AJAX

Web Application Development

WAD

HYPE, MECHANICS, DEMOS

The Hype

In 2006, an Amazon.co.uk search for ‘ajax’
under Books / Computer and Internet returned

52 books

By 2010 the same search returned

1417 books

In 2014…

3000+ Books

ean up the speed and appearance of your
eb applications with Ajax

Ajax

FOR

DUMMI

Code an
Ajax
on com

A Reference
for the
Rest of Us!®

EE eTips at dummies.com®

eve Holzner, PhD
hor of Physics For Dummies

Making Everything Easier!™

JavaScript® & AJAX

FOR

DUMMIES®

Learn to:

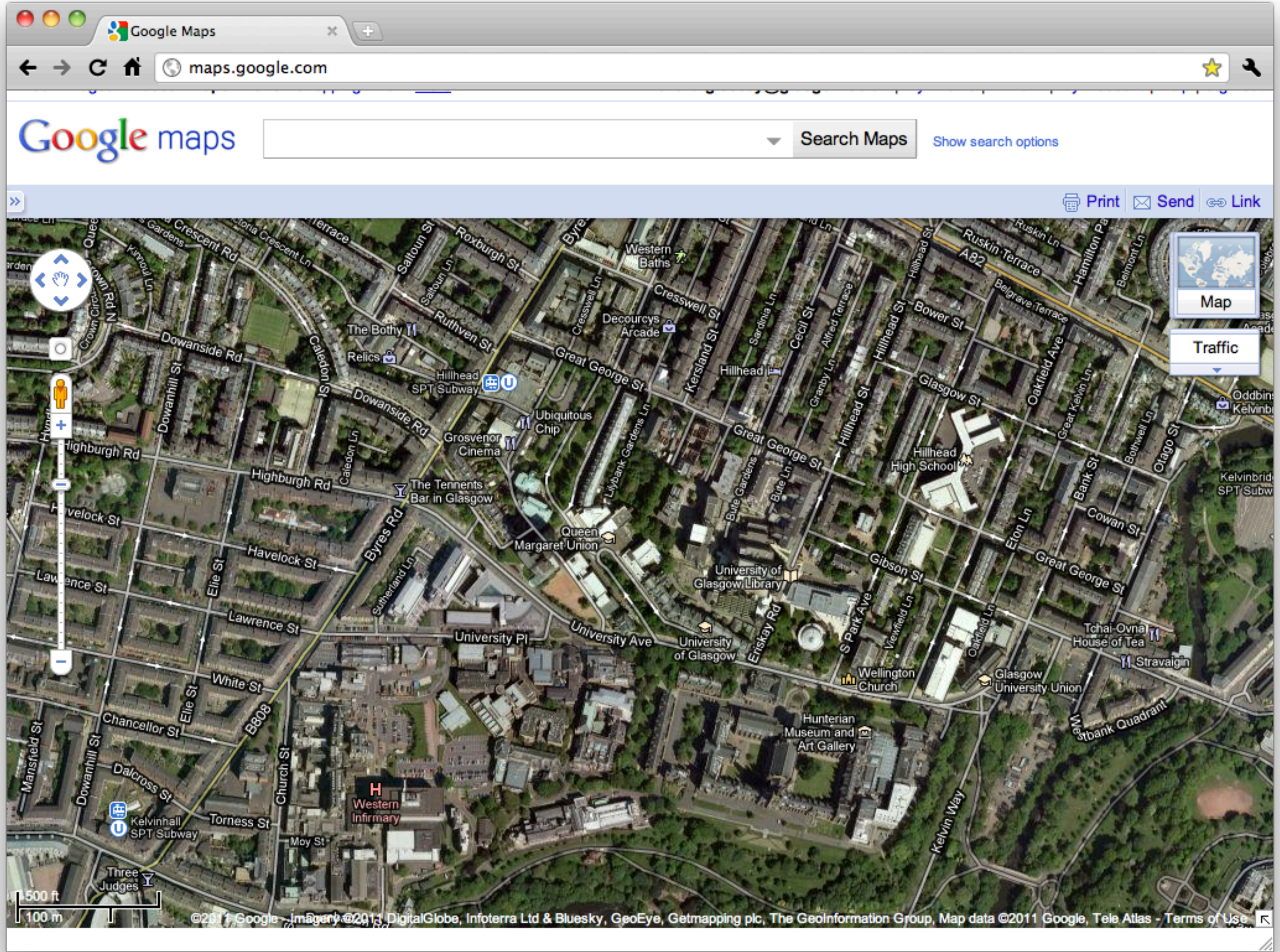
- Master basic JavaScript as a Web design and application development tool
- Write your own programs
- Use JavaScript with AJAX, XML, and JSON
- Design an interface, animate images, program menus, and manage cookies

Andy Harris
Author of HTML, XHTML, and CSS
All-in-One For Dummies

Ajax

- A key technology set underlying web apps
  - **Asynchronous JavaScript** and **XML**

- Critically, all of the components have existed in some form since the late 1990's
  - An example of where browsers introducing non-standard features has been a positive thing



