THE jQuery ESSENTIALS

IN 60 MINS OR LESS

WITH ADDY OSMANI @addyosmani

DEVOXX, BELGIUM, 2011.

```
$('knowledge').appendTo('you');
```

What is jQuery? Why should I use it?

Teach me how to use the API.

Introduction

What is jQuery?

It’s basically JavaScript

made more accessible.

Well..

With JavaScript, cross-browser Ajax looks like:

```
function sendRequest(url,callback,postData) {
    var req = createXMLHTTPObject();
    if (!req) return;
    var method = (postData) ? "POST" : "GET";
    req.open(method,url,true);
    req.setRequestHeader('User-Agent','XMLHTTP/1.0');
    if (postData)
        req.setRequestHeader('Content-type','application/x-www-form-urlencoded');
    req.onreadystatechange = function () {
        if (req.readyState != 4) return;
        if (req.status != 200 && req.status != 304) {
            return;
        }
        callback(req);
    }
    if (req.readyState == 4) return;
    req.send(postData);
}
```

```
var XMLHttpFactories = [
    function () {return new XMLHttpRequest()},
    function () {return new ActiveXObject("Msxml2.XMLHTTP")},
    function () {return new ActiveXObject("Msxml3.XMLHTTP")},
    function () {return new ActiveXObject("Microsoft.XMLHTTP")}
];

function createXMLHTTPObject() {
    var xmlhttp = false;
    for (var i=0;i<XMLHttpFactories.length;i++) {
        try {
            xmlhttp = XMLHttpFactories[i]();
        }
        catch (e) {
            continue;
        }
        break;
    }
    return xmlhttp;
}
```

Well..

With JavaScript, cross-browser Ajax looks like:

```
function sendRequest(url,callback,postData) {
    var req = createXMLHTTPObject();
    if (!req) return;
    var method = (postData) ? "POST" : "GET";
    req.open(method,url,true);
    req.setRequestHeader('User-Agent','XMLHTTP/1.0');
    if (postData)
        req.setRequestHeader('Content-type','application/x-www-form-urlencoded');
    req.onreadystatechange = function () {
        if (req.readyState != 4) return;
        if (req.status != 200 && req.status != 304) {
            return;
        }
        callback(req);
    }
    if (req.readyState == 4) return;
    req.send(postData);
}
```

```
var XMLHttpFactories = [
    function () {return new XMLHttpRequest()},
    function () {return new ActiveXObject("Msxml2.XMLHTTP")},
    function () {return new ActiveXObject("Msxml3.XMLHTTP")},
    function () {return new ActiveXObject("Microsoft.XMLHTTP")}
];

function createXMLHTTPObject() {
    var xmlhttp = false;
    for (var i=0;i<XMLHttpFactories.length;i++) {
        try {
            xmlhttp = XMLHttpFactories[i]();
        }
        catch (e) {
            continue;
        }
        break;
    }
    return xmlhttp;
}
```

Whoa!

Well..

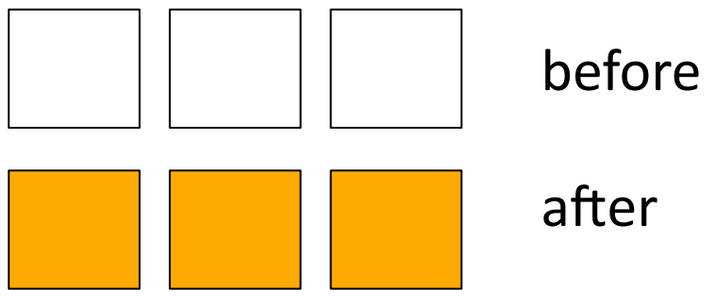
But with jQuery, it’s as simple as this:

```
// Get the latest attendees
$.get("/attendees.jsp",
  data: { range: 'latest'},
  function( html) {
    $("#results").append( html );
  }
});
```

Well..

With JavaScript, you would style elements like this..

```
var elems = document.getElementsByClassName('attendee'),
    len  = elems.length,
    i=0;

for(; i<len; i++) {
  elems[i].style.backgroundColor = 'orange';
}
```

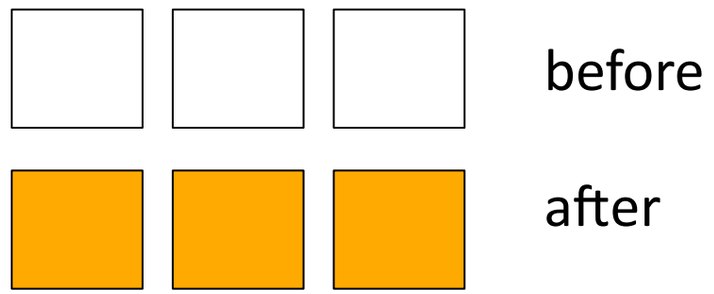


Well..

With JavaScript, you would style elements like this..

```
var elems = document.getElementsByClassName('attendee'),
    len  = elems.length,
    i=0;

for(; i<len; i++) {
  elems[i].style.backgroundColor = 'orange';
}
```

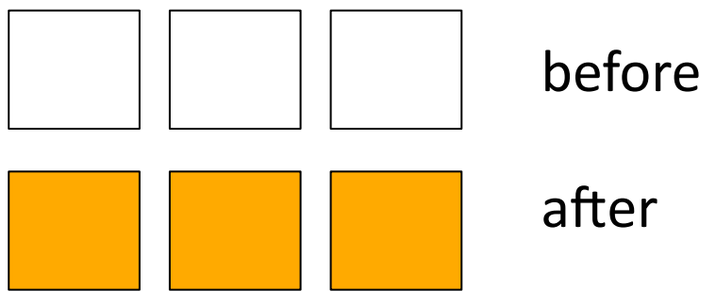
Hold on!



Well..

With jQuery, it's just a simple one-liner:

```
$('.attendee').css('backgroundColor','orange');
```



It’s lightweight (31kb)

Extensively tested

*QUnit unit tests exist for the entire API

Let’s you write less

and do more

“What does that mean?”

jQuery allows you to easily select elements on a page and manipulate or add some new behaviour to them.

jQuery..

- Simplifies **traversing** the DOM
- Powerful **selection** engine
- Eases element **styling** through JavaScript
- Powerful methods for element **animation**
- Cross-browser **Ajax** interactions
- DOM **data-storage**
- and more!

jQuery..

Normalises many **cross-browser quirks** so don't have to worry about them

Helps minimize issues with this bastard:

Hi! I'm IE6

jQuery..

- It's **open-source**
- Great for **beginners** wishing to 'break' into front-end web development
- Developers from any other language background can start using it **easily**

“Why should I use it?”

Reasons like this..

twitter Search Home Profile Messages Who To Follow addy_osmani

@oscsparky
Mark Evans

Dear @jquery once again you have saved the day and make what I thought was going to be a nightmare a complete pleasure #thanks

28 Oct via TweetDeck ☆ Favorite ↻ Retweet ↩ Reply

Mentioned in this Tweet

jquery jQuery · Unfollow
Write less, do more. Official news and updates from the jQuery team.

About Help Blog Status Jobs Terms Privacy Advertisers Businesses Media Developers Resources © 2011 Twitter

and maybe this..

twitter Search Home Profile Messages Who To Follow addy_osmani

@johndavidback
John David Back

Thank you, John Resig, for $.wrap(); You are the greatest wrapper I know. #jquery

31 Oct via YoruFukurou ☆ Favorite ↻ Retweet ↩ Reply

About Help Blog Status Jobs Terms Privacy Advertisers Businesses Media Developers Resources © 2011 Twitter

But mostly..

- jQuery is extensively documented
- Large online community here to help
  - Forums
  - Blogs
  - Tutorials
  - Twitter
  - IRC (#jquery on freenode)
  - Conferences
  - Books

Who uses jQuery?

IBM

Aol.

