24/4/18 Day1

Created Player and

public string Name { get; set; }

public int Money { get; set; }

public int Position = 0;

List<Property> Properties = new List<Property>();

Created abstract class Property and Regular, Utility, Railway subclass of Property

* This is due to different pay rent functions

Created Dice class

* This is due to only 1 Dice method should be used throughout the entire game

Created enum Colour Groups

* Convenience

Separate project for a specific game board (e.g. Stowe School, UK)

* Testing, as well as customization

5/1/18 Day 2

Created new project called Project.Model to keep all the classes that contains core functionalities that will be used.

* Cleaner, also makes more sense as those classes will be used regardless what UI it is.

Creating board class

* Core functions should be separate from console or any UI
* Board is going to handle all the events on the board
  + Eg. Player moving, dice
    - Updated: only functions, calling in Console/GUI
    - All the actual user decision are stored in Console/GUI, all the procedures/actions stored in separate class
  + ~~Therefore, class Dice is merged into this class~~
    - Reverted, this is due to Console/GUI should be able to access to roll dice, get value and any other progress

Added class Space

* All locations, eg start, jail, etc, will be sub-class of this
  + Polymorphism

Change PayRent to more generic name (TBD)

* Polymorphism

Association between Property and Player: Many to Single

player.Location = (player.Location + squares) % 41;

* Keep location <=40

To be decided:

The board knows where the players (and the spaces?) are

The player knows where the space they are

The space knows what players they have

18/5/6

To be decided:

The board knows where the players (and the spaces? Whether spaces have location instance) are

The player knows where the space they are

~~The space knows what players they have~~ -> space object should not need to know about player on them (nothing to do before the game start), waste of memory if no player on the space aka something feels weird logically

Simple functional programming for cards:

* Passing execute operation function as variable

Project.Config made

* For default gameboard, stowe gameboard, custom gameboard, etc.
  + Having function that makes create and pass back style of properties, characters, etc.

18/5/10

(and the spaces? Whether spaces have location instance)

* Considering removing location instance on Spaces since they are the same as the index the way I am setting up now. However, instance is more secured and flexible than index. Also, you can retrieve the location of the space from the object directly.

~~The board knows where the players are -> dictionary, with player as key and nullable int as location~~

~~The player knows where the space they are -> nullable int instance for location~~

~~·nullable int because null means the player is out the game~~

~~Game board knows where the players are~~

* ~~1. I would need a collection of players for the sake of keep check who is in the game and whose turn it is anyway~~
  + ~~Likely dictionary due to customizable key~~

~~While loop that loops through the player collection~~

~~If not gameboard check, then a List as collection.~~

18/6/18

Finish default gameboard ver.1

Player object and pieces?

Static function folder to contain all function for now

Rent as new instance property on each regular houses