

REST(Representational State Transfer)

- 자원(서비스 할 대상)을 화면을 통해서 표시하던 것에서 자원에 대한 정보를 xml 또는 JSON으로 표시
- URI는 하나의 고유한 리소스를 대표하도록 설계

자원 식별

URI를 통해 자원을 고유하게 식별한다.

표현 방식

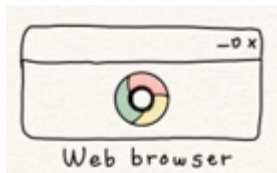
JSON또는 XML 형태로 자원을 표현한다.

상태 전이

HTTP 메소드를 통해 자원의 상태를 변경한다.

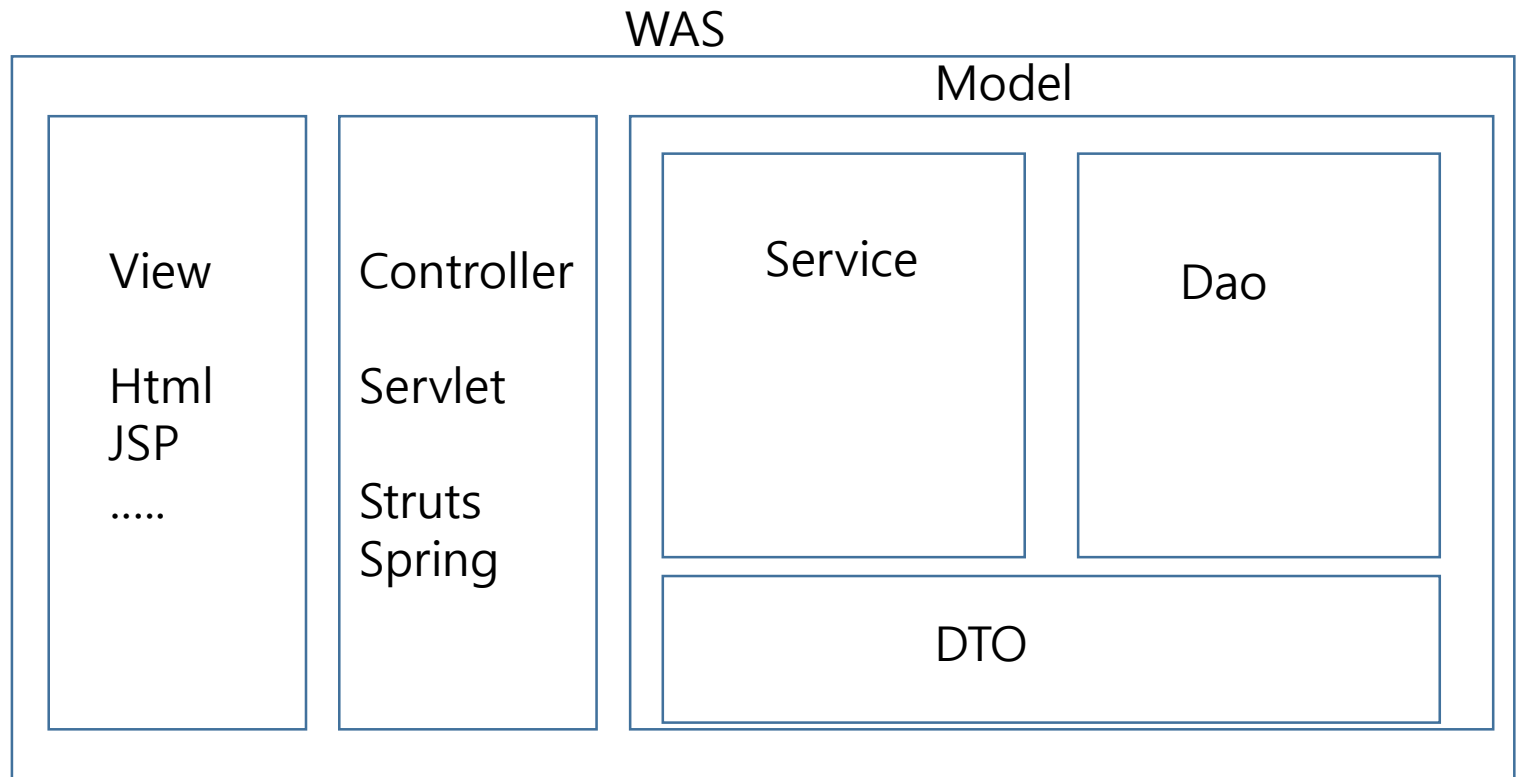
서비스 아키텍처의 변화

1. Web만 서비스



요청

응답



다양한 플랫폼 지원을 위한 아키텍처 변화



웹 전용 시스템

단일 플랫폼을 위한 통합 아키텍처



다중 플랫폼 시대

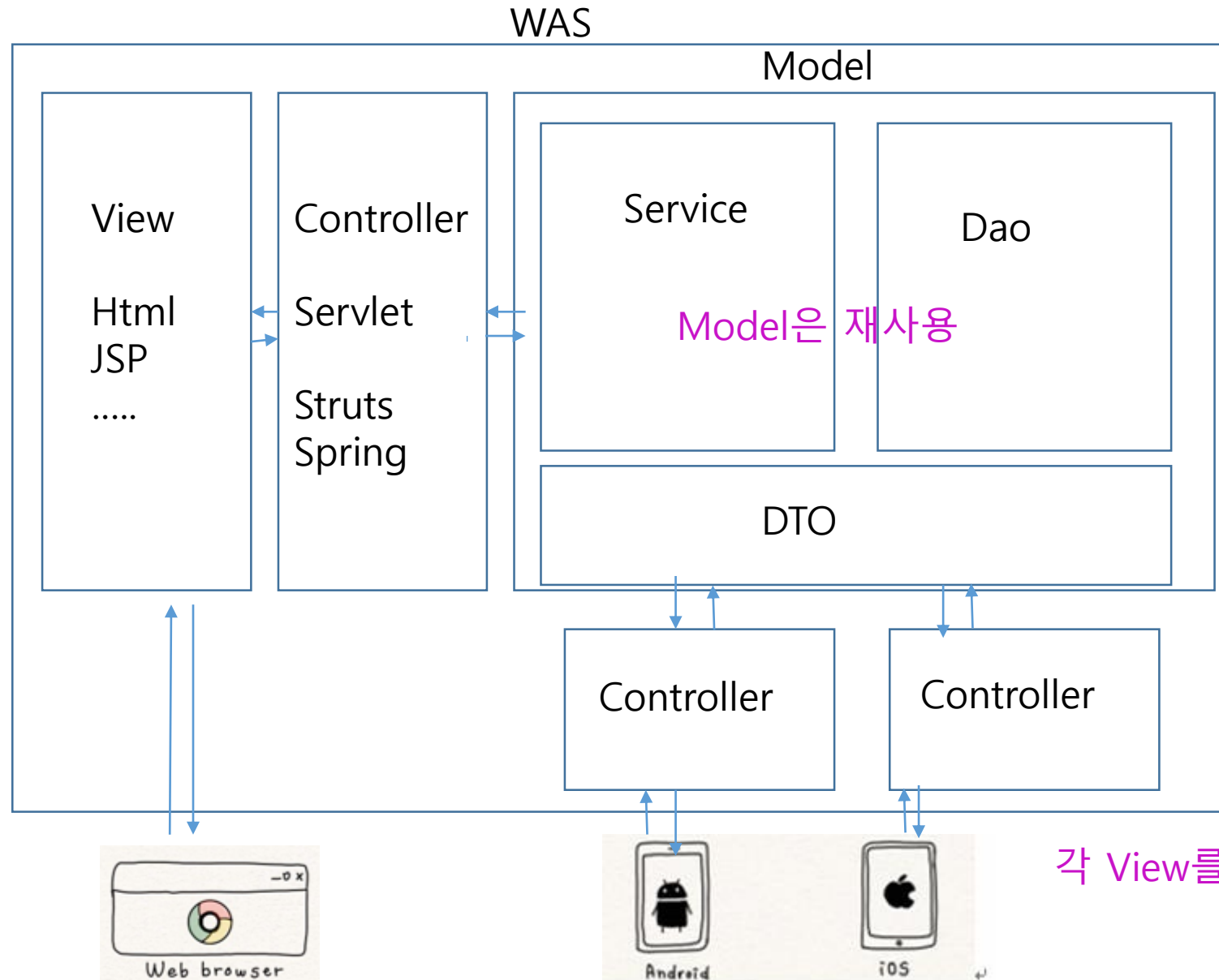
웹, 안드로이드, iOS 등 다양한 클라이언트 지원



모델 재사용

다양한 뷰를 위한 컨트롤러 분리 필요

2. View가 다변화 된 시대 (Web, android, ios,)



