AI 활용 빅데이터분석
풀스택웹서비스 SW 개발자 양성과정

React

부산대학교 소프트웨어교육센터
PUSAN NATIONAL UNIVERSITY SOFTWARE EDUCATION CENTER

엘리먼트(Element)

- **엘리먼트는 리액트 앱의 가장 작은 단위**
  - **리액트 엘리먼트는 불변객체**
    - 엘리먼트를 생성한 이후에는 해당 엘리먼트의 자식이나 속성을 변경할 수 없음

./src/app.js

```
import './App.css';

function App() {
  return (
    <div>
      현재 시간 : {new Date().toLocaleTimeString()}
    </div>
  );
}

export default App;
```

컴포넌트를 가지고 와서 엘리먼트로 생성

./src/index.js

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

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

reportWebVitals();
```

렌더링 후 생성된 엘리먼트는 변경안됨

```
<!DOCTYPE html>
<html lang="ko">
▶ <head>…</head>
▼ <body>
    <noscript>You need to enable JavaScript to run this app.
    </noscript>
  ▼ <div id="root">
    ▼ <div>
        "현재 시간 : "
        "오후 12:51:08" == $0
      </div>
    </div>
  </body>
</html>
```

엘리먼트 랜더링

- 랜더링 된 엘리먼트 업데이트하기
  - 새로운 엘리먼트를 생성하고 root.render()로 전달

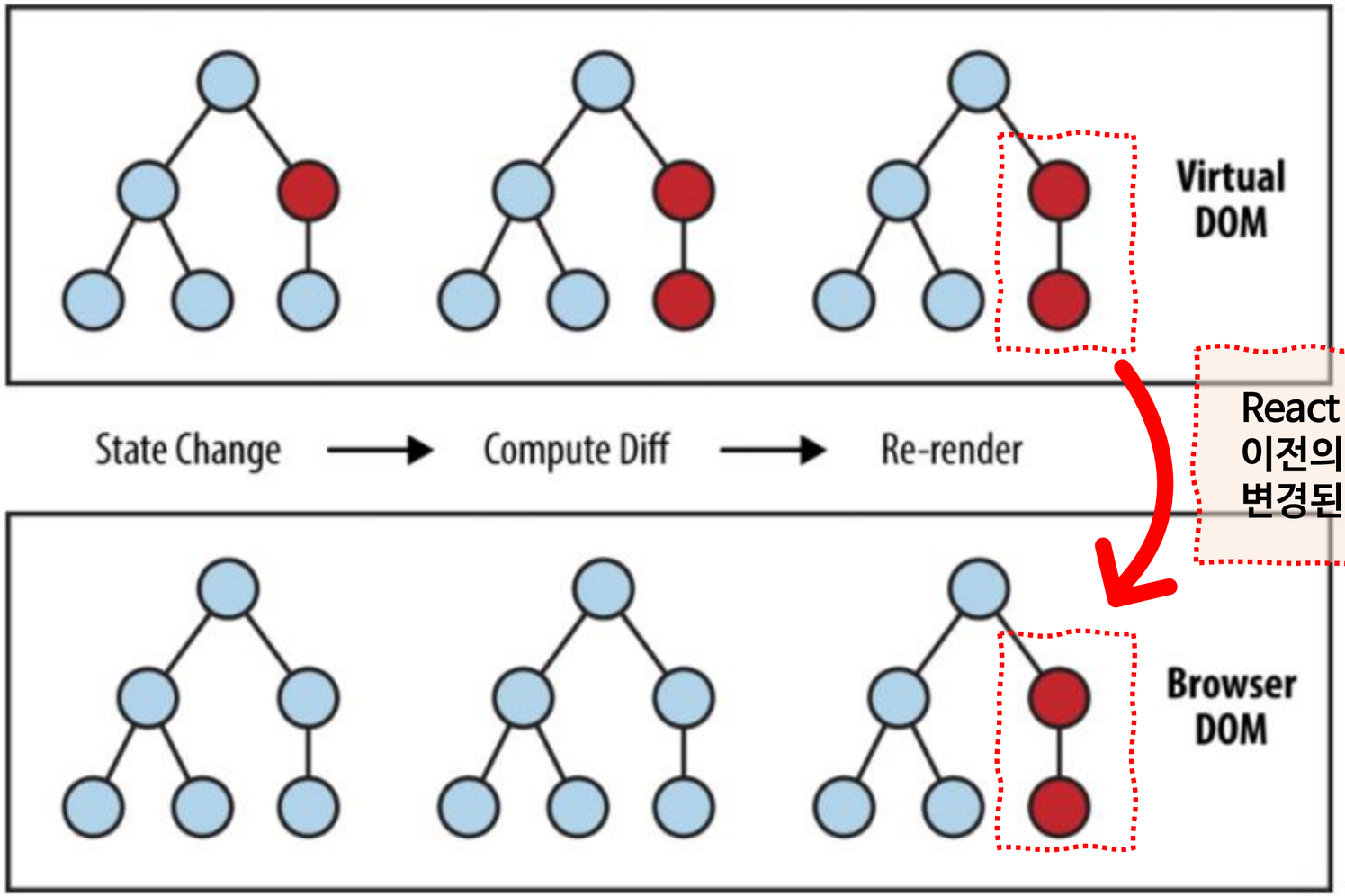
./src/index.js

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

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

//1초에 한번씩 재 랜더링
setInterval(tick, 1000);

reportWebVitals();
```

