

# VS / node / react 설치

## 1 VS 코드 설치 (설치 전 자리에 있는지 확인)

<https://code.visualstudio.com/docs/?dv=win64user>

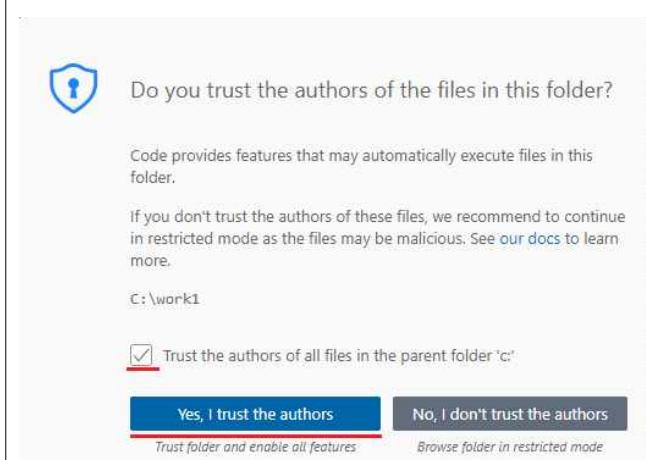
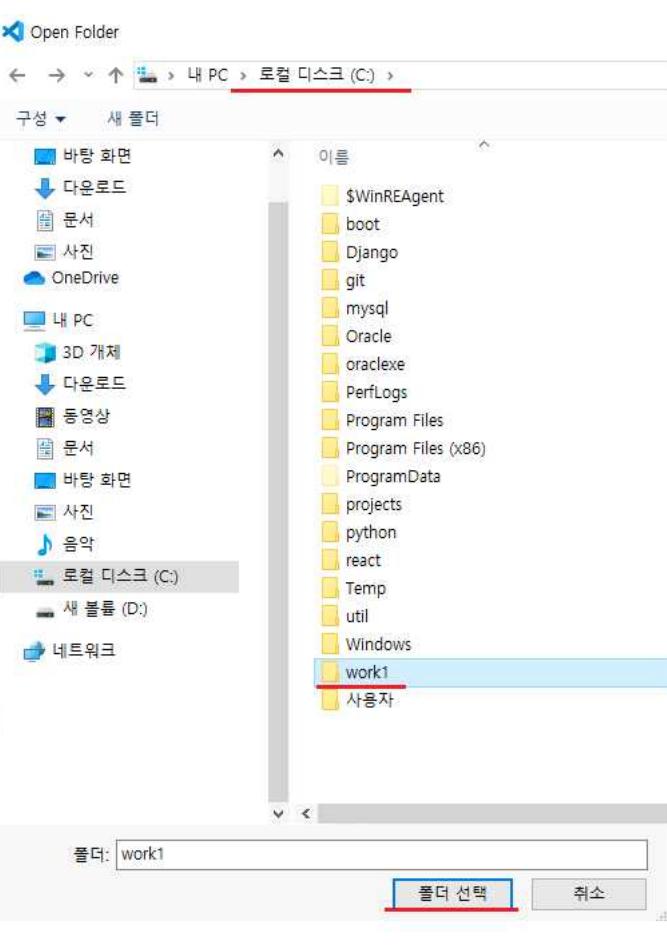
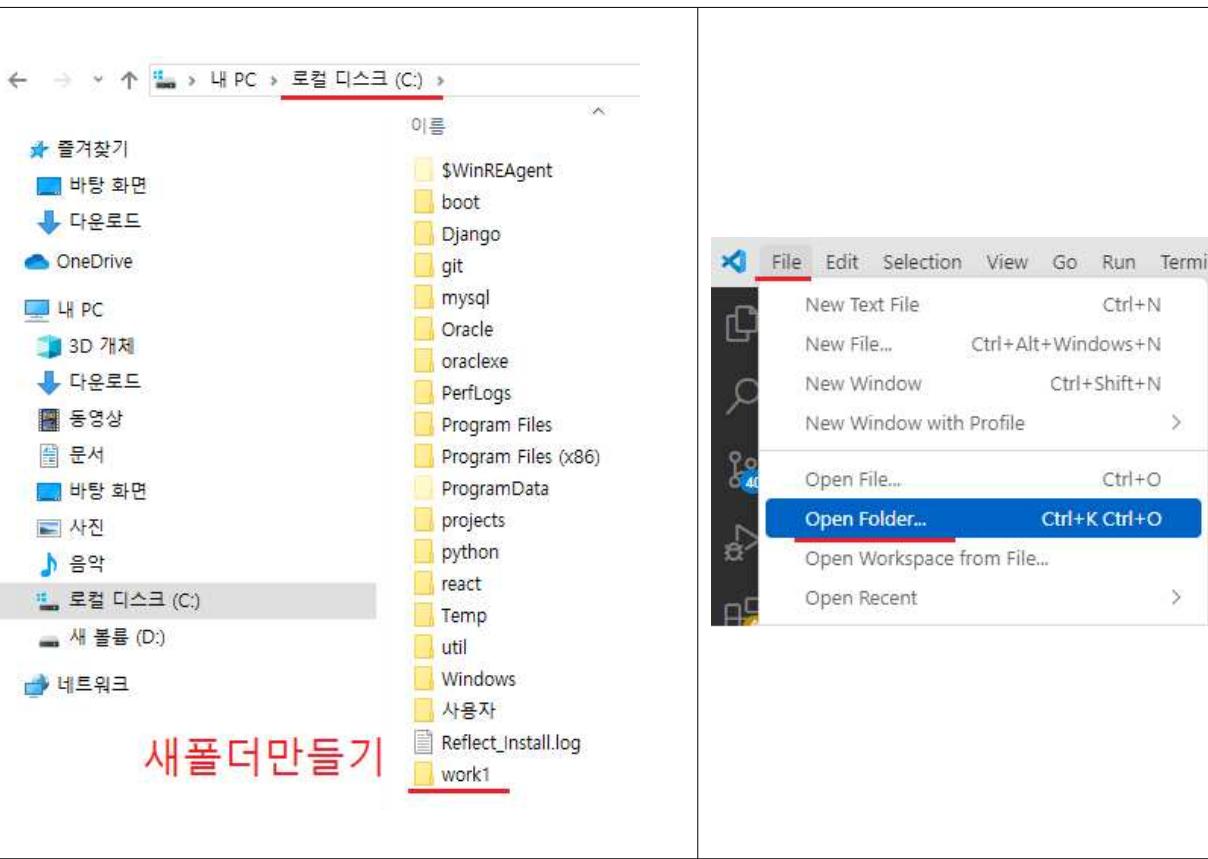
The screenshot shows the Visual Studio Code documentation page. At the top, there is a navigation bar with links for Documentation for Visual Studio, Visual Studio Code, Docs, Updates, Blog, API, Extensions, FAQ, and GitHub Copilot. Below the navigation bar, there is a search bar with placeholder text "접속하자마자 vs 코드 파일 다운로드 받아짐". To the right of the search bar are icons for a magnifying glass and a download button labeled "Download". A red box highlights the "Download" button with the text "안된다면 버튼 클릭". In the center of the page, there is a large callout box with the text "Thanks for downloading VS Code!" and a link "Download not starting? Try this [direct download link](#)". On the left side of the page, there is a sidebar with links for Overview, SETUP, GETTING STARTED, USER GUIDE, SOURCE CONTROL, TERMINAL, GITHUB COPILOT, LANGUAGES, NODEJS / JAVASCRIPT, TYPESCRIPT, PYTHON, JAVA, C++, and C#. The main content area features a section titled "Visual Studio Code documentation" with a sub-section "Getting started". It includes two cards: "Set up Visual Studio Code" (with a wrench icon) and "Getting started" (with a rocket icon). Both cards have descriptive text below them.

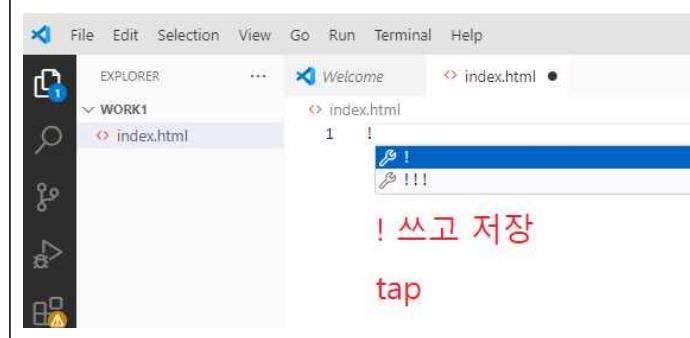
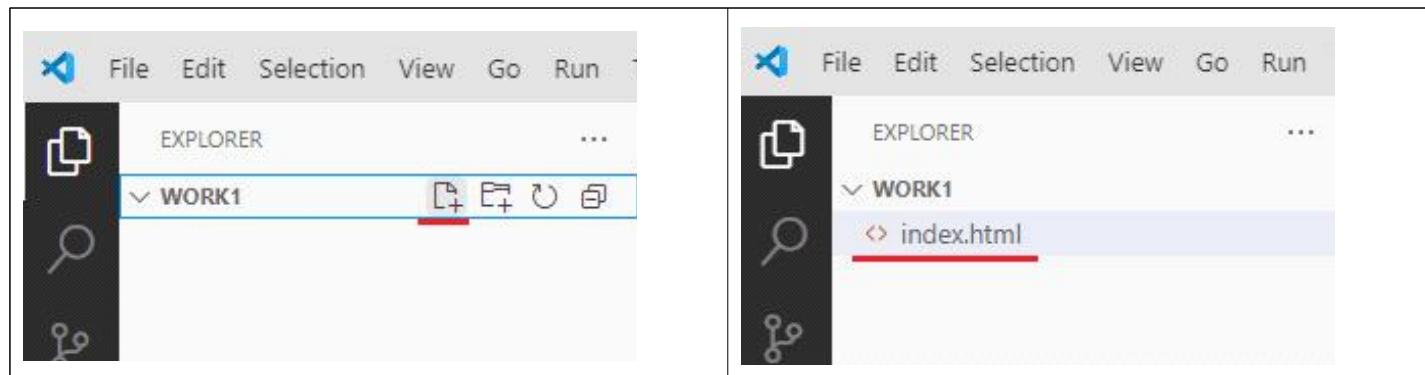
# 1 - 3 비주얼스튜디오 확장 프로그램 깔기

비주얼 스튜디오에서	<p>3 설치</p>
Open in browser	Alt + B
Live Server	Live Server → 실시간으로 브라우저에서 적용된 내용을 확인 (Alt + B)
auto rename tag	태그의 시작/종료를 동시에 수정
prettier	코드 자동 정렬 (Ctrl + Shift + B)
css Peek	HTML에서 class에 적용된 CSS 속성으로 바로 이동 (<Ctrl> + <마우스 클릭>)
Auto import	파일명을 입력하면 자동 import
Auto import -es6	
Indent Rainbow	들여쓰기를 색깔로 구분해서 가독성을 높임
Reactjs code snippets	rsc로 함수 만들어줌 (자동완성)
ES7 + React/Redux/React-Native snippets	rfce - 함수만들어줌(자동완성)

