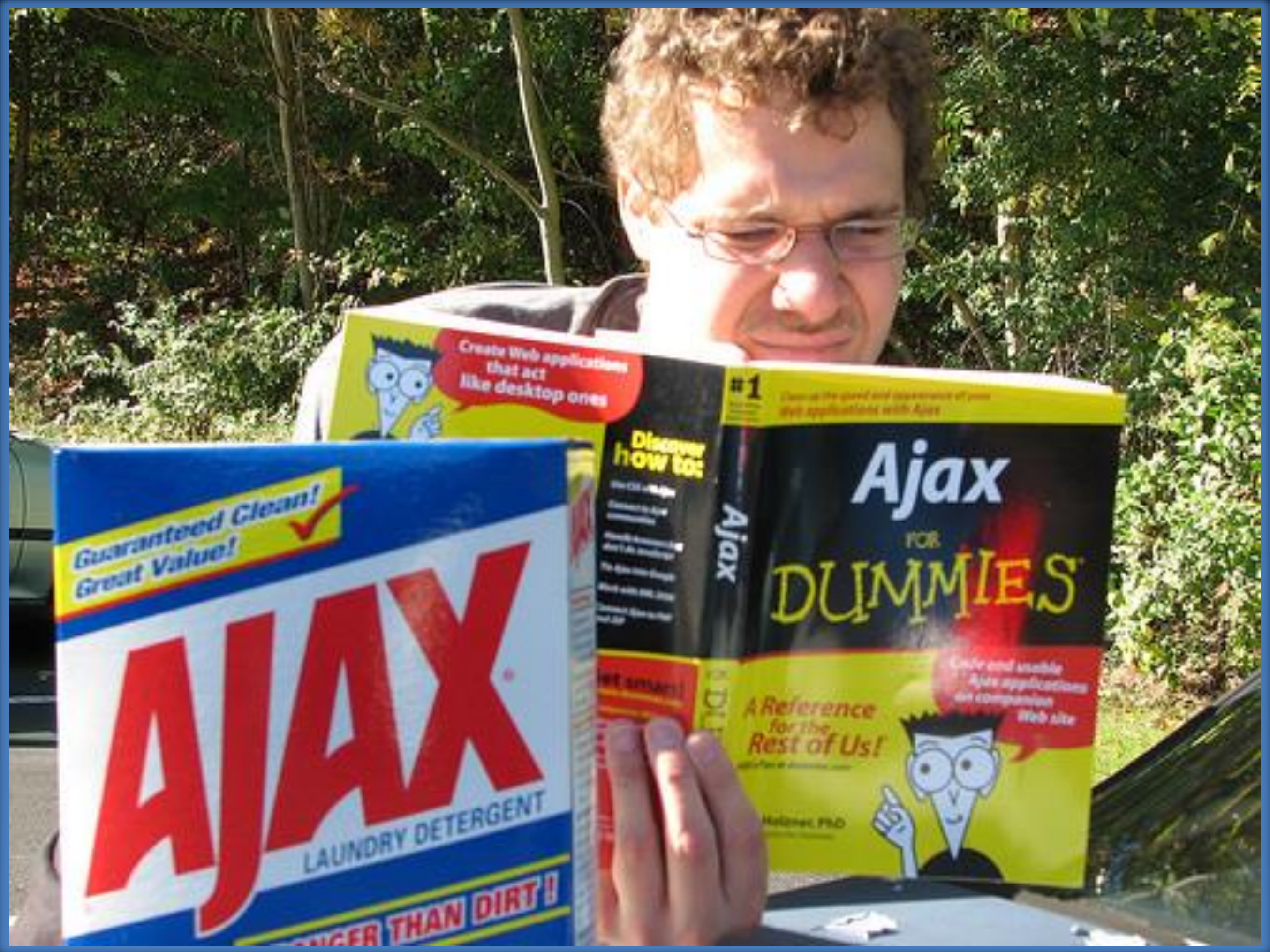

Guaranteed Clean!
Great Value!
AJAX
LAUNDRY DETERGENT
NGER THAN DIRT!
Create Web applications that act like desktop ones
Discover how to:
Ajax
#1
Ajax FOR DUMMIES
Code and usable Ajax applications on companion Web site
A Reference for the Rest of Us!
Holzner, PhD