1초에 한번씩 재 랜더링

```
<!DOCTYPE html>
<html lang="ko">
 ▶ <head>…</head>
 ▼ <body> == $0
    <noscript>You need to enable JavaScript to run this app.
    </noscript>
   ▼ <div id="root">
     ▼ <div>
         "현재 시간 : "
         "오후 1:10:24"
       </div>
     </div>
  </body>
</html>
```

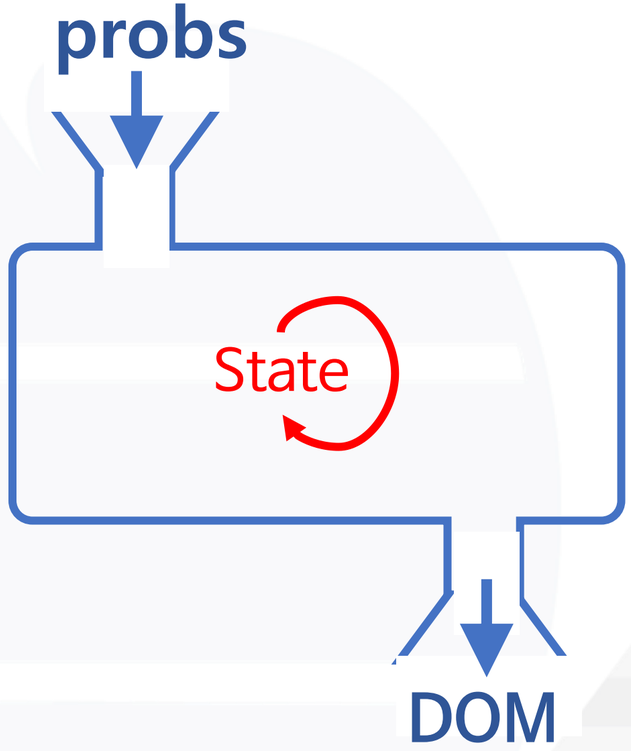
리액트 가상돔(Virtual DOM)



React DOM은 해당 엘리먼트와 그 자식 엘리먼트를
이전의 엘리먼트와 비교하고
변경된 부분만 실제 DOM에 업데이트

컴포넌트(Componet)

