webMethods API Gateway CORS Guide - 10.11

- **References**
  - CORS 란
  - CORS의 올바른 해결법
  - Simple Request vs Preflight Request
  - CORS 403 에러
  - webMethods API Gateway CORS 설정 방법 (165p 참고)

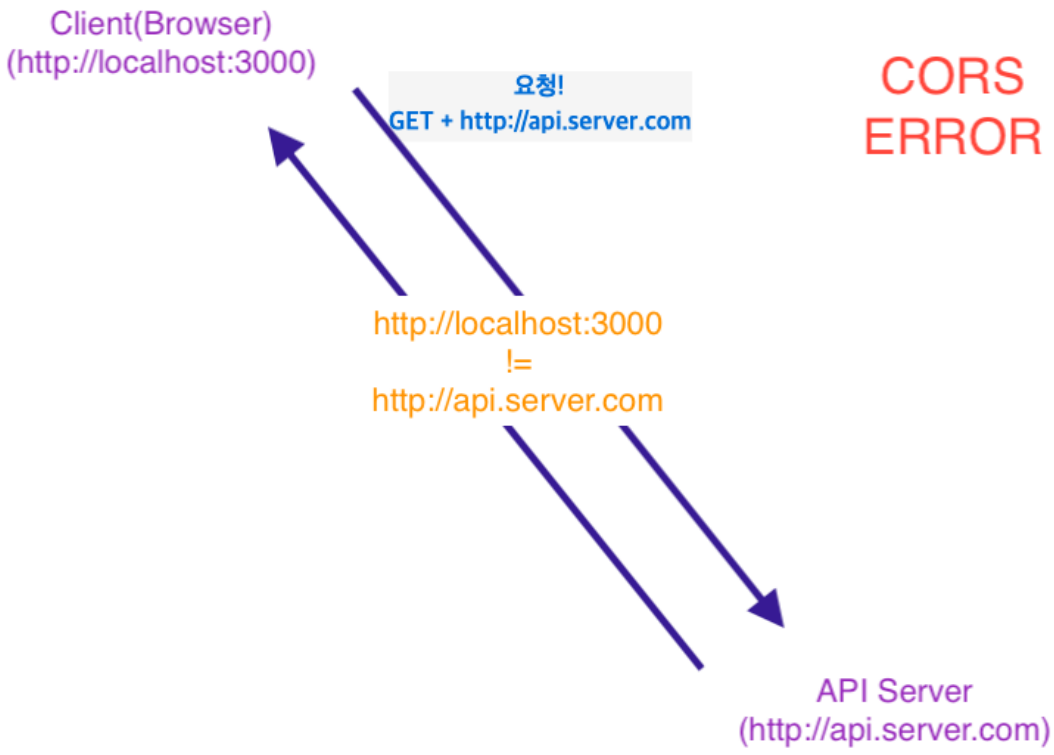
- **SOP (Same Origin Policy) 보안 정책이란?**
  - 대부분의 웹 브라우저에서 Same Origin Policy 를 준수
  - 내가 접속한 사이트(Origin)에서 다른 Origin에 요청한 것을 기본적으로 제한하여 해킹 공격 방어
  - 다른 Origin에서 온 HTML 특정 Tag (<img>,<script>,<link>,<iframe> 등) 는 임베딩 할 수 있게 허용

- **CORS (Cross-Origin Resource Sharing, 교차 출처 리소스 공유) 란**
  - 교차 호출이 해킹에 활용된다는 보안 취약점을 해결하기 위해 등장 (브라우저 - 웹 서버 간)
  - 브라우저에서 보안적인 이유로 Cross-Origin HTTP 요청을 제한
    - 호출 발생 시점부터 Origin을 체크하도록 보안 강화

- 이러한 호출을 위해서는 서버 측의 동의가 필요
  - HTTP header 를 사용한 메커니즘
  - 다른 출처(서버 측)의 응답에 올바른 CORS 헤더가 포함되지 않는 한 해당 API 를 사용하는 웹 애플리케이션은 애플리케이션이 로드 된 동일한 출처의 리소스만 요청할 수 있다

- **Origin 이란?**
  - [Protocol]:///[HOST IP 주소 또는 도메인 명]:[포트번호]

- **CORS 에러가 발생하는 원인**

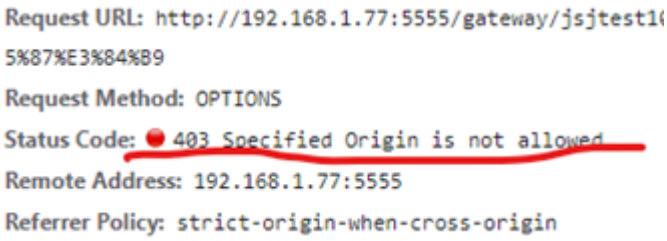


  - 대부분의 CORS Error 는 Origin의 불일치에서 발생
    - 프로토콜 -> HTTP 또는 HTTPS 로 프로토콜을 일치
    - 도메인 -> 도메인을 일치 시킨다. (IP 또는 도메인으로)

