# Getting Started with the Phaser Virtual Joystick Plugin

Version 1.0.0
March 31st 2015

By Richard Davey

## Phaser Virtual Joystick Classes

VirtualJoystick - Phaser.VirtualJoystick
Stick - Phaser.VirtualJoystick.Stick
DPad - Phaser.VirtualJoystick.DPad
Button - Phaser.VirtualJoystick.Button

## How to add the plugin to your game

This plugin has been extensively tested against Phaser 2.2.2 and 2.3.0. We do not guarantee it will work with earlier builds.

You will find the distribution files in the `dist` folder.

The file `phaser-virtual-joystick.js` is for use during production.

The file `phaser-virtual-joystick.min.js` is a minified version of the plugin for use when you go live.

Finally the file `phaser.2.3.1-virtual-joystick.js` is a complete version of Phaser 2.3.1 with the plugin ready installed. Please note this is the 'full' version of Phaser, including both P2 Physics and Arcade Physics.

There are three ways to include the plugin in your game:

### 1. Add it into your html file

You can include the plugin within the `<head>` block of your html page:

```
<script src="phaser.min.js" type="text/javascript"></script>
<script src="phaser-virtual-joystick.min.js" type="text/javascript"></script>
```

You may need to edit the paths depending on your environment.

### 2. Load it into your game via Phaser

Phaser can load script files automatically, so rather than modifying the html you could load the plugin from within your game code:

```
function preload() {

    game.load.script('joystick', 'phaser-virtual-joystick.min.js');

}
```

Again you will need to ensure the path is correct.

Once the file has been loaded it will be automatically added to the document and available for use within your game.

### 3. Merge the plugin into your build process

The final way to include the plugin is to ensure it's part of your build process. The plugin should be included after Phaser but before your game code. If you are using Grunt there is the handy `grunt-contrib-concat` package which will concat files together.

The following Gruntfile snippet will concat the plugin source files:

```
concat: {

    src: [
        'VirtualJoysticks/plugin/src/Pad.js',
        'VirtualJoysticks/plugin/src/Stick.js',
        'VirtualJoysticks/plugin/src/Button.js',
        'VirtualJoysticks/plugin/src/DPad.js',

        ... your game source files go here

    ],
    dest: 'your-game.js'

}
```

## How to enable the plugin

As with the other Phaser plugins you need to load it into the Plugin Manager:

```
var pad = this.game.plugins.add(Phaser.VirtualJoystick);
```

The object returned is a `Phaser.VirtualJoystick` object. Please see the API Docs for details about the methods and properties available.

## How to create new skins

In the `psd` folder you will find the Photoshop files used to create the skins provided with this plugin.

The process we went through is as follows:

### 1. Export each layer to a PNG

We exported each layer of the stick skin into a PNG file. You can find the results of these in the sub-folders 'Arcade', 'DPad' and 'Generic'. The names of the files is important:

For an analogue stick there should be a 'base.png' which represents the base frame and a 'stick.png', which is the part the player grabs hold of.

You can give the files other names, but if you do so then you must specify the frame names in the `addStick` method.

### 2. Add the buttons

If you are creating buttons as well (although not all games need them) then ensure you have both an 'up' and 'down' state for your buttons and export them as PNGs as well.

### 3. Create a texture atlas

We have provided the Texture Packer files we used in order to make the skin texture atlases. However you can use any atlas tool you like. The important thing is that it exports into a Phaser compatible format: either a JSON Array, a JSON Hash or a Starling XML file.

Once you've created your atlas it needs to be loaded into your game:

```
preload: function () {

    this.load.atlas('arcade', 'skins/joystick.png', 'skins/joystick.json');

}
```

The key given when loaded (in this case `arcade`) is the texture key you'll give when creating a new Stick or DPad object:

```
this.stick = this.pad.addStick(0, 0, 100, 'arcade');
```

### Design Tips

Although you can change the opacity of the stick via `Stick.alpha`, we suggest you try and perfect the opacity levels in Photoshop before exporting. Try your design over the top of a variety of grabs from your game to see what sort of levels and colors work best.

The skins provided are 256 x 256 pixels in size. This is so you can scale them down responsively and retain clarity. We suggest you design to this resolution as well.

Keep in mind that all of the math behind the plugin works on the principals of circles. It's not essential that the base frame be circular, but the stick nearly always should be, as the hit area needs to align with the image you create perfectly.

If you have a motion locked joystick then the base can be any shape you like - however the stick should still be circular.

## Check the Examples

We provided a whole bunch of examples with this plugin for you to learn from. They cover all of the core features, and the API Docs should fill in any blanks. The plugin is extremely flexible, and you've always got access to the underlying source code should you need to tweak it.

## Further Questions?

Please check the Examples provided. They will cover lots of use cases not covered here.

But you're also welcome to ask on the Phaser Forum. Just be sure to mention in your post you are using the Virtual Joystick plugin.

If you think you've found a bug in the plugin then please report it. Either via Github Issues or email support@phaser.io.

## License

The Phaser Virtual Joystick Plugin is (C) Copyright 2015 Photon Storm Limited and not for distribution. Please do not share it or pirate it, thank you.