****

**Technical Document for:**

**RPG Hero**

**The best mobile RPG money can buy**

“It’s dangerous to go alone, take RPG Hero with you!”™

All work Copyright ©(2014) by Walk Home Games

Version # 1.00

December 6th, 2014

Table of Contents

[Architecture 3](#_Toc407053661)

[Overview 3](#_Toc407053662)

[Start Scene 3](#_Toc407053663)

[Player 4](#_Toc407053664)

[Inventory Scene 4](#_Toc407053665)

[Inventory (Work In Progress) 6](#_Toc407053666)

[InventoryItem 6](#_Toc407053667)

[InventoryItem Types (Work in Progress) 6](#_Toc407053668)

[InventoryItemDatabase 7](#_Toc407053669)

[InventorySlot 7](#_Toc407053670)

[ItemInformation 7](#_Toc407053671)

[Store Scene 7](#_Toc407053672)

[Clerk 7](#_Toc407053673)

[Level Select Scene 9](#_Toc407053674)

[Level Scene 11](#_Toc407053675)

[Icon 14](#_Toc407053676)

[IconState 14](#_Toc407053677)

[IconSlot 14](#_Toc407053678)

[IconSpawner 14](#_Toc407053679)

[Enemy 15](#_Toc407053680)

[EnemyState 15](#_Toc407053681)

[EnemySpawner 15](#_Toc407053682)

[PlayerStats 16](#_Toc407053683)

[Character Scene 16](#_Toc407053684)

# Architecture

## Overview

RPG Hero will consist of roughly 25 scenes, there will be

* A Start Scene that will introduce the player to the game and allow them to start the game.
* An Inventory Scene that will show the player’s inventory and allow them to equip weapons, shields, and magic,

* A Store Scene that will allow the player to buy potions, weapons, shields, and repair hammers in addition to selling unused weapons and shields.

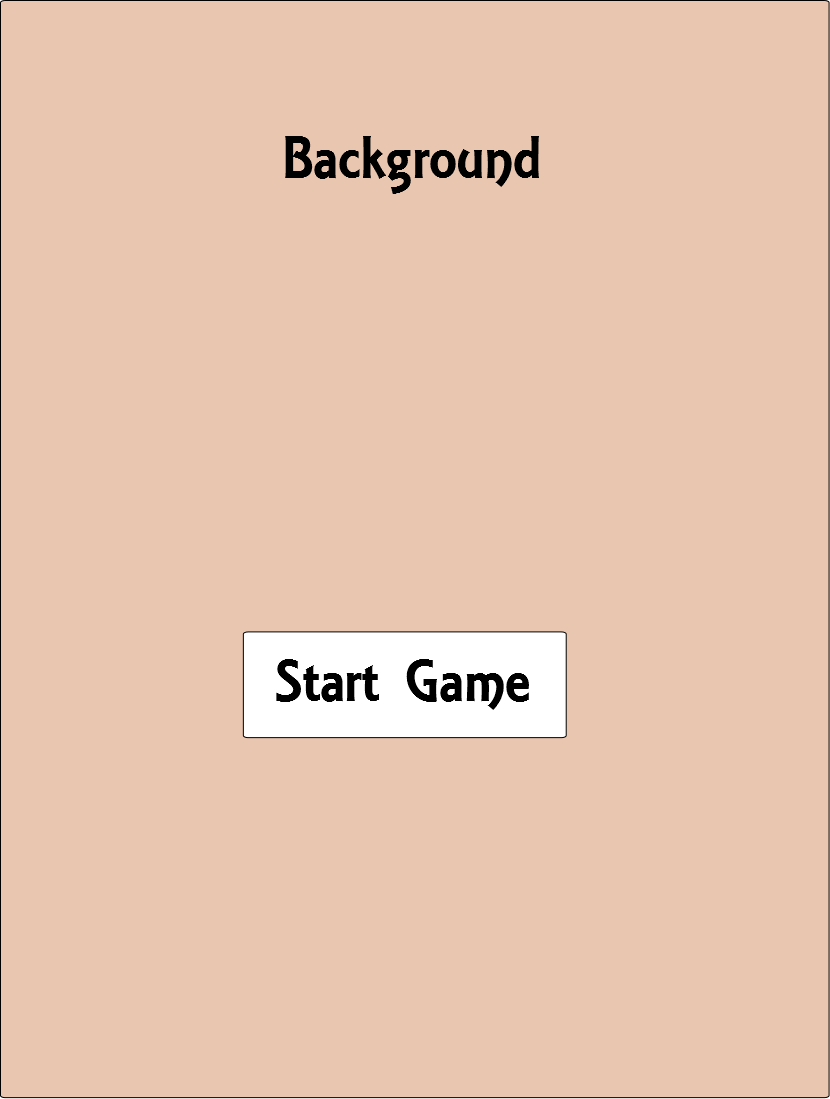
* A Character Scene that will allow the player to check there current stats and experience or level up their stats when they have gained enough experience.

* A Level Select Scene will represent the main hub for the player, from here they can access their inventory, the store, their character page and the level they wish to play next.
* Lastly, there will be 20 Level Scenes to represent each level in the game. Each scene will represent a level that the player must complete to unlock the next level. Each scene will control the spawning of the enemies and the icons.

The UI System used will be Unity’s new UI system that has been introduced into Unity 4.6.