Google

amazon.com

twitter

BBC

ESPN

DELL

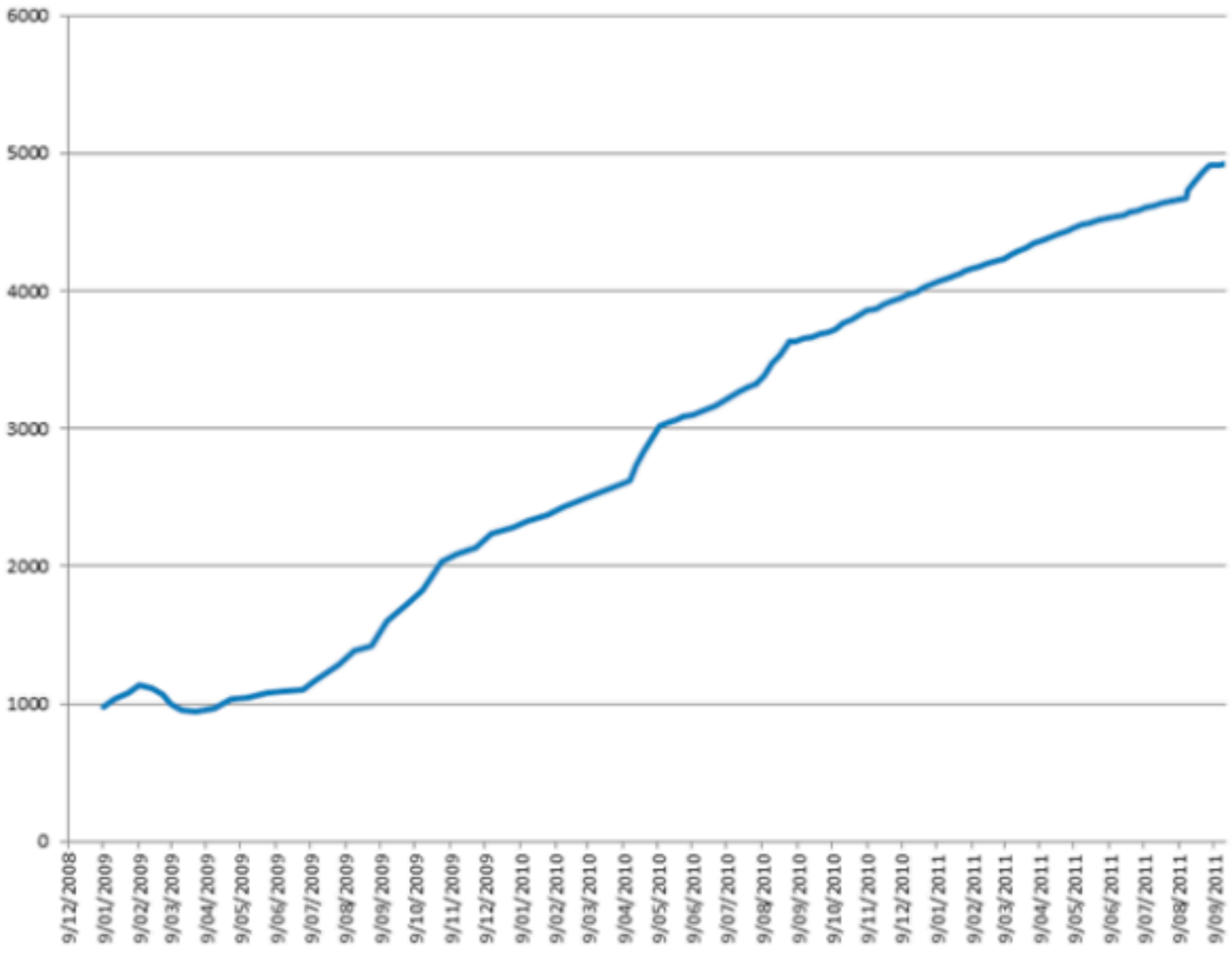
ELECTRONIC ARTS™

Microsoft®

Usage: Top 10K Sites

Trends

BuiltWith.com has been compiling weekly trends of website technology usage since 2008. The following chart shows jQuery usage in the top 10k sites on the Internet -



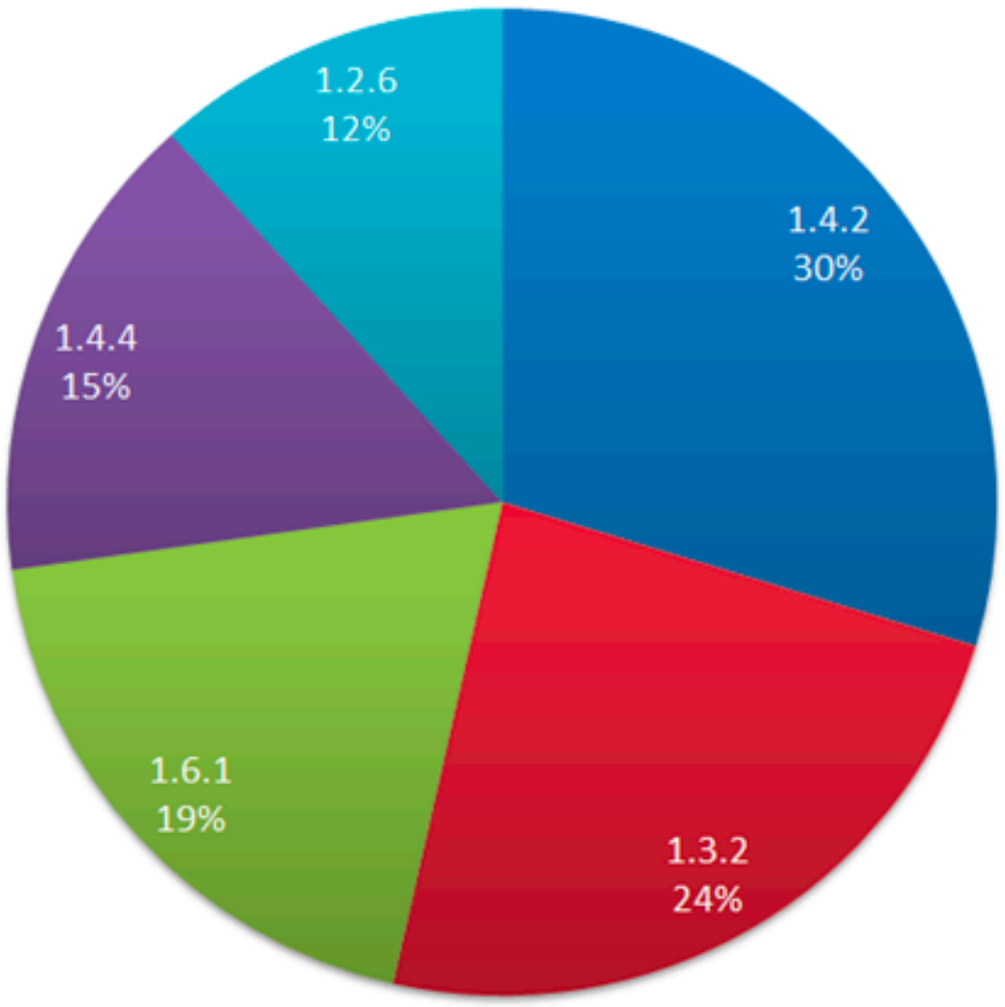
jQuery Usage in the Top 10k Sites since December 2008

Over 50% Using jQuery

Usage: Top 1 Million Sites

jQuery Version Usage

In the top million sites as provided by Quantcast, 311,654 sites report using jQuery, 60% of those use 5 different versions of the jQuery library.



Top 5 Versions of jQuery in use in the Top Million Sites

Over 60% Using jQuery

Where Are We Now?

Project status

Some history

- jQuery was first released in January, 2006.
- Initially developed by John Resig

Some history

- We've gone through several versions since then
  - 1.2.*
  - 1.3.*
  - 1.4.*
  - 1.5.*
  - 1.6.*
- Project now supported by:
  - 30+ Open-source contributors
  - Multiple sub-teams

Status

- We've just released **jQuery 1.7.1**

jQuery Plugins UI Meetups Forum Blog About Donate

jQuery
write less, do more.

Download Documentation Tutorials Bug Tracker Discussion

BLOG

Search jQuery

PAST ENTRIES

- Plugins Site Update: The Old Is New Again
- What Is Happening To The jQuery Plugins Site?
- jQuery Conference 2012: UK - Training Workshops Announced
- Call for jQuery 1.8 Ideas
- jQuery 1.7.1 Released
- Getting Board of jQuery
- jQuery 1.7.1 RC1 Released
- Upcoming jQuery Events
- Building a Slimmer jQuery

BLOG » JQUERY 1.7.1 RELEASED

Posted November 21st, 2011 by dmethvin

Here in the United States, we're celebrating Thanksgiving this week. For those of you living elsewhere in the world, it's a time when we install and test new versions of Javascript libraries while feasting on Mom's homemade goodies. Kind of like a code sprint, but with better food. We invite everyone worldwide to join us in these traditions.

To kick off the festivities, the jQuery Team is quite thankful to be releasing version 1.7.1! In this go-round we made Pilgrim's progress on a cornucopia of bugs, listed below. We are serving up our delicious copies on the jQuery CDN, fresh and warm from the oven:

- http://code.jquery.com/jquery-1.7.1.min.js
- http://code.jquery.com/jquery-1.7.1.js

These latest files should also be up on the major CDNs shortly, but please be patient since this is a holiday week for them as well.

Status

- Lots of new **changes**:
  - $.Callbacks
  - Better Events API
    - .on()
    - .off()
  - Performance optimisations
- A little of the API has been deprecated
- Looking to clean-up ‘cruft’ by 1.8

Status

- This is one of the **first** presentations to incorporate these changes
- Useful to keep in mind if reading jQuery books as some of these are **already** out of date
- You'll get an **up-to-date** run-through today!

Getting setup

Let’s get jQuery on your page

‘How do I start using this thing?’

Setup

- Head over to jQuery.com:

jQuery Plugins UI Meetups Forum Blog About Donate

jQuery write less, do more.

Download Documentation Tutorials Bug Tracker Discussion

jQuery is a new kind of JavaScript Library.

jQuery is a fast and concise JavaScript Library that simplifies HTML document traversing, event handling, animating, and Ajax interactions for rapid web development. jQuery is designed to change the way that you write JavaScript.

✔ Lightweight Footprint ✔ CSS3 Compliant ✔ Cross-browser

GRAB THE LATEST VERSION!

