**jQuery**

**Part 1**

**Introduction**

Jquery is a javascript file that you include in your web pages. It lets you find elements using CSS-style selectors and then do something with the elements using jquery methods.

$() = shorthand for ‘jquery()’

What jquery does:

1. Find elements using css-style selectors:
   * 1. E.g. $(‘li.hot’) – *this find all of the <li> elements with a class of ‘hot’.*
2. Do something with the elements using jquery methods:
   * 1. $(‘li.hot’).addClass(‘complete’);

Where to find jquery: ***http://jquery.org***

E.g. of jquery:

$(‘li’).hide().fadeIn(1500);

$(‘li’).on(‘click’, function(){

$(this).remove();

});

Jquery doesn’t do anything you can’t already achieve with pure javascript. It allows you to achieve the same goals but in fewer lines of code.

*\*\*\*Look at p302-305 for methods and how to find elements\*\*\**

**Jquery methods that get and set data**

Get info:

E.g. Var content = $(‘li’);

This will only retrieve the content of the first list item.

Set info:

E.g. $(‘li’).html(‘text’);

This will update all of the list items.

\*Note that when a var contains a jquery object, it is often given a name beginning with the ‘$’ symbol. E.g. var $listItems = $(‘li’);\*

**Part 2**

**Looping**

Looping is quicker in jquery than if using just plain javascript

E.g. $(‘li em’).addClass(‘className’);

$(‘li.hot’).addClass(‘otherClassName’);

**Chaining**

If you want to use more than one jquery method at a time, you can using dot notation.

E.g. $(‘li[id!=”one”]’).hide().delay(500).fadeIn(1400);

Most methods used to update the jquery selection can be chained. However, methods that retrieve info from the DOM cannot be chained.

*\*if one method in the chain does not work, the rest will not run either\**

**Checking a page is ready to work with**

$(document).ready(function(){

//insert code;

});

Shorthand

$(function(){

//insert code;

});

**Retrieving element content**

.html ()

This retrieves html inside the first element, along with any descendants.

Use .each() if you want to retrieve the value of every element

.text()

This returns content from every element in the jquery selection, along with text from descendants.

To get content from <input> or <textarea>, use the .val() method.

E.g.

1)

var $listHTML = $(‘ul’).html();

$(‘ul’).append($listHTML);

2)

Var $listText = $(‘ul’).text();

$(‘ul’).append(‘<p>’+$listText+’</p>’);

3)

Var $listItemHTML = $(‘li’).html();

$(‘li’).append(‘<i>’+$listItemHTML+’</i>’);

4)

Var $listItemText = $(‘li’).text();

$(‘li’).append(‘<i>’+$listItemText+’</i>’);

**Updating elements**

.html (), .text(), .replaceWith(), .remove()

Using a function to update content:

$(‘li.hot’).html(function(){

Return ‘<em>’ + $(this).text()+’</em>’;

});

**Changing content**

$(function(){

$(‘li:contains(“pine”)’).text(‘almonds’);

$(‘li.hot’).html(function(){

Return ‘<em>’+$(this).text()+’</em>’;

});

$(‘li#one’).remove();

});

**Inserting elements**

1. Create new elements in a jquery object
   1. Var $newFragment = $(‘<li>’);
   2. Var $newItem = $(‘<li class=’hot’>item</li>’);
2. Use a method to insert the content into the page
   1. .before(); - before selected element
   2. .after(); - after selected element
   3. .Prepend(); - at the start of the selected element (opening tag)
   4. .Append(); - at the end of the selected element (closing tag)

**Adding new content**

E.g. $(function(){

$(‘ul’).before(‘<p>just updated</p>’);

$(‘li.hot’).prepend(‘+ ‘);

Var $newListItem = $(‘<li><em>soy sauce</em></li>’);

$(‘li:last).after($newListItem);

});

**Part 3**

**Getting and setting attribute values**

.attr()

To get = e.g. $(‘li#one’).attr(‘id’);

To update = e.g. $(‘li#one’).attr(‘id’,’hot’);

.removeAttr()

e.g. $(‘li#one’).removeAttr(‘id’);

.addClass()

This adds a new class, it does not overwrite an existing value

.removeClass()

This removes class completely from an element

e.g. $(function(){  
 $(‘li#three’).removeClass(‘hot’);

$(‘li.hot’).addClass(‘favorite’);

