



Seoul
Software
ACademy

웹 개발자 부트캠프 과정

SeSAC x CODINGOn

With. 팀 리쳐드

React Style

React 컴포넌트 스타일링 방식

- 일반 CSS

- 소규모 프로젝트에 적합
- 이미 알고 있는 바로 그 CSS!!

- Sass (Syntactically Awesome Style sheets)

- 문법적으로 멋진 스타일시트
- CSS 전처리기를 이용해 복잡한 작업을 보다 쉽게 가능!!

- Styled Components

- JavaScript 안에서 CSS를 작성하도록 도와주는 CSS-in-JS 라이브러리
- 컴포넌트 기반의 설계 방식

잠깐! CSS 전처리기관?

자신만의 특별한 문법을 가지고 CSS를 생성하는 프로그램! 즉, 변수, 함수 개념을 사용하여 재활용성과 가독성을 높임

CSS(1)

- 기존 CSS 클래스를 이용할 경우, 컴포넌트별 클래스가 중복되지 않도록 클래스명을 지정해야 함
- 방법 1) 이름 규칙 정하기
 - 이름 - 클래스 형태로 이름 만들기 `App-logo`
 - BEM 네이밍 `app__title-primary`
- 방법 2) CSS Selector

```
/* App 컴포넌트 내부의 header 태그만을 의미 */
.App header { ... }
```

CSS(2)

- CSS Module 형식을 사용하게되면 자동으로 클래스명이 생성
 - css 파일명을 컴포넌트명.module.css로 생성
 - js파일에서 import해서 객체형태로 사용

```
import styled from "../App.module.css";

export default function App() {
  return <div className={styled.main}>Hello World</div>;
}
```

```
/* App.module.css */
.main {
  font-weight: 900;
}
```

SASS

Sass

- Syntactically Awesome Style Sheets
- CSS 전처리기 종류 중 하나
 - Sass/SCSS, PostCSS, Stylus, ...
- 복잡한 작업을 쉽게 할 수 있도록 도와주고, 스타일 코드의
재활용성과 코드의 가독성을 높여 유지보수를 더욱 쉽게 해준다.
- 가능한 확장자



`.SCSS`

`.sass`

CSS 전처리기 사용 방법

1. 자신만의 특별한 문법을 가지고 스타일을 꾸밈
2. 표준 CSS로 컴파일 (브라우저가 이해하는 CSS 문법으로 전환하는 작업 필요)

Sass 확장자 **.scss**

```
$font-stack: Helvetica, sans-serif;  
$primary-color: #333;  
  
body {  
    font: 100% $font-stack;  
    color: $primary-color;  
}
```


Sass 확장자 **.sass**

```
$font-stack: Helvetica, sans-serif  
$primary-color: #333
```

```
body
```

```
  font: 100% $font-stack  
  color: $primary-color
```

.sass VS .scss

- .sass 는 세미콜론(;) 과 중괄호({ }) 를 사용하지 않고 탭(tab) 을 이용해 스타일을 정의한다.
- .scss는 기존의 .css 파일과 비슷한 문법을 이용한다.
SCSS가 CSS와 거의 같은 문법으로 SASS 기능을 지원함 (더 넓은 범용성 + CSS와의 호환성)
- .sass 와 .scss 모두 변수를 선언할 때는 \$ 를 앞에 붙여줘야 한다.

Sass 사용하기

```
npm install node-sass
```

```
"dependencies": {
  "@testing-library/jest-dom": "^5.16.5",
  "@testing-library/react": "^13.4.0",
  "@testing-library/user-event": "^13.5.0",
  "node-sass": "^7.0.3",
  "react": "^18.2.0",
  "react-dom": "^18.2.0",
  "react-scripts": "5.0.1",
  "web-vitals": "^2.1.4"
},
```

Sass 활용

1. 변수 사용

```
// 변수 활용
$primary-color: #ff6347;
body {
  background-color: $primary-color;
}
```

2. 연산

```
// 연산
.container {
  width: 100% - 20px;
}
```

3. 믹스인

```
// 믹스인 (Mixin): 미리 CSS 코드 블록 정의하여 재사용
@mixin border-radius($radius) {
  -webkit-border-radius: $radius;
  border-radius: $radius;
}
.button {
  @include border-radius(10px);
}
```

Sass 활용

4. 중첩

```
// 중첩: 선택자를 중첩시켜 코드의 중복을 줄이고, 가독성 높임
div {
  ul {
    margin: 0;
    li {
      display: inline-block;
    }
  }
  &:hover {
    background-color: brown;
  }
}
```

5. 확장

```
//확장 (Extend): 기존의 선택자 스타일을 다른 선택자에게 상속
.btn {
  padding: 10px;
  margin: 10px;
  border: 1px solid black;
}
.btn-primary {
  @extend .btn;
  background-color: blue;
}
```

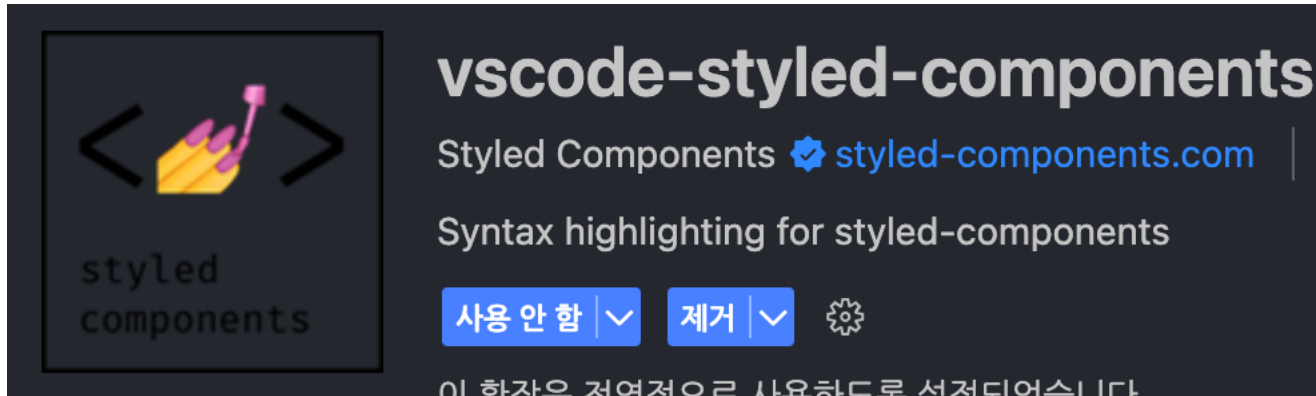
styled-components

사용 방법

설치 방법

```
npm install styled-components
```

VSCODE 사용자라면 아래 확장팩 설치



사용 방법

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

const StyledButton = styled.button`
  background-color: blue;
  color: white;
  padding: 10px 15px;
  border-radius: 4px;
`;

export default function App() {
  return (
    <div>
      <StyledButton>클릭</StyledButton>
    </div>
  );
}
```

HTML 태그 + 백틱