Travail pratique : Gestionnaire de liste de tâches (pondération 25%)

Ce TP consiste à réaliser une application JavaScript qui gère localement une liste de tâches mémorisée dans un tableau. Ce tableau est initialement vide, puis complété à l'aide d'un formulaire intégré dans la page, et mis à jour également avec des interactions de l'utilisateur.

Après quelques saisies de tâches, ce tableau pourrait contenir par exemple :

```
tTaches = [
  {"nom":"aaa","importance":"3","date":"21/04/2021, 15:41:05","id":1,"fait":false},
  {"nom":"ccc","importance":"1","date":"21/04/2021, 15:41:12","id":2,"fait":false},
  {"nom":"bbb","importance":"2","date":"21/04/2021, 15:41:29","id":3,"fait":false},
  {"nom":"éée","importance":"2","date":"21/04/2021, 15:41:36","id":4,"fait":false},
  {"nom":"ggg","importance":"2","date":"21/04/2021, 15:41:45","id":5,"fait":false},
  {"nom":"ddd","importance":"1","date":"21/04/2021, 15:42:16","id":6,"fait":false}
]
```

Chaque tâche est caractérisée par ses propriétés :

nom	une chaîne de caractères,
importance	le nombre 1 (haute), 2 (moyenne) ou 3 (basse),
date	la date et heure de création (générée automatiquement),
id	un numéro unique généré en incrémentant le plus grand numéro existant dans le tableau (la numérotation commence à 1),
fait	un booléen qui indique si la tâche est réalisée ou pas.

1. Ajout d'une tâche

Permettre la saisie d'un nom de tâche (au moins un caractère) et d'un niveau d'importance comme par exemple :

Traiter la mise à jour du tableau des tâches puis le réaffichage complet de la liste des tâches. Utiliser pour cela le template d'id "modeleListe" pour l'affichage de chaque ligne de tâche.

L'affichage de l'exemple précédent devient alors :

2. Tri de la liste des tâches

Au chargement initial de la page, la liste est triée sur les critères nom puis importance.
Vous pouvez choisir de trier aussi sur les critères importance puis nom avec la select de tri.

3. Suppression d'une tâche

Au clic sur l'action "effacer" d'une ligne de tâche, cette tâche est supprimée du tableau à partir de son "id". Le tableau est réaffiché complètement après cette suppression.

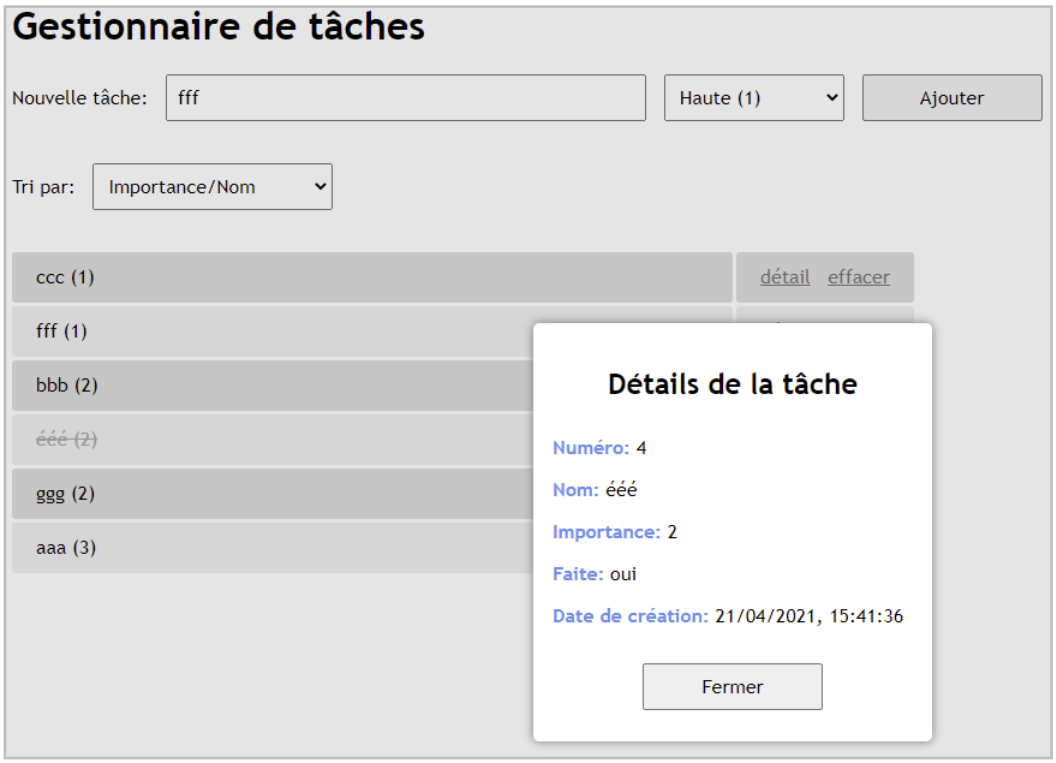
Par exemple après avoir cliqué sur "effacer" de la ligne contenant "ddd (1)" :

