
	Nom du Module	
	BILAN TP DE PROGRAMMATION	

Nom : PION

Prénom : Maxence

Nom du TD/TP : TP TO DO LIST

I. Instructions

Format

- Le bilan peut être **rédigé** sous forme de document écrit ou présenté sous forme de diaporama si cela est plus adapté.
- Le document final doit être soumis au format **PDF**. Assurez-vous que les fichiers sont bien nommés (évitez les espaces et utilisez des noms explicites, ex. : **bilanTP_NomPrenom.pdf**).

Clarté et Précision

- Assurez-vous que toutes les sections sont complétées avec soin. Prenez le temps de rédiger des explications claires et argumentées.
- Présentez les informations dans un ordre logique, avec des titres et sous-titres bien marqués pour faciliter la lecture.

Respect des délais

- Remettez votre bilan dans les délais impartis pour permettre une évaluation rapide et efficace. Les travaux remis en retard peuvent ne pas être pris en compte.

II. Fonctionnalités Implémentées

Ajout de tâches : Permet à l'utilisateur d'ajouter une nouvelle tâche via un champ texte.

Affichage des tâches : Affiche toutes les tâches dans un tableau interactif.

Marquage des tâches terminées : Permet de cocher une case pour indiquer qu'une tâche est terminée.

Suppression de tâches : Supprime une tâche de la liste.

Filtres : Offre des options pour afficher uniquement les tâches terminées, non terminées ou toutes les tâches.

Interface principale de l'application :

Liste de Tâches

Terminée	Numéro	Libellés	Actions
<input type="checkbox"/>	1	tache 1	<input type="button" value="Supprimer"/>
<input type="checkbox"/>	2	tache 2	<input type="button" value="Supprimer"/>

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Liste de Tâches</title>
  <link rel="stylesheet" href="style.css">
  <script defer src="script.js"></script>
</head>
<body>
  <div class="container">
    <h1>Liste de Tâches</h1>
    <div class="formulaire">
      <input type="text" placeholder="Ajouter une nouvelle tâche...">
      <button onclick="ajouterTache()">Ajouter</button>
    </div>
    <div class="filtres">
      <button onclick="afficherToutes()">Afficher toutes</button>
      <button onclick="afficherTerminees()">Afficher terminées</button>
      <button onclick="afficherNonTerminees()">Afficher non terminées</button>
    </div>
    <table>
      <thead>
        <tr>
          <th class="col-numero">Terminée</th>
          <th class="col-numero">Numéro</th>
          <th class="col-libelle">Libellés</th>
          <th class="col-actions">Actions</th>
        </tr>
      </thead>
      <tbody>
        <tr>
          <td><input type="checkbox"></td>
          <td>1</td>
          <td>tache 1</td>
          <td><input type="button" value="Supprimer"></td>
        </tr>
        <tr>
          <td><input type="checkbox"></td>
          <td>2</td>
          <td>tache 2</td>
          <td><input type="button" value="Supprimer"></td>
        </tr>
      </tbody>
    </table>
  </div>
</body>
</html>
```

L'interface principale présente un champ de texte pour ajouter une tâche, trois boutons pour filtrer les tâches (toutes, terminées, non terminées) et un tableau pour afficher les tâches ajoutées.

Ajout d'une tâche :

Terminée	Numéro	Libellés	Actions
<input type="checkbox"/>	1	tache 1	<input type="button" value="Supprimer"/>
<input type="checkbox"/>	2	tache 2	<input type="button" value="Supprimer"/>

```
function ajouterTache() {
  const champTexte = document.querySelector('input');
  const description = champTexte.value.trim();

  if (description === "") {
    alert("Veuillez entrer une description de tâche.");
    return;
  }

  taches.push(description);
  termine.push(false);
  champTexte.value = '';
  ajouterTacheHTML(taches.length - 1);
}

function ajouterTacheHTML(index) {
  const tbody = document.querySelector('table tbody');
  const nouvelleLigne = `
    <tr>
      <td>
        <input type="checkbox" id="${index}" ${termine[index] ? "checked" : ""} onchange="cocher(event)">
      </td>
      <td>${index + 1}</td>
      <td>${termine[index] ? 'class="barre"' : ''}>${taches[index]}</td>
      <td><button onclick="supprimerTache(${index})">Supprimer</button></td>
    </tr>
  `;
  tbody.insertAdjacentHTML('beforeend', nouvelleLigne);
}
```

Lorsque l'utilisateur clique sur le bouton "Ajouter", la fonction récupère la valeur du champ de texte, l'ajoute à la liste des tâches. Ensuite, elle appelle **ajouterTacheHTML** pour mettre à jour le tableau et afficher la tâche.

Marquage d'une tâche comme terminée :

Terminée	Numéro	Libellés	Actions
<input type="checkbox"/>	1	tache 1	<button>Supprimer</button>

