HTML5 and CSS3 for Mobile Web Applications

Prof. Paul Krause, University of Surrey
Lecture 10
Working with jQuery

Objectives of this lecture

- Use of jQuery's document ready handler
- Selecting document elements
- Creating new document elements
- Putting it all together with an example

Using jQuery to add dynamic functionality

Date	Singer	Title
1960	Adam Faith	Poor Me
1970	Lee Marvin	Wandrin Star
1970	Norman Greenbaum	Spirit in the Sky
1980	The Pretenders	Brass in Pocket
1982	Jam	A Town Called Malice
1990	Elton John	Sacrifice
1991	Queen	Innuendo
2001	Bob the Builder	Dirrty

- Dynamically updating the appearance of an html document is simple:
  - $("table tr:nth-child(odd)").addClass("striped");

Using jQuery to add dynamic functionality

Date	Singer	Title
1960	Adam Faith	Poor Me
1970	Lee Marvin	Wandrin Star
1970	Norman Greenbaum	Spirit in the Sky
1980	The Pretenders	Brass in Pocket
1982	Jam	A Town Called Malice
1990	Elton John	Sacrifice
1991	Queen	Innuendo
2001	Bob the Builder	Dirrty

Date	Singer	Title
1960	Adam Faith	Poor Me
1970	Lee Marvin	Wandrin Star
1970	Norman Greenbaum	Spirit in the Sky
1980	The Pretenders	Brass in Pocket
1982	Jam	A Town Called Malice
1990	Elton John	Sacrifice
1991	Queen	Innuendo
2001	Bob the Builder	Dirrty

- Dynamically updating the appearance of an html document is simple:
  - $("table tr:nth-child(odd)").addClass("striped");

Using jQuery to add dynamic functionality

Date	Singer	Title
1960	Adam Faith	Poor Me
1970	Lee Marvin	Wandrin Star
1970	Norman Greenbaum	Spirit in the Sky
1980	The Pretenders	Brass in Pocket
1982	Jam	A Town Called Malice
1990	Elton John	Sacrifice
1991	Queen	Innuendo
2001	Bob the Builder	Dirrty

Date	Singer	Title
1960	Adam Faith	Poor Me
1970	Lee Marvin	Wandrin Star
1970	Norman Greenbaum	Spirit in the Sky
1980	The Pretenders	Brass in Pocket
1982	Jam	A Town Called Malice
1990	Elton John	Sacrifice
1991	Queen	Innuendo
2001	Bob the Builder	Dirrty

- Dynamically updating the appearance of an html document is simple:
  - $("table tr:nth-child(odd)").addClass("striped");

This jQuery selector collects all the odd rows of the table

Using jQuery to add dynamic functionality

Date	Singer	Title
1960	Adam Faith	Poor Me
1970	Lee Marvin	Wandrin Star
1970	Norman Greenbaum	Spirit in the Sky
1980	The Pretenders	Brass in Pocket
1982	Jam	A Town Called Malice
1990	Elton John	Sacrifice
1991	Queen	Innuendo
2001	Bob the Builder	Dirrty

Date	Singer	Title
1960	Adam Faith	Poor Me
1970	Lee Marvin	Wandrin Star
1970	Norman Greenbaum	Spirit in the Sky
1980	The Pretenders	Brass in Pocket
1982	Jam	A Town Called Malice
1990	Elton John	Sacrifice
1991	Queen	Innuendo
2001	Bob the Builder	Dirrty

- Dynamically updating the appearance of an html document is simple:
  - $("table tr:nth-child(odd)").addClass("striped");

This jQuery selector collects all the odd rows of the table

jQuery then iterates through the collected elements and adds a class to them

Using jQuery to make JavaScript “Unobtrusive”

- You should all be clear on the importance of keeping style information out of the <body> of an html document
    - Why?
- It is perhaps even more important to keep *behaviour* out of the <body> of an html document.

Let’s start with some static content

```
<body>
    <h1 id="myTitle">Some jQuery examples</h1>
    <p id="myParentDiv">The fun starts here:</p>
    <p>And ends here!</p>
</body>
```

Now add event handlers when the page loads

- We could try something like this:

```
<script>
    window.onload = function() {
    // Code to add event handlers goes in here
    };
</script>
```

- We need to have all elements created before we can add event handlers

- But, this will also wait until all images and external elements have been loaded

jQuery's document ready handler

- jQuery provides a simple cross-browser compatible method of executing code as soon as the DOM tree has been parsed:

```
$(document).ready(function() {
   $("table tr:nth-child(odd)").addClass("striped");
});
```

- This wraps the document instance in the jQuery function ($( )), applies the ready method, passing in the function to be called when the document is ready to be manipulated

