



*Desenvolvimento de Aplicações WEB*  
***Frameworks***

Profa. Joyce Miranda

*Materiais de Referência:* <http://www.caelum.com.br/apostila-java-web>

# Frameworks

---

## ► Framework

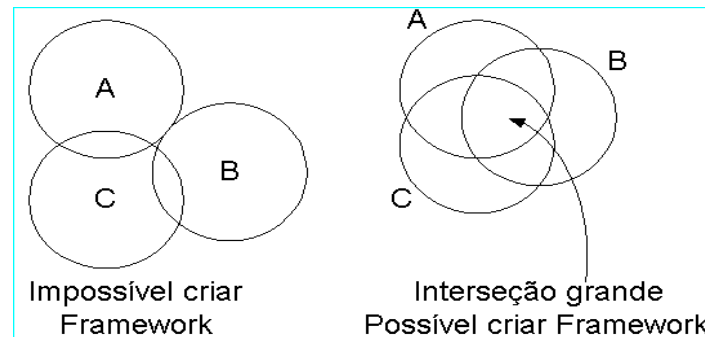
### ► Objetivo Principal

#### ► Reusabilidade

- Aumento da produtividade e redução de esforço de desenvolvimento

### ► Definições

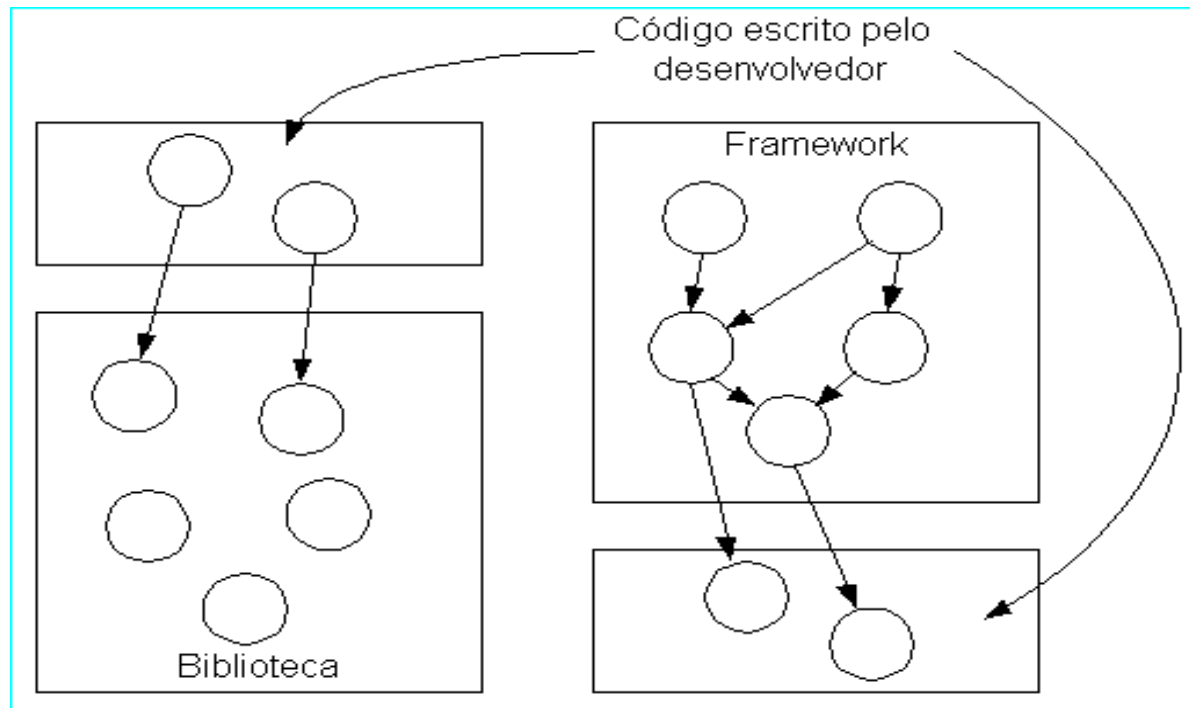
1. Solução projetada para encapsular conhecimento aplicável a diferentes aplicações inseridas em um mesmo contexto.



2. Software parcialmente completo projetado para ser instanciado.

# Frameworks

## ► Bibliotecas x Frameworks



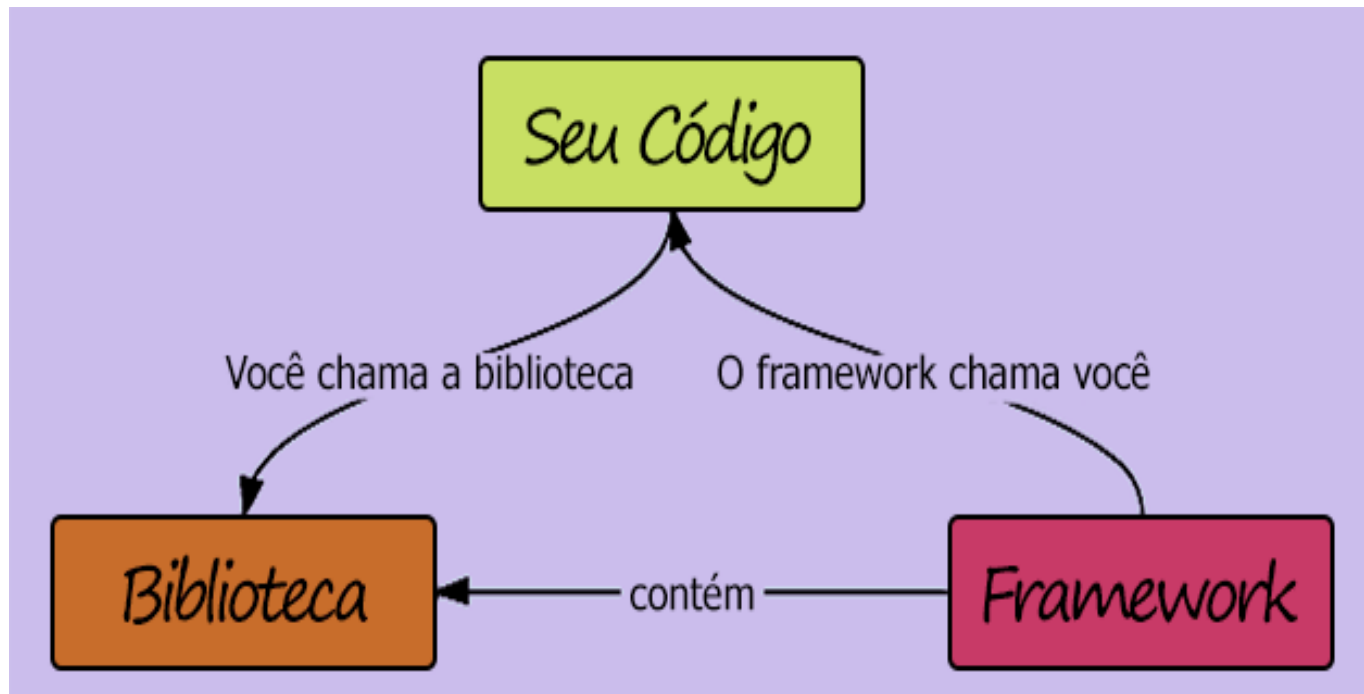
- Classes utilitárias independentes
- Modelo de colaboração não definido
- Aplicação faz chamada à biblioteca
- Reutilização Individual**

- Classes interligadas
- Modelo de colaboração pré-definido
- Framework faz chamada à aplicação
- Reutilização em conjunto**