$(‘ul’).attr(‘id’,’group’);

});

**Getting and setting CSS properties**

How to get a CSS property:

Var bgColor = $(‘li’).css(‘background-color’);

How to set a CSS property:

$(‘li’).css(‘background-color’,’red’);

Setting multiple properties (using literal notation):

$(‘li’).css({

‘background-color’:’red’,

‘font-family’:’sans-serif’

});

e.g. $(function(){

Var bgColor = $(‘li’).css(‘background-color’);

$(‘ul’).append(‘<p>color was: ‘ +bgColor+’</p>’);

$(‘li’).css({

‘background-color’:’red’,

‘border’:’1px solid black’,

‘color’:’white’

});

});

**Working with each element in a selection**

Jquery allows you to recreate the functionalist of a loop on a selection of elements, using the .each() method.

.each() – allows you to perform one or more statements on each of the items in the selection of elements that is returned by a selector, rather like a loop in javascript.

It takes one parameter: a function (named or anonymous), containing the statements you want to run on each element.

E.g. $(‘li’).each(function(){

Var ids = this.id;

$(this).append(‘<em>’+ids+’</em>’);

});

**Part 4**

**Event methods**

The .on() method is used to handle all events.

Behind the scenes, JQuery handles all the cross-browser issues

E.g. $(‘li’).on(‘click’,function(){  
 $(this).addClass(‘new’);

});

JQuery events

|  |  |
| --- | --- |
| UI | Focus, blur, change |
| Keyboard | Input, keydown, keyup, keypress |
| Mouse | Click, dblclick, mouseup, mousedown, mouseover, mousemove, mouseout, hover |
| Form | Submit, select, change |
| Document | Ready, load, unload |
| Browser | Error, resize, scroll |

E.g. $(function(){

Var ids = ‘’;

Var $listItems = $(‘li’);

$listItems.on(‘mouseover click’,function(){

ids = this.id;

$listItems.children(‘span’).remove();

$(this).append(‘<span>’+ids+’</span>’);

});

$listItems.on(‘mouseout’,function(){

$(this).children(‘span’).remove();

});

});

**The event object**

|  |  |
| --- | --- |
| **Property** | **Description** |
| Type | Type of event (e.g. click, mouseover…) |
| Which | Button or key that was pressed |
| Data | An object literal containing extra info passed to the function when the event fires |
| Target | DOM element that initiated the event |
| pageX | Mouse position from left viewport |
| pageY | Mouse position from top of viewport |
| timestamp | Number of milliseconds from Jan 1st 1970, to when the event was triggered |
| **Method** | **Description** |
| preventDefault() | Prevents the default |
| stopPropagation() | Stops the event bubbling up to ancestors |

E.g. $(function(){

$(‘li’).on(‘click’,function(e){

$(‘li span’).remove();

Var date = newDate();

Date.setTime(e.timeStamp);

Var clicked = date.toDateString();

$(this).append(‘<span>’+clicked+’ ’+e.type+’</span>’);

});

});

**Additional parameters for event handlers**

The .on() method has 2 optional properties that let you:

1. Filter the initial jquery selection to respond to a subset of the elements
2. Pass extra info into the event handler using object literal notation

.on(events[,selector][,data],function(e));

* Events = the event you want to respond to
* Selector = provide a second selector (optional)
* Data = you can pass extra info to the function that is called when the event is triggered
* Function = code block to run when the function is called

E.g. $(function(){

Var listItem, itemStatus, eventType;

$(‘ul’).on(

‘click mouseover’,

‘:not(#four)’,

{status:’important’},

Function(e){

listItem = Item: ‘+ e.target.textContent + ‘</br>’;

itemStatus = ‘Status: ‘ + e.data.status + ‘</br>’;

eventType = ‘Event: ‘+ e.type;

$(‘#notes’).html(listItem+itemStatus+eventType);

}

);

});

**Part 5**

**Effects**

|  |  |
| --- | --- |
| **Basic Effects** |  |
| .show() | Displays elements |
| .hide() | Hides elements |
| .toggle() | Toggles between showing and hiding elements |
| **Fading Effects** |  |
| .fadeIn() | Fades in elements |
| .fadeOut() | Fades out elements |
| .fadeTo() | Changes opacity of selected elements |
| .fadeToggle() | Hides or shows elements by changing their opacity |
| **Sliding Effects** |  |
| .slideUp() | Hides selected elements with a sliding motion |
| .slideDown() | Shows selected elements with a slide motion |
| .slideToggle() | Hides or shows with a sliding toggle |
| **Custom Effects** |  |
| .delay() | Delays execution of subsequent items in a que |
| .stop() | Stops animation if currently running |
| .animate() | Creates custom animation (p334) |