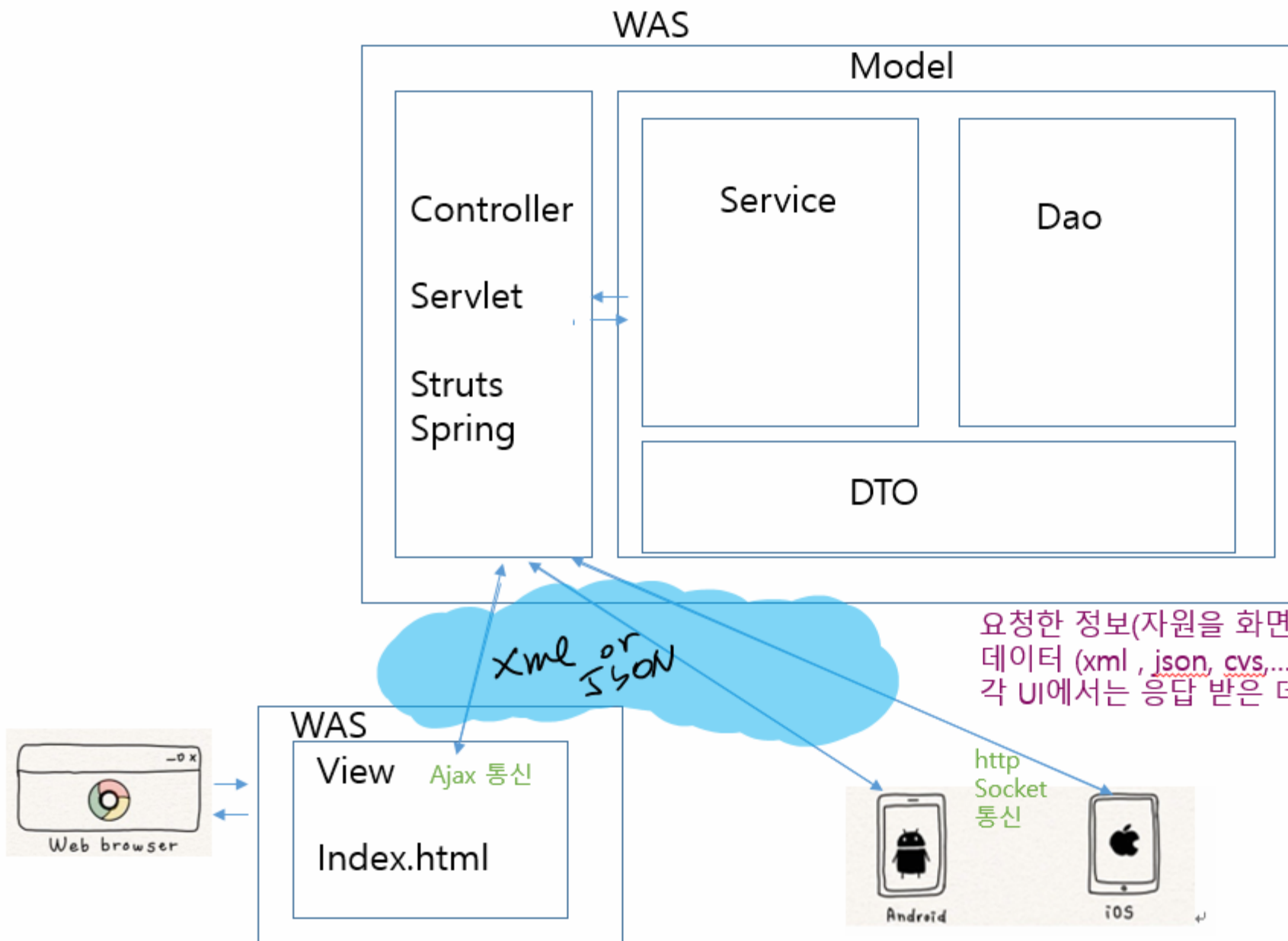
각 View를 위한 Controller가 필요

3. REST – FUL Service

자원에 대한 요청을 URI로 하고
구별하고 상세는 **method** 방식으로 구별

Restful 에서는 모두 사용

insert : C → post
Select : R → get
update : U → put
delete : D → delete



요청한 정보(자원을 화면이 아닌
데이터 (xml, json, cvs,...)로 응답하고
각 UI에서는 응답 받은 데이터로 화면을 만든다.

