State Management
Cookies and Sessions

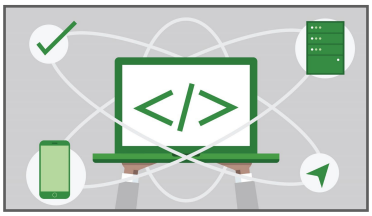


Table of Content

HTTP Cookies
Usage and Control

What Are Cookies?

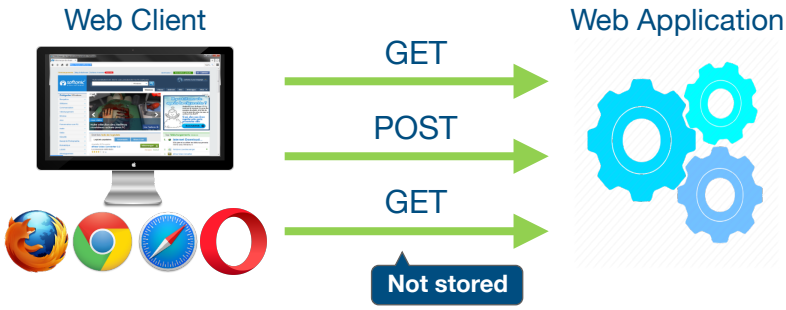
- A **small file** of **plain text** with no executable code
  - Sent by the server to the client's browser
  - Stored by the browser on the client's device (computer, tablet, etc.)
  - Hold small piece of data for a particular client and a website

What Are Cookies Used for?

- **Session management**
  - Logins, shopping carts, game scores, or anything else in the server should remember
- **Personalization**
  - User preferences, themes, and other custom settings
- **Tracking** (Third Party Cookies - till end of year 2023)
  - Recording and analyzing user behaviour

Session Management

- The HTTP protocol is **stateless**
  - It **doesn't store** information about the requests



Stateless HTTP - the Problem

- The **server does not know** if two requests come from the same client
- **State management** problems
  - Navigation through pages requires **authentication each time**
  - Information about the pages is lost **between the requests**
  - **Harder personalization** of page functionality

Stateless HTTP - the Cookie Solution

- A reliable **mechanism** for websites to **remember stateful information**
  - to know whether the user is **logged in or not**
  - to know **which account** the user is logged in with
  - to record the user's **browsing activity**
  - to remember pieces of information **previously entered** into form fields (usernames, passwords, etc.)

How Are Cookies Used?

- The response holds the cookies to be saved within the **Set-Cookie** header

```
HTTP/1.1 200 OK
Set-Cookie: lang=en
```

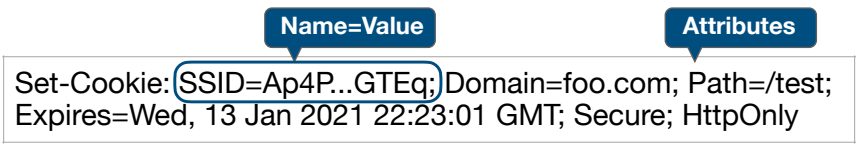
- The request holds the specific web site cookie within the **Cookie** header

```
GET /index HTTP/1.1
Cookie: lang=en
```

Server-Client Cookies Exchange

Cookie Structure

- The cookie consists of **Name**, **Value** and **Attributes** (optional)
- The attributes are **key-value pairs** with additional information
- Attributes are not included in the requests
- Attributes are used by the client to control the cookies



Scope

- Defined by the attributes **Domain** and **Path**
- **Domain** – defines the website that the cookie belongs to
- **Path** – Indicates a **URL** path that must exist in the requested resource before sending the **Cookie** header

```
Set-Cookie: SSID=Ap4P...GTEq; Domain=foo.com; Path=/test;
Expires=Wed, 13 Jan 2021 22:23:01 GMT; Secure; HttpOnly
```

Lifetime

- Defined by the attributes **Expires** and **Max-Age**
- **Expires** – defines the date that the browser should delete the cookie
- By default the cookies are deleted after the end of the session
- **Max-Age** – interval of seconds before the cookie is deleted

```
Set-Cookie: SSID=Ap4P...GTEq; Domain=foo.com; Path=/test;
Expires=Wed, 13 Jan 2021 22:23:01 GMT; Secure; HttpOnly
```

