158.258 Web Development/Application Design

HTML5 Local Storage

Computer Science & Information Technology, *College of Sciences, Massey University (AKLI, DISD & MTUI)*

Learning Objectives

At the end of this topic, you should be able to:

1. Explain how to work with local Web storage.
2. Explain the difference between *Local Storage* and *Session Storage.*
3. Choose the appropriate local Web storage for your Web application.
4. Store and retrieve data stored in the local Web storage.
5. Write scripts in JavaScript that store and retrieve data in local Web storage.

HTML5 Local Storage

Purpose: *To allow web applications to store data within the user's browser.*

Cookie	Web Storage	Local Storage
Version Supported	all	>= HTML5
Size	< 4K	> 5M
Data to the server?	Sometimes, yes	Never
Security	Lower	Higher

Local Storage and Session Storage

HTML5 **local storage** provides two objects for storing data on the client:

(1) Local storage (window.localStorage):

- Stores data with **no expiration date**.
- The data **will not be deleted** when the browser is closed — will be available the next day, week, or year.

(2) Session storage (window.sessionStorage):

- Stores data for one session.
- Data is lost when the browser tab is closed.
- The sessionStorage object is equal to the localStorage object, except that it stores the data for only one session.

Local Storage - Browser Support Test

Example: using setItem() and getItem()

```
    <script> // Check browser support
function storageTest() {
    r = document.getElementById("stResult")
    if (typeof(Storage) !== "undefined") {
        localStorage.setItem("Local storage", "YES");        // Store a value
        r.innerHTML = localStorage.getItem("Local_storage"); // Get it back
        r.style.color = 'green'
    } else {
        r.style.color = 'red'
        r.innerHTML = "It seems not!";
    }
}
</script>
<button onclick="storageTest()">Run localStorage test</button>
<p><b>Does your browser support local storage?</b></p>
<p id="stResult">Don't Know Yet</p>
```

Run localStorage test v1

Does your browser support local storage?

Don't Know Yet

http://www.w3schools.com/html/tryit.asp?filename=tryhtml5_webstorage_local

Local storage - Using array syntax

It's possible to use 'array-style' syntax with local storage

```
    <script> // Check browser support
function storageTest() {
    r = document.getElementById("stResult")
    if (typeof(Storage) !== "undefined") {
        localStorage["Local-storage"] = "YES";       // Store a value
        r.innerHTML = localStorage["Local-storage"]; // Get it back
        r.style.color = '#FF00FF'
    } else {
        r.innerHTML = "It seems not!";
    }
}
</script>
<button onclick="storageTest()">Run localStorage test</button>
<p><b>Does your browser support local storage?</b></p>
<p id="stResult">Don't Know Yet</p>
```

Run localStorage test v2

Does your browser support local storage?

Don't Know Yet

Local Storage - Using object property syntax

It's possible to use 'object-style' syntax (the dot notation) with local storage

```
    <script> // Check browser support
function storageTest() {
    r = document.getElementById("stResult")
    if (typeof(Storage) !== "undefined") {
        localStorage.Local storage = "YES";       // Store a value
        r.innerHTML = localStorage.Local_storage; // Get it back
        r.style.color = '#FF00FF'
    } else {
        r.innerHTML = "It seems not!";
    }
}
</script>
<button onclick="storageTest()">Run localStorage test</button>
<p><b>Does your browser support local storage?</b></p>
<p id="stResult">Don't Know Yet</p>
```

Run localStorage test v3

Does your browser support local storage?

Don't Know Yet

Local storage Syntax

Using Method syntax

- localStorage.setItem(key, value) // both as strings
- value = localStorage.getItem(key) // key is a string

Array Syntax

- localStorage['key'] = value
- value = localStorage['key']

Object property syntax

- localStorage.key = value // value is a string, KEY IS NOT
- value = localStorage.key

Using Local Storage - with numbers

Counts the total number of times a user has clicked a button. The count proceeds after the page is reopened.

Example: Click Counting

Click me!

Click the button to see the counter increase.

Close the browser tab (or window), and try again, and the counter will continue to count (is not reset).

---

http://www.w3schools.com/html/tryit.asp?filename=tryhtml5_webstorage_local_clickcount

Example of Session Storage

Counts the total number of times a user has clicked a button. The count restarts after the page is reopened.

