Simple Windows Manager

First we need to create a new GameObject in scene and attaches it WindowsController.cs, this class manages each window and popup that we'll create.

There's an instance in *Demo* scene.

Create a new Window

```
using UnityEngine;
using System.Collections;

public class UISampleWindow : UIWindow
{
    protected override void Setup()
    {
        Debug.Log("Updated window content.");
    }
}
```

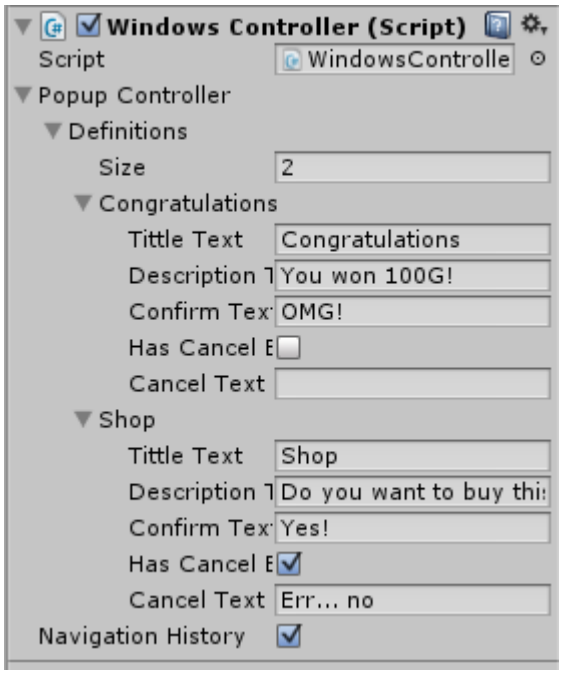
This is an example how UIWindow works, UIWindow is an abstract class, so you can't use directly over a game object, we need to inherit it, in this case we create a new class using UIWIndow, there's an abstract method called Setup(), you should write your window's update there, for example:

```
using UnityEngine;
using System.Collections;

public class UISampleWindow : UIWindow
{
    [SerializeField]
    private Image picture;

    [SerializeField]
    private List<Sprite> sprites;

    private int index = 0;

    protected override void Setup()
    {
        picture.sprite = sprites[index];

        if (index < sprites.Count - 1)
            index++;
        else
            index = 0;
    }
}
```

This means that every time we open this window, it'll assign a new sprite to the image called "picture".

UIWindow contains Show() and Close(), to these methods you should assign to the buttons that you want to use to show or close the window, it also contains Back() this works when we have activate Navigation History in WindowsController, it closes the opened window and opens the last one.

WindowsController also manages popups:



You can access to them writing:

```
 WindowsController.Instance.PopupController.Show(PopupDefinitions.ConfirmCancelExample,
ConfirmMethod CancelMethod, true);
```

It needs a popup definition, that is an enum:

```
public enum PopupDefinitions
{
    ConfirmExample = 0,
    ConfirmCancelExample
}
```

I wrote there the same two options as defined in WindowController, the first one doesn't have "Has Cancel Button" checked, so that's "ConfirmExample" and has as number the index of popup controller definition, in this case 0, it's our first popup, the second one is "ConfirmCancelExample", it doesn't have a number because enum structures automatically set one number in order, the first one is set 0, so the second one should be 1 like index of our second popup in popup controller definition.

It also needs two methods, these methods will be triggered when we press confirm button or cancel button.

The last bool is opcional, "true" if you want to close all windows and delete navigation history when users press Confirm Button.

If you have troubles or want to suggest something, feel free to write me:
picios@outlook.com