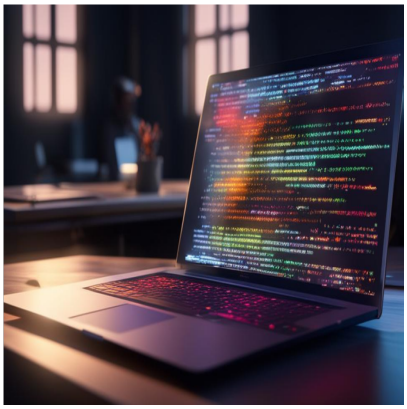


5 способов стилизации React с помощью CSS

Описание

Знаете ли вы, что более 97% веб-сайтов используют CSS для оформления? Каскадные таблицы стилей, или CSS, позволяют разработчикам создавать хорошо выглядящие, удобные для сканирования и презентабельные веб-страницы. Язык CSS определяет, как документы представляются пользователю. В данном случае документ – это файл, написанный на языке разметки, таком как XML или HTML.

Что такое стилизация в React?

Простота создания, запуска и поддержки приложений React – главная причина их популярности. React – это библиотека JavaScript, а не фреймворк, предлагающий больше, чем просто готовые функции и фрагменты кода. Наличие многократно используемых компонентов, гибкость, стабильность кода, скорость и производительность – вот некоторые из причин, по которым React занимает высокие позиции среди фреймворков и библиотек JavaScript. Стилизация в React – это процесс придания различным компонентам приложения React визуальной привлекательности с помощью CSS. Однако стоит отметить, что React использует JSX (JavaScript и XML) вместо HTML в качестве языка разметки. Одно из основных отличий заключается в том, что HTML использует `.class` для обозначения классов, в то время как JSX использует `.className` для обозначения того же самого.

Почему стоит стилизовать React с помощью CSS?

- **Сделайте свое приложение отзывчивым.** Современные веб-приложения должны быть доступны как на маленьких, так и на больших экранах. CSS позволяет применить медиа-запросы к приложению React и сделать его отзывчивым к разным размерам экрана.
- **Ускорьте процесс разработки.** Вы можете использовать одно и то же правило CSS в нескольких компонентах вашего приложения React.
- **Сделайте приложение React поддерживаемым.** Изменить внешний вид определенных компонентов/частей приложения легко с помощью CSS.
- **Улучшенный пользовательский опыт.** CSS обеспечивает удобное форматирование. В React, где текст и кнопки расположены в логичных местах, легко ориентироваться и пользоваться.

Существует несколько подходов, которые разработчики могут использовать для стилизации своих приложений React. Ниже приведены некоторые из наиболее распространенных;

Запись встроенных стилей

Встроенные стили – это самый простой подход к стилизации приложения React, поскольку пользователям не нужно создавать внешнюю таблицу стилей. Стилизация CSS применяется непосредственно к коду React. Стоит отметить, что встроенные стили имеют высокий приоритет над другими стилями.

Таким образом, если у вас есть внешняя таблица стилей с некоторым форматированием, она будет отменена встроенным стилем. Это демонстрация встроенной стилизации в приложении React.

```
import React from 'react';import ReactDOM from 'react-dom/client';const Header
      <h1 style={{backgroundColor: "lightblue"}}>HelloWorld!!!!!</h1>
      <h2>?????????????? ?????!</h2>
    </> );}const root = ReactDOM.createRoot(document.getElementById('root'));r
```

В отображаемом элементе появится h1 со светло-голубым фоном.

Плюсы линейной стилизации

- **Быстро.** Это самый простой подход, поскольку вы добавляете стиль непосредственно к тегу, который хотите стилизовать.
- **Обладает большими преимуществами**. Встроенные стили переопределяют внешние таблицы стилей. Таким образом, с его помощью можно сосредоточиться на определенной функциональности, не меняя все приложение.
- **Отличное решение для создания прототипов.** Вы можете использовать встроенные стили для тестирования функциональности, прежде чем включать форматирование во внешнюю таблицу стилей.

Недостатки инлайн-стилей

- **Может быть утомительным.** Непосредственная стилизация каждого тега отнимает много времени.
- **Ограниченность.** Вы не можете использовать такие возможности CSS, как селекторы и анимации, с помощью встроенных стилей.
- Большое количество встроенных стилей делает код JSX нечитаемым.

Импорт внешних таблиц стилей

Вы можете написать CSS во внешнем файле и импортировать его в приложение React. Этот подход можно сравнить с импортом CSS-файла в тег <head> HTML-документа. Для этого нужно создать CSS-файл в директории вашего приложения, импортировать его в целевой компонент и написать правила стиля для вашего

приложения. Чтобы продемонстрировать работу внешних таблиц стилей CSS, вы можете создать файл CSS и назвать его App.css. Затем вы можете экспортировать его следующим образом.

```
import { React } from "react";import "./Components/App.css";function App() { r
    </div> );}export default App;
```

Приведенный выше фрагмент кода импортирует внешнюю таблицу стилей в компонент App.js. Файл App.css находится в папке Components.

