LocalStorage

Persistencia de los datos

- HTTP es un protocolo sin estado, los datos no persisten entre una llamada y otra.
- Las cookies son el mecanismo utilizado habitualmente para esto, que son accesibles por el cliente y el servidor. La cookie viaje incrementando el tamaño de la petición, además se pueden deshabilitar.
- En muchos casos no es necesario que los datos se comuniquen entre cliente y servidor.
- HTML5 y los nuevos navegadores permiten implementar mecanismos de almacenamiento en el cliente.

API de almacenamiento

- La API de almacenamiento Web proporciona un mecanismo para poder almacenar datos en el navegador que permanecen cuando la página se recarga.
- Storage es un objeto local del navegador. sessionStorage y localStorage son propiedades de sólo lectura basados en los objetos WindowLocalStorage y WindowSessionStorage que implementa Window.

Window.sessionStorage: mantiene un área de almacenamiento separada mientras dure la sesión, aun con recarga de página.

Window.localStorage: mantiene un área de almacenamiento aun cuando el navegador se cierra y se abra nuevamente.

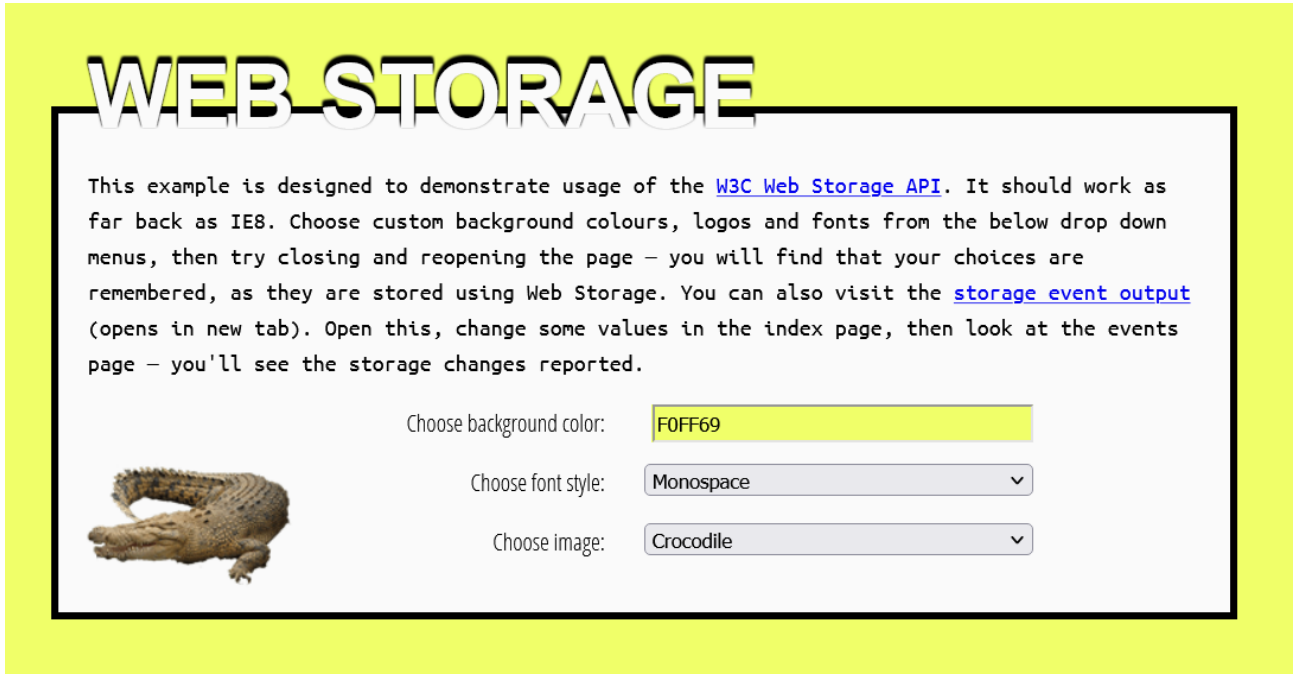
Manipulando el Storage

- Los objetos almacenan clave, valor. Cadenas de caracteres.
- Se acceden a estos valores como un objeto o con los métodos:
  - Storage.getItem()
  - Storage.setItem()
  - Storage.removeItem()
  - Storage.key()
  - Storage.length()
  - Storage.clear()

```
localStorage.colorSetting = '#a4509b';
localStorage['colorSetting'] = '#a4509b';
localStorage.setItem('colorSetting', '#a4509b');
```

- Ejemplo para analizar

https://mdn.github.io/dom-examples/web-storage/



Aunque recargue la página o cierre el navegador o una pestaña la información se mantiene.

JS

```
if(!localStorage.getItem('bgcolor')) {
  populateStorage();
} else {
  setStyles();
}
```

Para saber si la página fue visitada.

```
function setStyles() {
  var currentColor = localStorage.getItem('bgcolor');
  var currentFont = localStorage.getItem('font');
  var currentImage = localStorage.getItem('image');
```

Para tomar un valor de la memoria.

```
function populateStorage() {
  localStorage.setItem('bgcolor', document.getElementById('bgcolor').value);
  localStorage.setItem('font', document.getElementById('font').value);
  localStorage.setItem('image', document.getElementById('image').value);

  setStyles();
}
```

Para setear un valor en la memoria.

