

AI 활용 빅데이터분석 풀스택웹서비스 SW 개발자 양성과정

React



리액트 (React)

- 사용자 인터페이스를 만들기 위한 JavaScript 라이브러리
 - 컴포넌트(component) : 사용자가 정의한 태그
 - 선언형
 - 데이터가 변경됨에 따라 적절한 컴포넌트만 효율적으로 갱신하고 렌더링
 - 컴포넌트 기반
 - 스스로 상태를 관리하는 캡슐화된 컴포넌트를 조합해 복잡한 UI 생성
 - 한 번 배워서 어디서나 사용
 - Node 서버에서 렌더링을 할 수도 있고, React Native를 이용하면 모바일 앱 개발
 - 참고
 - <https://ko.reactjs.org/> , <https://react.dev/>
 - <https://react.dev/learn/react-developer-tools>

리액트 (React)

- SPA (Single-page application)

- 하나의 HTML 페이지에 애플리케이션 실행에 필요한 JavaScript와 CSS 같은 모든 자산을 로드하는 애플리케이션

- 웹 사이트 전체 페이지를 하나의 페이지에 담아 동적으로 화면을 바꿔가며 표현하는 것

- 페이지 또는 후속 페이지의 상호작용은 서버로부터 새로운 페이지를 불러오지 않으므로 페이지가 다시 로드되지 않음

- React를 사용하여 싱글 페이지 애플리케이션을 만들 수 있지만, 필수 사항은 아님

새로운 React 앱 만들기

- Create React App

- 새로운 싱글 페이지 애플리케이션으로 React의 간편한 환경을 제공
- 개발 환경을 설정하고, 최신 JavaScript를 사용하게 해주며, 좋은 개발 경험과 프로덕션 앱 최적화

- 설정

- Node 설치 (Node 14.0.0 이상 , npm 5.6 이상 버전이 필요)
 - <https://nodejs.org/en/>

새로운 React 앱 만들기

```
PS C:\minNote\OneDrive - pusan.ac.kr\강의자료\2022_0919_K디지털\React\work> npx create-react-app ./my01
```

Creating a new React app in C:\minNote\OneDrive - pusan.ac.kr\강의자료\2022_0919_K디지털\React\work\my01.

Installing packages. This might take a couple of minutes.
Installing react, react-dom, and react-scripts with cra-template...

added 1394 packages in 1m

210 packages are looking for funding
run `npm fund` for details

Initialized a git repository.

Installing template dependencies using npm...

added 56 packages in 9s

210 packages are looking for funding
run `npm fund` for details
npm audit fix --force

Run `npm audit` for details.

Created git commit.

Success! Created my01 at C:\minNote\OneDrive - pusan.ac.kr\강의자료\2022_0919_K디지털\React\work\my01
Inside that directory, you can run several commands:

```
npm start
Starts the development server.
```

```
npm run build
Bundles the app into static files for production.
```

```
npm test
Starts the test runner.
```

```
npm run eject
Removes this tool and copies build dependencies, configuration files
and scripts into the app directory. If you do this, you can't go back!
```

We suggest that you begin by typing:

```
cd my01
npm start
```

Creating a new React app in C:\minNote\OneDrive - pusan.ac.kr\강의자료\2022_0919_K디지털\React\work\my01.

Installing packages. This might take a couple of minutes.
Installing react, react-dom, and react-scripts with cra-template...

added 1394 packages in 1m

210 packages are looking for funding
Bundles the app into static files for production.

```
npm test
Starts the test runner.
```

```
npm run eject
Removes this tool and copies build dependencies, configuration files
Compiled successfully!
```

