

The behavioural layer (JavaScript & jQuery)

Webpage Design

Web design layers

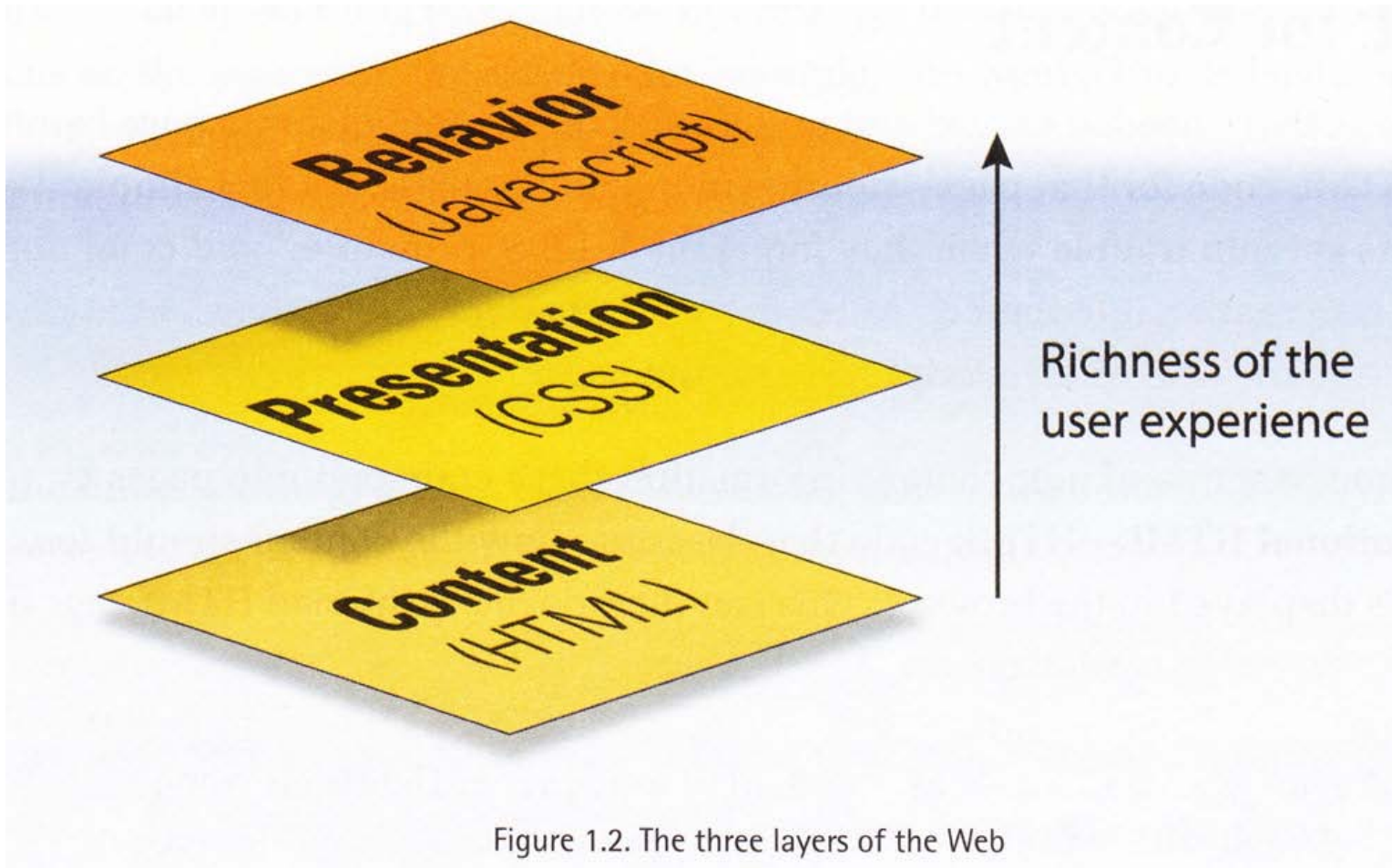


Figure 1.2. The three layers of the Web

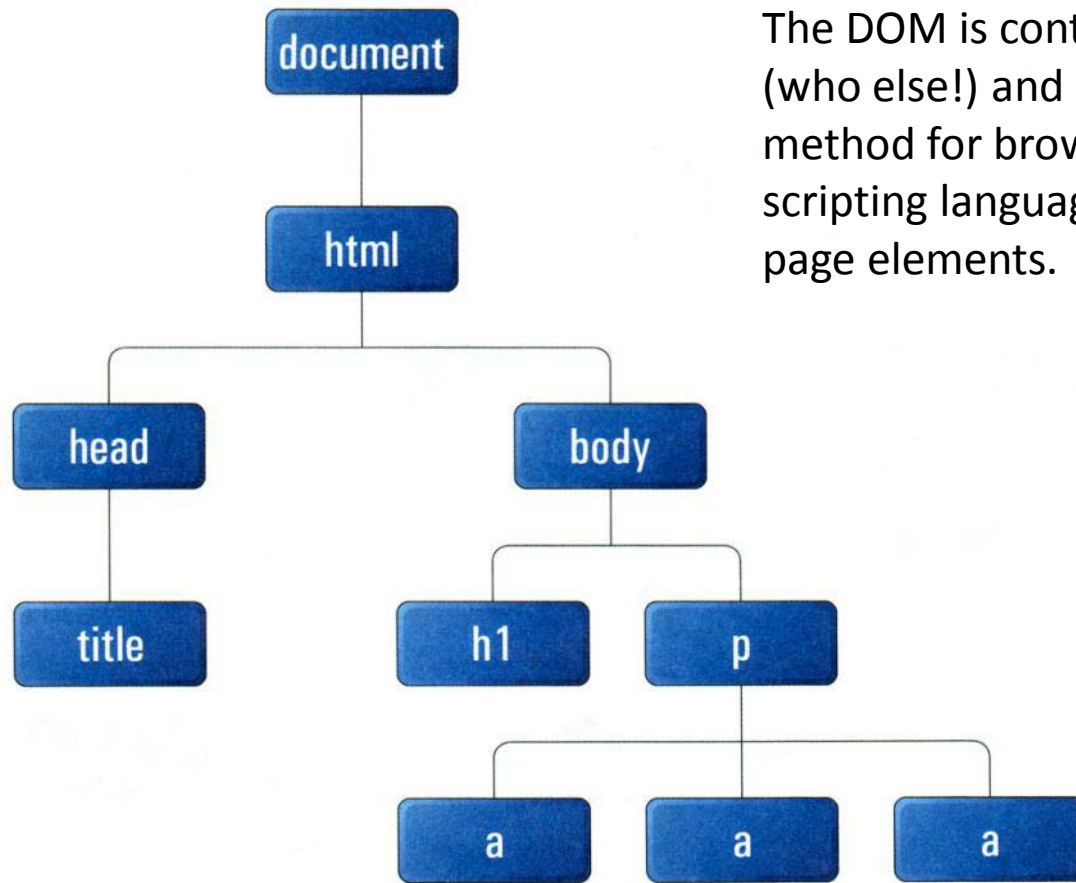
JavaScript

- Originally developed by Netscape with Sun Microsystems in 1995
- Microsoft followed with Jscript
- JavaScript now standardised by ECMA*
- Got a bad name because of misuse by some designers
- Has now been rehabilitated and used with the DOM and other web standards (basis of Ajax**)

* European Computer Manufacturers Association

** Asynchronous JavaScript And XML

The Document Object Model



The DOM is controlled by W3C (who else!) and is a standard method for browsers and scripting languages to access page elements.

Figure 3.2. The DOM tree, including the document node

Assigning objects to variables using the DOM

```
<body>
```

```
var body = document.getElementsByTagName("body")[0];
```

[Example](#) (body background-color)

```
<div id= "box">some content</div>
```

```
var box = document.getElementById("box");
```

[Example](#) (#box background-color)

Changing CSS values

A single value:

```
function changeText(){  
document.getElementById("box").style.fontSize = "150%";  
}
```

[Example](#) (#box text-size)

Multiple values:

```
function changeStuff(){  
var box = document.getElementById("box");  
box.style.backgroundColor = "#EFAE4C";  
box.style.border = "10px solid #798492";  
box.style.paddingLeft = "100px";  
}
```

[Example](#) (#box various)

A Jump Menu

```
<script type="text/javascript" src="/javascript/jump.js"></script>
```



Course Stuff
teaching and learning materials

Home | Contact | Upload | Files | Reading | Sign-in

Select Course

- Select Course
- Landscape Science & Techniques
- Digital Landscapes
- Advanced Representation
- Landscape Digital Design
- CAD & Visualisation
- Webpage Design
- Website Planning
- Net Art/Criticism

