Ajax

```
AJAX(AsynchronousJavaScriptAndXML)는 비동기 자바스크립트
비동기적 자바스크립트 동작을 하는 기술들을 통들어서도 AJAX라고 함

서버로부터의 응답(response) 확인
Ajax에서 서버로부터의 응답을 확인하기 위해 사용하는 XMLHttpRequest 객체
줄여서 XHR 라고함

1.  XMLHttpRequest
2.  Fetch
3.  Axios

1.  XMLHttpRequest
형식)
  new XMLHttpRequest()  : Ajax 기능을 수행하는 통신 객체
  method :  백엔드 페이지에 접속하는 방식 (GET,POST,PUT,DELETE)
  onreadystatechange  : xhr객체의 상태가 변화할 때 마다 호출되는 이벤트
  xhr.open(method, url)
  xhr.send()  : 요청

>> XMLHttpRequest의 readyState
0: UNSET - 요청이 생성된 상태로 아직 서버에 요청을 보내지 았았음
1: OPENED - open() 함수가 호출된 상태로, 요청이 초기화
2: HEADERS_RECEIVED - send() 함수 호출 후, 서버에 요청한 결과의 Header를 수신한 상태
3: LOADING - send() 함수 호출 후, 서버에 요청한 결과를 받아오는 중
4: DONE - 서버에 요청한 결과를 받은 상태이며, 결과는 성공이거나 실패했음을 알 수 있음
 XMLHttpRequest.DONE 완료 이것만 있어도됨

const xhr = new XMLHttpRequest()
 xhr.onreadystatechange = e => {
    if (e.target.readyState == XMLHttpRequest.DONE) {
        Ajax의 처리 결과를 구현하는 위치
    }
 ))
 요청을 초기화
 xhr.open(method, url);
 xhr객체가 요청을 백엔드에 전송
 xhr.send();

>> status 프로퍼티
status 프로퍼티는 서버의 문서 상태를 나타냅니다.
 - 200 : 서버에 문서가 존재함.
 - 404 : 서버에 문서가 존재하지 않음.
 - 403 : 접근거부
 - 500 : 서버의 응답이 없음

 특징
 초기 기획때부터 인터페이스를 제공하지 않아 브라우저간의 불일치가 발생
 IE 지원.
```

```
 2. Fetch
 Promise 기반 => 현재 비동기 처리 프로그래밍 방식과 잘 어울림
이미 잘 명세된 API가 제공됨으로써 브라우저간의 불일치가 없음
IE 대부분에서 동작하지 않음

  fetch(url)
    .then((response)=>(response.json()))
    .then((data)=> console.log(data))


 3. Axios
 promise 기반으로 다루기가 쉽다
 크로스 브라우징에 신경을 많이썼기에 브라우저 호환성
 axios.get(url)
 .then(response => console.log(response.data) )
```