Unit - 1

XML HttpRequest

```
+ Update a web page without reloading the page
+ Request data from a server - after the page has loaded
+ Receive data from a server  - after the page has loaded
+ Send data to a server - in the background

+ The responseText property returns the server response as a text string.
```

Window.history

```
+ The window.history object contains the browsers history.
```

Javascript Basics

```
Javascript Objects : A JavaScript object is a collection of named values

+ A primitive value is a value that has no properties or methods

+ JavaScript variables can contain single values : var person = "John Doe";   //
All strings are objects

+ The values are written as name : value pairs (name and value separated by a
colon). :
        var person = {firstName:"John", lastName:"Doe", age:50, eyeColor:"blue"};

+ Methods are actions that can be performed on objects

+ Objects are mutable: They are addressed by reference, not by value.
        var person = {firstName:"John", lastName:"Doe", age:50, eyeColor:"blue"}
        var x = person;
        x.age = 10;           // This will change both x.age and person.age

+ You can add new properties to an existing object by simply giving it a value.

+ Delete Keyword
        * The delete keyword deletes a property from an object:
        var person = {firstName:"John", lastName:"Doe", age:50, eyeColor:"blue"};
        delete person.age;   // or delete person["age"];

        * The delete keyword deletes both the value of the property and the
property itself.
```

```
        * After deletion, the property cannot be used before it is added back
again.

        * The delete operator is designed to be used on object properties. It has
no effect on variables or functions.

        * The delete operator should not be used on predefined JavaScript object
properties. It can crash your application.

        * The delete keyword does not delete inherited properties, but if you
delete a prototype property, it will affect all objects inherited from the
prototype.

+ The call() and apply() methods are predefined JavaScript methods.
        var person1 = {
                fullName: function() {
        return this.firstName + " " + this.lastName;
                }
        }
        var person2 = {
          firstName:"John",
          lastName: "Doe",
        }
        person1.fullName.call(person2);  // Will return "John Doe"
        whenever you want to invoke an other objects methods on an abject you
invoke call() or apply()
```

AJAX = Asynchronous JavaScript And XML.

```
- A browser built-in XMLHttpRequest object (to request data from a web server)
- JavaScript and HTML DOM (to display or use the data)

- XML Http Request
        + Holds the status of the XMLHttpRequest.
                - 0: request not initialized
                - 1: server connection established
                - 2: request received
                - 3: processing request
                - 4: request finished and response is ready

- Using A Callback function
        - A callback function is a function passed as a parameter to another
function.

        - loadDoc("url-1", myFunction1);
          loadDoc("url-2", myFunction2);
- Need for Asynchronous Requests -:
        + By sending asynchronously, the JavaScript does not have to wait for the
```

```
server response, but can instead:
        + execute other scripts while waiting for server response
        + deal with the response after the response is ready
- Synchronous XMLHttpRequest (async = false) is not recommended because the
JavaScript will stop executing until the server response is ready. If the server
is busy or slow, the application will hang or stop.

- AJAX Mechanics
        + Hidden Frames
        + XMLHttpRequest
        + Images
        + Javascript
        + Stylesheet

- Hidden IFrames
        – An iframe, reserved for the server data, is hidden initially. The main
window which houses the iframe, is visible.
        - On an event in the main window, the ‘src’ of the hidden iframe is
changed to point to a server resource.
        - The server returns data to the hidden iframe. The main window then makes
the frame visible (if need be) or the hidden frame updates the main window with
the data it received
        - Can Store history
        - Domain restrictions is a disadvantage.
```

Unit - 2

Predictive Fetch Pattern

- The Predictive Fetch pattern is a relatively simple idea that can be somewhat difficult to implement: the Ajax application guesses what the user is going to donext and retrieves the appropriate data

Difference between JSON and XML

```
1. JSON is less verbose. Saves bandwidth compared to XML.
2. On the web, JSON is easily parsed compared to XML (which needs a separate
parser)
3. XML is rendered very cleanly by most browsers (highly readable and organized).
4. XML can be automatically converted into html without writing a single line of
JS.
(using XSL Stylesheets). JSON may need extensive js code to do the same.

5. One of the most significant advantages that XML has over JSON is its ability to
communicate mixed content, i.e. strings that contain structured markup.

6. In XML we can (and need) to create Namespaces explicitly. In JSON an object
is a namespace by itself.
```

Submission Throttling

```
JSON.parse()
+ A common use of JSON is to exchange data to/from a web server.
+ When receiving data from a web server, the data is always a string.
+ Parse the data with JSON.parse(), and the data becomes a JavaScript object.

While Performing Submission throttling it is mandatory that we use local storage
to get the cached results.
We pass the php object url as parameter to the get item function.
this.sendTerm=function(){
                        object.url="suggest.php?term="+object.search.value;
                        //check if the search term is available in cache
                        // i.e. Local Storage
                        if(localStorage.getItem(object.url)){
                                var
cacheRes=JSON.parse(localStorage.getItem(object.url))
                                object.populateFood(cacheRes);
                                console.log(localStorage.getItem(object.url));
                                console.log("from browser cache")
                        }
                        else{
                        object.xhr.onreadystatechange=object.showResult;
                        console.log(object.search.value)
                        console.log(this)
                        object.xhr.open("GET",object.url,true);
                        object.xhr.send();
                        }
                }
Refer to suggest.html
The localstorage sends an array which is retreved through the php script.
This array is parsed inorder to convert it into a javascript array.
```

RSS

```
XML Parsing
this.processNews = function()
{
        if(xhr.readyState == 4 && xhr.status == 200)
        {
                root = xhr.responseXML.documentElement;
                item = root.getElementsByTagName("item")[0];
                //get title and link
                title = item.getElementsByTagName("title")[0];
                link = item.getElementsByTagName("link")[0];

                //Create an anchor
                anchor = document.createElement("a");
                anchor.innerHTML = title.firstChild.nodeValue;
                anchor.href = link.firstChild.nodeValue;

                obj.divinner.innerHTML = "";
                obj.divinner.appendChild(anchor);

        }
}
```