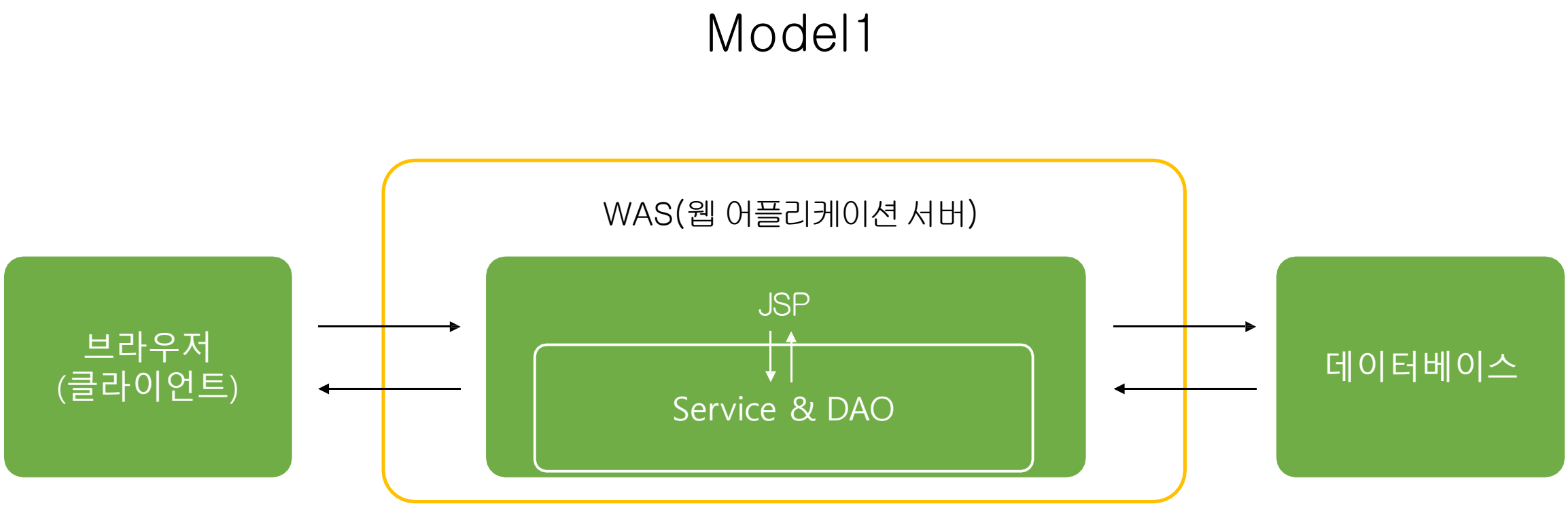
Spring Framework

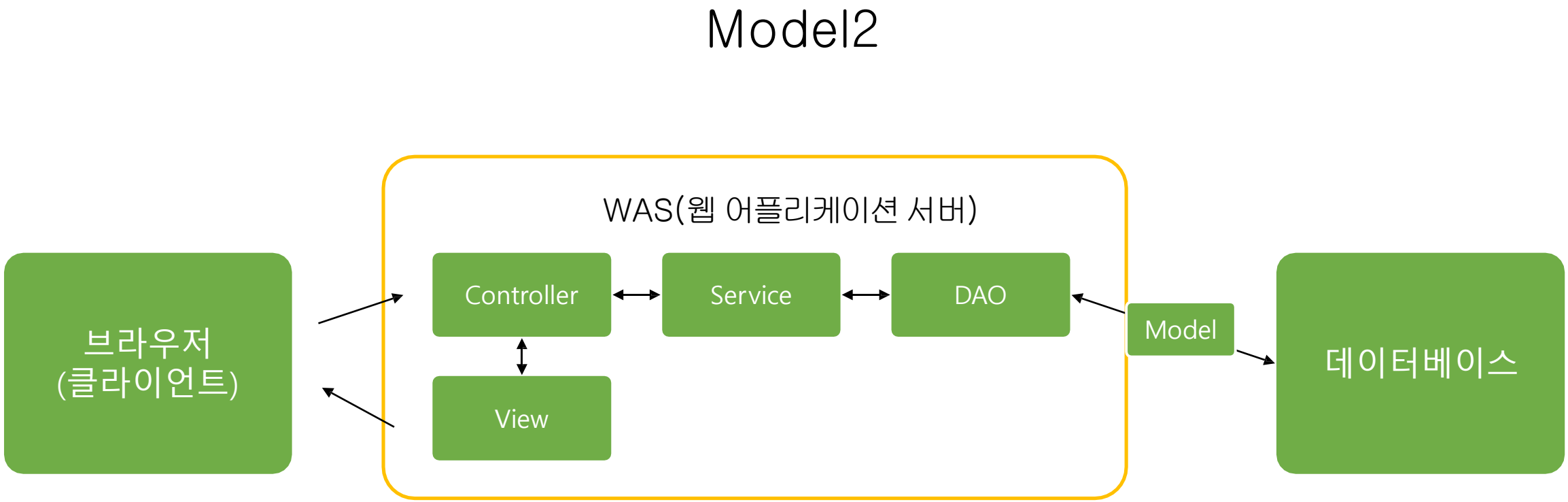
웹 프로그램 설계모델

1. 웹프로그래밍 설계모델
2. 스프링 프레임워크 설계모델
3. DispatcherServlet 설정
4. Controller - @Controller
5. Controller - @RequestMapping
6. Controller - Model 타입 파라미터
7. View객체
8. 전체 웹프로그램 구조

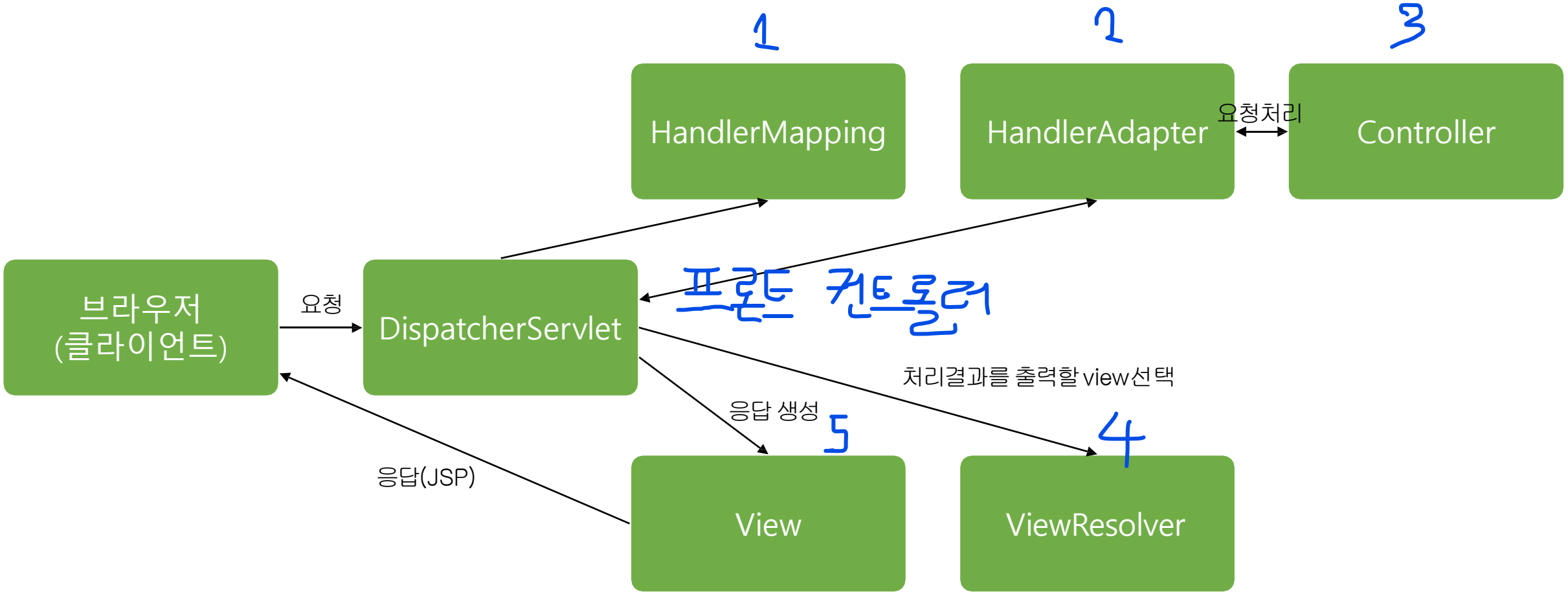
1. 웹프로그래밍 설계모델



1. 웹 프로그래밍을 구축하기 위한 설계 모델



2 : 스프링 MVC플레임워크 설계 구조



3 : DispatcherServlet 설정

web.xml에 서블릿을매핑

WEB-INF폴더의 web.xml파일 만들고, <servlet>태그와 <servlet-mapping>태그를 이용한다.

```
<servlet>
 <servlet-name>서블릿 별칭</servlet-name>
 <servlet-class>서블릿명(패키지 이름을 포함한 전체서블릿명)</servlet-class>
</servlet>
<servlet-mapping>
 <servlet-name>서블릿별칭</servlet-name>
 <url-pattern>/맵핑명</url-pattern>
</servlet-mapping>
```

