[React 공부 3일차]

1. React State 값 변경하기

```
let [like, upLike] = useState(0);

for (let i = 0; i < 3; i++) {
  listItem.push(
    <div className="list">
    <h4>{ titleName[i] } <span>❤️</span> { like } </h4>
    <p>2월 17일 발행</p>
  </div>
  )
}

return (
```

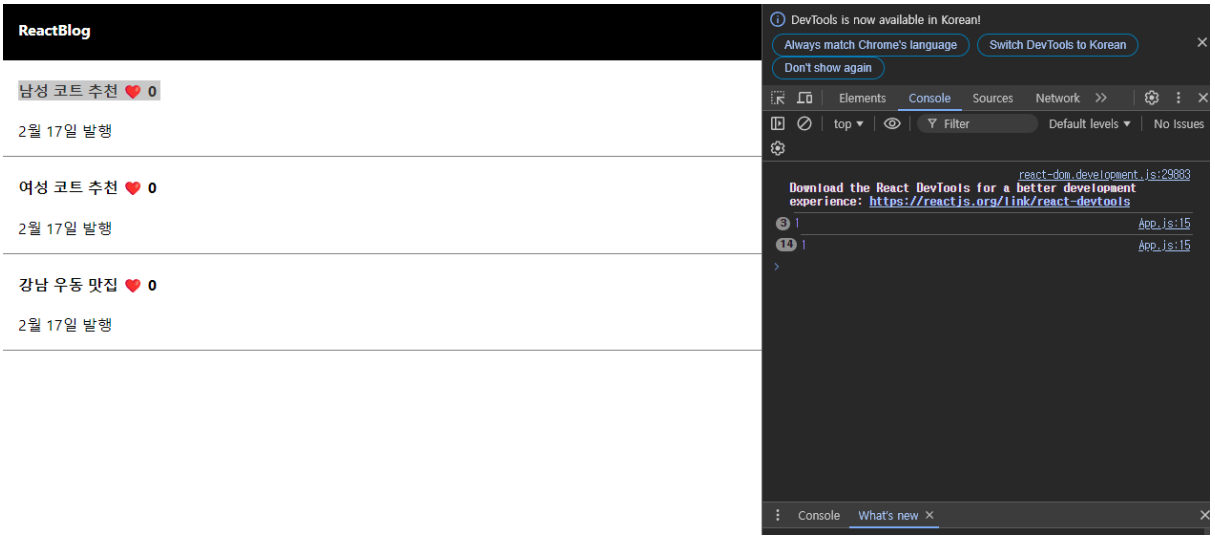
저번에 만들어두었던 useState 문을 통하여 좋아요 버튼을 구현하고있는중입니다.

```
function upLikeBtn(){
  console.log(1);
}

for (let i = 0; i < 3; i++) {
  listItem.push(
    <div className="list">
    <h4>{ titleName[i] } <span onClick={ upLikeBtn }>❤️</span>
    <p>2월 17일 발행</p>
  </div>
  )
}

return (
```

이번에 사용해볼문법은 onClick 문인데 위에 사진과같이 span 자체에 onclick 을 넣어준 다음 function 기능을 새로 따로 생성해 준 후 onclick 에 데이터바인딩을 해줍니다.

