INTRODUCTION

Hi !

First of all, thanks for purchasing "File Browser 2.0". Many improvements have been made since the first versions, taking into consideration the few questions people had with the previous one. Here is a list of what has been done :

- Full rework of the code base from scratch.
- New modes have been added. The File Browser now support opening files and directories, as well as saving files.
- The UI now work with modular blocks. Use the simple interface provided, and drop the new content, like a Search module, Thumbnail generation, a Favorite system, etc...
- Mobile support should be functional. The base module works should on any platform, and most of the modules should. More about this in the dedicated sections of this documentation.
- Easily customize the UI. Helper scripts have been made to change the color scheme at a simple click.

If you previously owned the asset, you will see how much it changed. And you are new onboard, well, enjoy ! Still, I would greatly appreciate it if you took some time to leave a review, or simply contact me by mail to give me to tell me what you think about it. Of course, if you have any question, go ahead and get in touch.

Asset Store link : http://u3d.as/cAL
E-Mail : huguenin.benjamin@gmail.com

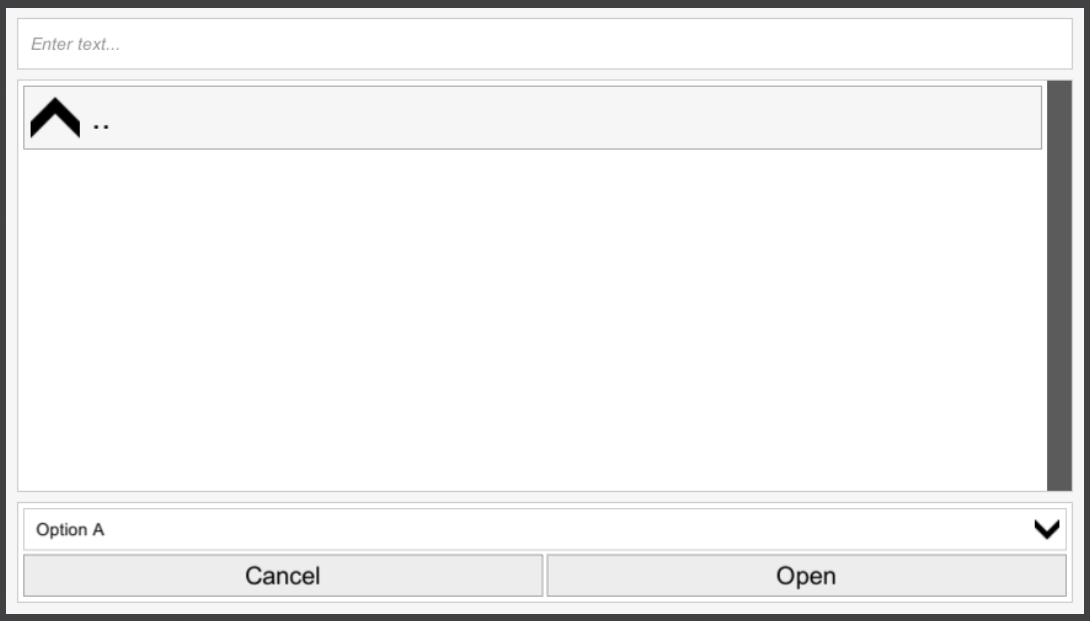
And if you want to take a look at my other projects, here is my website :

http://www.ellenack.com

Once again, thanks a lot for choosing this asset !

Benjamin Huguenin

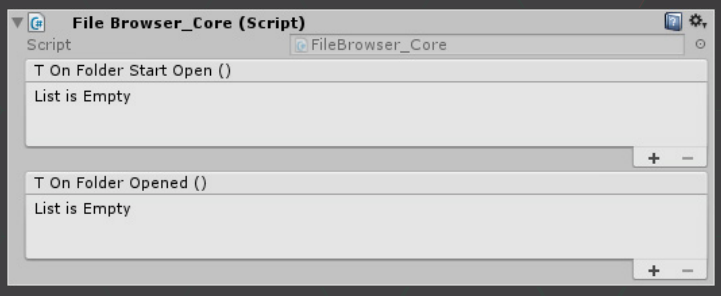
THE MAIN VIEW



The main view is devided in several areas :

