Chapter 15

How to work with browser, objects, cookies, and web storage

Objectives

- How to script browser objects
- The Tutorial application
- How to use cookies
- The Task List application
- How to use web storage
- How to use Chrome to work with items in the browser

How to script browser objects

How to script browser objects

- We will learn two objects of browser are **location** and **history** object.
- ***Location object*** lets you work with the URL for a web page.
- ***History object*** lets you work with the pages that are stored in your browser’s history.

How to use the location object

- The example URL

http://www.murach.com:8181/javascript/location.html?first=G&last=Hopper#result

- Properties of the location object

Property	Description	Value in the URL above
`href`	The complete URL of the web page	`Complete URL`
`protocol`	The protocol portion of the URL	`http:`
`hostname`	The host name portion of the URL	`www.murach.com`
`port`	The port number of the web server	`8181`
`host`	The host name and port number	`www.murach.com:8181`
`path`	The path to the web page	`/javascript/location.html`
`search`	The query string from the URL	`?first=G&last=Hopper`
`hash`	The anchor name from the URL	`#result`

How to use the location object (cont.)

- Methods of the location object

Method	Description
`reload(force)`	Reloads the current web page.
`replace(url)`	Loads a new page in the browser and overwrites the current page in the history

- Examples
  - How to load a new web page

```
location.href = http://www.murach.com;
location = http://www.murach.com;
```

  - How to reload a web page

```
location.reload();
location.reload(true);
```

  - How to load a new page and overwrite the current history page

```
location.replace(“http://www.murach.com”);
```

How to use the history object

- The history object represents the user’s history list of viewed web page.
- One property of the history object

Property	Description
`lenght`	The number of URLs in the history object.

- Methods of the history object

Method	Description
`back()`	Goes back one step in the URL history.
`forward()`	Goes forward one step in the URL history.
`go(position)`	Goes forward or back specified number of steps in the URL history.
`go(substring)`	Goes to the most recent URL in the history that contains the substring.

How to use the history object (cont.)

- How to use the back() method

```
history.back();
```

- How to use the forward() method

```
history.forward();
```

- How to use the go() method
  - Go forward two URLs

```
history.go(2);
```

  - Go back three URLs

```
history.go(-3);
```

The Tutorial application
(Page 440-443)

How to use cookies

An introduction to cookies

- A ***cookie*** is a short text string that is stored by the browser as a name/value pair.
- Cookies let a web server or web page store information in a user’s browser.
- When you request a web page, the server can return a cookie as part of the HTTP response.
- A ***session cookie*** is deleted when the web browser is closed.
- A ***persistent cookie*** is saved by the web browser after the browser is closed.

An introduction to cookies (cont.)

- Attributes of a cookie

Attributes	Description
`max-age`	The lifetime of the cookie in seconds.
`path`	The path on the web server that can see the cookie.
`domain`	The domain name that can see the cookie.
`secure`	If present, the cookie must be encrypted when it transmitted, and it can only transmitted use security protocol.

- Cookie example

```
email=grace@gmail.com; path=/
username=ghopper; max-age=1814400; path=/
```

How to create cookies

- Two functions for working with cookies

Functions	Description
`encodeURIComponent(value)`	Encode values that contains semicolons, commas, or white space.
`decodeURIComponent(value)`	Decode values that have been encoded.

- How to create a session cookie

```
var cookie =“tasks” + “=”;
cookie += encodeURIComponent(“Feed dog\nWater plants”);
cookie += “; path=/”;
document.cookie = cookie;
```

How to create cookies (cont.)

- How to create a persistent cookie

```
var cookie =“tasks” + “=”;
cookie += encodeURIComponent(“Feed dog\nWater plants”);
cookie += “; max-age=” + 21*24*60*60;
cookie += “; path=/”;
document.cookie = cookie;
```

- How to add multiple cookies to the document.cookie object

```
document.cookie = “email=john@doe.com; path=/”;
document.cookie = “username=ghopper; max-age= 1814400
path=/”;
document.cookie = “email=grace@gmail.com path=/”;
```

