스프링 MVC

Spring 개요

- 기업용 Application을 쉽게 개발할 수 있도록 도와주는 프레임워크
  - 빠른 구현 시간, 관리 용이성 증가, 개발자의 역량 획일화, 검증된 아키텍처의 재사용과 일관성 유지

- Spring Boot
  - 최소한의 설정만으로도 Spring 기반 Application 개발
  - XML이나 자바 기반 설정을 이용한 Bean 정의나 Servlet 설정을 하지 않아도 동작

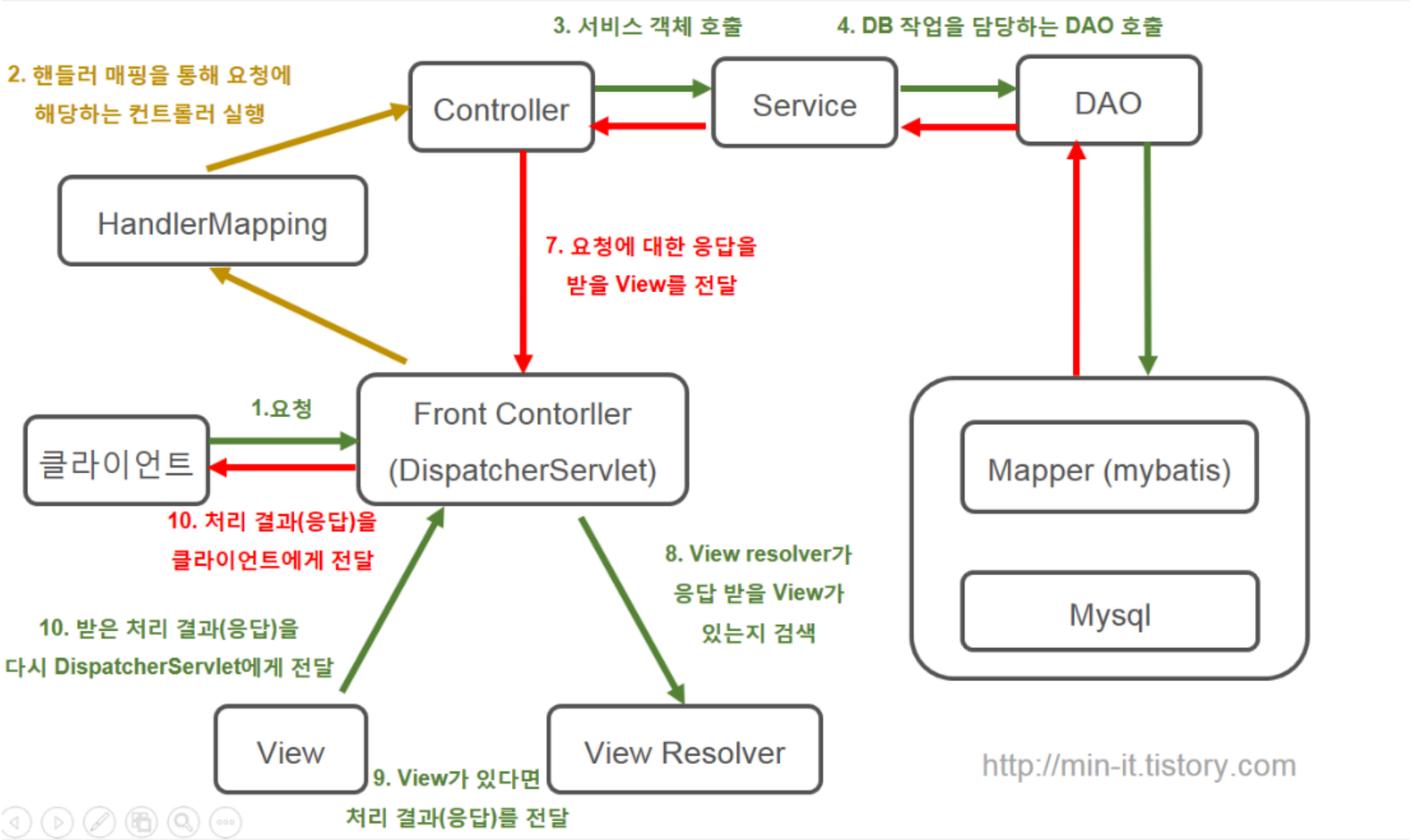
Spring MVC
웹 개발 모델

Spring MVC

- Spring 프레임워크에서 제공하는 웹 모듈
- 백엔드 프로그래밍의 기본 프레임워크
- Web 서버에 특화되어 만들어진 모듈로 개발자가 해야할 영역을 명확히 구분함
- MVC 는 Model-View-Controller 의 약자로, 기본 시스템 모듈을 MVC 로 나누어 구현

