Aplikacje Internetowe 1

Laboratorium 3: JS i DOM na przykładzie Todo List

Data laboratorium:	27.10.2022
Termin sprawozdania:	10.11.2022
lmię i nazwisko:	Szymon Buckowski

0. Wprowadzenie

Tutaj należy umieścić 2-3 zdania opisujące zawartość sprawozdania i streszczające przebieg laboratorium. Wypełnienie tej rubryki warunkuje dalsze sprawdzenie sprawozdania.

Celem laboratorium było stworzenie dynamicznej listy czynności do zrobienia (Todolist). Składa się ona z listy, pola wyszukiwarki, pola dodawania oraz przycisku usuwania. Do wczytywania i zapisywania zawartości listy należało wykorzystać Local Storage.

1. Strona HTML ze wszystkimi elementami

Należy przedstawić HTML ze wszystkimi wymaganymi elementami, tj. pole wyszukiwarki, lista, pole dodawania, przycisk usuwania. 1 punkt.

Search task		
• Zadanie 2		
2022-11-26		
Delete		
• Zadanie 1		
2022-11-22		
Delete		
• Task 3		
Delete		
• Task 2		
2022-11-24		
Delete		
• Task 1		
2022-11-23		
Delete		
	dd . mm . rrrr	Submit

2. Wczytywanie i zapisywanie listy z/do Local Storage

▼ 🖹 Lokalna pamięć	Key	Value
file:///C:/Users/sbuck/Desktop/AI/Lab_3/Lab_3.htm	0	["Task 1","2022-11-23"]
▶ 🗄 Ciasteczka	1	["Task 2","2022-11-24"]
▶ 🗄 IndexedDB	2	["Task 3",""]
Pamięć podręczna	3	["Zadanie 1","2022-11-22"]
Pamięć sesji	4	["Zadanie 2","2022-11-26"]

Lista jest wczytywana w kolejności klucza malejąco, w celu wyświetlenia jej na stronie od najnowszego zadania do najstarszego.

Dla każdej pozycji w local storage skrypt tworzy pozycje listy która składa się z 3 elementów - nazwy zadania, daty (opcjonalnie) oraz przycisku usuwania. Na koniec pozycja jest dodawana do listy.

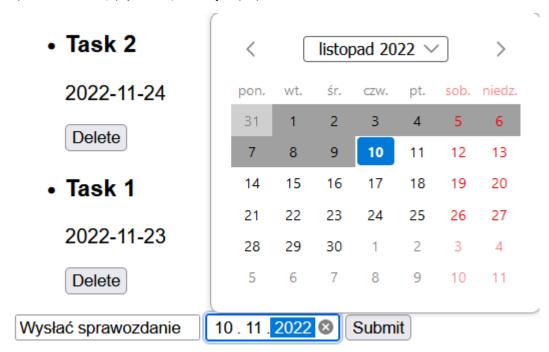
```
function showTODO() {
var list = document.getElementById('lista');
var max = Number.MAX VALUE;
var \max 2 = -1;
 for(var j = 0; j<localStorage.length; j++){</pre>
 for(var i = 0; i<localStorage.length; i++){</pre>
     var klucz = Number(localStorage.key(i));
     if(klucz>max 2 && klucz<max)
         \max 2 = klucz;
max = max_2;
max_2 = -1;
var item = document.createElement("li");
item.className = "tasks";
var i1 = document.createElement("h3");
i1.textContent = JSON.parse(localStorage.getItem(max))[0];
i1.id = "task name";
var i2 = document.createElement("p");
i2.textContent = JSON.parse(localStorage.getItem(max))[1];
var i3 = document.createElement("button");
// i3.innerHTML = max;
i3.innerHTML = 'Delete';
i3.className = "del";
i3.id = max;
item.appendChild(i1);
item.appendChild(i2);
item.appendChild(i3);
 list.appendChild(item);
```

Lista jest zapisywana w taki sposób, że dodanie nowego zadania dodaje je do local storage.

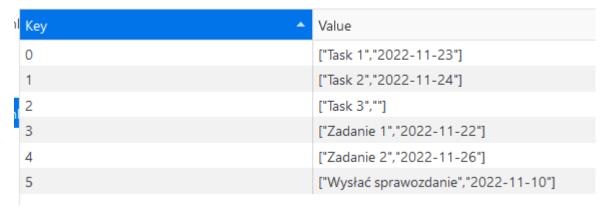
3. Dodawanie pozycji listy

Należy przedstawić jak dodawane są elementy do listy. 1 punkt.

Dodawanie elementów do listy odbywa się poprzez wpisanie w pole tekstowe na dole listy nazwy zadania, wybranie terminu wybrania zadania (opcjonalnie) i kliknięcie przycisku **Submit**.



