컴포넌트

안 화 수

컴포넌트

component ?

1. 리액트를 사용하여 애플리케이션의 인터페이스를 설계할때 사용자가 볼 수 있는 요소는 여러가지 컴포넌트로 구성된다.

2. 컴포넌트를 선언하는 방식은 클래스형 컴포넌트와 함수형 컴포넌트가 있다.

3. 클래스형 컴포넌트는 state 기능 및 라이프사이클 API를 사용할 수 있다.

4. 함수형 컴포넌트의 단점은 state와 라이프사이클 API를 사용할 수 없었지만, 리액트 v16.8 이후에 훅(Hooks)이라는 기능이 도입 되면서 해결 되었다.

5. 리액트 공식 매뉴얼에서는 함수형 컴포넌트와 Hooks을 사용하도록 권장하고 있다.

컴포넌트

component

6. 컴포넌트의 첫 글자는 항상 대문자로 시작 해야 한다.
   컴포넌트를 소문자로 시작하면 DOM 태그로 처리한다.

7. 컴포넌트를 이용해서 UI를 재사용 가능한 개별적인 여러 조각으로 나누고,
   각 조각을 개별적으로 나누어 코딩한다.

8. 자식 컴포넌트에게 값을 전달할때 props를 사용할 수 있다.

컴포넌트

- **프로젝트 생성**

```
npx    create-react-app    component01
```

컴포넌트

❖ **클래스형 컴포넌트** : src /App1.js

```
import React, {Component} from "react";
import './App.css';

// 클래스형 컴포넌트
class App1 extends Component{
    render(){
        const name = '클래스형 컴포넌트';
        return <div className="react">{name}</div>
    }
}

export default App1;
```

컴포넌트

- **함수형 컴포넌트** : src /App2.js

```
import React from "react";
import './App.css';

// 함수형 컴포넌트
function App2(){
    const name = '함수형 컴포넌트';
    return <div className="react">{name}</div>
}

export default App2;
```

JSX문법

- **외부 css파일 :** src/App.css

  App.css 파일에 아래의 내용을 추가한다.

```
.react{
        background-color : aqua;
        color : black;
        font-size : 48px;
        font-weight: bold;
        padding : 16px;
}
```

JSX문법

- **index.js 파일에서 컴포넌트 파일 import하기**

```
import App1 from './App1';
import App2 from './App2';

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
  <React.StrictMode>
      <App1 />
  </React.StrictMode>
);
```

컴포넌트

- **예1. 컴포넌트 나누기**

index.js – App.js – MyComponent.js
MyComponent2.js
MyComponent3.js
MyComponent4.js
MyComponent5.js
MyComponent6.js

컴포넌트

1. 함수형 컴포넌트 생성 : src/App.js

```
import React from 'react';
import MyComponent from  './MyComponent';
import MyComponent2 from './MyComponent2';
import MyComponent3 from './MyComponent3';
import MyComponent4 from './MyComponent4';
import MyComponent5 from './MyComponent5';
import MyComponent5 from './MyComponent6';

// 함수형 컴포넌트
//function App() {
  const App=()=>{
      return <MyComponent />
//    return <MyComponent2 name="React" />
//    return <MyComponent3 name="React">리액트</MyComponent3>
//    return <MyComponent4 name="React">리액트</MyComponent4>
//    return <MyComponent5 name="React" favoriteNumber={3}>리액트</MyComponent5>
//    return <MyComponent6 name="React" favoriteNumber={3}>리액트</MyComponent6>
  };
export default App;
```

컴포넌트

1. 함수형 컴포넌트 생성 : src/MyComponent.js

```
import React from 'react';

//function MyComponent(){
const MyComponent=()=>{ // ES6문법으로 정의한 화살표 함수(arrow function)
    return (
        <div><h1>함수형 컴포넌트 연습</h1></div>
    );
};

export  default  MyComponent;
```

props

❖ props

1. props는 프로퍼티(properties)의 줄임말 이다.
2. props는 부모 컴포넌트에서 자식 컴포넌트에게 데이터를 전달할 때 사용한다.
3. props를 전달받은 자식 컴포넌트에서는 데이터를 수정할 수 없다.

컴포넌트

2. 함수형 컴포넌트 생성 : src/MyComponent2.js
props를 이용해서 값전달 하기

```
import React from 'react';

// function MyComponent2(props){
const MyComponent2=(props)=>{
    return (
        <div>
            <h1>안녕 하세요?</h1>
            <h1>제 이름은 {props.name}입니다.</h1>
        </div>
    );
};

export default MyComponent2;
```

컴포넌트

3. 함수형 컴포넌트 생성 : src/MyComponent3.js
부모 컴포넌트 태그 사이의 값을 받은 때는 props.children으로 받는다.

```
import React from 'react';

// function MyComponent3(props){
const MyComponent3=(props)=>{
    return (
        <div>
            <h1>안녕 하세요?</h1>
            <h1>제 이름은 {props.name}입니다. <br />
                children 값은 {props.children} 입니다.
            </h1>
        </div>
    );
};
export default MyComponent3;
```

컴포넌트

4. 함수형 컴포넌트 생성 : src/MyComponent4.js
ES6의 비구조화 할당 문법을 통해 props 내부 값 추출하기

```
import React from 'react';

// function MyComponent4(props){
const MyComponent4=(props)=>{
    const {name, children} = props;      // props로 받은 값을 name과 children변수에 할당
    return (
        <div>
            <h1>안녕 하세요?</h1>
            <h1>제 이름은 {name}입니다. <br />
                children 값은 {children} 입니다.
            </h1>
        </div>
    );
};
export default MyComponent4;
```

