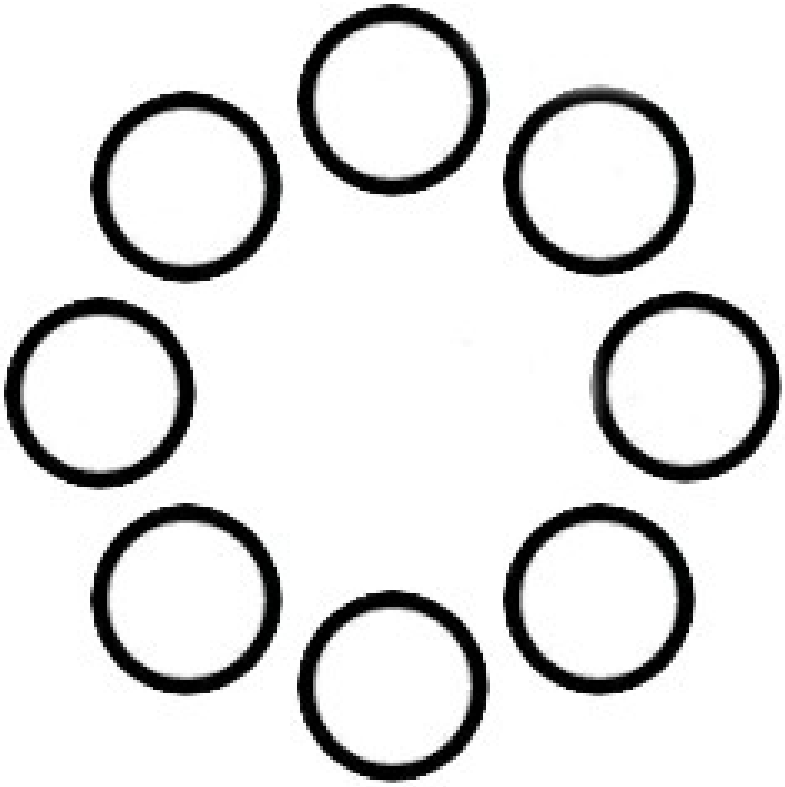


Custom Radial Menu

by Mr.Cojo Entertainment

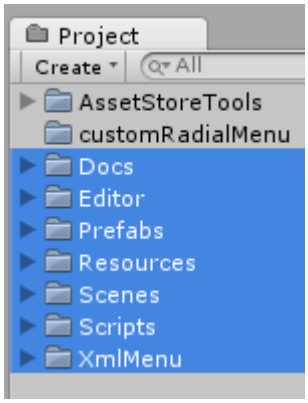
User Manual

Index

1.Introduction

First of all thank you to have chosen my radial menu as an asswet for your project!
Welcome to the user manual of this customizable radial menu.
Reading this document will help you to set up your own radial menu in a few clicks. The implementation and functionality of the menu itself then, will be coded by you, but feel free to dig into the example I created and use it as a reference for your projects.

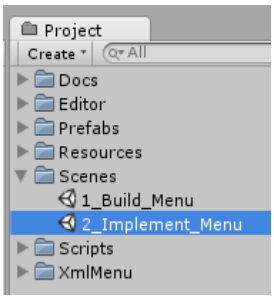
Note that the first thing that you have to do is extracting every sub folder of the **customRadialMenu** folder and put them in the root folder of the **Project** window (just drag the folders out of the **customRadialMenu** folder, then you can also delete the **customRadialMenu** folder, which should be empty by now), as you can see here:



So, now that you set up your project...let's start!

2.First of all...Play it!

The best way to understand how this menu works is using a pre made menu of course! I made a scene which you can play just to see how the menu looks like and works.
The scene is called **2_Implement_Menu** and you can find it in the **Scene** folder of the **Project** window of Unity.

