Size limitation, security,`localStorage` or`sessionStorage`?

We have already talked about client-side storage limits when we studied the HTML5 cache. We saw that the limit varies depending on various parameters. However, it's good to recall a few things from the specification:

- User agents (browsers) should limit the total amount of space allowed for storage areas.

- User agents may prompt the user when quotas are reached, allowing the user to grant more space to a site. This enables sites to store many user-created documents on the user's computer, for instance.

- User agents should allow users to see how much space each domain is using.

- A mostly arbitrary limit of five megabytes per origin is recommended (translation: give at least 5Mb per origin).

In many cases, local storage is all that your application will need for saving/loading data on demand. More complex ways to do it exist, such as IndexedDB, a No SQL database, that proposes transactions and usually comes with far more available space than local storage. IndexedDB usage is for advanced users and will be covered in the HTML5 part-2 course.

Additionally, there will be a limit on the amount of data that you can store there. Browsers enforce quotas that will prevent you from cluttering your users' drives excessively. These quotas can vary from platform to platform, but are usually reasonably generous for simple cases (around 5MB), so **if you are careful not to store anything huge there, you should be fine.**

Finally, keep in mind that this storage is not necessarily permanent. Browsers are inconsistent in how they allow for it to be wiped, but in several cases it gets deleted with cookies — which is logical when you think of how it can be used for tracking in a similar fashion.

For serious applications, you might want to synchronize existing data with the server on a regular basis, in order to avoid data loss (and in general, because users enjoy using the same service from multiple devices at once). This is a rather complex feat, and frameworks such as

Firebase can help. Such techniques are beyond the scope of this course and will not be covered.

`SESSIONSTORAGE` KEY/VALUES INSTEAD OF COOKIES?

Note that if all you need is to store session-based data in a manner that is more powerful than cookies, you can use the `sessionStorage` object which works in exactly the same way as localStorage, but the lifetime is limited to a single browser session (lifetime of your tab/window).

Also note that in addition to being more convenient and capable of storing more data than cookies, it has the advantage of being scoped to a given browser tab (or similar execution context).

Cookies' big danger: if a user has two tabs open to the same site, they will share the same cookies. Which is to say that if you are storing information about a given operation using cookies in one tab, that information will leak to the other side — this can be confusing if the user is performing different tasks in each.

> **By using `sessionStorage`, the data you store will be scoped and therefore not leak across tabs!**