|  |  |  |
| --- | --- | --- |
| **[Java]HTTP 통신** | 작성자 | 엄태권 |
| 작성일 | 2021.04.29 |
|  |  |

* **서버통신과 CORS**

Ajax 등의 방식을 통해 클라이언트에서 Controller 혹은 타 API 서버로 URL을 지정하여 데이터 통신이 가능함.

단순 우리서버(Java Server)와 클라이언트가 통신하는 경우 문제가 될 것이 없음.

* **예외상황**

**클라이언트 단에서 API 서버로 직접적인 호출을 할 경우 CORS error가 발생함.**

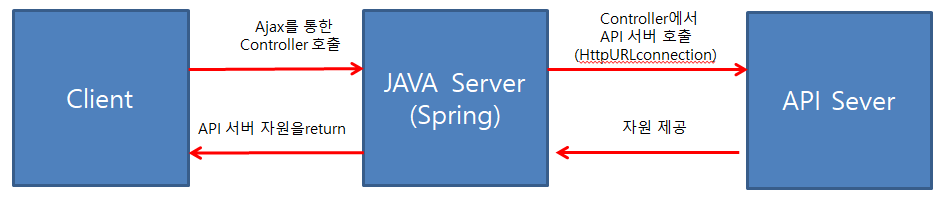
+) CORS 란 추가 HTTP헤더를 사용하여, 한 출처에서 실행중인 웹 애플리케이션이 다른 출처의 선택한 자원에 접근할 수 있는 권한을 부여하도록 브라우저에 알려주는 체제임.

**1)번 해결 방법**

무분별한 클라이언트(인증 X)가 직접적으로 API 서버에 자원을 요청할 경우 CORS error가 발생하므로, 직접적으로 응답 헤더의 Access-Control-Allow-Origin 이라는 값에 허용된 출처를 적어줌으로써 해결이 가능함.

**2)번 해결 방법**

결국 클라이언트 단에서는 JAVA서버(우리서버)로 데이터를 요청하는 것은 문제가 없기 때문에 아래와 같은 순서를 거쳐 정책을 벗어나지 않고 자원 공유가 가능함.



* **JAVA HTTPURLconnection**

JAVA 에서도 Http 통신이 가능하며 이를 사용하여 앞서 정책적인 부분을 지키면서 API서버와 통신하기 위해서 HTTPURLconnection을 사용함.

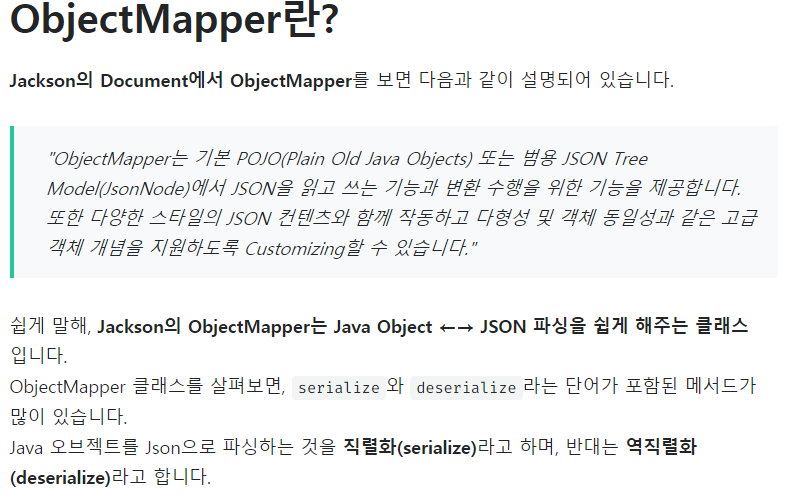
**//클라이언트 서버로부터 Ajax 요청을 처리하는 Controller**

@GetMapping(value = "/test.ajax", produces = MediaType.*APPLICATION\_JSON\_UTF8\_VALUE*)  
@ResponseBody  
public Map<String, String> postTest(String test, String test2) throws IOException {  
  
 Map<String, String> map = new HashMap<>();  
 map.put("test", test);  
 map.put("test2", test2);

**//ObjectMapper > 아래 설명 참조**

String json = new ObjectMapper().writeValueAsString(map);  
 *log*.info(json);

참조 : [클릭이동](https://velog.io/@conatuseus/RequestBody%EC%97%90-%EC%99%9C-%EA%B8%B0%EB%B3%B8-%EC%83%9D%EC%A0%95%EC%9E%90%EB%8A%94-%ED%95%84%EC%9A%94%ED%95%98%EA%B3%A0-Setter%EB%8A%94-%ED%95%84%EC%9A%94-%EC%97%86%EC%9D%84%EA%B9%8C-2-ejk5siejhh)



**//HTTP 통신할 URL 및 통신방식에 대한 설정**

URL url = new URL("http://localhost:5000/test");  
HttpURLConnection conn = (HttpURLConnection) url.openConnection();  
conn.setRequestMethod("POST");  
conn.setDoOutput(true);  
conn.setRequestProperty("Accept", "application/json");  
conn.setRequestProperty("Content-Type", "application/json");

**setRequestMethod : 전송방식을 규정**

**setDoOutPut : OutputStream으로 POST 데이터를 넘겨주겠다는 옵션**

**setRequestMethod : 전송 방식에 대한 규정옵션**

* **"http://localhost:5000/test” URL로 POST 방식의 통신을 할 것이며, 서버 Response Data를 JSON형식의 타입으로 요청하고, RequestBody 전달시 application/json로 서버에 전달 하겠다.**

**//Request Body에 Data를 담기위해 객체 생성 및 셋팅**

OutputStreamWriter outStream = new OutputStreamWriter(conn.getOutputStream(), "UTF-8");  
PrintWriter writer = new PrintWriter(outStream);  
writer.write(json);  
writer.flush();

* **UTF-8로 위의 JSON 타입의 데이터를 request body에 셋팅하겠다.**

**//응답받은 데이터에 대한 처리**

BufferedReader in = new BufferedReader(new InputStreamReader(conn.getInputStream()));  
String inputLine;  
String resultXmlText = "";  
while((inputLine = in.readLine()) != null) {  
 resultXmlText += inputLine;  
}  
in.close();  
conn.disconnect();

* **response 받은 데이터에 대해 출력을 진행하며, 출력진행후 inputStream을 종료 하고, 목적지서버와의 접속을 해지함.**

map.put("result", resultXmlText);  
  
 return map;  
}

* **API 서버로부터 응답받은 데이터를 map에 넣어 클라이언트 단에 return함.**