GB UNITY TEMPLATE

V2.0 by Milton Guasti

Thanks for checking out this template. Hopefully it'll save you some trouble and prevent you from having to solve shader and input issues during precious game jam time.

These are the main features you'll find:

- **GameBoy color palette shader**: Converts everything you throw at the main camera into a 160x144, four color image. This uses the Universal Rendering Pipeline, and the shader was created with Shader Graph.
- **Input System**: All the buttons of the Game Boy are already mapped to two keyboard sets and gamepad, using the new Unity Input System.
- **Sound System**: Play sounds and music without needing to set individual audio sources on every object.

You can see these features in action in some of the games I released on my Itch.io page.

NEW IN 2.0

- Fixed shader luminance detection on Unity 2022+
- Added mobile friendly touch controls
- Changed default left stick sensitivity to 0.2
- Added new examples featuring canvas and text rendering

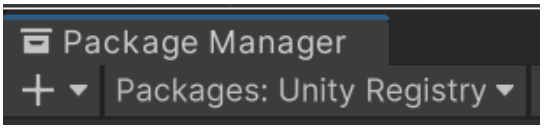
INSTALLATION

I recommend installing the latest LTS version of Unity from the Unity Hub.
This template was tested and packaged using Unity 2022.3.
Remember to install WebGL support too (optional, it's recommended if you're participating in a game jam).

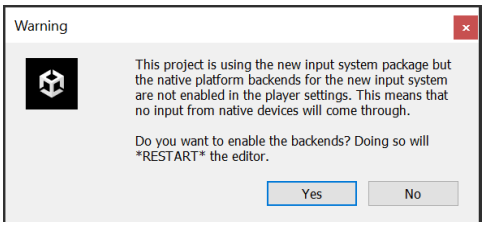
Create a new project.
I recommend using the template called "**Universal 2D Core**". It comes with several useful packages including TextMeshPro.

Once the project is created, select **Window - Package Manager** from the top menu.
Feel free to install any other packages you find useful. The only mandatory requirement for this template is the new input system.

Select **Unity Registry** in the top left drop-down:

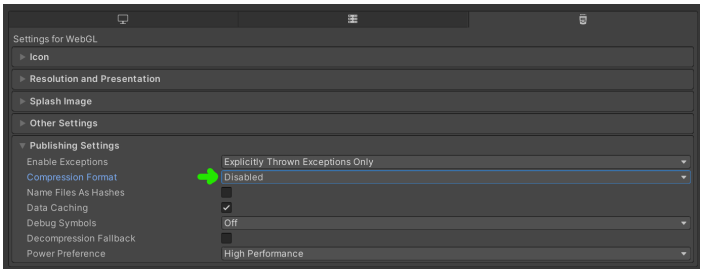


Find "**Input System**" and click **Install** on the bottom right corner.
After the package installs, this message will pop up:



Just click Yes, and wait for the editor to restart.

If you installed WebGL support, select **Edit - Project Settings** in the top menu.
Select the **Player** category, and on the WebGL tab, set Compression Format to **Disabled**.



In the top menu, select: **Assets - Import Package - Custom Package**
Select the **.unitypackage** file you just downloaded, and import everything.
Alternatively, you can drag the file into the Project panel.

Once everything is imported, you should be able to load and run the example scene.
It's located in:
Assets/GBTemplate/Example/Scenes

If you don't need the examples, the entire Examples folder is safe to delete.

NEW SCENE

You only need to add these two prefabs to an empty scene:

- **GBConsole**
- **Main Camera to RenderTexture**

Feel free to start adding content, creating your game controllers, and expanding the camera controller with tools like Cinemachine, etc.

In order to access most of the functionality of the template, you'll need a reference to GBConsoleController. This class is a singleton, and you can reference it pretty much anywhere.

CONSOLE CONTROLLER

Implementing gameplay, entities, game states, sprites, etc. shouldn't be any different than usual. However, when you need to access the "hardware" of the simulated GameBoy console you'll use methods inside GBConsoleController.

There's three controllers with useful methods within this object:

So, a typical script would contain:

```
private GBConsoleController gb;

void Start()
{
    //Getting the instance of the console controller, so we can access
its functions
    gb = GBConsoleController.GetInstance();
}
```

For example, if we want to play an AudioClip when the A button is pressed:

```
if (gb.Input.ButtonAJustPressed)
{
    gb.Sound.PlaySound(exampleSoundA);
}
```

DISPLAY

In this controller you'll find methods to change the color palette at runtime and fade the whole screen.

Color palettes are stored in an array. They are scriptable objects located in *Assets/GBTemplate/Palettes*.

COLOR PALETTES

```
UpdateColorPalette(int palNum)
PaletteCycleNext()
PaletteCyclePrev()
```

You can change the color palette during run time by specifying its index in the palette array or by cycling through them. These last ones are useful for an options screen.

