

TP : "Mini To-Do" persistant avec localStorage

Objectif pédagogique : savoir enregistrer, lire et supprimer des données dans **LocalStorage** pour garder l'état d'une petite application entre deux visites.

1. Contexte & résultat attendu

Vous allez créer une page HTML très simple qui permet d'ajouter des tâches dans une liste ;

si l'on rafraîchit ou ferme/ré-ouvre le navigateur, la liste réapparaît telle qu'on l'a laissée.

Étape 1 : Structure HTML minimale

```
<h1>Ma liste de tâches</h1>

<input id="taskInput" placeholder="Nouvelle tâche">
<button id="addBtn">Ajouter</button>

<ul id="taskList"></ul>

<button id="clearBtn">Tout effacer</button>
```

Étape 2 : Ajouter une tâche (sans persistance)

1. En JavaScript, écoutez le clic sur **Ajouter**.
2. Si le champ n'est pas vide :
 - ajoutez la valeur dans un **tableau** `tasks` conservé en mémoire,
 - appelez une fonction `render()` qui recrée la liste entière dans le DOM,
 - videz le champ.

Étape 3 : Sauvegarder la liste


1. Après chaque ajout ou suppression, **sérialisez** le tableau `tasks` :
`JSON.stringify(tasks)` .
2. Stockez-le : `localStorage.setItem('tasks', jsonString)` .

Étape 4 : Restaurer au chargement

Au tout début du script :

1. Lisez `localStorage.getItem('tasks')` .
2. S'il existe, reconvertissez : `tasks = JSON.parse(...)` ; sinon `tasks = []` .
3. Appelez `render()` pour afficher la liste à partir du tableau.

Étape 5 : Supprimer une tâche

1. Ajoutez une petite croix  ou bouton **Supprimer** dans chaque `` .
2. Au clic, retirez l'élément correspondant du tableau `tasks` , puis :
 - `localStorage.setItem('tasks', JSON.stringify(tasks))` ,
 - `render()` pour mettre à jour la vue.
3. Vérifiez que la suppression reste effective après un rafraîchissement.

Étape 6 : Vider toute la liste

1. Clic sur **Tout effacer** →
 - `tasks = []` ,
 - `localStorage.removeItem('tasks')` ,
 - `render()` pour vider l'affichage.

3. Bonus (optionnel)

| Bonus | Idée |
|---------------------|--|
| Compteur | Afficher "3 tâches restantes". |
| Édition | Double-clic sur une tâche → passer en champ texte pour modifier, puis sauvegarder. |
| Clavier only | Appuyer sur Entrée dans le champ pour ajouter. |

| | |
|---------------|---------------------------------------|
| Design | Un peu de CSS (flex, pastel, icônes). |
|---------------|---------------------------------------|

4. Critères de réussite

| Critère | Points clés |
|-----------------------------|--|
| Fonctionnel | Ajouter / supprimer / vider OK. |
| Persistance | La liste réapparaît après F5 ou redémarrage. |
| Code clair | Nom de variables explicite, fonctions courtes. |
| Sans données perdues | Pas de crash si <code>localStorage</code> est vide ou corrompu (penser <code>try ... catch</code>). |

5. Rappels rapides

- **API :** `setItem` , `getItem` , `removeItem` .
- Tout est **string** → toujours `JSON.stringify` / `JSON.parse` .
- Vérifiez la présence de la clé avant de parser.