

Spring Framework

19. Spring MVC 개요

CONTENTS

1

MVC 패턴의 개념과 모델2 아키텍처

2

Spring MVC 개념

3

Spring MVC 기반 웹어플리케이션 개발

학습 목표

- MVC 패턴의 개념과 모델2 아키텍처에 대하여 이해할 수 있습니다.
- Spring MVC 개념에 대하여 이해할 수 있습니다.
- Spring MVC 기반 웹 어플리케이션 개발에 대해 이해할 수 있습니다.

AOP는

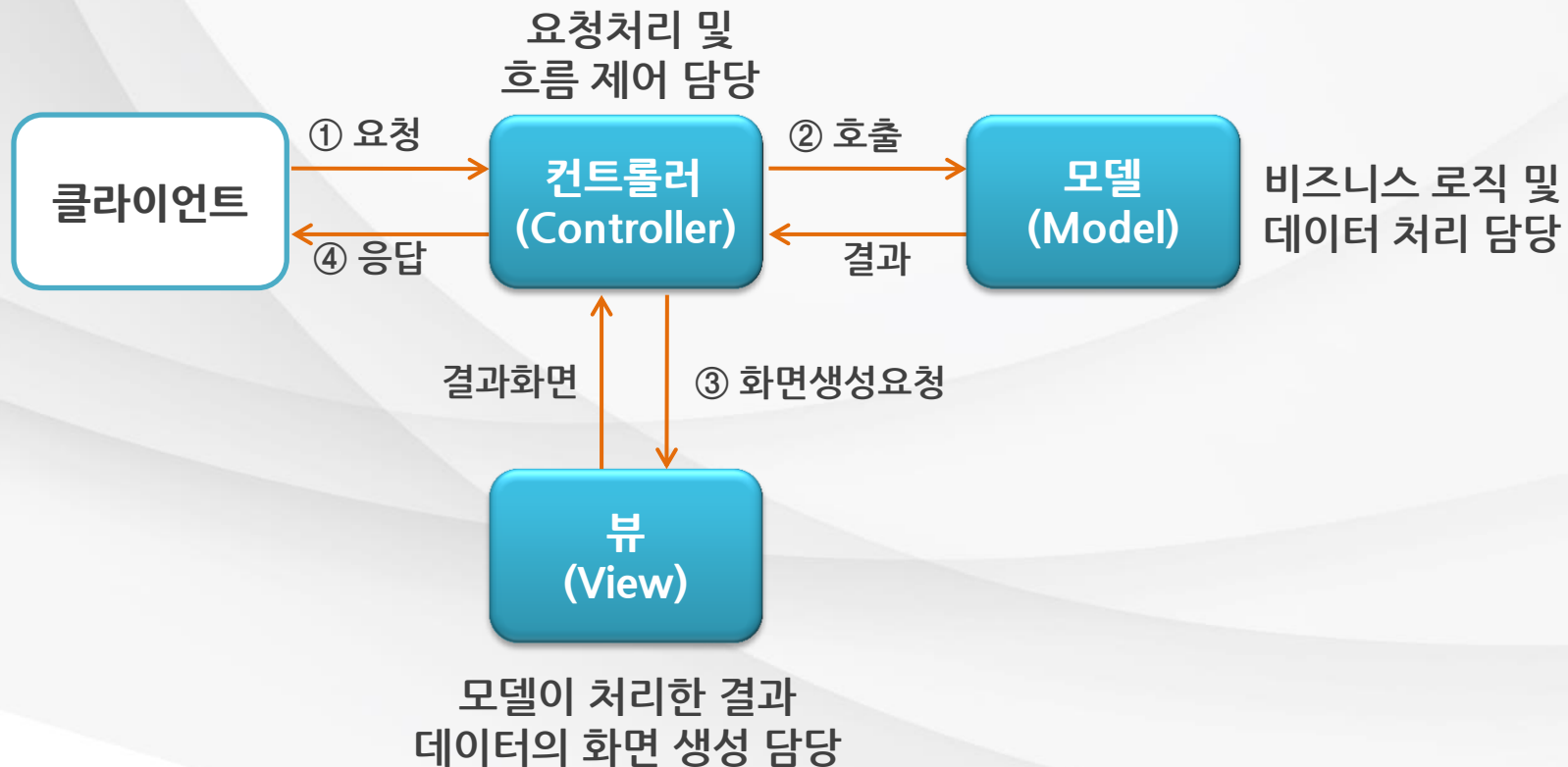
1. MVC 패턴의 개념과 모델2 아키텍처

■ MVC(Model-View-Controller) 패턴의 개념

모델-뷰-컨트롤러(Model-View-Controller, MVC)는 소프트웨어 공학에서 사용되는 **아키텍처 패턴**으로 MVC 패턴의 주 목적은 **Business logic**과 **Presentation logic**을 분리하기 위함이다.

- MVC 패턴을 사용하면, 사용자 인터페이스로부터 비즈니스 로직을 분리하여 애플리케이션의 시각적 요소나 그 이면에서 실행되는 비즈니스 로직을 서로 영향 없이 쉽게 고칠 수 있는 애플리케이션을 만들 수 있음
 - **Model** : 애플리케이션의 정보(데이터, Business Logic 포함)
 - **View** : 사용자에게 제공할 화면(Presentation Logic)
 - **Controller** : Model과 View 사이의 상호 작용을 관리

