COSC 617 – AJAX

Siddharth Kaza
Towson University, Computer and Information Sciences

Problems with Web 1.0

- Until 2005, browsers were really dumb
  - Web pages are not sufficiently dynamic
  - Full round-trip required for form field verification
  - Interfaces feel clunky and slow
  - Layout is difficult

Thick clients vs. Web 1.0

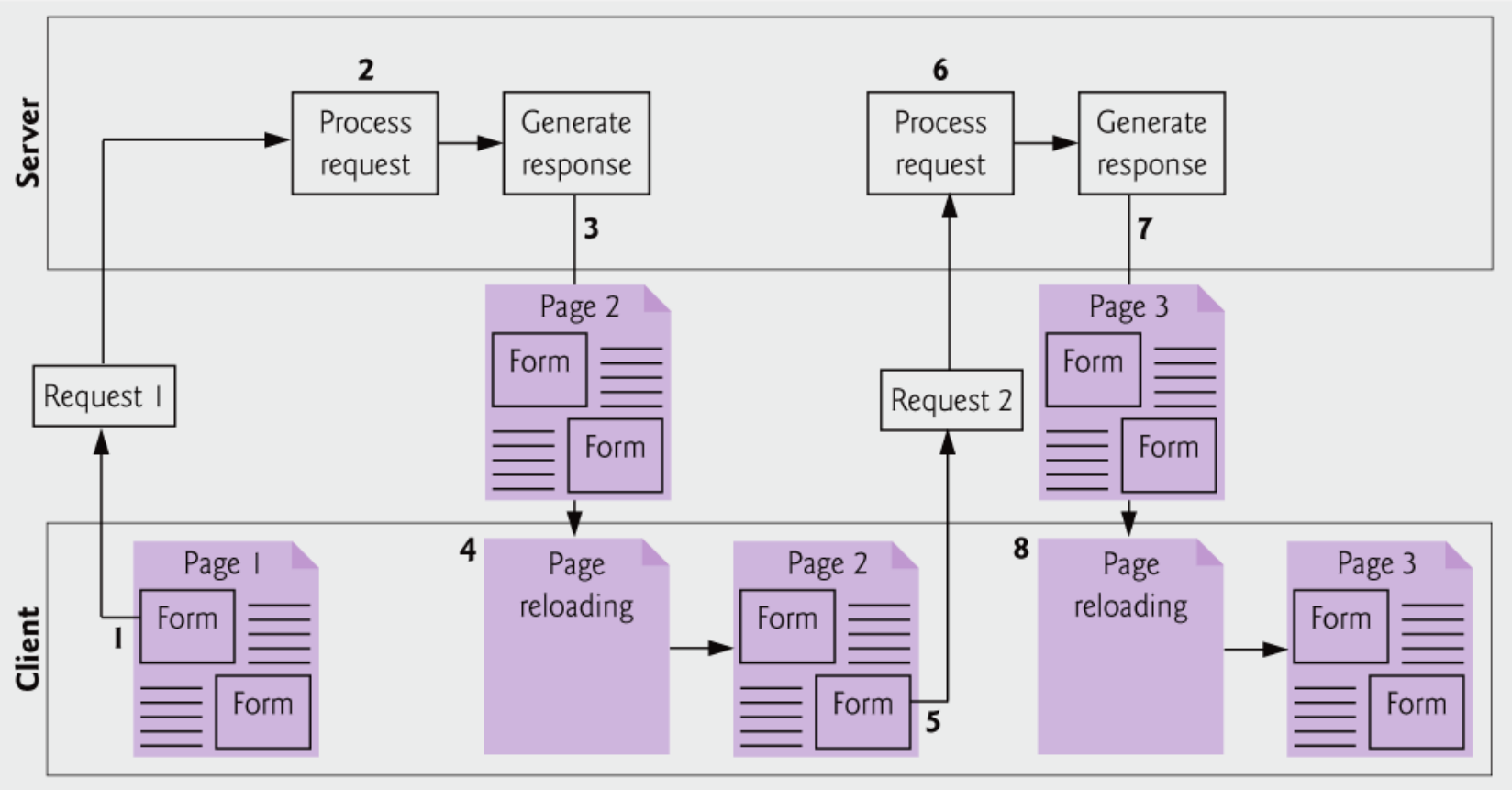
- Thick-clients
  - Full control of GUI
  - Layout of components
  - "Skins" - selectable look and feel
  - Update text labels, menu choices, etc. based on user inputs
  - Respond to mouse events
  - Asynchronous data access (in threaded environments)

- Web 1.0
  - None of the above

Traditional Web Applications

- Traditional web applications (look at the picture next slide)
  - User fills in the form's fields, then submits the form
  - Browser generates a request to the server, which receives the request and processes it
  - Server generates and sends a response containing the exact page that the browser will render
  - Browser loads the new page and temporarily makes the browser window blank
  - Client *waits* for the server to respond and *reloads the entire page* with the data from the response
- While a synchronous request is being processed on the server, the user cannot interact with the client web browser
- The synchronous model was originally designed for a web of hypertext documents

Traditional Web Applications (cont.)



5 from Deital P. J. & Deital, H. M., Internet and World Wide Web How to Program, Prentice Hall, 4e

Rich Internet Applications (RIA)

- RIA are Web applications that approximate the look, feel and usability of desktop applications
  - Two key attributes—performance and rich GUI

RIA - performance

- Comes from Ajax (Asynchronous JavaScript and XML), which uses client-side scripting to make web applications more responsive
- Ajax applications separate client-side user interaction and server communication,
  - and run them in parallel,
  - making the delays of server-side processing more transparent to the user
- "Raw" Ajax uses JavaScript to send asynchronous requests to the server, then updates the page using the DOM
- When writing "raw" Ajax you need to deal directly with cross-browser portability issues, making it impractical for developing large-scale applications

