iOS개발실무

2주차

담당: 김희숙
(jasmin11@hanmail.net)

리액트(React)

담당: 김희숙

(jasmin11@hanmail.net)

React 특징

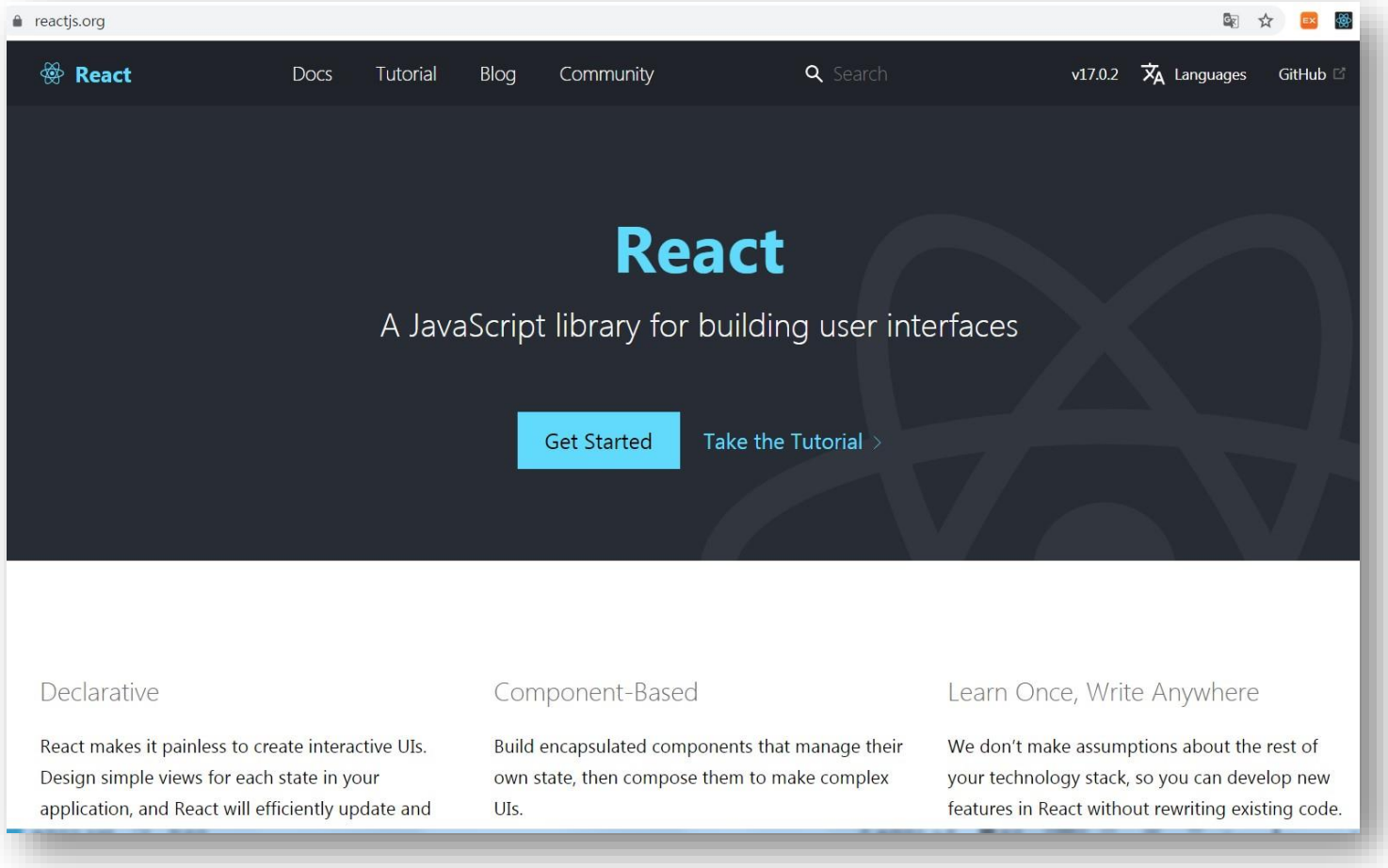
담당: 김희숙
(jasmin11@hanmail.net)

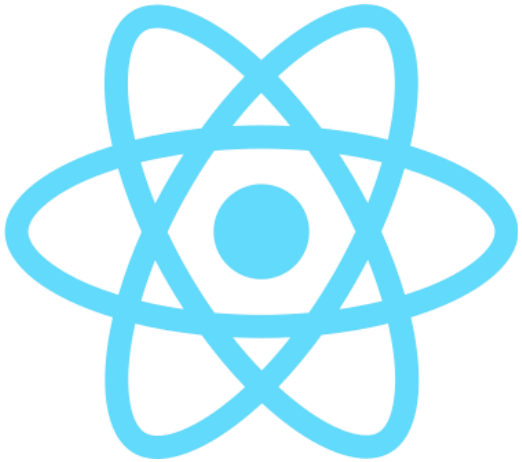
[요약] react.js

https://ko.reactjs.org/
https://reactjs.org/

- react.js
  - 자바스크립트 라이브러리 (프레임워크가 아니다)
  - 프론트엔드 라이브러리(Web App)

- 자바스크립트 라이브러리
  - jQuery, Underscore.js
- 자바스크립트 프레임워크
  - Angular: 2010년
  - React.js: 2013년
  - Vue.js: 2014년





[요약] react.js

https://ko.reactjs.org/

❖ react.js
- ❖ 자바스크립트 라이브러리 (프레임워크가 아니다)
- ❖ 프론트엔드 라이브러리
- ❖ 페이스북(Facebook) 개발
- ❖ DOM 관리와 상태 변화 관리를 최소화하고,
  개발자는 오직 기능 개발, 사용자 인터페이스에 보다 더 집중