동일 출처 정책과 CORS

SOP (Same-Origin Policy)

브라우저는 보안상 SOP를 따른다.

Ajax 통신은 동일한 출처의 리소스만 요청할 수 있다.

CORS (Cross-Origin Resource Sharing)

- 최초에 리소스를 제공한 출처(Origin)와 다른 곳에서 리소스를 요청하는 경우(cross-origin 요청)에는

특정 Http header를 사용하여 웹 어플리케이션의 cross-origin 요청을 브라우저가 제한적으로 허용하는 정책

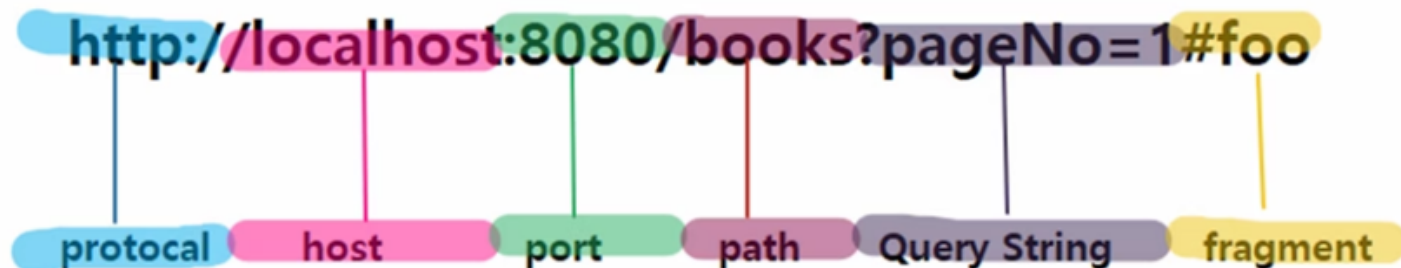
Access-Control-Request-Method : 본 요청에서 사용하는 메서드

Access-Control-Request-Headers: 본 요청에 포함될 헤더

Origin : 요청을 보낸 출처로 URL 중 scheme과 host, port만 명시

CORS 의 필요성

REST Full API를 서비스 하는 경우 web front는 다른 서버에서 서비스 되는 경우가 많으므로 cross-origin 요청을 필요한 경우가 많아지고 있기 때문에 안전하게 처리할 정책이 대두되었다



출처(Origin)

-서버의 위치를 의미하는 것으로 URL의 schema(protocol), host(IP or domain), Port로 이루어진 값(객체)를 말한다.

