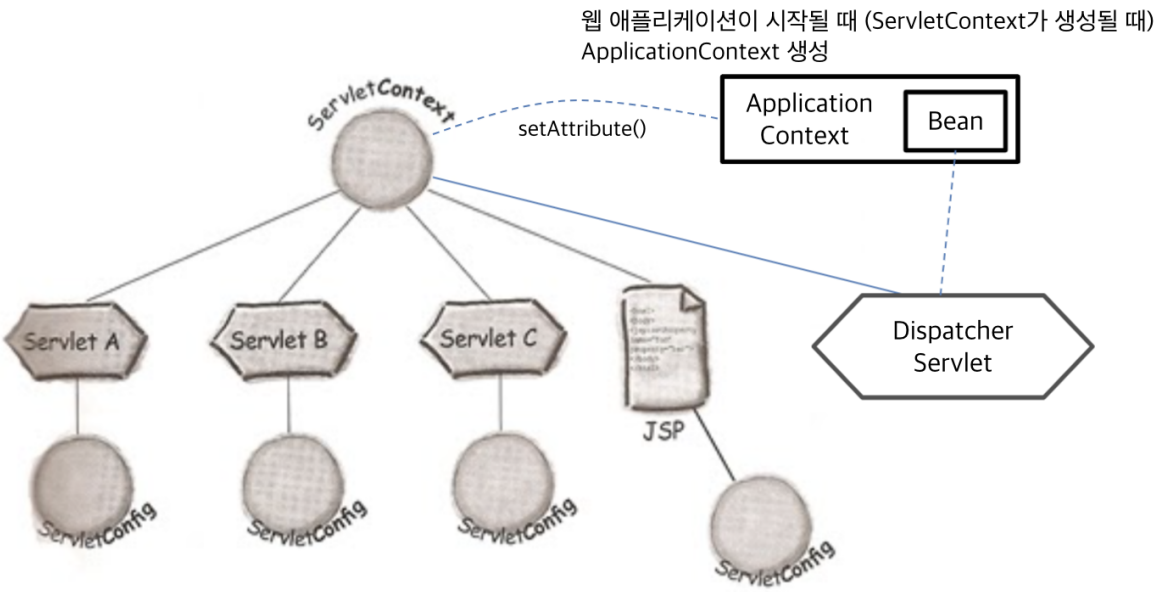
Spring MVC 2️⃣

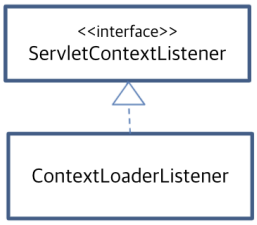
폼 처리

WebApplicationContext



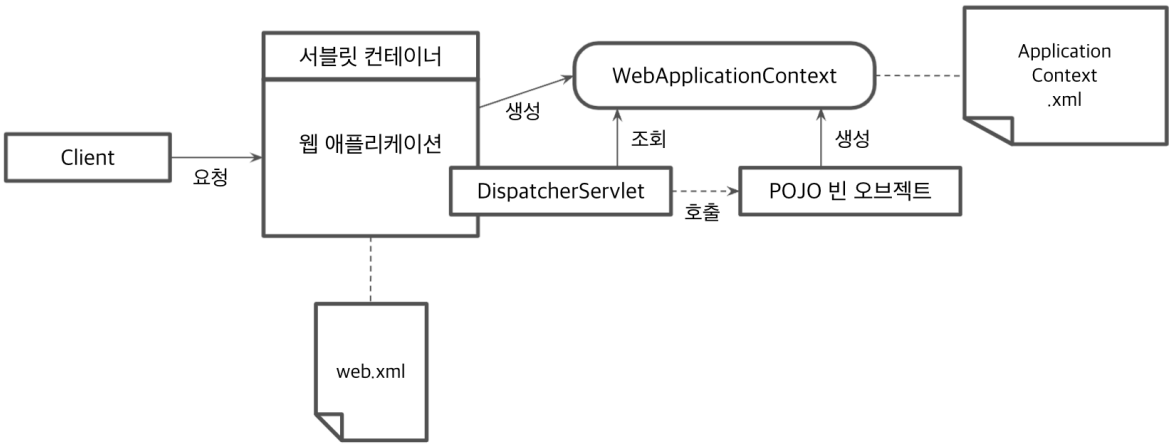
ContextLoaderListener - 루트 애플리케이션 컨텍스트 등록