FADE COROUTINES

```
FadeToBlack(float fadeSpeed)
FadeFromBlack(float fadeSpeed)
FadeToWhite(float fadeSpeed)
FadeFromWhite(float fadeSpeed)
```

These are coroutines that fade the screen to the darkest or brightest color. This does not alter the color palette of the display, but the input value of the shader. That means that fades will look like they did on the GameBoy.

A typical use case for the fade coroutines would be:

```
public IEnumerator ChangeScene()
{
    yield return gb.Display.StartCoroutine(gb.Display.FadeToBlack(2));

    //Insert your action / scene transition here

    yield return gb.Display.StartCoroutine(gb.Display.FadeFromBlack(2));
}
```

SOUND

Instead of placing audio sources in every object, sound is centralized within the console. You can send an AudioClip to be played by the console directly.

There are three audio sources implemented: Two for sound effects and one for background music.
In most cases, only one source for sound effects will be enough. If you need an extra looping sound playing besides the music (alarms, car engine, sirens, etc.), you can use this extra source.

SOUND PLAYBACK

```
PlaySound(AudioClip clip)
PlaySound(AudioClip clip, int channel)
LoopSound(AudioClip clip)
LoopSound(AudioClip clip, int channel)
PlayMusic(AudioClip clip)
PlayMusicOneShot(AudioClip clip)
StopAllSounds()
StopMusic()
```

If no channel is specified, the default is the first channel (0).

SOUND VOLUME

```
UpdateSoundVolume(float newVolume)
UpdateMusicVolume(float newVolume)
UpdateGlobalVolume(float newVolume)
```

If you add an options screen, you can use these to change the volume of the audio sources.

FADE OUT

```
FadeOutSoundChannel(int channel, float fadeTime)
FadeOutMusic(float fadeTime)
```

These will fade out the currently playing AudioClip over time, playback will stop once finished.

If we need to play an AudioClip, a typical use case would be:

```
gb.Sound.PlaySound(exampleSoundA);
```

INPUT

This controller features public properties describing the state of the console input. You should be able to access them directly once you have a reference to GBConsoleController.

DeadZone	float	Deadzone value for gamepad sticks, default is 0.2
Up **Down** **Left** **Right** **ButtonA** **ButtonB** **ButtonSelect** **ButtonStart**	bool	Returns true if the button/direction is being held
UpJustPressed **DownJustPressed** **LeftJustPressed** **RightJustPressed** **ButtonAJustPressed** **ButtonBJustPressed** **ButtonSelectJustPressed** **ButtonStartJustPressed**	bool	Returns true if the button/direction was pressed this frame
UpPressedTime **DownPressedTime** **LeftPressedTime** **RightPressedTime** **ButtonAPressedTime** **ButtonBPressedTime** **ButtonSelectPressedTime** **ButtonStartPressedTime**	float	Returns the time in seconds the button/direction has been pressed
ButtonAJustReleased **ButtonBJustReleased**	bool	Returns true if the button was released in this frame

In this example, if we start holding the A button a sound will start playing. It won't try to play the sound further if we hold the button.

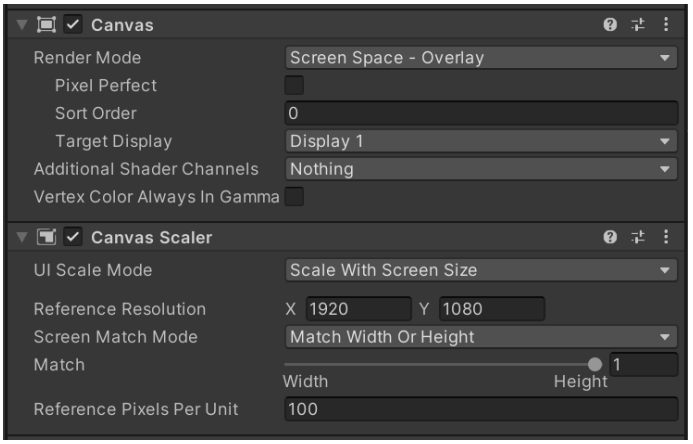
```
if (gb.Input.ButtonAJustPressed)
{
    gb.Sound.PlaySound(exampleSoundA);
}
```

TOUCH CONTROLS

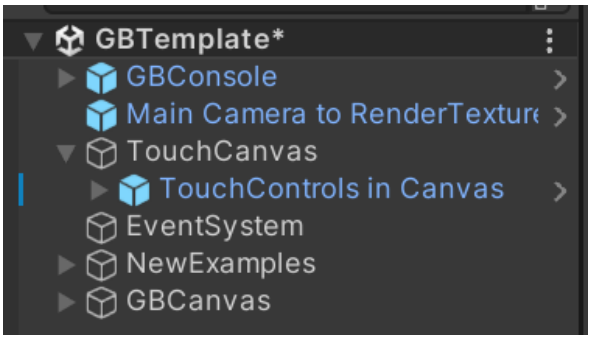
Thanks to the new Unity Input System, implementing touch controls can be very straightforward. A prefab containing a landscape layout is included in the template.

