실증 아카데미 리액트 특강

React.js란?

일시 2021년 01월 13일

주최 동아대학교

CONTENTS

01 **Front-End 프레임워크**
웹 프로그래밍의 분야
프레임워크란?
Angular.js, Vue.js, React.js

02 **React.js란?**
React.js란?
개발 환경 구축

03 **React.js의 특징**
Virtual DOM
컴포넌트와 단방향 데이터 전달
state와 lifecycle

04 **React.js의 장단점**

05 **다른 프레임워크와의 차이**
Angular.js
Vue.js

06 **JSX 문법**
JavaScript eXtension
JSX 사용하기
조건부 렌더링

01

Front-End 프레임워크

웹 프로그래밍의 분야

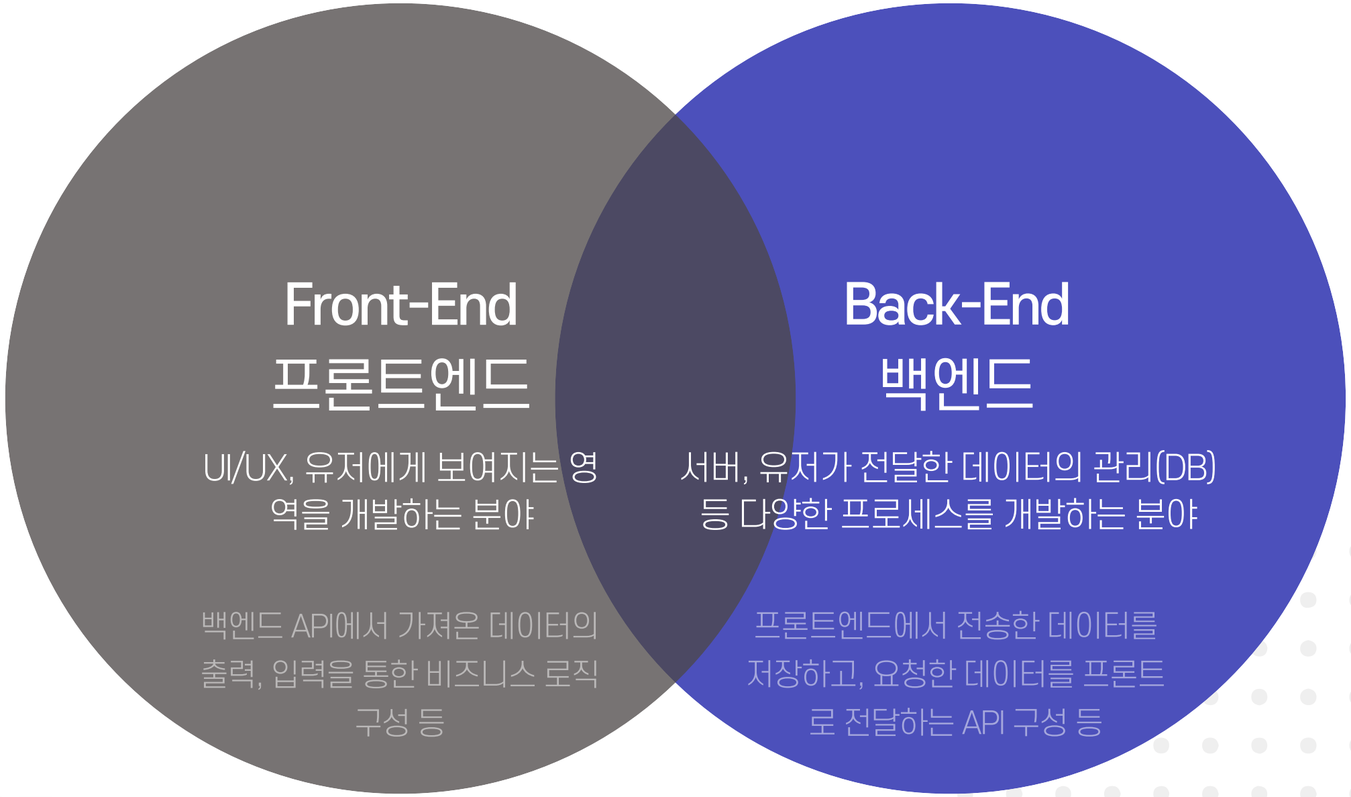
프레임워크란?