CHOOSE YOUR COMPRESSION LEVEL:

PRODUCTION *(31KB, Minified and Gzipped)*
DEVELOPMENT *(229KB, Uncompressed Code)*

Download(jQuery);

Current Release: v1.7

Click download

WHO'S USING JQUERY? Google DELL digg NBC CBS NETFLIX Technorati mozilla.org

LEARN JQUERY NOW!

What does jQuery code look like? Here's the quick and dirty:

```
$("p.neat").addClass("ohmy").show("slow");
```

RUN CODE

JQUERY RESOURCES

Getting Started With jQuery

- How jQuery Works
- Tutorials
- Using jQuery with other libraries
- jQuery Documentation

Developer Resources

- Mailing List
- Source code / Git
- Plugin Authoring
- Submit a New Bug Report

Setup

- Create a new HTML file
- Include jQuery using a <script> tag

```
<!DOCTYPE html>
<html>
<head>
<script src="jquery-1.7.min.js"></script>
<title>jQuery test</title>
</head>
<body>
</body>
</html>
```

Alternatively

- Load it from the Google CDN
- This can be faster than self-hosting

```
<!DOCTYPE html>
<html>
<head>
<script src="http://code.jquery.com/
jquery-1.7.min.js"></script>
<title>jQuery test</title>
</head>
<body>
</body>
</html>
```

Using jQuery

- In most cases, your code should run when the document has finished loading

```
<script type="text/javascript">
$(document).ready(function(){
    // your code should go here
});

// alternatively
$(function(){
    // your code should go here
});
</script>
```

What is $?

- `$` is an alias to `jQuery`
- Code can either use $ or just jQuery

```
$(document).ready(function(){
    //your code should go here
});

// same as..

jQuery(document).ready(function(){
    //your code should go here
});
```

jQuery Structure

- Core
- CSS
- Selectors
- Ajax
- Attributes
- Effects
- Properties
- Deferred Object
- Dimensions
- Events
- Manipulation
- Plugins
- Utilities
- Data

‘I want to select an element on a page...then do something with it’

Basic Selectors

Simple selectors for querying elements from the DOM

Code


```
//Basic selectors

// ID
$('#conference')

// class
$('.attendee')

// element
$('div')

//wildcard
$('*')

//attribute
$('input[name="attendeeName"]')
```

```
<div class="welcome"></div>

<ul id="conference">

    <li class="attendee">
    <a title="addy osmani"
     href="http://addyosmani.com">Addy</a>
    </li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Adam Sontag</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="text" name="attendeeName"
value="John"/>
```

Code


```
//Basic selectors

// ID
$('#conference')

// class
$('.attendee')

// element
$('div')

//wildcard
$('*')

//attribute
$('input[name="attendeeName"]')
```

```
<div class="welcome"></div>

<ul id="conference">

    <li class="attendee">
    <a title="addy osmani"
     href="http://addyosmani.com">Addy</a>
    </li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Adam Sontag</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="text" name="attendeeName"
value="John"/>
```

Code


```
//Basic selectors

// ID
$('#conference')

// class
$('.attendee')

// element
$('div')

//wildcard
$('*')

//attribute
$('input[name="attendeeName"]')
```

```
<div class="welcome"></div>

<ul id="conference">

    <li class="attendee">
    <a title="addy osmani"
     href="http://addyosmani.com">Addy</a>
    </li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Adam Sontag</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="text" name="attendeeName"
value="John"/>
```

Code


```
//Basic selectors

// ID
$('#conference')

// class
$('.attendee')

// element
$('div')

//wildcard
$('*')

//attribute
$('input[name="attendeeName"]')
```

```
<div class="welcome"></div>

<ul id="conference">

    <li class="attendee">
    <a title="addy osmani"
     href="http://addyosmani.com">Addy</a>
    </li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Adam Sontag</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="text" name="attendeeName"
value="John"/>
```

Code


```
//Basic selectors

// ID
$('#conference')

// class
$('.attendee')

// element
$('div')

//wildcard
$('*')

//attribute
$('input[name="attendeeName"]')
```

```
<div class="welcome"></div>

<ul id="conference">

    <li class="attendee">
    <a title="addy osmani"
     href="http://addyosmani.com">Addy</a>
    </li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Adam Sontag</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="text" name="attendeeName"
value="John"/>
```

Code


```
//Basic selectors

// ID
$('#conference')

// class
$('.attendee')

// element
$('div')

//wildcard
$('*')

//attribute
$('input[name="attendeeName"]')
```

```
<div class="welcome"></div>

<ul id="conference">

    <li class="attendee">
    <a title="addy osmani"
     href="http://addyosmani.com">Addy</a>
    </li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Adam Sontag</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="text" name="attendeeName"
value="John"/>
```

‘What if I want to select an element that’s a child of another element?...’

Filter Selectors

Selectors for filtering down jQuery collections further

Code


```
// Filter selectors

//first element in a result set
$('ul li:first')

//first child of the parent
$('ul li:first-child')

//last element in a result set
$('ul li:last')

//last child of the parent
$('ul li:last-child')

//all odd elements in a result set
$('ul li:odd')

//all even elements in a result set
$('ul li:even')
```

```
<div class="welcome"></div>

<ul id="conference">

    <li class="attendee">
    <a title="addy osmani"
     href="http://addyosmani.com">Addy</a>
    </li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Adam Sontag</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="text" name="attendeeName"
value="John"/>
```

Code


```
// Filter selectors

//first element in a result set
$('ul li:first')

//first child of the parent
$('ul li:first-child')

//last element in a result set
$('ul li:last')

//last child of the parent
$('ul li:last-child')

//all odd elements in a result set
$('ul li:odd')

//all even elements in a result set
$('ul li:even')
```

```
<div class="welcome"></div>

<ul id="conference">

    <li class="attendee">
    <a title="addy osmani"
     href="http://addyosmani.com">Addy</a>
    </li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Adam Sontag</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="text" name="attendeeName"
value="John"/>
```

Code


```
// Filter selectors

//first element in a result set
$('ul li:first')

//first child of the parent
$('ul li:first-child')

//last element in a result set
$('ul li:last')

//last child of the parent
$('ul li:last-child')

//all odd elements in a result set
$('ul li:odd')

//all even elements in a result set
$('ul li:even')
```

```
<div class="welcome"></div>

<ul id="conference">

    <li class="attendee">
    <a title="addy osmani"
     href="http://addyosmani.com">Addy</a>
    </li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Adam Sontag</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="text" name="attendeeName"
value="John"/>
```

Code


```
// Filter selectors

//first element in a result set
$('ul li:first')

//first child of the parent
$('ul li:first-child')

//last element in a result set
$('ul li:last')

//last child of the parent
$('ul li:last-child')

//all odd elements in a result set
$('ul li:odd')

//all even elements in a result set
$('ul li:even')
```

```
<div class="welcome"></div>

<ul id="conference">

    <li class="attendee">
    <a title="addy osmani"
     href="http://addyosmani.com">Addy</a>
    </li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Adam Sontag</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="text" name="attendeeName"
value="John"/>
```

Code


```
// Filter selectors

//first element in a result set
$('ul li:first')

//first child of the parent
$('ul li:first-child')

//last element in a result set
$('ul li:last')

//last child of the parent
$('ul li:last-child')

//all odd elements in a result set
$('ul li:odd')

//all even elements in a result set
$('ul li:even')
```

```
<div class="welcome"></div>

<ul id="conference">

    <li class="attendee">
    <a title="addy osmani"
     href="http://addyosmani.com">Addy</a>
    </li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Adam Sontag</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="text" name="attendeeName"
value="John"/>
```

Code


```
// Filter selectors

//first element in a result set
$('ul li:first')

//first child of the parent
$('ul li:first-child')

//last element in a result set
$('ul li:last')

//last child of the parent
$('ul li:last-child')

//all odd elements in a result set
$('ul li:odd')

//all even elements in a result set
$('ul li:even')
```

```
<div class="welcome"></div>

<ul id="conference">

    <li class="attendee">
    <a title="addy osmani"
     href="http://addyosmani.com">Addy</a>
    </li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Adam Sontag</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="text" name="attendeeName"
value="John"/>
```

Code


```
// Filter selectors

//first element in a result set
$('ul li:first')

//first child of the parent
$('ul li:first-child')

//last element in a result set
$('ul li:last')

//last child of the parent
$('ul li:last-child')

//all odd elements in a result set
$('ul li:odd')

//all even elements in a result set
$('ul li:even')
```

```
<div class="welcome"></div>

<ul id="conference">

    <li class="attendee">
    <a title="addy osmani"
     href="http://addyosmani.com">Addy</a>
    </li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Adam Sontag</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="text" name="attendeeName"
value="John"/>
```

Hierarchal Selectors

Parent/child selectors

Code


```
// Parent/child selectors

// a b, returns children of the
// parent a
$('ul li')

// a > b returns elements that are a
// child element of a
$('body > ul')

// returns the elements that are
// adjacent to the selector
$('label + input')

// a ~ b, returns b elements that
// are a sibling of a
$('p ~ ul')
```

```
<p>
<ul id="conference">

    <li class="attendee">
    <a title="addy osmani"
     href="http://addyosmani.com">Addy</a>
    </li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Adam Sontag</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>
</p>

<label>Conference newsletter:</label>
<input name="newsletter" />
```