■ MVC(Model-View-Controller) 패턴의 개념



I 각각의 MVC 컴포넌트의 역할

❖ 모델(Model) 컴포넌트

데이터 저장소(ex : 데이터베이스 등)와 연동하여 사용자가 입력한 데이터나 사용자에게 출력할 데이터를 다루는 일을 함

여러 개의 데이터 변경 작업(추가, 변경, 삭제)을 하나의 작업으로 묶는 트랜잭션을 다루는 일도 함

DAO 클래스, Service 클래스에 해당

I 각각의 MVC 컴포넌트의 역할

❖ 뷰(View) 컴포넌트

모델이 처리한 데이터나 그 작업 결과를 가지고 사용자에게 출력할 화면을 만드는 일을 함

생성된 화면은 웹 브라우저가 출력하고, 뷰 컴포넌트는 HTML과 CSS, Java Script를 사용하여 웹 브라우저가 출력할 UI를 만듦

Html과 JSP를 사용하여 작성할 수 있음

I 각각의 MVC 컴포넌트의 역할

❖ 컨트롤러(Controller) 컴포넌트

클라이언트의 요청을 받았을 때 그 요청에 대해 실제 업무를 수행하는 모델 컴포넌트를 호출하는 일을 함

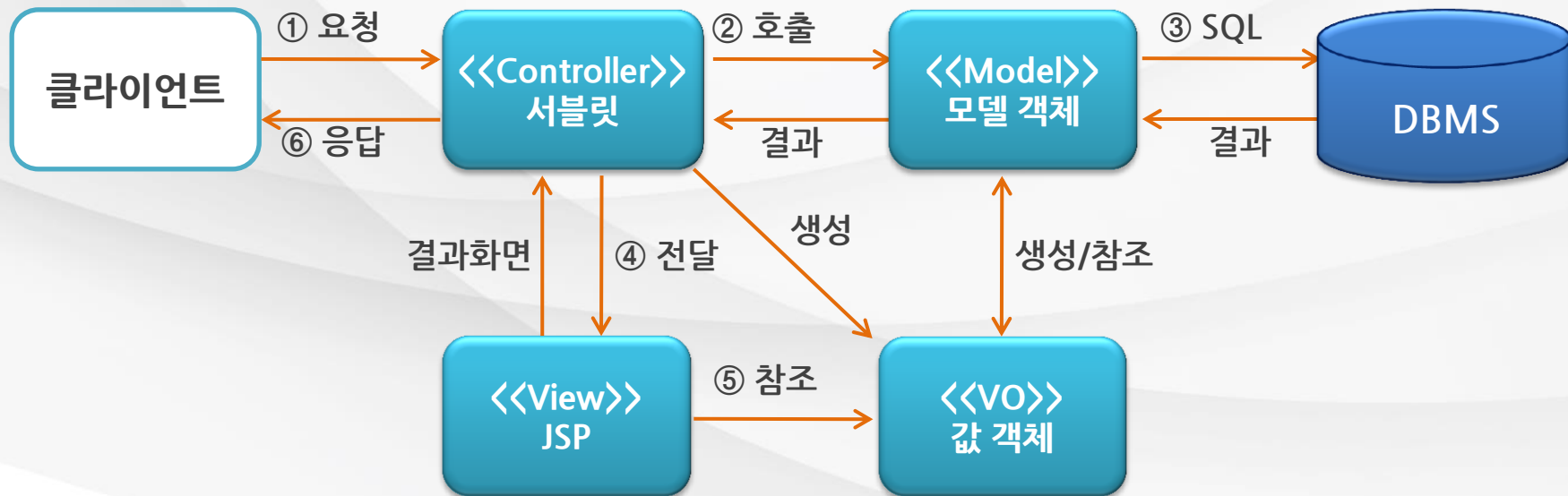
클라이언트가 보낸 데이터가 있다면, 모델을 호출할 때 전달하기 쉽게 데이터를 적절히 가공하는 일을 함

모델이 업무 수행을 완료하면, 그 결과를 가지고 화면을 생성하도록 뷰에게 전달(클라이언트 요청에 대해 모델과 뷰를 결정하여 전달)

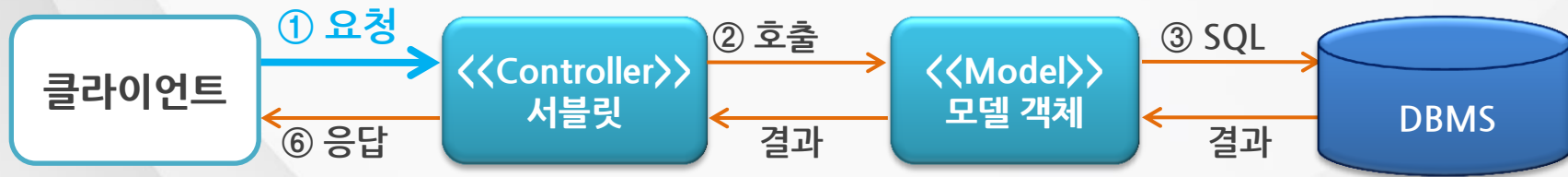
Servlet과 JSP를 사용하여 작성할 수 있음

■ 모델2 아키텍처 개념

- 모델 1 아키텍처 : Controller 역할을 JSP가 담당함
- 모델 2 아키텍처 : Controller 역할을 Servlet이 담당함



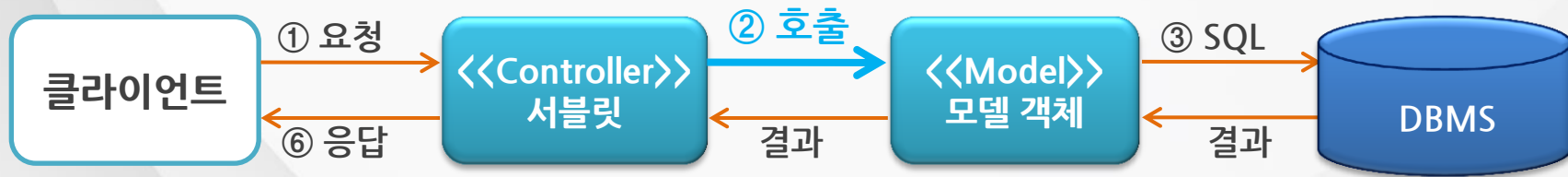
■ 모델2 아키텍처 호출 순서



① 웹 브라우저가 웹 애플리케이션 실행을 요청하면, 웹 서버가 그 요청을 받아서 서블릿 컨테이너(ex : 톰캣서버)에 넘겨준다.

서블릿 컨테이너는 URL을 확인하여 그 요청을 처리할 서블릿을 찾아서 실행한다.

■ 모델2 아키텍처 호출 순서

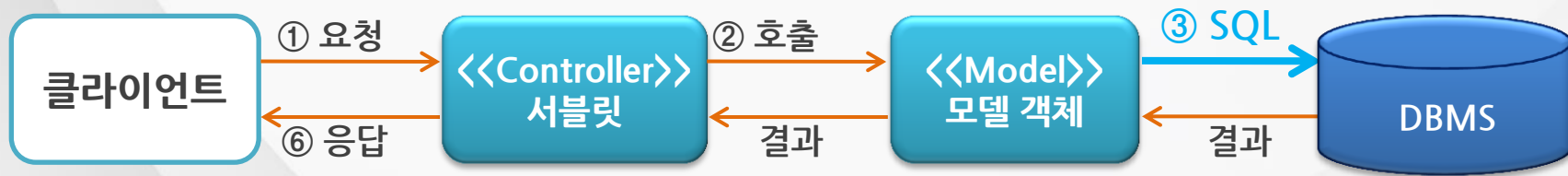


② 서블릿은 실제 업무를 처리하는 모델 자바 객체의 메서드를 호출한다.

만약 웹 브라우저가 보낸 데이터를 저장하거나 변경해야 한다면 그 데이터를 가공하여 VO 객체(Value Object)를 생성하고, 모델 객체의 메서드를 호출할 때 인자 값으로 넘긴다.