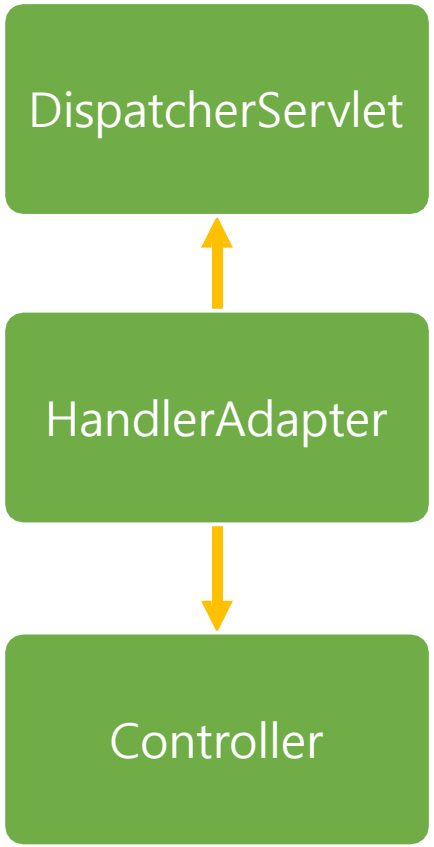
```
<servlet>
 <servlet-name>appServlet</servlet-name>
 <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
 <init-param>
  <param-name>contextConfigLocation</param-name>
  <param-value>/WEB-INF/spring/appServlet/servlet-context.xml</param-value>
 </init-param>
 <load-on-startup>1</load-on-startup>
</servlet>

<servlet-mapping>
 <servlet-name>appServlet</servlet-name>
 <url-pattern>/</url-pattern>
</servlet-mapping>
```

