**Local & Session Storage**

**Web Storage:**

Before HTML5, application data had to be stored in cookies, included in every server request. Web storage is more secure, and large amounts of data can be stored locally, without affecting website performance.

Unlike cookies, the storage limit is far larger (at least 5MB) and information is never transferred to the server.

Web storage is per origin (per domain and protocol). All pages, from one origin, can store and access the same data.

The Web Storage API provides mechanisms by which browsers can store key/value pairs, in a much more intuitive fashion than using cookies.

The two mechanisms within Web Storage are as follows:

* sessionStorage maintains a separate storage area for each given origin that's available for the duration of the page session (as long as the browser is open, including page reloads and restores)
* localStorage does the same thing, but persists even when the browser is closed and reopened.

The Window.sessionStorage and Window.localStorage properties (to be more precise, in supporting browsers the Window object implements the WindowLocalStorage and WindowSessionStorageobjects

A different Storage object is used for the sessionStorageand localStorage for each origin — they function and are controlled separately.

To illustrate some typical web storage usage, we have created a simple example, imaginatively called Web Storage Demo (check in Projects folder). The landing page provides controls that can be used to customize the colour, font and decorative image. When you choose different options, the page is instantly updated; in addition your choices are stored in localStorage, so that when you leave the page then load it again later on your choices are remembered.

* Local storage — The local storage uses the localStorage object to store data for your entire website, permanently. That means the stored local data will be available on the next day, the next week, or the next year unless you remove it.
* Session storage — The session storage uses the sessionStorage object to store data on a temporary basis, for a single window (or tab). The data disappears when session ends i.e. when the user closes that window (or tab).

sessionStorage + localStorage

The session storage and local storage are accessed via these two globally available JavaScript objects:

* sessionStorage
* localStorage

The sessionStorage object and localStorage object are accessed in the same way. It is only the life span and visibility of the data stored that is different.

**Setting Properties**

You can set properties on the sessionStorage and localStorage object just like with a normal JavaScript object. Here is an example:

sessionStorage.myProperty = "Hello World";

localStorage.myProperty = "Hello World";

The first line of this code example sets the session storage property myProperty to the value Hello World. The second line sets the local storage property myProperty to the value Hello World.

If your property names contain characters which are not allowed in JavaScript variable names, you will need to use the square bracket access mode to set a property, like this:

sessionStorage["Invalid JS Property Name"] = "Hello World";

localStorage["Invalid JS Property Name"] = "Hello World";

Or,

you can use the setItem() function, like this:

sessionStorage.setItem("Invalid JS Property Name", "Hello World");

**Getting Properties**

You can get properties from the sessionStorage and localStorage objects like this:

var myProp = sessionStorage.myProperty;

var myProp = localStorage.myProperty;

If the property name is not a valid JavaScript variable name, you will need to use the square bracket access method, like this:

var myProp = sessionStorage["myProperty"];

var myProp = localStorage["myProperty"];

Or,

you can use the getItem() function, like this:

var myProp = sessionStorage.getItem("myProperty");

var myProp = localStorage.getItem("myProperty");

**Deleting Properties**

You delete a session or local storage property like this:

delete sessionStorage.myProperty;

delete localStorage.myProperty;

Or

you can use the removeItem() function, like this:

sessionStorage.removeItem ("myProperty");

**Clearing the Local Storage**

If you want to delete all properties stored in the sessionStorage or localStorage objects, you can use the clear() function. Here is a clear() function call example:

sessionStorage.clear();

localStorage.clear();

**Reading Number of Properties Stored**

You can read the number of properties stored in the sessionStorage or localStorage objects using the lengthproperty, like this:

var length = sessionStorage.length;

var length = localStorage.length;

**Iterating Keys in the Local Storage**

You can iterate the keys (property names) of the key - value pairs stored in the sessionStorage or localStorage, like this:

for(var i=0; i < sessionStorage.length; i++){

var propertyName = sessionStorage.key(i);

console.log( i + " : " + propertyName + " = " + sessionStorage.getItem(propertyName));

}

The sessionStorage.length property returns the number of properties stored in the sessionStorage object.

The function key() returns the property name (or key name) of the property with the index passed as parameter to the function.

Store data in localStorage

**Let's stores data using localStorage as shown in the following example:**

Example

<!DOCTYPE HTML>

<html>

<body>

<script>

// Checking browser support

if (typeof (Storage) !== "undefined") {

// Store

localStorage.setItem("website", "abcd.com");

}

else {

alert = "Sorry, Web Storage is not supporting in your browser.";

}

</script>

</body>

</html>

Let’s understand the example:

* First of all, we are checking that our browser will support Web Storage or not.
* setItem(key,value) is a method. When we pass a key name and value, it will add that key to the storage, or update that key's value if it already exists.
* Here we are storing "abcd.com" in localStorage. Where key is "website" and value is "abcd.com".

