Overseer

by Lightbug

mail: lightbug14@gmail.com

This is a mini guide dedicated to explain the basics of Overseer.

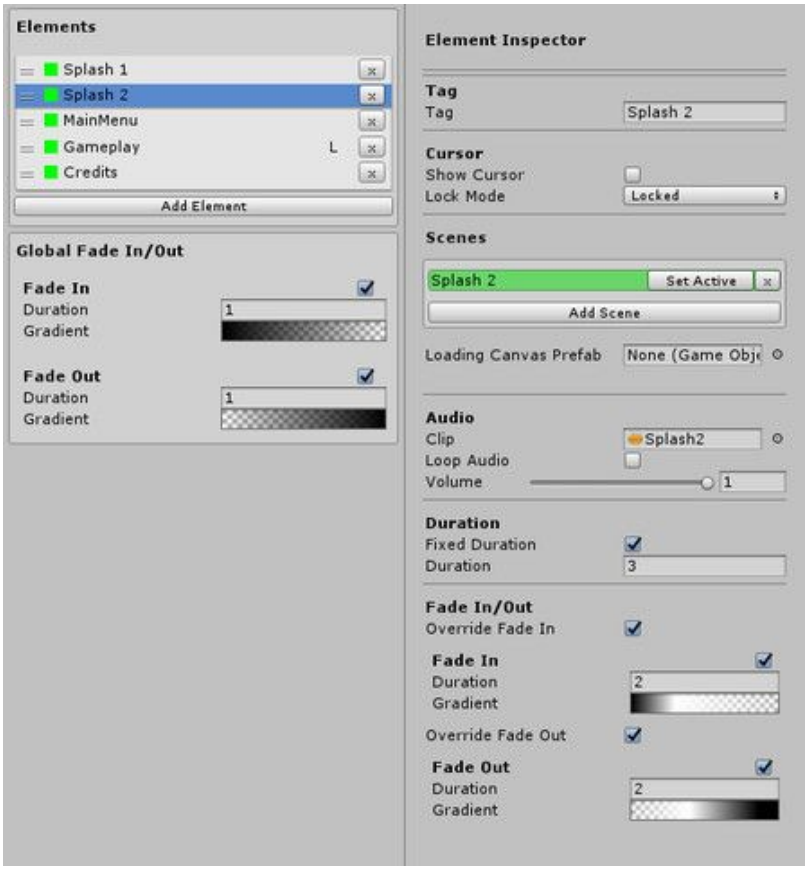
Future releases

At first (v1.0) the main goal was to make a better Unity's Scene Manager, a tool capable of organize all the scenes of your game (along with a one or two extra tools). It is intended to add more features to this asset:

- *Input Management*
- *Node Editor (for the elements)*
- *Menu Creator*
- *Save/Load basic Game Settings (Audio, Graphics, Inputs, Game Profiles, etc)*
- *Audio Manager (loops, ambience, 3D/Binaural effects, etc)*

Overseer Main Asset

The Overseer main editor can be opened by going to the top menu bar *“Overseer/Show Editor”* or by clicking the “Show Editor” button shown in the asset inspector (An Overseer Main Asset must be selected). The editor looks like this:



Elements: Formed by a reorderable list, each item corresponds to an element. The green/red square indicate if the element is valid (non empty and added to the build). The “L” indicate that the element uses a loading screen canvas.
Global Fade In/Out: Set the Fade In or Out for all the elements.
Tag: Works as an ID, useful for referencing an element with the GoToElement(string tag) method.
Cursor settings: Enable/Disable the cursor visibility and select the Lock Mode.
Scenes: Add all the scenes that you want to the element (multiples scenes will be loaded in Addictive mode). All the scenes (included in the build) are referenced by name.
Loading Canvas Prefab: Contains the Loading Screen canvas (the controller will automatically use it for loading the element scenes)
Audio: The audio clip will be played when the element is loaded.
Duration: Whether or not you want to play an element with an specific time duration. (useful for a splash screen).
Fade In/Out: Override or not the Global Fade In/Out settings.