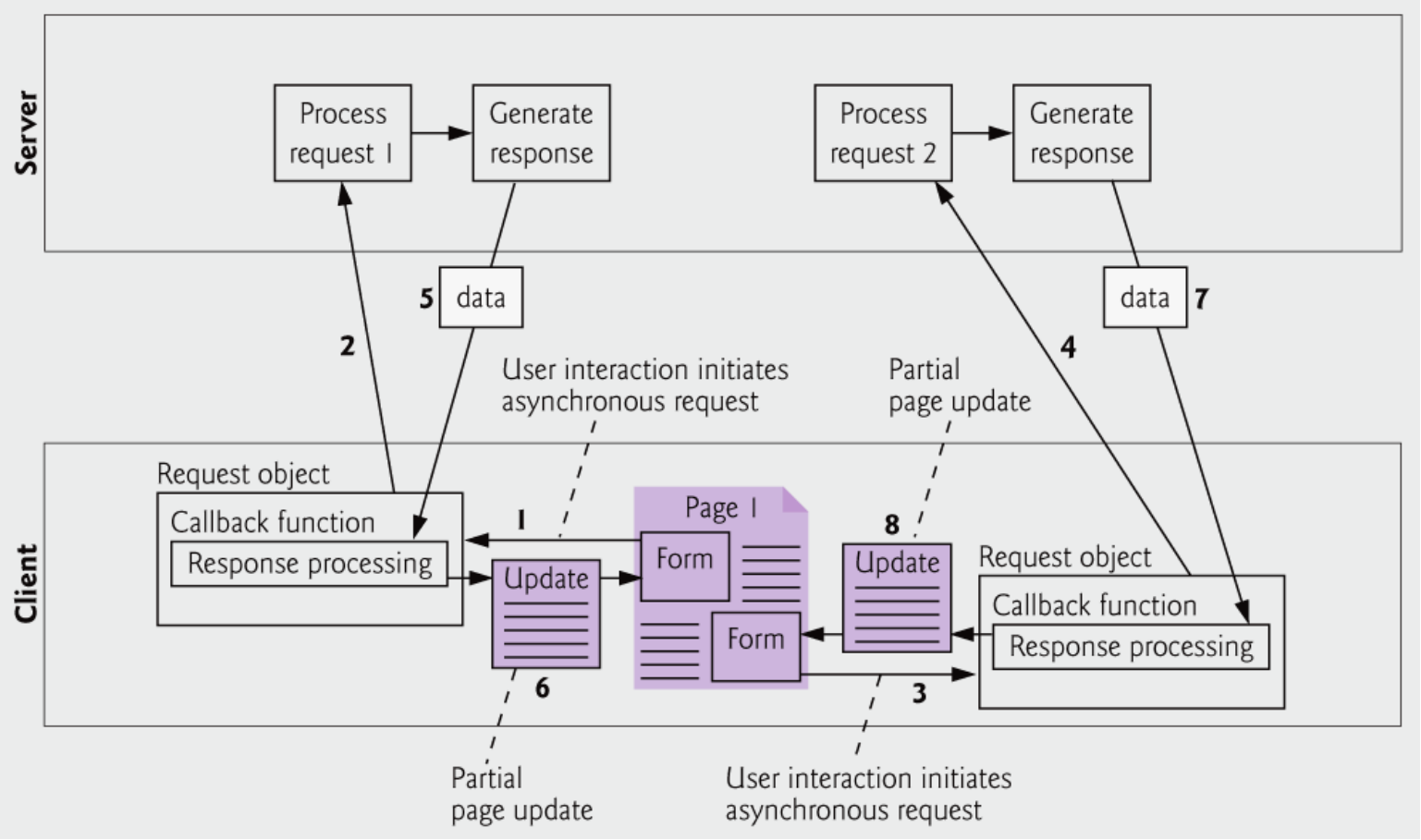
RIA (cont.)

- Portability issues
  - Hidden by Ajax toolkits, such as Dojo, Prototype and Script.aculo.us
  - Toolkits provide powerful ready-to-use controls that simplify JavaScript coding
- Achieve rich GUI in RIAs with
  - Ajax toolkits
  - RIA environments such as Adobe's Flex, Microsoft's Silverlight and JavaServer Faces
- XML and JSON are used to structure the data passed between the server and the client
- `XMLHttpRequest`
  - The Ajax component that manages interaction with the server
  - Commonly abbreviated as XHR.

AJAX Web Application

- In an Ajax application, when the user interacts with a page
  - Client creates an `XMLHttpRequest` object to manage a request
  - `XMLHttpRequest` object sends the request to and awaits the response from the server
  - Requests are asynchronous, allowing the user to continue interacting with the application while the server processes the request concurrently
  - When the server responds, the `XMLHttpRequest` object that issued the request invokes a callback function, which typically uses partial page updates to display the returned data in the existing web page *without reloading the entire page*
- Callback function updates only a designated part of the page
- Partial page updates help make web applications more responsive, making them feel more like desktop applications

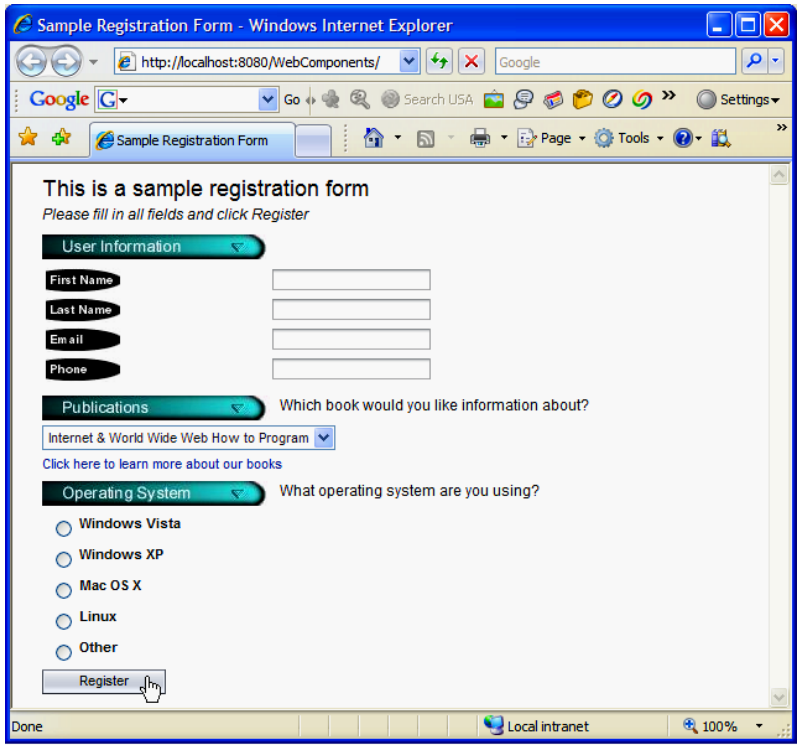
AJAX Web Application (cont.)

