COMP 4021

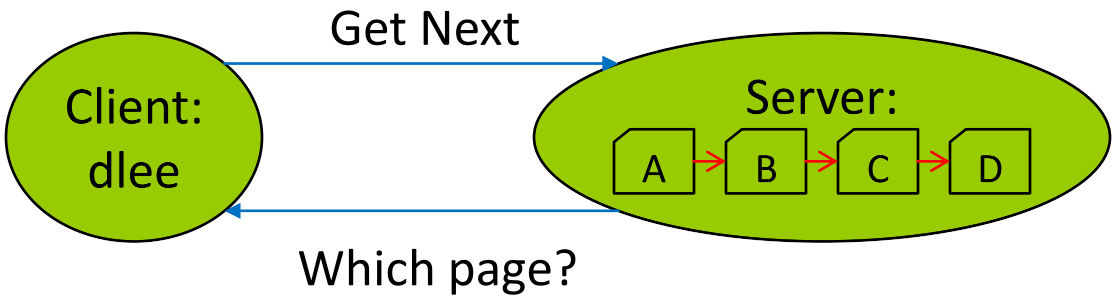
Internet Computing

Cookies

David Rossiter

Stateful Protocol

- The result of a request depends on the “state” of the communicating parties, e.g., the meaning of “next page” depends on what the “current page” is



- Server needs to remember two states: user ID and current page
  - [user=dlee, current_page=B] Get Next => returns C

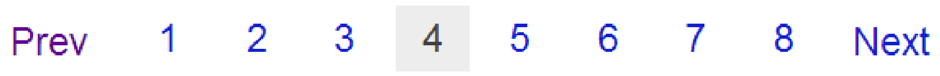
Stateful vs Stateless Protocol

Get a page given exact URL	Stateless	
Get the current page	Stateful	“current” depends on where you are
Get the “next” page	Stateful	“next” depends on where you are

- Stateful protocols are more efficient but stateless protocols are more scalable (i.e., suitable for internet) WHY?

HTTP is Stateless

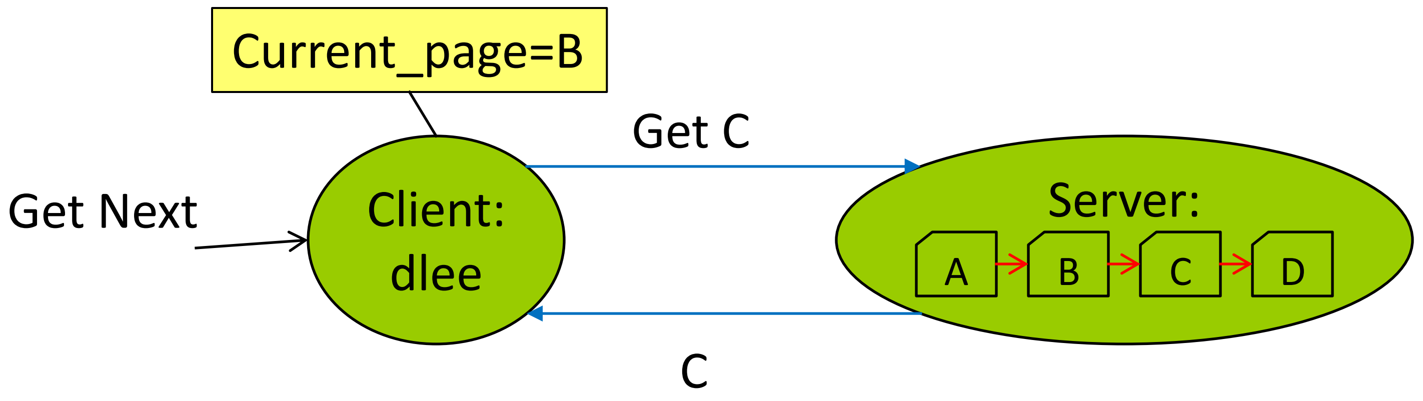
- An exact URL is required to get a page
- The returned page is the same whether or not you send the URL today or tomorrow or via different webpages (of course, page content could have been updated)
- Bing's navigation bar for the result pages



  - Current page showing results from 31 to 40
  - Prev URL: http://www.bing.com/search?q=4021&go=&qs=n&pq=4021 &sc=0-0&sp=-1&sk=&**first=21**&FORM=PQRE
  - Next URL: http://www.bing.com/search?q=4021&go=&qs=n&pq=4021 &sc=0-0&sp=-1&sk=&**first=41**&FORM=PORE

HTTP simulating Stateful Protocol

- Browser remember states; server does not remember states
- [current_page=B] Get Next => Get C



What Are Cookies?