E.g. $(function(){

$(‘h2’).hide().slideDown();

Var $li = $(‘li’);

$li.hide().each(function(index){

$(this).delay(700\*index).fadeIn(700);

});

$li.on(‘click’,function(){

$(this).fadeOut(700);

});

});

**Animation CSS properties**

The .animate() method allows you to create some of your own effects and animations by changing css properties.

.animate({

//styles you want to change

} [,speed][,easing][,complete]);

**Speed** = duration of animation in milliseconds (or slow, fast)

**Easing** = linear or swing (if not specified then swing is used)

**Complete** = is used to call a function that should run when the animation is finished (this is known as callback function).

You can animate any CSS property whose value can be represented as a number (not a string!)

The CSS property is written using camelCase (e.g. font-size is fontSize)

The CSS properties are specified using object literal notation

E.g. $(function(){

$(‘li’).on(‘click’,function(){

$(this).animate({

Opacity:0.0,

paddingLeft:’+=80’

}, 500, function(){

$(this).remove();

});

});

});

**Part 6**

**Traversing the DOM**

When you have a jquery selection, you can use these methods to access other element nodes relative to the initial selection.

|  |  |
| --- | --- |
| **Selector required** |  |
| .find() | All elements within current selection that match selector |
| .closest() | Nearest ancestor (not just parent), that matches selector |
| **Selector optional** |  |
| .parent() | Direct parent |
| .parents() | All parents |
| .children() | All children |
| .siblings() | All siblings |
| .next() | Next sibling |
| .nextAll() | All subsequent siblings |
| .prev() | Previous sibling |
| .prevAll() | All previous siblings |

E.g. $(function(){

Var $h2 = $(‘h2’);

$(‘ul’).hide();

$h2.append(‘<a class=”show”>Show</a>’);

$h2.on(‘click’,function()[

$h2.next().fadeIn(500).children(‘.hot’).addClass(‘complete’);

$h2.find(‘a’).fadeOut();

});

});

**Add and filter elements in a selection**

Once you have a jquery selection, you can add more elements to it, or you can filter the selection to work with a subset of elements

Adding element to a selection:

.add() = adds new content to the items in the existing selection and places the resulting content in a new jquery object.

Filtering with a second selector:

.filter() = finds elements in the matched set that in turn match a second selector

.find() = finds descendants of elements in the matched set that match a selector

.not()/:not() = finds elements that do not match the selector

.has()/:has() = finds elements from the matched set that have a descendant that matches the selector

:contains() = selects all elements that contain the text specified

Testing content:

.is() = checks whether current selection matches a condition (returns Boolean)

E.g. $(function(){

Var $listItems = $(‘li’);

$listItems.filter(‘.hot:last’).removeClass(‘hot’);

$(‘li:not(.hot)’).addClass(‘cool’);

$listItems.has(‘em’).addClass(‘complete’);

$listItems.each(function(){

Var $this = $(this);

If ($this.is(‘.hot’)){

$this.prepend(‘priority item: ‘);

}

});

$(‘li:contains(“honey”)’).append(‘ (local)’);

**Part 7**

**Finding items by order**

Each item returned by a jquery selector is given an index number, which can be used to filter the selection.

.eq() – the element that matches the index number

.lt() – elements with an index number less than the number specified

.gt() – elements with an index number greater than the number specified

E.g. $(function(){

$(‘li:lt(2)’).removeClass(‘hot’);

$(‘li’).eq(0).addClass(‘complete’);

$(‘li:gt(2)’).addClass(‘cool’);

});

**Selecting form elements**

Revise pages 342-343 for selectors and methods for forms

E.g. $(function(){

Var $newItemButton = $(‘#newItemButton’);

Var $newItemForm = $(‘#newItemForm’);

Var $textInput = $(‘input:text’);

$newItemButton.show();

$newItemForm.hide();

$(‘#showForm’).on(‘click’,function(){

$newItemButton.hide();

