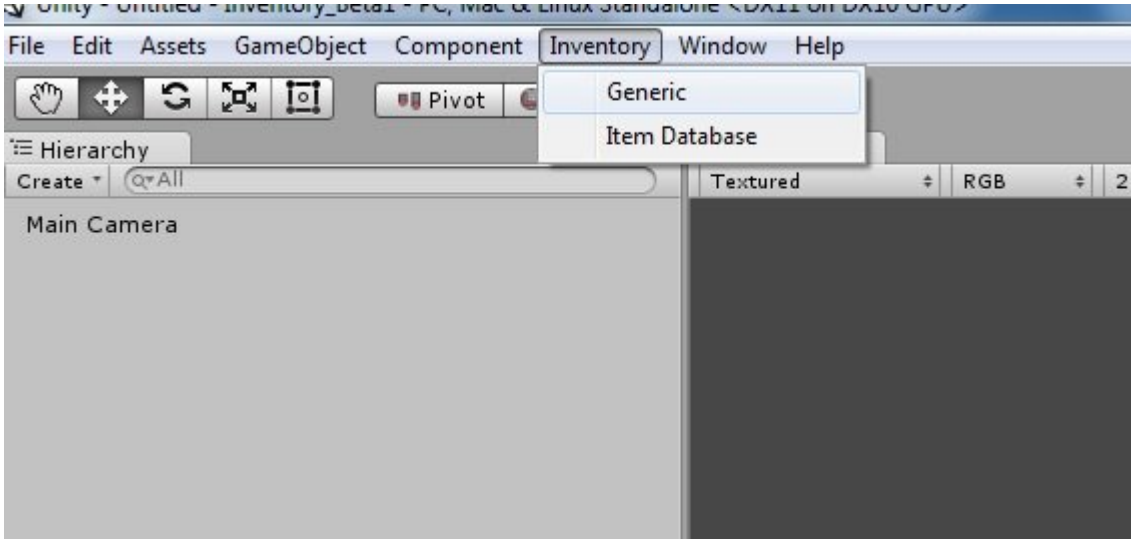
Documentation

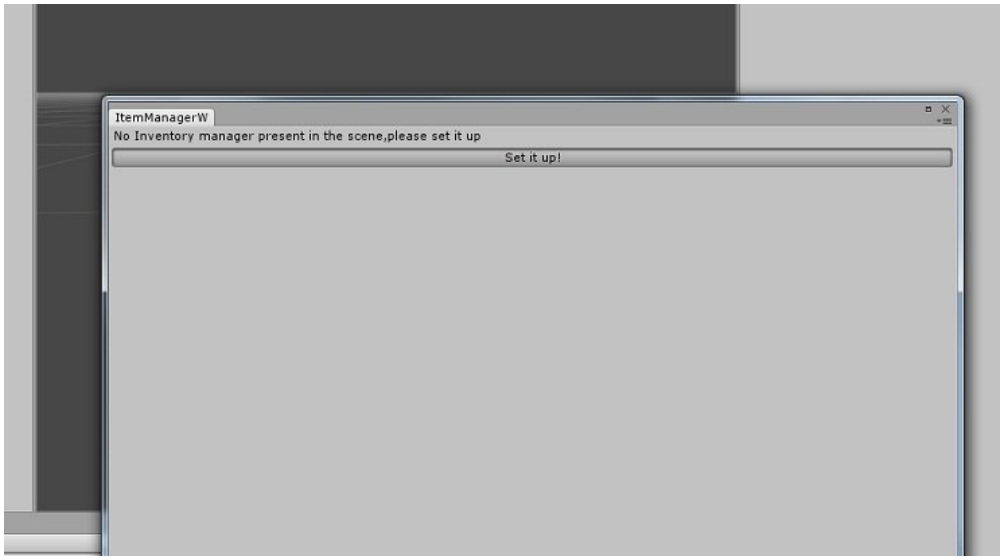
-Chris Paul-

Chapter I – Installing the inventory

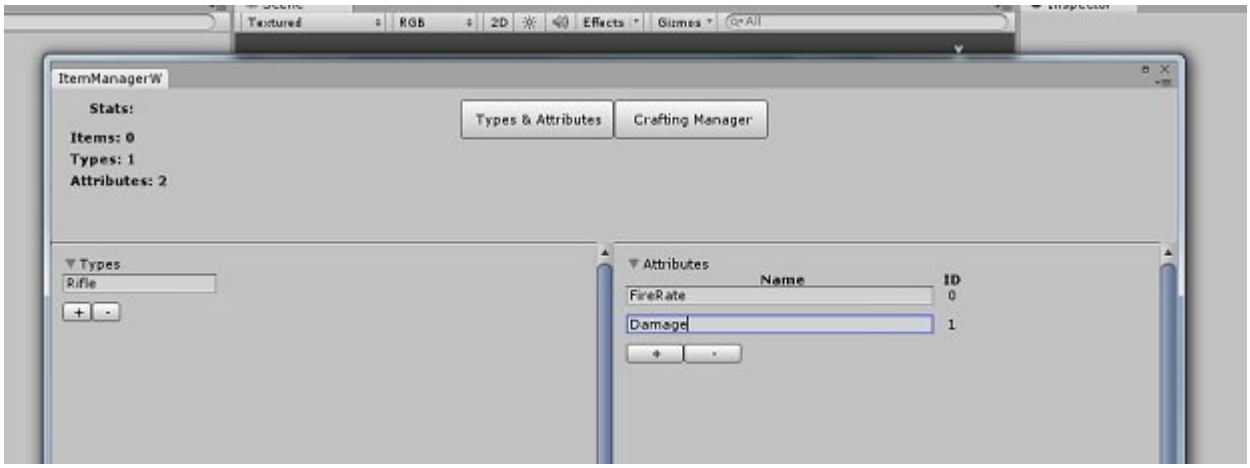
- The first step to installing the system is to enter the “Inventory” menu,and select any option from there:



The second step is to click the “Set it up!” button:



After clicking the button,the system will automatically create a Game Object called “InventoryManager” with a component with the same name,that is the heart of the system,which contains all the item definitions,type definitions,and also attributes,like “Power”,or “Attack Speed”,or “Fire Rate”,all of them customizable.



