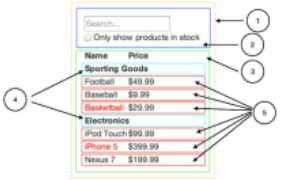
24.04.05

Component



컴포넌트

- Component : 입력(props)을 받아 출력(Element) 하는 역할
- 리액트는 Component 기반의 구조라는 특징을 가지고 있다.
- 리액트는 모든 페이지가 Component로 구성되어 있고 하나의 Component는 또 다른 여러 개의 Component의 조합으로 구성될 수 있다.
- 이러한 Component들을 마치 레고 블록을 조립하듯 끼워 맞춰 새로운 Component를 만들 수 있다.
- 프로그래밍에 있어 재사용이 가능한 각각의 독립된 모듈을 뜻한다
- 리액트로 화면을 구성하게 되면, 사용자가 볼 수 있는 여러 가지 컴포넌트로 구성되어 있다. 사용자에게 보여지는 UI 요소를 컴포넌트 단위로 구분하여 구현할 수 있다.
- Components are like functions that return HTML elements.
- 웹 페이지를 만드는 퍼즐 조각
- 개발자는 컴포넌트 조각을 설계하고 만든 컴포넌트를 조합해서 사용자 인터페이스(user interface, UI)를 구축한다. UI 조각인 컴포넌트를 모으면 전체 퍼즐 그림인 웹 페이지를 만들 수 있다.
- 자바스크립트의 함수처럼 작동해서 리액트 엘리먼트를 반환한다.
- 어떤 데이터 집합을 사용하든 같은 컴포넌트를 사용하면 모두 동일한 DOM 구조가 반환된다.

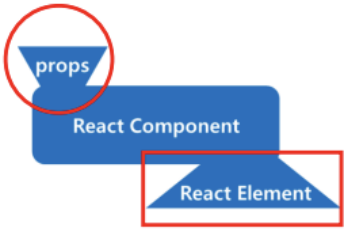
C - 함수로 구성

리엑트 - component 기반

재사용 가능

Component

자바 스크립트 컴포넌트와 리액트 컴포넌트



리액트 컴포넌트

- React Component에서의 입력은 props며 출력은 React Element가 된다.
- 결국 React Component가 해주는 역할은 어떠한 속성들을 입력으로 받아서 그에 맞는 React Element를 생성하여 리턴해 주는 것이다.
- React Element는 리액트 앱을 구성하는 가장 작은 빌딩 블록들이다.
- 자바스크립트 객체 형태로 존재하며 화면에 보이는 것을 기술한다.
- React Component를 만들고자 하는 대로 props, 즉 속성을 넣으면 해당 속성에 맞춰 화면에 나타날 Element를 만들어준다

함수로 구성되어있음 - 입출력

리엑트는 속성말고 입력을 받지X

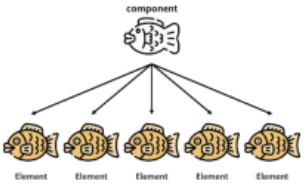
리엑트 element = 컴포넌트로 구성 ⇒ 최종적으로 html로 변환

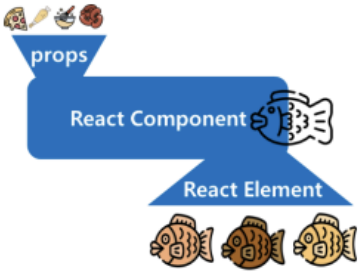
Component

리액트 컴포넌트

- props는 Component의 입력으로 들어간다.
- prop는 property의 의미로 재산이라는 뜻도 있지만 속성, 특성이라는 뜻도 가지고 있다.
- property를 줄여 prop이 되었고 여러 개라는 의미로 복수형 props가 되었다.
- 리액트에서 props는 리액트 컴포넌트의 속성이다.