$newItemForm.show();

});

$newItemForm.on(‘submit’,function(e){

e.preventDefault();

var newText = $textInput.val();

$(‘li:last’).after(‘<li>’ + text + ‘</li>’);

$newItemForm.hide();

$newItemButton.show();

$textInput.val(‘ ’);

});

});

**Cutting and copying elements**

Once you have a jquery selection, you can use these methods to remove those elements or make a copy of them:

|  |  |
| --- | --- |
| **Cut** |  |
| .remove() | Removes elements and descendants |
| .detach() | Same as remove(), but keeps a copy of them in memory |
| .empty() | Removes child nodes and descendants from any elements in a matched set |
| .unwrap() | Removes parents of matched set, leaving matched elements |
| **Copy** |  |
| .clone() | Creates a copy of the matched set, including any descedants |

E.g. $(function(){

Var $p = $(‘p’);

Var $clonedQuotes = $p.clone();

$p.remove();

$clonedQuotes.insertAfter(‘ul’);

Var $moveItem = $(‘#one’).detach();

$moveItem.appendTo(‘ul’);

});

**Box dimensions**

These methods all you to discover or update the width and height of all boxes on the page:

|  |  |
| --- | --- |
| **Retrieve or set box dimensions** |  |
| .height() | Height of box (no margin, border, padding) |
| .width() | Width of box (no margin, border, padding) |
| **Retrieve box dimensions only** |  |
| .innerheight() | Height of box plus padding |
| .innerwidth() | Width of box plus padding |
| .outerHeight() | Height of box plus padding and border |
| .outerWidth() | Width of box plus padding and border |
| .outerHeight(true) | Height of box plus padding, border and margin |
| .outerWidth(true) | Width of box plus padding, border and margin |

E.g. $(function(){

Var listHeight = $(‘#page’).height();

$(‘ul’).append(‘<p>height: ‘ + listHeight + ‘px</p>’);

$(‘li’).width(‘50%’);

$(‘li#one’).width(125);

$(‘li#two).width(‘75%’);

});

**Window and page dimensions**

The .height() and .width() methods can be used to determine the dimensions of both the browser and the HTML document.

There are also methods to get and set the position of the scroll bar.

.height() - height of jquery selection

.width() - width of jquery selection

.scrollLeft() - gets the horizontal position of the scrollbar for the first element in the jquery selection, or sets the horizontal scroll bar position for matched nodes

.scrollTop() - gets the vertical position of the scrollbar for the first element in the jquery , or sets the vertical scroll bar position for matched nodes

E.g. $(window).height(); E.g. $(document).height();

**Position of elements on the page**

The .offset() and .position() methods can be used to determine the position of elements on the page.

.offset() - gets or sets coordinates of the element relative to the top left hand corner of the document object

.position() - gets or sets coordinates of the element relative to any ancestor that has been taken out of normal flow (using CSS box offsets). If no ancestor is out of normal flow, it will return the same as .offset().

E.g. var offset = $(‘ul’).offset();

Var text = ‘left: ‘ + offset.left + ‘top: ‘ + offset.top;

Alert (text);

**Determining position of items on the page (p352-p353)**

\*\*note that I can not get this example code to work on my PC? Could no longer be available\*\*

In the following example, as the user scrolls down the page, a box slides into view if they get within 500px of the footer. This part of the page is called the endZone and you need to workout the height at which the endZone starts.

$(function(){

Var $window = $(window);

Var $slideAd = $(‘#slideAd’);

Var endZone = $(‘#footer’).offset().top - $window.height() – 500;

$window.on(‘scroll’,function(){

If ( (endZone) < $window.scrollTop() ){

$slideAd.animate({‘right’:’0px’},250);

}else{

$slideAd.stop(true).animate({‘right’:’-360px’},250);

}

});

});

**Part 8**

**Ways to include jquery in your page**

In addition to hosting the jquery file with the rest of your website, you can also use a version that is hosted by other companies. However, you should still include a fallback version.

A content delivery network (CDN) is a series of servers spread around the world designed to serve static files (javascript, HTML, CSS, images, videos) very quickly. The CDN tried to find a server near you, then sends files from that server so the data does not travel as far.

