# Flickr Image SlideShow

## In BackBone.js and JQuery

**Demo | GitHub**

**Project By: Jennifer Soh & James Griffith**

**5/9/2013**

ABSTRACT: This project is for IS217. The project features a slideshow that shows photos parsed from Flickr's API. The framework of the program is built behind Backbone.js.

# Table of Contents

## First prototype

At first, we were focused on what the slideshow should look like. Getting over our heads we had planned for more than we could do. After watching a few videos on YouTube and Lynda, James managed to get the first prototype working (see figure 1)

We presented this prototype to the class, and received mixed vibes. The problem with this prototype was the fact that James originally used a plug-in called Roundabout.js which essentially made the program very easy to code. The plug-in did most of the work while James just called the variables.

Later, we came to the conclusion that plug-ins were not the way to go.

## Second prototype

A few weeks later, Jen managed to find a good resource on how to build the second prototype. This slideshow was built from straight jquery. It used no plug-ins and it was simple to understand.

A milestone was reached when the images parsed on the screen and moved accordingly. We presented this to the class receiving less of a warm welcome than last time. After this presentation, we had a meeting deciding on what to do with the API.

Jen began to research the API and acquire an API key and secret.

## The API

'http://api.flickr.com/services/rest/?method=flickr.photos.search&api_key=ee78f78cac9824c29b1837f737726a3b&text=' + search + '&per_page=15&page=1&format=json&nojsoncallback=1'

Parsing data from Flickr was the hardest part for us to do. We had to move a lot of stuff and make a lot of variables that later on wasn't even necessary.

After about a week, Jen was able to make the API work. We presented this to the class as our pre-final presentation. The class liked this much better.

## Backbone.js

It was now James' time to buckle down and get to work on the backbone framework. Backbone is a library that gives structure to web apps. There are three main pieces to the backbone library. Models, Collections and Views.

For our project the breakdown was as follows:

**Model** -> Function
**Collection** -> Function
**View** -> Function -> API
JQuery Animation

**FlickrPhotoModel**

- Defining the whole thing

**FlickrPhotoCollection**

- Collecting all of the photos in the payload and holding them

**FlickrPhotoView**

-Load all those photos into the webpage and show them to the user

When the pseudo code was complete, it was time to set out the raw code of the script.

First we start with the model

*var FlickrPhoto = Backbone.Model.extend ({*

The model is what we are trying to parse. So in theory, a FlickrPhoto is a Flickr Photo.

Secondly, You need a collection for it all

*var FlickrPhotos = Backbone.Collection.extend({*

*model: FlickrPhoto*

Then, You need somewhere to SHOW YOU STUFF. This is the view

*var FlickrPhotoView = Backbone.View.extend ({*

Everything from the API to the animation goes into the view. If this was something bigger like a store or a to do list, we would have a temple as well with multiple views. For the store, you could have a nav_view, a content_view, and a manager_view.

Think of views like Drupal Nodes!

Finally, The API is stitched into the application with JSON. JSON is a file with data from an API. In this case, the JSON file is holding data from images. Titles, descriptions, tags, authors, media, and date_taken are all attributes of this file.

The JSON file is initiated with a $.getJSON request. The images are parsed with ID numbers instead of your regular .jpg file. You then have to manipulate the file name into what would be a .jpg file. In this demo, it is showing you all of the _m.jpg files from Flickr. Due to our novice knowledge of programming, it wasn't possible to learn how to parse larger photos so we had to distort them a bit using CSS.

The program starts off with no images. We declare this as "MinImages=0" The "MaxImages" variable is declared and set to 100 to cover up to 100 folders. You can virtually set this to anything.

The jquery is then brought into the equation while still working in the view piece of backbone. The jquery is used to make a fading effect on the image. The image is a controlled  tag that is exactly the height and width of the image so there isn't any overflow. Each time an image disappears. The jquery will perform what is called the .hide function. This function can be customized to any transition within the library. We chose fade because we were going for a TV slideshow kind of vibe.

After the first image loads, it needs to keep going. This was performed by writing an if statement. The IF statement states that if the image state is at 0, you need to add 1. So at 1 you get 2, at 3 you get 4 etc.

## The HTML

The JavaScript needs somewhere to dump its calculations so index.html is its home. Inside of this file, you have a div named "FlickrRotator". This div holds the images that parse after being pulled from Flickr.

---

## Extra Stuff

With such a limited knowledge of Backbone and JS, we couldn't really dig too deep into what we thought we could do in the time we had.

We wanted to add controls for stopping, playing and fast forwarding but it was too much for us to handle. If we had more time, it would have gotten done. Recently, we had altered the search API to accept search terms that should be taken when the document executes. However, the slideshow executes before the search term can be accepted, which wasn't resolved.