CORS 요청의 종류

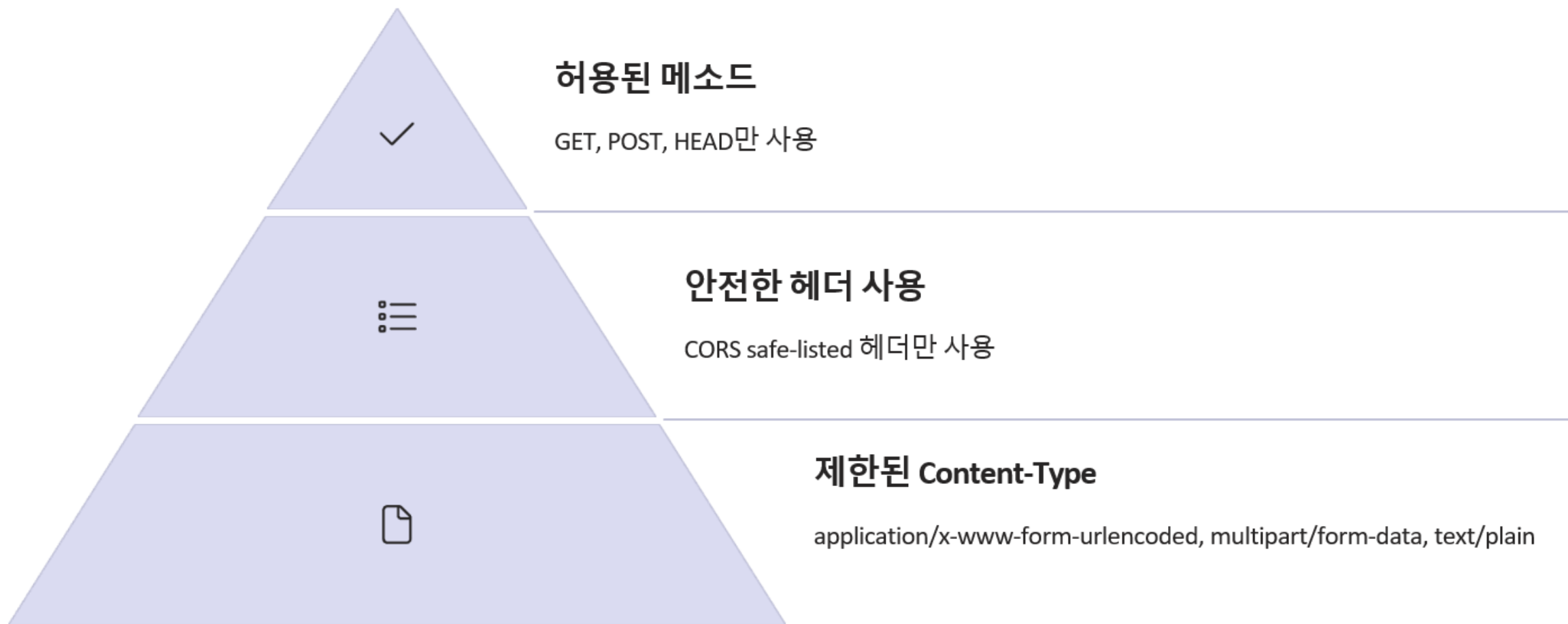
Preflight 요청

본 요청 전에 OPTIONS 메소드로 서버에 권한 확인을 먼저 수행합니다.

Simple 요청

특정 조건을 만족하면 예비 요청 없이 바로 서버에 요청합니다.

Simple 요청의 조건과 처리 방식



조건	Simple Request/ Preflight
GET 요청 + 아무 특별한 헤더 없음	Simple Request
POST 요청 + application/x-www-form-urlencoded Content-type	Simple Request
POST 요청 + application/json Content-Type	Preflight 발생
GET 요청 + Authorization: Bearer xxx 헤더 추가	Preflight 발생
PUT 요청	Preflight 발생
DELETE 요청	Preflight 발생

서버가 CORS를 허용한 경우의 응답

요청 전송

브라우저는 Origin 헤더를 포함하여 요청한다.

브라우저 처리

브라우저는 허용된 출처임을 확인하고 응답을 처리한다.



서버 처리

서버는 요청을 검토하고 허용된 경우 응답한다.

응답 반환

Access-Control-Allow-Origin 헤더가 포함된 응답을 보낸다.

• Simple request

Ex) http://localhost:9000에서 서버(http://localhost/bookproject/books)에 요청을 보낸 경우

서버에서 허용 한 경우

The image shows a web browser's developer tools interface. The left pane displays the 'General' tab for a request to `http://localhost/bookproject/books?pageNo=1&key=all&word=`. The request method is `GET` and the status code is `200 OK`. The right pane shows the 'Response' tab with a JSON array of book objects. The 'Access-Control-Allow-Origin' header is `http://localhost:9000`, and the 'Vary' header lists `Access-Control-Request-Headers`, `Access-Control-Request-Method`, and `Origin`. The 'Origin' header in the request is `http://localhost:9000`.

Request URL:	http://localhost/bookproject/books?pageNo=1&key=all&word=
Request Method:	GET
Status Code:	200 OK
Remote Address:	[::1]:80
Referrer Policy:	strict-origin-when-cross-origin

Response Headers	Raw
Access-Control-Allow-Origin:	http://localhost:9000
Connection:	keep-alive
Content-Type:	application/json
Date:	Mon, 13 Nov 2023 07:06:25 GMT
Keep-Alive:	timeout=60
Transfer-Encoding:	chunked
Vary:	Access-Control-Request-Headers
Vary:	Access-Control-Request-Method
Vary:	Origin