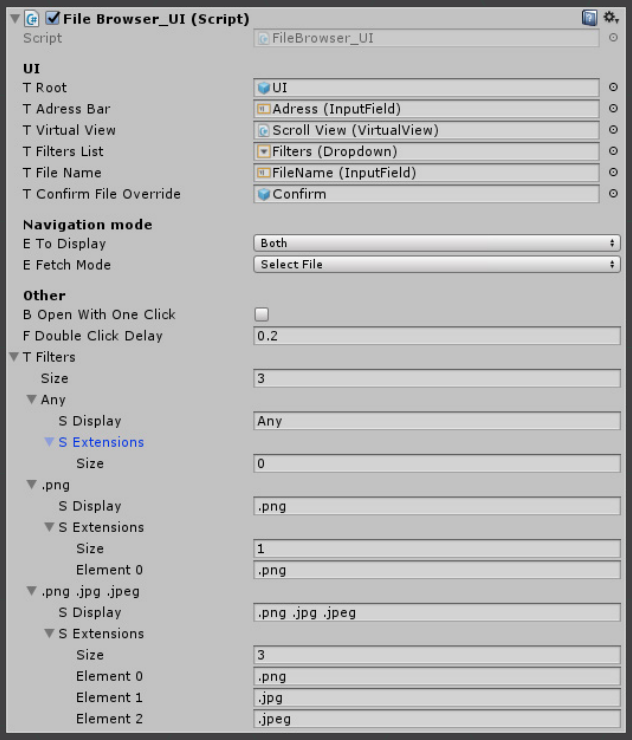
- The adress bar (at the top) : it will display the adress of the current folder at runtime. You can type an adress to try and access it.
- The folder view (in the middle) : it is a scroll view where all the files and directories will be displayed at runtime. By default, an entry to go to the parent folder is there.
- A dropdown list (just below the folder view) : when opening a file or a directory, you can select filters to only display files with the provided extension. The dropdown list is replaced by an input field when the file browser is in "Save" mode, so you can type the name of the file.
- Two buttons : "Cancel" close the file browser and return an empty string, "Open" will either open a directory, a file, or do nothing depending on the mode.

FILEBROWSER_CORE INSPECTOR



The FileBrowser_Core script is the one actually going trough the directories to return the lists of files and folders. You will find it on the "FileBrowserRoot_*" prefabs. There is two events where you can plug in your callbacks. "OnFolderStartOpen" is called at the beginning of the opening of the folder. No file has been retrieved yet. "OnFolderOpened" is called when the folder has been searched for files and directories, and the UI has been updated accordingly.

FILEBROWSER_UI INSPECTOR



Let's take a quick look at the FileBrowser_UI inspector. You will find it on the "FileBrowserRoot_*" prefabs.

> The UI :

There isn't much to say here, it simply reference the necessary objects for the FileBrowser to work properly. You shouldn't have to change anything here.

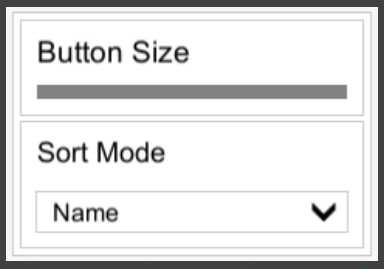
> The Navigation Mode :

- To Display : Here, you can select if you only want the viewer to display the files, or only the directories, or both of them.
- Fetch Mode : This is the most important choice. Here is an explanation of the modes :
  - ¤ Select File : The user will have to open a file for the browser to close.
  - ¤ Select Folder : The folder will have to select a folder and click open to close the file browser.
  - ¤ Save File : The user will have to go in a directory of his choice, enter a file name, and click open for the file browser to close.

> Other :

- Open with one click : Disable double click. Usefull for mobile targets.
- Double Click Delay : How long the delay between two clicks can be. If you double click and the delay between the two of them is higher than this value, the double click isn't processed.
- Filters : Here you can setup your own filters. A filter is defined by a display name, which can be anything, and a list of extensions to filter out. A filter with an empty list of extensions will not filter any file (the "Any" filter above is such an exemple). There must be at least one filter, and the default filter when opening the browser is the filter at the 0 position. Please refer to the image above for examples.