- Actually, we can write this more succinctly:

```
$(function() {
   $("table tr:nth-child(odd)").addClass("striped");
});
```

Back to our static html

```
<body>
    <h1 id="myTitle">Some jQuery examples</h1>
    <p id="myParentDiv">The fun starts here:</p>
    <p>And ends here!</p>
</body>
```

Now let’s do something with it

```
<script src="jquery-1.8.2.min.js"></script>
<script>
   $(function(){
     $('#myTitle')
     .bind('mouseover', function(event){
        $("<p class='bear'>I have a bear!</p><p>I don't</p>")
        .filter(".bear").click(function(){
                        alert("I'm a bear!");
                    }).end().appendTo("#myParentDiv");
        })
   })
</script>
```

Bind a mouseover event handler to myTitle

```
<script src="jquery-1.8.2.min.js"></script>
<script>
   $(function(){
     $('#myTitle')
     .bind('mouseover', function(event){
        $("<p class='bear'>I have a bear!</p><p>I don't</p>")
        .filter(".bear").click(function(){
                        alert("I'm a bear!");
                    }).end().appendTo("#myParentDiv");
        })
   })
</script>
```

Add some paragraphs

```
<script src="jquery-1.8.2.min.js"></script>
<script>
   $(function(){
     $('#myTitle')
     .bind('mouseover', function(event){
        $("<p class='bear'>I have a bear!</p><p>I don't</p>")
        .filter(".bear").click(function(){
                        alert("I'm a bear!");
                    }).end().appendTo("#myParentDiv");
        })
   })
</script>
```

bind an event handler to just one new paragraph

```
<script src="jquery-1.8.2.min.js"></script>
<script>
   $(function(){
     $('#myTitle')
     .bind('mouseover', function(event){
        $("<p class='bear'>I have a bear!</p><p>I don't</p>")
        .filter(".bear").click(function(){
                        alert("I'm a bear!");
                    }).end().appendTo("#myParentDiv");
        })
   })
</script>
```

bind an event handler to just one new paragraph

```
<script src="jquery-1.8.2.min.js"></script>
<script>
   $(function(){
     $('#myTitle')
     .bind('mouseover', function(event){
        $("<p class='bear'>I have a bear!</p><p>I don't</p>")
        .filter(".bear").click(function(){
                        alert("I'm a bear!");
                    }).end().appendTo("#myParentDiv");
        })
   })
</script>
```

This goes in the <head> of our document!

Let’s look at the $( ) function in more detail

jQuery uses CSS3 selectors

- You should be familiar with CSS selectors:
  - e.g. p a
  - Refers to the group of all links that are nested inside a paragraph element
- In jQuery, $(selector) will return the group of elements defined by selector
  - e.g. $("p a") will return the group of all links ...
  - selector may be any valid CSS3 selector, but be aware that not all of them are supported by all browsers

Some example selectors

Some example selectors

- $(“a”)

Some example selectors

- $(“a”)
- $(“#specialID”)

Some example selectors

- $(“a”)
- $(“#specialID”)
- $(“.specialClass”)

Some example selectors

- $(“a”)
- $(“#specialID”)
- $(“.specialClass”)
- $(“#specialID.specialClass”)

Some example selectors

- $(“a”)
- $(“#specialID”)
- $(“.specialClass”)
- $(“#specialID.specialClass”)
- $(“p a.specialClass”)

Attribute selectors

- use selector[attribute] to fetch the collection of elements defined by selector with a specific attribute
  - e.g. $(“input[type=text]”)
  - will return all text input elements in the document
- Some more examples ---->>>

Let’s pick some elements out of this fragment

```
 <li><a href="http://jquery.com">jQuery supports</a>
    <ul>
     <li><a href="css1.pdf">CSS1</a></li>
     <li><a href="css2">CSS2</a></li>
     <li><a href="css3">CSS3</a></li>
     <li>Basic XPath</li>
    </ul>
</li>
```

$(“a[href^=http://]”)

```
<li><a href="http://jquery.com">jQuery supports</a>
    <ul>
     <li><a href="css1.pdf">CSS1</a></li>
     <li><a href="css2">CSS2</a></li>
     <li><a href="css3">CSS3</a></li>
     <li>Basic XPath</li>
    </ul>
</li>
```

$(“a[href$=.pdf]”)

```
 <li><a href="http://jquery.com">jQuery supports</a>
    <ul>
     <li><a href="css1.pdf">CSS1</a></li>
     <li><a href="css2">CSS2</a></li>
     <li><a href="css3">CSS3</a></li>
     <li>Basic XPath</li>
    </ul>
</li>
```

