Documentation

I Overview

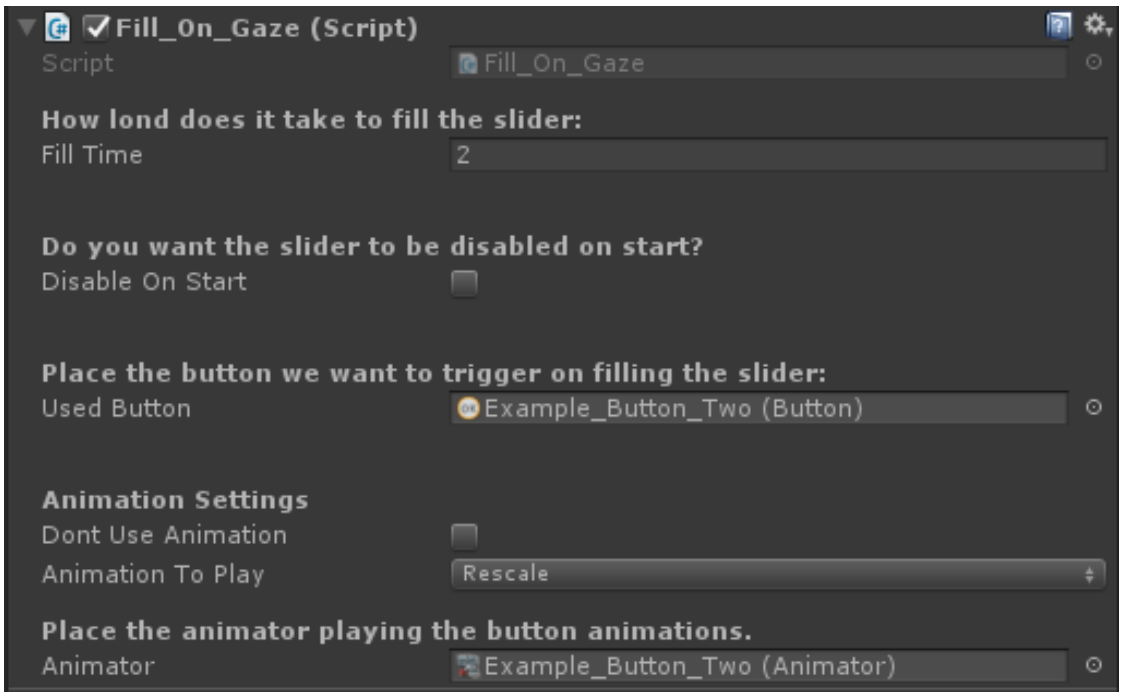
"Gaze Filling Slider Lite" helps you bring your gaze input to another level, allowing you to implement a great slider filling functionality into your project. It does not matter if you want to use it for a menu, a dialogue system, a game mechanic or something else, "**Gaze Filling Slider Lite**" will fit perfectly in your project. All the scripts are easy to read and full of comments to help the non-coders understand the basic principles that apply in the asset.

II Quick Setup

1. **Download the asset from the Asset Store.**
2. **Import it into your project.**

Note: If you see compile errors in your console after importing, that means you probably don't have the Oculus Utilities package already imported. After you do that, the errors will dissapear.

3. **Attach the "Fill_On_Gaze" script to the object that has a "Slider" component attached.**
4. **Set the time for filling the slider in the "Fill Time" field.**

