CS 416
Web Programming

AJAX

Dr. Williams
Central Connecticut State University

Input revisited - it's magic!

- Once of the input types we didn't touch upon earlier was hidden input

```
<input type="hidden" name="name"
  value="Chad"/>
```

- Form element doesn't show up on page but value of element gets submitted with rest of form
- Why do I care?
  - Very useful for keeping track of identifier (or other info) without having to show it to the user!

Topics for Lecture

- What is AJAX
- Walkthrough of making a server based dynamic page

What is AJAX

- Asynchronous JavaScript Technology
- It allows web pages to be dynamic based on server content
- This is accomplished through using Javascript to make a connection to the server and rewriting portions of the page based on the response
- Based on internet standards so works across browsers and platforms

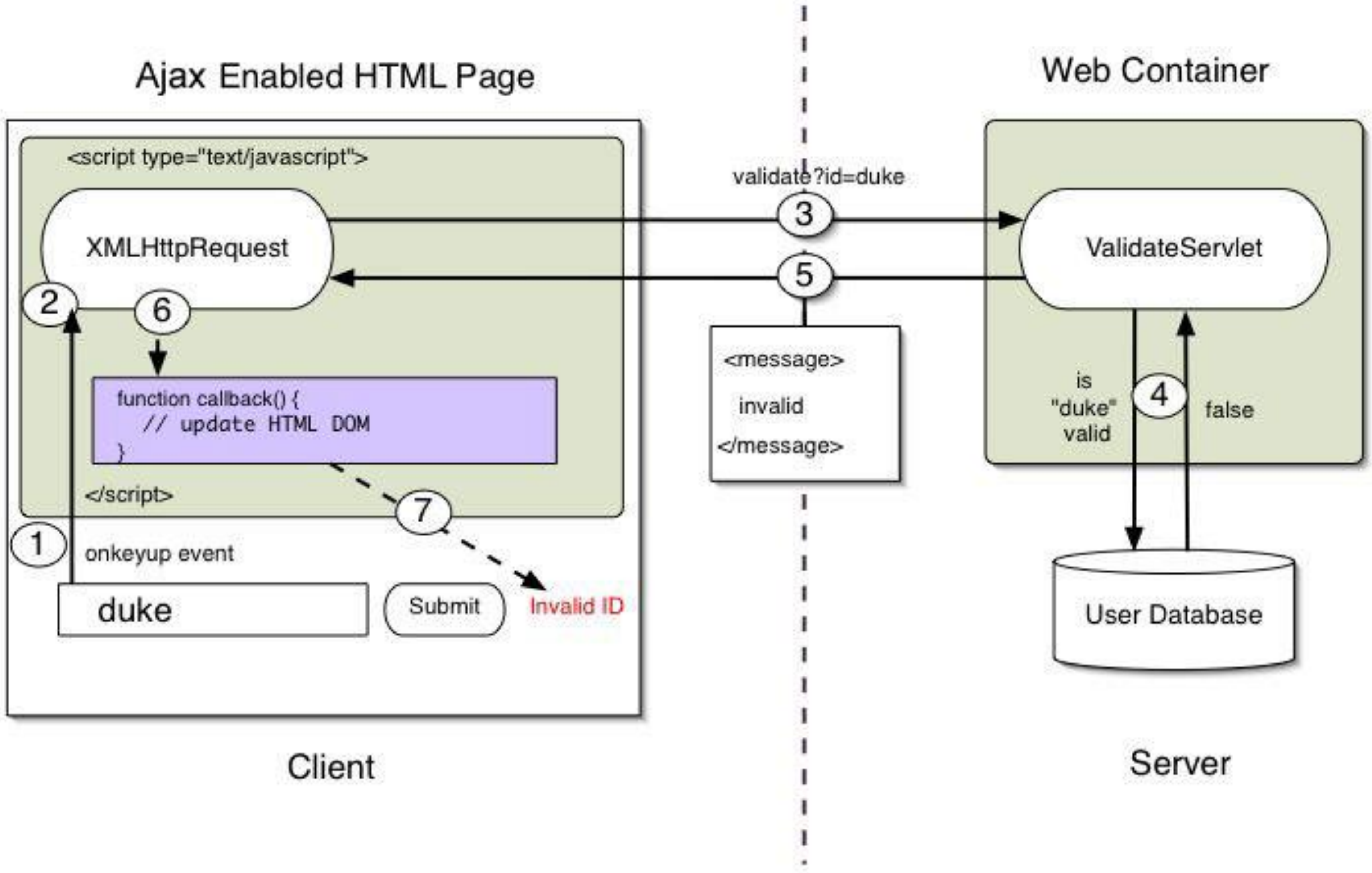
Some of its uses

- Real-time form validation
  - Checking serial numbers, zip codes, credit card validity
- Autocompletion
  - As the user types suggesting possible completions or useful completions
- Load on demand
  - Load portions of the page as they get downloaded (ex. Google maps)

More uses

- Rich user interface controls
  - Dynamically loaded trees, data tables, progress bars
- Refreshing data and server push
  - Live update of scores, stock ticker
- Page as an application
  - Create a single page as an application

AJAX flow


Ajax Enabled HTML Page
Web Container
<script type="text/javascript">
XMLHttpRequest
validate?id=duke
3
5
ValidateServlet
2
6
function callback() {
// update HTML DOM
}
<message>
invalid
</message>
is "duke" valid
4
false
</script>
7
1
onkeyup event
duke
Submit
Invalid ID
User Database
Client
Server

XMLHttpRequest

- The keystone of AJAX is the XMLHttpRequest object
- This object is used to establish connection to server
- Exchange data with the server
- Allows update of parts of page without reloading

Creating XMLHttpRequest

```
var xmlhttp;
if (window.XMLHttpRequest){
  // code for IE7+, Firefox, Chrome,
  Opera, Safari
  xmlhttp=new XMLHttpRequest();
}else{
  xmlhttp=new
  ActiveXObject("Microsoft.XMLHTTP");
}
```

Sending a request

- `xmlhttp.open("GET","ajax_info.txt",true);`
  `xmlhttp.send();`

Method	Description