When including jquery in your pages, you can try to load it from one of these CDNs. Then you can check if it loaded, and if not, you can include a version that is stored on your own servers (this is known as a fallback)

**Loading Jquery from a CDN**

Use a logical operator ( || ) to include your fallback in your HTML

E.g. <script src=”//ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js”></script>

<script>

Window.jquery || document.write(‘<script src=”js/jquery-1.10.2.js”></script>’)

</script>

**Where to place your script**

The position of <script> elements can affect how quickly a web page seems to load. Pages are loaded quicker if used before the closing </body> tag as opposed to in the <head> tag.

Where possible, consider using alternatives to scripts. E.g. use CSS for animations or HTML5 autofocus attribute rather than using the loadevent to bring focus to an element.

**Jquery documentation**

This booklet does not teach you everything about jquery, but it does list many of the most popular features, and you should know enough about jquery to understand how it works and how to make use of it in your scripts.

For an exhaustive list of the functionality provided in jquery, visit <http://api.jquery.com>

**Extending jquery with plugins**

Plugins are scripts that extend the functionality of the jQuery library. Hundreds have been written and are available to use.

Plugins offer functionality that is not included in the jQuery library. They usually deal with a particular task such as creating slideshows or video players, performing animations, transforming data, enhancing forms, and displaying new data from a remote server.

To get an idea of the number and range of plugins available, see <http://plugins.jquery.com>

**Javascript libraries**

Jquery is an example of what programmers call a javascript library. It is a javascript file that you include in your page, which then lets you use the functions, objects, methods and properties it contains.

The concept of a library is that is allows you to borrow code from one file and use its functions, objects, methods and properties in another script.

**Jquery example**

See page 363-365 for an example which uses many techniques used in jquery.

**Test**

Part 1

1. What is jquery and what does it do?
2. Change the class of an element
3. Revise p302-p305
4. How would you cach a jquery selection?
5. Create script so that certain elements fade in when the page is loaded

Part 2

1. Add a class to all list items
2. What is meant by chaining
3. Hide all list items with .hot class and delay before fading in
4. What are the differences between .html() and .text()?
5. Make all list items italic and replace one list item with a new food. Also remove list item when clicked
6. Add a paragraph before the list and a new list item at the end of the list

Part 3

1. List the 4 jquery methods used to get and set attributes
2. Practical: remove a class from an element, and add a different class to trigger new css code
3. How do you get and set css properties?
4. How do you set multiple css properties using literal notation? And give example
5. Use the .each() method to loop through a list and type the id of that list element at the of each list item

Part 4

1. Create some code that shows the ID name next to a list item when hovered over, and disappears when the cursor is moved away
2. Create code that shows date clicked after element
3. Crease some code using ‘function(e)’ and literal notation to show status (not containing(#four)), event triggered and element clicked

Part 5

* 1. Name 3 basic effects
  2. Name 4 fading effects
  3. Name 3 sliding effects
  4. Name 3 custom effects

1. Use the above to:
   * 1. Fade heading in
     2. Fade list item out when clicked
     3. Fade each list item in individually
2. What is the .animate() methods and what does it do?
3. Use the .animate() method to make list items move when clicked

Part 6

1. List the method that are used to traverse the DOM, that require a selector
2. List other methods used to traverse the DOM
3. Traverse the DOM so:
   * 1. After <h2> add a <a>show</a>
     2. When <a> clicked, fade list items in
     3. Fade <a> out
4. List and define the methods used to add and filter elements in a selection
5. Practical:
   * 1. Remove hot class from the last .hot item
     2. Add a ‘cool’ class to all, but .hot items
     3. Add a ‘complete’ class to list items with italics
     4. Make all list items with italics have the text ‘priority item:’ before it
     5. Add the text ‘ (local)’ after list items that contain the word “honey”

Part 7

1. List the 3 methods used to find items in a specific order, and give an example using <ul> element
2. Form elements – create code which:
   * 1. Hides form div
     2. Shows form div when ‘new item’ button clicked
     3. Adds specific item to list when ‘add’ button is clicked
     4. Hides form div when ‘add’ button is clicked, and show original button (‘new item’)
     5. Reverts form text value back to ‘ ‘;
3. List 2 methods used to cut elements and 1 method used to copy elements
4. Create code which:
   * 1. Copies <p> element and adds under <ul>
     2. Moves top list item to the bottom of the list
5. List 2 methods used to retrieve or set box dimensions
6. List 6 methods used to retrieve box dimensions only