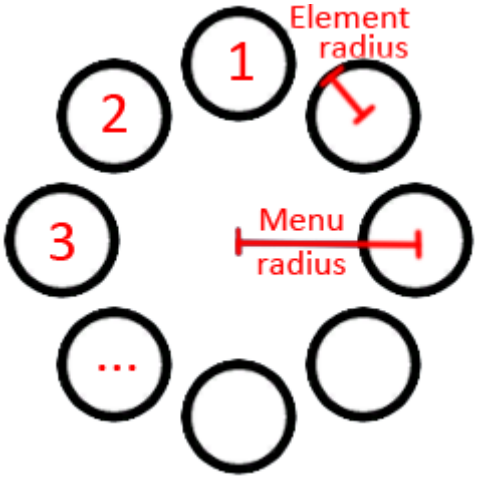


Open it and play it!

It is reccomended to play this scene at least once, in order to be able to understand better the next chapters of this manual!
Enjoy!

3.Basical theory behind the radial menu

I will go through this very quickly because is really something super easy. Basically i want you to understand how the menu is intended to be built and used.



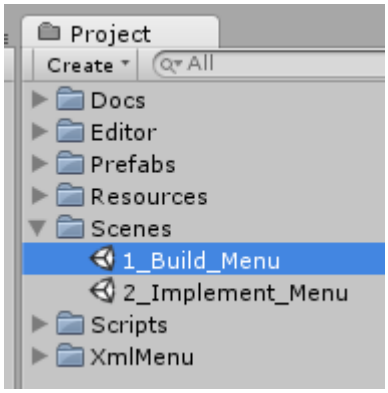
If this picture is not enough explanatory, the concept are basically three:

- The number of the elements starts from the top one and increase counterclockwise.
- The element radius is the value of the radius of the element itself.
- The menu radius is the distance between the center of an element and the center of the menu itself.

That's it!

4.Step 1: Building your own radial menu

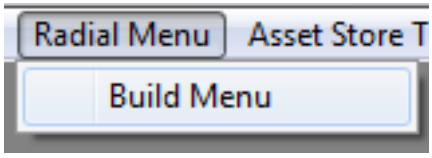
After opening the project in Unity, open the **Scene** folder, then open the scene **1_Build_Menu.**



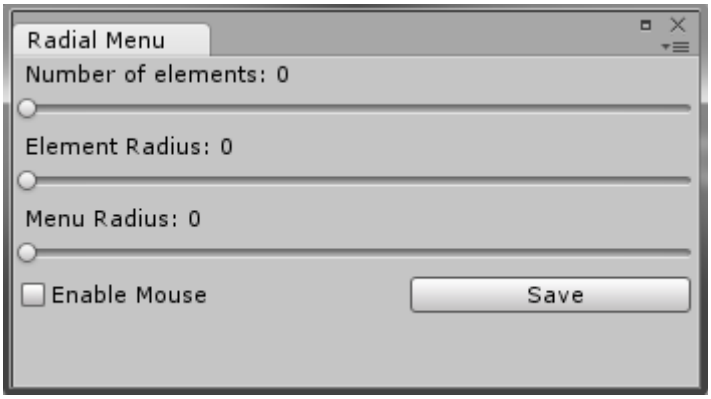
Now, you can press the play button 
in the Unity interface and start building your menu from this scene.

Note that is important to play the scene with the **Maximize on Play** option **disabled** (or you won't be able to assign your own textures to the elements of the menu).

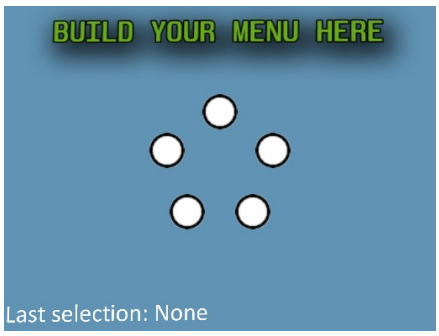
Now that the building scene is playing, to build up your menu you have to open the editor menu **Radial Menu → Build Menu.**



The window that will pop up will let you build the menu as you wish.