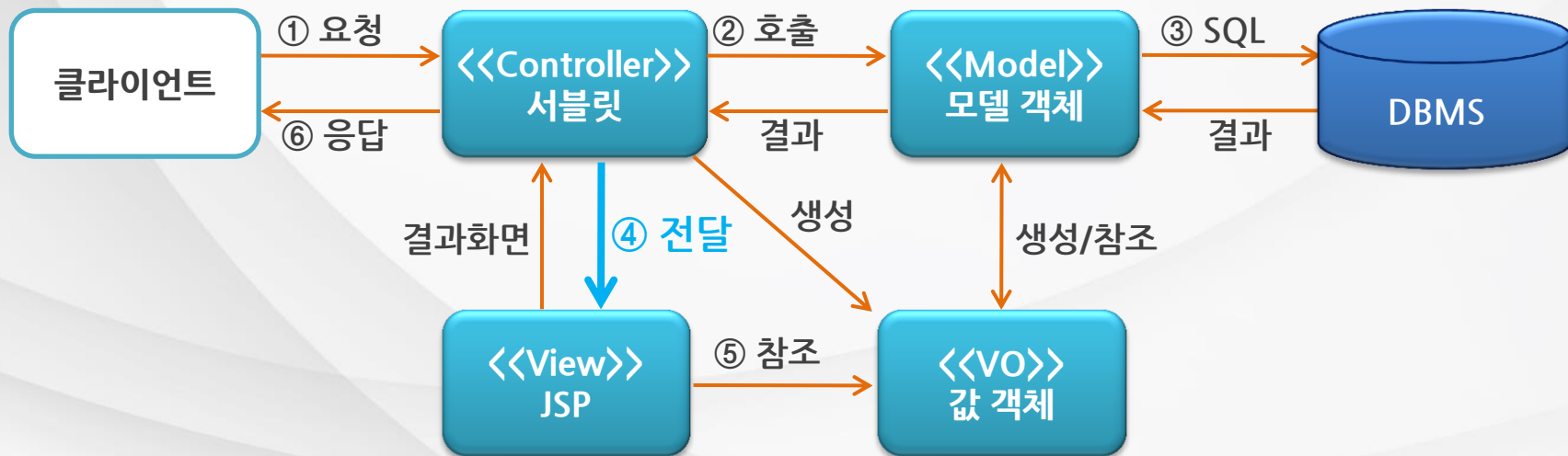
모델 객체는 엔터프라이즈 자바빈(EJB; Enterprise JavaBeans)일 수도 있고, 일반 자바 객체(POJO로 된 Service, DAO object) 일 수도 있다.

■ 모델2 아키텍처 호출 순서



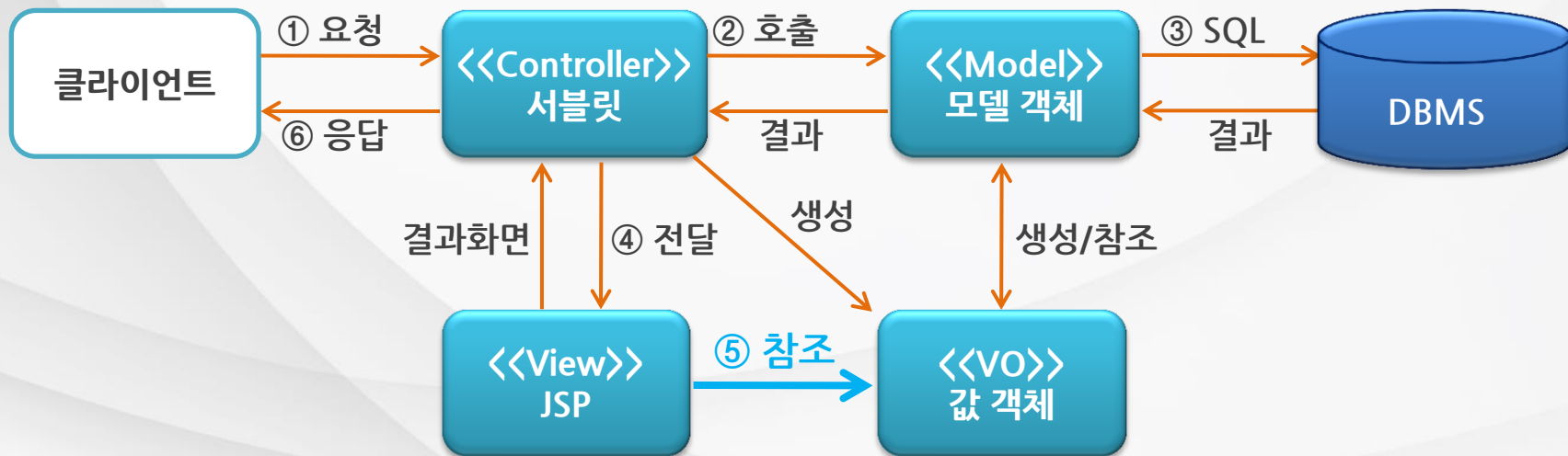
③ 모델 객체는 JDBC를 사용하여 매개변수로 넘어온 값 객체를 데이터베이스에 저장하거나, 데이터베이스로부터 질의 결과를 가져와서 VO 객체로 만들어 반환한다.

■ 모델2 아키텍처 호출 순서



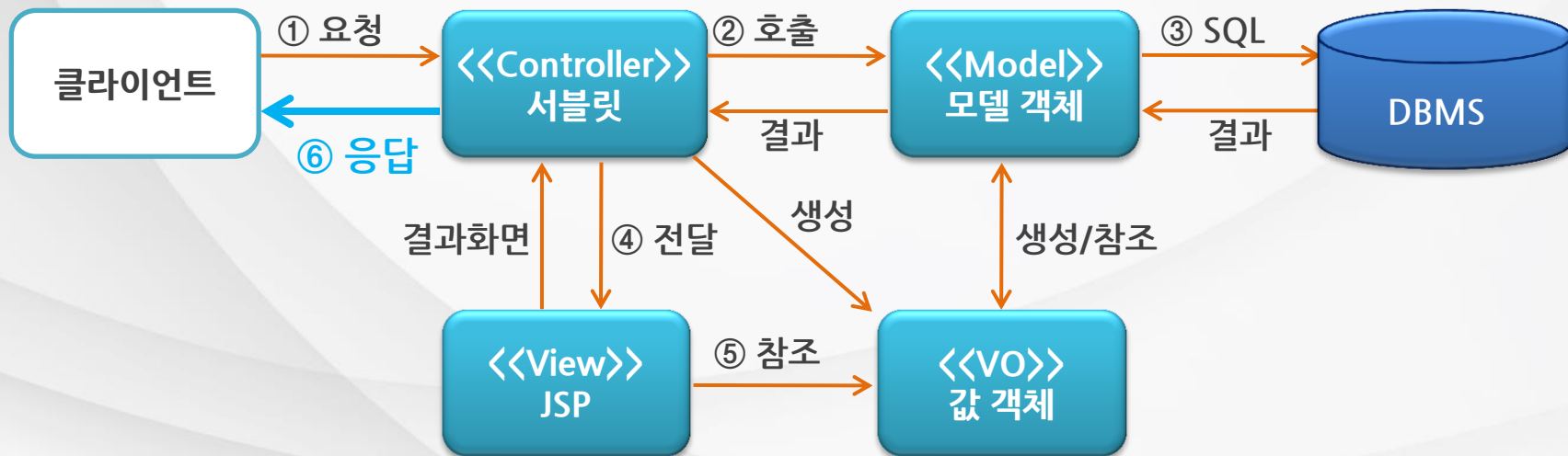
④ 서블릿은 모델 객체로부터 반환 받은 값을 JSP에 전달한다.

