COMP4021
Internet Computing

Cookies

David Rossiter and Gibson Lam

The Basic Idea of a Cookie

- Basically, a cookie is a small piece of text which is stored in your browser when you go to a web site
- After that, when you load a web page from the same web site, the cookie information is automatically given back to the web site
- In this way the web site can ‘remember’ things about your visit

Use of Cookies

- Cookies can be used in many ways
- For example, you could use cookies to store data for a webpage e.g. storing the highest score of a game
- The information will come back to your JavaScript code the next time the user loads the game
- For simple projects, you could actually forget about using typical JavaScript variables and only use cookies – then the values will be remembered even if the browser is closed

Use of Cookies

- Commonly, cookies are used for tracking what you have done with a web site
- For example, Amazon will track what you search for, and later it show adverts for similar products

Creating Cookies

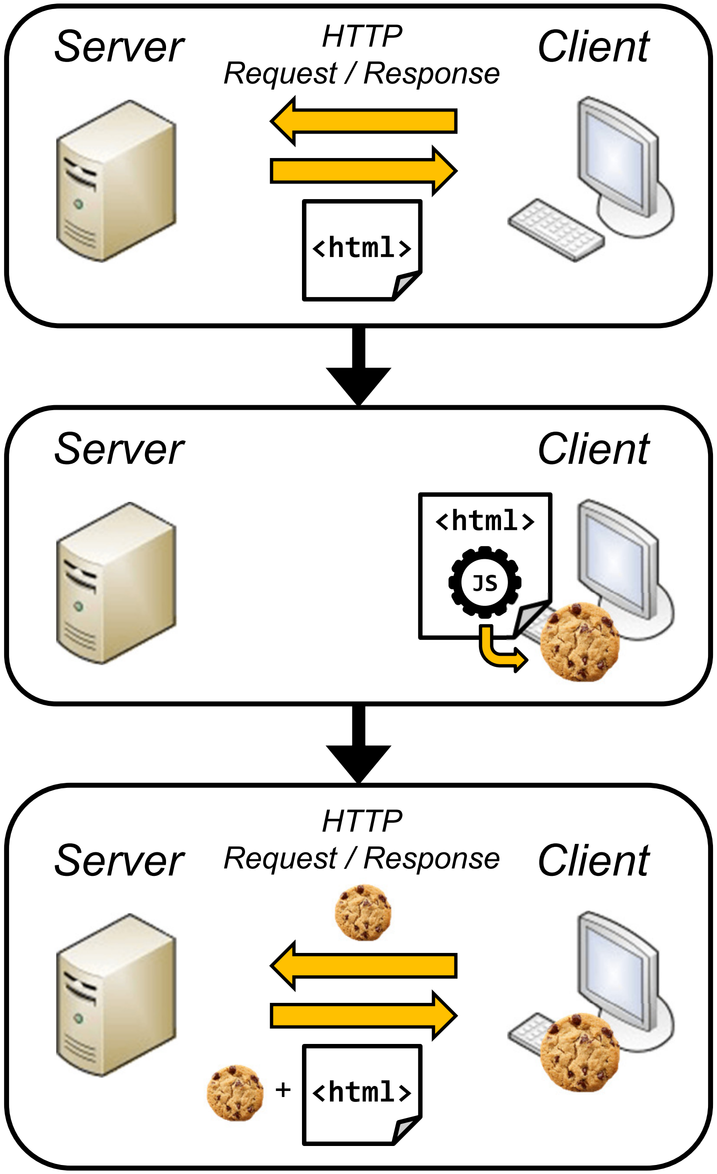
- Cookies can be created by
  1. Using JavaScript in the browser

     For example, you load a web page from a web site, JavaScript code in that web page can create a cookie in the browser which ‘belongs’ to that web site
  2. Using an HTTP header, from the server

     For example, you load a web page from a web site, information in the header from the web site can create a cookie in the browser which ‘belongs’ to that web site
- Different web sites cannot share their cookies
- E.g. if you have a cookie from starwars.com, then you visit disney.com, it will not be sent to disney.com

Creating a Cookie Using JavaScript

1. The browser requests for and gets an HTML page from the server
2. JavaScript in the HTML page creates a cookie in the browser
3. Later, if the browser requests any page from the same web site, the cookie content will be automatically sent to the server by the browser, and it will (probably) come back to the browser again