open(*method*,*url*,*async*)	Specifies the type of request, the URL, and if the request should be handled asynchronously or not. *method*: the type of request: GET or POST *url*: the location of the file on the server *async*: true (asynchronous) or false (synchronous)
send(*string*)	Sends the request off to the server. *string*: Only used for POST requests

HTTP Get vs. Post

- For GET the URL is built by combining the parameters in the same way they appear for a form's GET

```
MySerlvet?loginId=chad&password=pass
```

`encodeURIComponent` function helps encode special characters "Chad & Patti" get encoded to "Chad+%26+Patti"

```
var url = "MyServlet?id=" + encodeURIComponent(idField.value);
xmlhttp.open("GET","demo_get2.asp?fname=Henry&lname=Ford",true);
xmlhttp.send();
```

Should not be used if cached page will not work or changing data on the server

POST

- Similar to GET, but must set request header and rather than send being empty, request parameters placed in send

```
xmlhttp.open("POST","ajax_test.asp",true);

xmlhttp.setRequestHeader("Content-type",
  "application/x-www-form-urlencoded");

xmlhttp.send("fname=Henry&lname=Ford");
```

Server response

Property	Description
xmlhttp.responseText	get the response data as a string
xmlhttp.responseXML	get the response data as XML data

- With response text, server can send back any text to be processed and placed on the form.

```
document.getElementById("myDiv").innerHTML=xmlhttp.responseText;
```

- With response xml, response is read using XML DOM model

onreadystatechange event

- When a request is sent to the server the xmlhttp receives events relating to the status of the response
- To use it you specify a **function** to assign to the event
- The xmlhttp will then call that function every time the ready state changes

Ready state

readyState	Holds the status of the XMLHttpRequest. Changes from 0 to 4: 0: request not initialized 1: server connection established 2: request received 3: processing request 4: request finished and response is ready
status	200: "OK" 404: Page not found

```
xmlhttp.onreadystatechange=function()
  {
    if (xmlhttp.readyState==4 &&
        xmlhttp.status==200){
      document.getElementById("myDiv").innerH
TML=
         xmlhttp.responseText;
    }
  }
```