```
function cocher(event) {
  const caseId = parseInt(event.target.id);
  termine[caseId] = event.target.checked;
  majTableau();
}
```

Quand une case est cochée, cette fonction met à jour l'état de la tâche correspondante (terminée ou non) dans le tableau `termine`. Ensuite, elle appelle `majTableau` pour mettre à jour la tâche dans le tableau.

Filtrage des tâches :

Liste de Tâches

Ajouter une nouvelle tâche... Ajouter

Afficher toutes **Afficher terminées** Afficher non terminées

Terminée	Numéro	Libellés	Actions
<input checked="" type="checkbox"/>	1	tache-1	<button>Supprimer</button>
<input type="checkbox"/>	2	tache 2	<button>Supprimer</button>
<input checked="" type="checkbox"/>	3	tache-3	<button>Supprimer</button>
<input type="checkbox"/>	4	tache 4	<button>Supprimer</button>

Liste de Tâches

Ajouter une nouvelle tâche... Ajouter

Afficher terminées Afficher toutes Afficher non terminées

Terminée	Numéro	Libellés	Actions
<input checked="" type="checkbox"/>	1	tache-1	<button>Supprimer</button>
<input checked="" type="checkbox"/>	3	tache-3	<button>Supprimer</button>

```
function filtrerTaches(terminees) {
  const tbody = document.querySelector('table tbody');
  tbody.innerHTML = "";
  taches.forEach((tache, index) => {
    if (terminees === null || termine[index] === terminees) {
      ajouterTacheHTML(index);
    }
  });
}
```

Liste de Tâches

Ajouter une nouvelle tâche... Ajouter

Afficher toutes **Afficher non terminées** Afficher terminées

Terminée	Numéro	Libellés	Actions
<input type="checkbox"/>	2	tache 2	<button>Supprimer</button>
<input type="checkbox"/>	4	tache 4	<button>Supprimer</button>

Lorsqu'un bouton de filtrage est cliqué, cette fonction vide le tableau HTML et affiche uniquement les tâches correspondant au filtre sélectionné : toutes, terminées, ou non terminées.

Supprimer une tâche :

Terminée	Numéro	Libellés	Actions
<input type="checkbox"/>	2	tache 2	<button>Supprimer</button>
<input type="checkbox"/>	4	tache 4	<button>Supprimer</button>

```
function supprimerTache(index) {
  taches.splice(index, 1);
  termine.splice(index, 1);
  majTableau();
}
```

Quand l'utilisateur clique sur "Supprimer", cette fonction retire la tâche sélectionnée et son état associé des tableaux `taches` et `termine`. Elle appelle ensuite `majTableau` pour actualiser l'affichage.

III. Difficultés Rencontrées et Solutions Apportées

Difficultés :

- **Gestion des indices après suppression** : Les indices des tâches ne se mettaient pas à jour correctement après une suppression.
- **Problème d'affichage des filtres** : Les filtres provoquaient des doublons dans l'affichage des tâches.

Solutions :

- **Recalcul des indices** : Utilisation de la fonction `majTableau()` pour réorganiser les indices après suppression.
- **Réinitialisation du tableau** : Modification de la fonction `filtrerTaches()` pour effacer le tableau avant de le remplir à nouveau.

Critères d'évaluation :

- Difficultés bien identifiées et pertinentes.
- Solutions claires, adaptées et bien expliquées.

IV. Commentaires et Suggestions

Remarques générales :



- La structure du code est claire et suit une logique simple, ce qui facilite les modifications.
- L'utilisation de tableaux pour gérer les données est pratique pour ce projet de base.

Suggestions d'amélioration :

- Ajouter une fonctionnalité de recherche pour retrouver rapidement une tâche.
- Mettre en place un stockage local pour sauvegarder les tâches après rechargement de la page.

Focus sur les Apprentissages

- **Techniques** : Gestion du DOM, utilisation des tableaux pour stocker les données, manipulation des événements JavaScript.
- **Méthodologie** : Importance de bien planifier les fonctionnalités avant de coder et de tester chaque étape.

	Nom du Module	
	BILAN TP DE PROGRAMMATION	

Critères d'évaluation :

- Observations pertinentes et réflexion approfondie.
- Suggestions concrètes et réalistes.

V. Auto-évaluation

Évaluation de votre travail :

Tâches	Notes /10
Planification	8/10
Codage	9/10
Tests	7/10

Objectifs pour les futurs projets :

- Améliorer la gestion des erreurs dans le code.
- Ajouter des tests automatiques pour vérifier les fonctionnalités.

Critères d'évaluation :

- Auto-évaluation objective et bien argumentée.
- Objectifs pertinents et réalistes.

VI. Code

Lien GitHub :

- https://github.com/Frizx/TpToDoList_PionMaxence

Critères d'évaluation :

- Lien GitHub fonctionnel.
- README bien élaboré.
- Code propre, bien organisé et documenté