JQUERY TRICKSHOTS

100 **ADVANCED** TECHNIQUES

A BOOK BY TUTORIALZINE

IV. AJAX

AJAX is a fundamental building block for web apps. It allows you to send only the data that you need, saving bandwidth and speeding things up, making your sites feel native-like. In this chapter I will show you a number of tricks that you can use to enhance your applications and I'll explain a few of the new things that recent jQuery versions introduced.

34. Display file sizes next to download links

Did you know that you can send a HEAD request with AJAX and get the size of a file without downloading it? With jQuery this is very easy:

HTML

```
<a href="001.html" class="fetchSize">First Trickshot</a>
<a href="034.html" class="fetchSize">This Trickshot</a>
<a href="ball.png" class="fetchSize">Ball.png</a>
```

JS

```
// Loop all .fetchSize links
$('a.fetchSize').each(function(){

    // Issue an AJAX HEAD request for each one
    var link = this;

    $.ajax({
        type: 'HEAD',
        url: link.href,
        complete: function(xhr){

          var size = humanize(xhr.getResponseHeader('Content-Length'));

          // Append the filesize to each
         $(link).append(' (' + size + ')');

        }
    });

});


function humanize(size){
    var units = ['bytes','KB','MB','GB','TB','PB'];

    var ord = Math.floor( Math.log(size) / Math.log(1024) );
    ord = Math.min( Math.max(0,ord), units.length-1);

    var s = Math.round((size / Math.pow(1024,ord))*100)/100;
    return s + ' ' + units[ord];
}
```

This snippet places the size of the file in braces next to its name. The script issues a HEAD request, which only returns the headers and not the actual content of the file, which means that these requests are fast and lightweight.

```
First Trickshot (871 bytes)
This Trickshot (1.27 KB)
Ball.png (12.49 KB)
```

35. Simplify your Ajax calls with deferreds

Deferreds are a powerful tool. jQuery returns a new deferred object for every AJAX request, which makes them easier to work with. Here is how you can use deferreds to make your code more readable:

JS

```
// This is equivalent to passing a callback as the
// second argument (executed on success):

$.get('1.json').done(function(r){
    console.log(r.message);
});

// Requesting a file that does not exist. This will trigger
// the failure response. To handle it, you would normally have to
// use the full $.ajax method and pass it as a failure callback,
// but with deferreds you can can simply use the fail method:

$.get('non-existing.json').fail(function(r){
    console.log('Oops! The file is missing!');
});
```

As you will see in tip #55, deferreds have a lot of power behind them.

36. Run multiple AJAX requests in parallel

When working with APIs, you sometimes need to issue multiple AJAX requests to different endpoints. Instead of waiting for one request to complete before issuing the next, you can speed things up with jQuery by requesting the data in parallel, by using jQuery's **$.when()** function:

JS

```
$.when($.get('1.json'), $.get('2.json')).then(function(r1, r2){
    console.log(r1[0].message + " " + r2[0].message);
});
```

The callback function is executed when both of these GET requests finish successfully. **$.when()** takes the promises returned by two $.get() calls, and constructs a new promise object. The r1 and r2 arguments of the callback are arrays, whose first elements contain the server responses.

37. Get your IP with jQuery

Did you know you can get your public IP address with one line of JS? There is a free service that offers this for you and a get request is all that you need to do:

```
$.get('http://jsonip.com/', function(r){ console.log(r.ip); });
```

For the above snippet to work, your browser will have to support CORS (cross-origin request sharing). Otherwise a security exception would be thrown. In older browsers, you can use this version, which uses a JSON-P request:

```
$.getJSON('http://jsonip.com/?callback=?', function(r){ console.log(r.ip); });
```

38. The simplest AJAX request

jQuery offers a shorthand method for quickly loading content into an element via AJAX – the **.load()** method.