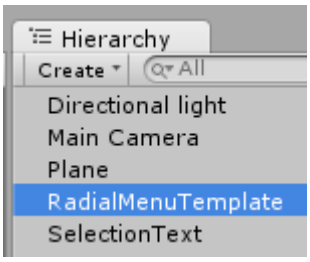


As you change the values of the sliders, you will notice that the menu will start to create in the scene, like you can see here:

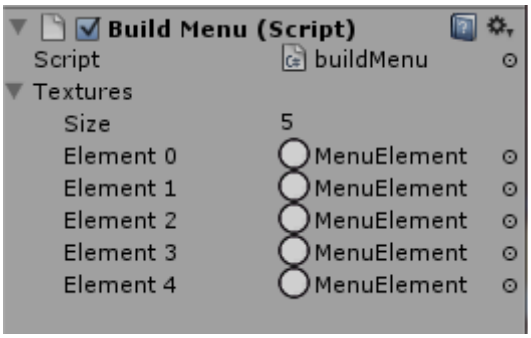


At this point, be sure to be satisfied with your selections, and you can already start dragging any image you want on every menu element. **Please, mind that if you change the number of elements value, the menu will re-draw showing the standard menu element texture, so if you made any change on the textures of the elements and then you change the value of the number of elements, you will lose your changes!**

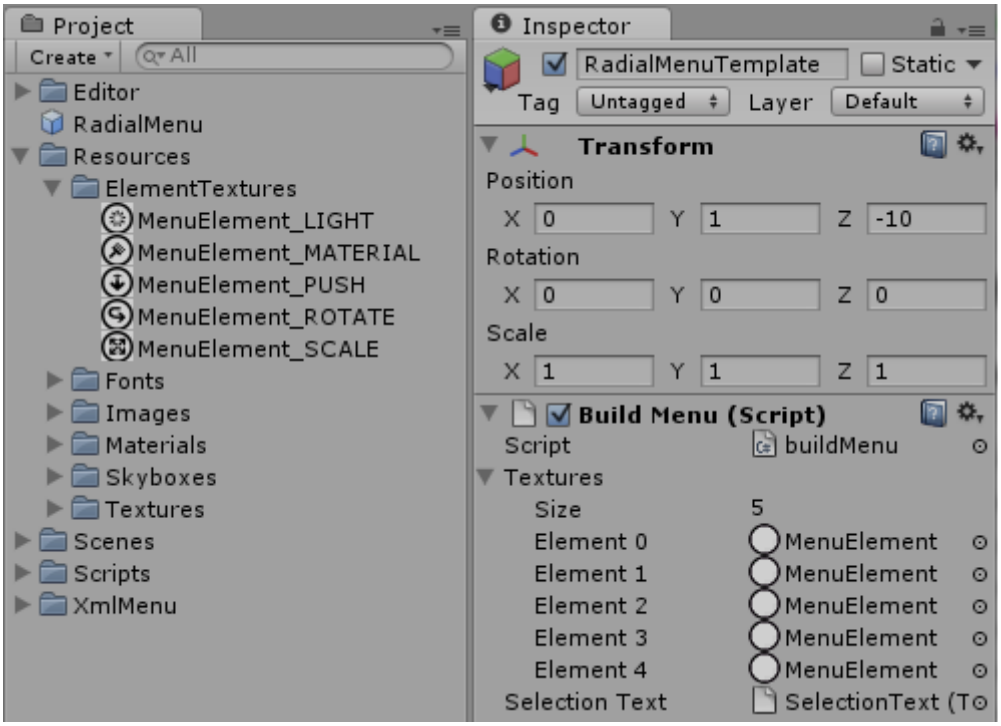
So now that you are happy with the shape/dimension/etc of your menu, you want to start adding your images to your menu. To do this, first select the RadialMenuTemplate object, form the **Hierarchy** window.