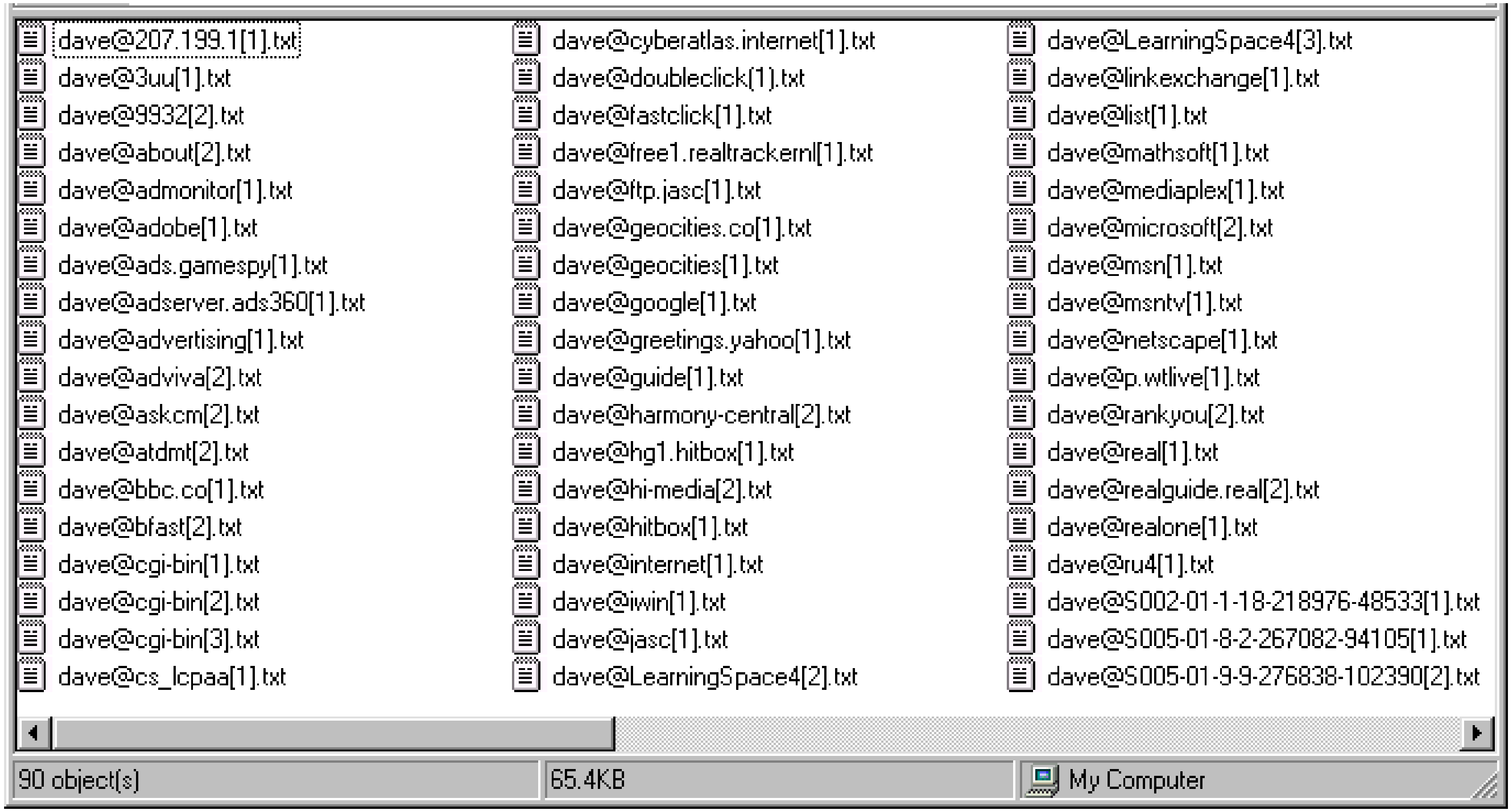
- Basically, cookies are just text messages that allow the browser/server to remember the "states" of the interaction
  - E.g., Whom am I talking to (user name), visit history, etc.
- A web site can store cookies in a browser and later read and modify them
- If you visit the web site again, then JavaScript code in the web page can read the cookies that were stored earlier
- Not just websites – files loaded from local disk can also use cookies (If cookies are handled by client-side JavaScript)

