TP N°1 – Javascript- To do list

Table des matières

II. Ajout de tâches

1. Résultat Attendu

Liste des tâches

Numéro	Libellé
1	Faire le TD de base de données
2	Réviser cours Réseaux
3	Terminer la rédaction du compte-rendu

2. HTML

1 : Créez un bouton "Ajouter une tâche" dans le HTML

Étape N°1 : - Créez un bouton dans votre interface utilisateur avec l'inscription "Ajouter une tâche".

Voici donc ma page html :

On y vois donc la structure standrard d'une page html, incluant également déjà les liens vers mes deux fichier css et javacript.
On y trouve également donc le bouton « Ajouter une tâche ».

2 : Créez un champ de texte pour la description de la tâche dans le HTML

Étape N°2 : - Créez un champ de texte où l'utilisateur pourra saisir la description de la tâche à ajouter.

L'image de l'étape précédente montre déjà la création de la balise <input>. Je vous laisse donc vous y référer.

ET c'est tout : vous n'avez plus rien à ajouter au fichier HTML

3. Javascript

3 : Ajoutez une fonction JavaScript pour gérer l'ajout de tâches

Étape N°3 : -En Javascript, définissez une variable de type tableau qui va contenir la liste des tâches

Voici donc mon tableau, vide pour le moment évidement :

```
JS main.js > ...
1   let todo = [];
```

Étape N°4 : Créez une fonction JavaScript qui sera appelée lorsque l'utilisateur clique sur le bouton "Ajouter une tâche". Cette fonction devra prendre la description de la tâche depuis le champ de texte et l'ajouter à la liste des tâches existantes. (méthode *push*)

4 : Utilisez un événement pour détecter le clic sur le bouton

Étape N°5 : - Utilisez un gestionnaire d'événements pour détecter quand l'utilisateur clique sur le bouton "Ajouter une tâche". Lorsque l'utilisateur appuie sur le bouton, une fonction *AjouterTache* est appelée.

5 : Questions préliminaires :

En JavaScript, pouvez-vous indiquer la commande pour accomplir les tâches suivantes :

Étape N°6 : Créer un tableau <table> en javascript puis rechercher la première occurrence de la balise `<table>` ?

Étape N°7 : Créer une balise `<td>` ?

Étape N°8 : Définir l'attribut `width` à 10 pixels pour une balise ?

Étape N°9 : Ajouter la balise `<td>` créée précédemment à la fin de la balise `<table>` ?

5bis : Mettez à jour l'interface utilisateur avec la nouvelle tâche

- Une fois que l'utilisateur a ajouté une tâche il faut mettre à jour l'interface utilisateur pour afficher cette nouvelle tâche dans la liste des tâches (tableau HTML).

Suivez ces étapes :

Étape N°10 : Modifiez la fonction *AjouterTache*.

Ce que fait la fonction :

- Récupère la valeur de l'input de *type texte*.
- Si cette valeur n'est pas vide :
- Elle est ajoutée au tableau *TableauTaches* (à definir en variable globale) - Affiche le tableau *TableauTaches* dans la console.
- Appelle la fonction *AjouterTacheHTML* avec la dernière valeur ajoutée au tableau.

La fonction *AjouterTacheHTML* crée une table HTML avec un en-tête et ajoute des éléments à cette table chaque fois que la fonction est appelée. L'élément est donné en paramètre.

Étape N°11 : Codez cette fonction en suivant ce pseudo-code :

