Web and Mobile Application Development

Ajax

Material created by:
David Augenblick, Bill Mongan, Dan Ziegler, Samantha Bewley, and Matt Burlick

What is AJAX?

AJAX, standing for **A**synchronous **Ja**vaScript and **X**ML, is a technique used to dynamically load information from the model without having to refresh the page. This could be desired after…

- A user provides input and presses a button
- A page updates every so many minutes with the latest information
- A user sends something and expects a response back

Ajax Overview – why is it necessary?

- When an application is running, it sometimes needs to obtain and display information from an external source, such as:
  - An internet application
  - An external database
- If this activity were to take place while the application (web page or mobile app screen) was running, the application is likely to freeze (or at least not show the response).
- Various programming and scripting languages provide features to avoid this:
  - Java – threads
  - Android – `asynctask` class
  - JavaScript – Ajax

Ajax Overview – why is it necessary?

- Ajax provides an interface between the browser and server to coordinate this activity.
- An additional benefit of using Ajax is that it does not require the laborious task of reloading an entire web page when an update takes place.
- The flow of the Ajax process is illustrated on the next 2 slides.

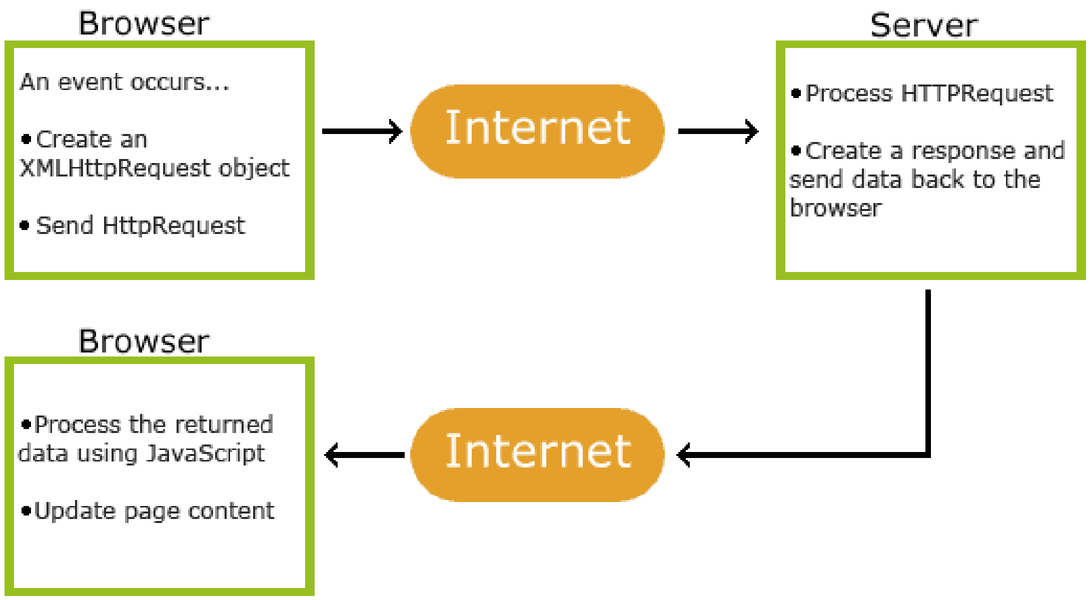
Understanding the Flow of Data

1. An event is defined as a click action or some other user interaction (or browser automated interaction) that sets off the AJAX request.

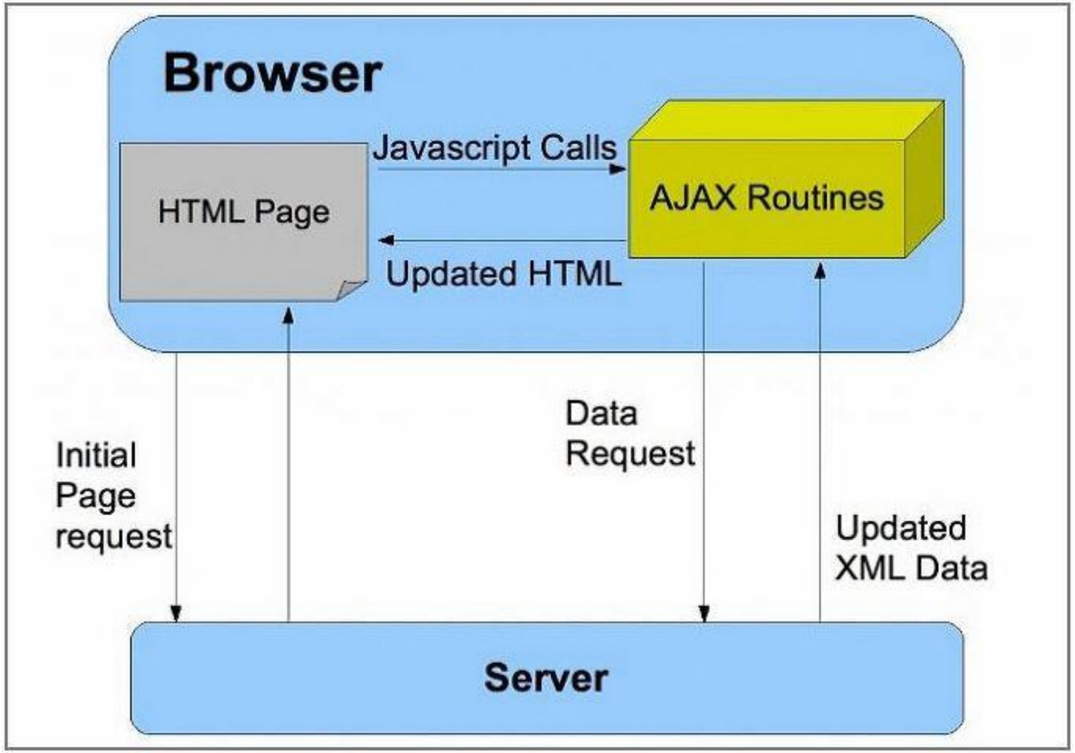
2. The request is created and is sent to the URL provided by the coder.

