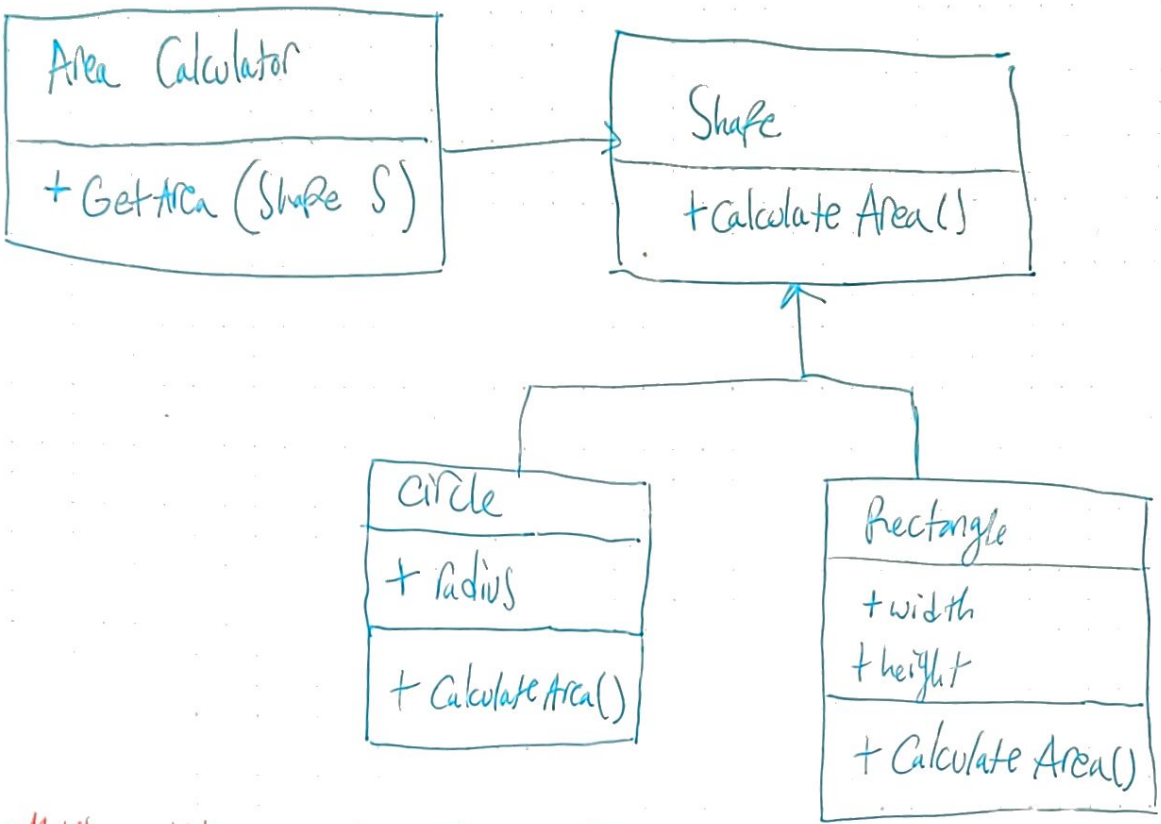
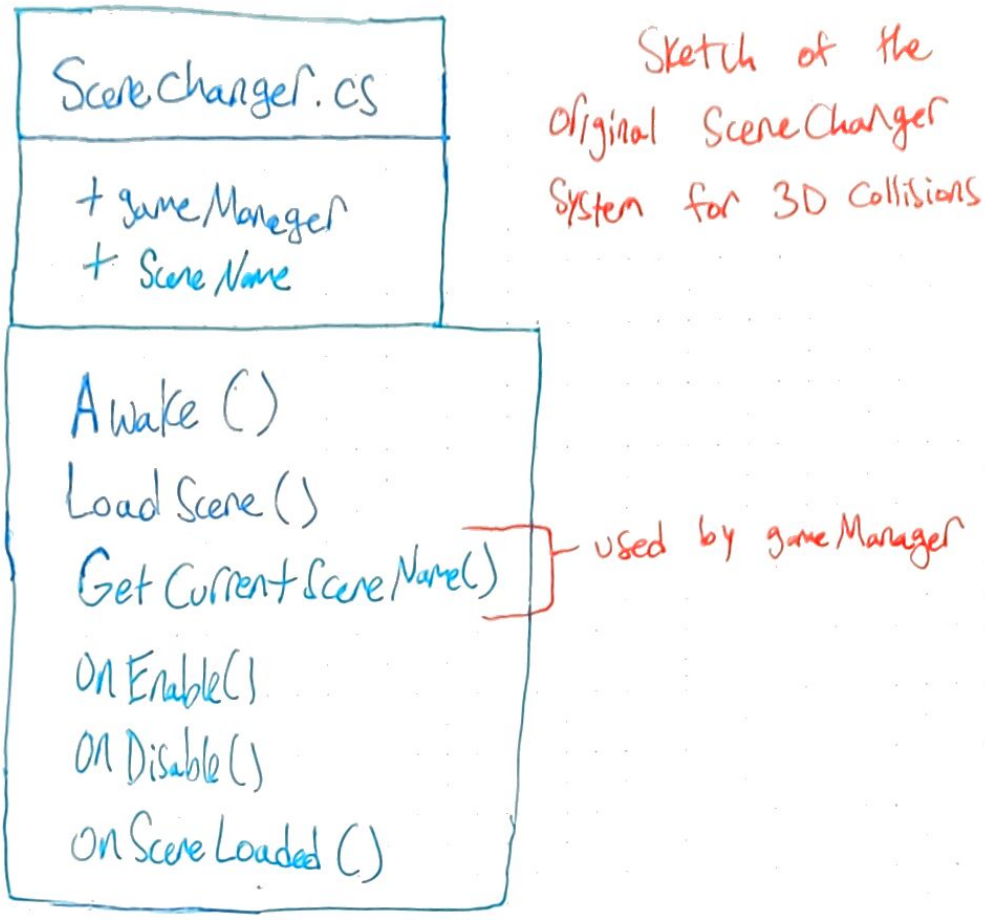