■ 모델2 아키텍처 호출 순서



⑤ JSP는 서블릿으로 부터 전달받은 값 객체를 참조하여 웹 브라우저가 출력할 결과 화면을 만들고, 웹 브라우저에 출력함으로써 요청 처리를 완료한다.

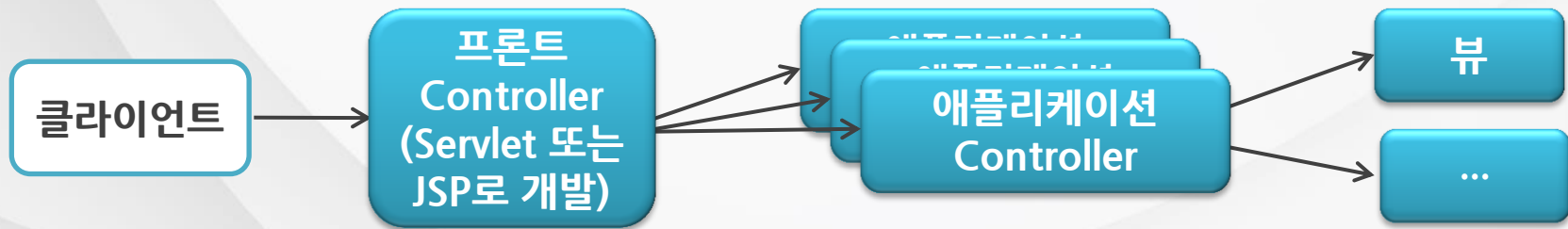
■ 모델2 아키텍처 호출 순서



⑥ 웹 브라우저는 서버로부터 받은 응답 내용을 화면에 출력한다.

■ Front Controller 패턴 아키텍처

❖ Front Controller 프로세스



- Front Controller는 클라이언트가 보낸 요청을 받아서 공통적인 작업을 먼저 수행
- Front Controller는 적절한 세부 Controller에게 작업을 위임
- 각각의 애플리케이션 Controller는 클라이언트에게 보낼 뷰를 선택해서 최종 결과를 생성하는 작업
- Front Controller 패턴은 인증이나 권한 체크처럼 모든 요청에 대하여 공통적으로 처리해야 하는 로직이 있을 경우 전체적으로 클라이언트의 요청을 중앙 집중적으로 관리하고자 할 경우에 사용

A person's hands are shown holding a smartphone, with the screen glowing. The background is dark with out-of-focus, colorful bokeh lights in shades of yellow, orange, and blue. A semi-transparent dark banner is at the bottom, containing a yellow decorative element and the title text.

2. Spring MVC 개념

■ Spring MVC의 특징

Spring은 DI 나 AOP 같은 기능뿐만 아니라,
서블릿 기반의 웹 개발을 위한 MVC 프레임워크를 제공

Spring MVC는 모델2 아키텍처와 Front Controller 패턴을
프레임워크 차원에서 제공

Spring MVC 프레임워크는 Spring을 기반으로 하고 있기 때문에
Spring이 제공하는 트랜잭션 처리나 DI 및 AOP등을 손쉽게 사용

■ Spring MVC와 Front Controller 패턴

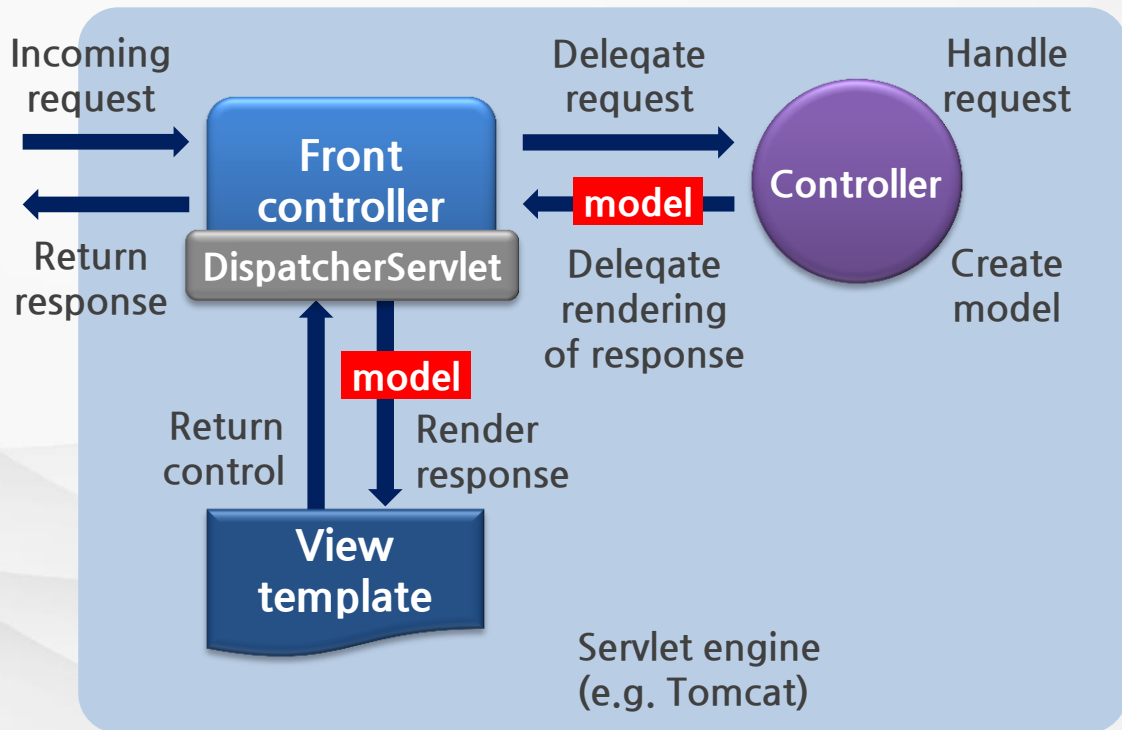
대부분의 MVC 프레임워크들은 Front Controller 패턴을 적용해서 구현

Spring MVC도 Front Controller 역할을 하는 DispatcherServlet이라는 클래스를 계층의 맨 앞단에 놓고, 서버로 들어오는 모든 요청을 받아서 처리하도록 구성

