홍익대학교 멋쟁이사자처럼 13기
프론트엔드 세션 2

React.js 기초

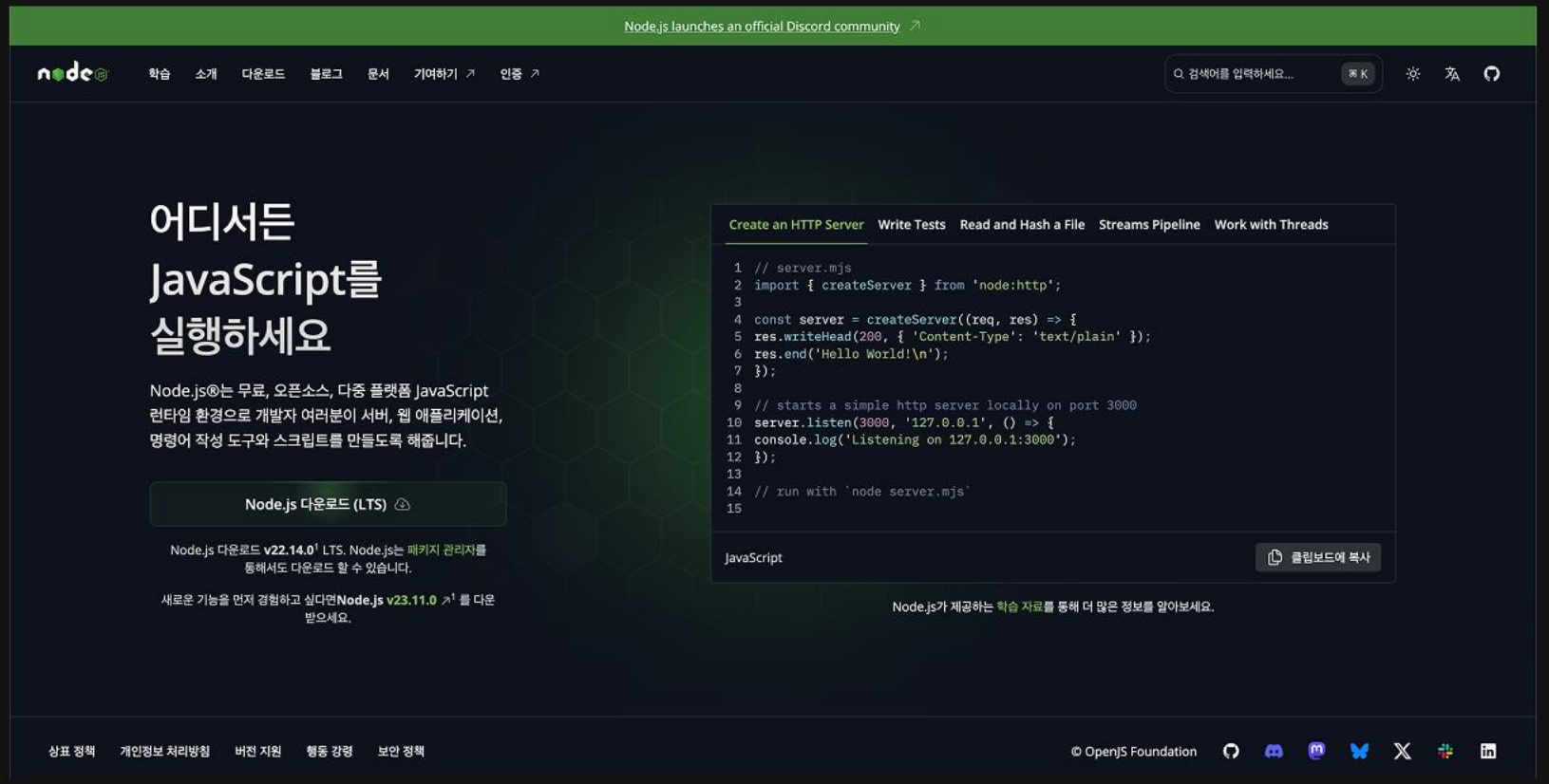
React / TailwindCSS / React-Router

심수연

홍익대학교 멋쟁이사자처럼 13기
프론트엔드 세션 2

React

Node.js



웹 브라우저가 아닌 환경에서도 JS 코드를 실행시켜주는 JavaScript 실행 환경(Run Time) = 구동기

React 특징

SPA(Single Page Application)를 구현하는 데 가장 많이 사용되는 JS 라이브러리

컴포넌트 사용 - 재사용에 최적화

Virtual DOM 사용 - 빠른 업데이트와 렌더링 속도 향상

JSX 사용 - 자바스크립트에 XML을 추가해 확장한 문법

- JSX는 브라우저에서 실행되기 전에 babel 이라는 JS 컴파일러를 통해 자바스크립트로 변환됨

```
// 실제 작성한 JSX 예시
function App() {
  return <h1>Hello</h1>
}
```

```
// 왼쪽처럼 작성하면, babel이 다음과 같이
// 자바스크립트로 해석

function App() {
  return React.createElement("h1", null,
"Hello");
}
```

