Tutorial - Image Reorderer

This page will walk you through an example of using the Infusion Reorderer's reorderImages() function to reorder image thumbnails in a collection.

This tutorial assumes that:

- you are already familiar with HTML, Javascript and CSS
- you are familiar with what the Image Reorderer is and does
- now you just want to know how to add it to your file.

For more general information about the Reorderer, see Reorderer. For technical API documentation, see Image Reorderer API and Advanced Reorderer API.

Tutorial: How to Use the Image Reorderer

Scenario

Suppose you're not satisfied with any of the image sharing applications currently available on the web, and you're convinced that you can write a better one. You want to use Infusion's Image Reorderer to let your users re-arrange the images in their collections.

There are five basic steps to using the Image Reorderer in your application:

- Setup: Download and install the Fluid Infusion library
- Step 1: Prepare your markup
- Step 2: Write the script
- Step 3: Add the script to your HTML
- Step 4: Apply styles

The rest of this tutorial will explain each of these steps in detail.

Status

This component is in Production status

On This Page

See Also

- Image Reorderer
- Image Reorderer API
- Advanced Reorderer API
- Talking to the Server Using The afterMove Event

Still need help?

Join the infusion-users mailing list and ask your questions there.

Setup: Download and install the Fluid Infusion library

1. Download a copy of the Fluid Infusion component library from:

- http://fluidproject.org/products/fluid-infusion/download-infusion/
  You only really need the "Minified deployment package," but if you want to actually look at the code, you should download the "Source package."

2. Unpack the zip file you just downloaded, and place the resulting folder somewhere convenient for your development purposes.
   The folder will have the release number in its name (e.g. infusion-1.4/). The rest of this tutorial will use infusion-1.4 in its examples, but if you downloaded a different version, you'll have to adjust.

Step 1: Prepare your markup

Let's suppose you're using a `<form>` with hidden `<input>` elements to record the ordering of the images in your collections. (For a description of how to use this `<form>` approach, see Talking to the Server Using The afterMove Event.) A simple example of this could be:

```
<form action="#">
    <a href="myImage1.jpg">
        <img src="myImage1.jpg" alt="image 1 thumbnail" />
        <span>Image 1</span>
        <input name="image 1" value="0" type="hidden" />
    </a>
    <a href="myImage2.jpg">
        <img src="myImage2.jpg" alt="image 2 thumbnail" />
        <span>Image 2</span>
        <input name="image 2" value="1" type="hidden" />
    </a>
    <a href="myImage3.jpg">
        <img src="myImage3.jpg" alt="image 3 thumbnail" />
        <span>Image 3</span>
        <input name="image 3" value="2" type="hidden" />
    </a>
    ...
</form>
```

The Image Reorderer needs to know about the 'container' of your image collection. In this case, that would be the `<form>` element. But since there may be other `<form>` elements in the markup, you will need to uniquely identify the `<form>` you want to use as a container.

The Reorderer accepts a jQuery selector, so you can choose any method that will uniquely identify the `<form>` element. We'll attach a unique ID to it:

```
<form action="#" id="reorder-images-form">
    ...
```

Note

This example uses an ID, but you might, for example, use a CSS class, or the element hierarchy - whatever works, so long as it uniquely identifies the right element.

Form elements and the Image Reorderer

Currently the Image Reorderer is limited to using `<form>` elements as its root container. This is a known issue and it is expected that in the future any container can be used as long as it is properly identified. For more information see (FLUID-4019).

You also need to tell the Reorderer which of your images should be reorderable and most of the time, that will likely be all of them. But perhaps you want the first image to *always* be first since it's the cover of the album - with the Image Reorderer this is possible. For this tutorial, though, we'll make all contained images movable.