HTML

```html
<p class="content"></p>
<p class="content"></p>
```

JS

```js
var contentDivs = $('.content');

// Fetch the contents of a text file:
contentDivs.eq(0).load('1.txt');

// Fetch the contents of a HTML file, and display a specific element:
contentDivs.eq(1).load('1.html #header');
```

Unlike the rest of the AJAX methods, here you can specify a CSS selector that will limit the result. Also the data isn't returned, but directly replaces the content of the element.

39. Serializing objects

If you are serious about AJAX, you should be familiar with jQuery's methods for encoding forms and plain objects. The result is a URL-friendly representation that you can include in your AJAX request. Here is how to use them:

HTML

```html
<form id="gform">
  <input type="text" name="performer" value="Gangnam K. Style" />
  <input type="text" name="email" value="psy@w3c.org" />
  <textarea name="lyrics">Na je nun ta sa ro un in gan jo gin yo ja</textarea>
</form>
```

JS

```js
var form = $('#gform');

// Turn all form fields into a URL friendly key/value string.
// This can be passed as argument of AJAX requests, or URLs.
console.log(form.serialize());

console.log('---------');
```

```
// You can also encode your own objects with the $.param method:
console.log($.param({'pet':'cat', 'name':'snowbell'}));

console.log('---------');

// With $.serializeArray() you can encode the form to an object
console.log(form.serializeArray());
```

Result

```
performer=Gangnam+K.+Style&email=psy
%40w3c.org&lyrics=Na+je+nun+ta+sa+ro+un+in+gan+jo+gin+yo+ja
---------
pet=cat&name=snowbell
---------
[{"name":"performer","value":"Gangnam K. Style"},
{"name":"email","value":"psy@w3c.org"},{"name":"lyrics","value":"Na je nun ta sa ro un
in gan jo gin yo ja"}]
```

A lot of the times these conversions are made behind the scenes, when you pass an object as the argument to one of the AJAX functions.

40. File uploads with jQuery

Modern browsers support the FormData API, which, among other things, allows you to send binary data easily through AJAX. Combine this with the HTML5 File API, and you get an easy way for uploading files without the need for additional plugins.

The idea is to obtain a reference to a file through a file input field. Then we will create a new FormData object and append the file reference. Lastly, we will pass the FormData object directly to jQuery's AJAX method. For the last step to work though, we will have to set two of its properties to false to prevent jQuery from processing the data.

HTML

```html
<input type="file" />
<button id="upload">Upload it!</button>
```

JS

```javascript
var fileInput = $('input[type=file]'),
    button = $('#upload');

button.on('click', function(){

    // Access the files property, which holds
    // an array with the selected files

    var files = fileInput.prop('files');

    // No file was chosen!
    if(files.length == 0) {
          alert('Please choose a file to upload!');
          return false;
    }

    // Create a new FormData object
    var fd = new FormData();

    fd.append('file', files[0]);

    // Upload the file to assets/php/upload.php, which only prints the filename

    $.ajax({
          url: './assets/php/upload.php',
          data: fd,
          contentType:false,
          processData:false,
          type:'POST',
          success: function(m){
                console.log(m);
          }
    });
});
```

Accessing the files property of the file input field will give us an array with file objects from the HTML5 File API. We can add them to the FormData object, and assign them as the data property of the AJAX request. The tricky part is setting ***contentType*** and ***processData*** to false, so that jQuery doesn't attempt to serialize the data (which would only break the request) and leave it to the browser. No file upload plugins needed!

