



Treinamento de Java

Spring MVC

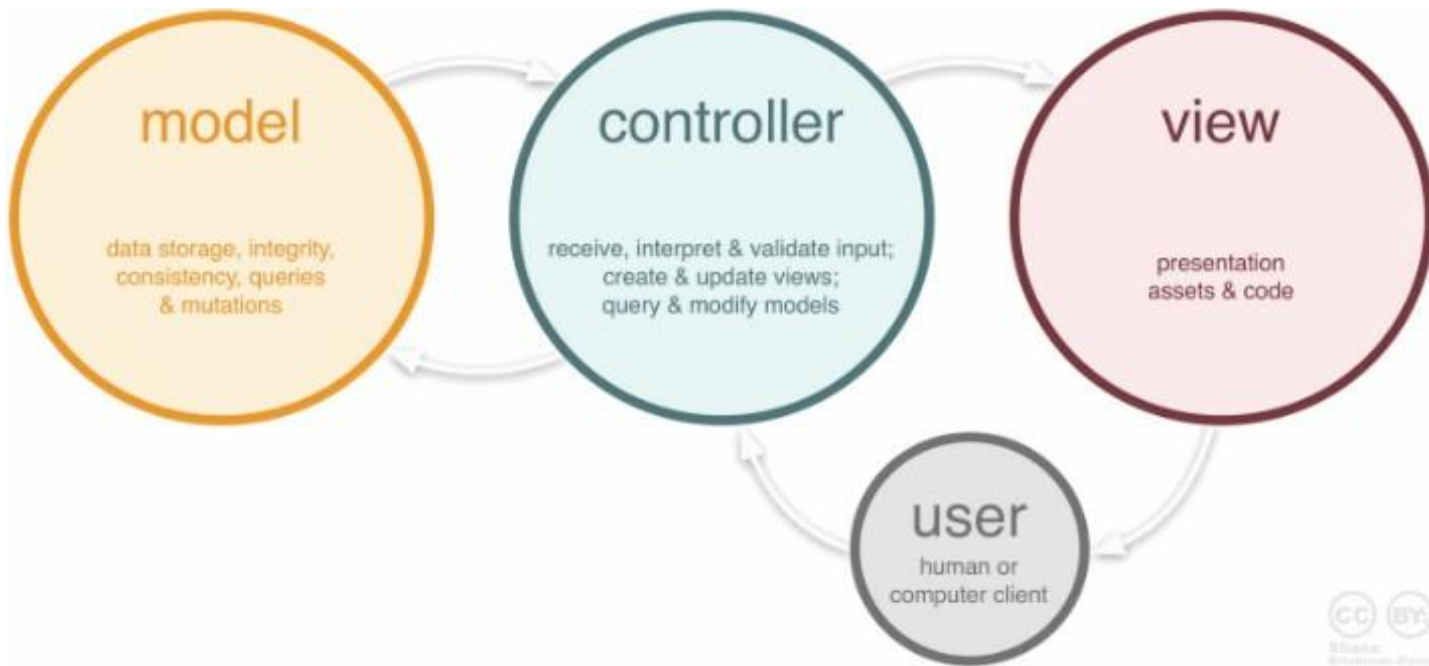
Abril de 2016

Fonte: PluralSight

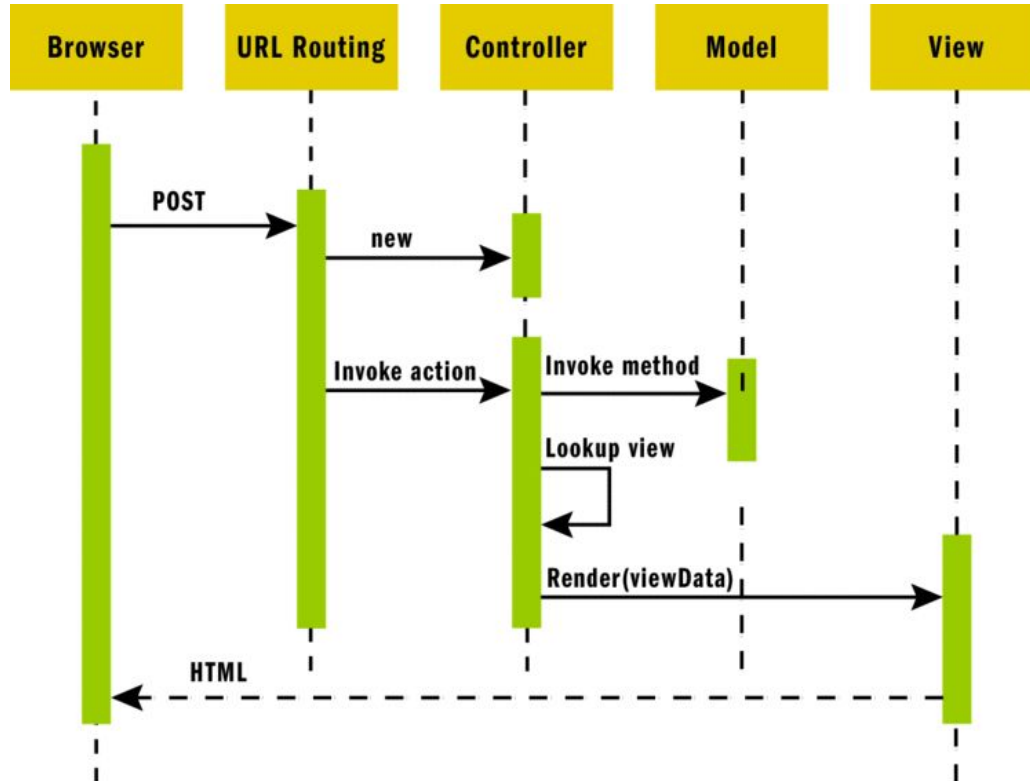


Spring MVC

O que é MVC?



Visão Básica



Configuração

Configuração -> New Project



New -> Spring Starter Project

Importante:

- > Java 1,7
- > Packaging war
- > Boot Version 1.1.12
- > Selecionar Web



Configuração -> Context



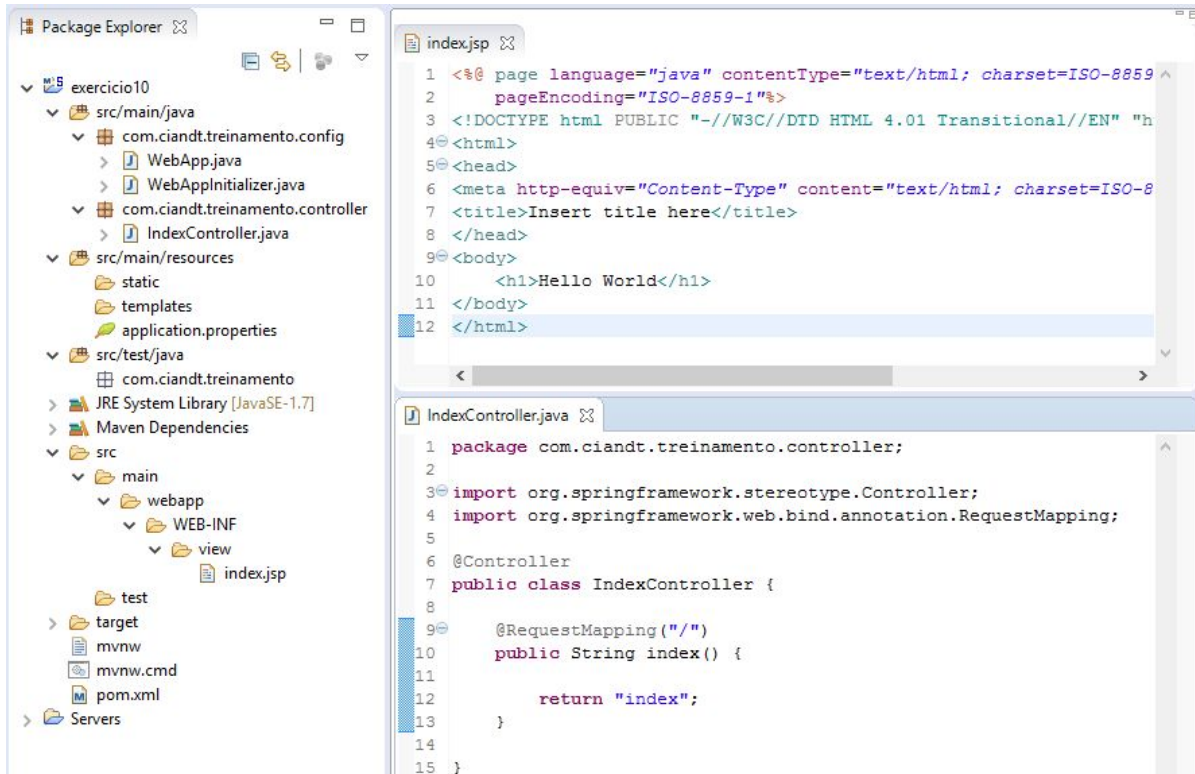
```
1 package com.ciandt.treinamento.config;
2
3+ import org.springframework.context.annotation.Bean;
4
5
6
7
8
9 @Configuration
10 @ComponentScan("com.ciandt.treinamento")
11 @EnableWebMvc
12 public class WebApp {
13
14-     @Bean
15     public InternalResourceViewResolver getViewResolver() {
16
17         InternalResourceViewResolver bean = new InternalResourceViewResolver();
18         bean.setPrefix("/WEB-INF/view/");
19         bean.setSuffix(".jsp");
20
21         return bean;
22     }
23
24 }
```