Things Stored in Cookies

- You can store anything you want in a cookie
- Examples of things commonly stored:
  - The date/ time you last visited the web site
  - How many times you have been to the web site
  - What things you clicked on (e.g. books, etc)
  - The highest score so far – for a game

Example Cookie Data - IE

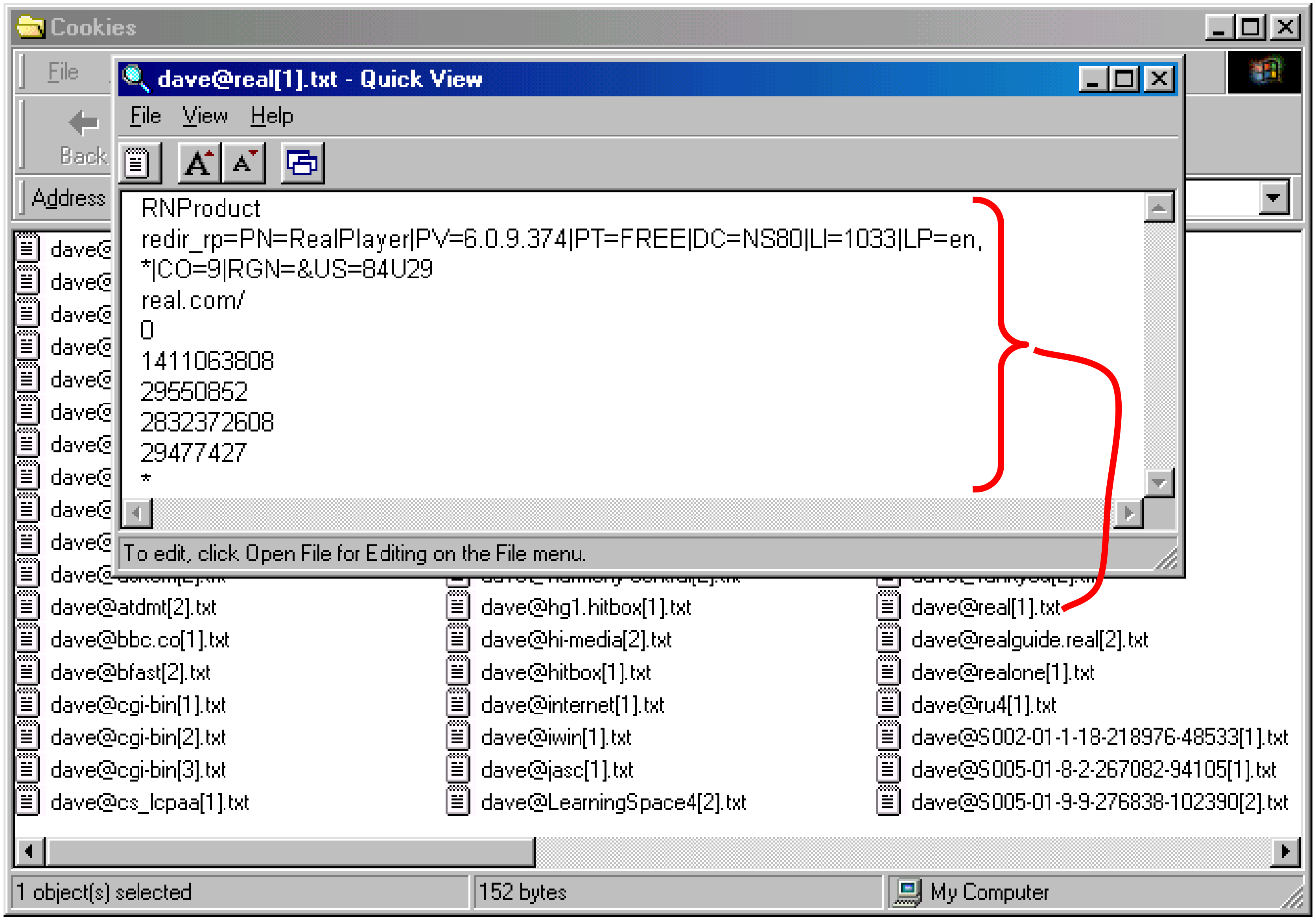
- For IE, each web site is given its own cookies file
- All cookie data for that web site is stored in that file



```
dave@207.199.1[1].txt          dave@cyberatlas.internet[1].txt   dave@LearningSpace4[3].txt
dave@3uu[1].txt                dave@doubleclick[1].txt           dave@linkexchange[1].txt
dave@9932[2].txt               dave@fastclick[1].txt             dave@list[1].txt
dave@about[2].txt              dave@free1.realtrackernl[1].txt   dave@mathsoft[1].txt
dave@admonitor[1].txt          dave@ftp.jasc[1].txt              dave@mediaplex[1].txt
dave@adobe[1].txt              dave@geocities.co[1].txt          dave@microsoft[2].txt
dave@ads.gamespy[1].txt        dave@geocities[1].txt             dave@msn[1].txt
dave@adserver.ads360[1].txt    dave@google[1].txt                dave@msntv[1].txt
dave@advertising[1].txt        dave@greetings.yahoo[1].txt       dave@netscape[1].txt
dave@adviva[2].txt             dave@guide[1].txt                 dave@p.wtlive[1].txt
dave@askcm[2].txt              dave@harmony-central[2].txt       dave@rankyou[2].txt
dave@atdmt[2].txt              dave@hg1.hitbox[1].txt            dave@real[1].txt
dave@bbc.co[1].txt             dave@hi-media[2].txt              dave@realguide.real[2].txt
dave@bfast[2].txt              dave@hitbox[1].txt                dave@realone[1].txt
dave@cgi-bin[1].txt            dave@internet[1].txt              dave@ru4[1].txt
dave@cgi-bin[2].txt            dave@iwin[1].txt                  dave@S002-01-1-18-218976-48533[1].txt
dave@cgi-bin[3].txt            dave@jasc[1].txt                  dave@S005-01-8-2-267082-94105[1].txt
dave@cs_lcpaa[1].txt           dave@LearningSpace4[2].txt        dave@S005-01-9-9-276838-102390[2].txt
```