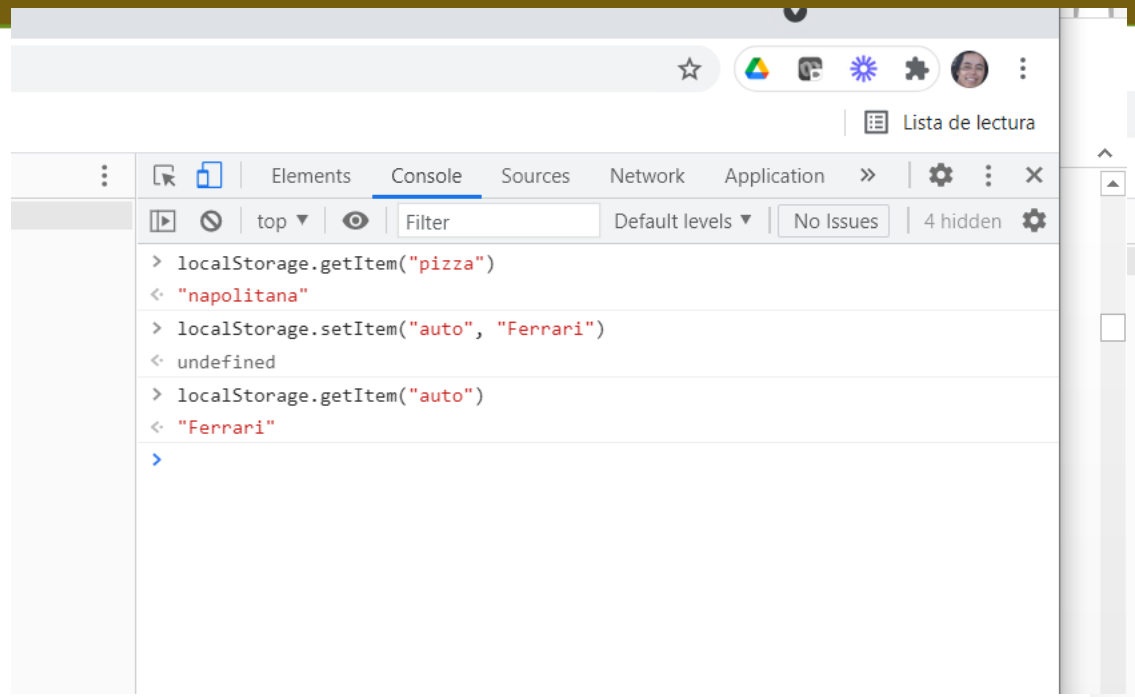
setItem()

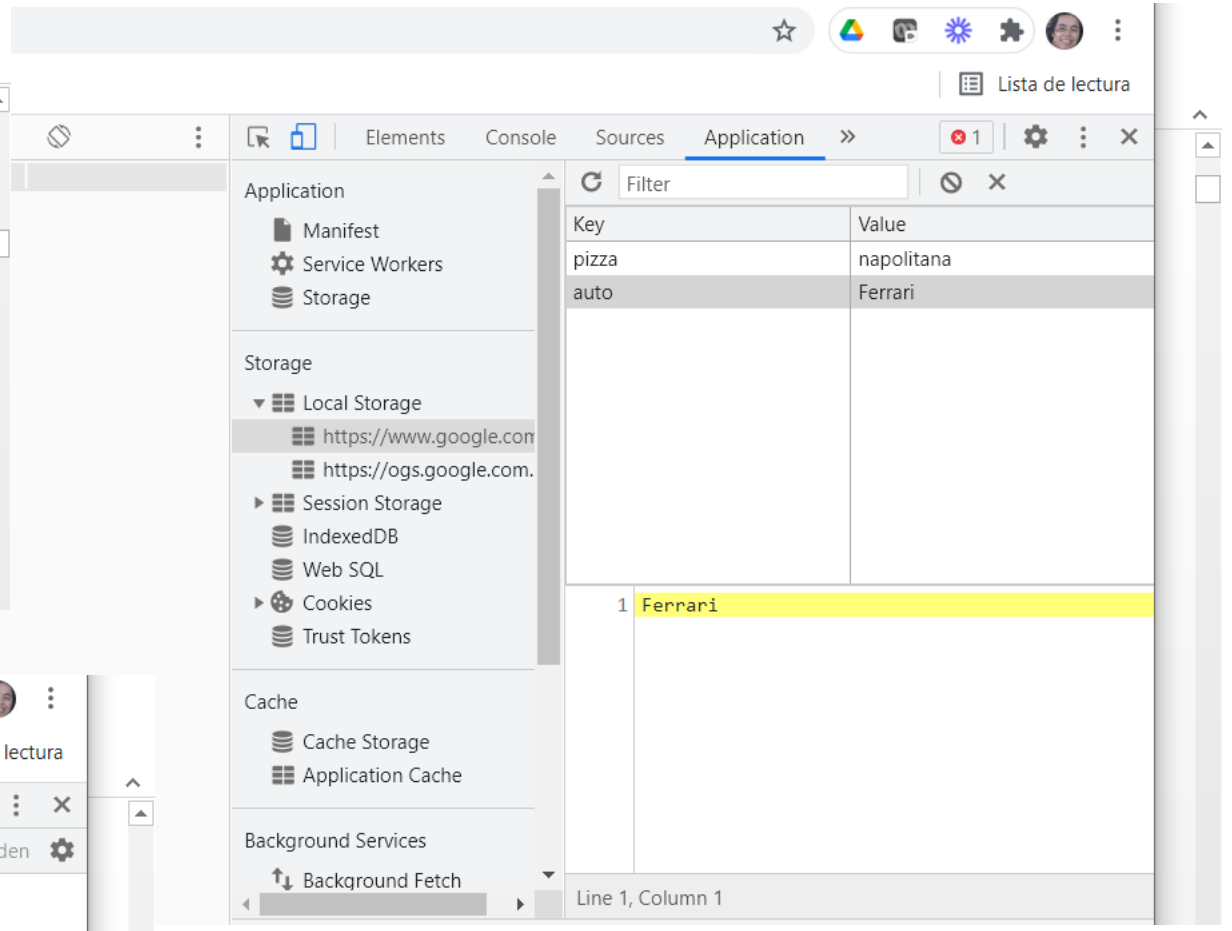
- Permite crear nuevos datos o actualizar los existentes.
- Dos argumentos: la llave del dato que se va a crear/modificar y el valor que se va a guardar.