Angular.js, Vue.js, React.js

웹 프로그래밍의 분야

프론트엔드와 백엔드

- 웹 프로그래밍은 크게 두 분야로 나뉨
- 두 분야를 모두 다룰 수 있는 개발자를 풀스택 개발자라고 칭함
- 응용 프로그래밍도 크게 다르지 않음
- 웹 퍼블리셔(마크업 개발자)라는 직군도 존재함
- 렌딩 페이지보다 웹 서비스에 어울림
- 웹 애플리케이션은 두 분야의 상호작용으로 이루어진 프로그램



Front-End 개발자

FE 개발자의 소양

- **웹 퍼블리싱(UI, 마크업)**

  UI/UX를 담당하기 때문에 기본적으로 유저에게 알맞는 화면을 렌더링할 수 있는 능력이 필요
  더 나아가 적절한 애니메이션 소양이 있으면 좋음

- **JavaScript, Webpack**

  웹 서비스는 유저와의 상호작용으로 이루어지며, 이는 JavaScript를 통해 구현되어야 함. 또 Webpack이라는 번들링 도구를 사용하는 것이 최근에 각광

- **HTTP 프로토콜, 서버에 대한 이해**

  백엔드에 HTTP로 요청을 하는 것은 물론, 백엔드가 결국 서버를 담당하기 때문에 이해가 필요

프레임워크란?

소프트웨어 개발의 뼈대 역할

- 어떠한 목적을 달성하기 위해 복잡하게 얽혀있는 문제를 해결하기 위한 구조
- 라이브러리의 상위 개념. 라이브러리는 프로그램의 구성요소로, 공통으로 사용될 수 있는 특정한 기능들을 모듈화 한 것을 칭하며, 이들을 묶어 프레임워크를 이룬다고 볼 수 있음
- 웹 프레임워크는 "웹 서버"를 구현하기 위한 목적으로 만들어진 프레임워크
- 프레임워크는 어떠한 목적을 효율적으로 달성하기 위해 좋은 방안이며, 백지상태에서 개발을 하는 난해함을 해결해 주는 가이드의 역할을 수행함
- 웹 프레임워크에서는 FE, BE 각각의 프레임워크가 존재하고, Java의 Spring과 같이 풀스택 프레임워크도 존재함
- 개발을 시작할 때 목적에 따라 적절한 프레임워크를 선택해야 함
- Spring, Django, Ruby on Rails, Laravel, Bootstrap, Express.js, Angular.js, Vue.js, React.js, 안드로이드, Cocoa 등

프론트엔드 프레임워크

Angular.js, Vue.js, React.js

JQuery는 일반적으로 라이브러리라 칭함

- **Angular.js**

  Google에서 운영 중인 프레임워크
  세 프레임워크 중 러닝커브가 가장 가파르며,
  TypeScript가 선택이 아닌 필수라 인기가 다소 낮음

- **Vue.js**

  에반 유에 의해 개발된 프레임워크
  최근 급속도로 확산되고 있으며 유연하고 가벼운 구조
  큰 생태계를 구축해 나가고 있음

- **React.js**

  Meta(Facebook)에서 개발한 라이브러리
  인기와 더불어 npm 패키지 수가 가장 많으며, 이해하기
  쉬운 구조로 생산성이 상당히 높아 가장 활성화

02

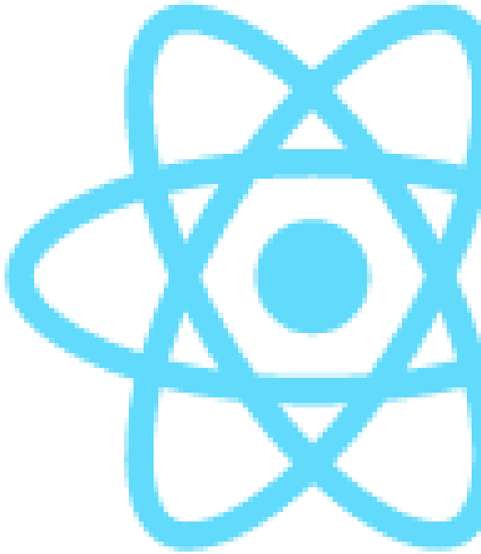
React.js란?