❖ 프론트엔드 라이브러리(Frontend Library)
- ❖ 동적인 웹 페이지를 보다 효율적으로 유지 보수하고 관리

❖ 정적 페이지
- ▪ 웹 서버에 이미 저장되어 있는 HTML 문서를 클라이언트에게 전달하여 받은 페이지

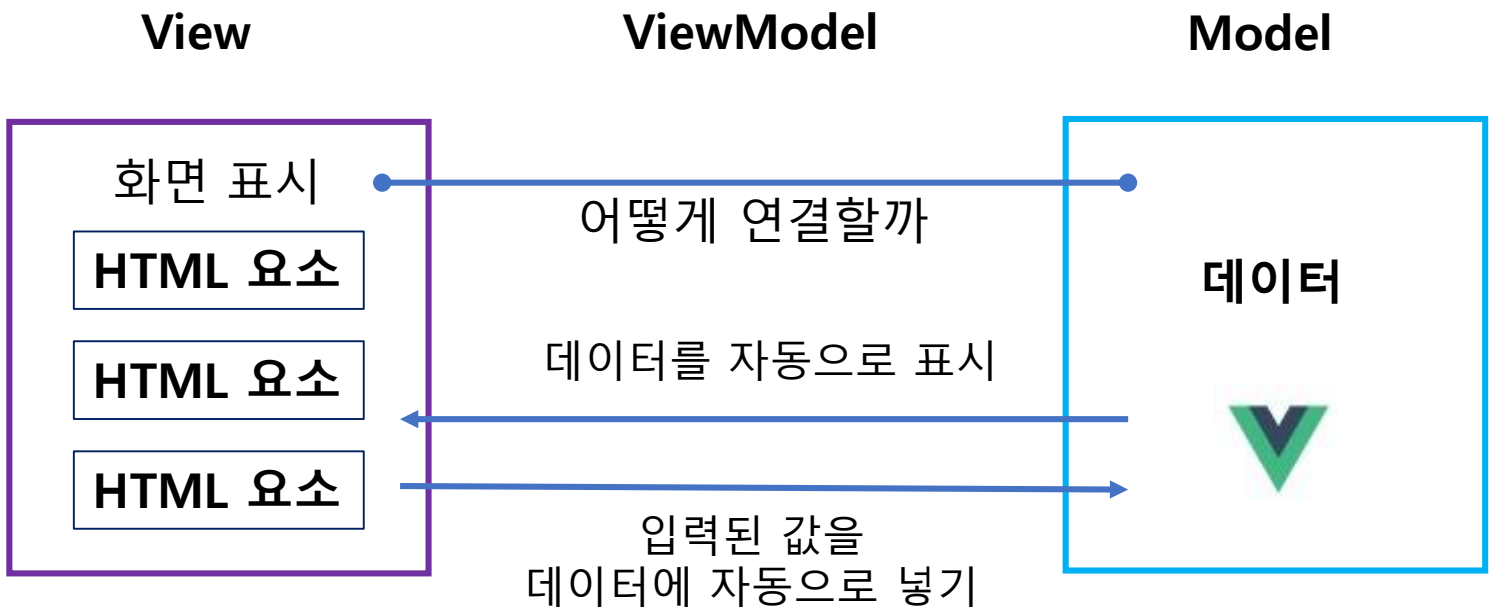
❖ 동적 페이지
- ▪ 사용자 행동 흐름에 따라 웹페이지의 구성을 다르게 하는 페이지
- ▪ 웹 애플리케이션: 사용자 인터랙션을 처리하기 위한 상태 변화가 많아짐

❖ 리액트(React)
- 2011년 facebook 개발
  뉴스피드에서 사용
- 2012년
  인스타그램에서 사용
- 2013년 오픈소스
- 2014년 확장
- 2015년
  React Native 출시
- 2016년 React 15
- React 16.8
  Hook
  2021년 3월
  안정화버전: 17.0.2

[특징] React vs. Vue.js

MVVM 패턴

❖ Vue.js: 데이터와 뷰를 연결해 주는 역할
  ❖ 뷰 모델(ViewModel)



❖ **MVVM**

1. **View**: 표시되는 요소
   데이터가 HTML 어느 부분에서
   어떤 형식으로 표시되는가
2. **ViewModel**: 어떤 방식으로 연결
   HTML 조작될 때 데이터가
   어떤 형식으로 변화되는가
3. **Model**: 데이터
   웹 페이지에서 바뀌는 부분은
   어디인가

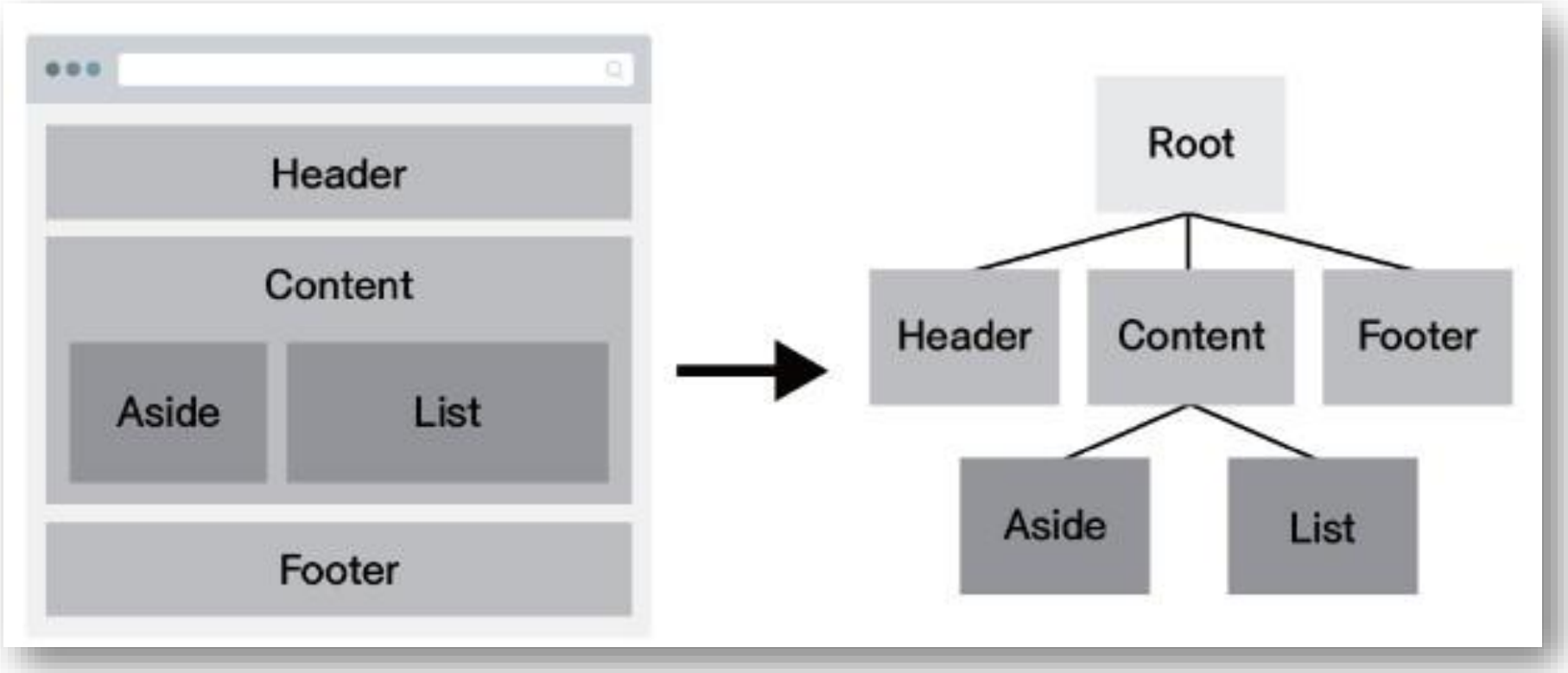
❖ **MVC**
   Model
   View
   Controller

