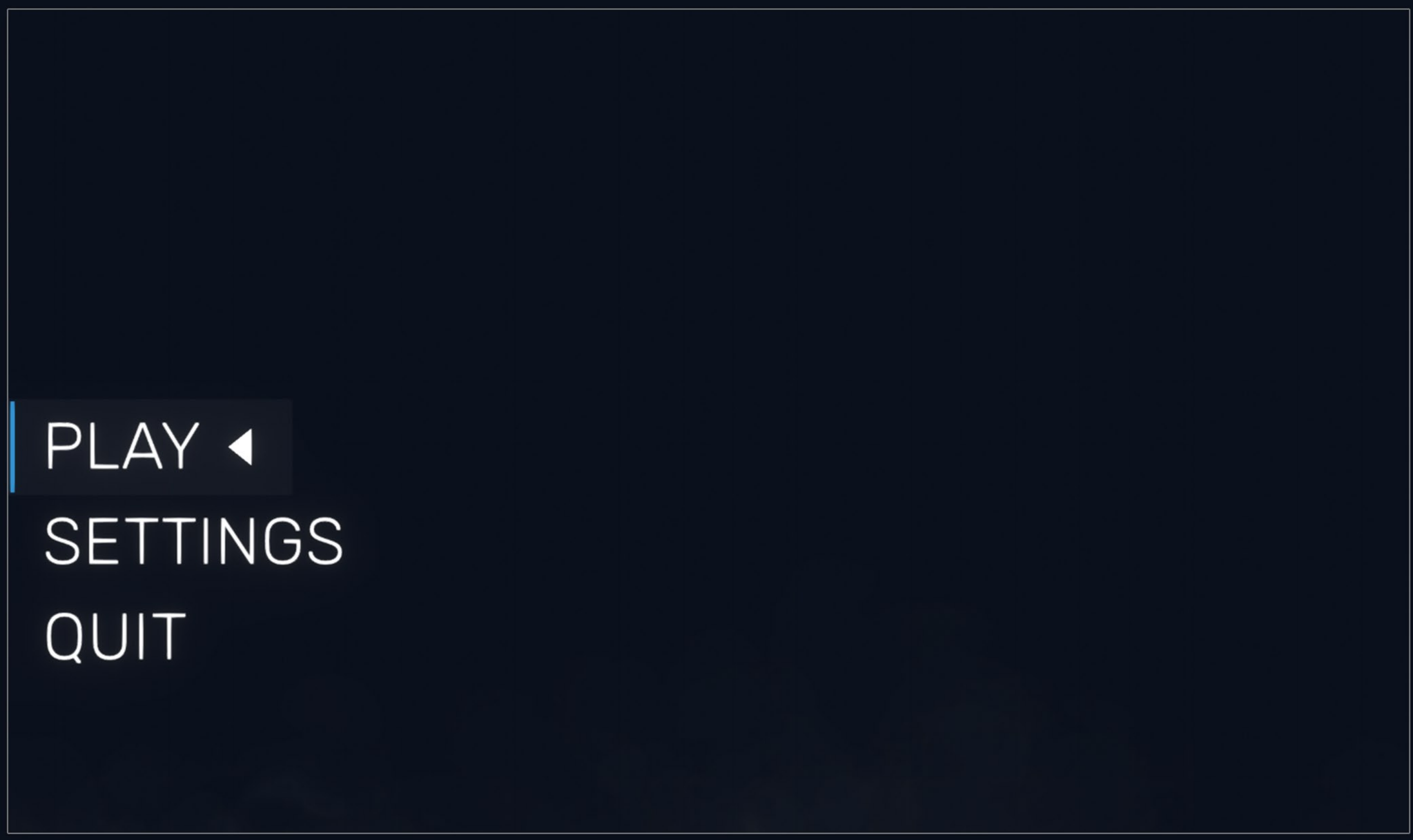
Clean Menu Documentation
V1.2

SLIMUI

1.0 INTRODUCTION

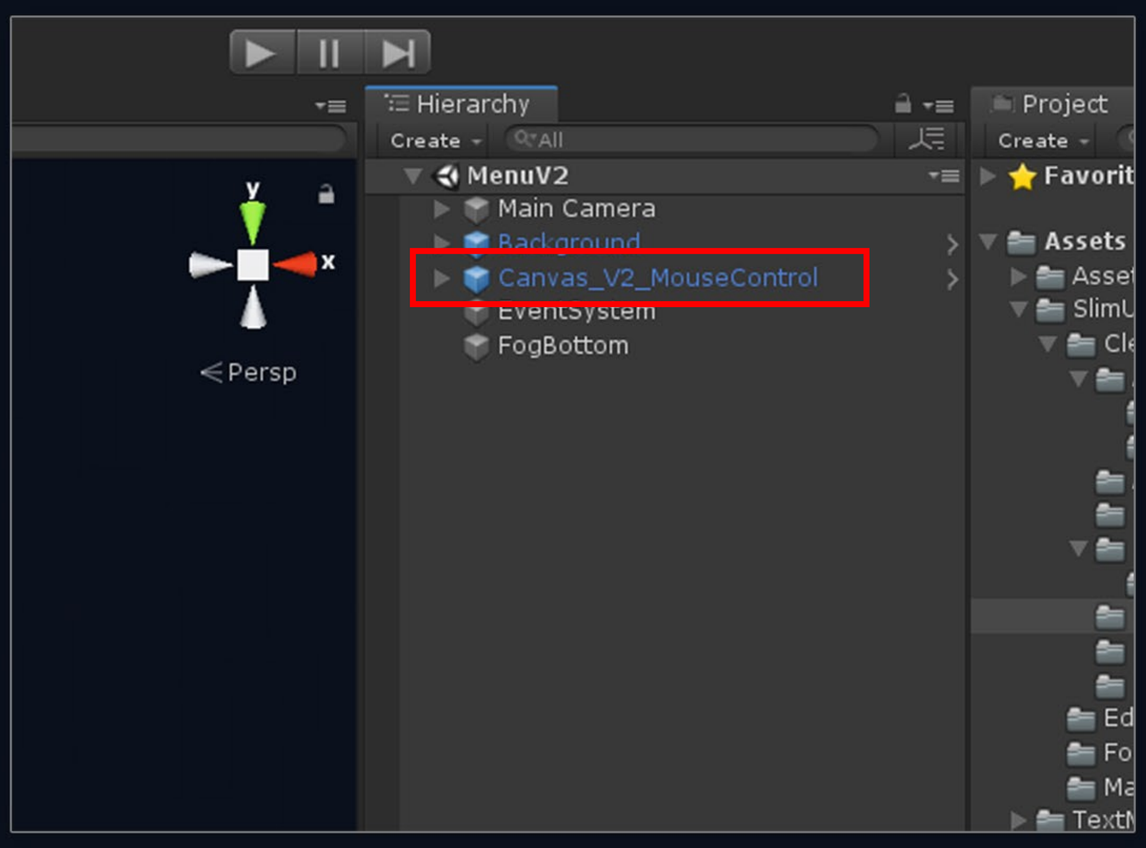
Clean Menu is a modern and sleek fully-functioning native uGUI main menu pack. Within minutes, you can set up the menu in your game. If you're upgrading to V1.2 from a legacy version, unfortunately they are not compatible because of the massive overhaul that this version is. It's refactored code, new prefabbed animations, and an entirely redone menu navigation layout built from the ground up using uGUI components.