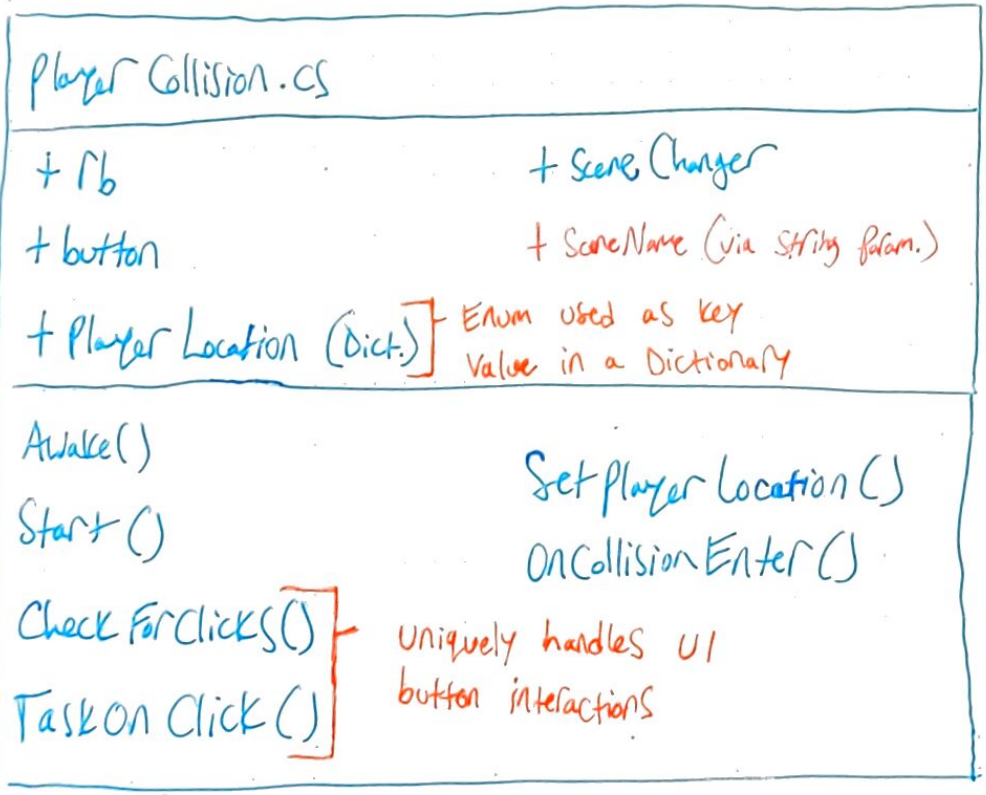
- Maybe this can be done with detection types in Scene Switching?

- Inheritance: "is a" relationship
  - "A rectangle is a Shape."
  - "A circle is a Shape."
- Composition: "has a" relationship
  - "A rectangle has a width and a height."
  - "A circle has a radius and a circumference."

A shared attribute between these Shapes is that they both have an area, it is just calculated differently.



Sketch of the original SceneChanger System for 3D collisions



```
Public Interface ISceneChanger {
    // Contains Shared logic
    // e.g. public String SceneName { get; set; }
}
```

This Interface would be applied to both 3D and 2D-based Scene Changes.

