**Урок 1. Введение в React Hooks: useState**

import React, { useState } from "react";

import TodoList from "./TodoList";

export default function App() {

  /\* Локальный стейт \*/

  /\* useState(начальное состояние) todos = начальное состояние, setTodos будет менять начальное состояние\*/

  const [todos, setTodos] = useState([

    { id: 1, title: "First todo", completed: false },

    { id: 2, title: "Second todo", completed: true },

  ]);

  const [todoTitle, setTodoTitle] = useState("");

  const addTodo = (event) => {

    /\* Проверяем совершали ли мы нажатие на клавишу Enter \*/

    if (event.key === "Enter") {

      setTodos([

        ...todos,

        {

          /\* Функция Date.now() возвращает текущее время в миллисекундах, и id просто является идентификатором этого значения времени. \*/

          id: Date.now(),

          title: todoTitle,

          completed: false,

        },

      ]);

      setTodoTitle(""); /\* очищаем поле ввода \*/

    }

  };

  return (

    <div className="container">

      <h1>Todo app</h1>

      <div className="input-field">

        <input

          type="text"

          value={todoTitle}

          onChange={(event) => setTodoTitle(event.target.value)}

          onKeyDown={addTodo} /\* Чтобы при нажатии Enter добовлять todo \*/

        />

        <label>Todo name</label>

      </div>

      <TodoList todos={todos} />

    </div>

  );

}

@import "~materialize-css/dist/css/materialize.min.css";

body {

  margin: 0;

  padding: 0;

}

.todo {

  margin-bottom: 1rem;

}

.todo.completed span {

  text-decoration: line-through;

}

.todo label {

  width: 100%;

  display: flex;

  align-items: center;

  justify-content: space-between;

  cursor: pointer;

  background: #f3f3f3;

  padding: 1rem;

}

.todo i {

  z-index: 2;

}

import React from 'react'

import TodoItem from './TodoItem'

export default function TodoList({todos}) {

  return (

    <ul>

      {todos.map(item => <TodoItem key={item.id} {...item} />)}

    </ul>

  )

}

import React, { useState } from "react";

export default function TodoItem({ title, id, completed }) {

  /\* Чтобы при нажатии чекбокса текст зачеркивался \*/

  const [checked, setChecked] = useState(completed);

  const cls = ["todo"];

  if (checked) {

    cls.push("completed");

  }

  return (

    <li className={cls.join(" ")}>

      {/\* соединили todo и completed \*/}

      <label>

        <input

          type="checkbox"

          checked={checked}

          onChange={() => setChecked(!checked)}

        />

        <span>{title}</span>

        <i className="material-icons red-text">delete</i>

      </label>

    </li>

  );

}

**LocalStorageУрок 2. Введение в React Hooks: useEffect**

const [todos, setTodos] = useState([]);

Очистили локальный стейт, пробелема в том что, при перезагрузке страницы стейт теряется и не сохраняется

import React, { useState, useEffect } from "react";

import TodoList from "./TodoList";

export default function App() {

  /\* Локальный стейт \*/

  /\* useState(начальное состояние) todos = начальное состояние, setTodos будет менять начальное состояние\*/

  const [todos, setTodos] = useState([]);

  const [todoTitle, setTodoTitle] = useState("");

  const addTodo = (event) => {

    /\* Проверяем совершали ли мы нажатие на клавишу Enter \*/

    if (event.key === "Enter") {

      setTodos([

        ...todos,

        {

          /\* Функция Date.now() возвращает текущее время в миллисекундах, и id просто является идентификатором этого значения времени. \*/

          id: Date.now(),

          title: todoTitle,

          completed: false,

        },

      ]);

      setTodoTitle(""); /\* очищаем поле ввода \*/

    }

  };

  /\*

Добавляется несколько кликов сразу remove чтобы убрать

const handleClick = () => console.log("click"); \*/

  /\* Будет вызываться при старте данного компонента

  Забиоаем todo которые сохранены локально \*/

  useEffect(() => {

    const raw = localStorage.getItem("todos") || [];

    setTodos(JSON.parse(raw));

  }, []);

  /\* Все что вносим в todo сохраняем в localStorage \*/

  useEffect(() => {

    document.addEventListener("click", handleClick);

    localStorage.setItem("todos", JSON.stringify(todos));

    /\*     return () => {

      document.removeEventListener("click", handleClick);