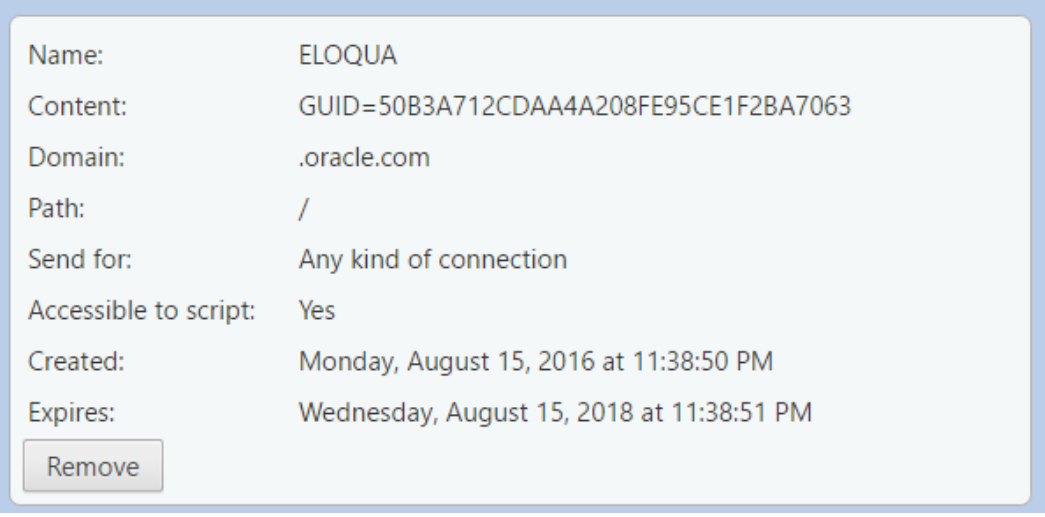
Security

- Security – flags do not have associated values
- **Security** – tells the browser to use cookies only via secure/encrypted connections
- **HttpOnly** – defines that the cookie cannot be accessed via client-side scripting languages

```
Set-Cookie: SSID=Ap4P...GTEq; Domain=foo.com; Path=/test;
Expires=Wed, 13 Jan 2021 22:23:01 GMT; Secure; HttpOnly
```

What is in the Cookie?

- The cookie file contains a table with **key-value** pairs



Examine Your Cookies

- Most cookies are stored in a **RDBMS**, usually **SQLite**
- Download **SQLite browser** from here

- Location of Mozilla cookies

```
C:\Users\{username}\AppData\Roaming\Mozilla\Firefox\Profiles\
{name}.default\cookies.sqlite
```

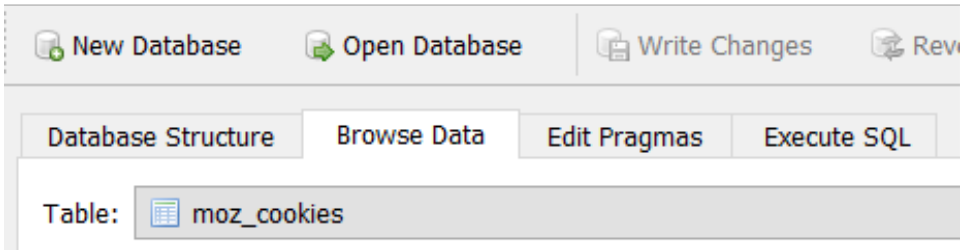
- MacOS:

```
/Users/{username}/Library/Application\ Support/Firefox/Profiles/{name}.default/
cookies.sqlite
```

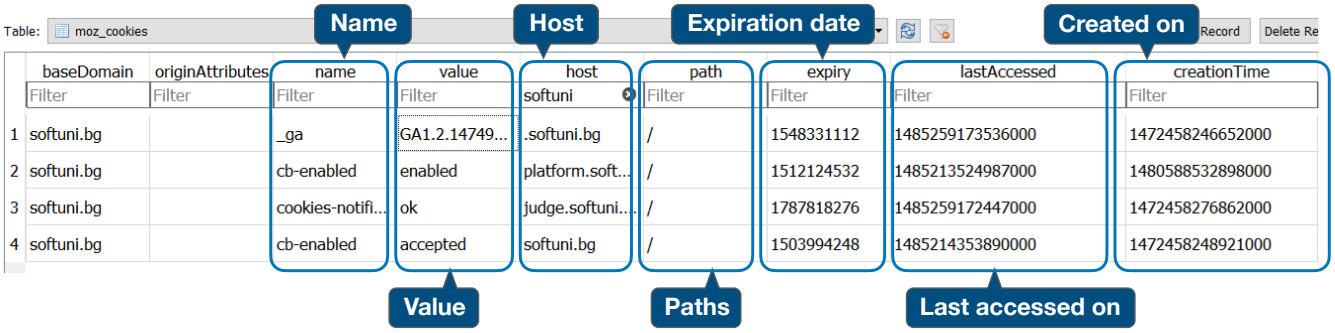
- Location of Chrome cookies

```
C:\Users\{username}\AppData\Local\Google\Chrome\UserData\Default\Cookies
```