## Screen Shots

```
var FlickrPhoto = Backbone.Model.extend ({
        PhotoModel: function(){
        console.log('after the model loads');
    }
});
var FlickrPhotos = Backbone.Collection.extend({
    model: FlickrPhoto,
    FlickrPhotoCollection: function(){
        console.log('after the collection loads');
    }
});
var FlickrPhotoView = Backbone.View.extend ({
    // init view
    GrabPhoto: function(){
        console.log('GrabPhoto Is Sucessful');
    },
```

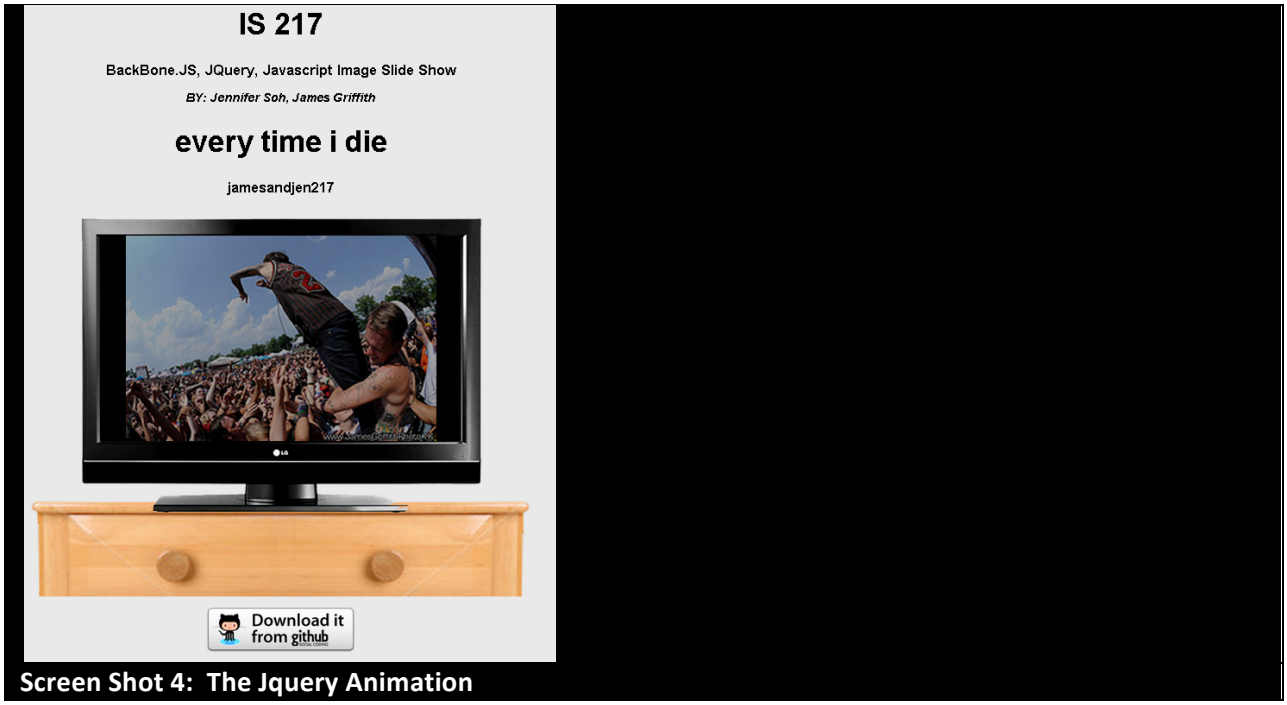
**ScreenShot 1: The Backbone Structure**

```
$.getJSON ("http://api.flickr.com/services/feeds/photos_public.gne?&tags=jamesandjen2l7&lang=en-us&format=json&jso
.done(function (data){

    $.each ( data.items, function (i, item) { $("#title").html(item.title);
                $("#description").html(item.tags);
                $("<img>").attr( { id: 'Api'+i, src: item.media.m } ).appendTo("#FlickrPhotoRotator");
        if (i === 10) {
        return false;
};
```

**ScreenShot 2: The API Broken Down**

```
        setInterval(function (){ $("#Api"+MinImages).hide("fade", {}), 1000;

            if (MinImages == MaxImages){
                    MinImages = 0;
                }else{
                MinImages = MinImages + 1;
                }
            }, 2000);
    },
```

**ScreenShot 3: The Jquery Animation**




**Screen Shot 4: The Jquery Animation**

*In ScreeShot 4, You will see that it says "Everytime I dIe" This is the name of the photo being parsed. Under the <h1> tag, you will see an <h2> with jamesandjen217. This is showing you the only tag that you are viewing.*

## Overall Experience

Overall, this project was a lot of work. There were a lot of all nighters and visits to the PC Mall after class but even after all over that we managed to get it done. At first we didn't really work well together and our schedules were bad, but after a talk with the professor, we managed to work everything out.

This entire project was a challenge for both of us. We learned a lot for it and hope to continue to climb the latter from here.

## What We've Learned

Before coming into this class, neither of us knew anything about JavaScript. The we learned about libraries, Node, MongoDB, Express and all other kinds of cool and technical things.

We learned overall, that becoming a successful programmer takes time and hardwork.

## Participation

- **Documentation # 1 – Jen and James**
- **Documentation #2 – Jen**
- **Prototype #1 – James**
- **Prototype #2 – Jen**
- **Initial API Configuration – Jen**
- **Prototype #3 – Jen**
- **Backbone.js Framework – James**
- **Backbone.js API Installation – James and Jen**
- **Final documentation – James and Jen**
- **Screen Shots – James**

## Credits

**BackBone.js – Framework Library**

**Jquery – Library for animation**

**Flickr API – API used for photos**

**Stackoverflow – For all of the research we did on it**

**GetHiFi – Flickr API Help** http://www.gethifi.com/blog/a-jquery-flickr-feed-plugin

**Viget** – Flickr Help http://viget.com/inspire/pulling-your-flickr-feed-with-jquery

## The Demo

http://web.njit.edu/~jag54/is217project/

## The GitHub

https://github.com/jamesgriffith/IS217-Final