[특징] 컴포넌트

컴포넌트: 화면의 한 영역

❖ 컴포넌트

- 레고 블록
- 컴포넌트를 조합하여 화면 구성
- 장점) 코드 재사용, 직관적인 코드 구조

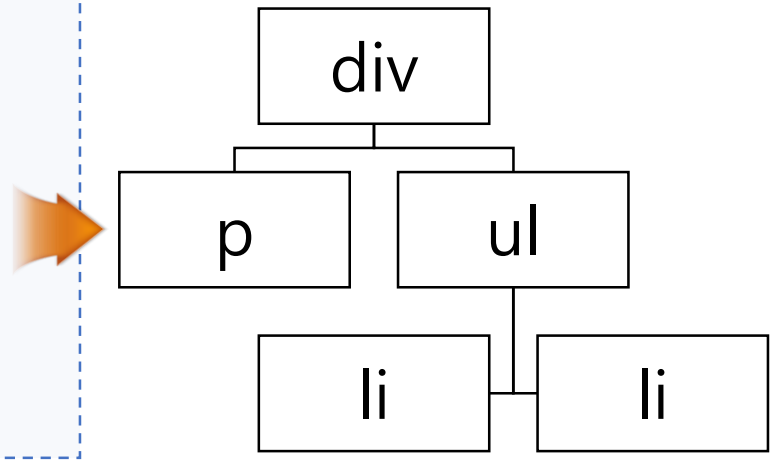


[특징] 문서 객체 모델(DOM)

가상 DOM

❖ DOM(**Document Object Model**): DOM 을 추상화한 트리 구조
- ❖ 트리 형태로 구조화된 텍스트 개념
- ❖ DOM을 표현하기 위한 언어로 HTML 사용
- ❖ DOM 기능 사용하면 화면의 내용을 생성, 수정, 삭제 가능

```
// HTML로 작성된 DOM 예시
<div>
    <p></p>
        <ul>
            <li></li>
            <li></li>
        </ul>
</div>
```



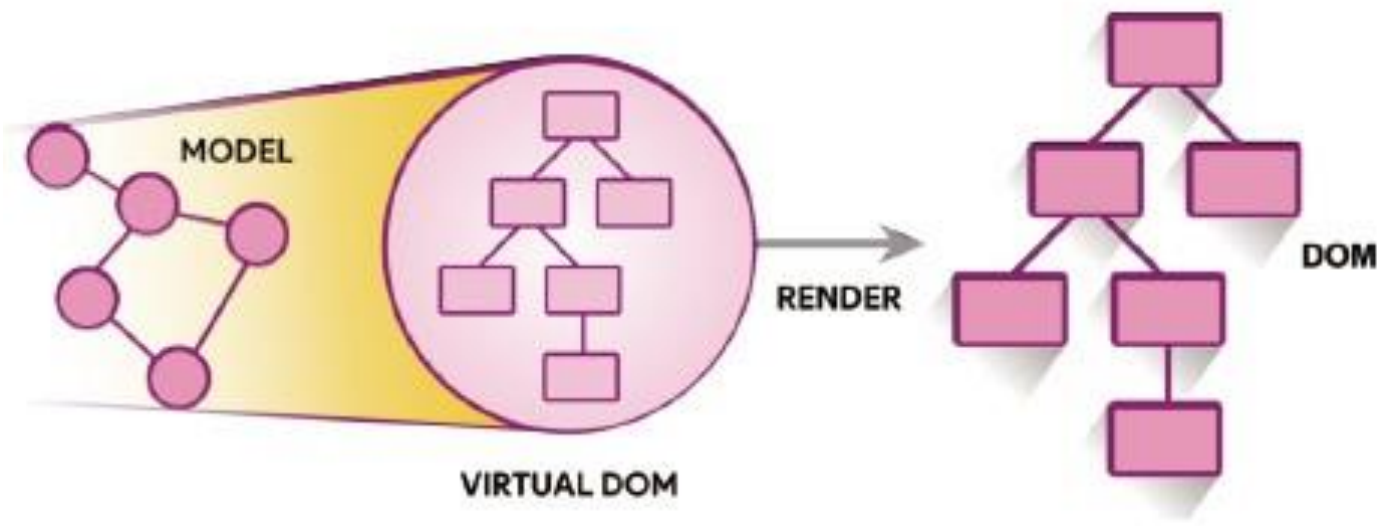
-SPA(Single Page Application):
DOM 동적인 수정을 전제로 하는 방법
애플리케이션이 구동되는 동안
DOM 변경이 굉장히 빈번하다

브라우저는 DOM 변경 요청을 받는대로
화면을 다시 렌더링한다

[특징] 가상 돔(Virtual DOM)

가상 DOM

❖ 가상 DOM(Virtual DOM): DOM 을 추상화한 트리 구조
  ❖ DOM을 렌더링하는 과정에서 성능저하를 해결하기 위해 고안
  ❖ DOM 내용 변경하고 싶을 때, 실제 DOM 이 아니라 가상 DOM을 대신 변경
  ❖ 화면을 조작하기 위한 기술
  ❖ 화면의 **DOM**을 추가하거나 삭제하는 등의 변경이 일어날 때마다 화면을 다시 그리는 것이 아니라
  자바스크립트 객체로 **DOM** 모양을 잡아 놓고
  화면의 렌더링 횟수를 최소화하여 브라우저의 부하를 줄인다



❖ **가상 DOM:**
- **몇 번을 수정하여도 실제 DOM 내용을 변경한 것이 아니므로 렌더링 발생하지 않는다**
- **모든 변경사항을 반영한 가상 DOM을 토대로 브라우저에게 실제 DOM을 렌더링하라는 요청한다**

React 개발 환경

담당: 김희숙
(jasmin11@hanmail.net)