You'll tell the Reorderer which items are to be orderable with another jQuery selector. The Image Reorderer understands a default class name for this purpose, but you can override that if you like. For this tutorial, we'll stick with the defaults classname `flc-imageReorderer-item`. Let's add that to each of the `<a>` elements:

```
<form action="#" id="reorder-images-form">
    <a href="myImage1.jpg" class="flc-imageReorderer-item">
        <img src="myImage1.jpg" alt="image 1 thumbnail" />
        <span>Image 1</span>
        <input name="image 1" value="0" type="hidden" />
    </a>
    <a href="myImage2.jpg" class="flc-imageReorderer-item">
        <img src="myImage2.jpg" alt="image 2 thumbnail" />
        <span>Image 2</span>
        <input name="image 2" value="1" type="hidden" />
    </a>
    <a href="myImage3.jpg" class="flc-imageReorderer-item">
        <img src="myImage3.jpg" alt="image 3 thumbnail" />
        <span>Image 3</span>
        <input name="image 3" value="2" type="hidden" />
    </a>
    ...
</form>
```

Note

As with the ID on the `<form>`, we can use any jQuery selector for the reorderable images. For example, we could attach a unique ID to each movable `<a>` with a unique prefix, maybe `pic-movable1`, `pic-movable2`, etc. Then we could use the jQuery selector `[fluid:id^=pic-movable]` to override the default selector.

Finally, you'll want to tell the Image Reorderer which part of your markup is the caption for the image. The Image Reorderer will use this information to help make your image collection more usable by people using assistive technologies, such as a screen reader.

You can identify the captions using a custom classname, or use the default selector classname `flc-reorderer-imageTitle`:

```
<form action="#" id="reorder-images-form">
    <a href="myImage1.jpg" class="flc-imageReorderer-item">
        <img src="myImage1.jpg" alt="image 1 thumbnail" />
        <span class="flc-reorderer-imageTitle">Image 1</span>
        <input name="image 1" value="0" type="hidden" />
    </a>
    <a href="myImage2.jpg" class="flc-imageReorderer-item">
        <img src="myImage2.jpg" alt="image 2 thumbnail" />
        <span class="flc-reorderer-imageTitle">Image 2</span>
        <input name="image 2" value="1" type="hidden" />
    </a>
    <a href="myImage3.jpg" class="flc-imageReorderer-item">
        <img src="myImage3.jpg" alt="image 3 thumbnail" />
        <span class="flc-reorderer-imageTitle">Image 3</span>
        <input name="image 3" value="2" type="hidden" />
    </a>
    ...
</form>
```

Step 2: Write the script

To make the HTML you just created do something special, you'll need to create a file to contain your initialization script - the script you write to apply the Reorderer to your image collection.

Create a file, say `image-collection.js`, and in this file, write a function that looks like this:

```
jQuery(document).ready(function () {
    return fluid.reorderImages("#reorder-images-form");
});
```

In this function call, the parameter to `reorderImages()`, `"#reorder-images-form"`, is a jQuery selector identifying the element with the ID `reorder-images-form`. That's all the information required by the `fluid.reorderList()` function.

By enclosing the function call inside `jQuery(document).ready()`, we ensure that the HTML is fully rendered before we apply the Reorderer to it.

Note

If you choose to use a custom selector for the movable items (instead of the default classname), you can override the default using options passed as the second parameter. Define an options block that specifies the selector you'd like, and pass it to the function:

```
jQuery(document).ready(function () {
    var opts = {
        selectors: {
            movables: "[id^=pic-movable]"
        }
    };
    return fluid.reorderImages("#reorder-images-form", opts);
});
```

For more information on selectors and other options, see the Image Reorderer API documentation.

Step 3: Add the script to your HTML

You'll need to add your initialization script, along with the Infusion library, to you HTML file. In the header of the file, link to the Javascript files with `<script>` tags:

```
<script type="text/javascript" src="infusion-1.2/InfusionAll.js"></script>
<script type="text/javascript" src="image-collection.js"></script>
```