For questions, comments, or feedback, make sure to join our Discord: **7cK4KBf**

1.1 GETTING STARTED

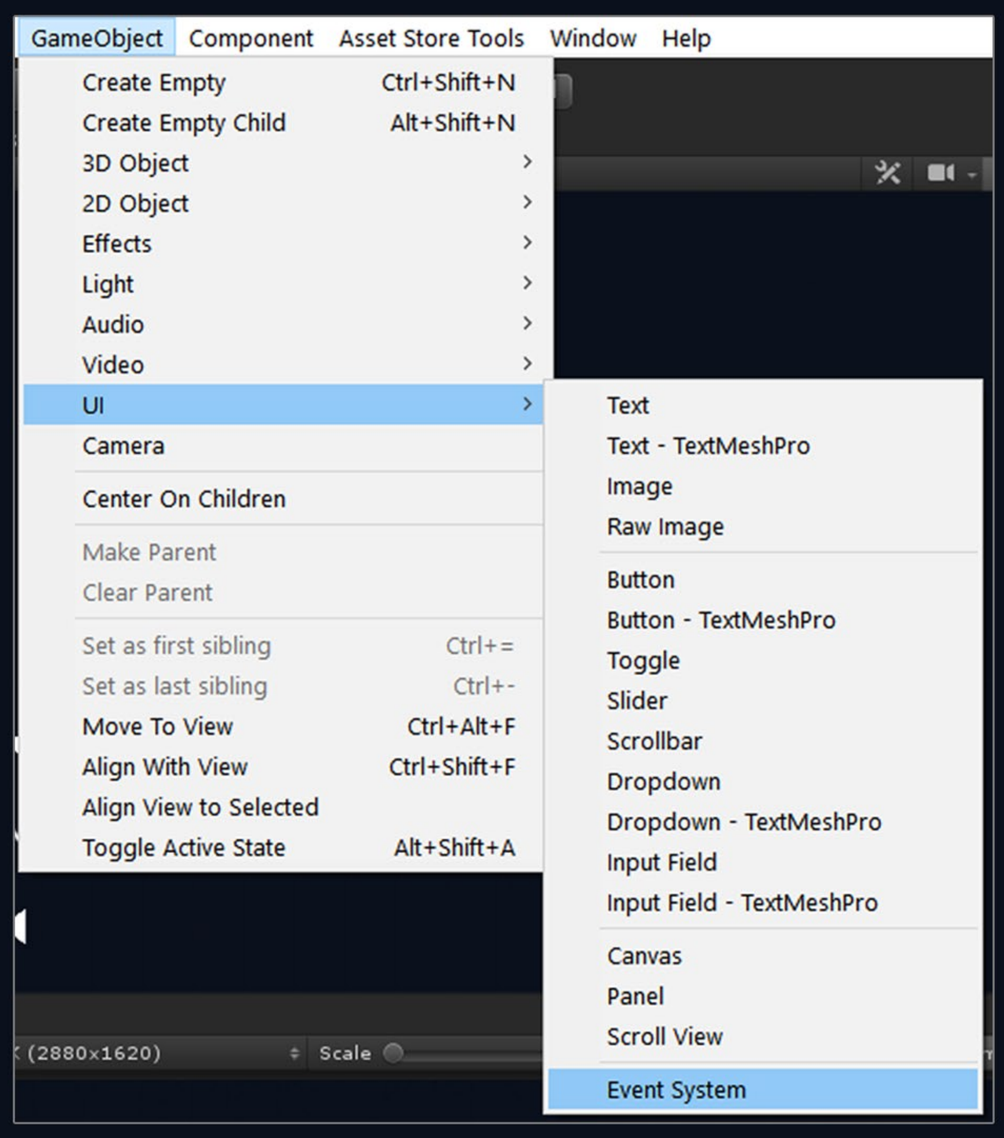
Everything you need can be found in the demo scene **MenuV2**. You can duplicate the scene and use it as a template or just import the prefab **Canvas_V2** manually in your scene. You only need to assign a camera to the canvas and add an EventSystem in the scene and you'll be able to interact with the menu instantly.



I recommend using the demo scene as a template because I set the Post Processing effects and UI VFX to save your time. It is optional and doesn't affect the UI interactions functionally but they are just for visual polish.

1.2 EVENT SYSTEM

In order for Unity to register UI actions in a scene, you need to have an **EventSystem** & **Input Module** attached to a game object. You can auto create one by selecting **GameObject/UI/EventSystem.**



1.3 INSPECTOR WALKTHROUGH

Screens - All the possible screens/sub-menus that you can switch to in the menu. This can be adjusted by adding or removing sub-menus. You cannot remove or re-order the first element in the array since **Element 0** is recognized as the home screen.

Use Splash Screen - If this is enabled, on the start of the scene load, an animated splash screen will appear with a user prompt.

