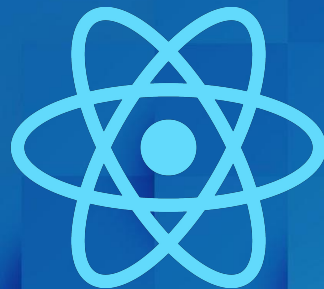




React и JSX

Автор курса: Алексей Тарасов



specialist.ru

- ✓ 15+ лет работы опыт работы в сфере образования и IT технологий,
- ✓ Сдал экзамены на ZCE (Zend Certified Engineer), 70-480 (Programming in HTML5 with JavaScript and CSS3)
- ✓ за 10 лет работы выпустил более 5000 слушателей в УЦ Специалист
- ✓ автор курсов React, Redux, Symfony

Веду линейку курсов по технологиям

- HTML, JavaScript, React, PHP, MySQL



Алексей Тарасов

Старший преподаватель УЦ
Специалист

Из чего состоит курс

- курс о популярной JS-библиотеке для UI
- **обязательно знание работы JS на клиентской стороне**
- 24 академических часа
- домашние задания
- консультации



Программа курса

Из чего состоит курс

Инструменты

Что такое React.js?

Как начать работу с React.js

Что такое JSX

Условная отрисовка

Что такое компоненты React?

Свойства компонента (props)

Композиция компонентов

Свойство children

Файловая структура React-проекта

Тестирование проекта

Сборка проекта

Обработчики событий

Состояние функционального компонента

Состояние классового компонента

Обработка ввода в `<input />`

Зачем нужно поднимать состояние

Хук `useEffect` и сторонние (side) эффекты

Как работать с контекстом?

Маршрутизация в SPA

Как реализовать модальное окно

Создание страницы с товарами (если будет время)

Хук `useReducer`

Обзор полезных инструментов

Подведение итогов



Инструменты

- VSCode + [Material Icon Theme]
- Node.js
- create-react-app
- Chrome + [React Developer Tools]



Что такое React.js?

- JS-библиотека для построения UI
- КОМПОНЕНТЫ
- СВОЙСТВА И СОСТОЯНИЕ



Как начать работу с React.js

- подключить через CDN
- с онлайн песочницей codesandbox
- Create React App (CRA)



Практика

- Откройте файл **01a-lab-install-CRA.pdf**, познакомьтесь с заданиями и выполните их



Что такое JSX

- элементы и дочерние элементы
- выражения в {}
- атрибуты
- className
- закрытие тегов
- инлайн-CSS



Условная отрисовка

- тернарный оператор
- `null` - ничего не отрисовывается



Практика

- Откройте файл **01b-lab-JSX.pdf**, познакомьтесь с заданиями и выполните их



Что такое компоненты React?

- части/фрагменты UI
- переиспользуемые кирпичики
- функциональный компонент
- классовый компонент



Свойства компонента (props)



```
1  function Hello(props) {  
2    return <h1>Привет, {props.name}</h1>;  
3  }  
4  
5  <Hello name={"world"} />
```



Практика

- Откройте файл **02-lab-react-components.pdf**, познакомьтесь с заданиями и выполните их



Композиция компонентов



```
1  <Item title='Название 1' />
2  <Catalog>
3      {items.map( item => (
4          <Item title={item.title} key={item.id} />
5      ) )}
6  </Catalog>
```



Практика

- Откройте файл **03-lab-composition.pdf**, познакомьтесь с заданиями и выполните их



Свойство children



```
1  function Wrapper(props) {  
2    return (  
3      <div className={"Wrapper Wrapper-" + props.color}>  
4        {props.children}  
5      </div>  
6    );  
7  }
```



Файловая структура React-проекта



```
1 components
2   |--- Button
3       Button.css
4       Button.jsx
5   |
6   |--- Footer
7       Footer.css
8       Footer.jsx
```

Тестирование проекта

- **npm test** - запуск теста



```
1  import React from "react";
2  import { render, screen } from "@testing-library/react";
3  import App from "../App";
4
5  test("renders learn react link", () => {
6    render(<App />);
7    const linkElement = screen.getByText(/learn react/i);
8    expect(linkElement).toBeInTheDocument();
9  });
```



Сборка проекта

- **npm run build** - запуск сборки



```
1 BUILD
2   └─ static
3       └─ css
4           main.073c9b0a.css
5           main.073c9b0a.css.map
6
7       └─ js
8           787.28cb0dcd.chunk.js
9           787.28cb0dcd.chunk.js.map
10          main.474b0368.js
11          main.474b0368.js.LICENSE.txt
12          main.474b0368.js.map
```

Обработчики событий



```
1  function Form() {  
2    const handleSubmit = (e) => {  
3      e.preventDefault();  
4      console.log("Отправлена форма.");  
5    };  
6  
7    return (  
8      <form onSubmit={handleSubmit}>  
9        <button type="submit">Отправить</button>  
10     </form>  
11  );  
12 }
```

Состояние функционального компонента



```
1  function Button() {  
2    const [counter, setCounter] = React.useState(0);  
3  
4    return (  
5      <button onClick={() => setCounter(counter + 1)}>Нажато: {counter}</button>  
6    );  
7  }
```