Webpage Design

Course code : DESI1046
Course co-ordinator : David Watson

Introduction

This course covers the design and implementation of web pages and small websites. It is concerned with the client-side technologies, XHTML and CSS and includes an introduction to

```
<form name="jumpmenu" id="jumpmenu" action="">  
<select name="jump" onchange="MM_jumpMenu('parent',this,0)">  
<option selected="selected" value="#">Select Course</option>  
<option value="/ENVT1053/">Landscape Science & Techniques</option>  
<option value="/ENVT1008/">Digital Landscapes</option>  
<option value="/ENVT1010/">Advanced Representation</option>  
<option value="/ENVT1016/">Landscape Digital Design</option>  
<option value="/ENVT1023/">CAD & Visualisation</option>  
<option value="/DESI1046/">Webpage Design</option>  
<option value="/DESI1047/">Website Planning</option>  
<option value="/DESI1054/">Net Art/Criticism</option>  
</select>  
</form>
```

A Jump Menu

jump.js

```
function MM_jumpMenu(targ,selObj,restore){ //v3.0
  eval(targ+".location='"+selObj.options[selObj.selectedIndex].value+"'");
  if (restore) selObj.selected = 0;
}

function MM_findObj(n, d, p) { //v4.0
  var p,i,x; if(!d) d=document; if(!p && document.frames.length) {
    p=parent; d=p.document; }
  if(!(x=d[n])&&d.all) x=d.all[n]; if(x!=undefined) return x;
  for(i=0;!x&&d.layers&&i<d.layers.length;i++) x=MM_findObj(n,d.layers[i].document);
  if(!x && d.getElementById) x=d.getElementById(n);
}

function MM_jumpMenu(targ,selObj,restore){ //v3.0
  var selObj = MM_findObj(selObj);
  MM_jumpMenu(targ,selObj,restore);
}
```



Typical reaction to complex javascript – it's not for designers, it's for coders.

What the...?

“...That rascally scripting language, cast as the black sheep of the web development family for so many years. JavaScript is how you add complex behaviours, sophisticated interactions, and extra pizzazz to your site.
...you just need to spend the next few years learning about programming languages: functions, classes, design patterns, prototypes, closures...”

From: jQuery, Novice to Ninja (Sitepoint)

Keeping markup clean

- The main problem with traditional approaches to JavaScript is that some of the code has to be placed inline (i.e. in the markup).
- Shouldn't we be separating JavaScript from our markup in the same way that we separate CSS from our markup?
- This can be done, using the DOM to target page elements but requires advanced JavaScript skills.
- We don't have or want to learn advanced JavaScript skills.
- What's the solution?

JavaScript Libraries/Frameworks

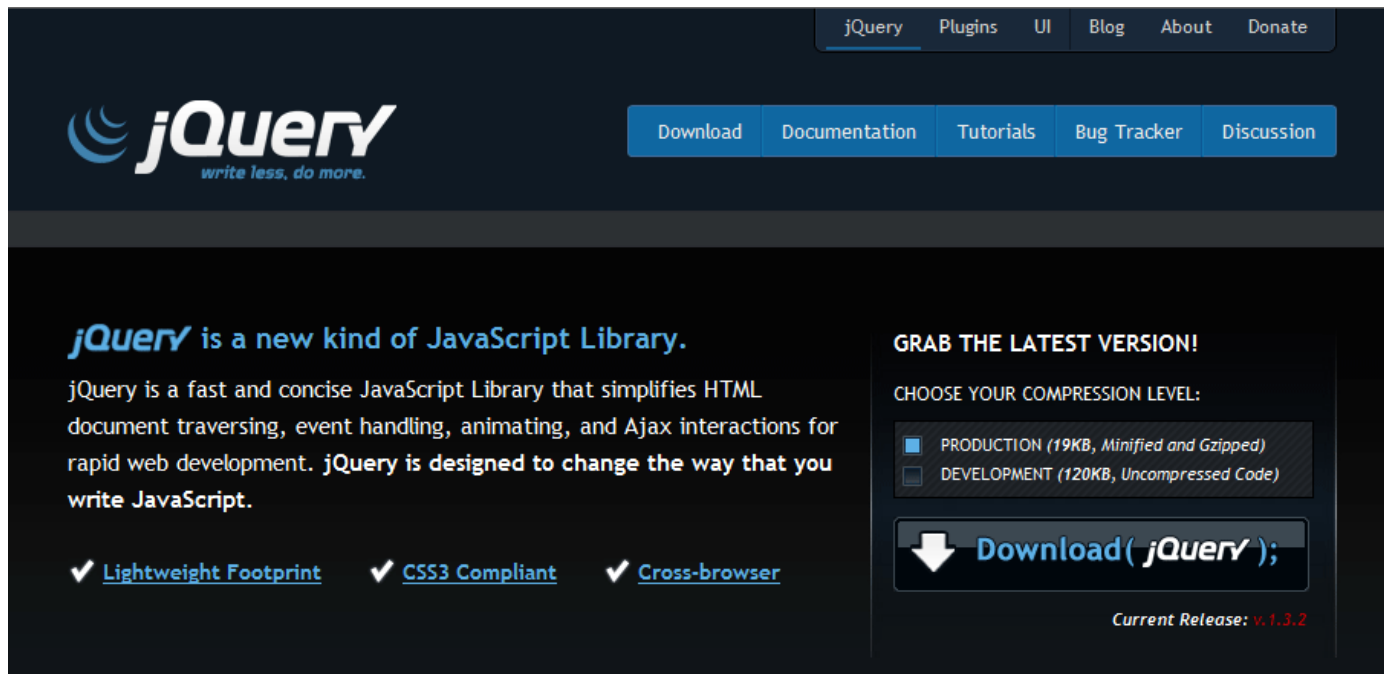
- JavaScript libraries are collections of pre-designed functions that can be added to webpages using a simplified JavaScript syntax.
- There are lots of JavaScript libraries (sometimes known as “frameworks”).
- Moo Tools, YUI, MoochiKit, Google Web Tool Kit, Spry and jQuery are just a few of the better known.
- Which one should we use?