# Frameworks

## ► Frameworks

- Esqueleto de uma aplicação
- Deve ser flexível e extensível

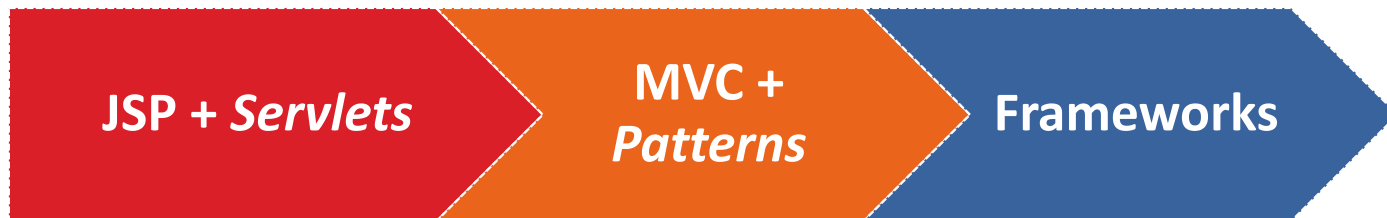


## Frameworks

---

### ► Frameworks WEB

#### ► Evolução

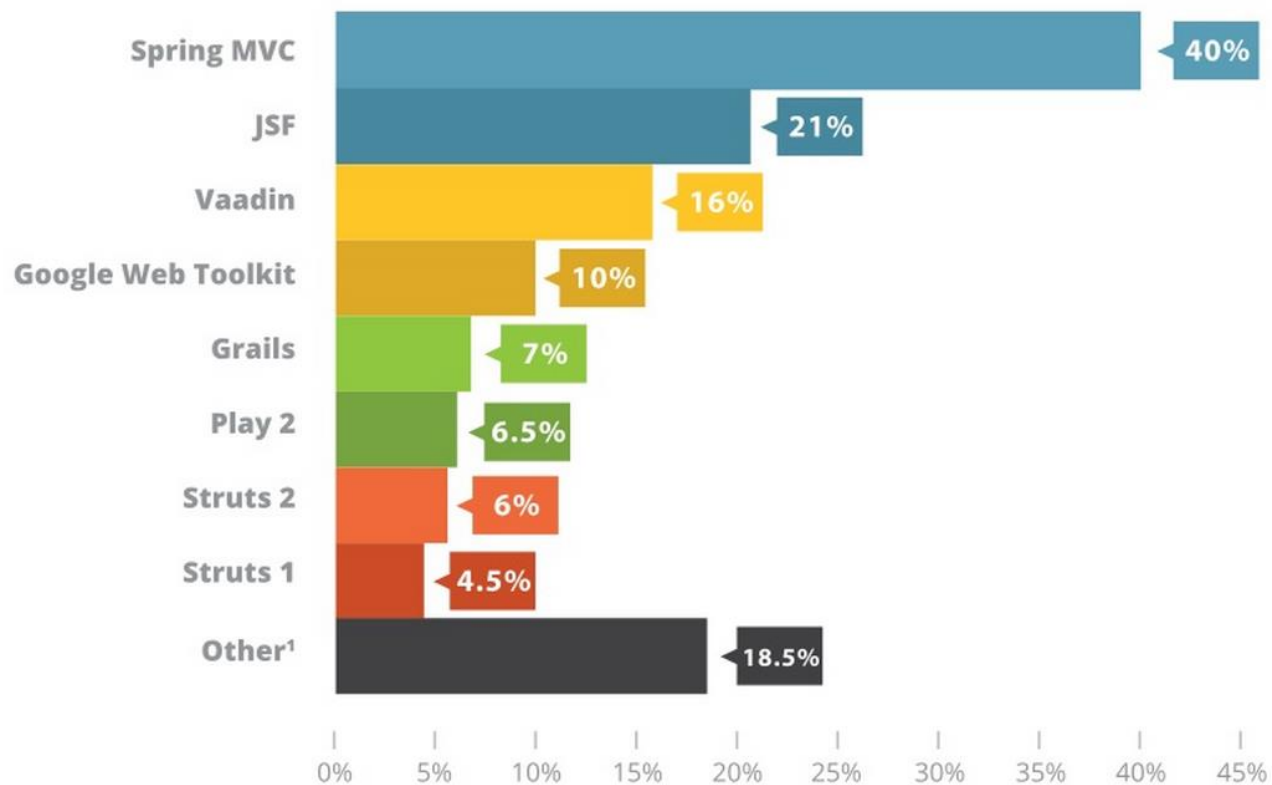


- Tornar o desenvolvimento web mais simples
  - Abstrair questões sobre arquitetura da aplicação
  - Manter foco na lógica de negócios

## Frameworks

### ► Frameworks WEB

**Web frameworks** in use \*



Fonte: 06 de janeiro de 2015

## Frameworks

- ▶ Criando o primeiro projeto SPRING
  - ▶ Netbeans – JAVA Web – SPRING Web MVC – Configuração

The screenshot shows the 'Frameworks' configuration window in NetBeans. At the top, it says 'Selecione os frameworks que você deseja utilizar em sua aplicação Web.' Below this is a list of frameworks with checkboxes: 'Spring Web MVC' (checked), 'JavaServer Faces', 'Struts 1.3.10', and 'Hibernate 4.3.1'. Below the list is a section titled 'Configuração Spring Web MVC' with two tabs: 'Bibliotecas' and 'Configuração'. The 'Configuração' tab is active, showing two text fields: 'Nome do emissor:' with the value 'dispatcher' and 'Mapeamento de emissores:' with the value '/'. A red dashed line is drawn across the top of the slide, passing behind the dialog box.