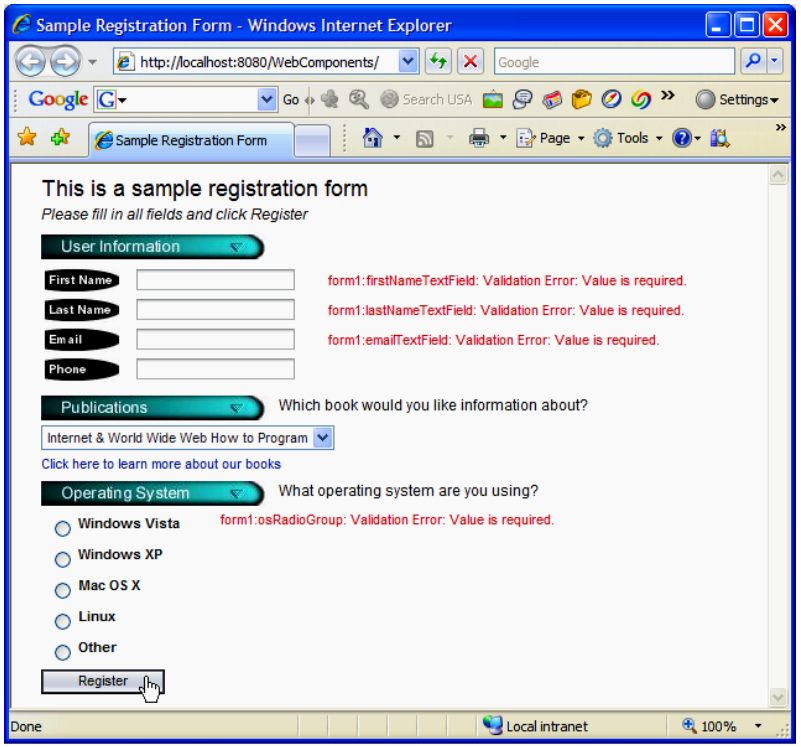


 from Deital P. J. & Deital, H. M., Internet and World Wide Web How to Program, Prentice Hall, 4e

Traditional vs. AJAX - A simple example

a) A sample registration form in which the user has not filled in the required fields, but attempts to submit the form anyway by clicking **Register**.

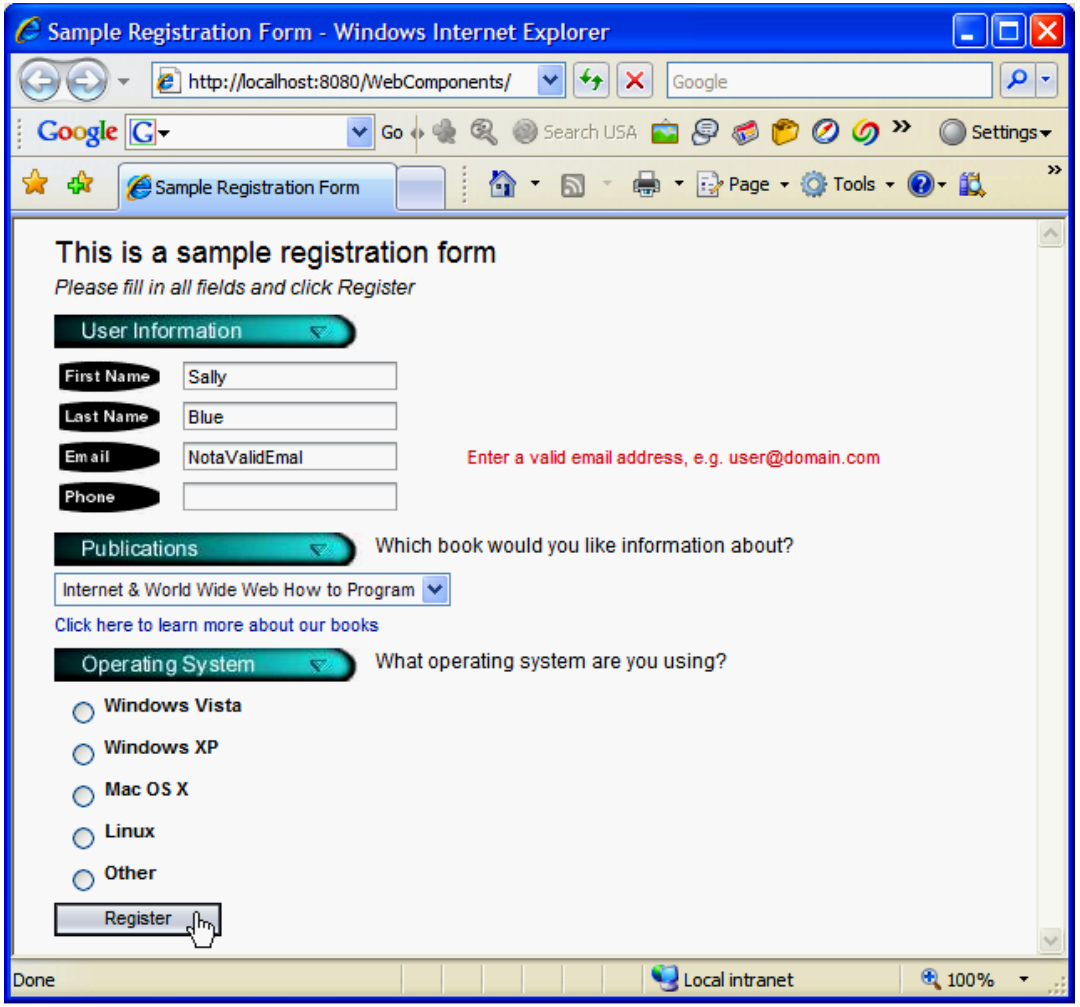




Classic XHTML form: User submits entire form to server, which validates the data entered (if any). Server responds indicating fields with invalid or missing data.

- from Deital P. J. & Deital, H. M., Internet and World Wide Web How to Program, Prentice Hall, 4e

Traditional vs. AJAX - A simple example



- Asynchronous requests could also be used to fill some fields based on previous fields' values (e.g., city based on zipcode)

Ajax-enabled form shows errors asynchronously when user moves to another field.

History of AJAX

- The term Ajax was coined by Jesse James Garrett of Adaptive Path in February 2005
- Ajax technologies (XHTML, JavaScript, CSS, dynamic HTML, the DOM and XML) have existed for many years
- In 1998, Microsoft introduced the `XMLHttpRequest` object to create and manage asynchronous requests and responses
- Popular applications like Flickr, Google's Gmail and Google Maps use the `XMLHttpRequest` object to update pages dynamically

Towards AJAX

- Javascript/ECMAScript
  - Insert code into web pages
  - Examine form fields to verify content
  - Respond to mouse events
  - Control browsers – open/close windows, etc.

- Document Object Model (DOM)
  - Hierarchical decomposition of contents of a web page
  - Or, more generally, XML document
  - Useful for programmatic manipulation of page content
    - By Javascript/ECMAScript

Towards Web 2.0

- Cascading style sheets (CSS)
- Specify visual attributes of XHTML elements
  - Separate from XHTML so the theoretically the style of the webpage can be changed completely by applying a different style sheet
  - The *class* and *id* selectors to select groups or one XHTML element
- Support grouping and layout.
- Can be manipulated via JavaScript
- "Cascading" - pages can get layout/visuals from multiple style sheets.

