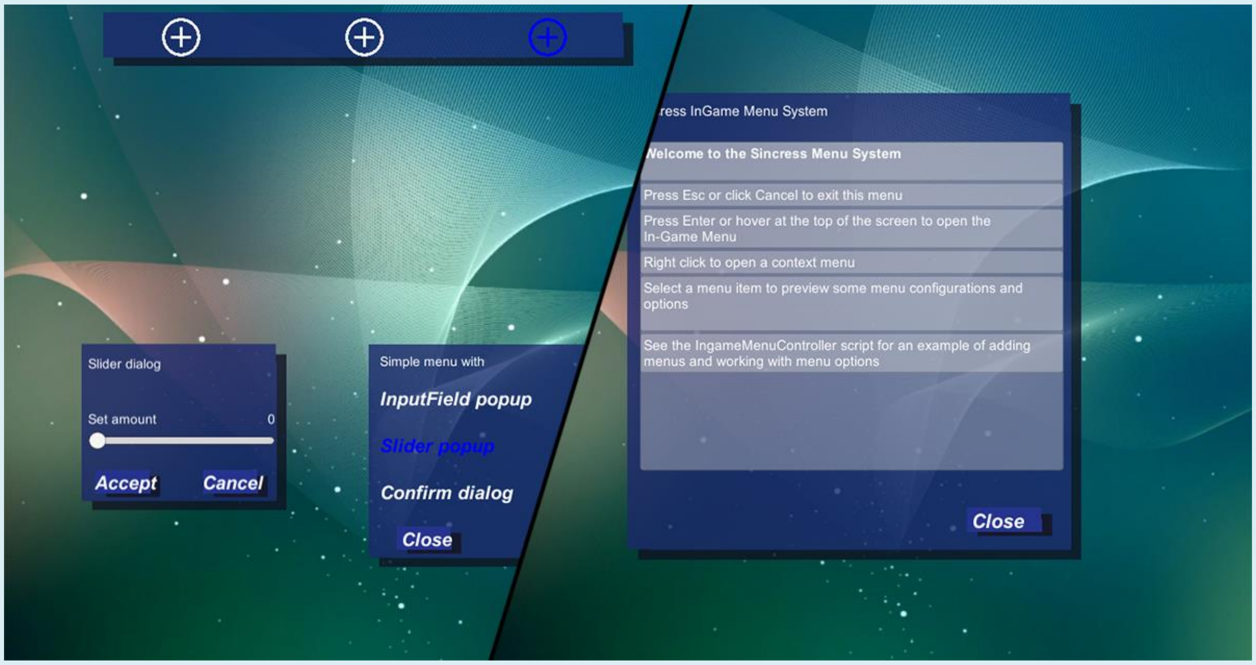
Ingame Menu Framework

Documentation and Guide



Welcome to the Sincress Menu System

Press Esc or click Cancel to exit this menu

Press Enter or hover at the top of the screen to open the In-Game Menu

Right click to open a context menu

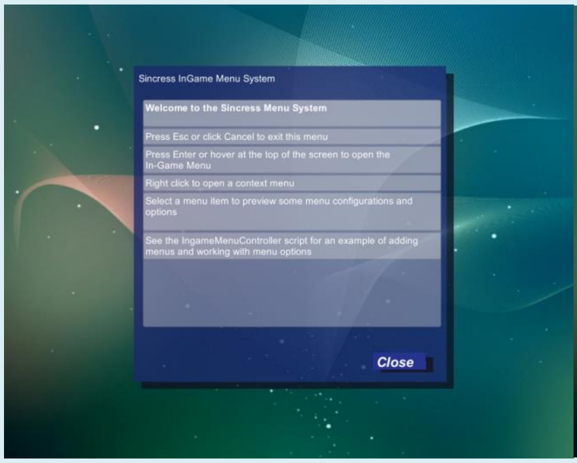
Select a menu item to preview some menu configurations and options

See the IngameMenuController script for an example of adding menus and working with menu options

1. User Interface Elements

The Ingame Menu Framework encompasses several UI elements which can be used in various projects or as a base for implementation of other elements. There is an animated dropdown menu which can be used as the main in-game menu. The project also contains **three Menus** (Simple, Scroll and Text) and **three Popup Dialogs** (Input ,Slider, Confirmation). These cover the basic use-cases found in most games, however specialized menus can easily be implemented using the framework. All the menus can be **navigated either by keyboard or by mouse**, providing a convenient and pleasing feel to both new and seasoned users of the game while keeping the interface intuitive.

sincress@gmail.com



Sincress InGame Menu System

Welcome to the Sincress Menu System

Press Esc or click Cancel to exit this menu

Press Enter or hover at the top of the screen to open the In-Game Menu

Right click to open a context menu

Select a menu item to preview some menu configurations and options

See the IngameMenuController script for an example of adding menus and working with menu options

Close

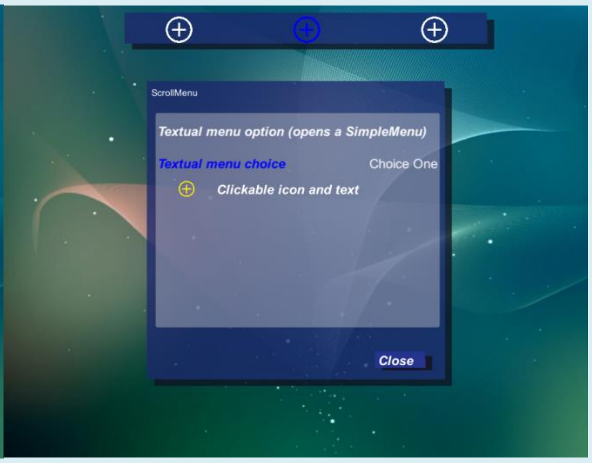


Image 1 - ScrollText menu with text items added; Image 2 – ScrollMenu with clickable items of different types

The **simple menu** can contain several options and scales with the number of options. The **scroll menu** is similar, but of fixed size, and can contain any number of options, which will be scrollable if overflowing. **Text menus** are similar to scroll menus, but only contain non interactable text (which can be colored and bolded). All the menus use the same base class, *MenuController* and inherit from it, each having a title text and a Close button.