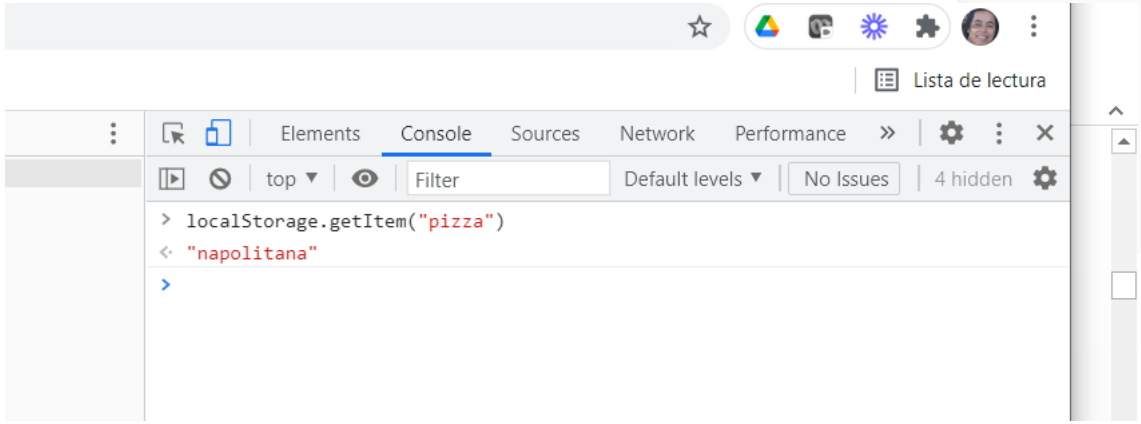
```
function populateStorage() {
  localStorage.setItem('bgcolor', document.getElementById('bgcolor').value);
  localStorage.setItem('font', document.getElementById('font').value);
  localStorage.setItem('image', document.getElementById('image').value);

  setStyles();
}
```

```
bgcolorForm.onchange = populateStorage;
fontForm.onchange = populateStorage;
imageForm.onchange = populateStorage;
```

JS


Lista de lectura
Elements
Console
Sources
Network
Application
top
Filter
Default levels
No Issues
4 hidden
> localStorage.getItem("pizza")
"napolitana"
> localStorage.setItem("auto", "Ferrari")
undefined
> localStorage.getItem("auto")
"Ferrari"


Lista de lectura
Elements
Console
Sources
Application
Application
Manifest
Service Workers
Storage
Storage
Local Storage
https://www.google.com
https://ogs.google.com.
Session Storage
IndexedDB
Web SQL
Cookies
Trust Tokens
Cache
Cache Storage
Application Cache
Background Services
Background Fetch
Filter
Key
Value
pizza
napolitana
auto
Ferrari
1 Ferrari
Line 1, Column 1


Lista de lectura
Elements
Console
Sources
Network
Performance
top
Filter
Default levels
No Issues
4 hidden
> localStorage.getItem("pizza")
"napolitana"

Consideraciones

- Cualquiera lo puede ver. No es recomendable para almacenamiento de datos sensibles.
- Se trabaja con strings. Recordar JSON.strignify y JSON.parser
- Antes de tomar un valor con getItem, verificar que tenga valores válidos.

- https://html.spec.whatwg.org/multipage/webstorage.html#introduction-15