setTimeout() Method

- setTimeout() method calls a function or evaluates an expression after a specified number of milliseconds.

```
setTimeout(function(){ alert("Hello"); }, 3000);
```

setInterval() Method

- setInterval() method calls a function or evaluates an expression at specified intervals (in milliseconds).
- setInterval() method will continue calling the function until clearInterval() is called, or the window is closed.

```
setInterval(function(){ alert("Hello"); }, 3000);
```

Cookies

- Your server sends some data to the visitor's browser in the form of a cookie
- Cookies are plain text data record of 5 variable-length fields.
  - **Expires** – The date the cookie will expire. If this is blank, the cookie will expire when the visitor quits the browser.
  - **Domain** – The domain name of your site.
  - **Path** – The path to the directory or web page that sets the cookie. This may be blank,.
  - **Secure** – If this field contains the word "secure", then the cookie may only be retrieved with a secure server.
  - **Name = Value** – Cookies are set and retrieved in the form of key-value pairs.

- When exchanging data between a browser and a server, the data can only be text.
- JSON is text, and we can convert any JavaScript object into JSON, and send JSON to the server.
- We can also convert any JSON received from the server into JavaScript objects.
- This way we can work with the data as JavaScript objects, with no complicated parsing and translations.

*JSON

- The *Session Storage* is designed for scenarios where the user is carrying out a single transaction, but could be carrying out multiple transactions in different windows at the same time.

*Session Storage

- The *Local Storage* is designed for storage that spans multiple windows, and lasts beyond the current session. In particular, Web applications may wish to store megabytes of user data, such as entire user-authored documents or a user's mailbox, on the client side for performance reasons.
- Again, cookies do not handle this case well, because they are transmitted with every request.

*Local Storage

- The XMLHttpRequest object can be used to exchange 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.

*AJAX

XMLHttpRequest Object Properties

- **onreadystatechange** Defines a function to be called when the readyState property changes
- **readyState**
  - 0: request not initialized
  - 1: server connection established
  - 2: request received
  - 3: processing request
  - 4: request finished and response is ready
- **responseText** Returns the response data as a string
- **responseXML** Returns the response data as XML data
- **status**
  - 200: "OK"
  - 403: "Forbidden"
  - 404: "Not Found"
- **statusText** Returns the status-text (e.g. "OK" or "Not Found")