```
// IMPORTANT : SOUS WINDOWS, UTILISER UN SERVEUR LOCAL COMME XAMPP (PAR EXEMPLE) POUR QUE LA SAUVEGARDE
// ET LE CHARGEMENT DU COOKIE FONCTIONNE !
// SOUS UBUNTU : PAS BESOIN D'UTILISER UN SERVEUR LOCAL, MAIS LE COOKIE SERA STOCKE EN SESSION
// (IL SERA EFFACE SI LE NAVIGATEUR EST FERME), NE FONCTIONNE DONC QUE SI LA PAGE
// EST RECHARGEE

cd /home/event/discovery_piscine
mkdir cell33
cd cell33
mkdir ex03
cd ex03

cp /home/event/discovery_piscine/cell32/ex02/calc.html  index.html

vim index.html
i

<!DOCTYPE html>
<html lang="en">
        <head>
                <title>TO DO</title>
                <meta charset="UTF-8">
                <meta http-equiv="X-UA-Compatible" content="IE=edge">
                <meta name="viewport" content="width=device-width, initial-scale=1.0">
                <style>
                        html, body, .container {
                                height: 100%;
                        }

                        body {
                                margin: 0;
                        }

                        .container {
                                display: flex;
                                // pour que le bouton se place sous la div et non à droite
                                // (disposition en colonne et non en ligne)
                                flex-direction: column;
```

```
                        justify-content: center;
                        align-items: center;
                }

                // mise en forme de la div contenant la liste des TODOS
                #ft_list {
                        margin: 20px;
                        background-color: pink;
                        border: 1px dashed black;
                }

                // mise en forme des TODOS individuellement
                // (ils seront ajoutés dynamiquement en javascript)
                .todo_div {
                        height: 20px;
                        width: 100px;
                        margin: 10px;
                        border: 1px solid grey;
                        background-color: skyblue;
                        text-align: center;
                }

                // pointeur (main) lors du survol de la souris sur le bouton
                button:hover {
                        cursor: pointer;
                }
        </style>
</head>
<body>
        // pour flex
        <div class="container">
                // contiendra la liste des TODO
                <div id="ft_list"></div>
                // bouton pour créer les TODO
                <button id="new_btn">New</button>
        </div>
        // lien vers le javascript
        <script src="todo.js"></script>
```

```
        </body>
</html>

echap
:wq
enter

vim todo.js
i

// pour sélectionner le bouton de création de TODO
const newBtn = document.getElementById("new_btn");
// pour sélectionner la div qui accueillera la liste des TODO
const ftList = document.getElementById("ft_list");
// tableau des TODO
let todoList = [];

// lorsque le bouton NEW sera cliqué (création de la TODO)
newBtn.addEventListener("click", function() {

        // on affiche un prompt, qui demandera de créer un TODO
        // un prompt contient un champ texte, un bouton "Annuler" et un bouton "OK"
        // avec un message au-dessus du champ texte (ici, Fill a new TO DO)
        // la valeur du champ texte sera récupérée dans la variable todoText
        let todoText = prompt("Fill a new TO DO");

        // si le texte récupéré n'est pas "null" (non initialisé, n'existe pas)
        // et que son nombre de caractère est
        // supérieur à zéro (après avoir supprimé les espaces blancs avant et après les caractères, grâce à trim)
        if (todoText !== null && todoText.trim().length > 0) {
                // on définit le contenu de ce nouveau TODO à la valeur du texte
                // sans les espaces avant et après le texte
                let newContent = todoText.trim();

                // on exécute la fonction pour ajouter ce TODO avec cette valeur (son contenu texte)
                addTodo(newContent);
        }
});
```

```
// fonction pour ajouter un TODO, avec son contenu texte
function addTodo(content) {
	// on crée la div qui contiendra le TODO
	let todoDiv = document.createElement("div");
	// on ajoute la classe "todo_div" à cette div
	todoDiv.classList.add("todo_div");
	// on attribue à la div, comme contenu texte, le texte de la TODO
	todoDiv.textContent = content;

	// on insère la div todoDiv (1er paramètre) ainsi créée dans ftList, avant le premier enfant de ftList
	// (2ème paramètre)
	// cette div sera ainsi toujours insérée avant le 1er enfant
	// Remarque : si ftList n'a pas encore d'enfants, la div todoDiv sera insérée quand même
	ftList.insertBefore(todoDiv, ftList.firstChild);

	// lorsque la div de la TODO sera cliquée (pour afficher la fenêtre de confirmation de suppression)
	todoDiv.addEventListener("click", function() {
		// on affiche un popup de confirmation avec le message :
		// "Do you want to remove that TO DO ?"
		// si l'utilisateur clique sur OK, window.confirm() retournera true, on rentrera donc
		// dans le bloc d'instructions
		// s'il clique sur Annuler,  window.confirm() retournera false, on ne rentrera donc
		// pas dans le bloc d'instructions
		if (window.confirm("Do you want to remove that TO DO ?"))
