Power Joysticks

User Guide V1.0

The Little Game Factory

Content

Overview 2

Getting Started 2

Power Joystick / D-Pad 3

- Basic Settings 3
- Layout 8
- Customize 10
- Axis to Button 13
- Events 13

Power Button 14

- Basic Settings 14
- Layout 15
- Customize 17
- Button to Axis 20
- Events 21

Simple Texture Editor 22

- Overlay Texture 23
- Combine 2 Textures 23
- Tint Texture 24
- Change Brightness and Contrast 24
- Invert Texture 25

Good To Know 26

Public Functions List 28

Overview

The Power Joysticks package is an easy to use mobile input solution. It includes powerful, flexible and customizable Joysticks, D-Pads and Buttons. The package uses unity's cross platform input manager to handle inputs. This documentation was designed to provide you with an overview of the features and functionality of the Power Joysticks package. However, it is recommended that you try out the prefabs and demo scene to get a quicker and better understanding of all the features.

Getting Started

Before you can use the Power Joysticks package in your project, you need to import the CrossPlatformInput package of the standard assets.

Go to Assets -> Import Package -> CrossPlatformInput

Switch the platform in the Build Settings to Android or iOS.

After you have imported the CrossPlattformInput package you can import the Power Joysticks package into your project.

Steps to setup your first controller:

1. Drag and drop the PowerJoystickCanvas and EventSystem prefabs into your scene. (Assets/PowerJoysticks/Prefabs/)
2. Drag and drop a joystick / d-pad / button prefab into the PowerJoystickCanvas game object. (Assets/PowerJoysticks/Prefabs/...)
3. Alternatively to step 1 and 2 you can also drag and drop a premade controller into your scene. (Assets/PowerJoysticks/Prefabs/Controller/) + (Assets/PowerJoysticks/Prefabs/EventSystem.prefab)
4. Press Play to test the joystick / d-pad / button / controller.

You can preview and test the joystick, d-pad and button prefabs in the DemoScenes. (Assets/PowerJoysicks/DemoScenes/).

Power Joystick / Power D-Pad

The Power Joystick and the Power D-Pad shares almost the same settings:

Basic Settings

Axis

Select whether you only want to use the horizontal or vertical axis, or both axes of the joystick / d-pad. If you use the option Axis to Button (see below), the corresponding axis must be included in the selection. If this axis is only used as button input, the field for the name of the axis can remain empty. To still display visual effects for this axis you can assign any unique name to it.

Horizontal Axis Name

The name of the horizontal axis that the cross platform input manager will register. CrossPlatformInputManager.GetAxis(Horizontal Axis Name) returns the value of the virtual horizontal axis.
Make sure that you assign a name for an axis only once.

Vertical Axis Name

The name of the vertical axis that the cross platform input manager will register. CrossPlatformInputManager.GetAxis(Vertical Axis Name) returns the value of the virtual vertical axis.
Make sure that you assign a name for an axis only once.

Movement Rang (Joystick only)

The movement range of the thumbstick:

Dead Zone
Defines the dead zone of the joystick / d-pad.

The Dead Zone is an area around the center of the joystick that does not recognize any input when the thumbstick is moved within this area.

Joystick Mode (Joystick only)
Select one of the following joystick modes:

1. Analog: For fully analog movement.
2. Analog Eight Directions: For analog movement in eight directions.
3. Analog Four Directions: For analog movement in four directions.
4. Digital Eight Directions: For digital movement in eight directions.
5. Digital Four Directions: For digital movement in four directions.

For digital input the axis values are either 0 or 1.
It is recommended to define a dead zone for all modes except the Analog mode.

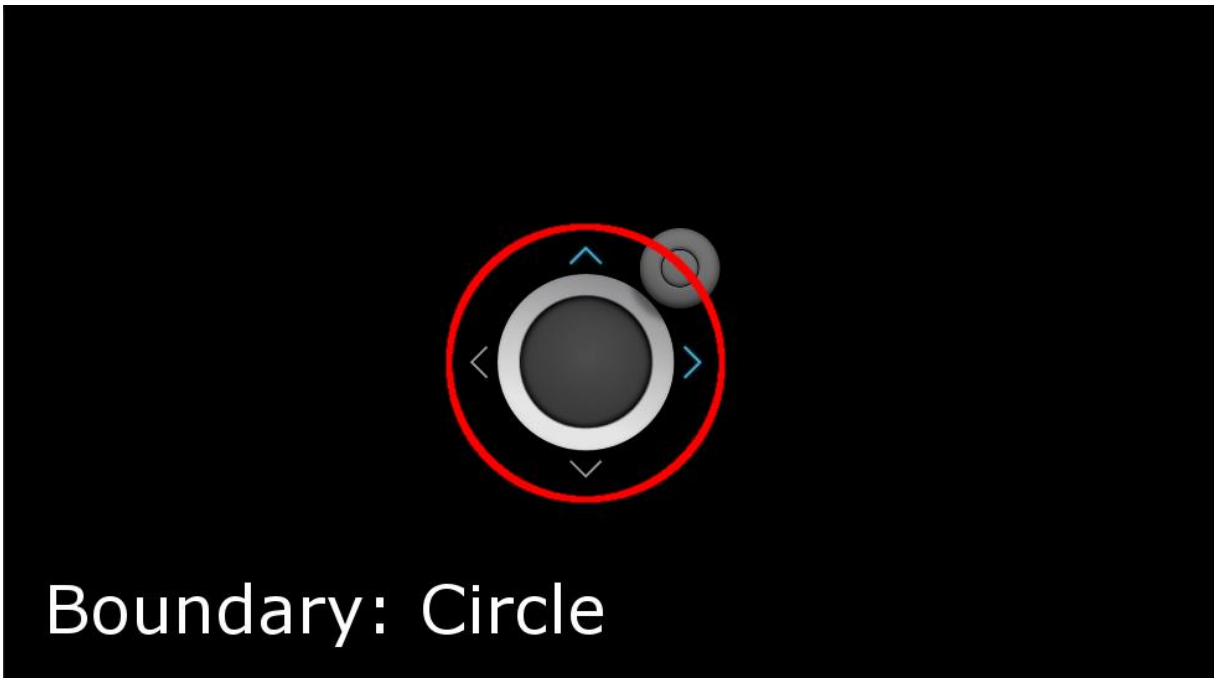
D-Pad Mode (D-Pad only)
Select one of the following d-pad modes:

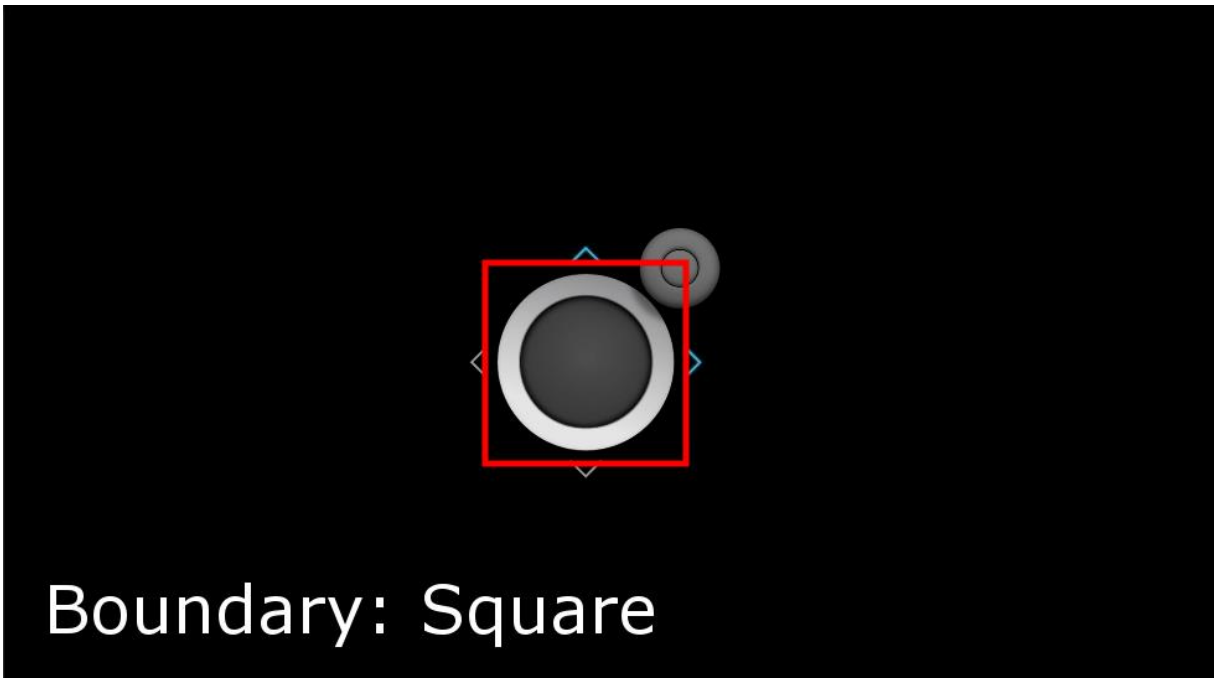
1. Digital Eight Directions: For digital movement in eight directions.
2. Digital Four Directions: For digital movement in four directions
3. Analog Eight Directions: For analog movement in eight directions.
4. Analog Four Directions: For analog movement in four directions.
5. Analog: For fully analog movement.

For digital input the axis values are either 0 or 1.
It is recommended to define a dead zone for all modes except the Analog mode.

Boundary (Joystick only)

The shape of the boundary of the joystick, circle or square:





Visibility Mode

Select one of the following visibility modes:

1. Always Visible (Note: Static joysticks and Static d-pads are always visible)
2. On Touch: (For dynamic position mode only – joystick / d-pad is only visible if the user touches the device)
3. At Start and On Touch (For dynamic position mode only – joystick / d-pad is only visible if the user touches the device and at the start of the scene)
4. Only at Start: (For dynamic position mode only – joystick/d-pad is only visible at the start of the scene)
5. Not Visible: (For dynamic position mode only – joystick / d-pad is invisible)

Reference Camera

The reference camera is responsible for positioning and scaling the joystick / d-pad. The default reference camera is the main camera, which should be correct in most cases.

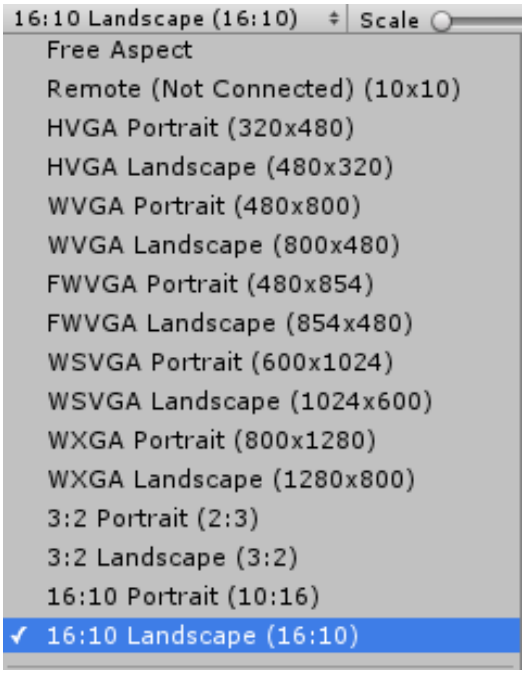
Make sure that there is always a camera tagged “Main Camera” in your scene when you drag a joystick / d-pad into the scene. Otherwise the joystick / d-pad will not be initialized.

Layout

Orientation

You can define individual positions and scaling for Landscape and Portrait orientation.

Choose the orientation for which you want to edit position and scaling values and make sure that the corresponding orientation is selected in the editor too:



Joystick / D-Pad Alignment

Sets the alignment of the joystick / d-pad.
Left Bottom, Right Bottom, Left Top, Right Top

Joystick / D-Pad Position X

Sets the x / horizontal position of the joystick / d-pad on the screen.

Joystick / D-Pad Position Y

Sets the y / vertical position of the joystick / d-pad on the screen.