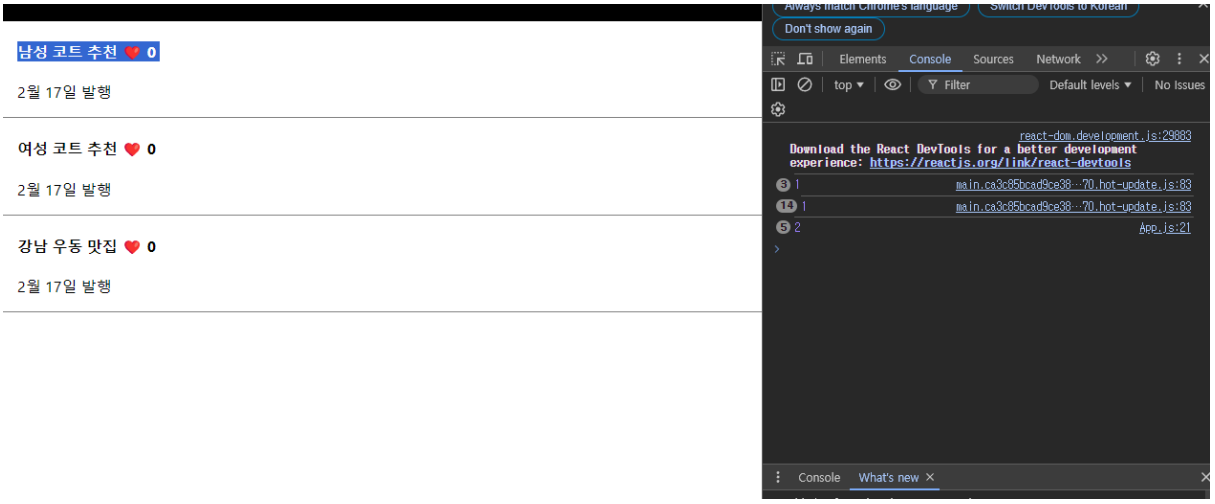


이런식으로 정상적으로 작동하는 모습이 확인이 됩니다.

```
for (let i = 0; i < 3; i++) {
  listItem.push(
    <div className="list">
    <h4>{ titleName[i] } <span onClick={ () => { } }>❤️</span>
    <p>2월 17일 발행</p>
  </div>
  )
}
```

```
<h4>{ titleName[i] } <span onClick={()=>{console.log(2)}}>❤️</sp
```

function을 따로 나눠사용해도 좋지만 onClick 자체에 기능을 구현하여서 바로바로 적용이 가능하게도 할 수 있는데 이때에 사용하는것은 (괄호) ⇒ {중괄호} 문법을 사용하여 function에 기능을 대신해줍니다.



똑같이 잘 작동하는 모습이 보입니다.

```
istItem.push(
 <div className="list">
 <h4>{ titleName[i] } <span onClick={()=>{ like = like + 1 }}>❤️
 <p>2월 17일 발행</p>
'div>
```

이렇게 like = like + 1 로 작성하거나 like + 1 을 작성해서 기능을 만들어줍니다.

하트 모양을 눌렀을때 1 이 올라가는모습이 정상적으로 보입니다.

```
or (let i = 0; i < 3; i++) {
 listItem.push(
   <div className="list">
   <h4>{ titleName[i] } <span onClick={()=>{ upLike(like + 1) }}>
   <p>2월 17일 발행</p>
 </div>
 )
```

이제 useState 문을 이용하여 재렌더링이 즉각즉각 보이도록 하기위해 뒤에 변경하는 useState변수값을 입력해주고 이때에는 변수뒤에 (괄호)를 넣어주고 이안에 기능을 넣어줍니다.

하고나면 이런식으로 즉각즉각 반응하는것이 보입니다.

2. Onclick 응용하기

```
let post = '역삼 우동 맛집';
const listItem = [];
let [titleName, setTitle] = useState(
  [{'id':1, 'value':'남성 코트 추천'}
  ,{'id':2, 'value':'여성 코트 추천'}
  ,{'id':3, 'value':'강남 우동 맛집'}]
);
let [logo, setLogo] = useState('ReactBlog');
let [like, upLike] = useState(0);
```

이제 숫자 올라가는것은 확인하였고, 변경하기 버튼을 누를시 이미 적혀있던 titleName이 변경되는 버튼을 만들어봅니다.