3 : DispatcherServlet 설정

초기화 파라미터에서 지정한 파일(servlet-context.xml)을
이용해서 스프링 컨테이너를 생성

초기화 파라미터에서 스프링 설정 파일을 지정하지 않은 경우
서블릿별칭을 이용해서 스프링 컨테이너 생성

4 : @Controller



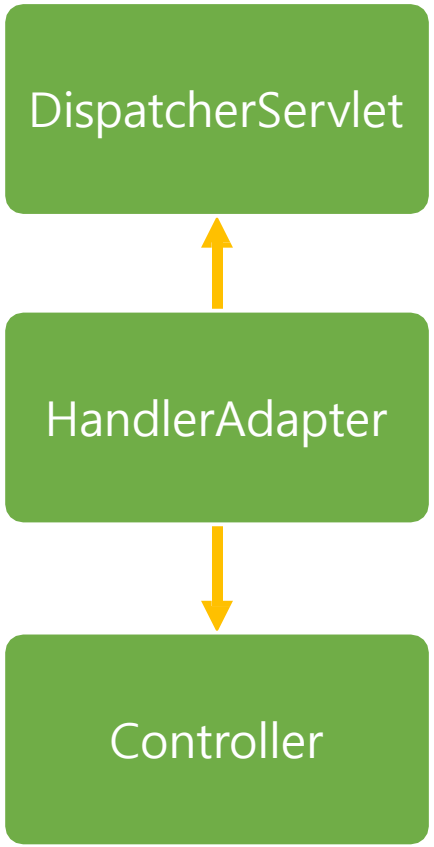
servlet-context.xml

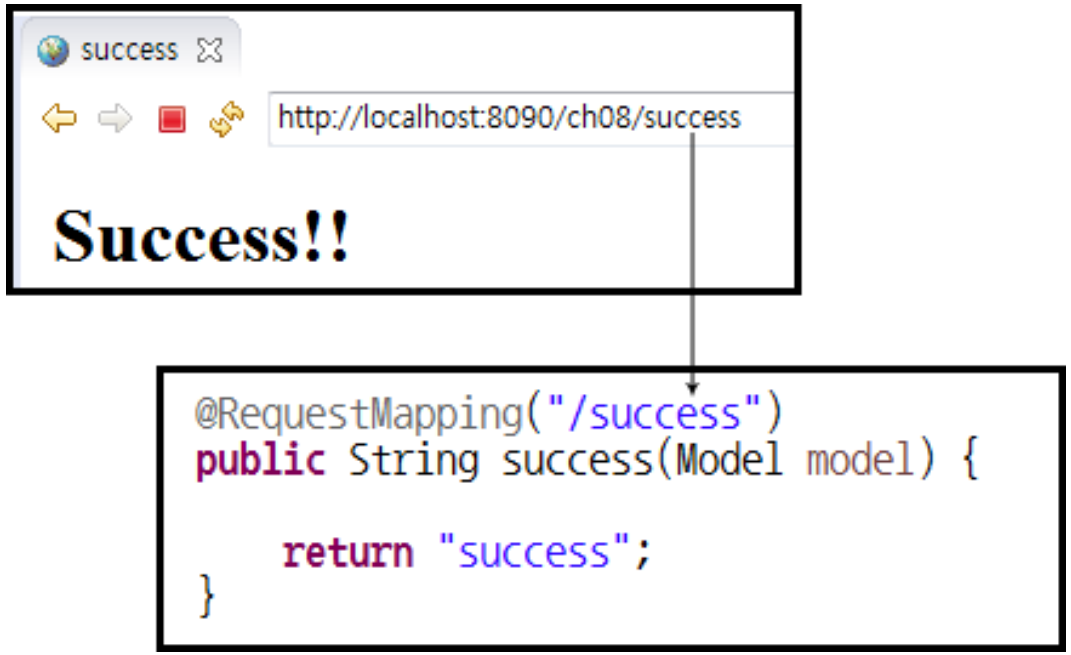
<annotation-driven />

Controller 객체로 사용할 클래스 정의

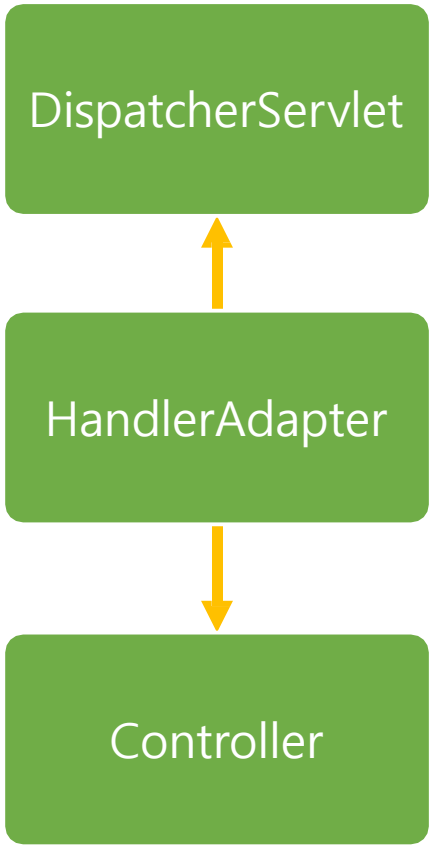
```
@Controller
public class HomeContoller {
  ….
}
```

5 : @RequestMapping





6: Model 타입의 파라미터

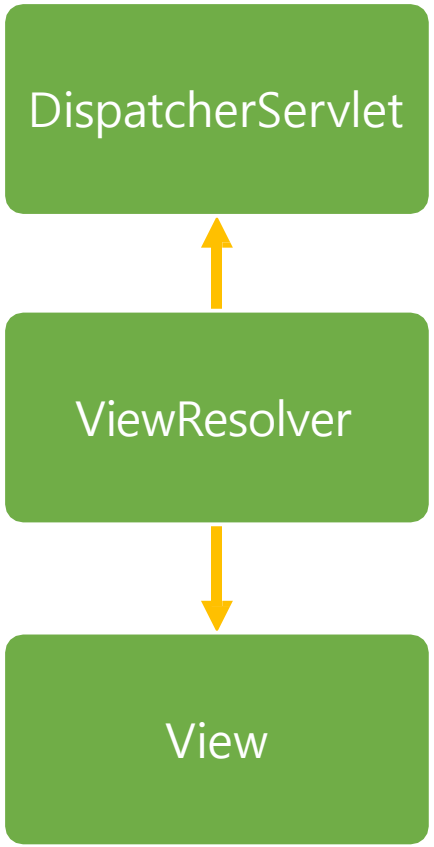



```
@RequestMapping("/success")
public String success(Model model) {
```

```
model.setAttribute("tempData", "model has data!!");
```

- 개발자는 Model 객체에 데이터를 담아서 DispatcherServlet에 전달할 수 있다.
- DispatcherServler에 전달된 Model데이터는 View에서 가공되어 클라이언트한테 응답처리 된다.

7 : View 객체




```
<beans:bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">
  <beans:property name="prefix" value="/WEB-INF/views/" />
<beans:property name="suffix" value=".jsp" />
</beans:bean>
```

```
@RequestMapping("/success")
public String success(Model model) {

    return "success";
}
```

```
<beans:bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">
    <beans:property name="prefix" value="/WEB-INF/views/" />
    <beans:property name="suffix" value=".jsp" />
</beans:bean>
```

JSP파일명 : /WEB-INF/views/success.jsp

JSP파일명 = return String값 + prefix값 + suffix값

8: 전체적인 웹프로그래밍 구조