README

## Removed feature

**Board**

* Shuffle()
* Swap()
* ShiftDownItems()
* FillGapsWithNewItems()
* GetHorizontalMatches()
* GetVerticelMatches()

**Board Controller**

* FindMatchesAndCollapse()
* ShowHint()
* Update() <old version>
* CollapseMatches()
* ShuffleBoardCoroutine()
* ShiftDownItemsCoroutine()

**LevelMoves**: Previously the condition for eGameState::GameOver was all 16 moves had to be made. Now that condition is removed. The condition for gameOver is now that the Backpack is full.

## Added features:

**Singleton**: Add Singleton.cs to Utils folder, Some class now become Singleton : GameManager, Backpack, BoardController

**Cell:**

* Variable: isInteractable – some cells in BottomBoard is overlapped by cells in UpperBoard, so player cannot click on it. True if player can click on it, otherwise False
* Variable: countOverlapped, private set: How many cells in the UpperBoard overlap this cell?
* ToggleInteractable(bool value): Set value of isInteractable. If value is False, it will darken the renderer color.
* AddOverlapped(int offset): Modify value of countOverlapped

**Backpack**: Mono Behaviour

* **Start()** function to construct 5 empty Cells horizontally
* **AddToBackpack**(Cell c): Move the item from a Cell into available Cell in List<Item>, using Dotween
* **FindMatchesAndCollapse()** function: if there are 3 identical Items in Backpack, then delete those 3 Items from Backpack, use Dotween to shorten List<Item>

**Board:**

* Refactor Fill() into FillWithRandomItem()
* Modify FillWithRandomItem() so that total of each Item is divisible by 3

**Board Controller**

* m\_board now become upperboard and lowerboard
* Become partial class: split declaration into 2 scripts: the main one and function for AutoPlayMode
* ProcessClick(): When player clicks on a Cell containing Item, that is "Item != null", then execute AddToBackpack(Cell) function of Backpack, disabled in AutoPlayMode
* IEnumerator AutoPlayRoutine(): active in AutoPlayMode
* StartGame() : added BottomBoard
* Update(): Receive signal from Input.MouseButtonDown(0)

**GameManager:**

* eLevelMode : added NormalMode
* eStateGame: add Win state
* Win() : happened when both bottom\_board and upper\_board are cleared

**UPanelWin**

* Deployed fromm IMenu
* Show this panel when eStateGame == Win