- Déclaration de la fonction AjouterTacheHTML avec un paramètre item. - Recherche de la première balise <table> dans le document HTML - Affiche dans la console le tableau trouvé (si présent) ou null (si absent). - Si aucune table n'a été trouvée - crée un nouvel élément table.
- Ajoute la nouvelle table créée à la fin du corps du document HTML.Crée un nouvel élément <caption> pour le tableau.Assignation de "Liste des tâches" comme texte à l'élément caption.
- Ajoute l'élément caption à la table.
- Crée un élément d'en-tête de table <thead>.
- Crée une nouvelle ligne pour l'en-tête.
- Crée la première cellule d'en-tête.
- Définit le texte de la première cellule d'en-tête comme "Numéro".
- Définit la largeur de la première cellule d'en-tête à 10 pixels.
- Crée la deuxième cellule d'en-tête.
- Définit le texte de la deuxième cellule d'en-tête comme "Libellé".
- Ajoute la première cellule d'en-tête à la ligne d'en-tête.
- Ajoute la deuxième cellule d'en-tête à la ligne d'en-tête.
- Ajoute la ligne d'en-tête à l'élément **thead**.
- Ajoute l'élément **thead** à la table.
- Crée un élément de corps de table **<tbody>**.
- Ajoute l'élément **tbody** à la table.
- Recherche de l'élément **<tbody>** dans le tableau et assignation à une variable - Crée une nouvelle ligne pour le corps de la table.
- Crée la première cellule de la ligne.
- Crée la deuxième cellule de la ligne.

- Définit le contenu de la première cellule comme la longueur du tableau **TableauTaches** (le numéro de la tâche).
- Style la première cellule pour que son contenu soit centré horizontalement et verticalement.
- Définit le contenu de la deuxième cellule comme la valeur du paramètre **item**.
- Ajoute la première cellule à la nouvelle ligne.
- Ajoute la deuxième cellule à la nouvelle ligne. - Ajoute la nouvelle ligne au corps du tableau.

6 : Réinitialisez le champ de texte après l'ajout

Étape N°12 : - Après avoir ajouté une tâche avec succès, assurez-vous de réinitialiser le champ de texte afin que l'utilisateur puisse ajouter d'autres tâches sans avoir à effacer manuellement le champ.

Étape N°13 : Si le champ du texte est vide, ne pas ajouter d'élément dans le tableau

4. CSS

7 : Appliquez du CSS pour améliorer l'apparence du bouton "Ajouter une tâche"

Étape N°14 : Ajoutez du CSS pour styliser le bouton "Ajouter une tâche" :

- backCouleur de fond bleue
- Texte blanc
- Espace intérieur pour le bouton
- Pas de bordure
- Coins arrondis
- Taille du texte 1 em
- Le curseur se transforme en main lorsque vous passez dessus
- Animation de transition pour l'effet de survol
- Pas de contour lorsque le bouton est sélectionné
- Couleur de fond un peu plus foncée lors de la survol
- Couleur de fond encore plus foncée lors du clic

8 : Stylisez le champ de texte de la description de la tâche

Étape N°15 : - Utilisez du CSS pour rendre le champ de texte plus attrayant. Pour cela :

- Espacement intérieur autour du texte de 5 pixels*/
- Bordure de 2 pixel de couleur ccc
- Taille de police de 16 pixels