React.js란?
개발 환경 구축

React.js란?

공식 홈페이지: https://ko.reactjs.org/

- 사용자 인터페이스(UI)를 만들기 위한 JavaScript 라이브러리
- 공식 홈페이지에서 라이브러리라고 소개, 다른 프레임워크에 간편하게 붙여서 사용하는 것이 가능
- 일반적인 HTML에 <script> 태그로도 추가할 수 있으나, Node.js와 npm, yarn을 활용한 Create React App(CRA) 환경이 잘 구축되어 있음
- SPA를 전제로 하고 있으며, HTML 요소를 컴포넌트로 나누어서 정리
- React 문법으로 안드로이드, iOS 앱을 개발할 수 있는 React Native라는 프레임워크도 존재
- 굉장히 많은 npm 모듈, 라이브러리 생태계로 생산성을 극대화



개발 환경 구축

Create React App을 구성하는 방법

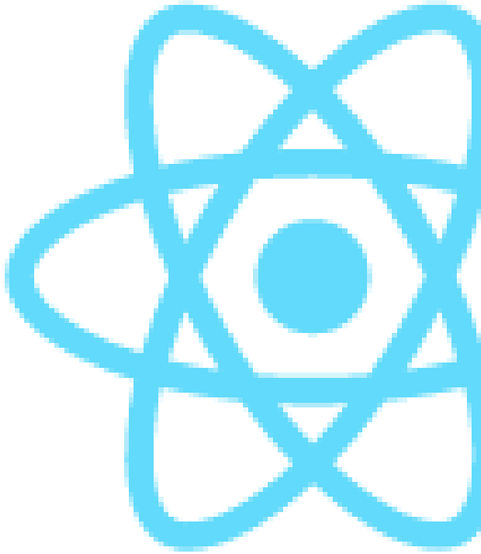
- npm을 설치하면 사용할 수 있는 npm 패키지 실행 도구인 npx를 활용

```
cd Desktop
npx create-react-app my-app
```

- 위 명령을 실행하면 바탕화면에 my-app이라는 폴더가 생성

```
cd my-app
npm start
```

- my-app 폴더로 이동 후 시작 명령어를 실행하면 브라우저가 켜짐과 동시에 CRA 개발 서버 실행



이 외에도 npm init react-app my-app 과 yarn create react-app my-app 이라는 명령어도 동일한 동작

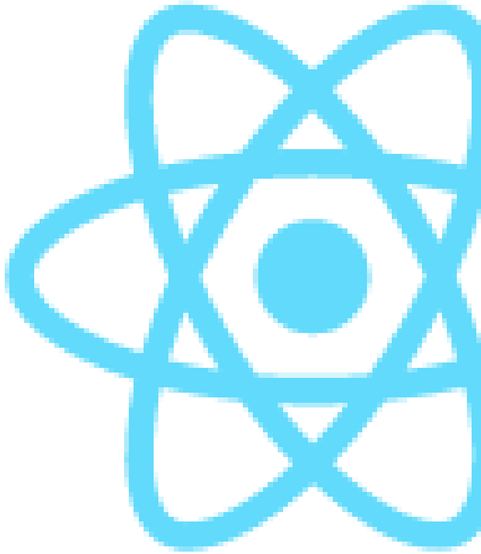
개발 환경

Create React App의 구성

```
my-app
├── README.md
├── node_modules
├── package.json
├── .gitignore
├── public
│   ├── favicon.ico
│   ├── index.html
│   └── manifest.json
└── src
    ├── App.css
    ├── App.js
    ├── App.test.js
    ├── index.css
    ├── index.js
    ├── logo.svg
    └── serviceWorker.js
    └── setupTests.js
```