Code


```
// Parent/child selectors

// a b, returns children of the
// parent a
$('ul li')

// a > b returns elements that are a
// child element of a
$('body > ul')

// returns the elements that are
// adjacent to the selector
$('label + input')

// a ~ b, returns b elements that
// are a sibling of a
$('p ~ ul')
```

```
<p>
<ul id="conference">

    <li class="attendee">
    <a title="addy osmani"
     href="http://addyosmani.com">Addy</a>
    </li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Adam Sontag</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>
</p>

<label>Conference newsletter:</label>
<input name="newsletter" />
```

Code


```
// Parent/child selectors

// a b, returns children of the
// parent a
$('ul li')

// a > b returns elements that are a
// child element of a
$('body > ul')

// returns the elements that are
// adjacent to the selector
$('label + input')

// a ~ b, returns b elements that
// are a sibling of a
$('p ~ ul')
```

```
<p>
<ul id="conference">

    <li class="attendee">
    <a title="addy osmani"
     href="http://addyosmani.com">Addy</a>
    </li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Adam Sontag</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>
</p>

<label>Conference newsletter:</label>
<input name="newsletter" />
```

Code


```
// Parent/child selectors

// a b, returns children of the
// parent a
$('ul li')

// a > b returns elements that are a
// child element of a
$('body > ul')

// returns the elements that are
// adjacent to the selector
$('label + input')

// a ~ b, returns b elements that
// are a sibling of a
$('p ~ ul')
```

```
<p>
<ul id="conference">

    <li class="attendee">
    <a title="addy osmani"
     href="http://addyosmani.com">Addy</a>
    </li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Adam Sontag</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>
</p>

<label>Conference newsletter:</label>
<input name="newsletter" />
```

Code


```
// Parent/child selectors

// a b, returns children of the
// parent a
$('ul li')

// a > b returns elements that are a
// child element of a
$('body > ul')

// returns the elements that are
// adjacent to the selector
$('label + input')

// a ~ b, returns b elements that
// are a sibling of a
$('p ~ ul')
```

```
<p>
<ul id="conference">

    <li class="attendee">
    <a title="addy osmani"
     href="http://addyosmani.com">Addy</a>
    </li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Adam Sontag</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>
</p>

<label>Conference newsletter:</label>
<input name="newsletter" />
```

‘I work with forms. Anything that can help selections there?’

Form Filters

Selectors for handling form elements (e.g. inputs, password fields, checkbox etc.)

Code


```
//form filters

//returns elements that are
enabled
$('input:enabled')

//returns elements that are
enabled
$('input:disabled')

//returns checked items
$(':checked')

//returns selected items
$('select option:selected')
```

```
<div class="welcome"></div>

<input type="text" name="attendeeName"
value="John"/>

<input disabled=”disabled” type="text"
name="attendeeAge" value="28"/>

<input type="checkbox" checked="checked"
value="enterContest" />

<select name="prizes">
    <option>Apple iPad</option>
    <option selected="selected">
      Google Chromebook
    </option>
</select>
```

Code


```
//form filters

//returns elements that are
enabled
$('input:enabled')

//returns elements that are
enabled
$('input:disabled')

//returns checked items
$(':checked')

//returns selected items
$('select option:selected')
```

```
<div class="welcome"></div>

<input type="text" name="attendeeName"
value="John"/>

<input disabled=”disabled” type="text"
name="attendeeAge" value="28"/>

<input type="checkbox" checked="checked"
value="enterContest" />

<select name="prizes">
    <option>Apple iPad</option>
    <option selected="selected">
      Google Chromebook
    </option>
</select>
```

Code


```
//form filters

//returns elements that are
enabled
$('input:enabled')

//returns elements that are
enabled
$('input:disabled')

//returns checked items
$(':checked')

//returns selected items
$('select option:selected')
```

```
<div class="welcome"></div>

<input type="text" name="attendeeName"
value="John"/>

<input disabled=”disabled” type="text"
name="attendeeAge" value="28"/>

<input type="checkbox" checked="checked"
value="enterContest" />

<select name="prizes">
    <option>Apple iPad</option>
    <option selected="selected">
      Google Chromebook
    </option>
</select>
```

Code


```
//form filters

//returns elements that are
enabled
$('input:enabled')

//returns elements that are
enabled
$('input:disabled')

//returns checked items
$(':checked')

//returns selected items
$('select option:selected')
```

```
<div class="welcome"></div>

<input type="text" name="attendeeName"
value="John"/>

<input disabled=”disabled” type="text"
name="attendeeAge" value="28"/>

<input type="checkbox" checked="checked"
value="enterContest" />

<select name="prizes">
    <option>Apple iPad</option>
    <option selected="selected">
      Google Chromebook
    </option>
</select>
```

Code


```
//form filters

//returns elements that are
enabled
$('input:enabled')

//returns elements that are
enabled
$('input:disabled')

//returns checked items
$(':checked')

//returns selected items
$('select option:selected')
```

```
<div class="welcome"></div>

<input type="text" name="attendeeName"
value="John"/>

<input disabled=”disabled” type="text"
name="attendeeAge" value="28"/>

<input type="checkbox" checked="checked"
value="enterContest" />

<select name="prizes">
    <option>Apple iPad</option>
    <option selected="selected">
      Google Chromebook
    </option>
</select>
```

“I’d like to select elements using indices or equations. Can you help?”

nth-child Filters

Filtering using the CSS nth-child selector

Code


```
//nth-child filters

//nth child in a result set
$('ul li:nth-child(2)')

//all odd numbered results
$('ul li:nth-child(odd)')

//all even numbered results
$('ul li:nth-child(even)')

//all elements based on a formula.
here it's every 2nd child
$('ul li:nth-child(2n)')
```

```
<div class="welcome"></div>

<ul id="conference">

    <li class="attendee">
    <a title="addy osmani"
     href="http://addyosmani.com">Addy</a>
    </li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Adam Sontag</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="text" name="attendeeName"
value="John"/>
```

Code


```
//nth-child filters

//nth child in a result set
$('ul li:nth-child(2)')

//all odd numbered results
$('ul li:nth-child(odd)')

//all even numbered results
$('ul li:nth-child(even)')

//all elements based on a formula.
here it's every 2nd child
$('ul li:nth-child(2n)')
```

```
<div class="welcome"></div>

<ul id="conference">

    <li class="attendee">
    <a title="addy osmani"
     href="http://addyosmani.com">Addy</a>
    </li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Adam Sontag</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="text" name="attendeeName"
value="John"/>
```

Code


```
//nth-child filters

//nth child in a result set
$('ul li:nth-child(2)')

//all odd numbered results
$('ul li:nth-child(odd)')

//all even numbered results
$('ul li:nth-child(even)')

//all elements based on a formula.
here it's every 2nd child
$('ul li:nth-child(2n)')
```

```
<div class="welcome"></div>

<ul id="conference">

    <li class="attendee">
    <a title="addy osmani"
     href="http://addyosmani.com">Addy</a>
    </li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Adam Sontag</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="text" name="attendeeName"
value="John"/>
```

Code


```
//nth-child filters

//nth child in a result set
$('ul li:nth-child(2)')

//all odd numbered results
$('ul li:nth-child(odd)')

//all even numbered results
$('ul li:nth-child(even)')

//all elements based on a formula.
here it's every 2nd child
$('ul li:nth-child(2n)')
```

```
<div class="welcome"></div>

<ul id="conference">

    <li class="attendee">
    <a title="addy osmani"
     href="http://addyosmani.com">Addy</a>
    </li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Adam Sontag</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="text" name="attendeeName"
value="John"/>
```

Code


```
//nth-child filters

//nth child in a result set
$('ul li:nth-child(2)')

//all odd numbered results
$('ul li:nth-child(odd)')

//all even numbered results
$('ul li:nth-child(even)')

//all elements based on a formula.
here it's every 2nd child
$('ul li:nth-child(2n)')
```

```
<div class="welcome"></div>

<ul id="conference">

    <li class="attendee">
    <a title="addy osmani"
     href="http://addyosmani.com">Addy</a>
    </li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Adam Sontag</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="text" name="attendeeName"
value="John"/>
```

More Pseudo Selectors

Selectors for handling whether selectors are visible or hidden and more

Code


```
//more pseudo selectors

// returns elements that are hidden
$('li:hidden')

// returns elements that are visible
$('li:visible')

// returns elements that don't match
// the condition supplied
$('input:not(:checked)')

// returns elements that are equal to
// the index supplied
$('ul li:eq(1)')

// returns elements that are greather
// than the index supplied
$('ul li:gt(2)')

// returns elements in a set less than
// the index supplied
$('ul li:lt(2)')
```

```
<style type="text/css">
.hidden{ display:none}
</style>

<ul id="conference">

    <li class="hidden attendee">Addy</li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="checkbox"/>
```

Code


```
//more pseudo selectors

// returns elements that are hidden
$('li:hidden')

// returns elements that are visible
$('li:visible')

// returns elements that don't match
// the condition supplied
$('input:not(:checked)')

// returns elements that are equal to
// the index supplied
$('ul li:eq(1)')

// returns elements that are greather
// than the index supplied
$('ul li:gt(2)')

// returns elements in a set less than
// the index supplied
$('ul li:lt(2)')
```

```
<style type="text/css">
.hidden{ display:none}
</style>

<ul id="conference">

    <li class="hidden attendee">Addy</li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="checkbox"/>
```