Web Technology

Lecture 14: AJAX and jQuery

AJAX

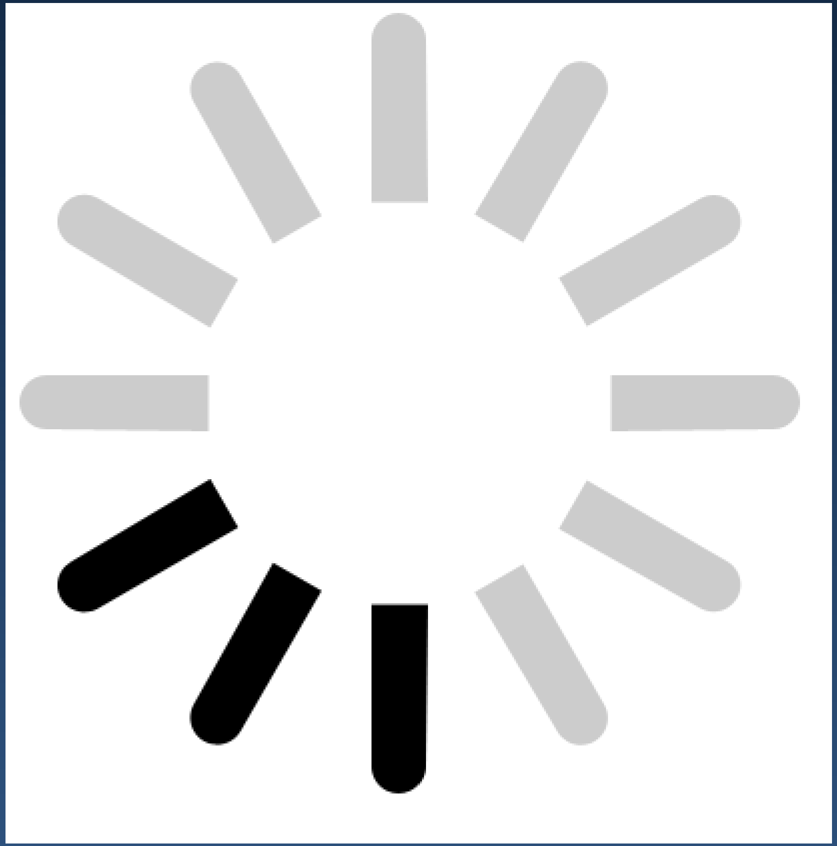
AJAX

Asynchronous

JavaScript

And

XML



AJAX

- AJAX is a set of technologies which incorporates:
  - Standards-based presentation using XHTML and CSS;
  - Dynamic display and interaction using the Document Object Model (DOM);
  - Data interchange and manipulation using XML or JSON
  - Asynchronous data retrieval
  - And JavaScript binding everything together
- AJAX is not a programming language but a technique!