In this moment,you can start the creation process.

By clicking “Inventory/Generic” you can acces the editor for types,attributes and crafting,and by clicking “Inventory/ItemDatabase” you can acces the database and start creating the items.

Chapter 2 – Windows,containers,slots

-Slot-

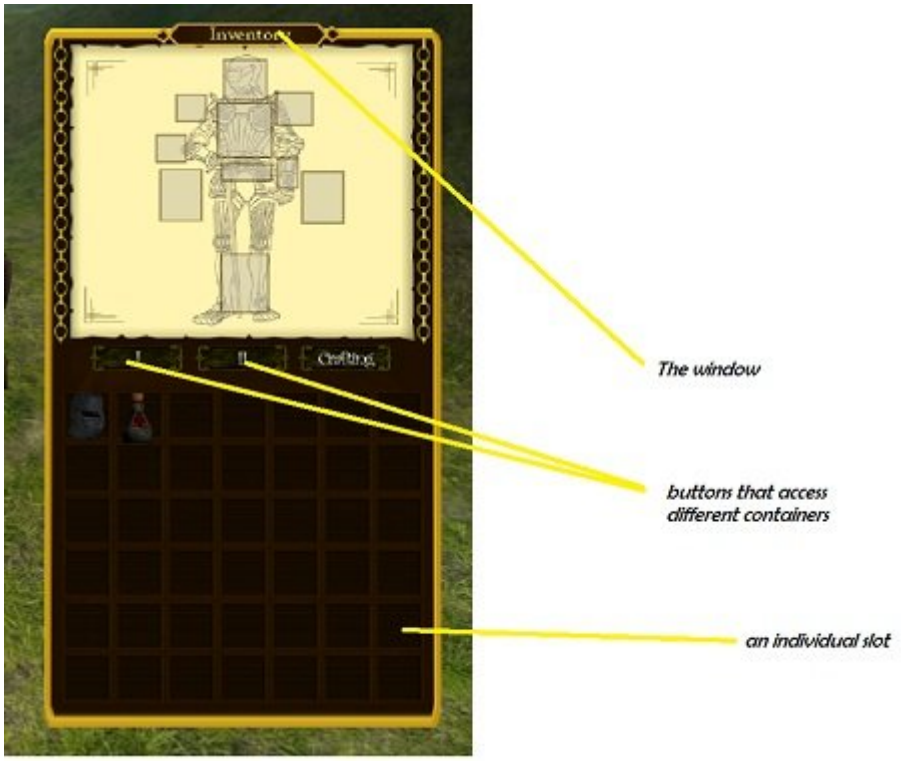
- The system lives around the concept of “slot”,in my inventory system,a slot is a game object with a “Slot” script attached,it also has an “Image” component for displaying the graphic of the slot.