AJAX

- How do these technologies interact:
  - XHTML builds web forms and identifies fields
  - JavaScript code facilitates communication with server applications (can be VBScript, but only in Internet Explorer)
  - DOM will be used to work with the structure of the HTML

from: http://www.ibm.com/developerworks/web/library/wa-ajaxintro1.html?S_TACT=105AGX08&S_CMP=EDU

JavaScript / ECMAScript

- ECMAScript is the proper name
  - www.ecmascript.org
- Object-oriented
- dynamically-typed
- Interpreted
- Theoretically supported by major browsers
  - But.. "the best thing about standards is that there are so many of them..."
- Version 3 1999
- Version 4 -under development – never completed
- Version 5 – current

Before we get to Javascript

- Be Careful!
  - Many incompatibilities
  - Many usability /accessibility issues

- Provide a graceful default
  - Functionality that will work if ECMAScript support is broken, turned off, etc.
  - Good for accessibility purposes as well

```
1  <?xml version = "1.0" encoding = "utf-8"?>
2  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5  <!-- Fig. 6.2: welcome.html -->
6  <!-- Displaying a line of text. -->
7  <html xmlns = "http://www.w3.org/1999/xhtml">
8     <head>
9        <title>A First Program in JavaScript</title>
10       <script type = "text/javascript">
11          <!--
12          document.writeln(
13             "<h1>Welcome to JavaScript Programming!</h1>" );
14          // -->
15       </script>
16    </head><body></body>
17 </html>
```

Script begins

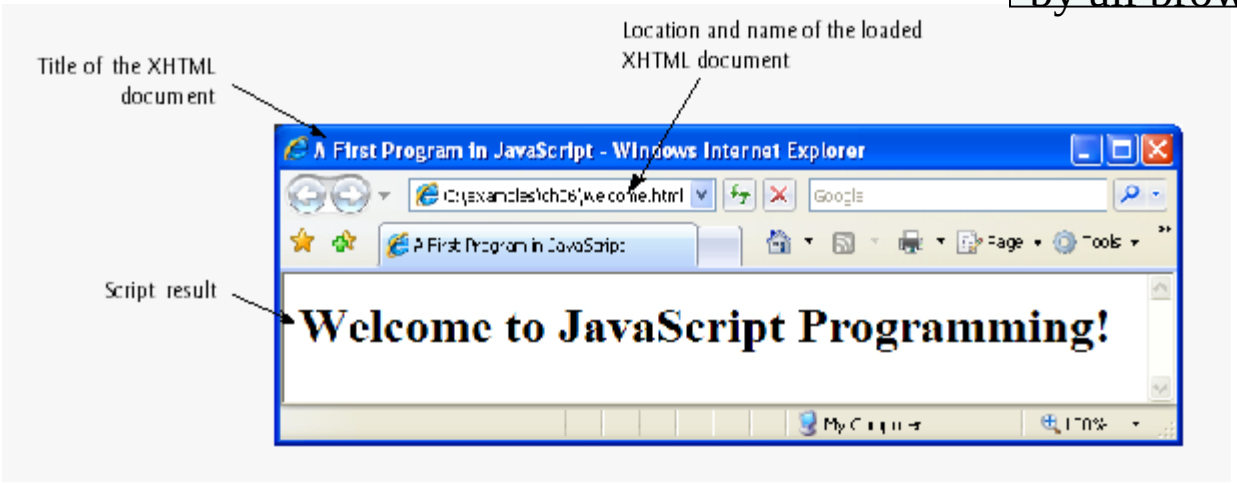
Specifies t… using the J… lang…

Write… messa… document

Prevents older browsers that do not support scripting from displaying the text of the script

XHTML comment delimiter, commented for correct interpretation by all browsers

Script ends



Attach an handler to an event

```
<SCRIPT type="‘text/JavaScript">
<!-- Beginning of JavaScript -
        function MsgBox (textstring) {
        alert (textstring) }
// - End of JavaScript - -->
</SCRIPT>
```

- Later, in a form

```
<INPUT NAME="text1" TYPE=Text>
<INPUT NAME="submit" TYPE=Button VALUE="Show Me"
onClick="MsgBox(form.text1.value)">
```

- From http://www.webteacher.com/javascript/ch01.html

Other possibilities

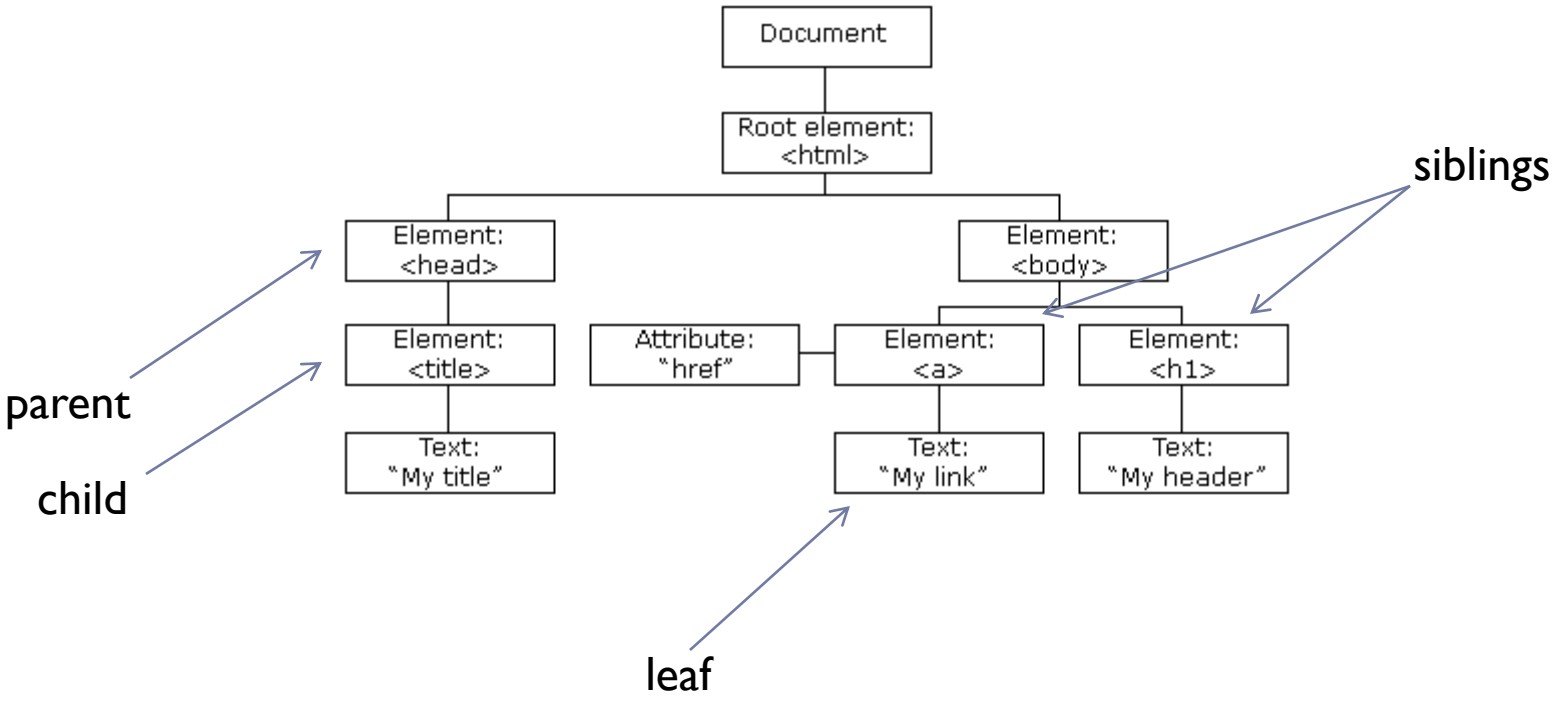
- Change images on mouse over
- Validate form contents
  - Javascript comes with all the control statements, many data structures (arrays), objects, functions
  - So there is a lot you can do client side
- Go wild? Or not.
  - Differentiate between what users will like, and what you think is "cool"
- Really gets interesting with DOM and CSS

