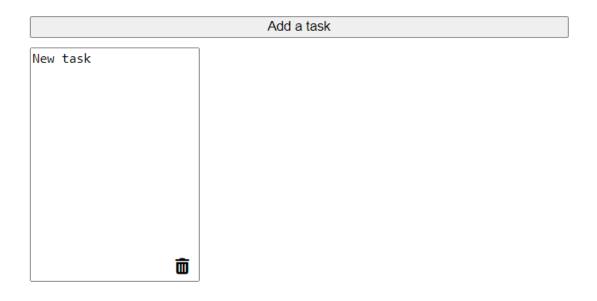
TODO - Exercice quidé

Pensez à effectuer des **recherches** sur chaque fonction que vous croiserez au fil de cet exercice guidé afin d'en comprendre tous les aspects. N'hésitez pas à faire des **console.log()** pour comprendre ce qui se trouve dans vos variables. A **tester** chaque bout de code que vous ne comprenez pas.

I. Créer l'interface

Créer le css pour obtenir une interface telle quelle :

Todo list



```
<body>
11
         <div id="wrapper">
12
13
              <h1>Todo list</h1>
14
15
              <div id="todo">
16
                  <div id="count"></div>
17
                  <button id="btn">Add a task/button>
18
                  <div id="todoCards">
19
                      <div class="todoCard">
20
21
                          <textarea class="task" maxlength="200" cols="20" rows="15">New task</textarea>
22
                          <span class="delBtn"><i class="fa fa-trash" aria-hidden="true"></i></span>
23
                      </div>
24
25
                  </div>
26
              </div>
27
28
         </div>
         <script src="script.js"></script>
```

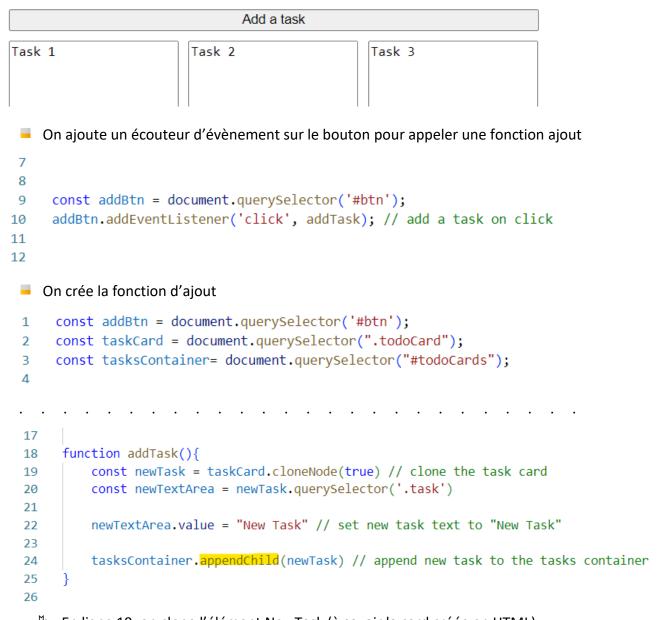
On pourra utiliser fontAwesome pour l'icône 🔤

ELAN
14 rue du Rhône
67100 STRASBOURG
TEL: 03.88.30.78.30
www.elan-formation.fr

TODO - Exercice quidé

II. Ajout d'une todoCard

Faire fonctionner le bouton « Add a task »



- En ligne 19, on clone l'élément NewTask (à savoir la card créée en HTML)
- 🔖 En ligne 20 et 22, on définit la valeur de la zone de texte afin qu'elle ne soit pas vide
- 🖔 En ligne 24, on ajoute cette nouvelle carte dans le DOM (afin de l'afficher. Faites le test d'enlever cette ligne et d'en constater le changement)

FLAN 14 rue du Rhône 67100 STRASBOURG TEL: 03.88.30.78.30

TODO - Exercice quidé

III. Supprimer une todoCard

Faire fonctionner le bouton de suppression



On ajoute un écouteur d'évènement sur le bouton pour appeler une fonction suppression

```
const delBtn = document.querySelector('.delBtn');
5
    delBtn.addEventListener('click', function() { // delete default task on click
6
        deleteTask(taskCard); // target the right task
7
8
      });
```

On crée la fonction de suppression

```
function deleteTask(task){
28
         task.remove(); // remove the task
29
30
```

On ajoute l'écouteur d'évènement sur l'élément cloné dans la fonction ajout, afin de faire fonctionner le bouton suppression sur les nouvelles cards

```
13
     function addTask(){
14
         const newTask = taskCard.cloneNode(true) // clone the task card
         const newDelBtn = newTask.querySelector('.delBtn')
15
         const newTextArea = newTask.querySelector('.task')
16
17
         newTextArea.value = "New Task" // set new task text to "New Task"
18
         newDelBtn.addEventListener('click', function() { // add delete event listener to new task
19
20
             deleteTask(newTask); // target the new task
21
         });
22
         tasksContainer.appendChild(newTask) // append new task to the tasks container
23
         updateCount();
24
25
```

IV. Compter le nombre de listes

A vous de jouer!

Ajouter un compteur de cards

- Ajouter une div dans l'html où l'on indiquera le nombre de cards (le compteur)
- Faire une fonction JS qui récupère la div fraichement créée, un moyen de compter le nombre de cards puis mettant à jour le compteur
- Appeler cette fonction dans le JS (afin d'initialiser le compteur quand on charge la page), puis l'appeler depuis les 2 fonctions ajout et suppression (pour mettre le compteur à jour)

ELAN 14 rue du Rhône 67100 STRASBOURG TEL: 03.88.30.78.30 www.elan-formation.fr