41. Work with Facebook's Graph

The Graph API is a very powerful interface to Facebook's huge pool of social data. There still are some parts of it that are publicly available. The API is available under the graph.facebook.com subdomain. Here is an example with Tutorialzine's FB page:

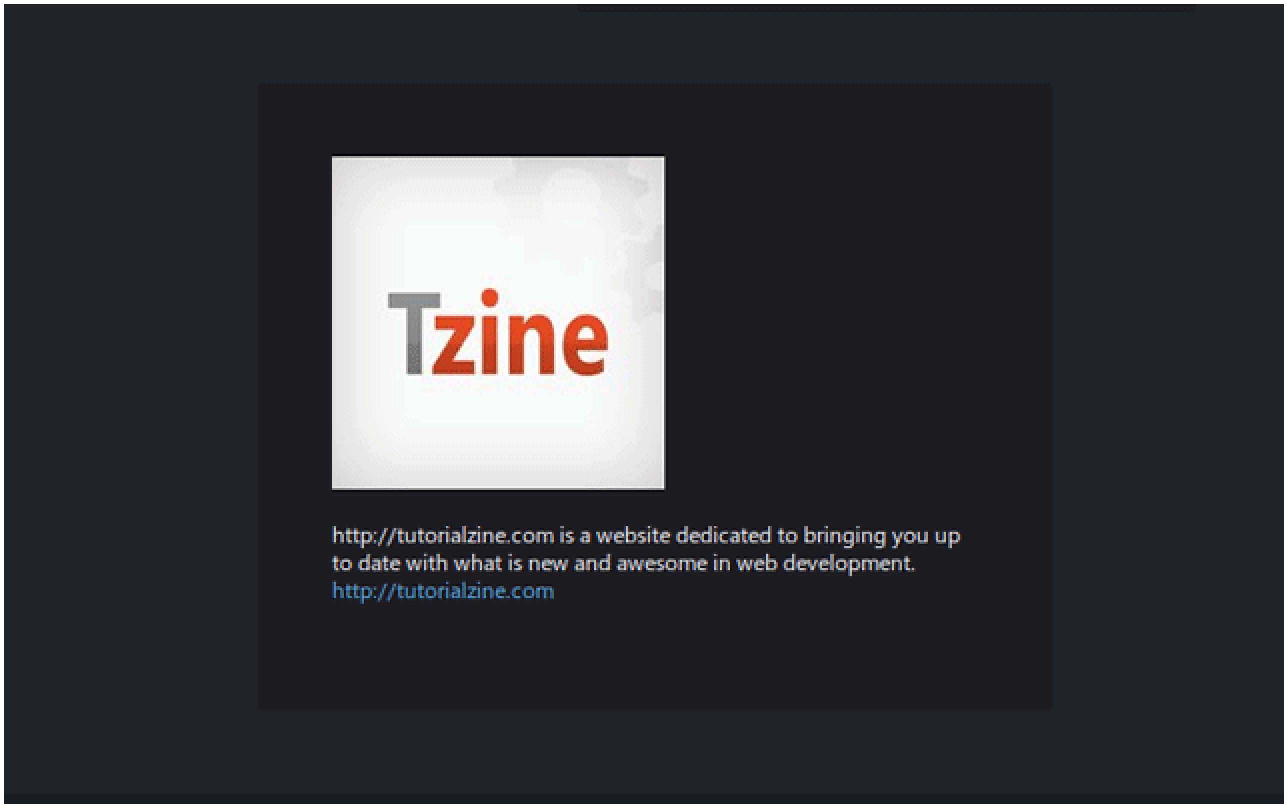
HTML

```
<div id="fbdata"></div>
```

JS

```
// Fetch the publicly accessible data on Tutorialzine's Page
var api = 'http://graph.facebook.com/Tutorialzine/?callback=?',
    holder = $('#fbdata');


$.getJSON(api, function(r){

    // This will always give the current picture
    holder.append('<img src="http://graph.facebook.com/Tutorialzine/picture/?
type=large">');

    holder.append('<p>'+ r.about +'</p>')
    holder.append('<a href="'+ r.website +'">'+ r.website +'</a>');

});
```

This prints the description of the page, link and the profile photo.


http://tutorialzine.com is a website dedicated to bringing you up to date with what is new and awesome in web development.
http://tutorialzine.com