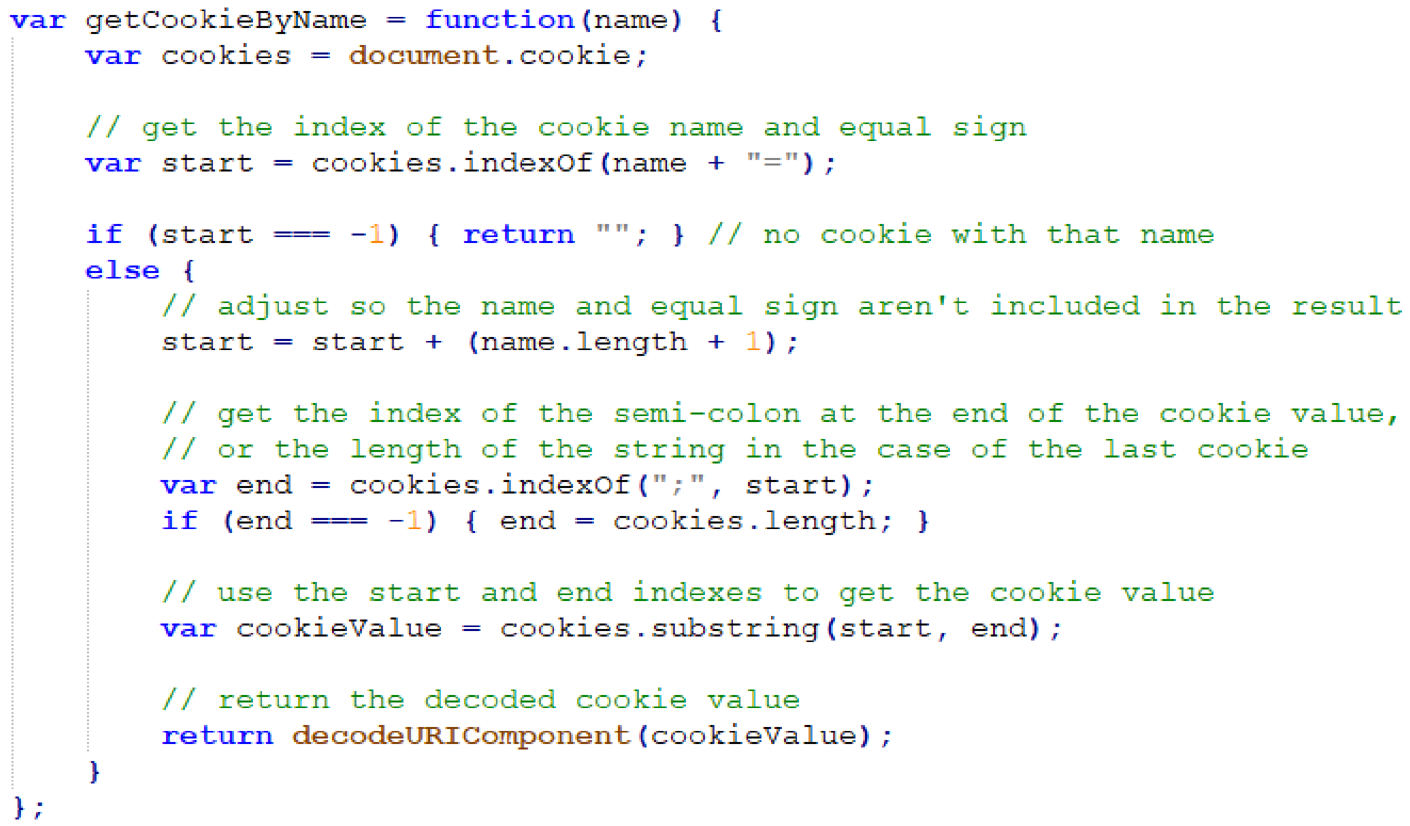
How to read cookies

- The cookies are stored in document.cookie object.
- Assume there are three cookies in the document.cookie object

```
username=ghopper; status=active;
tasks=Water%20plants;
```

How to read cookies(cont.)

- A getCookieByName() function that gets a cookie by name



```
var getCookieByName = function(name) {
    var cookies = document.cookie;

    // get the index of the cookie name and equal sign
    var start = cookies.indexOf(name + "=");

    if (start === -1) { return ""; } // no cookie with that name
    else {
        // adjust so the name and equal sign aren't included in the result
        start = start + (name.length + 1);

        // get the index of the semi-colon at the end of the cookie value,
        // or the length of the string in the case of the last cookie
        var end = cookies.indexOf(";", start);
        if (end === -1) { end = cookies.length; }

        // use the start and end indexes to get the cookie value
        var cookieValue = cookies.substring(start, end);

        // return the decoded cookie value
        return decodeURIComponent(cookieValue);
    }
};
```

How to use getCookieByName() function

```
var tasks= getCookieByName(“tasks”);
```