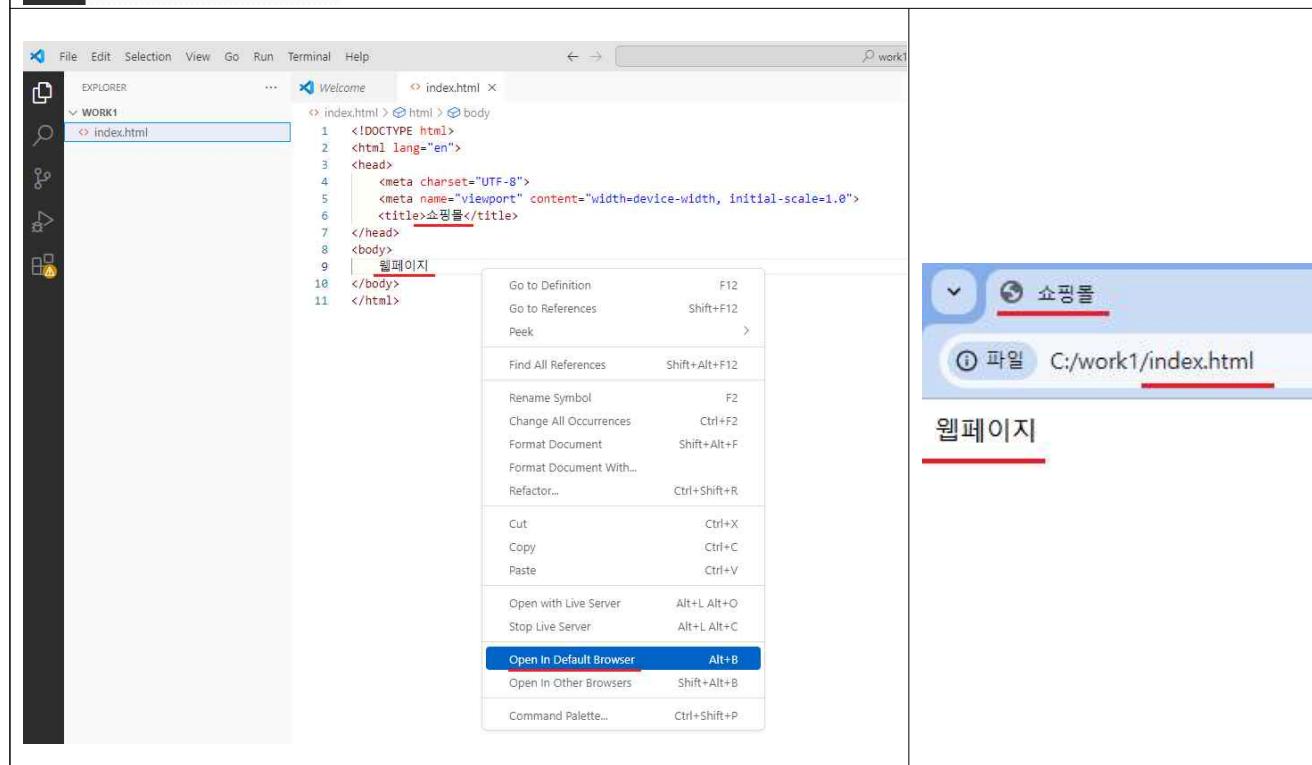
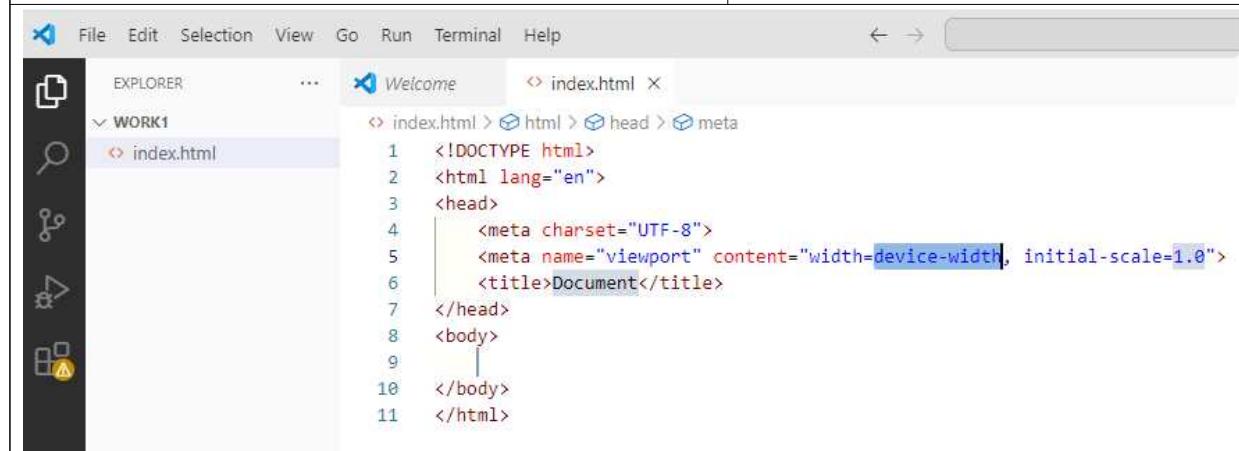
# 1 - 4 비주얼스튜디오 – 설정 (생략)

## 작업폴더만들고 index.html 파일만들기



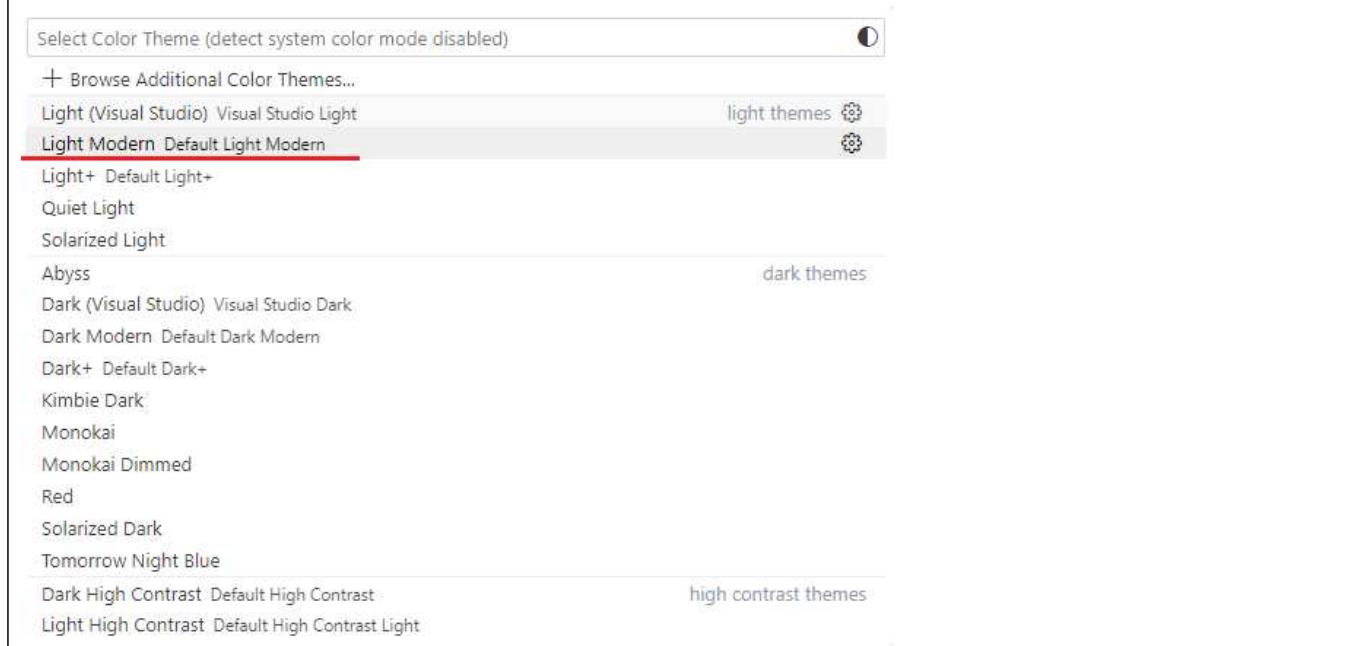
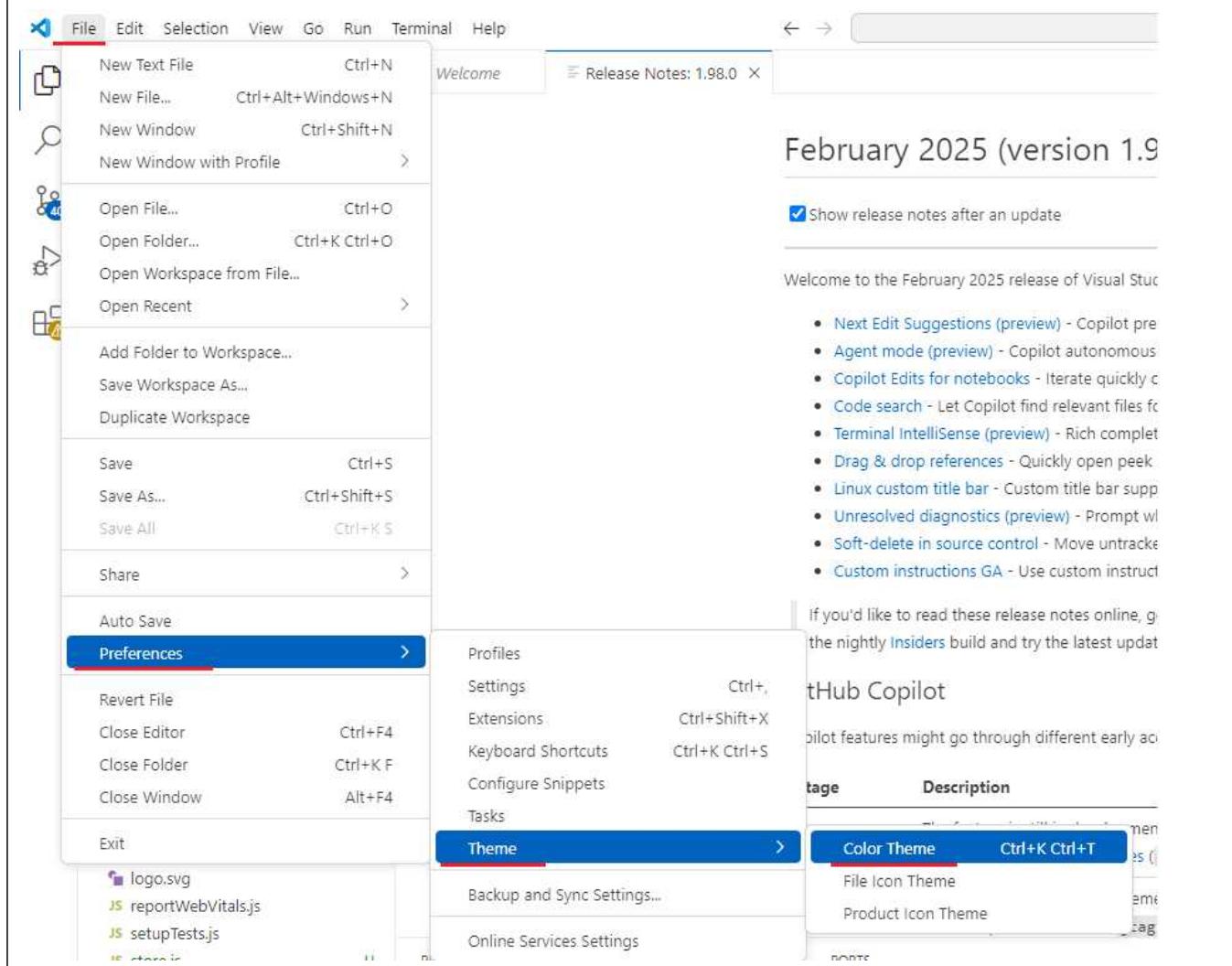