Document Object Model (DOM)

- "The Document Object Model is a platform- and language neutral interface
  - that will allow programs and scripts to dynamically access and update the
    - content,
    - structure
    - and style of documents.
  - The document can be further processed and the results of that processing can be incorporated back into the presented page." - www.w3.org/DOM/

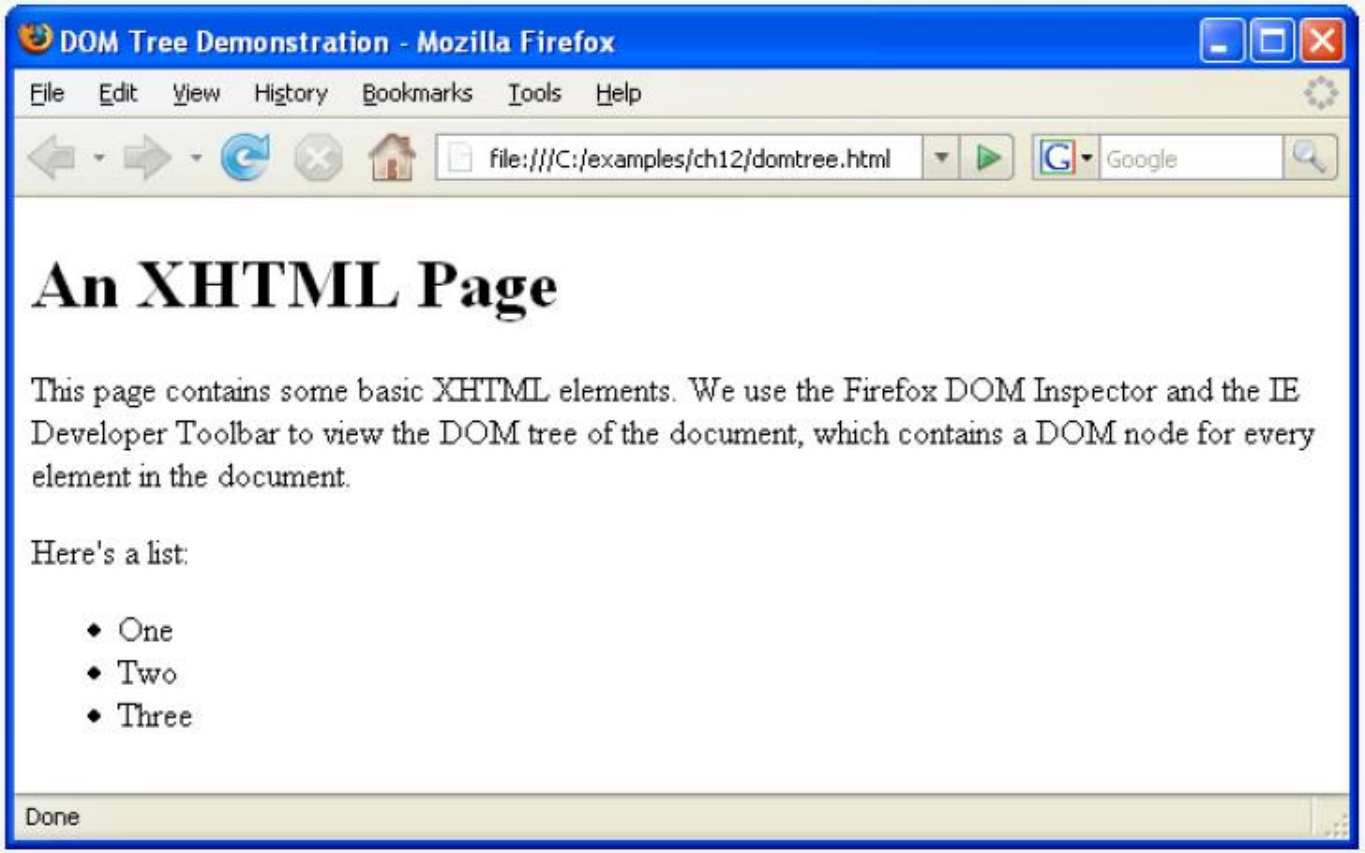
- A web page is no longer a static entity
- With the DOM, it can be examined, changed, etc.
  - Often (but not always) by Javascript

DOM Tree



From: http://www.w3schools.com/HTMLDOM/dom_nodetree.asp

a) The XHTML document is rendered in Firefox.

