

## JS I DOM NA PRZYKŁADZIE LISTY TODO

### SPIS TREŚCI

Spis treści	1
Cel zajęć	1
Rozpoczęcie	1
Uwaga	2
Wymagania	2
Strona HTML	3
Klasa Todo	3
Dodawanie pozycji listy	4
Usuwanie pozycji listy	4
Edycja pozycji listy	5
Odczyt / Zapis LocalStorage	6
Wyszukiwanie	7
Commit projektu do GIT	8
Podsumowanie	8

### CEL ZAJĘĆ

Celem głównym zajęć jest zdobycie następujących umiejętności:

- przemieszczania się po drzewie DOM;
- dodawania, usuwania, edytowania elementów drzewa DOM.

W praktycznym wymiarze utworzona zostanie dynamiczna lista czynności do zrobienia (lista To Do).

### ROZPOCZĘCIE

Rozpoczęcie zajęć. Powtórzenie metod przemieszczania się po drzewie DOM.

Wejściówka?

## UWAGA

Ten dokument aktywnie wykorzystuje niestandardowe właściwości. Podobnie jak w LAB A wejdź do Plik -> Informacje -> Właściwości -> Właściwości zaawansowane -> Niestandardowe i zaktualizuj pola. Następnie uruchom ten dokument ponownie lub Ctrl+A -> F9.

## WYMAGANIA

W ramach LAB B przygotowane powinny zostać:

- pojedyncza strona HTML ze skryptem ładowanym z zewnętrznego pliku JS
- lista zadań
- na dole listy pole tekstowe do dodawania nowych zadań, pole typu data/czas do określenia terminu wykonania zadania, przycisk dodawania zadania
- walidacja nowych zadań: co najmniej 3 znaki, nie więcej niż 255 znaków, data musi być pusta albo w przyszłości
- na górze listy pole wyszukiwarki
- po wpisaniu w wyszukiwarkę co najmniej 2 znaków na liście wyświetlają się wyłącznie pozycje zawierające wpisaną w wyszukiwarkę frazę
- wyszukiwana fraza zostaje wyróżniona w każdym wyniku wyszukiwania
- kliknięcie na dowolną pozycję listy zmienia ją w pole edycji; kliknięcie poza pozycję listy zapisuje zmiany
- obok każdej pozycji listy znajduje się przycisk Usun / Śmietnik
- wpisy na liście zapisują się do Local Storage
- po odświeżeniu strony lista wypełnia się wpisami z Local Storage

Mockupy:

Mockup of the application interface showing a list of tasks. The list includes items like 'chocolate', 'macaroon', 'chupa chups', 'candy canes', and 'bon bons' with associated dates and delete buttons. At the bottom, there is a text input field labeled 'do zrobienia...', a date picker set to '2000-01-01', and a 'Zapisz' button.

Mockup of the application interface showing a task being edited. The 'chupa chups' item is selected, and its details are shown in a form with a date picker and a 'Zapisz' button. The bottom bar remains the same with the 'do zrobienia...' field, date picker, and 'Zapisz' button.



Mockup of the application interface showing search results. The search bar contains the text 'on'. The list displays items like 'macaroon' and 'bon bons' that match the search criteria. The bottom bar is consistent with the previous mockups.

## STRONA HTML

Prace rozpocznij od implementacji HTML z danymi wpisanymi „na sztywno”. Upewnij się, że wstawione zostały wszystkie wymagane elementy – pole wyszukiwarki, lista, pole dodawania, przycisk usuwania. To laboratorium koncentruje się na JS, więc może być ładne, ale nie musi. Nie trać za dużo czasu na CSS.

Wstaw zrzut ekranu przedstawiający stronę HTML z polem wyszukiwarki, listą, polem dodawania, przyciskami usuwania:

### Lista zadań

- aaa 
- vbbbb 




Punkty:	0	1
---------	---	---

## KLASA TODO

Pierwszym instynktem może być chęć dodania zachowań bezpośrednio do elementów listy w drzewie DOM. Chociaż na krótką metę wydaje się być to najprostsze rozwiązanie, za chwilę okaże się krótkowzroczne i trudne do implementacji przy kolejnych punktach 😊

Najlepszym sposobem rozwiązania tego laboratorium jest utworzenie klasy Todo (albo po prostu obiektu z kilkoma metodami). Bez względu na przyjętą strategię, należy w tym nowoutworzonym bycie utworzyć tablicę `tasks` oraz metodę `draw()`, która wyczyści `div` z obecną wizualizacją zadań do zrobienia i wygeneruje ją na nowo na podstawie tablicy `tasks`.