        - 포트번호 -> 포트번호를 일치 시킨다.
    - CORS 기능을 지원하는 서버 측 설정에서의 문제의 경우 (브라우저에서 개발자 도구 F12 > Network 탭에서 확인 가능) → 403 Error 발생
        - 허용 된 헤더 가 아닌 경우

헤더가 허용되지 않았을 때 에러 (403)

        - 허용 된 메소드가 아닌 경우 (**Preflight 메시지는 OPTIONS 메소드를 사용하므로 반드시 허용 메소드에 OPTIONS 포함**)
        - 허용 된 Origin이 아닌 경우



Origin 허용 불가 (403)

Host 허용 불가 (403)

- **브라우저의 리소스 요청 방식 (Preflight request vs Simple request)**

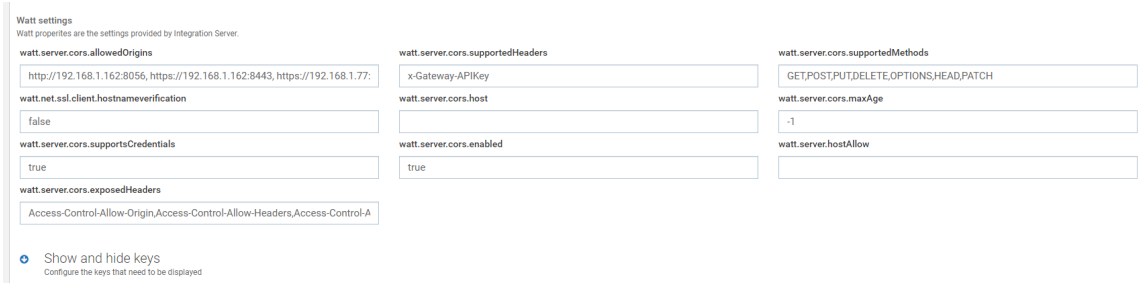
- Preflight request 란
  - CORS 이슈의 어떤 조건을 만족하는 경우 해당 리소스에 접근하기 위한 사전 요청
  - Simple request 와 다르게 '사전검사'를 한 후 브라우저에서 실제 요청을 보낸다.
  - Simple request 의 조건이 충족되지 않는 경우의 요청일 때 Preflight request 를 보낸다.
  - 해당 요청은 HTTP 메소드의 OPTIONS를 이용하여 이루어진다.
- Simple request 란
  - CORS preflight 를 발생 시키지 않는 요청
  - 다음의 조건을 만족하는 경우의 요청
    - GET, HEAD, POST 중 하나의 HTTP Method
    - User agent 에 의해 자동으로 정해지는 헤더
      - Connection
      - User-Agent
      - Fetch spec에 정의 된 forbidden header name이 아닌 경우
      - Fetch spec에 정의 된 CORS-safelisted request-header 를 포함하는 경우
    - Content-type 헤더의 값이 다음 중 하나일 때
      - application/x-www-form-urlencoded
      - multipart/form-data
      - text/plain
    - XMLHttpRequest.upload 에 아무런 이벤트 핸들러, 리스너가 등록되어있지 않은 경우
    - ReadableStream object가 요청에 포함되어 있지 않을 때

- **요청 방식에 따른 CORS 에러 해결 방법**
  - Simple Request 의 경우

        - 브라우저가 받은 요청이 어느 Origin에서 시작했는지 헤더 추가
        - 서버 측 응답 헤더 **Access-Control-Allow-Origin** 에 요청 측 Origin 값을 넣어 준다.
    - Preflight Request 의 경우
        - 서버 측 응답 헤더에서 Access-Control-Allow-* 로 시작하는 헤더들의 설정이 필요 (HTTP 응답헤더 참고)
            - "Access-Control-Allow-Origin": 허용하는 Origin 나열
            - "Access-Control-Allow-Headers" : 허용하는 헤더 나열
            - "Access-Control-Allow-Methods": 허용하는 HTTP Method 나열
            - "Access-Control-Expose-Headers" : 브라우저가 접근할 수 있는 서버의 헤더 (화이트 리스트)
            - "Access-Control-Max-Age" : Preflight request 요청 결과를 캐싱하는 시간
            - "Access-Control-Allow-Credentials" : 자격 증명(쿠키, Authorization 헤더, TLS 클라이언트 인증서) 모드의 경우 JavaScript 코드에 응답을 노출할지 여부를 브라우저에 알려준다 (True일 때)