How to delete cookies

- To delete a cookie, you set its max-age attribute to 0.
- The cookie to delete

```
task=Feed dod; max-age=1814400; path=/
```

- How to delete a cookie

```
var cookie =“tasks=”;
cookie +=“; max-age=” + 0;
cookie +=“; path=/”;
document.cookie= cookie;
```

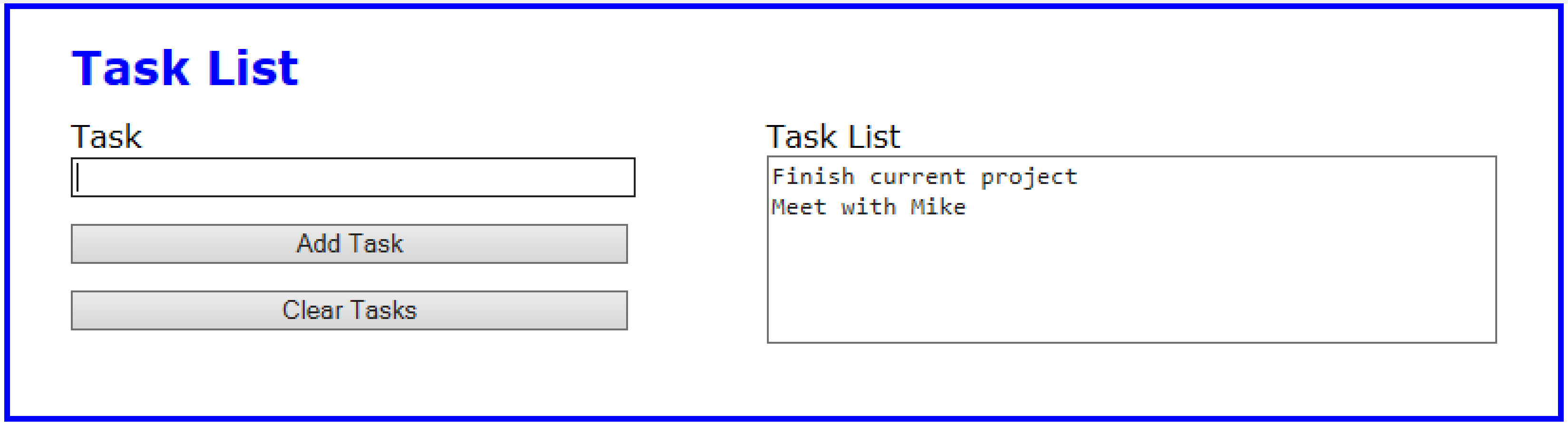
- A **deleteCookie()** function that delete a cookie

```
var deleteCookie =function(name){
   document.cookie = name + “=‘’;max-age=0; path=/”;
};
deleteCookie(“task”);
```

The Task List application

The Task Application

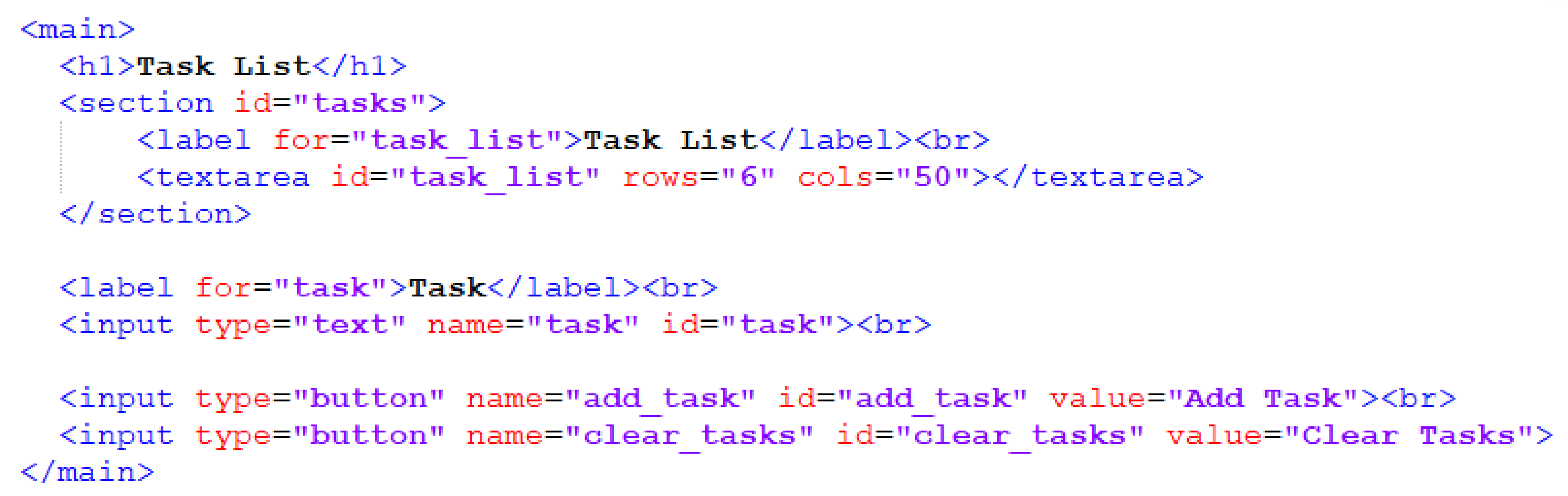
- The user interface



```
Finish current project
Meet with Mike
```