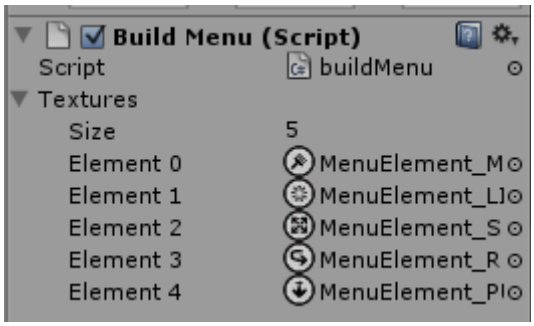
Be sure also, that in the **Inspector** (inside the **Build Menu** script attached to the object) the array of textures that you have to fill in is expanded (if not, click on the triangle to expand it)



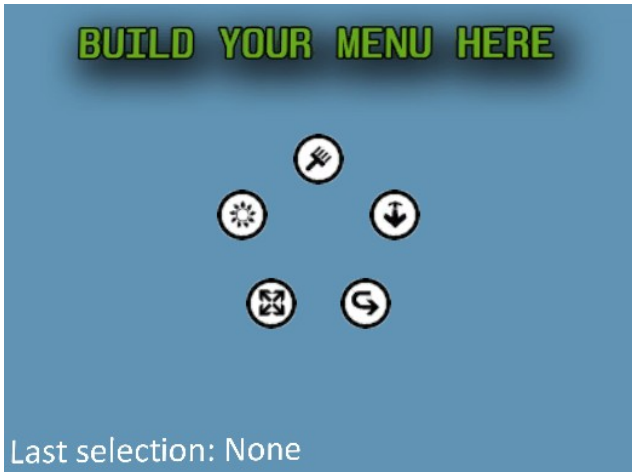
You should now have the editor in front of you. Personally, i like the 2 by 3 view like this:



As you can see, in the Project window, under **Resources->Element Texture** are stored the images of the example menu I made.
Note that you have to put your custom images into this folder (or you can always change the code to refer over the folder you desire).
If you don't have the images you used when building the menu in the right folder, a message in the console will be displayed anyway.
Now, the only thing you have to do, is dragging the images from the **Project** window, into the array in the **Inspector**, as you desire.



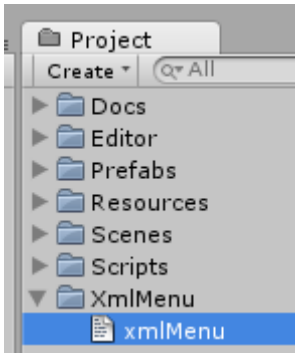
While you are doing that, you can already notice that in the scene, the menu is also changing.



When you are done with this, you can start test your menu also in this building scene. Activating the **Enable mouse** toggle, in the **Build Menu** window will enable the mouse on the scene, so by pressing the **Right Mouse Button** you will show the menu, and releasing it over an item you will select it (as I hope you already tried in the test map in the beginning). When a selection is made, you will notice it on the text in the bottom left corner of the scene.

When you are satisfied with everything about your menu, you are ready to save it, pressing the **Save** button of the **Build Menu** window.

When you do so, a XML file (named XmlMenu.xml) will be created in the **XMLMenu** folder. This file stores all the information of the menu that you hae just created (so this is pretty important).



Please note that if there is already a file named XmlMenu.xml in the XMLMenu folder, it will be automatically overwritten with your new settings, every time you press the **Save** button.

My suggestion is to save a copy of your XML files, anytime you produce one (maybe giving them a significant name, so you can recognize them at first sight, in the operating system).
So, whenever you want to "import" a new menu, you just have to:

- Copy the XML related to that configuration into the **XmlMenu** folder.
- Rename it as XmlMenu.xml.