프로젝트 생성 시의 폴더 구조

- node_modules 폴더: npm 모듈이 저장된 폴더
- package.json: npm 프로젝트 및 의존성 관리 파일
- public 폴더: React 앱의 정적 파일 폴더
- index.html: React 앱이 실행될 HTML 문서
- src 폴더: 프로젝트의 소스 코드가 저장된 폴더
- index.js: 프로젝트의 시작 소스 파일(엔트리 포인트)
- App.js: App 컴포넌트 파일



03

React.js의 특징

Virtual DOM

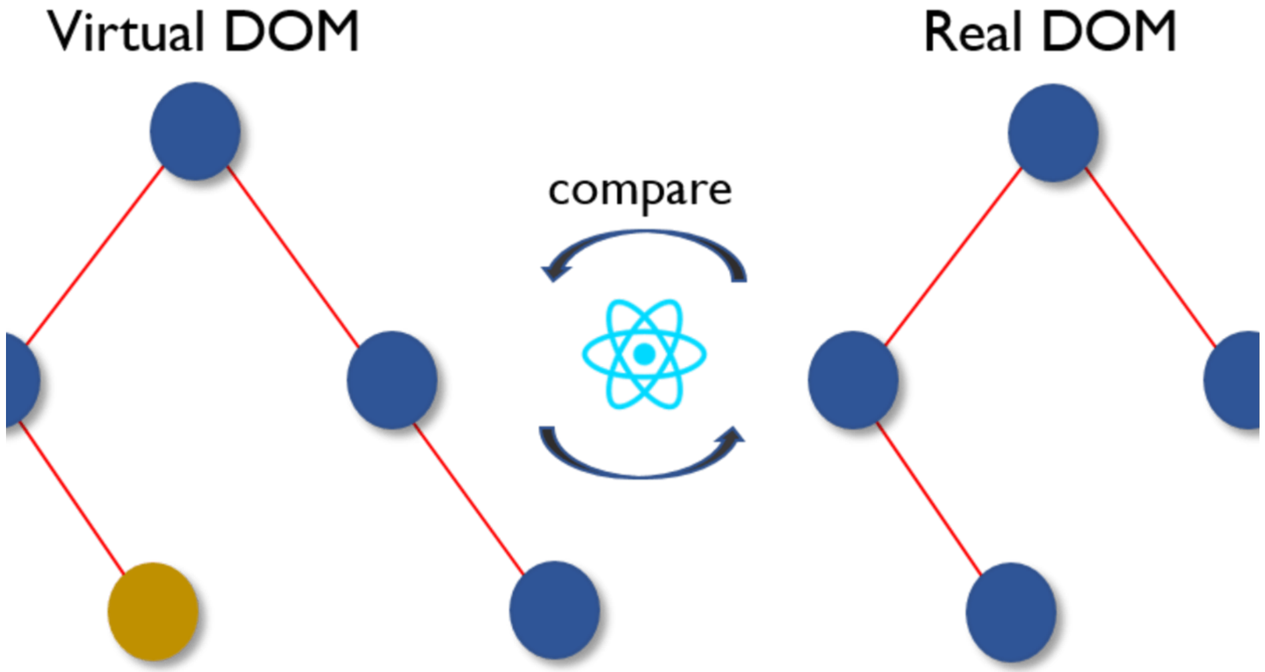
컴포넌트와 단방향 데이터 전달

state와 lifecycle

Virtual DOM

변화를 실제 DOM에 바로 반영하지 않고, 가상으로 표현 성능 향상

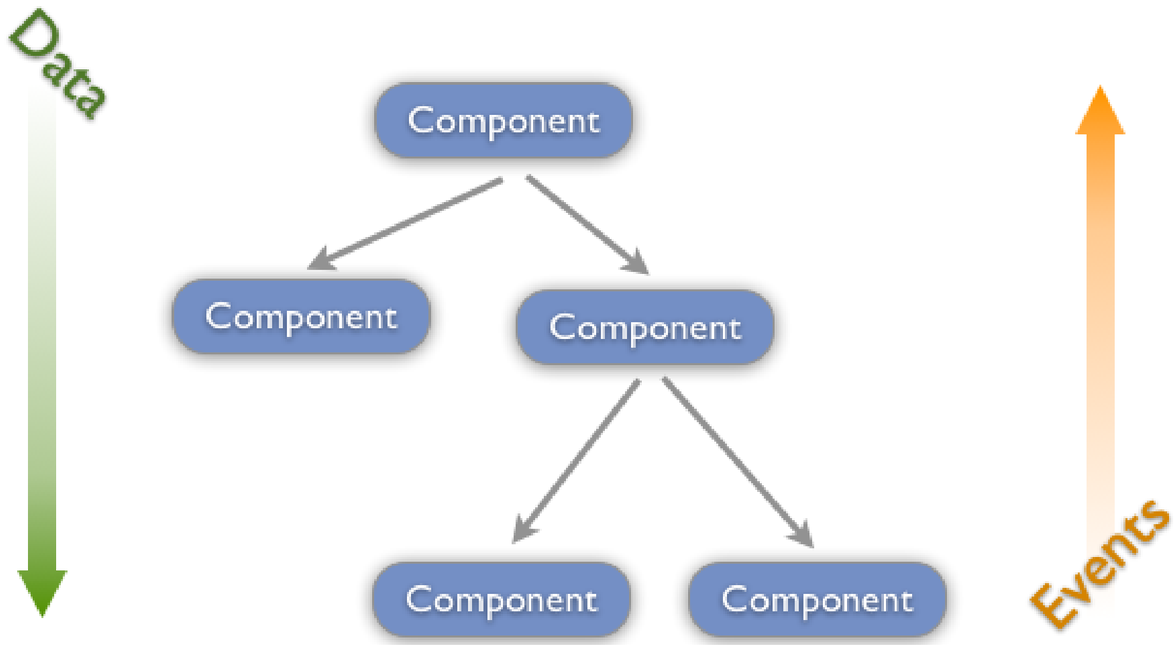
- 브라우저 렌더링은 비용(시간, 부하)이 굉장히 큰 연산이므로, 데이터의 변경이 잦은 웹 서비스에서는 이를 최소화하는 방향으로 개발이 진행되어야 함(최적화)
