GFile Browser

GFile Browser is a graphical file browser made with Unity5 UI to use in your Unity3D Games because Unity3D itself doesn't involve any file browser.
This one is relatively simple but fully-customizable to your needs.
The whole layout can be changed in whatever way you like.
Currently only works for PC!

Preparing

First of all you need to make a script and attach it to the MainCamera.

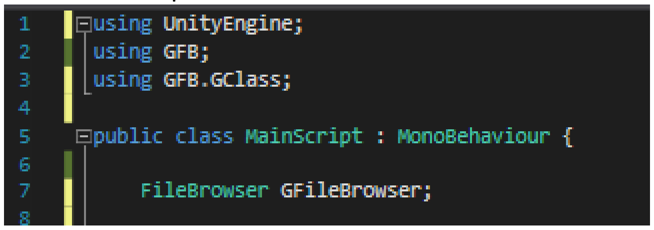
Now initialize the file browser plugin by adding to your code

1. using GFB;
2. using GFB.GClass;

And inside your monobehaviour class make a filebrowsercomponent variable

1. FileBrowser GFileBrowser;

You will end up like this :



```
using UnityEngine;
using GFB;
using GFB.GClass;

public class MainScript : MonoBehaviour {

    FileBrowser GFileBrowser;
```

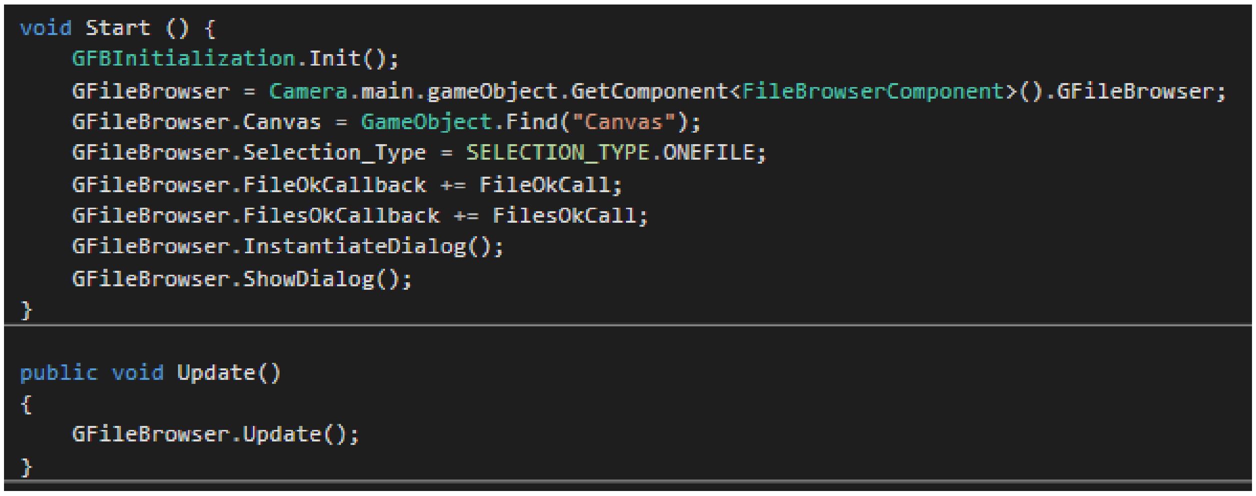
Now to actually instantiate the dialog to make it usable for your game all you need to do is initialize the GFB, set the variable to the actual component,change the canvas name, the sort method, the selection type and last but not least the callbacks

To do this add this piece of code to your Start function

1. GFBInitialization.Init();
2. GFileBrowser = Camera.main.gameObject.GetComponent<FileBrowserComponent>().GFileBrowser;
3. GFileBrowser.Canvas = GameObject.Find("Canvas");
4. GFileBrowser.Selection_Type = SELECTION_TYPE.MULTIPLEFILES;
5. GFileBrowser.FileOkCallback += FileOkCall;
6. GFileBrowser.FilesOkCallback += FilesOkCall;
7. GFileBrowser.InstantiateDialog();
8. GFileBrowser.ShowDialog();