To enable touch controls in your game, first create a Canvas in your scene. An **EventSystem** object will be created too. Do not delete it, or the button presses won't register.

Apply these settings to the Canvas component:

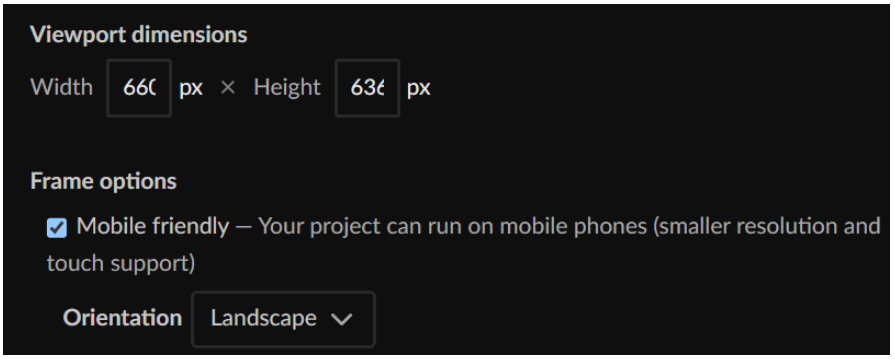


You can then drag the **TouchContros In Canvas** prefab under the recently created Canvas:



The example scene has touch controls already set up.
From here on, you can customize the layout to your liking.

If you're deploying a WebGL build on itch.io, you can set these parameters to enable touch controls whenever you view the page on mobile:



EXAMPLE CONTENT

All of the examples are now organized in a menu. They use a canvas set to **Screen Space - Camera**, and text is being rendered using TextMeshPro. The free font **Early GameBoy** is already imported. If you want to add more fonts, check out the General Tips section.

MAIN MENU

You can choose to enable specific groups in the hierarchy, using **SimpleMenuController** to control the cursor and trigger events on button presses. You will find the reference to these events in the Start() method of **GBTemplateExamplesManager**.
Video on Observer Pattern

INPUT TESTER

This example shows a canvas overlay of a joypad with buttons indicating the user input. There's also a worldspace group with a simple movement script. These can be used as the early building blocks of most games.

OPTIONS MENU

This screen expands on the simple menu, adding commands for left and right input. Values are updated directly with a reference to **GBConsoleController**.

CONFIRM MENU

An example of a horizontal menu, using the same simple menu controller. The result of the player's choice is controlled using Unity Events.
Video on Unity Events

DIALOG EXAMPLE

This component is quite complex, but extremely versatile. **DialogBoxController** takes care of animating the panel and the typing text. You will need to create **DialogData** scriptable object resources, and a reference to **DialogBoxController**.
The provided example chooses random pieces of dialog from a list.

FADE EXAMPLE

An example of calling a fade to black/white coroutine, executing a piece of code (in this case rotating the starfield), and calling a fade from black/white to complete a transition.

GENERAL TIPS

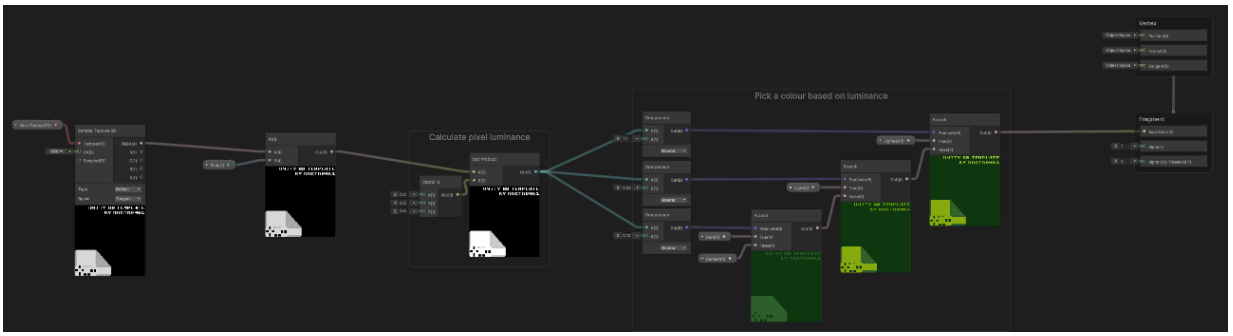
SPRITES

The shader will output four colors. If you want to have precise control over the colors that will be displayed, you can use the following colors in your sprites:

Darkest	#000000
Dark	#a8a8a8
Light	#d9d9d9
Lightest	#ffffff

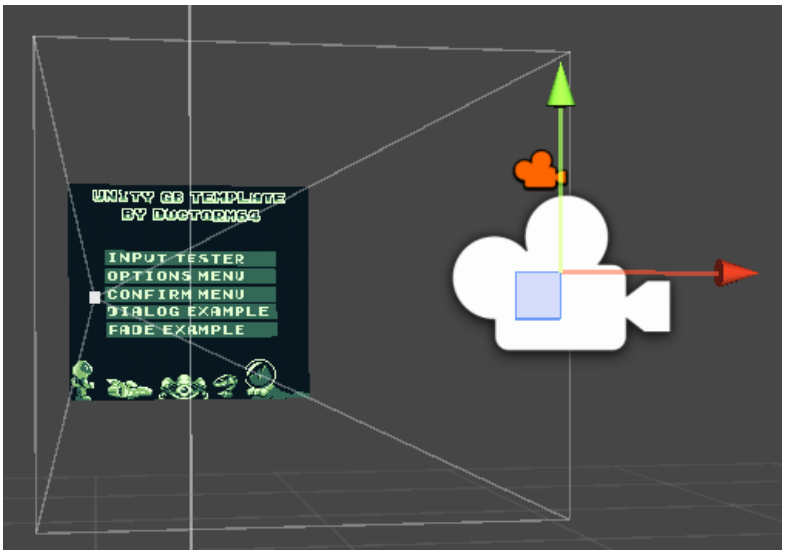
GAMEBOY SHADER

I originally used a shader based on [THIS TUTORIAL](#).
I later remade it using ShaderGraph to add the screen fade functionality.



You can find it in *Assets/GBTemplate/Saders/GameBoy*

The render texture is applied to a Quad, where the main camera is pointing.
These are 100 units above the origin point (y = 100), so try to stay away from that location.



2D PACKAGES

In the latest Unity versions, the project template “**Universal 2D Core**” comes with a bundle of useful 2D tools already in the package manager.
The official Aseprite importer and the Tileset tools have proven to be very useful in game jam scenarios. You should explore the rest of the packages, you might find something useful.

TEXT

You can use the provided font, but if you want to import more fonts to use with TextMeshPro, you can follow the steps in this tutorial:
https://pavcreations.com/pixel-perfect-fonts-in-unity-the-practical-guide/

EXAMPLE CONTENT CREDITS

Font:
https://www.dafont.com/es/early-gameboy.font

Music:
https://modarchive.org/index.php?request=view_by_moduleid&query=191117

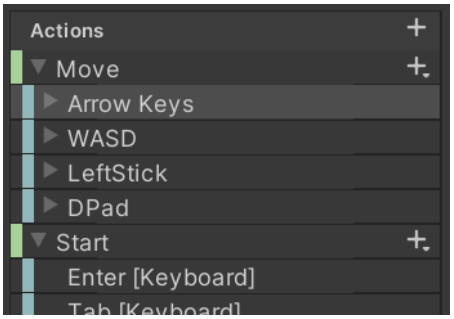
FAQ

- I'm getting this exception when importing the package:

```
Assets\GBTemplate\Input\InputMaster.cs(15,19): error CS0234: The type or
namespace name 'InputSystem' does not exist in the namespace
'UnityEngine' (are you missing an assembly reference?)
```

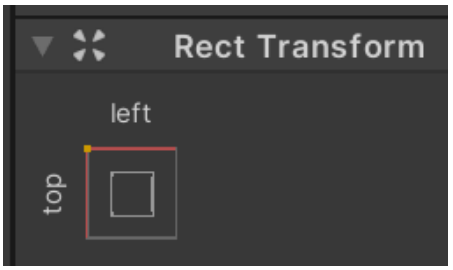
You need to install the new input system using the Unity Package Manager. Scroll up and read the install instructions again.

- How do I change the control bindings?
In the Project tab, go to *Assets/GBTemplate/Input*.
You'll find an asset called InputMaster. Double click it, and select the action you want to edit. Each one has multiple bindings.



Video tutorial on control bindings

- The text in my UI Canvas shakes a lot when moving the camera!
Try setting the anchor point of the text to one of the corners instead of the center.



- What's the license for this template?
CC0 - Public Domain - Feel free to use all of this however you like. Credit is not necessary but appreciated. Also, if you use this in your game please let me know. It's great to see how my work helped others.

- Do you provide tech support for this? I need a feature!
Sorry, I'm sharing these resources I made in my free time, which is very scarce lately. Feel free to ask, but I can't promise I'll respond in time.

- How can I contact you?
Email: doctorm64[at]live.com
Itch.io page - Twitter