Code


```
//more pseudo selectors

// returns elements that are hidden
$('li:hidden')

// returns elements that are visible
$('li:visible')

// returns elements that don't match
// the condition supplied
$('input:not(:checked)')

// returns elements that are equal to
// the index supplied
$('ul li:eq(1)')

// returns elements that are greather
// than the index supplied
$('ul li:gt(2)')

// returns elements in a set less than
// the index supplied
$('ul li:lt(2)')
```

```
<style type="text/css">
.hidden{ display:none}
</style>

<ul id="conference">

    <li class="hidden attendee">Addy</li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="checkbox"/>
```

Code


```
//more pseudo selectors

// returns elements that are hidden
$('li:hidden')

// returns elements that are visible
$('li:visible')

// returns elements that don't match
// the condition supplied
$('input:not(:checked)')

// returns elements that are equal to
// the index supplied
$('ul li:eq(1)')

// returns elements that are greather
// than the index supplied
$('ul li:gt(2)')

// returns elements in a set less than
// the index supplied
$('ul li:lt(2)')
```

```
<style type="text/css">
.hidden{ display:none}
</style>

<ul id="conference">

    <li class="hidden attendee">Addy</li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="checkbox"/>
```

Code


```
//more pseudo selectors

// returns elements that are hidden
$('li:hidden')

// returns elements that are visible
$('li:visible')

// returns elements that don't match
// the condition supplied
$('input:not(:checked)')

// returns elements that are equal to
// the index supplied
$('ul li:eq(1)')

// returns elements that are greather
// than the index supplied
$('ul li:gt(2)')

// returns elements in a set less than
// the index supplied
$('ul li:lt(2)')
```

```
<style type="text/css">
.hidden{ display:none}
</style>

<ul id="conference">

    <li class="hidden attendee">Addy</li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="checkbox"/>
```

Code

```
//more pseudo selectors

// returns elements that are hidden
$('li:hidden')

// returns elements that are visible
$('li:visible')

// returns elements that don't match
// the condition supplied
$('input:not(:checked)')

// returns elements that are equal to
// the index supplied
$('ul li:eq(1)')

// returns elements that are greather
// than the index supplied
$('ul li:gt(2)')

// returns elements in a set less than
// the index supplied
$('ul li:lt(2)')
```

```
<style type="text/css">
.hidden{ display:none}
</style>

<ul id="conference">

    <li class="hidden attendee">Addy</li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="checkbox"/>
```

Code


```
//more pseudo selectors

// returns elements that are hidden
$('li:hidden')

// returns elements that are visible
$('li:visible')

// returns elements that don't match
// the condition supplied
$('input:not(:checked)')

// returns elements that are equal to
// the index supplied
$('ul li:eq(1)')

// returns elements that are greather
// than the index supplied
$('ul li:gt(2)')

// returns elements in a set less than
// the index supplied
$('ul li:lt(2)')
```

```
<style type="text/css">
.hidden{ display:none}
</style>

<ul id="conference">

    <li class="hidden attendee">Addy</li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="checkbox"/>
```

‘What about elements containing some specific text or elements?’

Content Filters

Filters for narrowing down elements with specific content

Code


```
//content filters

// :has(a) all elements with a
descendant that matches a
$('div:has(p)')

//:contains(a) the element
contains a
$('li:contains("Dan")')

//returns elements that are empty
$(':empty')

//returns the parent of li
$('li:parent')
```

```
<div class="welcome">
    <p>attendee</p>
</div>

<p>
<ul id="conference">

    <li class="attendee">Addy</li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee"></li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>
</p>
```

Code


```
//content filters

// :has(a) all elements with a
descendant that matches a
$('div:has(p)')

//:contains(a) the element
contains a
$('li:contains("Dan")')

//returns elements that are empty
$(':empty')

//returns the parent of li
$('li:parent')
```

```
<div class="welcome">
    <p>attendee</p>
</div>

<p>
<ul id="conference">

    <li class="attendee">Addy</li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee"></li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>
</p>
```

Code


```
//content filters

// :has(a) all elements with a
descendant that matches a
$('div:has(p)')

//:contains(a) the element
contains a
$('li:contains("Dan")')

//returns elements that are empty
$(':empty')

//returns the parent of li
$('li:parent')
```

```
<div class="welcome">
    <p>attendee</p>
</div>

<p>
<ul id="conference">

    <li class="attendee">Addy</li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee"></li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>
</p>
```

Code


```
//content filters

// :has(a) all elements with a
descendant that matches a
$('div:has(p)')

//:contains(a) the element
contains a
$('li:contains("Dan")')

//returns elements that are empty
$(':empty')

//returns the parent of li
$('li:parent')
```

```
<div class="welcome">
    <p>attendee</p>
</div>

<p>
<ul id="conference">

    <li class="attendee">Addy</li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee"></li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>
</p>
```

Code


```
//content filters

// :has(a) all elements with a
descendant that matches a
$('div:has(p)')

//:contains(a) the element
contains a
$('li:contains("Dan")')

//returns elements that are empty
$(':empty')

//returns the parent of li
$('li:parent')
```

```
<div class="welcome">
    <p>attendee</p>
</div>

<p>
<ul id="conference">

    <li class="attendee">Addy</li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee"></li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>
</p>
```

jQuery Collections

Understanding what $() returns

Code


```
// Collections

// this returns a jQuery collection
// your selection wrapped inside a
// jQuery object that can be further
// manipulated
$('ul li')

// we can treat it a little like an array
$('ul li').length //5

// we can iterate over it..
$('ul li').each(function(i, x){
    console.log($(this));
});

// and we can also call methods on it
$('ul li').hide(); //hides these elements
```

```
<div class="welcome"></div>

<ul id="conference">

    <li class="attendee">
    <a title="addy osmani"
     href="http://addyosmani.com">Addy</a>
    </li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Adam Sontag</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="text" name="attendee_name"
value="John"/>
```

“I’ve heard you can chain methods with jQuery...how does that work? Why would I use it?”

Method Chaining

Chaining method calls to write shorter, less repetitive code

Code


```
// Chaining

// When you call a method on
// a jQuery object another
// jQuery object is usually returned
// Because of this, we're able to
// easily 'chain' together methods

$('ul')
    .find('.attendee')
    .addClass('arrived')
    .removeClass('pending');
```

```
<div class="welcome"></div>

<ul id="conference">

    <li class="attendee">
    <a title="addy osmani"
     href="http://addyosmani.com">Addy</a>
    </li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Adam Sontag</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="text" name="attendee_name"
value="John"/>
```

Code


```
// Accessing the original selection

// We sometimes want to access the
// original selection again and we
// can use .end() to achieve this..

$('ul')
    .find('.attendee')
    .addClass('arrived')
    .removeClass('pending')
  .end()
    .addClass('seated');
```

```
<div class="welcome"></div>

<ul id="conference">

    <li class="attendee">
    <a title="addy osmani"
     href="http://addyosmani.com">Addy</a>
    </li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Adam Sontag</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="text" name="attendee_name"
value="John"/>
```

‘Do I re-query the DOM every time I want to do something with an element?’

Caching

How to avoid re-querying the DOM unless absolutely necessary

jQuery

- jQuery makes **selecting** elements simple
- However, behind the scenes a lot of work can be involved in **querying** the DOM
- By caching selections, we avoid having to **re-query the DOM unless necessary**

Code


```
// uncached selections

$('.attendee').fadeIn();

$('.attendee').css('color','green');

$('.attendee').on('click',function(){
    console.log('hello world');
})
```

```
<div class="welcome"></div>

<ul id="conference">

    <li class="attendee">
    <a title="addy osmani"
     href="http://addyosmani.com">Addy</a>
    </li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Adam Sontag</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="text" name="attendee_name"
value="John"/>
```

Code


```
// uncached selections

$('.attendee').fadeIn();

$('.attendee').css('color','green');

$('.attendee').on('click',function(){
    console.log('hello world');
})
```

```
<div class="welcome"></div>

<ul id="conference">

    <li class="attendee">
    <a title="addy osmani"
     href="http://addyosmani.com">Addy</a>
    </li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Adam Sontag</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="text" name="attendee_name"
value="John"/>
```

Code


```
// cache the selection
var attendees = $('.attendee');

// use as needed
attendees.fadeIn();

attendees.css('color','green');

attendees.on('click', function(){
    console.log('hello world');
});
```

```
<div class="welcome"></div>

<ul id="conference">

    <li class="attendee">
    <a title="addy osmani"
     href="http://addyosmani.com">Addy</a>
    </li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Adam Sontag</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="text" name="attendee_name"
value="John"/>
```

Code


```
//but this also supports chaining!

var attendees = $('.attendee');

attendees
    .fadeIn()
    .css('color','green')
    .on('click', function(){
        console.log('hello world');
    });
```

```
<div class="welcome"></div>

<ul id="conference">

    <li class="attendee">
    <a title="addy osmani"
     href="http://addyosmani.com">Addy</a>
    </li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Adam Sontag</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="text" name="attendee_name"
value="John"/>
```

‘How do I modify the style of elements using jQuery?’

