Урок 5.1. Введение в React

Материалы

- Скачать NodeJS
- NVM - Node Version Manager
- NVM for Windows

Summary

Подготовка и создание приложения

Шаг 1. Установка NodeJS

1. Зайти на сайт NodeJS, скачать последнюю LTS версию (обычно кнопка слева)
2. Установить, следуя инструкциям установщика
3. Зайти в консоль (терминал), написать `node --version`, если вывелась версия - you're good to go :)

Шаг 2. Установка NVM (Linux, MacOS, возможно сработает на Windows в GitBash)

1. Открыть терминал, выполнить команду:

   ```
   curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.39.0/install.sh | bash
   ```

2. Перезагрузить терминал, написать `nvm -v`, если вывелась версия, you're good to go.

Шаг 2. Установка NVM (Windows)

1. Скачать установщик по ссылке в материалах (NVM for Windows)
2. Или скачать сразу по этой ссылке
3. Распаковать архив, запустить `nvm-setup.exe`
4. Следовать инструкциям установщика

5. Открыть консоль, написать `nvm -v`, если версия вывелась, значит всё хорошо.

Шаг 3. Создание React-приложения

1. Открыть терминал (консоль)
2. Перейти в директорию, где хотите, чтобы разместилось приложение
3. Выполнить команду `npx create-react-app react-app`, где `react-app` - название приложения (под него создастся отдельная директория)
4. Подождать, пока скрипт завершит выполнение, может занять некоторое время

Шаг 4. Открытие проекта

1. Открыть WebStorm (VSCode)
2. Открыть директорию `react-app` (ту, которая была создана скриптом)

Шаг 5. Создание файла `.nvmrc`

Файл `.nvmrc` необходим, чтобы все знали, какая версия Node используется в проекте

1. Создать в корне файл `.nvmrc`
2. Написать строчку с версией Node (например, `v16.13.0`). Если не уверены, какая у вас сейчас Node, можно в термминале (прямо в WebStorm) написать `nvm current`

Шаг 6. Создание директории под компоненты

Создайте директорию `components` в директории `src`

Шаг 7. Запуск

1. Откройте терминал в WebStorm (в VSCode тоже есть такой), либо можно просто открыть терминал или консоль в директории с проектом
2. Выполнить `npm run start`
3. Браузер должен открыться сам и открыть `localhost:3000`, если этого не произошло, можно сделать это вручную

Создание компонента в React

Подготовка

1. В директории `components` создайте новую и назовите её так, как будет называться компонент, например, `MySuperComponent`
2. Внутри созданной директории создайте 2 файла: `MySuperComponent.js` и `MySuperComponent.css`

`MySuperComponent.js`

```
// Функциональный подход
// Импортируем css файл
import './MyComponent.css';
// Функцию называем так же, как и файл js
function MySuperComponent() {
    // Возвращаем код разметки (JSX)
    // Внимание! Обязательно должен быть только ОДИН корневой тег.
    return (
        <div className="MyComponent">
            Hello, Functional Component!
        </div>
    );
}

// Экспортируем созданную функцию
export default MyComponent;
```

```
// Классовый подход
// Импортируем React
import React from 'react';
// Импортируем css файл
import './MySuperComponent.css';
// Создаём и одновременно экспортируем класс
// Класс называется так же, как и файл js
// Наследуем класс от React.Component
export default class MySuperComponent extends React.Component {
    // Создаём функцию render()
    render() {
        // Возвращаем код компонента (JSX)
        // Внимание! Обязательно должен быть только ОДИН корневой тег.
        return (<div>
            Hello Class Component!
        </div>);
    }
}
```

Использование компонента в `App.js`

```
// App.js
import './App.css';
// Импортируем компонент (путь должен подсказать VSCode, но можно написать
// самим по аналогии с тем, что представлено здесь
import MySuperComponent from './components/MySuperComponent/MySuperComponent';

function App() {
    return (
        <div className="App">
            {/* Используем название компонента как тег */}
            {/* Кстати, вот так пишутся комментарии внутри кода JSX */}
            <MySuperComponent/>
        </div>
    );
}

export default App;
```