Keep in mind that the `InfusionAll.js` file is minified - all of the whitespace has been removed - so it isn't really human-readable. If you're using the source distribution and you want to be able to debug the code, you'll want to include each of the required files individually. This would look like this:

```
<script type="text/javascript" src="../../../../lib/jquery/core/js/jquery.js"></script>
<script type="text/javascript" src="../../../../lib/jquery/ui/js/jquery.ui.core.js"></script>
<script type="text/javascript" src="../../../../lib/jquery/ui/js/jquery.ui.widget.js"></script>
<script type="text/javascript" src="../../../../lib/jquery/ui/js/jquery.ui.mouse.js"></script>
<script type="text/javascript" src="../../../../lib/jquery/ui/js/jquery.ui.draggable.js"></script>
<script type="text/javascript" src="../../../../framework/core/js/FluidDocument.js"></script>
<script type="text/javascript" src="../../../../framework/core/js/jquery.keyboard-a11y.js"></script>
<script type="text/javascript" src="../../../../framework/core/js/Fluid.js"></script>
<script type="text/javascript" src="../../../../framework/core/js/FluidDOMUtilities.js"></script>
<script type="text/javascript" src="../../../../framework/core/js/FluidView.js"></script>
<script type="text/javascript" src="../../../../framework/core/js/DataBinding.js"></script>
<script type="text/javascript" src="../../../../framework/core/js/FluidIoC.js"></script>
<script type="text/javascript" src="../../../../components/reorderer/js/ReordererDOMUtilities.js"></script>
<script type="text/javascript" src="../../../../components/reorderer/js/GeometricManager.js"></script>
<script type="text/javascript" src="../../../../components/reorderer/js/Reorderer.js"></script>
<script type="text/javascript" src="../../../../components/reorderer/js/ImageReorderer.js"></script>
<script type="text/javascript" src="image-collection.js"></script>
```

But all of these individual files are not necessary to make it work - the `InfusionAll.js` file has everything you need.

That's it! That's all you need to do to make your images reorderable!

Step 4: Apply styles

You can style your image gallery any way you choose (of course), or use the default Infusion Image Reorderer style.

Using the default styles

You can take advantage of the Image Reorderer styles provided with the component by simply adding the default styling class names to your markup. The Image Reorderer will take care of the rest.

There are three things you'll want to add styling classnames to:

1. the reorderer container element, using `fl-imageReorderer` and `fl-reorderer-horizontalLayout`,
2. the reorderable elements themselves, using `fl-imageReorderer-item`, and
3. the captions, using `fl-imageReorderer-caption`.

```
<form action="#" id="reorder-images-form" class="fl-imageReordererfl-reorderer-horizontalLayout">
    <a href="myImage1.jpg" class="flc-imageReorderer-item fl-imageReorderer-item">
        <img src="myImage1.jpg" alt="image 1 thumbnail" />
        <span class="flc-reorderer-imageTitle fl-imageReorderer-caption">Image 1</span>
        <input name="image 1" value="0" type="hidden" />
    </a>
    <a href="myImage2.jpg" class="flc-imageReorderer-item fl-imageReorderer-item">
        <img src="myImage2.jpg" alt="image 2 thumbnail" />
        <span class="flc-reorderer-imageTitle fl-imageReorderer-caption">Image 2</span>
        <input name="image 2" value="1" type="hidden" />
    </a>
    <a href="myImage3.jpg" class="flc-imageReorderer-item fl-imageReorderer-item">
        <img src="myImage3.jpg" alt="image 3 thumbnail" />
        <span class="flc-reorderer-imageTitle fl-imageReorderer-caption">Image 3</span>
        <input name="image 3" value="2" type="hidden" />
    </a>
    ...
</form>
```

The `fl-reorderer-horizontalLayout` will lay the images out horizontally, and will make sure that the drop marker shows up between the thumbnails properly.

Customizing the styles

If you choose to use CSS classname different than the defaults, you can override the defaults using the `options` parameter to the `reorderImages()` function.

Specify your classnames using the {[styles]} option, and the required style names:

```
var opts = {
    styles: {
        defaultStyle: "myMovableImage",
        imageTitle: "myImageCaption"
    },
};
return fluid.reorderImages("#reorder-images-form", opts);
```

There are actually many styles used by the Reorderer, affecting how the thumbnails look when the cursor hovers over them, what the avatar looks like while it's being dragged, what the drop marker looks like, and more. For a complete list of styles, see the full technical documentation: Image Reorderer API