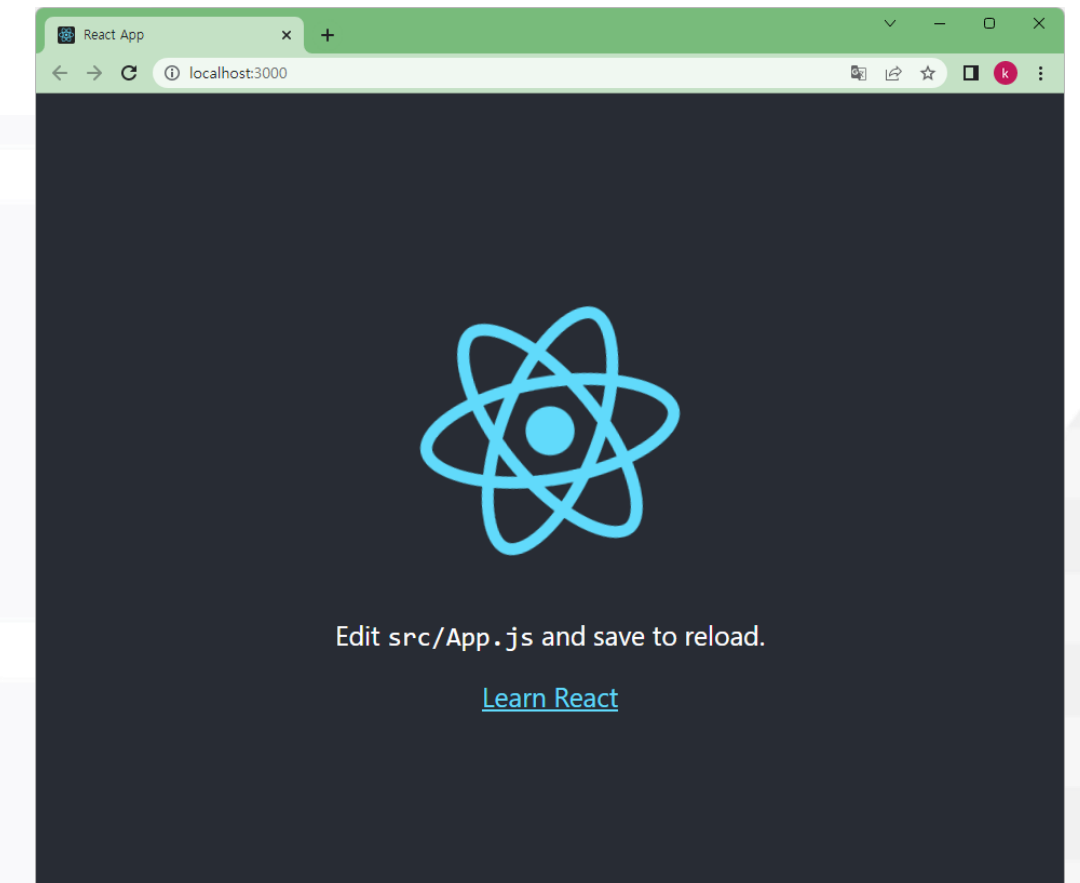
You can now view my01 in the browser.

```
Local:      http://localhost:3000
On Your Network: http://192.168.137.1:3000
```