- 파일저장하기 index.html
  - ! + Tab 키



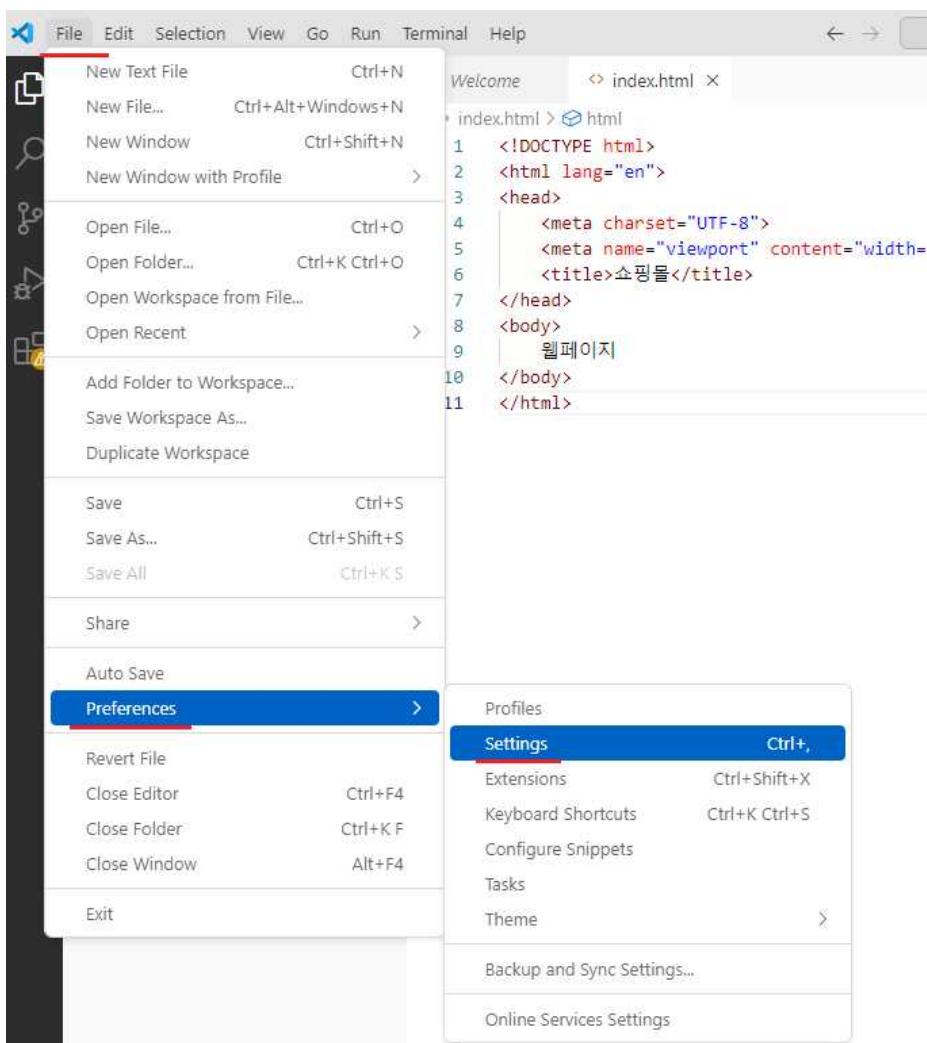
# 비주얼스튜디오 - 밝은 테마 사용하기

## File > Preferences > Theme > Color Theme

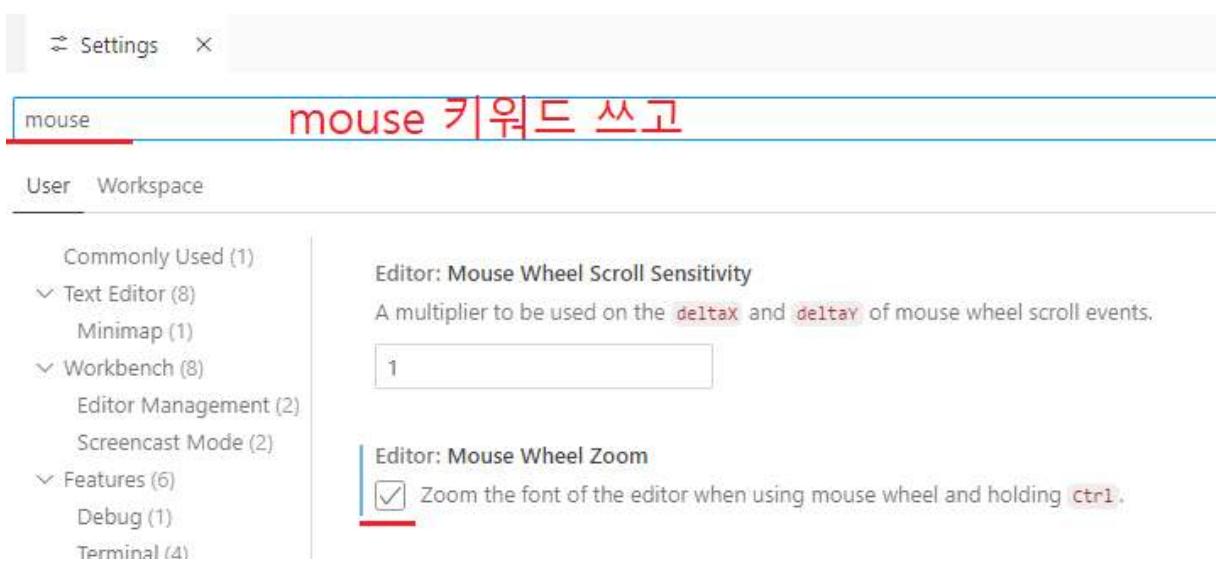


# 비주얼스튜디오 - 마우스휠로 폰트 키우기

File > Preferences > Settings

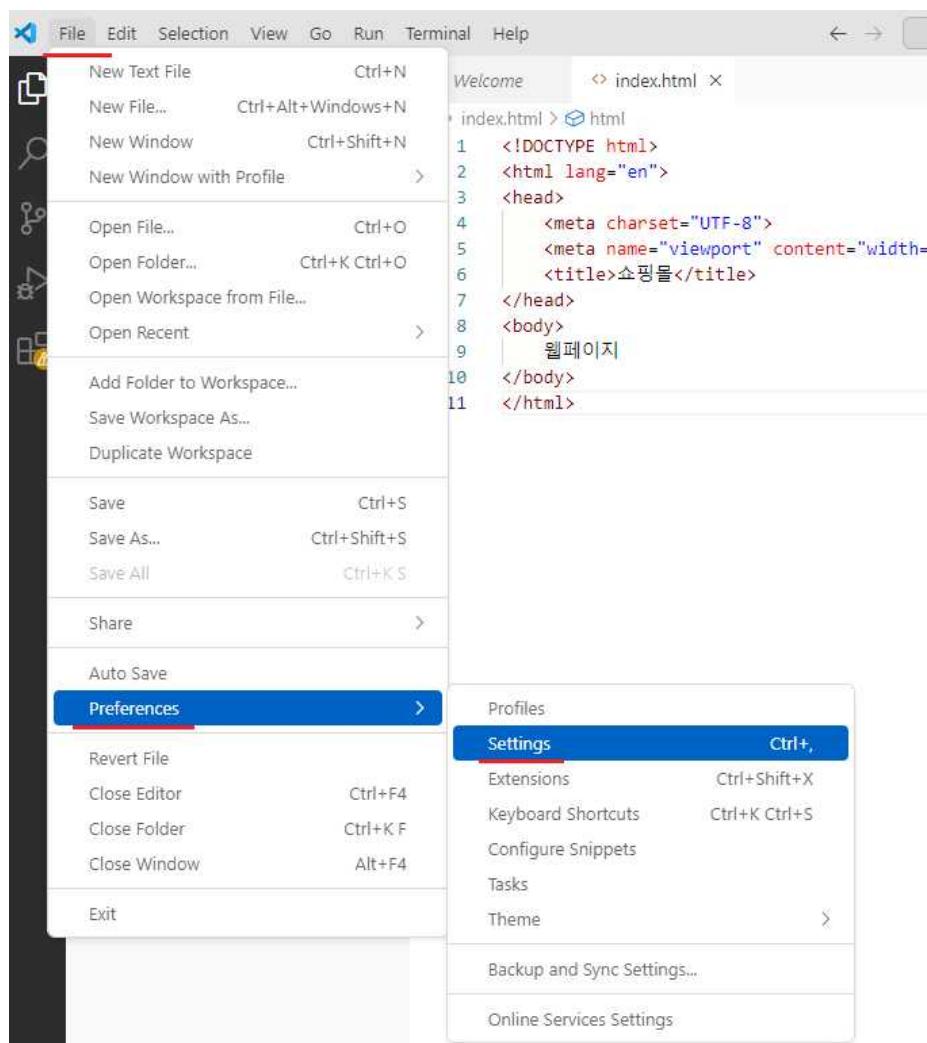


settings > mouse키워드 써주고 > Mouse Wheel Zoom 체크하기

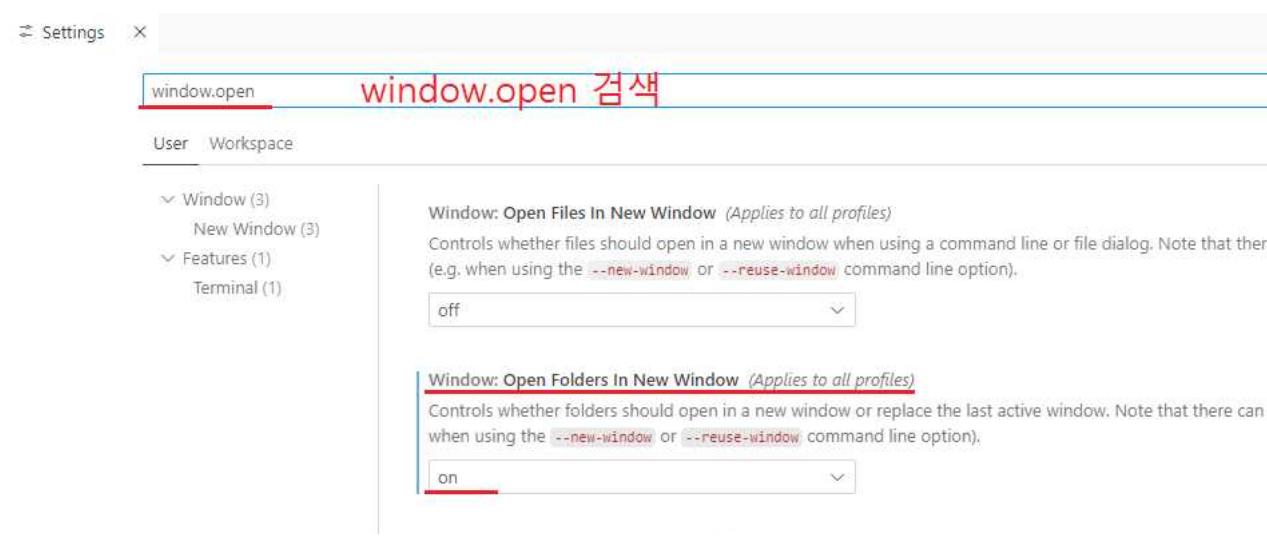


## VSCode에서 파일을 항상 새 창에서 열기, 프로젝트 두 개이상 동시에 작업 설정

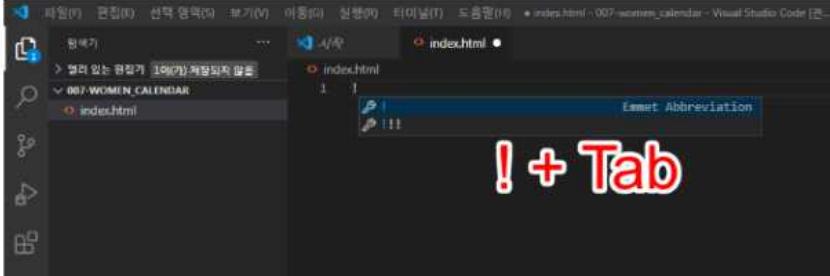
File > Preferences > Settings



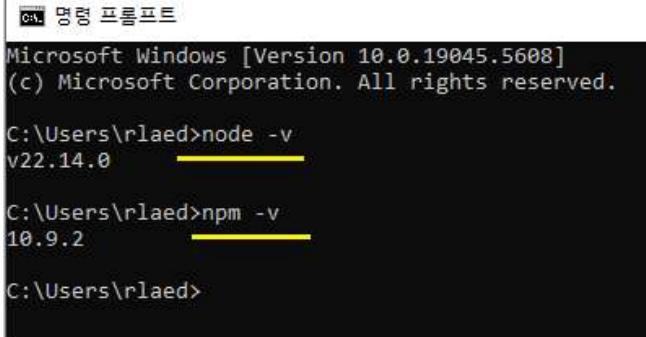
검색 : window : Open Files In New Window 에서 On으로 설정



## 비주얼스튜디오 단축키

html 자동완성하기	1. 파일저장하기 : index.html 2. ! + Tab 키  <b>! + Tab</b> 
