**Panda Jam Test (report)**

I’ve made a panda jam prototype with a random level generator and I’ve included the extra features below:

• bombs (destroy all blocks with the same color).

• firework (destroy a column above the block clicked).

• ice (needs to be destroyed 2x).

• screen shake, particles, music and sound effects.

There are two scenes. Please, find 'Menu' scene and play.

**Architectural decisions:**

1. I like to split UI elements, level elements and management scripts in different containers, i.e., in both scenes there are empty objects in hierarchy tab to handle with;

2. There are patterns as singleton, pooling and factory;

3. 'GameDirector' and 'SoundManager' scripts aren't destroyed on scene transition, with the intention to maintain player's evolution data and to not stop the background music;

4. In 'Menu' scene there is just a script to activate the ‘Play’ button;

5. In Game scene:

⁃ There are two cameras, one to show the UI and other to show all level elements. This last camera is the main one and there is a script called 'ScreenShake' to add a little shaking when the player combines blocks;

⁃ To handle with game rules, there is a script called GridManager. This script is responsible to build the initial grid (w/ blocks) and to use another script ('Block Spawner') to spawn groups of new blocks. In this last script, there is a method used by ‘New Row’ button to add a new row of blocks from grid's bottom, otherwise, this script will spawn by it own every fixed time;

⁃ The 'Block' script is the one to handle with block's rules (all data and mouse click). In this script there is a class called 'ClickedBehaviour' (responsible to handle with different behaviours);

⁃ The 'Toy' script is the one to handle with the object that will be rescued by the player;

⁃ All UI elements are manipulated by 'UIManager' script, to centralize all visual feedbacks;

⁃ 'GlobalVariables' script holds all informations used by more then one script.

**Problems you encountered:**

1. I've let to create the items (bomb, firework and ice) at the end and I've realized it was a mistake. At first I've made whole gameplay with blocks and I hadn't prepared a interface or abstract class to help with the upcoming changes. Then, to avoid a big change my solution was to use polymorphism as a property inside of 'Block' script, just to handle with player's inputs. At the end, it have worked well.