[설치] 웹 브라우저: Chrome

Chrome 웹 스토어

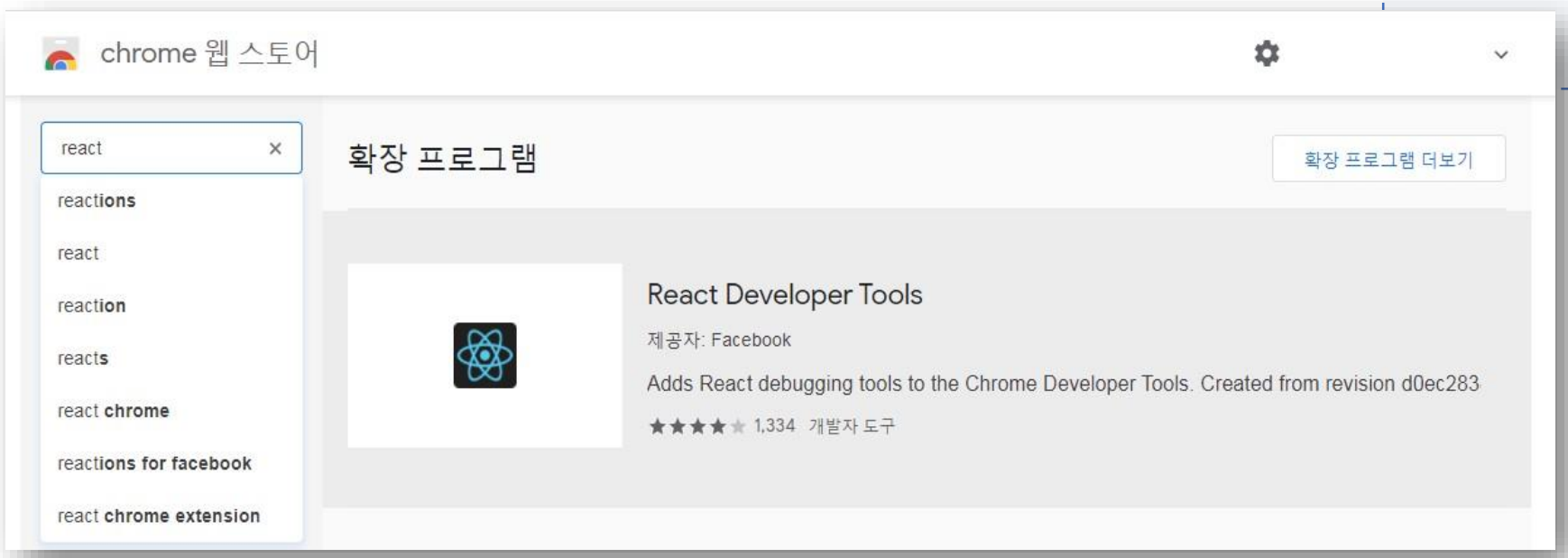
❖ 설치환경:

1. Chrome
2. Node.js
3. Visual Studio Code
4. **React Developer devtools** ⇒ Chrome 확장 프로그램

❖ Chrome 확장 프로그램:

- [도구 더보기]-[확장프로그램]
  - 왼쪽 상단 햄버거 버튼
  - 왼쪽 하단

    [Chrome 웹스토어 열기]
- **react** 검색
  - **react developer tools**
  - **[Chrome 에 추가]**



에어비앤비 www.airbnb.com
npm www.npmjs.com
넷플릭스
https://www.netflix.com/kr/
인스타그램
https://www.instagram.com/

[설치] React Developer tools

Chrome 웹 스토어

❖ React 로 개발된 사이트:

- 에어비앤비
  www.airbnb.com
- npm
  www.npmjs.com
- 넷플릭스
  https://www.netflix.com/kr/
- 인스타그램
  https://www.instagram.com/
- 드롭박스
  https://www.dropbox.com/ko/
- 트위터
  https://twitter.com/?lang=ko
- 우버
  https://www.uber.com/kr/ko/

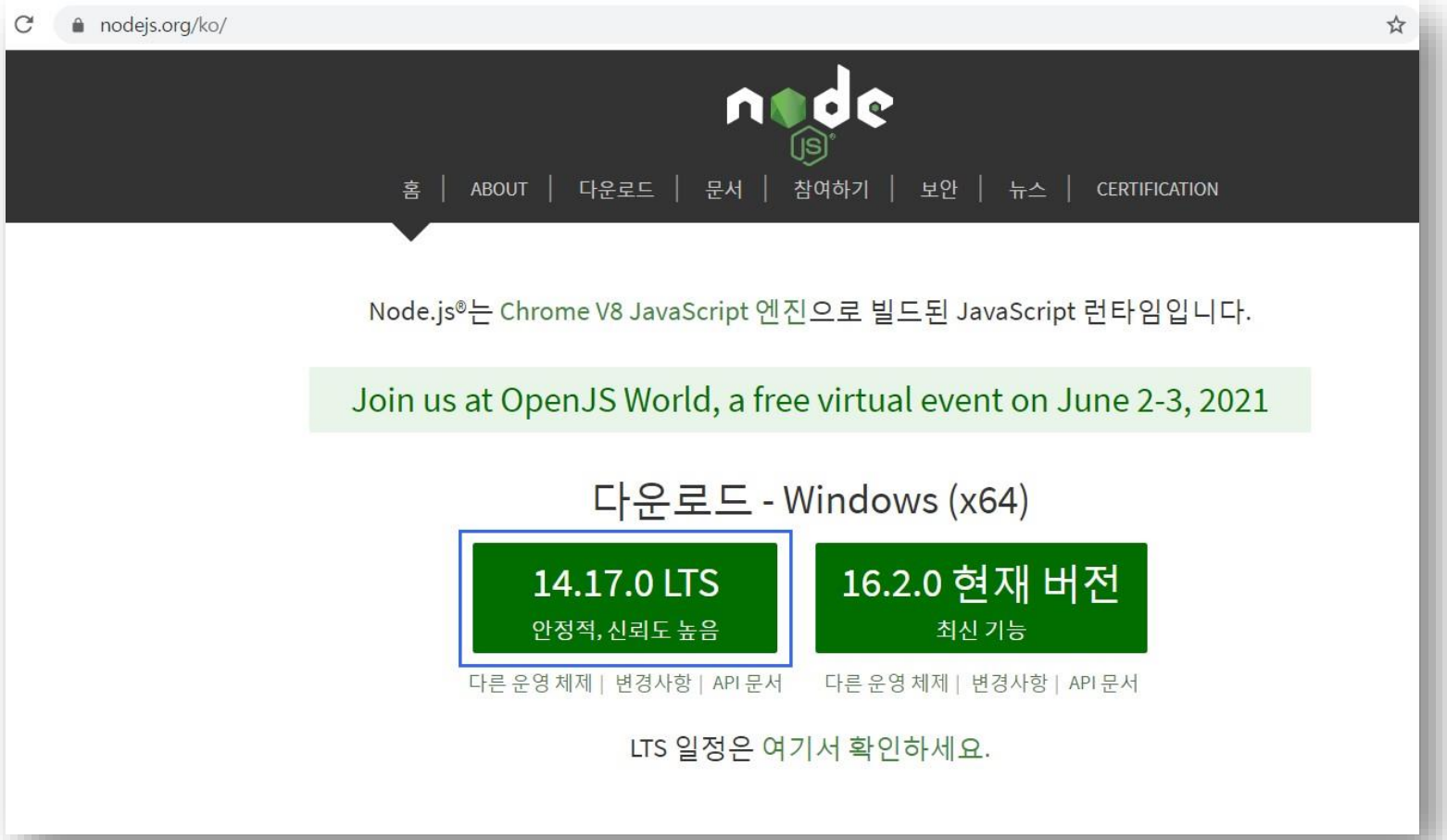
❖ Chrome 확장 프로그램:

- [도구 더보기]-[확장프로그램]
  - 왼쪽 상단 햄버거 버튼
  - 왼쪽 하단
    [Chrome 웹스토어 열기]
- **react** 검색
  - **react developer tools**
  - **[Chrome 에 추가]**

[설치] Node.js

https://nodejs.org/ko/download/

- Node.js: javascript 런타임 환경(runtime)
  - 자바스크립트로 이루어진 서버 사이드 언어
  - V8 엔진: 구글이 크롬 성능향상을 위해 만든 자바스크립트 엔진을 오픈소스로 공개



NPM 사용

node –v
npm -v

[설치] 웹 에디터: VS Code

https://code.visualstudio.com/

❖ 설치환경:

1. Chrome
2. Node.js
3. Visual Studio code
4. **React Developer devtools** ⇒ Chrome 확장 프로그램

❖ VSCode 확장 프로그램:

- **React Code Snippet**:
  - **rcc <tab> 키: 클래스 컴포넌트**
  - **rsc <tab> 키: 화살표함수형**
  - **rsf <tab> 키: 함수형 컴포넌트**
- **Material icon Theme**
  - 기본설정>파일 아이콘 테마> **Material icon Theme**

✓ **live server**

<Ctrl+Shift+X> live server 검색

단축키 <Alt+L+O> 웹서버가 동작 중인 브라우저 창이 열린다

<Alt+L+O> live server start

<Alt+L+C> live server stop

[실습] 첫 번째 react.js

CRA: create-react-app

- CRA: create-react-app
  - 보일러 플레이트(boiler-plate)
  - 개발을 바로 시작할 수 있도록 만든 개발환경 구성
  - 리액트 개발을 시작할 수 있도록 프로젝트 구조, 설정 작업 등을 자동으로 진행해 주는 도구

