

# JQuery



## □ **Events**

- ▣ Browsers are preprogrammed to recognize certain actions such as clicking, page loading, mouse movements etc.
- ▣ You write programs to respond to these events

## □ **Two Step process**

- ▣ Identify the element that you want to respond to the event
- ▣ Assign an event and create a function to run when event occurs

# JQuery



- **Mouse Events**

- ▣ Click, dblclick, mousedown, mouseup, mouseover, mouseout, mousemove

- **Document/Window Events**

- ▣ Load, resize

- **Form Events**

- ▣ Submit, reset, change, focus, blur

- **Keyboard Events**

- ▣ Keypress, keydown

# JQuery

## □ Techniques for using Events

### ▣ Inline Events

`<a href="somepage.html" onmouseover = "alert('Some Message');" >Link</a>`

- Add the word "on" to the event
- Add directly inside of HTML
- Downside: JS is now scattered throughout your HTML

### ▣ Traditional Technique

- Assign the event to an element but stay out of the HTML
- Within the `<script>` tags in head or body add `window.onload=message;`
  - ▣ Here a function called message is called after the page loads
  - ▣ function is assigned but not run immediately

# JQuery

## □ Techniques for using Events

- Previous techniques only allow 1 function assigned to 1 event per element (tag)
- Introduce W3C Event Listeners
  - Firefox, Safari and Opera handle differently than IE

## □ JQuery Way

- Select element
  - `$ ("a")`
- Assign an event
  - `$ ("a").mouseover ()`
- Pass function to event or use anonymous function
  - `$ ("a").mouseover (MyFunction)`

# JQuery

- Example of mouseover event that shows a submenu when menu selected:

```
$ ("#menu") .mouseover (function () {  
    $ ("#submenu") .show () ;  
}) ;
```

- Uses anonymous function

# JQuery

- Stopping a normal event action
    - ▣ Example: when a link is clicked the URL is followed
      - To stop that action:
        - The action is part of the event object
        - We can reference the event object and call `.preventDefault()`;
- ```
$("#menu").click(function(evt) {  
    //Some JavaScript code here  
    Evt.preventDefault();  
});
```

# JQuery

## □ Forms

### ▣ Form example

#### ■ Selecting Form Elements

#### ■ Assign an ID to it

```
<input name="user" type="text" id="user">  
var userfield = $("#user");
```

#### ■ Manipulating Selected form element

##### ■ val will get the value of the element:

```
var userfield = $('#user').val();
```

## □ Selecting all form elements of certain type:

### ▣ \$(" :text")

#### ■ Selects all text fields

### ▣ Use with :input ( all form elements), :password, :radio, :checkbox, :submit, :image, :reset, :button, :file, :hidden

### ▣ Can use descendant selectors too \$(" #signup : text")

# JQuery

- Set the value of a form element

```
var fieldvalue = $("#total").val(Yourvalue);
```

- Determining if box is checked

```
if ($("#total").attr("checked")) {  
    Do stuff if box is checked  
}  
else {  
    Do stuff if box is not checked  
}
```



# JQuery

## □ Form Events

### ▣ Submit

```
$(document).ready(function() {  
    $("#signup").submit(function() {  
        if ($("#username").val() == "")  
        {  
            alert ("Name is required");  
            return false;  
        }  
    })  
});
```

# JQuery

## □ Focus

- ▣ Example: Auto erases default text in a field when it gets the focus

`<input name="username" type="text" id="username" value="Type your user name">`

```
$("#username").focus(function() {  
    var field = $(this);  
  
    if(field.val()==field.attr("defaultValue"))  
    {  
        field.val("");  
    }  
});
```

# JQuery

## □ Blur

```
<input name="quantity" type="text" id="qty">
```

```
$("#qty").blur(function) {  
    var fieldValue = $(this).val();  
  
    if (isNaN(fieldValue))  
    {  
        alert("Please enter a number");  
    }  
});
```

# jQuery



## □ Click

- ▣ If any radio button is clicked

```
$ (":radio") .click (function () {  
    do stuff  
}) ;
```

- Can add focus to the first element of the form:

```
$ ( 'username' ) .focus;
```

