ToiletPaper #101

React Styled-Components

Author: C. D.

✗ Problem

“Two CSS properties walk into a bar. A barstool in a completely different bar falls over.” – Any developer who has ever dealt with Cascading Style Sheets (CSS) knows the issue meant in this joke: CSS has global reach. Unwanted side effects and bugs are the results, which causes unnecessary extra development effort. In current projects, I was particularly confronted with the following problems:

- Collisions of class names
- Conflicting CSS rules
- Difficulties in eliminating “dead code”

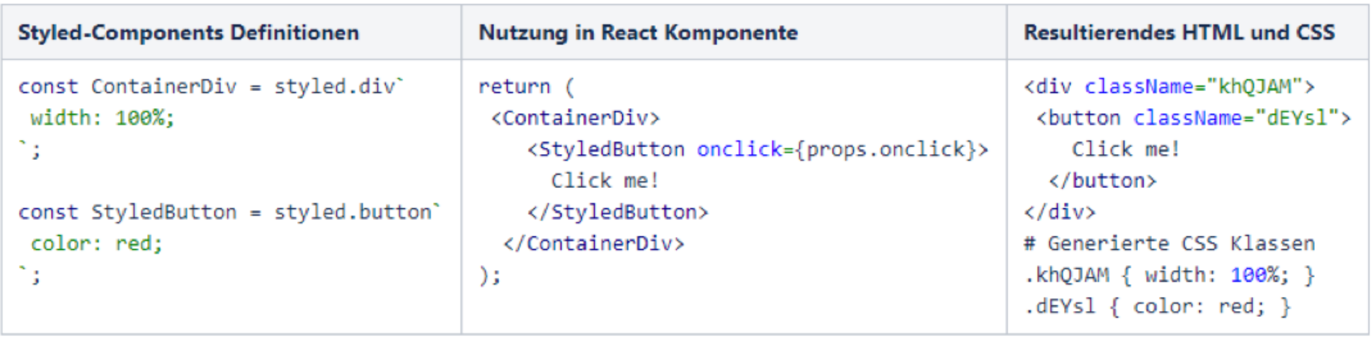
✓ Solution

For React applications, **styled-components** provide an elegant solution to the above-mentioned problems. Styled-components define CSS-based styling in a modular way for other React components. They are **declared directly in the JavaScript code** and act as wrappers for other components in the virtual DOM.

Mapping components and styles is no longer done manually during development (for example via class names), but is implicitly regulated by the library using generated class names. In this way, you can affect the core problem – the global reach of CSS.

By assigning the CSS rules – specified in a styled-component – only to the respective components, **the scope is limited** to the components the programmer has determined. At the same time, the defined styling remains **reusable**, is clearly separated from functional and state-related components and integrates intuitively into the code structure of a React application.

➔ Example



Styled-Components Definitionen	Nutzung in React Komponente	Resultierendes HTML und CSS

const ContainerDiv = styled.div`
width: 100%;
```
;

const StyledButton = styled.button
```
color: red;
`;
	return (
	<ContainerDiv>
	<StyledButton onclick={props.onclick}>
	Click me!
	</StyledButton>
	</ContainerDiv>
	);
		<div className="khQJAM">
		<button className="dEYsl">
		Click me!
		</button>
		</div>
		# Generierte CSS Klassen
		.khQJAM { width: 100%; }
		.dEYsl { color: red; }

+ Further Aspects

- Documentation of styled-components: https://www.styled-components.com/docs
- Blogpost about CSS evolution: https://medium.com/@perezpriego7/css-evolution-from-css-sass-bem-css-modules-to-styled-components-d4c1da3a659b
- Examples for direct alternatives to styled-components: Emotion (https://emotion.sh/) or Glamorous (https://glamorous.rocks/)
- Other solution approaches for the scoping problem: React Inline Styles (https://reactjs.org/docs/dom-elements.html#style) or CSS Modules (https://github.com/css-modules/)