- npx
  - 1회성으로 최신버전의 노드 패키지를 내려 받아 설치해 주는 노드 패키지
- npm
- yarn

[실습] 첫 번째 react.js

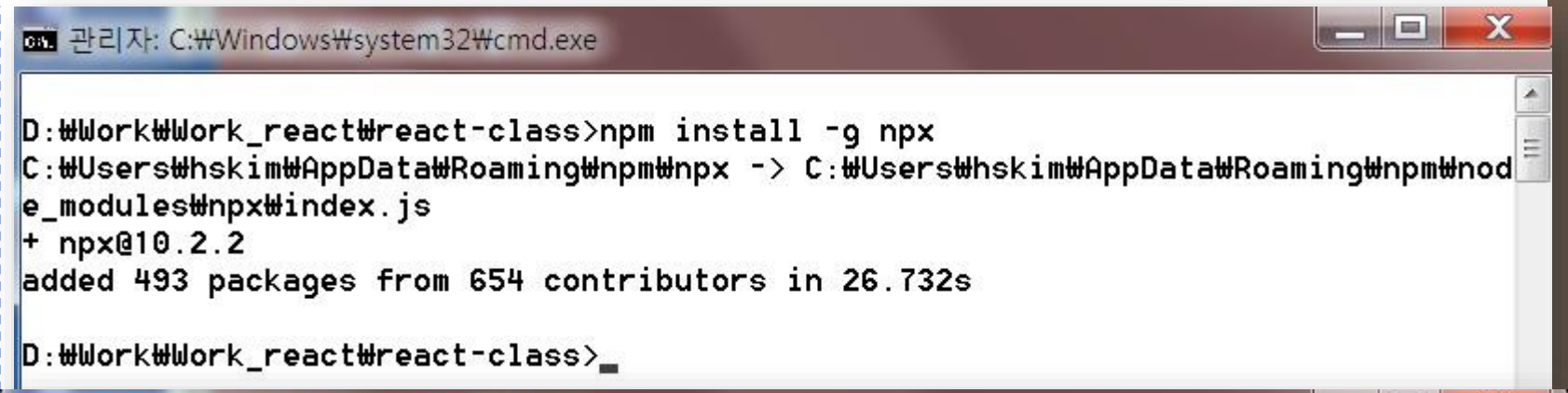
CRA: create-react-app

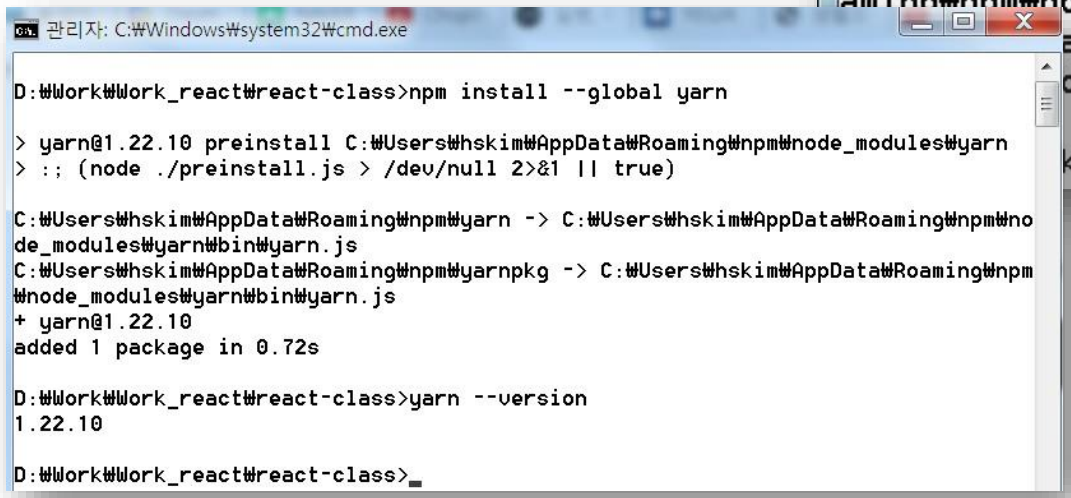
npm install -g npx

npm install -g create-react-app

(yarn 1.22 검색)
npm install --global yarn

yarn --version



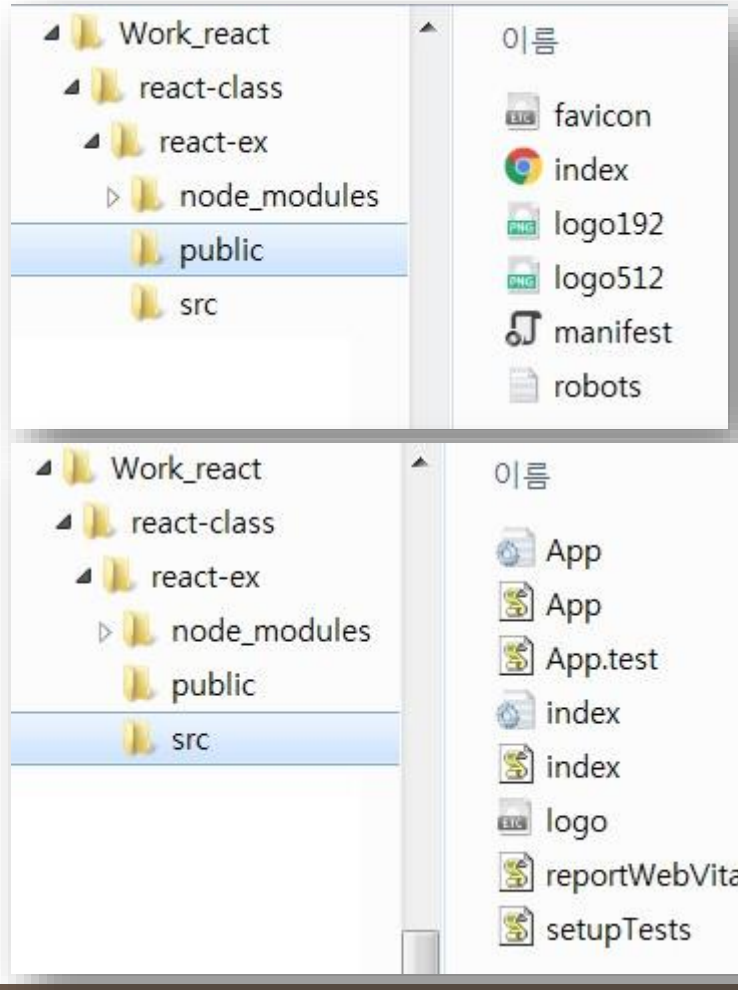


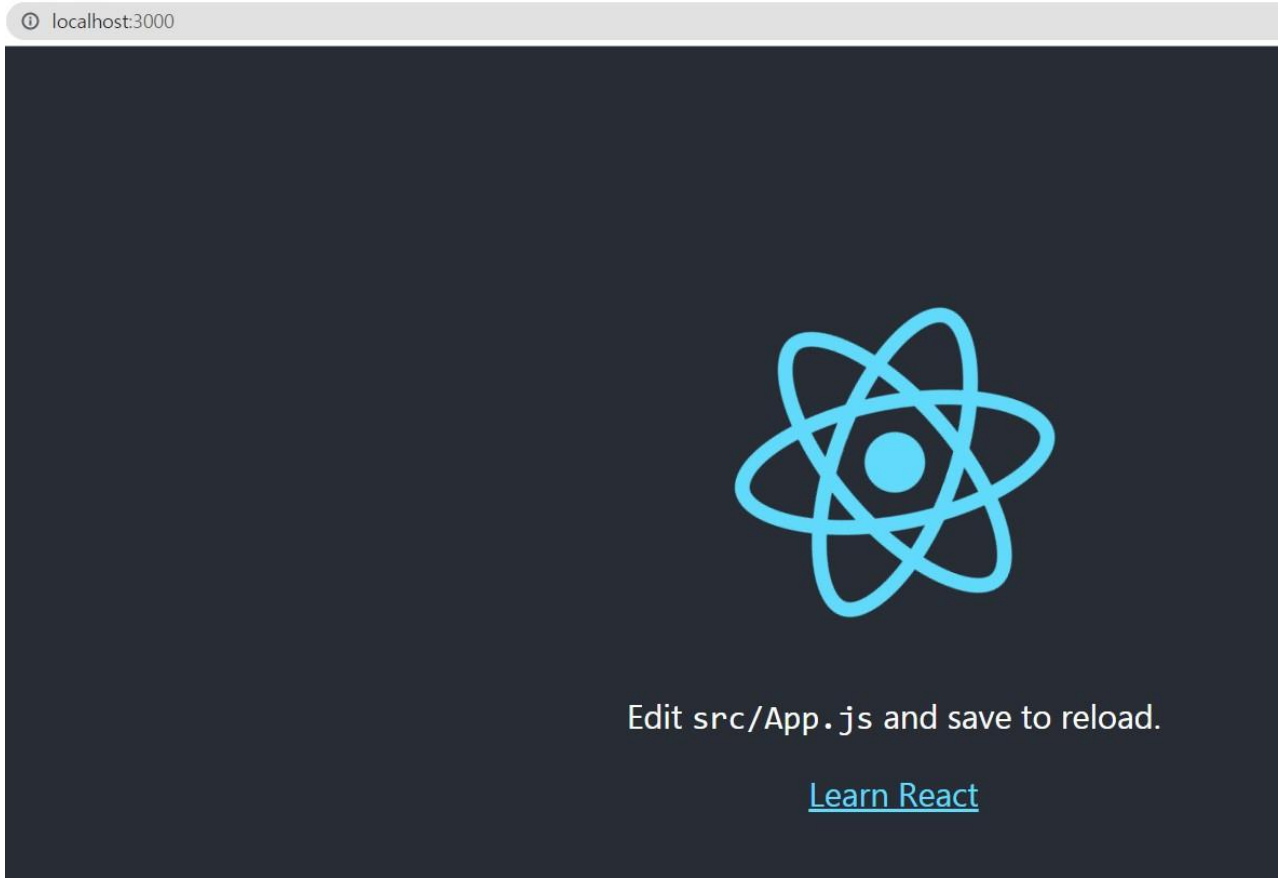
[실습] 첫 번째 react.js

CRA: create-react-app

```
# 첫 번째 react 예제
npx create-react-app react-ex
```

```
npm start
```





