accenture

Strategy | Digital | Technology | Operations

High performance. Delivered.

Practical Java Application Development with Spring

Module 2: Componentes do Spring MVC

Agenda

2f4869676820706572666f726d616e63652e2044656c6976657265642e2f4869676820706572666f726d616e63652e2044656c6976657265642e2f4869676820706572666f726d616e63652e2044656c6976657265642e2f4869676820706572666f726d616e63652e2044656c6976657265642e2f4869676820706572666f726d616e63652e2044656c6976657265642e2f48696768207

- J2EE Presentation Layer Overview
- Spring MVC – Introduction
- **Spring MVC – Components**
- Spring MVC – Features
- References
- Module Summary
- Appendix
  - J2EE Presentation Layer Technologies
  - Spring Web Application Development Stack
  - Struts MVC Vs. Spring MVC

Spring MVC – Components

2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070

- Spring contém os seguintes componentes:

Components	Description
Dispatcher servlet	• Servlet principal que atua como uma porta de entrada para as solicitações de entrada • Direciona as solicitações de entrada para os controladores apropriados
Handler mappings	• Mapeia solicitações de entrada para o controlador individual
Controller	• Lida com solicitações individuais • Contém lógica de navegação • Chama o objeto de serviço apropriado com base na lógica de negócios
ModelAndView	• Associe a visão ao pedido • Contém dados do modelo
ViewResolver	• Mapeia a visão lógica para a visão real
View	• Renderiza a saída

Spring MVC – Components Interaction (1 of 2)

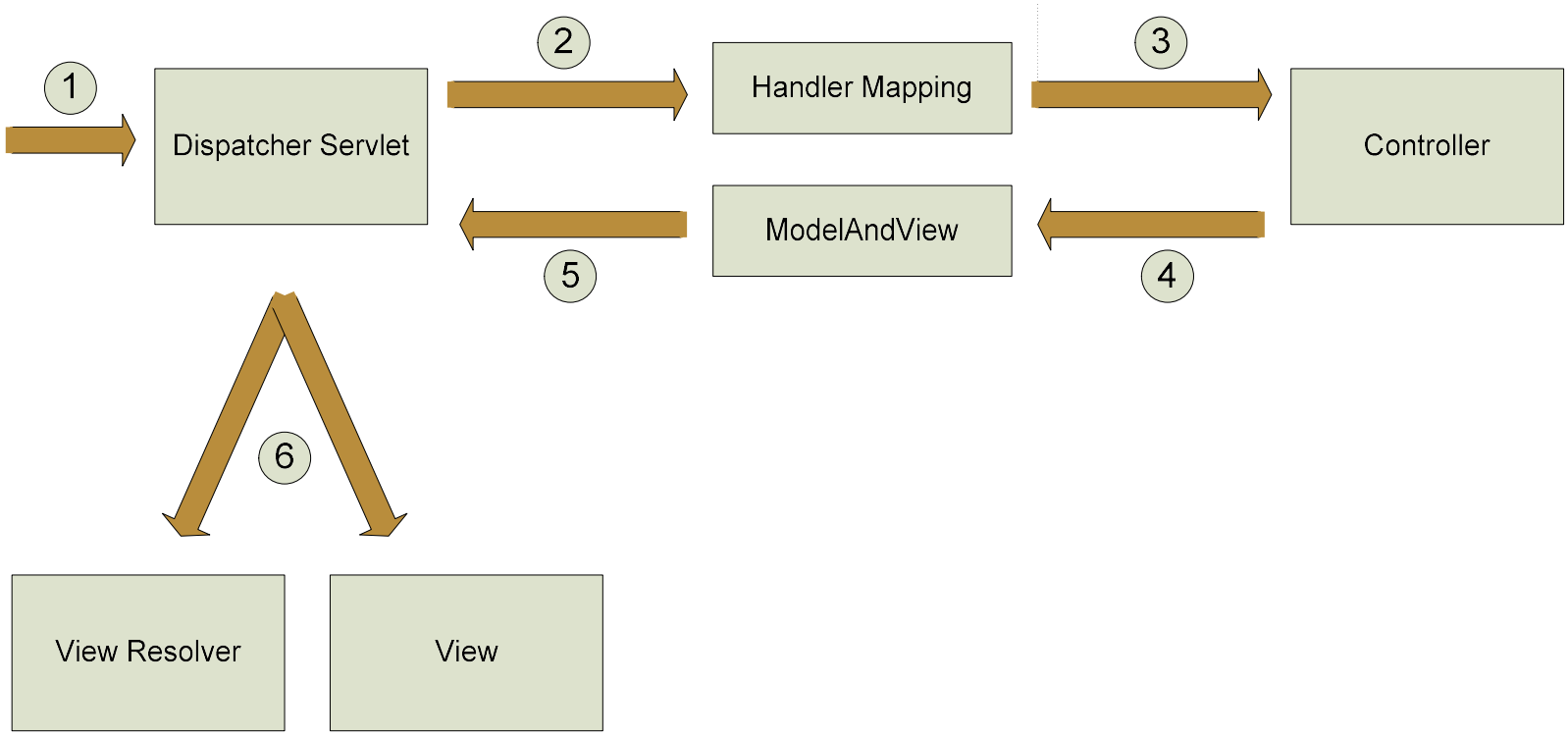
2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f48696768207

- Spring MVC flow:
  1. **Dispatcher servlet**, que atua como um **front controller** para a aplicação, recebe as solicitações do cliente.
  2. Utiliza o **handler mapping** para direcionar as requisições para um **controller** apropriado.
  3. A requisição é um designada para o **controller** apropriado.
  4. O **controller** processa a requisição e retorna o **model and view** na forma de um objeto do tipo **ModelAndView**.
  5. O objeto **ModelAndView** é então enviado de volta para o **front controller** (dispatcher servlet).
  6. O **front controller** então resolve a view atual atrávez do objeto View Resolver object e uma view apropriada será rendered de volta para o usuário.

Spring MVC – Components Interaction (2 of 2)

2f4869676820706572666f726d616e63652e2044656c6976657265642e2f4869676820706572666f726d616e63652e2044656c6976657265642e2f4869676820706572666f726d616e63652e2044656c6976657265642e2f4869676820706572666f726d616e63652e2044656c6976657265642e2f4869676820706572666f726d616e63652e2044656c6976657265642e2f48696768207

- Spring MVC flow diagram:

