Antd 적용 예제

- Header 컴포넌트
  Affix와 Button 컴포넌트 이용 → 페이지 상단 고정

```
<Affix offsetTop={0} onChange={affixed => console.log(affixed)}>
      <Button className='headerLogo'/>
</Affix>
<Affix offsetTop={0} onChange={affixed => console.log(affixed)}>
    <Button className='headerProfile'/>
</Affix>
```

- Notice 컴포넌트
  Typhography, Button, Badge, Affix 컴포넌트 이용
  → Typhography 컴포넌트를 이용하여 text 작성 (ellipsis props 이용하여 말줄임 적용)
  → Affix 컴포넌트를 이용하여 확성기 아이콘 상단 고정
  → 확성기 알림 Badge

```
<div className="NoticeBar"
        style={{
            width: '100%',
            height:'42px',
            backgroundColor: '#2C3340',
      }}
>
  <Affix offsetTop={0} onChange={affixed => console.log(affixed)}>
    <div className="notice-bar-icon"/>
  </Affix>
    <Text
        ellipsis
        style={{
            margin:'10px 50px 0 10px',
            color:'#fff',
            fontSize:'12px',
        }}
    >
        '꼴지 탈출+리그 첫 승'맨유, 리버풀 2-1 격파 '래시포드 ,'꼴지 탈출+리그 첫 승'맨유, 리버풀 2-1 격파 '래시포드
    </Text>
</div>
```

- Club Notice 컴포넌트
  Button, Badge 컴포넌트 이용 → 이벤트 알림 Badge

- Tab 컴포넌트
  ant design의 Tab import 실패 → 기존 작업물의 스크립트 활용 (JQuery)
  < typescript에서 jquery 사용법 >
  1. `npm install @types/jquery` → Jquery 설치
  2. .tsx에서 .js와 .d.ts를 import
  적용 실패로 인해 Jquery로 작성되어있던 스크립트 javascript로 재작성 (아래 코드 참조)

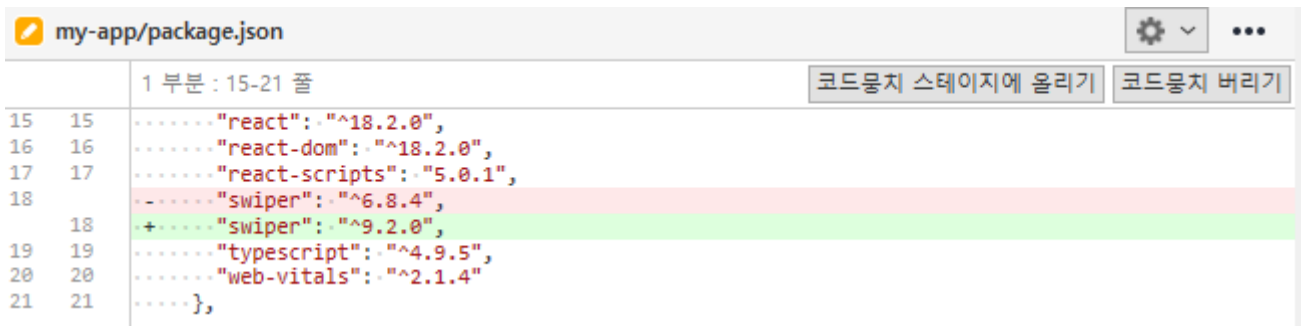
```
const mainTabList = document.querySelectorAll(".main-tab > li");

mainTabList.forEach((li) => {
    li.addEventListener('click', (event) => {
        event.preventDefault();
        mainTabList.forEach((li) => {
            li.classList.remove('main-tab-on');
        });
        li.classList.add('main-tab-on');
    });
});
```

위의 코드의 경우 스크립트가 로드되는 시점을 컴포넌트가 마운트되는 시점이 일치되지않아 스크립트가 렌더링 되는 직후 적용되지 않는 문제점 발생
→ useEffect를 사용하여 스크립트가 로드되는 시점을 컴포넌트가 마운트되는 시점과 동일하게 맞춰주는 작업

```
useEffect(() => {
    const mainTabList = document.querySelectorAll(".main-tab > li");
    mainTabList.forEach((li) => {
      const handleClick = (event) => {
        event.preventDefault();
        mainTabList.forEach((li) => {
          li.classList.remove("main-tab-on");
        });
        li.classList.add("main-tab-on");
      };
      li.addEventListener("click", handleClick);
      return () => {
        li.removeEventListener("click", handleClick);
      };
    });
}, []);
```