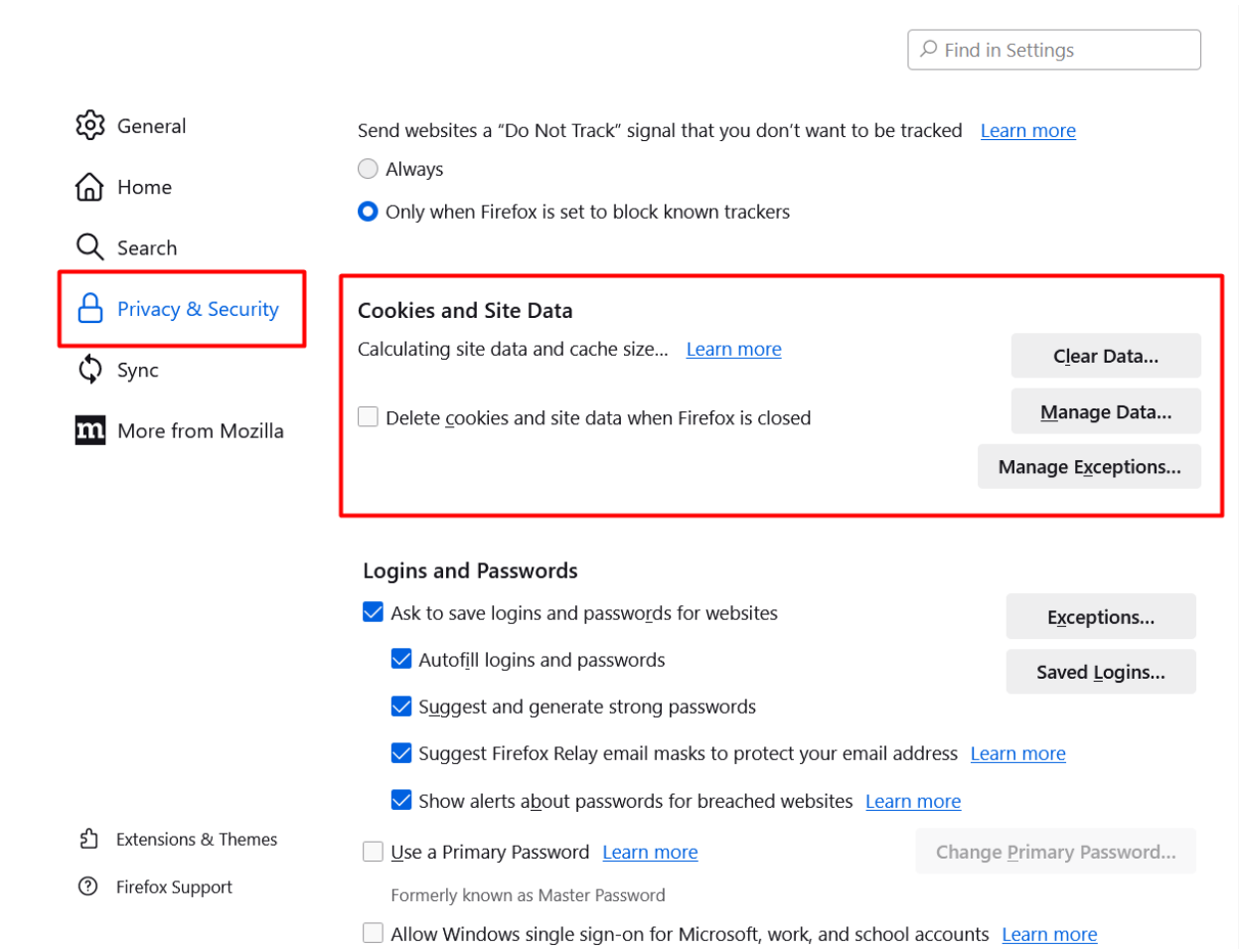
- Open the file with the SQLite browser

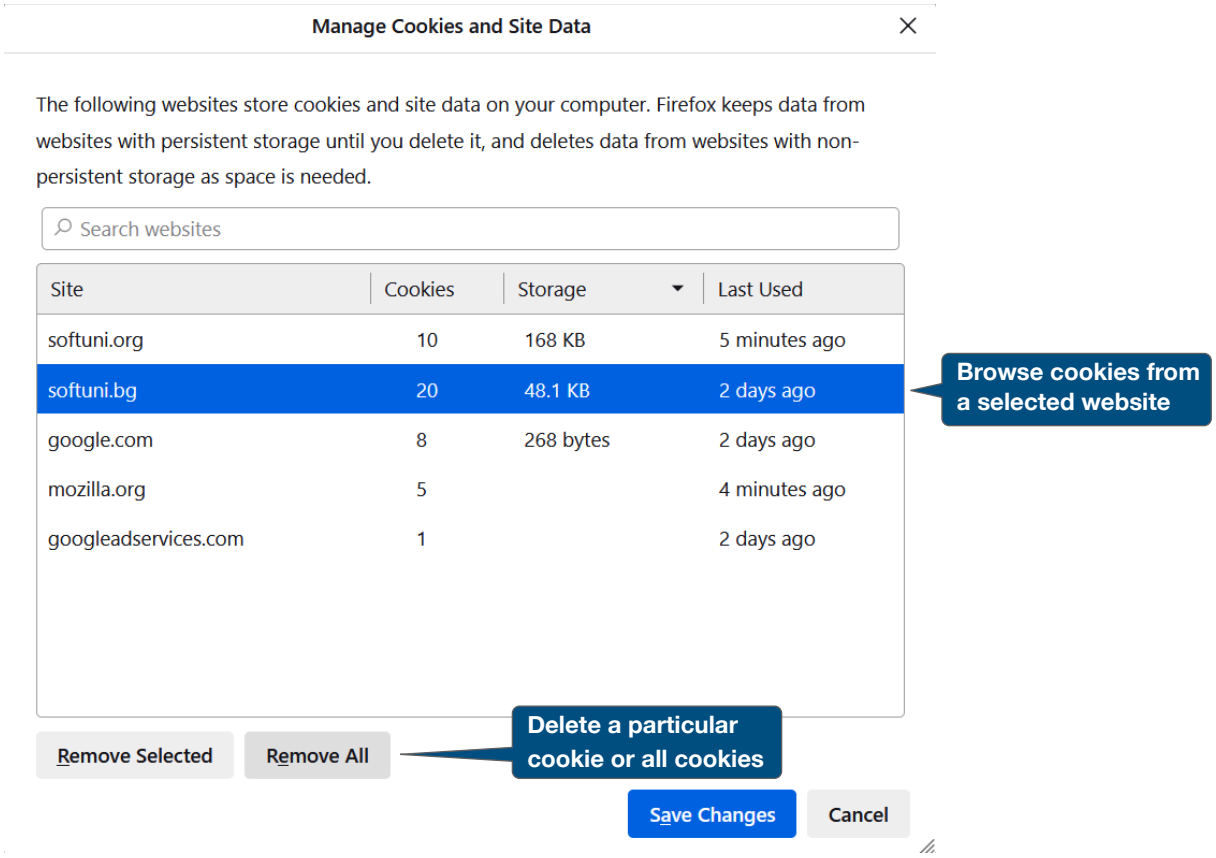


- Browse the cookies table

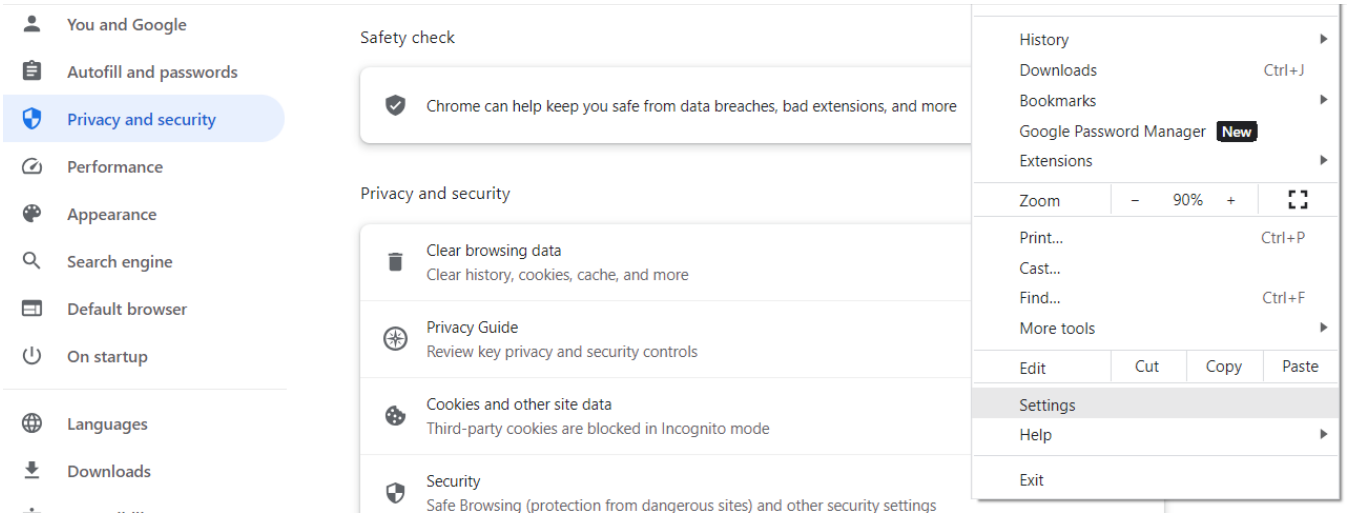


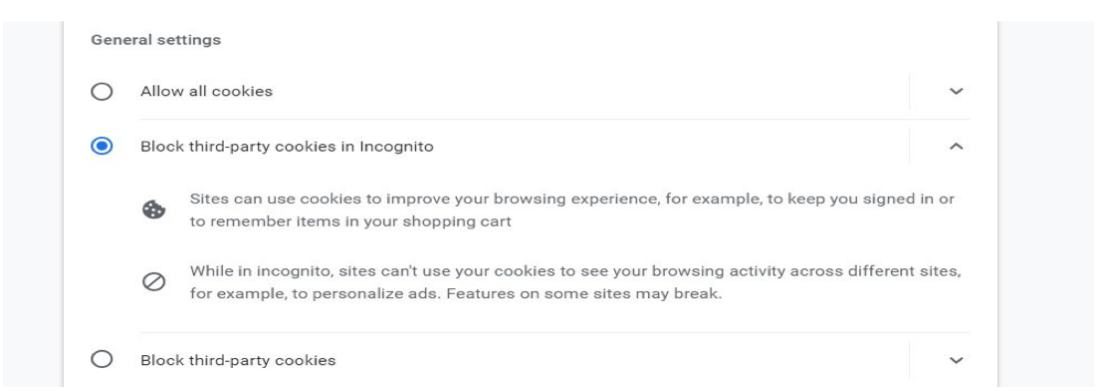
Control Your Cookies – Firefox Browser





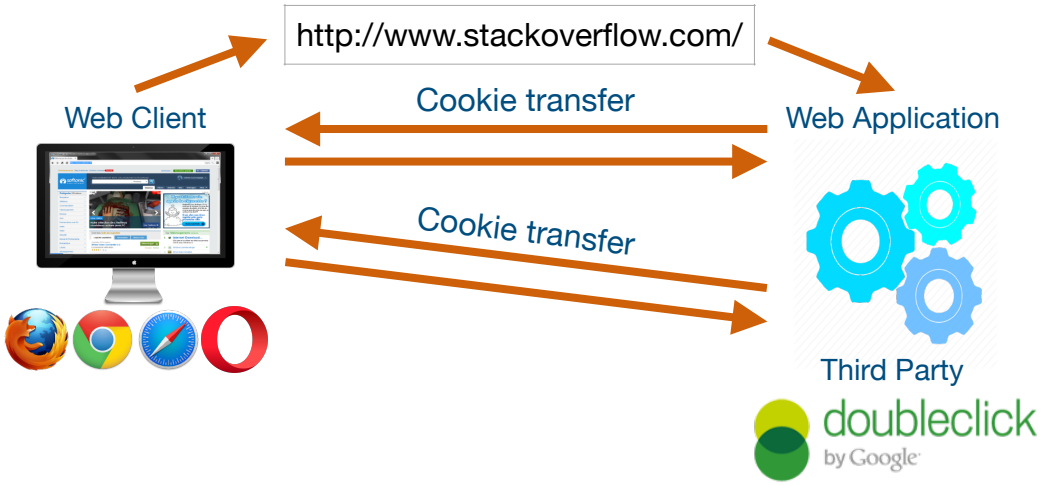
Control Your Cookies – Chrome Browser





Third Party Cookies

- Cookies stored by an **external party** (different **domain**)
- Mainly used for advertising and tracking across the web
- By the end of 2023, Google will stop the use of third-party cookies



Web Storage

- Data is stored in **local storage** objects that have no expiration dates.
- Data is stored for one session in a **session storage** object (data is lost when the browser tab is closed).

Local Storage

- Pros (advantages)
  - There is **no expiration date** for the data kept in local storage.
  - The storage limitation is approximately **10MB**.
  - Data from local storage is **never** sent to the server
- Cons (disadvantages)
  - Since local storage data is in plain text, it is not designed to be secure.
  - Since the data is restricted to strings, serialization is required.
  - Only the client side, not the server side, is capable of reading data.