90 object(s) | 65.4KB | My Computer

Example Cookie Data - IE



```
RNProduct
redir_rp=PN=RealPlayer|PV=6.0.9.374|PT=FREE|DC=NS80|LI=1033|LP=en,
*|CO=9|RGN=&US=84U29
real.com/
0
1411063808
29550852
2832372608
29477427
*
```

To edit, click Open File for Editing on the File menu.

dave@atdmt[2].txt	dave@hg1.hitbox[1].txt	dave@real[1].txt
dave@bbc.co[1].txt	dave@hi-media[2].txt	dave@realguide.real[2].txt
dave@bfast[2].txt	dave@hitbox[1].txt	dave@realone[1].txt
dave@cgi-bin[1].txt	dave@internet[1].txt	dave@ru4[1].txt
dave@cgi-bin[2].txt	dave@iwin[1].txt	dave@S002-01-1-18-218976-48533[1].txt
dave@cgi-bin[3].txt	dave@jasc[1].txt	dave@S005-01-8-2-267082-94105[1].txt
dave@cs_lcpaa[1].txt	dave@LearningSpace4[2].txt	dave@S005-01-9-9-276838-102390[2].txt

1 object(s) selected | 152 bytes | My Computer

Cookie Limitations

- A browser can hold up to 300 cookies (or 4 KB per cookie)
- There is a maximum of 20 cookies per web site
  - If you already have 20 cookies and you add a new cookie the browser finds the oldest cookie for that web site and throws it away
- A cookie has a byte size limitation of 4096 bytes (4KB)
  - If you add a new cookie and this makes the size larger than 4kB, then the browser finds the oldest cookie for that web site and throws it away
- All of these limitations may be different from version to version and from browser to browser