Joystick / D-Pad Size

Sets the size of the joystick / d-pad

Position Mode
Select Static for a static joystick / d-pad that stays in place.

Select Dynamic to define a larger area in which you can touch and use the joystick / d-pad. The joystick / d-pad then appears at the touch location.

Dynamic Area Position X (Dynamic position mode only)
Sets the x / horizontal position of the dynamic touch area on the screen.

Dynamic Area Position Y (Dynamic position mode only)
Sets the y / vertical position of the dynamic touch area on the screen.

Dynamic Area Width (Dynamic position mode only)
Sets the width of the dynamic touch area.

Dynamic Area Height (Dynamic position mode only)
Sets the height of the dynamic touch area.

Snap Back to Start Position (Dynamic position mode only)
If this option is enabled the joystick / d-pad will snap back to its original defined position when the finger is lifted.

Show / Hide Dynamic Area in Editor
Shows or hides the dynamic touch area. (The touch area is always hidden in play mode)

Customize

Joystick Sprite (Joystick only)
Select / Change the joysticks (background) sprite.

Joystick Tint (Joystick only)
Tints the joystick sprite with the selected color. You can change the opacity of the sprite here too.

Thumbstick Sprite (Joystick only)
Select / Change the thumbstick sprite.

Thumbstick Tint (Joystick only)
Tints the thumbstick sprite with the selected color. You can change the opacity of the sprite here too.

D-Pad Sprite A / D-Pad Sprite B / D-Pad Sprite C (D-Pad only)
Select / Change the d-pad sprites. One d-pad consists of three sprites for direction visualization.

D-Pad Tint (D-Pad only)
Tints the d-pad sprites with the selected color. You can change the opacity of the sprite here too.

Color Fade EFX
Set start and end colors for the joystick and thumbstick sprites here to create a color fade / lerp effect.

On touch, the sprites colors will lerp from start colors to end colors in the given time (Fade Time).

When the finger is lifted again, the sprites colors will lerp from end colors to start colors in the given time.

The previously set colors/tints will be overwritten if this effect is used.

Scale EFX

On touch, the complete joystick / d-pad including all EFX layers will be scaled by the Scale Factor in the given Scale Time.

Arrows EFX / Direction Indicator EFX A / Direction Indicator EFX B

These three EFX slots are further graphic layers for the joystick. They share all the same functionality and are designed to visualize the axis values (like a direction indicator).

Sprite: The EFX Sprite (Will be cloned and rotated by 90 degrees four times).

Tint Color at Zero: The color of the sprite if the corresponding axis returns a value of 0.

Tint Color at One: The color of the sprite if the corresponding axis returns a value of 1.

Fade Out Speed: Defines how fast the color will lerp back to Tint Color at Zero when the finger has been lifted and all axis returns 0.

Flash: If set to true the sprite will fade out and in in a sinus curve with the given frequency (Flash Speed).

The easiest way to understand these EFXs is to test one of the premade prefabs.

Rotation EFX A / Rotation EFX B / Rotation EFX C

These three EFX slots are further graphic layers for the joystick. They share all the same functionality and are designed to visualize the axis values by rotating a sprite towards the corresponding direction of the axis input values.

Sprite: The EFX

Tint Color at Zero: The color of the sprite if the corresponding axis returns a value of 0.

Tint Color at One: The color of the sprite if the corresponding axis returns a value of 1.

Fade Out: If set to true, the EFX sprite will fade out (alpha = 0) if the finger has been lifted, even if the Tint Color at Zero has an alpha value greater then zero. If set to false, the EFX sprite will lerp to Tint Color at Zero and stays on screen pointing to the direction of the last corresponding axis input.

Fade Out Speed: Defines how fast the color will fade out when the finger has been lifted and all axis returns 0.

Flash: If set to true the sprite will fade out and in in a sinus curve with the given frequency (Flash Speed).

Rotation Offset Angle: Changes the rotation of the efx sprite.

The easiest way to understand these EFXs is to test one of the premade prefabs.

Force Analog Color Lerp for Digital Input (for all Direction / Rotation EFX slots)

All EFX colors will lerp analog and based on the touch position even if the joystick / d-pad mode is set to digital and axis will only either return 0 or 1.

Shadow EFX

Creates a dropshadow for the joystick / d-pad.

Sprites: Select the shadow sprite.
Tint: Sets the color of the shadow sprite.

Remove unused EFX Sprites / Images

If you are satisfied with all the EFX settings you can press this button to remove unused sprites / images from the joystick / d-pad. Otherwise unused textures may be included in the sprite atlas / build of your game.

Axis to Button

Use / Don't Use Positive / Negative X / Y Button as Axis
You can use any axis of the joystick / d-pad to trigger a button.

Button Name
The name of the button that the cross platform input manager will register.

CrossPlatformInputManager.GetButton(Button Name),
CrossPlatformInputManager.GetButtonDown(Button Name),
CrossPlatformInputManager.GetButtonUp(Button Name)

returns the states of the virtual button.

Make sure that you assign a name for a button only once.

Trigger Button At
If the axis input value is greater or equal to this value, the button will be triggered. (GetButtonDown = true / GetButton = true)

If the axis input value is smaller to this value, the button will be released. (GetButtonUp = true)

Events

Enable / Disable Tap Events
You can tap the joystick / d-pad multiple times to trigger an OnTap event.

Tap Count
Tap count to trigger an event.

Tap Time Window
Time in which a tap must be made to count.

Tap Completed
Defines if the tap is registered on touch or when the finger has been lifted

Power Button

Basic Settings

Button Name
The name of the button that the cross platform input manager will register.

CrossPlatformInputManager.GetButton(Button Name),
CrossPlatformInputManager.GetButtonDown(Button Name),
CrossPlatformInputManager.GetButtonUp(Button Name)

returns the states of the virtual button.

Enable Auto Fire
Enables / Disables auto fire mode: GetButtonDown and GetButtonUp are triggered in a repetitive interval when the button is pressed.

Auto Fire Delay
Defines the auto fire delay.

