Using cookies in JavaScript

Cookies are small items of data, each consisting of a name and a value, stored on behalf of a website by visitors' web browsers. In JavaScript, cookies can be accessed through the `document.cookie` object, but the interface provided by this object is very primitive. Cookies.js is a JavaScript object that allows cookies to be created, retrieved, and deleted through a simple and intuitive interface.

Download Cookies.js

Download one of the files below and either incorporate it into your code or serve it as a separate file.

File	Size	Description
Cookies.js	661 bytes	Minified version
Cookies.src.js	3,957 bytes	Full version, with comments

The code creates a global object, `Cookies`, which has functions to create, retrieve, and delete cookies.

Creating cookies

Cookies can be created using the `set` function, which takes the cookie name and value as parameters:

```
1 // create a cookie
2 Cookies.set('theme', 'green');
```

A cookie set in this way will be deleted when the visitor closes their web browser, will only be accessible within the current directory on the current domain name, and will be sent over both encrypted and unencrypted connections. These features can be controlled through an optional third parameter, which is an object with four possible properties.

The `expiry` property allows the cookie can be given an expiry date. Cookies with an expiry date will persist between browsing sessions, and only be deleted when the expiry date is reached or the visitor instructs the browser to do so. The value of the property can be either a date on which the cookie will expire or a number of seconds after which the cookie should expire:

```
1 // create a cookie that expires after one hour
2 Cookies.set('theme', 'green', {expiry : 3600});
3
4 // create a cookie that expires on 1st January 2030
5 Cookies.set('name', 'Stephen Morley', {expiry : new Date(2030, 0, 1)});
```

Every cookie is accessible only within a particular path, which defaults to the current directory — for example, a cookie set on a page at http://example.com/news/ would by default be accessible from http://example.com/news/archives/ but not from the home page of the site. The `path` property allows an alternative path to be specified:

```
// create a cookie that is accessible anywhere on the site
Cookies.set('theme', 'green', {path : '/'});

// create a cookie that is accessible only within the news directory
Cookies.set('country', 'uk', {path : '/news/'});
```

Every cookie is accessible only on a particular domain, which defaults to the current domain and its subdomains — for example, a cookie set on news.example.com would by default be accessible from archives.news.example.com but not from the main domain example.com. The `domain` property allows an alternative domain to be specified:

```
// create a cookie that is accessible on all subdomains of example.com
Cookies.set('theme', 'green', {domain : '.example.com'});
```

Finally, the `secure` property can be set to true to instruct the browser to send the cookie only over encrypted connections:

```
// create a cookie that is sent only over encrypted connections
Cookies.set('theme', 'green', {secure : true});
```

Retrieving cookies

The value of a cookie can be retrieved using the `get` function, which takes the cookie name as a parameter:

```
// retrieve the value of the theme cookie
var theme =  Cookies.get('theme');
```

If there is no cookie with the specified name, the value `undefined` is returned. There may be more than one cookie with the same name if they were set for different paths or subdomains. In this case the get function returns the most specific cookie (the one set for the longest path). Passing `true` as a second parameter to the `get` function will cause it to return an array containing the values of all cookies with the specified name, in order of specificity:

```
// retrieve the values of any theme cookies
var themes =  Cookies.get('theme', true);
```

Deleting cookies

A cookie can be deleted using the `clear` function, which takes the cookie name as a parameter:

```
// delete the theme cookie
Cookies.clear('theme');
```

If the cookie was set for a path or domain other than the current path and domain, these must be passed to the `clear` function through its optional second parameter:

```
// delete the site-wide theme cookie
Cookies.clear(
    'theme',
    {
      path   : '/',
      domain : '.example.com'
    });
```