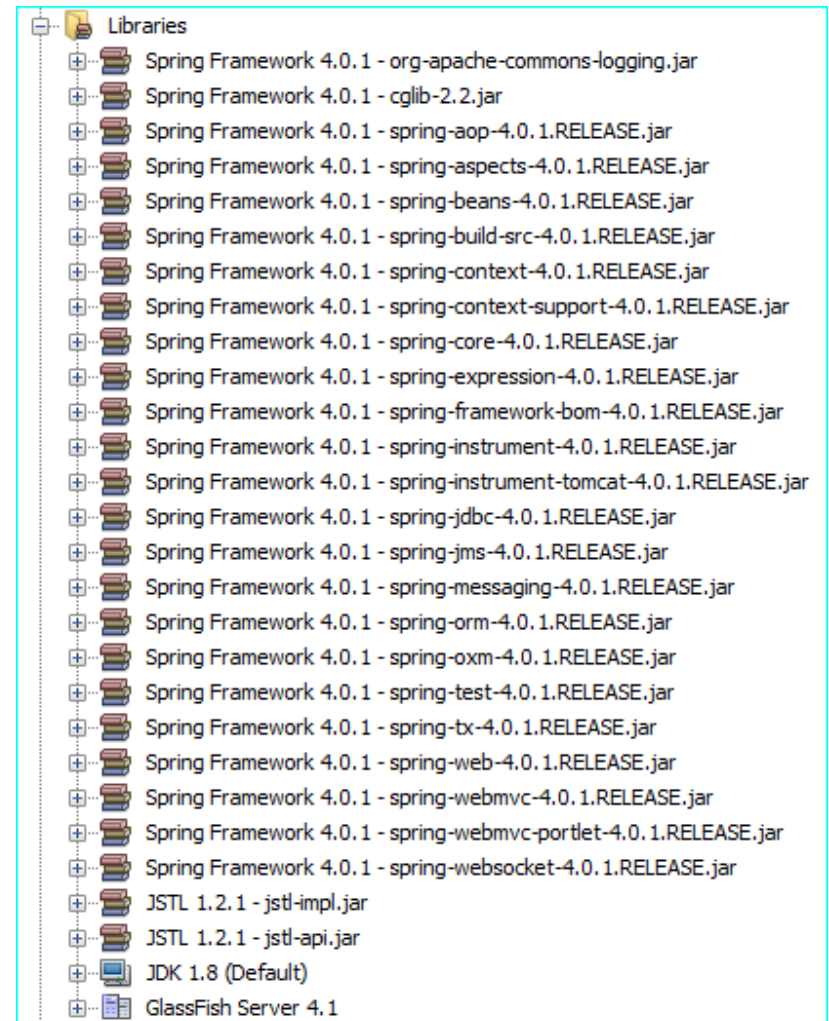
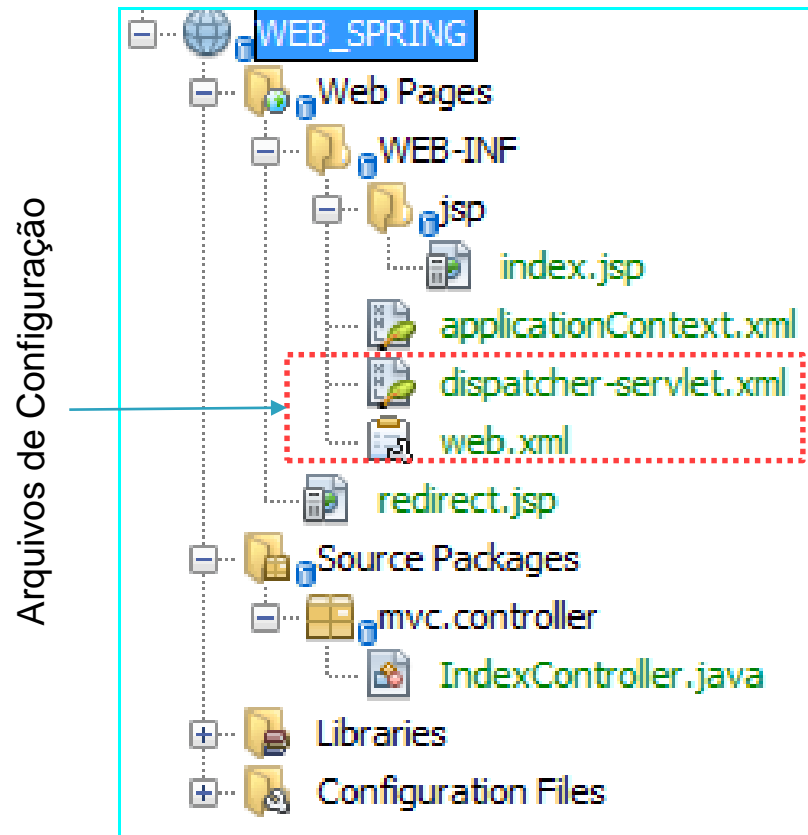
**Front-Controller**

**Quais requisições serão atendidas pelo *Front-Controller***

# Frameworks

## ► Projeto SPRING

### ► Estrutura do Projeto





## Frameworks

### ► Configurando Projeto SPRING

#### ► web.xml

##### ► XML de configuração do SPRING

```
10 <servlet>
11     <servlet-name>dispatcher</servlet-name>
12     <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
13     <load-on-startup>2</load-on-startup>
14 </servlet>
15 <servlet-mapping>
16     <servlet-name>dispatcher</servlet-name>
17     <url-pattern>/</url-pattern>
18 </servlet-mapping>
```

Indica qual classe fará o papel do  
*Front Controller*

Quais requisições serão atendidas  
pelo Front-Controller

## Frameworks

### ► Configurando Projeto SPRING

#### ► dispatcher-servlet.xml

incluir

```
<mvc:annotation-driven />  
<context:component-scan base-package="mvc.*" />
```

Define o pacote onde as suas classes Controller serão buscadas.

Define a pasta onde ficarão as Views

```
<bean id="viewResolver"  
      class="org.springframework.web.servlet.view.InternalResourceViewResolver"  
      p:prefix="/WEB-INF/jsp/"  
      p:suffix=".jsp" />
```

```
3 <beans xmlns="http://www.springframework.org/schema/beans"
4       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5       xmlns:p="http://www.springframework.org/schema/p"
6       xmlns:aop="http://www.springframework.org/schema/aop"
7       xmlns:tx="http://www.springframework.org/schema/tx"
8       xmlns:context="http://www.springframework.org/schema/context"
9       xmlns:mvc="http://www.springframework.org/schema/mvc"
10      xsi:schemaLocation="
11        http://www.springframework.org/schema/mvc
12        http://www.springframework.org/schema/mvc/spring-mvc-4.0.xsd
13        http://www.springframework.org/schema/context
14        http://www.springframework.org/schema/context/spring-context-4.0.xsd
15        http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans
16        http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop
17        http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx
18      " >
19
20     <mvc:annotation-driven />
21     <context:component-scan base-package="mvc.*" />
22
23     <bean id="viewResolver"
24           class="org.springframework.web.servlet.view.InternalResourceViewResolver"
25           p:prefix="/WEB-INF/jsp/"
26           p:suffix=".jsp" />
27
28 </beans>
```

Diagram illustrating the inclusion of Spring schemas in the XML configuration:

- The `xmlns:context` and `xmlns:mvc` attributes are highlighted with a dashed blue box.
- The `xsi:schemaLocation` attribute is highlighted with a dashed blue box.
- The `http://www.springframework.org/schema/mvc` and `http://www.springframework.org/schema/mvc/spring-mvc-4.0.xsd` URLs are highlighted with a dashed blue box.
- The `http://www.springframework.org/schema/context` and `http://www.springframework.org/schema/context/spring-context-4.0.xsd` URLs are highlighted with a dashed blue box.
- The `http://www.springframework.org/schema/beans` and `http://www.springframework.org/schema/beans` URLs are highlighted with a dashed blue box.
- The `http://www.springframework.org/schema/aop` and `http://www.springframework.org/schema/aop` URLs are highlighted with a dashed blue box.
- The `http://www.springframework.org/schema/tx` and `http://www.springframework.org/schema/tx` URLs are highlighted with a dashed blue box.

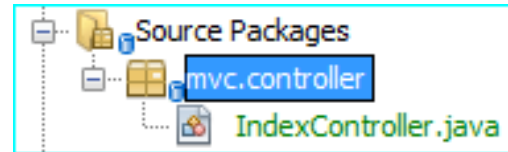
The word "inclure" (French for "include") is written next to the `xmlns:context` and `xmlns:mvc` attributes, with arrows pointing to the corresponding schema locations in the `xsi:schemaLocation` attribute.