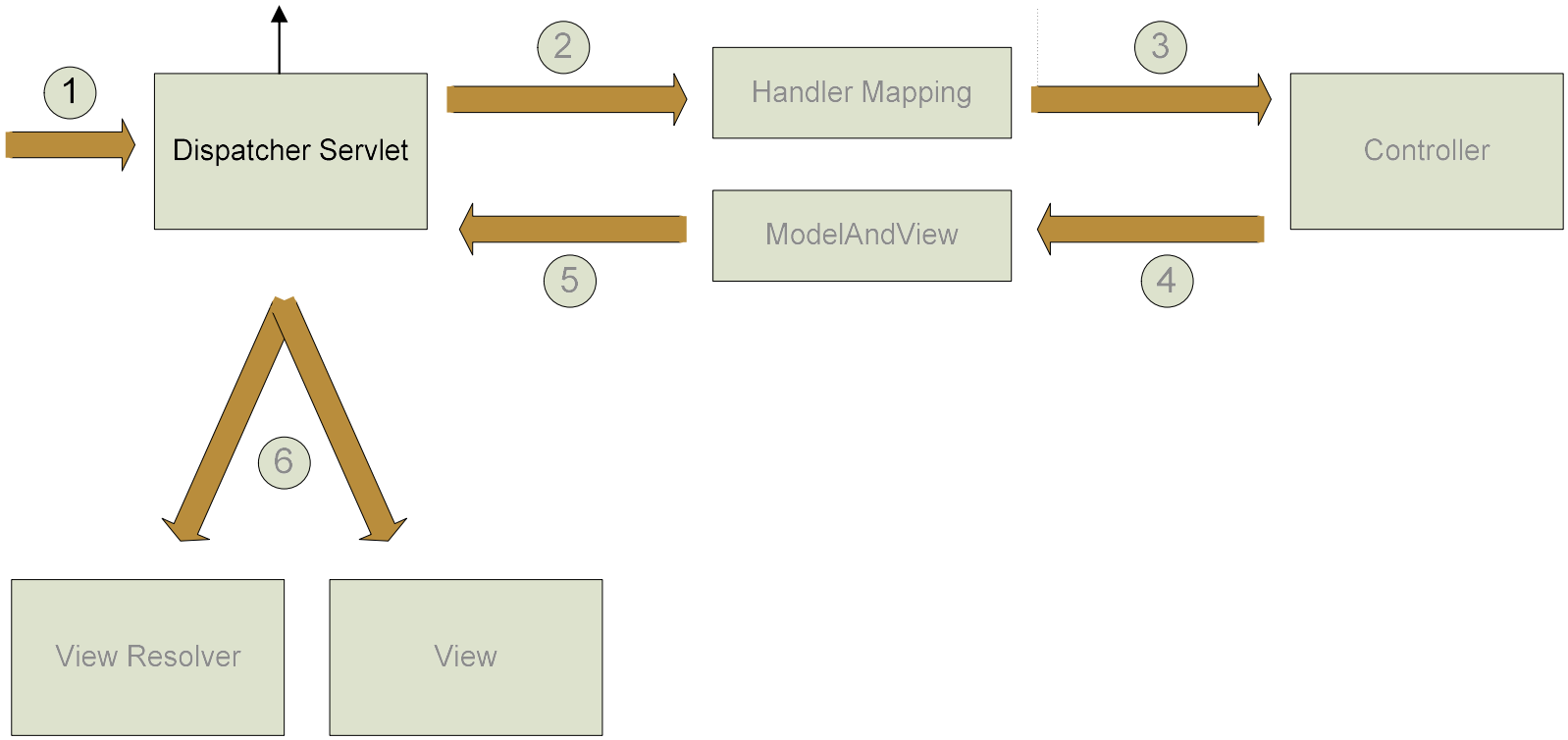


Spring MVC Components Interaction – Dispatcher Servlet

2f4869676820706572666f726d616e63652e2044656c6976657265642e2f4869676820706572666f726d616e63652e2044656c6976657265642e2f4869676820706572666f726d616e63652e2044656c6976657265642e2f4869676820706572666f726d616e63652e2044656c6976657265642e2f4869676820706572666f726d616e63652e2044656c6976657265642e2f48696768207

- Spring MVC flow diagram:

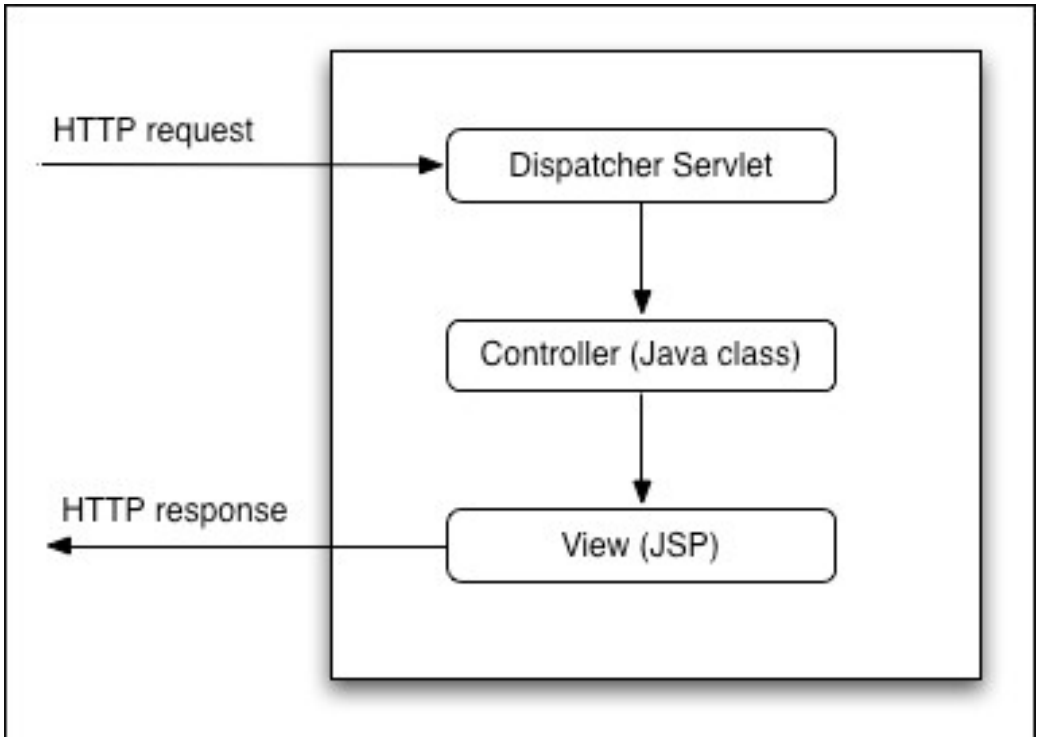
  - Servlet principal que atua como uma porta de entrada para as solicitações de entrada
  - Direciona as solicitações de entrada para os controladores apropriados



Spring MVC Components – Dispatcher Servlet (1 of 2)

2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f48696768207

- Spring MVC's **front controller**;
- Ponto de entrada para todas as solicitações;
- Conhece todos os **contollers** da aplicação;
- Tem a habilidade de identificar qual **controller** irá servir a solcitação através do **handler mapping**
- Definido no arquivo **web.xml**



Spring MVC – Components – Dispatcher Servlet (2 of 2)

2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f48696768207

- Dispatcher Servlet

web.xml

É necessário adicionar o Servlet que vai responder as requisições do Spring no web.xml com um mapeamento

Sem o Servlet do Spring ele não pode responder a requisições feitas ao servidor

Spring MVC – Components – Dispatcher Servlet (2 of 2)

2f4869676820706572666f726d616e63652e2044656c6976657265642e2f4869676820706572666f726d616e63652e2044656c6976657265642e2f4869676820706572666f726d616e63652e2044656c6976657265642e2f4869676820706572666f726d616e63652e2044656c6976657265642e2f4869676820706572666f726d616e63652e2044656c6976657265642e2f4869676820706572666f726d616e63652e2044656c6976657265642e2f48696768207

