

230116_2반_실습

React

Component

component란?

- 리액트로 만들어진 앱을 이루는 최소한의 단위
- 컴포넌트는 데이터(props)를 입력받아 View(state) 상태에 따라 DOM Node를 출력하는 함수.
- 컴포넌트 이름은 항상 대문자로 시작하도록 한다.
- UI를 재사용 가능한 개별적인 여러 조각으로 나누고, 각 조각을 개별적으로 나누어 코딩한다.

component의 종류

1. 함수형 컴포넌트 (Stateless Functional Component)

- 다양한 Function 형태로 컴포넌트화할 수 있습니다.

```
const title = "기능 컴포넌트화하기";
function Desc() {
  return (
    <h2>Function 형태로 컴포넌트화할 수 있습니다.</h2>
  );
}
const Body = () => <p> 본문 내용이 나오는 영역입니다. </p>;
const Footer = () => {return <p> Footer 영역입니다.</p>;};

// 함수형 컴포넌트
function App() {
  return (
    <>
      <h1>{title}</h1>
      <Desc></Desc>
      <Body />
      <Footer />
    </>
  );
}

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

2. 클래스형 컴포넌트 (Class Component)

- 컴포넌트 구성 요소, 리액트 생명주기를 모두 포함하고 있습니다.
- Class 형태로도 컴포넌트화할 수 있습니다.

```
const title = "기능 컴포넌트화하기";
function Desc() {
  return (
    <h2>Function 형태로 컴포넌트화할 수 있습니다.</h2>
  );
}
const Body = () => <p> 본문 내용이 나오는 영역입니다. </p>;
const Footer = () => {return <p> Footer 영역입니다.</p>;};

// 클래스 형 컴포넌트
class App extends React.Component {
  render() {
    return (
      <>
        <h1>{title}</h1>
        <Desc></Desc>
        <Body />
        <Footer />
      </>
    );
  }
}

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

props

- 프로퍼티, props(properties의 줄임말) 라고 한다.
- 상위 컴포넌트가 하위 컴포넌트에 값을 전달할때 사용한다.
- 프로퍼티는 수정할 수 없다는 특징이 있다.
- Props를 이용하여 컴포넌트에 값을 넘길 수 있습니다.

```
const Body = (props) => {
  return
    <>
      <p> 제목 : {props.title}</p>;
      <p> 내용 : {props.desc}</p>;
    </>
};

function App() {
  return (
    <Body title="제목입니다." desc="설명입니다." />
  );
}
```

```
};
```

Event

bind() 함수

- 기본적으로 render 함수 안에서 `this` 는 render 함수가 속해있는 그 Component 자체를 가리킨다.
- `strict mode` 가 적용된 일반 함수 내부의 `this` 에서는 전역객체인 window객체가 아닌 `undefined` 가 바인딩 되는데, class 내부는 암묵적으로 `strict mode` 가 적용되기 때문에 여기서 `this` 는 `undefined` 로 바인딩 된다.



strict mode

애플리케이션 내의 잠재적인 문제를 알아내기 위한 도구로, strict 모드는 개발 과정 중에만 적용 된다. 배포가 되고나면 strict 모드는 저절로 작동하지 않는다.

- `Arrow function` 에서는 `bind`, `apply`, `call` 을 사용 할 수 없다.

Hook

useState()

- `setState()`는 리액트의 함수형 컴포넌트 내에서 상태를 관리하기 위해 사용하는 hooks 인 `useState()`를 통해 반환되는 함수이다.
constructor 안에서 state값을 바꾸는 것은 가능 하지만 생성된 후 state값을 바꾸는 방법은 `setState` 를 사용한다.

```
this.state.mode = 'welcome'  
//안됨 -> 리액트 모르게 바꾸는 것 -> 렌더링이 안됨  
  
this.setState({  
  mode: 'welcome'  
})  
//리액트에게 알려줘서 다시 렌더링된다.
```

setState()의 특징

1. 기본적으로 비동기로 동작한다.

2. 연속적으로 호출했을 때 리액트 내부적으로는 BATCH 처리를 한다.
3. state 객체를 넘겨줄 수 있을 뿐만 아니라 새로운 state를 반환하는 함수를 인자로 넘겨 줄 수 있다.

setState 사용법

1. import

```
import React, { useState } from 'react';
```

2. 선언 방법

```
const [state, setState] = useState(initialState);  
const [ 데이터, 데이터변경함수 ] = useState(초기값(생략가능));
```