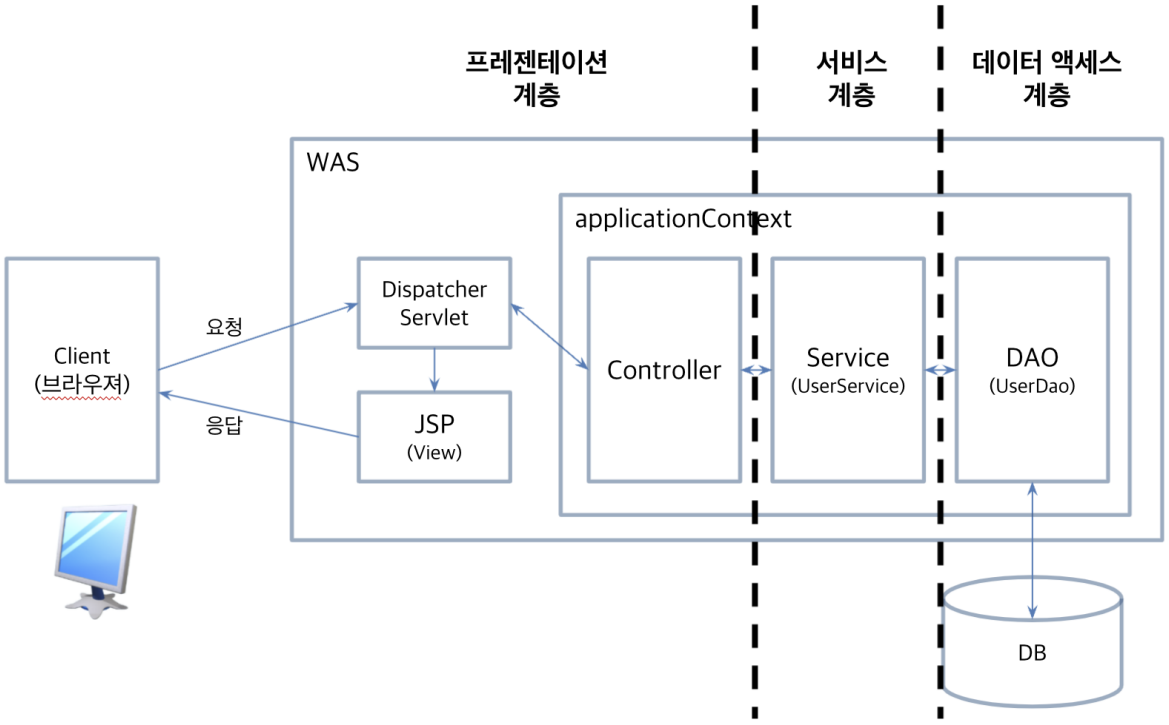
```
<listener>
    <listener-class>org.springframework.web.context.ContextLoaderListener
    </listener-class>
</listener>

web.xml
```