Następnie zadanie jest zapisywane do local storage - dostaje **key** będący wartością o jeden większą od największego klucza (w celu utrzymania kolejności dodania), a dane są przechowywane w **value** jako tablica.



Po dodaniu zadania do local storage cała lista jest od nowa wczytywana na stronę i dodane zadanie pojawia się na górze listy. Na koniec pole dodawania zadania i terminu są czyszczone. Search task · Wysłać sprawozdanie 2022-11-10 Delete • Zadanie 2 2022-11-26 Delete Zadanie 1 2022-11-22 Delete • Task 3 Delete • Task 2 2022-11-24 Delete Task 1

2022-11-23

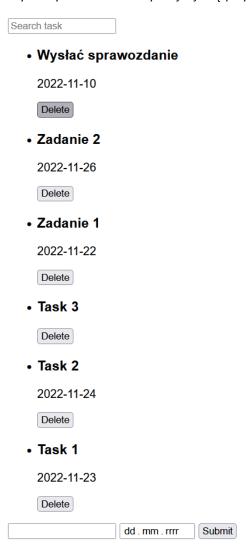
Delete

dd . mm . rrrr Submit

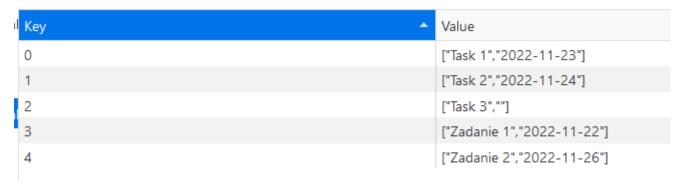
4. Usuwanie pozycji listy

Należy przedstawić jak usuwane są elementy listy. 1 punkt.

Przy każdym elemencie listy znajduje się przycisk **Delete**, który służy do usunięcia danego zadania.



Id przycisku jest równe wartości klucza przypisanego danemu zadaniu w local storage, dzięki czemu zadanie jest znajdowane w local storage i usuwane z niego, po czym lista jest wczytywana ponownie na stronę już bez usuniętego zadania.

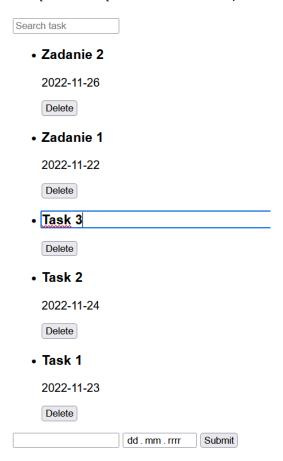


Search task		
• Zadanie 2		
2022-11-26		
Delete		
• Zadanie 1		
2022-11-22		
Delete		
• Task 3		
Delete		
• Task 2		
2022-11-24		
Delete		
• Task 1		
2022-11-23		
Delete		
	dd . mm . rrrr	Submit

5. Edycja pozycji listy

Należy przedstawić edycję pozycji listy. 1 punkt.

Kliknięcie w nazwę zadania umożliwia edytowanie nazwy.



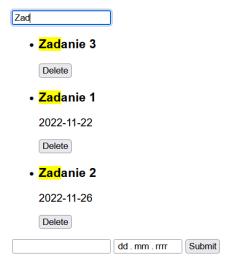
Po kliknięciu poza zadanie zapisuje zmianę do local storage.



6. Wyszukiwanie

Należy udokumentować funkcjonowanie wyszukiwarki. 1 punkt.

Po wpisaniu co najmniej 3 znaków w wyszukiwarkę, skrypt pobiera je z pola tekstowego, czyści listę na stronie i iteruje się po local storage porównując wartości z wyszukiwarki i pierwszej wartości w tablicy **value** dla danego elementu przy pomocy wyrażeń regularnych. Jeżeli znalazł dopasowanie wyświetla takie zadanie na stronie dodając je na górze listy.



Edytowanie oraz usuwanie zadania również działa dla wyszukanych zadań.

7. Oznaczanie wyszukiwanej frazy na liście

Należy przedstawić, jak wyszukiwana fraza jest wyróżniana na liście wyszukanych zadań. 1 punkt.

W funkcji wyświetlającej wyszukane zadanie po stworzeniu elementu odpowiadającego za nazwę zadania i zaciągnięciu do niego nazwy z local storage tworzy zmienną będącą frazą wpisaną w wyszukiwarce i podmienia **innerHTML** tego elementu zamieniając wyszukiwaną frazę na taką samą tylko z wyróżnieniem.



8. Linki i uwagi

W tej sekcji należy umieścić link do działającej aplikacji listy zadań oraz dowolne uwagi.

. . .