Google Maps
maps.google.com
Google maps
Search Maps
Show search options
Print
Send
Link
Map
Traffic
Western Baths
Cresswell St
Great George St
Hillhead SPT Subway
Ubiquitous Chip
Grosvenor Cinema
The Tennents Bar in Glasgow
Queen Margaret Union
University of Glasgow Library
University Ave
University Pl
University of Glasgow
Wellington Church
Glasgow University Union
Hunterian Museum and Art Gallery
Hillhead High School
Western Infirmary
Kelvinhall SPT Subway
Three Judges
Byres Rd
Highburgh Rd
Dowanside Rd
Havelock St
Lawrence St
White St
Chancellor St
Dalcross St
Torness St
Gibson St
Glasgow St
Kelvin Way
Westbank Quadrant
Stravaigin
Tchai-Ovna House of Tea
500 ft
100 m
©2011 Google - Imagery ©2011 DigitalGlobe, Infoterra Ltd & Bluesky, GeoEye, Getmapping plc, The GeoInformation Group, Map data ©2011 Google, Tele Atlas - Terms of Use

Ajax

- Jesse James Garret coined the term in his 2005 essay:
  - "*Ajax: A New Approach to Web Applications*"
  - http://www.adaptivepath.com/ideas/essays/archives/000385.php

- Generated an enormous amount of hype and energy in web development ever since

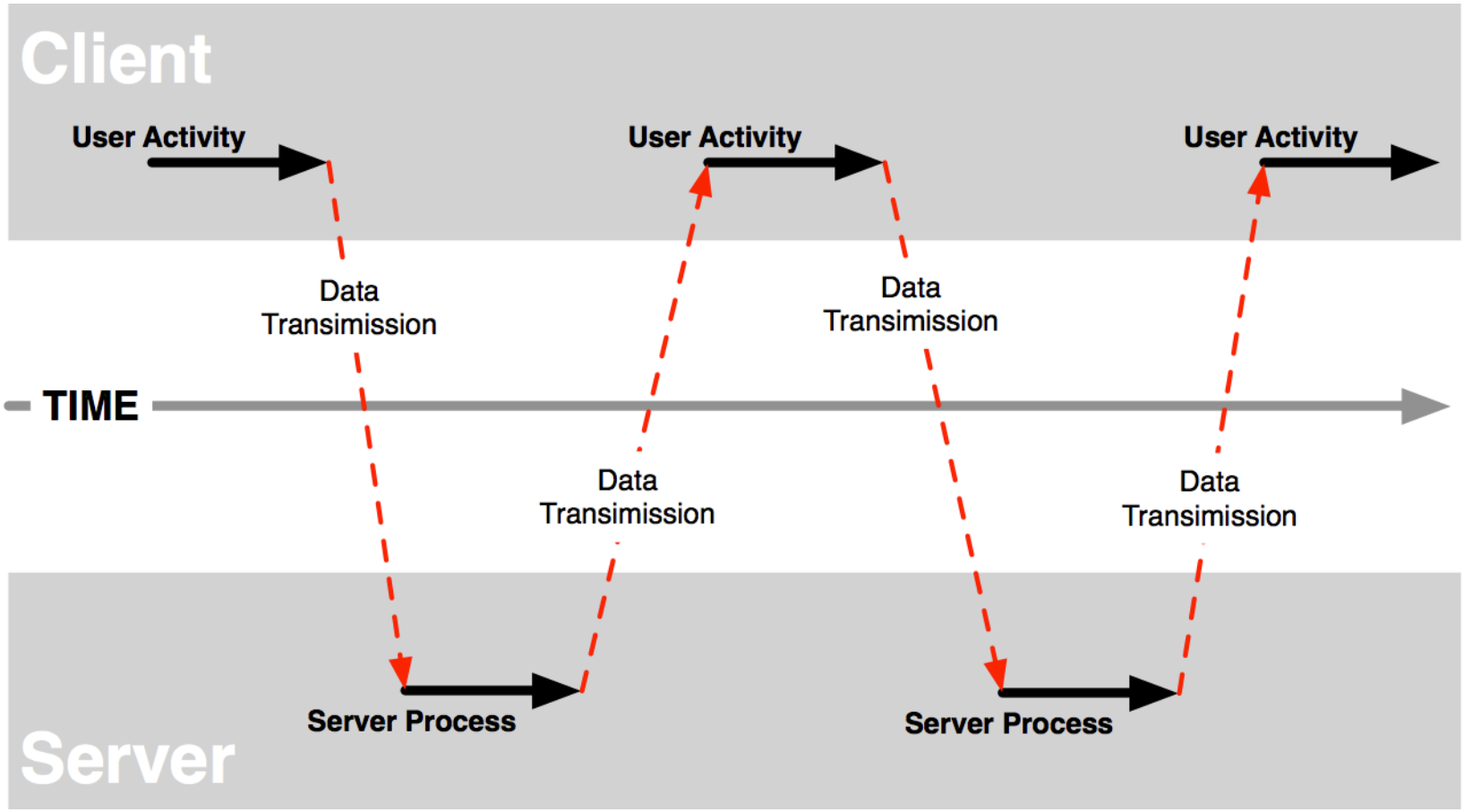
Technology Set

- standards-based presentation using XHTML and CSS;
- dynamic display and interaction using the Document Object Model;
- data interchange and manipulation using XML and XSLT;
- asynchronous data retrieval using XMLHttpRequest;
- and JavaScript binding everything together

What does Ajax do?

