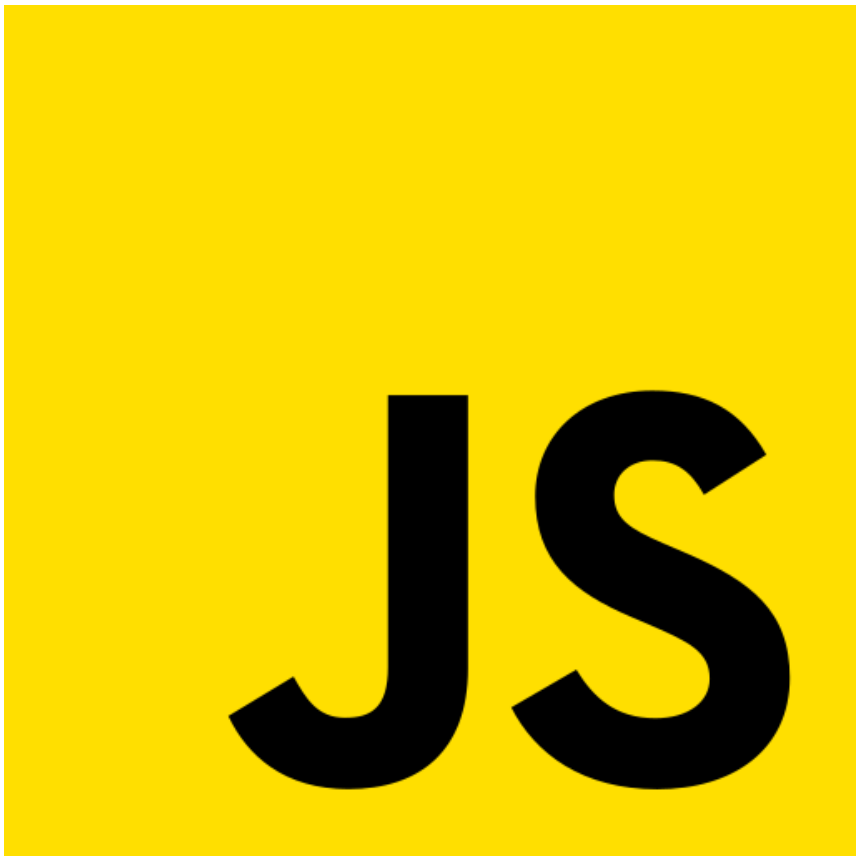
JavaScript

Sprint 0 Lab 1



Felipe Izquierdo Romero

Índice

1. Descripción del problema

Para el laboratorio de JavaScript se pide como ejercicio hacer un administrador de tareas. El laboratorio proporciona una maqueta del index.html y el estilo CSS y hay que hacer con un script que puedas añadir tareas marcarlas como realizadas y eliminar tareas.

2. Desarrollo

Para el desarrollo de este ejercicio se explicarán las partes del código para entender el funcionamiento general del script.

Primero se declaran las variables del input de texto del botón de añadir nota y de la lista donde añadir cada nota. Se crea además una lista vacía para guardar las notas por detrás de la vista.

Al botón de añadir nota se le añade un listener y al hacer clic llamara a la función addNota().

```
1  // Pon a prueba tus conocimientos sobre JavaScript aqui
2
3  const inputText = document.querySelector('#taskInput');
4  const botonInput = document.querySelector('#addTaskButton');
5  const taskList = document.querySelector('#taskList');
6
7  let ListaTareas = [];
8
9  botonInput.addEventListener('click',addNota);
10
```

Ilustración 1- Declaracion de variables

La función addNota() crea una variable que se añadirá a la lista de tareas, y en la vista se añadirá una tarea con el título un botón de borrado y un checkbutton.

```
10
11 function addNota(){
12
13     let Tarea = crearTareas(inputText.value);
14     ListaTareas.push(Tarea);
15     taskList.append(PatronDeTarea(inputText.value,Tarea.id));
16     inputText.value='';
17
18     console.log(ListaTareas.length);
19
20     ListaTareasConsola();
21 }
```

Ilustración 2- Función addNota

Dentro de esta función se llama a tres funciones más:

- crearTareas(): crea una tarea para guardar en al lista de tareas.
- PatronDeTarea(): retorna un patron de una tarea de la vista.
- ListaTareasConsola(): para observar en la consola las tareas que hay en segundo plano en la lista de tareas.