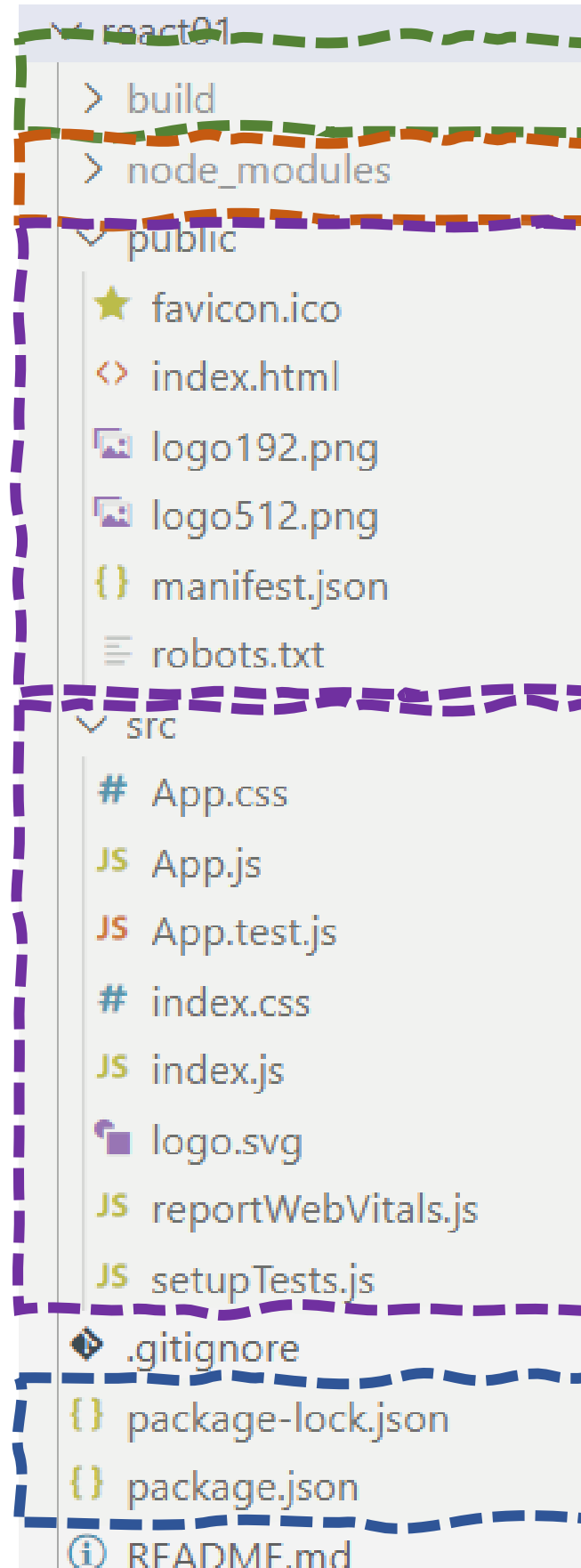
Note that the development build is not optimized.
To create a production build, use `npm run build`.

프로젝트명은 소문자로 작성

```
npx create-react-app my-app
cd my-app
npm start
```



create-react-app 프로젝트 폴더



• 빌드를 한 경우에 생성 : `npm run build` 후 `npx serve -s build`로 실행

• **node_modules**: React 앱을 실행하기 위해 설치한 Node 모듈 저장

• **public**: 웹 브라우저에 실제로 보이는 정적 파일(static files) 저장

• **src**: 모든 컴포넌트(components) 파일 저장; UI 조각 저장

- **package-lock.json**: npm이 node_modules 또는 package.json을 수정할 때 디펜던시 버전 정보(버전명) 저장
- **package.json**: 프로젝트에 대한 메타데이터(metadata) 및 디펜던시 버전 정보(범위) 저장

public > index.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <link rel="icon" href="%PUBLIC_URL%/favicon.ico" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <meta name="theme-color" content="#000000" />
    <meta
      name="description"
      content="Web site created using create-react-app"
    />
    <link rel="apple-touch-icon" href="%PUBLIC_URL%/logo192.png" />
    <link rel="manifest" href="%PUBLIC_URL%/manifest.json" />
    <title>React App</title>
  </head>
  <body>
    <noscript>You need to enable JavaScript to run this app.</noscript>
    <div id="root"></div>
  </body>
</html>
```

React의 결과물이 들어가는 부분으로
src 폴더의 index.js파일을 수정

- CRA 프로젝트의 단일 HTML 파일이며, 웹 애플리케이션의 진입점

- public 폴더 내에 위치

- 마운트 포인트가 되는 DOM 요소인 <div id="root"></div>를 제공

- 리액트 애플리케이션의 모든 컴포넌트는 이 <div> 안에 렌더링

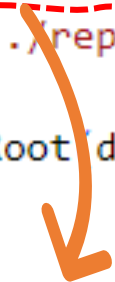
- 렌더링된 결과물이 바로 SPA에서 말하는 단 1개의 페이지

src > index.js

```
import React from 'react';
import ReactDOM from 'react-dom/client';
import './index.css';
import App from './App';
import reportWebVitals from './reportWebVitals';

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>
);

//measuring performance in your app
reportWebVitals(console.log);
```

A red dashed box highlights the import statement `import App from './App';` and another red dashed box highlights the `<App />` component usage within the `<React.StrictMode>` block in the `root.render()` call. An orange arrow points from the `App` in the import statement to the `<App />` in the JSX element.

- src 폴더

- 만든 모든 컴포넌트, 즉 UI 조각들이 저장되는 곳

- index.js

- 리액트 애플리케이션의 JavaScript 진입점

- 리액트와 DOM을 연결하여 리액트 애플리케이션의 최상위 컴포넌트인 App 컴포넌트를 импорт

- ReactDOM.render() 메서드를 사용하여 App 컴포넌트를 index.html의 `<div id="root"></div>`에 렌더링

src > index.js