Reference Camera
The reference camera is responsible for positioning and scaling the button. The standard reference camera is the main camera, which should be correct in most cases.

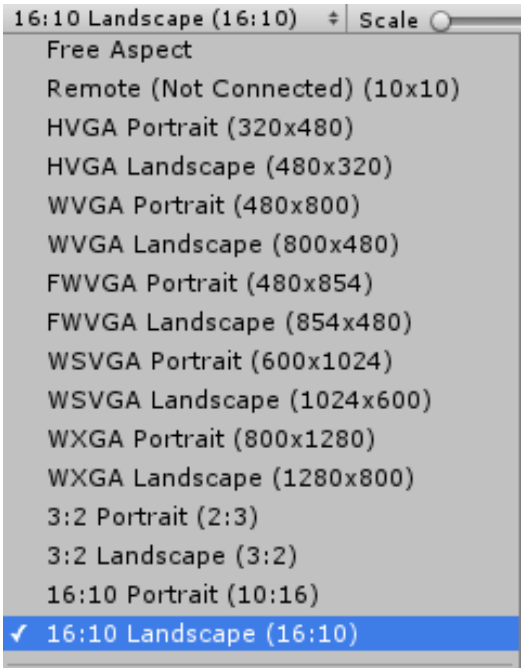
Make sure that there is always a camera tagged “Main Camera” in your scene when you drag a button into the scene. Otherwise the button will not be initialized.

Layout

Orientation

You can define individual positioning and scaling for Landscape and Portrait orientation.

Choose the orientation for which you want to edit position and scaling values and make sure that the corresponding orientation is selected in the editor too:



Button Alignment

The alignment of button:Left Bottom, Right Bottom, Left Top, Right Top

Button Position X

Sets the x / horizontal position of the button on the screen.

Button Position Y

Sets the y / vertical position of the button on the screen.

Button Size

Sets the size of the button.

Button Rotation
The rotation of the button. The Power Up, Refresh and Shadow layers are not affected by this value.

Touch Area Size
Sets the size of the touchable area around the button.

Show / Hide Touch Area in Editor
Shows / Hides the touch area in editor (the touch area is always hidden in play mode)

Customize

Button Sprite
The Sprite of the button.

Button Tint
Color of the button.

Button Hover Sprite
The Sprite of the button when pressed. (can be left empty).

Button Hover Tint
Color of the button when pressed.

Color Fade EFX
Set start and end colors for the button sprites here to create a color fade / lerp effect.

On touch, the sprites colors will lerp from start colors to end colors in the given time (Fade Time).

When the finger is lifted again, the sprites colors will lerp from end colors to start colors in the given time.

The previously set colors/tints will be overwritten if this effect is used.

Scale EFX
On touch, the complete button including all EFX layers will be scaled by the Scale Factor in the given Scale Time.

Underlay EFX / Overlay EFX A / Overlay EFX B
These three EFX slots are further graphic layers for the button. They share all the same functionality and are designed to visualize the button state (pressed or released).

Sprite: The EFX Sprite.

Underlay / Overlay Start Color: The color of the sprite if the button is not pressed.

Underlay / Overlay End Color: The color of the sprite if the button is pressed.

Fade In Time: Defines how fast the color will lerp to End Color when the button is pressed.

Fade Out Time: Defines how fast the color will lerp back to Start Color when the finger has been lifted and the button state is released.

Power Up EFX

This EFX slot is a further graphic layer for the button. It is designed to visualize the button state (the button is pressed for a certain time).

Sprite: The EFX Sprite. (Gets filled over time when the button is pressed)

Power Up Start Color: The color of the sprite if the button is not pressed.

Power Up End Color: The color of the sprite if the button is pressed for the Power Up Time + Power Up Delay (time) and the sprite is 100% filled.

Power Up Delay: Defines the time the button has to be pressed before the Power Up Time is counted up and the sprite gets filled.

Power Up Time: The time in which the sprite gets filled and the color will lerp from Start Color to End Color

Power Up Cancel Speed: The speed at which the power up time decreases.

If set to max, the current power up time will instantly set to 0 – Power Up Delay.

Power Up Origin: Sets the origin point of the fill process.

Flashing at End: If set to true and the sprite is completely filled, it will fade out and in in a sinus curve with the given frequency (Flash Speed).

If the finger is lifted and the EFX sprite is completely filled (the Power Up Time + Power Up Delay time has passed) an OnPowerUp event will be triggered.

Refresh EFX

This EFX slot is a further graphic layer for the button. It is designed to visualize the button state (refreshing / inactive). After pressing the key, the key is inactive for the duration of the refresh time.

Sprite: The EFX Sprite. (Filling decreases over time)

Refresh Start Color: The color of the sprite if the button is not refreshing.

Refresh End Color: The color of the sprite if the button is refreshing, the Refresh Time has passed, and the sprite is completely unfilled.

Refresh Time: Sets the time in which the button will be refreshed.

Refresh Origin: Sets the origin point of the unfill process.

If the Refresh Time has passed and the button is active again, an OnRefreshed event will be triggered.

Shadow EFX

Creates a dropshadow for the button.

Sprite: Select the shadow sprite.
Tint: Sets the color of the shadow sprite.

Remove unused EFX Sprites / Images

If you are satisfied with all the EFX settings you can press this button to remove unused sprites / images from the button. Otherwise unused textures may be included in the sprite atlas / build of your game.

Button to Axis

Enable / Disable Button to Axis

You can use the button to set an axis input value.

Axis Name

The name of the axis that the cross platform input manager will register.

Negative Axis

Axis value will be -1 instead of 1 on button down.

Events

Enable / Disable Tap Events

You can tap the button multiple times to trigger an OnTap event.

Tap Count

Tap count to trigger an event.

Tap Time Window

Time in which a tap must be made to count.

Tap Completed

Defines if the tap is registered on touch or when the finger has been lifted.

Enable / Disable Hold and Release Events

You can press the button for a given time and then release it to trigger a OnHoldAndRelease event.

Hold Time

Time that the button has to be pressed to trigger the OnHoldAndRelease event on release.