AJAX Demonstration

BBC News

http://www.bbc.co.uk/

Google Maps

http://maps.google.com/

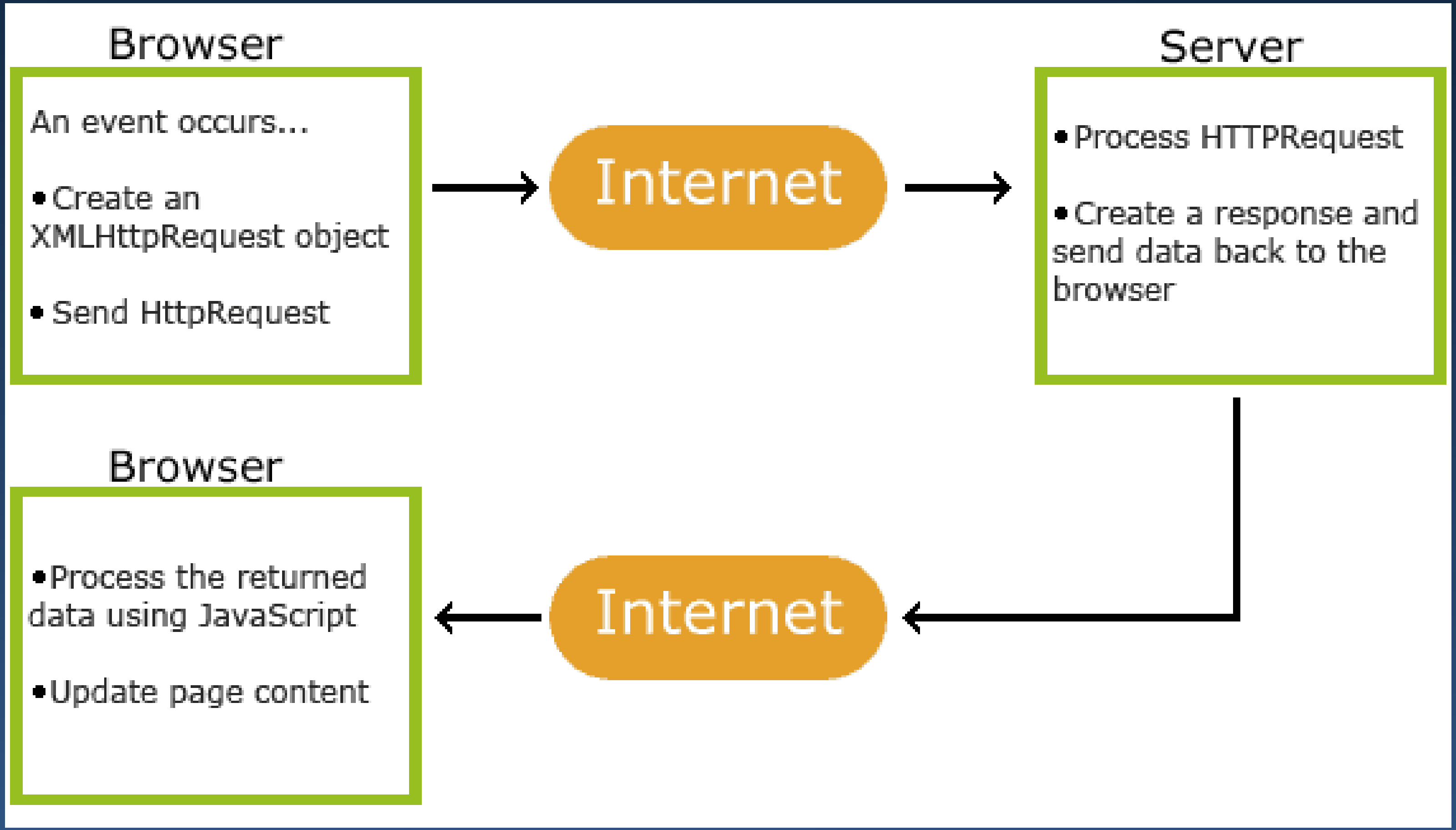
Traditional HTTP Approach

- User clicks a link or presses a form button
- User agent (browser) makes an HTTP request using either GET or POST
- Server processes request and sends a response
- Browser loads the response in a new web page

How does Ajax work?

- Traditional use of HTTP causes pages to be reloaded when data is exchanged
- AJAX uses JavaScript's `XMLHttpRequest` method to exchange data with the server without reloading the current page.
  - This enables quick, small transactions
  - Client can communicate more often with the server
  - More flexible user-interfaces can be built
  - Web applications can look/act more like desktop apps

How does Ajax work?



Browser

An event occurs...

- Create an XMLHttpRequest object
- Send HttpRequest

Internet

Server

- Process HTTPRequest
- Create a response and send data back to the browser

Internet

Browser

- Process the returned data using JavaScript
- Update page content

Javascript <---- AJAX ----> jQuery

```
<script type="text/javascript">
var http = false;
 if(navigator.appName == "Microsoft
Internet Explorer") {
   http = new
ActiveXObject("Microsoft.XMLHTTP");
 } else { http = new
XMLHttpRequest();
 }
function replace() {
 http.open("GET", "script.php",
                             true);
http.onreadystatechange=function()
{
  if(http.readyState == 4) {
     document.getElementById('foo')
    .innerHTML = http.responseText;
  }
}
 http.send(null);
}
</script>
```

```
$.ajax({
    url: ‘script.php',
   type: 'GET',
   dataType: ‘html',
   success: function(html){
// do something with html
         }
       })
```

jQuery

What is jQuery?

- jQuery is a library of JavaScript Functions.
- jQuery is a lightweight "write less, do more" JavaScript library.
- The jQuery library contains the following features:
  - HTML element selections
  - HTML element manipulation
  - CSS manipulation
  - HTML event functions
  - JavaScript Effects and animations
  - HTML DOM traversal and modification
  - AJAX
  - Utilities

Adding the jQuery Library to Your Pages

- The jQuery library is stored a single JavaScript file, containing all the jQuery functions.
- It can be added to a web page with the following mark-up:

```
<head>
   <script src="jquery.js">  </script>
</head>
```

- Please note that the `<script>` tag should be inside the page's `<head>` section.
- Can also use a CDN:

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/2.2.0/jquery.min.js"></script>
```

jQuery Syntax

- The jQuery syntax is tailor made for **selecting** HTML elements and perform some **action** on the element(s).
- Basic syntax is: `$(selector).action()`
- A dollar sign to define jQuery
- A (selector) to "query (or find)" HTML elements
- A jQuery action() to be performed on the element(s)

Query Syntax Examples

- `$(this).hide()`
  For hiding the current HTML element.
- `$("#test").hide()`
  For hiding the element with `id="test"`.
- `$("p").hide()`
  For hiding all `<p>` elements.
- `$(".test").hide()`
  For hiding all elements with `class="test"`.

The Document Ready Function

```
$(document).ready(function(){

    // jQuery functions go here...