Using JavaScript for Cookies

- JavaScript controls cookies through the document.cookie property

- You can read cookies by reading it
- You can make a cookie by changing it

Reading the Cookies

- document.cookie is a string containing all of the cookies associated with the current web site
- To see all the cookies do: alert(document.cookie)
- If document.cookie has three cookies it will have this structure:

cookieName1=cookieValue1;cookieName2=cookieValue2;cookieName3=cookieValue3

  - For example: name=dave;score=8900;total_time=46
- So you need JavaScript string handling to extract the individual data out of the string

Saving/Updating a Cookie

- Because document.cookie is just a string, changing it is straightforward
- For example, the following JavaScript statement sets two cookies:
  document.cookie = "name=peter;score=260";

Example JavaScript

- Put the date/time in a cookie, and shows it in the web page

```
<head>   <script>
  var today = new Date();
  now = today.getHours() + ":" + today.getMinutes() + ":" +
     today.getSeconds();

  document.cookie = "last_visit=" + now;
  document.writeln(document.cookie );
</script>  </head>
```

Same as “write” but with a newline

Cookie Expiry Time

- It is a good idea to also set a time for the cookie to ‘die’ (expire)
- Here a cookie is made with a specific time to expire:

```
document.cookie = "name=peter;
  expires=Tue, 17-Mar-08 00:00:01 GMT";
```

Deleting a Cookie

- The way to delete a cookie is to set the date/time of the cookie to a date/time that has already finished (i.e. 1970, or one second ago, or one hour ago)
- The browser will then automatically remove the cookie
- For example:

```
document.cookie = "name=peter;
  expires=Thu, 01-Jan-70 00:00:01 GMT";
```

Altering a Cookie

- What if you have already made a cookie, but now you want to change it?
- Cookies can be altered by simply
  - reading the document.cookie string
  - changing the string as appropriate
  - copying the string back to document.cookie

Functions for Handling Cookies

- You can define functions to help with handling cookies
- The source code for a set of routines for handling cookies is shown on the next few slides
  - setCookie()
  - getCookie()
  - deleteCookie()

Cookie Handling - setCookie

```
function setCookie(name, value, expires, path, domain, secure) {
    var curCookie = name + "=" + escape(value) +
      ((expires) ? "; expires=" + expires.toGMTString() : "") +
      ((path) ? "; path=" + path : "") +
      ((domain) ? "; domain=" + domain : "") +
      ((secure) ? "; secure" : "");
    document.cookie = curCookie;    }
```

- This code creates a cookie (using parameters passed to it)
- The expiry time needs to be given to it in milliseconds
- You can see that there are other possible parameters for a cookie not discussed here – path, domain, and secure

Cookie Handling - getCookie

```
function getCookie(name)  {
    var dc = document.cookie;
    var prefix = name + "=";
    var begin = dc.indexOf("; " + prefix);
    if (begin == -1) {
      begin = dc.indexOf(prefix);
      if (begin != 0) return null;
    } else  begin += 2;
    var end = dc.indexOf(";", begin);
    if (end == -1) end = dc.length;
    return unescape(dc.substring(begin + prefix.length, end));    }
```

Use string functions to extract data from the cookie string

cookie not found

Position (or Index) where prefix appears in cookie string

Start and end positions of the name cookie in cookie string

Convert %20 to " ", etc.

Cookie Handling - deleteCookie

- **path** - path of the cookie (must be same as path used to create cookie)
- **domain** - domain of the cookie (must be same as domain used to create cookie)

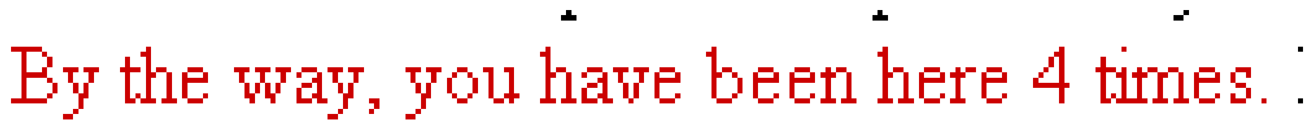
```
function deleteCookie(name, path, domain) {
    if (getCookie(name)) {
     document.cookie = name + "=" +
     ((path) ? "; path=" + path : "") +
     ((domain) ? "; domain=" + domain : "") +
     "; expires=Thu, 01-Jan-70 00:00:01 GMT";    }    }
```

