22.12.29

📌Http 요청 데이터의 3가지 형식(클라이언트 → 서버)

1) GET - 쿼리 파라미터

- Http 메시지 헤더에 querystring 데이터 추가
- 검색, 필터, 페이징 시 주로 사용

2) POST - HTML Form

- Http 메시지 바디에 querystring 형식의 데이터 추가
- content-type: application/x-www-form-urlencoded

*1)과 2)는 서버에서 request 객체에 의해 parameter로 파싱되어 사용되며, 요청 파라미터라고 함.

3) POST Message Body에 데[이터 직접 담아 요청

- **HTTP API**에서 주로 사용, JSON, XML, TEXT..
- POST, PUT, PATCH

📌Front Controller

- **프론트 컨트롤러 서블릿** 하나로 클라이언트의 요청을 받음 (기존 - url 마다 서블릿이 매핑되어 있었음)
- 공통 처리 가능, 요청에 맞는 컨트롤러를 찾아 호출
- 스프링 웹 MVC의 **DispatcherServlet**이 FrontController 패턴으로 구현되어 있음

📌어댑터 패턴

- 한 프로젝트 안에서 여러 컨트롤러 인터페이스를 사용하고 싶음.

  → 어댑터를 사용하여 이를 적용할 수 있다.
- 해당하는 종류의 어댑터만 있으면 다양한 방식의 컨트롤러들을 적용할 수 있으므로, 컨트롤러의 이름을 더 넓은 범위인 **핸들러**로 변경
- 어댑터가 핸들러에 대한 일종의 인터페이스로 사용됨 → 각 레이어 간 추상화가 심화됨

📌멱등성

- 동일한 요청을 한 번 보내는 것과 여러 번 연속으로 보내는 것이 같은 효과를 지니고, 서버의 상태도 동일하게 남을 때, 해당 HTTP 메서드가 **멱등성**을 가졌다고 말한다. 다른 말로는, 멱등성 메서드에는 통계 기록 등을 제외하면 어떠한 **부수 효과(side effect)**도 존재해서는 안된다.
- 멱등성을 따질 때 **실제 서버의 백엔드 상태**만 보면 되고, 각 요청에서 반환하는 **응답 코드는 달라질 수 있음**.

  EX) DELETE 요청 시 한 번이나 여러 번 연속적으로 보내는 것이나 서버는 동일한 상태를 가지지만, 클라이언트에게 보내는 응답은 달라질 수 있다.(이미 리소스가 존재하지 않을 때 DELETE → 404)
- **GET, HEAD, PUT, DELETE** 가 멱등성을 가짐.
- **POST는 비멱등성**을 갖는다. POST는 기존 리소스를 수정하는 데도 사용되지만, 새로운 리소스를 생성하는데도 사용된다. 이 때, 여러 번의 POST 요청은 요청 만큼의 리소스를 생성하여 서버의 상태를 변경 시킬 수 있다.
- **PATCH**는 설계에 따라 멱등성과 비멱등성을 가질 수 있다. PACTH 메소드는 기본적으로 서버의 리소스의 상태를 부분 수정한다는 특성을 만족하면 되는데, 만약 {”operation”: “add”, “age”: 10}와 같은 방식으로 PATCH가 설계되면, PATCH 메서드를 여러 번 호출할 경우 리소스의 상태가 계속 변경된다. 따라서 멱등성을 보장할 수 없다. { “name” : "kim"} 과 같은 방식으로 서버 리소스를 수정한다면, 멱등성을 보장할 수 있다.

참고

- MDN, https://developer.mozilla.org/ko/docs/Glossary/Idempotent
- 인프런, https://www.inflearn.com/questions/110644/patch-메서드가-멱등이-아닌-이유