- Ajax **eliminates the need to reload** a web page in order to get new content from the server
- This removes the **start-stop interaction** where a user has to wait for new pages to load.
- An intermediate layer (**Ajax Engine**) is introduced into the communication chain between client and server


The
Mechanics



Traditional Client/Server Synchronous Communication Model

Ajax Components

JavaScript can manipulate the DOM of a webpage to **create, modify and remove content and style**

JavaScript event handlers can be attached to **events generated by the user** and browser

XML can **model data** and we can access it using DOM

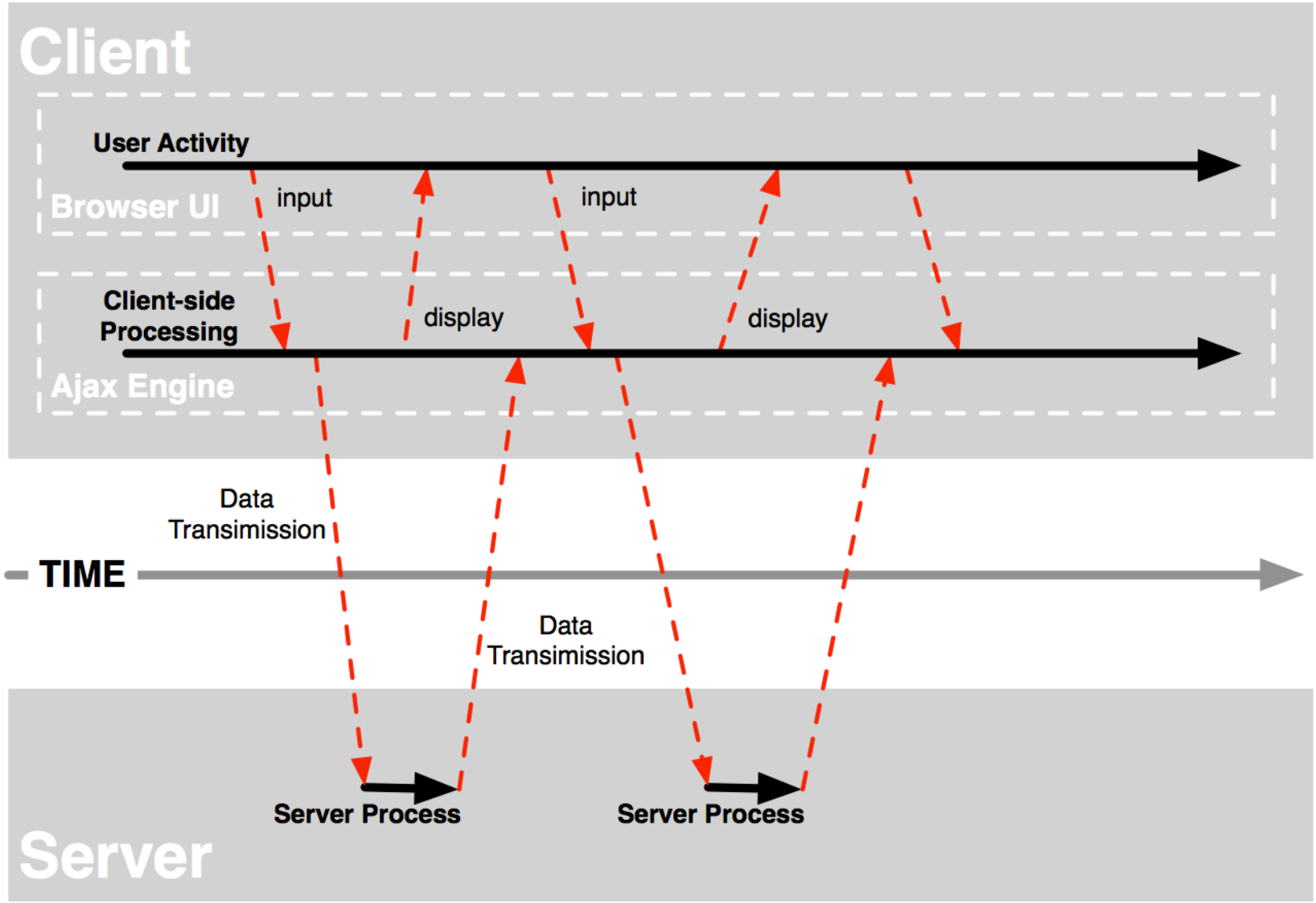
So how does Ajax achieve asynchronous interaction and communication with the server?