3. The script at the URL provided parses the request and takes appropriate action.

4. The data is returned back to the browser and the page is updated accordingly.


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

Understanding the Flow of Data



Ajax Overview – how to invoke Ajax

- There are 2 ways to incorporate Ajax into a JavaScript script:

1. Directly – using JavaScript's `XMLHTTPRequest` object
    - Limitations – Has difficulty processing cross-domain (external) requests.
2. Via jQuery's Ajax feature

- Examples for each method will be discussed in this slide set.

Ajax Overview – how to invoke Ajax

- First we'll use JavaScript’s `XMLHttpRequest` object to do AJAX.
- This JavaScript object:
  - Enables transfer of data to/from a web server using HTTP
  - Establishes a connection between the client and server for this purpose
- Many data formats supported, including JSON, HTML, and XML
- HOWEVER it is very strict about how/if you can make requests to servers outside of your own domain ☹
  - Fortunately jQuery combined with JSONP data type provides a work around for this ☺!
  - JSONP is a data type that is considered safe to send cross-domain.

Ajax Overview – how to invoke Ajax

- The `XMLHttpRequest` object comes with a variety of methods and properties
- The two most important methods are
  1. `open`(method, URL,async)
     - method ==>"GET", "POST", etc.
     - URL ==> source of the data to be transferred
     - Async ==> true/false. True if script processing continues after the "send" method has been called w/o waiting for the response
  2. `send()` - sends the request

Ajax Overview – how to invoke Ajax

- The XMLHttpRequest object also has several useful attributes associated with it:
    - `onreadystatechange` – We can bind a function to this that will be called whenever there is a state change
    - `readyState` – defines the current state of the `XMLHttpRequest` object
        - 0 ==> request not yet initialized
        - 1 ==> request has been set up
        - 2 ==> request has been sent
        - 3 ==> request in process
        - 4 ==> request has been competed
    - `status` – returns a status as a code
        - 200 = OK
        - 404 = Not Found
    - `responseText` – the response from the server (as a string)

Example

- Let's make an asynchronous request to the site

  http://jsonplaceholder.typicode.com/posts/1
- To do this we'll need to:
  - Create an `XMLHttpRequest` object.
  - Bind to its `onreadystatechange` attribute a function that will be called when the state of the object changes
- In particular if the object's `readyState` code is 4 or 200 (done) then populate a `div` with the content of the response (which is a JSON object)

Example

```
<script>
function request (){
     var xhttp = new XMLHttpRequest();
     xhttp.onreadystatechange = function()  {
          if(xhttp.readyState == 4 && xhttp.status == 200) {
               var json = $.parseJSON(xhttp.responseText);
               $("#current").html(json.title);
          } // end of "if" statement
     } // end of function().....
     var URL = "http://jsonplaceholder.typicode.com/posts/1";
     xhttp.open("GET",URL,true);
     xhttp.send();
}
$(document).ready(request);
</script>
```

Simplifying AJAX with jQuery

- jQuery’s implementation of AJAX tries to make things more organized and intuitive.
- Although it’s often more code
- **Nevertheless, for this course we’ll stick with using jQuery’s AJAX**

Ajax Via jQuery's Feature

```
function request (){
        var URL = " http://jsonplaceholder.typicode.com/posts/1”;
        $.ajax({
                type: "GET",
                url: URL,
                contentType:  "application/json; charset=utf-8",
                data: “{}”,
                dataType: "jsonp",
                success: function(msg) {
                        var json = msg;  //NOTE since we said we’re getting back jsonp, jQuery did the parsing for us!
                        $(“#current").html(json.title);
                },
                error: function (xhr, ajaxOptions, thrownError) {
                        $(“#current").html(“Error fetching “ + URL);
                }
        });
}
```

Ajax Via jQuery's Feature

Here's an explanation on some of the attributes that are part of the `$.ajax` function's parameter

- `$.ajax` – jQuery's ajax function
- `type: "GET"` - we want to perform a GET operation to bring back data from the server side
- `url`: address of the source of the server side data
- `data`: data to be sent to the server
- `contentType`: the format of the data we're sending to the server
- `dataType: "jsonp"` - (json with padding) – allows browser to interpret response as JSON from any source (not just local server). If type is json or jsonp, string response is automatically automatically parsed into json
- `msg` – the data return in the format specified by `dataType`

Summary

- AJAX is a very common way for you to be able to grab data from your model and put it into your view dynamically.
- It doesn’t take much code to set up, and is an extremely powerful tool to have in developing robust, clean, and high-functioning web applications.

References

http://www.w3Schools.com/ajax/ajax_examples.asp

http://api.jquery.com/jquery.ajax/