APPEARANCE MODULE

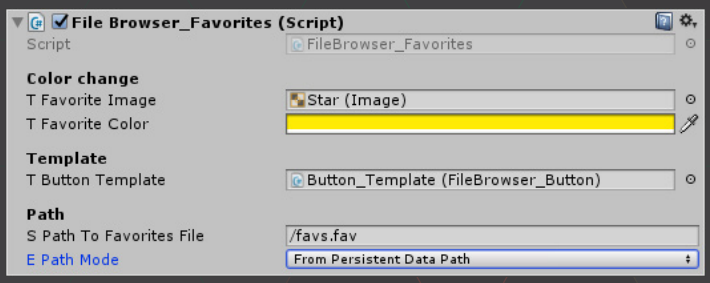


The Appearance module let the user control what the file view will look like. The size let you have smaller and bigger buttons. You can change the min and max values of the slider to fit your needs. Then, there is the sort mode. This one is pretty straightforward and allow you to sort the files with the provided mode. To add new sort options, follow these steps :

1) Go in "FileBrowser_Appearance.cs".
2) Add an option to the "SortMode" enum.
3) At the bottom of the script, add a sorting function.
4) Add a branch to the switch in the "Sort" function using your enum value and calling your sorting function.
5) In the UI, in the dropdown of the prefab you can see above, add a new entry with the exact name of the enum value you added to "SortMode".

Done !

FAVORITES MODULE



The Favorite module is pretty simple to setup. It is composed of a button to add or remove a folder as a favorite, and a view where buttons are added dynamically for each of your favorite folders.

The inspector let you select a few things. When in a favorite folder, you can change the color of an image to show it is a favorite. By default, the star is selected, and will turn yellow when in a favorite folder. Then, a template button that is instantiated for each favorite folder. You should not have to change it, but it is here. Finally, to save the favorite folders through several instances of the game, you must provide a path. It can either be absolute (not recommended), relative to the data path or relative tothe persistent data path.

FOLDER MANAGEMENT MODULE

The Folder Management module provide you with the most common tools you might need when using a file browser. There isn't much to say about the module itself, but the setup need a few extra steps compared to others.

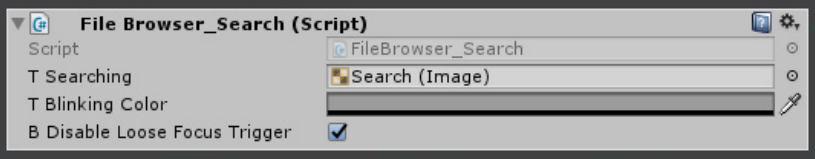
1) Under the "FolderManagement" gameobject, you will find a disabled gameobject named "Folder-Management_Confirm".
2) Move it as a child of the "UI" gameobject, which is a child of the "FileBrowserRoot_*" itself. It should be at the same level of hierarchy as "Confirm" and should be place under it (not as a child, but actually just under it as a sibling). It will break the prefab instanciation, but it is necessary.
3) Enable the "FolderManagement_Confirm" gameobject.
4) Check its RectTransform and set Left, Top, Bottom and Right at 0.

And you are done. To make sure you did good, simply enable "DeleteConfirm" under "FolderManage-ment_Confirm" . The UI should be completely covered by a dark panel, and a confirmation screen should show up. Disable it once you are done.

HISTORY MODULE

The History module is pretty simple to setup. Simply drop it and you are done. You can now go back to the folders you previously visited.

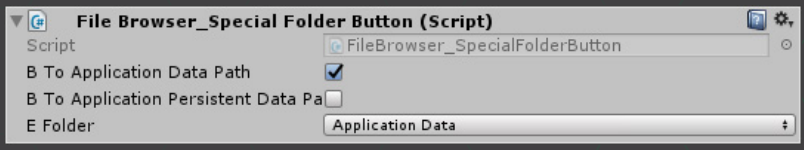
SEARCH MODULE



The Search module is also pretty easy to setup. It is composed of an input field to type your search, and a cancel button to simply cancel an ongoing search.

To give feedback on the fact that a search is under way, the default method is to make an image blink. The default selected image is the back of the search panel. It will blink back and forth between the default color and the one you provide in the "Blinking Color" field. Finally, because the search is launched when the "OnEndEdit" of the InputField is triggered, it will start a search when the field loose focus. To disable this, enable the "DisableLooseFocusTrigger".

SPECIAL FOLDER MODULE



The Special Folder module is a bit different than the others. In fact, it is just here to regroup several sub-modules that are the Special Folder Buttons. The buttons do not depend on each other, so you can add and remove buttons as you want. To create your own :

1) Duplicate one of the provided button of the module.
2) Change the icon (the image which is a child of the button).
3) Change the label (the text which is a child of the button).
4) On the button, search for the "FileBrowser_SpecialFolderButton" component (see above).
5) Select the folder you want. The options are ordered by priority !
   - If "ToApplicationDataPath" is set to true, clicking the button will take you there.
   - Else, if "ToApplicationPersistentDataPath" is set to true, clicking the button will take you there.
   - Else, it will try to access the folder specified in the enum provided.