A slot can be placed anywhere in the scene,and have any component attached,or children.With a slot being a game object,this is what brings a lot of flexibility to the system,since the slot it’s entirely customizable by the user.

Slots can receive input messages,like clicking,hovering,dragging,and so on...

-Containers and windows-

- Slots must be contained by the container,and the containers must by contained by a window.So the hierarchy is : “Window” – “Container” – “Slot”.



So when you want to create a layout like this,you create slots from the main menu,then you place them,more likely you will use automatic layout that Unity GUI provides,and then you create a container and place the slots in the container,maybe repeat the process for multiple containers,and then finally you create a window and place the containers in it.

Containers are game objects with a “Container” script attached.

Windows are game objects with an “InventoryWindow” script attached.

You can use this system to make FPS inventories,just like for making RPG inventories,it doesn’t matter,as long as you follow this hierarchy,everything that you build on top of this system is entirely your responsibility.

You will find out in the next chapters why you need to follow this hierarchy of InventoryWindow-Container-Slot.

Chapter 3 – Adding features

By now,you've already created the items and also set up the look of the windows,the buttons,and the slots.

The system handles the core aspects related to an inventory,like storing items,modifying,dragging and dropping,splitting,sounds when you consume items,events to happen when you are using one,and so on;

It would be impossible for me,and for any person or team,to make an inventory system that can satisfy every single user,this is why I build a solid,abstract system that you can build easily on top of it.

- THE EVENT SYSTEM-

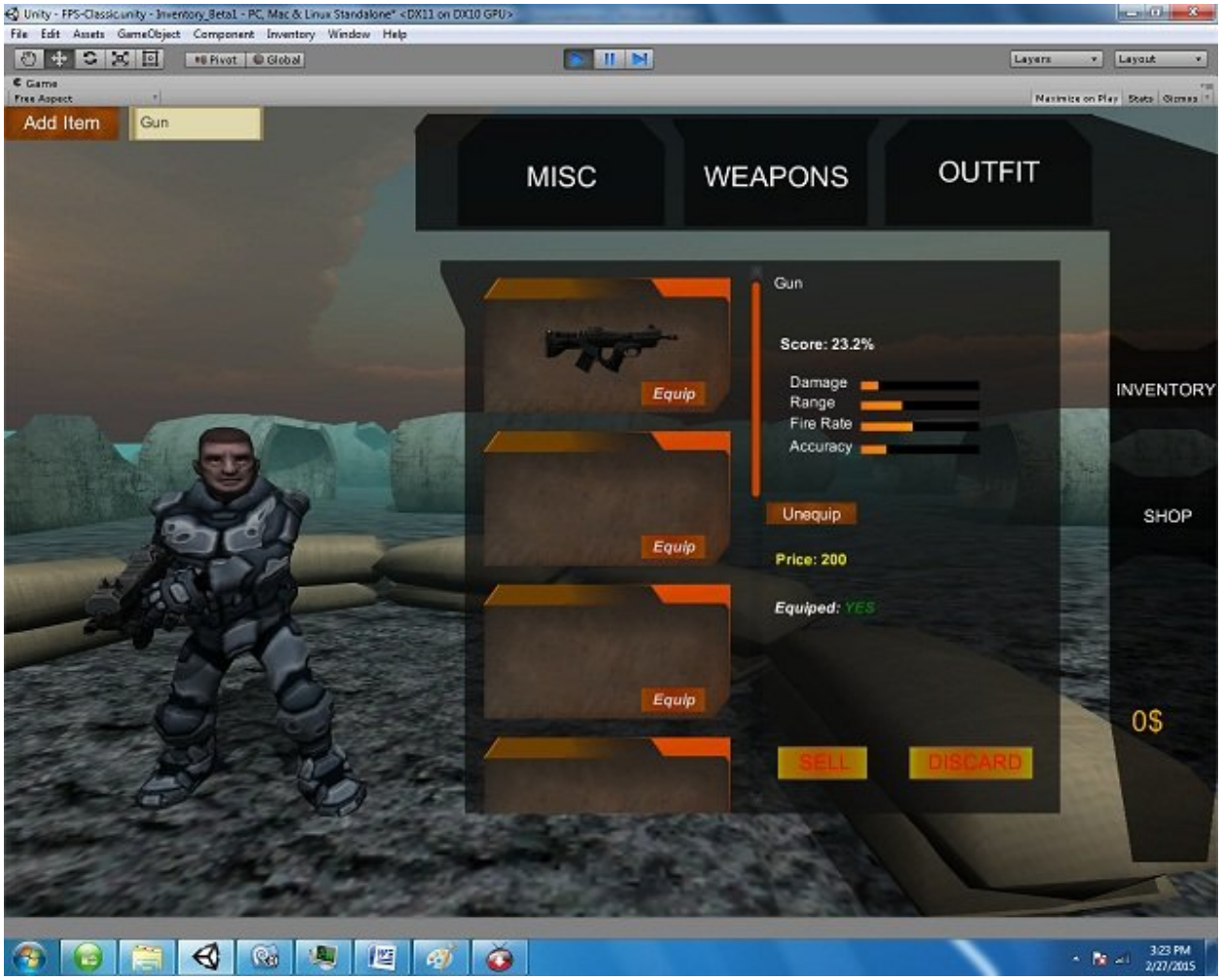
- Every time you move an item,delete it,swap it,add it or start dragging one,or maybe just hovering over a specific slot,that slot is firing some events,what does that mean? ...

