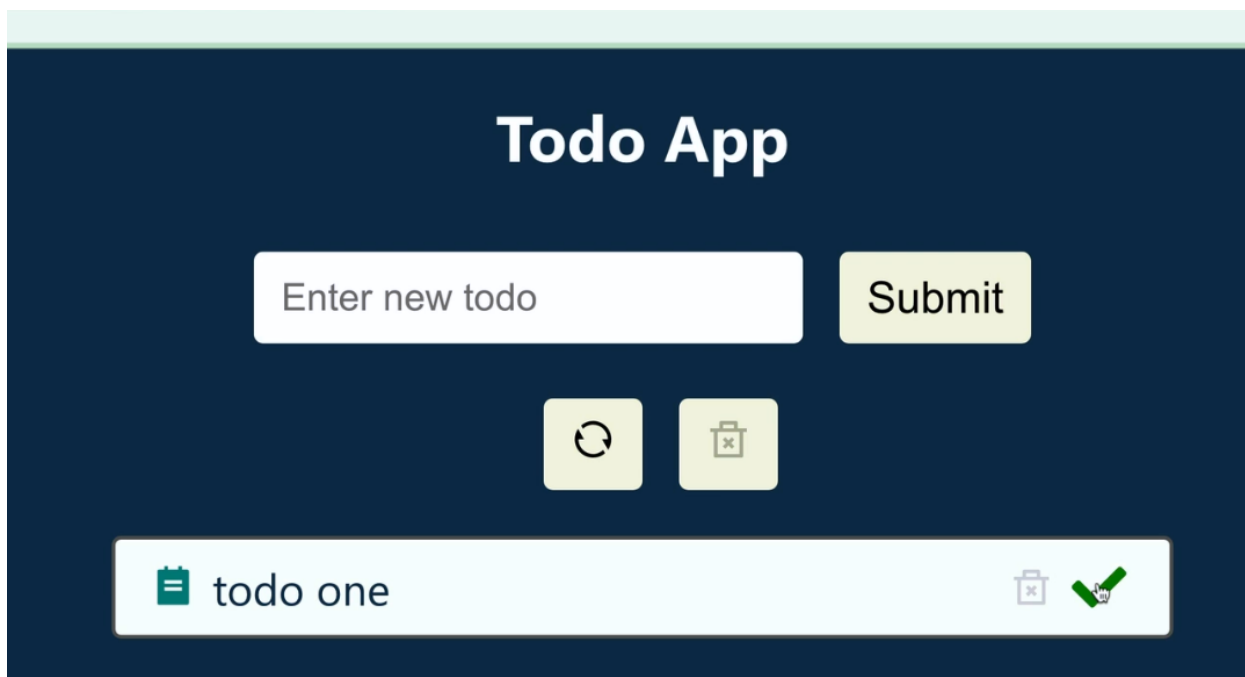


Проект Todo App v2



Изменение структуры данных на массив объектов

Структура массива для хранения задач в данном случае не подходит, т.к. теперь рядом с текстом задачи будут находиться поля `isCompleted` и `id`. `isCompleted` будет хранить информацию (`true/false`) о выполнении задачи, а `id` уникальный номер задачи. Поэтому организуем объект для хранения этих свойств.

Для `id` уникальность зададим с помощью внешнего npm-пакета, который и обеспечит уникальность

<https://www.npmjs.com/package/uuid?activeTab=readme>

(!!!Предварительно остановить сервер)

```
npm i uuid
```

Запустим сервер и подключим библиотеку

```
import { v4 as uuidv4 } from 'uuid';
```