- 기존 JavaScript는 브라우저에 리렌더링을 요청할 때 정확하게 필요한 부분만 리렌더링하기엔 어려움이 있음
- 이 기술을 지연 평가 혹은 재조정이라고 함
- React.js는 Virtual DOM이라는 가상의 DOM 객체를 만들어 실제 DOM과 비교를 한 후, 차이가 생긴 요소만 실제로 렌더링에 반영함
- 렌더링 연산을 줄임으로써 성능 향상



컴포넌트와 단방향 데이터 전달

React는 컴포넌트를 최소 단위(노드)로 트리 형태를 나타내고 있음

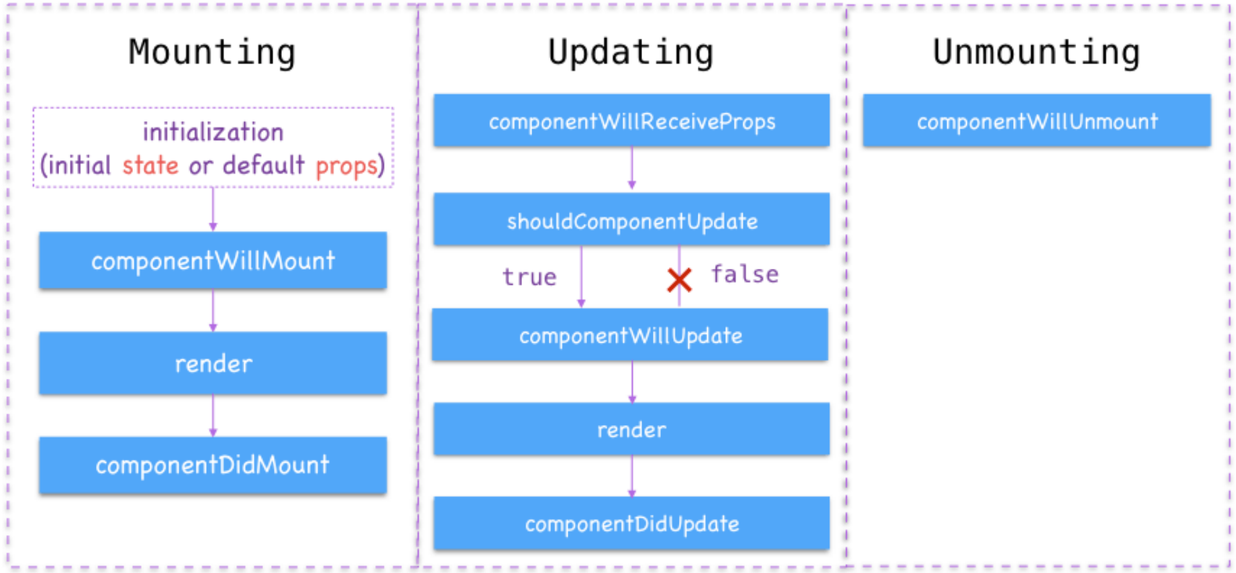
- React는 컴포넌트라는 단위의 집합으로 구성된 트리 구조
- Facebook을 예시로 들면, 상단 헤더 컴포넌트, 헤더 컴포넌트 내의 유저/로그인 영역 컴포넌트, 피드 컴포넌트 등
- 각 컴포넌트는 state를 통해 현재 상태를 나타내며, props를 통해 하위 컴포넌트로 데이터를 전달(단방향)