$(“a[href*=jquery.com]”)

```
<li><a href="http://jquery.com">jQuery supports</a>
    <ul>
     <li><a href="css1.pdf">CSS1</a></li>
     <li><a href="css2">CSS2</a></li>
     <li><a href="css3">CSS3</a></li>
     <li>Basic XPath</li>
    </ul>
</li>
```

$(“li:has(a)”])

```
<ul>
 <li><a href="css1.pdf">CSS1</a></li>
 <li><a href="css2">CSS2</a></li>
 <li><a href="css3">CSS3</a></li>
 <li>Basic XPath</li>
</ul>
```

Compare with $(“li a”])

```
<ul>
 <li><a href="css1.pdf">CSS1</a></li>
 <li><a href="css2">CSS2</a></li>
 <li><a href="css3">CSS3</a></li>
 <li>Basic XPath</li>
</ul>
```

Generating new html

Very simple

- $(“ Hello New World ”)
  - will create a new paragraph element that is ready to be added into the document
  - But you need to do something with it!
- $(“ Hello New World ”).appendTo(“#someElement”);
- Will append it to the element with ID someElement
- Alternatives include prependTo( )

Look at our example again

```
<html>
    <head>
        <meta charset="UTF-8">
        <title>Some jQuery Examples</title>
        <script src="jquery-1.8.2.min.js"></script>
        <script>
            $(function(){
                $('#myTitle')
                .bind('mouseover', function(event){
                    $("<p class='bear'>I have a bear!</p><p>I don't</p>")
                    .filter(".bear").click(function(){
                        alert("I'm a bear!");
                    }).end().appendTo("#myParentDiv");
                })
            })
        </script>
    </head>
    <body>
        <h1 id="myTitle">Some jQuery examples</h1>
        <p id="myParentDiv">The fun starts here:</p>
        <p>And ends here!</p>
    </body>
</html>
```

Define the document ready handler

```
<script>
    $(function(){
        $('#myTitle')
        .bind('mouseover', function(event){
            $("<p class='bear'>I have a bear!</p><p>I don't</p>")
            .filter(".bear").click(function(){
                alert("I'm a bear!");
            }).end().appendTo("#myParentDiv");
        })
    })
</script>
```

When the document is ready
bind a mouseover event handler to myTitle

```
<script>
   $(function(){
      $('#myTitle')
      .bind('mouseover', function(event){
         $("<p class='bear'>I have a bear!</p><p>I don't</p>")
         .filter(".bear").click(function(){
             alert("I'm a bear!");
         }).end().appendTo("#myParentDiv");
      })
   })
</script>
```

On “mouseover” create two new paragraphs One has class ‘bear’

```
<script>
   $(function(){
      $('#myTitle')
      .bind('mouseover', function(event){
         $("<p class='bear'>I have a bear!</p><p>I don't</p>")
         .filter(".bear").click(function(){
             alert("I'm a bear!");
         }).end().appendTo("#myParentDiv");
      })
   })
</script>
```

Filter out from that element set only those with class ‘bear’

```
<script>
   $(function(){
      $('#myTitle')
      .bind('mouseover', function(event){
         $("<p class='bear'>I have a bear!</p><p>I don't</p>")
         .filter(".bear").click(function(){
             alert("I'm a bear!");
         }).end().appendTo("#myParentDiv");
      })
   })
</script>
```

Bind a “click” event handler to the new elements of class ‘bear’

```
<script>
   $(function(){
      $('#myTitle')
      .bind('mouseover', function(event){
         $("<p class='bear'>I have a bear!</p><p>I don't</p>")
         .filter(".bear").click(function(){
             alert("I'm a bear!");
         }).end().appendTo("#myParentDiv");
      })
   })
</script>
```

Revert back to the full set of new elements

```
<script>
   $(function(){
      $('#myTitle')
      .bind('mouseover', function(event){
         $("<p class='bear'>I have a bear!</p><p>I don't</p>")
         .filter(".bear").click(function(){
             alert("I'm a bear!");
         }).end().appendTo("#myParentDiv");
      })
   })
</script>
```

And append them to myParentDiv

```
<script>
    $(function(){
        $('#myTitle')
        .bind('mouseover', function(event){
            $("<p class='bear'>I have a bear!</p><p>I don't</p>")
            .filter(".bear").click(function(){
                alert("I'm a bear!");
            }).end().appendTo("#myParentDiv");
        })
    })
</script>
```

What we have done

- This covers the basics of jQuery
- Note that you will still need to write your own control logic in JavaScript
- More on jQuery will be covered in the lab classes
- For much more on jQuery, see for example *jQuery in Action:*
  - http://www.manning.com/bibeault/