- Dispatcher Servlet

web.xml

```
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="http://java.sun.com/xml/ns/javaee"
    xmlns:jsp="http://java.sun.com/xml/ns/javaee/jsp"
    xmlns:web="http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
    xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
    http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd" id="WebApp_ID" version="2.5">
  <display-name>UniversityPortal</display-name>
  <welcome-file-list>
    <welcome-file>redirect.jsp</welcome-file>
  </welcome-file-list>
  <display-name>UniversityPortal</display-name>
  <servlet>
    <servlet-name>UniversityPortal</servlet-name>
    <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class
    <load-on-startup>1</load-on-startup>
  </servlet>
  <servlet-mapping>
    <servlet-name>UniversityPortal</servlet-name>
    <url-pattern>*.htm</url-pattern>
  </servlet-mapping>
  <jsp-config>
    <taglib>
      <taglib-uri>/spring</taglib-uri>
      <taglib-location>/WEB-INF/lib/spring-form.tld</taglib-location>
    </taglib>
  </jsp-config>
</web-app>
```

Redirect.jsp é o arquivo de boas-vindas.

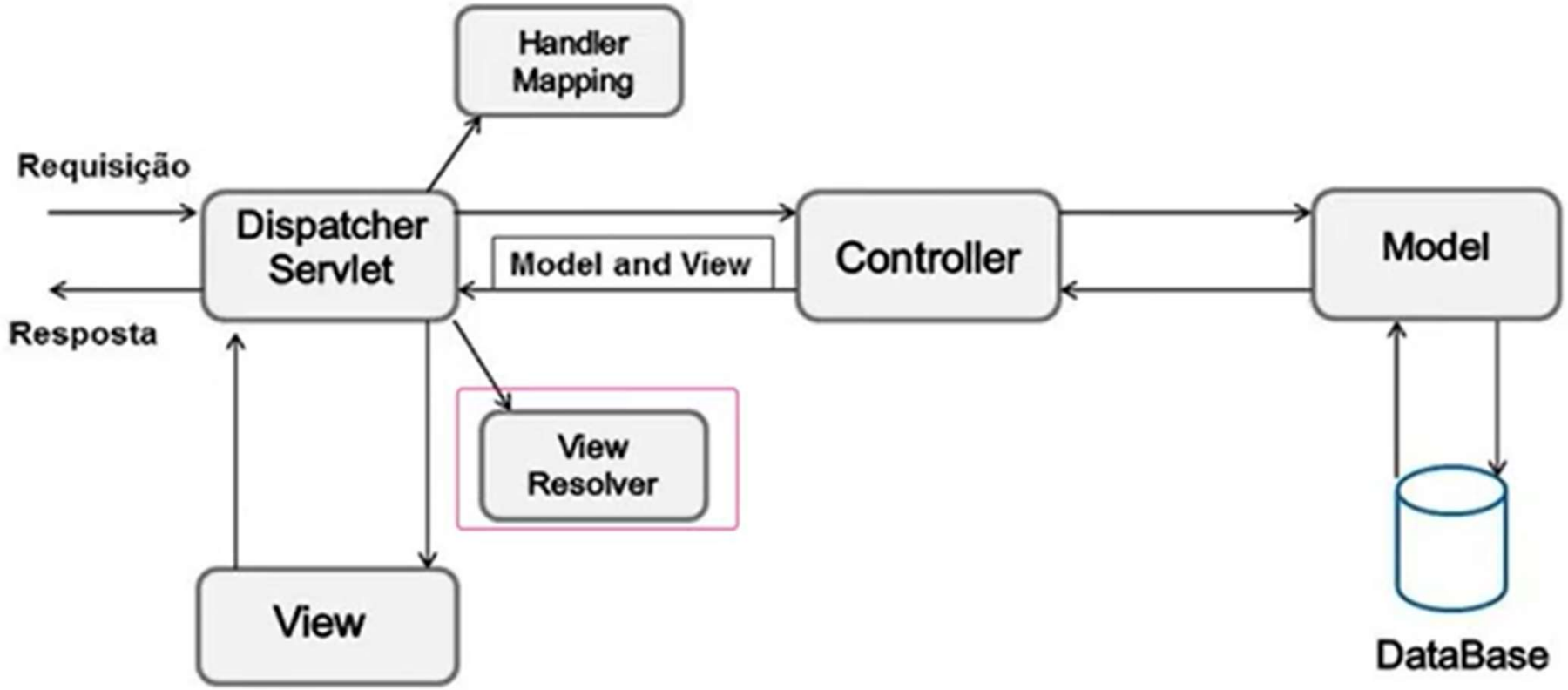
Defina DispatcherServlet em web.xml. Ele tratará de todas as solicitações do UniversityPortal.

Todas as solicitações de URL que terminam com a extensão .htm serão correspondidas para o aplicativo UniversityPortal.

Spring MVC Components Interaction – Handler Mapping

2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070

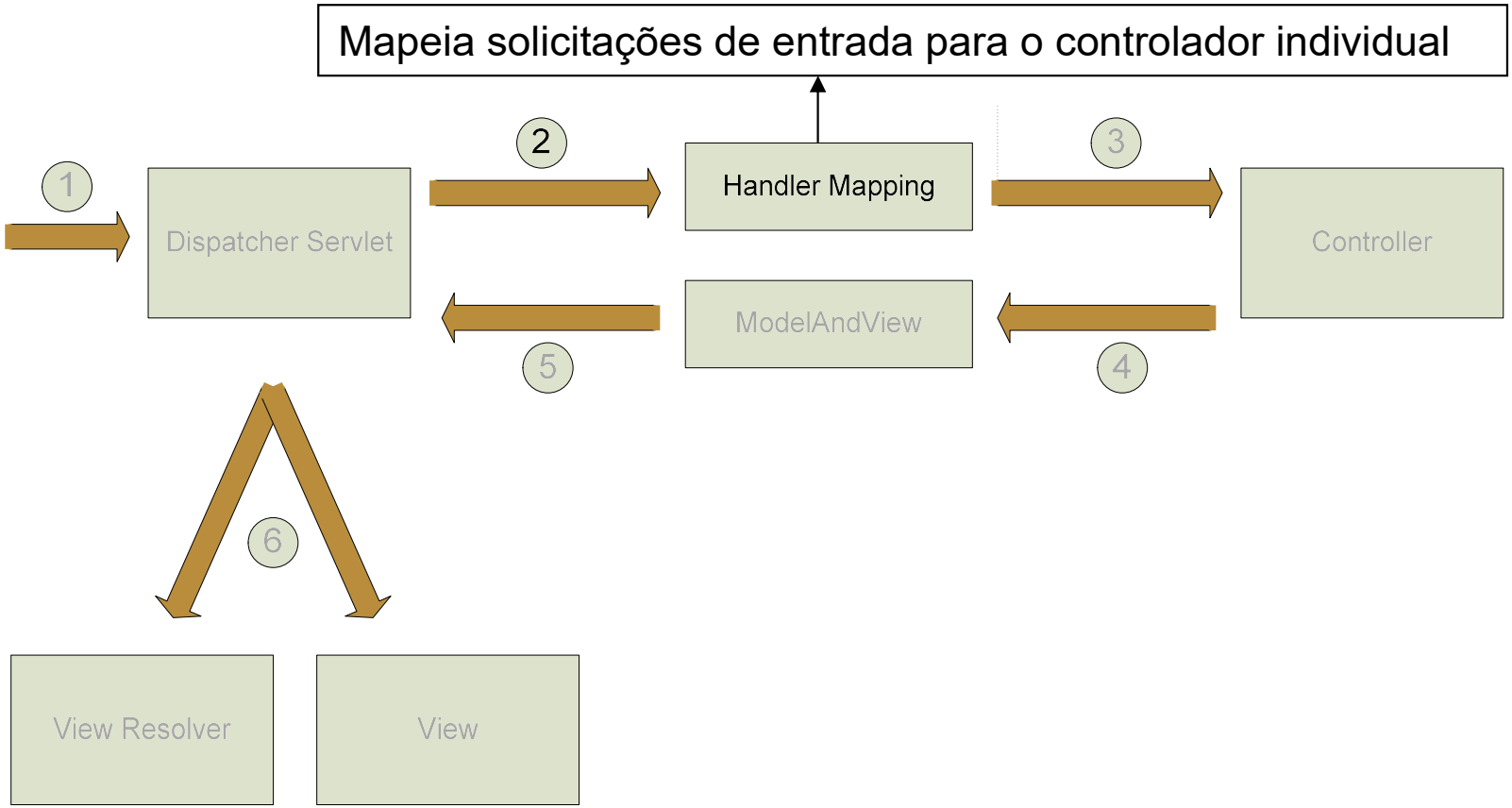
- Spring MVC flow diagram:



Spring MVC Components Interaction – Handler Mapping

2f4869676820706572666f726d616e63652e2044656c6976657265642e2f4869676820706572666f726d616e63652e2044656c6976657265642e2f4869676820706572666f726d616e63652e2044656c6976657265642e2f4869676820706572666f726d616e63652e2044656c6976657265642e2f4869676820706572666f726d616e63652e2044656c6976657265642e2f48696768207

- Spring MVC flow diagram:



Spring MVC – Components – Handler Mapping (1 of 6)

2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070

- Mecanismo que estabelece um link entre uma solicitação de URL recebida e uma classe de controlador;
- Inspeciona o URL path e o mapeia para um controlador cujo nome da classe corresponde ao caminho da URL definido no **RequestMapping**;
- Dentro da classe **controller**, pode haver um ou mais métodos de manipulador anotados pela anotação **@RequestMapping**.

  – **@RequestMapping** é usado para mapear URLs:
    - Ela suporta os métodos Post, Get, Put, Delete e Patch

Spring MVC – Components – Handler Mapping (2 of 6)

2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070

Annotations	Usage	Comments
@Controller	Define uma classe de controlador	O seguinte elemento no arquivo de metadados de configuração XML permite que o Spring detecte automaticamente os controladores com a anotação @Controller: <context:component-scan>
@RequestMapping	Define o mapeamento do manipulador	Para permitir que o Spring detecte automaticamente o mapeamento do manipulador com a anotação @RequestMapping.

Spring MVC – Components – Handler Mapping (3 of 6)

2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070

LoginController.java

```
import org.springframework.stereotype.Controller;


@Controller
@RequestMapping("/login.jsp")
// Class LoginController.
public class LoginController
{
//do something

    @RequestMapping
    public String showLoginForm(ModelMap model)
    {
        LoginForm loginform = new LoginForm();
        model.addAttribute(loginform);
        return "index";
    }
}
```

Mapeamento de solicitação em nível de classe. Este mapeamento significa que qualquer solicitação para /login.jsp invocará LoginController

Spring MVC – Components – Handler Mapping (4 of 6)

2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070

LoginController.java

```
import org.springframework.stereotype.Controller;


@Controller
@RequestMapping("/login.jsp")
// Class LoginController.
public class LoginController
{
//do something

    @RequestMapping
    public String showLoginForm(ModelMap model)
    {
        LoginForm loginform = new LoginForm();
        model.addAttribute(loginform);
        return "index";
    }
}
```

Instância de LoginForm é criada

LoginForm é adicionado ao modelo usando o método addAttribute. Veremos no próximo slide como obter referência à instância LoginForm usando @ModelAttribute

Index.jsp é a visão retornada

Spring MVC – Components – Handler Mapping (5 of 6)

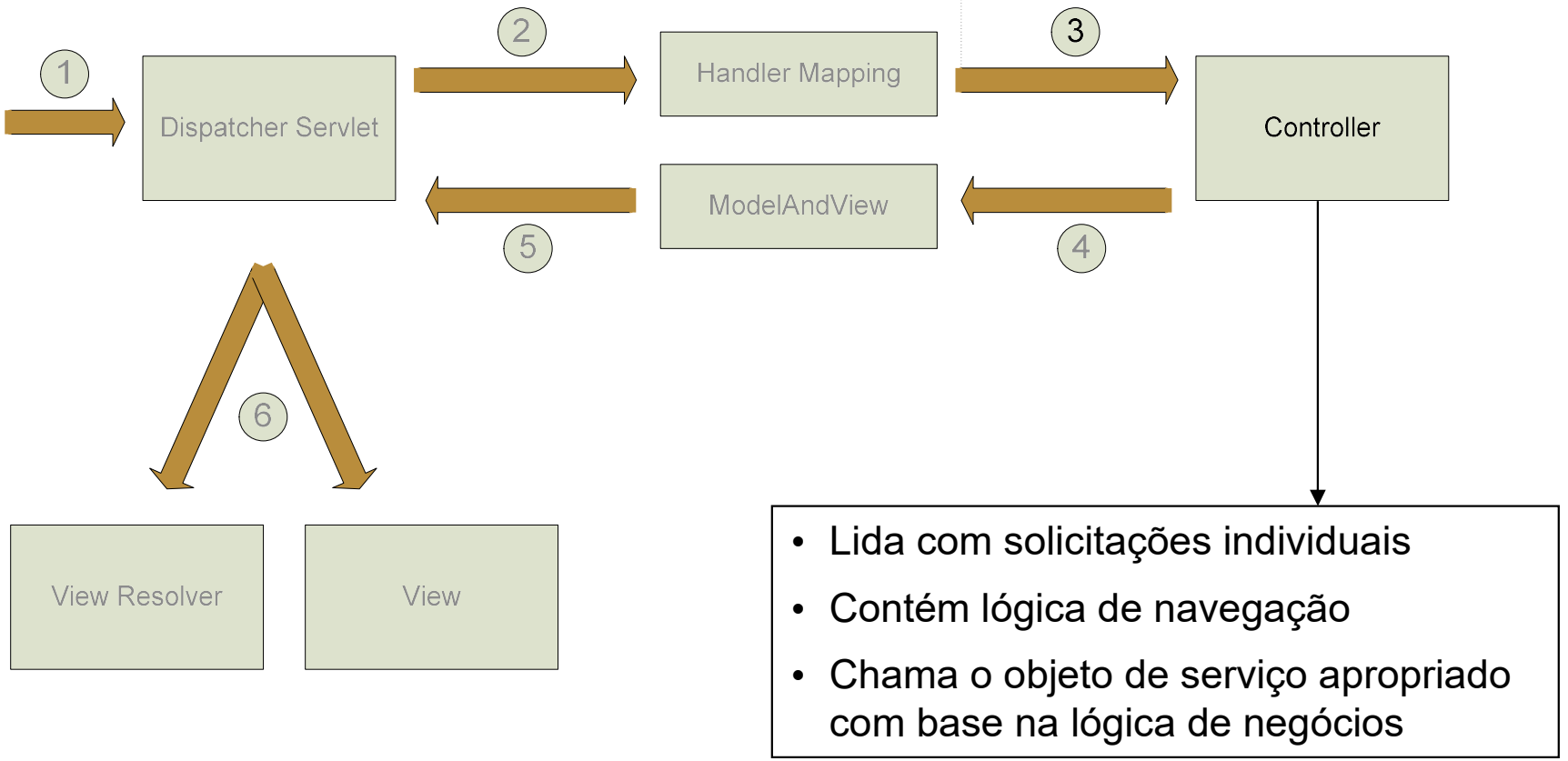
2f4869676820706572666f726d616e63652e2044656c6976657265642e2f4869676820706572666f726d616e63652e2044656c6976657265642e2f4869676820706572666f726d616e63652e2044656c6976657265642e2f4869676820706572666f726d616e63652e2044656c6976657265642e2f4869676820706572666f726d616e63652e2044656c6976657265642e2f48696768207

```java
@Controller
public class HelloControler{


// exemplo 1
  @RequestMapping( value ="/")
  public String hello(){
    return "Hello Spring MVC";
  }
// exemplo 2
  @RequestMapping( value ="/cadastro", method = POST)
  public String cadastros(){
    // logica aqui
   return "cadastro ok";
  }


}
```

