Skyward File Browser

General use

As shown in the example scenes, the file browser is used by having a script that creates an instance of the *"Plugins/SkywardFileBrowser/Prefabs/SkywardFileBrowser.prefab"* prefab. Or if you prefer, another method that creates an object in your scene with a SkywardFileBrowser component.

In your script, call the open file or save file method on the SkywardFileBrowser class. For example:

```
fileBrowser.OpenFile(path, Output, ClosedWindow, extensions);
```

- Path is the path of a directory that the file browser will open when it starts.
- Extensions is an array containing file types the file browser should show, for example: `{ "jpg", ".png", "TXT" };`

The middle two arguments are the 2 callbacks the file browser can give. The first argument is an Action that takes a string array as input. Such as the first method in the image below. The file browser uses this callback to send the output of the user's selection to your script. In save mode this will only be one item and in open mode this can be one or many depending on the settings.

The second argument is an Action that doesn't have any arguments itself. This callback is used to let you know that the file browser window has been closed and can be a method such as the second method in the image below.

```
// The function recieving the output from the filebrowser
private void Output (string[] output) {
    // Simply print the paths to show something happened.
    // Replace this with your own code to use the File Browser output
    foreach (string path in output) {
        Debug.Log("File browser output: " + path);
    }
}

// The function called when the file browser is closed
private void ClosedWindow () {
    // Print a message to show the closing event
    Debug.Log("File browser window closed");
}
```

Settings

The file browser uses a class called SfbSettings to allow some customization in its inner workings. A setting can be changed [File browser reference].Settings.[setting] = [new value]. For example:

```
fileBrowser.Settings.ShowHiddenFiles = true;
```

See also the example scene scripts. Here is a list of all settings and a short description for each:

AllTypesShowOnlySetExtensions

Default: true.
The user can select an extension in the file browser window. Only files of that type will be shown in the browser panel. One of the options the user can select is to show all file types. With this setting the browser will only show files with extensions used when opening the file browser instead of all file types.

AllowFolderAsOutput

Default: true.
Allows the user to select folders as output.

AllowFileAsOutput

Default: true.
Allows the user to select files as output.

DoubleClickTime

Default: 300.
The time in milliseconds between clicks that register as a double click.

HiddenOpacity

Default: 0.55f.
The opacity used when displaying hidden files and folders. This makes them look less accessible.

KeepBrowserInMemoryWhenClosed

Default: true.
The file browser creates objects for itself in the scene hierarchy. Keep these objects when the browser is closed or destroy them.

MaxRecentEntries

Default: 5.
The amount of recent entries to remember and show in the location browser panel.

RequireFileExtensionInSaveMode

Default: false.
Enforce the user to enter an extension in save mode.

RemoveLocationWhenMissing

Default: false.
Removes recent or favorite entries when the file browser can't load the file or folder because it is missing. A file might be deleted or on an unplugged drive for example.

RemoveLocationOnDelete

Default: true.
Removes the associated recent and favorite entries when a file or folder is deleted from within the file browser.

RestrictOutputToOneFile

Default: false.
This will disable the open button when the user selects more than multiple files.

SaveSettingsToPlayerPrefs

Default: true.
Saves recent and favorite entries to the unity player prefs. At least this setting or SaveSettingsToDisk must be enabled to allow the file browser to load bookmarks after closing.

SaveSettingsToDisk

Default: false.
Saves recent and favorite entries to disk. At least this setting or SaveSettingsToPlayerPrefs must be enabled to allow the file browser to load bookmarks after closing.

SettingsSaveFileName

Default: “SfbSettings”.
The file name to use when saving recent and favorite entries. This setting applies to both SaveSettingsToPlayerPrefs and SaveSettingsToDisk.

SettingsSaveFolder

Default: “”.
This setting is used to determine where to store settings when using the setting SaveSettingsToDisk.

ShowHiddenFiles

Default: false.
Allow users to browse hidden files and folders.

TooltipDelay

Default: 500.
The time in milliseconds a user has to hover over an entry in the file browser to show the tooltip. The tooltip is used to show file names that are too long to fit in the browser window.

Customization/Prefabs

All of the prefabs that come with the asset can be changed in every way except for the requirements in this document. That includes images, opacity and other things that work in the Unity UI normally. That also includes the order, layering and parenting of all object.