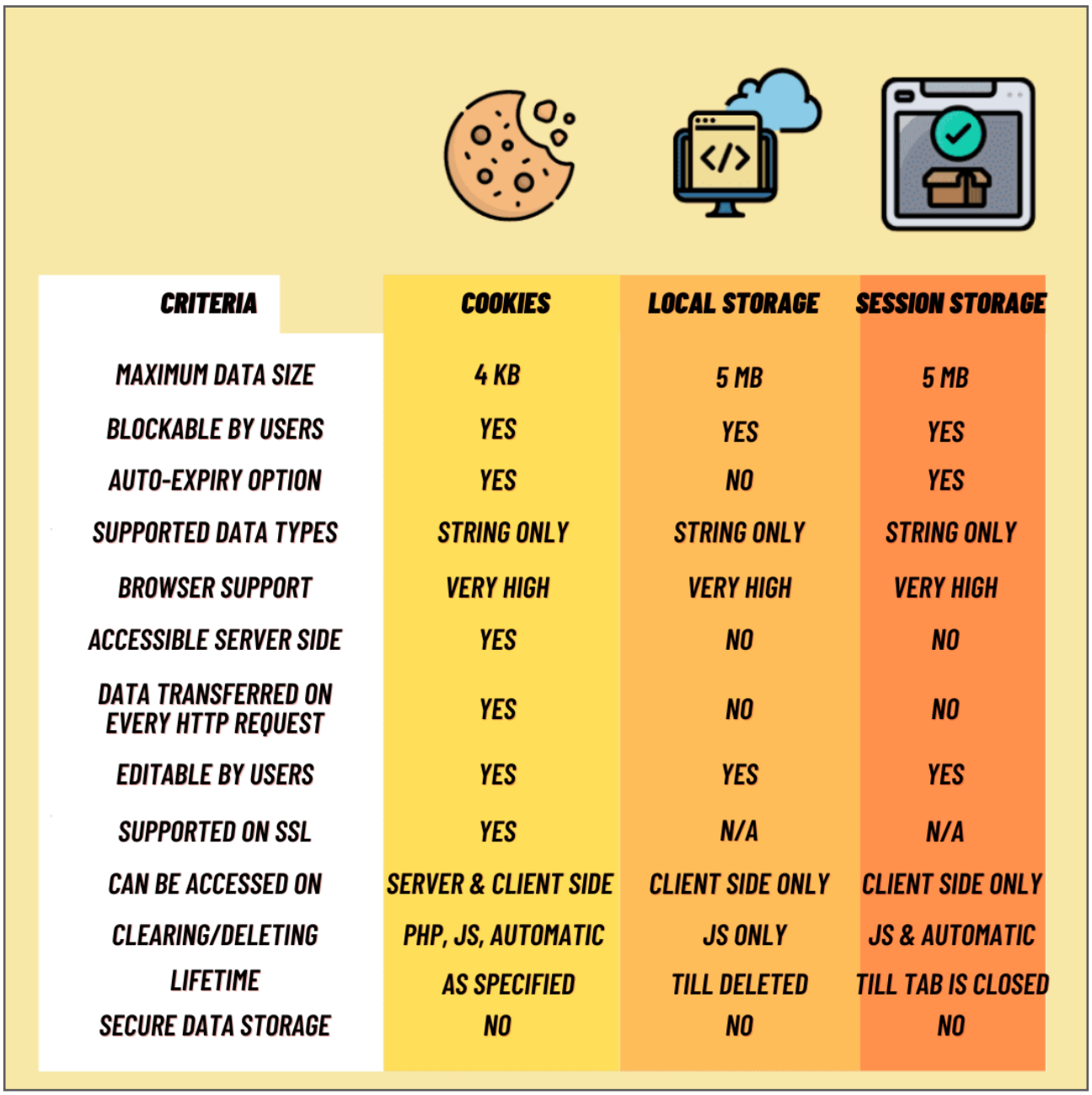
Session Storage

- **Session Storage** is a way of storing data on the client side of an application. It's similar to local storage, but with a few key differences:
  - Data is only available to the site that created it.
  - Data is not shared with other sites.
  - Data is not persistent, meaning it is only available for the duration of the user's session on a site.
  - Data is specific to the browser tab in which it was created.

Differences

Local Storage	Session Storage	Cookies
It allows **10MB** of data to be stored.	It allows **5MB** of data to be stored.	The storage capacity is limited to **4KB** of data.
The stored data is not deleted when the browser is closed.	The data is stored only for the session and will be deleted when the browser is closed.	The data can be set to expire at a certain time.
Introduced in **HTML5**	Introduced in **HTML5**.	Cookies are the oldest (**HTML4**) and most well known mechanism.
Useful for storing data that the user will need to access later, such as offline data.	Great way to improve the performance of your web applications.	Cookies are a good choice for storing data that should not be persisted for a long time, such as session IDs.
The data **IS NOT** sent with the request from the client to server.	The data **IS NOT** sent with the request from the client to server	The data **IS** sent with the request from the client to server.