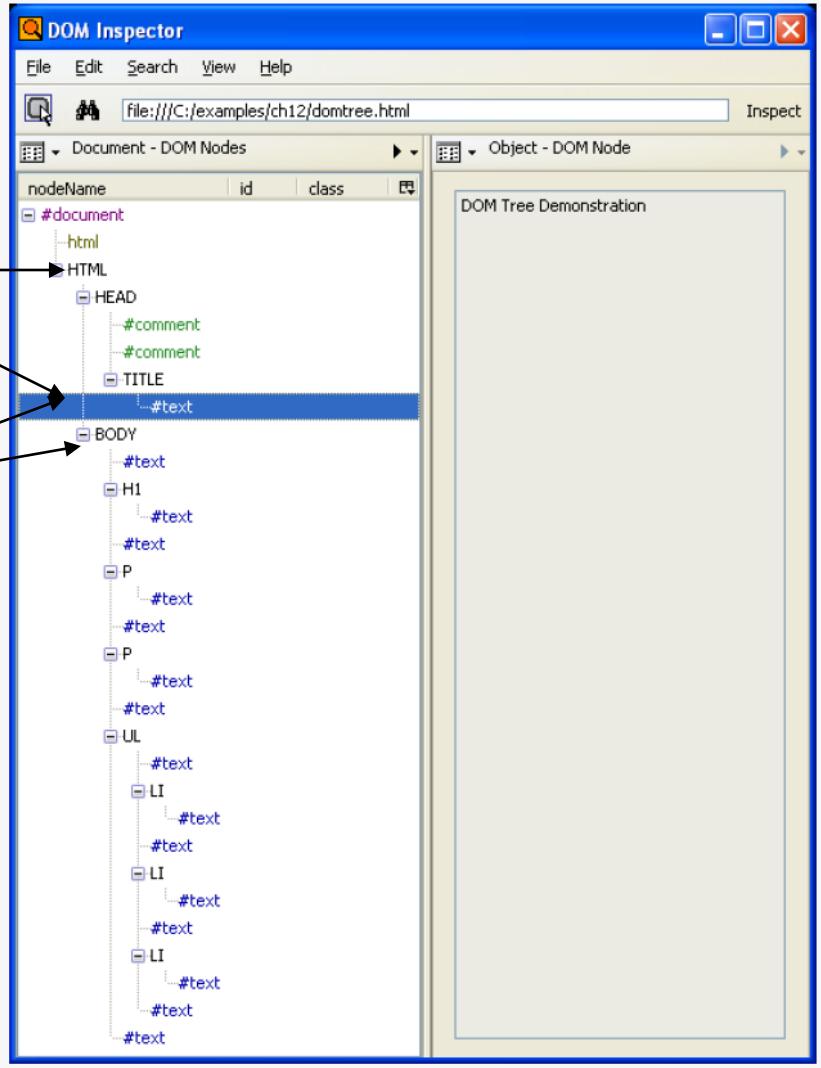


from Deital P. J. & Deital, H. M., Internet and World Wide Web How to Program, Prentice Hall, 4e

b) The Firefox DOM inspector displays the document tree in the left panel. The right panel shows information about the currently selected node.

HEAD and BODY nodes are siblings

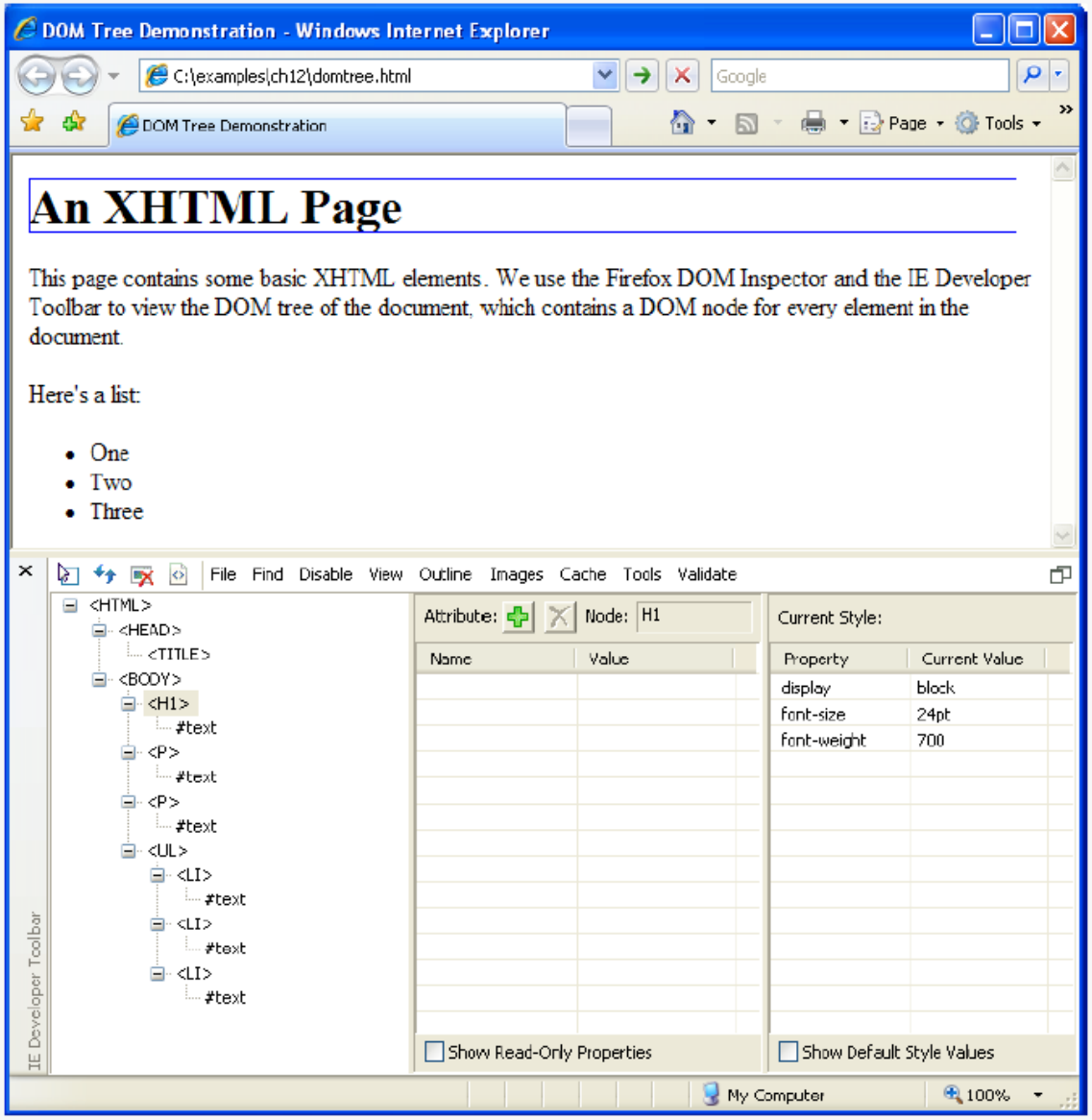
The BODY node is the parent of the H1 node



Tools
Firefox DOM Inspector
Firebug

from Deital P. J. & Deital, H. M., Internet and World Wide Web How to Program, Prentice Hall, 4e

c) The Internet Explorer Web Developer Toolbar displays much of the same information as the DOM inspector in Firefox in a panel at the bottom of the browser window.



Tools

from Deital P. J. & Deital, H. M., Internet and World Wide Web How to Program, Prentice Hall, 4e

DOM properties

- DOM properties get values/objects from the DOM
  - x.innerHTML
  - x.nodeName
  - x.nodeValue
  - x.parentNodes
  - x.childNodes
  - x.attributes
  - x.getElementByTagName

DOM (cont.)

- Since the document is a tree, you can use various tree navigation elements.
- Each item implements the Node interface
  - Methods for document structure
- Node.firstChild
- Node.lastChild
- Node.childNodes.length
- Nodes.childNodes[0]..Node.childNodes[length-1]
- Siblings: nextSibling, prevSibling
- Node.parentNode

DOM (cont.)