	붕어빵 틀	Componenet
	붕어빵	Element
	붕어빵 재료	props



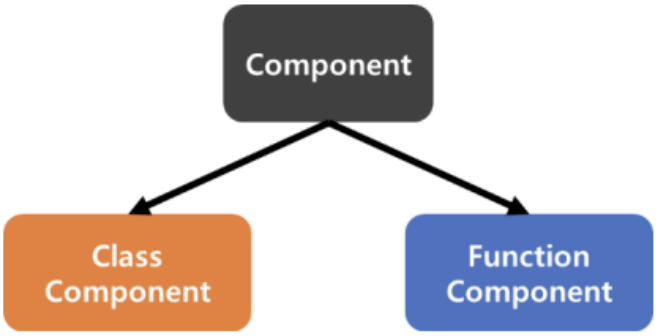


- 같은 붕어빵이라도 넣는 재료에 따라 다른 맛이 난다.
- 이처럼 props는 같은 React Component에서 눈에 보이는 글자나 색깔 등의 속성을 바꾸고 싶을 때는 사용하는 컴포넌트의 속 재료이다.
- 결국 React Component가 해주는 역할은 어떠한 속성들을 입력으로 받아서 그에 맞는 React Element를 생성하여 리턴해 주는 것이다.

Component

리액트 컴포넌트

- React에서 Componenet는 Class Componenet와 Function Component로 나뉜다.



Component

Function Component

- React Component는 pure 함수 같은 역할을 해야 한다.
- 즉, 리액트의 Component를 일종의 함수라고 생각한다.

```
function Welcome(props){
        return <h1>안녕, {props.name}</h1>;
}
```

- 이 함수는 하나의 props객체를 받아 인사말이 담긴 React element를 리턴하기 때문에 React Component라고 할 수 있다.
- 이런 것을 Function Component라고 한다.
- Function Component는 코드가 간단하다는 장점을 가지고 있다.

pure 함수 - 내가 이 함수를 호출하면 무조건 같은 결과를 출력해야한다

inpure 함수 - 실행시킬때마다 결과값이 달라진다 → 전역변수의 값을 변경시킨다 / 외부의 값을 변경시킨다

JS 클래스와 함수 컴포넌트를 사용할 수 있다 (함수 컴포넌트를 더 많이 사용)

, {props.name}</

리엑트 Component (DOM element → html 태그)

Component

Class Component

- 자바스크립트 ES6의 class라는 것을 사용해서 만들어진 형태의 Component다.
- Class Component의 경우 Function Component에 비해 몇 가지 추가적인 기능을 가지고 있다.

```
class Welcome extends React.Component{
        render(){
            return <h1>안녕, {this.props.name}</h1>;
    }
}
```

- 이 함수는 Function Component를 Class Component로 만든 것이다.

Component

Component name

- Component의 이름은 항상 대문자로 시작해야 한다.
- 리액트는 소문자로 시작하는 Component를 DOM 태그로 인식하기 때문이다.
- div, span과 같이 사용하는 것은 내장 컴포넌트라는 것을 뜻하며 div나 span과 같은 문자열 형태로 react.createElement에 전달된다.
- Component 이름이 소문자로 시작했다면 리액트는 내부적으로 이것을 Component가 아닌 DOM 태그로 인식한다.
- 그러므로 항상 컴포넌트의 이름은 대문자로 시작해야 한다.

```
class Car extends React.Component {
  render() {
    return <h2>Hi, I am a Car!</h2>;
  }
}
```

```
function Car() {
  return <h2>Hi, I am a Car!</h2>;
}
```

모든 HTML 컴포넌트

```
npx create-react-app comp
```

react파일에서 생성 (폴더이름 무조건 소문자)

헤더파일 열고 rfc

헤더입니다

안녕하세요! 메인입니다.

