CORS

박재훈

https://youtu.be/JQjVkP6ucUI

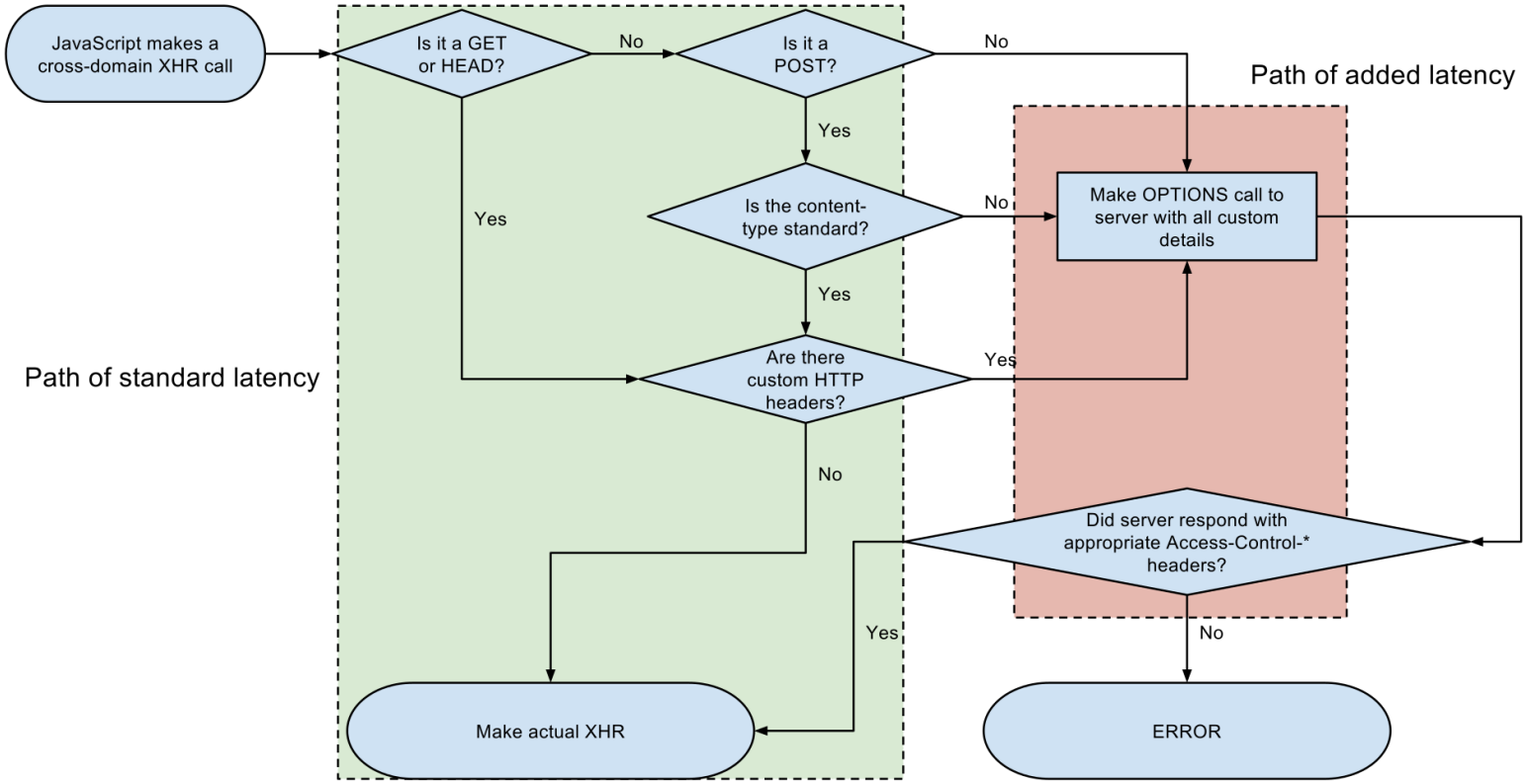
CORS

- 교차 출처 리소스 공유 (Cross-Origin Resource Sharing)

