CSS i React

Olika sätt att styla React-komponenter

- **Global CSS** – Vanlig CSS-fil importeras globalt.
- **Inline Styling** – CSS skrivs direkt i `style` -attributet.
- **CSS Modules** – Modulbaserad CSS för lokal styling.
- **Styled Components** – CSS-in-JS med TypeScript-stöd.

Global CSS

- Precis som i HTML kan en **global CSS-fil** importeras.
- Klasserna i filen kan användas överallt i applikationen.

```
import "./App.css";
```

Nackdel: Global CSS kan leda till **stilkonflikter** mellan komponenter.

Scoped Styling

I React är det rekommenderat att **undvika global styling**.

Varje komponent bör ha **sin egen styling** för att undvika oväntade effekter.

Detta kallas **scoped styling** och kan lösas med olika tekniker.

Inline Styling

- **CSS skrivs som ett objekt** direkt i `style`-attributet.
- CSS-attribut skrivs i **camelCase**.
- Värden måste vara **strängar** eller annan tillåten datatyp.

```
const buttonStyle = {
  backgroundColor: "#333333",
  borderRadius: "3px",
  padding: "5px 10px",
  color: "white",
};

const Button = () => <button style={buttonStyle}>En stylad knapp!</button>;
```

Nackdel: Inline-styling blir snabbt svårhanterad vid stora projekt.

CSS Modules

- CSS Modules ger **lokal styling** för en specifik komponent.
- Filer döps till `[filnamn].module.css` .
- Importeras direkt i komponenten.
- Se nästa sida

```css
/* button.module.css */
.button {
  background-color: #333333;
  border-radius: 3px;
  padding: 5px 10px;
  color: white;
}
```

```jsx
import styles from "./button.module.css";

const ButtonCSSModule = () => (
  <button className={styles.button}>Klicka här</button>
);
```

Fördel: Hindrar stilkonflikter och håller CSS lokal till komponenten.

Styled Components

- **CSS-in-JS**-teknik där styling skrivs direkt i JavaScript/TypeScript.
- Komponentens styling är **helt isolerad**.
- Möjlighet att använda **JavaScript-logik** direkt i CSS.

Installation

```
npm install styled-components
npm install --save-dev @types/styled-components
```

Exempel på Styled Components

```
import styled from "styled-components";

const StyledButton = styled.button`
  background-color: #333333;
  border-radius: 3px;
  padding: 5px 10px;
  color: white;
`;

const ButtonStyledComponents = () => <StyledButton>Klicka här</StyledButton>;
```

Fördel: Möjliggör komponentbaserad styling och dynamiska stilar.

Styled Components med Props

- CSS kan anpassas dynamiskt genom att använda **props**.

```
const StyledButton = styled.button<{ color: string }>`
  background-color: ${(props) => props.color};
  border-radius: 3px;
  padding: 5px 10px;
  color: white;
`;

const App = () => <StyledButton color="red">Klicka här!</StyledButton>;
```

Användningsområde: Skapa återanvändbara och konfigurerbara komponenter.

Pseudoklasser och Child-selektorer i Styled Components

- **Pseudoklasser** används med `&`.
- **Child-selektorer** kan också användas för att styla underliggande element.

```
const MyStyledButton = styled.button`
  &:hover {
    background-color: red;
  }
`;

const MyStyledList = styled.ul`
  & > li {
    color: red;
  }
`;
```