La función crearTareas() contiene un bloque if else para ver si esta vacía la lista y en ese caso acceder al id del ultimo valor y sumarle uno , de esa manera se autoincrementará y nunca se repetirá ningún identificador.

Además de crear un objeto Tarea con id, nombre de la tarea y un valor booleano que se usará para comprobar si la tarea se completó o no.

Finaliza con el retorno de la tarea creada.

```
function crearTareas(textoNota){
    let id;
    if(ListaTareas.length > 0){
        let ultimoId = parseInt(ListaTareas[ListaTareas.length-1].id.split('-')[1])
        id = 'task-' + (ultimoId+1);
    }else{
        id = 'task-' + 1;
    }

    let nombre = textoNota;
    let completada = false;

    let Tarea = {
        'id':id,
        'nombre':nombre,
        'completada':completada
    }
    return Tarea;
    //console.log(id);
}
```

Ilustración 3- Funcion crearTareas ()

La función PatroDeTarea() es mas extensa pero no mucho más difícil.

Primero crea los elementos html que deben tener la tarea y les da el estilo CSS que aporta el laboratorio.

A continuación se añaden los elementos hijos a cada elemento contenedor con la función append().

Después se añaden los listeenr de los botones de eliminar y de checkear que la tarea se completó.

Retorna el elemento grafico creado.

```
function PatronDeTarea(textoNota,id){

    let modeloTarea = document.createElement('li');
    let article = document.createElement('article');

    let inputCheckBox = document.createElement('input');
    let span = document.createElement('span');
    let trash = document.createElement('i');

    inputCheckBox.type = 'checkbox';
    span.innerHTML = textoNota;

    inputCheckBox.classList.add('task-checkbox');
    span.classList.add('task-text');
    trash.classList.add('fa','fa-trash');


    modeloTarea.append(article);
    article.append(inputCheckBox);
    article.append(span);
    modeloTarea.append(trash);

    trash.addEventListener('click',() => {
        modeloTarea.remove();
        ListaTareas = ListaTareas.filter(Tarea => Tarea.id !== id);
        ListaTareasConsola();
    });
    inputCheckBox.addEventListener('change', function(event) {
        for(let i = 0;i<ListaTareas.length;i++){
            if(ListaTareas[i].id === id){
                ListaTareas[i].completada =  event.target.checked;
            }
        }
        ListaTareasConsola();
    });

    return modeloTarea;
}
```

Ilustración 4- PatronDeTarea()

Por ultimo ListaTareasConsola() es una función sencilla que limpia la consola e imprime cada elemento de la lista de tareas y sus propiedades.

```
83  function ListaTareasConsola(){
84      console.clear();
85      ListaTareas.forEach((Tarea) =>{
86
87          console.log(Tarea.id);
88          console.log(Tarea.nombre);
89          console.log(Tarea.completada);
90      });
91  }
```

Ilustración 5- ListaTareasConsola()

Para finalizar queda aclarar que la maqueta del laboratorio tiene una Tarea prediseñada que habría que eliminar o comentar en el html puesto que sino habría una tarea sin funcionalidad que no se podría eliminar de la lista de la interfaz gráfica.

```
13  <body>
14      <main>
15          <header class="header">
16              <h1>Mis tareas del dia</h1>
17          </header>
18          <article class="container">
19              <div class="form">
20                  <input type="text" id="taskInput" placeholder="Ingresa una tarea">
21                  <button id="addTaskButton" class="btn">+</button>
22              </div>
23              <ul id="taskList" class="task-list">
24                  <!-- <li>
25                      <article>
26                          <input type="checkbox" class="task-checkbox">
27                          <span class="task-text">Tarea 1</span>
28                      </article>
29                      <i class="fa fa-trash"></i>
30                  </li> -->
31              </ul>
32          </article>
33      </main>
34      <script src="script.js"></script>
35  </body>
36
```

Ilustración 6- Tarea comentada en el html