푸터입니다

```
import logo from './logo.svg';
import './App.css';
import Header from './funcComp/Header';
import Main from './funcComp/Main';
import Footer from './funcComp/Footer';
```

```jsx
function App() {
  return (
    <div className="App">
      <Header/>
      <Main/>
      <Footer/>
    </div>
  );
}

export default App;
```

```jsx
import React from 'react'

export default function Header() {
  return (
    <div>
        <header>
            <h1>헤더입니다</h1>
        </header>
    </div>
  )
}

-----------------------------------------------

import React from 'react'

export default function Main() {
  return (
    <div>
        <main>
            <h1>안녕하세요! 메인입니다.</h1>
        </main>
    </div>
  )
}
```

```
-----------------------------------------------

import React from 'react'

export default function Footer() {
  return (
    <div>
        <footer>
            <h1>푸터입니다</h1>
        </footer>
    </div>
  )
}
```

헤더입니다

메인입니다

푸터입니다

```
import logo from './logo.svg';
import './App.css';
import Header from './classComp/Header';
import Main from './classComp/Main';
import Footer from './classComp/Footer';

function App() {
  return (
    <div className="App">
```

```
      <Header/>
      <Main/>
      <Footer/>
    </div>
  );
}

export default App;
```

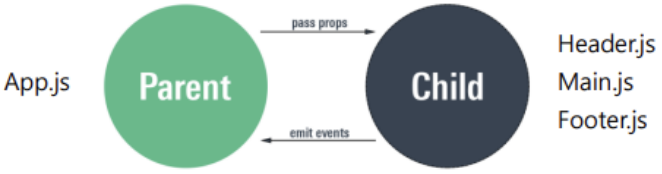
```
import React, { Component } from 'react'

export default class Header extends Component {
  render() {
    return (
        <div>
        <header>
            <h1>헤더입니다</h1>
        </header>
    </div>
    )
  }
}

-----

import React, { Component } from 'react'

export default class Main extends Component {
  render() {
    return (
        <div>
        <header>
            <h1>메인입니다</h1>
        </header>
    </div>
    )
  }
}
```

```
-----

import React, { Component } from 'react'

export default class Footer extends Component {
  render() {
    return (
        <div>
        <header>
            <h1>푸터입니다</h1>
        </header>
    </div>
    )
  }
}
```

Props

프로퍼티(속성)

- 프로퍼티, props(properties의 줄임말)이다.
- 상위 컴포넌트가 하위 컴포넌트에 값을 전달할때 사용한다.(단방향 데이터 흐름 갖는다.)
- 프로퍼티는 수정할 수 없다는 특징이 있다.(자식입장에선 읽기 전용인 데이터이다.)
- React 컴포넌트는 props를 이용해 서로 통신한다.
- 모든 부모 컴포넌트는 props를 줌으로써 몇몇의 정보를 자식 컴포넌트에게 전달할 수 있다.
- 부모 컴포넌트가 자식 컴포넌트에게 물려준 데이터를 의미한다.