Состояние классowego компонента



```
1 class Button extends React.Component {  
2   constructor(props) {  
3     super(props);  
4     this.state = { counter: 0 };  
5   }  
6   render() {  
7     return (  
8       <button  
9         onClick={() => {  
10           this.setState({ counter: this.state.counter + 1 });  
11         }}  
12       >  
13         Нажато: {this.state.counter}  
14       </button>  
15     );  
16   }  
17 }
```



Обработка ввода в `<input />`



```
1  function App() {  
2    const [text, setText] = React.useState("");  
3  
4    return (  
5      <div className="App">  
6        <input value={text} onChange={(ev) => setText(ev.target.value)} />  
7        <hr />  
8        {text} <hr />  
9      </div>  
10   );  
11 }
```



Практика

- Откройте файл **04-lab-state.pdf**, познакомьтесь с заданиями и выполните их



Зачем нужно поднимать состояние



```
1  const Button = (props) => (  
2    <button onClick={() => props.addClick(props.counter + 1)}>  
3      Нажато: {props.counter}  
4    </button>  
5  );  
6  
7  export default function App() {  
8    const [counter, setCounter] = React.useState(0);  
9  
10   return (  
11     <div className="App">  
12       Нажато: {counter}  
13       <Button counter={counter} addClick={setCounter} />  
14       clicked: {counter}  
15     </div>  
16   );  
17 }
```

- чтобы состояние было общим

- состояние поднимается до общего предка

- в сложных приложениях состояние выносится



Хук useEffect и сторонние-эффекты



```
1  function App() {  
2    useEffect(() => {  
3      document.title = `Новый title страницы`;   
4    });  
5  
6    return <div className="App">Приложение</div>;  
7  }
```



Практика

- Откройте файл **05-lab-basket-component-with-state.pdf**, познакомьтесь с заданиями и выполните их



Как работать с контекстом?

- КОНТЕКСТ позволяет передать данные через дерево компонентов



```
1  const ThemeContext = React.createContext('light');
2
3  export class App extends React.Component {
4    render() {
5      return (
6        <ThemeContext.Provider value="dark">
7          <Toolbar />
8        </ThemeContext.Provider>
9      );
10   }
11 }
12
13 const Toolbar = () => (
14   <div>
15     <ThemedButton />
16   </div>
17 );
```



```
1  class ThemedButton extends React.Component {
2    static contextType = ThemeContext;
3    render() {
4      return <Button theme={this.context} />;
5    }
6  }
```



Практика (если будет время)

- Откройте файл **lab-usecontext-theme.pdf**, познакомьтесь с заданиями и выполните их *

* примечание: эти и последующие задания выполняются без привязки друг ко другу. Учтите это при работе



Маршрутизация в SPA

- SPA - Single Page Application
- маршруты
- ссылки
- компоненты



Практика (если будет время)

- Откройте файл **lab-custom-routing.pdf**, познакомьтесь с заданиями и выполните их *

* примечание: эти и последующие задания выполняются без привязки друг ко другу. Учтите это при работе



Как реализовать модальное окно



```
1  const [isShowAlert, setIsShowAlert] = React.useState(false)
```



```
1  {isShowAlert ? <Modal /> : null}
```



Практика (если будет время)

- Откройте файл **lab-modal-window.pdf**, познакомьтесь с заданиями и выполните их *

* примечание: эти и последующие задания выполняются без привязки друг ко другу. Учтите это при работе



Создание страницы с товарами

(если останется время)

- сортировка на стороне клиента
- изменение отображения товаров
- изображения товаров
- рейтинг товаров



Практика (если будет время)

- Откройте файл **lab-catalog-items.pdf**, познакомьтесь с заданиями и выполните их *

* примечание: эти и последующие задания выполняются без привязки друг ко другу. Учтите это при работе



Xyk useReducer



```
1  const initialState = { count: 0 };
2
3  function reducer(state, action) {
4    switch (action.type) {
5      case "increment":
6        return { count: state.count + 1 };
7      case "decrement":
8        return { count: state.count - 1 };
9      default:
10       throw new Error();
11    }
12  }
```



```
1  function Counter() {
2    const [state, dispatch] = useReducer(reducer, initialState);
3    return (
4      <>
5        Count: {state.count}
6        <button onClick={() => dispatch({ type: "decrement" })}>-</button>
7        <button onClick={() => dispatch({ type: "increment" })}>+</button>
8      </>
9    );
10 }
```



Обзор полезных инструментов

- Контейнеры состояния (**Redux**, MobX)
- Тестирование (Testing Library, Jest)
- Работа с формами (Formik, React Hook Forms, react-select, react-final-form)
- Стилизация компонентов (styled components, classnames, Material UI, Ant Design, React-Bootstrap)
- Построение графиков (Recharts, react-flow, react-vis)
- Маршрутизация (**react-router**)
- Next.js (React.js + **Node.js**)
- **TypeScript**

* выделены темы по которым есть курсы в Специалисте



Домашняя работа

- выполните лабораторные работы применительно к собственному проекту
- Напишите код компонента `Timer`. Компонент должен принимать параметр `seconds` - через сколько секунд таймер должен сработать (показать уведомление)
- Программист Василий после работы с DOM на курсе JavaScript, решил написать компонент `Clock`, которые показывает время по Москве, Берлину и Пекину. Реализуйте код этого компонента с использованием состояний
- Напишите код компонента `Form`, который не отправляет форму, пока не заполнены его поля



Учебный центр «СПЕЦИАЛИСТ» – Ваш путь к успеху

 info@specialist.ru

 +7 (495) 232-32-16