9 : Stylisez la liste des tâches

Étape N°16 : - Appliquez du CSS pour améliorer l'apparence de la liste des tâches. Complétez et copiez le css concernant la table :

```
/* Styles généraux pour les tables */ table {   width: _____; /* La table occupe la
totalité de la largeur de son conteneur parent */
  _____: separate; /* Chaque cellule possède sa propre bordure individuelle */
  border-spacing: _____; /* Suppression de tou espacement entre les cellules de la table */
margin: _____ 0; /* Ajout d'une marge de 20px au-dessus et en dessous de la table pour l'espacer
des éléments adjacents */   box-shadow: _____ 15px rgba(0, 0, 0, 0.15); /* Ajout d'une ombre douce
autour de l'ensemble de la table */   border: _____ _____#3498db; /* Ajout d'une bordure bleue de 2
pixels d'épaisseur autour de l'ensemble de la table */
}

/* Styles pour l'en-tête de table */ table
th {
  __________: #555; /* Définit une couleur de fond sombre (grise) pour les en-têtes des colonnes */
_____: white; /* La couleur du texte dans les en-têtes est mise en blanc pour contraster avec le
fond sombre */
  _____: 10px 15px; /* Ajout d'un espacement interne pour éloigner le texte des bordures des
entêtes */   border: _____ _____ #666; /* Bordure grise foncée d'1 pixel autour de chaque en-
tête */
}

/* Styles pour les cellules de la table */ table
td {
  _____: 10px 15px; /* Espacement intérieur pour éloigner le texte des bordures de chaque cellule
*/
  _____: 1px solid #ddd; /* Bordure gris clair d'1 pixel autour de chaque cellule pour délimiter
les cellules */
}

/* Effet de survol pour les cellules de la table */ table
td:hover {
  __________: 0 0 15px rgba(0, 0, 0, 0.2); /* Ombre douce appliquée à la cellule lorsqu'elle est
survolée, donnant un effet de "soulèvement" */
  _____________: #ddd; /* Assombrissement léger du fond de la cellule survolée pour indiquer
l'interaction */
}
```

```
/* Styles pour l'en-tête (caption) de la table */ caption
{
  _____: 10px; /* Espacement intérieur autour du texte du titre (caption) */
  background-color: __________; /* Couleur de fond bleue pour faire ressortir le titre de la table
*/
  color: _____; /* Couleur de texte blanche pour contraster avec le fond bleu */   font-size:
_____; /* Augmentation de la taille du texte pour mettre en évidence le titre */   font-weight:
_____; /* Le texte du titre est rendu gras pour attirer l'attention */   text-_____: 1px 1px 1px
#666; /* Une ombre légère est ajoutée derrière le texte pour lui donner de la profondeur et du
relief */
  __________: 0 -4px 8px rgba(0, 0, 0, 0.1) inset; /* Une ombre interne est appliquée en bas du
titre pour donner l'impression qu'il est "enfoncé" */
  ____________________________; /* Une bordure bleue foncée est ajoutée en bas pour séparer
visuellement le titre du reste de la table */
}

/* Effet de survol pour l'en-tête de la table */ caption:hover
{
  _______________: #2980b9; /* Lors du survol du titre, sa couleur de fond devient légèrement plus
foncée pour indiquer l'interaction */
}
```

III. Marquer les tâches comme terminées

Objectif : Permettre à l'utilisateur de marquer une tâche comme terminée en cochant une case.

Remarques :

Les instructions fournies ici sont destinées à vous orienter et ne sont pas exhaustives. Il est essentiel que vous compreniez les étapes afin de pouvoir compléter le code de manière autonome.

Pour prendre en compte l'état de la tâche (terminée ou non), nous allons utiliser un tableau de booléens.

1. Ajout d'une colonne "Terminée" avec des cases à cocher :

Étape N°17 : Ajoutez en javascript, dans la fonction *AjouterTacheHTML,* une colonne dont le titre est *Terminée* (balise th) dans laquelle on trouvera des cases à cocher. Une pratique courante est de placer la case à cocher avant le numéro ou le libellé de la tâche pour permettre une sélection rapide et intuitive par l'utilisateur. Pensez à modifier la largeur, ainsi que les alignements verticalement et horizontalement.

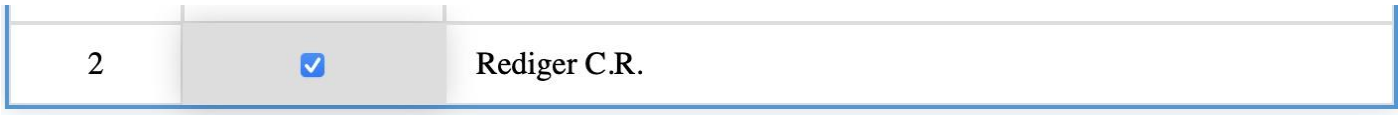
Étape N°18 : Créez un autre tableau de booléen *TableauTermine*, qui va représenter l'état *Terminée* de chaque tâche. Sa longueur sera la même que le tableau *TableauTaches*. Gérez ce tableau à la suite de *TableauTaches* (dans *ajouterTache()*) . Au démarrage toutes les valeurs sont à *false ;*

Étape N°19 : Insérez un *td* à droite du numéro et y mettre, avec *appendChild* ,un *input* dont les attributs sont *checkbox* pour *type* et *TableauTermine.length* pour *l'id*.

Étape N°20 : Comment pouvez-vous ajouter un gestionnaire d'événements à chaque case à cocher afin de surveiller les changements d'état ?

Étape N°21 : Associez cet événement à une fonction nommée Cocher.

