Program State

Lesson Time: 30 Minutes

Which Level is Mario On?

Before we continue on to the next lesson, first we need to talk about Program State, or simply "**State**". To understand **State,** we'll use one of the most famous video games of all time to illustrate it - Super Mario Brothers for the Nintendo NES.

Imagine you are playing Super Mario Brothers. At any time you can press the pause button and pause the game. When you press pause, the game is frozen in time. Everything that is happening in that moment is program state.

Examples of state in Super Mario Brothers

- Right now, is Mario big or small?
- Right now, Is Mario jumping, running, walking, or swimming?
- Right now, which level is Mario on?
- Right now, how many lives does Mario have?

We unpause the game, play for a few more seconds, and all of the state of the program has changed.

The management of state in all programming is tough, and it's even tougher in web apps.

Let's look at examples of state in a web application.

- Right now, how many users are online using your site?
- Right now, how many users are first time visitors?
- Right now, what was the last thing the user has clicked on?
- Right now, has the user logged into the website, or browsing as a guest?

Imagine two people. The first person's name is Browser, everyone calls her WB. The second person's name is Server, everyone calls him Sir. WB and Sir work together and have to talk to each other all of the time, but they have one big problem--Sir can't remember anything he tells WB. As soon as Sir tells WB anything, he forgets it! He doesn't even remember meeting WB before, and couldn't tell you if he's ever seen her before!

Fortunately, WB and Sir have found a make to make their situation work. Anytime Sir tells WB something important, WB writes it down. Then, when WB and SIr talk again, WB shows Sir the notes of what they talked about last time. Sir immediately remembers and they pick up work right where they left off.

HTTP protocol is a **stateless protocol**. That means, HTTP can not tell the web browser anything about the state of the server, and cannot tell the server anything about the browser. .

So how do we bridge the gulf between the browser and the server to let them share information? There are two technologies we can use that we'll talk about in the next lesson.

1. Cookies
2. Web Storage

Key Terms	State, Cookies, Stateless, Stateful, Cookies, Web Storage
Lesson Files	
Additional Resources	https://www.atlantic.net/managed-server-hosting/what-is-stateless-stateful-models-web-development/
Further Learning	

Cookies and Web Storage

Lesson Time: 30 Minutes

Cookies are the oldest way to share information between the web browser and the web server. Cookies are small text files a web server places on your computer. Any kind of information could be stored in the cookie, such as your username, the last time you visited the site, or preferences about settings on the site.

Cookies are created on the web server, and passed to your browser, where they are stored on the computer. JS can read the cookies saved on your computer, as well as create new ones too with the **document.cookie** property.

Cookies by default are deleted when the browser window closes. However, the cookie can be "kept alive" and remain on your computer if an expiration date has been set on the cookie.

Web Storage is a newer way of sharing information between the browser and the server. Web storage has two modes

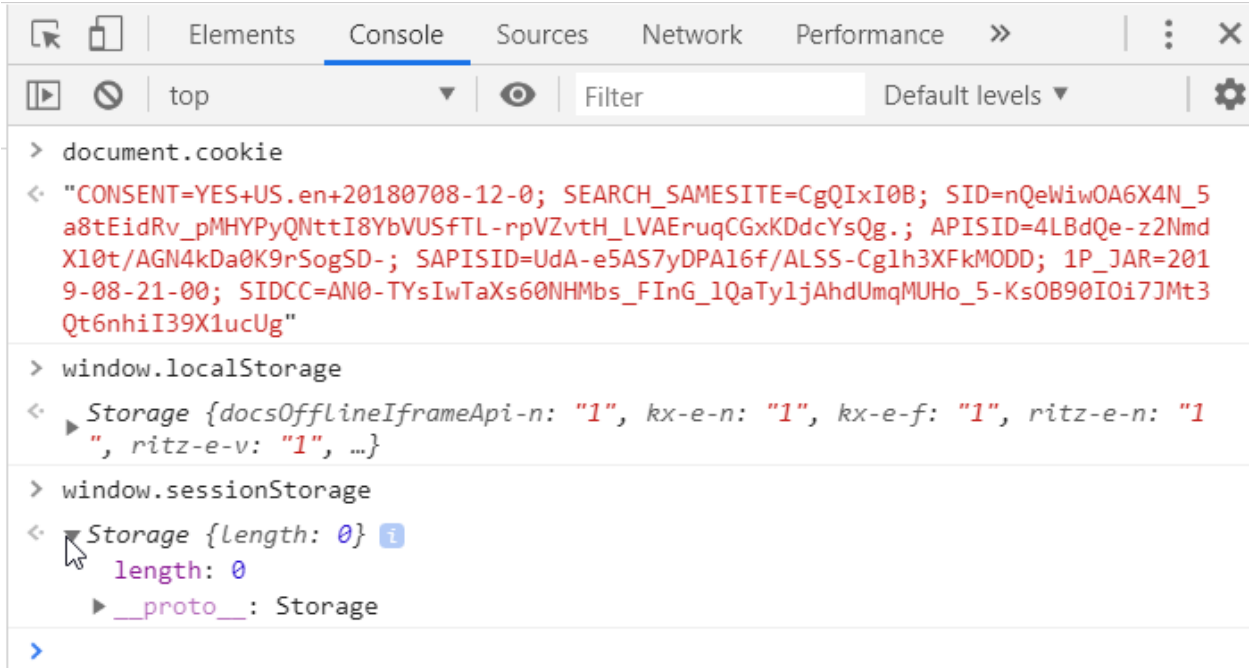
1. localStorage. Data stored in browser is stored forever
2. sessionStorage. Data stored in the browser is deleted when the browser window is closed.

JS can access web storage with the window.localStorage and window.sessionStorage properties.

To see this in action, do these steps in your web browser.

1. Visit a site you've previously visited before.
2. In Chrome, right click the page and choose Inspect
3. On the developer tools window, click Console to open the JS console
4. Type **document.cookie** and press enter The cookies for this site will appear.
5. Type **window.localStorage.** Data this site has permanently saved to your browser will appear.

6. Type **window.sessionStorage.** Data this site has saved to your browser for just this visit will appear.



The general flow works like this:

1. On the SERVER BACKEND, the server sends cookies and/or pushes data down to Web Storage localStorage or sessionStorage.
2. As soon as the server has sent the data & closed the connection, the server forgets. Remember, HTTP is stateless.
3. When needed, JS reads cookie and Web Storage data, and sends it back to the server.
4. The server “remembers” you after reading the data pushed by JS.