There are three simple dialog menus, which can be used independently or as sub-menus to one of the three above mentioned menus. One is the **Slider popup**, a simple popup with a title text, an amount slider, and two buttons: Okay and Cancel. The **Input popup** is basically the same, but with an input field in place of the amount slider. The **Confirmation popup** displays a message and asks you whether you wish to proceed.

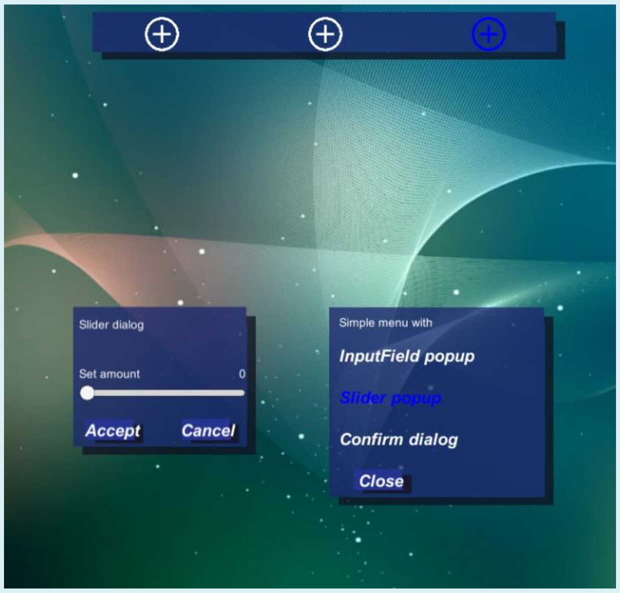


Figure 3 – SimpleMenu with a popup slider dialog

2. Setup Guide

To quickly start using this menu system, open the Example Scene. It contains most of the provided UI elements and makes them through various menus. Pressing the Enter key opens the in-game menu which animates from the top of the screen. It contains a pre-defined number of icons which are shortcuts to various menus in this example scene. Right clicking on the screen opens a context simple menu near the mouse click.

One component is needed to implement the Ingame Menu Framework in your project: the Canvas with the *CanvasController* script and the reference to the *UIElements* ScriptableObject and an Audio Source. The ***CanvasController*** does most of the underlying work for the menu system: It contains methods for opening and closing individual menus and keeps track of which menus are open using a stack, showing only the topmost UI menu. Therefore the menus need to be open using the *OpenMenu()* method, or *OpenMenuAtPosition()* for context menus. Invoking *CloseMenu()* closes the top-most menu by removing it from the stack, and shows the one before it, if any. These methods are accessible from anywhere in the scene via the singleton reference of the *CanvasController* like so:

```
CanvasController.Instance.CloseMenu()
```

The *OpenMenu* methods take, as arguments, a reference to the menu prefab you wish to open. The menu prefabs are available through the UIElements scriptable object referenced . Along with references to all available menus, this script also plays the UI sounds through the audio source, controlled by publicly accessible methods. The second argument, *isSubMenu*, tells the menu controller whether the opened menu will be a menu which overlays previous menus or a submenu (for example; a dialog) of the topmost menu. When opening the menu at a given screen position using *OpenMenuAtPosition*, it is also possible to tell the canvas controller if the previous (topmost) menu should be hidden and overlayed with the current menu.

Each menu will, by default, be closed by pressing the Esc button, which also cancels the current popup. Menus can be navigated by keys or mouse. The code for this is located mainly within the base *MenuController* class which is supplemented by specialized functionality of each of the three menus. Here is a step-by-step on how to open a menu from one of you game scripts. Let's assume we wish **to open a Scroll menu.**

1. We invoke `CanvasController.OpenMenu` with `UIElements.Instance.ScrollMenu` as the argument

2. The return value of OpenMenu is the reference to our menu. However, we only need its *ScrollMenu* script, so we store it in a variable called *scrollMenu*.

3. Now let's add a menu option: call `scrollMenu.AddMenuOption("option 1")` and store the return value in a variable.

4. The return value is the option's OnClick listener, to which we add functionality by writing a lambda: `onClick.AddListener( () => {} );`

In the brackets we can code any functionality which happens when this menu option is selected. The process of **adding a sub-menu** to a menu is not as simple, but an example can be found in the *SimpleMenu*'s OnOptionSelected method. A SubMenu has to be instantiated and positioned manually, as no framework for its manipulation is implemented in the CanvasController. Its anchor is the upper left corner which enables easy positioning, whether the SubMenu is a popup dialog, or a menu such as the Simple menu.

Always consult the Example Scene if unsure of how to implement an existing feature of the Ingame Menu Framework. I hope you enjoy using this UI framework and that it helps you finish your project.

Sincress