예외가 발생했을 때 일관된 방식으로 처리하는 것도 Front Controller의 역할

■ DispatcherServlet 클래스

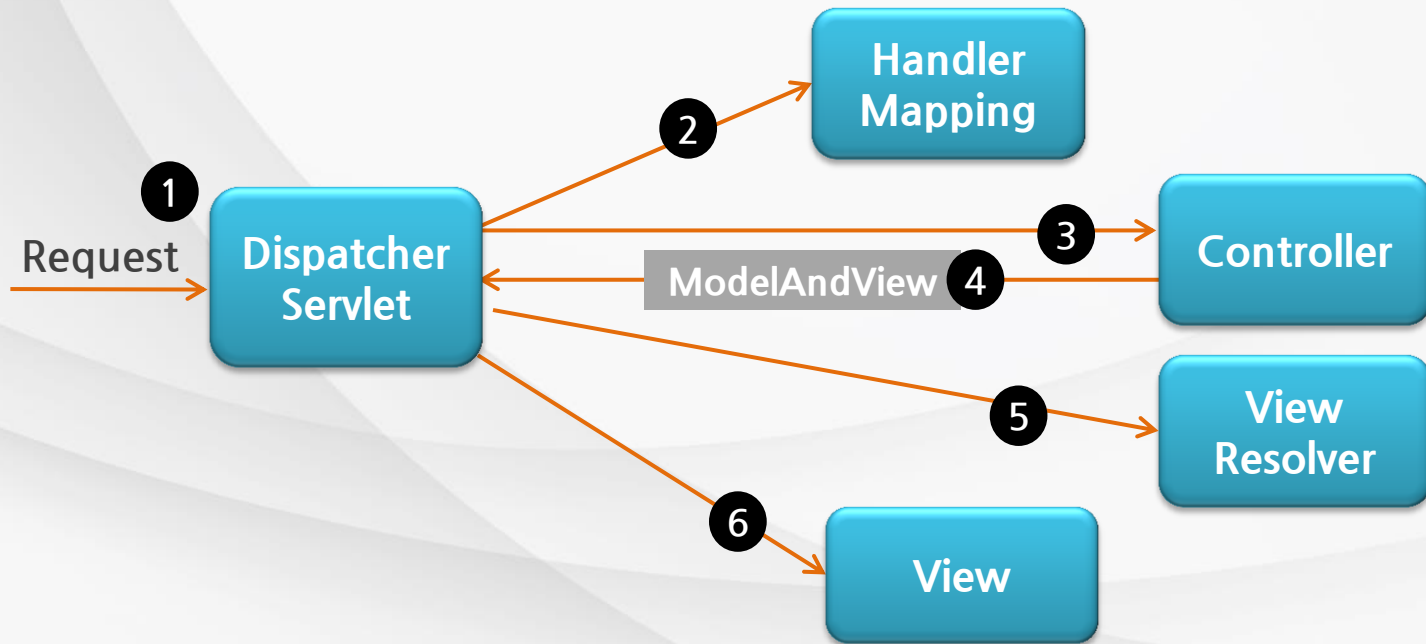
- ◉ Front Controller 패턴 적용
- ◉ web.xml에 설정
- ◉ 클라이언트로부터의 모든 요청을 전달 받음
- ◉ Controller나 View와 같은 Spring MVC의 구성요소를 이용하여 클라이언트에게 서비스를 제공



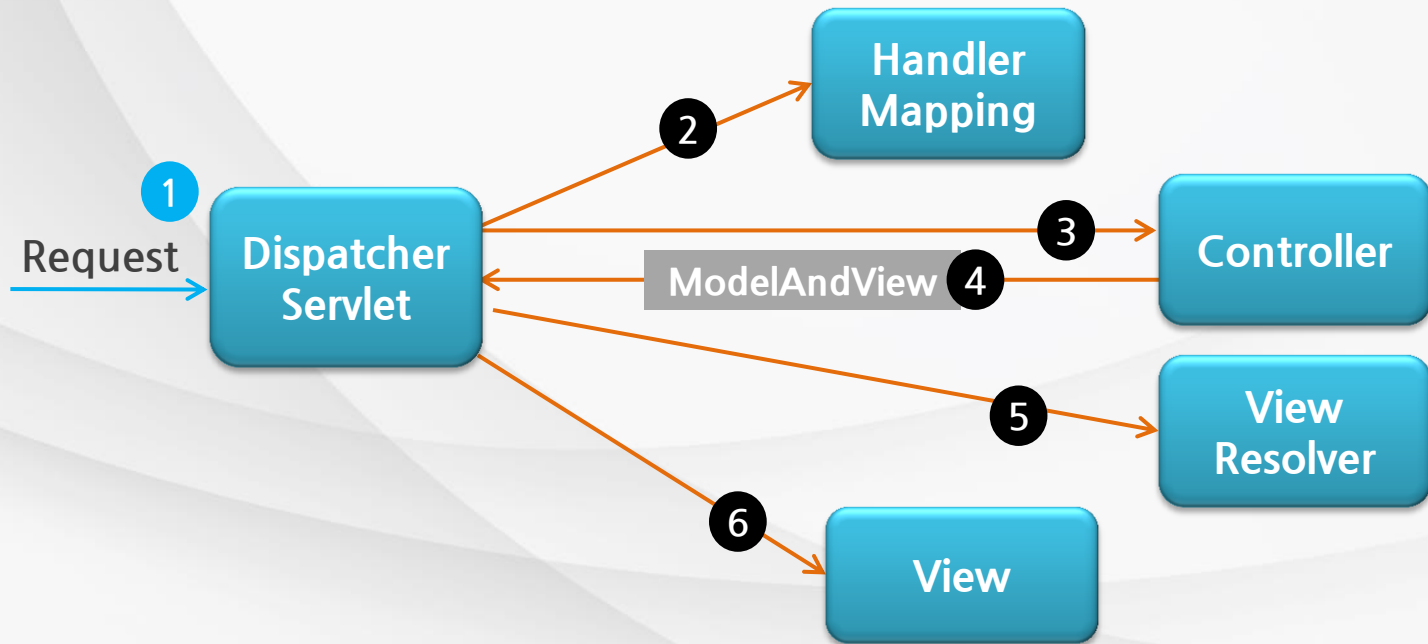
■ Spring MVC의 주요 구성 요소

구성요소	설 명
DispatcherServlet	클라이언트의 요청을 받아서 Controller에게 클라이언트의 요청을 전달하고, 리턴한 결과값을 View에게 전달하여 알맞은 응답을 생성
HandlerMapping	URL과 요청 정보를 기준으로 어떤 핸들러 객체를 사용할지 결정하는 객체이며, DispatcherServlet은 하나 이상의 핸들러 매핑을 가질 수 있음
Controller	클라이언트의 요청을 처리한 뒤, Model를 호출하고 그 결과를 DispatcherServlet에게 알려 줌
ModelAndView	Controller가 처리한 데이터 및 화면에 대한 정보를 보유한 객체
View	Controller의 처리 결과 화면에 대한 정보를 보유한 객체
ViewResolver	Controller가 리턴한 뷰 이름을 기반으로 Controller 처리 결과를 생성할 뷰를 결정

