AJAX - Sending Request

AJAX applications use XMLHttpRequest objects to initiate or manage data requests sent to the web servers and handle or monitor the data sent by the web servers in a very effective manner. AJAX support the following types of requests –

- GET request
- POST request
- PUT request
- DELETE request

To create a connection and send a request to the web server XMLHttpRequest object provide the following two methods:

open() – It is used to create a connection between a web browser and the web server.

send() – It is used to send a request to a web server.

open() Method

The open() method is used to establish an asynchronous connection to the web server. Once the secure connection is established now you are ready to use various properties of XMLHttpRequest, or you can send requests, or handle the responses.

Syntax

```
open(method, url, async)
```

Where, the open() method takes three parameters –

- **method** – It represents the HTTP method that is used to establish a connection with the web server(Either GET or POST).
- **url** – It represents the file URL which will be opened on the web server. Or we can say server(file) location.
- **async** – For asynchronous connection set the value to true. Or for synchronous connection set the value to false. The default value of this parameter is true.

To use the open() method first we create an instance of the XMLHttpRequest object. Then we call the open() method to initialise the request with HTTP GET or POST method and the URL of the server.

The GET option is used to retrieve a moderate amount of information from the web server whereas the POST option is used to retrieve a larger amount of information. So both GET and POST options can configure the XMLHttpRequest object to work with the given file.

In the open() method, the filename or location or path of an AJAX application can be specified by either using an absolute path or a relative path. Where the absolute path is a path which specifies the exact location of the file, for example −

```
Myrequest.open("GET", "http://www.tutorialspoint.com/source.txt")
```

Here "source.txt" is the name of the file and "http://www.tutorialspoint.com" is the place where the source.txt file is stored.

The relative path is used to specify the location of a file according to the location on the web server in relation to the web application file, for example −

```
Myrequest.open("GET", "my file.txt")
```

Syntax

```
Myrequest.send()
```

send() Method

The send() method is used to send the request to the server. You can also pass an argument to the send() method.

Sending Request

To send a request to the server first we need to create an instance of XMLHttpRequest object then we create a callback function which will come into action after getting a response from the web server. Then we use the open() method to establish an asynchronous connection between the web browser and the web server then using send() function we send the request to the server.

Example

Here in the following code, we are fetching a specified record from the server. To fetch the data from the server we click on the "Click Here" button. So when we click on the "Click Here" button, the showDoc() function is called. Inside the displayDoc() function, first, an object of XMLHttpRequest is created. Then we create a call-back function to handle the server response. Then we call the open() method of the XHR object to initialise the request with HTTP GET method and the URL of the server that is "https://jsonplaceholder.typicode.com/todos/3" which fetches a single to-do list whose id = 3 from the JSONPlaceholder API. Then we call send() function to send the request.

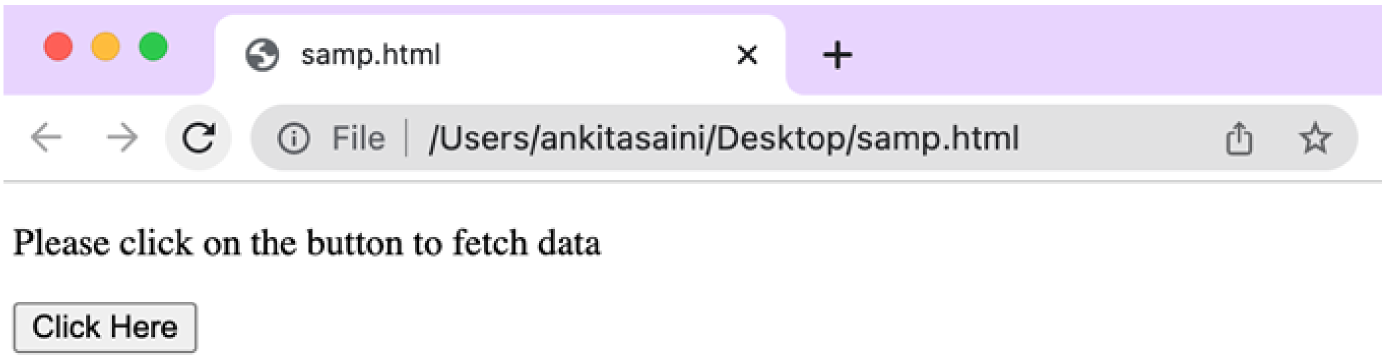
Open Compiler

```
<!DOCTYPE html>
<html>
<body>
<script>
   function ShowDoc() {
   // Creating XMLHttpRequest object
   var myhttp = new XMLHttpRequest();
   // Creating call back function
   myhttp.onreadystatechange = function() {
      if (this.readyState == 4 && this.status == 200) {
         document.getElementById("example").innerHTML = this.responseText;
      }
   };
  // Open the given file
  myhttp.open("GET", "https://jsonplaceholder.typicode.com/todos/3", true);
  // Sending the request to the server
  myhttp.send();
}
</script>

<div id="example">
  <p>Please click on the button to fetch data</p>
  <button type="button" onclick="ShowDoc()">Click Here</button>
</div>
</body>
</html>
```

Output

After clicking on "Click Here" button we will get the following record from the server.



Please click on the button to fetch data

So when the server responds to the request, the "onreadystatechange" property calls the callback function with the current state of the XMLHttpRequest object. If the "ready state" property is set to 4(that means the request is completed) and the "status" property is set to 200(that means the successful response), then the response data is extracted from the "responseText" property and display the HTML document with the help of "innerHTML" property of the example element.

Conclusion

So this is how we can send requests using XMLHttpRequest. Among all these requests GET and POST are the most commonly used request for fetching and sending data to/from the server. Now in the next article, we will see the type of request supported by AJAX.