Off Screen Indicator. Instructions

- Package Content
  - DemoScenes
  - Scripts
  - Scripts / FirstPersonVRCharacter
  - Scripts / Helpers
  - Sprites
- Using the package from scratch
- Adding and removing Indicators in real time
- How to use VR (tested with Oculus)

Package Content

DemoScenes

DemoScript.cs - Example code for demos, show how easily you can add a offscreen arrow in realtime via code.

FPSCanvasTest.unity - Demo Scene, where you can see how it works OffScreenIndicator with a FirstPersonCharacter Standard Asset, and displaying the arrows in Canvas.

FPSVRTest.unity - Demo Scene, using VR (Tested in Oculus Rift DK2) instead of Canvas.

ThirdPersonTopViewTest.unity - Demo Scene, using ThirdPersonCharacter with a cenital top view, and displaying the arrows in Canvas.

Scripts

Core classes.

Scripts / FirstPersonVRCharacter

You can remove this folder if you are not using VR. In this folder you can find the FirstPersonCharacter Prefab modified to work with VR, and display arrows in world space in a given distance to the viewer (easily configurable). The MouseLook is modified to change only character rotation, and pov camera will be moved only with the VR device.

Scripts / Helpers

Convenience methods for debugging.

Sprites

Example sprites used in the demo scenes.

Using the package from scratch

https://youtu.be/UdVvfvWVRME

1.Create empty 3D Project
2.Import OffScreenIndicator package
3.Add a plane for the ground
4.Add a cube to use as target
5.Create a Canvas adding for example a Text
6.Create empty GameObject to add Logic
7.Locate the script OffScreenIndicator/Scripts/OffScreenIndicator.css
8.Add it as a script component to the empty GameObject
9.Let's add one Indicator
10.Set color and sprites for the Indicator when the target is on screen and off screen, if you don't set sprite it won't be displayed
11.Add the cube as a target, the IndicatorID is the desired element of the Indicators Array
12.Set the Canvas object in the script component
13.Press Play, and see how the Indicator in the canvas follows the target
14.You can force the sprite to rotate pointing to the target
15.Let's add the FPSController from Standard Assets
16.Disable the MainCamera and verify the FPSController Camera is tagged as MainCamera
17.Move Character Controller and check how the Indicator works
18.You can change the transition from the onscreen sprite to the offscreen sprite, as the duration
19.You can change the indicator size and the padding to borders

Adding and removing Indicators in real time

https://www.youtube.com/watch?v=XO79GIaVLg0

1.Let's add a button to add a new target
2.Create a new script to add the logic for the button
3.This code is included in the DemoScenes folder
4.This is the important stuff, the method:
AddIndicator(Transform target, int indicatorID)
adds the indicator of the given transform using the Indicator id.
5.Now link the button with the script
6.Let know the script which object to instantiate
7.Tune the OffScreenIndicator parameters to your needs
8.You can create all the indicators you want with their given colors, sprites and transitions
9.You can add an offset applied in the Canvas space (pixel units)
10.Let's use a random indicator in the AddTarget script

How to use VR (tested with Oculus)

https://www.youtube.com/watch?v=pjBmPCPvu64

1.Locate the prefab
OffScreenIndicator/Scripts/FirstPersonVRCharacter/Prefabs/FPSVRController.prefab
2.Remove the old FPSController. Place FPSVRController on the scene
3.Go to File > Build Settings > Player Settings > Other Settings and check the Virtual Reality Supported checkbox
3.The component OffScreenIndicator must have checked the Virtual Reality Supported checkbox
4.In the scene window you can see the debug information of the OffScreenIndicator, the green plane is where arrows are drawn
5.Configure VR parameters to prevent the clipping, remember that the UI in VR has to be in the world space