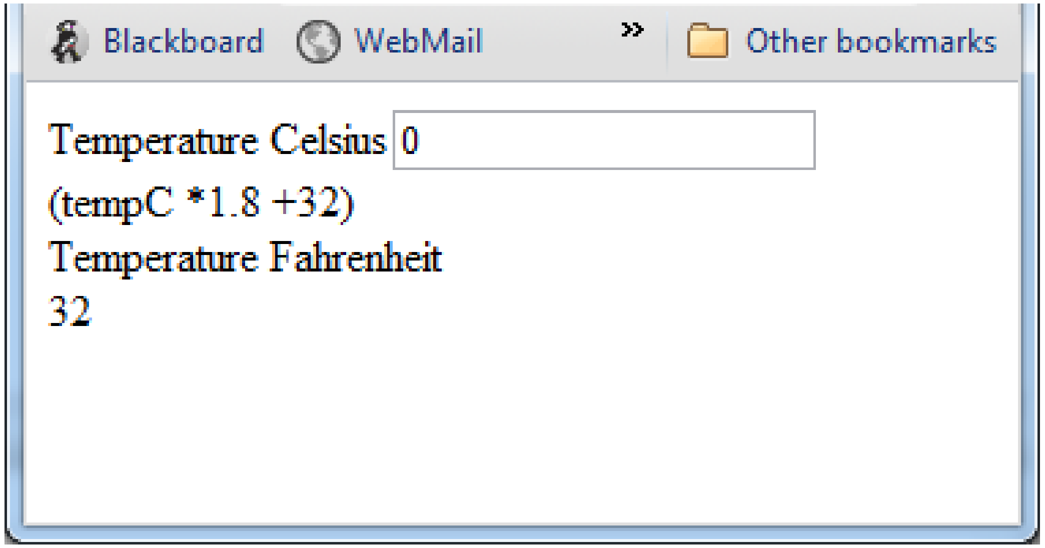
Acting on user entry

- Respond to any javascript event on the webpage and call a javascript function to retrieve content

```
<input type="text" id="userid" name="id" onkeyup="validate();"/>
```

- Walkthrough of AJAXTable
- Walkthrough AJAXAddition

AJAX
Temperature conversion

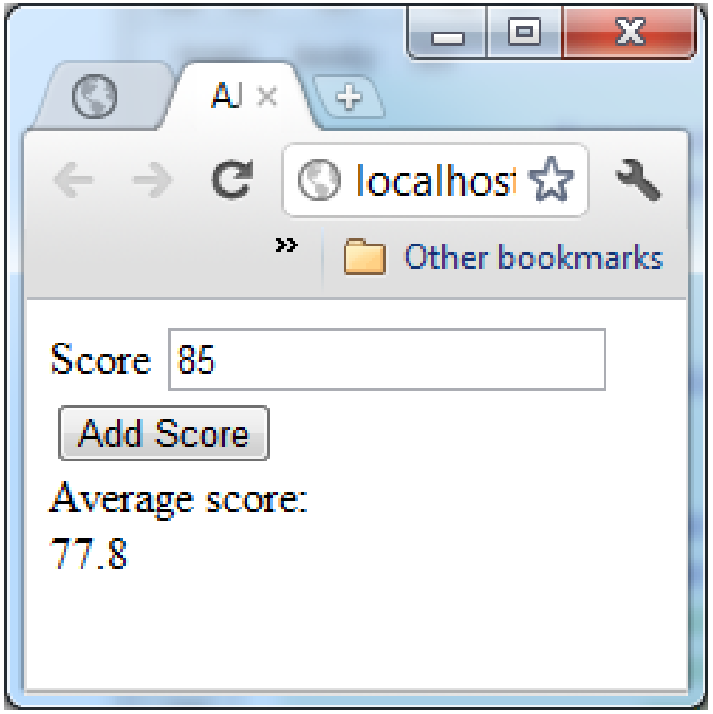


Temperature Celsius 0
(tempC *1.8 +32)
Temperature Fahrenheit
32

- You are given the shell of the page (AJAXTempConversion.html)
- Complete page
- Create Servlet

Review from last class

- Create a page/servlet to return the average of all scores added via AJAX (there is a shell of the HTML named AJAXAverage)
- Behavior is every time the user clicks Add Score the new average is output to the page



Score 85

Average score:
77.8

Topics for Lecture

- Structured content
  - Creating structured content
  - Processing XML

XML responses

- Often times a response from a server will be an xml document rather than plain text
- Allows structured content, allowing the page to determine what to show
- On server side this may be a static XML document, a document built from database results, or content built that is specific to that call
- Key aspect is the page decides format rather than the server
  - More flexible display

XML responses

Why XML response?