코드 정렬하기	1. Ctrl + A 키를 눌러 코드 전체를 모두 선택 2. Ctrl 키를 누른 상태에서 K 와 F 를 차례대로 클릭 (json 정렬도 같음) 3. prettier 코드 자동 정렬 (Ctrl + Shift + B)
브라우저 열기	Alt + B

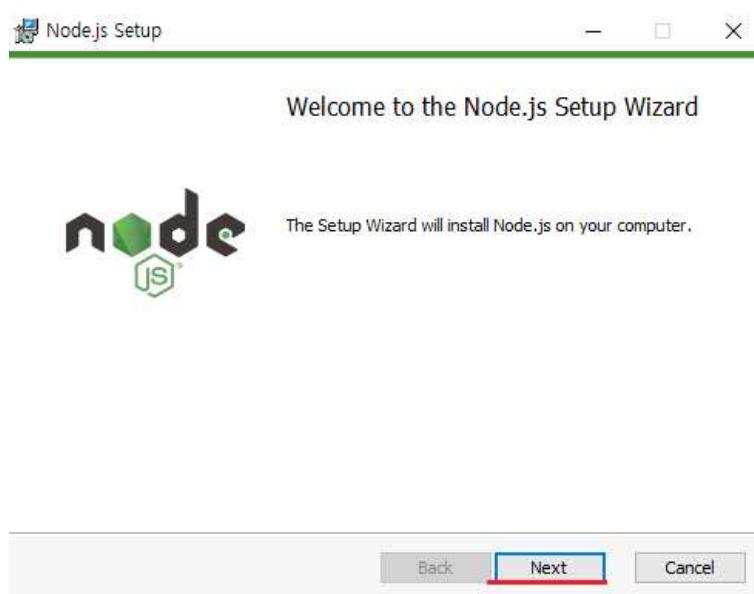
## 2 node 설치 (설치 전 cmd에서 node 있는지 확인, 없으면 설치)

cmd에서 node가 있는지 확인	
<a href="https://nodejs.org/ko/">https://nodejs.org/ko/</a>	

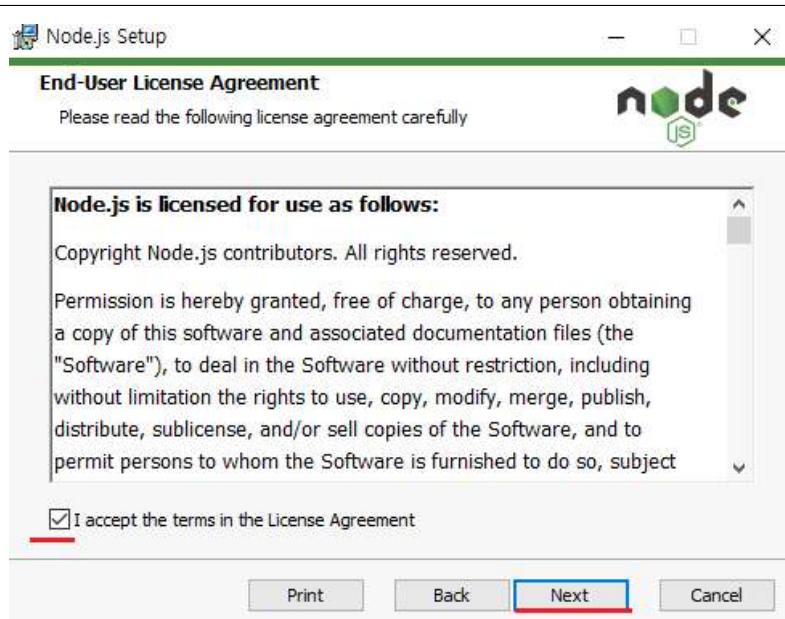
내컴퓨터에서  
다운로드 폴더 확인 후  
설치



설치



동의 체크 설치



설치 끝나고  
cmd에서 확인

node -v  
npm -v

```
C:\ 명령 프롬프트
Microsoft Windows [Version 10.0.19045.5608]
(c) Microsoft Corporation. All rights reserved.

C:\Users\rlaed>node -v
v22.14.0

C:\Users\rlaed>npm -v
10.9.2

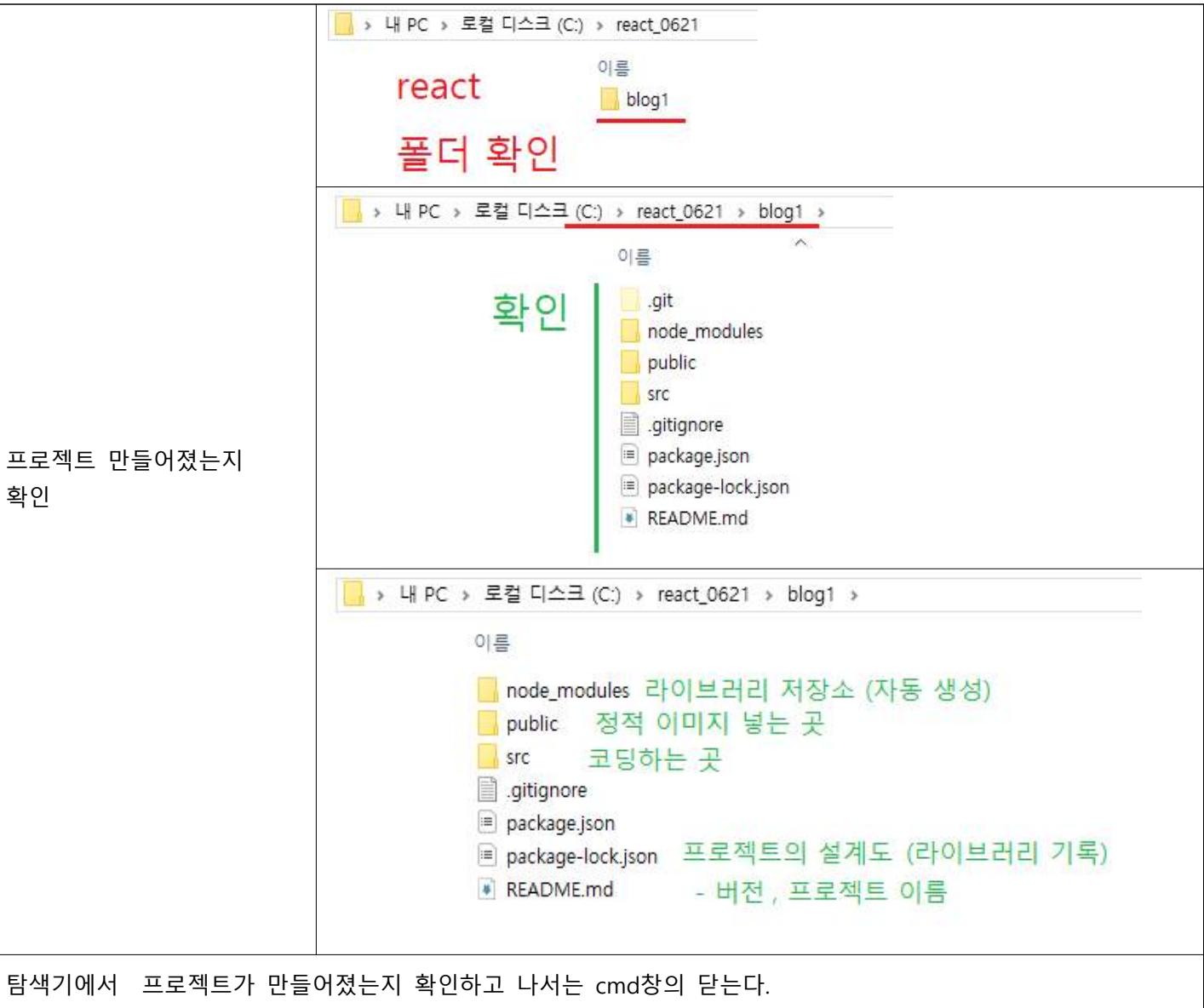
C:\Users\rlaed>
```