    }; \*/

  }, [todos]);

  return (

    <div className="container">

      <h1>Todo app</h1>

      <div className="input-field">

        <input

          type="text"

          value={todoTitle}

          onChange={(event) => setTodoTitle(event.target.value)}

          onKeyDown={addTodo} /\* Чтобы при нажатии Enter добовлять todo \*/

        />

        <label>Todo name</label>

      </div>

      <TodoList todos={todos} />

    </div>

  );

}

**Урок 3. Введение в React Hooks: useContext**

Сейчас в компоненте TodoItem мы используем локальный стейт и когда жмём чекбокс то он не сохраняет изменения чекбокса

Создали файл context.js

import React from "react";

export const Context = React.createContext();

import React, { useState, useEffect } from "react";

import TodoList from "./TodoList";

import { Context } from "./context"; /\* Вызываем контекст \*/

export default function App() {

  /\* Локальный стейт \*/

  /\* useState(начальное состояние) todos = начальное состояние, setTodos будет менять начальное состояние\*/

  const [todos, setTodos] = useState([]);

  const [todoTitle, setTodoTitle] = useState("");

  const addTodo = (event) => {

    /\* Проверяем совершали ли мы нажатие на клавишу Enter \*/

    if (event.key === "Enter") {

      setTodos([

        ...todos,

        {

          /\* Функция Date.now() возвращает текущее время в миллисекундах, и id просто является идентификатором этого значения времени. \*/

          id: Date.now(),

          title: todoTitle,

          completed: false,

        },

      ]);

      setTodoTitle(""); /\* очищаем поле ввода \*/

    }

  };

  /\*

Добавляется несколько кликов сразу remove чтобы убрать

const handleClick = () => console.log("click"); \*/

  /\* Будет вызываться при старте данного компонента

  Забиоаем todo которые сохранены локально \*/

  useEffect(() => {

    const raw = localStorage.getItem("todos") || [];

    setTodos(JSON.parse(raw));

  }, []);

  /\* Все что вносим в todo сохраняем в localStorage \*/

  useEffect(() => {

    /\*  document.addEventListener("click", handleClick); \*/

    localStorage.setItem("todos", JSON.stringify(todos));

    /\*     return () => {

      document.removeEventListener("click", handleClick);

    }; \*/

  }, [todos]);

  /\* Удаление todo \*/

  const removeTodo = (id) => {

    setTodos(

      todos.filter((todo) => {

        return todo.id !== id;

      })

    );

  };

  /\* Делаем todo зачёркнутым \*/

  const toogleTodo = (id) => {

    setTodos(

      todos.map((todo) => {

        if (todo.id === id) {

          todo.completed = !todo.completed;

        }

        return todo;

      })

    );

  };

  return (

    /\* Передаем компонентам контекст \*/

    <Context.Provider value={{ toogleTodo, removeTodo }}>

      <div className="container">

        <h1>Todo app</h1>

        <div className="input-field">

          <input

            type="text"

            value={todoTitle}

            onChange={(event) => setTodoTitle(event.target.value)}

            onKeyDown={addTodo} /\* Чтобы при нажатии Enter добовлять todo \*/

          />

          <label>Todo name</label>

        </div>

        <TodoList todos={todos} />

      </div>

    </Context.Provider>

  );

}

Теперь удаляем локальный state из TodoItem и берем его из Контекста

import React, { useState, useContext } from "react";

import { Context } from "./context"; /\* Вызываем контекст \*/

export default function TodoItem({ title, id, completed }) {

  const cls = ["todo"];

  if (completed) {

    cls.push("completed");

  }

  /\* Вызываем контекст от которого определили и функции которые передвали \*/

  const { toogleTodo, removeTodo } = useContext(Context);

  return (

    <li className={cls.join(" ")}>

      {/\* соединили todo и completed \*/}

      <label>

        <input

          type="checkbox"

          checked={completed}

          onChange={() => toogleTodo(id)}

        />

        <span>{title}</span>

        <i className="material-icons red-text" onClick={() => removeTodo(id)}>

          delete

        </i>

      </label>

    </li>

  );

}

**Урок 4. Введение в React Hooks: useReducer**

Сейчас проблема что все методы которые меняют стейт они хранятся в компоненте

И это поддердживать сложно и для этого используем хук useReducer

Который позволяет вынести логику в отдельный файл (работает как Redux)