state와 lifecycle

React의 렌더링 프로세스(알고리즘)

- React의 렌더링 프로세스는 state와 깊게 연관되어 있으며, state가 변경되면 React 컴포넌트 트리에서 리렌더링 여부를 결정함
- 크게 컴포넌트 생성(마운트), 변경(업데이트), 제거(언마운트)의 생명주기(lifecycle)이 있으며, state가 변경됐을 때, 해당 state와 연관된 컴포넌트(props 등)들은 생명 주기 알고리즘을 통해 리렌더링 여부를 판단하고 실제 DOM에 반영함
- 이 생명주기와 리렌더링 여부를 효과적으로 사용해야 효과적인 성능의 웹 애플리케이션 개발 가능



04

React.js의 장단점

React.js의 장점

- **Component로 모든 것을 관리**

  일반적으로 개발을 하게 되면 Controller, directive, template, model, view 처럼 분리를 하게 되는데, React는 Component 단 하나로 관리함

- **배우기가 간단하고, 복잡함이 적음**

  JSX라는 문법을 통해 마치 HTML처럼 직관적으로 개발을 하기 때문에, 처음 배우기 쉽고 복잡하지 않음
  또한 방대한 라이브러리를 지원하고 있음

- **뛰어난 메모리 관리**

  Virtual DOM이나 GC 등으로 메모리 및 성능을 알아서 관리해주기 때문에 최소한의 개발 지식만으로 좋은 효율을 뽑아내기에 용이

- **다른 프레임워크/라이브러리와 혼용**

  다른 프로젝트와 혼용해서 사용할 수 있기 때문에 개발이 완료된 서비스에도 적응이 가능함
  또한 서버 사이드 렌더링과, 클라이언트 사이드 렌더링 모두 지원하여 적용의 범위가 매우 넓음

React.js의 단점

- **보여지는 부분에만 관여**

  데이터 모델링, Routing, Ajax 등등의 기능을 제공하지 않고, View 외 기능들은 직접 구현하거나 외부 라이브러리에 의존해야 하기 때문에 JavaScript 배경 지식이 부족하면 개발에 어려움이 있을 수 있음

- **비용이 많이 발생함**

  실제로 3대 프레임워크, 그리고 JQuery와 비교하였을 때 React.js가 가장 로딩 시간이 길며, 메모리 관리와 별개로 빌드 파일의 용량이 매우 무거움

- **웹의 핵심 가치를 부정함**

  SPA 자체의 문제이며, 웹은 스트리밍(실시간 다운로드)가 근본 가치인데, SPA는 모든 JavaScript를 다운 받은 후에 다시 리소스를 다운 받지 않아 로딩이 길어짐