The Task Application

- The HTML code



```html
<main>
  <h1>Task List</h1>
  <section id="tasks">
      <label for="task_list">Task List</label><br>
      <textarea id="task_list" rows="6" cols="50"></textarea>
  </section>

  <label for="task">Task</label><br>
  <input type="text" name="task" id="task"><br>

  <input type="button" name="add_task" id="add_task" value="Add Task"><br>
  <input type="button" name="clear_tasks" id="clear_tasks" value="Clear Tasks">
</main>
```

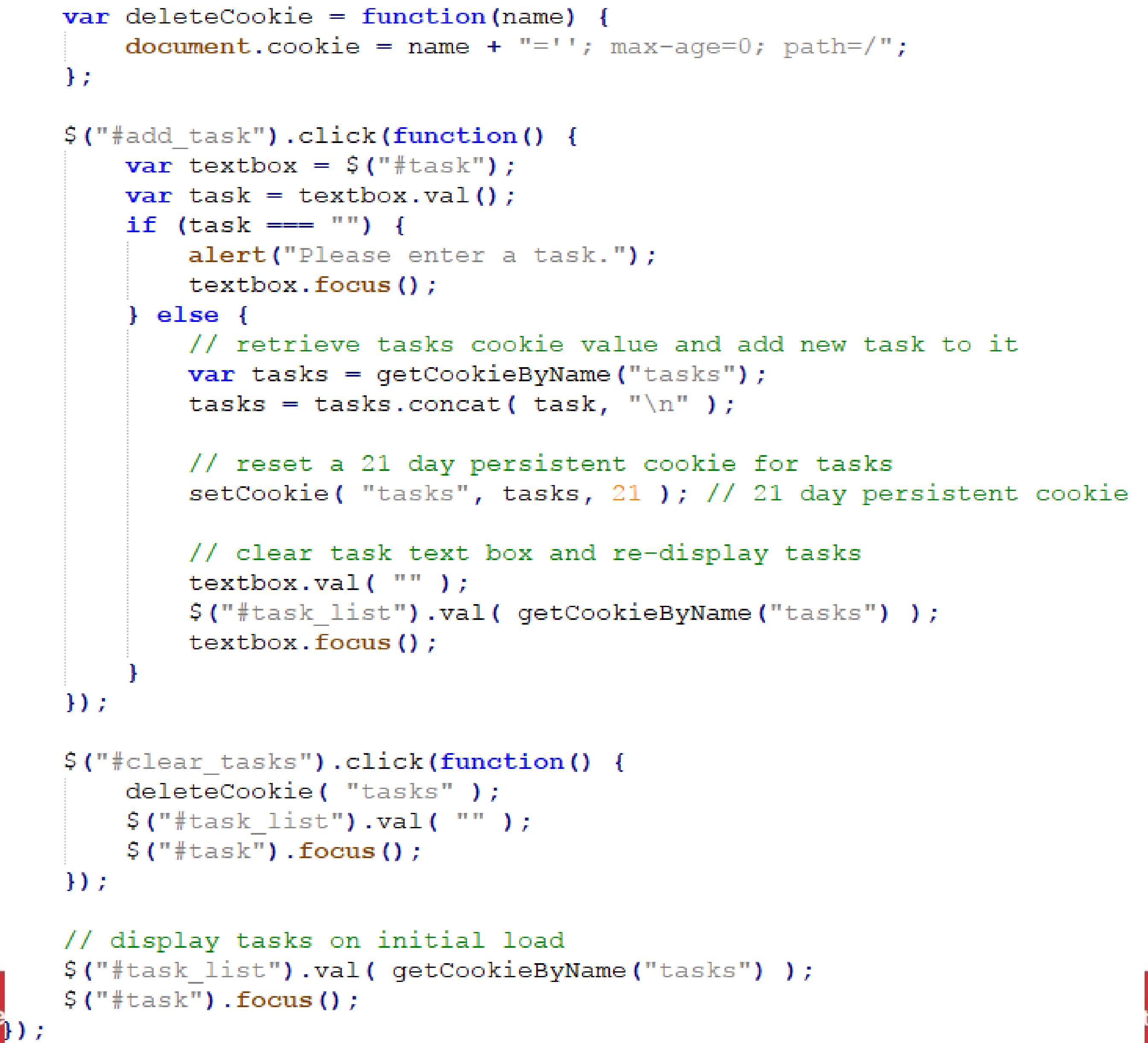
The Task Application

- The JavaScript code

```
$(document).ready(function(){
    var setCookie = function(name, value, days) {
        // concatenate cookie name and encoded value
        var cookie = name + "=" + encodeURIComponent(value);

        // if there's a value for days, add max-age to cookie
        if (days !== undefined) {
            cookie += "; max-age=" + days * 24 * 60 * 60;
        }
        // add path to cookie and then set
        cookie += "; path=/";
        document.cookie = cookie;
    };

    var getCookieByName = function(name) {
        var cookies = document.cookie;

        // get the index of the cookie name and equal sign
        var start = cookies.indexOf(name + "=");

        if (start === -1) { return ""; } // no cookie with that name
        else {
            // adjust so the name and equal sign aren't included in the result
            start = start + (name.length + 1);

            // get the index of the semi-colon at the end of the cookie value,
            // or the length of the string in the case of the last cookie
            var end = cookies.indexOf(";", start);
            if (end === -1) { end = cookies.length; }

            // use the start and end indexes to get the cookie value
            var cookieValue = cookies.substring(start, end);

            // return the decoded cookie value
            return decodeURIComponent(cookieValue);
        }
    };
```

The Task Application

- The JavaScript code



```
var deleteCookie = function(name) {
    document.cookie = name + "=''; max-age=0; path=/";
};

$("#add_task").click(function() {
    var textbox = $("#task");
    var task = textbox.val();
    if (task === "") {
        alert("Please enter a task.");
        textbox.focus();
    } else {
        // retrieve tasks cookie value and add new task to it
        var tasks = getCookieByName("tasks");
        tasks = tasks.concat( task, "\n" );

        // reset a 21 day persistent cookie for tasks
        setCookie( "tasks", tasks, 21 ); // 21 day persistent cookie

        // clear task text box and re-display tasks
        textbox.val( "" );
        $("#task_list").val( getCookieByName("tasks") );
        textbox.focus();
    }
});

$("#clear_tasks").click(function() {
    deleteCookie( "tasks" );
    $("#task_list").val( "" );
    $("#task").focus();
});

// display tasks on initial load
$("#task_list").val( getCookieByName("tasks") );
$("#task").focus();
});
```

How to use web storage

Web Storage introduction

- **Web storage** lets the web page use JavaScript to store data in key/value pair like cookie, but it is over limit of cookie.
- **Web storage** is currently supported by every modern browser.
- There are **local** and **session** storage. Local storage can store persist, session storage will be remove when session end.