- UI를 재사용 가능한 개별적인 조각으로 사용자 정의 태그 생성
  - props라고 하는 임의의 입력을 받은 후, 화면에 어떻게 표시되는지를 기술하는 React 엘리먼트를 반환
  - 컴포넌트는 반드시 하나의 요소를 반환
  - 여러 요소가 있다면 프레그먼트로 감싸서 반환(〈〉…〈/〉)
  - 컴포넌트 명은 반드시 대문자로 시작해야 함
  - JSX 문법으로 작성
    - **JSX**내에 자바스크립트 표현식은 **{}**안에 작성
    - 하이픈은 카멜케이스로 표시해야함
      - background-color => backgroundColor
    - **class** 속성은 **className**으로 사용
    - 태그는 반드시 닫아야 함
    - 주석 : **{/* */}**



함수 컴포넌트와 클래스 컴포넌트

클래스 컴포넌트

```
import React from 'react';
import '../App2.css';

class BoxRow extends React.Component {
  render() {
    return (
      <div className='mvRow'>
      <div className='mvCol1'>2</div>
      <div className='mvCol2'>물방울을 그리는 남자</div>
      <div className='mvCol3'>6,114,200원</div>
      <div className='mvCol4'>626명</div>
      <div className='mvCol5'><i class="fa-sharp fa-solid fa-circle
    </div>
    );
  }
}

export default BoxRow;
```

함수 컴포넌트

```
import React from 'react';
import '../App2.css';

const BoxRow = (probs) => {
  return (
    <div className='mvRow'>
      <div className='mvCol1'>2</div>
      <div className='mvCol2'>물방울을 그리는 남자</div>
      <div className='mvCol3'>6,114,200원</div>
      <div className='mvCol4'>626명</div>
      <div className='mvCol5'><i class="fa-sharp fa-solid fa-
    </div>
  );
}

export default BoxRow;
```

컴포넌트 구조

```
import React from 'react';
import '../App2.css';

const BoxRow = (probs) => {
  return (
    <div className='mvRow'>
      <div className='mvCol1'>{probs.rank}</div>
      <div className='mvCol2'>{probs.movieNm}</div>
      <div className='mvCol3'>{probs.salesAmt}</div>
      <div className='mvCol4'>{probs.audiCnt}</div>
      <div className='mvCol5'>{probs.salesChange}</div>
    </div>
  );
}

export default BoxRow;
```

- React로 작성한 코드를 브라우저가 읽을 수 있도록 변환
- 상위 컴포넌트에 작성되었다면 하위에서는 생략 가능
- React 17버전부터 작성하지 않아도 됨

- 컴포넌트를 호출할 때 전달되는 값 => object

- 이 파일을 다른 곳에서 import 하기 위해 작성
- 한 파일 내에서 한번만 사용

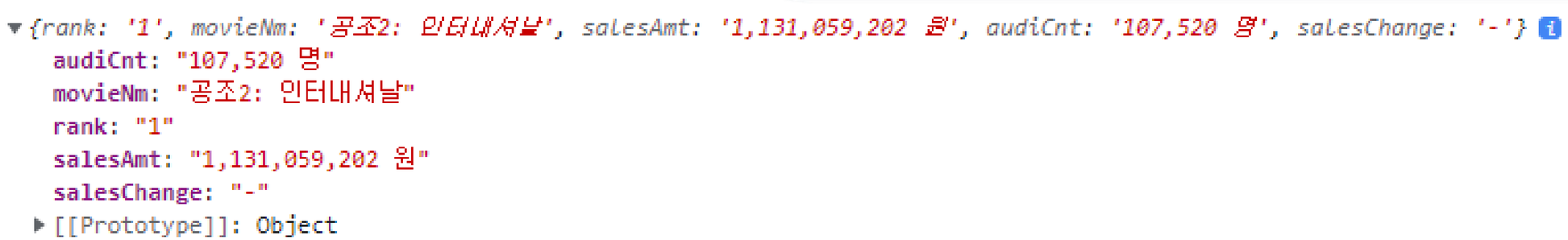
- 여러 개의 함수를 export 할 경우는 내보내고자 하는 함수명 앞에 export를 붙이고 import 하는 곳에서는 import 뒤에 {} 안에 원하는 함수를 넣으면 됨