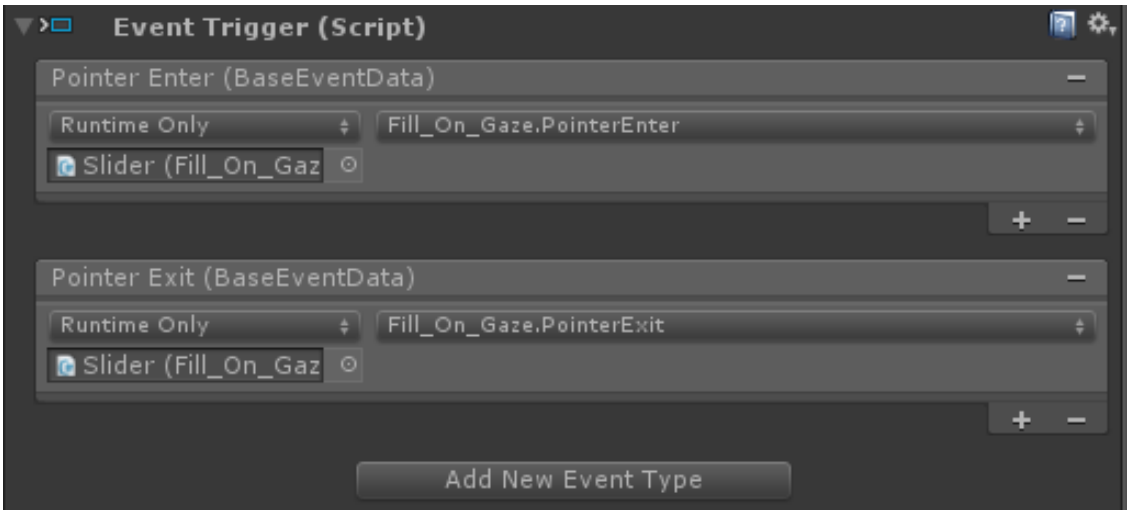


5. **Place the button you want to gaze upon in the "Used Button" field.**
6. **Click on "Disable On Start" if you want the script to automatically deactivate the object holding it on Start().**
7. **Click on "Don't Use Animation" if you dont want any animation to be played on full slider.**

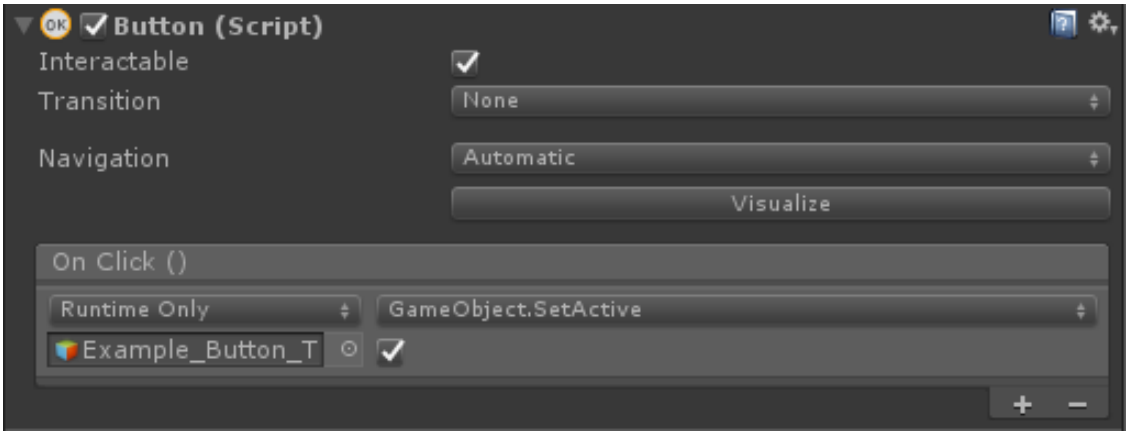
8. **If you stick with the animation routine, you can choose between three default animations - “ColorChange”, “Shake” and “Rescale”.**
9. **In the “Animator” field place the Animator component that is attached to your button object. If you don’t have one and you are not sure how to set it up, examine the example scene provided with this asset.**

Note: If you want to add a custom animation to the routine - Custom Animation

10. **On your button object, add an "Event Trigger" component. Once this is done, add two new event types - "OnPointerEnter" and "OnPointerExit". From the "Add to List" button add one field for both the event types. In them, drag the object holding the slider component and from the "Functions" panel choose "Fill_On_Gaze --> PointerEnter" or "Fill_On_Gaze --> PointerExit" depending on the event type.**

