Styled-components en React

Julian F. Latorre

Índice

1. Introducción

styled-components es una biblioteca popular para React que permite escribir CSS en JavaScript. Esta guía te mostrará cómo usar styled-components de manera efectiva en tus proyectos de React.

2. Conceptos Básicos

2.1. Instalación

Para comenzar, instala styled-components en tu proyecto React:

```
npm install styled-components
```

2.2. Creación de un Componente Estilizado Básico

Aquí tienes un ejemplo simple de cómo crear un componente estilizado:

```
import styled from 'styled-components';

const Button = styled.button`
  background-color: #4CAF50;
  border: none;
  color: white;
  padding: 15px 32px;
  text-align: center;
  text-decoration: none;
  display: inline-block;
  font-size: 16px;
  margin: 4px 2px;
  cursor: pointer;
`;

// Uso
function App() {
  return <Button>Click me</Button>;
}
```

3. Técnicas Intermedias

3.1. Props Dinámicas

Puedes usar props para crear estilos dinámicos:

```
const Button = styled.button`
  background-color: ${props => props.primary ? '#4CAF50' : '#008CBA
    '};
  color: white;
  padding: 10px 20px;
  border: none;
  border-radius: 5px;
`;

// Uso
<Button primary>Primary Button</Button>
<Button>Secondary Button</Button>
```

3.2. Extendiendo Estilos

Puedes extender estilos de componentes existentes:

```
const Button = styled.button`
  color: palevioletred;
  font-size: 1em;
  margin: 1em;
  padding: 0.25em 1em;
  border: 2px solid palevioletred;
  border-radius: 3px;
`;
```

```
9
10 const TomatoButton = styled(Button)`
11   color: tomato;
12   border-color: tomato;
13 `;
```

3.3. Estilos Globales

Crea estilos globales para toda tu aplicación:

```
1  import { createGlobalStyle } from 'styled-components';
2
3  const GlobalStyle = createGlobalStyle`
4    body {
5      margin: 0;
6      padding: 0;
7      font-family: Open-Sans, Helvetica, Sans-Serif;
8    }
9  `;
10
11 // En tu componente principal
12 function App() {
13   return (
14     <>
15       <GlobalStyle />
16       {/* Resto de tu aplicaci n */}
17     </>
18   );
19 }
```

4. Mejores Prácticas

4.1. Organización de Componentes

Es una buena práctica mantener tus componentes estilizados en archivos separados:

```
1  // Button.styles.js
2  import styled from 'styled-components';
3
4  export const Button = styled.button`
5    /* estilos aqu  */
6  `;
7
8  // Button.js
9  import React from 'react';
10 import { Button } from './Button.styles';
11
12 const MyButton = ({ children }) => {
13   return <Button>{children}</Button>;
14 };
15
16 export default MyButton;
```

4.2. Uso de Temas

styled-components proporciona un ThemeProvider para manejar temas:

```
import { ThemeProvider } from 'styled-components';

const theme = {
  colors: {
    primary: '#4CAF50',
    secondary: '#008CBA',
  },
};

function App() {
  return (
    <ThemeProvider theme={theme}>
      {/* Tus componentes aqu  */}
    </ThemeProvider>
  );
}

// En tus componentes
const Button = styled.button`
  background-color: ${props => props.theme.colors.primary};
`;
```

5. Ejercicio Práctico

Para poner en práctica los conceptos aprendidos en esta guía, te proponemos el siguiente ejercicio:

5.1. Enunciado

Crea una pequeña aplicación de React que muestre una lista de tareas (todo list) utilizando styled-components. La aplicación debe incluir:

- Un componente `TaskList` que muestre la lista de tareas.
- Un componente `TaskItem` para cada tarea individual.
- Un componente `AddTaskForm` para añadir nuevas tareas.
- Utiliza un tema global para los colores y fuentes.
- Implementa estilos condicionales (por ejemplo, tareas completadas vs. pendientes).

5.2. Requerimientos

1. Usa styled-components para todos los estilos.
2. Implementa props dinámicas para los estilos condicionales.

3. Utiliza el ThemeProvider para gestionar un tema global.
4. Organiza tus componentes y estilos en archivos separados.
5. Añade al menos una animación simple (por ejemplo, al completar una tarea).

5.3. Ejemplo de Estructura

Aquí tienes un esquema básico para empezar:

```
// App.js
import React from 'react';
import { ThemeProvider } from 'styled-components';
import { GlobalStyle } from './styles/GlobalStyle';
import TaskList from './components/TaskList';
import AddTaskForm from './components/AddTaskForm';

const theme = {
  colors: {
    primary: '#4CAF50',
    secondary: '#008CBA',
    background: '#f4f4f4',
    text: '#333',
  },
  fonts: {
    main: 'Arial, sans-serif',
  },
};

function App() {
  return (
    <ThemeProvider theme={theme}>
      <GlobalStyle />
      <div>
        <h1>Mi Lista de Tareas</h1>
        <AddTaskForm />
        <TaskList />
      </div>
    </ThemeProvider>
  );
}

export default App;

// components/TaskItem.js
import styled from 'styled-components';

const StyledTaskItem = styled.li`
  // A ade tus estilos aqu
`;

const TaskItem = ({ task }) => {
  return <StyledTaskItem>{task.text}</StyledTaskItem>;
};

export default TaskItem;
```

```
47
48 // Implementa los dem s componentes de manera similar
```

5.4. Desafío

Intenta implementar un toggle de modo oscuro/claro utilizando styled-components y el contexto de tema.