4. Modification du statut "fait" d'une tâche

Au clic sur le libellé d'une tâche son statut "fait" est inversé. S'il était "false" (valeur initiale) il devient "true" pour indiquer que la tâche est réalisée et réciproquement.
L'affichage est modifié, par exemple une tâche faite est grisée et barrée, ainsi en cliquant sur "ééé (2)" afficher :

5. Affichage du détail d'une tâche dans une fenêtre modale

Au clic sur l'action "détail" d'une ligne de tâche, afficher toutes les données d'une tâche dans une fenêtre modale, en utilisant le template d'id "modeleDetail", comme par exemple :

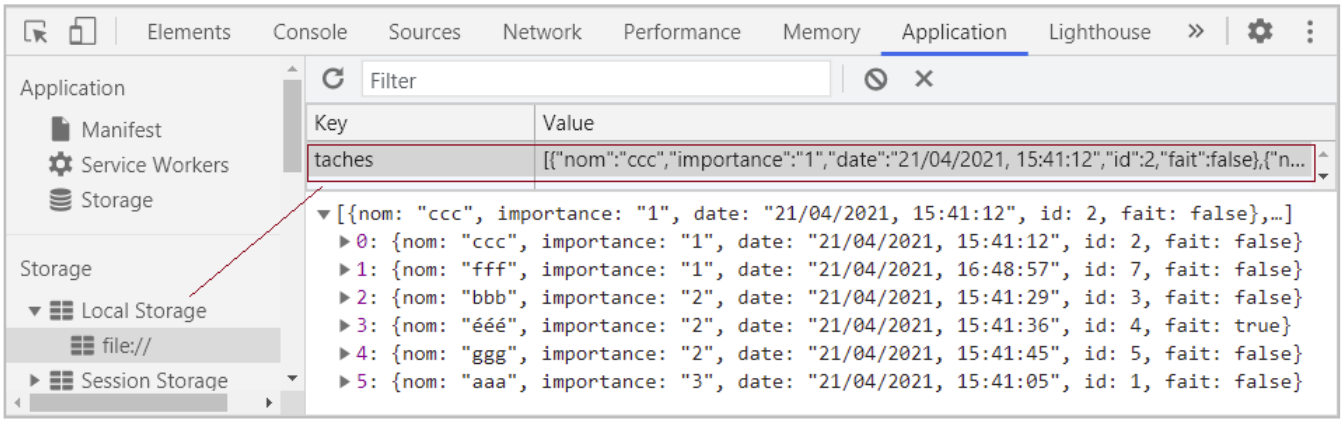


La fenêtre modale peut être gérée avec une balise dialog.

6. Sauvegarde de la liste des tâches en localStorage

Pour pouvoir réutiliser cette application, il est nécessaire de sauvegarder le tableau des tâches en localStorage après chaque mise à jour (tableau complet au format JSON dans un seul item), de manière à le restaurer à chaque chargement initial de la page.

Exemple de stockage en localStorage :



Barème :

- Ajout d'une tâche dans le tableau à partir du formulaire	3 points
- Affichage complet de la liste de tâches après chaque modification du tableau en utilisant le template d'id "modeleListe" pour afficher chaque tâche de la liste	3 points
- Exploitation de la select de tri en combinant deux critères et en réaffichant le tableau trié	2 points
- Suppression d'une tâche dans le tableau au clic sur l'action "effacer" et réactualisation de l'affichage	2 points
- Modification du statut "fait" au clic sur le libellé de la tâche et réactualisation de l'affichage	2 points
- Affichage de toutes les données d'une tâche au clic sur l'action "détail" dans une fenêtre modale, en utilisant le template d'id "modeleDetail"	3 points
- Mémorisation du tableau des tâches en localStorage après chaque mise à jour (tableau complet au format JSON dans un seul item)	1 point
- Rechargement du tableau en localStorage à chaque chargement initial de la page	1 point
- Qualité du code (organisation, indentation, nommage des variables et des fonctions, commentaires)	3 points

Consignes :

Vous pouvez modifier les codes CSS et HTML fournis à titre d'exemple.

Vous pouvez rassembler la gestion des listeners dans un module principal (main.js) et les manipulations du tableau des tâches dans un module dédié et/ou dans une classe.

Commentez votre code lorsque c'est nécessaire; chaque fonction doit avoir un en-tête JSDoc.

Remise :

Remettez votre dossier compressé avec tous les composants (html, css et js) sur Léa.