XmlHttpRequest Object

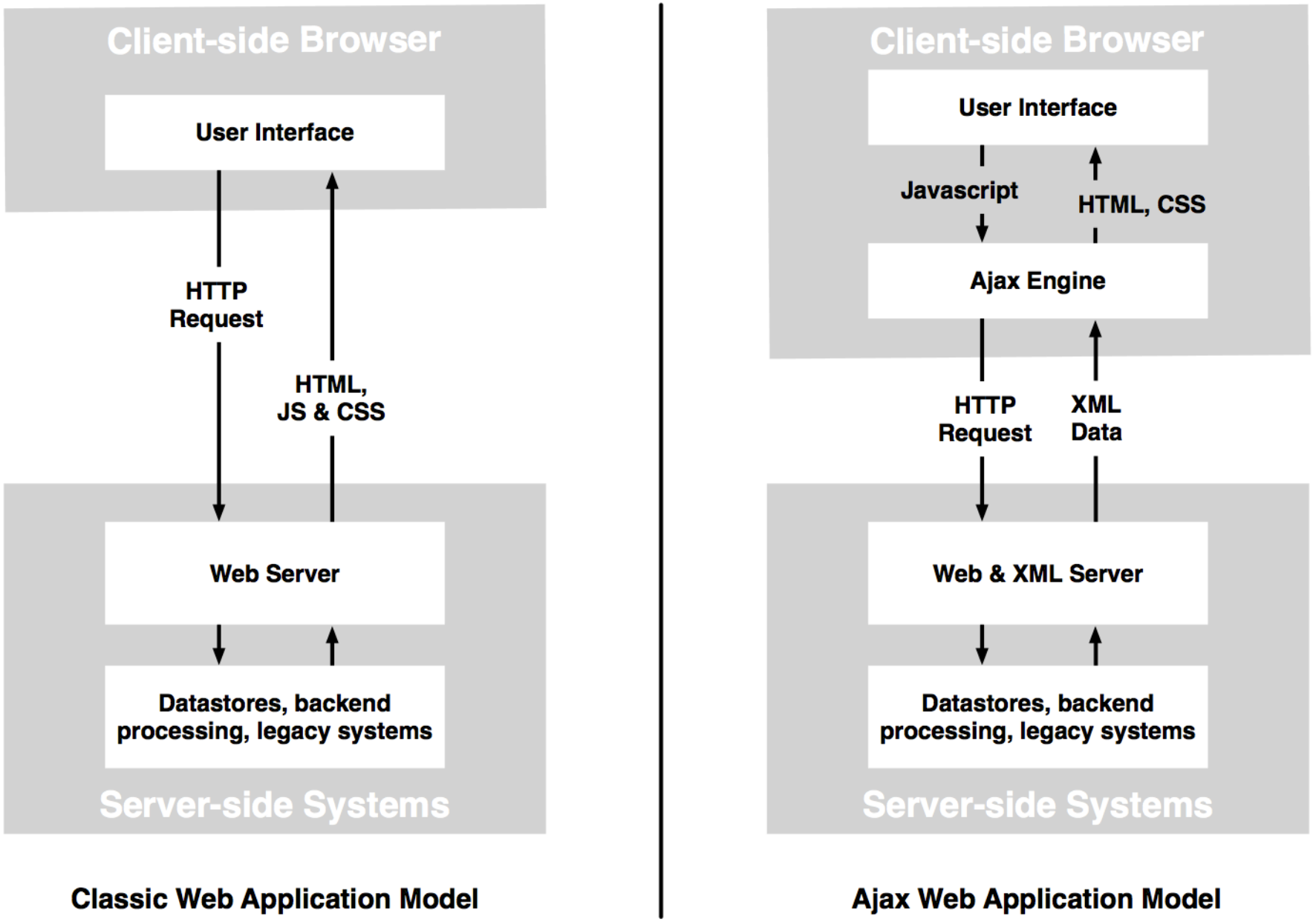
Introduced by Microsoft in Internet Explorer 5

The XHR is an object that is part of the DOM and is built into most modern browsers

It can communicate with the server by sending HTTP requests (much like normal client/server communication)



AJAX Client/Server Asynchronous Communication Model



Architectural Comparison

XmlHttpRequestObject

- Independent of <form> or <a> elements for generating HTTP GET/POST requests
- It does not block script execution after sending an HTTP request
- As with **content** and **style**, JavaScript can now programmatically manage **HTTP communication**

XHR Properties

- **readyState** property
  - cycles through several states as it sends an HTTP request to the server, waits whilst the request is processed, and when it receives a response from the server (values 0-4)
- **onreadystatechange** property
  - accepts an EventListener value, specifying the method that the object will invoke whenever the readyState value changes
- **status** property
  - The status property represents the HTTP status code and is of type short (e.g. 200 = OK, 404 = Not Found)

XHR Properties

- **responseXML** property
  - represents the XML response data when the complete HTTP response has been received (when readyState is 4), and when the Content-Type header specifies the MIME (media) type as text/xml, application/xml, or ends in +xml
- **responseText** property
  - contains the text of the HTTP response received by the client
  - XML is not the only method to model data in Ajax applications. A popular alternative is JSON (JavaScript Object Notation)

responseXML

Simple XML to model an address book entry

```
<Person>
    <firstName>John</firstName>
    <lastName>Smith</lastName>
    <age>25</age>
    <address>
        <streetAddress>21 2nd Street</streetAddress>
        <city>New York</city>
        <state>NY</state>
        <postalCode>10021</postalCode>
    </address>
    <phoneNumber type="home">212 555-1234</phoneNumber>
    <phoneNumber type="fax">646 555-4567</phoneNumber>
    <companyName />
</Person>
```

responseText (JSON)