42. Access weather information

Weather data is another place where a good API is crucial. Open Weather Map provides free weather information that you can use through their JSON API. Here is an example:

```
// Request weather data:
var api = 'http://openweathermap.org/data/2.1/find/name?q=paris,france&callback=?';


$.getJSON(api, function(r){

    // This will always give the current picture
    console.log(r.list[0].name + ', ' + r.list[0].sys.country);
    console.log(r.list[0].main);

    // Temperatures are in kelvin, subtract 273.15 to convert to Celsius,
    // or use the formula (kelvin*⁹/₅ - 459.67) for Fahrenheit
});
```

Result

```
Paris, FR
{"temp":277.02,"pressure":1020,"humidity":80,"temp_min":276.15,"temp_max":277.59}
```

You get real time data. Notice that the temperature is in Kelvin; you will need to manually convert it to a format for presenting to users.

43. Fetch your latest Tumblr posts

In the spirit of the great APIs of old, the popular blogging service Tumblr offers an easy way for you to fetch the posts of any blog as JSON. Here is how their API is used:

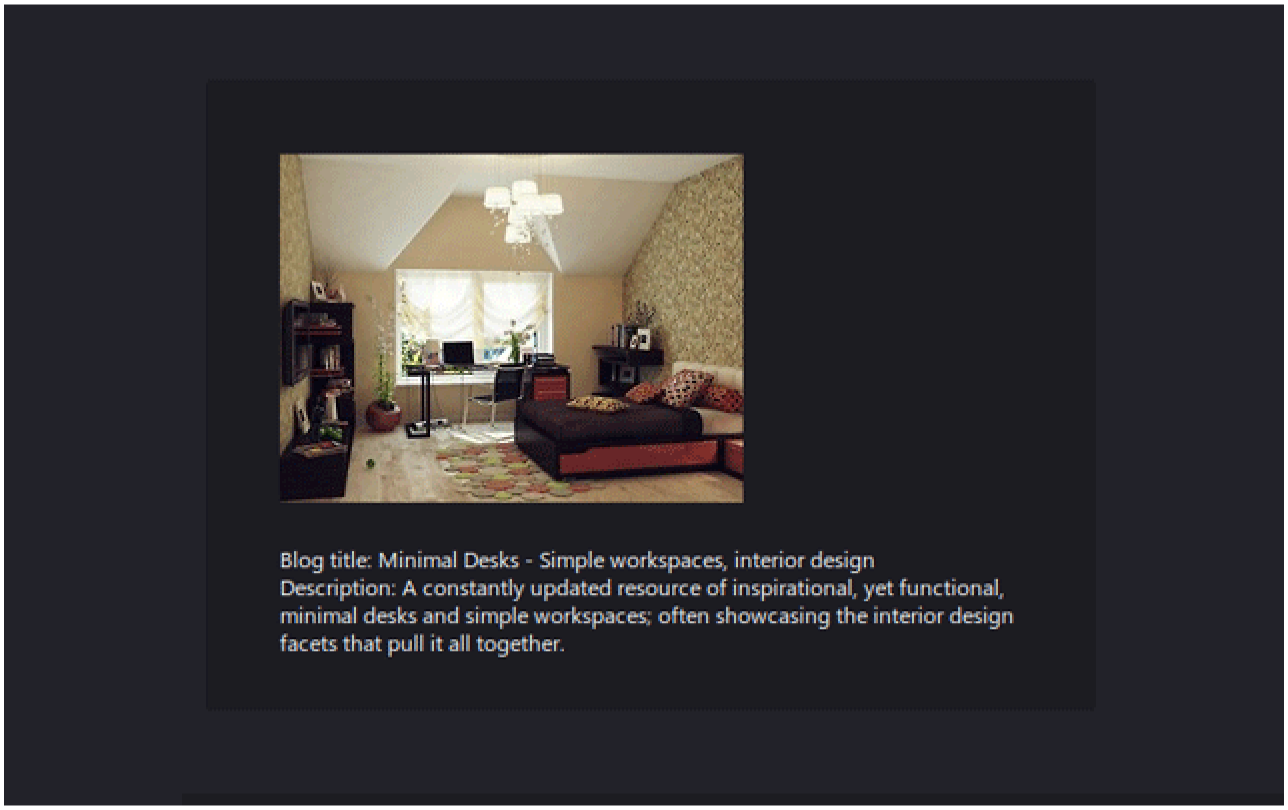
HTML

```
<div id="post"></div>
```

JS

```
// Define some variables
var blog = 'minimaldesks.tumblr.com',
     api  = 'http://' + blog + '/api/read/json?callback=?',
     post = $('#post');


$.getJSON(api, function(r){

     console.log('Blog title: ' + r.tumblelog.title);
     console.log('Description: ' + r.tumblelog.description);

     // If this post has a photo, show it
     if(r.posts[0]['photo-url-250']){
           post.append('<img src="' + r.posts[0]['photo-url-250'] + '" />');
     }
     else{
           console.log('Latest post: ' + r.posts[0]['regular-title']);
     }

});
```

This gives you details about the blog and the last post with a photo. It even works with custom domains – simply replace the tumblr URL with the domain.


Blog title: Minimal Desks - Simple workspaces, interior design
Description: A constantly updated resource of inspirational, yet functional, minimal desks and simple workspaces; often showcasing the interior design facets that pull it all together.

44. Find the geographic location of an IP address

There are services online that can tell you the city and country of an IP address. Here is how to use one of them – freegeoip.net:

```
// Define some variables
var ip = '', // you can optionally put an ip address here
    api  = 'http://freegeoip.net/json/' + ip + '?callback=?';