# What is jQuery?



- JavaScript Library
- Functionality
  - ▣ DOM scripting & event handling
  - ▣ Ajax
  - ▣ User interface effects
  - ▣ Form validation

# Why jQuery?



- Lightweight – 14kb (Minified and Gzipped)
- Cross-browser support (IE 6.0+, FF 1.5+, Safari 2.0+, Opera 9.0+)
- CSS-like syntax – easy for developers/non-developers to understand
- Active developer community
- Extensible - plugins

# JQuery



- Powerful JavaScript library
  - ▣ Simplify common JavaScript tasks
  - ▣ Access parts of a page
    - using CSS or XPath-like expressions
  - ▣ Modify the appearance of a page
  - ▣ Alter the content of a page
  - ▣ Change the user's interaction with a page
  - ▣ Add animation to a page
  - ▣ Provide AJAX support
  - ▣ Abstract away browser quirks

# Example – Show/Hide the old way

`<a href="#" onclick="toggle_visibility('foo');">Click here to toggle visibility of #foo</a>`

```
function toggle_visibility(id) {  
    var e = document.getElementById(id);  
  
    if(e.style.display == 'block')  
        e.style.display = 'none';  
    else  
        e.style.display = 'block';  
}
```



# Example – Show/Hide with jQuery

---

```
$().ready(function() {  
    $("a").click(function() {  
        $("#more").toggle("slow");  
        return false;  
    });  
});
```

# Introductory Sample

```
<html>
<body>
<h1>Cities of the World</h1>
<dl>
<dt>Paris</dt>
  <dd>Chic, fashionable, expensive rude</dd>
<dt>Sydney</dt>
  <dd>Opera house but no culture, Mardi Gras,
    fireworks</dd>
</dl>
</body>
</html>
```

```
h1 {font-size: 2.5em;
    margin-bottom: 0;}

.emphasize {font-style:
            italic; color:red;}
```

## Cities of the World

Paris

Chic, fashionable, expensive rude

Sydney

Opera house but no culture, Mardi Gras, fireworks

# Basic JQuery



- Selecting part of document is fundamental operation
- A JQuery object is a wrapper for a selected group of DOM nodes
- `$()` function is a factory method that creates JQuery objects
- `$("dt")` is a JQuery object containing all the "dt" elements in the document

# Basic JQuery

- .addClass() method changes the DOM nodes by adding a 'class' attribute
  - ▣ The 'class' attribute is a special CSS construct that provides a visual architecture independent of the element structures
- \$("dt").addClass("emphasize") will change all occurrences of <dt> to <dt class="emphasize">
- See also .removeClass()

# Basic JQuery

- To make this change, put it in a function and call it when the document has been loaded and the DOM is created

```
function doEmph() { $("dt").addClass("emphasize") }  
<body onLoad="doEmph()">
```

- We had to alter the HTML (bad)
- Structure and appearance should be separated!
- Also, onLoad waits until all images *etc* are loaded. Tedious.

# Basic JQuery

- JQuery provides an independent scheduling point after DOM is created and before images are loaded
  - ▣ `$(document).ready(doEmph);`
- No HTML mods required. All done in script.
- Better solution:

```
$(document).ready(function(){  
    $("dt").addClass("emphasize")  
});
```

```
<html><head>  
<script src="jquery.js" type="text/javascript"></script>  
<script src="test.js" type="text/javascript"></script>  
...
```