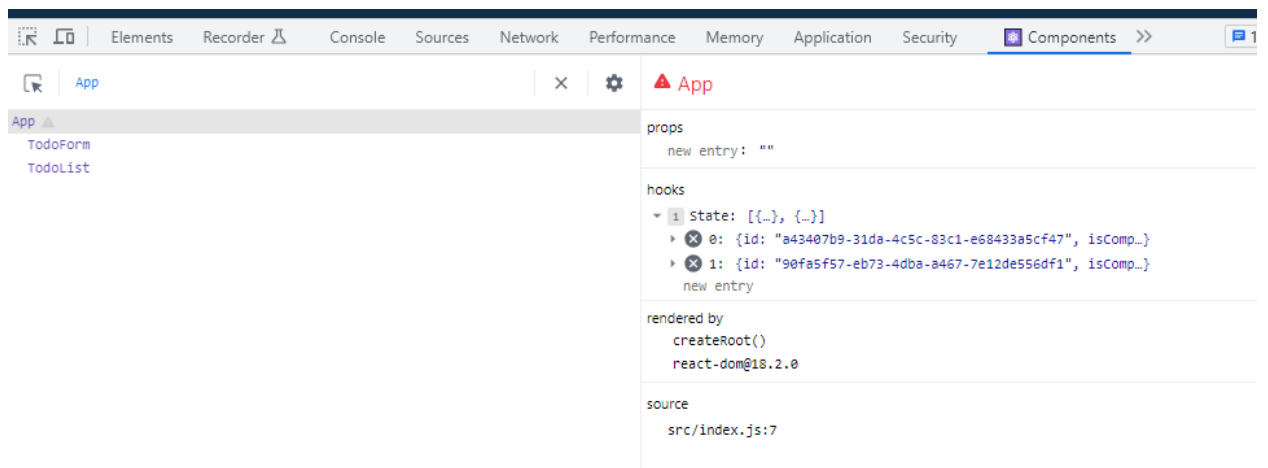
Поскольку у нас появился объект с идентификатор `id`, то предыдущий идентификатор `index` необходимо заменить во всех компонентах проекта

```
JS App.js U X JS TodoList.js U JS Todo.js U
25-todo-app-v2 > src > JS App.js > App > addTodoHandler > newTodo > id
6
7 function App() {
8   const [todos, setTodos] = useState([]);
9
10  const addTodoHandler = (text) => {
11    const newTodo = {
12      text: text,
13      isCompleted: false,
14      id: uuidv4(),
15    };
16
17    setTodos([...todos, newTodo]);
18  };
19
20  const deleteTodoHandler = (id) => {
21    setTodos(todos.filter((todo) => todo.id !== id));
22  };
23
24  return (
25    <div className="App">
26      <h1>Todo App</h1>
27      <TodoForm addTodo={addTodoHandler} />
28      <TodoList todos={todos} deleteTodo={deleteTodoHandler} />
29    </div>
30  );
31 }
32
33 export default App;
```

```
JS App.js U JS TodoList.js U JS Todo.js U X
25-todo-app-v2 > src > components > Todos > JS Todo.js > ...
1 import { RiTodoFill } from 'react-icons/ri';
2 import styles from './Todo.module.css';
3
4 function Todo({ todo, deleteTodo }) {
5   return (
6     <div className={styles.todo} onDoubleClick={() => deleteTodo(todo.id)}>
7       <RiTodoFill className={styles.todoIcon} />
8       <div className={styles.todoText}>{todo.text}</div>
9     </div>
10  );
11 }
12
13 export default Todo;
14
```

```
JS App.js U JS TodoList.js U X JS Todo.js U
25-todo-app-v2 > src > components > Todos > JS TodoList.js > TodoList > todos.map() callba
1  import Todo from './Todo';
2  import styles from './TodoList.module.css';
3
4  function TodoList({ todos, deleteTodo }) {
5    return (
6      <div className={styles.todoListContainer}>
7        {!todos.length && <h2>Todo list is empty</h2>}
8        {todos.map((todo) => (
9          <Todo key={todo.id} todo={todo} deleteTodo={deleteTodo} />
10       ))}
11      </div>
12    );
13  }
14
15  export default TodoList;
16
```