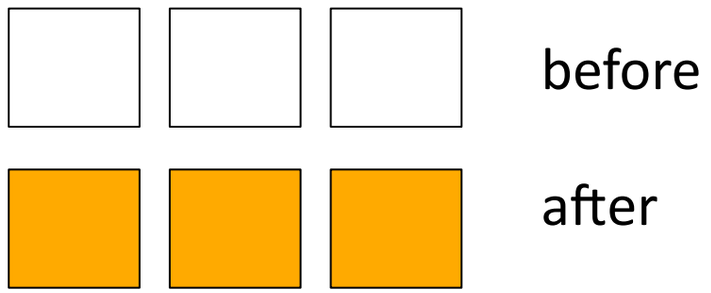
CSS

Methods for getting and setting CSS-related properties of elements

As we saw earlier..

– With JavaScript, you would style elements like this..

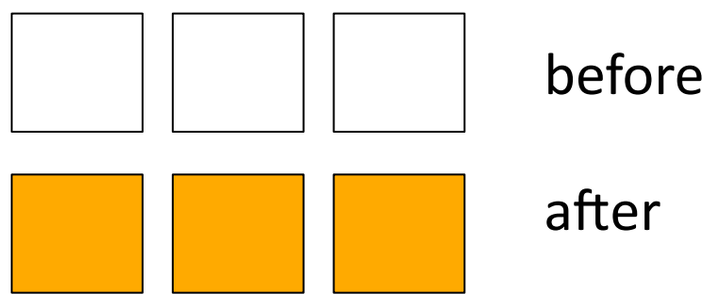
```
var elems = document.getElementsByClassName('attendee'),
    len  = elems.length,
    i=0;

for(; i<len; i++) {
  elems[i].style.backgroundColor = 'orange';
}
```



But..

– With jQuery, it's just a simple one-liner:

```
$('.attendee').css('backgroundColor','orange');
```



The .css() method

- Supports a few different **signatures**:
  - `$(elem).css(name)` - gets a CSS property
  - `$(elem).css(name, value)` - sets a property
  - `$(elem).css({multiple properties})` - set multiple properties and values

CSS

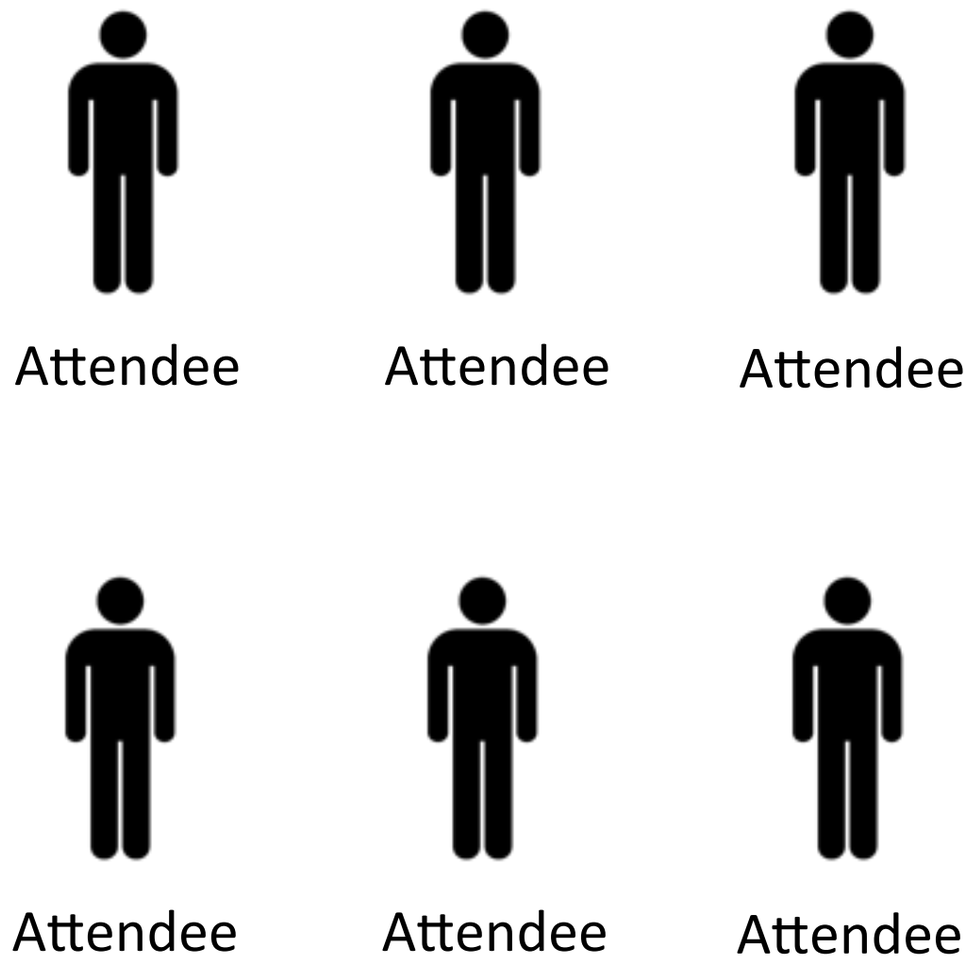
- We also support some other **utility** methods:
  - `$(elem).addClass('className');`
  - `$(elem).removeClass('className');`
  - `$(elem).toggleClass('className');`
  - and more!.

Code

```
$('.attendee')
            .removeClass('arrived')
            .addClass('attending')
            .css({
                'font-size':'20px',
                'color':'orange',
                'border':'1px solid
                black',
            })
            .toggleClass('seated');
```

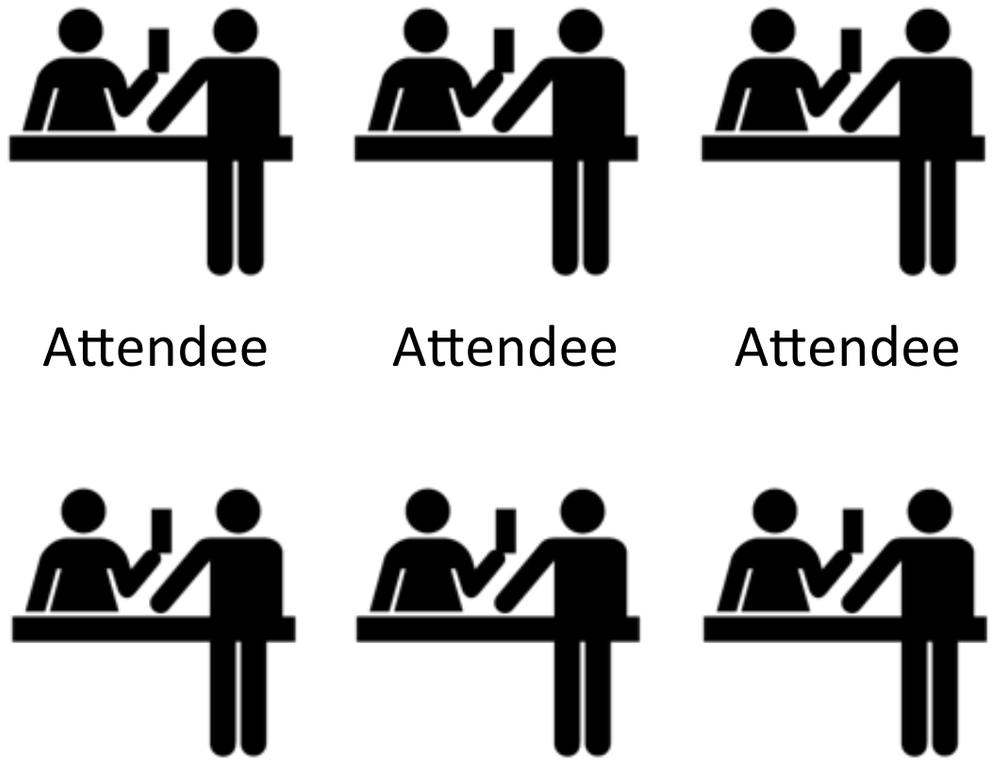
Code


```
$('.attendee')
            .removeClass('arrived')
            .addClass('attending')
            .css({
                'font-size':'20px',
                'color':'orange',
                'border':'1px solid
                black',
            })
            .toggleClass('seated');
```



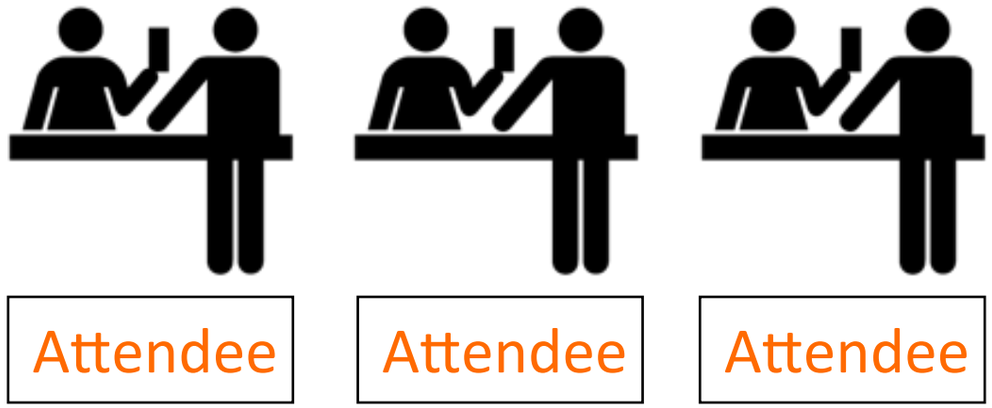
Code

```
$('.attendee')
            .removeClass('arrived')
            .addClass('attending')
            .css({
                'font-size':'20px',
                'color':'orange',
                'border':'1px solid
                black',
            })
            .toggleClass('seated');
```