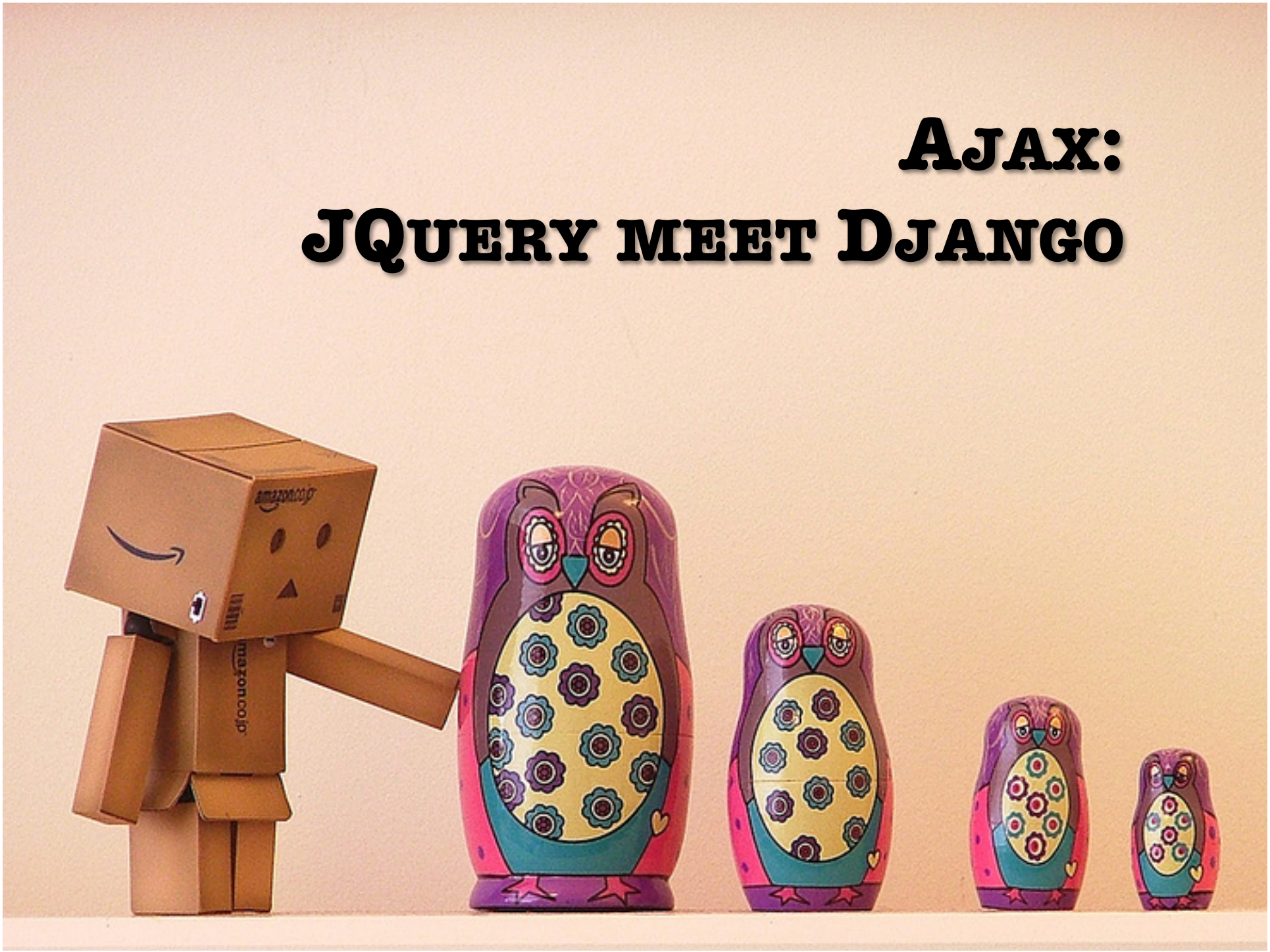
- Same data as before, but uses fewer characters:

```
{
    "firstName": "John",
    "lastName": "Smith",
    "age": 25,
    "address": {                                        // address object
        "streetAddress": "21 2nd Street",
        "city": "New York",
        "state": "NY",
        "postalCode": "10021"
    },
    "phoneNumbers": [                                   // array of objects
        { "type": "home", "number": "212 555-1234" },
        { "type": "fax", "number": "646 555-4567" }
    ],
    "companyName": null
}
```

AJAX

- # Eliminates the need to reload complete pages
- # Stop-Start Interaction no longer interrupts the user
- # Improves the interactive experience in web apps