Using the Functions

- Two examples follow
- Example 1 – A web counter
  - Each time you visit the page, it adds one to a counter stored in a cookie
- Example 2 – A name tracker
  - The name of the user is stored in a cookie and is shown every time the page is visited

Example 1 - Web Counter

- Use a cookie to count how many times someone has visited a particular web page
- The following script displays the number of times the user has visited, assuming just one person uses the browser)
- Reload the page to see the counter increment



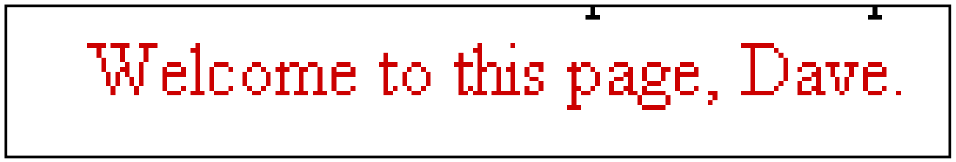
By the way, you have been here 4 times.

Example 1 - Web Counter

```
var now = new Date();       // create an instance of the Date object
now.setTime(now.getTime() + 365*24*60*60*1000);  // expires in 365 days
                                                  // getTime() and setTime() work in msec
var visits = getCookie("counter");
if (!visits) {
    visits = 1;        // if the cookie wasn't found, this is the first visit
    document.write("By the way, this is your first time here.");
    } else {
        visits = parseInt(visits) + 1; // increment the counter
        document.write("By the way, you have been here " + visits + " times.");
    }
setCookie("counter", visits, now); // set the new cookie
                  name,     value,   expire
```

Example 2 - Name Tracker

- The following script asks the user for his/ her name, and "remembers" the input
- It then welcomes the user each time he/ she accesses the page, without asking again for the name


Welcome to this page, Dave.

Example 2 - Name Tracker

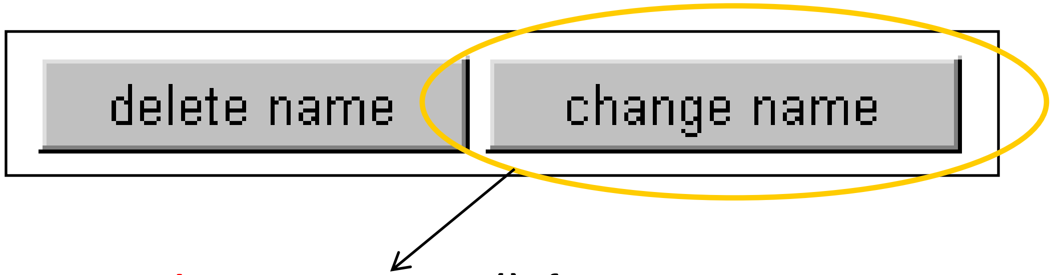
```
var now = new Date();       // create an instance of the Date object

now.setTime(now.getTime() + 365 * 24 * 60 * 60 * 1000);
var username = getCookie("username");
 // if the cookie wasn't found, ask for the name
if (!username) username = prompt("Please enter your name:", "");

setCookie("username", username, now);       // set the new cookie
if (username) {
      document.write("Welcome to this page, " + username + ".");
      setCookie("username", username, now);
      } else document.write("You refused to enter your name.");
```

Example 2 - Name Tracker

- Altering a cookie, e.g., allow user to change to a new name



```
function changeName() {
    var now = new Date();
    // cookie will expire one year later
    now.setTime(now.getTime() + 365 * 24 * 60 * 60 * 1000);
    username = prompt("Please enter your name:", "");
    setCookie("username", username, now);     }
```

Take Home Message

- Cookies is a quick-and-dirty way of storing information (or states) about an interaction
- Sizes of cookies are limited
- Storing cookies as strings make cookies hard to access and maintained
- HTML5 has better support on “local storage”