### 3 react 프로젝트 만들기

탐색기로  
작업 폴더 만들기

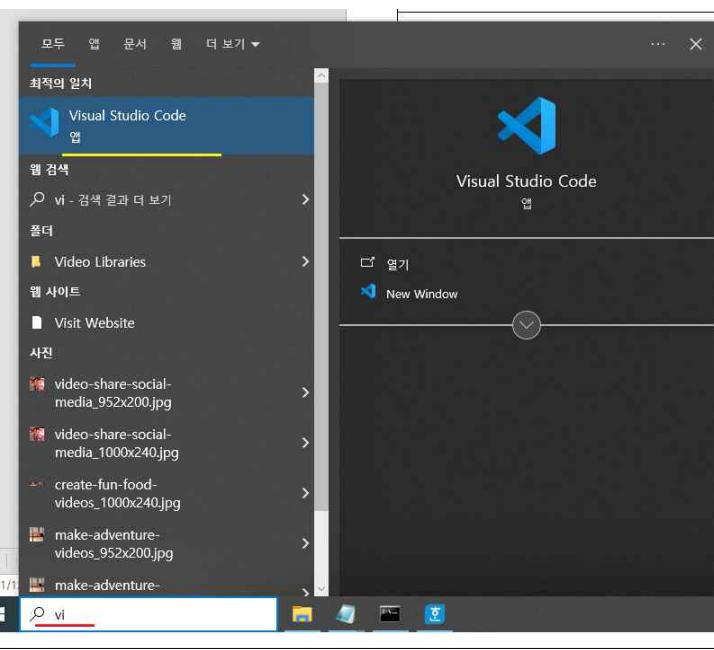
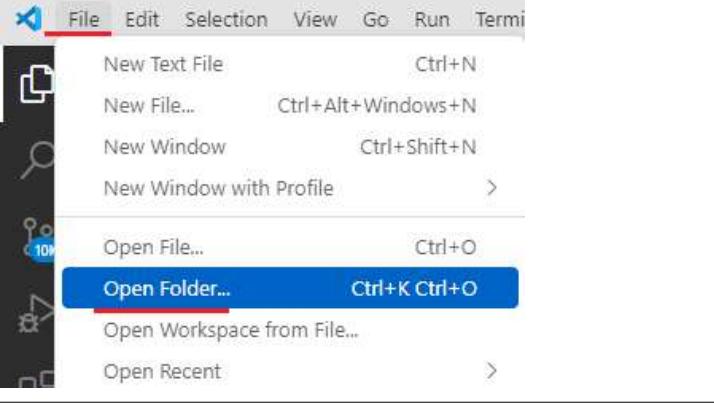
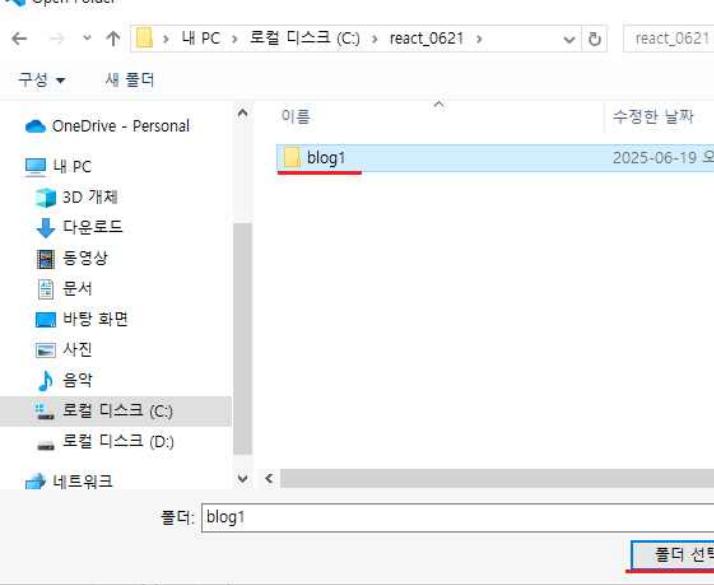
▶ 내 PC ▶ 로컬 디스크 (C:) ▶ react\_0621

	<pre>     관리자: 명령 프롬프트 Microsoft Windows [Version 10.0.19045.5965] (c) Microsoft Corporation. All rights reserved.  C:\Users\Administrator&gt;User -2025VRCIA&gt;cd C:\react_0621 C:\react_0621&gt;npx create-react-app blog1 Creating a new React app in C:\react_0621\blog1.  Installing packages. This might take a couple of minutes. Installing react, react-dom, and react-scripts with cra-template...  added 1324 packages in 38s  269 packages are looking for funding   run `npm fund` for details  Initialized a git repository.  Installing template dependencies using npm... added 18 packages, and changed 1 package in 4s  269 packages are looking for funding   run `npm fund` for details Removing template package using npm...  removed 1 package, and audited 1342 packages in 3s  269 packages are looking for funding   run `npm fund` for details    9 vulnerabilities (3 moderate, 6 high)  To address all issues (including breaking changes), run:   npm audit fix --force  Run `npm audit` for details.  Created git commit. 확인  Success! Created blog1 at C:\react_0621\blog1 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. </pre>
위의처럼 안뜨면	리액트 프로젝트가 생성되지 않으면 참고 url: <a href="https://blog.naver.com/ysboo2/223757816691">https://blog.naver.com/ysboo2/223757816691</a>
cmd에서 코드 복사 붙여넣기	cmd > 프로젝트 경로 > 아래코드 붙여넣고 실행 (아래코드 한줄, 한번에 복사) <pre> npm install react@18 react-dom@18 @testing-library/jest-dom@5.17.0 @testing-library/react@13.4.0 @testing-library/user-event@13.5.0 web-vitals </pre> npx create-react-app blog1



탐색기에서 프로젝트가 만들어졌는지 확인하고 나서는 cmd창의 닫는다.

#### 4 react 프로젝트 실행되는지 확인

<p>비주얼 스튜디오 열기</p>	
<p>vi 코드에서 Open Folder</p>	
<p>blog1 폴더 열기</p>	

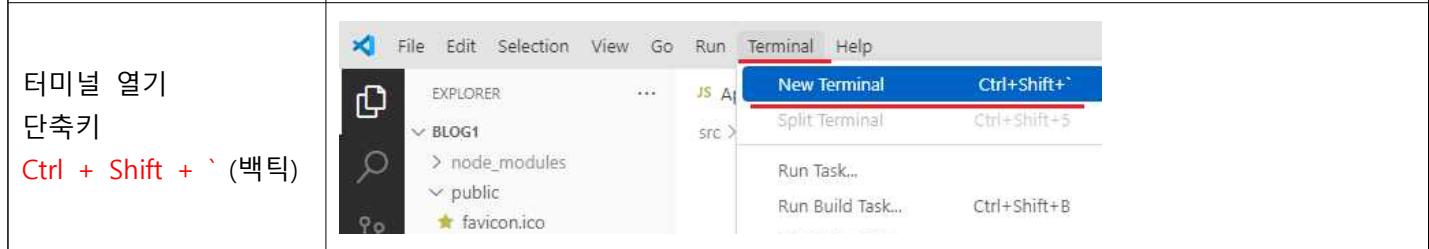
App.js 선택

프로젝트 확인

```

File Edit Selection View Go Run Terminal Help
EXPLORER ... JS App.js X
src > JS App.js ...
1 import logo from './logo.svg';
2 import './App.css';
3
4 function App() {
5   return (
6     <div className="App">
7       <header className="App-header">
8         <img src={logo} className="App-logo" alt="logo" />
9         <p>
10            Edit <code>src/App.js</code> and save to reload.
11        </p>
12        <a
13          className="App-link"
14          href="https://reactjs.org"
15          target="_blank"
16          rel="noopener noreferrer"
17        >
18          Learn React
19        </a>
20      </header>
21    </div>
22  );
23 }
24
25 export default App;
26

```



터미널에서 npm start

프론트 서버 실행

React 앱은 http://localhost:3000 포트에서 실행

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

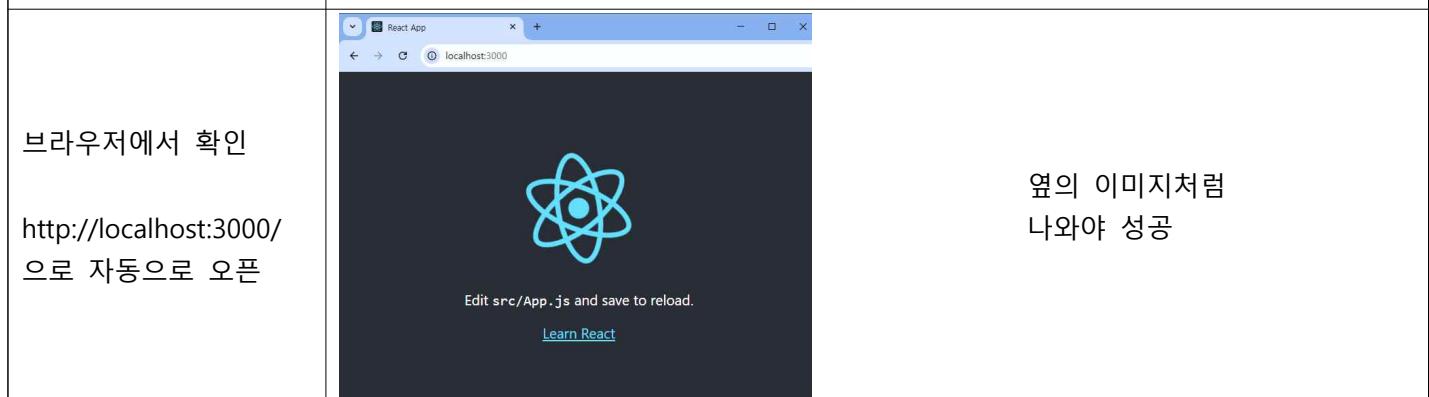
○ Compiled successfully!

You can now view blog1 in the browser. 확인

Local: http://localhost:3000  
On Your Network: http://192.168.0.39:3000

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

webpack compiled successfully



## 리액트 프로젝트가 크롬 브라우저로 안 뜬다면

참고 url	react 프로젝트 기본 브라우저가 크롬이 아닐경우 (크롬 브라우저) <a href="https://blog.naver.com/ysboo2/223841170588">https://blog.naver.com/ysboo2/223841170588</a>
--------	---

### 기본 브라우저가 크롬이 아닐경우

윈도우 창에서 검색 기본앱	
기본앱 아래쪽에 웹브라우저 > 앱선택 > 구글 크롬 선택	
크롬으로 열리는지 확인	

## App.js > 작업 header를 다 지우고 > 123만 작성 > 터미널에서 npm start > 브라우저에서 확인

File Edit Selection View Go Run Terminal Help

EXPLORER BLOG1

- node\_modules
- public
  - favicon.ico
  - index.html
  - logo192.png
  - logo512.png
  - manifest.json
  - robots.txt
- src
  - App.css
  - App.js M
  - App.test.js
  - index.css
  - index.js
  - logo.svg
  - reportWebVitals.js
  - setupTests.js
- .gitignore
- package-lock.json
- package.json
- README.md

JS App.js M ×

```

1 import logo from './logo.svg';
2 import './App.css';
3
4 function App() {
5   return (
6     <div className="App">
7       | 123
8     </div>
9   );
10 }
11
12 export default App;
13

```

React App localhost:3000

123

File Edit Selection View Go Run Terminal Help

EXPLORER BLOG1

- node\_modules
- public
  - favicon.ico
  - index.html 3
  - index192.png
  - index512.png
  - manifest.json
  - robots.txt
- src
  - App.css
  - App.js 1 M
  - App.test.js
  - index.css
  - index.js 2
  - logo.svg
  - reportWebVitals.js
  - setupTests.js
- .gitignore
- package-lock.json
- package.json
- README.md

JS App.js M × 1

JS index.html × 3

JS index.js × 2

```

1 import logo from './logo.svg';
2 import './App.css';
3
4 function App() {
5   return (
6     <div className="App">
7       | 123
8     </div>
9   );
10 }
11
12 export default App;
13