Spring MVC Components Interaction – Controller

2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070

- Spring MVC flow diagram:

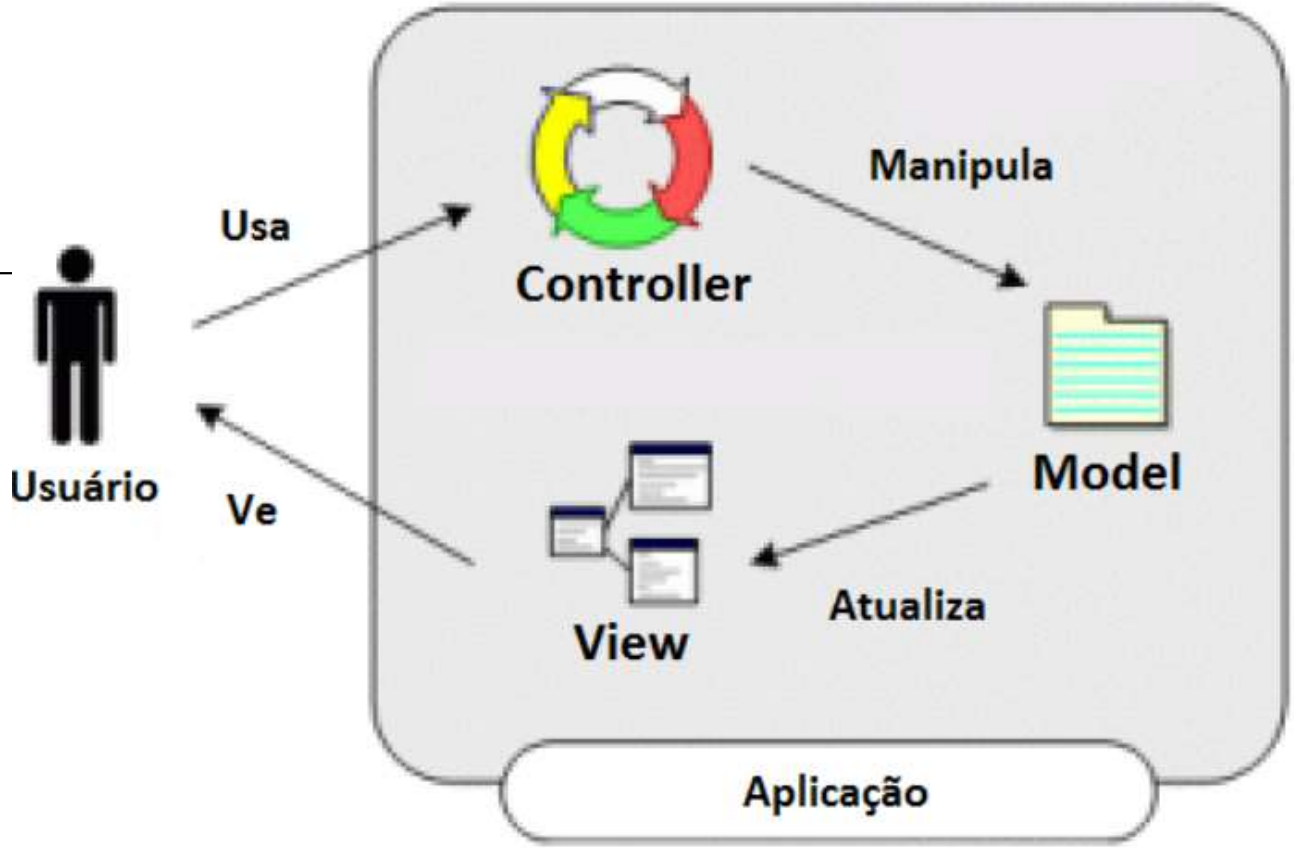


Spring MVC – Components – Controllers

2f4869676820706572666f726d616e63652e2044656c6976657265642e2f4869676820706572666f726d616e63652e2044656c6976657265642e2f4869676820706572666f726d616e63652e2044656c6976657265642e2f4869676820706572666f726d616e63652e2044656c6976657265642e2f4869676820706572666f726d616e63652e2044656c6976657265642e2f48696768207

- Recebe a requisição;
- Interpretar a entrada do usuário e transformá-la em um modelo;;
- **@Controller** annotation especifica um controller.

```
import org.springframework.stereotype.Controller;

@Controller
public class LoginController
{
    //do something…
}
```



Spring MVC – Components – Controllers

2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f48696768207

- **@Controller** annotation especifica um controller.
- é uma anotação comum usada para marcar uma classe como *Spring MVC Controller*,
- O controller e seus metodos são mapeados para solicitar uma URI usando **@RequestMapping**;
- São conhecidos como handlers;
- Podem existir multiplos controller em uma aplicação;.