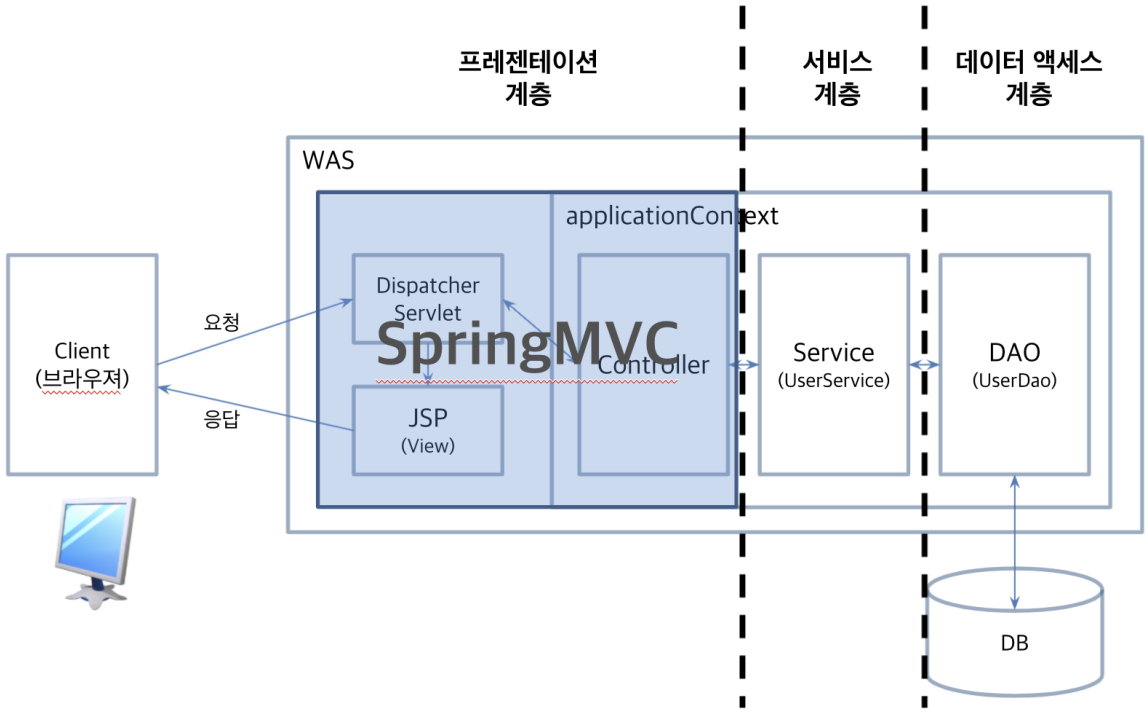


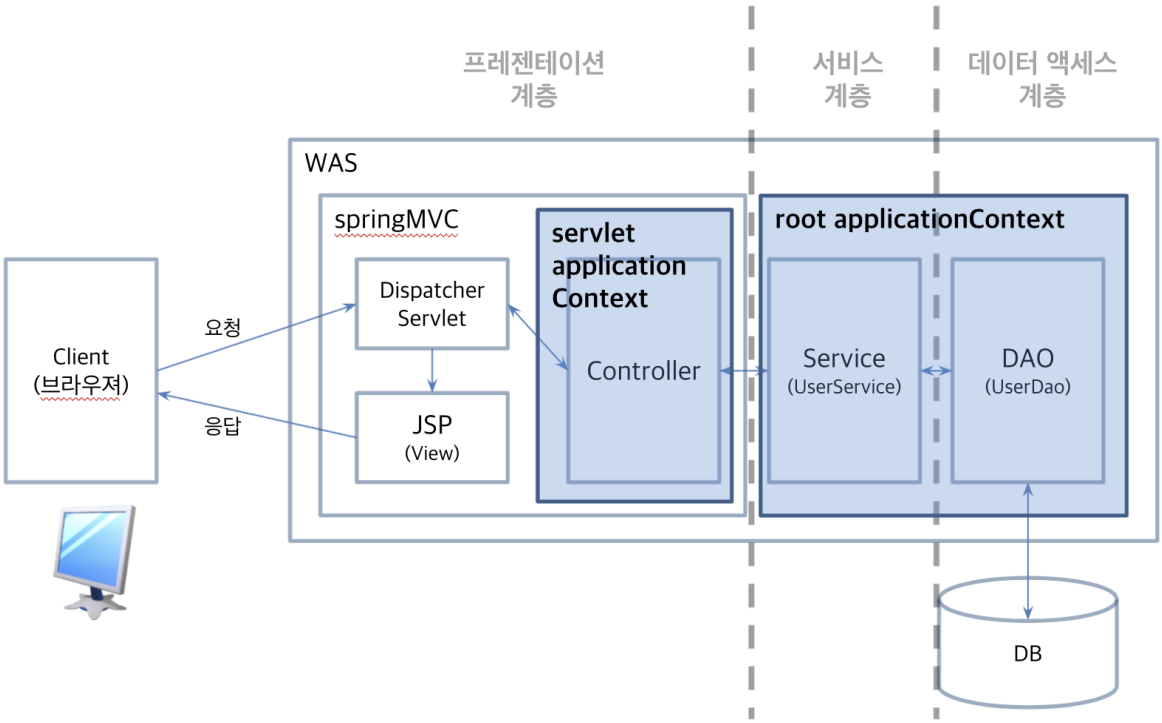

```
<!-- 옵션 : 설정파일 위치 직접 지정 -->
<context-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>
        /WEB-INF/daoContext.xml
        /WEB-INF/applicationContext.xml
    </param-value>
</context-param>

web.xml
```

웹 환경에서 스프링 애플리케이션이 동작하는 방식


서블릿 컨테이너
웹 애플리케이션
Client
요청
생성
WebApplicationContext
조회
생성
Application
Context
.xml
DispatcherServlet
호출
POJO 빈 오브젝트
web.xml


프레젠테이션
계층
서비스
계층
데이터 액세스
계층
WAS
applicationContext
Client
(브라우져)
요청
응답
Dispatcher
Servlet
JSP
(View)
Controller
Service
(UserService)
DAO
(UserDao)
DB


