



# **COMP 4021**

# **Internet Computing**

## **AJAX**

Dr. Kenneth LEUNG

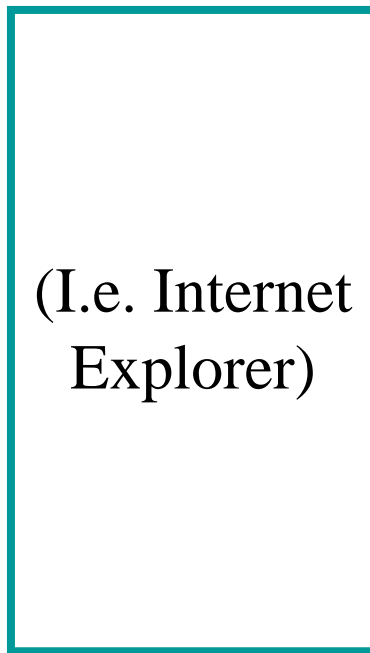
Slides created by Dr. David ROSSITER

# 'Traditional' Communication

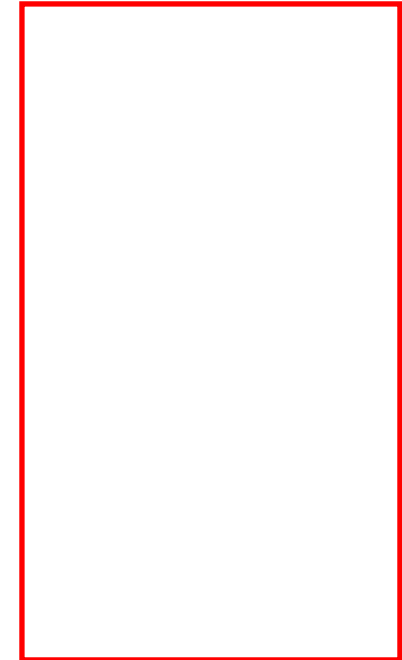
- You already know that the most common way to give information to a server is by using a form
- Typically, the server then sends a web page back to the browser
- With this method, the transaction happens 'once only'
- This is the 'traditional' style of handling information on the Internet for the past 20 years

# 'Traditional' Operation

Client



Server



*1. Client sends  
request i.e.  
GET index.html*



*2. Server sends  
new web page*



*or maybe redirects  
the browser to a  
web page*

*3. New web page is  
shown in the browser*

# Updating an Entire Web Page

- To periodically update information on a web page automatically, you can do either of these:
  - Method 1: Refresh the web page (or go to another web page) using an html 'meta' tag
  - Method 2: Use JavaScript to tell the browser to reload the web page (or go to another web page)
- With both methods, even if only small parts of the page need to be updated, the whole thing must be updated

# Method 1 – Using a Meta Tag

- The HTML refresh meta tag can be used to redirect a web page after a certain time interval

`<META http-equiv="refresh" content="<time>;URL=<url>" />`

- For example the following tag redirects the browser to the 303 main page 10 seconds after loading

`<META http-equiv="refresh"  
content="10;URL=http://course.cse.ust.hk/comp303" />`

- If the URL field is empty the page will refresh itself after the specified time value, e.g.

`<META http-equiv="refresh" content="5" />`

*Refresh the same page after 5 seconds*

## Method 2 – Using JavaScript

- You can use JavaScript to refresh the page content
- The following example refreshes a web page after 5 seconds

```
<html>  
  <body onload=  
    "setTimeout('window.location.reload()', 5000)">  
    Hello, I will be reloaded in 5 seconds.  
  </body>  
</html>
```

# AJAX

- Ajax uses a different style of communication to the 'load a complete page' style
- Ajax = **A**synchronous **J**avaScript **a**nd **X**ML
- It is a method for code in a web page to send a request to the server any time it wants to, without 'leaving' the current web page
- This is called 'asynchronous communication'
- Typically a small message is sent back from the server to the browser (not a complete web page)