Code

```
$(‘.attendee’)
            .removeClass(‘arrived’)
            .addClass(‘attending’)
            .css({
                ‘font-size’:‘20px’,
                ‘color’:‘orange’,
                ‘border’:‘1px solid
                black’,
            })
            .toggleClass(‘seated’);
```



Code


```
$('.attendee')
            .removeClass('arrived')
            .addClass('attending')
            .css({
                'font-size':'20px',
                'color':'orange',
                'border':'1px solid
                black',
            })
            .toggleClass('seated');
```

Code

```
var attendees = $('.attendee'),

// Returns 'orange'
currentTextColor = attendees.css('color'),

// Returns '1px solid black'
currentBorder    = attendees.css('border'),

// Returns '20px'
currentFontSize  = attendees.css('font-size');
```

Code


```
// Single property changes
$('#container').css('backgroundColor','orange');

// Multiple property changes
$('.attendee').css({
   'width':'200',
   'height':'100',
   'color':'red',
   'backgroundColor':'blue'
});

// Working with cached elements
var myDiv = $('div');
myDiv.css('color','green');
```

‘I want things to swoosh from one side to another or just fade in. How can I animate?’

Effects

Short-hand and long-hand approaches for animating elements

Animation

- There are both short-hand and long-hand methods of doing animation:
  - $(elem).show()
  - $(elem).hide()
  - $(elem).fadeIn()
  - $(elem).fadeOut()
  - $(elem).animate()

Short-hand: Hide and Show


```
// cache the selection
var attendees = $('.attendee');

// Hide all attendees
attendees.hide();
```

```
<div class="welcome"></div>

<ul id="conference">

    <li class="attendee">
    <a title="addy osmani"
     href="http://addyosmani.com">Addy</a>
    </li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Adam Sontag</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="text" name="attendee_name"
value="John"/>
```

Short-hand: Hide and Show


```
// cache the selection
var attendees = $('.attendee');

// Show all attendees
attendees.show();
```

```
<div class="welcome"></div>

<ul id="conference">

    <li class="attendee">
    <a title="addy osmani"
     href="http://addyosmani.com">Addy</a>
    </li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Adam Sontag</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="text" name="attendee_name"
value="John"/>
```

Short-hand: Fade-in/Fade-out


```
// cache the selection
var attendees = $('.attendee');

// Fade-out all attendees
attendees.fadeOut();

// Fade-in all attendees
attendees.fadeIn();
```

```
<div class="welcome"></div>

<ul id="conference">

    <li class="attendee">
    <a title="addy osmani"
     href="http://addyosmani.com">Addy</a>
    </li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Adam Sontag</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="text" name="attendee_name"
value="John"/>
```

Short-hand: A little better


```
// cache the selection
var attendees = $('.attendee');

// Show all attendees over time
attendees.show(500, function(){
   alert(‘there you are!’);
});

// Fade-out all attendees over time
attendees.fadeOut(500, function(){
   alert(‘you disappeared!’);
});
```

```
<div class="welcome"></div>

<ul id="conference">

   <li class="attendee">
   <a title="addy osmani"
    href="http://addyosmani.com">Addy</a>
   </li>
   <li class="attendee">Dan Heberden</li>
   <li class="attendee">Adam Sontag</li>
   <li class="attendee">Mathias Bynens</li>
   <li class="attendee">Douglas</li>

</ul>

<input type="text" name="attendee_name"
value="John"/>
```

Long-hand: animate()


```
attendees.animate({
    'width': '200',
    'height': '300',
    'opacity': '0.5',
    'marginLeft': '20'
    },
    200,
    function(){
        //animation complete
});
```

Long-hand: animate()


```
attendees.animate({
    'width': '200',
    'height': '300',
    'opacity': '0.5',
    'marginLeft': '20'
    },
    200,
    function(){
        //animation complete
});
```

What would you like to animate?

Long-hand: animate()


```
attendees.animate({
    'width': '200',
    'height': '300',
    'opacity': '0.5',
    'marginLeft': '20'
    },
    200,
    function(){
        //animation complete
});
```

Duration for the animation (ms)

Long-hand: animate()


```
attendees.animate({
    'width': '200',
    'height': '300',
    'opacity': '0.5',
    'marginLeft': '20'
    },
    200,
    function(){
        //animation complete
});
```

Callback function for when the animation completes.

Step-functions:


```
attendees.animate({
  opacity: '0.5',
  height: '50%'
},{
    // step: a callback function fired at
    // each step of an animation
step: function(now, fx) {
    var data = fx.elem.id + ' ' + fx.prop + ': ' + now;
    $('body').append('<div>' + data + '</div>');
  }
});
```

Easing functions:


```
$('#container').animate({
    width: ['toggle', 'swing'],
    height: ['toggle', 'swing'],
    opacity: 'toggle'
  }, 500,
  'linear',
  function() {
      $(this).after('<div>Animation complete.</div>');
});
```

Easing functions:


```
$('#container').animate({
    width: ['toggle', 'swing'],
    height: ['toggle', 'swing'],
    opacity: 'toggle'
  }, 500,
  // an easing function that sets the speed at which the
  // animation progresses at different points in the
  // animation. Think of 'linear' as smooth.
  'linear',
  function() {
      $(this).after('<div>Animation complete.</div>');
});
```

“I want to get or post content using Ajax, so I don’t have to reload the page. Is that hard to do?”

Ajax

Performing asynchronous HTTP requests

Ajax

- jQuery supports both **short** and **long-hand** methods for making Ajax requests
  - $.get()
  - $.post()
  - $.getJSON()
  - $.ajax()
  - and others
- Cross-browser XHR **without** the headaches!

Short-hand: Get and Post


```
// Get data
$.get('attendees.html', function( data ) {
   $('.result').html(data);
});

// Post data
$.post('attendees.php',
{ name: 'John' }, function( data ){
   $('.result').html(data);
});
```

Short-hand: Get and Post


```
// Get data
$.get('attendees.html', function( data ) {
  $('.result').html( data );
});

// Post data
$.post('attendees.php',
{ name: 'John' }, function( data ){
  $('.result').html( data );
});
```

Short-hand: Get JSON


```
$.getJSON('attendees.json', function( data ) {
  var attendees = [];
  $.each( data, function( key, val ) {
    attendees.push('<li id="' + key + '">' + val
+'</li>');
  });

  $('<ul/>', {
    'class': 'attendees',
    html: attendees.join('')
  }).appendTo('body');
});
```

Getting JSON


```
$.getJSON('attendees.json', function( data ) {
  var attendees = [];
  $.each( data, function( key, val ) {
    attendees.push('<li id="' + key + '">' + val
+'</li>');
  });

  $('<ul/>', {
    'class': 'attendees',
    html: attendees.join('')
  }).appendTo('body');
});
```

Iterating over our data
and pushing into an array

Getting JSON


```
$.getJSON('attendees.json', function( data ) {
  var attendees = [];
  $.each( data, function( key, val ) {
    attendees.push('<li id="' + key + '">' + val
+'</li>');
  });

  $('<ul/>', {
    'class': 'attendees',
    html: attendees.join('')
  }).appendTo('body');
});
```

Dynamically generate a list containing our items and append to the document body

Ajax: Retrieving content


```
// Retrieve the latest version of a web page

$.ajax({
   url: "attendees.html",
   cache: false,
   // What to do if this is successful:
   success: function(html){
      $("#results").append(html);
   },
   // What to do if this fails:
   error: function(){
       //something went wrong
   }
});
```

Ajax: Retrieving content


```
// Retrieve the latest version of a web page

$.ajax({
   url: "attendees.html",
   cache: false,
   // What to do if this is successful:
   success: function(html){
      $("#results").append(html);
   },
   // What to do if this fails:
   error: function(){
       //something went wrong
   }
});
```

Ajax: Retrieving content


```
// Retrieve the latest version of a web page

$.ajax({
   url: "attendees.html",
   cache: false,
   // What to do if this is successful:
   success: function(html){
      $("#results").append(html);
   },
   // What to do if this fails:
   error: function(){
       //something went wrong
   }
});
```

Ajax: Retrieving content


```
// More future-proof version:
var jqxhr = $.ajax({
      url: "attendees.html",
      cache: false
    })
    .done(function(){
      // success!
    })
    .fail"(function(){
      // error!
    });

// Better as .success() and .error() are going.
```

Ajax: We also support..


```
var request = $.ajax({
    url: 'attendees.html',
    type: 'POST',
    data: {id : attendeeID},
    dataType: 'html',
    cache: 'false'
});

request.done(function(data){
    console.log('Received:' + data);
});

request.fail(function(data, status){
    console.log('failed because:' + status);
});
```

Advanced Ajax: Deferreds


```
// Let's make a request for a file
function getFile(){
    return $.get('attendees.html');
}

// All $.ajax/short-hand ajax methods support returning
// a promise for something to be completed. This allows us
// to make non-blocking calls to servers.
$.when( getFile() )
    .then(function(){
        console.log( 'I fire when getFile() has completed!' );
    })
    .fail(function(){
        console.log( 'I fire if requests fail!' );
    });
// Read the above as 'when' getFile() is resolved, then do
something. If this fails do something else.
```

“How do I attach events to elements? I’d like to have something happen when I click or hover over them”

Events

Registering behaviours which are applied when a user interacts with the browser

jQuery

- Before jQuery 1.7:
  - Many ways to attach event handlers to elements
    - **.bind()** - basic attachment of a handler to an element
    - **.delegate()** - supports attaching handlers to elements that have been added dynamically after page-load.
    - **.live()** - similar to delegate, but not as great with larger groups of elements.
  - and the following for 'unbinding' handlers:
    - .unbind()
    - .undelegate()

‘Whoa. Wait....three methods? But how do I know which to use? Couldn’t this be made simpler?’

jQuery

- As of jQuery 1.7:
  - We now have:
    - **.on()**
  - and unbind..
    - **.off()**
  - for all possible cases
  - Much simpler to understand.