**We can store value this way also:**

**Example**

<!DOCTYPE HTML>

<html>

<body>

<script>

// Checking browser support

if (typeof (Storage) !== "undefined") {

// Store

localStorage.website = "abcd.com";

}

else {

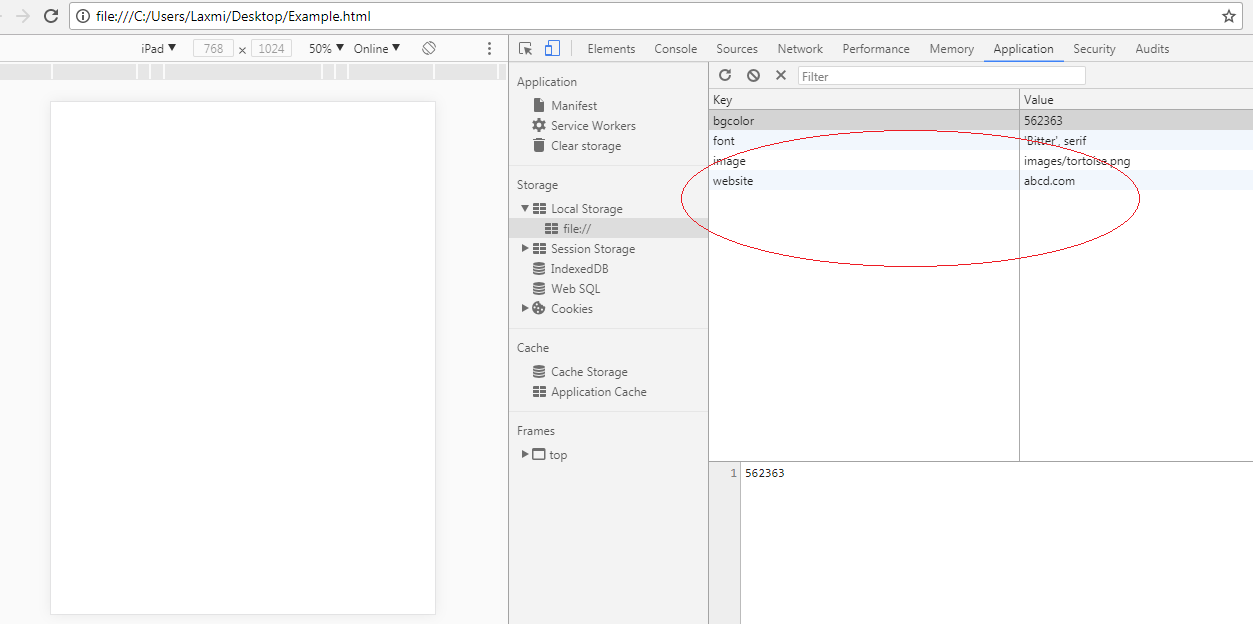
alert = "Sorry, Web Storage is not supporting in your browser.";

}

</script>

</body>

</html>

****

**Retrieve data from localStorage object**

Let's retrieve data as shown in the following example:

Example

<!DOCTYPE HTML>

<html>

<body>

<script>

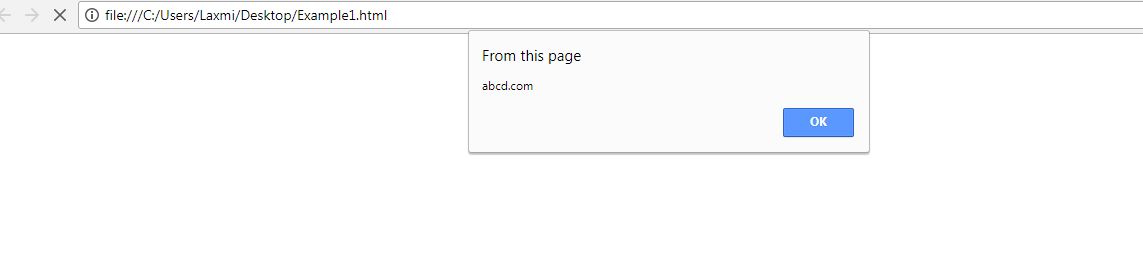
// Retrieving

alert(localStorage.website);

</script>

</body>

</html>



Delete data from localStorage object

* We can Delete data from localStorage object using removeItem(key) method. When we pass a key name, it will remove that key from the object.
* We can also use clear() method. It will remove all keys from the object.

Example

<!DOCTYPE HTML>

<html>

<body>

<script>

// It will remove "website" key from the object.

localStorage.removeItem(website);

// It will remove all keys from the object.

localStorage.clear();

</script>

</body>

</html>