■ Spring MVC의 주요 구성 요소의 요청 처리 과정

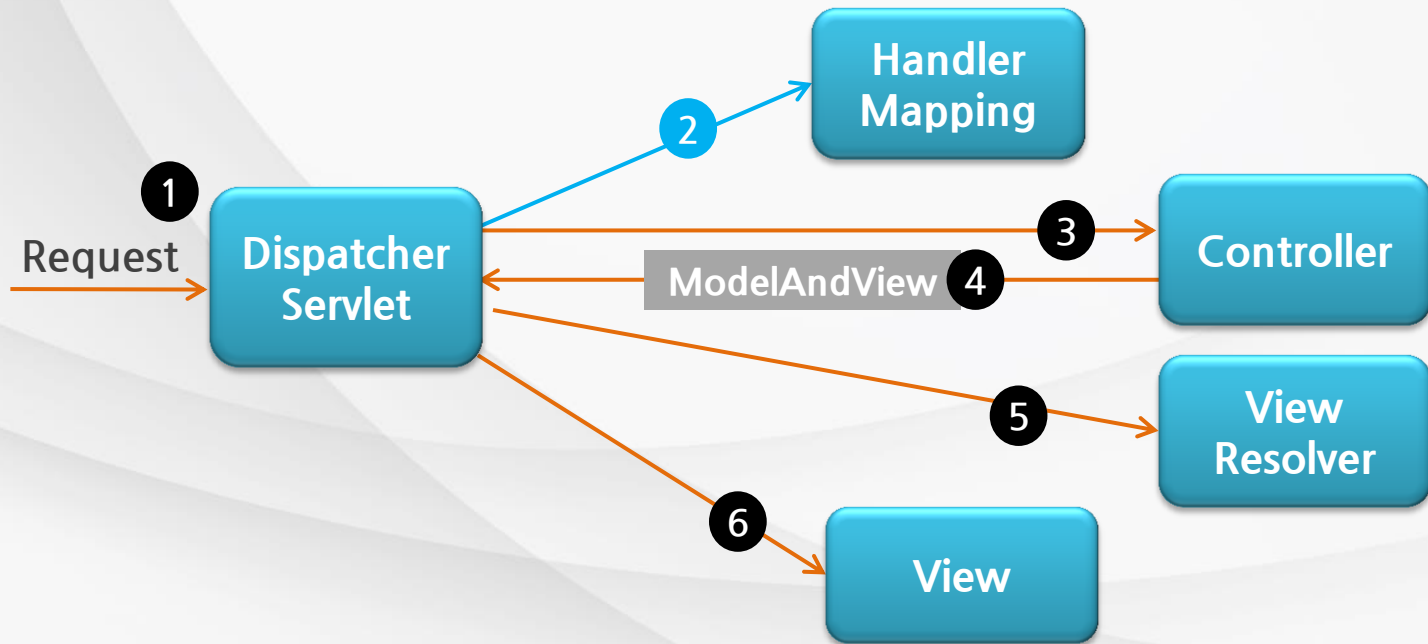


■ Spring MVC의 주요 구성 요소의 요청 처리 과정



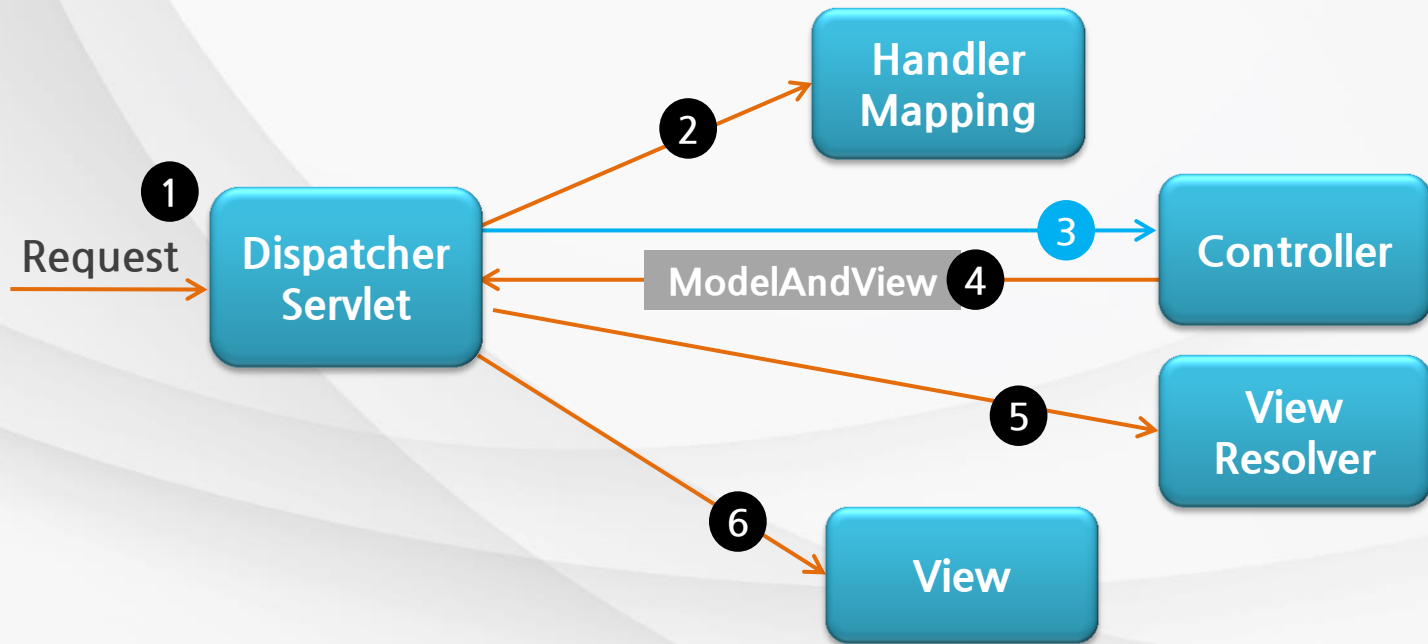
① 클라이언트의 요청이 DispatcherServlet에게 전달된다.