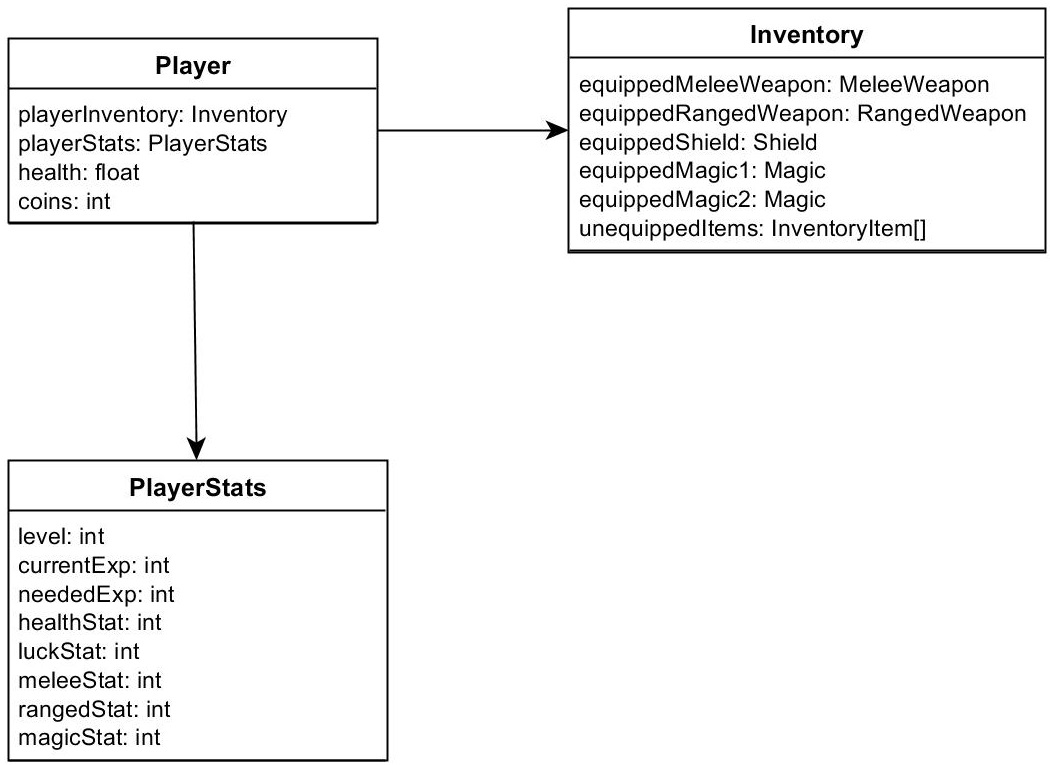
## Start Scene

The Start Scene will have a “Background” Game Object and a “Start Game” Button (see below), when the player presses the Button, the game will switch over to the Level Select Scene. The “Start Game” Button will have a script attached to it that will load the Level Select Scene.



## Player

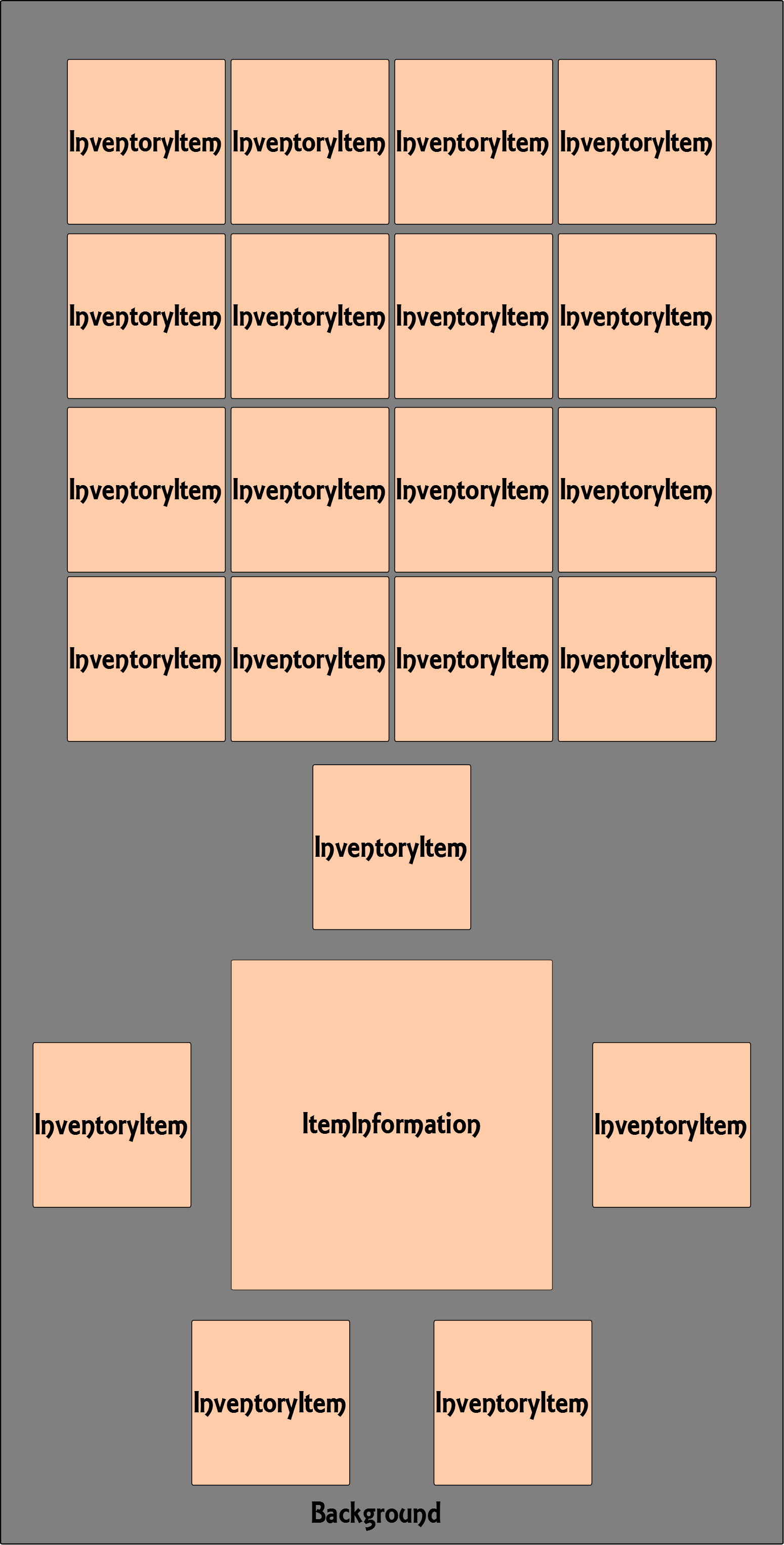
* The Player will be represented by a singleton,
* The Player Class will have to be saved whenever the Player changes their stats or inventory
* The Player Class will have one Inventory instance and one PlayerStats instance
* The Player class will be used by the different scenes to access the Player’s information such as their Inventory or stats in order to load in some information for the scene.
* playerInventory is an instance of the Inventory class that will represent the items the Player has.
* playerStats is an instance of the PlayerStats class that will represent the stats the Player has.
* health is the float used to display the health the Player currently has, if it reaches zero the Player will return to the level select scene.
* coins is the integer used to manage the number of coins the Player has.