[실습] [.src/index.js][./src/App.js][./public/index.html]

Hello React

App.js

```
// ./src/App.js
import React from 'react';
import './App.css';

function App() {
  return (
    <div>Hello React</div>
  );
}

export default App;
```

index.js

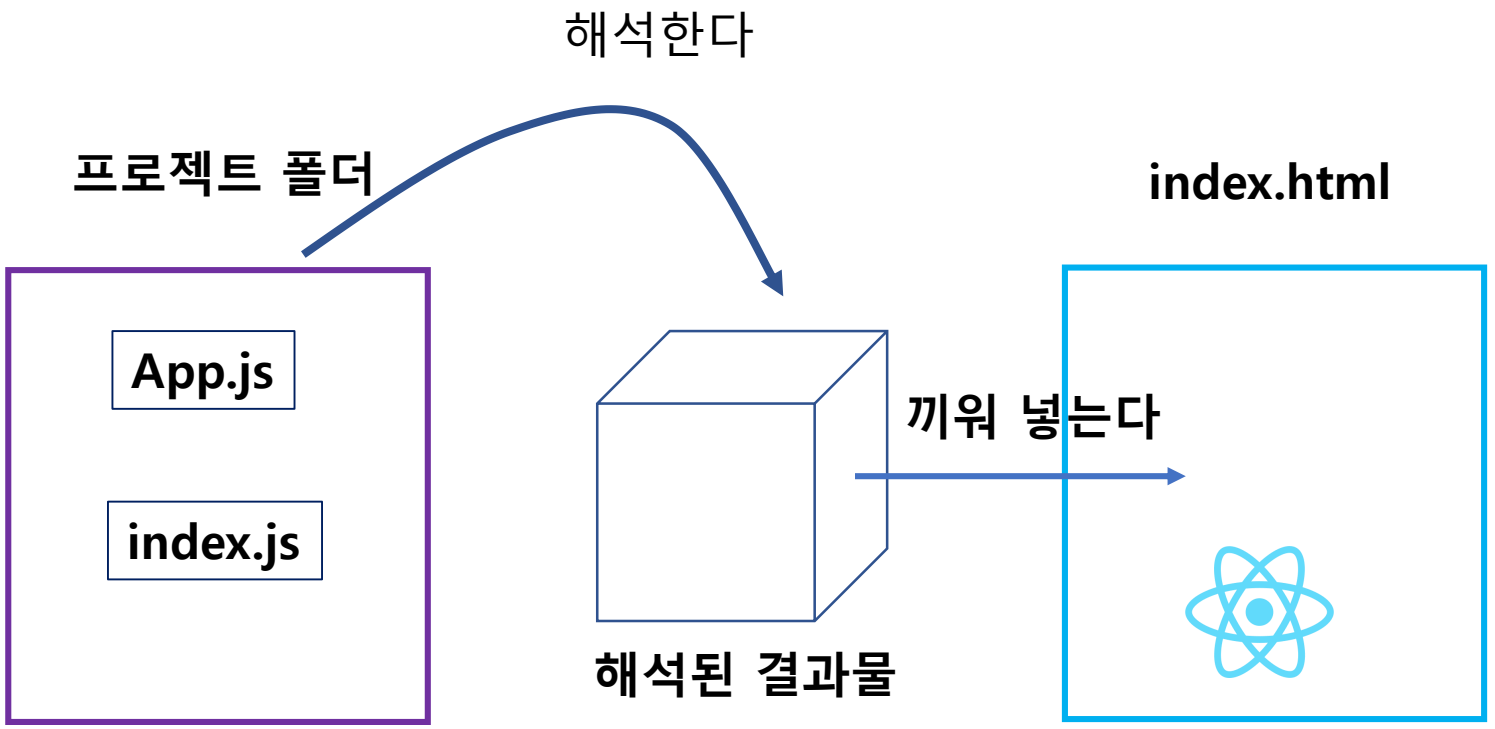
```
// ./src/index.js

import React from 'react';
import ReactDOM from 'react-dom';

import App from './App';

ReactDOM.render(
    <App />, document.getElementById('root')
);
```

index.html

```
// ./public/index.html
<!DOCTYPE html>
<html lang="en">
  <head>
중간생략
  </head>
  <body>
    <noscript>You need to enable JavaScript to run this app.</noscript>
    <div id="root"></div>
  </body>
</html>
```

[동작] React 동작원리

동작원리



❖ **React**

index.html 의

중간에 넣을 결과물을

프로젝트 폴더 파일
App.js 을 해석하여
만들어 넣는다

컴포넌트

담당: 김희숙
(jasmin11@hanmail.net)

[실습] [.src/index.js][./src/App.js][./public/index.html]

Hello React

App.js

```
// ./src/App.js
import React from 'react';
import './App.css';

function App() {
  return (
    <div>Hello React</div>
  );
}

export default App;
```

index.js

```
// ./src/index.js

import React from 'react';
import ReactDOM from 'react-dom';

import App from './App';

ReactDOM.render(
    <App />, document.getElementById('potato')
);
```

index.html

```
// ./public/index.html
<!DOCTYPE html>
<html lang="en">
  <head>
중간생략
  </head>
  <body>
    <noscript>You need to enable JavaScript to run this app.</noscript>
    <div id="potato"></div>
  </body>
</html>
```

예제) potato

[실습] 컴포넌트

컴포넌트

```
# potato 컴포넌트 만들기
npx create-react-app react-ex01-01
```

```
// ../src/App.js
ReactDOM.render(                                          // 1)
    <App />, document.getElementById('root')      // 2)
);
```

// 1) ReactDOM.render() 첫 번째 인자 → 컴포넌트가 반환하는 것들을 화면에 그린다

// 1) ReactDOM.render() 두 번째 인자 → 함수를 해석해서 그려질 위치 지정

// 2) <App /> HTML 태그가 아니다 → JSX

// 2) <App /> 컴포넌트

[요약] React

- ❖컴포넌트
- ❖JSX
- ❖props

[실습] [.src/index.js][./src/App.js][./public/index.html]

Potato 컴포넌트

index.js (오류)

```
// ./src/index.js
// 오류

import React from 'react';
import ReactDOM from 'react-dom';

import App from './App';

ReactDOM.render(
    <App /> < Potato />,
document.getElementById('root')
);
```

Potato.js

```
// ./src/Potato.js
import React from 'react'

function Potato() {
    return <h3>I like potato</h3>
}

export default Potato;
```