## Frameworks

### ► Configurando Projeto SPRING

#### ► 1ª Lógica

- Crie a classe IndexController



```
6   package mvc.controller;
7
8   import org.springframework.stereotype.Controller;
9   import org.springframework.web.bind.annotation.RequestMapping;
10
11  @Controller          ← Indica que os métodos dessa classe
12  public class IndexController {          são ações'
13
14      @RequestMapping("/") ← Indica qual será a URL utilizada
15      public String index() {             para invocar o método
16          return "index"; ← Indica a View que será oferecida
17      }                                   como resposta: index.jsp
18  }
```

## Frameworks

---

### ► Projeto SPRING

- Incluindo uma nova lógica...
  - Modifique sua página index.jsp

```
<body>
  <a href="${pageContext.request.contextPath}/welcome">
    Welcome Link
  </a>
</body>
```

Requisição

index.jsp

## Frameworks

---

### ► Projeto SPRING

- Adicionando uma nova ação na classe IndexController

```
@RequestMapping("/welcome")  
public ModelAndView welcome() {  
    String viewName = "welcome";  
    String var = "message";  
    String content = "WELCOME TO SPRING WORLD!";  
    return new ModelAndView(viewName, var, content) ;  
}
```

IndexController

- Criar *View* welcome.jsp

```
<html>  
    <body>  
        <h1>${message}</h1>  
    </body>  
</html>
```

welcome.jsp

## Frameworks

---

### ► Projeto SPRING

#### ► Estudo de Caso – Tarefas

- Adicionar ao projeto o driver de conexão JDBC MySQL.

Tarefa
- id : Long - descricao : String - finalizado : boolean - dataFinalizacao : Calendar
+ adicionaTarefa(tarefa : Tarefa) : boolean

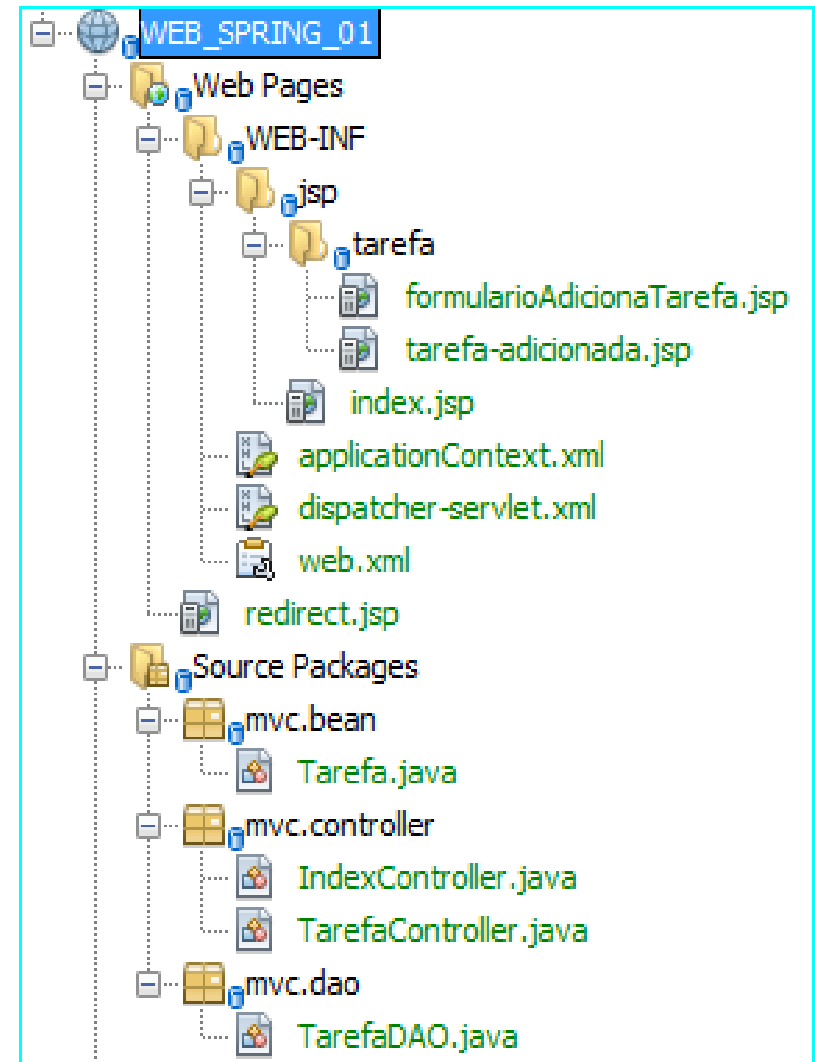
```
1 create database gestaoTarefas;  
2  
3 use gestaoTarefas;  
4  
5 create table tarefas (  
6 id BIGINT NOT NULL AUTO_INCREMENT,  
7 descricao VARCHAR(255),  
8 finalizado BOOLEAN,  
9 dataFinalizacao DATE,  
10 primary key (id)  
11 );
```

BD

## Frameworks

### ► Projeto SPRING

- Estudo de Caso – Tarefas
  - Estrutura

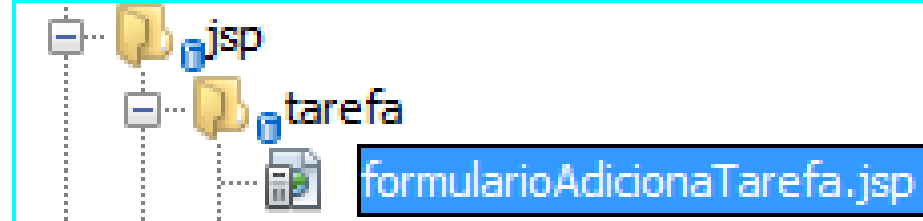






```
1 package mvc.bean;
2
3 import java.util.Calendar;
4
5 public class Tarefa {
6
7     private Long id;
8     private String descricao;
9     private boolean finalizado;
10    private Calendar dataFinalizacao;
11
12    public Tarefa() {}
13
14    public Long getId() {...3 lines }
15
16
17
18    public void setId(Long id) {...3 lines }
19
20
21
22    public String getDescricao() {...3 lines }
23
24
25
26    public void setDescricao(String descricao) {...3 lines }
27
28
29
30    public boolean isFinalizado() {...3 lines }
31
32
33
34    public void setFinalizado(boolean finalizado) {...3 lines }
35
36
37
38    public Calendar getDataFinalizacao() {...3 lines }
39
40
41
42    public void setDataFinalizacao(Calendar dataFinalizacao) {...3 lines }
43
44
45
46 }
```

## Frameworks



### ► Projeto SPRING

#### ► Estudo de Caso – Tarefas

```
6 <html>
7   <body>
8     <h1>Formulário de Adição de Tarefas</h1>
9     <form action="<c:url value='/adicionaTarefa'/" method="post">
10       Descrição
11       <br>
12       <textarea name="descricao" rows="5" cols="100" ></textarea>
13       <br>
14       <input type="submit" value="Adicionar">
15     </form>
16   </body>
17 </html>
```