A typical controller class looks like this:

```
@Controller
public class MyController {
    @RequestMapping(value = "/my-handler-path", method = RequestMethod.GET)
    public String myHandlerMethod(...) {
        .....
    }
}
```

Spring MVC – Components – Controllers


```
@Configuration
public class MyWebConfig {
    @Bean
    public MyController myController() {
        return new MyController();
    }
}
```

A typical controller class looks like this:

```
@Controller
public class MyController {
    @RequestMapping(value = "/my-handler-path", method = RequestMethod.GET)
    public String myHandlerMethod(...) {
        .....
    }
}
```

Spring MVC – The Model

2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f48696768207

```
@Controller
public class MyMvcController {
    @RequestMapping(value = "/my-uri-path")
    public String prepareView(Model model) {
        model.addAttribute("msg", "msg-value");
        .....
    }
}
```

Spring MVC – The View

2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f48696768207

```
<%@ page language="java"
    contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<html>
<body>
Message : ${msg}
</body>
</html>
```

Spring MVC – The ViewResolver

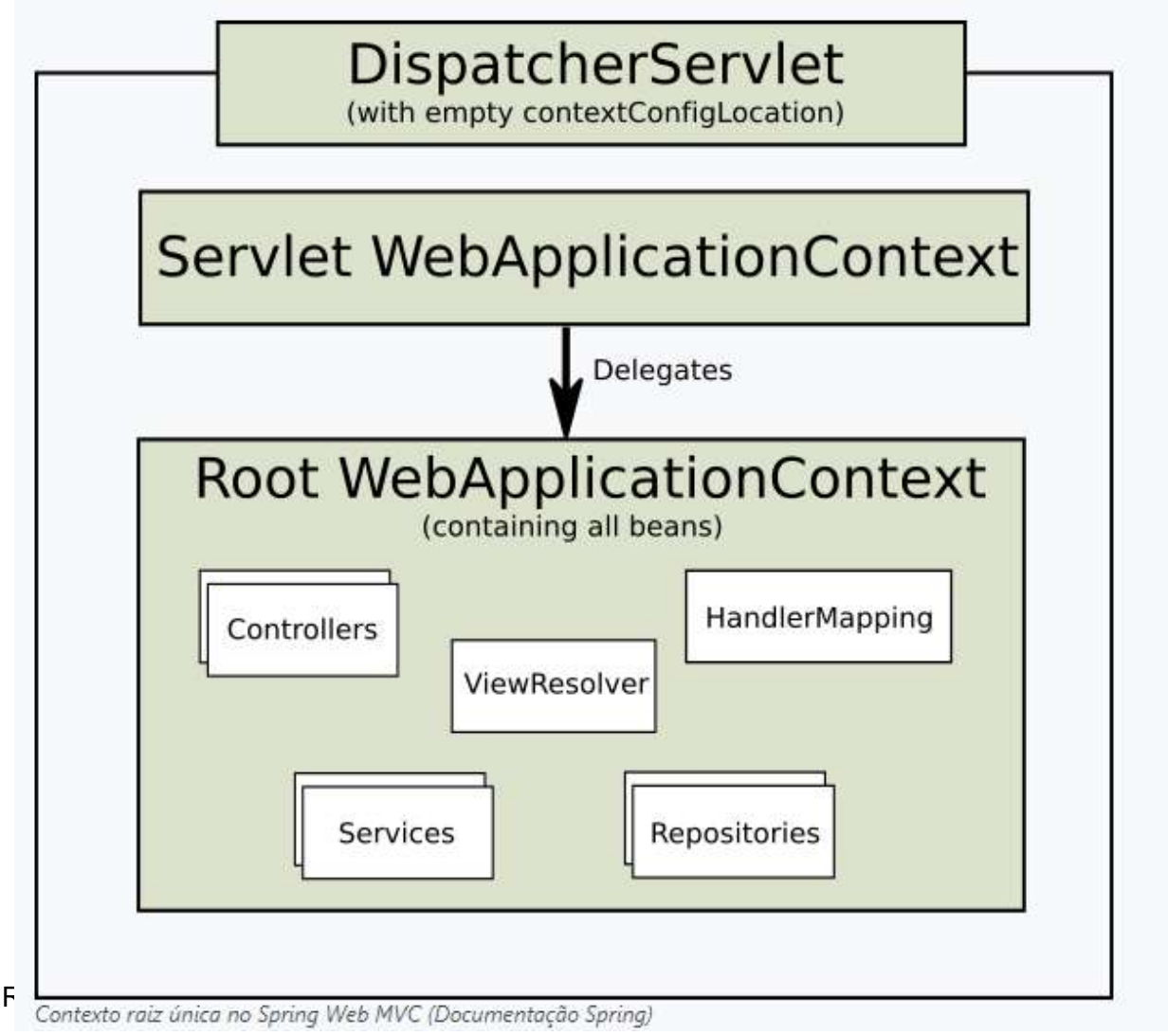
2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f48696768207

```
@Configuration
public class MyWebConfig {

    @Bean
    public ViewResolver viewResolver() {
        InternalResourceViewResolver viewResolver =
                        new InternalResourceViewResolver();

        viewResolver.setPrefix("/WEB-INF/views/");
        viewResolver.setSuffix(".jsp");
        return viewResolver;
    }
}
```

Spring MVC – The DispatcherServlet

2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f48696768207

O que é o DispatcherServlet?



Contexto raiz única no Spring Web MVC (Documentação Spring)

Spring MVC – The WebApplicationInitializer

2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f48696768207

O que é o WebApplicationInitializer?

Interface fornecida pelo Spring MVC

Funciona mapeando o acesso ao servlet por meio de código;

Spring MVC – The WebApplicationInitializer

O que é o WebApplicationInitializer?

```java
public class MyWebInitializer implements WebApplicationInitializer {

    @Override
    public void onStartup(ServletContext servletContext) throws ServletException {
        AnnotationConfigWebApplicationContext ctx =
                    new AnnotationConfigWebApplicationContext();
        ctx.register(MyWebConfig.class);
        ctx.setServletContext(servletContext);

        //using servlet 3 api to dynamically register
        //spring dispatcher servlet
        ServletRegistration.Dynamic servlet =
                    servletContext.addServlet("springDispatcherServlet",
                                             new DispatcherServlet(ctx));

        servlet.setLoadOnStartup(1);
        servlet.addMapping("/");
    }
```

Spring MVC – The AnnotationConfigWebApplicationContext

2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070

O que é AnnotationConfigWebApplicationContext?