How to use local and session storage

- The syntax for working with local or session storage

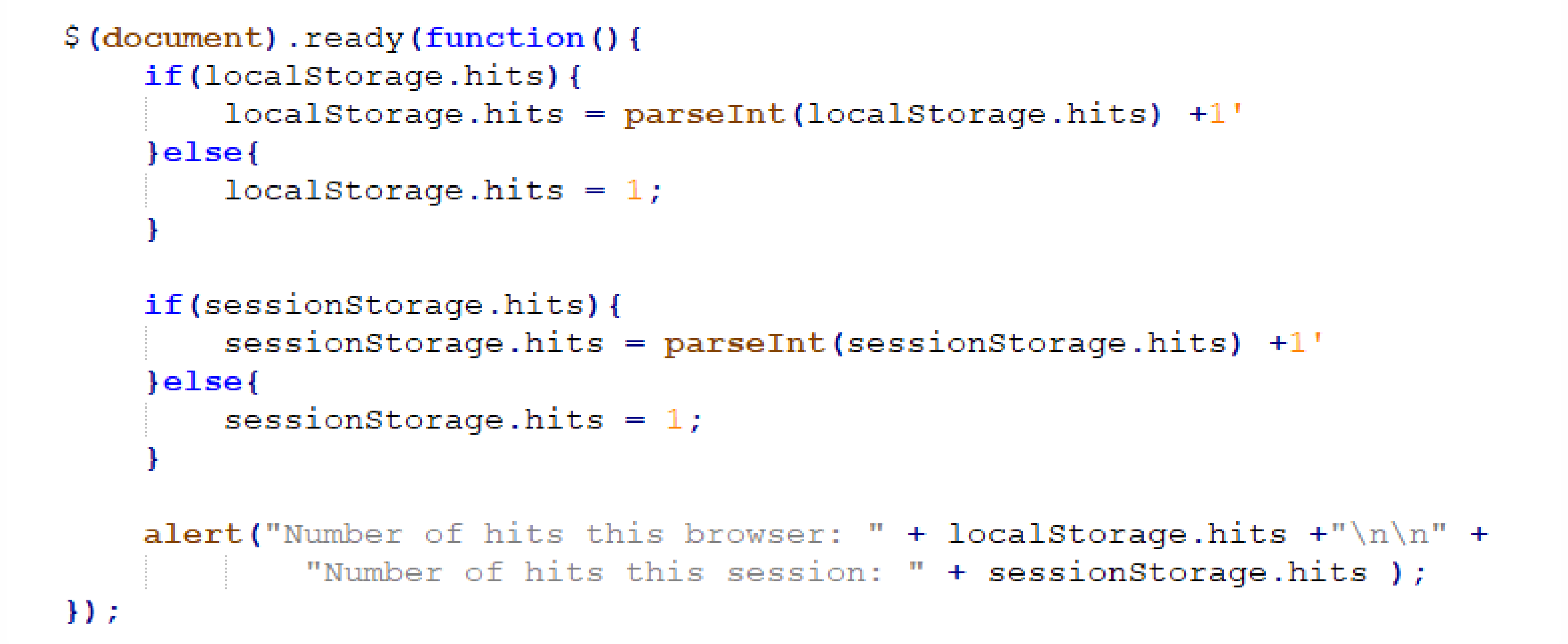
```
localStorage.setItem(“itemname”,”value”)
localStorage.getItem(“itemname”)
localStorage.removeItem(“itemname”)
localStorage.clear()

sessionStorage.setItem(“itemname”,”value”)
sessionStorage.getItem(“itemname”)
sessionStorage.removeItem(“itemname”)
sessionStorage.clear()
```

- The shortcut syntax for getting and saving an item

```
localStorage.itemname
sessionStorage.itemname
```

How to use local and session storage (cont.)