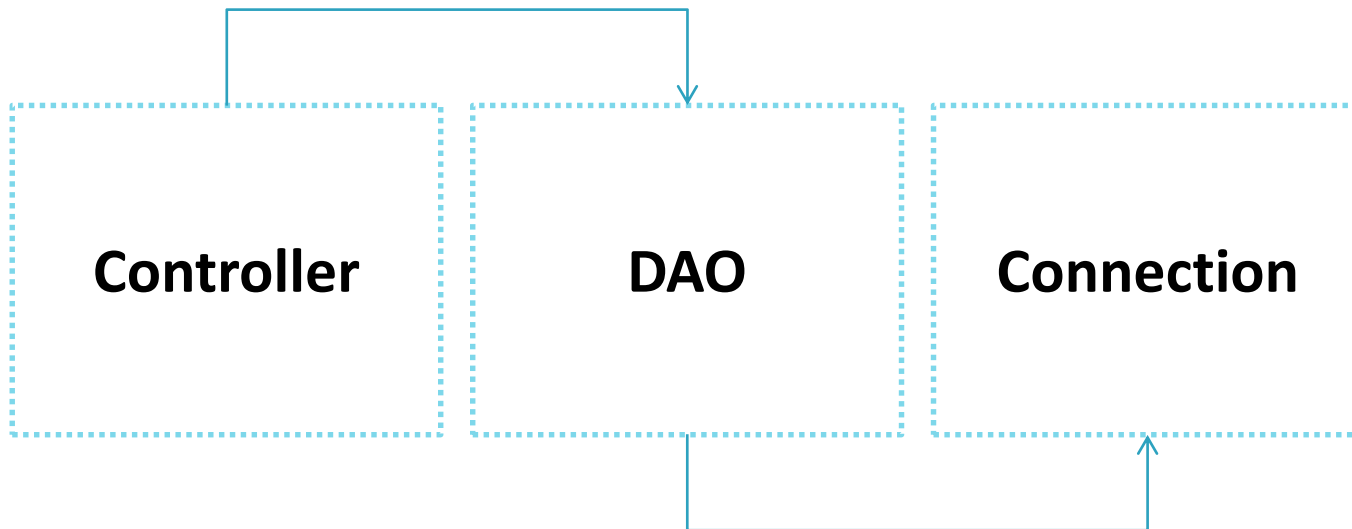
## Frameworks

---

### ► Projeto SPRING

#### ► Forte acoplamento

- Vários métodos são dependentes de uma ou mais classes.
- A aplicação cria dependência chamando diretamente uma classe específica.



## Frameworks

---

### ► Projeto SPRING


#### ► Forte acoplamento – Exemplo

- Vários métodos da classe dependem de JdbcTarefaDao

- ☐ Criação de objetos DAO em cada método do Controller.

```
@Controller
public class TarefasController {

    @RequestMapping("adicionaTarefa")
    public String adiciona(Tarefa tarefa) {
        JdbcTarefaDao dao = new JdbcTarefaDao();
        dao.adiciona(tarefa);
        return "tarefa/adicionada";
    }
}
```



## Frameworks

---

### ► Projeto SPRING

#### ► Forte acoplamento – Exemplo

- Acoplamento com a classe ConnectionFactory.
- A classe está com a responsabilidade de resolver a dependência, mas apenas precisamos da conexão,

```
public class JdbcTarefaDao {  
  
    private final Connection connection;  
  
    public JdbcTarefaDao() {  
        try {  
            this.connection = new ConnectionFactory().getConnection();  
        } catch (SQLException e) {  
            throw new RuntimeException(e);  
        }  
    }  
  
    //métodos omitidos  
}
```

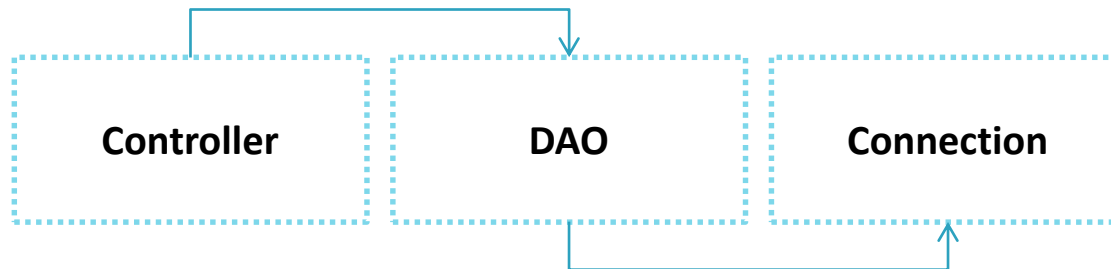


## Frameworks

---

### ► Projeto SPRING

#### ► Menos acoplamento



- Padrão de Projeto – *Dependency Injection* (DI) – Injeção de Dependência
  - ❑ Retira da classe a responsabilidade de resolver suas dependências e a delega para algum container que será o responsável por controlar os objetos que ligam a dependência.
- *Spring – Container DI*
  - ❑ Resolve e liga as dependências

## ► Projeto SPRING

### ► @Autowired

Injeção de dependência

Avisa ao SPRING que ele precisa resolver e injetar a dependência



```
14 @Controller
15 public class TarefaController {
16
17     private final TarefaDAO dao;
18
19     @Autowired
20     public TarefaController(TarefaDAO dao) {
21         this.dao = dao;
22     }
23
24     @RequestMapping("/formAdicionaTarefa")
25     public String form() {
26         return "tarefa/formularioAdicionaTarefa";
27     }
28
29     @RequestMapping("/adicionaTarefa")
30     public String adiciona(Tarefa tarefa) {
31         dao.adicionaTarefa(tarefa);
32         return "tarefa/tarefa-adicionada";
33     }
34
35 }
```

## Frameworks

### ► Projeto SPRING

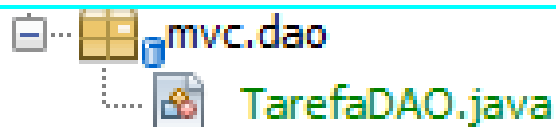
#### ► @Repository

Classes do tipo DAO

Permite que o SPRING  
instancie a classe

#### ► DataSource

Provedor de Conexão



```
16 @Repository
17 public class TarefaDAO {
18
19     private final Connection connection;
20
21     @Autowired
22     public TarefaDAO(DataSource dataSource) {
23         try {
24             this.connection = dataSource.getConnection();
25         } catch (SQLException ex) {
26             throw new RuntimeException(ex);
27         }
28     }
29
30     public boolean adicionaTarefa(Tarefa tarefa) {
31         String sql = "insert into tarefas (descricao) values (?)";
32         try (
33             // prepared statement para inserção
34             PreparedStatement stmt = connection.prepareStatement(sql)) {
35             // seta os valores
36             stmt.setString(1, tarefa.getDescricao());
37             // executa
38             stmt.execute();
39         } catch (SQLException e) {
40             throw new RuntimeException(e);
41         }
42         return true;
43     }
44 }
```



## Frameworks

### ► Projeto SPRING

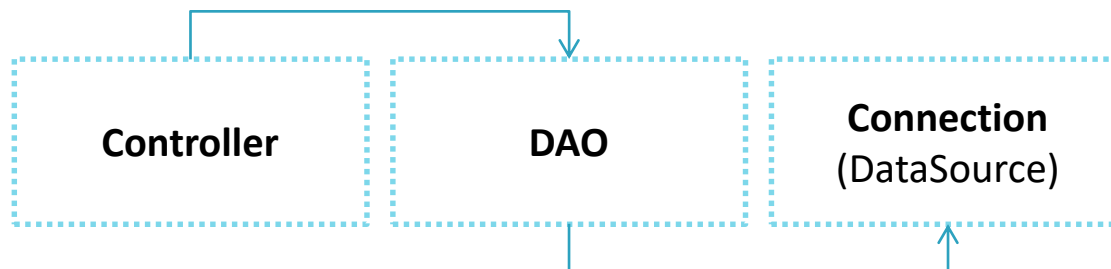
#### ► DataSource

##### ► XML do SPRING

```
<bean id="dataSource"
      class="org.springframework.jdbc.datasource.DriverManagerDataSource"
      p:driverClassName="com.mysql.jdbc.Driver"
      p:url="jdbc:mysql://localhost/gestaotarefas"
      p:username="root"
      p:password="" />
```

Configurações de acesso ao BD

dispatcher-servlet.xml



## Frameworks

---

### ► Projeto SPRING

- Validando formulários no lado do servidor

## Formulário de Adição de Tarefas

Descrição

A Descrição deve ser preenchida!