Так же обратим внимание на состояние компонента App.js на вкладке Components



Добавление и стилизация кнопок удаления и завершения задач Todo.module

```
JS App.js U    JS TodoList.js U    JS Todo.js U X    Todo.module.css U
25-todo-app-v2 > src > components > Todos > JS Todo.js > Todo
1  import { RiTodoFill } from 'react-icons/ri';
2  import { RiDeleteBin2Line } from 'react-icons/ri';
3  import { FaCheck } from 'react-icons/fa';
4  import styles from './Todo.module.css';
5
6  function Todo({ todo, deleteTodo }) {
7    return (
8      <div className={styles.todo} onDoubleClick={() => deleteTodo(todo.id)}>
9        <RiTodoFill className={styles.todoIcon} />
10       <div className={styles.todoText}>{todo.text}</div>
11       <RiDeleteBin2Line className={styles.deleteIcon} />
12       <FaCheck className={styles.checkIcon} />
13     </div>
14   );
15 }
16
17 export default Todo;
```

```

1  .todo {
2      display: flex;
3      align-items: center;
4      padding: 10px 20px;
5      margin: 15px 0;
6      font-size: 1.5rem;
7      border-radius: 5px;
8      border: 2px solid #555;
9      color: #112d49;
10     background-color: #fbfef9;
11 }
12
13 .todoText {
14     width: 100%;
15     text-align: left;
16 }
17
18 .todoIcon {
19     font-size: 1.8rem;
20     margin-right: 10px;
21     color: teal;
22 }
23
24 .completedTodo {
25     background-color: unset;
26     border-color: gray;
27     color: gray;
28 }
29
30 .todo.completedTodo .todoIcon,
31 .todo.completedTodo .checkIcon,
32 .todo.completedTodo .deleteIcon {
33     color: gray;
34 }
35
36 .checkIcon,
37 .deleteIcon {
38     cursor: pointer;
39     color: lightgrey;
40     padding: 0 7px;
41     font-size: 1.8rem;
42     transition: transform 0.3s;
43 }
44
45 .checkIcon:hover,
46 .deleteIcon:hover {
47     cursor: pointer;
48     transform: scale(1.3);
49 }
50
51 .checkIcon:hover {
52     color: green;
53 }
54
55 .deleteIcon:hover {
56     color: red;
57 }
58

```

Функционал удаления отдельных задач

Т.к. функционал удаления задачи у нас уже реализован, то скопируем эту функцию и добавим ее через свойство `onClick` в компонент `RiDeleteBin2Line`, а из общего `<div>` удалим свойство `onDubleClick`

```
JS Todo.js U X JS App.js U JS TodoList.js U Todo.module.css U
25-todo-app-v2 > src > components > Todos > JS Todo.js > [e] default
1  import { RiTodoFill } from 'react-icons/ri';
2  import { RiDeleteBin2Line } from 'react-icons/ri';
3  import { FaCheck } from 'react-icons/fa';
4  import styles from './Todo.module.css';
5
6  function Todo({ todo, deleteTodo }) {
7    return (
8      <div className={styles.todo}>
9        <RiTodoFill className={styles.todoIcon} />
10       <div className={styles.todoText}>{todo.text}</div>
11       <RiDeleteBin2Line
12         className={styles.deleteIcon}
13         onClick={() => deleteTodo(todo.id)}
14       />
15       <FaCheck className={styles.checkIcon} />
16     </div>
17   );
18 }
19
20 export default Todo;
21
```

Функционал завершения отдельных задач

Здесь нам необходимо свойство `isCompleted` менять с `false` на `true` или обратно. Это называется `toggle` (переключатель) задачи. Для этого создадим новую функцию `toggleTodoHendler` в компоненте `App.js`. В дальнейшем эта функция будет передана в компонент `TodoList`, а он передаст ее в компонент `List`

JS App.js M X

JS TodoList.js M

JS Todo.js M

25-todo-app-v2 > src > JS App.js > default

```
1  import { useState } from 'react';
2  import { v4 as uuidv4 } from 'uuid';
3  import TodoForm from '../components/Todos/TodoForm';
4  import TodoList from '../components/Todos/TodoList';
5  import './App.css';
6
7  function App() {
8    const [todos, setTodos] = useState([]);
9
10   const addTodoHandler = (text) => {
11     const newTodo = {
12       text: text,
13       isCompleted: false,
14       id: uuidv4(),
15     };
16
17     setTodos([...todos, newTodo]);
18   };
19
20   const deleteTodoHandler = (id) => {
21     setTodos(todos.filter((todo) => todo.id !== id));
22   };
23
24   const toggleTodoHandler = (id) => {
25     setTodos(
26       todos.map((todo) =>
27         todo.id === id
28         ? { ...todo, isCompleted: !todo.isCompleted }
29         : { ...todo }
30       )
31     );
32   };
33
34   return (
35     <div className="App">
36       <h1>Todo App</h1>
37       <TodoForm addTodo={addTodoHandler} />
38       <TodoList
39         todos={todos}
40         deleteTodo={deleteTodoHandler}
41         toggleTodo={toggleTodoHandler}
42       />
43     </div>
44   );
45 }
46
47 export default App;
```