Étape N°22 : Dans cette fonction Cocher, comment afficheriez-vous un message approprié dans la console, selon que la case vient d'être cochée ou décochée ?



Fonction Cocher

Étape N°23 : Complétez la fonction change.

Consignes :

- Lorsque vous définissez la fonction de rappel pour l'événement "change", utilisez event comme paramètre.
- Cet objet event représente l'événement qui a été déclenché.
- Grâce à ce paramètre, la fonction peut accéder aux détails de l'événement, tels que l'élément qui a déclenché l'événement via event.target.
- Par exemple, vous pouvez récupérer l'ID de cet élément et savoir si la case a été cochée ou non en utilisant les propriétés id et checked de event.target.

Étape N°24 : Affichez dans la console, l'identifiant de la case à cocher qui a appelé la fonction ainsi que si elle est cochée ou non.

Étape N°25 : Cet identifiant, est également l'index de la ligne en soustrayant un (car la première valeur est à l'index 0).

Étape N°26 : Mettez à jour le tableau *TableauTermine* (false = pas terminé, true = terminé)

Étape N°27 : Avec querySelectorAll, sélectionnez tous les balises *td* dans une variable *tousLesTD* et les afficher dans la console.

Étape N°28 : Expliquez cette ligne :

```
    let texteBrut = tousLesTD[(identifiantCase-1) * 3+2].innerText;
```

et notamment l'index identifiantCase-1) * 3+2

Si la tâche est terminée, il faudrait barrer le texte du libellé comme ceci :

Numéro	Terminée ?	
1	☑	~~Rédiger C.R.~~

Étape N°29 : Quelle est la balise html qui permet de barrer un texte ?

Complétez ce code :

```
    if (event.target.checked)          tousLesTD[(identifiantCase-1) * 3+2].innerHTML =
_______ + texteBrut+_________;      else      tousLesTD[(identifiantCase-1) *
3+2].__________ = __________;
```

Étape N°30 : Terminez si nécessaire, le code de la fonction *Cocher*

IV. Filtrage des tâches

Objectif :

Ajouter des options de filtrage pour afficher toutes les tâches, les tâches terminées ou les tâches non terminées.

Étape N°31 : Créez un élément <select> dans votre interface utilisateur. Cet élément servira de sélecteur pour les options de filtrage. Ajoutez trois options à ce sélecteur : "Toutes les tâches", "Tâches terminées" et "Tâches non terminées". Assurez-vous d'attribuer des valeurs appropriées à chaque option.

Étape N°32 : Ajoutez un gestionnaire d'événements pour le sélecteur <select> que vous avez créé à la question précédente. Lorsque l'utilisateur choisit une option, appelez une fonction de filtrage appropriée en fonction de la valeur de l'option sélectionnée.

Étape N°33 : Dans votre code JavaScript, créez une fonction filterTasks qui accepte un argument représentant le type de filtre (par exemple, "all", "completed", "uncompleted"). Cette fonction doit mettre à jour l'affichage en fonction du filtre sélectionné. Par exemple, si le filtre "completed" est sélectionné, la fonction doit afficher uniquement les tâches terminées.

V. Suppression des tâches

Objectifs :

Permettre à l'utilisateur de supprimer une tâche de la liste.

Étape N°34 : Créez une fonction JavaScript qui supprime une tâche de la liste lorsqu'un bouton de suppression est cliqué. Quel événement JavaScript devez-vous utiliser pour détecter ce clic ?

Étape N°35 : Comment pouvez-vous associer la fonction de suppression aux boutons de suppression de chaque tâche dans votre interface utilisateur HTML ? Quelle propriété ou attribut HTML est nécessaire pour identifier chaque bouton de suppression ?

Étape N°36 : Lorsque la fonction de suppression est déclenchée, comment pouvez-vous identifier quelle tâche spécifique doit être supprimée de la liste ? Comment pouvez-vous accéder à l'élément HTML de cette tâche ?

Étape N°37 : Comment pouvez-vous mettre à jour l'interface utilisateur HTML pour refléter la suppression de la tâche ? Quelle méthode JavaScript pouvez-vous utiliser pour supprimer un élément HTML de la page ?