

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	3
Usuwanie pozycji listy	4
Edycja pozycji listy.....	5
Odczyt / Zapis LocalStorage	6
Wyszukiwanie.....	7
Commit projektu do GIT.....	9
Podsumowanie.....	9

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 przycisku Usuń / Śmiertnik
- wpisy na liście zapisują się do Local Storage
- po odświeżeniu strony lista wypełnia się wpisami z Local Storage

Mockupy:

The left mockup shows a search bar at the top. Below it is a table-like structure with four rows of tasks. Each row contains a checkbox, a task name, a date (e.g., 2000-01-01), and a trash icon. At the bottom are three input fields: 'do zrobienia...', a date picker, and a 'Zapisz' button.

<input type="checkbox"/>	chocolate	2000-01-01
<input type="checkbox"/>	macaroon	
<input type="checkbox"/>	chupa chups	

The right mockup is similar but the 'chupa chups' row is highlighted with a yellow background, indicating it's selected for editing. The input fields at the bottom remain the same.

<input type="checkbox"/>	chocolate	2000-01-01
<input type="checkbox"/>	macaroon	
<input type="checkbox"/>	chupa chups	
<input type="checkbox"/>	candy canes	2000-01-05
<input type="checkbox"/>	bon bons	

This mockup shows a search bar at the top with the text 'on'. Below it is a table-like structure with two rows of tasks. The first row has a yellow background and contains a checked checkbox, the task name 'macaroon', and a trash icon. The second row contains an unchecked checkbox, the task name 'bon bons', and a trash icon. At the bottom are three input fields: 'do zrobienia...', a date picker, and a 'Zapisz' button.

<input checked="" type="checkbox"/>	macaroon	
<input type="checkbox"/>	bon bons	

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:

The screenshot shows a user interface for a task management application. At the top is a search bar with placeholder text "Szukaj zadań...". Below the search bar is a list of three tasks, each represented by a row with a title and a red "Usuń" (Delete) button. At the bottom of the list is a form with fields for "Wpisz nowe zadanie" (Enter new task) and "dd.mm.rrrr" (Date), followed by a green "Dodaj zadanie" (Add task) button. The bottom of the page features a standard browser navigation bar with icons for home, back, forward, and search.

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 wyczyszczy `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()`.

DODAWANIE POZYCJI LISTY

Wstaw zrzut ekranu listy przed dodaniem nowego zadania:

Szukaj zadań...

Wpisz nowe zadanie dd.mm.yyyy Dodaj zadanie

Wstaw zrzut ekranu listy po dodaniu nowego zadania:

Szukaj zadań...

Zrobić labA na AI Usuń

Zrobić labB na AI - 01.11.2024 Usuń

Zrobić labC na AI - 08.11.2024 Usuń

Wpisz nowe zadanie dd.mm.yyyy Dodaj zadanie

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

USUWANIE POZYCJI LISTY

Wstaw zrzut ekranu listy przed usunięciem wybranego zadania:

Szukaj zadań...

Zrobić labA na AI Usuń

Zrobić labB na AI - 01.11.2024 Usuń

Zrobić labC na AI - 08.11.2024 Usuń

Wpisz nowe zadanie dd.mm.yyyy Dodaj zadanie

Wstaw zrzut ekranu listy po usunięciu zadania:

Szukaj zadań...

Zrobić labB na AI - 01.11.2024 Usuń

Zrobić labC na AI - 08.11.2024 Usuń

Edytuj Dodaj zadanie

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

EDYCJA POZYCJI LISTY

Wstaw zrzut ekranu listy przed edycją wybranego zadania:

Szukaj zadań...

Zrobić labB na AI - 01.11.2024 Usuń

Zrobić labC na AI - 08.11.2024 Usuń

Edytuj Dodaj zadanie

Wstaw zrzut ekranu listy w trakcie edytowania zadania i daty:

Szukaj zadań...

Zrobić labB na AI - 01.11.2024 Usuń

Zrobić labC na AI Edytuj Usuń

Edytuj Dodaj zadanie

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

Szukaj zadań...

Zrobić labB na AI - 01.11.2024	<button>Usuń</button>
Zrobić kolejne laby na AI - 15.11.2024	<button>Usuń</button>

Wpisz nowe zadanie dd.mm.rrrr Dodaj zadanie

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:

Szukaj zadań...

Zrobić labB na AI - 01.11.2024	<button>Usuń</button>
Zrobić kolejne laby na AI - 08.11.2024	<button>Usuń</button>

Wpisz nowe zadanie dd.mm.rrrr Dodaj zadanie

▼ [{id: 1730466956836, text: "Zrobić labB na AI", date: "2024-11-01"},...]

▶ 0: {id: 1730466956836, text: "Zrobić labB na AI", date: "2024-11-01"}

▶ 1: {id: 1730466977357, text: "Zrobić kolejne laby na AI", date: "2024-11-08"}

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:

Szukaj zadań...

Zrobić labB na AI - 01.11.2024 Usuń

Zrobić kolejne laby na AI - 08.11.2024 Usuń

Wynieść kota Ali Usuń

Ala ma kota Usuń

DODAJ Dodaj zadanie

```
▼ [{"id": 1730466956836, "text": "Zrobić labB na AI", "date": "2024-11-01"}, ...]
▶ 0: {"id": 1730466956836, "text": "Zrobić labB na AI", "date": "2024-11-01"}
▶ 1: {"id": 1730466977357, "text": "Zrobić kolejne laby na AI", "date": "2024-11-08"}
▶ 2: {"id": 1730468134198, "text": "Wynieść kota Ali", "date": ""}
▶ 3: {"id": 1730468187715, "text": "Ala ma kota", "date": ""}
```

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:

Szukaj zadań...

Zrobić labB na AI - 01.11.2024

Usuń

Zrobić kolejne laby na AI - 08.11.2024

Usuń

Wynieść kota Ali

Usuń

Ala ma kota

Usuń

Wpisz nowe zadanie

dd.mm.yyyy



Dodaj zadanie

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

lab

Zrobić labB na AI - 01.11.2024

Usuń

Zrobić kolejne laby na AI - 08.11.2024

Usuń

Wpisz nowe zadanie

dd.mm.yyyy



Dodaj zadanie

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:

labB

Zrobić labB na AI - 01.11.2024

Usuń

Wpisz nowe zadanie

dd.mm.yyyy



Dodaj zadanie

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:

...link, <https://github.com/mm51621/main/tree/main/AI/labB>

PODSUMOWANIE

W kilku zdaniach podsumuj zdobyte podczas tego laboratorium umiejętności.

Poszerzyłem wiedzę związaną z HTML, CSS i JavaScript

Wykorzystanie local storage do przechowywania danych między sesjami

Obsługa zdarzeń i walidacja danych

Obsługa funkcjonalności takich jak dodawanie, edycja i usuwanie w JavaScript

...podsumowanie...

Zweryfikuj kompletność sprawozdania. Utwórz PDF i wyślij w terminie.