props와 state

2023.06.15.

01. props
02. React Developer Tool
03. state

01. props

2023.06.16.

props : 컴포넌트의 속성. 상위 컴포넌트가 하위 컴포넌트에 값을 전달할 때 사용한다. 고정된 서식에 내용을 수정할 수 있는 컴포넌트를 만들 수 있게 함하며, HTML과 비유하면 태그의 속성을 정의하고 사용하는 것으로 볼 수 있다.

1. class 컴포넌트

1.1 정의

구문

```
class 컴포넌트명 extends Component {
  render() {
    return (
      ...
      {this.props.프롬스명}
      ...
    );
  }
}
```

1.2 호출(사용)

구문

```
<컴포넌트명 프롭스명=“값” .../>
```

2. 함수 컴포넌트

2.1 정의

구문

```
function 컴포넌트명({프롭스명, 프롭스명, ...}) {
  return(
    ...
    {프롭스명}
    ...
  );
}
```

2.2 호출(사용)

구문

```
<컴포넌트명 프롭스명=“값” .../>
```

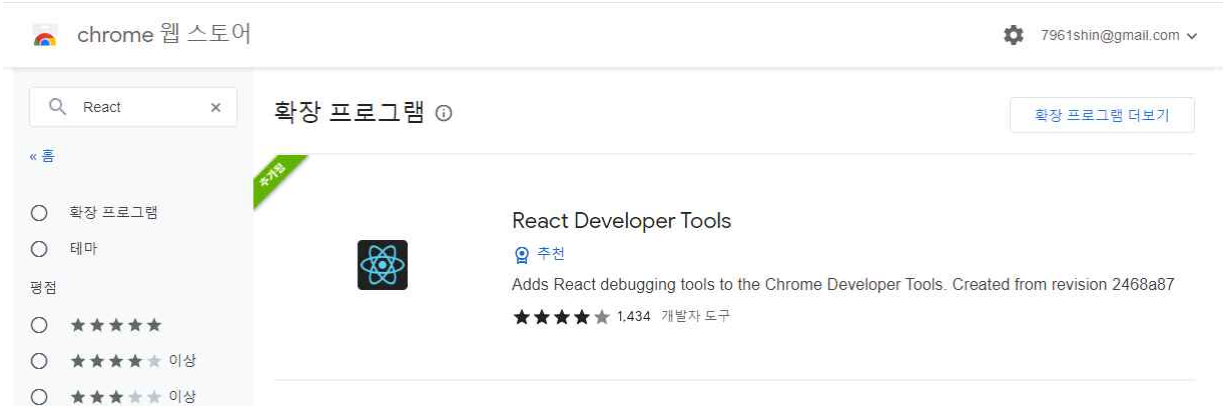
02. React Developer Tool

2023.06.15.