48

JS App.js M


JS TodoList.js M X

JS Todo.js M

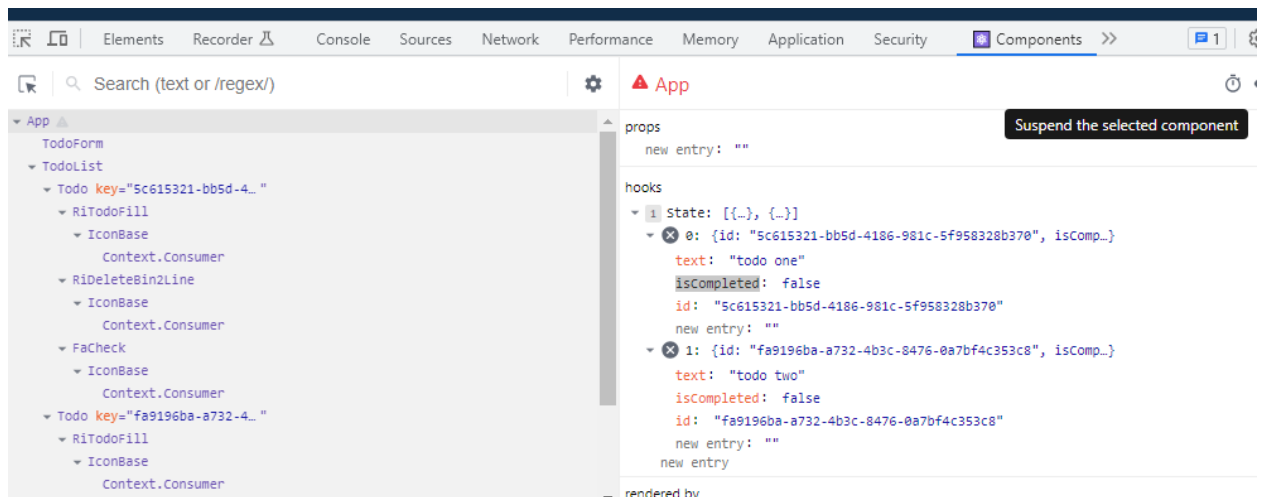
25-todo-app-v2 > src > components > Todos > JS TodoList.js > TodoList > todos.r

```
1  import Todo from './Todo';
2  import styles from './TodoList.module.css';
3
4  function TodoList({ todos, deleteTodo, toggleTodo }) {
5    return (
6      <div className={styles.todoListContainer}>
7        {!todos.length && <h2>Todo list is empty</h2>}
8        {todos.map((todo) => (
9          <Todo
10             key={todo.id}
11             todo={todo}
12             deleteTodo={deleteTodo}
13             toggleTodo={toggleTodo}
14           />
15        ))}
16      </div>
17    );
18  }
19
20  export default TodoList;
```

```
JS App.js M JS TodoList.js M JS Todo.js M X
25-todo-app-v2 > src > components > Todos > JS Todo.js > Todo
1 import { RiTodoFill } from 'react-icons/ri';
2 import { RiDeleteBin2Line } from 'react-icons/ri';
3 import { FaCheck } from 'react-icons/fa';
4 import styles from './Todo.module.css';
5
6 function Todo({ todo, deleteTodo, toggleTodo }) {
7   return (
8     <div className={styles.todo}>
9       <RiTodoFill className={styles.todoIcon} />
10      <div className={styles.todoText}>{todo.text}</div>
11      <RiDeleteBin2Line
12        className={styles.deleteIcon}
13        onClick={() => deleteTodo(todo.id)}
14      />
15      <FaCheck
16        className={styles.checkIcon}
17        onClick={() => toggleTodo(todo.id)}
18      />
19    </div>
20  );
21 }
22
23 export default Todo;
24
```

В видимой части приложения ни чего не происходит при нажатии на , но в консоли на вкладке Components мы видим изменения isCompleted при

нажатии на 



Теперь необходимо поменять поведение в интерфейсе.

