Cookies

What is a cookie?

- You may have heard of cookies before!
- Sometimes they are incorrectly categorized as a **bad thing**
- For web developers, however, they can be a very useful tool

What is a cookie?

- A cookie is a piece of browser technology that allows websites to store small key-value pairs of data on a user's local machine
- Every cookie is set to expire eventually, removing itself from the users' machine
- May also be referred to as a "web cookie", "browser cookie", or "HTTP cookie"

Cookie use cases

- Saving user preferences for a web app - like the status of a sliding navigation element (open or closed)
- Remembering that a user is logged in by giving them an access key that eventually expires
- Track where a user has gone on your website and where they came from

Cookie Caveats

- Users can disable cookies or erase them since they live on their machines, so it's not good to rely on them
- Values can also be changed by users, which can interfere with site security - see the next slide for more on this
- The EU passed some strict laws stating that you have to allow users of your website to deny the use of cookies if they'd prefer - so make sure you're compliant!

Using Cookies

- Cookies are a native browser feature and can be used with plain JavaScript.
- Many developers, however, use a library wrapper around the cookie functionality for ease of use and cross-browser compatibility

Native Cookie Code

- Cookies are stored in the `document` object along with all other parts of the DOM
- To add a new cookie, give a key-value pair:

```
document.cookie = "hello=test"
```

- Try running this code in the JavaScript console, then going to (in Chrome Dev Tools) Applications -> Cookies (under the "Storage" section). You should see your key value pair!

Using cookies.js

- `Cookies.js` is an incredibly light wrapper around basic cookie functionality
- It doesn't even require jQuery - it only needs JavaScript and cookies enabled to work.

  github.com/ScottHamper/Cookies

Serve a local directory with WEBrick

You'll need a webserver running to store cookies locally. It's easy to serve a local directory over HTTP with WEBrick. Just run the following command.

```
echo -e '\nalias served="ruby -run -e httpd . -p 5000"' >> ~/.bash_profile && source ~/.bash_profile
```

Then cd to the directory you want to serve and type 'served'. Point your browser to http://localhost:5000 and voilà!

Source: coderwall.com

Using cookies.js

- First, require `Cookies.js` inside of your HTML file like you would any other JavaScript library (i.e. jQuery)

```
<script src="/path/to/cookies.js"></script>
```

- Open the html file in your browser and try to run some `Cookies.js` commands from the console:

```
Cookies.set("Test Key", "Test Value")
Cookies.get("Test Key")
// > "Test Value"
```

Exercises

- Create a web page that asks a user for their name using a form. When they click a button, have their name stored in a cookie. When they refresh the page, if their name has been set, welcome them to the page by retrieving the stored cookie value.
- On a separate page, create a basic jQuery animation. Don't show the animation if the user has already been cookied/been to the page.