CASUAL GAME DEVELOPER
-Match-3 Case-

1. Goal of the Case

Our request from the applicant is a limited match-3 prototype that is developed in Unity with C# to match Object Oriented Programming principles.

2. Technical Details

The main issues to be considered during development are:
- A code that works with good performance
- Working with 2D Sprites without using canvas elements
- Optimized visual assets
- Overdraw and Drawcall optimization
- Proper naming of the folders, assets and prefabs in the Project

3. Deadline

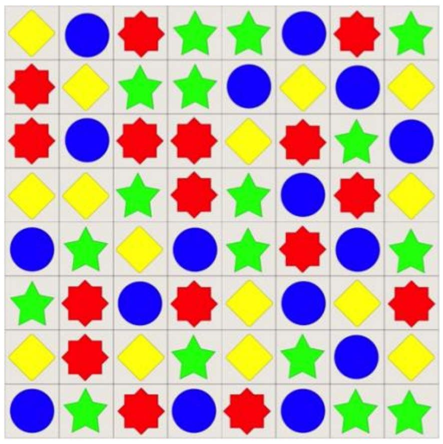
Projected duration to complete this case is 7 days. Projects that is sent after this duration will not be evaluated.

4. Context

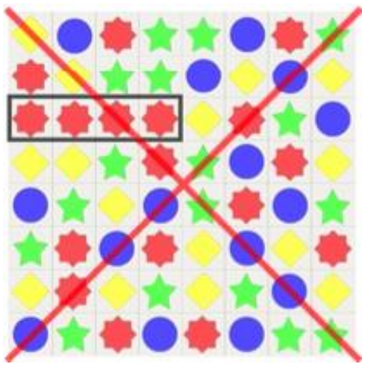
4.1. Creating the Board

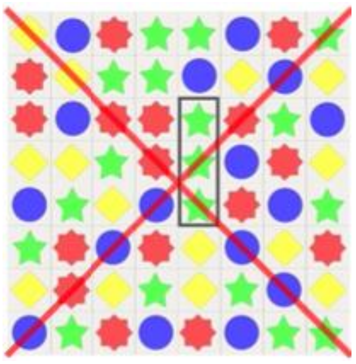
In this step, we expect you to create a 8x8 game field with tiles using Unity 2D Sprites.

Provided that the board consists of 8 rows and 8 columns, a total of 64 empty squares (tiles) should consist of 4 different colors (yellow, blue, green and red) of the game elements (drop) randomly arranged. You can find an example of the randomly distributed board is below:



The most important point to be considered while creating the board is that 3 or more of the same color shouldn't match automatically in the initial state of the board creation. Otherwise, the board creation algorithm will be incorrect. You can find the examples below:





4.2. Swipe Logic

Swipe movement is a necessary mechanic to make matches in the game.

The swipe begins by tapping any drop on the board. Without lifting the finger, it is dragged and dropped on another piece on the lower, upper, left or right side.

If, as a result of this move, three or more drops of the same color coincide in line, a valid match is made. Swapped drops are replaced by the animation and disappear from the board. If the move is invalid, swiped drops return to their initial position by the animation again. (You can see an example watching Video1 in the case folder)

Swipe movement can't be applied between empty or cross tiles. (Video2)

4.3. Spawning and Falling Drops

Drops that disappear from the board after a valid match must be replaced by the drops on the top line.

The filling process occurs when the drops above the empty till fill the tiles below by animating similar to free fall.

While the drops are falling down, new drops must fall over the tiles on the 8th line at the same time (Spawner mechanic). Spawned drops should be randomly generated. (Video3)

We should be able to choose whether the top tiles can be spawners or not. Thus, after matching in that column, new drops will not be dropped because that column does not have a spawner.(Video4)

Hint: Sometimes we may need to Instantiate and Destroy the same gameObject frequently (For example bullet prefab for a gun game). However, frequent use of Instantiate and Destroy operations may not be good for performance. Therefore, instead of destroying the objects we want to use all the time, we can deactivate them and then use them again when necessary.

Note: After the match, it does not need to be animated as in the example gifs. Objects can simply scale and disappear. While the drops are falling from above, they do not need to be animated, stretched as in the example gifs.

Good luck!

CRATOONZ