É quem chama todas as classes baseadas em java e anotadas em spring diretamente;

Exemplo:

2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f48696768207

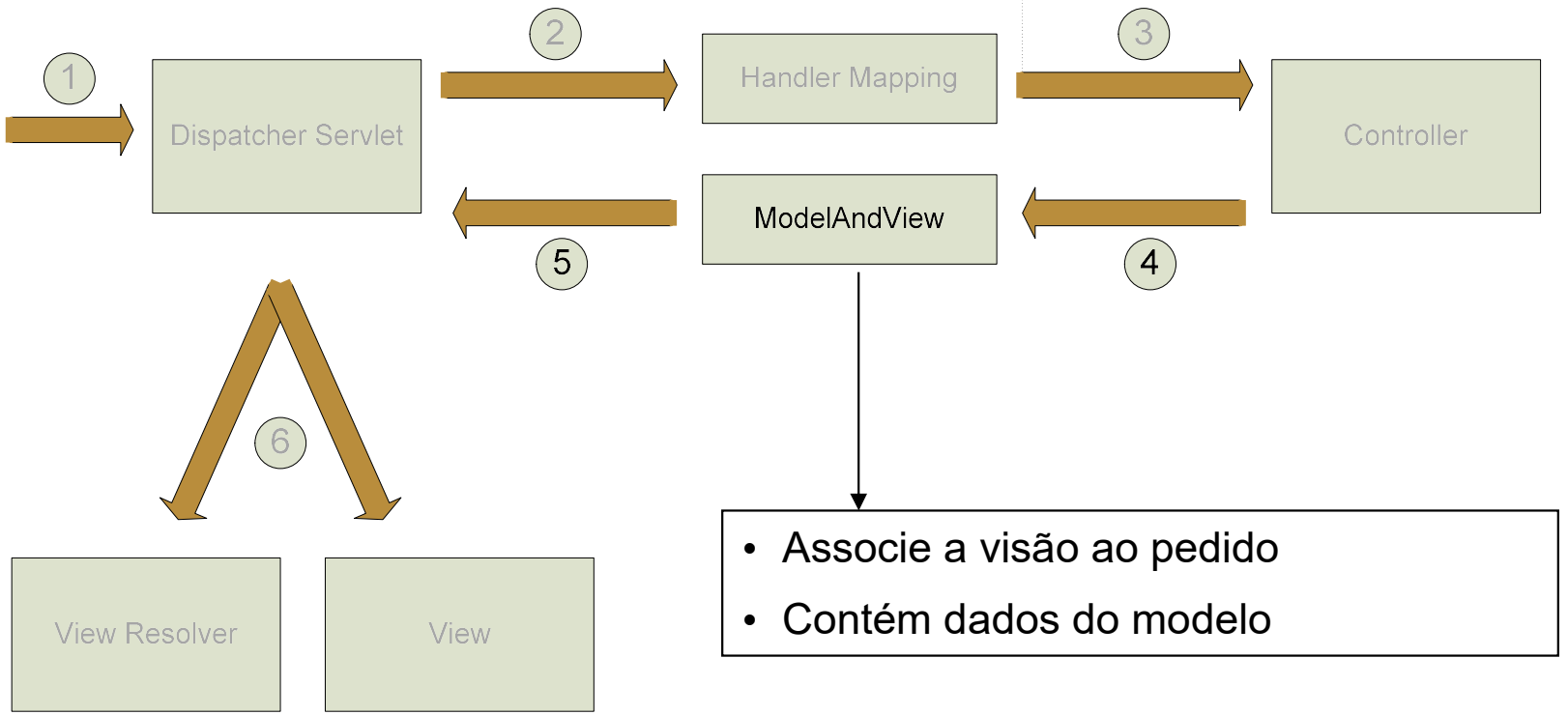
Exemplo:

Spring-mvc-example.zip

Spring MVC Components Interaction – ModelAndView

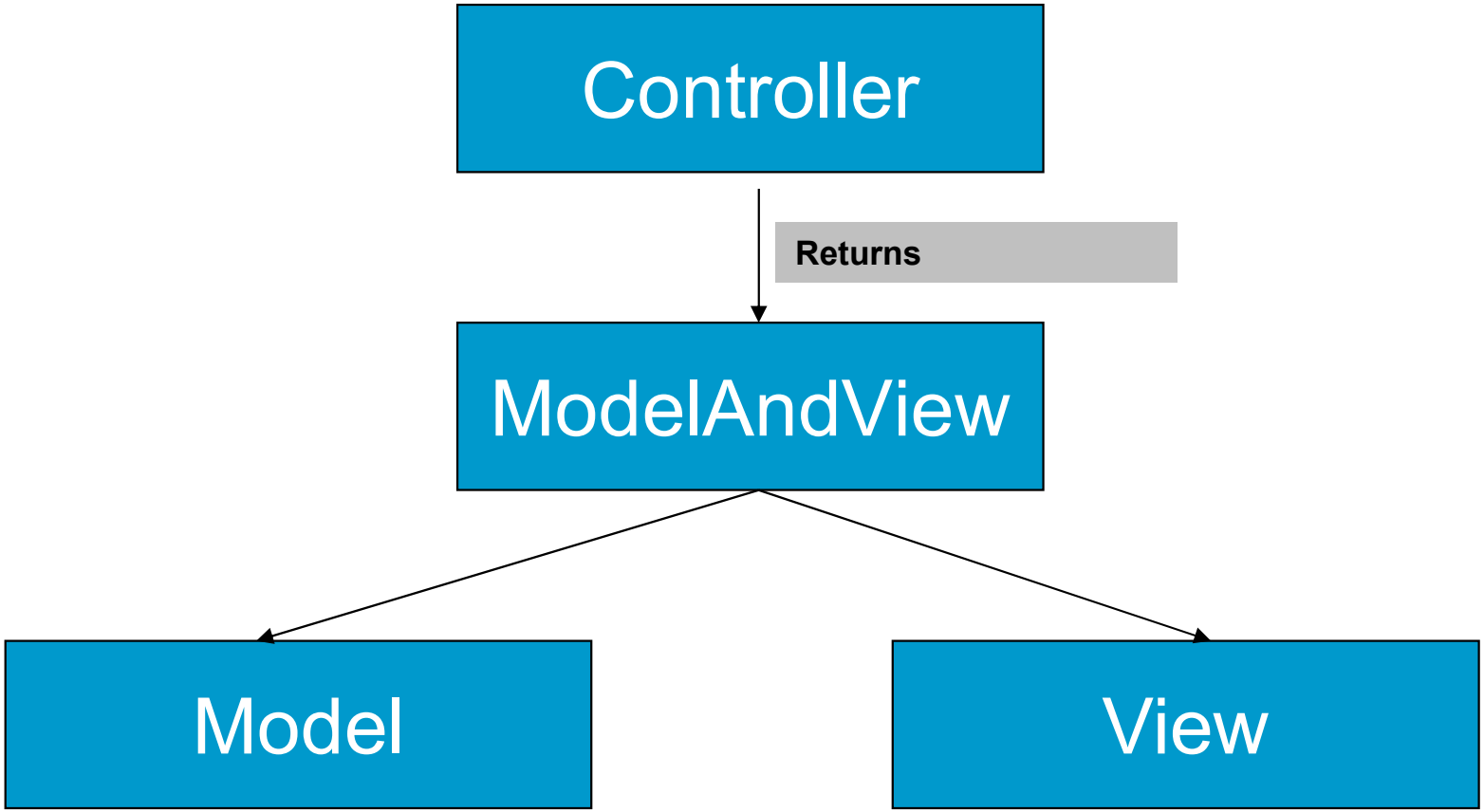
2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070

- Spring MVC flow diagram:



Spring MVC – Components – ModelAndView (1 of 3)

2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070



Spring MVC – Components – ModelAndView (2 of 3)

2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f48696768207

- ModelAndView
  - O controlador retorna o objeto **ModelAndView** para o despachante.
  - **ModelAndView** object contains:
    - Model
      - **@ModelAttribute**: annotation que liga um parâmetro de método ou valor de retorno de método a um atributo de modelo;
    - View
      - Pode ser uma string com o nome de exibição ou referência a um objeto de exibição;
      - Precisa ser resolvida pelo objeto **ViewResolver** object