Maybe there's a good Interface for Collisions & UI that's already ~~built~~ built-in?

```
Public Interface ICollidable {
    // Stuff about Collision
}
```

This interface would only be applied to 3D Collision-based Scene Changers.

Private Dictionary

- Maybe 2 parameters?
- GameObject & then a SceneAsset as Key & Value, respectively.

Goal: the user should be able to drag-and-drop into Inspector

private Dictionary to be added to on the object

Scene Changer 3D:

- Needs the OnCollisionEnter()

Scene Changer 2D:

- Needs the CheckForClicks() and TaskOnClick() methods.

Comparing the dictionary from PlayerCollision.cs:

```
SetPlayerLocation (PlayerLocation location) {
    transform.position = locationPositions[location];
}
```

SetPlayerLocation: Method to call
PlayerLocation: enum (key)
location: id in Dictionary
locationPositions: actual Dictionary

```
GetDictionaryScene (GameObject otherObject) {
    Scene = SceneDictionary[otherObject]
}
```

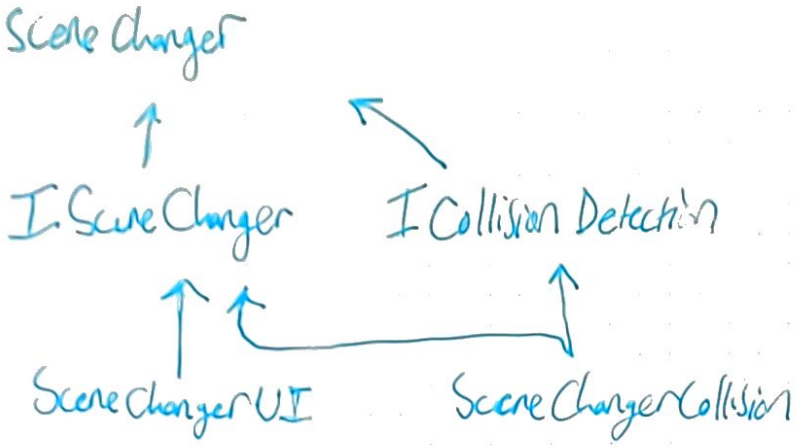
GetDictionaryScene: Method to call
GameObject: key
otherObject: id in Dictionary
Scene: The Scene we want to switch to
SceneDictionary: Dictionary

Collision Handler

```
Void OnCollisionEnter():
    // get the object in question &
    // get the object you're colliding with
}
```

PlayerMovement (should have some kind of inheritance or interface)

```
void OnCollisionEnter():
    //Maybe perform a switch case to
    // see what they are colliding with?
```



Scene Changer UI and Scene Changer Collision "are a" form of a Scene Changer (Inheritance)

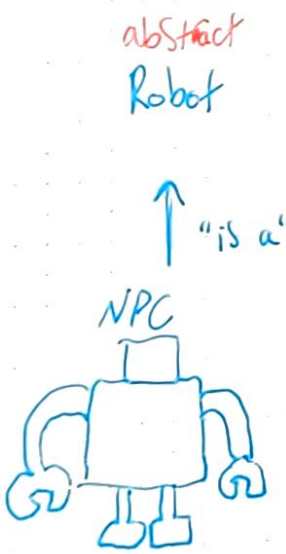
What methods do these new classes share?

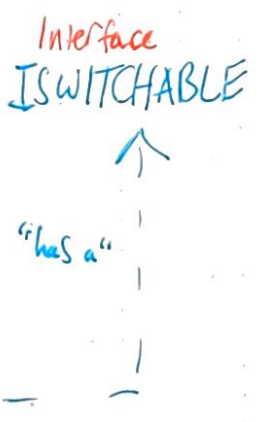
- Load Scene()
- Get Current Scene Name()
- On Enable()
- On Disable()
- On Scene Loaded

Protected properties:

- Scene Name (String)

Perhaps interface segregation?





For Scene Changers:

- Both "are a" Scene Switcher (inheritance).
- But Scene Changer Collision "has a" collider while Scene Changer UI doesn't.

Load Scene (Scene)

↑ This should be the value asset associated with the dictionary.

Methods	IScene Changer	ICollidable	IClickable
Awake()	X		
Start()	X		
Get Dictionary()	X		
Load Scene ()	X		
Get Current Scene Name()	X		
On Enable()	X		
On Disable()	X		
On Scene Loaded()	X		
On Collision Enter()		X	
Check For Clicks()			X
Task On Click()			X
Check For Key()	X		
Get Dictionary Scene()	X		