ITMD 465/565
Rich Internet Applications

Lecture 7

Fall 2019 – October 2, 2019

Tonight's Agenda

- Discuss midterm (end of next week on blackboard)
- AJAX

AJAX

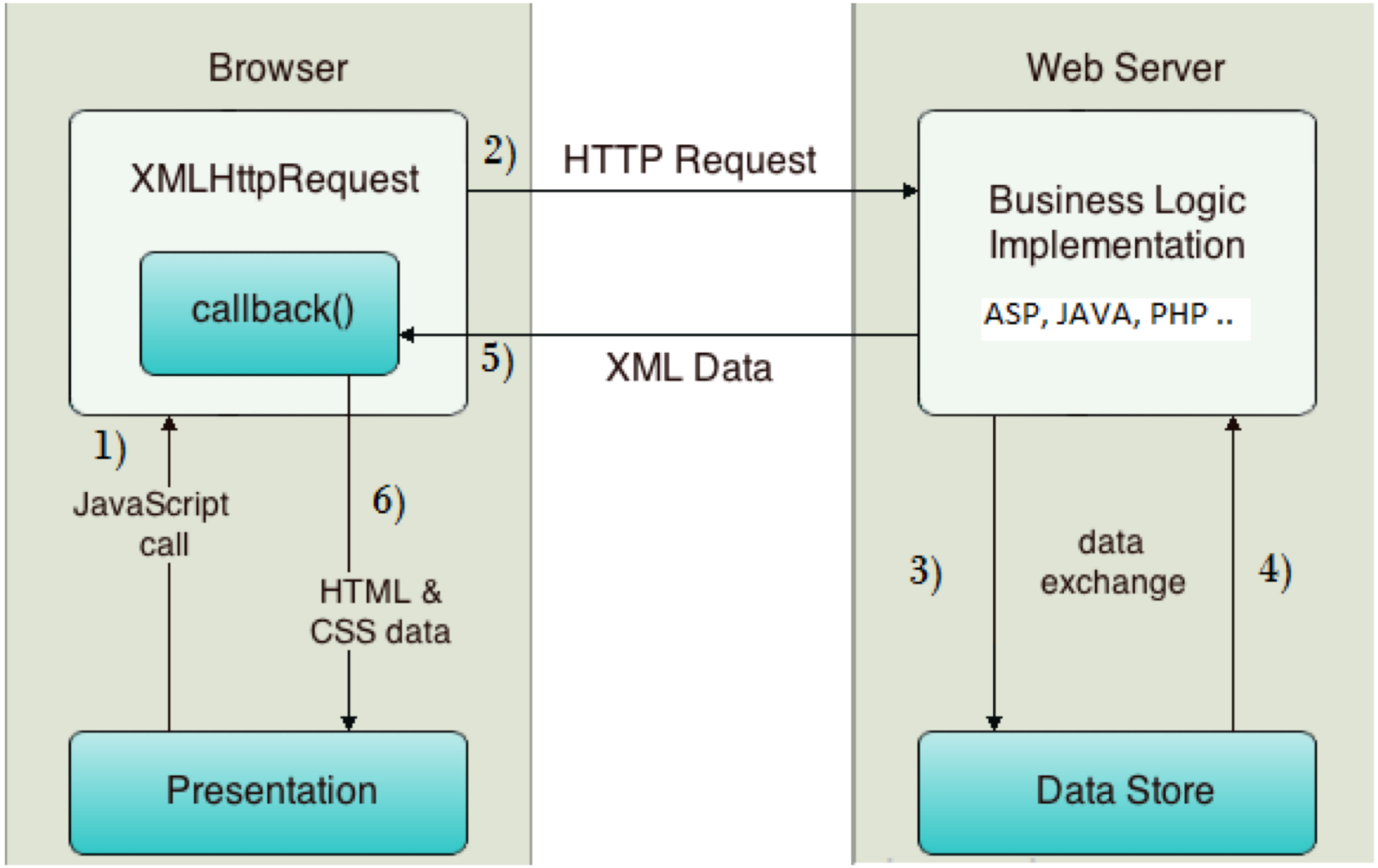
AJAX

- AJAX – **A**synchronous **J**avaScript **A**nd **X**ML
- AJAX is not a programming or markup language
- AJAX is a combination of other technologies:
  - Browser built-in object **XMLHttpRequest** to request data from a server
  - JavaScript and HTML DOM to modify the page once data is received
- Use of XML is not required even though it is in the name. XML, JSON, text/HTML are common
- Request is sent to server and response is received asynchronously and processed by JavaScript without a full page reload. This allows you to update parts of a web page without reloading the entire page.