실습과제

- HTML/CSS 프로젝트1를 리액트로 변경
  - Font Awesome 아이콘 React 사용
    - https://fontawesome.com/v5/docs/web/use-with/react

컴포넌트 probs

- 컴포넌트는 사용자 정의 태그임으로 태그의 속성을 가짐
- 이러한 속성들은 컴포넌트 생성시 함수의 매개변수로 받음
- 이 매개변수는 속성들을 가지는 object
- 읽기 전용으로 수정해서는 안됨



컴포넌트 합성

```
//import React from 'react';
import '../App2.css';
// console 창에 경고 메시지 지우기
console.warn = console.error = () => {};

//컴포넌트 합성
const BoxCol = (probs) => {
  let cn = `mvCol${probs.idx}` ;
  return <div className={cn}>{probs.item}</div>
}

const BoxRow = (probs) => {
  console.log(probs)
  return (
    <div className="mvRow">
      <BoxCol idx="1" item={probs.rank} />
      <BoxCol idx="2" item={probs.movieNm} />
      <BoxCol idx="3" item={probs.salesAmt} />
      <BoxCol idx="4" item={probs.audiCnt} />
      <BoxCol idx="5" item={probs.salesChange} />
    </div>
  );
}

export default BoxRow;
```

- 컴포넌트는 자신의 출력에 다른 컴포넌트를 참조할 수 있음

실습과제

- 컴포넌트 probs를 오브젝트로 전달해서 컴포넌트 생성

```
function App() {
  const items = [
    { rank : "1",  movieNm : "공조2: 인터내셔날",  salesAmt : "1,131,059,202 원", audiCnt : "107,520 명",  salesChange : "-" },
    { rank : "2",  movieNm : "정직한 후보2",  salesAmt : "927,485,030 원", audiCnt : "89,931 명",  salesChange : "-" }
  ];
  return (
    <div className="mvBox">
      <header className="header_title">
        <h1>박스오피스</h1>
      </header>
      <div className='mvList'>
        <div className='mvRow0'>
          <div className='mvCol1'></div>
          <div className='mvCol2'>영화명</div>
          <div className='mvCol3'>매출액</div>
          <div className='mvCol4'>관객수</div>
          <div className='mvCol5'>증감율</div>
        </div>
        <BoxRow mlist={items} />
      </div>
    </div>
  );
}
```

여러 개의 컴포넌트 렌더링

- 여러 개의 엘리먼트를 배열에 저장하고 중괄호를 이용하여 JSX에 포함할 수 있음
  - 자바스크립트 map()함수를 사용

```
const Cols = (item) =전달된 값은 object
  let colTag = [] ;
  for(let [k, v] of Object.entries(item).entries()) 배열의 인덱스와 값추출
    console.log(k, v)
    if (k == 5) break ;  object를 배열로 변환
    colTag.push(<BoxCol idx={k+1} item={v[1]} />)   엘리먼트 배열 만들기
  }
  return colTag ;  엘리먼트 배열 반환
}

const BoxRow = (probs) => {
  const items = [...probs.mlist];
  const divRows = items.map((item) =>
                    <div className="mvRow" onClick={() => setmovieCd(item.movieCd)}> {Cols(item)} </div>
                  ) ;
  let [movieCd, setmovieCd] = useState(null);

  return (
    <>
    { divRows }   엘리먼트 배열
    <BoxContent movieCd={movieCd} />
    </>
  );
}
```

함수 호출

배열의 항목을 반복 실행하면서 엘리먼트를 반환하고 생성된 엘리먼트 리스트를 출력

컴포넌트 이벤트 추가

- React 엘리먼트에서 이벤트를 처리
  - React의 이벤트는 소문자 대신 캐멀 케이스(camelCase)를 사용
  - JSX를 사용하여 문자열이 아닌 함수로 이벤트 핸들러를 전달

```
<div className="mvRow" onClick={() => show(item)}>
```