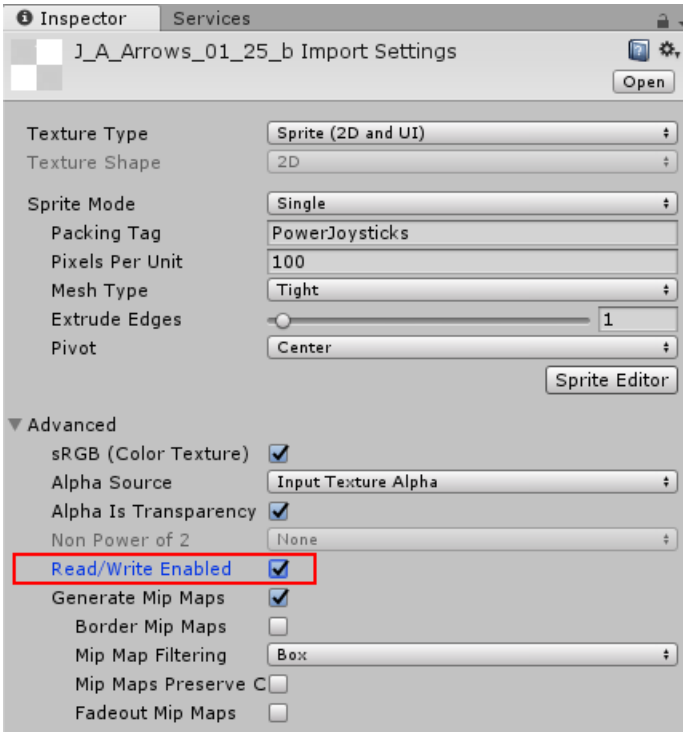
Simple Texture Editor

By combining different sprites, adjusting the color and activating / deactivating the EFX layers, many different designs can already be realized.

To create even more individual designs, the Power Joysticks package contains additional tools for simple texture editing.

Please note that these tools are in an experimental state and are designed to handle the Power Joysticks textures with a maximum size of 512 x 512 pixels. It is not recommended to edit bigger textures with these tools.

Textures must be made Read/Write Enabed to be edited with these tools:



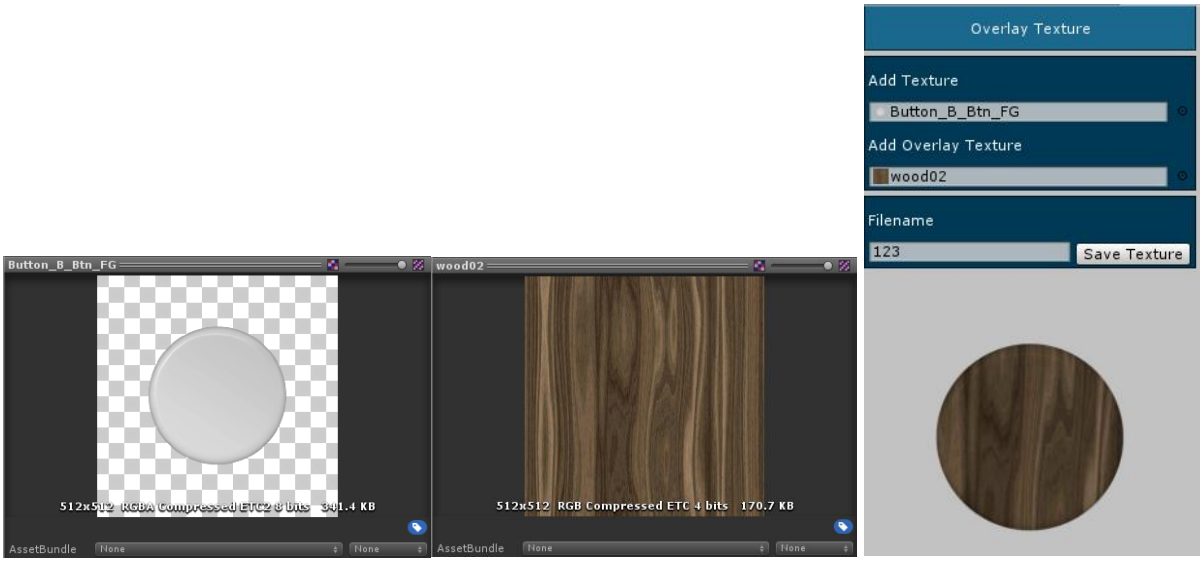
Go to **"Window/The Little Game Factory/Power Joysticks/Simple Texture Editor"** to open the tools.

All edited textures will be saved to:
Assets/PowerJoysticks/UserTextures/filename_powerjoysticks.png

All textures in your project containing _powerjoysticks.png in their filename are imported with special import settings. If you want to change the import settings of these textures you will have to rename them.