- JavaScript that uses local and session storage for hit counters



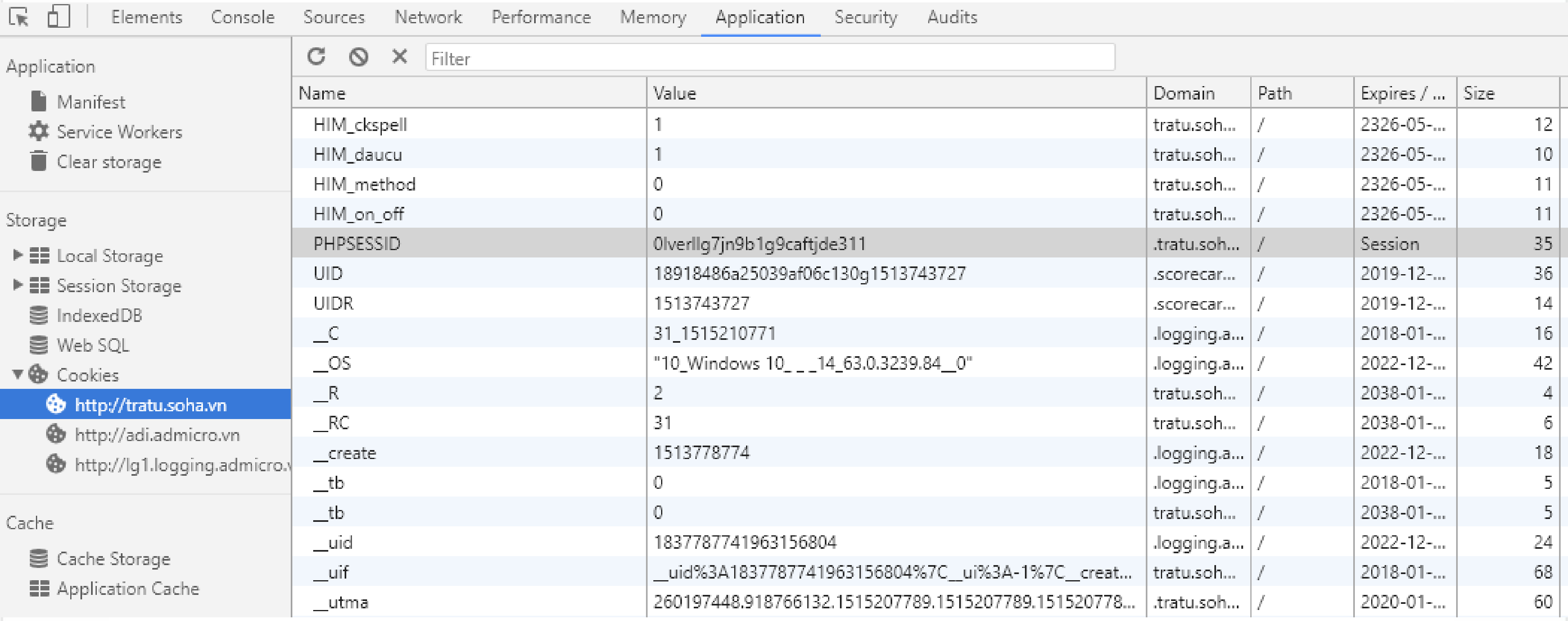
```
$(document).ready(function(){
    if(localStorage.hits){
        localStorage.hits = parseInt(localStorage.hits) +1'
    }else{
        localStorage.hits = 1;
    }

    if(sessionStorage.hits){
        sessionStorage.hits = parseInt(sessionStorage.hits) +1'
    }else{
        sessionStorage.hits = 1;
    }

    alert("Number of hits this browser: " + localStorage.hits +"\n\n" +
            "Number of hits this session: " + sessionStorage.hits );
});
```