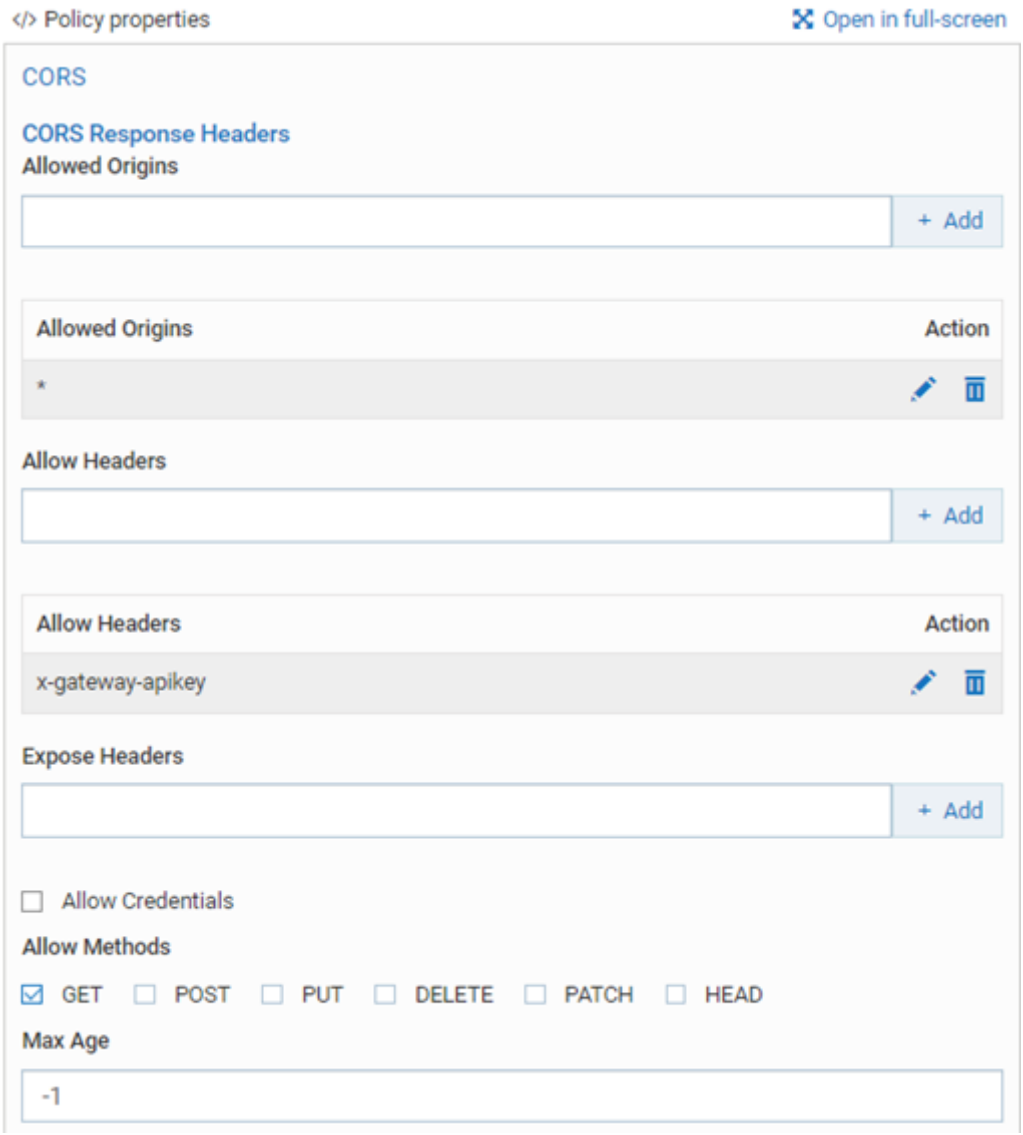
- **webMethods API Gateway 의 CORS 기능**
    - Extended Settings 에서 CORS 정책 지원
    - Response Processing Policy 에서 CORS 정책을 지원 (API Gateway의 API 단위 CORS 정책 설정)

- **Extended Settings 설정 방법 (165p - Configuring Integration Server to Accept CORS Request 참고)**



- watt.server.cors.enabled = true (API Gateway 의 CORS 기능 켜기)
- watt.server.cors.exposedHeaders = value1, value2 ... (API Gateway 에서 Client 브라우저 쪽으로 Expose 할 헤더 목록)
  - Access-Control-Allow-Origin
  - Access-Control-Allow-Headers
  - Access-Control-Allow-Methods
  - Access-Control-Expose-Headers
  - Access-Control-Max-Age
  - Access-Control-Allow-Credentials
- watt.server.cors.host = <hostname>:<port> 로컬네임서버나 DNS 서버를 사용하는 경우 허용 Origin 설정, 테스트 필요
- watt.server.cors.maxAge = <number of seconds> Preflight Request 의 요청 결과 캐싱 시간 설정 (default = -1 ⇒ disable)
- watt.server.cors.supportsCredentials = true , Access-Control-Allow-Credentials 헤더 지원 여부 설정
- watt.server.cors.supportedHeaders = header1,header2,header3... CORS 요청을 허용하는 요청 헤더를 지정. Preflight Request에 응답할 때 이 헤더를 Access-Control-Allow-Headers 응답 헤더 값으로 포함 (x-Gateway-APIKey 와 같은 Custom Header 설정에 사용)
- watt.server.cors.supportedMethods = GET,POST,PUT,PATCH,DELETE, "OPTIONS", 허용 HTTP 메소드 설정 Preflight request 는 OPTIONS 메소드를 사용하기에 설정하지 않으면 403 에러 발생

- **API Policy 설정 방법 예시 (Response Processing > CORS 설정)**



- Allowed Origins -> * (모두 허용)
- Allow Headers -> x-gateway-apikey (API Gateway의 Default API Key 헤더)