# Example Use of Ajax - Google

- Example - interactive entering of search terms:  
<http://www.google.com>
- As soon as you enter a letter in the google search field, all the letters entered by the user so far are sent to the Google server
- Google immediately responds with a list of the most popular search terms that start with those letters (user can then use arrow keys to select one)



a	
amazon	11,510,000,000 results
argos	16,500,000 results
amazon.com	641,000,000 results
aol	1,280,000,000 results
ask jeeves	575,000,000 results
aol.com	300,000,000 results
apple	1,780,000,000 results
ask.com	1,140,000,000 results
ask	5,700,000,000 results
autotrader	148,000,000 results

aj	
ajax	147,000,000 results
ajc	114,000,000 results
ajc.com	1,600,000 results
aj wright	63,100,000 results
ajax tutorial	25,800,000 results
ajilon	2,780,000 results
aj.com	73,300,000 results
ajit	5,050,000 results
ajb	4,530,000 results
aja	297,000,000 results

ajax	
ajax tutorial	25,800,000 results
ajax tutorials	51,400,000 results
ajax examples	3,790,000 results
ajaxian	2,230,000 results
ajax framework	33,900,000 results
ajax php	138,000,000 results
ajax example	17,800,000 results
ajax4jsf	199,000 results
ajax asp.net	9,170,000 results
ajax amsterdam	3,530,000 results

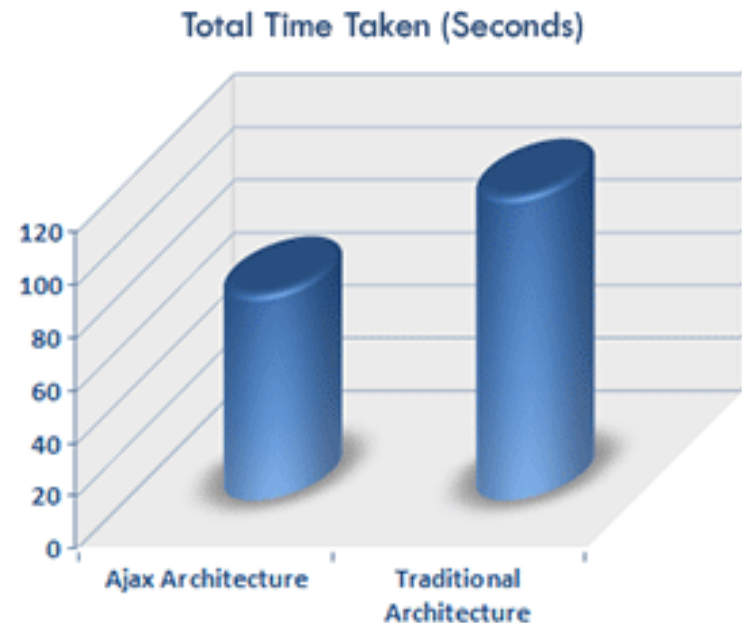
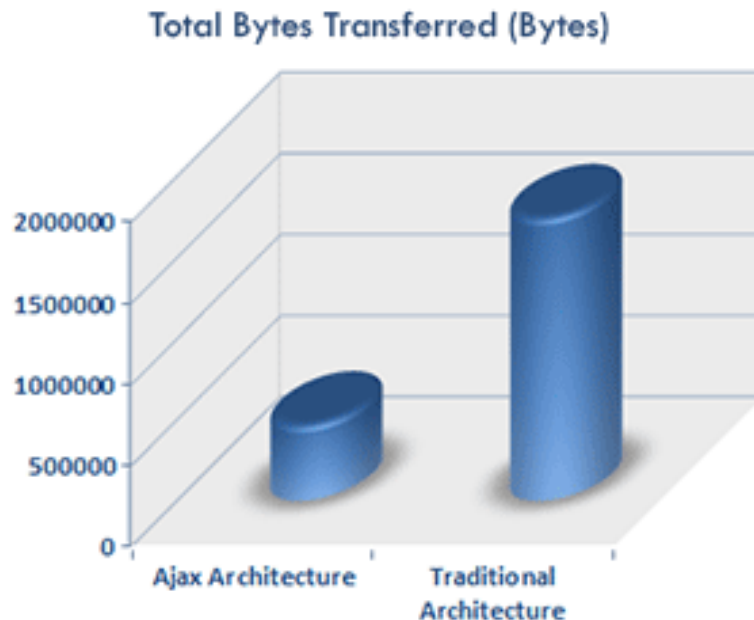
ajal	
ajax	147,000,000 results
ajax tutorial	25,800,000 results
ajay devgan	776,000 results
ajax tutorials	51,400,000 results
ajax examples	3,790,000 results
ajay	124,000,000 results
ajaxian	2,230,000 results
ajanta	898,000 results
ajax framework	33,900,000 results
ajax php	138,000,000 results