SameSite cookies explained:
https://web.dev/articles/samesite-cookies-explained

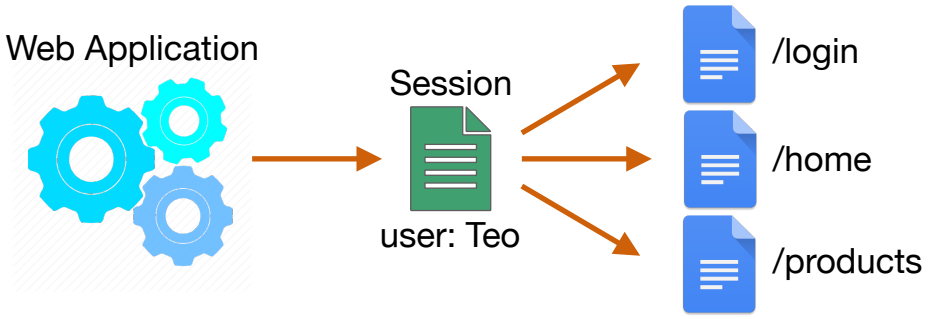


CRITERIA	COOKIES	LOCAL STORAGE	SESSION STORAGE
MAXIMUM DATA SIZE	4 KB	5 MB	5 MB
BLOCKABLE BY USERS	YES	YES	YES
AUTO-EXPIRY OPTION	YES	NO	YES
SUPPORTED DATA TYPES	STRING ONLY	STRING ONLY	STRING ONLY
BROWSER SUPPORT	VERY HIGH	VERY HIGH	VERY HIGH
ACCESSIBLE SERVER SIDE	YES	NO	NO
DATA TRANSFERRED ON EVERY HTTP REQUEST	YES	NO	NO
EDITABLE BY USERS	YES	YES	YES
SUPPORTED ON SSL	YES	N/A	N/A
CAN BE ACCESSED ON	SERVER & CLIENT SIDE	CLIENT SIDE ONLY	CLIENT SIDE ONLY
CLEARING/DELETING	PHP, JS, AUTOMATIC	JS ONLY	JS & AUTOMATIC
LIFETIME	AS SPECIFIED	TILL DELETED	TILL TAB IS CLOSED
SECURE DATA STORAGE	NO	NO	NO

HTTP Session

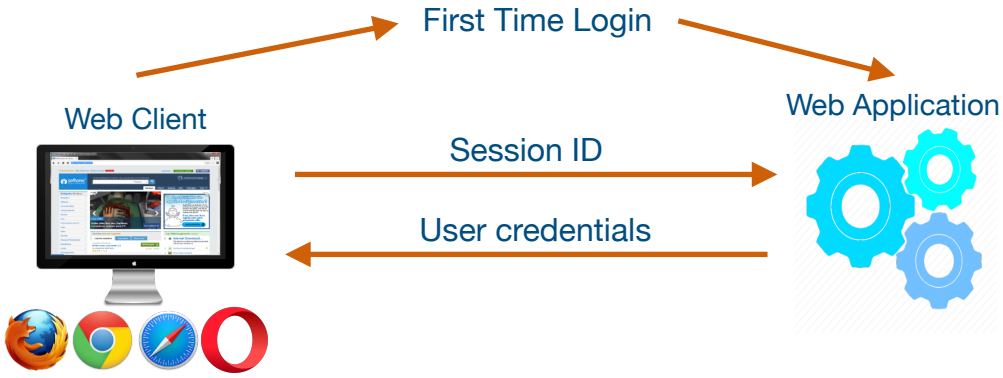
What Are Sessions?

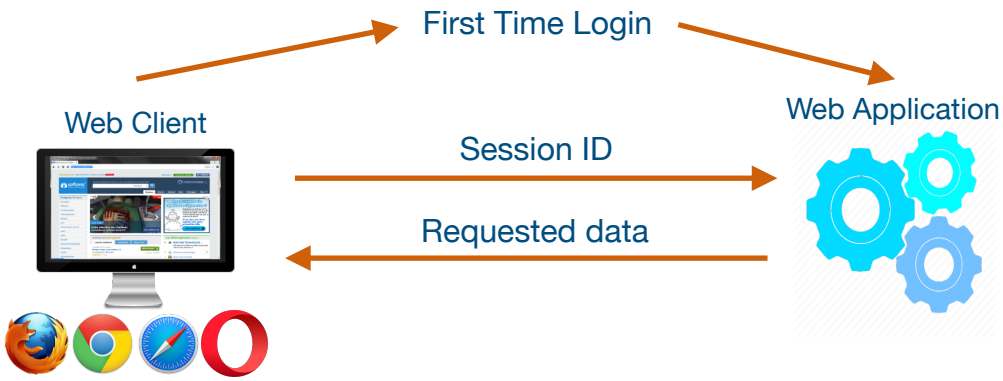
- A way to store information about a user to be used across **multiple pages**

