Web Application Security

Rajendra Kachhwaha
rajendra1983@gmail.com

September 23, 2015

Outline

Introduction to AJAX:

1. What is AJAX
2. Why & When use AJAX
3. What is an AJAX Web Application Model
4. Working of AJAX Web Application Model
5. AJAX Engine
6. XMLHttpRequest Object

What is AJAX

1. AJAX is neither a new programming language nor a new platform for developing websites. AJAX stands for Asynchronous JavaScript and XML, is a new approach, which display the refreshed contents on a Web page by using the Page Update approach rather than the Page Replacement approach.
2. AJAX uses client side scripts to make asynchronous call to a server & loads only those parts of a Web page that needs to be changed.
3. In AJAX, a call to server for page refreshment takes place in the background and the user does not even get a hint of when the request is sent to the server and when the new data is retrieved from the server.

Why & When use AJAX:

The most common benefits of AJAX are pretty easy to list:

1. No more full page refreshes.
2. Enables responsive web applications that feel like Windows applications.
3. Reduces the amount of data that must be exchanged between the server and client.

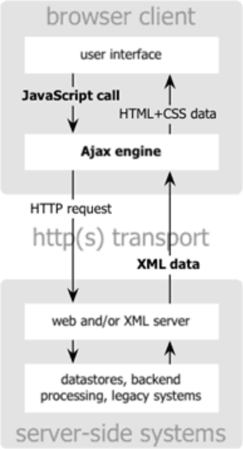
Common examples of these interactions are:

1. A form that validates values with some server process.
2. A drop down list that loads values in response to another elements action.
3. Voting or rating input elements & Multi-tab interfaces.
4. Any grid operations (such as sorting, selecting, editing, filtering, etc.)

What is an AJAX Web Application Model:

1. An AJAX application eliminates the start-stop-start-stop nature of interaction on the Web by introducing an intermediary-an AJAX engine-between the user and the server.
2. Instead of loading a webpage, at the start of the session, the browser loads an AJAX engine-written in JavaScript & usually tucked away in a hidden frame.
3. This engine is responsible for both rendering the interface the user sees & communicating with the server on the user's behalf.

What is an AJAX Web Application Model:

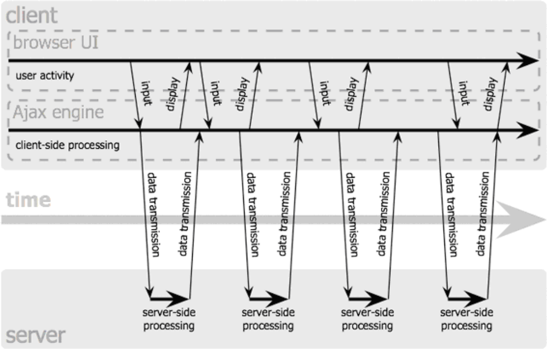


Working of AJAX Web Application Model

1. Every user action that normally would generate an HTTP request takes the form of a JavaScript call to the AJAX engine instead.
2. Any response to a user action that does not require a trip back to the server-such as simple data validation, editing data in memory, and even some navigation-the engine handles on its own.
3. If the engine needs something from the server in order to respond-if it is submitting data for processing, loading additional interface code, or retrieving new data-the engine makes those requests asynchronously, usually using XML, without stalling a user's interaction with the application

cont.

Working of AJAX Web Application Model

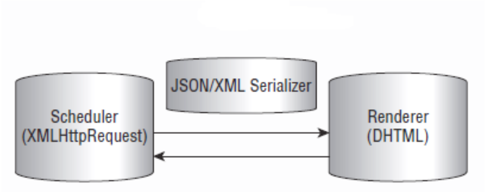


Basic technologies involved:

1. Standard based representation using Cascading Style Sheet (CSS) & Dynamic display and interaction using the Document Object Model (DOM).
2. Data interchange and manipulation using XML.
3. Asynchronous data retrieval using XMLHttpRequest object.
4. JavaScript binding everything together.

AJAX Engine:

For an AJAX to work, we need to make an AJAX Engine first. An AJAX engine takes complete control over the client-server communications and the rendering of the new information to ensure that these communications and renderings do not interrupt the user interactions. Following figure shows an AJAX engine.



AJAX Engine:

- Scheduler: The scheduler uses AJAX technologies such as XMLHttpRequest to send data to and receive data from the server in an asynchronous fashion.
- Renderer: The renderer component of the AJAX engine uses DHTML to dynamically update only those portions of the current page that need refreshing without re-rendering or re-loading the entire page.
- JSON/XML Serializer: The client and server exchange data in JSON or XML format. The JSON/XML serializer has two main responsibilities:1.Serialize the client data, which are JavaScript objects, into their JSON or XML representations before these objects are sent to the server. 2.Deserialize JavaScript objects from the JSON or XML data received from the server.