# Speed Advantages of AJAX

- Usually the request sent to the server is small in size (i.e. <150 bytes)
- Also, the response from the server to the client is small (maybe <200 bytes)
- So these small sizes mean quick communication
- Another point is that usually when the data is received the JavaScript code only updates a small part of the web page, not the entire display
- This is much quicker than updating the entire web page display, which is the 'traditional' style

# Advantages - Efficiency

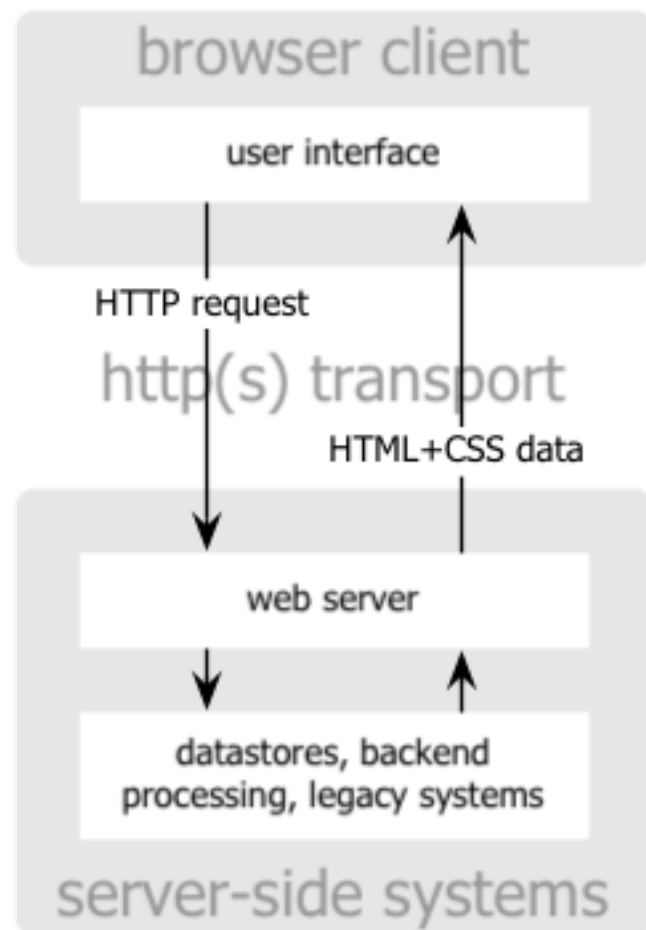
- Here is some data from one real-world system comparing a traditional approach with an Ajax approach