How to use Chrome to work with items in the browser

How to view and delete cookies

- You can use Chrome to view or delete cookies.
- Steps to view and delete cookies:
  1. Open Chrome
  2. Press F12 to open developer tools
  3. Click application tab
  4. Select Cookies link in the right panel
  5. You can select to view or delete cookies

How to view and delete cookies (cont.)

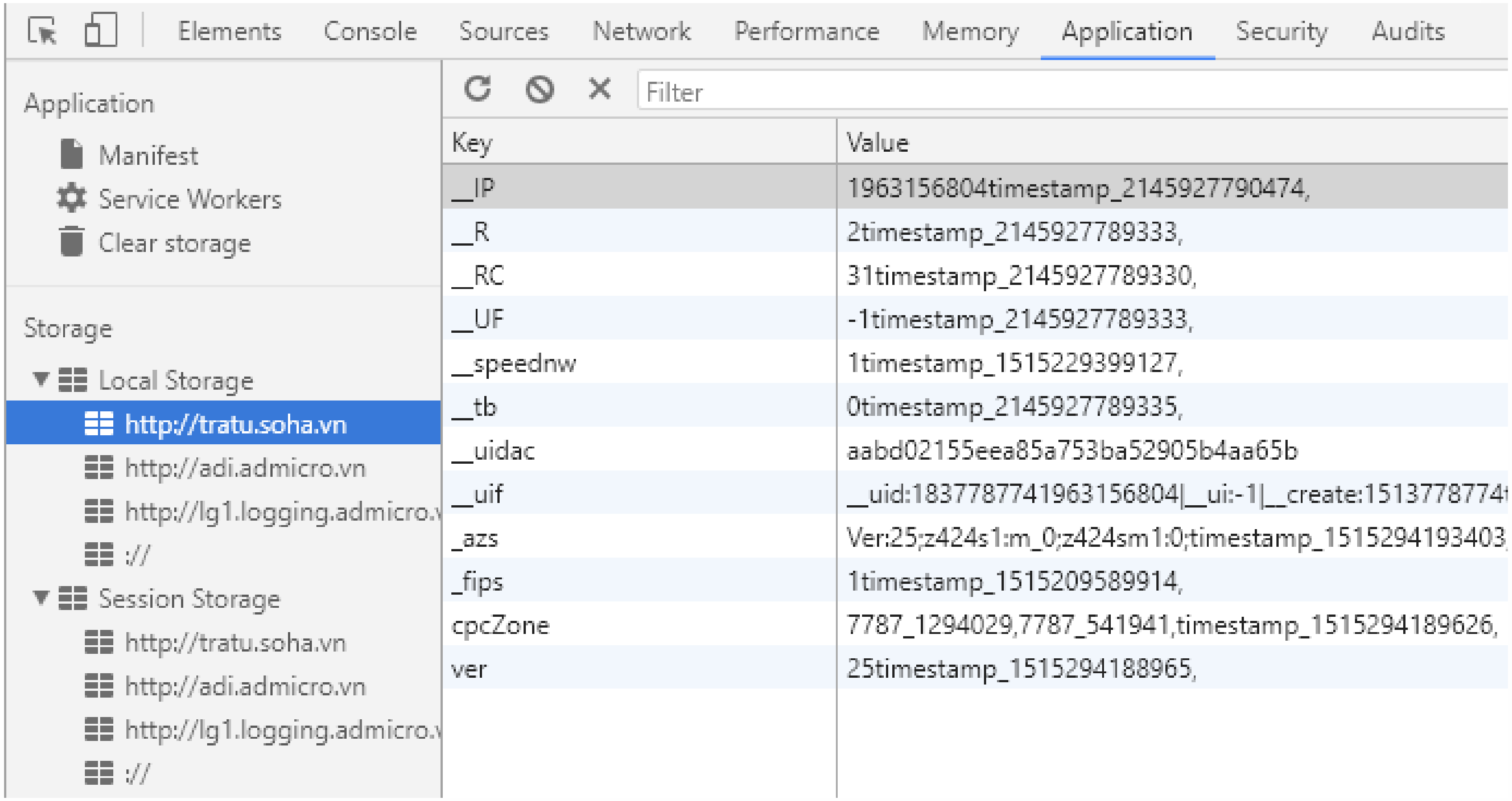


Name	Value	Domain	Path	Expires / ...	Size
HIM_ckspell	1	tratu.soh...	/	2326-05-...	12
HIM_daucu	1	tratu.soh...	/	2326-05-...	10
HIM_method	0	tratu.soh...	/	2326-05-...	11
HIM_on_off	0	tratu.soh...	/	2326-05-...	11
PHPSESSID	0lverllg7jn9b1g9caftjde311	.tratu.soh...	/	Session	35
UID	18918486a25039af06c130g1513743727	.scorecar...	/	2019-12-...	36
UIDR	1513743727	.scorecar...	/	2019-12-...	14
__C	31_1515210771	.logging.a...	/	2018-01-...	16
__OS	"10_Windows 10_ _ _14_63.0.3239.84__0"	.logging.a...	/	2022-12-...	42
__R	2	tratu.soh...	/	2038-01-...	4
__RC	31	tratu.soh...	/	2038-01-...	6
__create	1513778774	.logging.a...	/	2022-12-...	18
__tb	0	.logging.a...	/	2018-01-...	5
__tb	0	tratu.soh...	/	2038-01-...	5
__uid	1837787741963156804	.logging.a...	/	2022-12-...	24
__uif	__uid%3A1837787741963156804%7C__ui%3A-1%7C__creat...	tratu.soh...	/	2018-01-...	68
__utma	260197448.918766132.1515207789.1515207789.15152077 8...	.tratu.soh...	/	2020-01-...	60

How to view, edit and delete items in web storage