```

```

<!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" />
<!--
  manifest.json provides metadata used when your web app is installed on a user's mobile device or desktop. See https://developers.google.com/web/fundamentals/
-->
<link rel="manifest" href="%PUBLIC_URL%/manifest.json" />
<!--
  Notice the use of %PUBLIC_URL% in the tags above.
  It will be replaced with the URL of the 'public' folder during the build.
  Only files inside the 'public' folder can be referenced from the HTML.
  Unlike "/favicon.ico" or "favicon.ico", "%PUBLIC_URL%/favicon.ico" will work correctly both with client-side routing and a non-root public URL.
  Learn how to configure a non-root public URL by running `npm run build`.
-->
<title>React App</title>
</head>
<body>
  <noscript>You need to enable JavaScript to run this app.</noscript>
  <div id="root"></div>
<!--
  This HTML file is a template.
  If you open it directly in the browser, you will see an empty page.
  You can add webfonts, meta tags, or analytics to this file.
-->

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

On Your Network: http://192.168.0.39:3000

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

webpack compiled successfully  
PS C:\react\_0621\blog1> ^C  
Compiled with warnings.

eslint

React 구조	React는 public/index.html이라는 HTML 템플릿 하나만 있고, 진짜 화면 내용은 src/App.js처럼 자바스크립트(컴포넌트)로 채워지는 구조	
동작순서	1 index.html	<div id="root"></div> 라는 빈 공간이 있음 (HTML 템플릿)
	2 index.js	root라는 그 빈 공간을 찾아서, App 컴포넌트를 거기다가 넣어줌 <code>const root = ReactDOM.createRoot(document.getElementById('root'));</code> <code>root.render(&lt;App /&gt;);</code>
	3 App.js	실제로 보이는 화면 내용을 여기서 만들고 구성함

index.html은 틀이고, 실제 내용은 App.js부터 컴포넌트로 구성

### 그림으로 이해 1

index.html contains an empty <div>  
with the id “root”

#### 1. index.html

```
<div id="root">/div>
```



index.js finds the root and renders the  
App component in it

#### 2. index.js

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



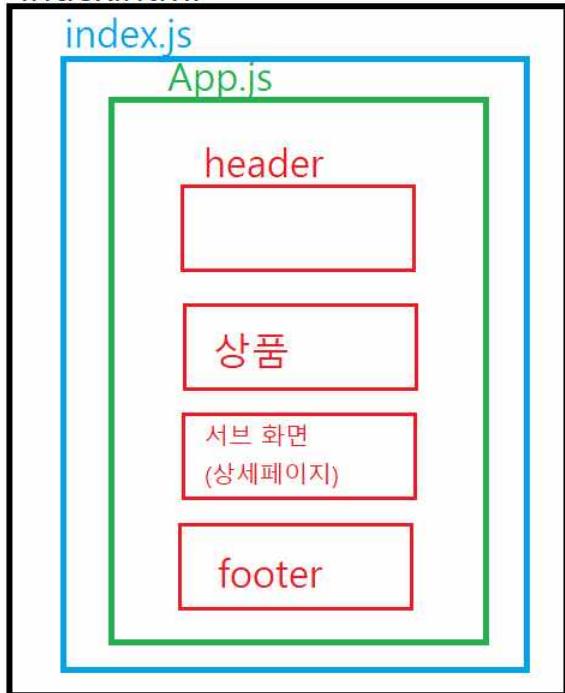
App.js component is created in root

#### 3. App.js

### 그림으로 이해 2

기존처럼 html을 여러개 만드는게 아니라 하나의 html안에 한페이지에 컴포넌트로 채움(SPA)

index.html



App.js에서  
모든 컴포넌트를 다 넣고 SPA

조건이 되면  
컴포넌트가 보였다 안보였다 제어한다.

App.js는 모든 컴포넌트의 중심이고,  
조건(if, useState, 라우터)에 따라 화면을 바꿔가며 보여주는 SPA 구조

SPA의 단점	문제는 서버에서 렌더링하면 검색이 잘되는데 리액트로 만든 SPA는 검색에 노출이 잘 안되어서 Next.js를 많이 사용한다. Next.js는 TypeScript가 기본이다
	일반적인 React(SPA)는 모든 화면을 브라우저에서(JavaScript로) 그리는 방식이다. 그래서 검색 엔진(구글, 네이버)이 페이지 내용을 제대로 읽지 못할 수 있다. 반면에 Next.js는 페이지를 서버에서 미리 만들어서(SSR) 브라우저에 보내준다. 그래서 검색 엔진에도 잘 노출되고, SEO(검색 최적화)에 유리하다. 또한, Next.js는 기본 설정에 TypeScript도 쉽게 쓸 수 있게 되어 있어서, 대규모 프로젝트나 팀 개발에 많이 사용되고 있어요.
SPA	Single Page Application 하나의 페이지에서 모든 기능이 동작하는 앱 처음에 HTML 하나만 로딩하고, 이후에는 페이지를 다시 새로고침하지 않고 필요한 부분만 JavaScript로 바꿔서 보여주는 방식 한 페이지 안에서 필요한 화면만 바꿔 보여주는 웹앱
SSR	Server Side Rendering (서버 사이드 렌더링) 사용자가 요청하면 서버가 HTML을 만들어 보내줌 검색엔진이 쉽게 읽을 수 있어 SEO에 유리함
SEO	Search Engine Optimization 검색 엔진에 잘 노출되게 만드는 기술, 검색엔진 최적화 Search → 검색 Engine → 검색엔진 (구글, 네이버 등) Optimization → 잘 보이도록 최적화

## 리액트에서 함수 만드는 방법 – 대문자로 컴포넌트를 만들고 src폴더 아래 > Dog.js 만들기

### 함수표현식 1 (변수에 담아쓰는 방법)

The screenshot shows two views of a VS Code interface. On the left, the Explorer sidebar shows a file tree with a folder named 'src' containing 'Dog.js'. The right pane shows the code editor with the following content:

```
const Dog = () => {
  return (
    <div>
      강아지
    </div>
  );
};

export default Dog;
```

A red box highlights the word 'Dog' in the first line of the code. The text '함수표현식' and '(변수에 저장하는방식)' is overlaid on the right side.

### 함수선언식2 – 타입스크립트에서는 함수선언식을 선호 (호이스팅)

The screenshot shows two views of a VS Code interface. On the left, the Explorer sidebar shows a file tree with a folder named 'src' containing 'Cat.js'. The right pane shows the code editor with the following content:

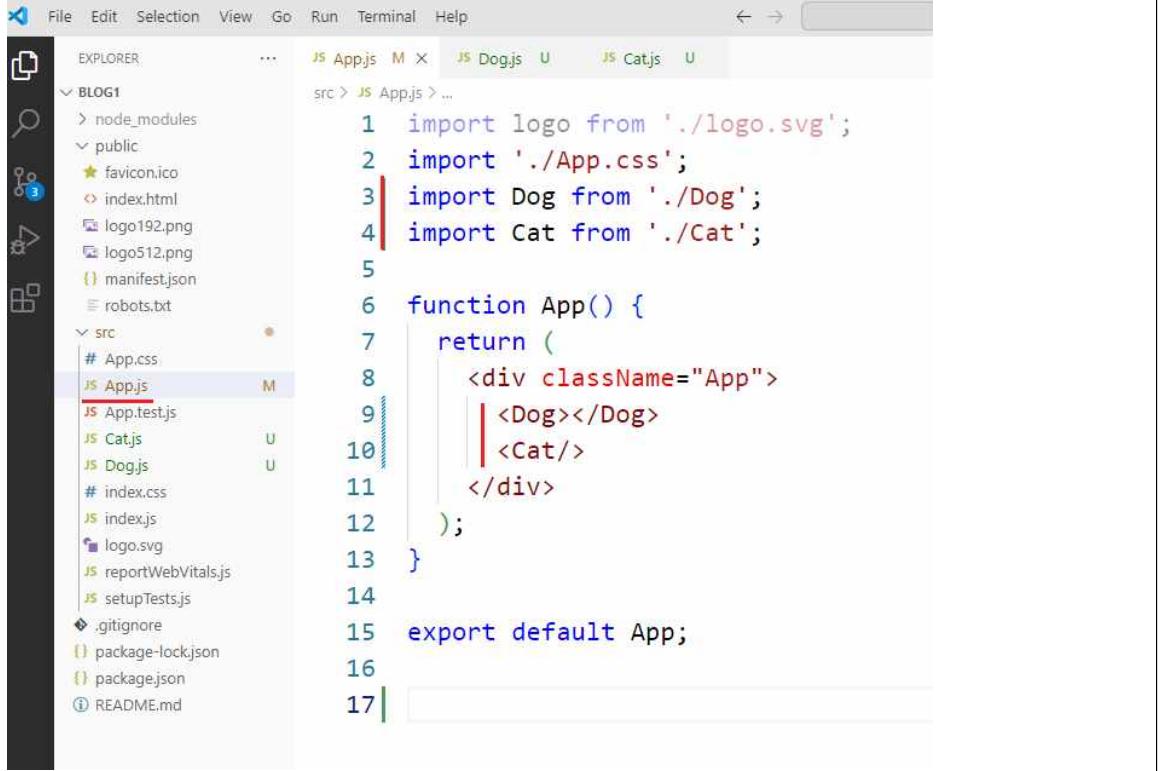
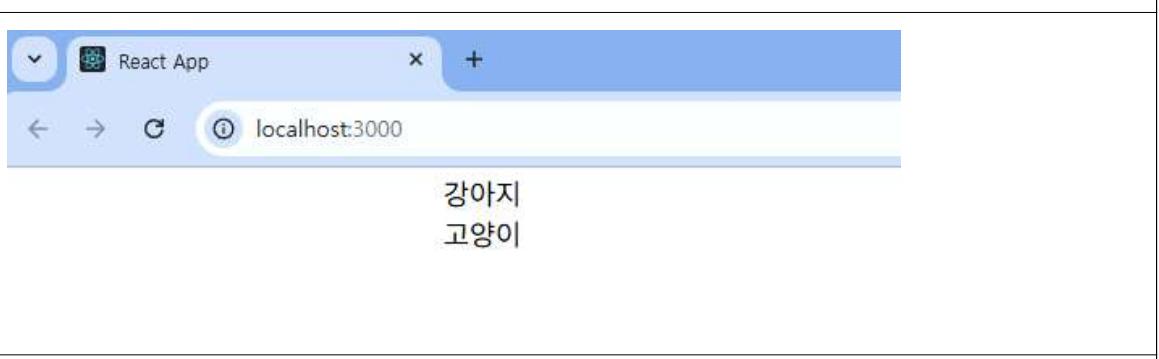
```
import React from 'react';