Ajax:
jQuery meet Django



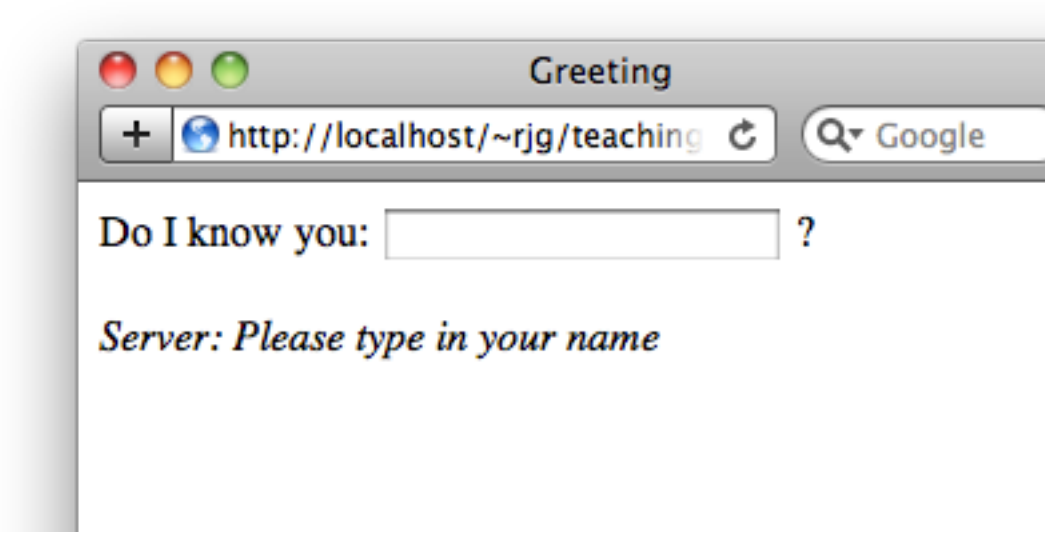
AJAX IN ACTION

Ajax Example – c. 2010

- Webpage that recognizes a user's name
  - **client-side script communicates** what the user is typing **to a server-side script**
  - server-side script checks the user input against it's own list of names to match
  - web page is **updated continuously** to indicate if there is a match or not (without reloading)

HTML

```
<html>

<head>
    <title>Greeting</title>
    <script type="text/javascript" src="engine.js"></script>
</head>

<body onload='process( );'>
    Do I know you:
    <input type="text" id="myName" /> ?
    <br><br>
    <div id="divMessage" />
</body>

</html>
```



JavaScript

```
// variable for XmlHttpRequest object
var xmlhttp = createXmlHttpRequestObject( );

// retrieves the XMLHTTPRequest object (if supported)
function createXmlHttpRequestObject( ) {
    var xmlhttp;
    try {
        xmlhttp = new XMLHttpRequest();
    }
    catch (e) {
        xmlhttp = false;
    }

    if (!xmlhttp)
        alert("Error creating the XMLHTTPRequest object");
    else
        return xmlhttp;
}
```

get reference to XmlHttpRequest Object

JavaScript

```
// make asynchronous HTTP request using XMLHTTPRequest object
function process( ) {
    // proceed only if the xmlhttp object is not busy
    if (xmlhttp.readyState == 4 || xmlhttp.readyState == 0) {
        // retrieve name typed by user from text field
        name = document.getElementById("myName").value;

        // execute the greeting.php script on the server
        xmlhttp.open("GET", "greeting.php?name="+name, true);

        // define the method to handle the server responses
        xmlhttp.onreadystatechange = handleServerResponse;

        // make the server request
        xmlhttp.send(null);
    }
    else {
        // if connection is busy, try again after one second
        setTimeout('process()', 1000);
    }
}
```

communicate with server asynchronously

PHP

```php
<?php
	// declare mime-type output as XML
	header('Content-Type: text/xml');
	//  generate XML header
	echo '<?xml version="1.0" encoding="UTF-8" standalone="yes"?>';
	// create the <response> element
	echo '<response>';
	// retrieve the user name
	$name = $_GET['name'];
	// generate output depending on the user name
	$users = array('RICHARD', 'JAMES', 'BOB');

	if (in_array(strtoupper($name), $users)) {
		echo 'Server: Hello, ', htmlentities($name) . '!!';
	}
	else if (trim($name) == '') {
		echo 'Server: Please type in your name';
	}
	else {
		echo 'Server: ' . htmlentities($name) . ', I do not know you!';
	}
	echo '</response>';
?>
```