Choosing a JavaScript Library

- In general, this will depend on the requirements of your project and your level of JavaScript experience.
- By far the most popular library is jQuery. It is well supported and documented, concise, fast and, most important of all, easy to use.
- However, some libraries may be better suited to specific tasks.
- One of the most important features of jQuery is that it is designed to work “cross-browser”.

jQuery

- jQuery is a JavaScript library that we can use to implement advanced techniques without needing to know advanced JavaScript.



The screenshot shows the jQuery website homepage. At the top, there is a navigation bar with links for 'jQuery', 'Plugins', 'UI', 'Blog', 'About', and 'Donate'. Below this is the jQuery logo with the tagline 'write less, do more.' and a secondary navigation bar with links for 'Download', 'Documentation', 'Tutorials', 'Bug Tracker', and 'Discussion'. The main content area features a heading 'jQuery is a new kind of JavaScript Library.' followed by a paragraph describing the library's features. Below the paragraph are three checkmarks indicating 'Lightweight Footprint', 'CSS3 Compliant', and 'Cross-browser'. On the right side, there is a section titled 'GRAB THE LATEST VERSION!' with a 'CHOOSE YOUR COMPRESSION LEVEL:' section containing two radio buttons: 'PRODUCTION (19KB, Minified and Gzipped)' (selected) and 'DEVELOPMENT (120KB, Uncompressed Code)'. Below this is a large 'Download (jQuery);' button with a download icon. At the bottom right, it says 'Current Release: v.1.3.2'.

<http://jquery.com>

Starting to use jQuery

- To begin using jQuery, all you need to do is to download the small .js file and link to it from the head section of your page or simply link to the .js file hosted elsewhere.

```
<script type="text/javascript" src="/javascript/jquery-1.3.2.min.js"></script>
```

OR

```
<script type="text/javascript" src="http://code.jquery.com/jquery-latest.js"></script>
```

- Once you have done that, you can start using jQuery in your pages.

jQuery in practice

- Most jQuery functions perform an action on a page element when triggered by a user action on another page element.
- For example, the background colour of a paragraph could be changed by clicking on a link.
- Each page element can be identified using a selector and selectors in jQuery work in exactly the same way as they do in CSS.
- The result of this selection method means that the JavaScript can be completely separate from the markup in the same way as CSS.