컴포넌트

5. 함수형 컴포넌트 생성 : src/MyComponent5.js
함수의 매개변수로 바로 값 전달하기

```
import React from 'react';

// 함수의 매개변수로 바로 값 전달하기
//function MyComponent5({name, children, favoriteNumber})  {
const MyComponent5=({name, children, favoriteNumber}) =>{
    return (
        <div>
            <h1>안녕 하세요?</h1>
            <h1>제 이름은 {name}입니다. <br />
                    children 값은 {children} 입니다.
            </h1>
            <h1>제가 좋아하는 숫자는 {favoriteNumber}입니다.</h1>
        </div>
    );
};
export default MyComponent5;
```

컴포넌트

6. 함수형 컴포넌트 생성 : src/MyComponent6.js
ES6의 비구조화 할당 문법을 사용하여 내부의 값 추출하기

```
import React from 'react';

//function MyComponent6(props){
const MyComponent6=(props)=>{
    const {name, children, favoriteNumber} = props;
    return (
        <div>
            <h1>안녕 하세요?</h1>
            <h1>제 이름은 {name}입니다. <br />
                  children 값은 {children} 입니다.
            </h1>
            <h1>제가 좋아하는 숫자는 {favoriteNumber}입니다.</h1>
        </div>
    );
};
export default MyComponent6;
```

컴포넌트

- **예2. 컴포넌트 나누기**

index.js – Library.js – Book.js

컴포넌트

- **index.js 수정 : src/index.js**

```
import React from 'react';
import ReactDOM from 'react-dom/client';
import './index.css';
import App from './App';
import Library from './Library';

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
  <React.StrictMode>
            {/*  <App /> */}
            <Library />
  </React.StrictMode>
);
```

컴포넌트

- **부모 컴포넌트 생성 : src/Library.js**

```
import React from 'react';
import Book from './Book’;

// 부모 컴포넌트
// function Library() {
const  Library=()=>{
    return (
        <div>
            <Book name="처음 만난 파이썬" numOfPage={300} />
            <Book name="처음 만난 AWS" numOfPage={400} />
            <Book name="처음 만난 리액트" numOfPage={500} />
        </div>
    );
}
export default Library;
```

컴포넌트

- **자식 컴포넌트 생성 : src/Book.js**

```
import React from 'react';

// 자식 컴포넌트
// function Book(props) {
const Book=(props)=>{
    return (
        <div>
            <h1>이 책의 이름은 {props.name}입니다.</h1>
            <h2>이 책은 총 {props.numOfPage}페이지로 이뤄져 있습니다.</h2>
            <br />
        </div>
    );
}
export default Book;
```

컴포넌트

- **프로젝트 생성**

npx create-react-app component02

- **예3. 컴포넌트 나누기**

index.js – App.js – component – Header.js
Main.js
Footer.js

컴포넌트

- **부모 컴포넌트 생성 : src/App.js**

```
import './App.css';
import Header from './component/Header';
import Main from './component/Main';
import Footer from './component/Footer';

function App() {
  return (
    <div className="App">
        <Header/>
        <Main name='홍길동' color='blue'/>
        <Footer/>
    </div>
  );
}
export default App;
```

컴포넌트

- **자식 컴포넌트 생성 : src – component - Header.js**

```
import React from "react";

function Header(props){

    return(
        <div>
            <header>
                <h1>헤드 입니다.</h1>
            </header>
            <hr></hr>
        </div>
    );
}
export default Header;
```

컴포넌트

- **자식 컴포넌트 생성 : src – component - Main.js**

```
import React from "react";
// 자식 컴포넌트에게 2개의 값 전달하기
function Main({name, color}) {
    return(
        <div>
            <main>
                <h1>메인 입니다.</h1>
                <h1 style={{color}}>내이름은 {name}입니다.</h1>
            </main>
            <hr></hr>
        </div>
    );
}
export default Main;
```

컴포넌트

- **자식 컴포넌트 생성 : src – component - Footer.js**

```
import React from "react";

function Footer(props){
    return(
        <div>
            <footer>
                <h1>푸터 입니다.</h1>
            </footer>
        </div>
    );
}
export default Footer;
```

컴포넌트

- **프로젝트 생성**

npx create-react-app component03

- **예4. 컴포넌트 나누기**

index.js – App.js – component – Header.js
Main.js
Footer.js

컴포넌트

❖ 부모 컴포넌트 생성 : src/App.js (1/2)

```
import "./App.css";
import Header from "./components/Header";
import Main from "./components/Main";
import Footer from "./components/Footer";

 스프레드 연산자로 여러개의 값 전달하기
function App() {
  const mainProps = { // Main컴포넌트에 Props로 전달할 값을 mainProps객체 생성
      name : '홍길동',
      location : '서울시',
      favorList : ['파스타','빵','떡볶기'],
   };
```

컴포넌트

❖ 부모 컴포넌트 생성 : src/App.js (2/2)

```
return (
    <div className="App">
        <Header />
        <Main {...mainProps} />     {/* 스프레트 연산자 */}
        <Footer />
    </div>
  );
}

export default App;
```

컴포넌트

- **자식 컴포넌트 생성 : src – component - Header.js**

```
const Header = () => {
  return (
    <header>
      <h1>header</h1>
    </header>
  );
};

export default Header;
```

컴포넌트

- **자식 컴포넌트 생성 : src – component - Main.js**

```
// function Main({name, location, favorList}){
const Main = ({name, location, favorList}) =>{

    return(
        <div>
            {name}은 {location}에 거주 합니다. <br/>
            {favorList.length}개의 음식을 좋아합니다.
        </div>
    );
}

export default Main;
```

컴포넌트

- **자식 컴포넌트 생성 : src – component - Footer.js**

```
const Footer = () => {
  return (
      <footer>
         <h1>footer</h1>
      </footer>
  );
};

export default Footer;
```