And this piece of code to your Update function

1. GFileBrowser.Update()



```
void Start () {
    GFBInitialization.Init();
    GFileBrowser = Camera.main.gameObject.GetComponent<FileBrowserComponent>().GFileBrowser;
    GFileBrowser.Canvas = GameObject.Find("Canvas");
    GFileBrowser.Selection_Type = SELECTION_TYPE.ONEFILE;
    GFileBrowser.FileOkCallback += FileOkCall;
    GFileBrowser.FilesOkCallback += FilesOkCall;
    GFileBrowser.InstantiateDialog();
    GFileBrowser.ShowDialog();
}

public void Update()
{
    GFileBrowser.Update();
}
```

Let's analyse this piece of code.

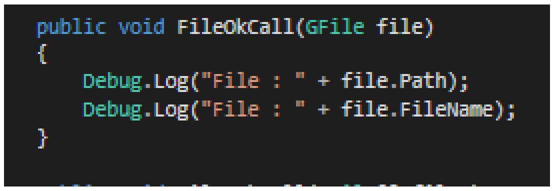
GFBInitialization.Init() will add the component "FileBrowserComponent" to your Main Camera in order to have the file browser working.The second line of code will actually grab this component and assign it to "GFileBrowser"

Canvas variable is used to show the GFile Browser where to instantiate the dialog. If you have changed your canvas, change the variable to your canvas.

Selection_Type indicates how many selections you can have.There are 2 supported selection types.

1. ONEFILE (Will only return one file and can only select one file)
2. MULTIPLEFILES (Will return all files selected, allows more than 1 selections)

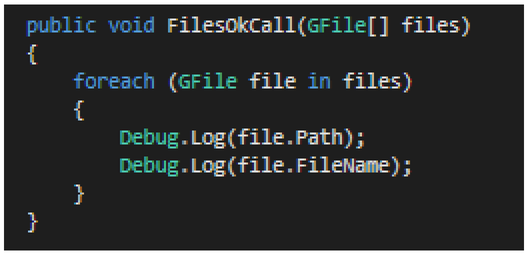
FileOkCallback is the function that will be called from the GFile Browser when the user is done selecting a file (FileOkCallback is used for ONEFILE Selection type). Just make a function with a string parameter like this :



```
public void FileOkCall(GFile file)
{
    Debug.Log("File : " + file.Path);
    Debug.Log("File : " + file.FileName);
}
```

In this example when the user is done selecting his file, the parameter file will return a class named GFile which contains 2 strings "Path" which is the path to the file and "FileName" which is the file name.

FilesOkCallback is the function that will be called from the GFile Browser when the user is done selecting multiple files(FilesOkCallback is used for MULTIPLEFILES Selection type). Just make a function with a GFile[] parameter like this:



```
public void FilesOkCall(GFile[] files)
{
    foreach (GFile file in files)
    {
        Debug.Log(file.Path);
        Debug.Log(file.FileName);
    }
}
```

In this example when the user is done selecting his file, the array of string "files" will return the paths of the files the user selected.

If no files or file are selected in both cases, there will NOT be any callback!.

InstantiateDialog will instantiate the dialog and must be called once.

ShowDialog will show the dialog in the game viewport and can be called multiple times...

To hide the dialog yourself you can call "CloseDialog".

And you are ready to go!

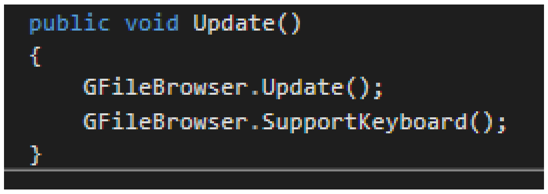
Supporting Keyboard Shortcuts

All that is supported for now is, when user hits the Enter (Return) button the dialog will automatically close and return the files or file he selected, if he hits the Escape Button it will go to the parent directory and if you use the Up or Down arrow key buttons it will select the file-folder approprietly.