Request Headers	Raw
Accept:	application/json, text/plain, */*
Accept-Encoding:	gzip, deflate, br
Accept-Language:	ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7
Connection:	keep-alive
Host:	localhost
Origin:	http://localhost:9000
Referer:	http://localhost:9000/

```
{
  "books": [
    {
      "isbn": "555555",
      "title": "4",
      "author": "4",
      "price": 4,
      "describ": "4",
      "img": null
    },
    {
      "isbn": "444444",
      "title": "4",
      "author": "4",
      "price": 4,
      "describ": "4",
      "img": null
    },
    {
      "isbn": "1313-12",
      "title": "boot unit test",
      "author": "ssafy",
      "price": 35000,
      "describ": "유닛 테스트",
      "img": null
    }
  ],
  "page": {
    "key": "all",
    "word": "",
    "pageLink": null,
    "pageNo": 1,
    "total": 7,
    "interval": 3,
    "start": 0
  }
}
```

• Simple request

Ex) `http://localhost:9000`에서 서버(`http://localhost/bookproject/books`)에 요청을 보낸 경우

서버에서 허용 안한 경우

▼ General	
Request URL:	<code>http://localhost/bookproject/books?pageNo=1&key=all&word=</code>
Request Method:	GET
Status Code:	200 OK
Referrer Policy:	strict-origin-when-cross-origin
▼ Response Headers <input type="checkbox"/> Raw	
Connection:	keep-alive
Content-Type:	application/json
Date:	Mon, 13 Nov 2023 07:03:53 GMT
Keep-Alive:	timeout=60
Transfer-Encoding:	chunked
▼ Request Headers <input type="checkbox"/> Raw	
Accept:	application/json, text/plain, */*
Accept-Encoding:	gzip, deflate, br
Accept-Language:	ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7
Connection:	keep-alive
Host:	localhost
Origin:	<code>http://localhost:9000</code>

✖ `books?pageNo=1&key=all&word=`

CORS error

xhr

✖ Access to XMLHttpRequest at '`http://localhost/bookproject/books?pageNo=1&key=all&word=`' from origin '`http://localhost:9000`'
'Access-Control-Allow-Origin' header is present on the requested resource.

▶ `AxiosError {message: 'Network Error', name: 'AxiosError', code: 'ERR_NETWORK', config: {...}, request: XMLHttpRequest, ...}`

응답 헤더에 `Access-Control-Request-Headers`와 `Method` 가 없어
브라우저에서 CORS 오류를 발생 시키고 응답 데이터를 주지 않는다

서버가 CORS를 허용하지 않은 경우



오류 발생

브라우저는 응답을 차단하고 오류를 콘솔에 출력한다



접근 제한

스크립트에서 응답 데이터에 접근할 수 없다.



해결 방법

서버 측에서 적절한 CORS 헤더를 설정해야 한다.

Preflight 요청의 구조



OPTIONS 메소드 사용

본 요청 전 권한 확인을 위한 예비 요청



특수 헤더 포함

Access-Control-Request-Method, Access-Control-Request-Headers, Origin



서버 응답 확인

허용 여부에 따라 본 요청 실행 결정

Preflight 요청 성공 시 흐름



예비 요청

OPTIONS 메소드로 권한 확인



서버 승인

적절한 CORS 헤더로 응답



본 요청 전송

실제 API 요청 실행



응답 처리

데이터 수신 및 처리

• Preflight request - 성공시

Name	×	Headers	Preview	Response	Initiator	Timing
books	▼	General				
books		Request URL:		http://localhost/bookproject/books		
data:image/svg+xml,...		Request Method:		OPTIONS		
		Status Code:		200 OK		
		Remote Address:		:::1:80		
		Referrer Policy:		strict-origin-when-cross-origin		
	▼	Response Headers	<input type="checkbox"/> Raw			
		Access-Control-Allow-Headers:		content-type		
		Access-Control-Allow-Methods:		PUT		
		Access-Control-Allow-Origin:		http://localhost:9000		
		Access-Control-Max-Age:		1800		
		Allow:		GET, HEAD, POST, PUT, DELETE, OPTIONS, PATCH		
		Connection:		keep-alive		
		Content-Length:		0		
		Date:		Mon, 13 Nov 2023 08:07:30 GMT		
		Keep-Alive:		timeout=60		
		Vary:		Access-Control-Request-Headers		
		Vary:		Access-Control-Request-Method		
		Vary:		Origin		
	▼	Request Headers	<input type="checkbox"/> Raw			
		Accept:		*/*		
		Accept-Encoding:		gzip, deflate, br		
		Accept-Language:		ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7		
		Access-Control-Request-Headers:		content-type		
		Access-Control-Request-Method:		PUT		