Условное добавление классов CSS

Для этого необходимо добавить новый класс `.completedTodo` для тех задач, у которых `isCompleted` имеет значение `true`. Сделаем это в родительском `<div>` компонента `Todo.js` через тернарный оператор

```

JS App.js M    JS TodoList.js M    JS Todo.js M X
25-todo-app-v2 > src > components > Todos > JS Todo.js > Todo
1  import { RiTodoFill } from 'react-icons/ri';
2  import { RiDeleteBin2Line } from 'react-icons/ri';
3  import { FaCheck } from 'react-icons/fa';
4  import styles from './Todo.module.css';
5
6  function Todo({ todo, deleteTodo, toggleTodo }) {
7    return (
8      <div
9        className={` ${styles.todo} ${
10      todo.isCompleted ? styles.completedTodo : ''
11        }` >
12      <RiTodoFill className={styles.todoIcon} />
13      <div className={styles.todoText}>{todo.text}</div>
14      <RiDeleteBin2Line
15        className={styles.deleteIcon}
16        onClick={() => deleteTodo(todo.id)}
17      />
18      <FaCheck
19        className={styles.checkIcon}
20        onClick={() => toggleTodo(todo.id)}
21      />
22    </div>
23  );
24  }
25
26
27  export default Todo;
28

```

Todo App

📅 todo one

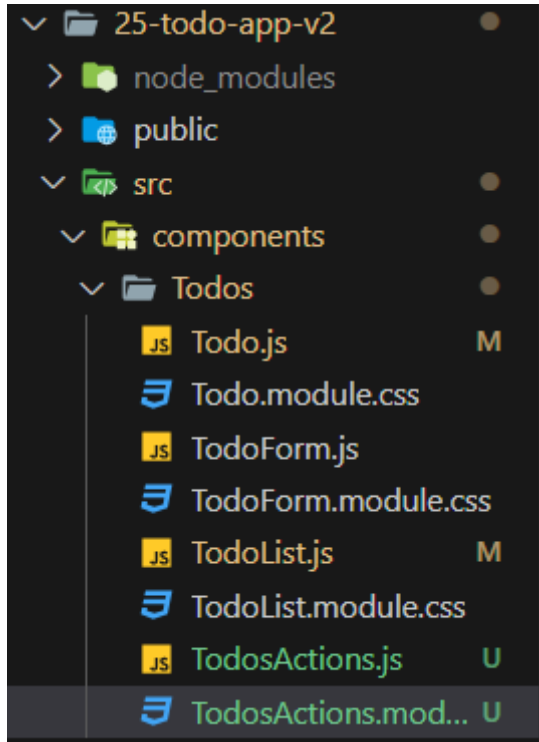


📅 todo two



Добавление блока с кнопками для всех задач

Для этого создадим новый компонент TodosActions, содержащий действия для всех задач в нашем приложении, а также добавим файл TodosActions.module.css



В компоненте TodosActions пока выполним разметку без какого-либо функционала и подключим этот компонент

```
JS TodosActions.js U X JS App.js M
25-todo-app-v2 > src > components > Todos > JS TodosActions.js > ...
1  function TodosActions() {
2    return (
3      <>
4        <button>Reset</button>
5        <button>Delete Completed</button>
6      </>
7    );
8  }
9
10 export default TodosActions;
11
```