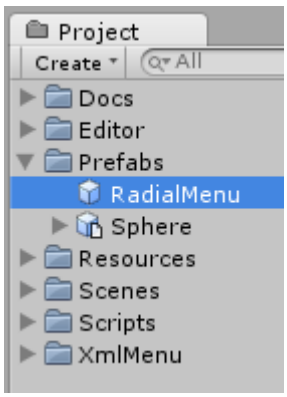
Well done! You created your own radial menu! Let's find out how to implement it into your project now!

5.Step 2: Implement your menu inside your project

Now, you want to use your own menu for your needings. To implement your own menu is so easy. It's really just a couple of clicks!

The only thing that you should mind before starting, is to have a XMLMenu.xml file in the **XMLMenu** folder. If you don't have it, a message in the console will be displayed anyway.

Ok, so now what you have to do is to implement your own menu.
Copy the **RadialMenu** prefab (that you can find in the **Prefabs** folder of the project) inside your scene.



This prefab conatins a copy of the script **showRadialMenu.cs** (which is basically the “core” of the radial menu itself), so to make the menu working, you should add this prefab in every scene you want the menu to be displayed.

Now, adding this prefab will show the menu. But it won't handle the selections you make with it.
To handle the selections basically you have to check if the Right Mouse Button is released, and if so, use the method **getSelection()** of the **showRadialMenu** script, to get the selected item index.

We could basically say that the following code is the one you should use in your script to handle the selections:

```
//Checks if RMB is released
if(Input.GetMouseButtonUp(1))
{
	//Checks what selection was made by the user
	switch (showRadialMenu.getSelection()){
		case 1: //Do something
			break;
		//		… add all the cases you want

		default: //Do nothing
			 break;
	}
}
```

But this is really up to you! Your creativity is the limit!

There are really different ways to handle the selections.
You can edit the script **showRadialMenu.cs** itself or (as I did in the example I provided, and I think it is more elegant) you can have a separated script which “cathces” the selections every time you make one and then, act as desired.

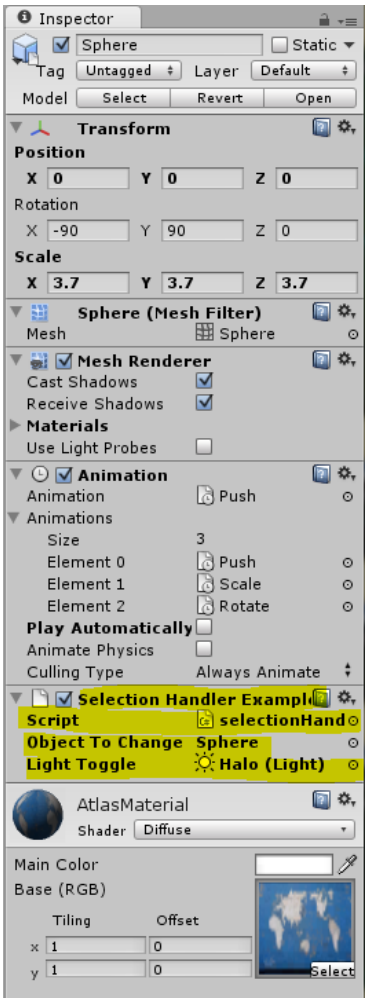
6.My implementation example

I will quickly show you how I handled the selections in my test scene:

Ok so, as you already noticed I have a **Sphere** which reacts over the selections I make on the radial menu.

To achieve it, first I created a script, named **selectionHandlerExample** which is the script that catches the selections I make on the radial menu, and for each specific case, it does something different.

Once i coded the handler, then what I did was just dragging the **selectionHandlerExample** script over the **Sphere** itself (and of course I also dragged the elements that actually interact with the menu over the relative variables of the script itself), as you can see here (highlited):



Feel free to dig into the **selectionHandlerExample** script and use it as a reference to build your own “handlers” for your menu.

7.Congratulations

Congratulations! You created your own radial menu and understood how to use it in your games/applications in Unity!

Good luck with your projects! If you liked my product then, support me! sites.google.com/site/mrcojoentertainment

Thank you!