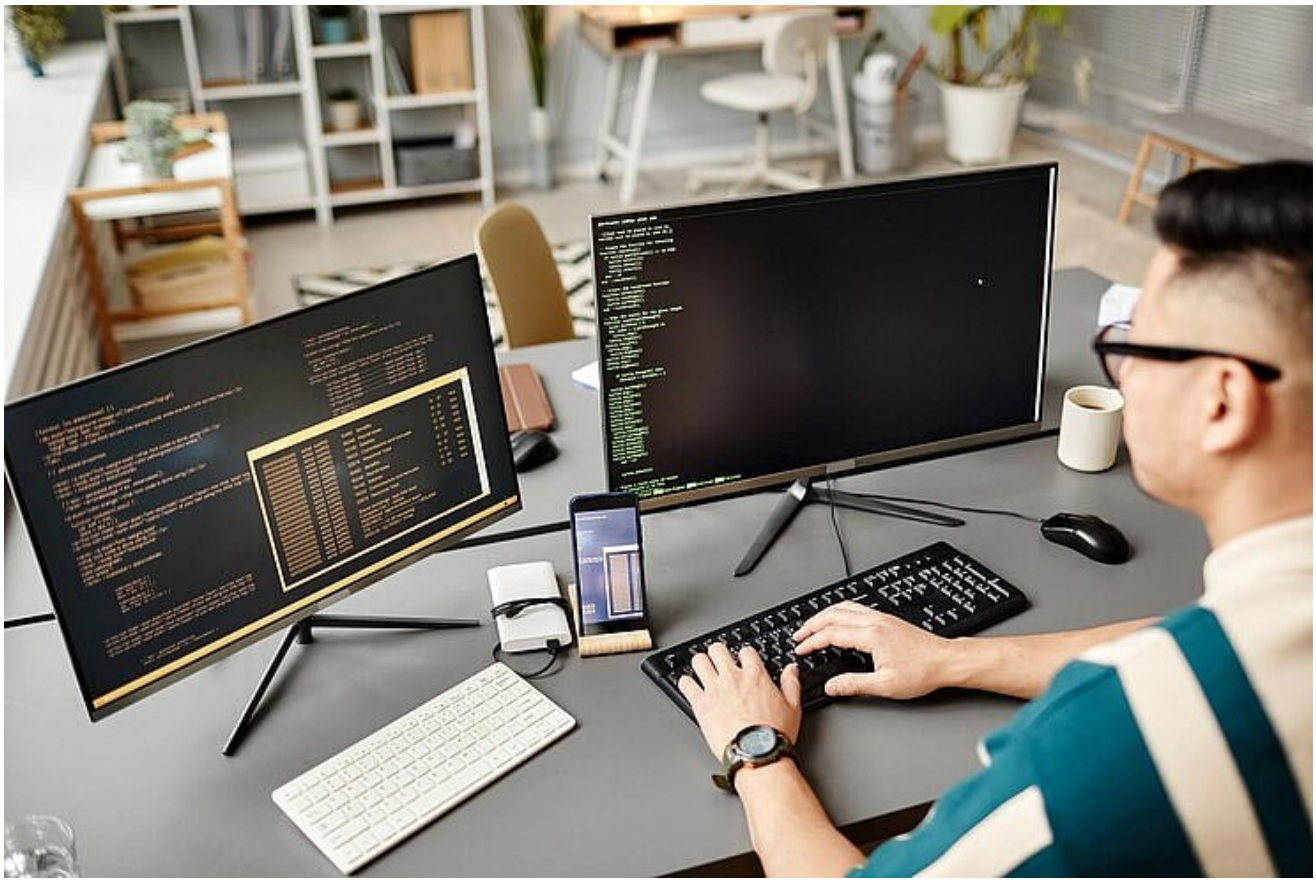
Плюсы внешних таблиц стилей CSS

- **Многоразовое использование**. Вы можете использовать одно и то же правило CSS в разных компонентах приложения React.
- **Делает код более презентабельным.** Понимание кода упрощается при использовании внешних таблиц стилей.
- **Предоставляет доступ к расширенным возможностям CSS**. Вы можете использовать псевдоклассы и расширенные селекторы при работе с внешними таблицами стилей.

Кон внешних таблиц стилей CSS

- Требуется надежное соглашение об именовании, чтобы не допустить переопределения стилей.

Используйте модули CSS



Приложения React могут быть очень большими. Имена анимаций и классы CSS по умолчанию имеют глобальное распространение. Эта настройка может быть проблематичной при работе с большими таблицами стилей, так как один стиль может переопределять другой. Модули CSS решают эту проблему путем локального определения имен анимации и классов. Такой подход гарантирует, что имена классов будут доступны только в том файле/компоненте, где они нужны. Каждое имя класса получает уникальное программное имя, что позволяет избежать конфликтов. Чтобы реализовать модули CSS, создайте файл с именем `.module.css`. Если вы хотите назвать свою таблицу стилей style, то имя файла будет **style.module.css**. Импортируйте созданный файл в ваш компонент React, и вы будете готовы приступить к работе. Ваш CSS-файл может выглядеть примерно так;

```
/* styles.module.css */.font { color: #f00; font-size: 30px;}
```

Вы можете импортировать модуль CSS в App.js следующим образом;

```
import { React } from "react";import styles from "./styles.module.css";functio
```

Плюсы использования модулей CSS

- Легко интегрируется с SCSS и CSS
- Избегайте конфликтов имен классов

Минусы использования модулей CSS

- Ссылки на имена классов в модулях CSS могут запутать новичков.