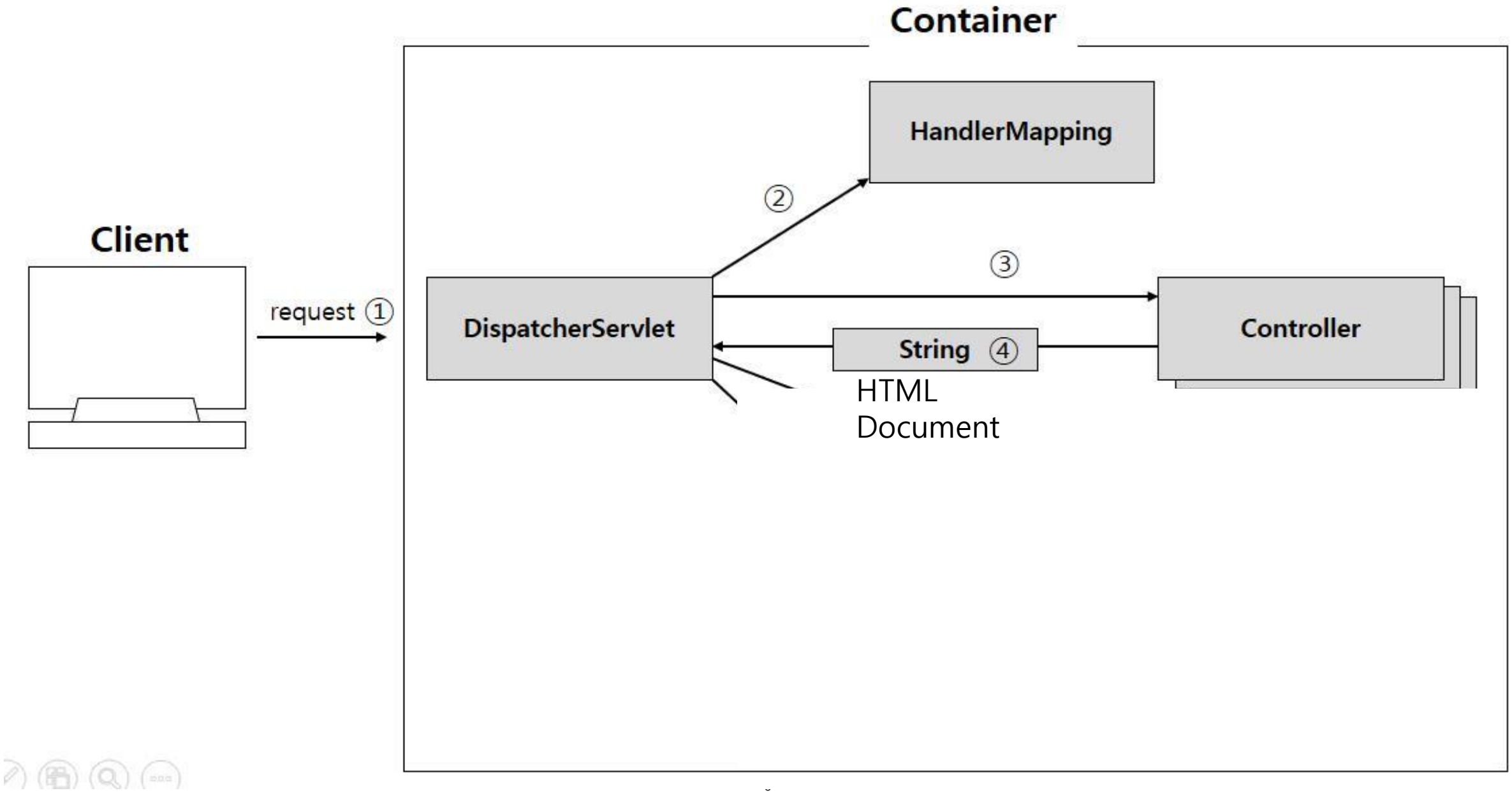
기 능	구성 요소	개발 주체
Model	데이터베이스 클래스 관련	자바 개발자(Database관련)
View	HTML 페이지	웹 디자이너
Controller	브라우저 요청 처리	자바 개발자