Configuração -> Without web.xml



```
1 package com.ciandt.treinamento.config;
2
3+ import javax.servlet.ServletContext;
4
11
12 public class WebAppInitializer implements WebApplicationInitializer {
13
14-     @Override
15     public void onStartUp(ServletContext servletContext) throws ServletException {
16
17         AnnotationConfigWebApplicationContext context = new AnnotationConfigWebApplicationContext();
18         context.register(WebApp.class);
19
20         servletContext.addListener(new ContextLoaderListener(context));
21
22         ServletRegistration.Dynamic dispatcher = servletContext.addServlet("DispatcherServlet",
23             new DispatcherServlet(context));
24         dispatcher.setLoadOnStartup(1);
25         dispatcher.addMapping("/");
26
27     }
28
29 }
```


Configuração -> Hello World!



Configuração -> Tomcat 7



Run On Server

Select which server to use

How do you want to select the server?

☐ Choose an existing server

☒ Manually define a new server

[Download additional server adapters](#)

Select the server type:

type filter text

- Apache
 - Tomcat v3.2 Server
 - Tomcat v4.0 Server
 - Tomcat v4.1 Server
 - Tomcat v5.0 Server
 - Tomcat v5.5 Server
 - Tomcat v6.0 Server
 - Tomcat v7.0 Server**
 - Tomcat v8.0 Server
- Basic
- ObjectWeb
- Pivotal
 - Cloud Foundry
 - Pivotal tc Server v2.5 - v2.9
 - Pivotal tc Server v3.0 - v3.1

Publishes and runs J2EE and Java EE Web projects and server configurations to a local Tomcat server

Server's host name: localhost

Server name: Tomcat v7.0 Server at localhost

☐ Always use this server when running this project

[?](#) < Back Next > Finish Cancel

Run On Server

Tomcat Server

Specify the installation directory

Name: Apache Tomcat v7.0

Tomcat installation directory: C:\Tools\apache-tomcat-7.0.64

apache-tomcat-7.0.47 Download and Install...

JRE: jre7

Installed JREs...