- Can modify with
  - insertBefore()
  - replaceChild()
  - appendChild()
  - removeChild()
  - cloneNode()

AJAX Without Toolkits - "Raw" Coding

- To initiate an asynchronous request
  - Create an instance of the `XMLHttpRequest` object
  - Use its `open` method to set up the request, and its `send` method to initiate the request

- Security
  - `XMLHttpRequest` object does not allow a web application to request resources from servers other than the one that served the web application
  - Prevents Cross Site Scripting (XSS)
  - You can implement a server-side proxy—an application on the web application's web server—that can make requests to other servers on the web application's behalf

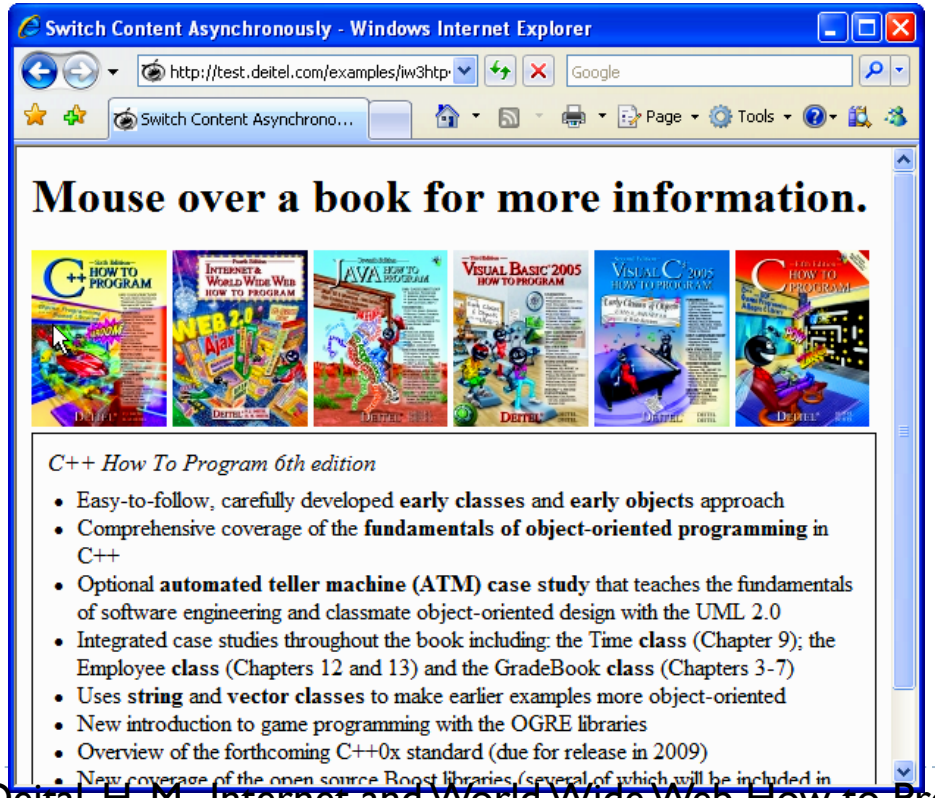
AJAX Without Toolkits - "Raw" Coding

- Creating a XMLHTTPRequest object is done in the try… catch block
- A callback function is registered as the event handler for the `XMLHttpRequest` object's `onreadystatechange` event
  - Whenever the request makes progress, the `XMLHttpRequest` calls the `onreadystatechange` event handler.
  - Progress is monitored by the `readyState` property, which has a value from 0 to 4
  - The value 0 indicates that the request is not initialized and the value 4 indicates that the request is complete.

AJAX Without Toolkits - "Raw" Coding (cont.)

- AIM - To show the description of the book when the user hovers over it.

a) User hovers over *C++ How to Program* book cover image, causing an asynchronous request to the server to obtain the book's description. When the response is received, the application performs a partial page update to display the description.



from Deital P. J. & Deital, H. M., Internet and World Wide Web How to Program, Prentice Hall, 4e

Outline

SwitchContent
ml

of 5)

```
1  <?xml version = "1.0" encoding = "utf-8"?>
2  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5  <!-- Fig. 15.5: SwitchContent.html -->
6  <!-- Asynchronously display content without reloading the page. -->
7  <html xmlns = "http://www.w3.org/1999/xhtml">
8  <head>
9     <style type="text/css">
10       .box { border: 1px solid black;
11              padding: 10px }
12    </style>
13    <title>Switch Content Asynchronously</title>
14    <script type = "text/javascript" language = "JavaScript">
15       <!--
16       var asyncRequest; // variable to hold XMLHttpRequest objec
17
18       // set up and send the asynchronous request
19       function getContent( url )
20       {
21          // attempt to create the XMLHttpRequest and make the re
22          try
23          {
24             asyncRequest = new XMLHttpRequest(); // create request object
25
26             // register event handler
27             asyncRequest.onreadystatechange = stateChange;
28             asyncRequest.open( 'GET', url, true ); // prepare the request
29             asyncRequest.send( null ); // send the request
30          } // end try
```

The program attempts to execute the code in the `try` block. If an exception occurs, the code in the `catch` block will be executed

`object` and store it in `asyncRequest`

Set the event handler for the `onreadystatechange` event to the function `stateChange`

The request will be a `GET` request for the page located at `url`, and it will be asynchronous

```
31          catch ( exception )
32          {
33             alert( 'Request failed.' );
34          } // end catch
35       } // end function getContent
36
37       // displays the response data on the page
38       function stateChange()
39       {
40          if ( asyncRequest.readyState == 4 && asyncRequest.status == 200 )
41          {
42             document.getElementById( 'contentArea' ).innerHTML =
43                asyncRequest.responseText; // places text in contentArea
44          } // end if
45       } // end function stateChange
46
47       // clear the content of the box
48       function clearContent()
49       {
50          document.getElementById( 'contentArea' ).innerHTML = '';
51       } // end function clearContent
52       // -->
```

Notify the user that an error occurred

If the request has completed successfully, use the DOM to update the page with the `responseText` property of the request object

Outline

`SwitchContent.html`

(2 of 5)

```
53    </script>
54 </head>
55 <body>
56    <h1>Mouse over a book for more information.</h1>
57    <img src =
58       "http://test.deitel.com/examples/iw3htp4/ajax/thumbs/cpphtp6.jpg"
59       onmouseover = 'getContent( "cpphtp6.html" )'
60       onmouseout = 'clearContent()'/>
61    <img src =
62       "http://test.deitel.com/examples/iw3htp4/ajax/thumbs/iw3htp4.jpg"
63       onmouseover = 'getContent( "iw3htp4.html" )'
64       onmouseout = 'clearContent()'/>
65    <img src =
66       "http://test.deitel.com/examples/iw3htp4/ajax/thumbs/jhtp7.jpg"
67       onmouseover = 'getContent( "jhtp7.html" )'
68       onmouseout = 'clearContent()'/>
69    <img src =
70       "http://test.deitel.com/examples/iw3htp4/ajax/thumbs/vbhtp3.jpg"
71       onmouseover = 'getContent( "vbhtp3.html" )'
72       onmouseout = 'clearContent()'/>
73    <img src =
74       "http://test.deitel.com/examples/iw3htp4/ajax/thumbs/vcsharphtp2.jpg"
75       onmouseover = 'getContent( "vcsharphtp2.html" )'
76       onmouseout = 'clearContent()'/>
```