- Page decides format rather than the server
- In general it is better to have display logic on server, but there are exceptions
- <u>Very useful if building shared service</u>
    - Consider a feed of sports scores from the NFL to all news carriers
    - NFL builds single interface and XML response
    - News carriers specify how to handle the XML individually

XML document

```
<CATALOG>
    <CD>
        <TITLE>War</TITLE>
        <ARTIST>U2</ARTIST>
    </CD>
    <CD>
        <TITLE>Wasting Light<TITLE>
        <ARTIST>Foo Fighters<ARTIST>
    </CD>
</CATALOG>
```

Root

Child Elements

CD library service

- AJAXCDArtistsTitle.html
- AJAXCDSecondClient.html

XML document design

Guidelines:

- Cross between object model and data model
- If there is a 0-1/1-1 relationship the tag should be directly on the object (ex. TITLE)
- If there is a 0-many there should be a tag labeling the collection then repeated tag inside (ex. CATALOG/CD and TRACKS/TRACK)

```
<CATALOG>
    <CD>
        <TITLE>War</TITLE>
        <ARTIST>U2</ARTIST>
    </CD>
    <CD>
        <TITLE>Wasting Light</TITLE>
        <ARTIST>Foo Fighters</ARTIST>
        <TRACKS>
          <TRACK>Bridge Burning</TRACK>
          <TRACK>Rope</TRACK>
        </TRACKS>
    </CD>
</CATALOG>
```

Creating XML phone book service

- Design the XML structure to return the following
  - Multiple people
  - Each person has a name, city, state and zero to many work phone numbers AND zero to many personal phone numbers
- Create an example return document
- How might the XML structure change if each phone number had a description as well?

XML syntax

- With Document Object Model(DOM) you are able to specify retrieving nodes or tags within the XML
- The root of the document (the tag which hold everything in the XML) can be retrieved using

```
rootNode=xmlhttp.responseXML.documentElement
```

- With that element portions of the XML document can be retrieved by knowing what tags you are interested in

Accessing child elements

- Any child elements can be accessed by specifying the name of the tag you want to retrieve

```
cds = rootNode.getElementsByTagName("CD");
```

- Return will be an array of all elements matching that tag
- From that tag element you can access sub tag(s) elements in the same way

```
titles =
cds[0].getElementsByTagName(“TITLE”);
```

Accessing child elements cont.

- Be aware when you access the child elements with `getElementsByTagName` it will match all child tags with that name not just immediate child tags

```
<CDs>
  <CD>
    <ARTIST>Bob Dylan</ARTIST>
  </CD>
  <CD>
    <ARTIST>Nine Inch Nails</ARTIST>
  </CD>
</CDs>
```

Accessing Tag value

- Once you have the tag you want the value of (ex. TITLE) you access the node value of that tag

```
value = titles[0].firstChild.nodeValue
```

- By doing this structured data can be read from the response rather than just text

Printing CD Info

- Print the Artist and Title in a table

Printing Street and State

- Using the StudentInfo.xml output the street and state

Creating structured results

- Creating structured results is very useful when returning complex information often such as multiple results
- In creating the results you essentially use the servlet to build your xml file

- Ex. StudentGradesServlet, AJAXFindByNameServlet

XML phone book service revisited

- Using XML designed before…

- Write psuedo code for javascript side to output in this format with only the personal phone numbers printed:

  Bob – New Britain, CT
    - 860-867-5309
    - 888-345-1234

  Charles – Hamden, CT
    - 280-112-3581

Student grades

- **Step1:** Create web page that looks like the previous ones, where you click to get student grades
- Call the StudentGradesServlet to get the XML and display the student names and grades
- **Step2:** Update your page so that there is a text box where the user can type characters
- Modify the servlet so that the characters entered will filter so only names that begin with the characters entered will be returned
- **Step 3:** Make it so you can add new names and grades on your page