



# JavaScript and jQuery for Designers

## Lesson 4: Getting Fancy

Michael Slater, CEO  
Christopher Haupt, CTO  
[course-support@webvanta.com](mailto:course-support@webvanta.com)  
888.670.6793

**[www.webvanta.com](http://www.webvanta.com)**

# Welcome!



- Four sessions
  - 1: JavaScript & jQuery Basics
  - 2: Setting Up Plugins
  - 3: Making Forms Work Well
  - **4: Getting Fancy**
- Course Portal has all the links you need
  - [www.online-web-courses.com](http://www.online-web-courses.com)

# Questions

- Ask questions via chat during the live course
- Join the discussion group
  - Link at the course portal
- Email us
  - [course-support@webvanta.com](mailto:course-support@webvanta.com)
  - *Please use the discussion group* for non-private questions



Michael Slater  
Cofounder & CEO



Christopher Haupt  
Cofounder & CTO

# Goals for This Lesson

- Learn some more things you can do with jQuery, generally without plugins
- See some more troubleshooting techniques
- Establish a foundation for going further on your own

# Topics

- Using implicit and explicit loops
- jQuery events: beyond "click" and "ready"
- Using jQuery effects: beyond hide and show
- Selectors and filters: beyond simple CSS
- Using cookies to give your pages memory
- Ajax: Using jQuery to load new content without reloading the page
- Variable scope and good programming practices
- *Note: sample code files are a key part of the content!*

# Homework Solution

- Create a form with the following behaviors:
  - Fields for name, street, city, state, and zip
  - Validate that no fields are blank, that the state field has exactly two characters, and that the zip is a five-digit number
  - Add a checkbox labeled "Provide additional contact info?"
  - If the box is checked, show fields for phone number and email, with appropriate validations
- See file homework/homework.html

# Implicit and Explicit Loops

- Acting on a CSS selector affects *everything* that matches
  - `$(".special").css("font-weight", "bold");`  
(make all the elements with the class special bold)
  - `$("#greenlist li").css("color", "green");`  
(make green all the list items that are children of an element with the id greenlist)
- You can also create explicit loops with `.each`
  - Allows you to execute a function for each item

# Using "this"

- "this" is a keyword that lets you access the DOM element on which a function has been called
  - Inside a loop
  - In an event handler
- Use `$(this)` to create a jQuery object
- See files
  - `loops-and-this/each.html`
  - `events/event-this.html`
- Note: If you have a function within a function, then you need to cache "this" in a variable outside of the inner function to be able to access it there



# Chaining

- jQuery methods return the original object, so you can chain them together
  - `$( "#myID" ).css( "color", "green" ).show( );`  
(assumes element is initially hidden; set color to green and then show the element)
- Chains can be arbitrarily long
- With effects, chain is executed sequentially
- See file `chaining/chaining.html`
- Don't chain just to set multiple css attributes
  - See file `chaining/multiple-css.html`

# Events

- Mouse events
  - click, dblclick, mousedown, mouseup, mouseover, mouseout, hover, mousemove
- Document events
  - ready, load, unload, resize, scroll
- Form events
  - submit, reset, change, focus, blur
- Keyboard events
  - keypress, keydown, keyup

# Using Events

- Event behavior varies somewhat among browsers, so test thoroughly
  - e.g., Firefox fires the resize event once when resizing is done, whereas other browsers fire the event repeatedly during resizing
- Most events have a shortcut form that is simply the name of the event
  - e.g., `$(document).resize(function-to-execute);`
- General-purpose "on" method (jQuery 1.7)
  - e.g., `$(document).on("resize", function-to-execute);`
- Older variants: bind, delegate, live

# Lots More to Events

- If you want to become a more advanced jQuery programmer, there's lots more you can do with events
  - Event delegation: set events on the parent element to effectively attach events to all its children
  - Removing event handlers
  - Work with event objects

# Effects

- The basics
  - `hide`, `show`, `toggle`
- Fading
  - `fadeIn`, `fadeOut`, `fadeTo`, `fadeToggle`
- Sliding
  - `slideUp`, `slideDown`, `slideToggle`
- Custom effects
  - `animate`