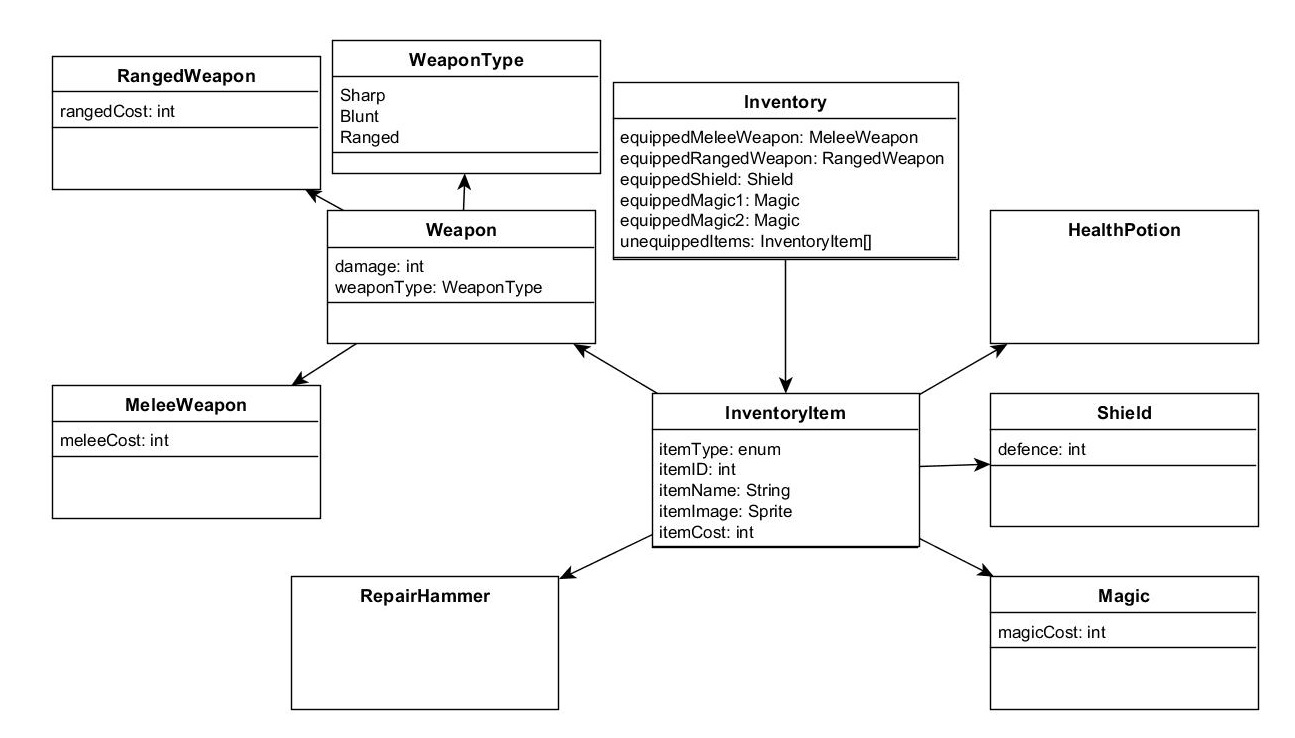


## Inventory Scene

The Inventory Scene will have a

* Background Game Object,
* A grid of InventorySlot Game Objects that will display the Player’s Inventory and
* An ItemInformation Game Object that will display the information of the selected item and compare it to the currently equipped item.





## Inventory (Work In Progress)

The Inventory Class will consist of an array of InventoryItems. The inventory class will also store the items the Player has equipped.

* equippedMeleeWeapon is the MeleeWeapon that the Player has equipped
* equippedRangedWeapon is the RangedWeapon that the Player has equipped
* equippedShield is the Shield that the Player has equipped
* equippedMagic1 is the first Magic power that the Player has equipped
* equippedMagic2 is the second Magic power that the Player has equipped
* unequippedItems is an array of InventoryItems that the Player does not have equipped

## InventoryItem

An InventoryItem is any item that is contained in the Player’s inventory, InventoryItem’s will be divided by ItemType. The types will be MeleeWeapon, RangedWeapon, Shield, Magic, HealthPotion, and RepairHammer.

* itemType will be the type of the item
* itemName will be the name of the item
* itemID will be the ID associate with the item (will be used to store the Player’s inventory when saving)
* itemImage will be the sprite attached to the item
* itemCost will be the coin cost of the item (Magic items will have an invalid cost since they cannot be purchased or sold)

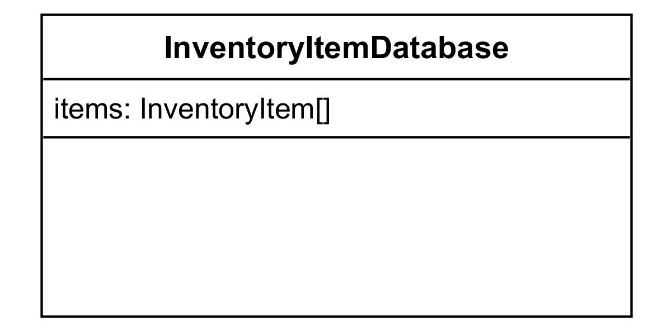
## InventoryItem Types (Work in Progress)

* ~~RangedWeapon: rangedCost is the amount that will be taken from the Player’s Ranged Stat per use (i.e. the higher the Player’s Ranged stat the more times they can use the item)~~
* ~~MeleeWeapon: meleeCost is the amount that will be taken from the Player’s Melee Stat per use (i.e. the higher the Player’s Melee stat the more times they can use the item)~~
* Weapon: Base class that determines how much damage the Weapon can do and what type of weapon it is
* Shield: defence is the attack damage the shield can block before it needs to be repaired (shields will not regenerate)
* ~~Magic: magicCost is the amount that will be taken from the Player’s Magic Stat per use (i.e. the higher the Player’s Magic stat the more times they can use the item)~~

## InventoryItemDatabase

InventoryItemDatabase is the class that will contain all items that the Player can buy throughout the entire play through of RPG Hero.

* items: array of all of the InventoryItems in the database



## InventorySlot

InventorySlots are Game Objects that will be used to display InventoryItems. Each slot can contain one InventoryItem.

## ItemInformation

ItemInformation will display the currently selected items information (i.e its type, name, cost, etc.) and the currently equipped items information (if applicable).