Overseer Controller

This is the main component, it handles all the scenes transitions, loading progress, audio, fade in and out, etc. This component needs to be added to the scene, only one instance of it will remain (Singleton). It requires an Overseer Main Asset to work, you can create one by right clicking over the project view (Create/Overseer/Main Asset)

By entering playmode (hitting the play button) the “starting element” will be loaded first. This is useful for testing different elements of the list.

Referencing the Overseer controller

In order to use some Overseer’s functionalities you will need to get a reference to the instance. Since this component is a singleton you can get the reference to it by doing the following:

```
OverseerController controller = OverseerController.Instance;
```

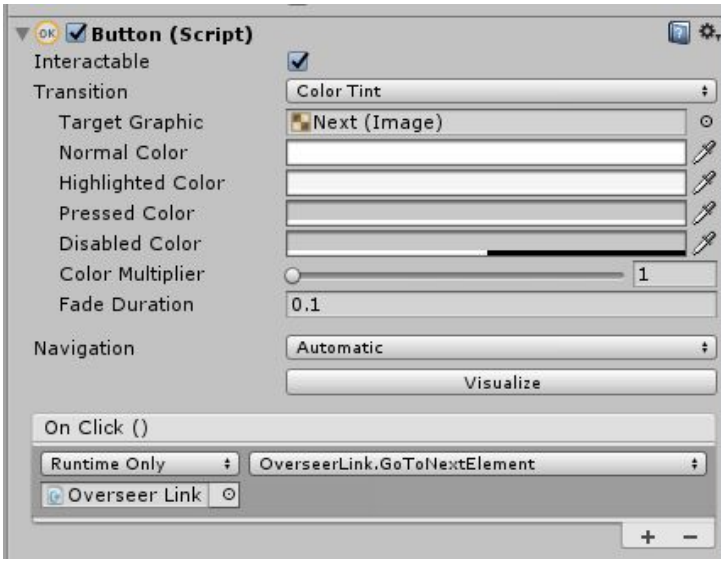
This way you can call use singleton instance by code, calling the following methods:

- GoToNextElement
- GoToPreviousElement
- GoToElement
- ReloadCurrentElement
- ReloadCurrentElementWithFadeOut
- QuitApplication
- QuitApplicationWithFadeOut

Using the Overseer controller with Unity’s events

If you are using Unity’s Events you will need to reference a gameObject with the Overseer Controller in it. Since the controller is a singleton, it is not recommended to use directly an Overseer Controller as a reference for the event, this is because the singleton instance might not be “the one” (even if you add one into the scene). This can be solved by adding an **Overseer Link** component to the scene, using it as a reference component. This component is a Link between the current scene and the main controller instance, from here you can call any significant Overseer Controller public methods (listed above).

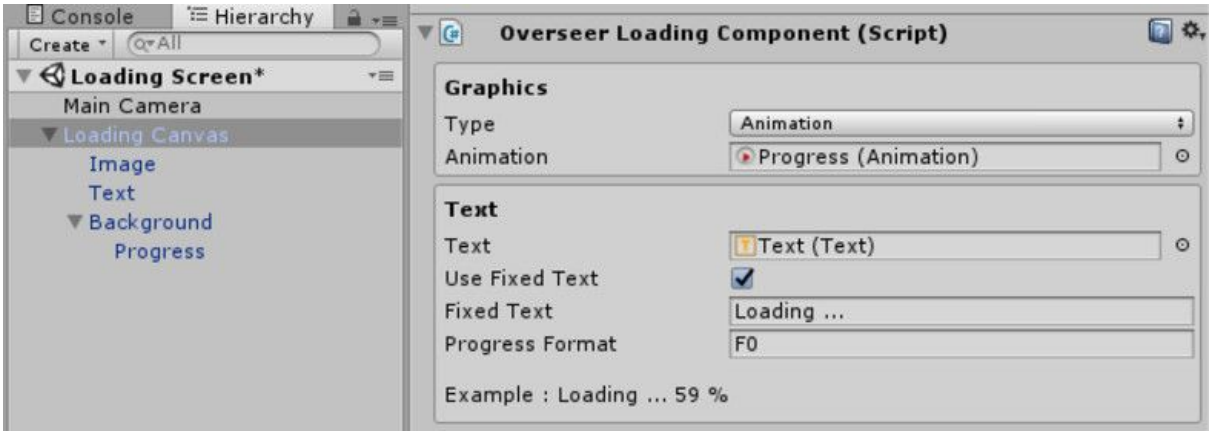
For example, the following component belongs to a Unity's UI Button, the event is calling the method "GoToNextElement" from the OverseerLink component:



Using a Loading Screen canvas

1 - Create a new **Canvas** (UI), for example let's call it "Loading Canvas".
2 - Fill the canvas gameObject with the elements of your choice, these elements will act as progress indicators, whether they are Text (0% to 100%), Animation (0 to clip length) or an Image (fillAmount 0 to 1).
3 - Add an **Overseer Loading Component** to the scene.
4 - Assign a text reference to the text field(if you want to use a text field). You can select the fixed text and some options for the progress.
5 - Select the progress main indicator type, Image or Animation, make sure to assign the references as well.

For example: Selecting Animation, referencing the Animation component from the "Progress" GameObject and referencing the Text component from the "Text" GameObject



6 - Create a prefab from the Loading Scene Canvas.
6 - In your **Overseer Main Asset** assign the prefab to the Loading Canvas Prefab field.

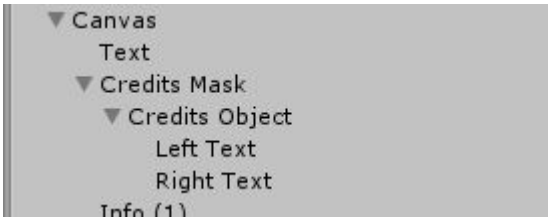
Creating the credits

99.9% of the credits screens used in games behave almost the same, they:

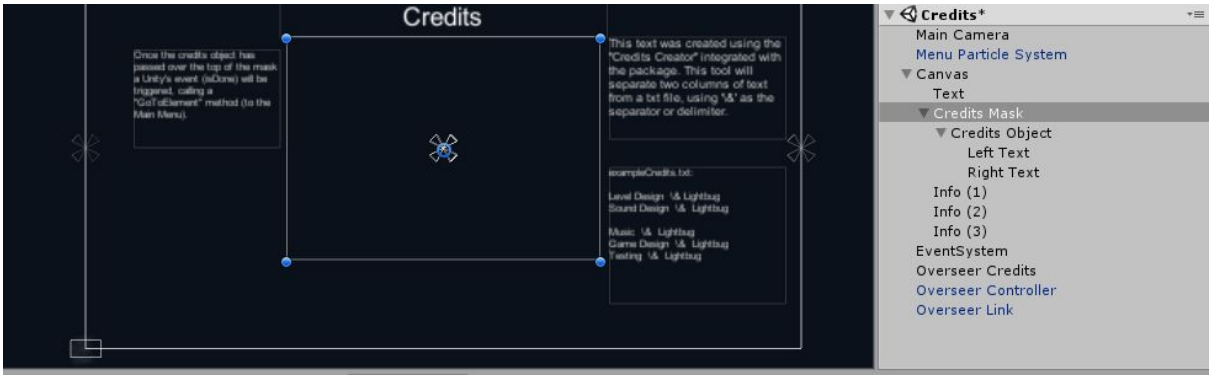
- Show their content inside a UI mask (visible inside)
- Move their content upwards
- The content is separated into two columns, profession/job & person (or vice versa).

Overseer offers a **Credits Creator**(Overseer/Credits Creator) to generate both columns of text (Unity's Text components) from an external txt file. Although you could do whatever you want with the content generated, Overseer integrate a **Overseer Credits** do the typical movement and to trigger an event at the end of it. In order to use this component you add it to an object (anyone) and reference the **Credits Mask** (UI Mask) and the **Credits Object** (the moving UI element).

Here is, as an example, the Credits Mask and the Credits Object used in the "Credits" scene.



Credits Mask:



Credits Object:

IMPORTANT: Both the Credits Object and the Credits Mask must have assigned a Middle-Center Anchor preset to them.