To enable this support add to your Update function

1. GFileBrowser.SupportKeyboard();

Just like this :



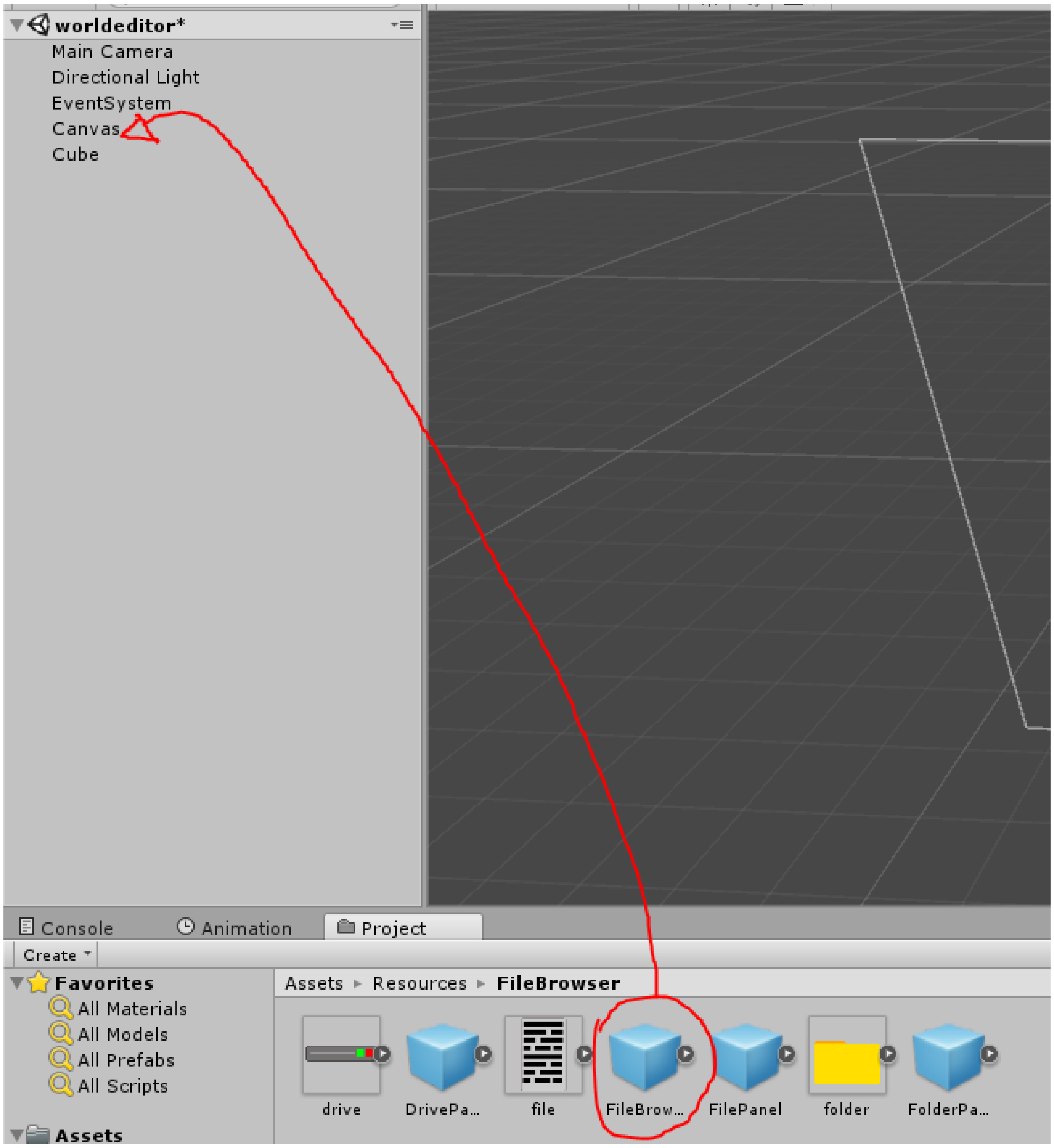
```
public void Update()
{
    GFileBrowser.Update();
    GFileBrowser.SupportKeyboard();
}
```

Customizing the dialog

To actually customize the dialog, is fairly easy to do.