1. 크롬용 React 개발 도구

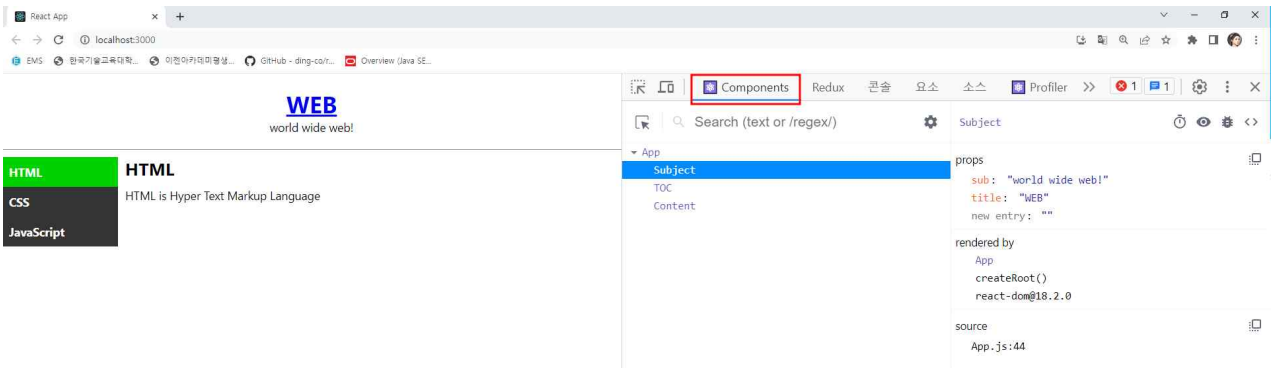
① chrome.google.com/webstore/

② React로 검색 후 React Developer Tools를 설치



③ 브라우저 재실행

2. 크롬 개발자 도구에서 Components 탭



03. state

2023.06.15.

state : 컴포넌트에 대한 데이터 또는 정보를 포함하는 데 쓰이는 리엑트 내장 객체. state는 시간이 지남에 따라 변경될 수 있으며, 변경될 때 마다 컴포넌트의 render()가 다시 호출 된다.
state는 생성자에서 초기화 된다.

1. class 컴포넌트

1.1 정의

구문

```
class 컴포넌트명 extends Component {
  constructor(props) {
    super(props);
    this.state = {
      스테이트명: 값,
      스테이트명: [요소, 요소, ...],
      스테이트명: {속성명: 값, 속성명: 값, ...},
      ...
    }
  }

  render() {
    ...
  }
}
```

1.2 state 값 사용

구문

```
{this.state.스테이트명}
```

1.3 state 값 설정

구문

```
this.setState({스테이트명: 값});
```

2. key props

각 리스트<li>의 항목은 'key'프롭스를 정의하여야 한다. key 프롭스의 값은 중복되지 않아야 한다.

3. 함수 컴포넌트

3.1 Hook(훅)

- 함수 컴포넌트에서 React state와 생명주기 기능(lifecycle features)을 연동(hook into)할 수 있게 해주는 라이브러리 함수

3.1.1 Hook의 사용규칙

- 같은 훅을 여러 번 호출 할 수 있다.
- 함수 컴포넌트 몸통이 아닌, 몸통 안 복합 실행문의 { }에서는 사용할 수 없다.
- 비동기 함수(async 키워드가 붙은 함수)는 콜백함수로 사용할 수 없다.

3.1.1 React에서 지원하는 Hooks

① useState : 컴포넌트의 state를 관리할 수 있다.

② useEffect : 랜더링 이후에 실행할 코드를 만들 수 있다.

③ useContext : 부모 컴포넌트와 자식 컴포넌트 간의 변수와 함수를 전역적으로 정의할 수 있다.

④ useReducer : state 업데이트 로직을 reducer 함수에 따로 분리할 수 있다.

⑤ useRef() : 컴포넌트나 HTML 요소를 참조로 관리할 수 있다.

⑥ forwardRef : useRef로 만든 레퍼런스를 상위 컴포넌트로 전달 할 수 있디

⑦ useImpertiveHandle : useRef로 만든 레프런스의 상태에 따라 실행할 함수를 정의할 수 있다.

⑧ useMemo, useCallback : 의존성 배열에 적힌 값이 변할 때만 값, 함수를 다시 정의할 수 있다.(재랜더링시 정의 안함)

⑨ useLayoutEffect : 모든 DOM 변경 후 브라우저가 화면을 그리기(render)전에 실행되는 기능을 정할 수 있다.

⑩ useDebugValue : 사용자 정의 Hook의 디버깅을 도와 준다.

3.2 useState

- React에서 사용자의 반응에 따라 화면을 바꿔주기(랜더링) 위해 사용하는 트리거 역할을 하는 변수
- React가 state를 감시하고, 변경된 정보에 따른 화면을 보여 준다.

3.2.1 정의 구문

```
import {useState} from “react”;

function 컴포넌트명() {
  const [스테이트명, set스테이트명] = useState(초기값);
}

※ 스테이트명 : getter, set스테이트명 : setter
```

3.2.2 state 불러오기

구문

```
{게터명}
```

3.2.3 state 설정하기

구문

```
set스테이트명 = 값;
```