Используйте стилизованные компоненты

Стилизованные компоненты позволяют разработчикам создавать компоненты с использованием CSS в коде JavaScript. Стилизованный компонент действует как компонент React, который поставляется со стилями. Стилизованные компоненты предлагают динамическую стилизацию и уникальные имена классов. Чтобы начать использовать Styled Components, вы можете установить пакет в корневую папку с помощью этой команды;

```
npm install styled-components
```

Следующим шагом будет импорт стилизованных компонентов в ваше приложение React. Ниже приведен фрагмент кода App.js, в котором используются стилизованные компоненты;

```
import { React } from "react";import styled from "styled-components";function
```

В отрисованном приложении будут присутствовать вышеуказанные стили, импортированные из Styled Components.

Плюсы стилизованных компонентов

- **Он предсказуем**. Стили при таком подходе к стилизации вложены в отдельные компоненты.
- **Не нужно зацикливаться на соглашениях об именовании классов.** Просто напишите свои стили, а пакет позаботится обо всем остальном.
- **Вы можете экспортировать стилизованные компоненты в качестве реквизитов.** Styled Components преобразуют обычный CSS в компоненты React. Таким образом, вы можете повторно использовать этот код, расширять стили с помощью реквизитов и экспортировать.

Кон из стилизованных компонентов

- Для начала работы необходимо установить библиотеку стороннего производителя.

Синтаксические таблицы стилей (SASS/ SCSS)

SASS обладает более мощными инструментами и возможностями, которые отсутствуют в обычном CSS. Вы можете писать стили в двух разных стилях, руководствуясь этими расширениями: **.scss и .sass**.

Синтаксис SASS похож на синтаксис обычного CSS. Однако при написании стилевых правил в SASS открывающие и закрывающие скобки не нужны. Простой фрагмент SASS будет выглядеть следующим образом;

```
nav ul margin-right: 20px padding: 0 list-style: list li display: block a disp
```

Чтобы начать использовать SASS в своем приложении React, вам нужно сначала скомпилировать SASS в обычный CSS. После настройки приложения с помощью команды Create React App вы можете установить node-sass, который позаботится о компиляции.

```
npm install node-sass
```

Затем вы можете создать свои файлы и присвоить им расширения .scss или .sass. Затем вы можете импортировать файлы обычным способом. Например;

```
??????????? "./Components/App.sass";
```

Плюсы SASS/SCSS

- В нем есть множество динамических функций, таких как миксины,

вложенность и расширение.
- При использовании SASS/SCSS вам не придется писать много шаблонов для CSS-кода.

Минусы SASS/SCSS

- Стили являются глобальными, и поэтому вы можете столкнуться с проблемами переопределения.

Какой подход к укладке лучше выбрать?

Поскольку мы обсудили пять подходов, вы хотите знать, какой из них самый лучший. В этой дискуссии трудно выделить абсолютного победителя. Однако эти соображения помогут вам принять взвешенное решение:

- Показатели производительности
- Нужна ли вам система проектирования
- Простота оптимизации кода

Встроенная стилизация подходит для простых приложений с небольшим количеством строк кода. Однако все остальные: внешний, SASS, стилизованные компоненты и CSS-модули – подходят для больших приложений.

Каковы лучшие практики поддержки CSS в большом React-

приложении?

- **Избегайте инлайн-стилей.** Написание встроенных стилей CSS для каждого тега в большом приложении React может быть утомительным. Вместо этого используйте внешнюю таблицу стилей, которая подходит для ваших нужд.
- **Линтуйте свой код.** Линтеры, такие как Stylelint, выделяют ошибки стилизации в вашем коде, чтобы вы могли исправить их на ранней стадии.
- **Регулярно проводите обзоры кода.** Написание CSS кажется увлекательным занятием, но регулярные обзоры кода позволяют легко выявить ошибки на ранней стадии.
- **Автоматизируйте тесты для ваших CSS-файлов.** Enzyme и Jest – замечательные инструменты, которые можно использовать для автоматизации тестов вашего CSS-кода.
- При работе с часто используемыми стилями, такими как цвета и поля, используйте служебные переменные и классы, так как это соответствует принципу “Не повторяйся” (DRY)**.**
- **Используйте соглашения об именовании, такие как Block Element Modifier.** Такой подход позволяет легко писать классы CSS, которые легко понять и повторно использовать.

Заключение

Выше перечислены некоторые способы, которые можно использовать для стилизации React. Выбор подхода к стилизации зависит от ваших потребностей, навыков и вкуса. Вы даже можете сочетать несколько подходов к стилизации, например, встроенные и внешние таблицы стилей, в своем приложении React.

Дата Создания
06.03.2024