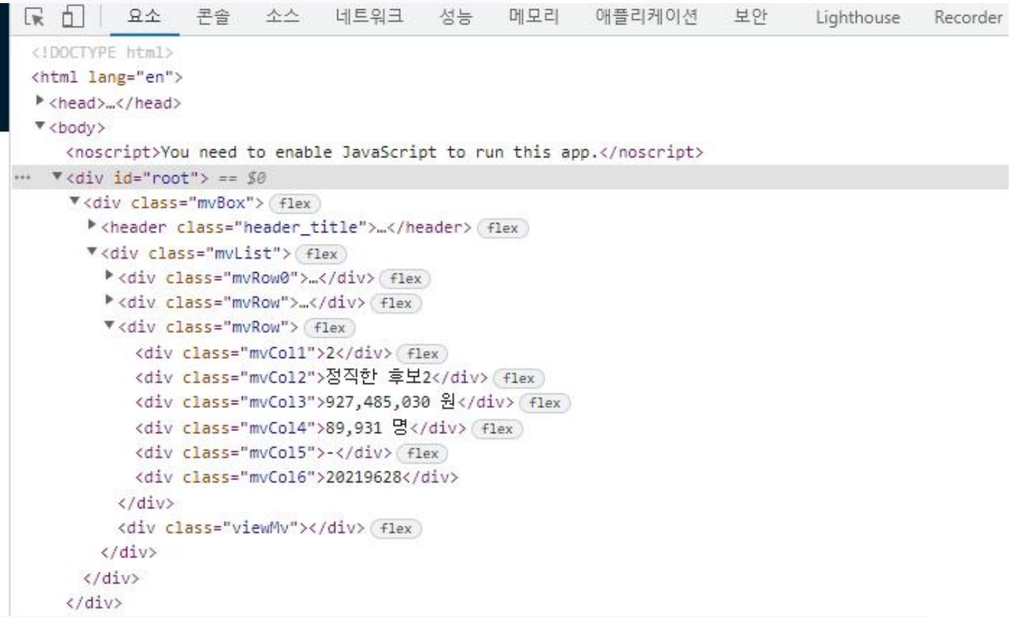
```
const show = (clickItem) => {
  console.log(setmovieCd(clickItem.movieCd))
}
```

컴포넌트 내부에 작성

생각해 보기

박스오피스

	영화명	매출액	관객수		증감율
1	공조2: 인터내셔날	1,131,059,202 원	107,520 명	-	20215601
2	정직한 후보2	927,485,030 원	89,931 명	-	20219628



컴포넌트 state

- state는 함수 내에 선언된 변수처럼 컴포넌트 안에서 관리되면서 변경시 랜더링 결과에 영향을 줌
  - 컴포넌트의 내부에서 변경 가능한 데이터를 관리
  - 리액트 16.8부터 Hooks 라는 기능이 도입되면서 함수형 컴포넌트에서도 상태를 관리가능
  - 함수 컴포넌트 안에서 useState Hook을 사용하여 state 관리

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

=> 반드시 상단에 추가

```
const BoxRow = (probs) => {
  const items = [...probs.mlist];
  let [movieCd, setmovieCd] = useState(null) ;

  return (
    <>
    { items.map((item) => <div className="mvRow" onClick={() => setmovieCd(item.movieCd)}> {Cols(item)} </div>) }
    <BoxContent movieCd={movieCd} />
    </>
  );
}
```

State 변수 선언은 useState()를 호출

- useState의 인수 : State 변수의 초기값
- useState의 반환값 : State변수, 해당 변수를 갱신하는 함수
- 반환값은 구조분해 할당(Destructuring assignment)을 통해 저장

컴포넌트 내에서 사용

변경이 되면 DOM에 바로 반영