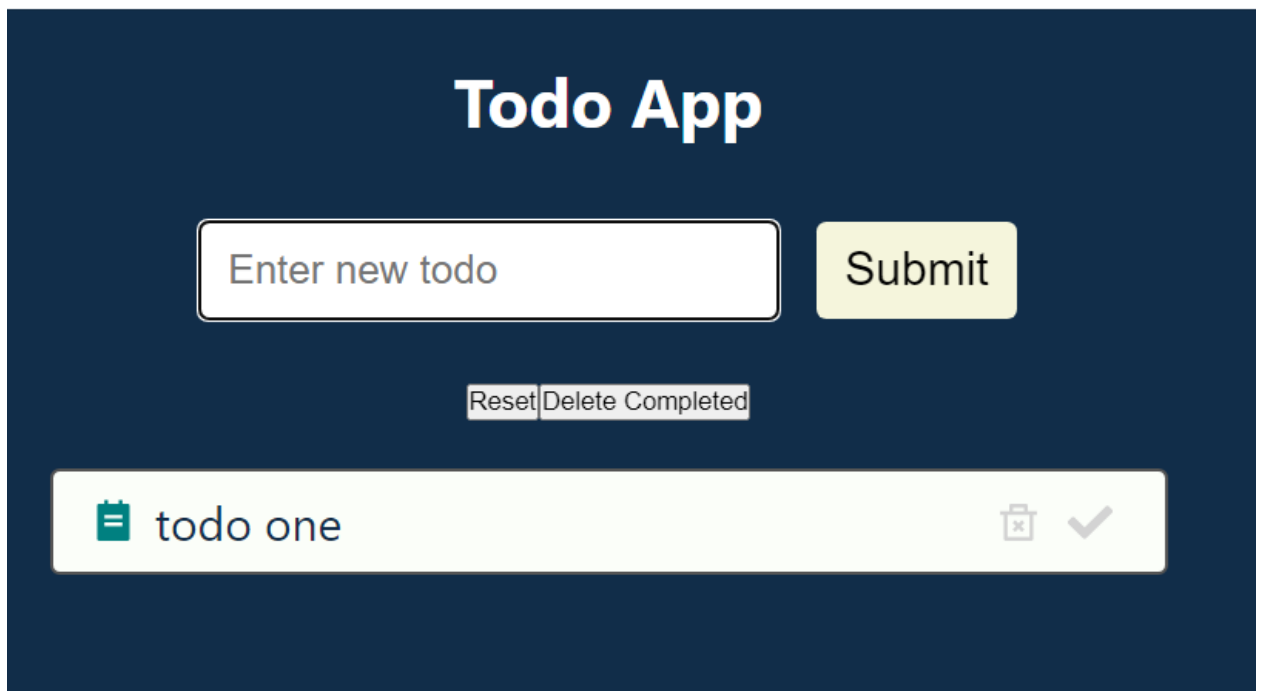
JS TodosActions.js U

JS App.js M X



25-todo-app-v2 > src > JS App.js > App

```
1  import { useState } from 'react';
2  import { v4 as uuidv4 } from 'uuid';
3  import TodoForm from '../components/Todos/TodoForm';
4  import TodoList from '../components/Todos/TodoList';
5  import TodosActions from '../components/Todos/TodosActions';
6  import './App.css';
7
8  function App() {
9    const [todos, setTodos] = useState([]);
10
11    const addTodoHandler = (text) => {
12      const newTodo = {
13        text: text,
14        isCompleted: false,
15        id: uuidv4(),
16      };
17
18      setTodos([...todos, newTodo]);
19    };
20
21    const deleteTodoHandler = (id) => {
22      setTodos(todos.filter((todo) => todo.id !== id));
23    };
24
25    const toggleTodoHandler = (id) => {
26      setTodos(
27        todos.map((todo) =>
28          todo.id === id
29            ? { ...todo, isCompleted: !todo.isCompleted }
30            : { ...todo }
31        )
32      );
33    };
34
35    return (
36      <div className="App">
37        <h1>Todo App</h1>
38        <TodoForm addTodo={addTodoHandler} />
39        <TodosActions />
40        <TodoList
41          todos={todos}
42          deleteTodo={deleteTodoHandler}
43          toggleTodo={toggleTodoHandler}
44        />
45      </div>
46    );
47  }
48
49  export default App;
```



Создание компонента Button

Создадим универсальный компонент Button, который можно использовать в любой части нашего приложения. Как правило кнопка содержит обработчик `onClick`, `children`, `title`, `disabled = false`. Children отвечает за то, что мы помещаем в компонент Button: текст, иконку. Title — обеспечивает отображение подсказки при наведении мыши. `disabled = false` — позволяет отключать какую-либо кнопку.

Внешний вид кнопок будет одинаковым.