Name	×	Headers	Payload	Preview	Response	Initiator	Timing
books	▼	General					
books		Request URL:			http://localhost/bookproject/books		
data:image/svg+xml,...		Request Method:			PUT		
		Status Code:			200 OK		
		Remote Address:			:::1:80		
		Referrer Policy:			strict-origin-when-cross-origin		
	▼	Response Headers	<input type="checkbox"/> Raw				
		Access-Control-Allow-Origin:			http://localhost:9000		
		Connection:			keep-alive		
		Content-Length:			7		
		Content-Type:			application/json		
		Date:			Mon, 13 Nov 2023 08:07:30 GMT		
		Keep-Alive:			timeout=60		
		Vary:			Access-Control-Request-Headers		
		Vary:			Access-Control-Request-Method		
		Vary:			Origin		
	▼	Request Headers	<input type="checkbox"/> Raw				
		Accept:			application/json, text/plain, */*		
		Accept-Encoding:			gzip, deflate, br		
		Accept-Language:			ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7		
		Connection:			keep-alive		
		Content-Length:			115		
		Content-Type:			application/json;charset=UTF-8		
		Host:			localhost		
		Origin:			http://localhost:9000		

- Preflight request – 서버에서 허용하지 않은 경우

Name	×	Headers	Payload	Preview	Response	Initiator	Timing
✖ books	▼	General					
		Request URL:	http://localhost/bookproject/books				
		Request Method:	PUT				
		Status Code:	● 200 OK				
		Referrer Policy:	strict-origin-when-cross-origin				
	▼	Response Headers	<input type="checkbox"/> Raw				
		Connection:	keep-alive				
		Content-Length:	7				
		Content-Type:	application/json				
		Date:	Mon, 13 Nov 2023 08:18:15 GMT				
		Keep-Alive:	timeout=60				
	▼	Request Headers	<input type="checkbox"/> Raw				
		Accept:	application/json, text/plain, */*				
		Accept-Encoding:	gzip, deflate, br				
		Accept-Language:	ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7				
		Connection:	keep-alive				
		Content-Length:	112				
		Content-Type:	application/json;charset=UTF-8				
		Host:	localhost				
		Origin:	http://localhost:9000				
		Referer:	http://localhost:9000/				

✖ Access to XMLHttpRequest at '<http://localhost/bookproject/books>' from origin '<http://localhost:9000>' has been blocked by CORS policy: the requested resource.

▶ `AxiosError {message: 'Network Error', name: 'AxiosError', code: 'ERR_NETWORK', config: {...}, request: XMLHttpRequest, ...}`

✖ ▶ PUT <http://localhost/bookproject/books> net::ERR_FAILED 200 (OK)

• CORS with credentials

- CORS는 보안상의 이유로 쿠키를 요청으로 보낼 수 없도록 막는다.
- credentials을 통해 쿠키를 사용할 수 있도록 서버측에서 허용한다.

```
5 @Configuration
7 public class WebConfig implements WebMvcConfigurer {
30     @Override
31     public void addCorsMappings(CorsRegistry registry) {
32         System.out.println("*****CORS 설정");
33         registry.addMapping("/") // 프로그램에서 제공하는 URL
34             // .allowedOrigins("*") // request를 허용할 출처를 명시, 전체 허용 (
35             // .allowedOriginPatterns("*") // request를 허용할 출처를 명시, 전체 허용 (
36             // WhiteList 설정 - 특정 URL:port로 요청 오는 것을 허용하기
37             .allowedOrigins("http://localhost:9000", "http://localhost:9001")
38             .allowedHeaders("*") // 어떤 헤더들을 허용할 것인지
39             .allowedMethods("*") // 어떤 메서드를 허용할 것인지 (GET, POST,
36             // 쿠키 요청을 허용한다(다른 도메인 서버에 인증하는 경우에만 사용해야하며, true 설정시 !
37             // JSESSIONID도 cookie로 저장할 수 있다
38             .allowCredentials(true)
39             // .maxAge(3500); // preflight 요청에 대한 응답을 브라우저에
40             ;
41     }
42 }
```