function Cat() {
  return (
    <div>
      고양이
    </div>
  );
}

export default Cat;
```

A red box highlights the word 'Cat' in the first line of the code. The text '함수선언식' and '타입스크립트쓸때는 함수선언식을 선호 (호이스팅)' is overlaid on the right side.

## App.js 에 컴포넌트 가져오기

App.js	 <pre> File Edit Selection View Go Run Terminal Help EXPLORER ... JS App.js M X JS Dog.js U JS Cat.js U src &gt; JS App.js &gt; ... 1 import logo from './logo.svg'; 2 import './App.css'; 3 import Dog from './Dog'; 4 import Cat from './Cat'; 5 6 function App() { 7   return ( 8     &lt;div className="App"&gt; 9       &lt;Dog&gt;&lt;/Dog&gt; 10      &lt;Cat/&gt; 11    &lt;/div&gt; 12  ); 13} 14 15 export default App; 16 17 </pre>
App.js	<pre> import './App.css'; import Dog from './Dog'; import Cat from './Cat';  function App() {   return (     &lt;div className="App"&gt;       &lt;Dog&gt;&lt;/Dog&gt;       &lt;Cat/&gt;     &lt;/div&gt;   ); } export default App; </pre>
터미널	npm start
브라우저 확인	 <p>강아지 고양이</p>
터미널 프론트 서버 종료	Ctrl + C



**article**

**section**

**NEW**

**Discovery 5**

랜드로버가 만든 청룡 SUV, 디스커버리의 신형 모델이 다음 달 공개된다.

[시승 예약 하기](#)

**시승후기**

믿고 타는 LAND ROVER

[시승 예약 하기](#)

[예약 하기](#)

**이달의 추천 모델**

**Discovery 43% 할인**

[구입하기](#)

**BUILD YOUR OWN**

- 내가 맞는 랜드로버 구성하기
- 모든 각도에서 보는 내 차
- 구성 내용 저장하기
- 내 랜드로버 공유하기

**랜드로버 모델 살펴보기**

**고객지원**

랜드로버만들기
 experience day
 시승신청

뉴스레터 신청
 전시장 찾기
 더보기

**footer**

Jaguar Land Rover Limited: Registered office: Abbey Road, Whitley, Coventry CV3 4LF. Registered in England No: 1672070

header	헤더를 의미, 머리말
nav	네비게이션을 의미 / 메뉴,로그인 서브메뉴
section	여러 중심 내용을 감싸는 공간을 의미, 웹컨텐츠들을 그룹으로 묶어주는 역할을 담당
article	글자가 많이 들어가는 부분을 의미, 본문내용 웹페이지 상에서의 실제 내용을 의미
figure	이미지사진 등
aside	사이드에 위치하는 공간을 의미 (퀵메뉴, 서브메뉴)
footer	푸터를 의미, 웹사이트의 저작권 정보나 저작권표기

## 시맨틱 – 웹페이지

**Header**

The header features a promotional banner for 'Rocket Delivery' with a deadline of 12 PM today. It includes a search bar, user account options (로그인, 회원가입, 고객센터, 판매자 가입), and navigation links for categories like 쿠팡플레이, 로켓배송, 쿠팡레시피, 다시 구매, biz 쿠팡비즈, 쿠팡직구, 골드박스, 이달의신상, and 판매자특기.

**Sliser**

This page highlights the 'Galaxy Z Fold7 | Z Flip7' with a large image of the phones. A callout box says '오늘 마지막!' (Last day today!). It also shows a '쿠팡캐시+용량 업그레이드' (Coupang Cash + Capacity Upgrade) offer. To the right, there's a sidebar with various product categories and a coupon section.

**Main**

This section displays five food items with their prices and ratings:

제품 이미지	제품 이름	가격	평균 평점	리뷰 수
	맛있는 찰옥수수 미백 냉동 옥수수 특품 10개...	13,090원	★★★★★	(849)
	맛있는 찰옥수수 냉동 옥수수 10개, 옥수수종	13,140원	★★★★★	(5,714)
	야심찬 함흥냉면 10인분 + 냉면육수 x 10봉, 1세트...	11,900원	★★★★★	(2,867)
	오늘 담근 출각 김치	17,850원	★★★★★	(165)
	프리미엄갈치 총2kg 30토막, 약 200g(3토막), 5개	14,310원	★★★★★	(11,814)

**전세계 핫딜 로켓직구 글로벌특가**

This section shows five global deals with their prices and ratings:

제품 이미지	제품 이름	가격	평균 평점	리뷰 수
	지금 49% 할인 중 Tiger Pavilion 접이식 텐트형 모기장	6,990원	★★★★★	(37)
	지금 83% 할인 중 품기구산 약 살 막았네 케리형 오븐파토 포도맛 백, 5kg, 200g	3,950원	★★★★★	(1)
	지금 57% 할인 중 자수 경아지 고양이 이미지 커튼 바란스커튼 2종	13,020원	★★★★★	(68)
	지금 11% 할인 중 GNCMEGAMEN 40+ 비타박 프로그램, 30회분, 2개	84,020원	★★★★★	(4)
	지금 49% 할인 중 clever 반려견용 닐은 물다리, 화이트, 1개	17,990원	★★★★★	(12)

**Footer**

The footer contains the Coupang logo, company information, contact details (365고객센터, 1577-7011 (유료)), and legal notices (상호명 및 호스팅 서비스 제공: 쿠팡(주), 대표이사: 박대준, 서울시 송파구 송파대로 570, 사업자 등록번호: 120-88-00767, 통신판매업신고: 2017-서울송파-0680, 사업자정보 확인), and terms and conditions (채우지급보증 안내).

## 컴포넌트를 이용한 웹 페이지 – App.js

```

File Edit Selection View Go Run Terminal Help
EXPLORER ... App.js M ...
src > App.js > ...
1 import Header from './components/Header';
2 import Slider from './components/Slider';
3 import Main from './components/Main';
4 import Footer from './components/Footer';
5 function App() {
6   return (
7     <>
8       <Header></Header>
9       <Slider></Slider>
10      <Main/>
11      <Footer/>
12    </>
13  );
14}
15
16
17 export default App;
18

```

폴더	파일
components	Header.js Slider.js Main.js Footer.js

**Header.js**

단축키 rsc

```

import React from 'react';

const Header = () => {
  return (
    <div>
      헤더
    </div>
  );
}

export default Header;

```

**Slider.js**

단축키 rsc

```

import React from 'react';

const Slider = () => {
  return (
    <div>
      슬라이더
    </div>
  );
}

export default Slider;

```

**Main.js**

단축키 rfce

```

import React from 'react';

function Main() {
  return (
    <div>
      메인
    </div>
  )
}

export default Main;

```

**Footer.js**

단축키 rfce

```

import React from 'react';

function Footer() {
  return (
    <div>
      풋터
    </div>
  )
}

export default Footer;

```

**App.js**

```

import Header from './components/Header';
import Slider from './components/Slider';
import Main from './components/Main';
import Footer from './components/Footer';

function App() {
  return (
    <>
      <Header></Header>
      <Slider></Slider>
      <Main/>
      <Footer/>
    </>
  );
}

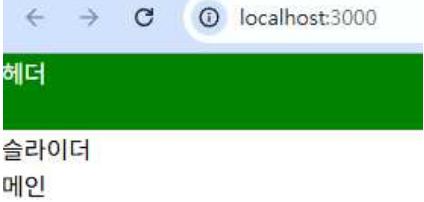
export default App;

```

localhost:3000

헤더  
슬라이더  
메인  
풋터

## Header.js 컴포넌트에 Style 입하기 – 1 인라인 스타일 style={} 머스테치(Mustache)

<p>EXPLORER</p> <p>... Header.js</p> <p>BLOG1</p> <ul style="list-style-type: none"> <li>&gt; node_modules</li> <li>&gt; public</li> <li>&gt; src           <ul style="list-style-type: none"> <li>components               <ul style="list-style-type: none"> <li>Cow.js</li> <li>Footer.js</li> <li><b>Header.js</b></li> <li>Main.js</li> <li>Slider.js</li> </ul> </li> <li>App.css</li> <li>App.js</li> <li>App.test.js</li> <li>Cat.js</li> <li>Dog.js</li> </ul> </li> </ul>	<pre> src &gt; components &gt; Header.js &gt; ... 1 import React from 'react'; 2 3 const Header = () =&gt; { 4   return ( 5     &lt;div style={{width:"100%", color:"white", height:"50px", backgroundColor:"green"}&gt; 6       헤더 7     &lt;/div&gt; 8   ); 9 } 10 11 export default Header; 12 13 </pre> <p style="color: red;">코드작성</p> <p style="color: green;">카멜기법</p>
<p>components &gt; Header.js</p>	<pre> import React from 'react';  const Header = () =&gt; {   return (     &lt;div style={{width:"100%", color:"white", height:"50px", backgroundColor:"green"}&gt;       헤더     &lt;/div&gt;   ); };  export default Header; </pre>
<p>터미널 확인 npm start</p>	 <p>localhost:3000</p> <p>헤더</p> <p>슬라이더</p> <p>메인</p> <p>풋터</p>
<p>style={{ }} 머스테치(Mustache)</p>	<p>JavaScript 객체를 표현하는 JSX 문법 바깥 { ... } → JSX 안에서 JavaScript 표현식을 넣기 위한 괄호 안쪽 { ... } → JavaScript의 객체 리터럴 (스타일 정의용) JSX 안에 객체를 넣는다.</p>

## slider.js 컴포넌트에 Style 입하기 – 2 props로 스타일 적용하기

EXPLORER

- BLOG1
- node\_modules
- public
- src
  - components
    - Cow.js
    - Footer.js
    - Header.js
    - Main.js
    - Slider.js
  - App.css
  - App.js
  - App.test.js
  - Cat.js
  - Dog.js
  - index.css
  - index.js
  - logo.svg
  - reportWebVitals.js
  - setupTests.js

Header.js U App.js M X

```

1 import Header from './components/Header';
2 import Slider from './components/slider';
3 import Main from './components/Main';
4 import Footer from './components/Footer';
5 function App() {
6     return (
7         <>
8             <Header/>
9             <Slider style={{width:"100%", color:"white", height:"200px", backgroundColor:"gold"}}></Slider>
10            <Main/>
11            <Footer/>
12        </>
13    );
14 }
15
16
17 export default App;

```

코드추가

components >  
App.js

```

import Header from './components/Header';
import Slider from './components/Slider';
import Main from './components/Main';
import Footer from './components/Footer';
function App() {
    return (
        <>
            <Header/>
            <Slider style={{width:"100%", color:"white", height:"200px", backgroundColor:"gold"}}></Slider>
            <Main/>
            <Footer/>
        </>
    );
}
export default App;

```

EXPLORER

- BLOG1
- node\_modules
- public
- src
  - components
    - Cow.js
    - Footer.js
    - Header.js
    - Main.js
    - Slider.js
  - App.css
  - App.js
  - App.test.js
  - Cat.js

Slider.js U ●

```

1 import React from 'react';
2
3 const Slider = (props) => {
4     return (
5         <div style={props.style}>
6             슬라이더
7         </div>
8     );
9 }
10
11 export default slider;

```

코드추가

Slider.js