- **소규모 프로젝트에 적용이 어려움**

  SPA와 같은 서비스를 개발하기 위한 라이브러리이기 때문에, 간단한 서비스에는 오히려 연산이 많아져 성능의 이점이 발휘되지 않을 수 있음

05

다른 프레임워크와의 차이

Angular.js

Vue.js

다른 프레임워크와의 차이

Angular.js

- React는 JSX 기반으로 TypeScript는 옵션이나 Angular는 TypeScript만을 사용해야함(러닝커브)
- React에서는 Virtual DOM을 사용하지만, Angular는 실제 DOM에서 변경 감지를 통해 렌더 업데이트
- React는 단방향 데이터 전달 방식이고, Angular는 양방향 데이터 전달 방식을 채택
- 애플리케이션 사이즈와 성능, 모바일 이식성은 Angular가 뛰어난 편

Vue.js

- React는 JSX 기반인데 반해, Vue는 HTML 기반 템플릿 구문으로 가시성이 좋음
- Vue는 React보다도 진입 장벽이 낮다고 평가되고 있음
- React보다 컴포넌트 재사용 측면에서 비효율적이라고 평가됨
- React에 비해 변화가 느리며, TypeScript 결합이 조금 부족함

06

JSX 문법

JavaScript eXtension
JSX 사용하기
조건부 렌더링

JavaScript eXtension

JavaScript를 확장한 문법으로, 문자열과 HTML의 결합

- React의 Element (컴포넌트의 렌더링 요소)를 생성
- JavaScript에서 HTML을 직관적으로 다룰 수 있는 표현식
- React는 JSX 사용이 필수는 아니지만, 대부분 적용하고 있으며, 시각적으로 매우 도움이 됨
- 동적으로 HTML 요소를 표현하기 때문에 변수를 이용한 연산과, 함수 호출 결과 등으로 상황에 맞게 유연한 렌더링을 할 수 있음
- 컴포넌트는 JSX 문법을 통해 Element의 집합을 반환
- JSX는 후에 JavaScript 코드로 컴파일됨

```jsx
import React from 'react';
import logo from './logo.svg';
import './App.css';

function App() {
  return (
    <div className="App">
      <header className="App-header">
        <img src={logo} className="App-logo" alt="logo" />
        <p>
          Edit <code>src/App.js</code> and save to reload.
        </p>
        <a
          className="App-link"
          href="https://reactjs.org"
          target="_blank"
          rel="noopener noreferrer"
        >
          Learn React
        </a>
      </header>
    </div>
  );
}

export default App;

```

JSX 사용하기

- 한 줄 이상의 JSX 코드는 대괄호('(', ')')로 감싸주어야 함 또한, JSX 코드 내에 여러 Element를 추가하려면 최상위 요소가 필요

```
const element = <h1>Hello, {name}</h1>; // 중괄호 내에 변수 사용 가능
const multiLine = (
    <div>
        <h1>Hello, {name}</h1>
    </div>
    <input /> // 다음과 같이 한 JSX에는 두 개의 최상위 Element 사용 불가.
    // 무조건 하나의 요소로 감싸줘야 함.
);
```

- HTML 속성 정의 가능

```
const element = <div tabIdnex="0"></div>;
// tab-index 요소와 같이 '-'이 있는 속성은 뒤 문자를 대문자로 연결
const elementImage = <img src={user.avatarUrl} />;
```

- 컴포넌트 표현하기

```
const InputComponent = () => {
    const name = "InputText";
    const onChange = (e) => {
        console.log(e.target.value);
    }

    return (
        <div className="component" style={{ width: '100px' }}>
            <input type="text" placeholder={name} onChange={onChange} />
        </div>
    );
}
```

조건부 렌더링

- JSX 내에는 if, switch 문을 사용할 수 없음 때문에 조건식을 사용하여 렌더링해야 함

```
const Condition = () => {
    const isRender = false;

    return (
        <div>
            {isRender ? <h1>렌더 가능</h1> : <p>렌더 불가</p>
        </div>
    );
}
```

- &&와 || 연산자를 통해 필요한 렌더링 하기

```
const Condition = () => {
    const isRender = false;

    return (
        <div>
            {isRender && <h1>true면 렌더</h1>}
            {isRender || <h1>false면 렌더</h1>}
        </div>
    );
}
```

실증 아카데미 리액트 특강

THANK YOU

(주) 인바이즈 박철현
(주) 인바이즈 개발팀