JSX 장점

보기 쉽고 익숙함	JS에서 HTML과 비슷하게 코드를 작성할 수 있는 것이 JSX의 가장 큰 장점이자 이를 사용하는 이유
오류 검사에 용이	JSX 코드에 오류가 있을 때, 바벨이 코드를 변환하는 과정에서 이를 감지 (ex | 태그를 닫지 않은 경우)
높은 활용도	JSX에서는 우리가 기존에 알고 있는 div나 span 같은 HTML 태그를 사용할 수 있을 뿐만 아니라, 앞으로 만들 컴포넌트도 JSX 안에서 작성할 수 있음

JSX 특징

반드시 하나의 부모 요소가 나머지 요소를 감싸는 형태

```
function App() {
  return (
    <h1>테스트</h1>
    <h1>테스트</h1>
  )
}
```

에러 발생

```
function App() {
  return (
    <div>
        <h1>테스트</h1>
        <h1>테스트</h1>
    </div>
  )
}
```

정상 작동 코드 (div로 감싸줌)

```
import React from 'react';

function App() {
  return (
        <React.Fragment>
            <h1>멋쟁이사자처럼</h1>
            <h1>홍익대학교</h1>
        </React.Fragment>
  );
}
```

fragment 사용

→ 같은 별도의 노드를 추가 X 여러 개의 자식 감쌀 수 O
→ 불필요한 div를 렌더링하는 것 생략 가능 !

- fragment는 **<> </>** 이렇게 나타낼 수 있음

JSX 특징

JSX 내부에서 자바스크립트 표현식을 사용할 수 있음
⇒ 자바스크립트 값을 JSX 안에서 렌더링 가능

```jsx
import React from 'react';

function App() {
        const name = "리액트";
        return (
        <>
            <h1>{name}</h1>
            <h2>이름</h2>
        </>
  );
}
```

← 자바스크립트 표현식 작성하려면
JSX 내부에서 코드를 { } 로 감싸면 됨 !

JSX 특징

JSX 내부의 자바스크립트 표현식에서는 if문을 사용할 수 없음
⇒ { } 내부에 삼항 연산자 사용

{A ? B : C}
: A가 맞으면 B, A가 아니면 C를 보여줌

```jsx
import React from 'react'

function App() {
  const name = '리액트'
  return (
    <div>{name === '리액트' ? <h1>리액트</h1>
  : <h2>리액트가 아님</h2>}</div>
  )
}

```

- **name이 '리액트'라면 <h1>리액트</h1>이 출력**
- **name이 리액트가 아니라면 <h2>리액트가 아님</h2>이 출력**

JSX 특징

&&를 사용한 조건부 렌더링

```
1 { condition ? '보여주기' : null }
2  //=> JSX에서는 null 값 외에도 false 값을 렌더
  링하면 아무것도 나타나지 않음. 따라서 이때는 삼항연
  산자 대신 다음과 같이 작성
3
4 { condition && '보여주기' }
```

- 특정 조건을 만족할 때와 만족하지 않을 때에 따라 다른 UI를 보여줘야 할 때
  ⇒ 삼항 연산자 사용

- 단순히 특정 조건을 만족할 때는 화면에 보여주고, 만족하지 않을 때는 보여주지 않는 경우
  ⇒ && 연산자를 이용한 조건부 렌더링

JSX 특징

class 대신 className 사용

- 리액트에서 class를 설정할 때는 class 키워드 대신 className으로 설정해야 함

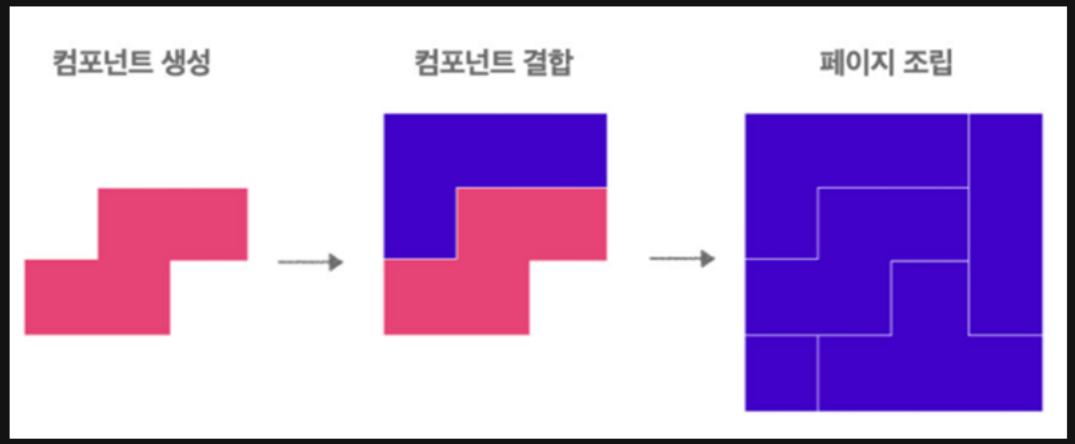
반드시 닫아야 하는 태그

- HTML 코드를 작성할 때는 가끔 태그를 닫지 않아도 괜찮은 경우가 있었음
  (ex | <input>)