Making Cookies in JavaScript

- You can easily create a cookie using `document.cookie` in JavaScript
- For example, if you want to create a cookie called “`name`” with a value of “`Dave`”, you can do this:

```
document.cookie = "name=Dave";
```

- A cookie always has a name and a value

The HTML File Has to be on a Server

- If you try to use a local file (=a file on your hard disk) to make a cookie, it won't work
- You must put the HTML page containing the JavaScript on a web server, then load it from the server – then it will work

Creating Multiple Cookies

- You can create many cookies:

```
document.cookie = "name=Dave";
document.cookie = "score=10800";
```

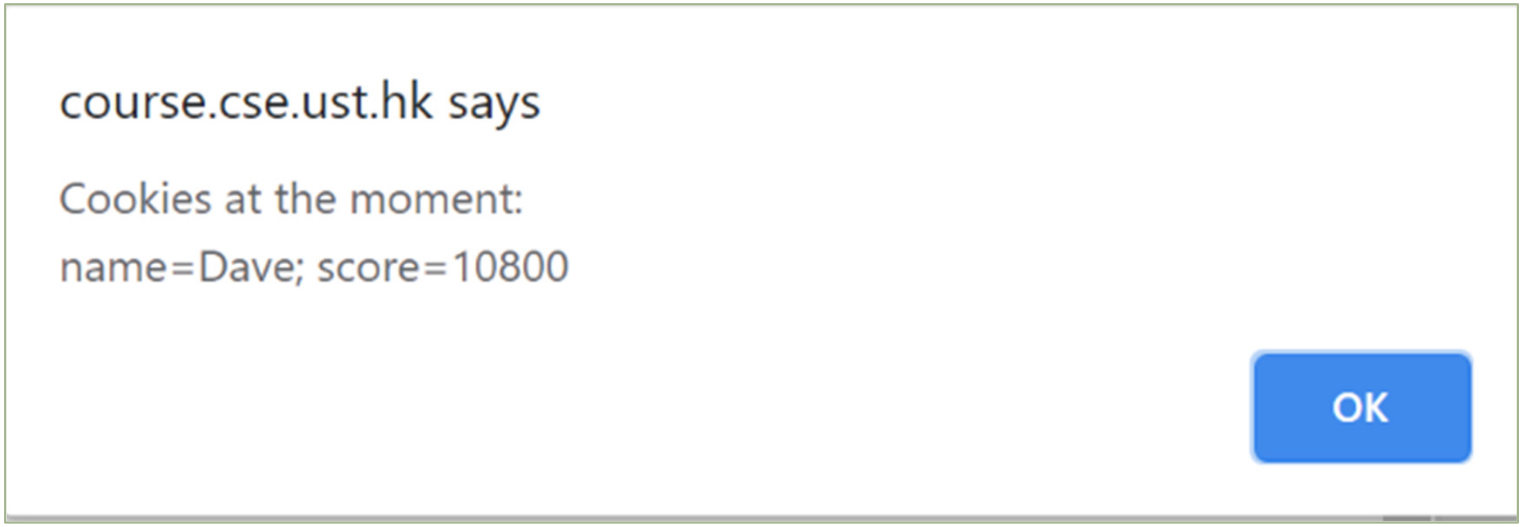
- The above 2 lines of JavaScript code create **two different cookies** – the second cookie created in the code above does not replace the first cookie

Cookies

- Officially, a browser needs to support 20 or more cookies
- Some browsers may allow more than this
- Officially, each cookie can be up to 4096 characters
- Some browsers may allow more than this

Reading Cookies

- To read the cookies, you simply read `document.cookie`
- For example, if you do `alert(document.cookie);` after creating the previous cookies, you will see this:



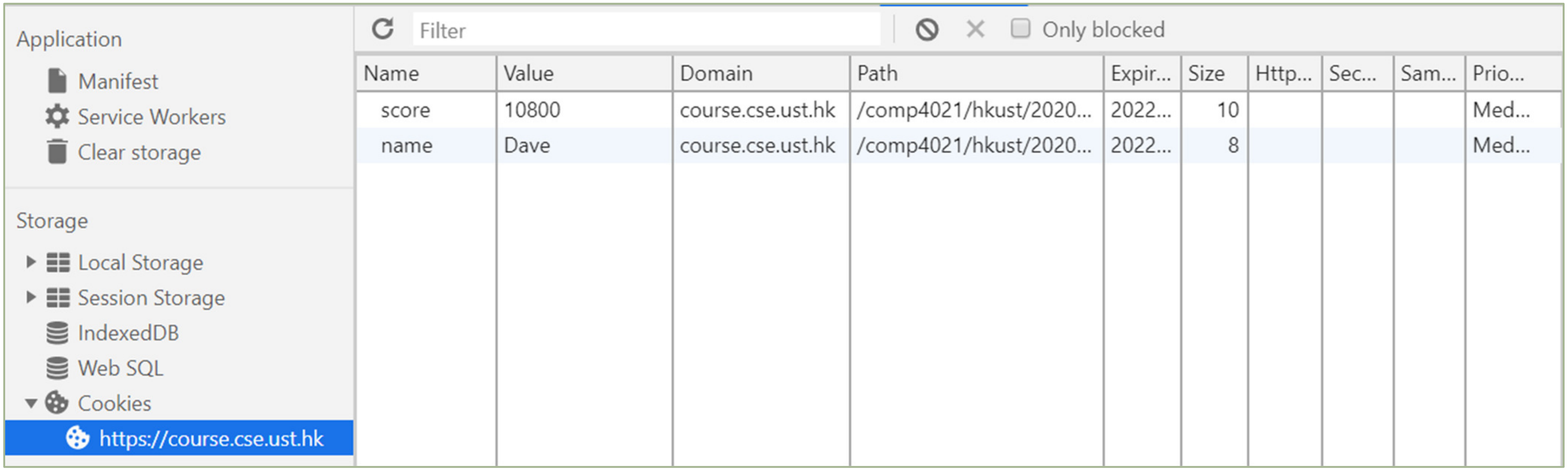
course.cse.ust.hk says

Cookies at the moment:

name=Dave; score=10800

Viewing Cookies in Chrome

- In Chrome you can use the developer tool (Application > Cookies) to look for the currently active cookies:



Name	Value	Domain	Path	Expir...	Size	Http...	Sec...	Sam...	Prio...
score	10800	course.cse.ust.hk	/comp4021/hkust/2020...	2022...	10				Med...
name	Dave	course.cse.ust.hk	/comp4021/hkust/2020...	2022...	8				Med...

The Cookie’s Expiry Date/Time

- If you do not specify the expiry date and time when you create a cookie, it will exist for the length of the current session **expiry by default**
- Simple speaking, the ‘current session’ means ‘until you stop the browser’
- So if you don’t use an expiry date/time the cookie will be gone when you close the browser

How Long the Cookie Lasts

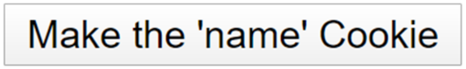
- You can set several things when you create a cookie
- One of the more useful options you can control is the expiry date/time of the cookie
- You can do that like this:

```
document.cookie =
  "difficulty=easy; expires=Sat, 01 Jan 2022 00:00:00 GMT";
```

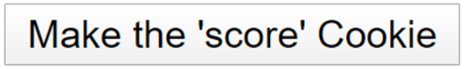
The expiry date and time

Example Program – Buttons 1/3

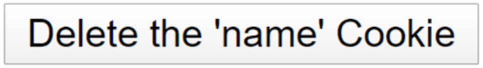
```
<body>
    <button onclick="makeNameCookie()">Make the 'name' Cookie</button>
<br><br>
```



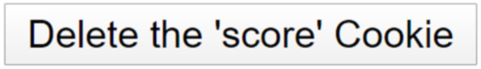
```
    <button onclick="makeScoreCookie()">Make the 'score' Cookie</button>
<br><br>
```



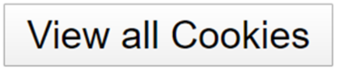
```
    <button onclick="deleteNameCookie()">Delete the 'name' Cookie</button>
<br><br>
```



```
    <button onclick="deleteScoreCookie()">Delete the 'score' Cookie</button>
<br><br>
```

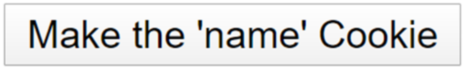


```
    <button onclick="viewAllCookies()">View all Cookies</button>
</body> </html>
```