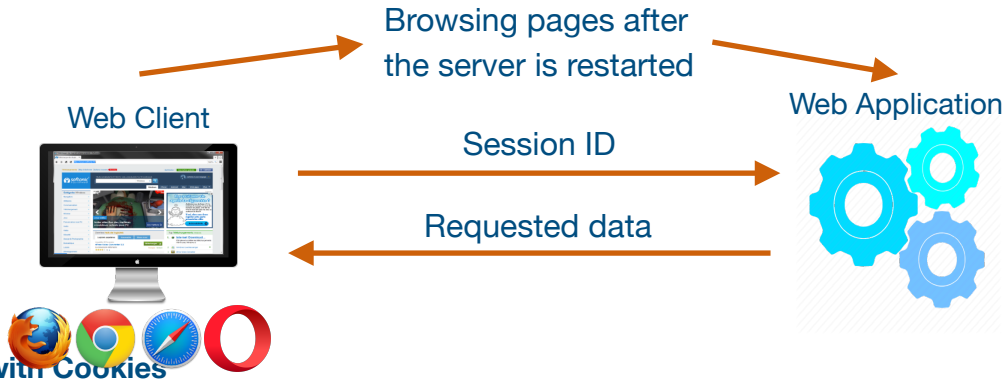


Session Management

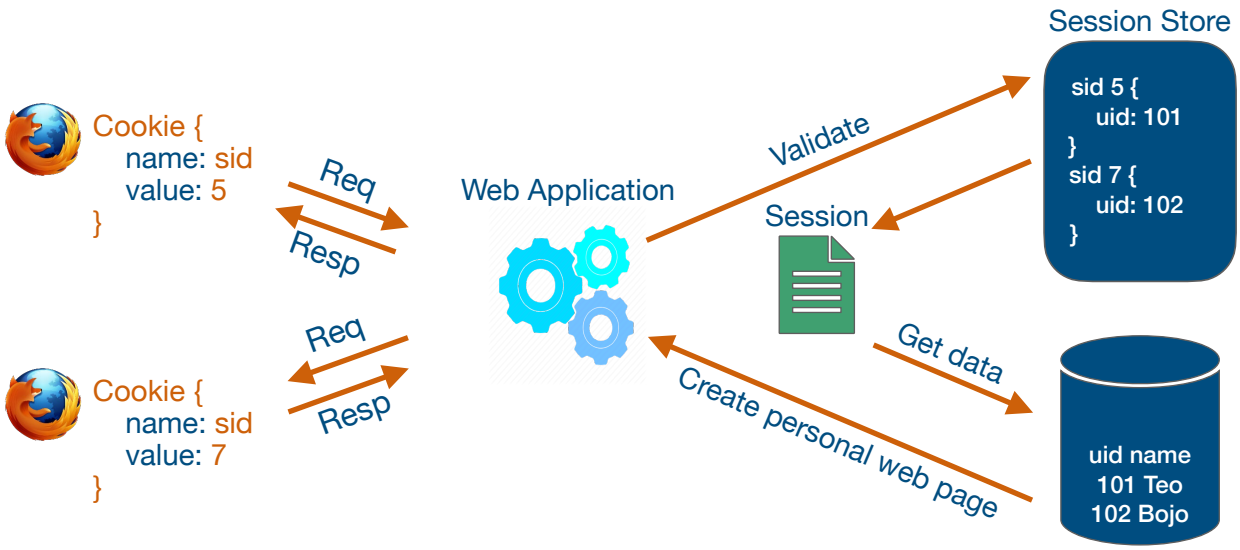
- The exchange mechanism is used between the user and the web application



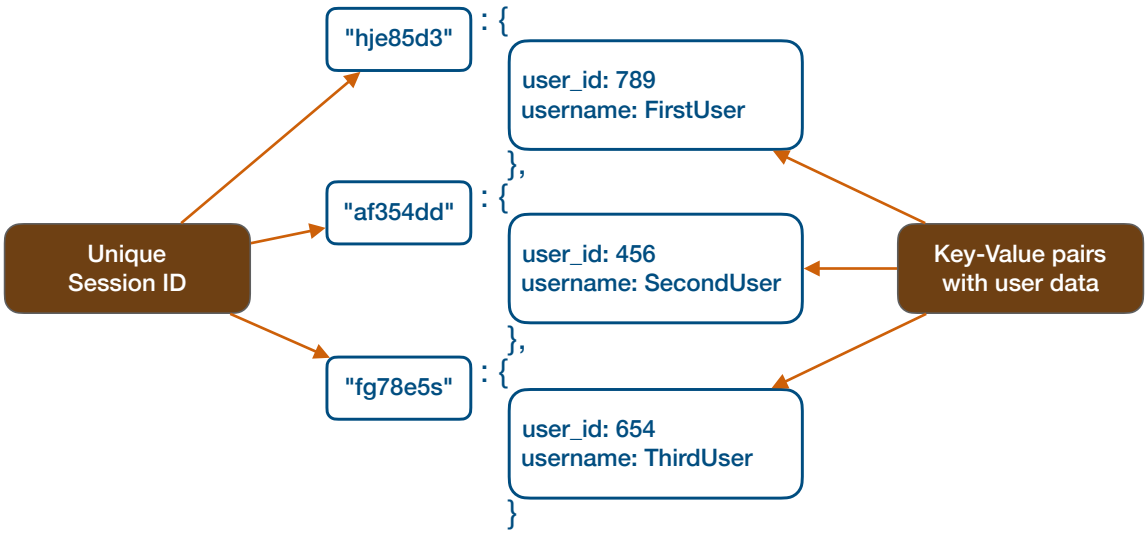




Relation with Cookies



Session Structure



Summary

- **Cookies** are client based stored information
  - They are created by web applications
  - Browsers sends them back to the application
- **Sessions** are server based information
  - They are used across multiple pages
  - Stores important info about the client