			// on supprime la div du TODO
			todoDiv.remove();
	});
}

// avant que la page ne soit "déchargée" (avant de quitter, fermer la page)
window.addEventListener("beforeunload", function() {
	// récupère la collection HTML (HTMLCollection) de tous les éléments ayant la classe
	// todo_div
	let todoDivs = document.getElementsByClassName("todo_div");
	let JSONString;
	let encodedTodoList;
```

```
        // sur chacun de ces éléments
        // ATTENTION : todoDivs est de type HTMLCollection, il faut donc le convertir en tableau
        // avec Array.from(todoDivs) avant
        // attention à passer todoDiv (chacun des éléments de TodoDivs , CREES AUTOMATIQUEMENT
        // en paramètre et à travailler dessus pour extraire le contenu texte
        Array.from(todoDivs).forEach(function(todoDiv) {
                        // on ajoute, au début du tableau todoList (comme on le ferait dans la div ft_list
                        // (le dernier élément ajouté apparaît tout en haut),
                        // le contenu texte de l'élément en cours
                        todoList.unshift(todoDiv.textContent);
        });

        // on transforme le contenu du tableau en chaîne JSON
        JSONString = JSON.stringify(todoList);

        // on encode cette chaîne JSON comme composante d'URI, ce qui remplace
        // les caractères spéciaux par des séquences d'échappement (encodage utf-8 du caractère)
        // par exemple, un espace sera traduit par %20
        encodedTodoList = encodeURIComponent(JSONString);

        // on exécute la fonction setCookie, pour créer le cookie, avec les paramètres
        // "TODO" (le nom du cookie), la liste encodée, et 7 (le nombre de jours pour la
        // validité du cookie
        setCookie("TODO", encodedTodoList, 7);
});

// fonction pour créer le cookie (avec son nom, son contenu et le nombre de jours pendant lequel il sera valide)
function setCookie(name, value, days) {
        // on initialise la variable expires vide
        let expires = "";

        // si on a lancé la fonction setCookie avec un nombre de jours pour la validité du cookie
        if (days) {
                // on obtient la date actuelle
                let date = new Date();
                // on calcule une nouvelle date en additionnant le nombre de
                // millisecondes depuis le 1er janvier 1970 et le nombre de millisecondes dans "days" jours
                // (7 jours ici) en multipliant par le nombre d'heures, de minutes, de secondes et de
```

```
                // millisecondes dans "days" jours
                date.setTime(date.getTime() + (days * 24 * 60 * 60 * 1000));
                // on convertit cette date au format chaîne de caractères UTC (avec le GMT)
                // le format UTC est YYYY-MM-DDTHH:MM:SSZ
                // et UTCString est sous la forme "Wed, 14 Jul 2021 12:00:00 GMT" par exemple
                // et on y ajoute "; expires=" devant
                expires = "; expires=" + date.toUTCString();
        }
        // le cookie est formé de son nom ("TODO") "="
        // sa valeur (la chaîne JSON encodée) ou une chaîne vide sinon
        // sa date d'expiration (plus précisément "; expires=date_d-expiration"
        // ou une chaîne vide si setCookie a été appelé sans days (dans ce cas, le cookie sera
        // un cookie de session, et il sera effacé si le navigateur est fermé)
        // IMPORTANT : il faut que la page soit ouverte sur un serveur (local ou distant) pour que
        // les cookies ne soient pas supprimés lorsque le navigateur est fermé !
        // cela fonctionne quand même lors du rechargement de la page (car le cookie est stocké en
        // session) si la page est en local cependant (MAIS SEULEMENT SOUS UBUNTU)
        document.cookie = name + "=" + (value || "") + expires + "; path=/" + "; SameSite=Strict";
}

window.addEventListener("load", function() {    // lorsque la page sera chargée
        let todoCookie = getCookie("TODO");     // on récupère le cookie "TODO"

        // si il a été trouvé
        if (todoCookie) {
                // on décode le cookie (les espaces seront écrits comme %20 sinon)
                let decodedTodoList = decodeURIComponent(todoCookie);
                // on définit la chaîne JSON qui accueillera la liste des todo
                let JSONTodoList;

                // si l'instruction suivante échoue (la transformation de la chaîne de caractère JSON en objet
                // JSON), on ira dans le catch, un message d'erreur sera indiqué dans la console,
                // et la 2ème instruction du try ne sera pas exécutée
                try {
                        // transformation en objet JSON
                        JSONTodoList = JSON.parse(decodedTodoList);
                        // pour chacun des items (TODO) de l'objet JSON
                        // on lance la fonction addTodo (pour créer un TODO) avec le contenu
```

```
                        // pour recréer les TODO une par une
                        JSONTodoList.forEach(addTodo);
                }
                // si la première instruction dans try échoue
                catch(e) {
                        // on affiche l'erreur dans la console
                        console.error("Cookie decode error", e);
                }
        }
});

// pour obtenir le cookie sauvegardé
function getCookie(name) {
        // on récupère la liste des cookies dans le tableau cookies
        // (en découpant la liste de tous les cookies, document.cookie,
        // selon le séparateur ";" (le séparateur de chacun des cookies))
        let cookies = document.cookie.split(";");

        // pour chacun de ces cookies (pour chacun des éléments du tableau de cookies),
        // du début à la fin du tableau (de 0 à la longueur du tableau non inclus),
        // en incrémentant l'indice de 1 à chaque boucle pour parcourir tout le tableau
        for (let i = 0 ; i < cookies.length ; i++) {
                // on supprime tous les espaces avant et après la chaîne de caractère du cookie
                let cookie = cookies[i].trim();
                // si le cookie commence par "TODO="
                if (cookie.startsWith(name + '='))
                        // on retourne la valeur du cookie après "TODO=" (après "TODO" + 1 caractère)
                        return cookie.substring(name.length + 1);
        }
        // on ne retourne rien si la fonction n'a pas déjà retourné quelque chose,
        // donc si le cookie "TODO" n'a pas été trouvé
        return null;
}

echap
:wq
enter
```

```
firefox index.html
```