Outline

SwitchContent.html

(3 of 5)

```
77    <img src =
78       "http://test.deitel.com/examples/iw3htp4/ajax/thumbs/chtp5.jpg"
79       onmouseover = 'getContent( "chtp5.html" )'
80       onmouseout = 'clearContent()'/>
81    <div class = "box" id = "contentArea"> </div>
82 </body>
83 </html>
```

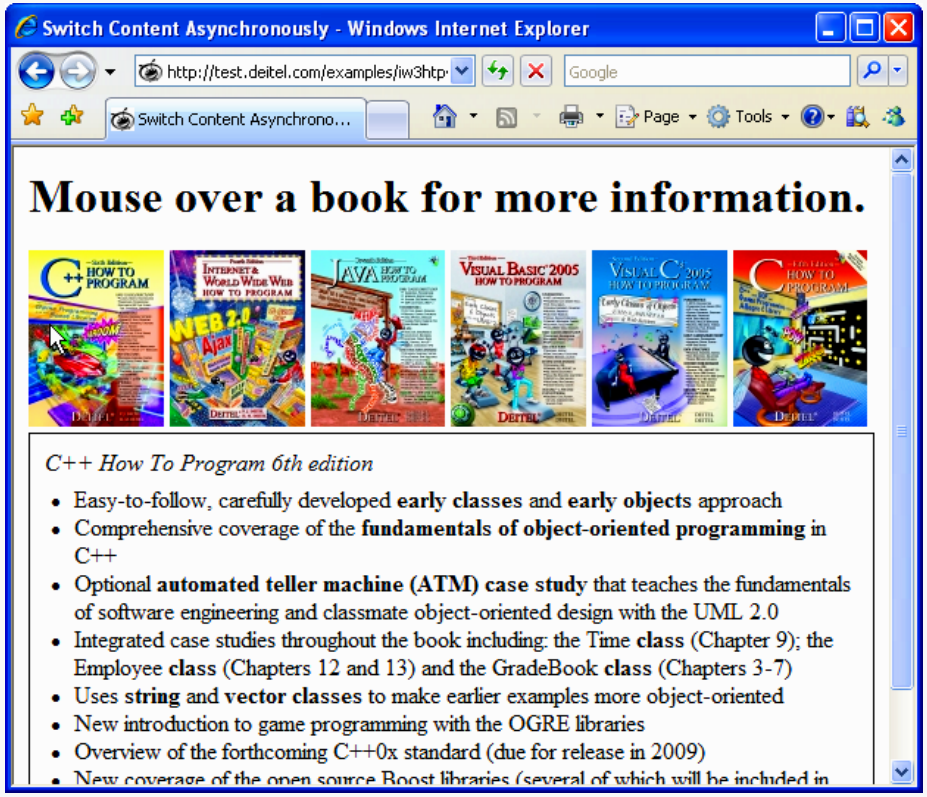
This `div` is updated with the description of the book that the mouse is currently hovering over

Outline

`SwitchContent.html`

(4 of 5)

a) User hovers over *C++ How to Program* book cover image, causing an asynchronous request to the server to obtain the book's description. When the response is received, the application performs a partial page update to display the description.

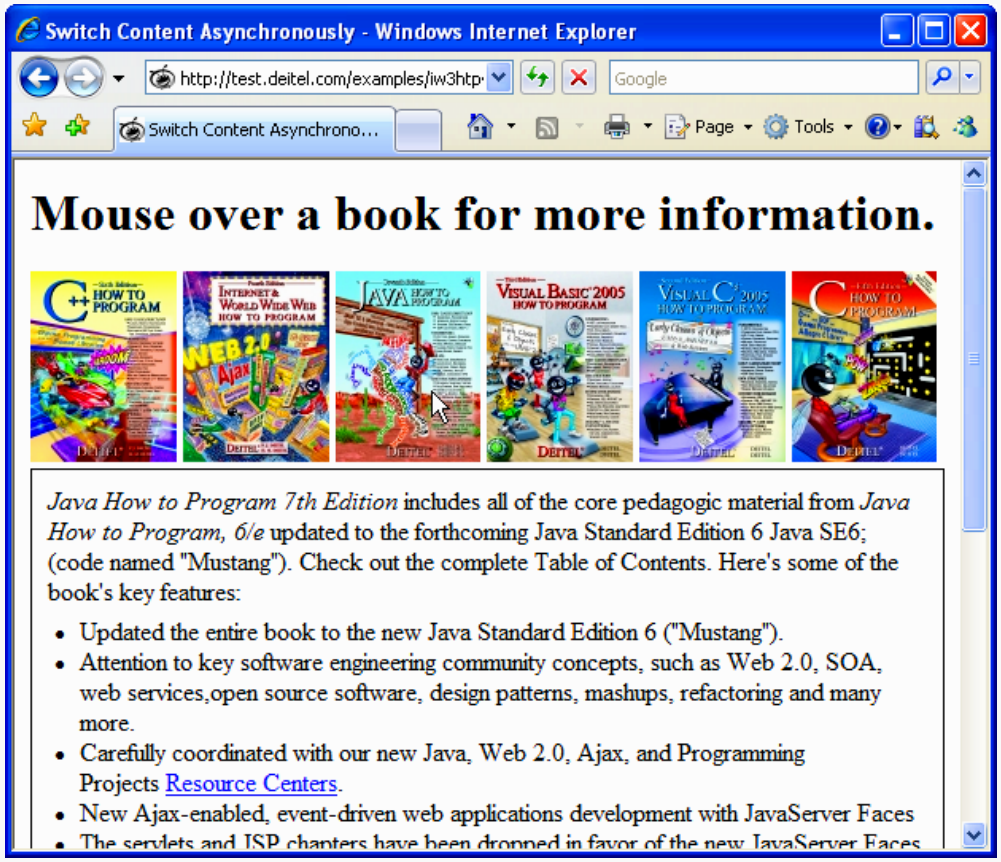


Outline

`SwitchContent.html`

(5 of 5)

b) User hovers over *Java How to Program* book cover image, causing the process to repeat.



AJAX with Toolkits (in Rails)

- Show a demo with Addressbook
  - https://github.com/siddharthkaza/addressbook_ajax_rails328