Example Program 2/3 - Creating Cookies

```
<script>
```



```
    function makeNameCookie() {
        document.cookie =
            "name=Dave; expires=Sat, 01 Jan 2022 00:00:00 UTC";
    }
```

Create a cookie using a future date/time

```
    function makeScoreCookie() {
        document.cookie =
            "score=10800; expires=Sat, 01 Jan 2022 00:00:00 UTC";
    }
```

Example Program 3/3 - Deleting and Showing Cookies

```
function deleteNameCookie() {
    document.cookie =
        "name=anything; expires=Thu, 01 Jan 1970 00:00:00 GMT";
}
```

The value doesn’t matter

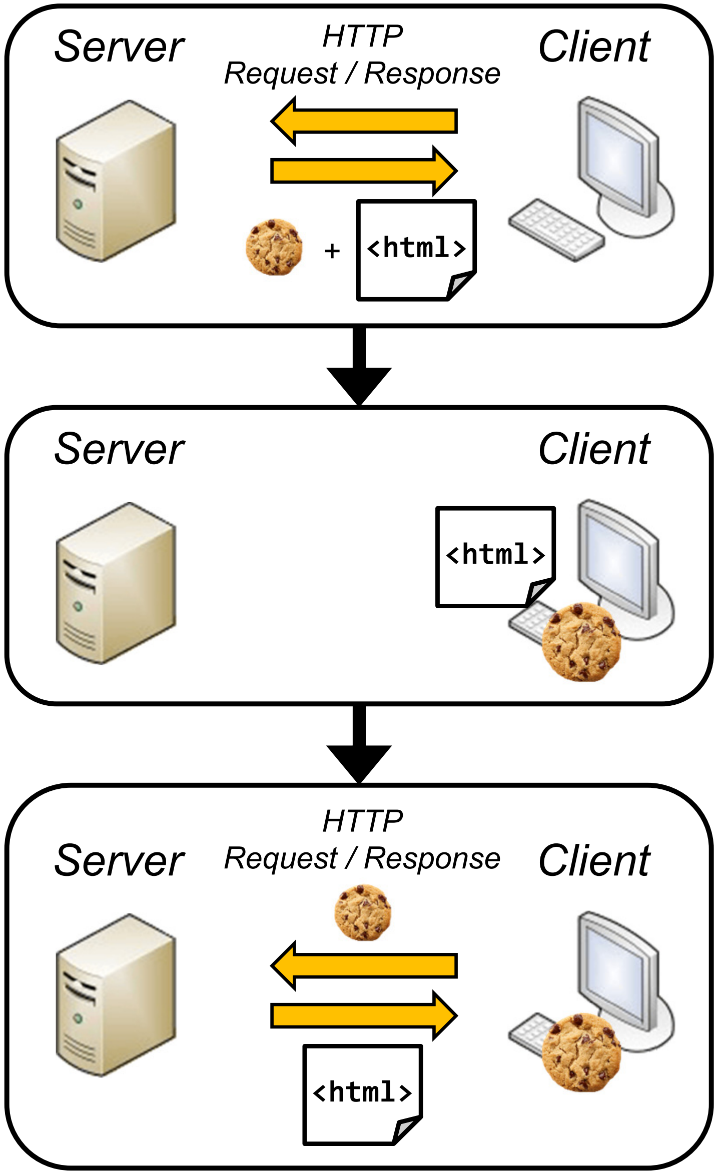
Delete a cookie using a past date/time

```
function deleteScoreCookie() {
    document.cookie =
        "score=anything; expires=Thu, 01 Jan 1970 00:00:00 GMT";
}
```

```
function viewAllCookies() {
    alert("Cookies at the moment:\n" + document.cookie);
}
</script></head>
```

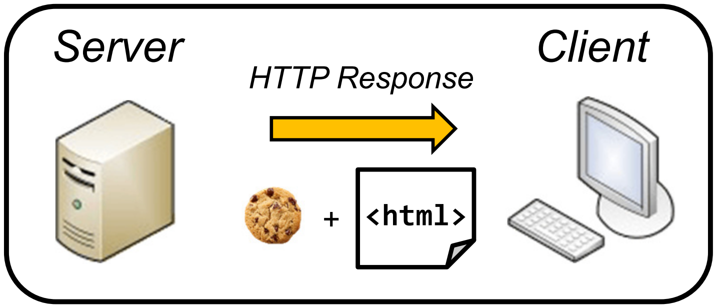
Creating a Cookie Using Server Code

1. The browser requests and gets an HTML page from the server; the server also sends a cookie to the browser
2. The browser stores the cookie automatically
3. Later, if the browser requests any page from the same web site, the cookie will be passed to the server



