**SCRIPTING DESIGN**

**SUMMARY:**

In this game we separate the scripts in four different folders, each one handling a different purpose:

Characters:

In this folder we put the scripts for all of the characters behaviours

LevelDev:

In this folder we put the scripts of those things of the levels that have interactions different than the regular collision.

Managers:

In this folder we put the scripts for managing general game functionalities that are present on all of the scenes.

UI:

In this folder we put the scripts for the management of the UI of all scenes.

Total scripts till v0.1: 20 scripts. 7 scripts for the basic game management and 13 for the specific game behavior.

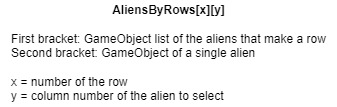
**IMPORTANT NOTES:**

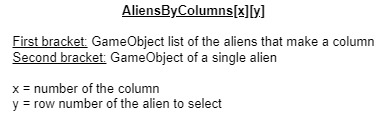
About the characters naming:

* The player ship is just called player. The enemies in the grid are called alien army, individually just alien. The ship that appears eventually is called RandomAlien.

About the alien army:

* The alien army data and generation happens in a script called “AlienArmy”.
* For each army action there is a specific script that gather the alien army information from the ‘AlienArmy’ script.
* For the individual interactions of the aliens there is an “Alien” script attached for each one of them.
* There is an “AlienArmyEditorGenerator” script in the test scene that generates the army on edit mode to quick review of the number of rows, columns, separations and other army features.
* For managing the movement of the army row by row we needed to focus on the army rows, but for doing the detection of the screen borders we needed to focus on the army columns so we end up doing a double lists of gameObjects, one that gathered the rows in the first brackets and individual aliens in the second brackets, and other double list for the columns. Explanation below:





At the end we know it’s not the a proper way of solving this problem so we better change that in a future version.

About the game manager:

* This script manages all of the game transition between scenes, the game losing and winning, the pausing functions: one by time stopping and one by game objects freezing; and finally, the data saving and loading.

About the input manager:

* This whole system is made with Unitys Input System. The normal inputs wont work. Install the Input system package for implementation.
* The inputs set-up are made in the input manager script, but the action made by each button, also called the call-back of the input, are made in the respective code of the action that happens. For example, if the player move with ‘wasd’ then the callbacks are set in the script where the player movement is done.

About the UI:

* There are UI scripts for specific components, for example, one for the HUD.
* The general managing of the menus is done in menu managers scripts, there are many managers for the specific scene type, for example, one for the level menu management and one for the main menu management. This managers are all derived classes of one ‘MenuManager’ script.

**DELEGATES:**

GameManager script:

* OnPauseByTime(bool pausing):

Methods that happen when pausing the game by setting the timeScale of the Time class of Unity.

* OnSceneLoaded():

Methods that happen when we finish loading a scene, like closing the loading screen.

* OnLevelStart():

Behavior when starting a level.

* OnLoseGame():

Behaviour when losing the game.

* OnWinGame():

Behaviour when losing the game.

* OnLevelPauseByFreezing(bool pausing):

For stopping manually the thing in the level to create a fake pause evading the use of the time class of Unity so we can continue using coroutines, animations and else.

InputManager script:

* OnMenuBackInput():

Tells whichever menu manager is at the scene that we pressed the back button.

AlienArmy script:

* OnArmyStart():

Tells the other scripts of the alien army that the army is done being set and can start its behavior.

* OnAlienDestroyed():

All the actions taken when a single alien is destroyed.

Player script:

* OnLosingLive():

What happens when the player loses a single live.