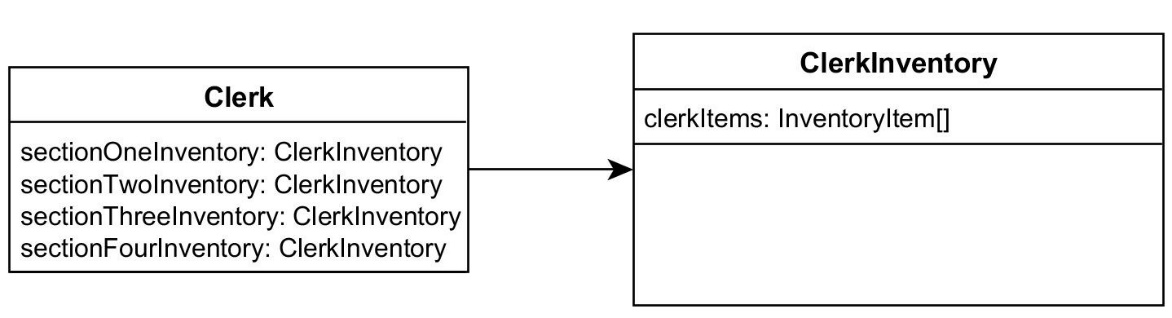
## Store Scene

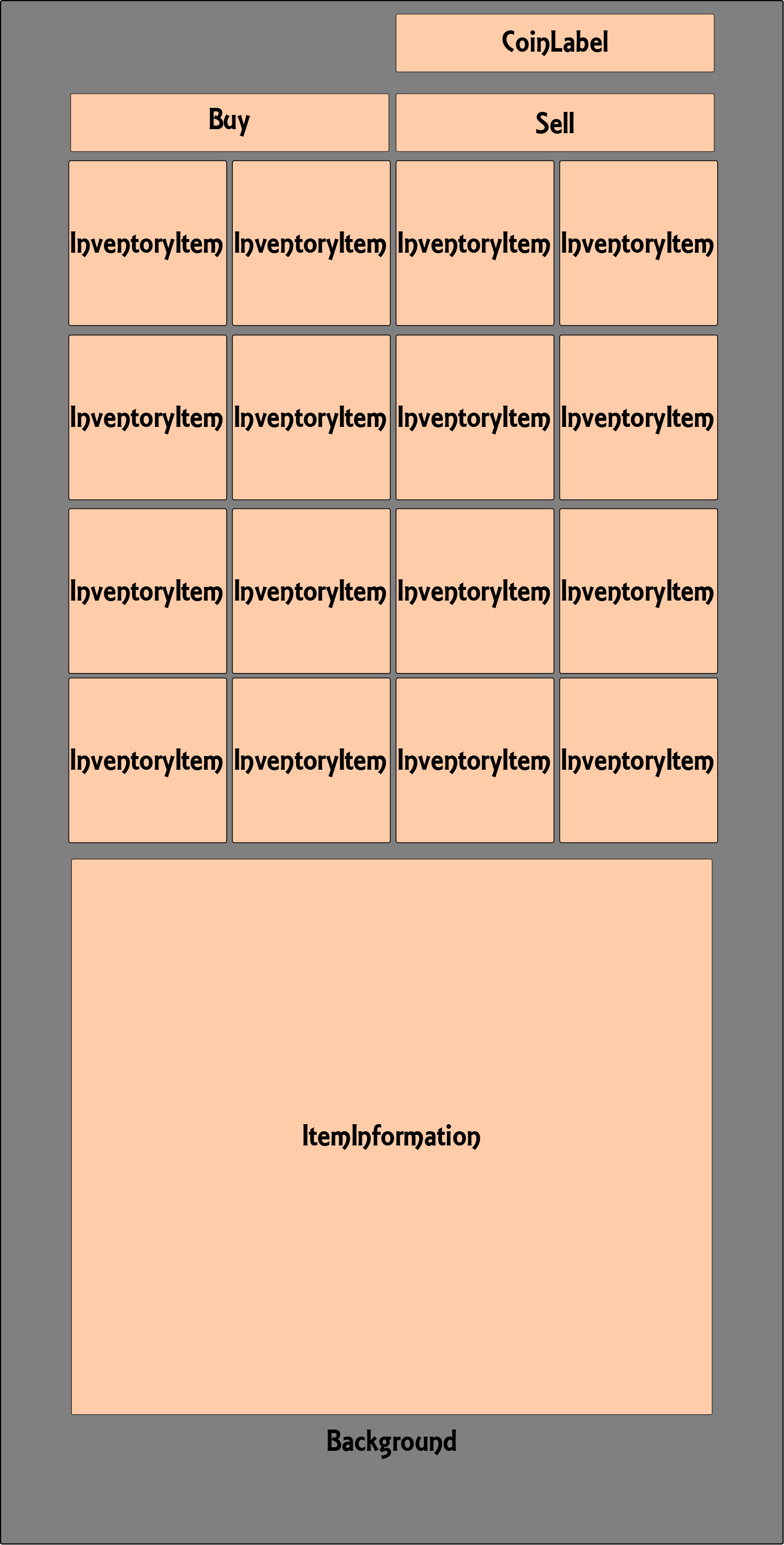
* There will be a Background Game Object
* There will be two buttons to switch between buy and sell mode
* There will be a grid of InventorySlot Game Objects that will display the Inventory of the Clerk or the Player depending on the mode (see InventorySlot above)
* There will be a CoinLabel that will display the amount of coins the Player has
* There will be an ItemInformation box that will display the current selected item and will contain a button allowing the Player to purchase or sell the current item. (see ItemInformation above)

## Clerk

The Clerk Class will represent the clerk for the Store. The Clerk will have a ClerkInventory associated with them. Their ClerkInventory will depend on the last section the Player unlocked.

* sectionOneInventory is the inventory the Player will see when they have unlocked the first section
* sectionTwoInventory is the inventory the Player will see when they have unlocked the second section (unlocks when the Player reaches level 6)
* sectionThreeInventory is the inventory the Player will see when they have unlocked the third section (unlocks when the Player reaches level 11)
* sectionFourInventory is the inventory the Player will see when they have unlocked the fourth section (unlocks when the Player reaches level 16)
* The ClerkInventory class will be an array of InventoryItems that will represent the InventoryItems the Clerk has for sale