W celu sprawdzenia poprawności działania, najlepiej dostać się do tablicy `tasks` i edytować jej zawartość, po czym ręcznie wywołać metodę `draw()`. Jeśli zawartość listy wyrenderuje się na nowo poprawnie – możemy iść dalej!


Zaimplementuj dodawanie, usuwanie, edycję pozycji listy – wszystko modyfikujące tablicę `tasks` i wywołujące na koniec metodę `draw()`.

Wstaw zrzut ekranu listy przed dodaniem nowego zadania:

## Lista zadań

Wyszukaj...

Nowe zadanie




dd . mm . rrrr 

Zapisz


Wstaw zrzut ekranu listy po dodaniu nowego zadania:

## Lista zadań

Wyszukaj...

- aaa (termin: 26.10.2024, 02:00:00) 
- vvvvv 
- bbbb 

Nowe zadanie

dd . mm . rrrr 

Zapisz

Punkty:

0




1

## USUWANIE POZYCJI LISTY


Wstaw zrzut ekranu listy przed usunięciem wybranego zadania:

## Lista zadań

Wyszukaj...

- aaa (termin: 26.10.2024, 02:00:00) 
- vvvvv 
- bbbb 

Nowe zadanie



dd . mm . rrrr 

Zapisz


Wstaw zrzut ekranu listy po usunięciu zadania:

# Lista zadań

Wyszukaj...

- aaa (termin: 26.10.2024, 02:00:00) 
- vvvvv 

Nowe zadanie

dd . mm . rrrr 

Zapisz

Punkty:

0


1

## EDYCJA POZYCJI LISTY


Wstaw zrzut ekranu listy przed edycją wybranego zadania:

# Lista zadań

Wyszukaj...

- aaa (termin: 26.10.2024, 02:00:00) 

Nowe zadanie


dd . mm . rrrr 

Zapisz


Wstaw zrzut ekranu listy w trakcie edytowania zadania i daty:

# Lista zadań

Wyszukaj...

- 26 . 10 . 2024 , 00 : 00 

Nowe zadanie

dd . mm . rrrr 

Zapisz

Wstaw zrzut ekranu listy po edycji zadania i daty. Upewnij się, że dane się zapisały i zadanie jest zmienione:

# Lista zadań

Wyszukaj...

- aaabbbbb (termin: 26.12.2025, 00:00:00) 🗑

Nowe zadanie

dd . mm . rrrr 📅

Zapisz

Punkty:

0

1

## ODCZYT / ZAPIS LOCALSTORAGE

Zastosowanie klasy Todo w realizacji tego laboratorium pozwala w bardzo łatwy sposób odczytywać i zapisywać stan listy do pamięci przeglądarki. Wystarczy serializacja / deserializacja za pomocą `JSON.parse()` i `JSON.stringify()`.

Wstaw zrzuty ekranu przedstawiające wygląd listy i zawartość local storage gdy na liście są pewne zadania:

### Lista zadań

Wyszukaj...

Nowe zadanie

dd . mm . rrrr 📅

Zapisz

🔍

🔧

📄

🐞

🌐

🛠

🔊

🧠

📄

🛡

🧵

🛑

InspektorKonsolaDebuggerSiećEdytor stylówWydajnośćPamięćDaneDostępnośćWątkiAdblock Plus