```

import React from 'react';

const Slider = (props) => {
    return (
        <div style={props.style}>
            슬라이더
        </div>
    );
}

export default Slider;

```

터미널 확인  
npm start

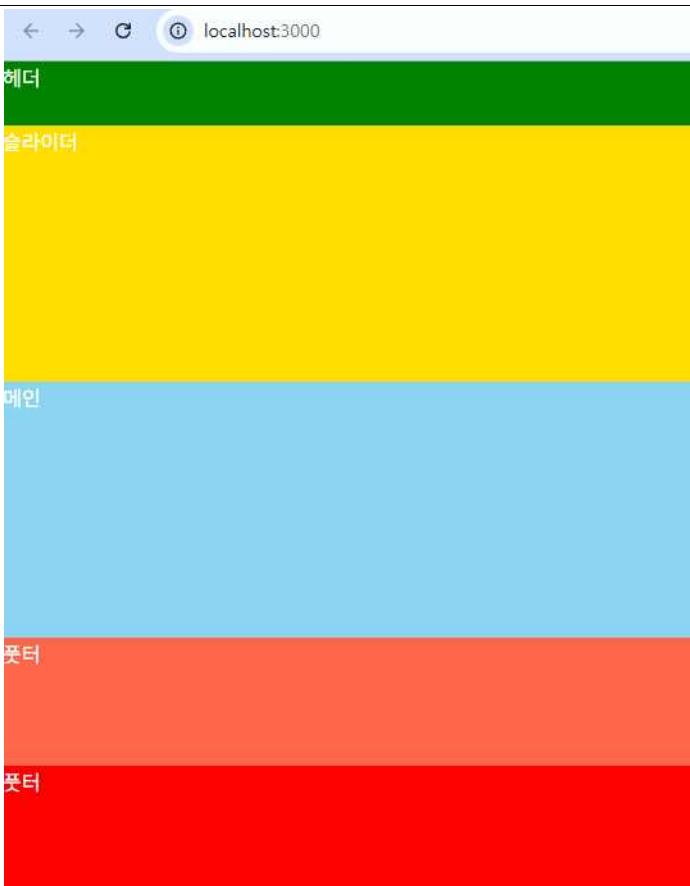
style={{ ... }}	부모 컴포넌트(App)에서 스타일을 자식(Slider)에게 전달함
props	자식 컴포넌트가 전달받은 속성을 담는 객체
rops.style	부모가 보낸 스타일 객체를 꺼내어 <div>에 적용함
props란	props는 properties(속성)의 줄임말로, 부모 컴포넌트가 자식 컴포넌트에게 값을 전달할 때 사용하는 객체 props는 자식 컴포넌트가 외부 데이터를 전달받는 방법

### Main.js 컴포넌트에 Style 입히기 – 3 스타일 변수

EXPLORER ... BLOG1 > node_modules > public src components Cow.js Footer.js Header.js Main.js Slider.js App.css App.js App.test.js Cat.js Dog.js index.css index.js logo.svg reportWebVitals.js setupTests.js .gitignore	<pre> 1 import React from 'react' 2 3 function Main() { 4   const style2 = { 5     width: '100%', 6     height: '200px', 7     backgroundColor: 'skyblue', 8     color: '#ffff' 9   } 10  return ( 11    &lt;div style={style2}&gt; 12      메인 13    &lt;/div&gt; 14  ) 15 } 16 17 export default Main 18 19 </pre>	코드추가	
components > Main.js	import React from 'react'  function Main() { //라인 스타일을 객체 형태로 정의 const style2 = {   width: '100%',   height: '200px',   backgroundColor: 'skyblue',   color: '#ffff' } return ( <div style={style2}>   메인 </div> ) }  export default Main	터미널 확인 npm start	
style2	스타일 속성을 가진 객체		
style={style2}	style={객체} 구조는 JSX에서 인라인 스타일을 적용하는 기본 문법 (JSX에서 style에 객체를 넣는 방식)		

<p>EXPLORER</p> <ul style="list-style-type: none"> <li>BLOG1           <ul style="list-style-type: none"> <li>node_modules</li> <li>public</li> <li>src               <ul style="list-style-type: none"> <li>components                   <ul style="list-style-type: none"> <li>Cow.js</li> <li>Footer.js</li> <li>Header.js</li> <li>Main.js</li> <li>Slider.js</li> <li>App.css</li> <li>App.js</li> <li>App.test.js</li> <li>Cat.js</li> </ul> </li> </ul> </li> </ul> </li> </ul>	<p>App.css M X</p> <pre> 38 } 39 40 41 .footerStyle { 42   background-color: tomato; 43   color: #ffff; 44   height: 100px; 45 } 46 47 </pre> <p style="color: red; font-size: small;">코드작성</p>	<p>App.css – 맨아래 추가</p> <pre> .footerStyle {   background-color: tomato;   color: #ffff;   height: 100px; } </pre>
<p>EXPLORER</p> <ul style="list-style-type: none"> <li>BLOG1           <ul style="list-style-type: none"> <li>node_modules</li> <li>public</li> <li>src               <ul style="list-style-type: none"> <li>components                   <ul style="list-style-type: none"> <li>Cow.js</li> <li>Footer.js</li> <li>Header.js</li> <li>Main.js</li> <li>Slider.js</li> <li>App.css</li> <li>App.js</li> <li>App.test.js</li> <li>Cat.js</li> <li>Dog.js</li> <li>index.css</li> <li>index.js</li> <li>logo.svg</li> </ul> </li> </ul> </li> </ul> </li></ul>	<p>Footer.js U X</p> <pre> 1 import './App.css'; 2 import React from 'react' 3 4 function Footer() { 5   return ( 6     &lt;div className="footerStyle"&gt; 7       풋터 8     &lt;/div&gt; 9   ) 10 } 11 12 export default Footer 13 </pre> <p style="color: red; font-size: small;">코드 추가</p>	<p>Footer.js</p> <pre> import './App.css'; import React from 'react'  function Footer() {   return (     &lt;div className="footerStyle"&gt;       풋터     &lt;/div&gt;   ) }  export default Footer </pre>
<p>터미널 확인</p> <pre>npm start</pre>		
<p>className 속성</p>	<p>Footer 컴포넌트에서 외부 CSS 파일(App.css)의 클래스를 불러와서, className 속성을 통해 JSX 요소에 스타일을 적용하는 방식</p>	

<p>EXPLORER</p> <ul style="list-style-type: none"> <li>BLOG1           <ul style="list-style-type: none"> <li>node_modules</li> <li>public</li> <li>src               <ul style="list-style-type: none"> <li>components                   <ul style="list-style-type: none"> <li>Cow.js</li> <li>Footer.js</li> <li>Footer2.js</li> <li>Header.js</li> <li>Main.js</li> <li>Slider.js</li> </ul> </li> <li>App.css</li> <li>App.js</li> <li>App.test.js</li> <li>Cat.js</li> <li>Dog.js</li> <li>index.css</li> <li>index.js</li> <li>logo.svg</li> </ul> </li> </ul> </li> </ul>	<p>App.css - 맨아래 추가</p> <pre> 39 40 41 .footerStyle { 42   background-color: tomato; 43   color:#fff; 44   height: 100px; 45 } 46 47 .footerStyle2 { 48   background-color: red; 49   color:#fff; 50   height: 100px; 51 } 52 53 </pre> <p>코드추가</p>
<p>EXPLORER</p> <ul style="list-style-type: none"> <li>BLOG1           <ul style="list-style-type: none"> <li>node_modules</li> <li>public</li> <li>src               <ul style="list-style-type: none"> <li>components                   <ul style="list-style-type: none"> <li>Cow.js</li> <li>Footer.js</li> <li>Footer2.js</li> <li>Header.js</li> <li>Main.js</li> <li>Slider.js</li> </ul> </li> <li>App.css</li> <li>App.js</li> <li>App.test.js</li> <li>Cat.js</li> <li>Dog.js</li> <li>index.css</li> <li>index.js</li> <li>logo.svg</li> </ul> </li> </ul> </li> </ul>	<p>Footer2.js</p> <pre> import React from 'react'  function Footer() {   return (     &lt;div className='footerStyle2'&gt;       풋터     &lt;/div&gt;   ) }  export default Footer </pre>
<p>EXPLORER</p> <ul style="list-style-type: none"> <li>BLOG1           <ul style="list-style-type: none"> <li>node_modules</li> <li>public</li> <li>src               <ul style="list-style-type: none"> <li>components                   <ul style="list-style-type: none"> <li>Cow.js</li> <li>Footer.js</li> <li>Footer2.js</li> <li>Header.js</li> <li>Main.js</li> <li>Slider.js</li> </ul> </li> <li>App.css</li> <li>App.js</li> <li>App.test.js</li> <li>Cat.js</li> <li>Dog.js</li> <li>index.css</li> <li>index.js</li> <li>logo.svg</li> <li>reportWebVitals.js</li> <li>setupTests.js</li> </ul> </li> </ul> </li> </ul>	<pre> 1 import './App.css'; 2 import Header from './components/Header'; 3 import Slider from './components/Slider'; 4 import Main from './components/Main'; 5 import Footer from './components/Footer'; 6 import Footer2 from './components/Footer2'; 7 8 function App() { 9   return ( 10     &lt;&gt; 11       &lt;Header/&gt; 12       &lt;Slider style={{width:"100%", color:"white", height:"200px", backgroundColor:"gold"}}&gt;&lt;/Slider&gt; 13       &lt;Main &gt; &lt;/Main&gt; 14       &lt;Footer&gt;&lt;/Footer&gt; 15       &lt;Footer2/&gt; 16     &lt;/&gt; 17   ); 18 } 19 20 export default App; 21 </pre> <p>코드추가</p>
<p>App.js</p> <p>코드추가</p> <p>전역 범위(global)</p>	<pre> import './App.css'; import Header from './components/Header'; import Slider from './components/Slider'; import Main from './components/Main'; import Footer from './components/Footer'; import Footer2 from './components/Footer2'; function App() {   return (     &lt;&gt;       &lt;Header/&gt;       &lt;Slider style={{width:"100%", color:"white", height:"200px", backgroundColor:"gold"}}&gt;&lt;/Slider&gt;       &lt;Main &gt; &lt;/Main&gt;       &lt;Footer&gt;&lt;/Footer&gt;       &lt;Footer2/&gt;     &lt;/&gt;   ); } </pre>

	<pre>         }         export default App;     </pre>
	
터미널 확인 npm start	Footer 컴포넌트에서 외부 CSS 파일(App.css)의 클래스를 불러와서, className 속성을 통해 JSX 요소에 스타일을 적용하는 방식

## Main 컴포넌트에 자식 컴포넌트(Dog, Cow) 포함하기

EXPLORER

- > BLOG1
- > node\_modules
- > public
- > src
  - > components
    - Cow.js
    - Footer.js
    - Footer2.js
    - Header.js
    - Main.js
    - Slider.js
  - App.css
  - App.js
  - App.test.js
  - Cat.js
  - Dog.js
  - index.css
  - index.js
  - logo.svg
  - reportWebVitals.js
  - setupTests.js
- .gitignore
- package-lock.json
- package.json
- README.md
- 컴포넌트.dno

... Main.js

```

src > components > Main.js > ...
  1 import React from 'react'
  2 import Dog from './Dog';
  3 import Cow from './Cow';
  4
  5 function Main() {
  6   const style2 = {
  7     width: '100%',
  8     height: '200px',
  9     backgroundColor: 'skyblue',
 10     color: '#fff'
 11   }
 12   return (
 13     <div style={style2}>
 14       메인
 15       <Dog></Dog>
 16       <Cow></Cow>
 17     </div>
 18   )
 19 }
 20
 21 export default Main
 22
 23
  
```

코드추가

Main.js

```

import React from 'react'
import Dog from './Dog';
import Cow from './Cow';

function Main() {
  const style2 = {
    width: '100%',
    height: '200px',
    backgroundColor: 'skyblue',
    color: '#fff'
  }
  return (
    <div style={style2}>
      메인
      <Dog></Dog>
      <Cow></Cow>
    </div>
  )
}

export default Main
  
```

터미널 확인

```
npm start
```