed in the editor
* Check on the Kick() function the ‘y’ component of the initial velocity. Don’t allow small values, as this will make the ball going up or down too slow...

**Brickcontroller.cs**

* Added a public variable with an instance of GamePlay (from the editor). This way, we can access directly to the score from the OnCollisionEnter2D() function

**Death.cs**

* Again, added a public variable with an instance of GamePlay
* Check if ‘lives’ is greater than 0, and only then call the Goal() function

**Mainmenu.cs**

* Nothing changed here

**Playercontroller.cs**

* Added a public (exposed in the editor) variable to hold the Rigidbody2D component. This way, we dont have to use the GetComponent<>() function in the Update() call, which is very costly...

**Topscore.cs**

* Added the update() function to check for the ‘Space’ pressed key
* Use the PlayerPrefs to retrieve the topscore

**Gameplay.cs**

* Removed the GamePlay instance, where the rest of the scripts were requesting an instance of this object... Instead, the scripts that use this instance get it from the exposed variable in the editor
* Another exposed public variable in the editor: The EdgeCollider2D that holds the actual game
* This variable is used in a new function that I wrote for outlining the edges of game area. The function DrawBorder(), which adds a LineRenderer and draws around the points of the EdgeCollider...
* In the function Goal(), I have changed how to access the velocity in the RigidBody component of the Ball: Instead of using the GetComponent<RigidBody>() function, I use a public exposed variable in the editor holding this value
* In the update() function there were a couple of unnecessary accesses to the ‘Score’ and ‘Lives’ variables: These variables are defined in this same scripts, so there is no need to access them like GamePlay.Instance.Score... Instead, we can use Score directly
* - Also, when ‘Lives’ reaches 0, I check whether the score is the maximum, and if it is the case, I save this value using the PlayerPrefs.SetInt()