Theorically, it should work on any platform. However, the enum values are system values, and may or may not be associated to a destination folder. This is beyond my control, so please refer to the documentation of your target platform to see what values you should use.

THUMBNAILS MODULE

The Thumbnails module is the last of the provided modules. It is also setup differently. You have to place it as a child of the button used as template for the file view. The button is at :

> FileBrowserRoot_*
>> UI
>>> MiddlePanel
>>>> Scroll View
>>>>> Viewport
>>>>>> Content
>>>>>>> Button_Template
>>>>>>>> PLACE THE THUMBNAIL MODULE HERE

Then, on "Button_Template", add a element to the "OnInit" event of the "FileBrowser_Button" component (NOT "OnClick" from the "Button" component). Reference the thumbnail module you just added, and select the function "GenerateThumbnails" with the dynamic "FileBrowser_Button". Press play and go to a folder containing PNG or JPEG, and you should see thumbnails !

HOW TO CUSTOMIZE ?

Two file browser prefabs are provided with the asset : "FileBrowserRoot_Simple" and "FileBrowserRoot_Full". The simple version is stripped of any module, and contains the very minimum to have the file browser run. The full version contains all the modules, and some elements have been taken out of their original modules to fit better in the UI general design (like the button to go to the parent folder and the favorite button). But what if you want to create your own version ? This is what we will cover now.

Add modules :

Adding modules is a pretty straightforward process, and is common to the majority of them (please refere to the documentation of the modules to make sure there is not another way to proceed). Starting with the "FileBrowserRoot_Simple" prefab, add a panel as a child of UI. It has to be above "Confirm", but can either be above or under "MiddlePanel". Adjust its size and make sur the middle panel is still visible under it (you can also adjust the size of the middle panel, of course). Then, simply drop a module under this new panel you just created, and you are done !

You can have several modules under a single panel, simply adjust their height and vertical position so they do not overlap. Having a good understanding of the snapping and stretching options of the RectTransform beforehand is recommended. You can refer to the "FileBrowserRoot_Full" prefab to see how I did it. However, I do not recommend removing modules from it to then plug your own. Because it is an already customized version of the asset, some references may be broken if you do.

Change the color :

You will find a "FileBrowser_Color" script on "FileBrowserRoot_*" and any module with a UI interface. It works in association with the "FileBrowser_ColorManager" that we will cover in a moment. This component is composed of three arrays : "ImagesColors", TextColors" and "OutlinesColors". Each of these array will let you associate labels to arrays of images, texts and outlines. "DarkWindow", "Icons", "TextPlaceholder" are some of the default labels you can find.

Then, if you look at the "FileBrowserRoot_*" , you will find the "FileBrowser_ColorManager" that we spoke about. It contains an array of "FileBrowser_Color", that you have to fill with any gameobject having such a component. Then, there is an array where you can associate colors to labels. Of course, these should be the same labels you used in the "FileBrowser_Color" . Change the colors, enable the boolean "Set" at the bottom of the script, and you can now modify the color of all the UI with a single click. The global color allow you to apply a color filter to all the UI.

HOW TO USE

First of all, the FileBrowser should be put under a canvas. By default, the file browser start closed. Then you will have to open it from script, and have a way to wait until a file or folder has been selected. When it is done, retrieve the path of the file, and you are done ! Here is such a way :

```
using UnityEngine;
using System.Collections;
using UnityEngine.UI;

public class FileBrowser_Demo : MonoBehaviour
{
	public string _sStartPath; // The path of the folder to display when opening the file browser
	public FileBrowser_UI.FetchMode _eMode; // The mode to use : select file, select folder, or save file
	public FileBrowser_UI.ToDisplay _eDisplay; // Display  only files, only folders, or both
	public string _sDefaultExtension; // The extension applied when saving a file. Empty means any extension can be provided
	public Text _tResult;

	void Update ()
	{
		if (!FileBrowser_UI.Instance._bIsOpen && Input.GetKeyDown(KeyCode.Space))
			StartCoroutine(WaitForResult());
	}

	IEnumerator WaitForResult()
	{
		FileBrowser_UI.Instance.ShowWindow(_sStartPath, _eDisplay, _eMode, _sDefaultExtension);

		while (FileBrowser_UI.Instance._bIsOpen)
			yield return null;

		_tResult.text = FileBrowser_UI.Instance._sResult;
	}
}
```