A simple jQuery function

```
<p id="change">Duis sed ligula odio, sed posuere magna...</p>
```

The target

```
<form>
```

The trigger

```
  <input type="button" value="Make Change"
```

```
  onclick="makeChange()" />
```

The HTML

```
</form>
```

```
.change_me {  
  background-color:#BADB35;  
}
```

The CSS

```
<script type="text/javascript">  
  function makeChange(){  
    $("#change").toggleClass('change_me');  
    return false;  
  }  
</script>
```

The JavaScript

This example uses the jQuery `.toggleClass()` function to add/remove a class from an element: [Example](#)

A breakdown

your function name

```
function makeChange(){
```

selector

CSS class name

```
jQuery $("#change").toggleClass('change_me');
```

jQuery function name

```
return false; end the function quietly
```

```
} close the function
```

jQuery example

```
<p>Duis sed ligula odio, sed posuere magna...</p>  
<p id="extra">Duis sed ligula odio, sed posuere magna...</p>  
<p id="link"><a href="#extra">More content&hellip;</a></p>
```

The HTML

```
#extra {  
display:none;  
}
```

The CSS

```
<script src="/javascript/jquery-1.3.2.min.js" type="text/javascript"></script>  
<script type="text/javascript">  
$(document).ready(function(){  
    $("#link a").click(function(){  
        $("#extra").toggle();  
        return false;  
    });  
});  
</script>
```

The JavaScript

This example uses the jQuery `.toggle()` function to show/hide page elements: [Example](#)

jQuery example (slide effect)

```
<p>Duis sed ligula odio, sed posuere magna...</p>
<p id="extra">Duis sed ligula odio, sed posuere magna...</p>
<p id="link"><a href="#extra">More content&hellip;</a></p>
```

The HTML
(structural layer)

```
#extra {
display:none;
}
```

The CSS
(presentation layer)

```
<script src="/javascript/jquery-1.3.2.min.js" type="text/javascript"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("#link a").click(function(){
        $("#extra").slideToggle();
        return false;
    });
});
</script>
```

The JavaScript
(behavioural layer)

This example uses the jQuery `.slideToggle()` function to show/hide page elements with a "slide" effect. Notice that no change is required to HTML or CSS: [Example](#)

Separation of the Layers

- Our ultimate goal is to separate the 3 layers of our page, the Structural, the Presentation and the Behavioural.
- Using a combination of XHTML, CSS and jQuery, a web designer can achieve this goal without needing to know how to work with JavaScript at a fundamental level.
- The glue that holds all these things together is the selector, whether ID, class, tag or combination.
- Once we define an id `<div id="element">`, this element can be targeted by both CSS and jQuery (`#element`).

jQuery UI

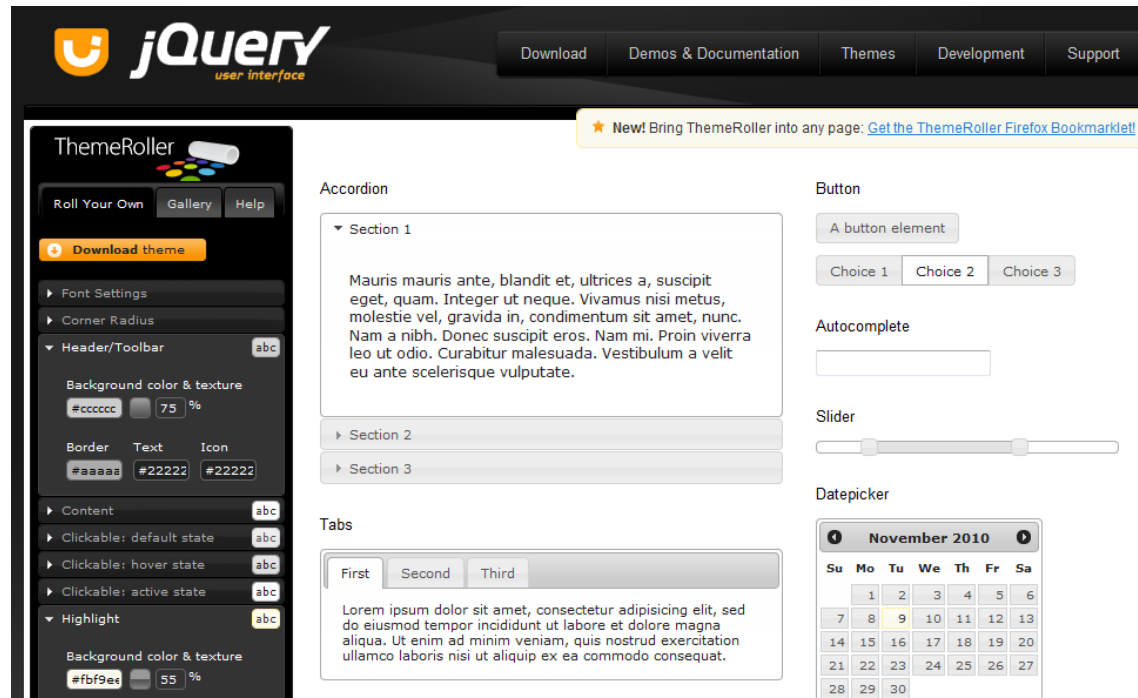
- jQuery UI takes the concept of a JavaScript library to the next level by providing a library of “widgets”, effects and utilities and not just functions.
- Each widget is composed of the core jQuery library, a widget specific function library, associated CSS and image files.
- jQuery UI provides standard solutions to common interface design methods.