});
```

- This is to prevent any jQuery code from running before the document is finished loading (is ready).
- Here are some examples of actions that can fail if functions are run before the document is fully loaded:
  - Trying to hide an element that doesn't exist
  - Trying to get the size of an image that is not loaded

jQuery Selectors

- jQuery selectors allow you to select HTML elements (or groups of elements) by element name, attribute name or by content.
- Selectors allow you to manipulate DOM elements as a group or as a single node.
- Different types of selector:
  - Element Selector
  - Attribute Selector
  - CSS Selector

jQuery Element Selectors

- jQuery uses CSS selectors to select HTML elements.
- `$("p")` selects all <p> elements.
- `$("p.intro")` selects all <p> elements with class="intro".
- `$("p#demo")` selects the first <p> element with id="demo".

jQuery Attribute Selectors

- jQuery uses expressions to select elements with given attributes.
- `$("[href]")` select all elements with an href attribute.
- `$("[href='#']")` select all elements with an href value equal to "#".
- `$("[href!='#']")` select all elements with an href attribute NOT equal to "#".
- `$("[href$='.jpg']")` select all elements with an href attribute that ends with ".jpg".

jQuery CSS Selectors

- jQuery CSS selectors can be used to change CSS properties for HTML elements.
- The following example changes the background-color of all p elements to yellow:

```
$("p").css("background-color","yellow");
```

jQuery Event Functions

- The jQuery event handling functions are core functions in jQuery.
- Event handlers are functions that are called when "something happens" in HTML. The term "**triggered (or "fired") by an event**" is often used.
- It is common to put jQuery code into event handler functions in the <head> section.

jQuery Event Example

```
<html>
<head>
<script src="jquery.js"></script>
<script>
$(document).ready(function(){
  $("button").click(
         function(){
                     $("p").hide();
                   }
         );
});
</script>
</head>

<body>
<h2>This is a heading</h2>
<p>This is a paragraph.</p>
<p>This is another paragraph.</p>
<button>Click me</button>
</body>
</html>
```

Example: http://jsfiddle.net/edo77uk/fL2nx/

jQuery Callback Functions

- A callback function is executed after the current animation (effect) is finished.
- JavaScript statements are executed line by line. However, with animations, the next line of code can be run even though the animation is not finished. This can create errors.
- To prevent this, you can create a callback function. The callback function will not be called until after the animation is finished.

jQuery Callback Example

- Typical syntax: `$(selector).hide(speed,callback)`
- The callback parameter is a function to be executed after the hide effect is completed:

```
$("p").hide(1000,function(){
  alert("The paragraph is now
                      hidden");
});
```

jQuery HTML Manipulation

- jQuery contains methods for changing and manipulating HTML elements and attributes.
- Changing HTML Content
  - `$(selector).html(content)` The html() method changes the contents (innerHTML) of matching HTML elements.
  - Example: `$("p").html("CS1518");`
- Adding HTML content
  - `$(selector).append(content)` The append() function appends content to the inside of matching HTML elements.
  - Example: `$("p").append(" CS1518");`

AJAX and jQuery

- jQuery provides a rich set of methods (functions) for AJAX web development.
- With jQuery AJAX, you can request TXT, HTML, XML or JSON data from a remote server using both HTTP Get and HTTP Post.
- And you can load remote data directly into selected HTML elements of your web page.
- The jQuery load function is a simple AJAX function. It has the following syntax:

```
$(selector).load(url,data,callback)
```

Low Level AJAX

- `$.ajax(options)` is the syntax of the low level AJAX function.
- `$.ajax` offers more functionality than higher level functions like load, get, and post, but it is also more difficult to use.
- The option parameter takes <name,value> pairs defining url data, passwords, data types, filters, character sets, timeout and error functions.

jQuery AJAX Example

```
<html>
<head>
<script src="jquery.js"></script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    htmlobj=$.ajax({url:"script.php",async:false});
    $("div").html(htmlobj.responseText);
  });
});
</script>
</head>
<body>

<div><h2>Let AJAX change this text</h2></div>
<button>Change Content</button>
</body>
</html>
```

jQuery AJAX Requests

Request	Description
`$(selector).load(url,data,callback)`	Load remote data into selected elements
`$.ajax(options)`	Load remote data into an XMLHttpRequest object
`$.get(url,data,callback,type)`	Load remote data using HTTP GET
`$.post(url,data,callback,type)`	Load remote data using HTTP POST
`$.getJSON(url,data,callback)`	Load remote JSON data using HTTP GET
`$.getScript(url,callback)`	Load and execute a remote JavaScript file

More about jQuery

http://jquery.com/

With jQuery, look at the following selector: `$("div.intro")`. What does it select?

A. The **first** `div` element with `id="intro"`

B. All `div` elements with `class="intro"`

C. The **first** `div` element with `class="intro"`

D. All `div` elements with `id="intro"`