컴포넌트 안에 컴포넌트를 품을 수 있다

→ app.js가 header, main, footer를 품었음

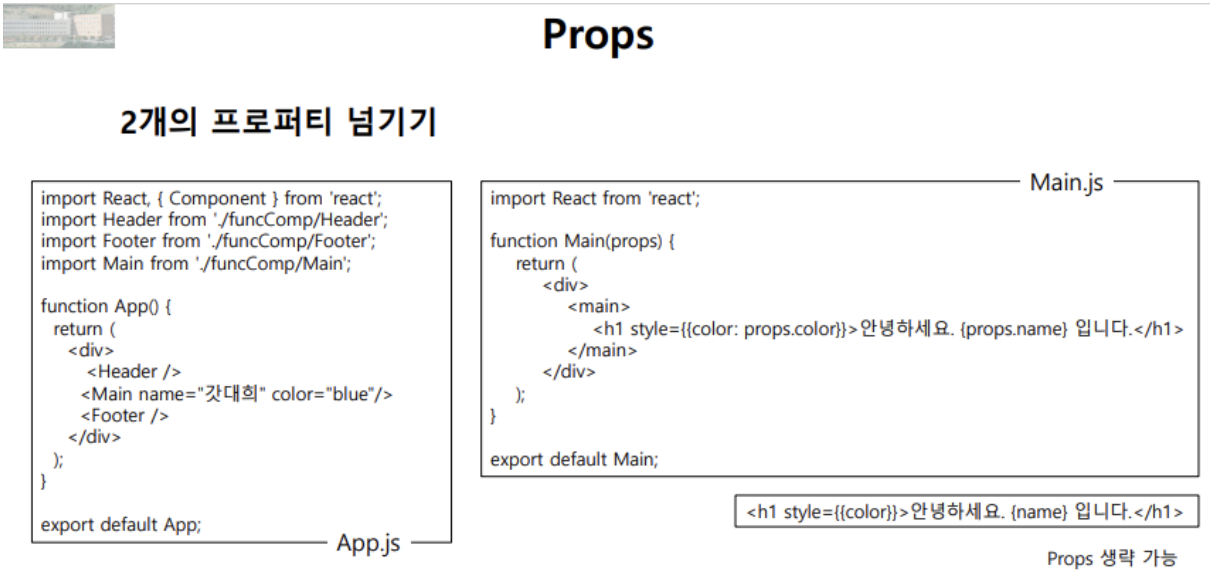
컴포넌트의 입력은 항상 props

컴포넌트 자체 내에서 값을 처리할 수 있다 - state

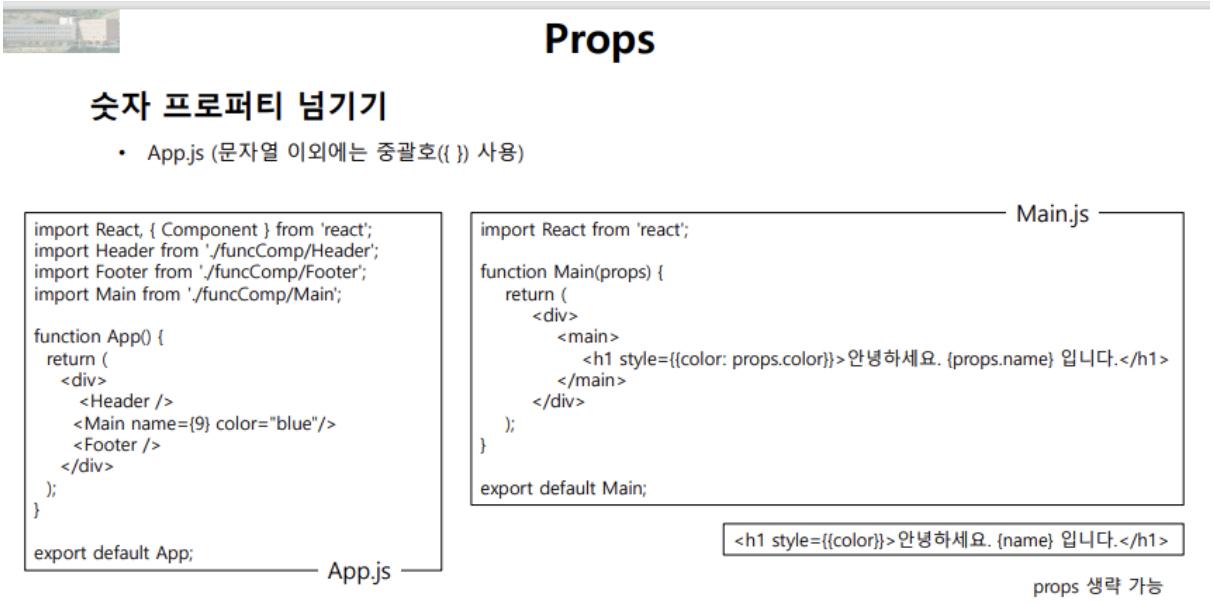
상위 컴포넌트에서 하위로 값을 전달할 수 있는데, 단방향으로 이루어짐

상위 → 하위

변수 명 주고 할당시키면됨



중괄호 사용



Props

프로퍼티의 자료형, 타입정의

- 프로퍼티의 자료형을 미리 선언할 수 있다.
- 리액트 엔진이 프로퍼티로 전달하는 값을 효율적으로 알 수 있고, 버그예방에도 도움이 된다.
- 리액트에서 제공하는 prop-types를 이용하여 각각의 자료형을 선언하면 된다.

```
import React from 'react';
import PropTypes from 'prop-types' // 프로퍼티 타입을 지정해주기 위해 사용 한다.

function Main({name, color}) {
    return (
        <div>
            <main>
                <h1 style={{color}}>안녕하세요. {name} 입니다.</h1>
            </main>
        </div>
    );
}

// 프로퍼티 타입 지정
Main.propTypes = {
  name: PropTypes.string
}

export default Main;
```

```
// 프로퍼티 타입 지정
Main.propTypes = {
  name: PropTypes.string
}
```

name을 타입 string으로 전달 할거다

Props

프로퍼티의 기본값 설정

- 기본값 설정
- 컴포넌트에 props 기본값을 설정하고 싶은 경우 defaultProps를 설정하면 된다.
  ex) Main.js (name 프로퍼티가 없는 경우, '디폴트'라는 값을 사용 하게 처리)

```
import React from 'react';
import PropTypes from 'prop-types' // 프로퍼티 타입을 지정해주기 위해 사용 한다.
function Main({name, color}) {
    return (
        <div>
            <main>
                <h1 style={{color}}>안녕하세요. {name} 입니다.</h1>
            </main>
        </div>
    );
}
// 프로퍼티 타입 지정
Main.propTypes = {
  name: PropTypes.string
}
// 프로퍼티 기본값 지정
Main.defaultProps = {
  name: '디폴트'
}
export default Main;
```

```
import React, { Component } from 'react';
import Header from './funcComp/Header';
import Footer from './funcComp/Footer';
import Main from './funcComp/Main';

function App() {
  return (
    <div>
      <Header />
      <Main color="blue"/>
      <Footer />
    </div>
  );