This example uses the jQuery UI Tabs widget to create a tabbed interface. Notice that the markup is very simple, as is the JavaScript function that makes it work: [Example](#)

Customizing your widget

- Widgets can easily be customized by editing the supplied images and CSS file so that they blend with your site design.

However, the jQuery UI site also provides a “ThemeRoller” so that you can create your own bespoke theme.

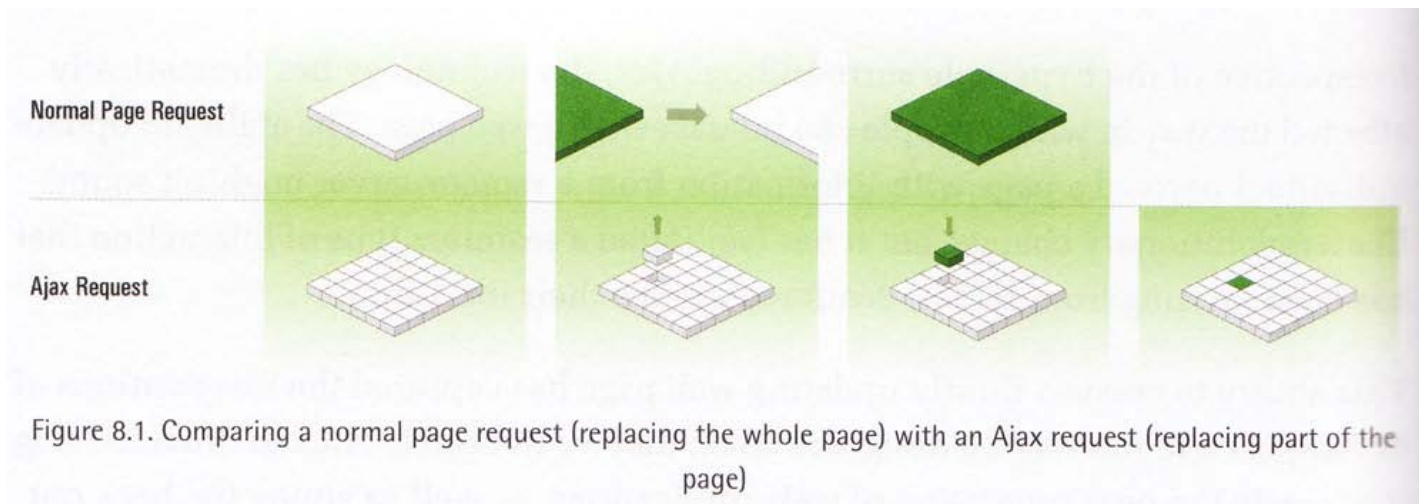


The screenshot displays the jQuery ThemeRoller interface, which is used for customizing jQuery UI widgets. The interface is divided into several sections:

- jQuery user interface** logo and navigation links: Download, Demos & Documentation, Themes, Development, Support.
- ThemeRoller** header with navigation: Roll Your Own, Gallery, Help, and a Download theme button.
- Font Settings** and **Corner Radius** sections.
- Header/Toolbar** section with options for Background color & texture (hex #ccccc, 75%), Border, Text, and Icon.
- Content** section with options for Clickable: default state, Clickable: hover state, and Clickable: active state.
- Highlight** section with options for Background color & texture (hex #fbf9e8, 55%).
- Accordion** widget preview showing Section 1, Section 2, and Section 3.
- Button** widget preview showing a button element and three choices.
- Autocomplete** widget preview showing a text input field.
- Slider** widget preview showing a horizontal slider.
- Datepicker** widget preview showing a calendar for November 2010.
- Tabs** widget preview showing three tabs: First, Second, and Third.

What is AJAX?

- **AJAX** (**A**synchronous **J**avaScript and **X**ML)
- Data is retrieved using the XMLHttpRequest object
- Allows parts of a page to be refreshed with new data from the server



```
function endSlideshow(){  
    $("#slideshow").toggleClass('end');  
    return false;  
}
```