Assuming you have everything correctly installed there should be a prefab on your Resources/FileBrowser called "FileBrowser".

Drag the prefab onto your Canvas and the dialog should be shown

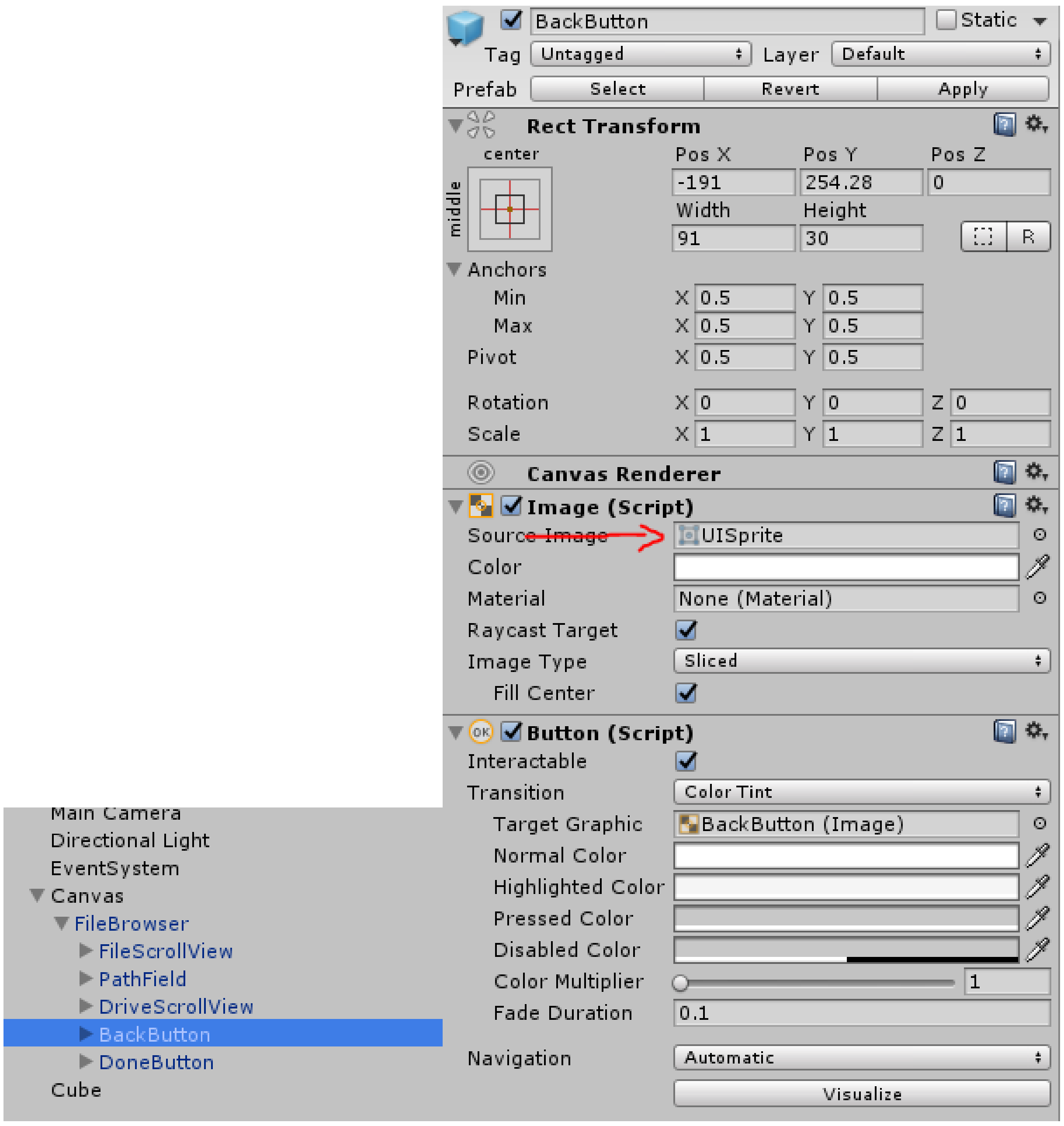


The file browser UI should be visible on your game viewport.

