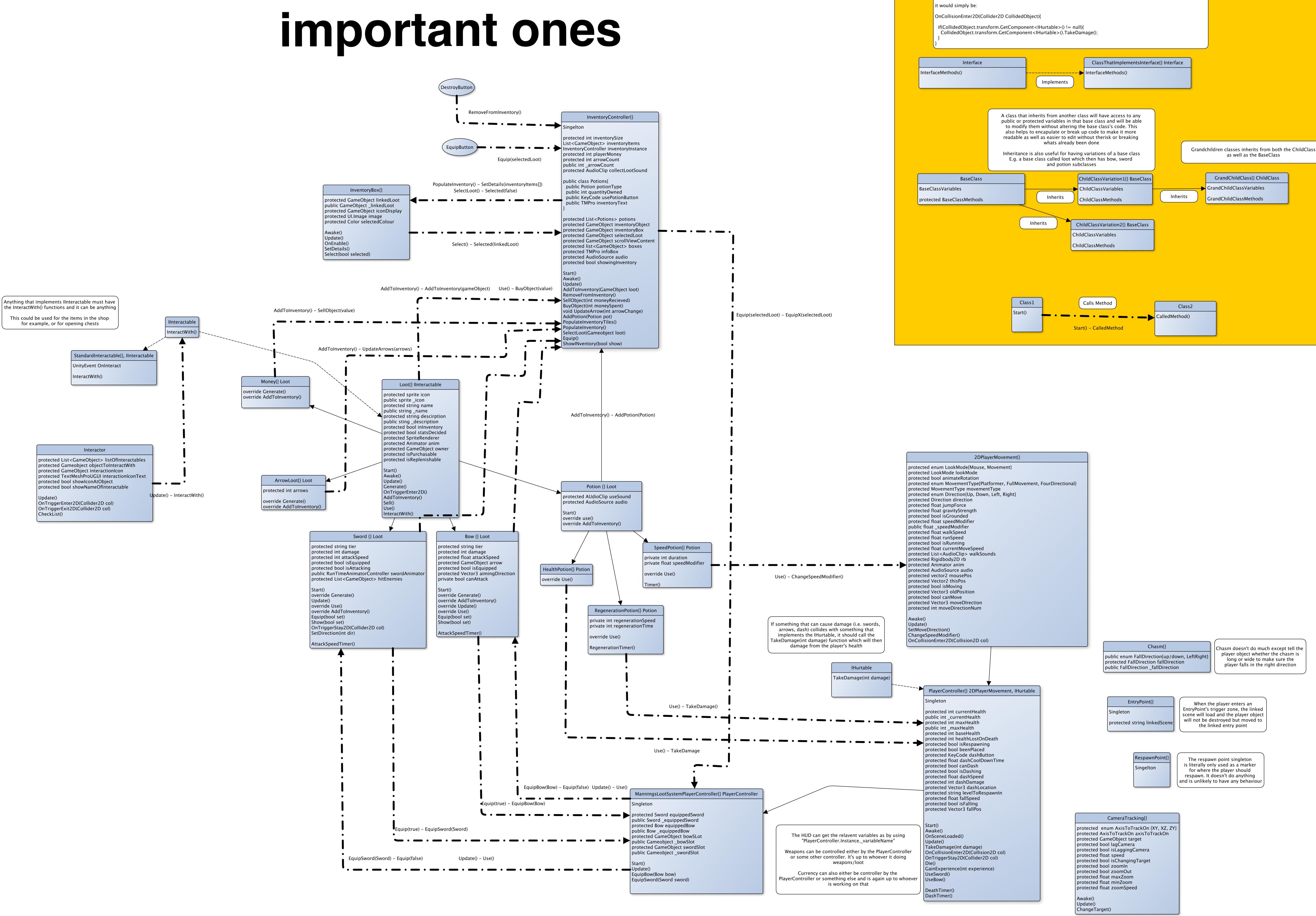
Still a bunch of "Calls Method" relations missing, but I got most of the important ones



An interface is a bit like a contract. Any class that implements an interface

implements a paticular interface and call a method for that interface rather than a method for a particular class.

A good example of this would be weapons and thinsg that can be hurt. Instead of checking for individual scripts like this:

Interfaces can also be checked i.e. A class can check if another class

if(CollidedObject.transform.GetComponent<EnemyScript>() != null){

if(CollidedObject.transform.GetComponent<EnemyScript>() != null){

if(CollidedObject.transform.GetComponent<DestructableScript>() != null){

MUST have that interface's methods.

EnemyScript.TakeDamage()

EnemyScript.TakeDamage()

EnemyScript.TakeDamage()

OnCollisionEnter2D(Collider2D CollidedObject){



When an IInteractable object enters or exits the trigger zone, it will be added or removed from the list of interactables

When the player presses e while something can be interacted with, it will call the InteractWith() function on that obbject