A Descrição deve conter [10-100] caracteres

Adicionar

## Frameworks

---

### ► Projeto SPRING

#### ► Validando Formulários – Jeito Não Recomendado

```
@RequestMapping("adicionaTarefa")
public String adiciona(Tarefa tarefa) {

    if(tarefa.getDescricao() == null || tarefa.getDescricao().equals("")) {
        return "tarefa/formulario";
    }

    JdbcTarefaDao dao = new JdbcTarefaDao();
    dao.adiciona(tarefa);
    return "tarefa/adicionada";
}
```



## Frameworks

---

### ► Projeto SPRING

#### ► Validando Formulários – Bean Validation

##### ► Validação da Camada de Modelo

- Regras de validação por meio de anotações

```
8 public class Tarefa {
9
10     private Long id;
11
12     @NotEmpty(message="{tarefa.descricao.vazia}")
13     @Size(min=10, max = 100, message="{tarefa.descricao.limite}")
14     private String descricao;
15
16     private boolean finalizado;
17
18     @DateTimeFormat(pattern = "yyyy-MM-dd")
19     private Calendar dataFinalizacao;
20
21     // ...
```

bean: Tarefa

Annotation	Example	Description
@NotEmpty	@NotEmpty private String firstName;	Checks if: object != null
@NotNull	@NotNull private Boolean isStudent;	Checks if: object != null and object size >= 1
@Size	@Size(min=1,max=20) private String lastName;	Checks for min and max size
@DateTimeFormat	@DateTimeFormat(pattern = "MM/dd/yyyy") private Date dob;	The custom pattern to use to format the field. This is used for display purpose on the form too.
@NumberFormat	@NumberFormat(pattern="#.##") private Double weight;	The custom pattern to format the Double value.
@Pattern	@Pattern(regexp="^[A-Za-z]*\$") private String lastName;	Support to Regular expressions. In this example restricts to lower and upper case characters only.
@NotBlank	@NotEmpty private String firstName;	Checks the size after trimming the String. Do not use to check for null.

## Frameworks

---

### ► Projeto SPRING

#### ► Validando Formulários – Bean Validation

##### ► ValidationMessages.properties

- ❑ Deve ser salvo dentro da pasta src

```
tarefa.descricao.vazia = A Descrição deve ser preenchida!  
tarefa.descricao.limite = A Descrição deve conter [{min}-{max}] caracteres
```

ValidationMessages.properties

## Frameworks

---

### ► Projeto SPRING

#### ► Validando Formulários – Bean Validation

##### ► Validação da Camada de Controle

- ❑ `@Valid`
- ❑ `BindingResult`

*ConstraintViolationException*

```
@RequestMapping("/adicionaTarefa")
public String adiciona(@Valid Tarefa tarefa, BindingResult result){

    if(result.hasErrors()){
        return "tarefa/formularioAdicionaTarefa";
    }

    dao.adicionaTarefa(tarefa);
    return "tarefa/tarefa-adicionada";
}
```

TarefaController

## Frameworks

### ► Projeto SPRING

#### ► Validando Formulários – Bean Validation

##### ► Validação da Camada de Visão : **form:errors**

```
5  <@taglib uri="http://www.springframework.org/tags/form" prefix="form" %>
6  <html>
7  <body>
8      <h1>Formulário de Adição de Tarefas</h1>
9      <form action="<c:url value='/adicionaTarefa'/'>" method="post">
10         <fmt:message key="tarefa.descricao"/>
11         <br>
12         <textarea name="descricao" rows="5" cols="100" ></textarea>
13         <br>
14         <form:errors path="tarefa.descricao" cssStyle="color:red"/>
15         <br>
16         <input type="submit" value="Adicionar">
17     </form>
18 </body>
19 </html>
```



## Frameworks

---

- ▶ Projeto SPRING
  - ▶ Validando Formulários

### Formulário de Adição de Tarefas

Descrição

A Descrição deve ser preenchida!

A Descrição deve conter [10-100] caracteres

Adicionar

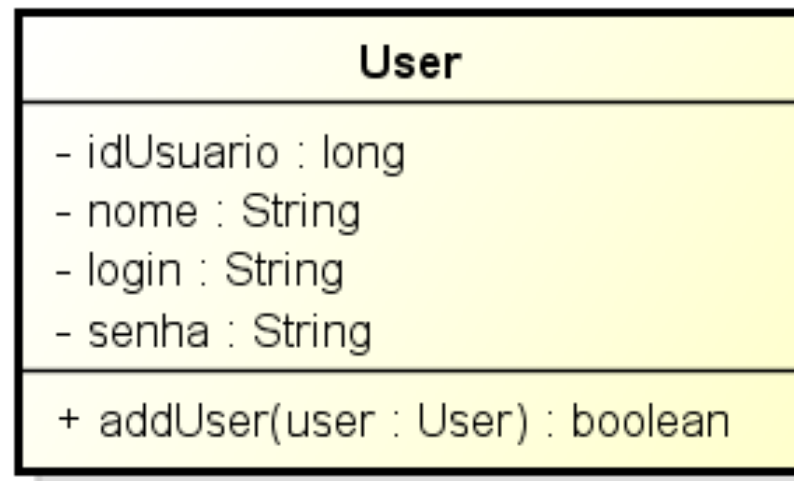
## Frameworks



### ► Prática

#### ► Utilizando o *Framework* SPRING

- Crie uma página responsável por receber informações referentes a um usuário e inserir no banco de dados.
- Implementar validação de formulário na camada de controle.



## Frameworks

---

### ► Projeto SPRING

#### ► Listar Tarefas

```
@RequestMapping("/listaTarefas")  
public String lista(Model model){  
    model.addAttribute("listaTarefas", dao.listarTarefas());  
    return "tarefa/listagem-tarefas";  
}
```

TarefaController

```
50 public List<Tarefa> listarTarefas() {  
51     List<Tarefa> listaTarefas = new ArrayList<Tarefa>();  
52     String sql = "select * from tarefas order by descricao";  
53     try (  
54         PreparedStatement stmt = connection.prepareStatement(sql)) {  
55         ResultSet rs = stmt.executeQuery();  
56         while (rs.next()) {  
57             Tarefa tarefa = new Tarefa();  
58             tarefa.setId(rs.getLong("id"));  
59             tarefa.setDescricao(rs.getString("descricao"));  
60             tarefa.setFinalizado(rs.getBoolean("finalizado"));  
61             //montando data  
62             Calendar data = Calendar.getInstance();  
63             if(rs.getDate("dataFinalizacao") != null)  
64             {  
65                 data.setTime(rs.getDate("dataFinalizacao"));  
66                 tarefa.setDataFinalizacao(data);  
67             }  
68             listaTarefas.add(tarefa);  
69         }  
70     } catch (SQLException e) {  
71         throw new RuntimeException(e);  
72     }  
73     return listaTarefas;  
74 }
```

```
7 <body>
8 <a href="<c:url value='/' />">Voltar</a> <br><br>
9 <table>
10 <tr>
11 <th>Id</th>
12 <th>Descrição</th>
13 <th>Finalizada?</th>
14 <th>Data de Finalização</th>
15 </tr>
16 <c:forEach items="${listaTarefas}" var="tarefa">
17 <tr>
18 <td>${tarefa.id}</td>
19 <td>${tarefa.descricao}</td>
20 <c:if test="${tarefa.finalizado eq false}">
21 <td>Não finalizada</td>
22 </c:if>
23 <c:if test="${tarefa.finalizado eq true}">
24 <td>Finalizada</td>
25 </c:if>
26 <td>
27 <fmt:formatDate value="${tarefa.dataFinalizacao.time}" pattern="dd/MM/yyyy"/>
28 </td>
29 </tr>
30 </c:forEach>
31 </table>
32 </body>
```