- 도메인 또는 포트가 다른 서버의 자원을 요청하는 매커니즘

CORS 요청의 종류

- Simple / Preflight
- Credential / Non-Credential



Simple Request

- GET, HEAD, POST 중 한 가지 방식 사용

- POST일 경우 Content-type이 아래 셋 중 하나여야 함
  - application/x-www-form-urlencoded
  - multipart/form-data
  - text/plain

- 커스텀 헤더를 전송하지 말아야 함

Preflight Request

- Simple Request에 해당하지 않는 경우
- 예비 요청(Preflight Request)와 본 요청(Actual Request)로 나뉨
- 각각의 응답을 서버가 알아서 처리

Request with/without Credential

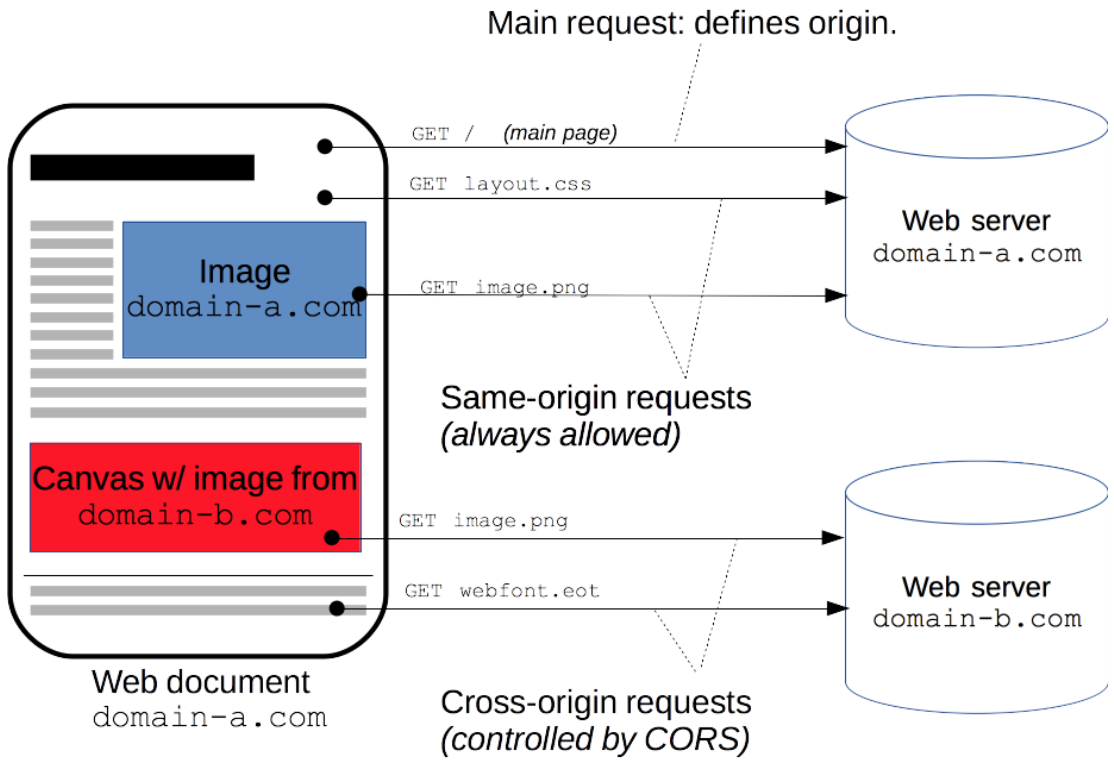
- without Credential : default

- with Credential : Cookie 혹은 Authentication 정보가 필요할 경우

동일 출처 정책

- 특별한 허가가 없는 이상 동일 출처 정책(Same-Origin Policy)에 의해 보안 목적으로 차단
- 특정 출처(Origin)에서 불러온 문서나 스크립트가 다른 출처에서 가져온 리소스와 상호작용하는 것을 제한하는 중요한 보안 방식
- 잠재적으로 해로울 수 있는 문서를 분리함으로써 가능한 공격 경로를 줄이는데 도움을 줌

동일 출처 정책



- https://domain-a.com에서 AJAX 방식 등을 통해서 https://domain-b.com로 자원을 요청할 경우,
- domain-a.com 프론트엔드에서 domain-a.com 웹 서버로의 요청은 같은 출처이기에 허용
- domain-b.com 웹 서버로의 요청은 출처가 다르기에 허용되지 않음