XMLHttpRequest Object:

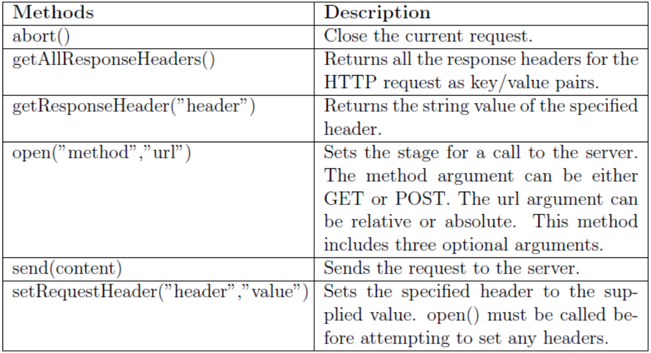
1. The XMLHttpRequest object is the heart of all asynchronous operations related to Ajax.
2. It is the object responsible for providing the asynchronous behavior through which Ajax-style applications can interact.
3. XMLHTTP is a protocol that is designed to package data as XML and send it via the network to a specific destination, or endpoint. This information is typically processed in some way, and the result is returned to the caller.

XMLHttpRequest Object:

```
var xmlHttp;
function createxmlHttpRequest()
{
if(window.ActiveXObject)
{
xmlHttp=new ActiveXObject(”Microsoft.XMLHTTP”);
}
else if(window.XMLHttpRequest)
{
xmlHttp=new XMLHttpRequest();
}
}
```

XMLHttpRequest Object Methods:

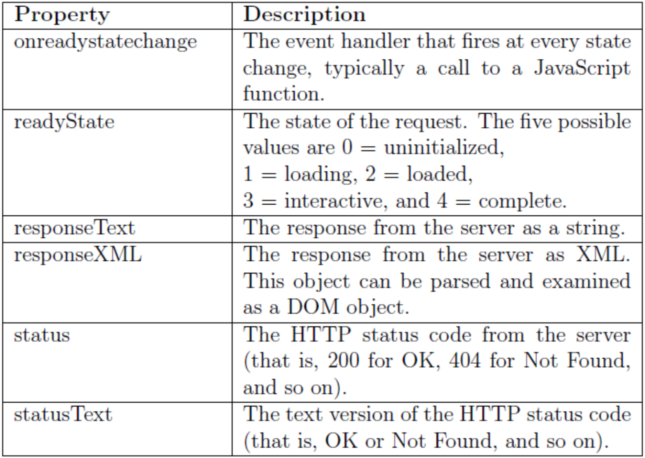
Following Table shows some typical methods of the XMLHttpRequest object.



Methods	Description
abort()	Close the current request.
getAllResponseHeaders()	Returns all the response headers for the HTTP request as key/value pairs.
getResponseHeader("header")	Returns the string value of the specified header.
open("method","url")	Sets the stage for a call to the server. The method argument can be either GET or POST. The url argument can be relative or absolute. This method includes three optional arguments.
send(content)	Sends the request to the server.
setRequestHeader("header","value")	Sets the specified header to the supplied value. open() must be called before attempting to set any headers.

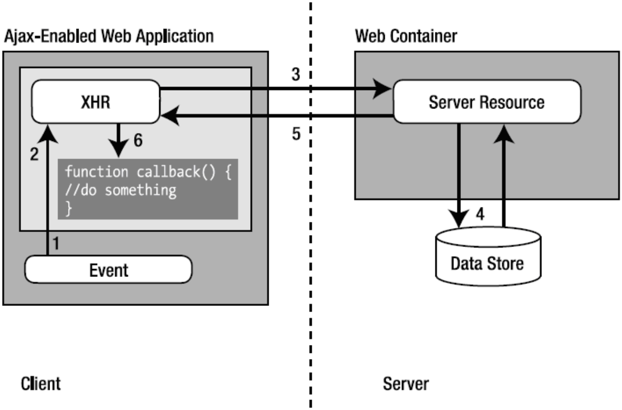
XMLHttpRequest Object Properties:

Following are the Properties of an XMLHttpRequest.



Property	Description
onreadystatechange	The event handler that fires at every state change, typically a call to a JavaScript function.
readyState	The state of the request. The five possible values are 0 = uninitialized, 1 = loading, 2 = loaded, 3 = interactive, and 4 = complete.
responseText	The response from the server as a string.
responseXML	The response from the server as XML. This object can be parsed and examined as a DOM object.
status	The HTTP status code from the server (that is, 200 for OK, 404 for Not Found, and so on).
statusText	The text version of the HTTP status code (that is, OK or Not Found, and so on).

An Example Interaction:



An Example Interaction:

```
var xmlHttp;
function validateEmail() {
var email = document.getElementById(”email”);
var url = ”validate?email=” + escape(email.value);
if (window.ActiveXObject) {
xmlHttp = new ActiveXObject(”Microsoft.XMLHTTP”);
}
else if (window.XMLHttpRequest) {
xmlHttp = new XMLHttpRequest();
}
xmlHttp.open(”GET”, url);
xmlHttp.onreadystatechange = handleStateChange;
xmlHttp.send(null);
}
```

An Example Interaction:

```
function handleStateChange()
{
if(xmlHttp.readyState == 4)
{
if(xmlHttp.status == 200)
{
- Process the server response here-
}
}
}
```