## jQuery Selectors

Selector	Example	Selects
<u>*</u>	\$("#*")	All elements
<u>#id</u>	\$("#lastname")	The element with id="lastname"
<u>.class</u>	\$(".intro")	All elements with class="intro"
<u>.class.class</u>	\$(".intro,demo")	All elements with the class "intro" or "demo"
<u>element</u>	\$("p")	All <p> elements
<u>e1,e2,e3</u>	\$("h1,div,p")	All <h1>, <div> and <p> elements
<u>:first</u>	\$("#p:first")	The first <p> element
<u>:last</u>	\$("#p:last")	The last <p> element
<u>:even</u>	\$("#tr:even")	All even <tr> elements
<u>:odd</u>	\$("#tr:odd")	All odd <tr> elements
<u>:first-child</u>	\$("#p:first-child")	All <p> elements that are the first child of their parent
<u>:first-of-type</u>	\$("#p:first-of-type")	All <p> elements that are the first <p> element of their parent
<u>:last-child</u>	\$("#p:last-child")	All <p> elements that are the last child of their parent
<u>:last-of-type</u>	\$("#p:last-of-type")	All <p> elements that are the last <p> element of their parent
<u>:nth-child(n)</u>	\$("#p:nth-child(2)")	All <p> elements that are the 2nd child of their parent
<u>:nth-last-child(n)</u>	\$("#p:nth-last-child(2)")	All <p> elements that are the 2nd child of their parent, counting from the last child
<u>:nth-of-type(n)</u>	\$("#p:nth-of-type(2)")	All <p> elements that are the 2nd <p> element of their parent
<u>:nth-last-of-type(n)</u>	\$("#p:nth-last-of-type(2)")	All <p> elements that are the 2nd <p> element of their parent, counting from the last child

<u>:only-child</u>	\$("p:only-child")	All <p> elements that are the only child of their parent
<u>:only-of-type</u>	\$("p:only-of-type")	All <p> elements that are the only child, of its type, of their parent
<u>parent &gt; child</u>	\$("div > p")	All <p> elements that are a direct child of a <div> element
<u>parent descendant</u>	\$("div p")	All <p> elements that are descendants of a <div> element
<u>element + next</u>	\$("div + p")	The <p> element that are next to each <div> elements
<u>element ~ siblings</u>	\$("div ~ p")	All <p> elements that are siblings of a <div> element
<u>:eq(<i>index</i>)</u>	\$("ul li:eq(3)")	The fourth element in a list (index starts at 0)
<u>:gt(<i>no</i>)</u>	\$("ul li:gt(3)")	List elements with an index greater than 3
<u>:lt(<i>no</i>)</u>	\$("ul li:lt(3)")	List elements with an index less than 3
<u>:not(<i>selector</i>)</u>	\$("input:not(:empty)")	All input elements that are not empty
<u>:header</u>	\$("":header")	All header elements <h1>, <h2> ...
<u>:animated</u>	\$("":animated")	All animated elements
<u>:focus</u>	\$("":focus")	The element that currently has focus
<u>:contains(<i>text</i>)</u>	\$("":contains('Hello'))	All elements which contains the text "Hello"
<u>:has(<i>selector</i>)</u>	\$("div:has(p)")	All <div> elements that have a <p> element
<u>:empty</u>	\$("":empty")	All elements that are empty
<u>:parent</u>	\$("":parent")	All elements that are a parent of another element
<u>:hidden</u>	\$("p:hidden")	All hidden <p> elements
<u>:visible</u>	\$("table:visible")	All visible tables
<u>:root</u>	\$("":root")	The document's root element
<u>:lang(<i>language</i>)</u>	\$("p:lang(de)")	All <p> elements with a lang attribute value starting with "de"



<u>[attribute]</u>	\$( "[href]" )	All elements with a href attribute
<u>[attribute=value]</u>	\$( "[href='default.htm']" )	All elements with a href attribute value equal to "default.htm"
<u>[attribute!=value]</u>	\$( "[href!='default.htm']" )	All elements with a href attribute value not equal to "default.htm"
<u>[attribute\$=value]</u>	\$( "[href\$='.jpg']" )	All elements with a href attribute value ending with ".jpg"
<u>[attribute =value]</u>	\$( "[title ='Tomorrow']" )	All elements with a title attribute value equal to 'Tomorrow', or starting with 'Tomorrow' followed by a hyphen
<u>[attribute^=value]</u>	\$( "[title^='Tom']" )	All elements with a title attribute value starting with "Tom"
<u>[attribute~=value]</u>	\$( "[title~='hello']" )	All elements with a title attribute value containing the specific word "hello"
<u>[attribute*=value]</u>	\$( "[title*='hello']" )	All elements with a title attribute value containing the word "hello"
<u>:input</u>	\$( ":input" )	All input elements
<u>:text</u>	\$( ":text" )	All input elements with type="text"
<u>:password</u>	\$( ":password" )	All input elements with type="password"
<u>:radio</u>	\$( ":radio" )	All input elements with type="radio"
<u>:checkbox</u>	\$( ":checkbox" )	All input elements with type="checkbox"
<u>:submit</u>	\$( ":submit" )	All input elements with type="submit"
<u>:reset</u>	\$( ":reset" )	All input elements with type="reset"
<u>:button</u>	\$( ":button" )	All input elements with type="button"
<u>:image</u>	\$( ":image" )	All input elements with type="image"
<u>:file</u>	\$( ":file" )	All input elements with type="file"
<u>:enabled</u>	\$( ":enabled" )	All enabled input elements
<u>:disabled</u>	\$( ":disabled" )	All disabled input elements
<u>:selected</u>	\$( ":selected" )	All selected input elements
<u>:checked</u>	\$( ":checked" )	All checked input elements