- Steps to view, edit and delete items in web storage :
  1. Open Chrome
  2. Press F12 to open developer tools
  3. Click application tab
  4. Select Local Storage or Session Storage link in the right panel
  5. You can select to view, edit or delete items

How to view, edit and delete items in web storage (cont.)



Key	Value
__IP	1963156804timestamp_2145927790474,
__R	2timestamp_2145927789333,
__RC	31timestamp_2145927789330,
__UF	-1timestamp_2145927789333,
__speednw	1timestamp_1515229399127,
__tb	0timestamp_2145927789335,
__uidac	aabd02155eea85a753ba52905b4aa65b
__uif	__uid:1837787741963156804|__ui:-1|__create:1513778774
_azs	Ver:25;z424s1:m_0;z424sm1:0;timestamp_1515294193403
_fips	1timestamp_1515209589914,
cpcZone	7787_1294029,7787_541941,timestamp_1515294189626,
ver	25timestamp_1515294188965,

Summary

- Two objects of browser are **location** and **history** object.
- ***Location object*** lets you work with the URL for a web page. ***History object*** lets you work with the pages that are stored in your browser's history.
- A ***cookie*** is a short text string that is stored by the browser as a name/value pair. Cookies let a web server or web page store information in a user's browser.
- **Web storage** lets the web page use JavaScript to store data in key/value pair like cookie, but it is over limit of cookie.
- You can use chrome to view/edit cookies or view/edit/delete items in web storage.

The End.