## Frameworks

### ► Projeto SPRING

- Redirecionando Requisições
  - Remover Tarefa

Id	Descrição	Finalizada?	Data de Finalização	Ação
37	Estudar Framework Spring.	Não finalizada		<a href="#">Alterar</a> <a href="#">Remover</a>
38	Estudar JavaScript.	Finalizada	18/12/2015	<a href="#">Alterar</a> <a href="#">Remover</a>
36	Estudar Modelo MVC.	Não finalizada		<a href="#">Alterar</a> <a href="#">Remover</a>

```
<a href="removeTarefa?id=${tarefa.id}"  
    onclick="return confirm('Deseja realmente excluir?')" > Remover </a>
```

## Frameworks

---

### ► Projeto SPRING

#### ► Redirecionando Requisições

##### ► Remover Tarefa

□ redirect

```
<a href="removeTarefa?id=${tarefa.id}"  
  onclick="return confirm('Deseja realmente excluir?')" > Remover </a>
```

```
@RequestMapping("/removeTarefa")  
public String remove(Long id) {  
    dao.removerTarefa(id);  
    return "redirect:/listaTarefas";  
}
```

TarefaController

## Frameworks

---

### ► Projeto SPRING

- Redirecionando Requisições
  - Remover Tarefa

```
public boolean removerTarefa(Long id){  
    String sql = "delete from tarefas where id = ? ";  
    try (  
        PreparedStatement stmt = connection.prepareStatement(sql)) {  
        stmt.setLong(1,id);  
        stmt.execute();  
    } catch (SQLException e) {  
        throw new RuntimeException(e);  
    }  
    return true;  
}
```

TarefaDao



## Frameworks

### ► Projeto SPRING

- Redirecionando Requisições
  - Alterar Tarefa

Id	Descrição	Finalizada?	Data de Finalização	Ação
37	Estudar Framework Spring.	Não finalizada		<a href="#">Alterar</a> <a href="#">Remover</a>
38	Estudar JavaScript.	Finalizada	18/12/2015	<a href="#">Alterar</a> <a href="#">Remover</a>
36	Estudar Modelo MVC.	Não finalizada		<a href="#">Alterar</a> <a href="#">Remover</a>

```
<a href="exibeTarefa?id=${tarefa.id}"> Alterar </a>
```

## Frameworks

---

### ► Projeto SPRING

- Redirecionando Requisições
  - Alterar Tarefa

```
<a href="exibeTarefa?id=${tarefa.id}"> Alterar </a>
```

```
@RequestMapping("/exibeTarefa")  
public String exibe(Long id, Model model){  
    model.addAttribute("tarefa", dao.buscarTarefaPorId(id));  
    return "tarefa/exibe-tarefa";  
}
```

TarefaController

## Frameworks

---

### ► Projeto SPRING

- Redirecionando Requisições
  - Alterando Tarefas

**Alterar tarefa - 37**  
Descrição:  

Estudar Framework Spring.

  
Finalizado? ☒ Em: 

18/12/2015

Alterar

exibe-tarefa.jsp

```
76 public Tarefa buscarTarefaPorId(Long id) {
77     String sql = "select * from tarefas where id = ? ";
78     try (
79         PreparedStatement stmt = connection.prepareStatement(sql)) {
80         stmt.setLong(1, id);
81         ResultSet rs = stmt.executeQuery();
82         Tarefa tarefa = new Tarefa();
83         if(rs.next()) {
84             tarefa.setId(rs.getLong("id"));
85             tarefa.setDescricao(rs.getString("descricao"));
86             tarefa.setFinalizado(rs.getBoolean("finalizado"));
87             //montando data
88             Calendar data = Calendar.getInstance();
89             if(rs.getDate("dataFinalizacao") != null)
90             {
91                 data.setTime(rs.getDate("dataFinalizacao"));
92                 tarefa.setDataFinalizacao(data);
93             }
94         }
95         return tarefa;
96     } catch (SQLException e) {
97         throw new RuntimeException(e);
98     }
99 }
```

<body>

<h3>Alterar tarefa - \${tarefa.id}</h3>

<form action="<c:url value='/alteraTarefa'/'>" method="post">

<input type="hidden" name="id" value="\${tarefa.id}" />

Descrição:

<br><form:errors path="tarefa.descricao" cssStyle="color:red"/> <br>

<textarea name="descricao" cols="50" rows="5">\${tarefa.descricao}</textarea>

<br>

Finalizado?

<input type="checkbox" name="finalizado"

value="true" \${tarefa.finalizado?'checked':''}/>

Em:

<form:errors path="tarefa.dataFinalizacao" cssStyle="color:red"/>

<fmt:formatDate var="fmtDate" value="\${tarefa.dataFinalizacao.time}"

pattern="yyyy-MM-dd"/>

<input type="date" name="dataFinalizacao" value="\${fmtDate}" >

<br><br>

<input type="submit" value="Alterar"/>

<br><br>

</form>

<a href="<c:url value='/'/'>">Voltar</a>

</body>

exibe-tarefa.jsp

<body>

<h3>Alterar tarefa - \${tarefa.id}</h3>

<form action="<c:url value='/alteraTarefa'/>" method="post">

<input type="hidden" name="id" value="\${tarefa.id}" />

Descrição:

<br><form:errors path="tarefa.descricao" cssStyle="color:red"/> <br>

<textarea name="descricao" cols="50" rows="5">\${tarefa.descricao}</textarea>

<br>

Finalizado?

<input type="checkbox" name="finalizado"

value="true" \${tarefa.finalizado?'checked':''}/>

Em:

<form:errors path="tarefa.dataFinalizacao" cssStyle="color:red"/>

<fmt:formatDate var="fmtDate" value="\${tarefa.dataFinalizacao.time}"  
pattern="yyyy-MM-dd"/>

<input type="date" name="dataFinalizacao" value="\${fmtDate}" >

<br><br>

<input type="submit" value="Alterar"/>

<br><br>

</form>

<a href="<c:url value='/' />">Voltar</a>

</body>

<body>

<h3>Alterar tarefa - \${tarefa.id}</h3>

<form action="<c:url value='/alteraTarefa'/'>" method="post">

<input type="hidden" name="id" value="\${tarefa.id}" />

Descrição:

<br><form:errors path="tarefa.descricao" cssStyle="color:red"/> <br>

<textarea name="descricao" cols="50" rows="5">\${tarefa.descricao}</textarea>

<br>

Finalizado?

<input type="checkbox" name="finalizado"

value="true" \${tarefa.finalizado?'checked':''}/>

Em:

<form:errors path="tarefa.dataFinalizacao" cssStyle="color:red"/>

<fmt:formatDate var="fmtDate" value="\${tarefa.dataFinalizacao.time}"  
pattern="yyyy-MM-dd"/>

<input type="date" name="dataFinalizacao" value="\${fmtDate}" >

<br><br>

<input type="submit" value="Alterar"/>

<br><br>

</form>

<a href="<c:url value='/'/'>">Voltar</a>

</body>

<body>

<h3>Alterar tarefa - \${tarefa.id}</h3>

<form action="<c:url value='/alteraTarefa'/'>" method="post">

<input type="hidden" name="id" value="\${tarefa.id}" />

Descrição:

<br><form:errors path="tarefa.descricao" cssStyle="color:red"/> <br>

<textarea name="descricao" cols="50" rows="5">\${tarefa.descricao}</textarea>

<br>

Finalizado?