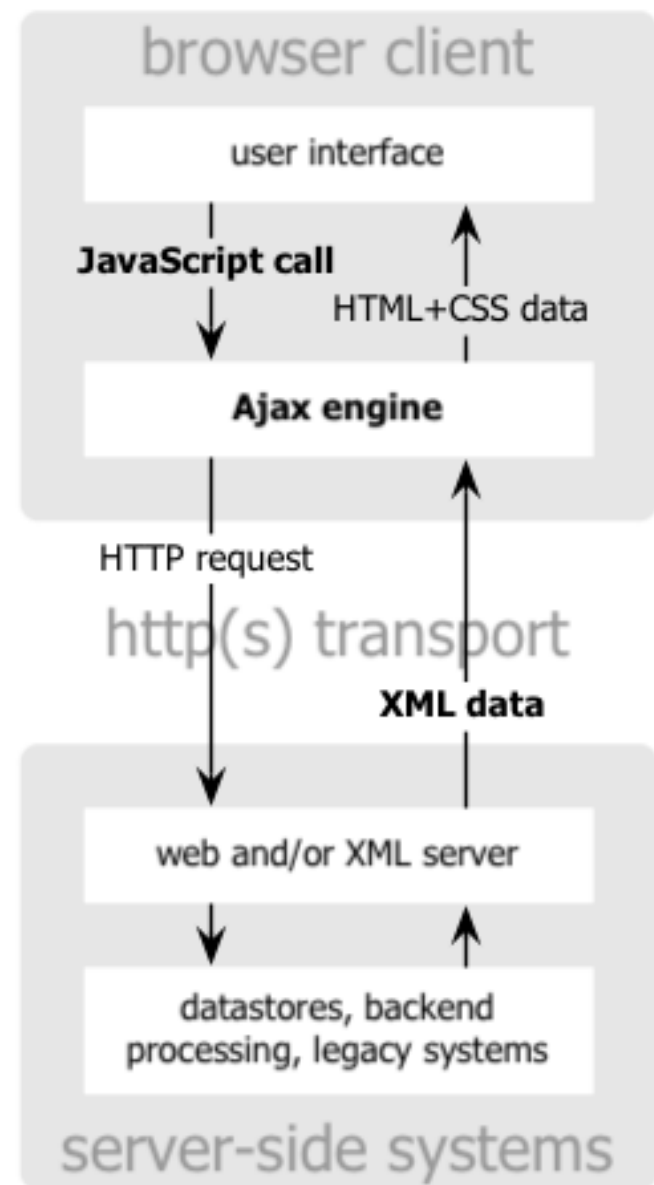
<http://www.ajaxinfo.com/default~viewart~18.htm>

# Ajax/Traditional Comparison

- The next slide shows a comparison between the 'traditional' method (left) and the Ajax method (right)
- Some things in the figure are slightly misleading i.e.
  - Both methods don't *have* to use a CSS file
  - Don't *have* to send XML data for the Ajax method, even simple text could be sent
  - Anything could be on the server, i.e. server may or may not have a database



classic  
web application model



Ajax  
web application model

# Client Side Programming for Ajax

- Create an XMLHttpRequest object
- Send the request to the server
  - `open()` to initialize connection to the server
  - `send()` to send the data
- A client side event listener knows when the XMLHttpRequest object has finished retrieving data, and runs a callback function
- The callback function
  - Checks whether the data is successfully retrieved
  - Does something appropriate with the data

# Basic Ajax Code

- Here the request sent to the server doesn't pass any data (which is unusual)
- This approach uses an Active X object, so it only works on IE

```
function getUpdate() {  
    request = new XMLHttpRequest("Microsoft.XMLHTTP");  
    request.onreadystatechange = stateChange; // see below  
    request.open("POST", "server.php", true);  
    request.send( null );  
}
```