프레젠테이션
계층
서비스
계층
데이터 액세스
계층
WAS
applicationContext
Dispatcher
Servlet
SpringMVC
Controller
JSP
(View)
Service
(UserService)
DAO
(UserDao)
Client
(브라우저)
요청
응답
DB


프레젠테이션
계층
서비스
계층
데이터 액세스
계층
WAS
springMVC
servlet
application
Context
root applicationContext
Dispatcher
Servlet
Controller
JSP
(View)
Service
(UserService)
DAO
(UserDao)
Client
(브라우저)
요청
응답
DB

REST(ful) API

REST

- **REST**(Representational State Transfer)는 월드 와이드 웹**과 같은 분산** 하이퍼미디어 **시스템을 위한** 소프트웨어 아키텍처**의 한 형식**
- 로이 필딩(Roy Fielding)의 2000년 박사학위 논문에서 소개되었고 필딩은 HTTP의 주요 저자 중 한 사람임
- 엄격한 의미로 **REST**는 네트워크 아키텍처 원리의 모음.
- '네트워크 아키텍처 원리'란 자원을 정의하고 자원에 대한 주소를 지정하는 방법
- 간단한 의미로는, 웹 상의 자료를 HTTP위에서 SOAP이나 쿠키를 통한 세션 트랙킹 같은 별도의 전송 계층 없이 전송하기 위한 아주 간단한 인터페이스 말함

출처: 위키피디아(https://ko.wikipedia.org/wiki/REST)

API

- In computer programming, an **application programming interface** (**API**) is a set of subroutine definitions, protocols, and tools for building application software. In

general terms, it is a set of clearly defined methods of communication between various software components. A good API makes it easier to develop a computer program by providing all the building blocks, which are then put together by the programmer. An API may be for a web-based system, operating system, database system, computer hardware or software library.

출처: 위키피디아(https://wikipedia.org/wiki/API)

> 💡 REST API = REST 아키텍쳐 스타일을 따르는 API

REST 아키텍쳐 스타일

- 클라이언트-서버
- 스테이트리스
- 캐시
- 균일한 인터페이스
- 계층화된 시스템

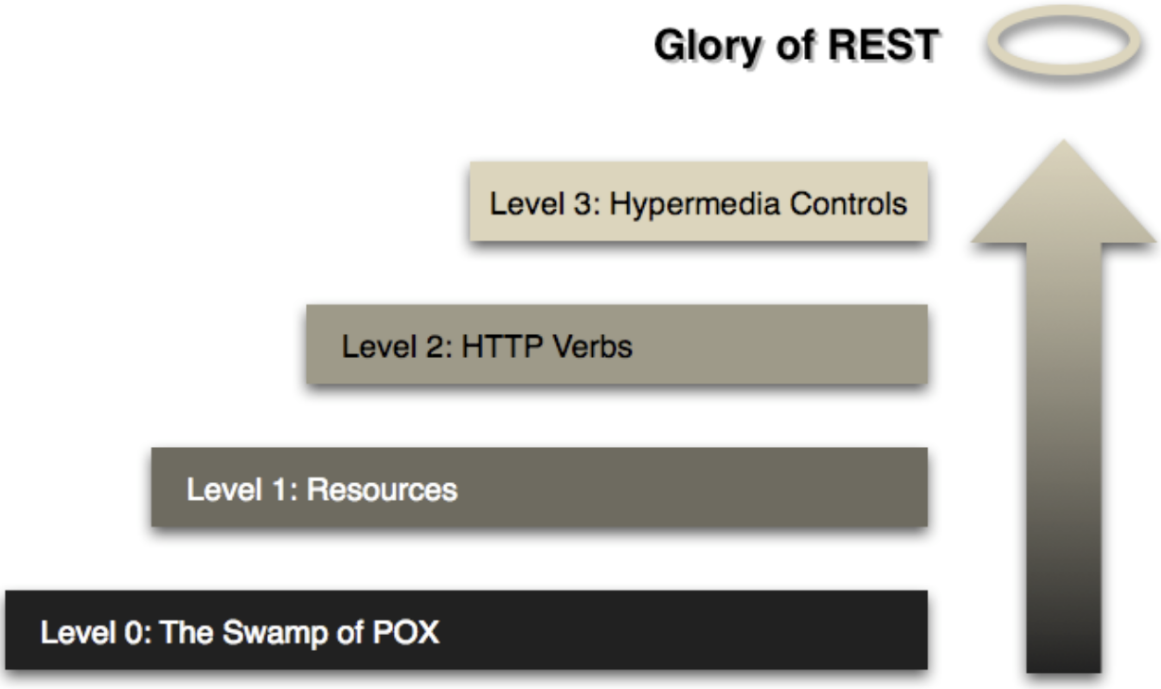
출처: https://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm

REST 아키텍쳐 스타일

- 클라이언트-서버 (client-server)
  - 사용자 인터페이스에 대한 관심을 데이터 저장에 대한 관심으로부터 분리함으로써 클라이언트의 이식성과 서버의 규모확정성을 개선한다.
- 스테이트리스 (stateless)
  - 클라이언트 서버의 통신에 상태가 없습니다. 모든 요청에는 필요한 모든 정보를 담고 있어 가시성이 좋고 요청 실패시 복원이 쉽기 때문에 신뢰성이 좋습니다. 상태를 저장할 필요가 없어 규모확장성이 개선됩니다.
- 캐시 (cache)
  - 캐시가 가능해야 합니다. HTTP가 가진 캐싱 기능이 적용 가능합니다. HTTP 프로토콜 표준에서 사용하는 Last-Modified태그나 E-Tag를 이용하면 캐싱 구현이 가능합니다.

- 균일한 인터페이스 (uniform interface)
  - URI로 지정한 리소스에 대한 조작을 통일되고 한정적인 인터페이스로 수행하는 아키텍처 스타일을 말합니다.
- 계층화된 시스템 (layered system)
  - REST 서버는 다중 계층으로 구성될 수 있으며 보안, 로드 밸런싱, 암호화 계층을 추가해 구조상의 유연성을 둘 수 있고 PROXY, 게이트웨이 같은 네트워크 기반의 중간매체를 사용할 수 있게 합니다.

Richardson Maturity Model



https://martinfowler.com/articles/richardsonMaturityModel.html

Representations

어떠한 리소스의 특정 시점의 상태를 반영하고 있는 정보이고 representation data와 representation metadata로 구성

GET /hello

```
HTTP/1.1 200 OK
Content-Length: 6
Date: Sun, 19 Mar 2017 10:20:47 GMT
Last-Modified: Sun, 19 Mar 2017 08:00:00 GMT
Content-Type: text/plain
Content-Language: en

hello
```

- representation data
  - hello
- representation metadata
  - Content-Type: text/plain
  - Content-Language: en

```
Content-Type: text/plain
Content-Language: ko

안녕하세요
```

```
Content-Type: application/json
Content-Language: en

{ “message” : “hello” }
```

GET /hello

```
Content-Type: text/plain
Content-Language: ko

안녕하세요
```

- representation data
  - 안녕하세요
- representation metadata
  - Content-Type: text/plain
  - Content-Language: ko

```
Content-Type: application/json
Content-Language: en

{ “message” : “hello” }
```

- representation data
  - { “message” : “hello” }
- representation metadata
  - Content-Type: application/json
  - Content-Language: en

https://blog.restcase.com/4-maturity-levels-of-rest-api-design/

HATEOAS

Hypermedia as the Engine of Application State (hey-dee-us)

```
{
   "id": "1",
   "contents": "공부합시다.",
```

```
    "createAt": "2020-01-01 12:00:00",
    "likes": 2,
    "likesOfMe": false,
    "comments": [],
    "writer": { "id": "2", "email": "harry@gmail.com", "name": "harry" },
    "links": [
       {"rel": "self", "action": "GET", "href": "/api/v1/posts/1"},
       {"rel": "deletePost", "action": "DELETE", "href": "/api/v1/posts/1" },
       {"rel": "getWriter", "action": "GET", "href": "/api/v1/users/1" },
       {"rel": "addComment", "action": "POST", "href": "/api/v1/posts/1/comments"}
    ]
}
```

API 설계

1. URI는 정보의 자원을 표현해야 한다. (리소스명은 동사보다는 명사를 사용)

```
❌ GET /members/delete/1
```

2. 자원에 대한 행위는 HTTP Method(GET, POST, PUT, DELETE 등)로 표현

```
✔️ DELETE /members/1
❌ GET /members/show/1
✔️ GET /members/1
✔️ POST /task/1/run
```

3. 슬래시 구분자(/)는 계층 관계를 나타내는 데 사용
4. URI 마지막 문자로 슬래시(/)를 포함하지 않는다.
5. 하이픈(-)은 URI 가독성을 높이는데 사용