- react 모듈에서 { useState }를 불러오고 useState()를 선언해서 사용하면 된다.
- useState의 변수값이 바뀌면 컴포넌트가 새롭게 렌더링 된다.
- 예제

```
const [count, setCount] = useState(0);
```

```
import React, { useState } from 'react';  
  
const Main = () => {  
  const [ cnt, setCnt ] = useState(0)  
  const updateCnt = () => setCnt(cnt + 1);  
  const clearCnt = () => setCnt(0);  
  return (  
    <div>  
      클릭한 횟수는 {cnt}번 입니다.  
      <div>  
        <button onClick={updateCnt}> 클릭해 보세요! </button>  
        <button onClick={clearCnt}> 초기화 하기! </button>  
      </div>  
    </div>  
  );  
};  
  
export default Main;
```

```

import React, { useState } from 'react';

const Main = () => {
  // let myName = "Snoopy"; // useState를 사용하여 변경
  const [ myName, setMyName ] = useState("Snoopy");

  function changeName() {
    /*
    myName = myName === "Snoopy" ? "Woodstock" : "Chalri";
    console.log(myName);
    document.getElementById("name").indderText = myName;
    */
    setMyName(myName = myName === "Snoopy" ? "Woodstock" : "Chalri");
  }

  return (
    <div>
      <h1>안녕하세요. {myName} 입니다.</h1>
      { /* <button
        onClick={() => {
          setMyName(myName = myName === "Snoopy" ? "Woodstock" : "Chalri");
        }}
      >Change</button> */ }
      <button onClick={changeName}>Change</button>
    </div>
  );
};

export default Main;

```

3. 객체(Object)도 상태변수(State Value)로 사용 가능하다.

```

import React, { useState } from 'react';

const Main = () => {
  const [ state, setState ] = useState({cnt : 0})
  const updateCnt = val =>
    setState({
      ...state,
      [val] : state[val] + 1
    })
  const { cnt } = state
  return (
    <div>
      클릭한 횟수는 {cnt}번 입니다.
      <div>
        <button onClick={updateCnt.bind(null, 'cnt')}> 클릭해 보세요! </button>
      </div>
    </div>
  );
};

export default Main;

```

4. useState를 호출하여 반환된 업데이트 함수는 setState와 유사하게 사용 가능 하다.

```
import React, { useState } from 'react';

const Main = () => {
  const [ cnt, setCnt ] = useState(0)
  // const updateCnt = () => setCnt(cnt + 1);
  // const clearCnt = () => setCnt(0);
  return (
    <div>
      클릭한 횟수는 {cnt}번 입니다.
      <div>
        <button onClick={() => setCnt(prevCnt => prevCnt + 1)}> 클릭해 보세요! </
button>
        <button onClick={() => setCnt(0)}> 초기화 하기! </button>
      </div>
    </div>
  );
};

export default Main;
```

```
import React, { Component } from 'react';
import Main from './component/Main';
import Wrapper from './component/Wrapper';

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

export default App;
```

클릭한 횟수는 0번 입니다.

클릭해 보세요! 초기화 하기!

클릭한 횟수는 0번 입니다.

클릭해 보세요! 초기화 하기!

클릭한 횟수는 0번 입니다.

클릭해 보세요! 초기화 하기!

react-router-dom

- 사용자가 입력한 주소를 감지하는 역할을 하며, 여러 환경에서 동작할 수 있도록 여러 종류의 라우터 컴포넌트를 제공한다.
- 이중 가장 많이 사용하는 라우터 컴포넌트는 BrowserRouter와 HashRouter이다.
 - BrowserRouter : HTML5를 지원하는 브라우저의 주소를 감지 한다.
 - HashRouter 해시 주소(<http://www.test.com/#test>)를 감지 한다.
- 설치

```
npm i react-router-dom
```

- 패키지 불러오기

```
import React, {Component} from 'react';  
import { BrowserRouter, Route, Link, Switch } from "react-router-dom";
```


- **BrowserRouter** - history API를 사용해 URL과 UI를 동기화하는 라우터입니다.
 - <BrowserRouter>태그로 컴포넌트 사용.
 - Header는 모든 URL에 공통 적용할 Component로 최상단에 위치 할 예정이다.
- **Route** - 컴포넌트의 속성에 설정된 URL과 현재 경로가 일치하면 해당하는 컴포넌트, 함수를 렌더링한다.
 - path속성에 경로, element속성에는 컴포넌트를 넣어 준다. 여러 라우팅을 매칭하고 싶은 경우 URL 뒤에 *을 사용하면 된다.
- **Link** - 'a'태그와 비슷합니다. to속성에 설정된 링크로 이동합니다. 기록이 history 스택에 저장됩니다.
 - 문법 : <Link to="경로">링크명</Link>
 - import { Link } from 'react-router-dom';
- **Switch** - 자식 컴포넌트 Route또는 Redirect중 매치되는 첫 번째 요소를 렌더링합니다. Switch를 사용하면 BrowserRouter만 사용할 때와 다르게 **하나**의 매치되는 요소만 렌더링한다는 점을 보장해줍니다. 사용하지 않을 경우 매치되는 모두를 렌더링합니다. 사용하지 않을 경우 매치되는 모두를 렌더링합니다.

CSS

CSS: Cascading Style Sheets | MDN

MDN CSS 학습지 는 CSS를 처음부터 알려주는 모듈로 구성되어 있습니다. 사전 지식도 필요하지 않습니다. CSS(Cascading Style Sheets)는 웹 페이지에 스타일과 레이아웃을 적용할 때 사용합니다.

 <https://developer.mozilla.org/ko/docs/Web/CSS>

 mdn web docs