No parameters are passed to the server program

Send the Ajax request to this PHP program

Here we use the POST method, can use the GET method if you like

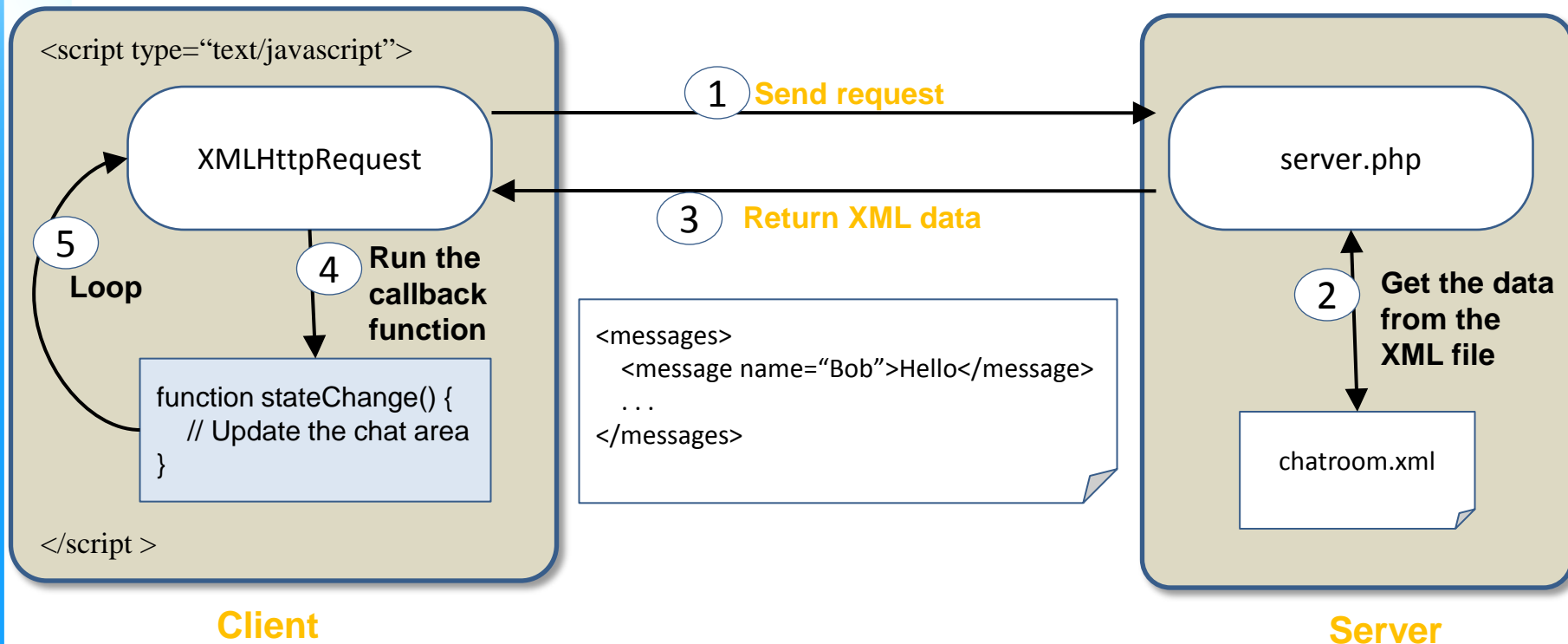
```
function stateChange() { // Callback function  
    // This function is run when the server responds  
    if (request.readyState == 4 && request.status == 200  
        && request.responseText) {  
        xmlDoc = new XMLHttpRequest("Microsoft.XMLDOM");  
        xmlDoc.loadXML(request.responseText);  
        // At this stage the XML is stored in a DOM structure.  
        // For example, the root node is xmlDoc.firstChild.  
    }  
}
```

We only need to run getUpdate() once. After that, whenever a response is received from server.php, stateChange() will be automatically executed.

'4' means a reply has been received from the server. '200' means the reply was normal, the server did not return an error. The third test is to check some actual data was returned by the server.

# A3 – Getting XML Data from the Server

- For A3, the client uses AJAX to get XML data from the server
- A request is sent to the server; XML data is sent back
- When we send the request, we send the last known size of the XML data, so the server can compare it to the current size, and send all the XML chat room data if the size is not the same





## A3 Project Code

- In the previous example code, we only needed to send the request once (in `getUpdate()`)
- After that, whenever a response is received from `server.php`, `stateChange()` will be automatically executed
- For our A3 project, the situation is a bit different, because each time we need to tell the server the size of the chat room XML in the client
- So that means we have to repeatedly send the request

```
var datasize = 0; |
```

The size of the XML data in the client is 0  
when the chat room client side code starts

```
function getUpdate() {  
    request = new ActiveXObject("Microsoft.XMLHTTP");  
    request.onreadystatechange = stateChange;  
    request.open("POST", "server.php", true);  
    request.setRequestHeader("Content-type",  
        "application/x-www-form-urlencoded");  
    request.send("datasize=" + datasize); } |
```

Send the size of the XML data  
in the client to the server  
The server.php compares this  
value with the size of the  
current XML data to check if  
the client has the latest XML  
data, and send it if required

```
function stateChange() { // Callback function  
    // This function is run when data is returned  
    if (request.readyState == 4 && request.status == 200  
        && request.responseText) {  
        xmlDoc = new ActiveXObject("Microsoft.XMLDOM");  
        xmlDoc.loadXML(request.responseText);  
        datasize = request.responseText.length; |  
        ... // XML received, update the chat display  
        getUpdate(); |  
    } }
```

Update the current size of the  
XML data for sending later

Repeat the whole procedure again