```
import React from 'react';
import ReactDOM from 'react-dom/client';
import './index.css';
import App from './App';
import reportWebVitals from './reportWebVitals';

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>
);
```

```
//measuring performance in your app
reportWebVitals(console.log);
```

reportWebVitals 파일에서 가져온 함수를 실행

- **console.log**를 넣어주면 개발창으로 앱의 퍼포먼스시간들을 분석하여 객체 형태로 보여줌
- **metric**(측정도구) 이름
name: 'CLS' | 'FCP' | 'FID' | 'LCP' | 'TTFB';
- 측정된 현재값 (값이 작을수록 빠른성능을 가짐)
value: number;
- 현재 측정값(**current value**)과 최신 측정값(**last-reported value**) 차이
delta: number;
- 특정 측정도구를 나타내는 유니크한 **ID** 값으로 중복되는 값들을 관리
id: string;
- 계산된 측정값들의 내용들이 배열로 나열
entries: (PerformanceEntry | FirstInputPolyfillEntry | NavigationTimingPolyfillEntry)[];

• reportWebVitals 함수

- Google의 웹 성능 측정 도구인 Web Vitals API를 활용하여, 사용자의 실제 경험을 반영하는 여러 중요 성능 지표를 측정

```
web-vitals.js:1
{name: 'TTFB', value: 8.599999904632568, delta: 8.599999904632568, entries: Array(1), id: 'v2-1664616015014-9097766879736'}
  delta: 8.599999904632568
  entries: [PerformanceNavigationTiming]
  id: "v2-1664616015014-9097766879736"
  name: "TTFB"
  value: 8.599999904632568
  [[Prototype]]: Object
```

src > App.js

```
import logo from './logo.svg';
import './App.css';

function App() {
  return (
    <div>
      <div className="AppH1"><img src={logo} className="App-logo2" alt="logo" /></div>
      <h1 className="AppH1">Hello World!</h1>
    </div>
  );
}