- JSX에서는 반드시 태그를 닫아주어야 함 (ex | <input />)
  ⇒ 태그를 닫지 않으면 virtual DOM에서 트리 형태의 구조를 만들지 못하기 때문에 오류가 생김

Component

리액트로 작성된 화면은 컴포넌트로 구성되어 있음



- 리액트로 만들어진 앱을 이루는 최소한의 단위
- 기존 웹 프레임워크가 가지고 있던 재사용성이 어렵다는 단점을 보완하기 위해 사용
- 부모 → 자식 컴포넌트로 데이터 전달 시 props를 사용
- 컴포넌트 이름은 대문자로 작성해야 함

React 폴더구조

src/	
── **assets/**	이미지, 폰트, svg 등 정적 리소스
── **components/**	재사용 가능한 컴포넌트 (Button, Card 등)
── **pages/**	페이지 단위 컴포넌트 (route 연결되는 화면)
── **layouts/**	Layout 컴포넌트 (공통 구조 - Header, Footer 등)
── **routes/**	라우팅 설정
── **hooks/**	커스텀 훅
── **utils/**	유틸 함수들 (formatDate, calc 등)
── **data/**	더미 데이터, json 파일
── **constants/**	고정값 (색상, 텍스트, 키값 등)
── **services/**	API 호출 함수 모음 (axios 등)
── **stores/**	상태관리 (예: recoil, zustand, context)
── **App.jsx**	앱 루트 컴포넌트
── **main.jsx**	엔트리 파일

홍익대학교 멋쟁이사자처럼 13기
프론트엔드 세션 2

TailwindCSS

TailwindCSS란?

유틸리티-퍼스트 CSS 프레임워크

CSS 클래스를 직접 작성하지 않고,
HTML에 미리 정의된 클래스들을 조합해서 디자인하는 방식

장점

생산성 향상
→ CSS 파일 안 봐도 클래스만으로 디자인 가능

반응형 디자인 용이
→ sm:, md:, lg: 같은 접두사만 붙이면 됨

클래스 이름 고민할 필요 없음

디자인 일관성 유지에 용이

홍익대학교 멋쟁이사자처럼 13기

TailwindCSS란?

기존 방식
.card { padding: 16px; background: white; }

→

TailwindCSS 방식
className="p-4 bg-white"

TailwindCSS 사용 예시 ⬇️

```html
<div class="flex flex-col items-center gap-6 p-7 md:flex-row md:gap-8 rounded-2xl">
  <div>
    <img class="size-48 shadow-xl rounded-md" alt="" src="/img/cover.png" />
  </div>
  <div class="flex items-center md:items-start">
    <span class="text-2xl font-medium">Class Warfare</span>
    <span class="font-medium text-sky-500">The Anti-Patterns</span>
    <span class="flex gap-2 font-medium text-gray-600 dark:text-gray-400">
      <span>No. 4</span>
      <span>·</span>
      <span>2025</span>
    </span>
  </div>
</div>
```

```jsx
<div className="pt-20 bg-black text-white min-h-screen">
  <Outlet />
</div>
```

Styled-Components는?

CSS-in-JS 라이브러리

자바스크립트 파일 안에서 CSS를 작성할 수 있도록 도와줌
React에서 컴포넌트 단위로 스타일을 분리하고 재사용 가능하게 함

```
import styled from "styled-components";
// 컴포넌트 선언하고, styled.태그종류를 할당하고
const BlueButton = styled.button`
//스타일 속성 작성하고
  background-color: blue;
    color: white;
`;
export default function App() {
  //리턴문 안에 스타일이 적용된 컴포넌트 사용 -> 렌더
    return <Bluebutton>Blue Button</Bluebutton>;
}
```

컴포넌트 기반 스타일링 → 깔끔

props로 스타일 변경 가능

조건부 스타일링 용이

---

but, 성능 이슈(실행 시점에 스타일 처리)나 유틸리티 클래스 기반 CSS의 대세로 인해 더 이상 업데이트 안 함

신촌톤 때 사용할지도..?

홍익대학교 멋쟁이사자처럼 13기
프론트엔드 세션 2

React-Router

React-Router란?

React Router는 **SPA** 구조에서 페이지 전환을 자연스럽게 처리할 수 있도록 도와주는 핵심 도구

- React 전용 라우팅 라이브러리
- SPA 환경에서 페이지 전환을 가능하게 해줌
- URL에 따라 다른 컴포넌트를 렌더링함

홍익대학교 멋쟁이사자처럼 13기
프론트엔드 세션 2

실습

📎 Playlist 사이트 만들기

함께 **Node 설치**부터 **React 프로젝트를 세팅**하고 **Playlist 사이트**를 만들어 볼까요?!

제출물은 다음과 같습니다. ⬇️

Node.js 설치부터 진행했던 실습 내용을 복습하며 노션에 정리 &
같이 만든 Playlist 사이트를 깃허브에 올리고 캡쳐해 노션에 업로드