■ Spring MVC의 주요 구성 요소의 요청 처리 과정(2)



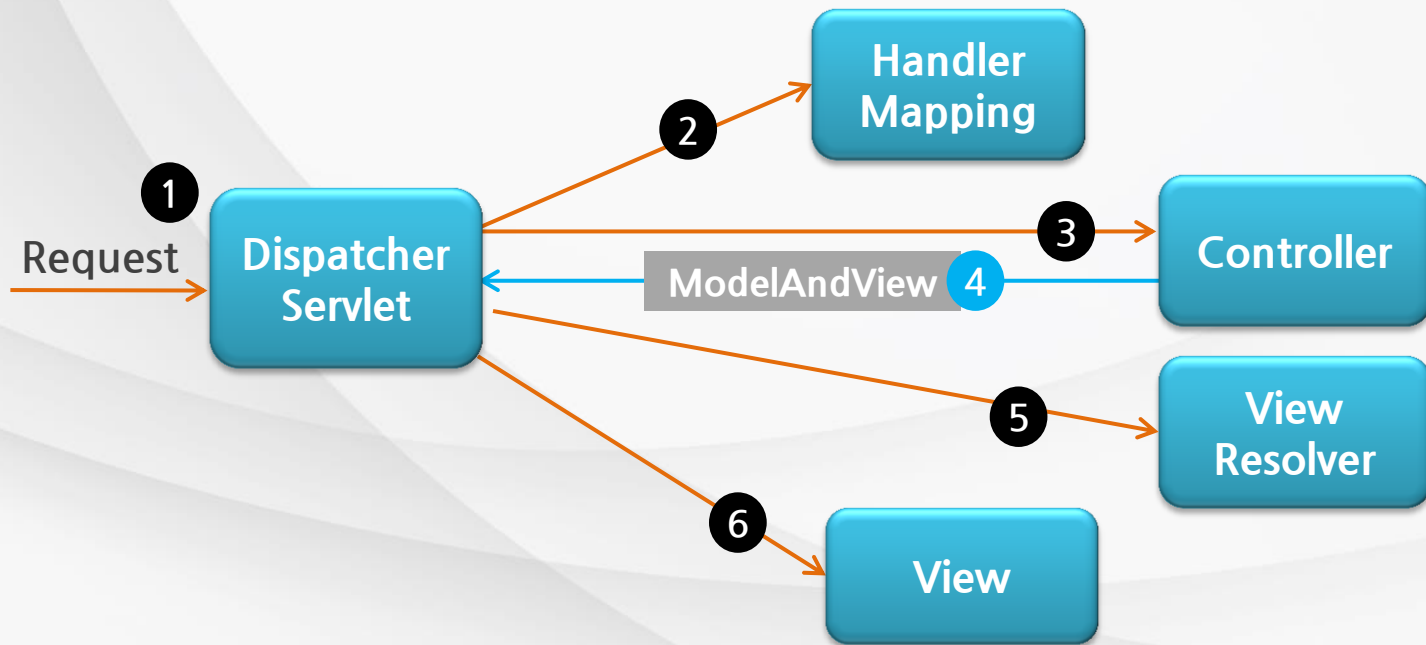
② DispatcherServlet은 HandlerMapping을 사용하여 클라이언트의 요청을 처리할 Controller를 획득한다.

■ Spring MVC의 주요 구성 요소의 요청 처리 과정(2)



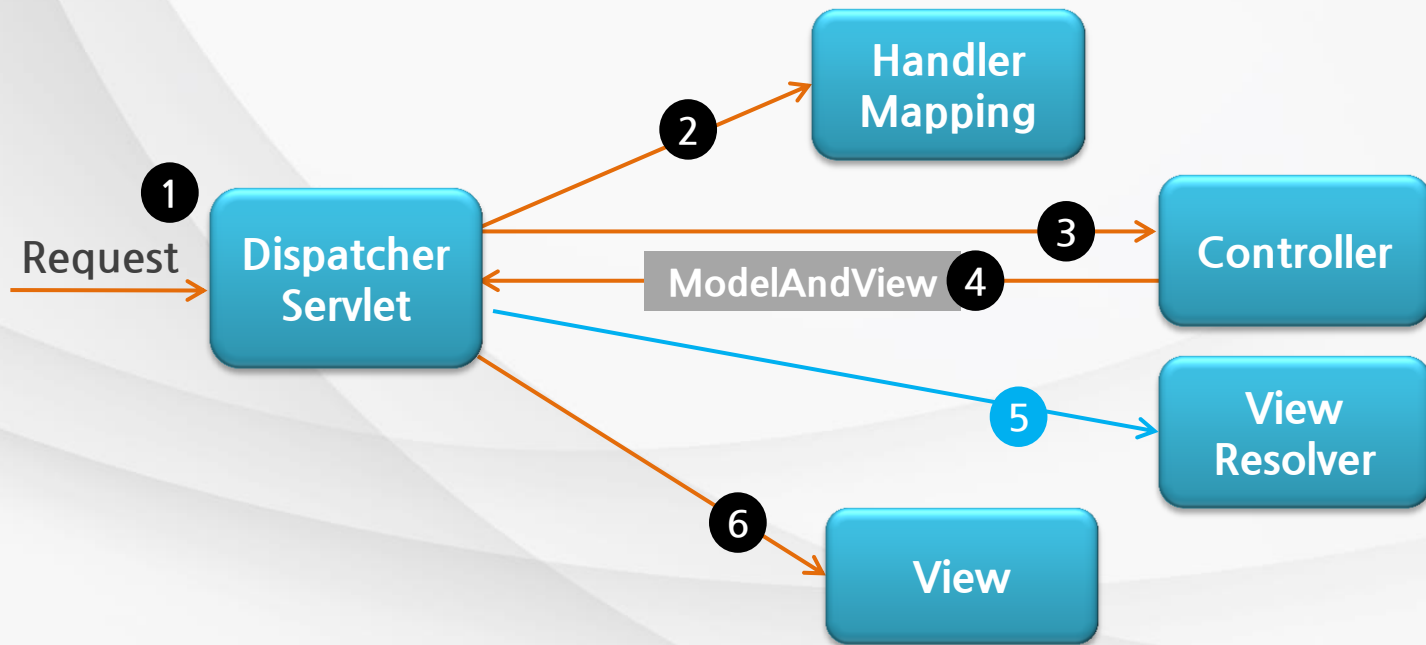
- ③ DispatcherServlet은 Controller 객체를 이용하여 클라이언트의 요청을 처리한다.