RestController

```
Annotation indicating a method parameter should be bound to the body of the web request. The body
of the request is passed through an HttpMessageConverter to resolve the method argument
depending on the content type of the request. Optionally, automatic validation can be applied by
annotating the argument with @Valid.

Supported for annotated handler methods.

Since:      3.0
See Also:   RequestHeader, ResponseBody, org.springframework.web.servlet.mvc.method.
            annotation.RequestMappingHandlerAdapter
Author:     Arjen Poutsma
```

```java
@Target(ElementType.PARAMETER)
@Retention(RetentionPolicy.RUNTIME)
@Documented
public @interface RequestBody {
```

```
Annotation that indicates a method return value should be bound to the web response body.
Supported for annotated handler methods.

As of version 4.0 this annotation can also be added on the type level in which case it is inherited and
does not need to be added on the method level.

Since:      3.0
See Also:   RequestBody, RestController
Author:     Arjen Poutsma
```

```java
@Target({ElementType.TYPE, ElementType.METHOD})
@Retention(RetentionPolicy.RUNTIME)
@Documented
public @interface ResponseBody {


}
```

A convenience annotation that is itself annotated with `@Controller` and `@ResponseBody`.

Types that carry this annotation are treated as controllers where `@RequestMapping` methods assume `@ResponseBody` semantics by default.

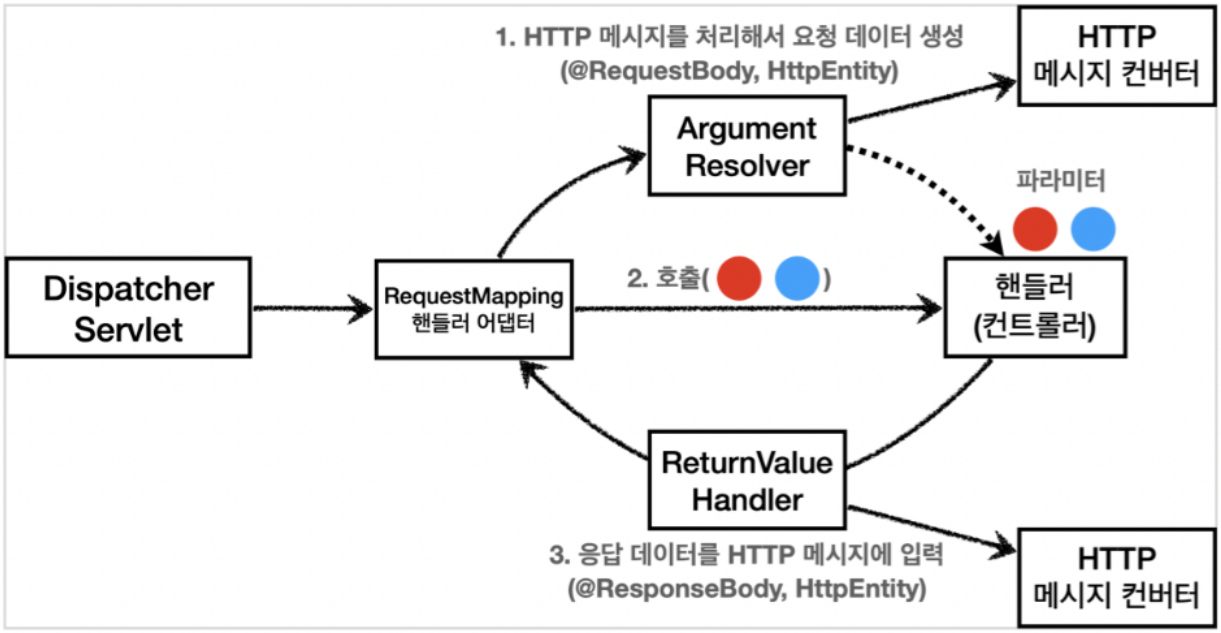
NOTE: `@RestController` is processed if an appropriate `HandlerMapping-HandlerAdapter` pair is configured such as the `RequestMappingHandlerMapping-RequestMappingHandlerAdapter` pair which are the default in the MVC Java config and the MVC namespace.