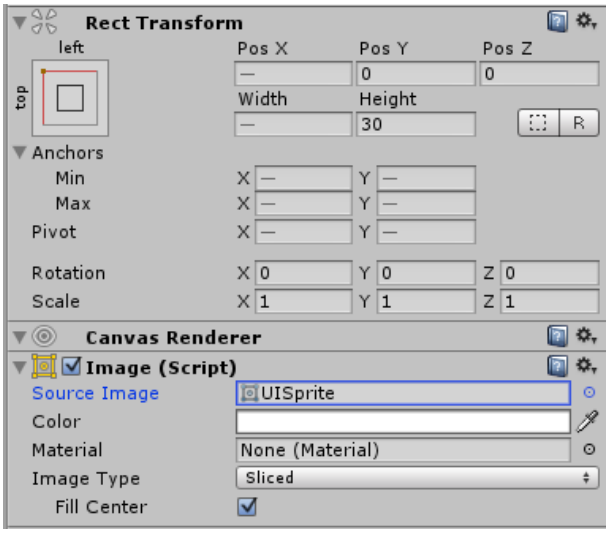
If you changed the base prefabs and you don't know how to fix it, reimport the asset and see how the components are used.

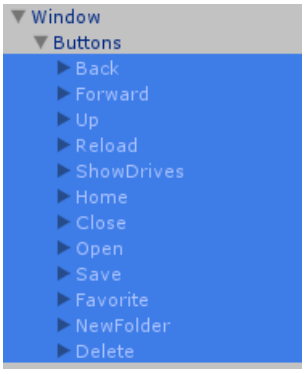
Tips

I would have liked to make the customization easier to do but the ways I could come up with would have sacrificed the strengths of unity's UI system.

Editing multiple objects at once

Unity allows you to edit the properties of multiple object at the same time. This means that editing the shared sprites of all buttons in the file browser is a lot easier.

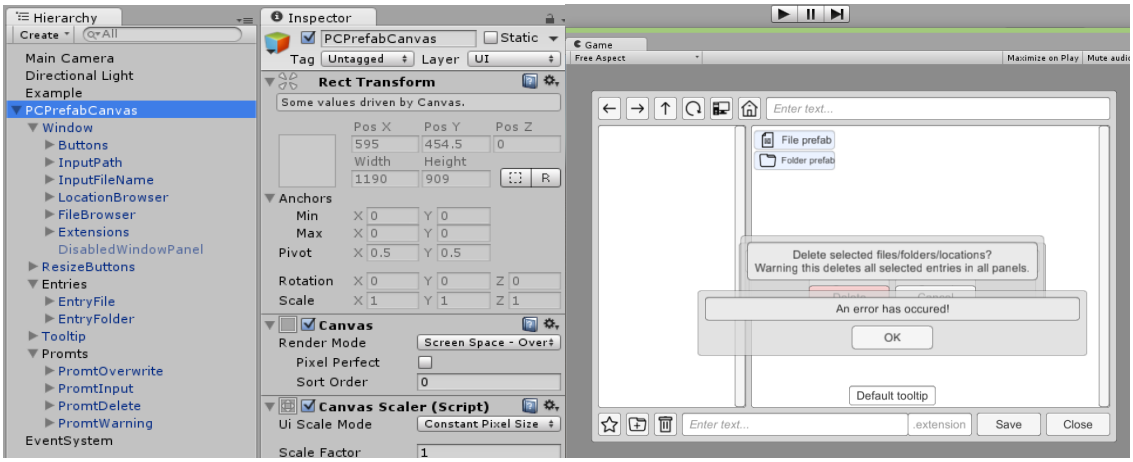




Simply select the buttons in the hierarchy and change all their height, anchor, sprites etc. at the same time.

Having prefabs in the hierarchy at all times

While creating the file browser I kept the prefabs for all components in the scene hierarchy all the time. They were in an empty gameobject that I could enable and disable when I needed to make a change. This allows you to preview your changes without having to start the game. Just remember to press apply at the top of the inspector to update the prefab or the changes will not have any effect.



Tooltip

This is prefab used by the file browser when the user hover their mouse over a file name that is too long for the file browser window. It can be any object as long as there is a child object with a text component.

Components

Unity components

Canvas

Just a canvas, no special component is required.

Event system

Unity creates one automatically when you create a canvas. If you can't interact with buttons and such, this is missing.

These components must be children or sub-children of an SfbWindow

SfbButton

Attach to a button and set an action for the file browser. In open mode, any save buttons will be hidden. In save mode, any open buttons will be hidden.

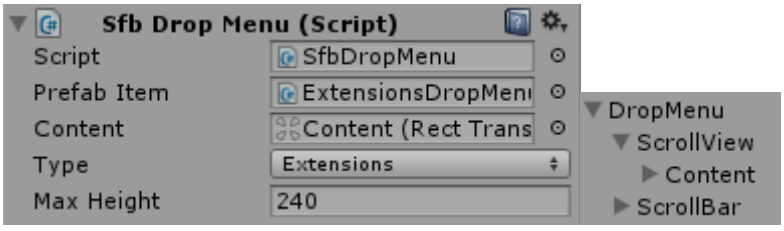
SfbDisabledWindowPanel

Attach to an overlay of the main window. When a popup window is open the overlay will be activated and disable interactions with the main window until the popup window is closed.