Создаем файл reducer.js где будет храниться и обрабатываться логика

import React, { useState, useEffect, useReducer } from "react";

import TodoList from "./TodoList";

import { Context } from "./context"; /\* Вызываем контекст \*/

import reducer from "./reducer";

export default function App() {

  /\* Есть state и  dispatch который будет менять это состояние\*/

  const [state, dispatch] = useReducer(

    reducer,

    JSON.parse(localStorage.getItem("todos"))

  );

  /\* Локальный стейт \*/

  /\* useState(начальное состояние) todos = начальное состояние, setTodos будет менять начальное состояние\*/

  const [todos, setTodos] = useState([]);

  const [todoTitle, setTodoTitle] = useState("");

  const addTodo = (event) => {

    /\* Проверяем совершали ли мы нажатие на клавишу Enter \*/

    if (event.key === "Enter") {

      dispatch({

        type: "add" /\* Параметр по которому буде менять state \*/,

        payload: todoTitle /\* который содержит состояние todoTitle \*/,

      });

      setTodoTitle(""); /\* очищаем поле ввода \*/

    }

  };

  /\*

Добавляется несколько кликов сразу remove чтобы убрать

const handleClick = () => console.log("click"); \*/

  /\*

  ИЗБАВИЛИСЬ ТАК КАК ПЕРЕДАЕМ ЕГО В REDUCER

  Будет вызываться при старте данного компонента

  Забиоаем todo которые сохранены локально \*/

  /\*   useEffect(() => {

    const raw = localStorage.getItem("todos") || [];

    setTodos(JSON.parse(raw));

  }, []);

 \*/

  /\* Все что вносим в todo сохраняем в localStorage \*/

  useEffect(() => {

    /\*  document.addEventListener("click", handleClick); \*/

    localStorage.setItem("todos", JSON.stringify(state));

    /\*     return () => {

      document.removeEventListener("click", handleClick);

    }; \*/

  }, [state]);

  /\* Логика будет в Reducer

Удаление todo \*/

  /\*   const removeTodo = (id) => {

    setTodos(

      state.filter((todo) => {

        return todo.id !== id;

      })

    );

  }; \*/

  /\* Делаем todo зачёркнутым \*/

  /\*   const toogleTodo = (id) => {

    setTodos(

      todos.map((todo) => {

        if (todo.id === id) {

          todo.completed = !todo.completed;

        }

        return todo;

      })

    );

  }; \*/

  return (

    /\* Передаем компонентам контекст \*/

    <Context.Provider

      value={{

        /\* toogleTodo, removeTodo \*/

        dispatch,

      }}

    >

      <div className="container">

        <h1>Todo app</h1>

        <div className="input-field">

          <input

            type="text"

            value={todoTitle}

            onChange={(event) => setTodoTitle(event.target.value)}

            onKeyDown={addTodo} /\* Чтобы при нажатии Enter добовлять todo \*/

          />

          <label>Todo name</label>

        </div>

        <TodoList todos={state} />

      </div>

    </Context.Provider>

  );

}

import React, { useContext } from "react";

import { Context } from "./context"; /\* Вызываем контекст \*/

export default function TodoItem({ title, id, completed }) {

  const cls = ["todo"];

  if (completed) {

    cls.push("completed");

  }

  /\* Вызываем контекст от которого определили и функции которые передвали \*/

  const { dispatch } = useContext(Context);

  return (

    <li className={cls.join(" ")}>

      {/\* соединили todo и completed \*/}

      <label>

        <input

          type="checkbox"

          checked={completed}

          onChange={() => dispatch({ type: "toggle", payload: id })}

        />

        <span>{title}</span>

        <i

          className="material-icons red-text"

          onClick={() => dispatch({ type: "remove", payload: id })}

        >

          delete

        </i>

      </label>

    </li>

  );

}

export default function (state, action) {

  switch (action.type) {

    case "add":

      return [

        ...state,

        {

          id: Date.now(),

          title: action.payload /\* Который передали из APP \*/,

          completed: false,

        },

      ];

    case "toggle":

      return state.map((todo) => {

        if (todo.id === action.payload) {

          todo.completed = !todo.completed;

        }

        return todo;

      });

    case "remove":

      return state.filter((todo) => todo.id !== action.payload);

    default:

      return state;

  }

}