그러기 위해선 기존에 있던 useState 문법을 조금 변경을해줘서 각각 배열에 id값을 부여합니다.

```
const changeValue = () => {
  setTitle((prevTitles) => {
    return prevTitles.map((item) =>
      item.id === 1
        ? { ...item, value: '여성 코트 추천' }
        : item
    );
  });
};
```

버튼에 넣어줄 function 기능을 만들어줍니다.

map 을 사용하여 id 값을 가져와 비교한뒤 id 값이 1인 경우 item 안에 들어있는 value값을 변경해주는 기능입니다.

```
for (let i = 1; i <= 3; i++) {
  let item = titleName.find(el => el.id === i);
  if (item) {
    listItem.push(
      <div className="list" key={item.id}>
      <h4>{item.value} <span onClick={()=>{ upLike(like+1) }}>❤️</span
      <p>2월 17일 발행</p>
    </div>
    )
  }
}
```

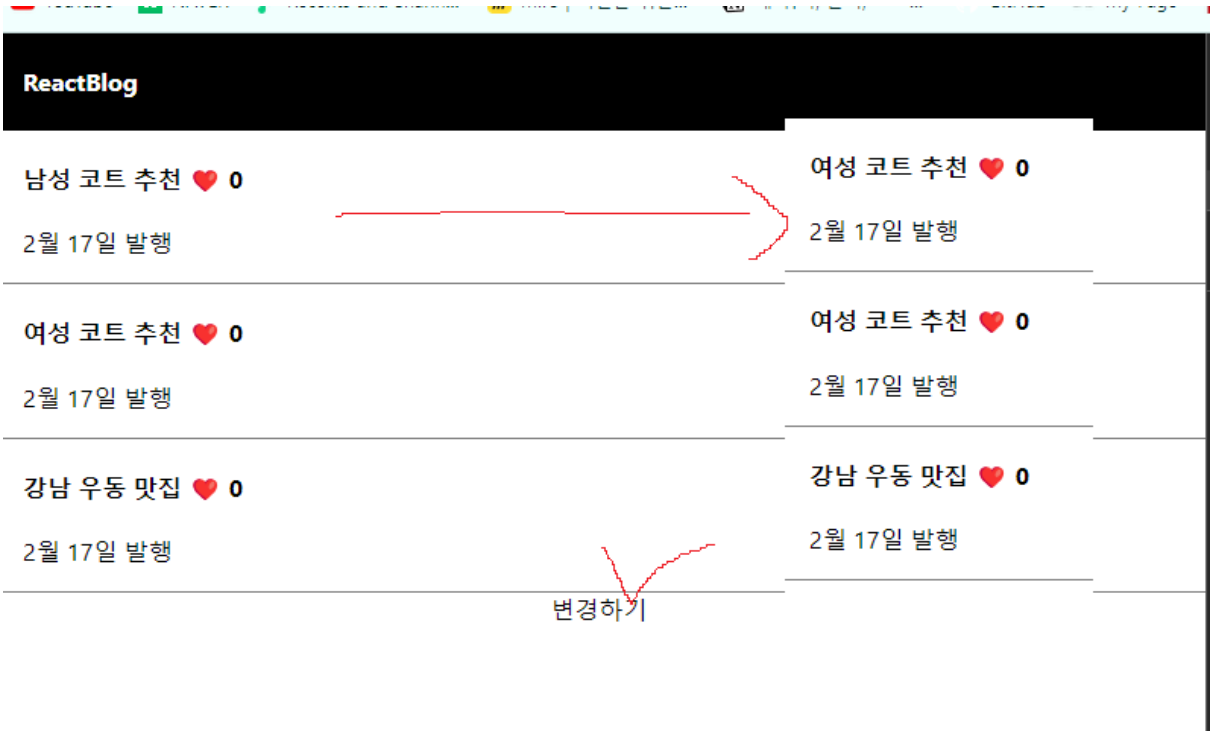
for 문을 돌려서 만들어주던 게시글 목록도 수정을 거쳐갑니다.

기존에는 출력만하면 상관없어서 키값에 상관없이 배열안에 값을 가져와 [0],[1],[2] 식으로 생성해주었지만 id값이 주어져서 item.value 값만 빼오고있습니다.

```
return (
  <div className="App">
    <div className="black-nav">
      <h4>{ logo }</h4>
    </div>
    {listItem}
    <span onClick={changeValue}>변경하기</span>
  </div>
);
}
```

기존에 있던 {listItem[0]}, {listItem[1]}, {listItem[2]} 를 삭제하고 하나로 적어도 정상작동합니다.

