gethifi.com

A jQuery Flickr Feed Plugin

View the Demo | Download the Zip

We often work with clients that maintain accounts with Twitter, Flickr, Youtube and other services in addition to their website. Often they will want to pull in data from one of their accounts to their website. With Flickr, this is pretty easy because they make a simple API available. Having worked with it a few times, we decided to make it even easier to pull photos from a public feed.

There are a few examples of this in use on the demo page. All of the photos on this page have been used with the generous permission of John Roberts (@jroberts13).

Plugin Overview

This plugin works by pulling a JSON feed from Flickr and applying the data it gets back to a template. For example, we can generate this list of pictures:

From the following jQuery:

```
4          id: '44802888@N04'

6     itemTemplate: '<li><img alt="{{title}}" src="{{image_s}}"
/></li>'
```

The plugin gets the feed from Flickr using AJAX and applies each image it gets back to the provided template.

The Plugin Options

There are a number of options available on the plugin, these are the defaults:

```
02   feedapi: 'photos_public.gne',

09    cleanDescription: true,

12   itemCallback: function(){}
```

Here is a description of each one:

- **flickrbase:** This is unlikely to be needed. It is used to set up the url the jQuery AJAX call will need to be made to.
- **feedapi:** There are a number of feeds that flickr makes available. The default is the public feed. Here is the list of all that are available: http://www.flickr.com/services/feeds/. This has only been tested on feeds that return photos. So, for example, don't expect good results using this plugin on the Forum discussion feeds.
- **limit:** Set how many items you want to loop through. Flickr seems to limit its feeds to 20, so that is the default.
- **qstrings**: *This is the most important setting*. This is used to request the correct feed. In my examples I use this to set the user id. These are automatically added to the request url. Depending on which feed you use (http://www.flickr.com/services/feeds/) there are different sets of query parameters available: http://www.flickr.com/services/feeds/.
- **cleanDescription:** Flickr puts all kinds of junk in the description it returns. By default this plugin is set to remove everything but the plain photo description.
- **useTemplate:** Set this to false if you don't want to use the plugins templating system.
- **itemTemplate:** The template rules are described below.

- **itemCallback:** You can add a callback on each item. The scope is set to the container and the item object is made available.

Typically, the only things that will need to be set are limit, qstrings.id and itemTemplate.

Using the Templates

In order to make it really easy to use any kind of markup needed with this plugin, a simple templating system has been build in. Here is an example of a template:

```
2 <a href="{{image_b}}"><img src="{{image_s}}" alt="
{{title}}" /></a>
```

You can see that this is just basic html with a few special tags mixed in. All of the tags are surrouned by double curly braces. The plugin works by putting in the correct information for each tag and each item into the template.

The tags that are available depend on what flickr returns for each item. For the Public Photos API these are: title, link, date_taken, description, published, author, author_id, tags, and image. For the image property, the plugin uses the Flickr URL scheme to make several sizes available. You can read about this at http://www.flickr.com/services/api/misc.urls.html. The image tags available are: image_s, image_t, image_m, image, image_b.

The Callback Parameter and Integration

The plugin's second parameter is a callback. This has the scope of the containing element and has the entire data response available. This is a great feature if you want to integrate this plugin with others. Let's say you want to integrate it with colorbox. If you call $('selector').colorbox() and then this plugin it won't work. This is becuase the plugin is adding elements to the page after the colorbox call. This is even true if you call colorbox after this plugin. Since this uses ajax, your images won't be added to the page until well after most of your javascript has run. (In computer time).

Instead, the trick is to use the callback function. This allows you to pass in code that you want to run after the images have loaded. So for example, you can do this:

```
01    $('#cbox').jflickrfeed({

04            id: '37304598@N02'

06       itemTemplate: '...example...'

08        $('#cbox a').colorbox();
```

Now the colorbox call won't be made on the photos until they are loaded.

Demo and Download

You can see a demo of this plugin on the demo page. Additionally you can download a zip of this plugin and the example.

As always, if you have any questions or comments, leave them below. Enjoy!