Manual

`Highlight UI` makes a single element pop out by covering the rest of the screen. It is perfect for creating tutorials and hint UI in games.

Basics

You can call one of the 3 static methods on the `HighlightUI` class to show the UI:

```
// Highlight a UI element
UIHighlight.ShowForUI(myRectTransform);

// Highlight a 3D Object
UIHighlight.ShowFor3DObject(myTransform);

// Highlight an user defined Rect on the screen (uses screen coordinates)
UIHighlight.ShowForScreenRect(myRect);
```

`Highlight UI` will create the overlay canvas if it doesn't exist already and show the desired highlight.

You can dismiss the highlight UI by calling `Dismiss`.

```
// Dismiss the highlight UI
UIHighlight.Dismiss();
```

Options

There are a few options you can modify to change how the highlight UI looks and behaves. These are the available options:

Dismiss On Click

If true, clicking on the highlight UI will dismiss it.

Dismiss Action

A `System.Action` to be called when the highlight UI gets dismissed. The action will be called exactly once.

Use this to chain actions to happen after the user dismisses the highlight UI.

Be careful when using dismiss action with `defaultOptions`. You should set it back to `null` if you don't want the action to get called on subsequent dismissals of the highlight UI.

Canvas Sort Order

The highlight UI uses an overlay canvas. Set the sort order where you want it in your game. You might want show additional UI on top of the highlight UI.

Color

The color of the highlight UI. Supports transparency.

Padding

How much space (in screen space pixels) to include as padding around the highlighted object.

Fade Duration

Duration in seconds for the fade animation. Applied to both the fade in and fade out animations.

Frame Sprite

The sprite to use around the highlighted object. If null, there is no frame, so a rectangle with sharp angles will show. You can use any sprite here to make a custom frame. Note that for clicks to go through, you'd need the sprite to have read/write access to its values and have transparent areas. See alphaHitTestMinimumThreshold on Unity documentation.

Frame Type

The image that is used for the frame will have this `Image Type` set on it. Typically `Sliced` and `Simple` should be used.

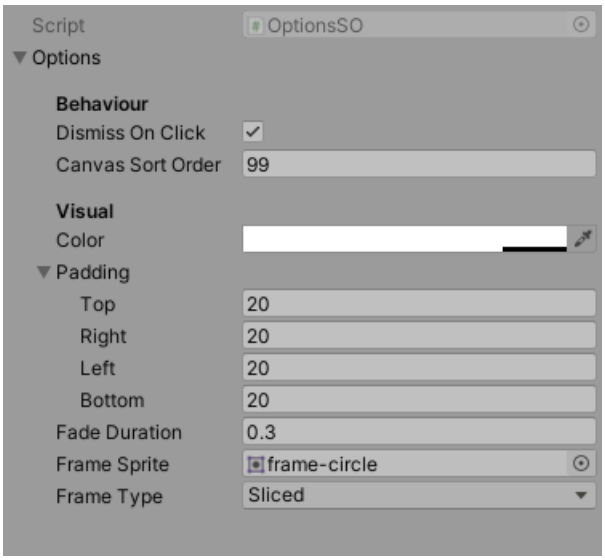
Setting the Options

You can define and use the options in different ways.

From Unity editor using scriptable object options

You can create and set the options in an scriptable object.

Create the scriptable object by going to the `Assets/Create/No Such Studio/UI Highlight/Options SO` menu. Then select the scriptable object and set the options in Unity Editor.



From scripts using the default options

You can also modify the options at runtime programmatically.

```
UIHighlight.defaultOptions.dismissOnClick = true;
```

By changing the `defaultOptions`, the change applies to all the future invocations.

From scripts using the explicit options parameter

You can also explicitly pass the options to the call to show the highlight UI. This way the options get used only for that call and won't affect subsequent calls.

Here we change the dismissOnClick option only for this single call.

```
var tmpOptions = new Options(UIHighlight.defaultOptions);
tmpOptions.dismissOnClick = true;
UIHighlight.ShowForUI(myRectTransform, tmpOptions);
```

Customizing the Highlight UI

`Highlight UI` creates the canvas and its children lazily on first invocation. On subsequent calls, it will try to reuse the old one by searching for the canvas by name. You can create additional children for the canvas and modify the hierarchy as you wish to add new UI elements and modify the components on the children. You should however not change the names as it will cause `Highlight UI` to create them again.

The minimal hierarchy looks like this: NoSuchStudio_HighlightCanvas | *panelTop* | panelBottom | *panelLeft* | panelRight |_ panelCenter

If this hierarcy exists in a scene, `Highlight UI` will use it.

Text Extention

The text functionality is an addon to the base library. It adds a text component to the hierarchy and places it in a place where there is enough space. You can take a look at the code under `extentions` directory and use it as a base to create your own extensions to the `Highlight UI` asset.

You can highlight an element with some contexual text like this:

```
UIHighlight.ShowForUI(myRectTransform);
UIHighlightText.Show("This is a highlighted UI.");
```

Contact

- Discord
- Unity Forum