Spring MVC – Components – ModelAndView (3 of 3)

2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f48696768207

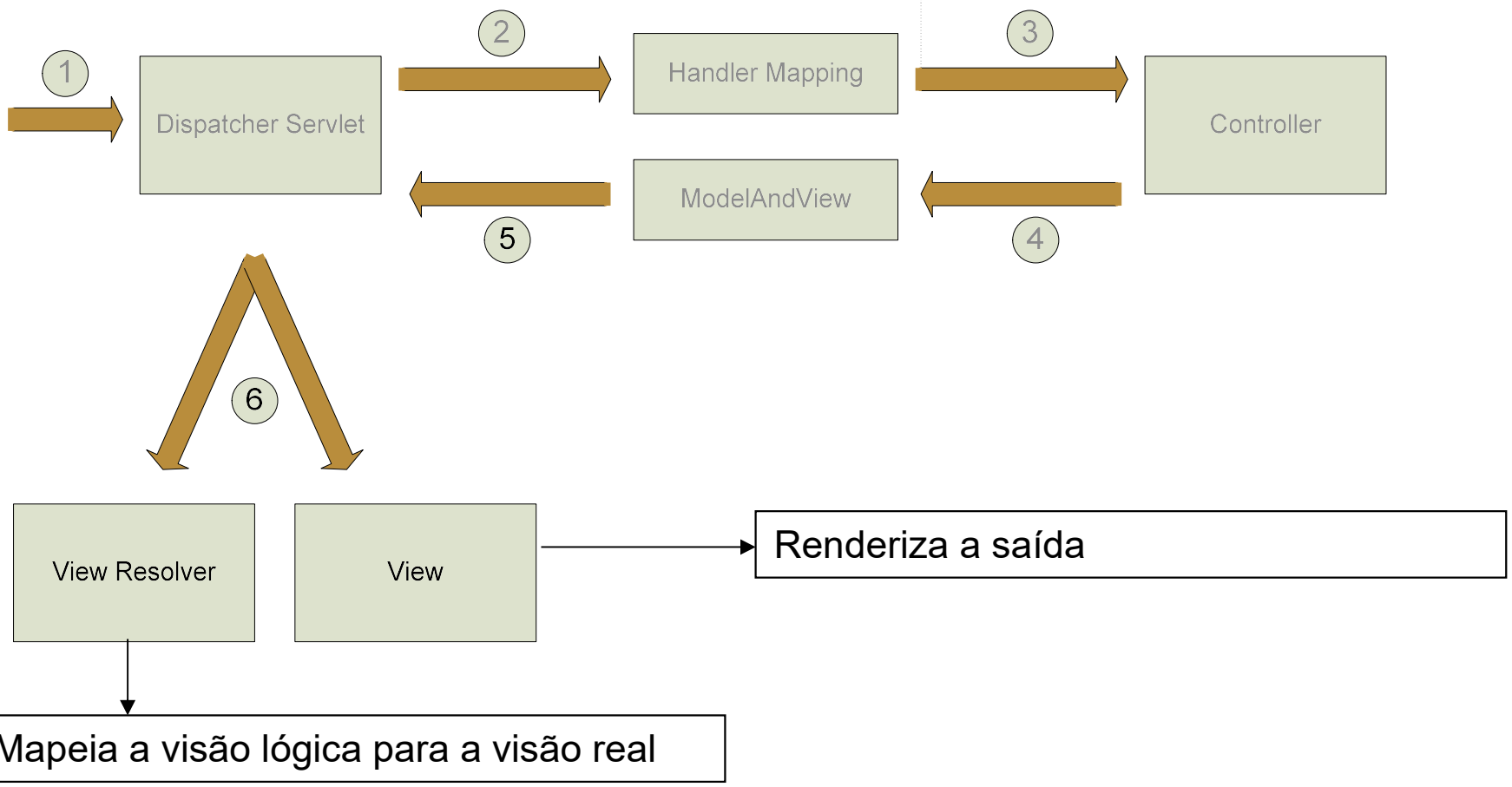
Em código...

```
protected ModelAndView handleRequestInternal
   (HttpServletRequest request,
                HttpServletResponse response) throws
   Exception {

   ModelAndView mv = new ModelAndView
   ( this.getView() );

   mv.getModel().put("objetos", this.getRepositorio
   ().pegarTodos( this.getClasse()) );

   return mv;
   }
```

Spring MVC Components Interaction – View And View Resolver

2f4869676820706572666f726d616e63652e2044656c6976657265642e2f4869676820706572666f726d616e63652e2044656c6976657265642e2f4869676820706572666f726d616e63652e2044656c6976657265642e2f4869676820706572666f726d616e63652e2044656c6976657265642e2f4869676820706572666f726d616e63652e2044656c6976657265642e2f48696768207

- Spring MVC flow diagram:



Spring MVC – View

2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f48696768207

- View
  - Retornado pelo **controller** para renderizar o **model**;
  - Gera respostas HTTP
  - A **View** retornada pode ser:
    - Concrete
      - new JstlView(“/WEB-INF/jsp/myPortal.jsp”)
    - View name
      - Nome da view que será resolvida pelo **DispatcherServlet**

Spring MVC – View Resolver

2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f48696768207

ViewResolvers

- São as classes responsáveis pela descoberta das views;
- Transformam nomes lógicos em uma requisição para a view apropriada;

Spring MVC – View Resolver

2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070

- **DispatcherServlet** recebe um modelo e um nome de exibição do objeto ModelAndView;

- Ele usa o **ViewResolver** para resolver o nome da exibição lógica retornada do objeto ModelAndView.
  - A ViewResolver is configured as a Bean

- Once **DispatcherServlet** has resolved a view name into a view object, it will render that view object.
- Uma vez que o **DispatcherServlet** tenha resolvido o nome de exibição em um objeto de exibição, ele renderizará esse objeto de exibição.

Spring MVC – View Resolver Example

2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f486967682070657266f726d616e63652e2044656c6976657265642e2f48696768207

Example: Construct a view using JSP

```
<!--Define a view resolver. InternalResourceViewResolver resolves logical view name to actual view-->
<bean id="viewResolver"
      class="org.springframework.web.servlet.view.InternalResourceViewResolver">
      <property name="prefix">
            <value>/WEB-INF/views/</value>
      </property>
      <property name="suffix">
            <value>.jsp</value>
      </property>
</bean>
```

Se o controlador retornar um novo **ModelAndView** ("successPage"), com base no código acima, successPage.jsp presente dentro de / WEB-INF / jsp / será gerado como uma visualização.

Questions and Comments

2f4869676820706572666f726d616e63652e2044656c6976657265642e2f4869676820706572666f726d616e63652e2044656c6976657265642e2f4869676820706572666f726d616e63652e2044656c6976657265642e2f4869676820706572666f726d616e63652e2044656c6976657265642e2f4869676820706572666f726d616e63652e2044656c6976657265642e2f48696768207

- What questions or comments do you have?