[?](#) < Back Next > Finish Cancel

Run On Server

Add and Remove

exercicio10 is required and cannot be removed from the server

Move resources to the right to configure them on the server

Available:

Configured: > exercicio10

Add >

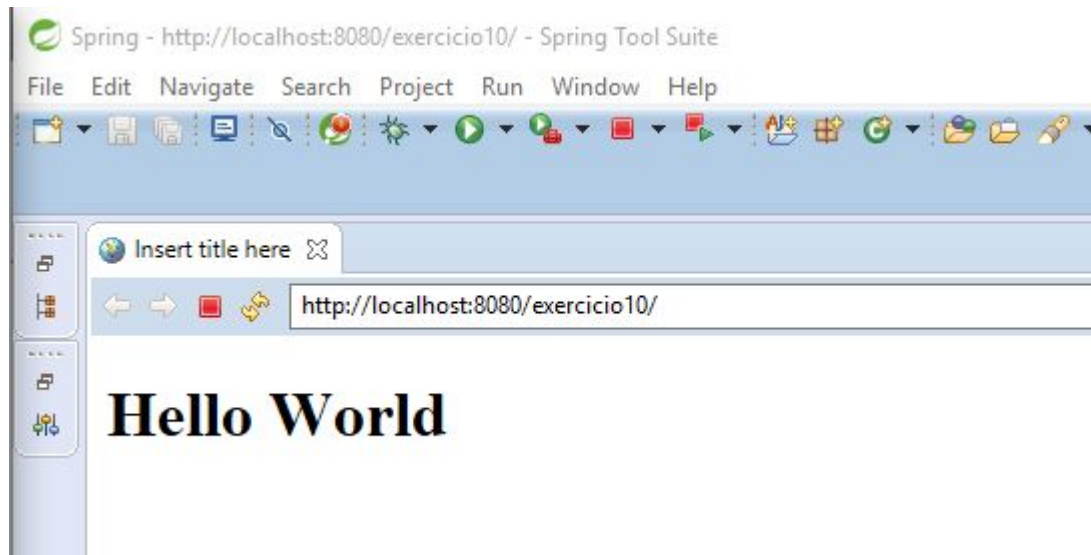
< Remove

Add All >>

<< Remove All

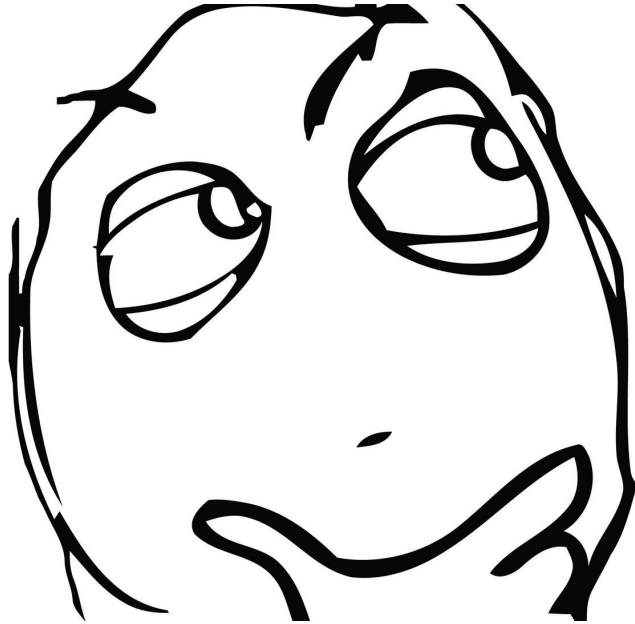
[?](#) < Back Next > Finish Cancel

Configuração -> It's alive!



Vamos a um exemplo?

CRUD Simples - Produto



Controller \ View

@Controller



```
@Controller
@RequestMapping("/appointments")
public class AppointmentsController {

    private final AppointmentBook appointmentBook;

    @Autowired
    public AppointmentsController(AppointmentBook appointmentBook) {
        this.appointmentBook = appointmentBook;
    }

    @RequestMapping(method = RequestMethod.GET)
    public Map<String, Appointment> get() {
        return appointmentBook.getAppointmentsForToday();
    }
}
```

@RequestMapping



```
@RequestMapping(path="/owners/{ownerId}/pets/{petId}", method=RequestMethod.GET)
public String findPet(@PathVariable String ownerId, @PathVariable String petId, Model model) {
    Owner owner = ownerService.findOwner(ownerId);
    Pet pet = owner.getPet(petId);
    model.addAttribute("pet", pet);
    return "displayPet";
}
```

```
@RequestMapping("/spring-web/{symbolicName:[a-z-]+}-{version:\\d\\.\\d\\.\\d}{extension:\\.[a-z]+}")
public void handle(@PathVariable String version, @PathVariable String extension) {
    // ...
}
}
```