Suppose you are doing an FPS inventory,and you have a button on every slot,that button is for equipping the item that is in that slot,here is how you do it:

-you create your regular button(hope you know how to do that)

-you attach it to the slot,make it a child of it

-trigger something when you fire up that button



Suppose you have a script that handles *equipping an item*, the button component that comes with Unity 4.6 +, allows you to select functions that you want to invoke when you press that button,so you select your function that handles equipping and you're good to go.

THAT EXAMPLE WAS SIMPLE...you were just using the button component,but consider the next one:

- You also want to display the name of the item somewhere on the slot,you can't just make a button and press it to display the item name,that won't work,what you can do is make a Text UI,make it a child of the slot,and then

make a script that checks the parent slot for it's item,when the item changes,or there is no item in the slot,you display something with the help of the "Text" component.

BUT, how do you know when you have to change the text? You will need to constantly check for that,in an Update() function...

That's bad,it would be nice if there is a way for us to change the text just when the parent slot changes,because if you have a lot of those in your game,I can guarantee you that it will slow everything down.

The system comes with an event system that fires up everytime something changes.

- The SLOT component has a few events that are triggered depending on what is happening,for example,if you want to update a text that displays the item that's currently in a slot,you just subscribe your function that updates the text,to that event.

EXAMPLE:

```
using UnityEngine.UI;
using UnityEngine;

public class MyScript : MonoBehaviour {
	public Text displayer;
	public Slot parentSlot;
	void Start() {
		parentSlot = GetComponentInParent<Slot>();
		parentSlot.OnChange += UpdateText;
	}
	void UpdateText() {
		Text.text = parentSlot.ItemName;
	}
}
```

- So,once again,

We want to display the item name in each slot,we do that by:

1.Creating a game object,and attaching it to the slot,making it a child of the slot.

2.Attach a "Text" component,and your script to the game object you have created.

3.In the script,we subscribe our function that updates the text,to the slot's event,with this line:

parentSlot.OnChange += UpdateText;

“parentSlot” is the reference of the slot that we’ve got in the Start().

“OnChange” is the event that fires up every time something changes in the slot.