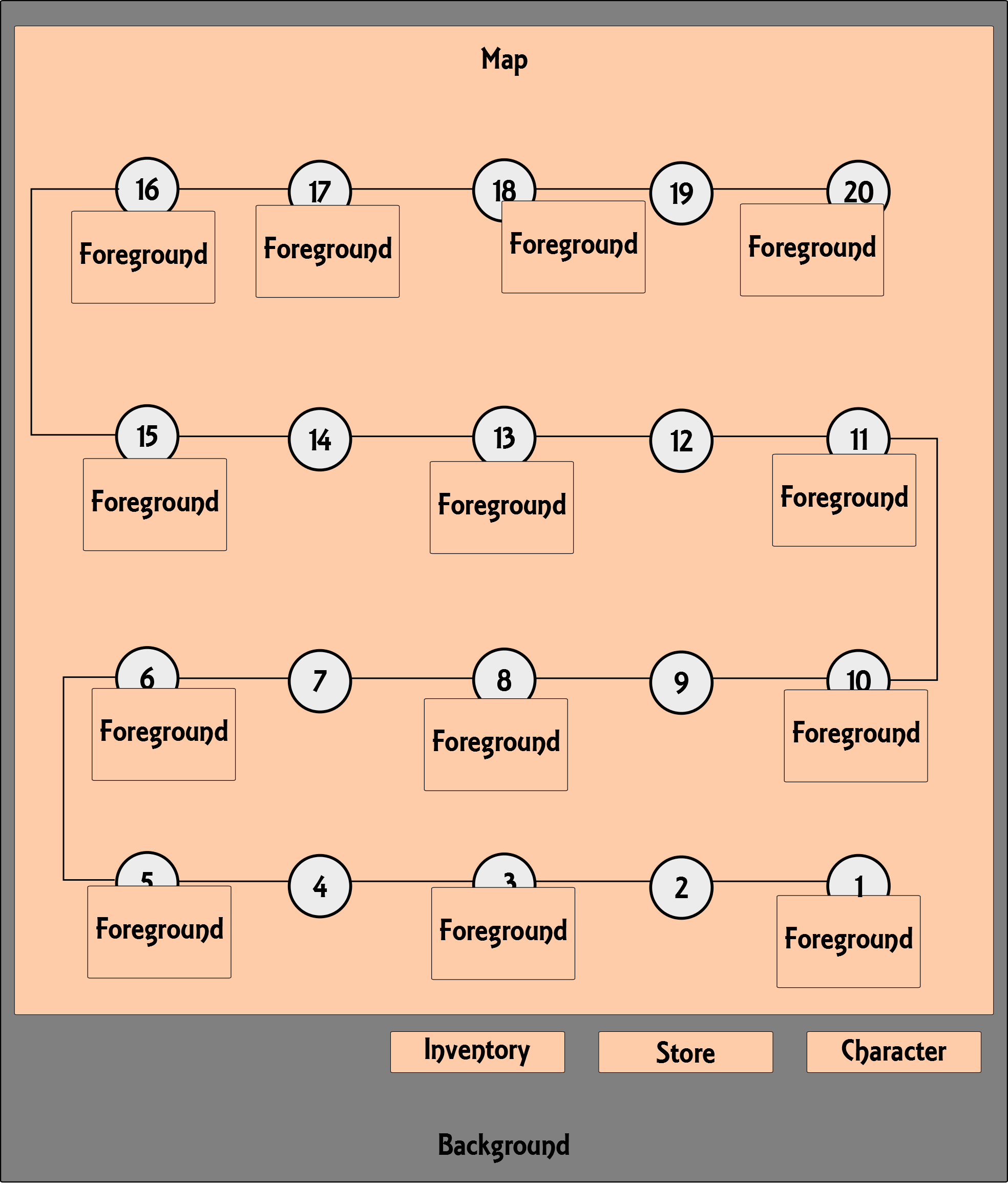


## Level Select Scene

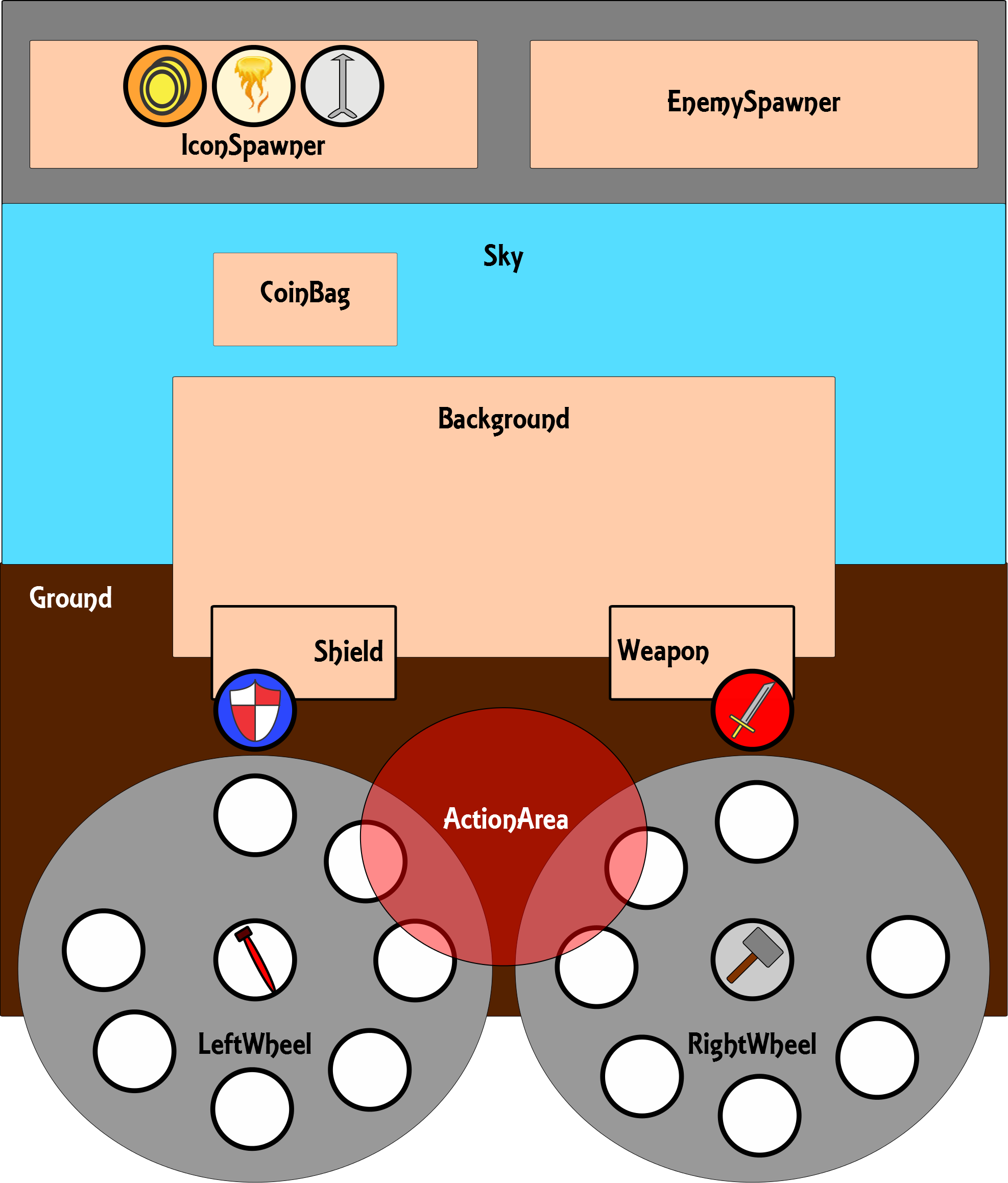
The Level Select Scene will have

* A Background Game Object
* A Map Game Object
* Buttons for Character, Inventory, and Store
* Some Foreground Game Objects on top of the map for visual effect
* Buttons for each of the 20 levels.

The Inventory, Store, Character, and 20 level buttons will take the Player to the respected scenes.