Code


```
// Binding a click event to elements
// with the class 'attendee'
$('.attendee').on('click', function(){
    $(this).css('color','red');
});

// Remove just the click event handler
$('.attendee').off('click','**');

// Remove all event handlers from
// attendees e.g. if others had been
// set
$('.attendee').off();
```

Binding: Same syntax


```
// Old way of doing things

$('a').bind('click', myHandler);

$('form')
.bind('submit', { val: 42 }, fn);

$('.comment')
.delegate('a.add', 'click', addNew);

$('.dialog')
.undelegate('a', 'click.myDlg');

$('a').live('click', fn);

$('a').die('click');
```

```
// The new, simpler, hotness

$('a').on('click', myHandler);

$('form')
.on('submit', { val: 42 }, fn);

$('.comment')
.on('click', 'a.add', addNew);

$('.dialog')
.off('click.myDlg', 'a');

$(document).on('click', 'a', fn);

$(document).off('click', 'a');
```

Delegation: Order changes


```
// Old way of doing things

$('a').bind('click', myHandler);

$('form')
.bind('submit', { val: 42 }, fn);

$('.comment')
.delegate('a.add', 'click', addNew);

$('.dialog')
.undelegate('a', 'click.myDlg');

$('a').live('click', fn);

$('a').die('click');
```

```
// The new, simpler, hotness

$('a').on('click', myHandler);

$('form')
.on('submit', { val: 42 }, fn);

$('.comment')
.on('click', 'a.add', addNew);

$('.dialog')
.off('click.myDlg', 'a');

$(document).on('click', 'a', fn);

$(document).off('click', 'a');
```

Live: More minor changes


```
// Old way of doing things

$('a').bind('click', myHandler);

$('form')
.bind('submit', { val: 42 }, fn);

$('.comment')
.delegate('a.add', 'click', addNew);

$('.dialog')
.undelegate('a', 'click.myDlg');

$('a').live('click', fn);

$('a').die('click');
```

```
// The new, simpler, hotness

$('a').on('click', myHandler);

$('form')
.on('submit', { val: 42 }, fn);

$('.comment')
.on('click', 'a.add', addNew);

$('.dialog')
.off('click.myDlg', 'a');

$(document).on('click', 'a', fn);

$(document).off('click', 'a');
```

Code: Prevent defaults


```
// .preventDefault() allows us to cancel
// only the *default* action

$("form").on("submit", function(event) {
  event.preventDefault();
});


// .stopPropagation() allows us to stop
// events from bubbling without preventing
// the standard form submit

$("form").on("submit", function(event) {
  event.stopPropagation();
});
```

```
<form action="attend.php"
method="POST">
<input type="checkbox"
name="attending" checked/>
</form>
```

Code: Prevent defaults


```
// .preventDefault() allows us to cancel
// only the *default* action

$("form").on("submit", function(event) {
  event.preventDefault();
});


// .stopPropagation() allows us to stop
// events from bubbling without preventing
// the standard form submit

$("form").on("submit", function(event) {
  event.stopPropagation();
});
```

```
<form action="attend.php"
method="POST">
<input type="checkbox"
name="attending" checked/>
</form>
```

“I usually just store information in variables, but I’d like to associate data with elements. Can jQuery help?”

Data

Storing and retrieving arbitrary data using specific DOM elements

jQuery

- jQuery has an internal **data cache**:
  - Can be used to store data in key/value pairs
  - Data is stored against any DOM elements
  - Data can be stored and retrieved, a little like a database (store)

Code


```
var last = $('.attendee:last');

// Set some data with $.fn.data()

last.data('firstName', 'John');

last.data('lastName', 'Resig');


// You can then access this data
// as follows:

console.log(
   last.data('firstName')
);
```

```
<div class="welcome"></div>

<ul id="conference">

    <li class="attendee">
    <a title="addy osmani"
     href="http://addyosmani.com">Addy</a>
    </li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Adam Sontag</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="text" name="attendee_name"
value="John"/>
```

Code


```
// utility methods
// May be deprecated in the future
// or moved to plugins

// does the DOM node or object
// have data attached?
$.hasData(lastAttendee); //true

// Removes data stored under
// a specific key
$.removeData(lastAttendee, 'firstName');

//removes all data
$.removeData(lastAttendee);

// Lower level data method for
// handling data associated with
// an element
$.data('.attendee:first', 'foo', 'bar');
//same as elem.data('foo','bar')
```

```
<div class="welcome"></div>

<ul id="conference">

    <li class="attendee">
    <a title="addy osmani"
     href="http://addyosmani.com">Addy</a>
    </li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Adam Sontag</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="text" name="attendee_name"
value="John"/>
```

“I want to iterate through the elements returned by a selector. How can I do that?”

Utilities

Iterating over collections and traversing the DOM for other subsets of collections

Code


```
//Iterators

// .each()
$(‘li’).each(function(index)){
   alert(index + ': ' + $(this).text
());
});

// but if we want to pass each
// element through a function..

// .map()
$(".welcome")
.append( $(“li”).map(function(){
      return $(this).val();
    })
.get()
.join(", ") );

// appends comma-separated list of
li values
```

```
<div class="welcome"></div>

<ul id="conference">

    <li class="attendee">
    <a title="addy osmani"
     href="http://addyosmani.com">Addy</a>
    </li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Adam Sontag</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="text" name="attendeeName"
value="John"/>
```

Code


```
//Iterators

// .each()
$('li').each(function(index){
   alert(index + ': ' + $(this).text
());
});

// but if we want to pass each
// element through a function..

// .map()
$(".welcome")
.append( $("li").map(function(){
      return $(this).val();
    })
.get()
.join(", ") );

// appends comma-separated list of
li values
```

```
<div class="welcome"></div>

<ul id="conference">

    <li class="attendee">
    <a title="addy osmani"
     href="http://addyosmani.com">Addy</a>
    </li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Adam Sontag</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="text" name="attendeeName"
value="John"/>
```

“I want to be able to check if something has been ‘checked’ or modify attributes like ‘href’. Can I?”

Attributes

Getting and settings DOM attributes of elements

jQuery

- jQuery supports getting and setting DOM **attributes** and **properties** of elements
  - attributes are generally strings
  - can be modified via .attr()
  - properties can be modified via .prop()

Code


```
// .attr() gets the attribute value
// for the first element in a set only

// gets the value of an attribute
$('a').attr('href');

// sets the value of an attribute
$('a').attr('href','http://google.com');


// we can also set multiple
// attributes at the same time
$('a').attr({
    title: 'Google!',
    alt: 'Googling',
    href: 'http://google.com'
});
```

```
<div class="welcome"></div>

<ul id="conference">

    <li class="attendee">
    <a title="addy osmani"
     href="http://addyosmani.com">Addy</a>
    </li>
    <li class="attendee">Dan Heberden</li>
    <li class="attendee">Adam Sontag</li>
    <li class="attendee">Mathias Bynens</li>
    <li class="attendee">Douglas</li>

</ul>

<input type="text" name="attendee_name"
value="John"/>
```

jQuery

- Only thing that can’t be altered like this is **type**
- Some browsers throw errors when this is attempted to be changed
- (Mostly IE)

jQuery

- Similar to attributes, jQuery also allows us to get and set **properties**
  - Difference is important
  - ‘checked’ is a property
  - ‘title’ is an attribute
  - Before 1.6, attr() sometimes took properties into account causing inconsistencies
  - This is now fixed

Code


```
// Prop examples

var elem = $('input');
var node = elem[0];

//true
alert( node.checked );

//true
elem.prop('checked');

//true
elem.is(':checked');

// change prop
elem.prop('checked','');

//or
elem.removeProp('checked');
```

```
<div class="welcome"></div>

<input type='checkbox' checked/>

<a href="http://google.com"
title="hello world">test</a>
```

Summary

What did we learn today?

jQuery can be a powerful addition to your front-end utility belt

We covered:

- Core concepts
  - Setup
  - Selection
  - Chaining
  - Caching

We looked through:

- API
  - Selectors
  - Events
  - Animation
  - Traversing
  - Attributes
  - Ajax
  - and more.

You can start using it right away

Go try it out at jQuery.com!

For questions or more from me:

@addyosmani or addyosmani.com

For a free jQuery book check out:

jqfundamentals.com