@RequestMapping



Host	localhost:8080
Accept	text/html,application/xhtml+xml,application/xml;q=0.9
Accept-Language	fr,en-gb;q=0.7,en;q=0.3
Accept-Encoding	gzip,deflate
Accept-Charset	ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive	300

```
@RequestMapping("/displayHeaderInfo.do")
public void displayHeaderInfo(@RequestHeader("Accept-Encoding") String encoding,
    @RequestHeader("Keep-Alive") long keepAlive) {
    //...
}
```

Validation

Validation JSR-303

@ Digits - javax.validation.constraints

@ Size - javax.validation.constraints

@ Min - javax.validation.constraints

@ NotNull - javax.validation.constraints

@ AssertFalse - javax.validation.constraints

@ AssertTrue - javax.validation.constraints

@ DecimalMax - javax.validation.constraints

@ DecimalMin - javax.validation.constraints

@ Future - javax.validation.constraints

@ Max - javax.validation.constraints

@ Null - javax.validation.constraints

@ Past - javax.validation.constraints

@ Pattern - javax.validation.constraints

@ Range - org.hibernate.validator.constraints

@ NotBlank - org.hibernate.validator.constraints

@ NotEmpty - org.hibernate.validator.constraints

ⓔ CompositionType - org.hibernate.validator.constraints

@ ConstraintComposition - org.hibernate.validator.constraints

@ CreditCardNumber - org.hibernate.validator.constraints

@ Email - org.hibernate.validator.constraints

@ Length - org.hibernate.validator.constraints

@ ModCheck - org.hibernate.validator.constraints

@ SafeHtml - org.hibernate.validator.constraints

@ ScriptAssert - org.hibernate.validator.constraints

@ URL - org.hibernate.validator.constraints

⊞ org.hibernate.validator.constraints.br

@ CNPJ - org.hibernate.validator.constraints.br

@ CPF - org.hibernate.validator.constraints.br

@ TituloEleitoral - org.hibernate.validator.constraints.br

Dojo!