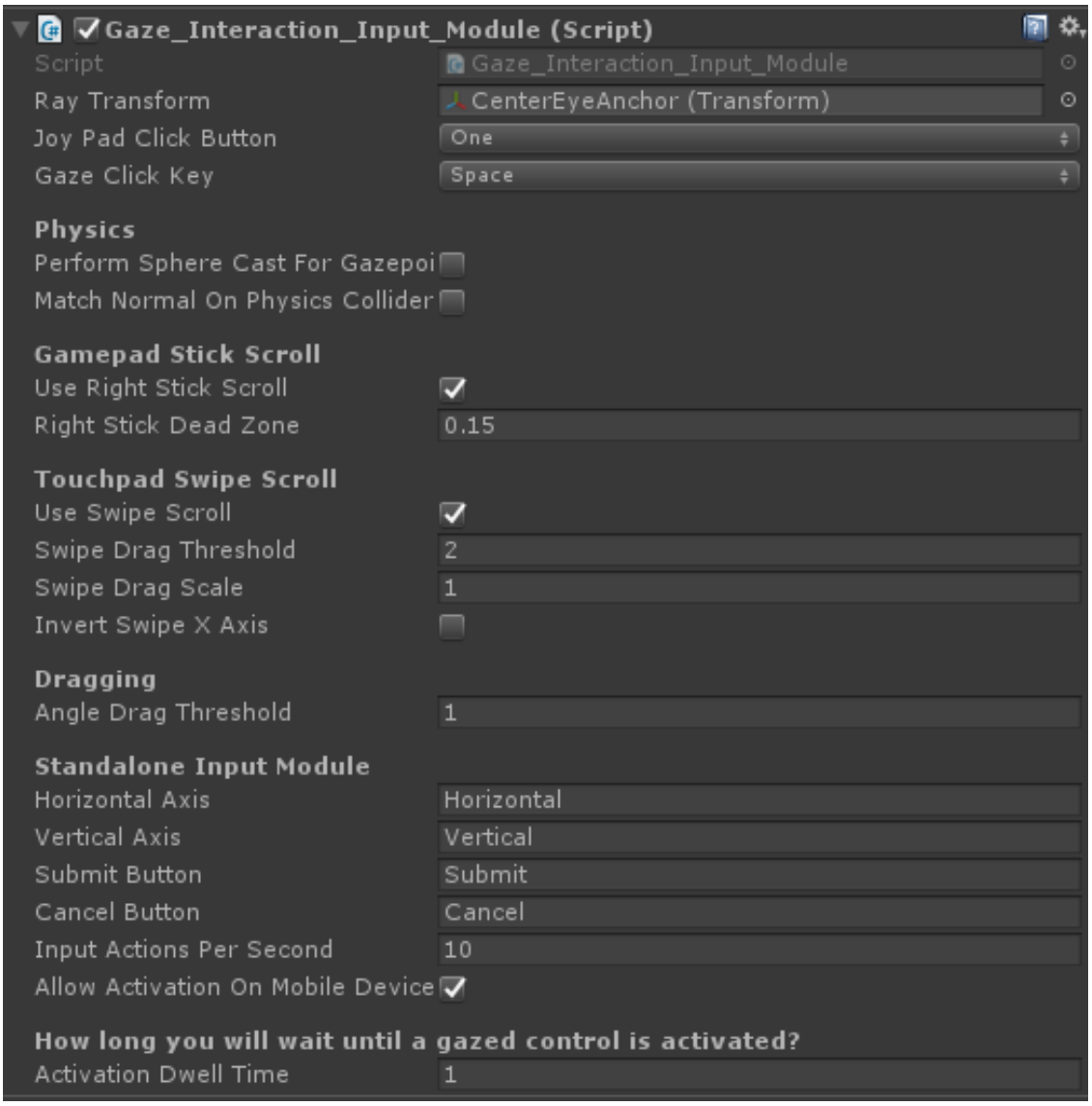


11. **On the "Button" component, add the action you want to be done into the "OnClick" event type field. This action/function will be done/invoked once the slider is filled.**



12. **In order for everything to work without a problem, attach the "Gaze_Interaction_Input_Module" script to the object holding the "EventSystem" in your scene. If you don't have one, now it's the best time to create it. In the "Ray Transform"**

field, place the object you want to start the ray/pointer from. Usually, this is the main camera that you are using.



13. **Finally, you want to have a Gaze Pointer in your scene. If the OVR Manager does not instantiate one at runtime and starts sending errors in your console, just create an empty object and attach the "Gaze_Pointer" script to it. This will solve the problem. For more cool looking and complex pointers, check out the documentation provided by Oculus.**

III Custom Animations

If you want to add a custom animation to the system, the first thing you want to do is to put it into the Animator that you've placed in the "Fill_On_Gaze" script. Then, open the script itself and go to line 27. In the brackets you can add a name for your new animation.

Example: public enum AnimationOptions { Shake, ColorChange, Rescale, NewAnimation }

On line 115 you will find the method "OnBarFiiled". Everything we want to happen when the slider is filled, we place here. From line 120 to line 131 you can see an if-block that is checking which animation have you chosen. You can add a simple condition check for your animation too.

If you want, you can delete the entire if-block and add a direct play to your animation. Check Animator.Play() for more information.

IV Contact us

If you are experiencing problems with the asset, you can reach us on e-mail – office@love2design.org. We usually respond within two business days. Thank you for your patience!