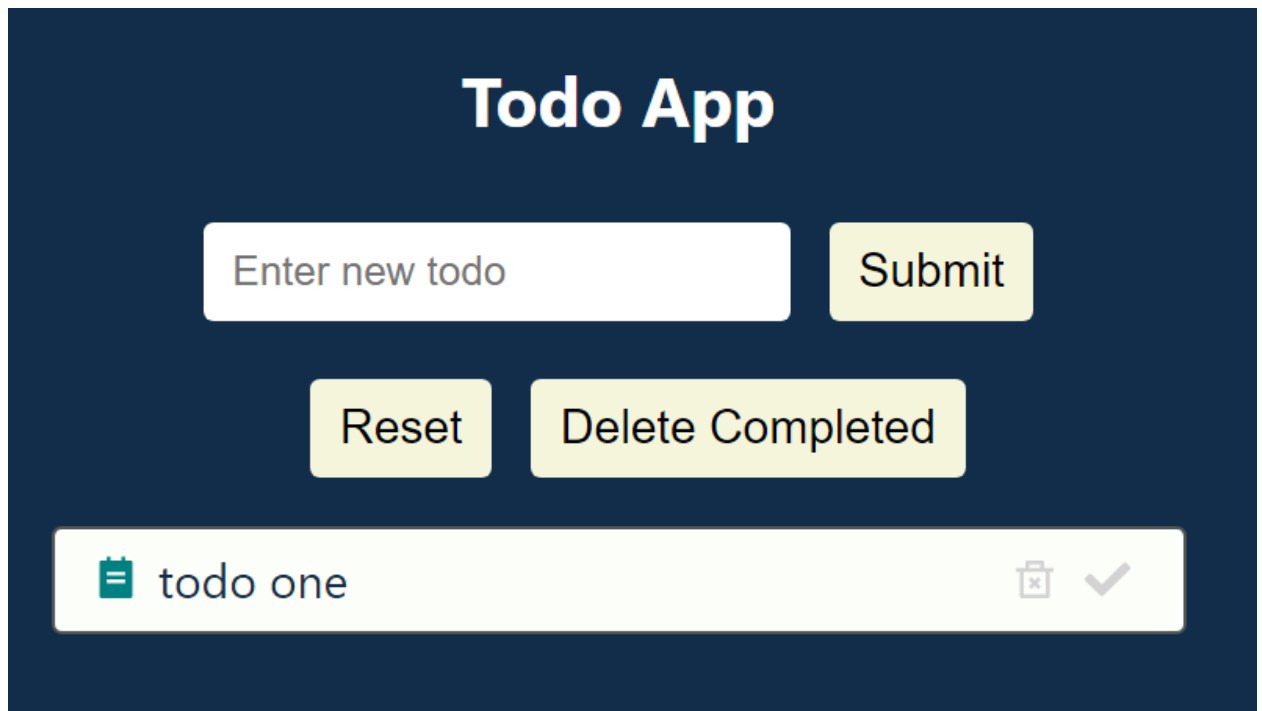
```
JS Button.js M X Button.module.css U
25-todo-app-v2 > src > components > UI > JS Button.js > ...
1  import styles from './Button.module.css';
2
3  function Button({ onClick, children, title, disabled = false }) {
4    return (
5      <button
6        className={styles.button}
7        onClick={onClick}
8        title={title}
9        disabled={disabled}
10     >
11       {children}
12     </button>
13   );
14 }
15
16 export default Button;
17
```


Добавим стили для этого компонента

```
Button.module.css U X JS Button.js M
25-todo-app-v2 > src > components > UI > Button.module.css > .button
1  .button {
2    margin-left: 20px;
3    height: 50px;
4    cursor: pointer;
5    background-color: beige;
6    font-size: 1.5rem;
7    padding: 10px 15px;
8    border: none;
9    border-radius: 5px;
10 }
11
12 .button:hover {
13   background-color: rgb(240, 240, 155);
14 }
15
```

Компонент готов и его можно использовать в компонентах TodoForm и TodoActions

```
JS TodosActions.js U X Button.module.css U JS Button.js M
25-todo-app-v2 > src > components > Todos > TodosActions.js > ...
1  import Button from '../UI/Button';
2  function TodosActions() {
3    return (
4      <>
5        <Button>Reset</Button>
6        <Button>Delete Completed</Button>
7      </>
8    );
9  }
10
11  export default TodosActions;
12
```




Поменяем на иконки текст на кнопках и добавим подсказки на кнопки

```
JS TodosActions.js U X Button.module.css U JS Button.js M
25-todo-app-v2 > src > components > Todos > JS TodosActions.js > ...
1  import { RiDeleteBin2Line, RiRefreshLine } from 'react-icons/ri';
2  import Button from '../UI/Button';
3
4  function TodosActions() {
5    return (
6      <>
7        <Button title="Reset Todos">
8          <RiRefreshLine />
9        </Button>
10       <Button title="Clear Completed todos">
11         <RiDeleteBin2Line />
12       </Button>
13     </>
14   );
15 }
16
17 export default TodosActions;
18
```

Todo App



 todo one

