# Experiment No: 17

**Using AJAX Create visual search feature to search using name for practical number 15 which list name, mobile number and email id of matching users.**

**Date:**

**Competency and Practical Skills:**

**Relevant CO:** 5

**Objectives:**

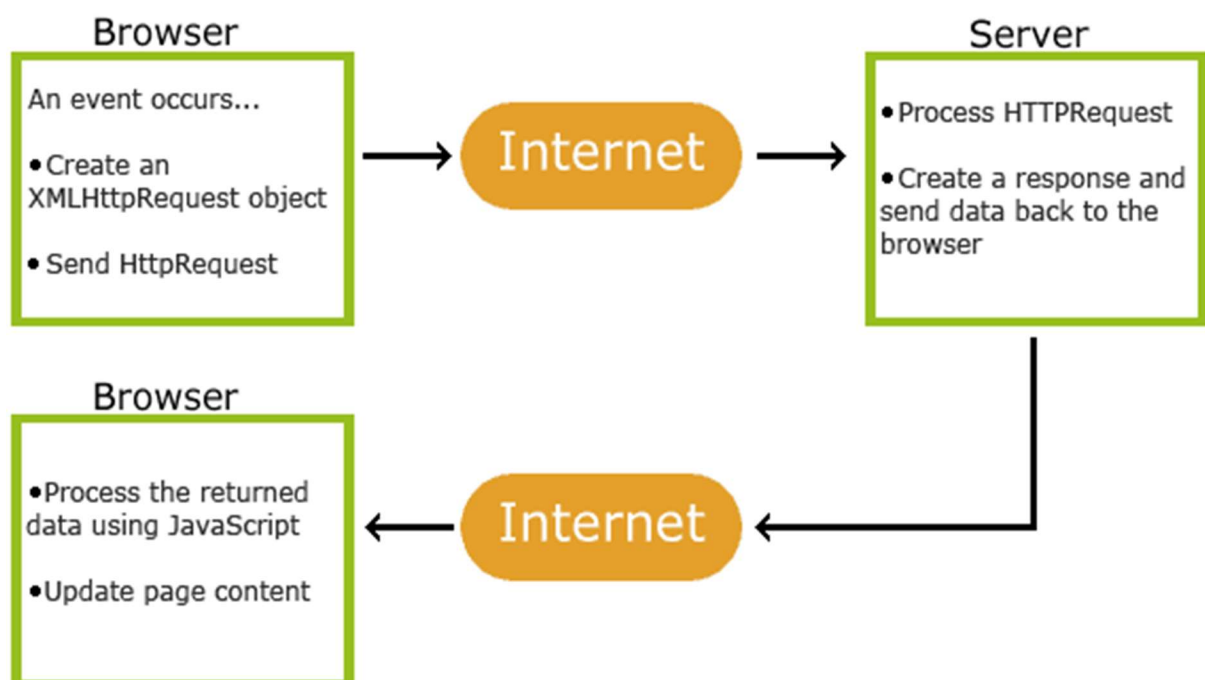1. To understand how Ajax works.

**Theory:**

What is AJAX?

AJAX = Asynchronous JavaScript and XML.

AJAX is not a programming language.

AJAX just uses a combination of:

    - A browser built-in XMLHttpRequest object (to request data from a web server)

    - JavaScript and HTML DOM (to display or use the data)

AJAX allows web pages to be updated asynchronously by exchanging data with a web server behind the scenes. This means that it is possible to update parts of a web page, without reloading the whole page.



Ref: https://www.w3schools.com/js/js_ajax_intro.asp

Steps:

1. An event occurs in a web page (the page is loaded; a button is clicked)
2. An XMLHttpRequest object is created by JavaScript
3. The XMLHttpRequest object sends a request to a web server
4. The server processes the request
5. The server sends a response back to the web page
6. The response is read by JavaScript
7. Proper action (like page update) is performed by JavaScript

The keystone of AJAX is the XMLHttpRequest object.

1. Create an XMLHttpRequest object

```
variable = new XMLHttpRequest();
```

2. Define a callback function

```
xhttp.onload = function() {
  // What to do when the response is ready
}
```

3. Open the XMLHttpRequest object

```
xhttp.open("GET", "ajax_info.txt");
```

4. Send a Request to a server

```
xhttp.send();
```

### XMLHttpRequest Object Methods

| Method | Description |
| --- | --- |
| new XMLHttpRequest() | Creates a new XMLHttpRequest object |
| abort() | Cancels the current request |
| getAllResponseHeaders() | Returns header information |
| getResponseHeader() | Returns specific header information |

| open(*method, url, async, user, psw*) | Specifies the request |
|---|---|
| | *method*: the request type GET or POST<br>*url*: the file location<br>*async*: true (asynchronous) or false (synchronous)<br>*user*: optional user name<br>*psw*: optional password |
| send() | Sends the request to the server<br>Used for GET requests |
| send(*string*) | Sends the request to the server.<br>Used for POST requests |
| setRequestHeader() | Adds a label/value pair to the header to be sent |

XMLHttpRequest Object Properties

| Property | Description |
|---|---|
| Onload | Defines a function to be called when the request is recieved (loaded) |
| onreadystatechange | Defines a function to be called when the readyState property changes |
| readyState | Holds the status of the XMLHttpRequest.<br>0: request not initialized<br>1: server connection established<br>2: request received<br>3: processing request<br>4: request finished and response is ready |
| responseText | Returns the response data as a string |

| responseXML | Returns the response data as XML data |
| --- | --- |
| Status | Returns the status-number of a request<br>200: "OK"<br>403: "Forbidden"<br>404: "Not Found"<br>For a complete list go to the Http Messages Reference |
| statusText | Returns the status-text (e.g. "OK" or "Not Found") |

**Call back function**

With the XMLHttpRequest object you can define a callback function to be executed when the request receives an answer. The function is defined in the onload property of the XMLHttpRequest object:

```
xhttp.onload = function() {
  document.getElementById("demo").innerHTML = this.responseText;
}
xhttp.open("GET", "ajax_info.txt");
xhttp.send();
```

**The onreadystatechange Property**

The readyState property holds the status of the XMLHttpRequest. The onreadystatechange property defines a callback function to be executed when the readyState changes. The status property and the statusText properties hold the status of the XMLHttpRequest object.

| readyState | Holds the status of the XMLHttpRequest.<br>0: request not initialized<br>1: server connection established<br>2: request received<br>3: processing request<br>4: request finished and response is ready |
| --- | --- |
| Status | 200: "OK"<br>403: "Forbidden"<br>404: "Page not found"<br>For a complete list go to the Http Messages Reference |

**Implementation:**

Using AJAX Create visual search feature to search using name for practical number 16 which list name, mobile number and email id of matching users.

**Output:**

**Conclusion:**

**Quiz:**

1. What is Ajax?
2. Explain XMLHttpRequest.

**Suggested Reference:**

- https://www.w3schools.com/xml/ajax_intro.asp

**References used by the students:**

**Rubric wise marks obtained:**

| Rubrics | 1 | 2 | 3 | Total |
|---------|---|---|---|-------|
| Marks   |   |   |   |       |