📁

📁

📁

📁

📁

📁

CiasteczkaIndexedDBLokalna pamięćPamięć podręcznaPamięć sesji

🔍

🔄

📄

🔍

Filtruj elementy

Key

Value

tasks

0

🔍

🔄

📄

🔍

Filtruj wartości

Dane

tasks: 0

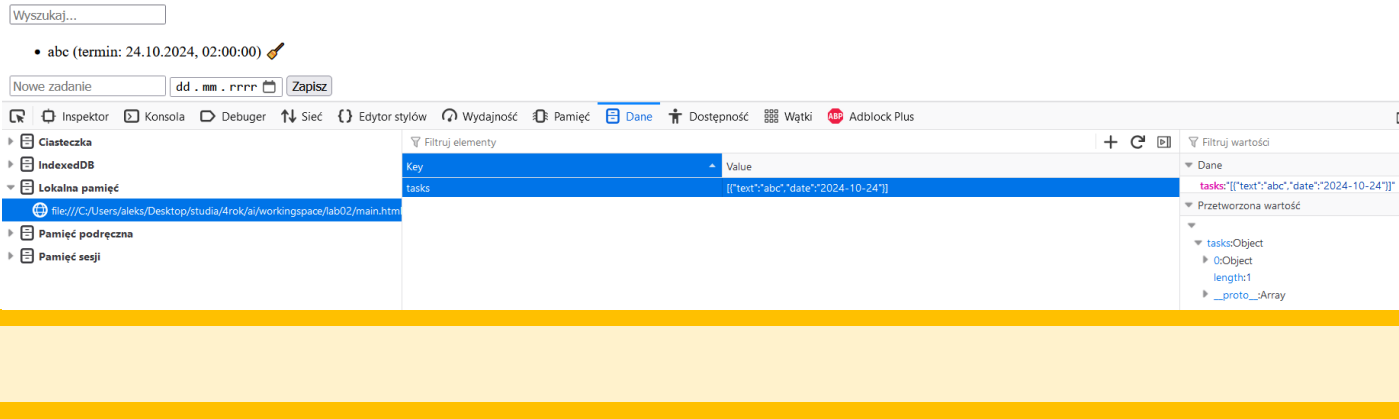
Przetworzona wart

tasks: Array

length: 0

\_\_proto\_\_: Array

Wstaw zrzuty ekranu przedstawiające wygląd listy i zawartość local storage po dodaniu nowej pozycji listy. Upewnij się, że widoczne w local storage są dane dotyczące nowego zadania:



Punkty:	0	1
---------	---	---

WYSZUKIWANIE

Na koniec zostało filtrowanie wyników. Proponowanym podejściem do tego tematu jest umieszczenie w klasie Todo właściwości `term` – frazy wyszukiwanej przez użytkownika. Następnie można utworzyć metodę `getFilteredTasks`, albo getter `filteredTasks`, która zwracać będzie te elementy tablicy `tasks`, które odpowiadają zapytaniu. Można użyć funkcji wyższego rzędu `filter()`.



Wstaw zrzut ekranu listy, gdy pole wyszukiwania jest puste:


### Lista zadań

- abc (termin: 24.10.2024, 02:00:00)
- def (termin: 30.10.2024, 01:00:00)
- adh

Wstaw zrzut ekranu listy, gdy w polu wyszukiwania wpisano wystarczająco dużo znaków, by zadziałało filtrowanie. Upewnij się, że chociaż 2 wyniki będą wciąż widoczne:

## Lista zadań

- abc (termin: 24.10.2024, 02:00:00) 
- adh 

Punkty:

0

1

Wstaw zrzut ekranu przedstawiający podświetlenie szukanej frazy w wynikach wyszukiwania, przykładowo dla frazy ko i zadania Ala ma kota otrzymujemy: Ala ma **kota**:

Punkty:

0

1

## COMMIT PROJEKTU DO GIT

Zacommituj i pushnij swoje rozwiązanie do repozytorium GIT.

Upewnij się, czy wszystko dobrze się wysłało. Jeśli tak, to z poziomu przeglądarki utwórz branch o nazwie lab-b na podstawie głównej gałęzi kodu.

Podaj link do brancha lab-b w swoim repozytorium:

**<https://github.com/Smokesiren/achastas/tree/main/lab-b>**

...link, np. <https://github.com/inazwisko/ai1-lab/tree/lab-b...>

## PODSUMOWANIE

W kilku zdaniach podsumuj zdobyte podczas tego laboratorium umiejętności.

...podsumowanie...



