Javascript

Comment enregistrer la data dans les navigateurs

- Les API `sessionStorage` et `localStorage` permettent de stocker des données dans le navigateur côté client. Ces données sont accessibles uniquement à partir du même domaine et ne sont jamais envoyées au serveur.

Utilisation de base

Principales méthodes disponibles :

1. `setItem(key, value)` : Stocker une valeur.
2. `getItem(key)` : Récupérer une valeur.
3. `removeItem(key)` : Supprimer une clé spécifique.
4. `clear()` : Supprimer toutes les données stockées.
5. `length` : Retourne le nombre d'éléments stockés.
6. `key(index)` : Récupère la clé à un index donné.

Exemple d’utilisation

```
// Stockage d'une donnée
sessionStorage.setItem('nom', 'Alice');
localStorage.setItem('nom', 'Bob');

// Récupération d'une donnée
console.log(sessionStorage.getItem('nom')); // "Alice"
console.log(localStorage.getItem('nom'));  // "Bob"
```

Exemple d’utilisation

```
// Suppression d'une clé
sessionStorage.removeItem('nom');
localStorage.removeItem('nom');

// Suppression de toutes les données
sessionStorage.clear();
localStorage.clear();
```

Différence entre sessionStorage et localStorage

Propriété	**sessionStorage**	**localStorage**
Durée de vie	Les données persistent uniquement pendant la session du navigateur (jusqu'à la fermeture de l'onglet ou de la fenêtre).	Les données persistent indéfiniment, même après la fermeture du navigateur ou le redémarrage du système.
Portée	Par onglet ou fenêtre.	Accessible à tous les onglets et fenêtres du même domaine.

Exemple de stockage d'une liste

```
let liste = ["One", "two", "three"];
localStorage.setItem("liste", liste.toString());
console.log(localStorage.getItem("liste"));

// ajouter un élément
let liste_lue = localStorage.getItem("liste").split(",");
console.log(liste_lue);

liste_lue.push("four");
localStorage.setItem("liste", liste_lue.toString());
console.log(localStorage.getItem("liste"));
```

JS Objects (Tableaux associatifs Ou Dictionnaires)

```
// Stocker un objet
const user = { id: 1, nom: 'Alice', email: 'alice@example.com' };
localStorage.setItem('user', JSON.stringify(user));

// Récupérer l'objet
const storedUser = JSON.parse(localStorage.getItem('user'));
console.log(storedUser.nom); // "Alice"
```

Bonnes pratiques

- Limitez la taille des données stockées (5 à 10 Mo en moyenne).
- Évitez de stocker des informations sensibles (comme des mots de passe).
- Utilisez un mécanisme pour vérifier si les données sont toujours à jour.

Exercices

Exercice 1 :

1. Créez une page web avec deux champs (nom et âge) et un bouton "Enregistrer".
2. Au clic sur le bouton, sauvegardez les valeurs dans le `localStorage`.
3. Ajoutez un autre bouton "Afficher" qui affiche les valeurs stockées dans une alerte.

Exercice 2 : Gestion de session avec `sessionStorage`

1. Créez une page qui affiche le nombre de fois que l'utilisateur l'a rechargée dans la session en cours.
2. Stockez et mettez à jour ce compteur avec `sessionStorage`.