“+=” means that we are adding our function on top of the existing ones that are also “subscribed” to that event,so that means you can subscribe as many functions,or methods,as you want to that specific event,and all the functions will execute when it happens,without you explicitly calling the function/method.

Chapter 4 – basic functionalities

The following are the functionalities that are provided by the system:

1.Adding an item – adding an item to a window it’s achieved by calling a function of the “InventoryWindow” script,called *AddItem().*

You can add an item by specifying it’s id,and the amount you want to add,if the item it’s not stackable,the amount doesn’t matter,it will always be 1,even if you specify something different.

2.Removing an item – removing an item from a window it’s also achieved by calling a function, called RemoveItem().

You can also specify the item id and the amount that you want to remove from the inventory.

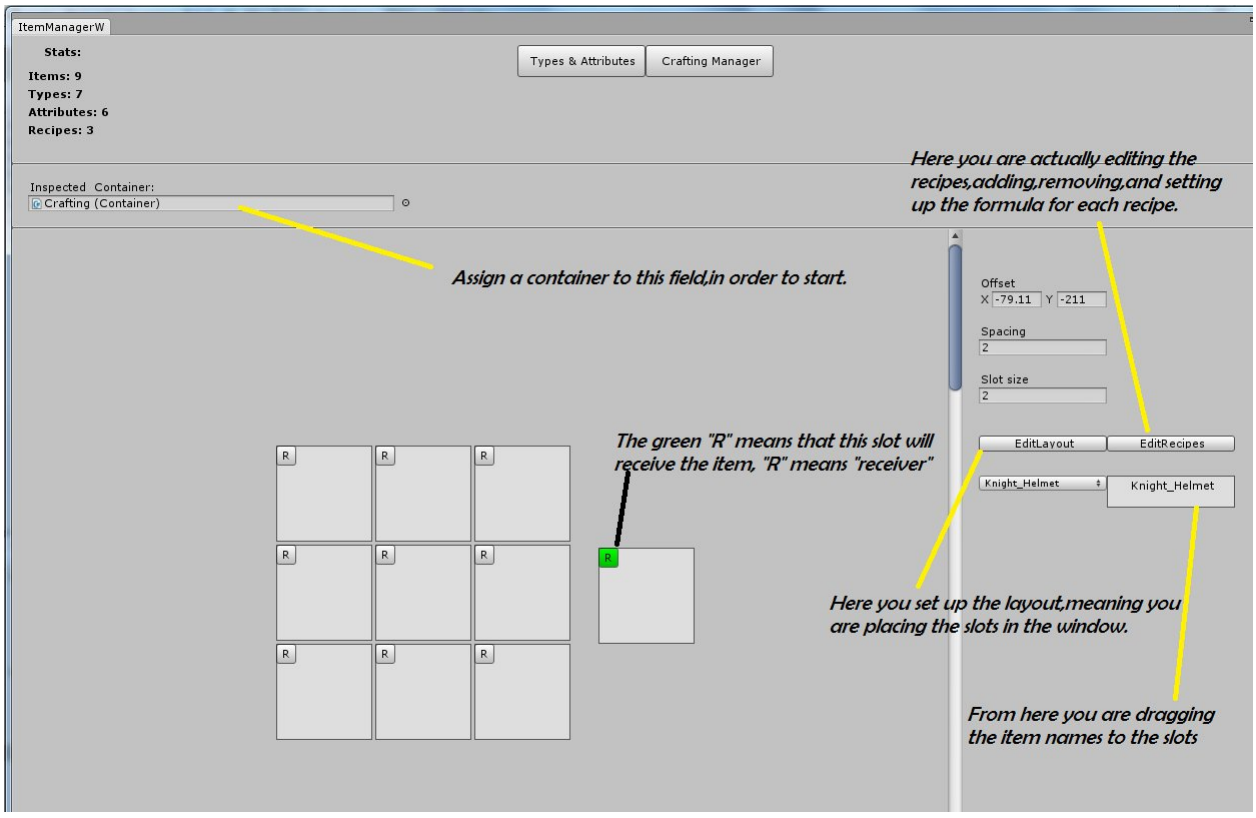
3.RemoveAll() – this function removes all the items from a specific window.