■ Spring MVC의 주요 구성 요소의 요청 처리 과정(2)



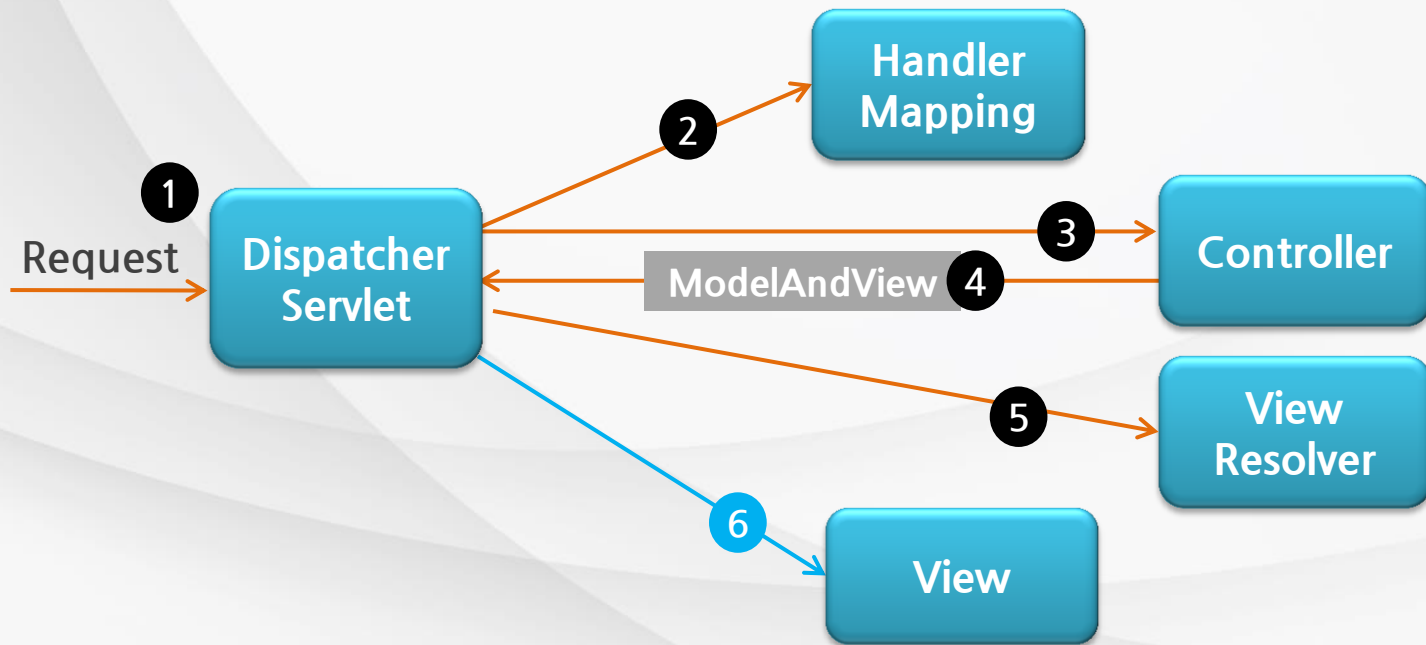
④ Controller는 클라이언트 요청 처리 결과와 View 페이지 정보를 담은 ModelAndView 객체를 반환한다.

■ Spring MVC의 주요 구성 요소의 요청 처리 과정



- ⑤ DispatcherServlet은 ViewResolver로부터 응답 결과를 생성할 View 객체를 구한다.

■ Spring MVC의 주요 구성 요소의 요청 처리 과정



⑥ View는 클라이언트에게 전송할 응답을 생성한다.

A person's hands are shown holding a smartphone, with the screen glowing. The background is dark with out-of-focus, colorful bokeh lights in shades of yellow, orange, and blue. A semi-transparent dark banner with a yellow decorative element on the left is positioned at the bottom.

3. Spring MVC 기반 웹어플리케이션 개발

■ Spring MVC기반 웹 어플리케이션 작성 절차

① 클라이언트의 요청을 받는 DispatcherServlet를 web.xml에 설정

② 클라이언트의 요청을 처리할 Controller를 작성

③ Spring Bean으로 Controller를 등록

④ JSP를 이용한 View 영역의 코드를 작성

⑤ Browser 상에서 JSP를 실행

■ Spring MVC기반 웹 어플리케이션 작성 절차(1)

① DispatcherServlet을 web.xml에 설정

```
*web.xml
35  <!-- The front controller of this Spring Web application, responsible for handling all
36  <servlet>
37      <servlet-name>springDispatcherServlet</servlet-name>
38      <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
39      <init-param>
40          <param-name>contextConfigLocation</param-name>
41          <param-value>classpath:/config/beans*.xml</param-value>
42      </init-param>
43      <load-on-startup>1</load-on-startup>
44  </servlet>
45
46  <!-- Map all requests to the DispatcherServlet for handling -->
47  <servlet-mapping>
48      <servlet-name>springDispatcherServlet</servlet-name>
49      <url-pattern>*.do</url-pattern>
50  </servlet-mapping>
51  </web-app>
```

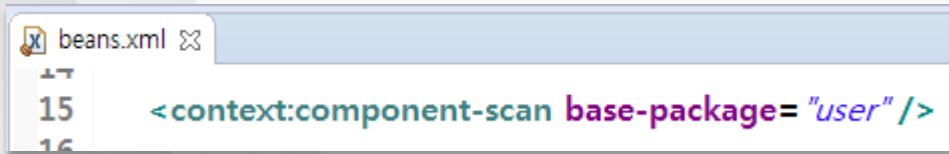

■ Spring MVC기반 웹 어플리케이션 작성 절차(2)

② 클라이언트의 요청을 처리할 Controller

```
*UserController.java ✕  
1  package user.controller;  
2  
3  import java.util.ArrayList;  
21  
22  @Controller  
23  public class UserController {  
24      @Autowired  
25      private UserService userService;  
26  
27      @RequestMapping("/getUser.do")  
28      public ModelAndView getUser(@RequestParam String id) {  
29          UserVO user = userService.getUser(id);  
30          return new ModelAndView("userInfo.jsp","user",user);  
31      }
```

■ Spring MVC기반 웹 어플리케이션 작성 절차(3)

③ Spring Bean으로 Controller를 등록



```
beans.xml
14
15 <context:component-scan base-package="user" />
16
```

■ Spring MVC기반 웹 어플리케이션 작성 절차(4)

④ JSP를 이용한 View 영역의 코드를 작성

```
*userInfo.jsp
3 <html>
4 <head>
5 <meta http-equiv="Content-Type" content="text/html; charset=EUC-KR">
6 <title>사용자 상세정보</title>
7 </head>
8 <body>
9 <div class="container">
10 <h2>사용자 상세정보</h2>
11 <table class="table table-bordered table table-hover">
12 <tr><td>아이디 :</td><td>${user.userId}</td></tr>
13 <tr><td>이름 :</td><td>${user.name}</td></tr>
14 <tr><td>성별 :</td><td>${user.gender}</td></tr>
15 <tr><td>도시명 :</td><td>${user.city}</td></tr>
16 </table>
17 </div>
18 </body>
19 </html>
```

■ Spring MVC기반 웹 어플리케이션 작성 절차(5)

⑤ Browser상에서 JSP를 실행한다.





학습정리

지금까지 [Spring MVC 개요]에 대해서 살펴보았습니다.

MVC 패턴의 개념과 모델2 아키텍처

- ◉ MVC 패턴의 개념 및 각 컴포넌트의 역할
- ◉ 모델2 아키텍처의 개념

Spring MVC 개요

- ◉ Spring MVC의 특징
- ◉ Spring MVC의 주요 구성요소와 요청 처리 과정

Spring MVC 기반 웹어플리케이션 개발

- ◉ Spring MVC 기반 웹 어플리케이션 작성 절차