Cookies in an HTTP Response

- Here is what happens in an HTTP response:
  - A server responds to an HTTP request with the requested HTML file
  - Also, at the same time the server can create cookies in the browser by putting the `Set-Cookie` instruction in the HTTP header



```
HTTP/1.1 200 OK
... some HTTP header info...
Set-Cookie: name=Dave
Content-Type: text/html
...the HTML content...
```

Using Cookies in PHP

- In PHP cookies can be created easily using the `setcookie()` function. Here is an example:

```
setcookie("level", "3");
```

Cookie name *Cookie value*

- Because the function creates an instruction in the HTTP header, you need to run the function before you send any output to the browser

Example PHP Program

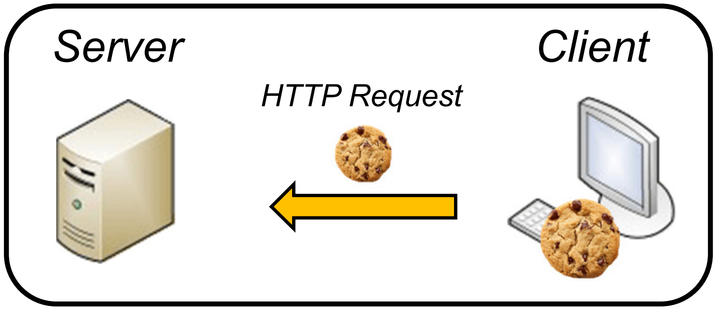
```
<?php
// Set the cookie
setcookie("level", "3");

// Now output an HTML
// page to the browser
?>
```

```
<!DOCTYPE html>
<html>
<head>
    <script>
    function viewCookies() {
        alert("The cookie level=3 was just created\n\n
                    All cookies:\n" + document.cookie);
    }
    </script>
</head>
<body onload="viewCookies()">
</body>
</html>
```

An Example Request With Cookies

- Here is an example HTTP request, assuming the browser has previously stored cookies for the same web site:

either created by javascript or by server-side code



```
GET /register.php HTTP/1.1
...some HTTP header information...
Cookie: name=Dave; score=21; level=3
```

- As you can see, this request has sent three previously stored cookies back to the server

Reading Cookies in PHP

- With PHP you can read the content of the cookies received by the server using `$_COOKIE[]`
- For example, you can read the ‘`score`’ cookie using this code:

```
$_COOKIE["score"]
```

- When you are programming, you can easily show all of the cookies to check what‘s happening:

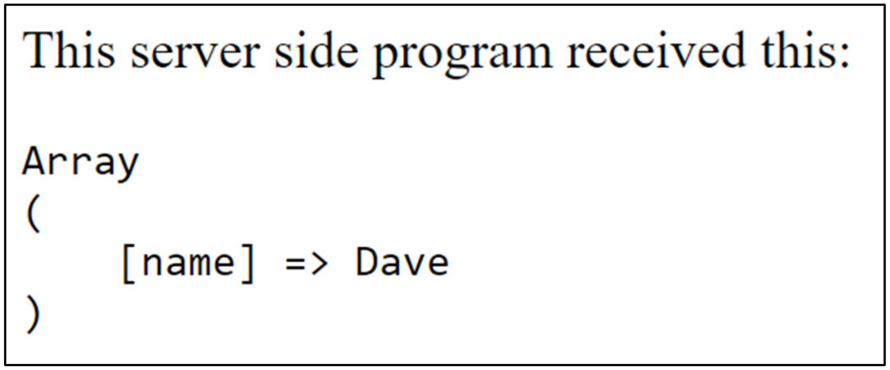
```
print_r($_COOKIE);
```

- `print_r` means ‘print recursive’

Example PHP Program

```
<!DOCTYPE html>
<html>
<head>
    <title>Viewing Cookies on the Server</title>
</head>
<body>
    <p>This server side program received this:</p>
    <pre><?php print_r($_COOKIE) ?></pre>
</body>
</html>
```

Example result:



This server side program received this:

```
Array
(
    [name] => Dave
)
```