문제점	해결방법	새롭게 발생한 문제점
antd Tab import 실패	기존 스크립트 활용 (Jquery)	.d.ts 파일 오류로 인해 적용 실패
Jquery 적용 실패	javascript 작성법으로 재작성	스크립트 로드 시점과 컴포넌트 마운트시점의 불일치로 인한 렌더링 적용 오류
시점불일치	useEffect를 통한 시점 일치작업	event객체의 타입이 any로 명시되어있지 않은 문제점 발생

- MainBanner 컴포넌트
  Swiper 라이브러리 활용
  `Cannot read properties of undefined (reading 'wrapperClass')`
  위와 같은 오류 발생 반복 → 발생원인 : swiper의 버전 이슈 → 해결 : 버전 업그레이드



- DividendBoard 컴포넌트
  Button, Typography, Radio 컴포넌트 이용
  → Button의 link props를 이용하여 토너먼트 버튼 생성
  → Typography 컴포넌트를 이용하여 text 작성 (ellipsis props 이용하여 말줄임 적용)
  → Radio.Group와 Radio.Button 컴포넌트를 이용하여 배당판 생성

```
// Button 컴포넌트 예시
<Button
    type="link"
    style={{
        fontSize:'13px',
        color:'#eee',
        fontWeight:'500',
    }}
>
    <span className='nation'>영국</span>
    <span className='league'>프리미어리그</span>
</Button>

// Typography 컴포넌트 예시
const { Text } = Typography;

<Text ellipsis className='league-table-group'>
    <Text className='league-table-l'>토르페도 N.노브고로드</Text>
    <span>vs</span>
    <Text className='league-table-r'>SKA 상트페테르부르크</Text>
</Text>

// Radio 컴포넌트 예시
<Radio.Group  onChange={onChange} buttonStyle="solid" className='d-b-btn-group'>
      <Radio.Button value="1" className='d-b-btn' disabled={disabled[0]} >
          <span>홈 승</span><span className='d-b-num'>1.93</span>
      </Radio.Button>
      <Radio.Button value="2" className='d-b-btn' disabled={disabled[1]} >
          <span>무</span><span className='d-b-num'>1.93</span>
      </Radio.Button>
      <Radio.Button value="3" className='d-b-btn' disabled={disabled[2]} >
```

```
          <span className='d-b-num'>1.93</span><span>원정 승</span>
      </Radio.Button>
      <Radio.Button value="4" className='d-b-btn' disabled={disabled[3]} >
          <span>홈 승</span><span className='d-b-num'>1.93</span>
      </Radio.Button>
      <Radio.Button value="5" className='d-b-btn accodian'>
          <span className='aco-position'>H</span><span className='aco-num'>0.5</span>
      </Radio.Button>
      <Radio.Button value="6" className='d-b-btn' disabled={disabled[4]} >
          <span className='d-b-num'>1.93</span><span>원정 승</span>
      </Radio.Button>
      <Radio.Button value="7" className='d-b-btn' disabled={disabled[5]} >
          <span>언더</span><span className='d-b-num d-b-under'>1.18</span>
      </Radio.Button>
      <Radio.Button value="8" className='d-b-btn accodian'>
          <span className='aco-position'>U/O</span><span className='aco-num'>2.5</span>
      </Radio.Button>
      <Radio.Button value="9" className='d-b-btn' disabled={disabled[6]} >
          <span className='d-b-num d-b-over'>1.93</span><span>오버</span>
      </Radio.Button>
</Radio.Group>
```

useState를 활용하여 Radio의 disabled props 스크립트 작성

```
const [disabled, setDisabled] = useState([false, false, false, false, false, false, false]);

const onChange = (e: RadioChangeEvent) => {
  if (e.target.value === 1) {
    setDisabled([false, true, true, true, true, true, true]);
  } else if (e.target.value === 2){
    setDisabled([false, false, false, true, true, false, false]);
  } else if(e.target.value === 3){
    setDisabled([false, false, false, true, true, false, true]);
  }else if(e.target.value === 4){
    setDisabled([true, true, true, false, false, false, false]);
  }else if(e.target.value === 6){
    setDisabled([true, true, true, false, false, false, false]);
  }else if(e.target.value === 7){
    setDisabled([false, false, false, false, false, false, false]);
  }else if(e.target.value === 9){
    setDisabled([false, false, false, false, false, false, false]);
  }
};

// 기존 e가 any타입으로 지정되어있어 e를 명시적으로 지정해주면서 추가적으로 RadioChangeEvent를 import 하였습니다 → 따라서 RadioChangeEvent import
```

- BottomBar 컴포넌트
  Affix 컴포넌트 이용 → 페이지 하단 고정

```
<Affix offsetBottom={0} onChange={affixed => console.log(affixed)}>
  <div className='main-btn'>홈</div>
</Affix>
```

- FixedElement 컴포넌트
  BackTop 컴포넌트 이용

```
<BackTop>
    <div className="ant-back-top-inner"/>
</BackTop>
```