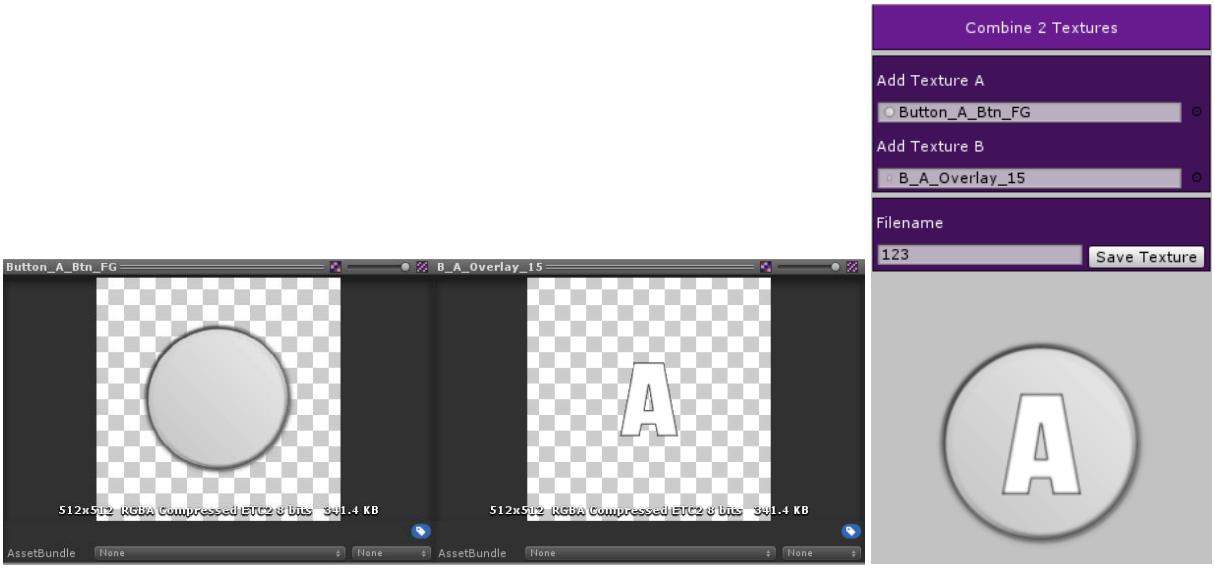
Overlay Texture

Overlay a texture with another:



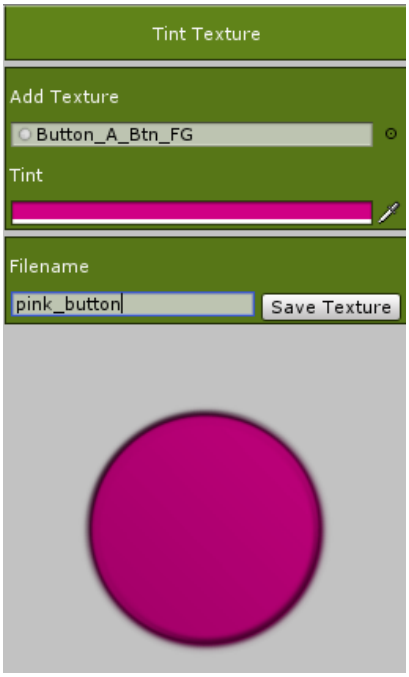
Combine 2 Textures

Combine two textures:



Tint Textures

Tint a texture:

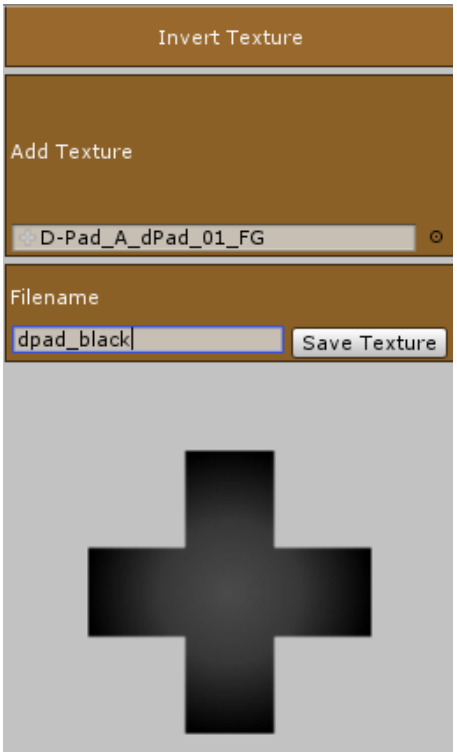


Change Texture Brightness and Contrast

Change the brightness and the contrast of a texture:

Invert Texture

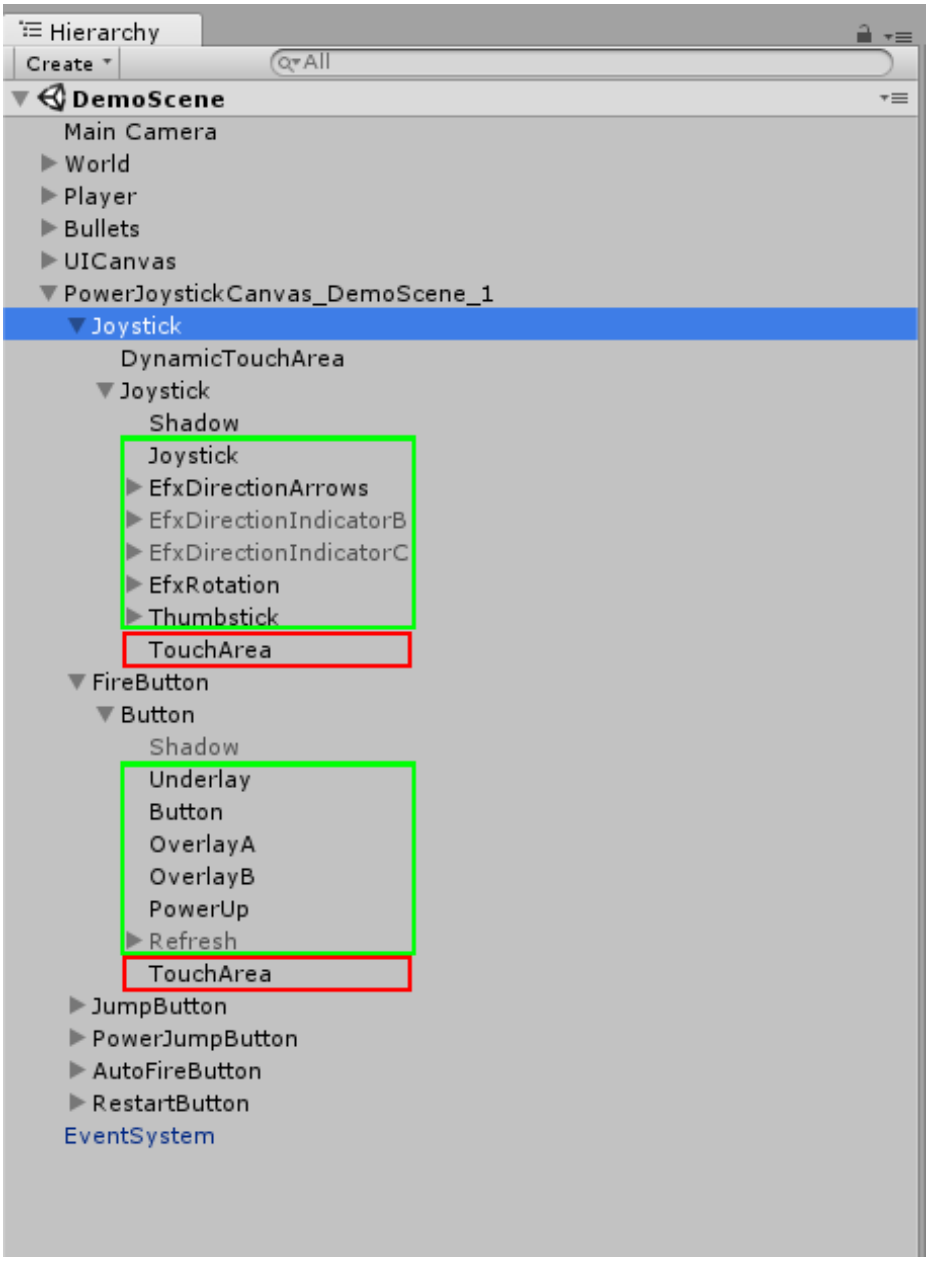
Invert the colors of a texture:



Good To Know

Layer Sorting

To achieve certain effects it can sometimes be useful to change the sorting of the layers of the joystick / d-pad / button. You can sort the game objects / layers of the joystick in the hierarchy window as you like. The TouchArea object should always be at the bottom of the hierarchy:



Folder Structure

The Power Joysticks package contains a lot of textures. In order to keep an overview of all prefabs and textures all prefab- and texture folders were meaningfully named. For example, if you want to edit a joystick prefab with the name PowerJoystick_A_01, you will find the corresponding textures in the folder Textures/Joysticks/A.

If you want to edit a controller prefab with the name PowerJoystickCanvas_GB_Controller, you will find the corresponding textures in the folder Textures/TLGF_Customized/GB.

If you want to edit a button prefab with the name PowerButton_Space_02, you will find the corresponding textures in the folder Textures/Space.

And so on...

Public Functions List

If you want to change a controller at runtime you can use the following setter and getter functions:

Joystick:

```
public void SetJoystickPositionLandscapeX (float pos)
public void SetJoystickPositionLandscapeY (float pos)
public void SetJoystickPositionPortraitX (float pos)
public void SetJoystickPositionPortraitY (float pos)
public void SetJoystickSizeLandscape (float scale)
public void SetJoystickSizePortrait (float scale)
public void SetJoystickLandscapeAlignment (JoystickLandscapeAlignment alignment)
public void SetJoystickPortraitAlignment (JoystickPortraitAlignment alignment)
public void SetJoystickMovementRange (float range)
public void SetJoystickDeadZone (float deadZone)
public void SetJoystickBoundary (Boundary b)
public void SetJoystickVisibilityMode (VisibilityMode vm)
public void SetJoystickPositionMode (PositionMode pm)
public void SetJoystickDynamicAreaLandscapePostionX (float pos)
public void SetJoystickDynamicAreaLandscapePostionY (float pos)
public void SetJoystickDynamicAreaPortraitPostionX (float pos)
public void SetJoystickDynamicAreaPortraitPostionY (float pos)
public void SetJoystickDynamicAreaLandscapeWidth (float width)
public void SetJoystickDynamicAreaLandscapeHeight (float height)
public void SetJoystickDynamicAreaPortraitWidth (float width)
public void SetJoystickDynamicAreaPortraitHeight (float height
public void SetJoystickSnapBackToStart (bool snapBack)
public void SetJoystickColor (Color c)
public void SetThumbstickColor (Color c)
public void SetJoystickFadeEfx (Color joystickStartColor, Color joystickEndColor, Color
thumbstickStartColor, Color thumbstickEndColor, float fadeTime)
public void SetJoystickFadeEfx (bool enable)
public void SetJoystickScaleEFX (float scaleFactor, float scaleTime)
public void SetJoystickScaleEfx (bool enable)
public void SetJoystickArrowsEfx (Sprite sprite, Color startColor, Color endColor, float fadeOutSpeed,
bool flash, float flashSpeed)
public void SetJoystickArrowsEfx (bool enable)
public void SetJoystickDirectionIndicatorBEfx (Sprite sprite, Color startColor, Color endColor, float
fadeOutSpeed, bool flash, float flashSpeed)
public void SetJoystickDirectionIndicatorBEfx (bool enable)
public void SetJoystickDirectionIndicatorCEfx (Sprite sprite, Color startColor, Color endColor, float
fadeOutSpeed, bool flash, float flashSpeed)
public void SetJoystickDirectionIndicatorCEfx (bool enable)
public void SetJoystickRoatationAEfx (Sprite sprite, Color startColor, Color endColor, float speed,
bool fadeOut, float fadeOutSpeed, bool flash, float flashSpeed)
public void SetJoystickRoatationAEfx (bool enable)
public void SetJoystickRoatationBEfx (Sprite sprite, Color startColor, Color endColor, float speed,
bool fadeOut, float fadeOutSpeed, bool flash, float flashSpeed)
public void SetJoystickRoatationBEfx (bool enable)
public void SetJoystickRoatationCEfx (Sprite sprite, Color startColor, Color endColor, float speed,
bool fadeOut, float fadeOutSpeed, bool flash, float flashSpeed)
public void SetJoystickRoatationCEfx (bool enable)
```

```
public void SetJoystickShadowEfx (Sprite joyShadowSprite, Color joyShadowColor, float joyShadowPosX,
float joyShadowPosY, Sprite thumbShadowSprite, Color thumbShadowColor, float thumbShadowPosX, float
thumbShadowPosY)
public void SetJoystickShadowEfx (bool enable)
public void SetTargetTapCount (int count)
public int GetTargetTapCount ()
public void SetTapCount (int count)
public int GetTapCount ()
```