그리고 span Onclick 기능에 미리 만들어두었던 changeValue 기능을 바인딩 시켜줍니다.



변경하기 글을 누르면 남성 코트 추천 글이 여성 코트 추천글로 변경되는것이 잘 확인됩니다.

다시 눌렀을때 남성 코트 추천으로 돌아오게 하게끔 function 기능을 수정해주면

```
const changeValue = () => {
  setTitle((prevTitles) => {
    return prevTitles.map((item) =>
      item.id === 1
        ? { ...item, value: item.value === '남성 코트 추천' ? '여성 코트 추천' : '남성 코트 추천' }
        : item
    );
  });
};
```

이런식으로 수정해주면 변경하기 버튼을 눌렀을때 item.id = 1 에 값이 남성 코트 추천 이면 여성 코트 추천으로 여성 코트 추천이면 남성 코트 추천으로 잘 변경되는것이 확인됩니다.

```
return (
  <div className="App">
    <div className="black-nav">
      <h4>{ logo }</h4>
    </div>
    {listItem}
    <span onClick={changeValue}>변경하기(function사용함)</span>
    <br></br>
    <span onClick={() => {
      let copy = [...titleName];
      copy[0] = {'id':1, 'value':'여성 코트 추천'}
      setTitle(copy);
    }}>
      변경하기(copy문)
    </span>
  </div>
);
}
```

function을 미리 만들어서 onClick 에 넣어주어도 좋고 아니면 () ⇒ {} 문을 사용하여 copy문을 써서 특정 배열만 변경시켜주는 기능을 만들어도 좋습니다.

3. sort(); 사용해보기

들어가기앞서 앞에 작성했던 코드를 수정을 좀 해줍니다.

```
function App() {

  let post = '역삼 우동 맛집';
  const listItem = [];
  let [titleName, setTitle] = useState(
    [{'id':1, 'value':'남성 코트 추천'}
    ,{'id':2, 'value':'냐니뇨냐뇨'}
    ,{'id':3, 'value':'강남 우동 맛집'}]
  );
  let [logo, setLogo] = useState('ReactBlog');
  let [like, upLike] = useState(0);

  const changeValue = () => {
    setTitle((prevTitles) => {
      return prevTitles.map((item) =>
        item.id === 1
          ? { ...item, value: item.value === '남성 코트 추천' ? '여성 코트 추천' : '남성 코트 추천' }
          : item
      );
    });
  };
```

```
    <div className="App">
      <div className="black-nav">
        <h4>{ logo }</h4>
      </div>

      {titleName.map((item =>
        <div className='list' key={item.id}>
          <h4>
            {item.value}
            <span onClick={()=>upLike(like+1)}>❤️</span> {like}
          </h4>
          <p>8월 19일 발행</p>
        </div>
      ))}

      <span onClick={changeValue}>변경하기(function사용함)</span>
      <br></br>
      <span onClick={() => {

        let copy = [...titleName];

        copy[0] = {'id':1, 'value':'여성 코트 추천'}
        setTitle(copy);
      }}>
        변경하기(copy문)
      </span>
      <br></br>
      <button onClick={()=>{

        let sortCopy = [...titleName];

        sortCopy.sort((a, b) => a.value.localeCompare(b.value));

        console.log(sortCopy);

        setTitle(sortCopy);

      }}>가나다 정렬</button>
    </div>
  );
}
```

for문을 돌려 listItem 배열에 추가해서 출력하면 방식을 삭제해주고 map 을 사용하여 react 홈페이지에 출력되게 변경한뒤 copy문법을 사용하여 sort 시켜 가나다순으로 글을 정렬해줍니다.

4. Component 만들어보기

js를 사용하다보면 페이지를 조금만 만들어도 div가 무수하게 많아지는것을 확인할 수 있습니다.

그래서 보기도힘들고 어디가 어딘지 찾기 힘들어서 component를 쓴다고합니다.

