Initial (Incomplete) UML Class Diagram for Player class

Be sure to use the given identifiers for the class, instance variables and methods as shown in each of the UML diagrams.

If you want to use a C# Property rather than the given Accessor and Mutator methods (eg GetName() and SetName() in Player class) be sure to replace all Accessors and Mutators with a Property implementation.

Player
-name: string **-location: Square** **-playerTokenImage: Image** **-playerTokenColour: Brush**
+Player() **+Player(string, Square)** **+SetName(string)** **+GetName(): string** **+SetLocation(Square)** **+GetLocation():Square** **+GetPlayerTokenImage(): Image** **+GetPlayerTokenColour(): Brush** **+SetPlayerTokenColour(Brush) // code provided below**

The Player class will need to have the reference to `System.Drawing` added as well as the directive `using System.Drawing;` so that `Brush` and `Image` are recognised in addition to a reference to the Square Class.

The following method is to be added to your Player Class either as a separate Mutator method as shown below or as the set block of playerTokenColour property.

```
public void SetPlayerTokenColour(Brush value){

    playerTokenColour = value;
    playerTokenImage = new Bitmap(1, 1);
    using(Graphics g = Graphics.FromImage(PlayerTokenImage)) {
      g.FillRectangle(playerTokenColour, 0, 0, 1, 1):
    }
} //end SetPlayerTokenColour
```

The default Constructor of Player should throw an exception of type `ArgumentException` with an error messages that its use is invalid.

All other methods, apart from *SetPlayerTokenColour(Brush)*, have simple and obvious implementations.

Additional functionality will be added to this class in future releases of the assignment specification.

However this current specification is sufficient for Part A of the assignment to be completed.