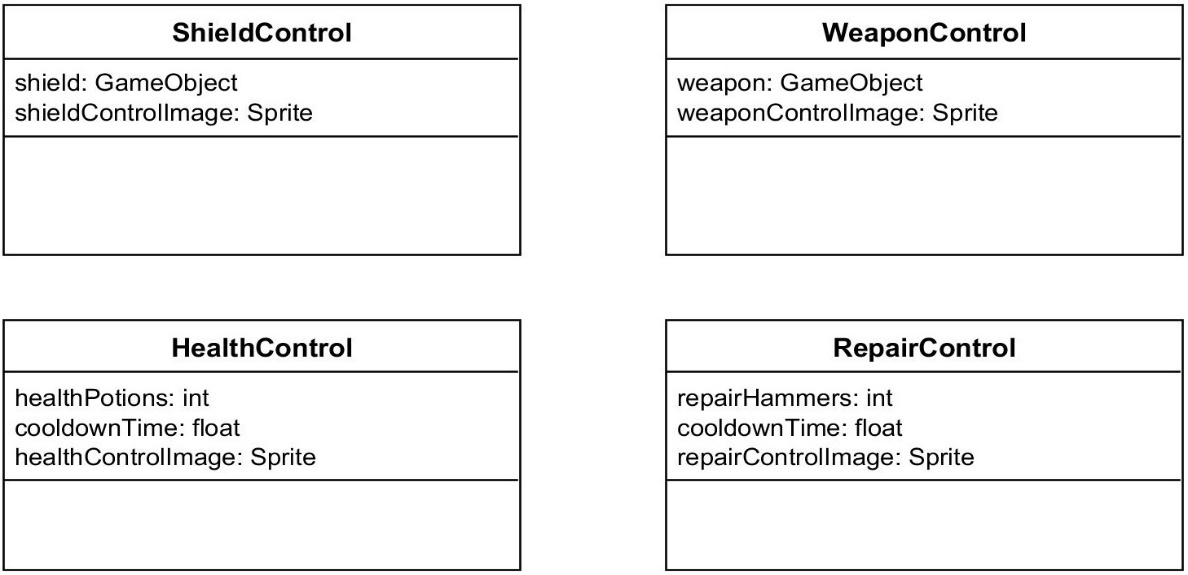


## Level Scene

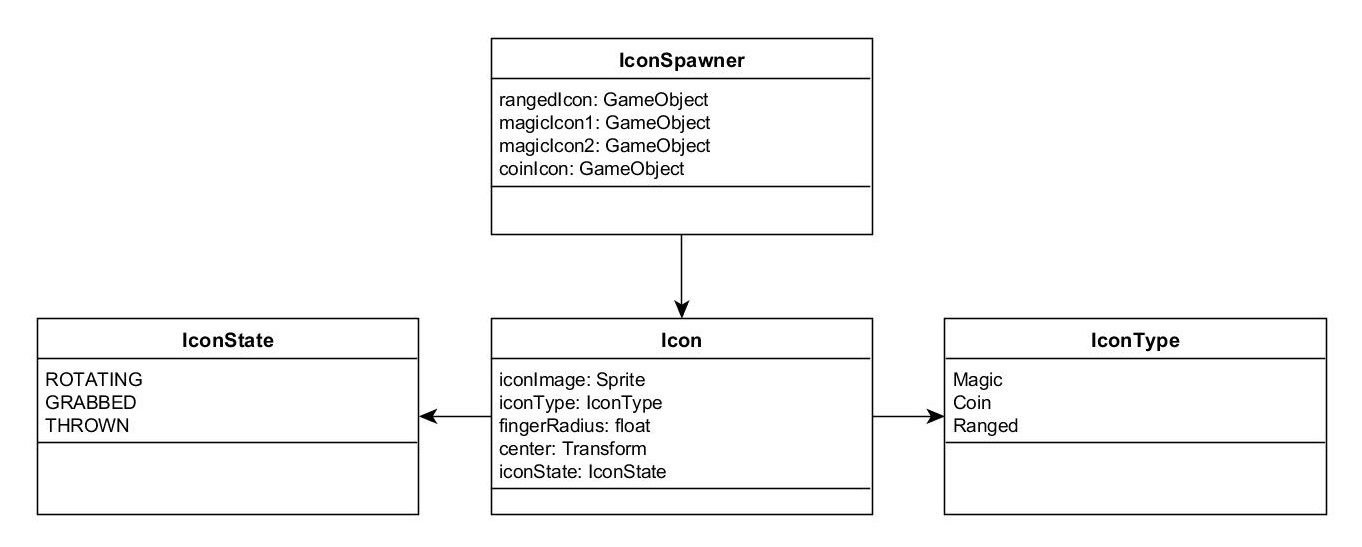


Each Level Scene will be similar, it will consist of a Ground Game Object, Sky Game Object, and Background Game Object. In the foreground will be two Wheel Game Objects, an ActionArea Game Object, a WeaponIcon, ShieldIcon, HealthIcon, RepairIcon, a CoinBag, 14 IconSlot Game Objects, a Weapon Game Object, and a Shield Game Object. There will also be IconSpawner and EnemySpawner which will be invisible Game Objects.

* The ActionArea is a Game Object that the Player will be required to bring certain icons into to activate them.
* The CoinBag is a Game Object that will collect CoinIcons when they are thrown at the CoinBag. The CoinBag will have a collider attached to it to detect collision and a script attached to it to increment the Player’s coin count when a CoinIcon is collected.
* The Weapon Game Object will represent the currently equipped weapon and will be seen when the Player uses the WeaponIcon Game Object.
* The Shield Game Object will represent the currently equipped shield and will be seen when the Player uses the ShieldIcon Game Object.



* The WeaponControl is the persistent icon that will allow the Player to attack. It will have a collider and a script attached that will access the Player’s equipped weapon and will control the Weapon Game Object.
  + weapon will be a reference to the Weapon GameObject to control it
  + weaponControlImage will be the sprite of the control
* The ShieldControl is the persistent icon that will allow the Player to defend. It will have a collider and a script attached that will access the Player’s equipped shield and will control the Shield Game Object.
  + shield will be a reference to the Shield GameObject to control it
  + shieldControlImage will be the sprite of the control
* The HealthControl is the persistent icon that will allow the Player to heal. It will have a collider and a script attached that will increment the Player’s health when it is used.
  + healthPotions will be the number of health potions the Player has
  + cooldownTime will be the time between uses of the health potion
  + healthControlImage will be the sprite of the control
* The RepairControl is the persistent icon that will allow the Player to repair an item automatically. It will have a collider and a script attached that will repair either the weapon or the shield depending on what it is dragged to.
  + repairHammers will be the number of repair hammers the Player has
  + cooldownTime will be the time between uses of the repair hammer
  + repairControlImage will be the sprite of the control



## Icon

The Icon class will represent an icon that can spawned in the game. Some examples are the RangedIcon, CoinIcon, and any MagicIcons. Icons will spin around one of the Wheel Game Objects depending on which side the IconSpawner spawns it on. Each Icon will have a collider attached to it to handle collisions with other game objects.

* iconImage will be the sprite attached to the icon
* iconType will be the type of icon (Magic, Coin, or Ranged)
* fingerRadius is the finger size (used for fat finger implementation, there is likely a better spot for this than in the Icon class)
* center is the Transform that the Icon will rotate around, i.e. the left wheel or the right wheel
* iconState is the state that the Icon is in.