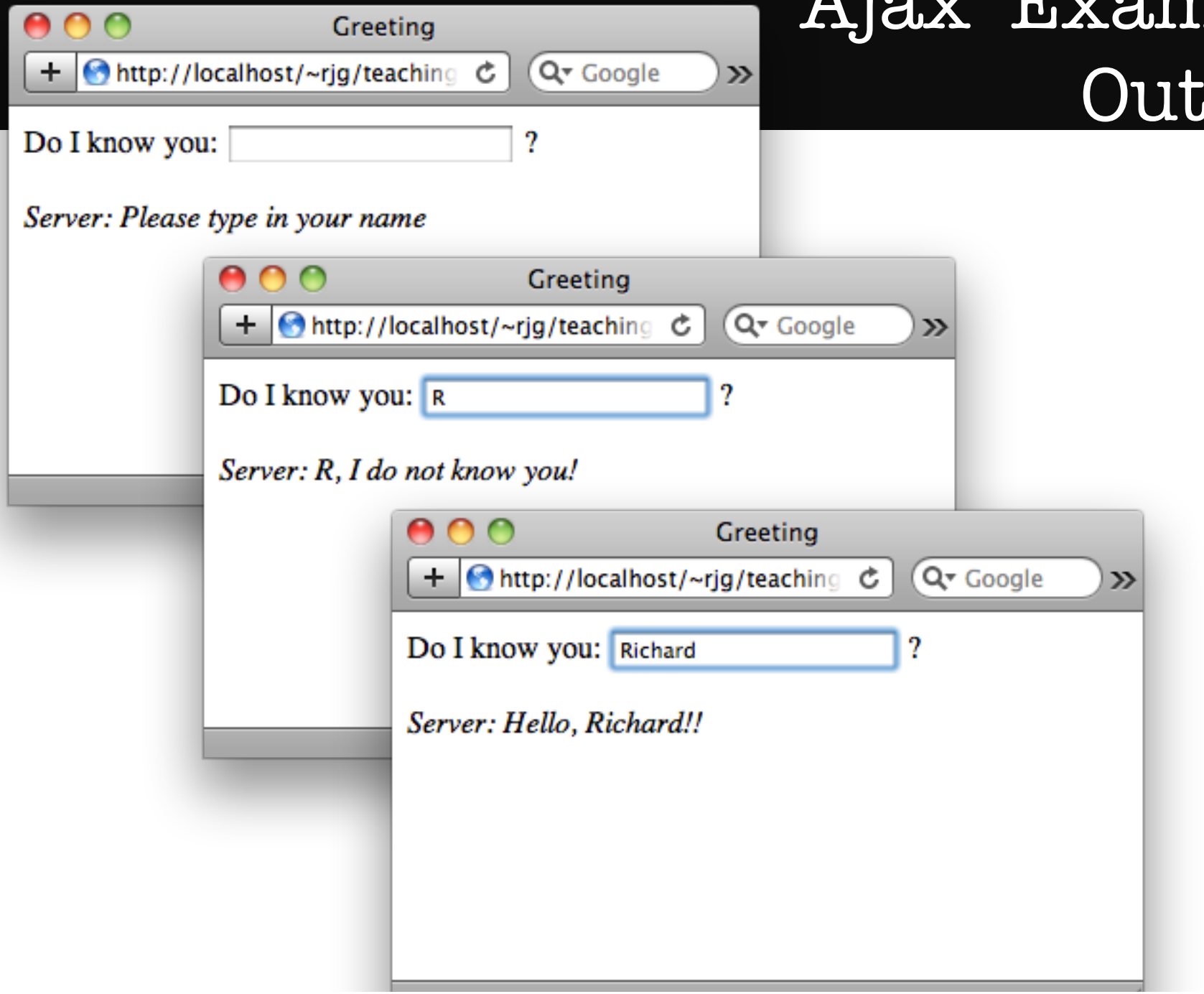
build response message in XML

JavaScript

```
function handleServerResponse() {
    // continue only if transaction has completed
    if (xmlhttp.readyState == 4) {
        // status code 200 indicates successful transaction
        if (xmlhttp.status == 200) {
            // extract the XML retrieved from the server
            xmlResponse = xmlhttp.responseXML;

            // obtain the document element of the XML structure
            xmlDocumentElement = xmlResponse.documentElement;

            // get the message (first child of the document element)
            helloMessage = xmlDocumentElement.firstChild.data;

            // update the client display using data received from the server
            document.getElementById("divMessage").innerHTML = '<i>' +
            helloMessage + '<i>';

            // restart the process( ) function
            setTimeout('process( )', 1000);
    }
    else
        alert("There was a problem:" + xmlhttp.statusText);
  }
}
```

handle XML response from server, use DOM to extract message from XML, use innerHTML to add content to our HTML