App.js (성공)

```
// ./src/App.js
import React from 'react';
import './App.css';
import Potato from './Potato'
function App() {
  return (
    <div>
      <h1>Hello React</h1>
      <Potato />
    </div>
}

export default App;
```

props

담당: 김희숙
(jasmin11@hanmail.net)

[실습] props

❖ props

- 컴포넌트에서 컴포넌트로 전달하는 데이터
- 컴포넌트로 데이터를 보내는 방법
- 리액트 컴포넌트로 넘어가는 매개변수

 # react 예제 (props)
npx create-react-app react-ex01-02-props

Tip) props 에 있는 데이터는
문자열 제외하고
모두 중괄호 { } 로 감싸야 한다

[실습] [.src/index.js][./src/App.js][./public/index.html]

Potato

App.js

```
// ./src/App.js
import React from 'react';
import './App.css';

function Potato() {
  return <h3>I Like Potato!</h3>
}

function App() {
  return (
    <div>
      <h1>Hello! </h1>
      <Potato />
    </div>
  );
}

export default App;
```

App.js 수정

```
// ./src/App.js
import React from 'react';
import './App.css';

function Potato() {
  return <h3>I Like Potato!</h3>
}

function App() {
  return (
    <div>
      <h1>Hello! </h1>
      <Potato />
      <Potato />
      <Potato />
      <Potato />
    </div>
  );
}

export default App;
```

[실습] [.src/index.js][./src/App.js][./public/index.html]

props

App.js

```
// ./src/App.js
import React from 'react';
import './App.css';

function Food() {
  return <h3>I Like Potato!</h3>
}

function App() {
  return (
    <div>
      <h1>Hello! </h1>
      <Food />
      <Food />
      <Food />
      <Food />
      <Food />
    </div>
  );
}

export default App;
```

App.js 수정

```
// ./src/App.js
import React from 'react';
import './App.css';

function Food(props) {
  return <h3>I Like {props.fav}!</h3>
}

function App() {
  return (
    <div>
      <h1>Hello! </h1>
      <Food fav = "kimchi" />
    </div>
  );
}

export default App;
```

[실습] [.src/index.js][./src/App.js][./public/index.html]

Food

App.js

```
// ./src/App.js
import React from 'react';
import './App.css';

function Food(props) {
  return <h3>I Like {props.fav}!</h3>
}

function App() {
  return (
    <div>
      <h1>Hello! </h1>
      <Food fav = "kimchi" />
    </div>
  );
}

export default App;
```

App.js 수정

```
// ./src/App.js
import React from 'react';
import './App.css';

function Food(props) {
  return <h3>I Like {props.fav}!</h3>
}

function App() {
  return (
    <div>
      <h1>Hello! </h1>
      <Food fav = "kimchi" />
      <Food fav = “ramen” />
      <Food fav = “samgiopsal” />
      <Food fav = “chukumi” />
    </div>
  );
}

export default App;
```

// 컴포넌트
// 재사용

[실습] [.src/index.js][./src/App.js][./public/index.html]

App.js

```
// ./src/App.js
import React from 'react';
import './App.css';

function Food(props) {
  return <h3>I Like {props.fav}!</h3>
}

function App() {
  return (
    <div>
      <h1>Hello! </h1>
      <Food fav = "kimchi" />
    </div>
  );
}

export default App;
```

App.js 수정

```
// ./src/App.js
import React from 'react';
import './App.css';

function Food({fav}) {
  return <h3>I Like {fav}!</h3>
}

function App() {
  return (
    <div>
      <h1>Hello! </h1>
      <Food fav = "kimchi" />
    </div>
  );
}

export default App;
```

map() 함수

❖ map() 함수

- 배열의 모든 원소마다 특정 작업을 하는 함수를 작용하고 그 함수가 반환한 결과를 모아서 배열로 반환한다

```
>const friends = ["dal", "mark", "lynn"]
>friends

>friends.map(current => {
   console.log(current);
   return 0;
});
```

```
// 이름 뒤에 하트 붙이기

friends.map(function(friend) {
  return friend +  " ♥";
})
```

[실습] [.src/index.js][./src/App.js][./public/index.html]

App.js

```
// const foodILike = [];
const foodILike = [
  {
    name: 'Kimchi',
    image: 'http://aeriskitchen.com/wp-content/uploads/2008/09/kimchi_bokkeumbap_02-.jpg'
  },
  {
    name: 'Samgyeopsal',
    image:
      'https://3.bp.blogspot.com/-
hKwIBxIVcQw/WfsewX3fhJI/AAAAAAAAALk/yHxnxFXcfx4ZKSfHS_RQNKjw3bAC03AnACLcBGAs/s400/DSC07624.jpg'
  },
  {
    name: 'Bibimbap',
    image:
      'http://cdn-image.myrecipes.com/sites/default/files/styles/4_3_horizontal_-
_1200x900/public/image/recipes/ck/12/03/bibimbop-ck-x.jpg?itok=RoXlp6Xb'
  },
 {
    name: 'Doncasu',
    image: 'https://s3-media3.fl.yelpcdn.com/bphoto/7F9eTTQ_yxaWIRytAu5feA/ls.jpg'
  },
  {
    name: 'Kimbap',
    image: 'http://cdn2.koreanbapsang.com/wp-content/uploads/2012/05/DSC_1238r-e1454170512295.jpg'
  },
];
```

react 예제 (props)
npx create-react-app react-ex01-03-props

[실습] [.src/index.js][./src/App.js][./public/index.html]

map()

App.js

```
// ./src/App.js
import React from 'react';
import './App.css';

function Food({name}) {
  return <h3>I Like {name}!</h3>
}

// 생략

function App() {
  return (
    <div>
      <h1>Food</h1>
      {foodILike.map(dish => (<Food name = {dish.name} />))}
    </div>
  );
}

export default App;
```

react 예제 (props)
npx create-react-app react-ex01-03-props

[실습] [.src/index.js][./src/App.js][./public/index.html]

map()

App.js

```
// ./src/App.js
import React from 'react';
import './App.css';

function Food({name, picture}) {
  return (
     <div>
        <h3>I Like {name}!</h3>
        <img src = {picture}  width = "50%" alt = {name} />
      </div>
  )
}
// 생략

function App() {
  return (
    <div>
      <h1>Food</h1>
      {foodILike.map(dish => (
           <Food name = {dish.name} picture = {dish.image} />
       ))}
    </div>    </div>
  );
}

export default App;
```

react 예제 (props)
npx create-react-app react-ex01-03-props