

== JQUERY ==

NUTS, BOLTS

AND
BLING



DOUG NEINER



doug@dougneiner.com



dougneiner



dcneiner



Doug Neiner

MY FAMILY

Crystal
Ruby, Olivia, Cody

Not pictured:
Ditto the cat



.appendTo
THE COMPANY DEDICATED TO jQUERY



I am a senior
designer here

I am a team
member here



 **jQuery**
write less, do more.

AUDIENCE

This presentation was crafted specifically for delivery at the Front End Design Conference in St. Petersburg, FL. The audience consisted of designers who have never used jQuery through developers who use jQuery regularly.

The slide portion of this presentation has a lot of getting started information, as well as some tricks you may or may not know. The live coding portion (available on Github) contains some more advanced techniques.

If you have any questions after watching this presentation and looking at the code, please let me know: doug@dougneiner.com (*Be sure to reference this presentation when emailing!*)

== JQUERY ==

NUTS, BOLTS

AND
BLING

WRITE WORKING CODE

Ugly, working code always trumps pretty, broken code

BASICS

JQUERY NUTS, BOLTS AND **BLING**

WRITING THE NAME

A. JQuery

C. jquery

B. JQuery

D. jQuery

* It's ok to write it in all caps when the rest of the context is all caps as well.

ADDING SCRIPTS

```
<div id="dan"> I am Dan </div>
```

```
<style>
```

```
#dan { display: none; }
```

```
</style>
```

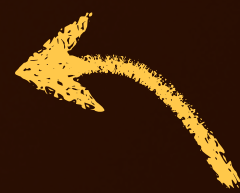
```
<div id="cherrie"> I am Cherrie </div>
```

```
<style>
```

```
#cherrie { background: red; }
```

```
</style>
```

HTML

You obviously
wouldn't do this

ADDING SCRIPTS

```
<div id="dan"> I am Dan </div>
```

```
<script>
```

```
$( "#dan" ).hide();
```

```
</script>
```

```
<div id="cherrie"> I am Cherrie </div>
```

```
<script>
```

```
$( "#cherrie" ).hide().fadeIn( 500 );
```

```
</script>
```

HTML

So please don't
do this

ADDING SCRIPTS

```
<div id="dan"> I am Dan </div>
<div id="cherrie"> I am Cherrie </div>
```

HTML

```
<script src="//ajax.googleapis.com/ajax/
libs/jquery/1.6.2/jquery.min.js"></script>
<script src="js/site.js"></script>
</body>
```

Better, but now
#dan and #cherrie
blink on for a split-
second before hiding

```
$( "#dan" ).hide();
$( "#cherrie" ).hide().fadeIn( 500 );
```

JS

ADDING SCRIPTS

```
<html class="no-js">
```

```
<head>
```

```
...
```

```
<script>
```

```
    document.documentElement.className =
```

```
        document.documentElement.className.replace( "no-js", "js" );
```

```
</script>
```

```
<link rel="stylesheet" href="css/layout.css" />
```

Now the two elements are hidden by CSS,
but only when we know JS is working



If you are using Modernizr, it
already does this, and you
don't need this code

HTML

```
.js #dan, .js #cherrie { display: none; }
```

CSS

ADDING SCRIPTS

```
<div id="dan"> I am Dan </div>  
<div id="cherrie"> I am Cherrie </div>
```

HTML

```
<script src="//ajax.googleapis.com/ajax/  
libs/jquery/1.6.2/jquery.min.js"></script>  
<script src="js/site.js"></script>  
</body>
```

Now we can remove
the two .hide()
statements and just
run fadeIn on #cherrie

```
$( "#cherrie" ).fadeIn( 500 );
```

JS

ADDING SCRIPTS

- Use at least one external script file for your site, not tons of in-page script blocks spread throughout your app.
- Include references to jQuery and your script just before the closing body tag
 - Primarily on web sites (*on JS-required apps inside <head> may be ok*)
 - Use no-js and js classes to provide styling until the script loads

FILE LAYOUT

```
jQuery( document ).ready( function () {  
    jQuery( "div" ).each( function () {  
        jQuery( this ).hide();  
    });  
    jQuery.getJSON( "...", function ( data ) {  
        jQuery.each( data, function () { ... });  
    });  
});
```

JS



Too... Much...
jQuery...