Splash Screen - The Splash Screen Rect Transform that will animate.

UI ELEMENTS & USER DATA

Resolution Drop Down - The drop-down uGUI component in the settings sub-menu that is used to populate screen resolutions at runtime.

Quality Buttons - An array for each quality setting that users are able to assign to. Each element in the array must be a **Button** type. If a quality setting is added or removed, the array should reflect that change.

Audio Slider - The slider component in the settings sub-menu that modifies and saves the master volume.

LOADING SCREEN ELEMENTS

Wait For Input - If this is enabled, once loading finishes for the next scene, the scene will wait for the user to press “Return” before switching. if this is disabled, the scene will auto switch as soon as loading completes.

Load Prompt Text - The TMP_Text Component in the loading screen that will display the user prompt if **Wait For Input** is enabled and the loading has completed.

New Scene Name - The string variable representing the name of the scene that will load on New Game. By default, the scene name is set to SecondScene.

Loading Bar - The Slider component in the loading screen that is used as the progress bar UI. This animates as loading progresses.

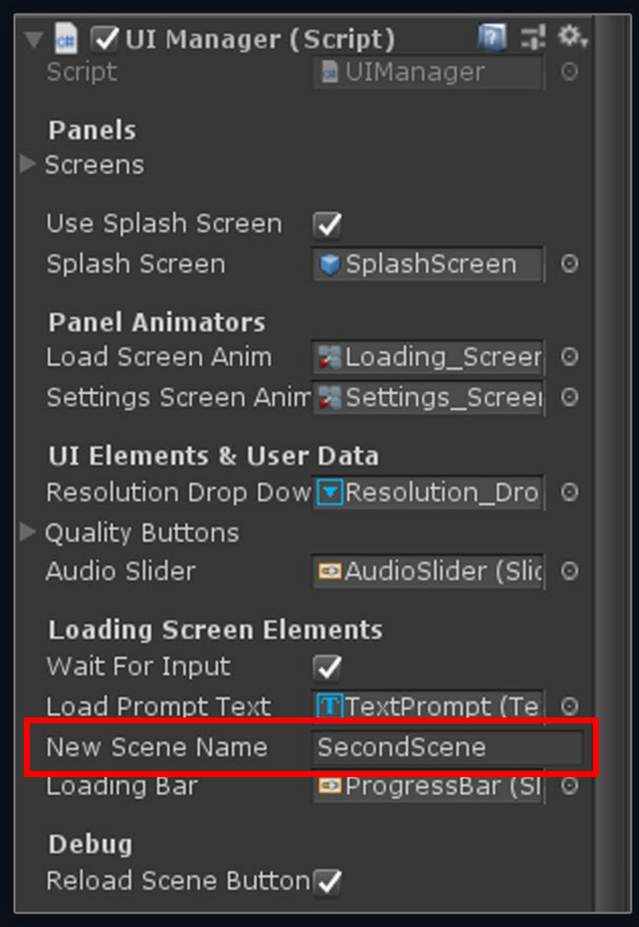
User Prompt Key - The key that users need to press in order to load the scene-scene.

Reload Scene Button - If enabled, pressing R while in the menu will reload the scene. This should be disabled when not in dev mode.

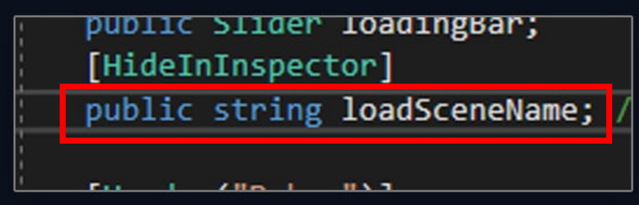
1.4 CHANGING SCENES

One of the features included with Clean Menu is the ability to assign New Game & Load Game scenes directly in the inspector. By default, the New Game scene is set to **SecondScene** which is the second demo scene included.

You can change this to whatever scene you want to load after the user starts a new game in the menu.