# Example

- JQuery uses chaining as follows

```
$ ( 'a:contains ("ECS") ' ) .parent () .addClass ("blue") ;
```

# JQuery Events

- bind(eventname, function) method
  - ▣ 'click'
  - ▣ 'change'
  - ▣ 'resize'

```
$ ("a[@href]").bind("click", function () {  
    $(this).addClass('red');  
});
```

# Other JQuery Effects

- `.css("property", "value")`
- `.css({"prop1":"value1", "prop2":"value2"...})`
- E.g.
  - ▣ `.css('color', 'red')`
- `.hide(speed)` or `.show(speed)`
  - ▣ Where speed is `'slow'`, `'normal'` or `'fast'`

# More JQuery Changes DOM

- `.attr({ "name", "value" })`
  - ▣ sets a new attribute (or many)
- `$("<i>hello</i>")`
  - ▣ Creates a new element
- `$("<i>hello</i>").insertAfter("div.chapter p");`
  - ▣ Creates element and inserts it into the document
- `.html()` or `.text()` or `.empty()`
  - ▣ will replace matched elements with newly created elements

# Example – Show/Hide the old way

```
<a href="#" onclick="toggle_visibility('foo');">Click here to toggle visibility of #foo</a>
```

```
function toggle_visibility(id) {  
    var e = document.getElementById(id);  
  
    if(e.style.display == 'block')  
        e.style.display = 'none';  
    else  
        e.style.display = 'block';  
}
```

# Example – Show/Hide with jQuery

```
$().ready(function() {  
    $("a").click(function() {  
        $("#more").toggle("slow");  
        return false;  
    });  
});
```

### Example – Ajax the old way

```
function GetXmlHttpRequest(handler) {  
    var objXmlHttp = null; //Holds the local xmlHTTP object instance  
  
    //Depending on the browser, try to create the xmlHttp object  
    if (is_ie){  
        var strObjName = (is_ie5) ? 'Microsoft.XMLHTTP' : 'Msxml2.XMLHTTP';  
        try{  
            objXmlHttp = new ActiveXObject(strObjName);  
            objXmlHttp.onreadystatechange = handler;  
        }  
        catch(e){  
            //Object creation errored  
            alert('Verify that activscripting and activeX controls are enabled');  
            return;  
        }  
    }  
    else{  
        // Mozilla | Netscape | Safari  
        objXmlHttp = new XMLHttpRequest();  
        objXmlHttp.onload = handler;  
        objXmlHttp.onerror = handler;  
    }  
    //Return the instantiated object  
    return objXmlHttp;  
}
```



# Example – Ajax with jQuery



```
$.get("serverscript.php", { name: "John", time: "2pm" }, function(data){  
    alert("Data Loaded: " + data);  
});
```

```
$.post("serverscript.php", { name: "John", time: "2pm" }, function(data){  
    alert("Data Loaded: " + data);  
});
```

# Example – Form Validation

```
$.ready(function()
{
    // validate the comment form when it is submitted
    $("#commentForm").validate();
});

<input id="cname" name="name" class="some other styles
    {required:true,minLength:2}" />
<input id="cemail" name="email" class="{required:true,email:true}" />
```