Changing the icons

To change the icons all you do is find the appropriate UI Game Object and change its sprite.

To change the sprites of "Back" and "Done" you select one of them and then in your Inspector you change the button sprite



BackButton ☐ Static
Tag: Untagged
Layer: Default
Prefab: Select | Revert | Apply

Rect Transform

center / middle

	Pos X	Pos Y	Pos Z
	-191	254.28	0
	Width	Height	
	91	30	

Anchors

Min	X 0.5	Y 0.5
Max	X 0.5	Y 0.5
Pivot	X 0.5	Y 0.5
Rotation	X 0	Y 0
Scale	X 1	Y 1

Rotation Z 0; Scale Z 1

Canvas Renderer

Image (Script)

Source Image	UISprite
Color	
Material	None (Material)
Raycast Target	✓
Image Type	Sliced
Fill Center	✓

Button (Script)

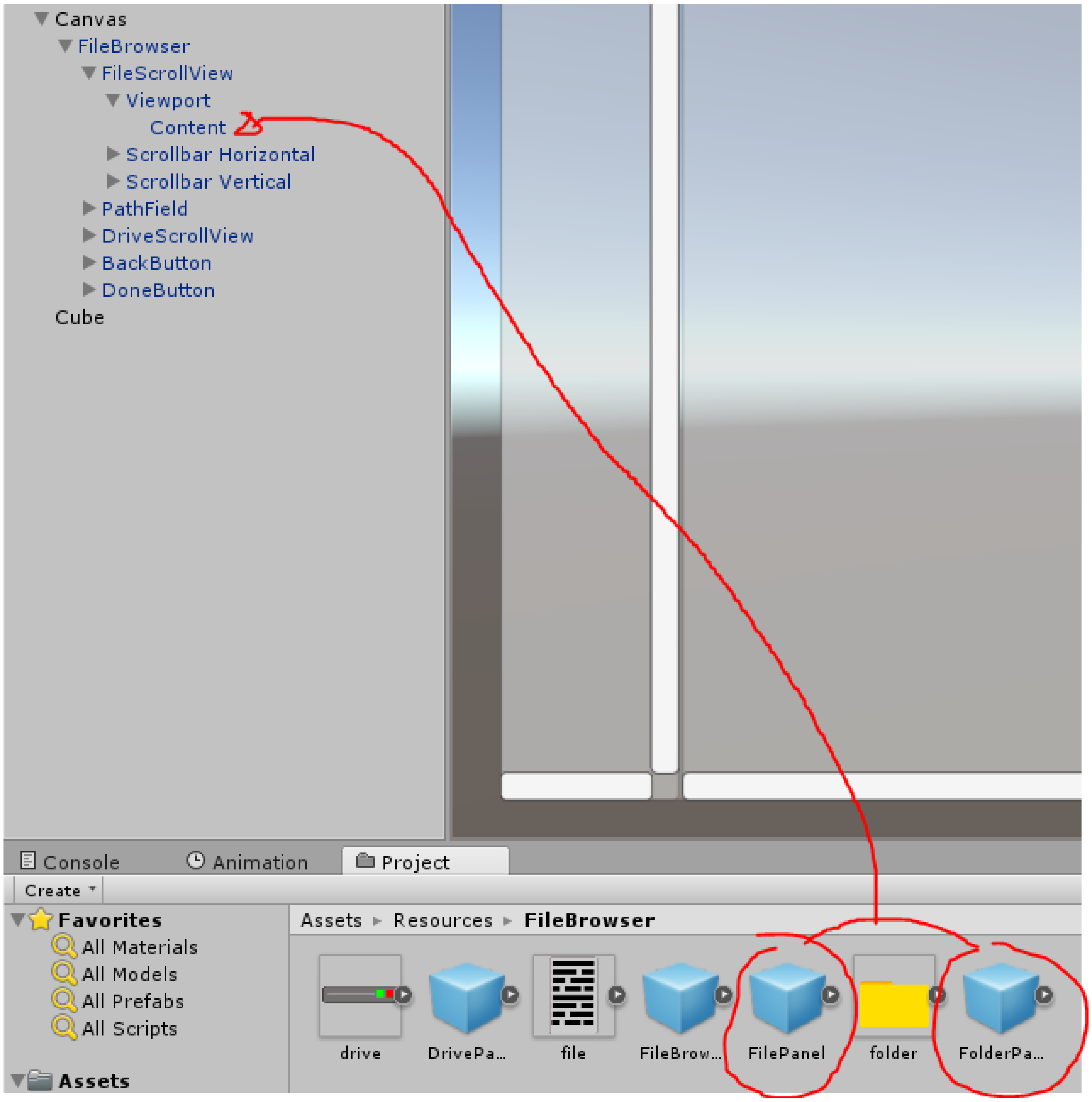
Interactable	✓
Transition	Color Tint
Target Graphic	BackButton (Image)
Normal Color	
Highlighted Color	
Pressed Color	
Disabled Color	
Color Multiplier	1
Fade Duration	0.1
Navigation	Automatic