$.getJSON(api, function(r){

    console.log('How is the weather in ' + r.city + ', ' + r.country_name + '?');

});
```

If no IP address is provided, the API will assume the address of the client making the request.

45. Scrape sites with YQL

YQL is the ultimate API for the JavaScript developer. It makes it possible to work with all kinds of third party APIs through a SQL-like interface. Here is how to use it to fetch and parse HTML from remote sites:

```
// Define variables

var query = 'select * from data.html.cssselect where
url="http://www.chucknorrisfacts.com/chuck-norris-top-50-facts" and css=".field-
content a"';
var yqlAPI = 'http://query.yahooapis.com/v1/public/yql?q=' + encodeURIComponent(query)
+ ' &format=json&env=store%3A%2F%2Fdatatables.org%2Falltableswithkeys&callback=?';


$.getJSON(yqlAPI, function(r){

    console.log('Chuck Norris Facts:');

    $.each(r.query.results.results.a, function(){
            console.log('----------');
            console.log(this.content);
    });

});
```

The results speak for themselves:

```
Chuck Norris Facts:
----------
Chuck Norris has the right to keep and arm bears.
----------
Chuck Norris can turn the lights off by clapping his eyelids twice.
----------
When Alexander Bell invented the telephone he had 3 missed calls from Chuck Norris
----------
Fear of spiders is aracnaphobia, fear of tight spaces is chlaustraphobia, fear of
Chuck Norris is called Logic
----------
Chuck Norris doesn't call the wrong number. You answer the wrong phone.
```

46. Use the global AJAX methods

You can simplify how your web app handles AJAX requests (and save yourself from writing some code), by utilizing the ajax global methods:

```
// Create an indicator that would be shown whenever an AJAX request occurs:

var preloader = $('<div>',{ 'class':'preloader' }).appendTo('body');
var doc = $(document);

// Show the preloader whenever you are making an AJAX request:

doc.ajaxStart(function(){
     preloader.fadeIn();
});

// Hide it when the ajax request finishes

doc.ajaxComplete(function(){
     // Keep it visible for 0.8 seconds after the request completes
     preloader.delay(800).fadeOut();
});

// It will show automatically every time you do an AJAX request:

$.get('1.json');
```

These methods are called every time that one of jQuery's AJAX methods is used, which makes them perfect for showing and hiding indicators.

Thank you for enjoying our free preview!

For more, purchase the full book from our website.