D-Pad:

```
public void SetDPadPositionLandscapeX (float pos)
public void SetDPadPositionLandscapeY (float pos)
public void SetDPadPositionPortraitX (float pos)
public void SetDPadPositionPortraitY (float pos)
public void SetDPadSizeLandscape (float scale)
public void SetDPadSizePortrait (float scale)
public void SetDPadLandscapeAlignment (DPadLandscapeAlignment alignment)
public void SetDPadPortraitAlignment (DPadPortraitAlignment alignment)
public void SetDPadDeadZone (float deadZone)
public void SetDPadVisibilityMode (VisibilityMode vm)
public void SetDPadPositionMode (PositionMode pm)
public void SetDPadDynamicAreaLandscapePostionX (float pos)
public void SetDPadDynamicAreaLandscapePostionY (float pos)  public void SetDPadDynamicAreaPortraitPost
ionX (float pos)
public void SetDPadDynamicAreaPortraitPostionY (float pos)
public void SetDPadDynamicAreaLandscapeWidth (float width)
public void SetDPadDynamicAreaLandscapeHeight (float height)
public void SetDPadDynamicAreaPortraitWidth (float width)
public void SetDPadDynamicAreaPortraitHeight (float height)
public void SetDPadSnapBackToStart (bool snapBack)
public void SetDPadColor (Color c)
public void SetDPadFadeEfx (Color dPadStartColor, Color dPadEndColor, float fadeTime)
public void SetDPadFadeEfx (bool enable)
public void SetDPadScaleEFX (float scaleFactor, float scaleTime)
public void SetDPadScaleEfx (bool enable)
public void SetDPadArrowsEfx (Sprite sprite, Color startColor, Color endColor, float fadeOutSpeed, bool
 flash, float flashSpeed)
public void SetDPadArrowsEfx (bool enable)
public void SetDPadDirectionIndicatorBEfx (Sprite sprite, Color startColor, Color endColor, float fadeO
utSpeed, bool flash, float flashSpeed)
public void SetDPadDirectionIndicatorBEfx (bool enable)
public void SetDPadDirectionIndicatorCEfx (Sprite sprite, Color startColor, Color endColor, float fadeO
utSpeed, bool flash, float flashSpeed)
public void SetDPadDirectionIndicatorCEfx (bool enable)
public void SetDPadRoatationAEfx (Sprite sprite, Color startColor, Color endColor, float speed, bool fa
deOut, float fadeOutSpeed, bool flash, float flashSpeed)
public void SetDPadRoatationAEfx (bool enable)
public void SetDPadRoatationBEfx (Sprite sprite, Color startColor, Color endColor, float speed, bool fa
deOut, float fadeOutSpeed, bool flash, float flashSpeed)
public void SetDPadRoatationBEfx (bool enable)
public void SetDPadRoatationCEfx (Sprite sprite, Color startColor, Color endColor, float speed, bool fa
deOut, float fadeOutSpeed, bool flash, float flashSpeed
public void SetDPadRoatationCEfx (bool enable)
public void SetDPadShadowEfx (Sprite joyShadowSprite, Color joyShadowColor, float joyShadowPosX, float
joyShadowPosY)
public void SetDPadShadowEfx (bool enable)
public void SetTargetTapCount (int count)
public int GetTargetTapCount ()
public void SetTapCount (int count)
public int GetTapCount)
```

Button:

```
public void SetButtonPositionLandscapeX (float pos)
public void SetButtonPositionLandscapeY (float pos)
public void SetButtonPositionPortraitX (float pos)
public void SetButtonPositionPortraitY (float pos)
public void SetButtonSizeLandscape (float scale)
```

```
public void SetButtonSizePortrait (float scale)
public void SetButtonTouchAreaSizeLandscape (float scale)
public void SetButtonTouchAreaSizePortrait (float scale)
public void SetButtonLandscapeAlignment (ButtonLandscapeAlignment alignment
public void SetButtonPortraitAlignment (ButtonPortraitAlignment alignment)
public void SetButtonEnableAutoFire (bool enable, float delay)
public bool GetButtonAutoFire ()
public float GetButtonAutoFireDelay ()
public void SetButtonColor (Color c)
public void SetButtonFadeEfx (Color buttonStartColor, Color buttonEndColor, float fadeTime)
public void SetButtonFadeEfx (bool enable)
public void SetButtonScaleEFX (float scaleFactor, float scaleTime)
public void SetButtonScaleEfx (bool enable)
public void SetButtonUnderlayEfx (Sprite sprite, Color startColor, Color endColor, float fadeInSpeed,
float fadeOutSpeed)
public void SetButtonUnderlayEfx (bool enable)
public void SetButtonOverlayAEfx (Sprite sprite, Color startColor, Color endColor, float fadeInSpeed,
float fadeOutSpeed)
public void SetButtonOverlayAEfx (bool enable
public void SetButtonOverlayBEfx (Sprite sprite, Color startColor, Color endColor, float fadeInSpeed,
float fadeOutSpeed)
public void SetButtonOverlayBEfx (bool enable)
public void SetButtonPowerUpEfx (Sprite sprite, Color startColor, Color endColor, float delay, float
time, PowerUpOrigin origin, bool flash, float flashSpeed)
public void SetButtonPowerUpEfx (bool enable)
public void SetButtonRefreshEfx (Sprite sprite, Color startColor, Color endColor, Color filStartColor,
Color fillEndColor, float time, RefreshOrigin origin)
public void SetButtonRefreshEfx (bool enable)
public void SetButtonShadowEfx (Sprite shadowSprite, Color shadowColor, float shadowPosX, float
shadowPosY)
public void SetButtonShadowEfx (bool enable)
public void SetTargetTapCount (int count)
public int GetTargetTapCount ()
public void SetTapCount (int count)
public int GetTapCount ()
public void SetHoldAndReleaseTargetTime (float time)
public float GetHoldAndReleaseTargetTime ()
public void SetCurrentHoldAndReleaseTime (float time)
public float GetCurrentHoldAndReleaseTime ()
public void SetPowerUpTime (float time)
public float GetPowerUpTime ()
public void SetPowerUpDelay (float time)
public float GetPowerUpDealy ()
public void SetCurrentPowerUpTime (float time)
public float GetCurrentPowerUpTime ()
public void SetRefreshTime (float time)
public float GetRefreshTime ()
public void SetCurrentRefreshTime  (float time)
public float GetCurrentRefreshTime ()
```