Visualize

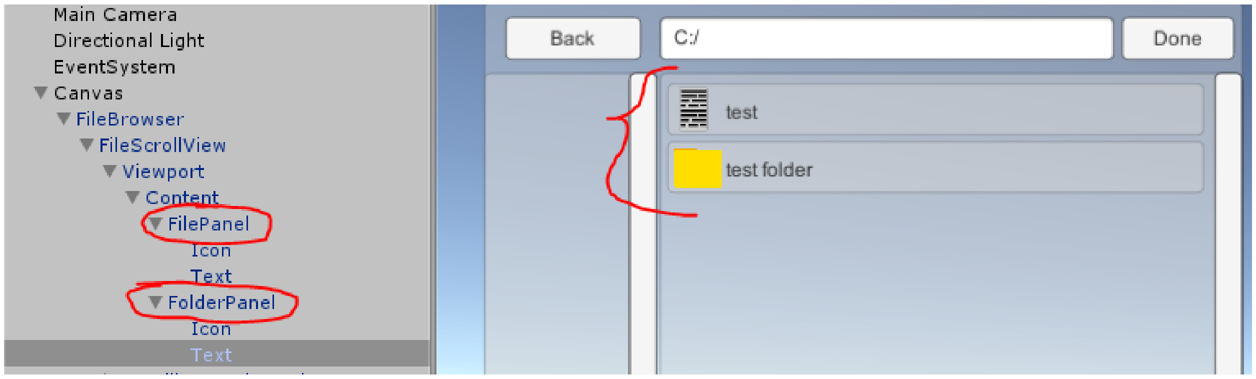
Main Camera
Directional Light
EventSystem
Canvas
- FileBrowser
  - FileScrollView
  - PathField
  - DriveScrollView
  - BackButton
  - DoneButton

Cube

To change the file icon or the folder icon, grab the "FolderPanel" or the "FilePanel" to "FileScrollView/Viewport/Content".