SfbDropMenu

Attach to a drop menu, the file browser will activate the object when the appropriate button is pressed. This component requires a ScrollView, there is an entry in the drop menu inspector for the ScrollView content. The drop menu will add any options to this content object. The only current type of drop menu is extension selection.

The entry Prefab Item takes a prefab with SfbDropMenuItem attached. This prefab is used to create the drop menu entries.



SfbDropMenuItem

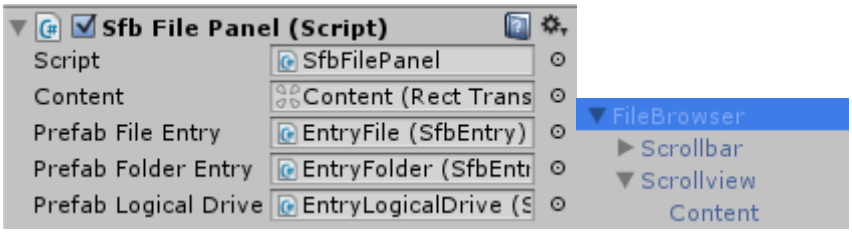
Required by SfbDropMenu.

SfbEntry

Required by SfbFilePanel and SfbLocationPanel. Works like a button. The entry Type sets what kind of entry this prefab is used for.

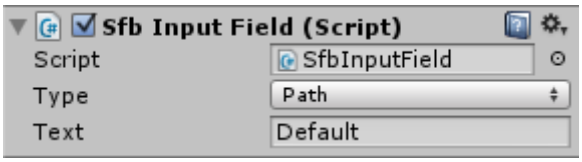
SfbFilePanel

The panel used for browsing. The file browser will add all entries to the object set as content. The prefabs set in the component are used to create the browser entries.



SfbInputField

Attach this component to input fields to use them with the file browser. The Text entry in the inspector is the default text used for the input field. The Type entry sets what the file browser uses the input field for.



SfbLocationPanel

The same as the SfbFilePanel. The location panel has 2 more prefabs, recent and favorite headers.

SfbWindow

This component tells the file browser what the main window is. A lot of components need this to be a parent in the file browser hierarchy.

These components can be attached to any window panel

SfbDraggable

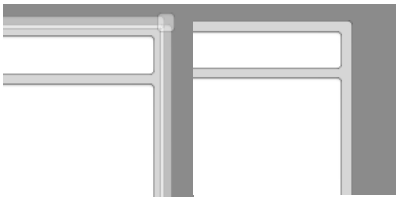
Add this to a window to make it draggable

SfbResizable

Add this to a window to make it resizable, this component requires a prefab containing buttons with the SfbResizeButton component attached.

SfbResizeButton

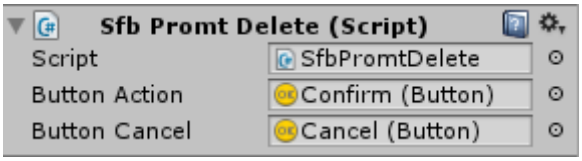
Required by the SfbResizable component. Attach this to a button and set which side of the window this button should resize. The button is generally invisible in use



These components are used to show a specific window to the user

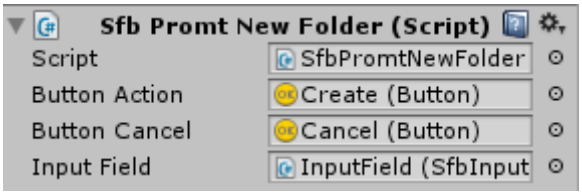
SfbPromtDelete

Required by the file browser. This component is used when the file browser warns the user about deleting the selected items. The component requires 2 buttons one for confirming the deletion of the selection and one for cancelling the deletion.



SfbPromtNewFolder

Required by the file browser. This is used when the user wants to create a new folder. The input field is used as the folder name, the action button confirms creating the folder.



SfbPromtOverwrite

Required by the file browser. This component is the same as SfbPromtDelete, except it is used when the user has selected an existing file as output in save mode. The action button in this case confirms the overwriting of selected files.

SfbPromtWarning

Required by the file browser. This is used when the user is shown a warning. The button confirm the user has read it and closes the window.

Closing statement

If you feel this guide is not detailed enough, or you would like some examples customizing the file browser let me know, skywardray@gmail.com