}

export default App;
```

Props

불리언 프로퍼티 사용하기

- true, false만 정의 가능한 자료형
- 중괄호로 감싸 전달할 필요 없이 프로퍼티의 이름만 선언하면 된다.

```
import React from 'react';
import PropTypes from 'prop-types'
function Main({color, name, maleYn}) {
    const msg = maleYn ? '남자' : '여자'; // 불리언 사용
    return (
        <div>
            <main>
                <h1 style={{color}}>안녕하세요. {name} 입니다. ({msg})</h1>
            </main>
        </div>
    );
}
Main.propTypes = {
    name: PropTypes.string
}
Main.defaultProps = {
    name: '디폴트'
}
export default Main;
```

```
import React, { Component } from 'react';
import Header from './funcComp/Header';
import Footer from './funcComp/Footer';
import Main from './funcComp/Main';

function App() {
  return (
    <div>
      <Header />
      <Main name="홍길동" color="blue" maleYn/>
<Footer />
    </div>
  );
}

export default App;
```

App.js에서 mailYn을 생략하면 false로 처리한다.
<Main name= " 홍길동" color="blue"/>


Desktop


Notebook


Pad

```
import React from 'react';
function FPhotoText(props) {
    const url = "img/"+ props.image + ".png"
    const label = props.label
    const boxStyle = {
        border: "1px solid silver",
        margin: "8px",
        padding: "4px"
    }
    return (
        <div style={boxStyle}>
            <img src={url} width="128" alt="computer"/>
            <span>{label}</span>
```

```
        </div>
    );
}
export default FPhotoText
```

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

function App() {
  return (
    <div className="App">
      <FPhotoText image = "desktop" label = "Desktop" />
      <FPhotoText image = "notebook" label = "Notebook" />
      <FPhotoText image = "pad" label = "Pad" />
    </div>
  );
}

export default App;
```