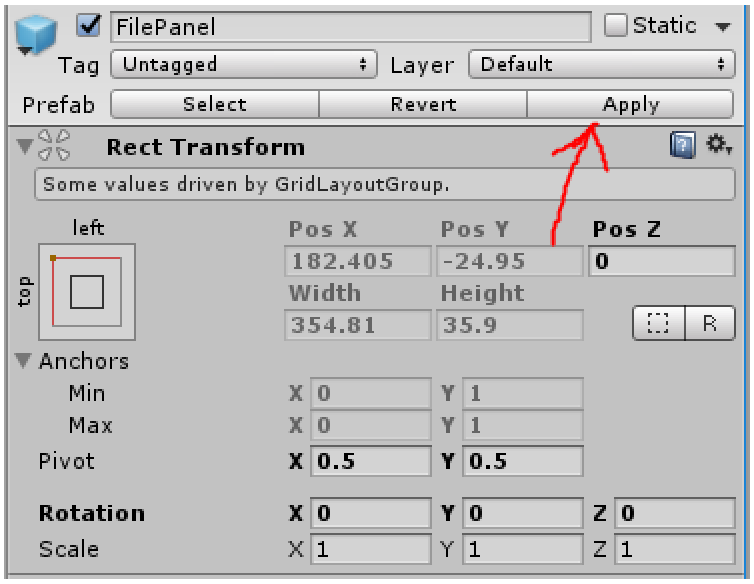


If you have done it correctly they will be shown in the file browser scroll view



You can now go ahead and change their icon (text doesn't matter).

After you have finished making changes to the "FilePanel" or the "FolderPanel" select them and hit apply to the inspector prefab menu



FilePanel ✓ Static
Tag Untagged Layer Default
Prefab

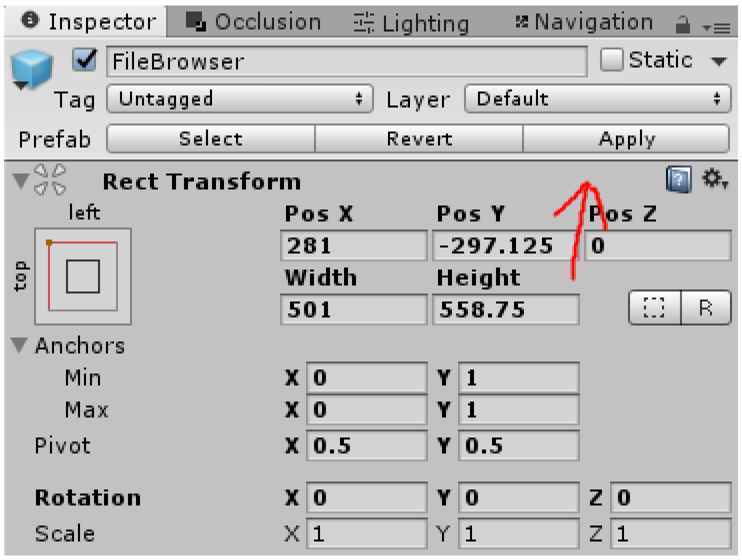
Some values driven by GridLayoutGroup.

	Pos X	Pos Y	Pos Z
	182.405	-24.95	0
	Width	Height	
	354.81	35.9	
Anchors			
Min	X 0	Y 1	
Max	X 0	Y 1	
Pivot	X 0.5	Y 0.5	
Rotation	X 0	Y 0	Z 0
Scale	X 1	Y 1	Z 1

And then remove them from the "Content".Generally the prefab of "FileBrowser" should have the scrollviews empty (content should have no children gameobjects).

Same goes for the DrivePanel but instead of dragging it into the FileScrollView, you must drag it to "DriveScrollView/Viewport/Content".

After finishing all of your changes, select "FileBrowser" and in the inspector prefab menu hit Apply.


Inspector | Occlusion | Lighting | Navigation

FileBrowser | Static

Tag: Untagged | Layer: Default

Prefab: Select | Revert | Apply

Rect Transform

	Pos X	Pos Y	Pos Z
left / top	281	-297.125	0
	Width	Height	
	501	558.75	

Anchors

	X	Y	Z
Min	0	1	
Max	0	1	
Pivot	0.5	0.5	
Rotation	0	0	0
Scale	1	1	1

And then delete "FileBrowser" from your canvas.