4.Contains() – this one searches for the existence of a specific item in the inventory,if it finds a single item that matches,it will return *true.*

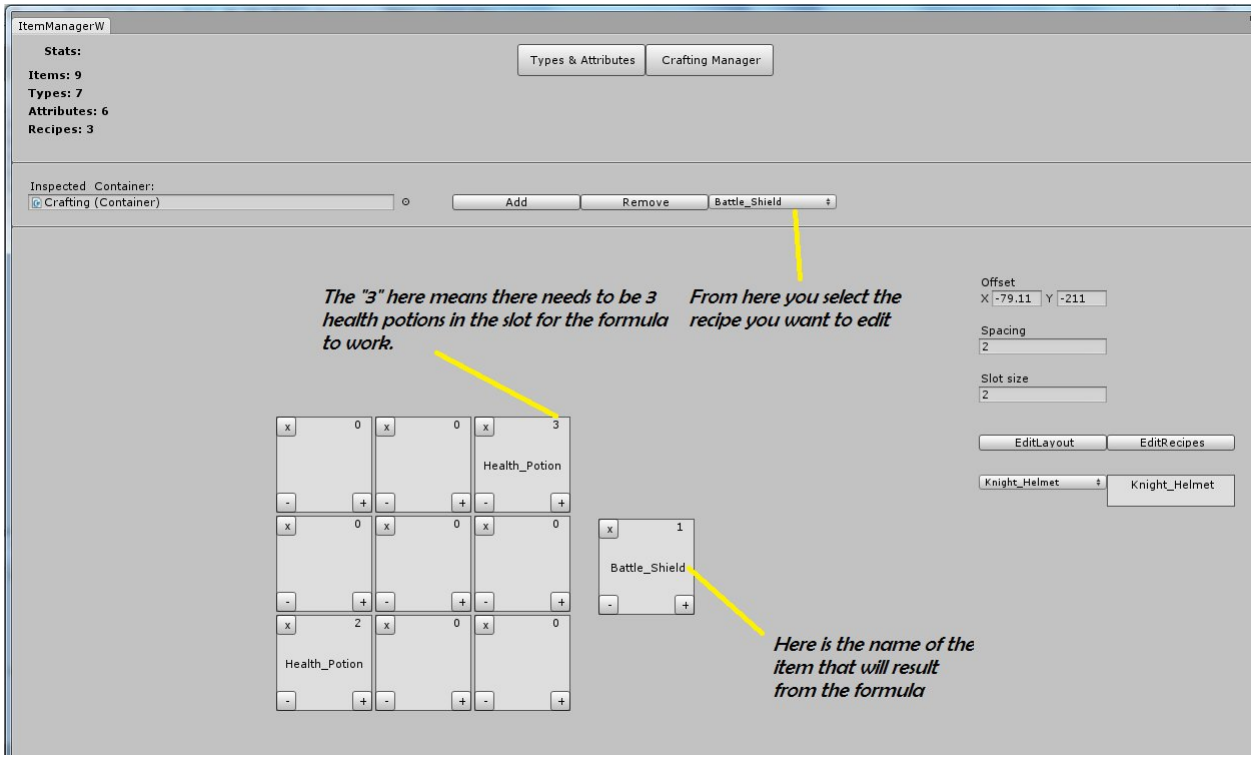
5.HowManyContains() – this method returns the number of items found with a specific id.

Chapter 5 – the crafting system

"Editing the layout"



“Editing the recipes”



FOR ALL THE SCRIPTING REFERENCE THAT IS AVAILABLE WITH THAT SYSTEM,PLEASE CHECK OUT THE .PDF DOCUMENT CALLED “ScriptingReference” THAT IS PROVIDED WITH THE PACKAGE.

This documentation represents a simple guide to starting using the inventory system,for more detailed tutorials please check out the video tutorials on my channel on youtube,and also the scripting reference.

- *My name is Chris ,I am the author of the inventory system, for any bug,information,tutorial and feature request,you can send an e-mail to cristianpavel40@yahoo.com , or comment on my youtube channel - ,and I’m*

going to try my best to debug the product,do a tutorial,or simply give you an advice.

Youtube channel - https://www.youtube.com/channel/UCPisFYJed_LBzF7fXDllHfg?spfreload=10