

JS I DOM NA PRZYKŁADZIE LISTY TODO

SPIS TREŚCI

| | |
|----------------------------------|----|
| Spis treści..... | 1 |
| Cel zajęć..... | 1 |
| Rozpoczęcie | 1 |
| Uwaga | 2 |
| Wymagania | 2 |
| Strona HTML..... | 2 |
| Klasa Todo..... | 3 |
| Dodawanie pozycji listy..... | 4 |
| Usuwanie pozycji listy..... | 6 |
| Edycja pozycji listy | 8 |
| Odczyt / Zapis LocalStorage..... | 10 |
| Wyszukiwanie | 13 |
| Commit projektu do GIT | 15 |
| Podsumowanie | 16 |

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 Usuń / 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 contains five items: chocolate (2000-01-01), macaroon, chupa chups, candy canes (2000-01-05), and bon bons. Each item has a checkbox on the left and a red 'X' icon on the right. Below the list is a form with a text input labeled 'do zrobienia...', a date input set to '2000-01-01', and a blue 'Zapisz' button.

Mockup of the application interface showing a task being edited. The list contains five items: chocolate (2000-01-01), macaroon, chupa chups, candy canes (2000-01-05), and bon bons. The 'chupa chups' item is selected, and its details are shown in a form below the list: a text input with 'chupa chups', a date input, and a blue 'Zapisz' button. The other items in the list have checkboxes and red 'X' icons.

Mockup of the application interface showing search results. The search bar at the top contains the text 'on'. The list below shows two items: 'macaroon' and 'bon bons', both highlighted in yellow. Each item has a checkbox on the left and a red 'X' icon on the right. Below the list is a form with a text input labeled 'do zrobienia...', a date input set to '2000-01-01', and a blue 'Zapisz' button.

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.

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

| ToDo List | | Wyszukaj |
|--|-------------------|----------|
| Dodaj do listy... | dd.mm.rrrr, --:-- | Add |
| <div> <div>pilka (2023-12-10)</div> <div>✎ ✕</div> </div> <div> <div>zakupy (2023-11-24)</div> <div>✎ ✕</div> </div> <div> <div>karate</div> <div>✎ ✕</div> </div> <div> <div>umyc auto (2023-11-25T19:20)</div> <div>✎ ✕</div> </div> <div> <div>lekarz (2023-11-10T15:16)</div> <div>✎ ✕</div> </div> <div> <div>market (2023-12-08T00:17)</div> <div>✎ ✕</div> </div> | | |

Punkty:

0

1

KLASA TODO

Pierwszym instynktem może być chęć dodania zachowań bezpośrednio do elementów listy. 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()`.

DODAWANIE POZYCJI LISTY

Wstaw zrzut ekranu listy przed dodaniem nowego zadania:

| ToDo List | | Wyszukaj |
|------------------------------|-------------------|----------|
| Dodaj do listy... | dd.mm.rrrr, --:-- | Add |
| <hr/> | | |
| pilka (2023-12-10) | | |
| zakupy (2023-11-24) | | |
| karate | | |
| umyc auto (2023-11-25T19:20) | | |
| lekarz (2023-11-10T15:16) | | |
| market (2023-12-08T00:17) | | |

Wstaw zrzut ekranu listy po dodaniu nowego zadania:

ToDo List

| | | |
|------------------------------|--|--|
| pilka (2023-12-10) | | |
| zakupy (2023-11-24) | | |
| karate | | |
| umyc auto (2023-11-25T19:20) | | |
| lekarz (2023-11-10T15:16) | | |
| market (2023-12-08T00:17) | | |
| siatkówka (2023-11-24T19:20) | | |

Punkty:















0

1

USUWANIE POZYCJI LISTY

Wstaw zrzut ekranu listy przed usunięciem wybranego zadania:

ToDo List

| | | |
|------------------------------|---|---|
| piłka (2023-12-10) |  |  |
| zakupy (2023-11-24) |  |  |
| karate |  |  |
| umyc auto (2023-11-25T19:20) |  |  |
| lekarz (2023-11-10T15:16) |  |  |
| market (2023-12-08T00:17) |  |  |
| siatkówka (2023-11-24T19:20) |  |  |

Wstaw zrzut ekranu listy po usunięciu zadania:

ToDo List

| | | |
|------------------------------|--|--|
| piłka (2023-12-10) | | |
| zakupy (2023-11-24) | | |
| karate | | |
| umyc auto (2023-11-25t19:20) | | |
| lekarz (2023-11-10t15:16) | | |
| siatkówka (2023-11-24t19:20) | | |

Punkty:













0

1

EDYCJA POZYCJI LISTY

Wstaw zrzut ekranu listy przed edycją wybranego zadania:

ToDo List



| | | |
|------------------------------|--|--|
| piłka (2023-12-10) |  |  |
| zakupy (2023-11-24) |  |  |
| karate |  |  |
| umyc auto (2023-11-25t19:20) |  |  |
| lekarz (2023-11-10t15:16) |  |  |
| siatkówka (2023-11-24t19:20) |  |  |

Wstaw zrzut ekranu listy w trakcie edytowania zadania i daty:


ToDo List






Add

pilka (2023-12-10)

 Save Cancel

karate

umyc auto (2023-11-25t19:20)

lekarz (2023-11-10t15:16)

siatkówka (2023-11-24t19:20)

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

ToDo List

Dodaj do listy...

dd.mm.rrrr, --:--

📅

Add

| | |
|------------------------------|-----|
| pilka (2023-12-10) | ✎ ✕ |
| boks - 2023-12-07 | ✎ ✕ |
| karate | ✎ ✕ |
| umyc auto (2023-11-25t19:20) | ✎ ✕ |
| lekarz (2023-11-10t15:16) | ✎ ✕ |
| siatkówka (2023-11-24t19:20) | ✎ ✕ |

| | | |
|---------|---|---|
| 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:

ToDo List

pilka (2023-12-10)



boks - 2023-12-07



karate



umyc auto (2023-11-25t19:20)



lekarz (2023-11-10t15:16)



savecancel



siatkówka (2023-11-24t19:20)



```
▼ [{name: "pilka", date: "2023-12-10"}, {name: "boks", date: "2023-12-07"}, {name: "karate", date: ""}, ...]  
  ▶ 0: {name: "pilka", date: "2023-12-10"}  
  ▶ 1: {name: "boks", date: "2023-12-07"}  
  ▶ 2: {name: "karate", date: ""}  
  ▶ 3: {name: "umyc auto", date: "2023-11-25T19:20"}  
  ▶ 4: {name: "lekarz", date: "2023-11-10T15:16"}  
  ▶ 5: {name: "market", date: "2023-12-08T00:17"}  
  ▶ 6: {name: "siatkówka", date: "2023-11-24T19:20"}
```

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:

ToDo List

Wyszukaj

Dodaj do listy...

dd.mm.rrrr, --:--

Add

| | | |
|------------------------------|--|--|
| piłka (2023-12-10) | | |
| boks (2023-12-07) | | |
| karate | | |
| umyc auto (2023-11-25T19:20) | | |
| lekarz (2023-11-10T15:16) | | |
| market (2023-12-08T00:17) | | |
| siatkówka (2023-11-24T19:20) | | |
| pływanie (2023-11-26T20:12) | | |

```
▼ [{name: "piłka", date: "2023-12-10"}, {name: "boks", date: "2023-12-07"}, {name: "karate", date: ""},...]  
  ▶ 0: {name: "piłka", date: "2023-12-10"}  
  ▶ 1: {name: "boks", date: "2023-12-07"}  
  ▶ 2: {name: "karate", date: ""}  
  ▶ 3: {name: "umyc auto", date: "2023-11-25T19:20"}  
  ▶ 4: {name: "lekarz", date: "2023-11-10T15:16"}  
  ▶ 5: {name: "market", date: "2023-12-08T00:17"}  
  ▶ 6: {name: "siatkówka", date: "2023-11-24T19:20"}  
  ▶ 7: {name: "pływanie", date: "2023-11-26T20:12"}
```

Punkty:

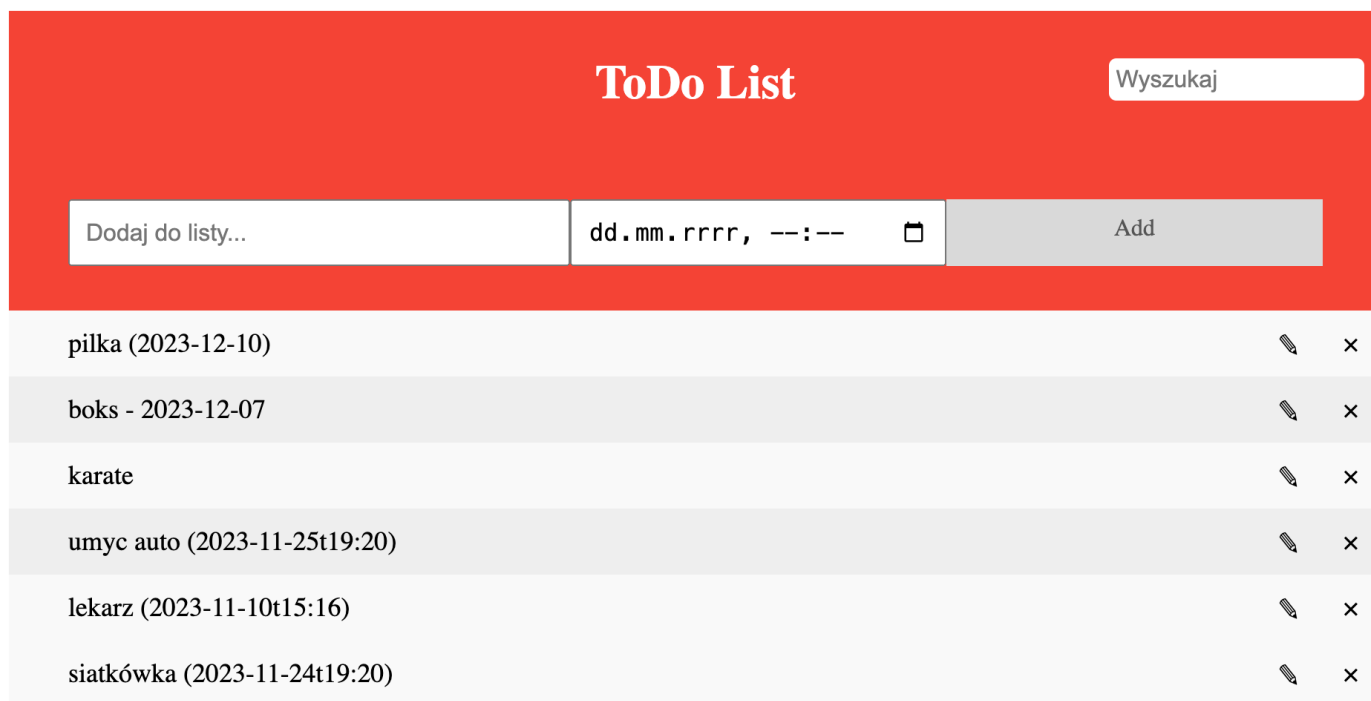
0

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:



| ToDo List | |
|--|---|
| <input type="text" value="Wyszukaj"/> | |
| <input type="text" value="Dodaj do listy..."/> | <input type="text" value="dd.mm.rrrr, --:--"/> <input type="button" value="Add"/> |
| piłka (2023-12-10) | |
| boks - 2023-12-07 | |
| karate | |
| umyc auto (2023-11-25t19:20) | |
| lekarz (2023-11-10t15:16) | |
| siatkówka (2023-11-24t19:20) | |

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

ToDo List

karate

lekarz (2023-11-10t15:16)

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

Wstaw zrzut ekranu przedstawiający podświetlenie szukanej frazy w wynikach wyszukiwania, przykładowo dla frazy `imp` i zadania `implementacja` otrzymujemy: `implementacja`:



The screenshot shows a web application titled "ToDo List". At the top right, there is a search bar containing the text "kar". Below the header, there is a white bar with a text input field labeled "Dodaj do listy...", a date/time picker showing "dd.mm.rrrr, --:--" with a calendar icon, and an "Add" button. Below this bar, there is a list of items. The first item is "karate" and the second item is "lekarz (2023-11-10t15:16)". Each item has a pencil icon for editing and a close icon for deleting.

| | | |
|---------|---|---|
| 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:

PODSUMOWANIE

W kilku zdaniach podsumuj zdobyte podczas tego laboratorium umiejętności.

Podczas wykonywania laboratorium poszerzyłem swoją wiedzę w zakresie javascriptu nauczyłem się wielu rzeczy związanych z tym językiem. Nową rzeczą dla mnie było wykonanie localStorage. Przez zadanie wiele się nauczyłem i z wieloma rzeczami się obyłem.

Zweryfikuj kompletność sprawozdania. Utwórz PDF i wyślij w terminie.