## IconState

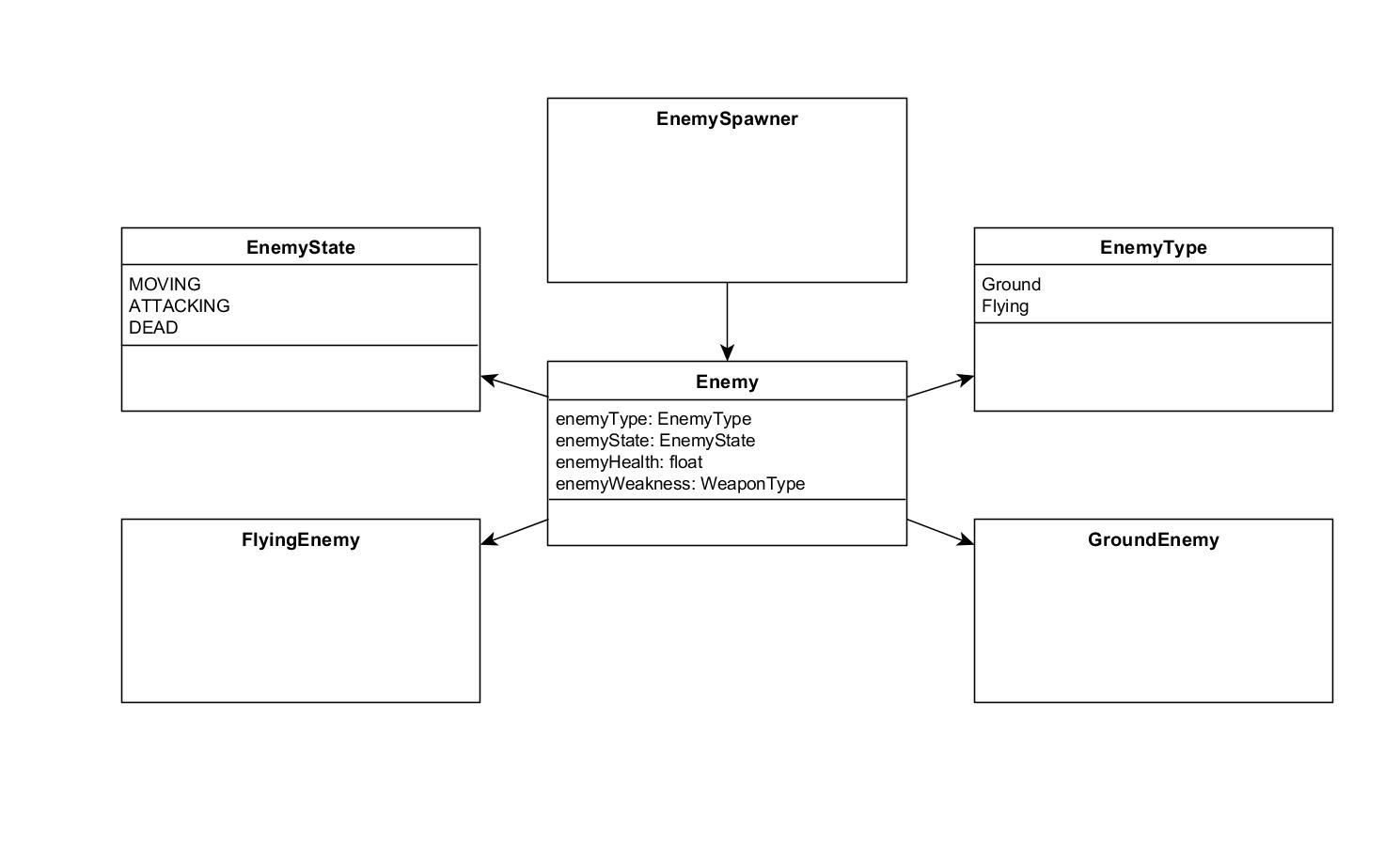
* ROTATING: the Icon is rotating around a wheel (has not been selected)
* GRABBED: the Icon has been selected by the Player
* THROWN: the Player has thrown the Icon

## IconSlot

An IconSlot will be a slot that an Icon can spawn in. Each IconSlot will hold one Icon. When an IconSlot is empty, the IconSlot will notify the IconSpawner and a new Icon will be placed into it if possible.

## IconSpawner

The IconSpawner class will be in charge of managing the spawning of icons in the level. This means it will have to look at the IconSlots in the scene and spawn a new icon in when appropriate. The IconSpawner will have to work with the Player’s Inventory and their Stats to determine what Icons are to be spawned in.



## Enemy

The Enemy class will represent an enemy in RPG Hero, the Enemy will control its own state and AI. When an enemy dies it will notify the EnemySpawner.

* enemyType is the Type of the enemy
* enemyState is the current state the enemy is in
* enemyHealth is how much health the enemy has
* enemyWeakness is the WeaponType that the enemy is susceptible to (can be NONE)

## EnemyState

* MOVING: the enemy is moving around but not attacking
* ATTACKING: the enemy is going to attack
* DEAD: the enemy died

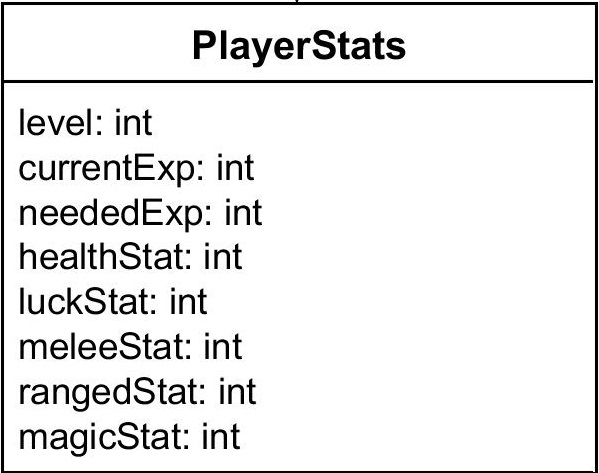
## EnemySpawner

The EnemySpawner will be in charge of managing the spawning of enemies in a level. It will spawn in the initial enemy or enemies and when one dies will have to either spawn another or let the level know it is finished.

## PlayerStats

PlayerStats is the class that will manage the Player’s stats. The stats will include what level the Player’s character is, their experience, their health stat, luck stat, melee stat, ranged stat, and magic stat.