Exercício 1

Fazer a seguinte API

Endpoint: <http://localhost:8080/dojo-02/v1>

GET - /cartoes

query - cpf

Retorna os cartões do cpf

GET - /cartoes/{id_cartao}/faturas

Header - cpf

Retorna as faturas do cartao

POST - /cartoes

Header - cpf

Body - Cartao

Retorna uma mensagem de sucesso

Dados de teste

CPF Válido 01234567890

Cartão válido 1234

Exercício 2

Fazer a seguinte API

Endpoint: <http://localhost:8080/dojo-02/v1>

GET - /pessoa

header - cpf

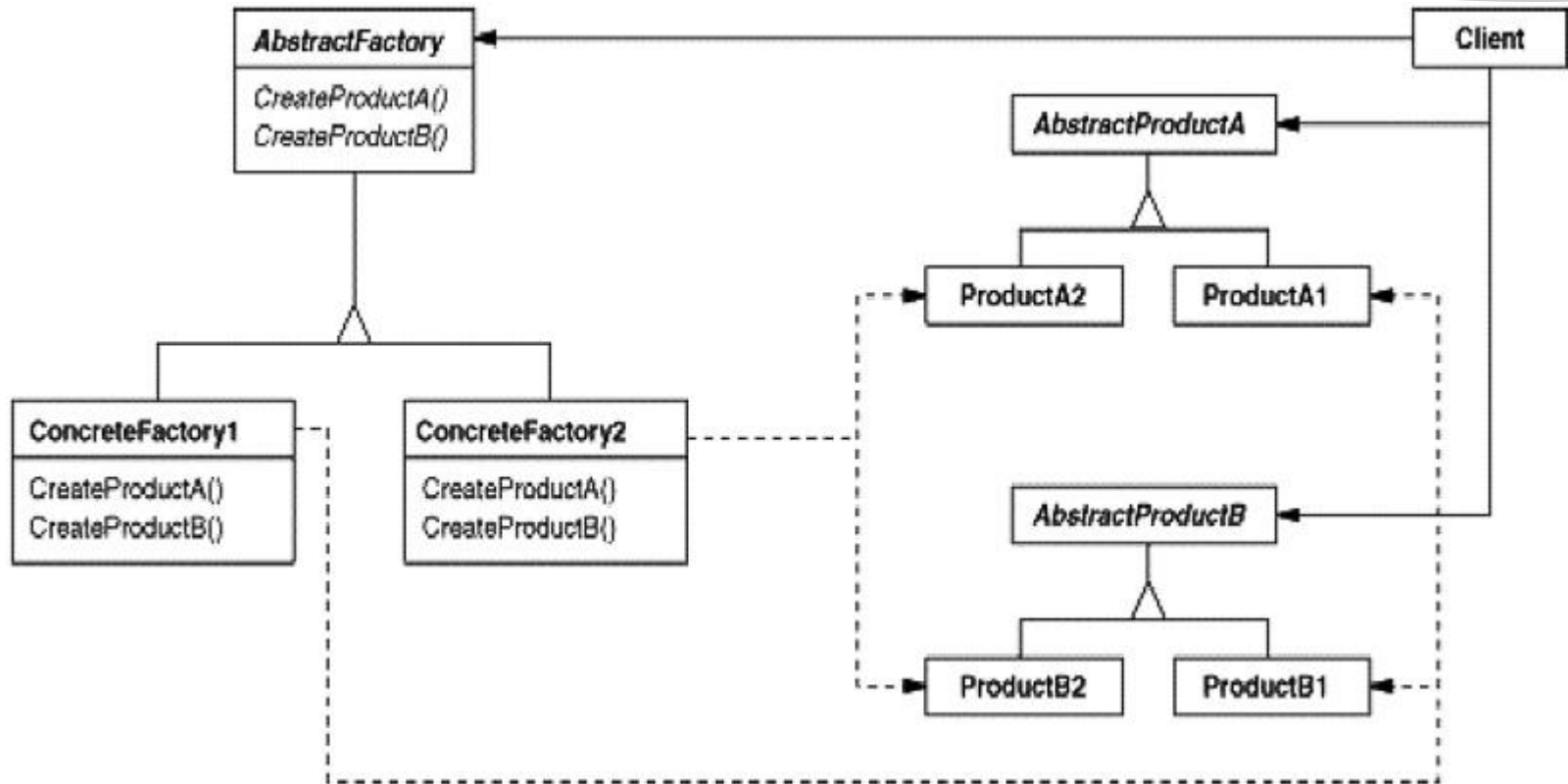
Retorna os dados da pessoa

Dados de teste

CPF Válido 01234567890

Cartão válido 1234


Design Patterns - Factory e AbstractFactory



Custom Validation


Vamos criar uma
Annotation!

Custom Validation



```
1 package com.ciandt.treinamento.validation;
2
3+ import javax.validation.ConstraintValidator;
4
5
6 public class TelefoneValidador implements ConstraintValidator<Telefone, String> {
7
8     @Override
9     public void initialize(Telefone constraintAnnotation) {
10     }
11
12     @Override
13     public boolean isValid(String value, ConstraintValidatorContext context) {
14         if (value == null) return false;
15
16         return value.matches("\\(\\d{2}\\)\\d{4,5}-\\d{4}");
17     }
18
19 }
```

Custom Validation



```
Telephone.java ☒ TelephoneValidador.java
1 package com.ciandt.treinamento.validation;
2
3+ import java.lang.annotation.Documented;
11
12 @Documented
13 @Constraint(validatedBy = TelefoneValidador.class)
14 @Target({ ElementType.METHOD, ElementType.FIELD })
15 @Retention(RetentionPolicy.RUNTIME)
16 public @interface Telefone {
17
18     String message() default "Telefone inválido";
19
20     Class<?>[] groups() default {};
21
22     Class<? extends Payload>[] payload() default {};
23
24 }
```