Since: 4.0

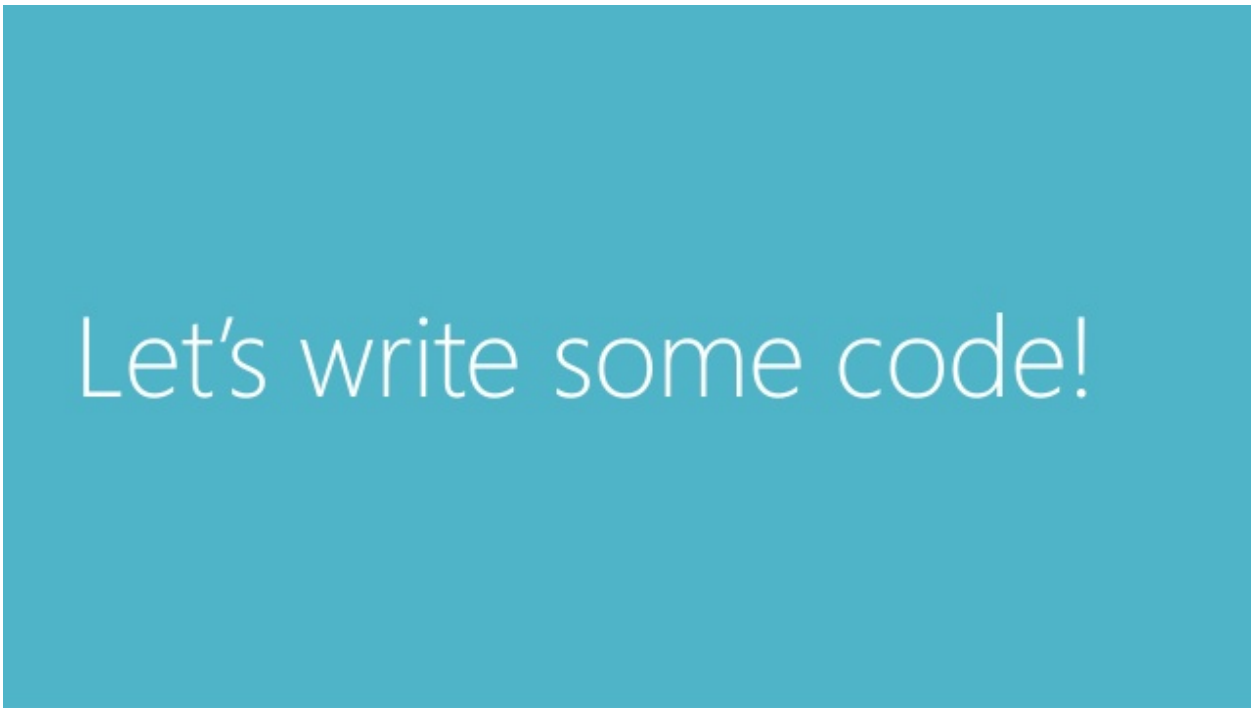
Author: Rossen Stoyanchev, Sam Brannen

```
@Target(ElementType.TYPE)
@Retention(RetentionPolicy.RUNTIME)
@Documented
@Controller
@ResponseBody
public @interface RestController {
```