MVC Framework 구조



(Spring MVC DispatcherServlet) 출처: https://min-it.tistory.com/7 [MIN-IT]

MVC Framework 구조 / Controller



@Controller / @ReqeustMapping

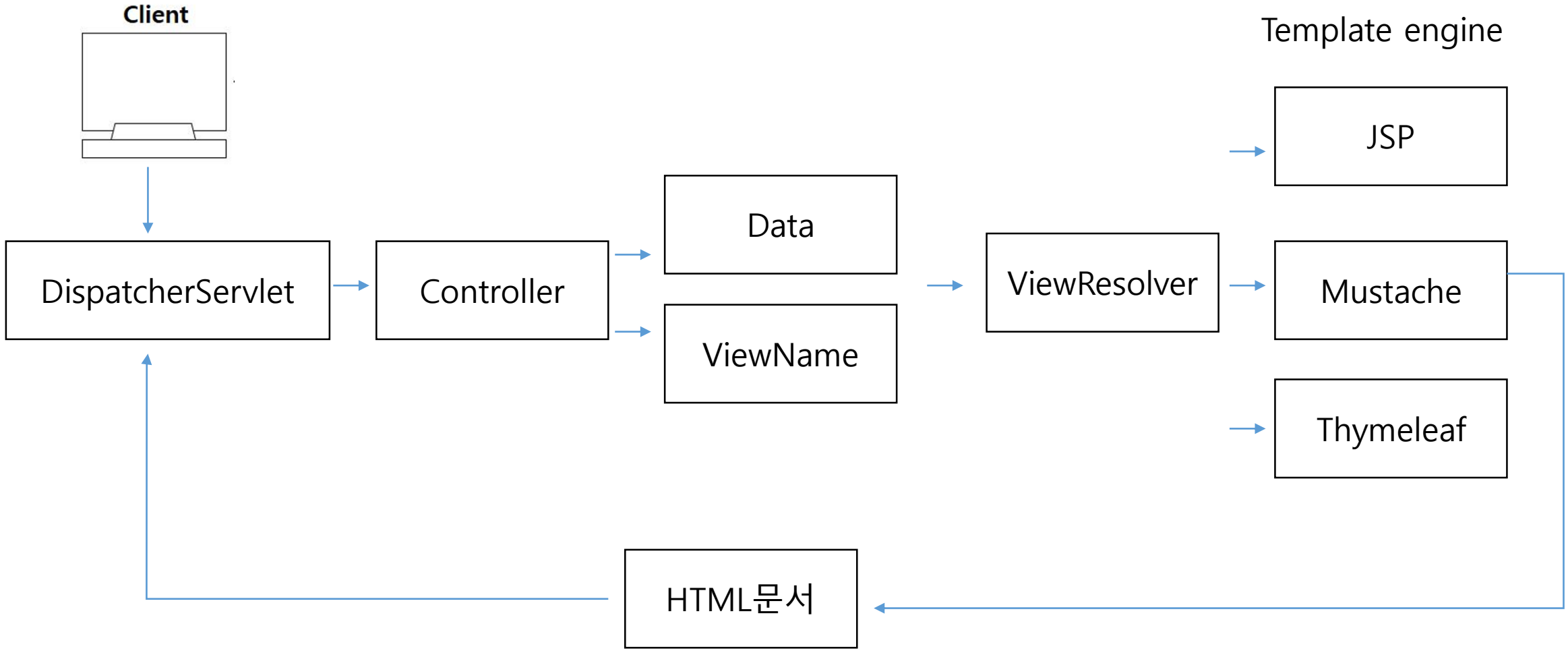
- @Controller가 붙은 클래스를 메모리에 생성하고 Controller 객체로 인식
- 클라이언트의 요청 path에 대해 실행될 메소드 매핑
- HTML 문서 리턴

```
@Controller
public class InsertBoardController {

        @RequestMapping(value="/insertBoard.do")
        @ResponseBody
        public string insertBoard() {

        }
}
```

MVC / View


Client
Template engine
JSP
Data
DispatcherServlet
Controller
ViewResolver
Mustache
ViewName
Thymeleaf
HTML문서