Ajax Example Output

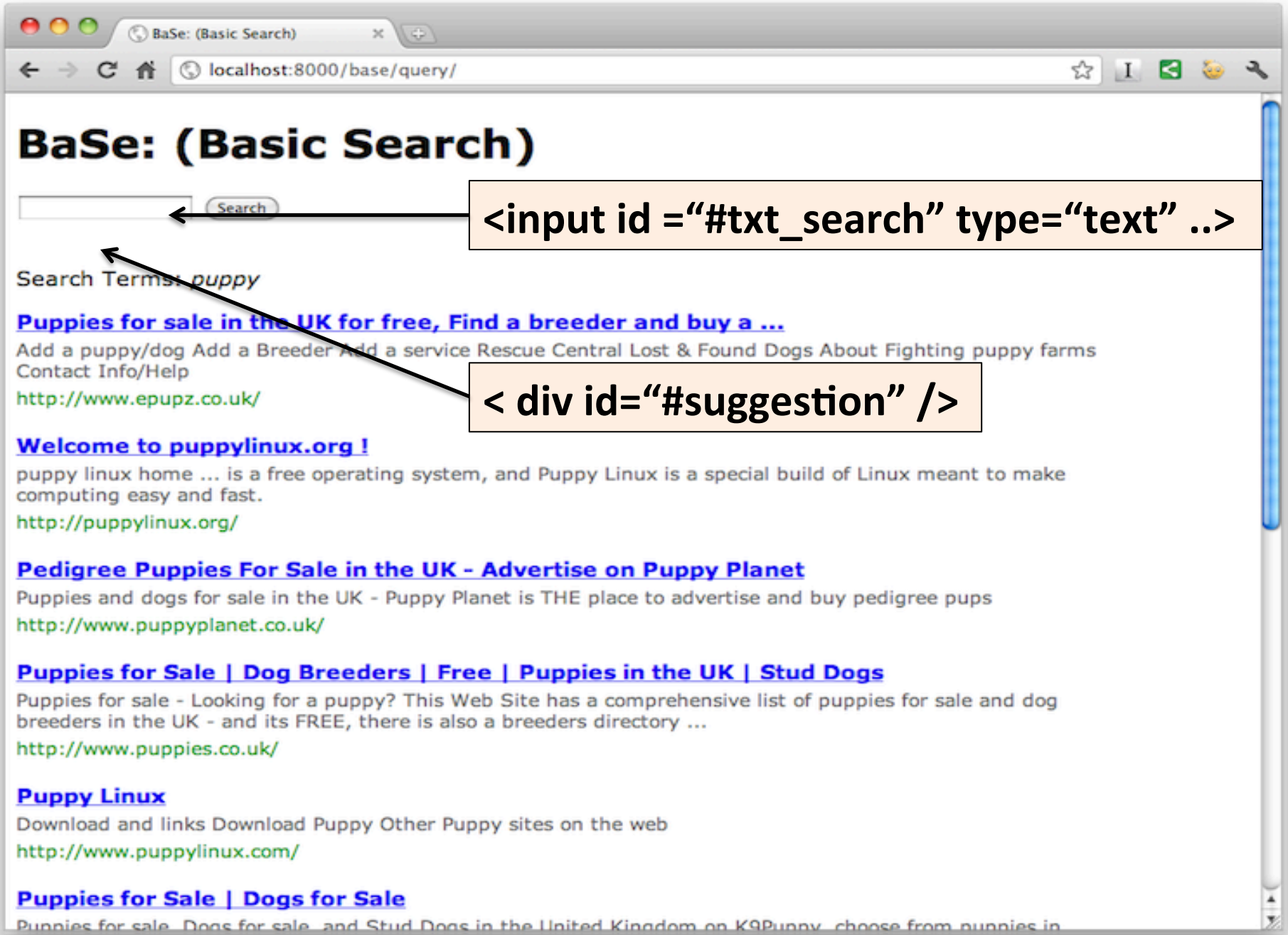


jQuery AJAX

Simplifying AJAX in web applications

```
$.ajax({type: ”GET",
        url: ”greeting.php",
        data: ”name=" + name,
        success: function(message) {
          // do something
        }
});
```

SEARCH SUGGESTION EXAMPLE


BaSe: (Basic Search)
localhost:8000/base/query/
BaSe: (Basic Search)
Search
<input id =“#txt_search” type=“text” ..>
Search Terms: puppy
< div id=“#suggestion” />
Puppies for sale in the UK for free, Find a breeder and buy a ...
Add a puppy/dog Add a Breeder Add a service Rescue Central Lost & Found Dogs About Fighting puppy farms Contact Info/Help
http://www.epupz.co.uk/
Welcome to puppylinux.org !
puppy linux home ... is a free operating system, and Puppy Linux is a special build of Linux meant to make computing easy and fast.
http://puppylinux.org/
Pedigree Puppies For Sale in the UK - Advertise on Puppy Planet
Puppies and dogs for sale in the UK - Puppy Planet is THE place to advertise and buy pedigree pups
http://www.puppyplanet.co.uk/
Puppies for Sale | Dog Breeders | Free | Puppies in the UK | Stud Dogs
Puppies for sale - Looking for a puppy? This Web Site has a comprehensive list of puppies for sale and dog breeders in the UK - and its FREE, there is also a breeders directory ...
http://www.puppies.co.uk/
Puppy Linux
Download and links Download Puppy Other Puppy sites on the web
http://www.puppylinux.com/
Puppies for Sale | Dogs for Sale

Views.py

```
def suggest(request):
    """generate search suggestions"""
    # get suggestion so far from post
    search = request.POST['search']
    # search for pattern from list
    suggestion = ""
    suggestion_list = ["puppy dog", "cats hate
dogs", "raining cats and dogs"]
    for s in suggestion_list:
      if s.startswith(search):
        suggestion = s
    # return suggestion
    response = HttpResponse(suggestion)
    return response
```

Jquery Code

```
$(document).ready(function() {
     $("#txt_search").keyup(function() {
          var search = $("#txt_search").val();
          if (search.length > 0) {
               $.ajax({type: "POST",
                    url: "suggest/",
                    data: "search=" + search,
                    success: function(message) {
                      $("#suggest").empty();
                      if (message.length > 0) {
                        message = "Do you mean: " + message + "?";
                        $("#suggest").append(message);
                      }
                    }
            });
          } else {
            // Empty suggestion list
            $("#suggest").empty();
          }
        });
      });
```