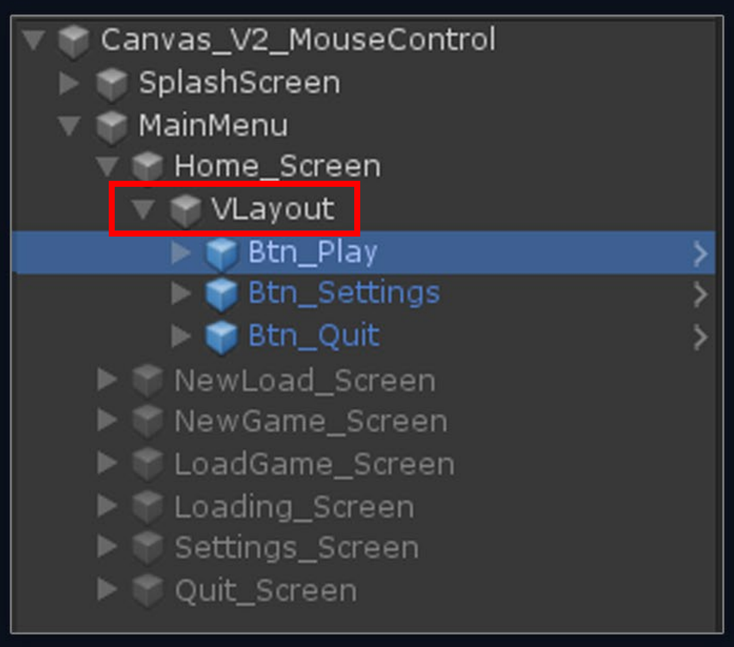
For Load Game, the scene name needs to be set as the loadSceneName string variable through your own code. If you pass in the scene name, the UIManager will handle the loading animations.



2.0 ADDING BUTTONS

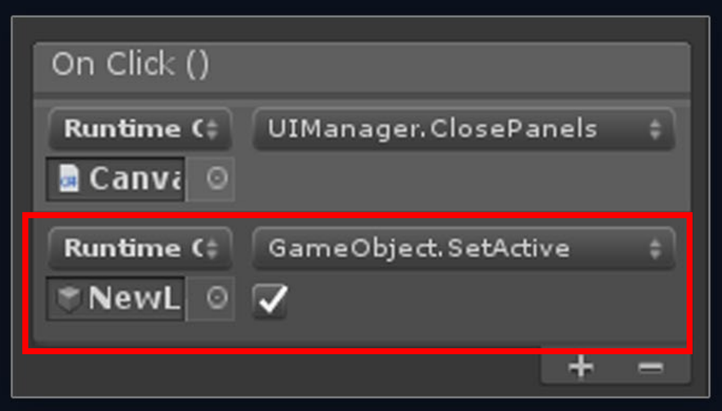
Every sub-menu has a Vertical Layout component that allows for quick and easy button ordering, adding, and removing. If you want to modify the button settings in a sub-menu, all you have to do is duplicate an existing button in a vertical layout group, and then change the SetActive(true) sub-menu game object.

Vertical Layout Element



Ctrl + D / Cmd + D

Button Component



2.1 ADDING A SUB-MENU

Adding or removing sub-menus is quick and easy. Simplfy follow these steps:

1) Duplicate an existing roo screen object like **Home_Screen.** For demonstration purposes, rename it to **Added_Screen.**

2) Add the screen to the Screens array in the UIManager, found on **Canvas_V2**, otherwise it will not enable/disable properly.

3) The button that you want to trigger the sub-menu load, go to the **Button** component and under **OnClick()**, change the GameObject.SetActive object to the new sub-menu object.

3.0 TIPS

Every variable on UIManager has a tooltip describing its function, so you don’t have to guess what they are for. Hover over the variable in the inspector to show the tooltip.

NOTE* Animations are used universally throughout the package, so be careful about changing them. If you want to change animation and not have it affect all other buttons that are prefabbed, make sure to duplicate the Animator component *AND* all of its individual menu animations and swap them out.

NOTE* You can rearrange any of the sub-menus in the Screens array, but do not move the first element as is seen as the Home Screen.

4.0 SUPPORT

For support questions, you can either email us on the Support page or join our Discord: **https://discord.gg/7cK4KBf**

Thank you for choosing SlimUI!