export default App;
```

• 직접 만들 컴포넌트들이 모인 루트 컴포넌트

- 애플리케이션의 구조와 라우팅을 정의하는 데 사용
- 모듈 시스템과 모듈 번들링 도구를 활용하여 애플리케이션의 핵심 로직과 스타일을 정의하고, 필요한 자원을 효율적으로 관리
- index.js에서 App.js의 기능을 사용하기 위해 export 키워드를 사용
 - 기본 내보내기(**export default**)를 사용함으로 index.js에서는 중괄호 없이 **import**하여 사용

CRA와 Webpack

- Webpack(웹팩)

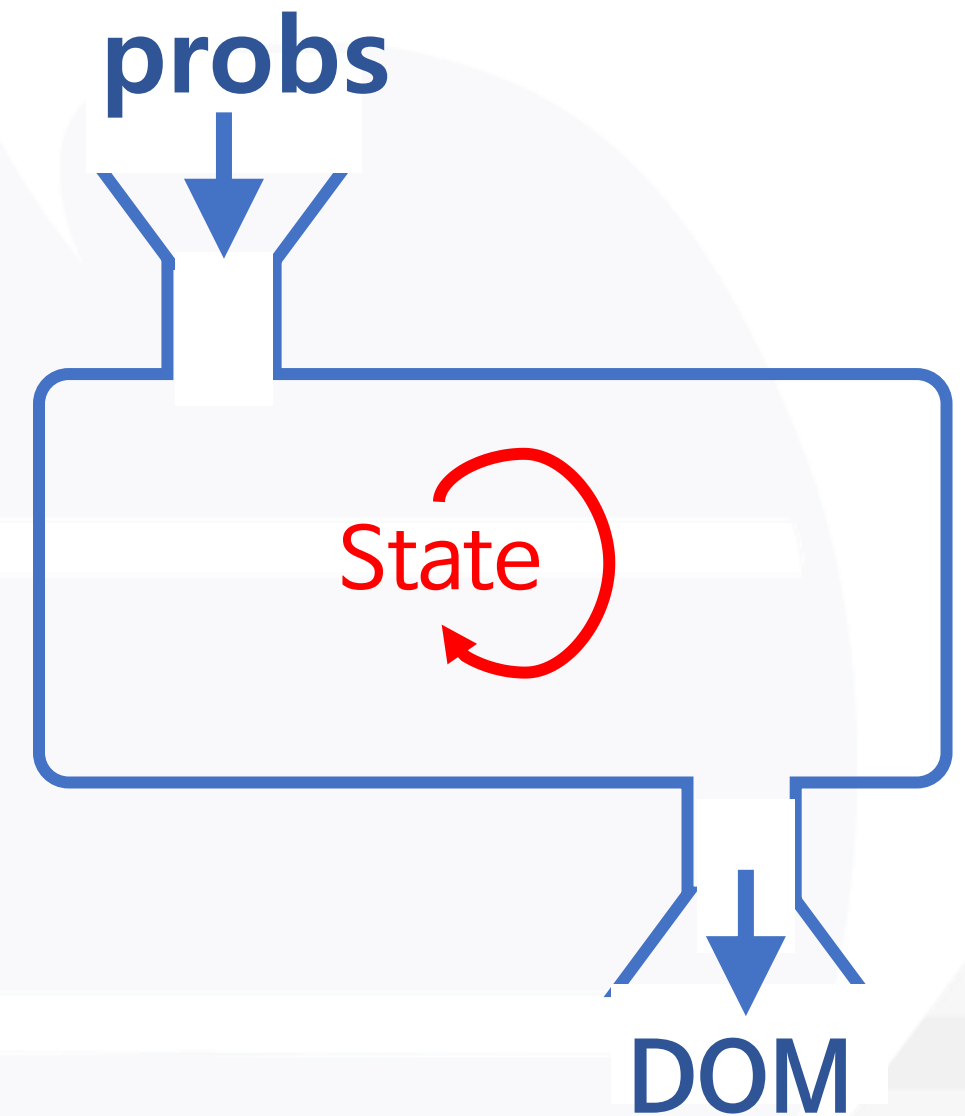
- ECMAScript 2015(ES6)부터 시작된 모던 자바스크립트는 모듈 시스템을 도입하여 코드를 개별적인 단위로 분리하여 재사용성을 높이고, 유지 관리를 용이
- 모듈 번들링을 위해 사용되는 도구
 - 모듈 번들링(Module Bundling) : 웹 애플리케이션을 구성하는 자바스크립트 모듈과 다른 자원(예: CSS, 이미지 파일)을 하나 또는 여러 개의 파일로 합치는 과정

- CRA(create-react-app)와 Webpack

- CRA를 사용하면 Webpack과 같은 빌드 도구가 사전 구성되어 있어서 복잡한 구성 없이 React 애플리케이션을 쉽게 생성, 개발 및 빌드
- index.html과 index.js의 연결도 직접적인 스크립트 링크가 아닌, Webpack 같은 빌드 도구를 통한 번들링 과정을 통해 연결 등 프론트엔드 개발의 기본적인 설정을 자동으로 처리
- npm start, npm run build, npm run test와 같은 명령어로 개발, 빌드, 테스트를 간단히 할 수 있게 해 줌

컴포넌트(Component)

- UI를 재사용 가능한 개별적인 조각으로 사용자 정의 태그 생성
 - props라고 하는 임의의 입력을 받은 후, 화면에 어떻게 표시되는지를 기술하는 React 엘리먼트를 반환
 - 컴포넌트는 반드시 하나의 요소를 반환
 - 여러 요소가 있다면 프래그먼트로 감싸서 반환(`<>...</>`)
 - 컴포넌트명은 반드시 대문자로 시작해야 함
 - JSX 문법으로 작성
 - JSX내에 자바스크립트 표현식은 `{}`안에 작성
 - 하이픈은 카멜케이스로 표시해야함
 - background-color => backgroundColor
 - class 속성은 className으로 사용
 - 태그는 반드시 닫아야 함
 - 주석 : `{/* */}`



src > 01 > Hello.js

Hello.js