HTTP 메시지 컨버터 위치



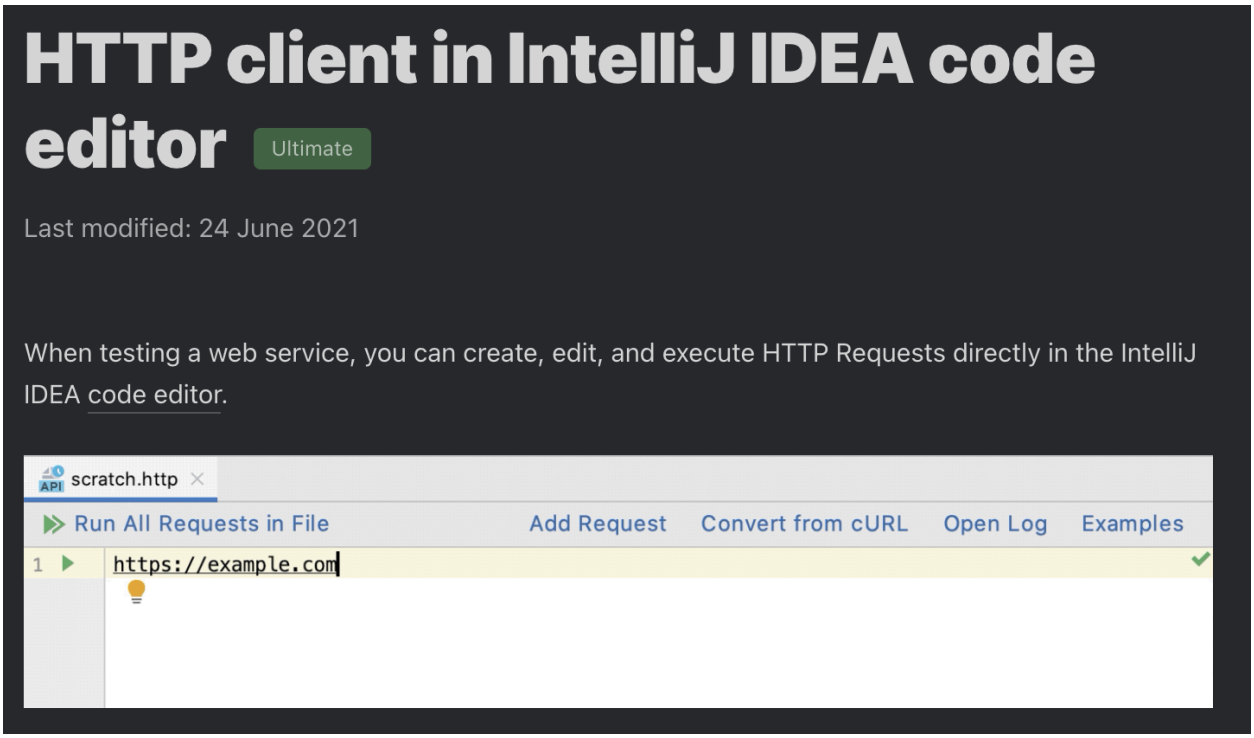
https://bepoz-study-diary.tistory.com/374




```
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-oxm</artifactId>
</dependency>
<dependency>
  <groupId>com.thoughtworks.xstream</groupId>
  <artifactId>xstream</artifactId>
  <version>1.4.17</version>
</dependency>
```

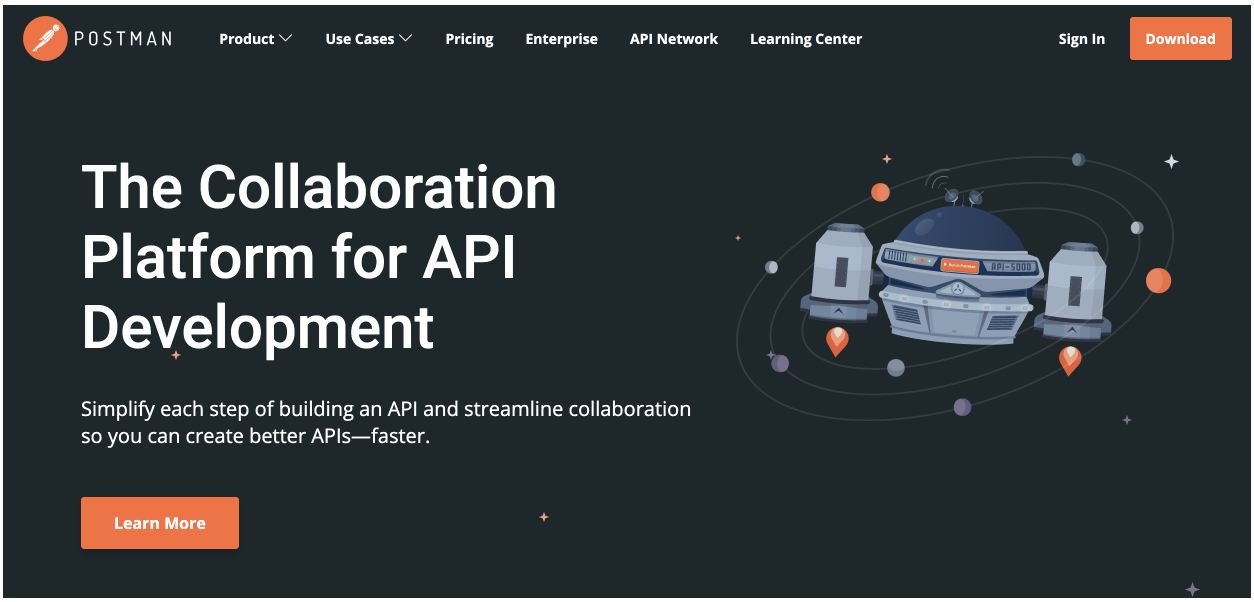
Http Client 도구 소개

IntelliJ HTTP Client



https://www.jetbrains.com/help/idea/http-client-in-product-code-editor.html

Postman



https://www.postman.com/