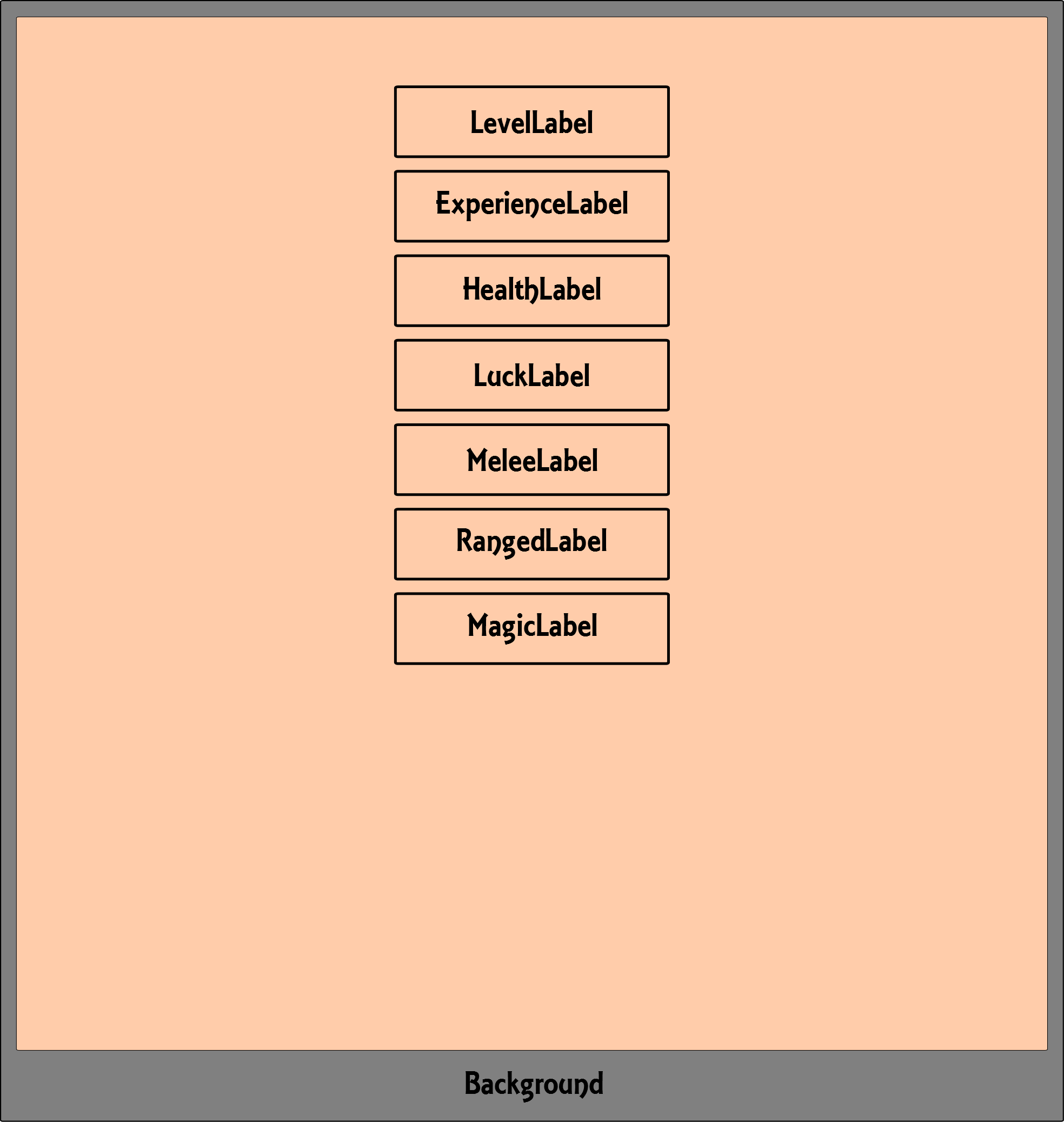
* level is the integer that will represent the Player’s current level
* currentExp is the integer that will represent the current amount of exp the Player has
* neededExp is the integer that will represent how much exp the Player needs to level up.
* healthStat is the integer that will represent the Player’s Health Stat
* luckStat is the integer that will represent the Player’s Luck Stat
* meleeStat is the integer that will represent the Player’s Melee Stat
* rangedStat is the integer that will represent the Player’s Ranged Stat
* magicStat is the integer that will represent the Player’s Magic Stat



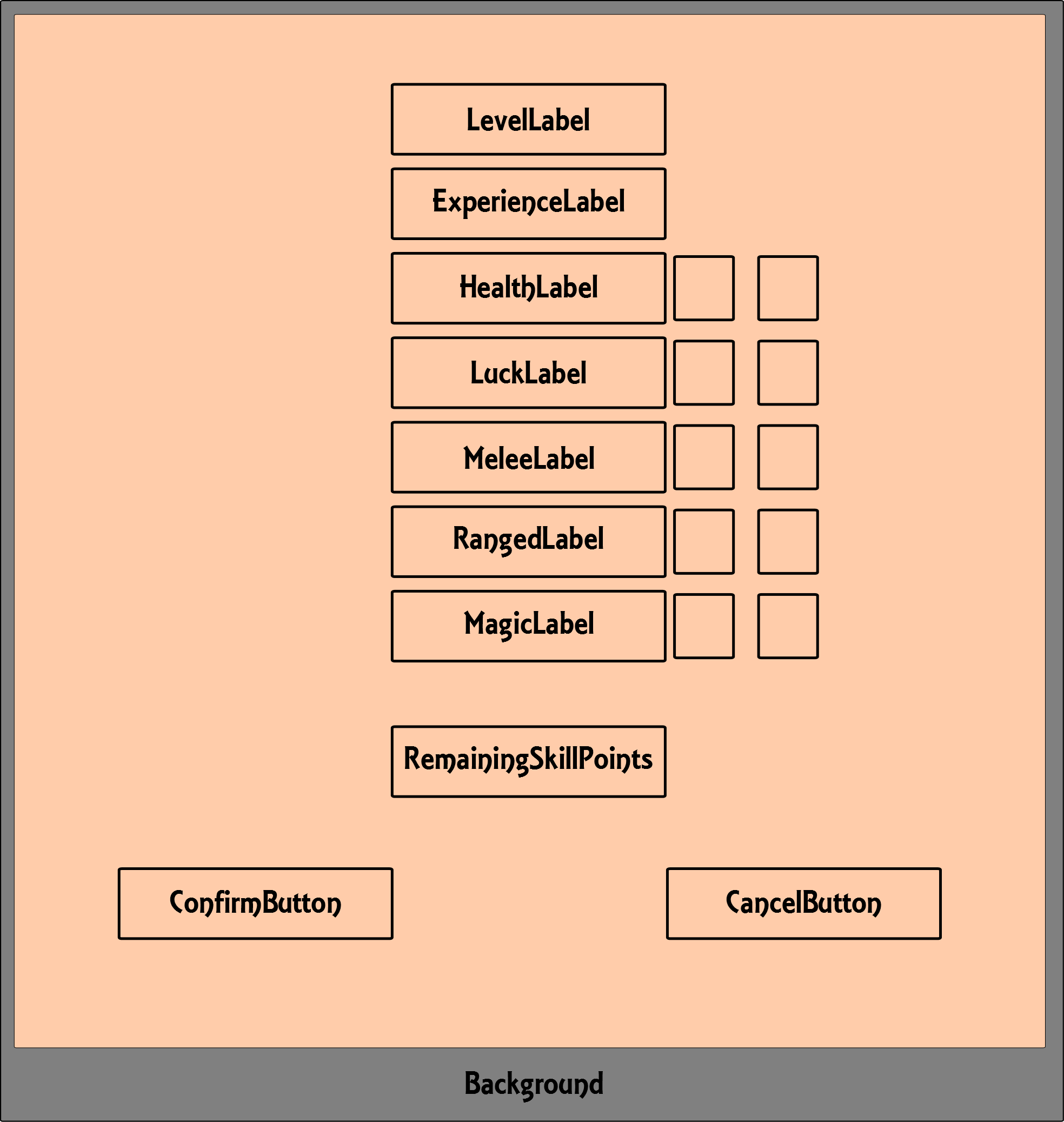
## Character Scene

The Character Scene will consist of a Background Game Object and Labels for all of the Player’s Stats.

* The LevelLabel will display the Player’s current level
* The ExperienceLabel will display the Player’s experience
* The HealthLabel displays the current value for the Player’s Health stat
* The LuckLabel displays the current value for the Player’s Luck stat
* The MeleeLabel displays the current value for the Player’s Melee stat
* The RangedLabel displays the current value for the Player’s Ranged stat
* The MagicLabel displays the current value for the Player’s Magic stat



Once the Player has enough experience points they will level up and the Character Scene will look like the image below



The majority of the scene is the same except each stat will have a plus and minus beside them for the Player to increase or decrease the stat.

* The RemainingSkillPoints Label will display how many skill points the Player has left to assign
* The ConfirmButton will allow the player to confirm their choices
* The CancelButton will allow them to leave the scene without assigning any points.