```
1 function Hello() {  
2   return(  
3     <h1>Hello React!!</h1>  
4   );  
5 }  
6  
7 export default Hello;
```

App.js

```
1 import logo from './logo.svg';  
2 import './App.css';  
3 import Hello from './01/Hello';  
4  
5 function App() {  
6   return (  
7     <div className="App">  
8       <header className="App-header">  
9         <img src={logo} className="App-logo" alt="logo" />  
10        <Hello />  
11      </header>  
12    </div>  
13  );  
14 }  
15  
16 export default App;
```

• 함수형 컴포넌트

– 자바스크립트 함수로 작성

– return문에 JSX 코드를 작성하여 반환

• JSX는 웹 브라우저에 보일 UI 조각인 html 요소

JSX(JavaScript XML)

- **JavaScript의 확장 문법**

- HTML과 유사한 문법을 사용하여 리액트 요소를 생성
- 공식 자바스크립트 문법은 아니지만, 리액트에서 UI 컴포넌트를 직관적으로 표현하는 데 널리 활용
- 브라우저가 실행하기 전에 코드가 번들링되는 과정에서 바벨을 사용하여 일반 자바스크립트 형태의 코드로 변환

JSX(JavaScript XML) 문법

- 단일 루트 요소 반환
 - 여러 개의 요소를 반환하고 싶다면 fragments라 불리는 `<></>` 를 사용
- 자바스크립트 표현식은 중괄호(`{}`) 내에 위치
- 자바스크립트 예약어와 같은 속성명을 사용할 수 없음
 - class 속성은 `className`, for 속성은 `htmlFor`
 - 속성명은 카멜표기법을 사용
- 스타일 적용
 - 스타일 이름을 카멜표기법으로 사용
 - 스타일은 오브젝트로 선언하여 표현식으로 작성하거나 인라인 스타일은 `{}` 안에 작성

JSX(JavaScript XML) 문법

- 반드시 종료 태그 작성

- 예) `<Hello />`

- 조건부 렌더링

- 삼항연산자를 이용하여 조건부 렌더링

- & & (AND) 연산자를 이용한 조건부 렌더링

- 특정조건을 만족할때만 내용을 보여주고, 만족하지 않을 때는 렌더링 하지 않는 경우

- || (OR) 연산자를 이용한 조건부 렌더링

- 컴포넌트내부에서 undefined만 반환하면 오류가 발생하므로 **OR**연산자를 사용하여 해당값이 undefind인 경우일때 사용할 값을 지정하여 오류를 방지

JSX 예제

```
export default function HelloDate() {  
  const current = new Date() ;  
  const hours = current.getHours() ;  
  const pstyle = {  
    backgroundColor : "yellow",  
    color : "black"  
  }  
  
  return (  
    <div style={pstyle}>현재시간 : {current.toLocaleTimeString()}</div>  
    <p className="th" style={{color : "blue", backgroundColor: "#FFFFFFF"}}>  
      {hours < 12 ? "오전" : "오후"}  
    </p>  
  </div>  
)  
}
```

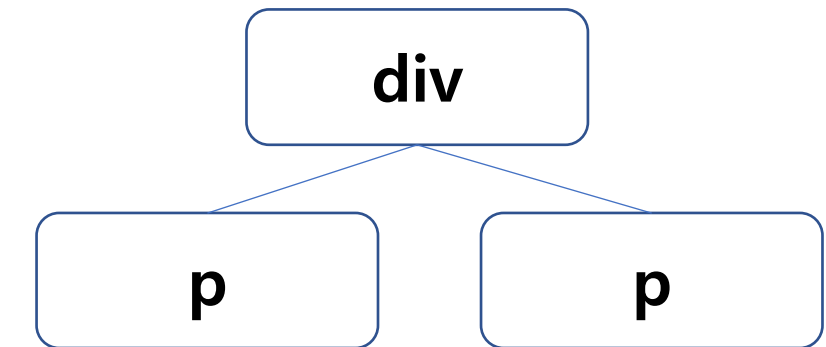
스타일을 오브젝트로 정의

스타일을 표현식으로

예약어는 사용할 수 없음

조건부 렌더링

스타일을 표현식으로 오브젝트로 정의



해결문제