FILE LAYOUT

```
(function ( $ ) {  
  var sample = { sample: true };  
  // DOM may not be complete  
  
  $( document ).ready( function () {  
    // DOM is ready. Come and get it!  
  });  
  
}( jQuery ));
```

JS

FILE LAYOUT

```
(function ( $ ) {  
  var sample = { sample: true };  
  // DOM may not be complete  
  
  $( document ).ready( function () {  
    // DOM is ready. Come and get it!  
  });  
  
}( jQuery ));
```

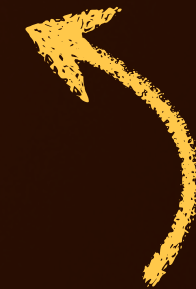
JS

This is called an
"Immediately
Invoking Function
Expression" or IIFE.

FILE LAYOUT (ALT)

```
jQuery( document ).ready( function ( $ ) {  
    // DOM is ready. Come and get it!  
});
```

JS



You can supply a parameter for the anonymous function callback, and it will reference jQuery.

use this as a shortcut when all your code must run on document.ready.

CONCEPTS

JQUERY NUTS, BOLTS AND **BLING**

ITERATION

// Explicit, you realize this is looping over items

```
$( "div" ).each(function () { ... });
```

// Implicit, you may not realize each call is looping

```
$( "div" )  
  .attr( "data-group", "doug" )  
  .addClass( "dougsGroup" )  
  .hide();
```

JS

CSS VS. CLASS

```
$( "div" ).css({  
  width: 20,  
  height: 500  
});
```

// Or:

```
$( "div" ).addClass( "tall" );
```

Use .css() when the values must be dynamic

Use classes when you know the values ahead of time.

```
.tall { width: 20px; height: 500px; }
```

TOGGLE METHODS

```
var div = $( "div" );  
if ( div.hasClass( "frontendrocks" ) {  
    div.addClass( "frontendrocks" );  
} else {  
    div.removeClass( "frontendrocks" );  
}
```

```
// A better way to write it  
div.toggleClass( "frontendrocks" );
```

JS

TOGGLE METHODS

// If you need it to match up to a variable

```
var something = true, div = $( "div" );
```

// This forces the class on or off based on `something`

```
div.toggleClass( "frontendrocks", something );
```

// Other methods support this too, check the API

```
div.slideToggle( 200, something );
```

```
div.toggle( something );
```

JS

UPDATING VALUES

```
var div = $( "div" );  
// Setting a single key/value  
div.attr( "key", "value" );  
  
// Setting more than one key/value at once  
div.attr( { "key": "value", "key2": "value2" } );  
  
// Setting the value using a callback function  
div.attr( "key" , function (i, oldValue ) {  
    return "newvalue";  
} );
```

UPDATING VALUES

```
// Other methods support it too, check the API
div.addClass( function (i, val) {

    // This returns an incremental class to add for each
    // item in the result set
    return "awesome-" + i;

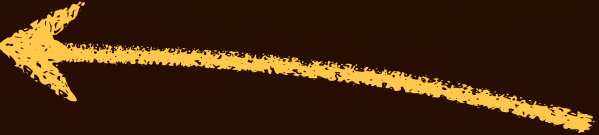
});
```

ASYNCHRONOUS CODE

```
var loaded = false;
```

```
$.get( "http://url.com", function ( data ) {  
    loaded = true;  
});
```

This method will run at
"some point in the future"



```
if ( loaded === true ) {  
    // Never runs  
}
```

This runs right away, and
will execute before the
callback for \$.get fires.

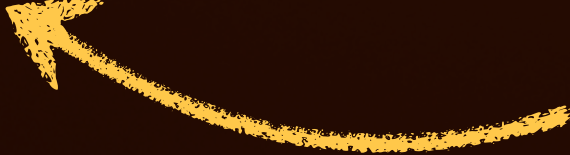


ASYNCHRONOUS CODE

```
var loaded = false;

$.get( "http://url.com", function ( data ) {
    loaded = data.loaded;
    if ( loaded === true ) {
        // This runs when loaded is true
    }
});
```

Put logic that depends on asynchronous code either in the callback or in a function you execute from the callback



LETS CODE!

Code is available here:

<https://github.com/dcneiner/jquery-bling>

JQUERY NUTS, BOLTS AND **BLING**

LINKS

- jQuery Tmpl
<https://github.com/jquery/jquery-tmpl>
- jQuery MockJax
<http://code.appendto.com/plugins/jquery-mockjax>
- jQuery doTimeout
<http://benalman.com/projects/jquery-dotimeout-plugin/>
- Alternate jQuery API
<http://jqapi.com/>



DOUG NEINER



doug@dougneiner.com



dougneiner



dcneiner



Doug Neiner