View Resolver / Template Engine

- 출력할 데이터를 준비하는 일은 컨트롤러가 담당
- ViewResolver는 이름과 확장자를 보고 실행할 뷰 컴포넌트를 찾는 일을 담당
- JSP, Thymeleaf 같은 뷰 컴포넌트로가 출력 역할 담당
- 서버-사이드 Template Engine
  - 서버에서 클라이언트에게 응답할 브라우저 화면을 만들어주는 역할

프로젝트 생성

- https://start.spring.io/

Project
○ Gradle Project ● Maven Project

Language
● Java ○ Kotlin ○ Groovy

Spring Boot
○ 3.0.0 (SNAPSHOT) ○ 3.0.0 (RC1) ○ 2.7.6 (SNAPSHOT) ● 2.7.5
○ 2.6.14 (SNAPSHOT) ○ 2.6.13

Project Metadata

Group	com.example
Artifact	hellospring
Name	hellospring
Description	Demo project for Spring Boot
Package name	com.example.hellospring
Packaging	● Jar ○ War
Java	○ 19 ○ 17 ● 11 ○ 8

Dependencies ADD DEPENDENCIES... CTRL + B

Spring Web WEB
Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

Lombok DEVELOPER TOOLS
Java annotation library which helps to reduce boilerplate code.

Thymeleaf TEMPLATE ENGINES
A modern server-side Java template engine for both web and standalone environments. Allows HTML to be correctly displayed in browsers and as static prototypes.

Thymeleaf

개요

- View Template
- 컨트롤러가 전달하는 데이터를 이용하여 동적으로 HTML 생성
- html태그를 기반으로하여 th:속성을 이용
- HTML로 진입장벽이 낮고 쉽게 배울 수 있다는 장점
- HTML에서 namespace 정의
    - <html lang="en" xmlns:th="http://www.thymeleaf.org">
- 서버에서 HTML을 동적으로 렌더링
- 순수한 HTML을 최대한 유지하려는 특징

기본 문법

간단한 표현

- 변수 표현식: `${...}`
- 선택 변수 표현식: `*{...}`
- 메세지 표현식: `#{...}`
- 링크 URL 표현식: `@{...}`
- 조각 표현식: `~{...}`

리터럴

- 텍스트: 'one text', 'Anothr one!', ...
- 숫자: 0, 34, 3.0, 12.3, ...
- 불린: `true`, `false`
- 널: `null`
- 리터럴 토큰: one, sometext, main, ...

문자 연산

- 문자 합치기: `+`
- 리터럴 대체: `|` The name is ${name} `|`

산술 연산

- Binary operators: `+`, `-`, `*`, `/`, `%`
- Minus sign (unary operatior): `-`

불린 연산

- Binary operators: `and`, `or`
- Boolean negation (unary operator): `!`, `not`

비교와 동등

- 비교: `>`, `<`, `>=`, `<=` (`gt, lt, ge, le`)
- 동등 연산: `==`, `!=` (`eq, ne`)

조건 연산

- If-then: `(if) ? (then)`
- If-then-else: `(if) ? (then) : (else)`
- Default: `(value) ?: (defaultvalue)`

https://catsbi.oopy.io/32a9458e-f452-4733-b87c-caba75f98e2d

기본 문법

- **문자 출력**

```
• <p th:text="${name}" />
• [[ ${name} ]]
• <th:block th:text="${name}" />
• <p th:utext="${html}">
• [( $(html) )]
```

- **객체 속성 출력**

```
• <p th:text="${book.title}" />
• title = [[ ${book.title} ]] <br/>
```

- **이미지, A**

```
• <img th:src="${image}" width="32">
• <img th:src="'/img/' + ${image}" width="32">
• <a th:href="|read?id=${id}|">go~~</a>
```

비교, 제어, 반복

비교연산자

```
<div th:text="${book.price >= 30}"></div>
<div th:text="${book.author == 'kim' ? 'ok' : 'no'}"></div>
```

if, else, switch

```
<p th:if="${age > 5}">age가 5보다 크다</p>
<p th:unless="${age > 5}">age는 5보다 작다</p>
<th:block th:switch="${name}">
<div th:case="김"> 당신은 김씨 입니다.</div>
```

반복

```
<th:block th:each="data:${datas}">
<th:block th:each="data,status:${datas}">
```

Fragment 정의

```
<div th:fragment="footer">
    <p> COPYRIGHT@ me</p>
</div>
```

Fragment 포함시키기

```
<th:block th:replace="~{header :: header}" />   // 파일명 :: 이름
<div th:insert="~{footer :: footer}"></div>
```