Advanced

If you wish to get the objects of the FileBrowser, you can do by using these variables

1. GFileScrollViewContent
2. GDriveScrollViewContent
3. GPathField
4. GBackButton
5. GDoneButton
6. GFavoriteScrollViewContent
7. GContextMenu
8. GBrowserGameObject
9. GRedirectButton

To redirect the dialog to a specific path, you can use the function RedirectTo(string path) or RedirectTo(GFolder folder).

For example

1. GFileBrowser.RedirectTo("D:/Movies");

If you wish to use a GFolder then continue reading about the GClass.

GClass

GClass was created to be used for the File Browser in order to make things easier.

General Classes

GFileBrowser uses the classes of GFolder,GFile and GDrive which keep the folder or file name and its path or the Drive Volume.

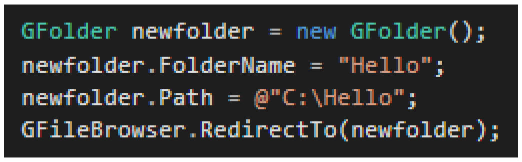
Class of GFile contains 2 variables, "FileName" and "Path".

Class of GFolder contains also 2 variables, "FolderName" and "Path"

Class of GDrive contains only 1 variable, "Volume"

In order to make your own GFolder to use it to redirect to this folder all you need to do is create a new instance of the class and just assign its variables.

For example :



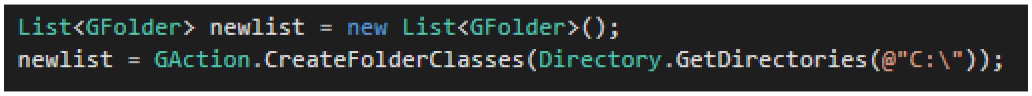
```
GFolder newfolder = new GFolder();
newfolder.FolderName = "Hello";
newfolder.Path = @"C:\Hello";
GFileBrowser.RedirectTo(newfolder);
```

Chances are you will not need GFile or GDrive for the current version of FileBrowser but it's the same procedure for creating a class.

GAction Class

GAction Class can create a list of GFile or a list of GFolder or a list of GDrive from a string array that contains paths.

To create a list, lets take for example all the folders from "C:\" and make them into a list easily using GAction



```
List<GFolder> newlist = new List<GFolder>();
newlist = GAction.CreateFolderClasses(Directory.GetDirectories(@"C:\"));
```

Here we take all the paths from the folders using "Directory.GetDirectories" (System.IO) and then use GAction.CreateFolderClasses so it will automatically create a list.

You can do the same with the functions CreateFileClasses, CreateDriveClasses.

You may use CreateDriveClasses like :

variable_name = GAction.CreateDriveClasses(Environment.GetLogicalDrives());

GSerializer Class

GSerializer is used to save the GFolders and GFiles with playerprefs, so file browser can remember the favorites. It should not be used in any case because it will not work for most of the occasions and may cause problems if used incorrectly.

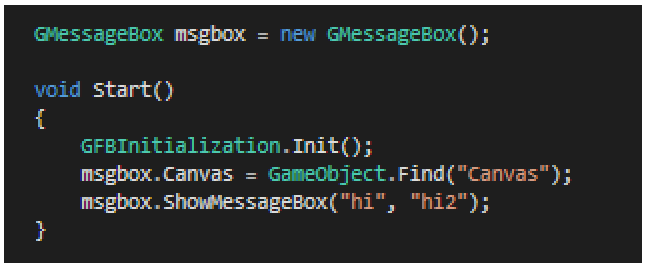
GMessageBox Class

GMessageBox is used to display a messagebox in the middle of the screen with a string.

If you wish to use the GMessageBox, declare a variable of the class, assign a gameobject to its canvas variable and use the function

- ShowMessageBox(string titlestring, string messagestring)

For example :



```
GMessageBox msgbox = new GMessageBox();

void Start()
{
    GFBInitialization.Init();
    msgbox.Canvas = GameObject.Find("Canvas");
    msgbox.ShowMessageBox("hi", "hi2");
}
```

To close the message box using code, simple use the function "CloseMessageBox()"