# Effect Options

- Parameters are duration, easing, callback
  - All are option and have default settings
- duration defaults to 400 ms
  - "slow" = 600 ms
  - "fast" = 200 ms
  - number = however many ms you want
- easing is how the speed varies during the effect
  - Default is "swing"
  - Other built-in option is "linear"
  - More options available with plugins
- callback allows you to specify a function to be executed when effect is completed

# Effect Chaining

- When you chain effects, they are executed in sequence
- Use the `.delay()` method if you want a pause
- With the `.animate()` method, you can specify multiple CSS properties to be animated simultaneously
- See files in the effects folder

# Selectors and Filters

- You can use any CSS selector, including descendant, child, sibling, and attribute
- jQuery filters provide even more ways to select items
  - `:even` and `:odd`
  - `:first` and `:last`
  - `:not( )`
  - `:has( )`
  - `:contains( )`
  - `:hidden` and `:visible`
- See file `filters/filters.html`



# Remembering Things

- Web pages are stateless
  - One page knows nothing about anything that came before it
- Cookies are the traditional method for storing information in the browser
- HTML5 adds session storage and web databases
- Query parameters can also be used to pass information between pages

# Using Cookies

- Cookies have several attributes
  - Name
  - Value
  - Domain name
  - Path
  - Expiration
- Session cookies are deleted when browser closed
- Cookie plugin
  - <https://github.com/carhartl/jquery-cookie>
- See files `cookies/read-cookie.html` and `cookies/write-cookie.html`
  - Note these files don't work locally

# Ajax

- Originally "asynchronous JavaScript and XML"
  - XMLHttpRequest requests data from server without loading a new page
  - In more general sense, may be no XML involved
- Simplest form:
  - Load a page fragment and inject it into the page
  - `$( "#mydiv" ).load( "fragment.html" );`
- See files `ajax/ajax.html`, `ajax/ajax-2.html`, `ajax/fragment.html`
  - Note: security model prevents this from working with local files; use a local or remote web server

# Many Ways to Manipulate DOM Content

- `$( "#myid" ).html( "<h2>Contents</h2>" );`
  - Replaces contents of element with specified HTML
- `$( "#myid" ).text( "contents" );`
  - Replaces text contents of element
- `$( "#myid" ).append( "<h2>Contents</h2>" );`
  - Adds HTML as the last child element
- `$( "#myid" ).prepend( "<h2>Contents</h2>" );`
  - Adds HTML as the first child element
- `$( "#myid" ).after( "<h2>Contents</h2>" );`
  - Adds HTML after the selected element
- `$( "#myid" ).before( "<h2>Contents</h2>" );`
  - Adds HTML before the selected element
- See file `ajax/manipulate.html`

# Variable Scope

- From where can a variable be accessed?
  - You can get into trouble if you don't understand this
- Global scope = accessible anywhere
  - The default behavior for any variable defined outside of a function
- Variables defined in a function have local scope
  - They are accessible only within that function, *if* they are defined using the "var" keyword
  - *If defined without the "var" keyword, they are global*
  - Functions inside functions have access to the containing function's variables
- See file `scope/scope.html`

# Getting In and Out of Trouble with Variable Scope

- If you are writing simple jQuery code, any variables you create are global
- Any other JavaScript on that page that uses the same variable name, if global, will conflict and likely cause misbehavior
- For safety, for code that is not in a document ready block, it's best to wrap your code in a self-calling function, which keeps your variables local

```
(function($) {  
    your code here  
})(jQuery);
```

# You've Made It!

- We hope you feel like you've come a long way in your ability to use JavaScript and jQuery
- If you just master the basic JavaScript syntax and targeting elements with jQuery, you can do a lot with jQuery plugins
- If you want to dive deeper, there's two major levels
  - More advanced jQuery functions
  - Pure JavaScript programming

# Next Steps

- Course videos and files will be available for at least one year
  - Repetition is key to learning
- Take on some simple jQuery tasks and work at them until you have them functioning
- Discussion group will remain open for your questions
- Read the book "JavaScript & jQuery: the missing manual"