문제점

- RESTful API
  - 백엔드를 따로 두고서 여러 기기에서 서버의 자원을 이용

- 프론트엔드 → 백엔드로의 자원 요청이 거부됨

해결 방안

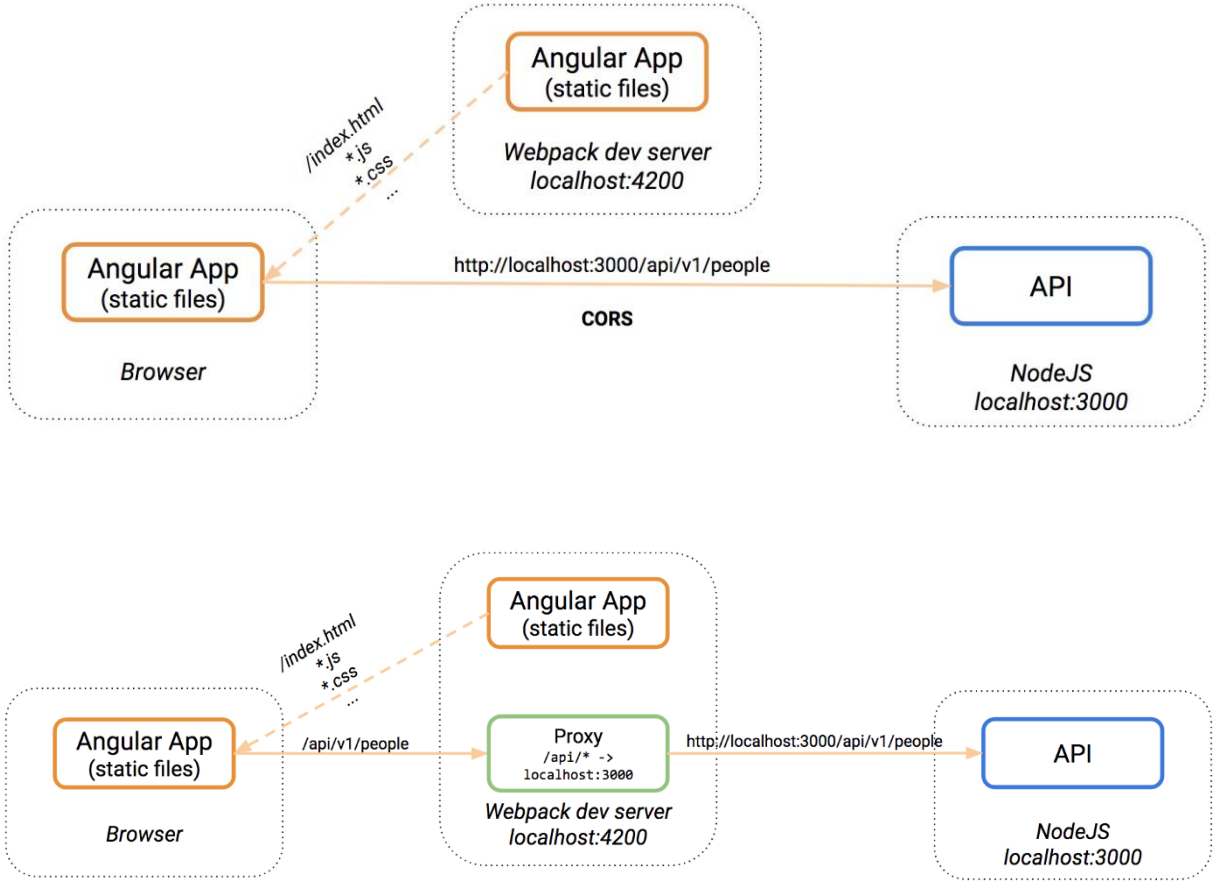
- Access-Control-Allow-Origin response 헤더 추가

```
app.get('/data', (req, res) => {
    res.header("Access-Control-Allow-Origin", "*");
    res.send(data);
});
```

- CORS 미들웨어 적용

해결 방안

- Proxy 설정



Q & A