다른말로는 modal 이라하는데 function 으로 미리 div를 생성해서 만들어두고 그 이름을 가져다가 어디든 붙여서 쓸 수 잇는 기능이라고합니다.

```
    <div className='modal'>
      <h4>제목</h4>
      <p>날짜</p>
      <p>상세내용</p>
    </div>
  </div>
);
```

```
.modal {
  margin-top: 20px;
  padding: 20px;
  background: #eee;
  text-align: left;
}
```

modal 용 div를 생성했지만 이미 같은곳에서 div가 너무많은 상황이라 이걸 app fuction 에서 빼내준다음 간략하게 <modal/> 만을 사용하여 불러와보도록 하겠습니다.

```
66
67          <Modal></Modal>
68
69        </div>
70      );
71    }
72
73    function Modal(){
74      return (
75        <div className='modal'>
76        <h4>제목</h4>
77        <p>날짜</p>
78        <p>상세내용</p>
79      </div>
80      )
81    }
82
83    export default App;
```

이런식으로 modal을 생성해주고 위에서는 간편하게 <modal></modal> 만으로 div를 불러 올 수 있게되었습니다.

하지만 component의 단점은 여러개의 div를 담을 수 없다는것인데 그걸 해결하기위해선 큰 틀을 잡아주는 div 를 만들어주고 그안에서는 여러 div를 쓸 수 있게됩니다.

그래도 나는 div 란 단어자체가 너무많아서 보기 싫다 하시는분들은 <></> 을 사용하여 깔끔하게 정리가 가능합니다.

```
      <Modal/>

    </div>
  );
}

function Modal(){
  return (
    <>
      <div className='modal'
        <h4>제목</h4>
        <p>날짜</p>
        <p>상세내용</p>
      </div>
    </>
  )
}
```

<></> 사용하고 위에선 <modal/> 로 닫아준 모습입니다.

이렇게 정상적으로 출력된 모습입니다.

컴포넌트를 사용하기 좋을때에는 간략하게

1. 반복적인 div를 사용할때
2. 큰 페이지들을 설계할때

3. 자주 변경되는 html ui 들을 설게할때에 좋습니다.

```
let [modal, setModal] = useState(false);
```

이제 안보이다가 누르면 modal 창이 보이는 버튼을 만들어보겠습니다.

modal 창의 상태를 보기위해 useState를 통해 true 값 false 값을 지정해줍니다.

기본은 보이지 말아야하니 false 값을 입력해 modal창을 숨겨놓습니다.

```
{
  1 == 2 ? '맞음': '아님'
}
<Modal/>
```

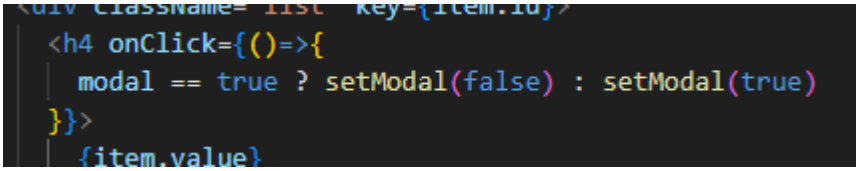
이제 if 문을 사용하여 fasle 값과 true 값을 비교하여 띄워주어야하는데 html 문에서는 if 문 사용이 불가능하기에 삼항연산자를 사용해줍니다. ?? == ? 트루값 : else 값 을 입력해주면 사용가능한 삼항연산자입니다.

```
<div className="list" key={item.id}>
  <h4 onClick={()=>{setModal(true)}}>
    {item.value}
```

이제 글에있는 제목을 누르면 useState 값 변경으로 Modal값을 true로 변경해줍니다.

클릭시 아래에 정상적으로 div가 출력되는것을 확인했습니다.

이것을 이제 다시누르면 사라지게끔 설계해봅시다.



이런식으로 삼항연산자를 넣어주면 modal 값이 true 인지 확인해보고 아니면 else 로 true 로바꿔주고 조건이 맞다면 false 값으로 바꿔줍니다.

이렇게 하면 다시누르면 숨겨지는 기능이 완성됩니다.

감사합니다.

연락처

✉️ alarm153@Naver.com 📱 +82 10-4750-0732 🏠 서울시 관악구 신림동