Лекция 6. Стилизация React-приложений

Изучите распространённые способы стилизации компонентов React. Сосредоточьтесь на применении стилей с помощью встроенных стилей и внешних CSS-файлов.

1. Варианты стилизации в React

React не устанавливает единственный способ стилизации компонентов. Вместо этого он предоставляет гибкость в выборе того, как применять стили в зависимости от потребностей вашего проекта.

Стилизация в React по-прежнему основана на **CSS**. Разница заключается в том, **как стили организованы и применяются** к компонентам.

В этом курсе вы сосредоточитесь на двух основных подходах к стилизации:

- Inline styles: styles are applied directly to elements using the style attribute and JavaScript objects;
- External CSS files: styles are written in traditional CSS files and applied using class names.

Оформление с помощью библиотек и фреймворков

В реальных проектах приложения React часто оформляются с использованием **сторонних библиотек** и фреймворков. Среди популярных вариантов можно выделить следующие:

- CSS-модули;
- Стилизованные компоненты;
- Эмоции;
- Tailwind CSS;
- Материальный пользовательский интерфейс (MUI);
- Интерфейс чакр.

Эти инструменты предлагают расширенные возможности, такие как стили с заданной областью видимости, системы проектирования и стилизация на основе утилит.

Почему мы не освещаем их здесь

Несмотря на свою мощь, библиотеки стилей добавляют **дополнительную сложность** и вводят новые концепции, выходящие за рамки базового React. Цель этого курса — заложить прочный фундамент, понять, как работают компоненты React, и избежать ненужных отвлекающих факторов на начальном этапе.

2. Использование встроенных стилей в React

Один из простых способов применения стилей в React — использование **встроенных стилей** , аналогично тому, как стили добавляются к HTML-элементам с помощью styleатрибута.

Ключевое отличие заключается в том, что в React значение атрибута styleпредставляет собой объект JavaScript, а не строку CSS.

Вот пример применения встроенных стилей непосредственно в JSX:

```
const App = () => (
  <>
    <h1
      style={{
        fontWeight: 700,
        fontSize: "32px",
        color: "rebeccapurple",
      }}
    >
      App title
    </h1>
  </>
);
```

В этом примере h1элемент получает стили через styleатрибут, который содержит объект JavaScript.

Важные правила для встроенных стилей

При использовании встроенных стилей в React следует помнить о следующих правилах:

- Названия свойств CSS, состоящие из двух и более слов, должны быть написаны в стиле camelCase.
  - font-weight→ fontWeight;
  - background-color→ backgroundColor.
- Значения обычно записываются в виде строк: "32px", "red", "rebeccapurple";
- Некоторые свойства могут принимать числа напрямую: fontWeight: 700.

Эти правила существуют потому, что встроенные стили пишутся на JavaScript, а не на обычном CSS.

Встроенные стили как отдельный объект

Для того чтобы JSX был чище и легче читался, встроенные стили можно хранить в отдельном JavaScript-объекте, а затем передавать в styleатрибут.

```
const appStyles = {
  fontWeight: 700,
  fontSize: "32px",
  color: "rebeccapurple",
};

const App = () => (
  <>
    <h1 style={appStyles}>App title</h1>
```

```
  </>
);
```

Когда использовать встроенные стили

Встроенные стили лучше всего подходят для:

- Мелкие компоненты;
- Динамичные стили;
- Быстрая визуальная корректировка.

Они не идеально подходят для больших макетов или сложной стилизации, поэтому часто предпочтительнее использовать внешние CSS-стили.

3. Стилизация компонентов с помощью внешних CSS-стилей

Хотя встроенные стили полезны для небольших или динамических стилизаций, большинство реальных приложений полагаются на **внешние CSS-файлы** . Такой подход позволяет отделить стили от логики компонентов и упрощает управление ими по мере роста проекта.

React поддерживает внешние CSS-стили «из коробки». Вы можете писать обычные CSS-стили и применять их к компонентам, используя имена классов, как в традиционной веб-разработке.

Создание внешнего CSS-файла

Сначала создайте CSS-файл и определите свои стили:

```
/* styles.css */
.title {
  font-size: 32px;
  font-weight: 700;
  color: rebeccapurple;
}
```

Этот CSS-файл содержит простой класс, который можно повторно использовать в разных компонентах.

Импорт CSS в компонент

Чтобы использовать стили, импортируйте CSS-файл в файл вашего компонента:

```
import "./styles.css";

const App = () => (
  <>
    <h1 className="title">App title</h1>
  </>
);
```

В React атрибут classзаписывается так, classNameпотому что class является зарезервированным ключевым словом в JavaScript.

Как работают внешние CSS-стили в React

CSS-файлы по умолчанию применяются глобально. Любой компонент может использовать класс, определенный в импортированном CSS-файле. Стили ведут себя так же, как в стандартном HTML и CSS. Это делает использование внешних CSS-стилей привычным и простым, особенно для разработчиков, пришедших из традиционной веб-разработки.

Когда использовать внешние CSS-стили

Использование внешних CSS-стилей — хороший выбор в следующих случаях:

- Стили используются совместно несколькими компонентами;
- Планировка становится более сложной;
- Вам необходимо четкое разграничение между структурой и стилем.

В крупных проектах использование внешних CSS-стилей часто приводит к более чистому и удобному для сопровождения коду.

Вопросы.

1. В чём ключевое различие между применением стилей в HTML и использованием встроенных стилей в React?
2. Какую нотацию следует использовать при определении имен свойств, состоящих из двух или более слов, во встроенных стилях?
3. Как применить CSS-класс к элементу в React?
4. Что можно сказать о внешних CSS-стилях в React?