AJAX Basic Steps

- Event occurs in page that triggers AJAX request (click or something)
- XMLHttpRequest object is created and configured
- The XMLHttpRequest object sends an asynchronous request to the server
- Server processes the request in back end code
- Server sends a response back to the XMLHttpRequest object including the results as response text. Check readyState and status to see if success.
- The XMLHttpRequest object uses the configured callback function to run and process the results if successful, or proper error response if not
- JavaScript in the callback function updates the HTML, DOM, and CSS of the page as needed

AJAX


Browser
Web Server
XMLHttpRequest
2)
HTTP Request
Business Logic Implementation
ASP, JAVA, PHP ..
callback()
5)
XML Data
1)
JavaScript call
6)
HTML & CSS data
3)
data exchange
4)
Presentation
Data Store

AJAX

- XMLHttpRequest (XHR)
  - http://en.wikipedia.org/wiki/XMLHttpRequest
  - https://developer.mozilla.org/en-US/docs/Web/API/XMLHttpRequest
  - https://developer.mozilla.org/en-US/docs/Web/API/XMLHttpRequest/Using_XMLHttpRequest
- All modern browsers and IE 7 + support the native XMLHttpRequest object.
- For IE < 7 support you would need to use the active x version. (not a problem anymore, if interested there are resources online that talk about this)

- Important Properties and Methods if xhr === XMLHttpRequest
  - xhr.onreadystatechange
  - xhr.readyState
  - xhr.status
  - xhr.responseText
  - xhr.open()
  - xhr.send();
- Ready State is an unsigned int
  - https://developer.mozilla.org/en-US/docs/Web/API/XMLHttpRequest/readyState
  - Can use constant as comparison instead of the number.
  - XMLHttpRequest.DONE same as 4

AJAX

- Same-Origin Policy
  - Script code must come from same server, domain, subdomain, protocol, port as the ajax call
  - http://en.wikipedia.org/wiki/Same_origin_policy
  - CORS will allow for cross-origin requests
  - https://enable-cors.org/
  - https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS
  - Basic CORS support can be achieved as long as the cross-origin resource providing the data sets the proper header
    - Access-Control-Allow-Origin: *

AJAX Basic Example

```
var myRequest = new XMLHttpRequest();

myRequest.onreadystatechange = function(){
  if (myRequest.readyState === 4) {
    if (myRequest.status === 200)
      var myArray = JSON.parse(myRequest.responseText);
      parseData(myArray);
    }
  }
};

myRequest.open('GET', 'http://libertyville.rice.iit.edu/scripts/data.php', true);

myRequest.send();

function parseData(arr) {
  console.log(arr);
}
```

AJAX Basic Example 2

```
var myRequest = new XMLHttpRequest();

myRequest.onreadystatechange = function(){
  if (myRequest.readyState === XMLHttpRequest.DONE) {
    if (myRequest.status === 200)
      var myArray = JSON.parse(myRequest.responseText);
      parseData(myArray);
    }
  }
};

myRequest.open('GET', 'http://libertyville.rice.iit.edu/scripts/data.php', true);

myRequest.send();

function parseData(arr) {
  console.log(arr);
}
```

AJAX Basic Example 3

```
var myRequest = new XMLHttpRequest();

myRequest.onreadystatechange = function(){
  if (myRequest.readyState === 4 && myRequest.status === 200) {
      var myArray = JSON.parse(myRequest.responseText);
      parseData(myArray);
  }
};

myRequest.open('GET', 'http://libertyville.rice.iit.edu/scripts/data.php', true);

myRequest.send();

function parseData(arr) {
  console.log(arr);
}
```

AJAX Events and Monitoring

- The XMLHttpRequest object allows us to listen for events that occur when the request is being processed, including progress, errors, and more.
- Must add before calling open()
- Doesn't work on file:// protocol
- Events:
  - progress
  - load
  - error
  - abort
  - Loadend
- An event object is the parameter of the event handler function
- See section on monitoring progress
  https://developer.mozilla.org/en-US/docs/Web/API/XMLHttpRequest/Using_XMLHttpRequest

AJAX Example using event for load

```
var myRequest = new XMLHttpRequest();

myRequest.addEventListener('load', parseData);

myRequest.open('GET', 'http://libertyville.rice.iit.edu/scripts/data.php', true);

myRequest.send();

function parseData(evt) {
  console.log(evt);
  var myArray = JSON.parse(evt.target.responseText);
  // code to process the array and modify the DOM
}
```

Fetch API

- Fetch API is a promise based api for doing AJAX requests.
- No support in IE but other modern browsers do support
- https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API
- https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API/Using_Fetch
- https://developers.google.com/web/updates/2015/03/introduction-to-fetch

```
fetch('http://libertyville.rice.iit.edu/scripts/data.php')
.then(function(response) {
        return response.json();
})
.then(function(myJson){
        console.log(JSON.stringify(myJson));
});
```

AJAX Libraries

- There is Ajax support built-in to most JavaScript libraries including jQuery
- jQuery ajax support is based on $.ajax(); function
- See jQuery docs for api reference
- http://api.jquery.com/
- Using libraries can simplify your use of Ajax
- New native fetch api is a more modern way to do ajax
- Popular AJAX library is Axios
  - Promise based HTTP client for the browser and node.js
  - https://www.npmjs.com/package/axios

Assignments

Reading/Assignments

- Lab will be posted soon.
- Continue Eloquent JavaScript, chapters 6 & 11
- Read through links in slides