Example: Click Counting

```
    <head>
    <script>
function clickCounter() {
    if(typeof(Storage) !== "undefined") {
        if (sessionStorage.clickcount) {
            sessionStorage.clickcount = Number(sessionStorage.clickcount)+1;
        } else {
            sessionStorage.clickcount = 1;
        }
        document.getElementById("result").innerHTML =
        "You have clicked the button " + sessionStorage.clickcount
        + " time(s) in this session.";
    } else {
        document.getElementById("result").innerHTML =
        "Sorry, your browser does not support web storage...";
    }
}
</script>
</head>
```

To play with HTML5, you need to have a good browser!

The following code works on Firefox version 48.0.2 and above, tested.

```
<html><body>
<div id = "result"></div>
<script>
    if (typeof(Storage) !== "undefined"){
        localStorage.setItem("lastname","smith");
        localStorage.setItem("firstname","tony");
        localStorage.setItem("age","30");

        var temp = "";
        for (var i = 0; i < localStorage.length; i++){
            var mykey = localStorage.key(i);
            var myvalue = localStorage.getItem(mykey);
            temp += mykey + ":" + myvalue + ";";
        }
        document.getElementById("result").innerHTML = temp;
}
</script></body></html>
```

Deleting values & Clearing Web Storage

To clean/remove all the values from local storage:

```
localStorage.clear();
```

To remove a specific item from the local storage:

```
localStorage.removeItem(key);
```

---

http://www.w3schools.com/html/tryit.asp?filename=tryhtml5_webstorage_session

Storing Structured Data (Arrays & Objects)

So far have stored only stored **strings** and **numbers**

Trying to save an array to localStorage

```
    <script>
function saveArray() {
    let x = [1, "222", 33.33]
    localStorage.arrayTest = x

    let y = localStorage.arrayTest
    alert("Value Retrieved was " + y + " Typeof = " + typeof(y))
}
</script>
<button onclick="saveArray()"> LocalStorage Array Test </button>
```

LocalStorage Array Test

Array assignment works but result is NOT an array, it's a string

Storing arbitrary data as strings

It's very useful to be able to save structured data (lists/objects) as strings

This would enable complex structured data to be

- saved to a file, and then reloaded
- the structure to be sent by email
- the structure to be send via a network connection

Python's 'Pickle' module does this

Javascript's JSON provides similar functionality

JSON - Javascript Object Notation

s = JSON.stringify(x)

provides a way to turn objects and arrays(**x**) into string **s**

x = JSON.parse(s)

reverses the process - recreate the original data structure

The thing being 'stringified' can be nested:

- it's OK to convert a list of objects which themselves contain lists …

IMPORTANT: JSON and methods

- JSON will save object properties (the data values)
- JSON.stringify(x) - does NOT SAVE THE METHODS of object x

Storing Structured Data (Arrays & Objects)

The methods **document.URL** and **document.title** are useful:

- document.URL : Returns the path to the current page (e.g. http://cnn.com)
- document.title : returns the *Title* of the current page

We can use these to create our own bookmark manager:

Remember this page Show Saved Pages Clear Bookmarked pages

Example Application of Local Storage: Scoping/personal namespaces

It's too easy to accidently create global variables

- this clutters the global namespace
- can result in accidentally using/setting names used by imported libraries

Your own namespace - a better alternative

Consider creating an object purely to store your own variables:

If you have three variables, *homepage*, *username* and *loggedIn* e.g.

```
      myVars = {homepage: null, loggedIn:false }
...
myVars.username = document.getElementById('userNameBox').value
```

This makes it easy to

- keep your variables separate from other modules' variables
- save your state to local storage
    - *for restoring at a later time*
    - *providing undo*

End of Topic Summary

Saving values locally

sessionStorage

can be used to store data until you close the session/tab

localStorage

can be used to store data **FOR THE CURRENT BROWSER** persistently.

Both store data as as **key/value** pairs where the value is a string.

Three syntax options for save/lookup

These all work:

- **.setItem(key, value)** / **.getItem(key)**
- array-like save/load with the key as a string: e.g. **localstorage['age']**
- method.property syntax: e.g. **localStorage.key**

Non-string values need to use JSON

- **JSON.stringify(item)** will convert *item* to a textual format
- **JSON.parse(s)** will convert *s* back into an object