<input type="checkbox" name="finalizado"

value="true" \${tarefa.finalizado?'checked':''}/>

Em:

<form:errors path="tarefa.dataFinalizacao" cssStyle="color:red"/>

<fmt:formatDate var="fmtDate" value="\${tarefa.dataFinalizacao.time}"  
pattern="yyyy-MM-dd"/>

<input type="date" name="dataFinalizacao" value="\${fmtDate}" >

<br><br>

<input type="submit" value="Alterar"/>

<br><br>

</form>

<a href="<c:url value='/'/'>">Voltar</a>

</body>



## Frameworks

---

### ► Projeto SPRING

- Redirecionando Requisições
  - Alterando Tarefas

```
@RequestMapping("/alteraTarefa")
public String altera(@Valid Tarefa tarefa, BindingResult result){
    if(result.hasErrors()){
        return "tarefa/exibe-tarefa";
    }
    dao.alterarTarefa(tarefa);
    return "redirect:/listaTarefas";
}
```

TarefaController

```
114 public boolean alterarTarefa(Tarefa tarefa) {
115     String sql = "update tarefas set descricao = ?,"
116         + " finalizado=?, dataFinalizacao=? where id = ? ";
117     try {
118         PreparedStatement stmt = connection.prepareStatement(sql) {
119             stmt.setString(1,tarefa.getDescricao());
120             stmt.setBoolean(2,tarefa.isFinalizado());
121             if(tarefa.getDataFinalizacao() != null){
122                 stmt.setDate(3, new Date(tarefa.getDataFinalizacao().getTimeInMillis()));
123             }else{
124                 stmt.setNull(3, java.sql.Types.DATE);
125             }
126             stmt.setLong(4,tarefa.getId());
127             stmt.execute();
128         } catch (SQLException e) {
129             throw new RuntimeException(e);
130         }
131         return true;
132     }
```

## Frameworks

### ► Projeto SPRING

#### ► Melhorando a Usabilidade

- Como carregar apenas o conteúdo que mudou com a nova requisição?

Id	Descrição	Finalizada?	Data de Finalização	Ação
37	Estudar Framework Spring.	Não finalizada		<a href="#">Alterar</a> <a href="#">Remover</a>

Id	Descrição	Finalizada?	Data de Finalização	Ação
37	Estudar Framework Spring.	Não finalizada <a href="#">Finalizar agora?</a>		<a href="#">Alterar</a> <a href="#">Remover</a>

```
<a href="#" onclick="finalizaAgora(${tarefa.id})"> Finalizar agora? </a>
```

## Frameworks

### ► Projeto SPRING

#### ► Melhorando a Usabilidade

##### ► AJAX - *Asynchronous Javascript and XML*

- ❑ Requisições assíncronas
- ❑ Utiliza árvore **DOM** para modificar o conteúdo sem recarregar a página
  - ❑ Economia de tráfego de internet e melhora da experiência do usuário
- ❑ Faz requisições utilizando um objeto Javascript e não um formulário

Extensible Markup Language

Linguagem Extensível de Marcação Genérica



Requisição Síncrona



- Requisição I
- Requisição II
- Requisição III

Requisição Assíncrona



## Frameworks

### ► Projeto SPRING

#### ► Melhorando a Usabilidade

##### ► AJAX

- Marcação das regiões específicas que se deseja alterar

```
<c:if test="${tarefa.finalizado eq false}">
<td id="tarefa_${tarefa.id}">
    Não finalizada <br>
    <a href="#" onclick="finalizaAgora(${tarefa.id})"> Finalizar agora? </a>
</td>
</c:if>
<td id="tarefa_data_${tarefa.id}">
    <fmt:formatDate value="${tarefa.dataFinalizacao.time}" pattern="dd/MM/yyyy"/>
</td>
```

Id	Descrição	Finalizada?	Data de Finalização	Ação
37	Estudar Framework Spring.	Não finalizada <a href="#">Finalizar agora?</a>		<a href="#">Alterar</a> <a href="#">Remover</a>

## Frameworks

---

### ► Projeto SPRING

#### ► Melhorando a Usabilidade

##### ► AJAX

- Suporte *jQuery* para trabalhar com AJAX

```
function finalizaAgora(id) {  
    $.post("finalizaTarefa", {'id' : id}, function(dadosDeResposta) {  
        // selecionando o elemento html através da  
        // ID e alterando o HTML dele  
        $("#tarefa_"+id).html("Finalizado");  
        $("#tarefa_data_"+id).html(dadosDeResposta);  
    });  
}
```

default.js

## Frameworks

---

### ► Projeto SPRING

#### ► Melhorando a Usabilidade

##### ► AJAX

```
@RequestMapping("/finalizaTarefa")
public void finaliza(Long id, HttpServletResponse response) {
    dao.finalizarTarefa(id);
    response.setStatus(200); //msg de sucesso
}
```

```
@RequestMapping("/finalizaTarefa")
public String finaliza(Long id, Model model) {
    dao.finalizarTarefa(id);
    model.addAttribute("tarefa", dao.buscarTarefaPorId(id));
    return "tarefa/dataFinalizada";
}
```

## Frameworks

---

- ▶ Projeto SPRING
  - ▶ Melhorando a Usabilidade
    - ▶ AJAX - jquery

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/fmt" prefix="fmt" %>  
  
<fmt:formatDate value="${tarefa.dataFinalizacao.time}" pattern="dd/MM/yyyy" />
```

dataFinalizada.jsp



## Frameworks

---

### ► Projeto SPRING

#### ► Melhorando a Usabilidade

##### ► AJAX - jquery

```
public boolean finalizarTarefa(Long id){  
    String sql = " update tarefas set finalizado=true, "  
                + " dataFinalizacao=now() where id = ? ";  
    try {  
        PreparedStatement stmt = connection.prepareStatement(sql);  
        stmt.setLong(1, id);  
        stmt.execute();  
    } catch (SQLException e) {  
        throw new RuntimeException(e);  
    }  
    return true;  
}
```

TarefaDao

## Frameworks

---

### ► Autenticação e Autorização

- Acesso a conteúdo restrito
- Protocolo HTTP não mantém informações sobre quem é quem entre uma conexão e outra



## Frameworks

---

### ▶ Autenticação e Autorização

#### ▶ **Cookie**

##### ▶ Par de strings armazenado no cliente

- ❑ Limitações com relação ao cliente utilizado
- ❑ Não é muito confiável
- ❑ Não é possível marcar um cliente com um objeto, apenas com Strings.

##### ▶ Funcionamento

- ❑ Um cookie salvo no cliente é enviado de volta ao servidor toda vez que o cliente efetuar uma nova requisição
- ❑ Assim, o servidor consegue identificar aquele cliente sempre com os dados que o cookie enviar

## Frameworks

---

### ▶ Autenticação e Autorização

#### ▶ Sessão

- ▶ É o tempo que o usuário permanece ativo no sistema
  - A cada página visitada, o tempo de sessão é zerado
  - Existe um tempo limite que define a duração da sessão
- ▶ Permite manter objetos associados a um cliente
- ▶ Independente de cliente
- ▶ Se os cookies estiverem desabilitados, a sessão não funcionará
  - Solução: técnica url-rewriting

## Frameworks

### ► Projeto SPRING – Login da Aplicação

#### ► Formulário de Login

```
7 <html>
8   <head>
9     <link rel="stylesheet" href="<c:url value='/resources/css/default.css' />"
10  </head>
11  <body>
12    <h2>Página de Login</h2>
13    <div class="msgErro">${msgLoginInvalido}</div><br>
14    <form action="<c:url value='/efetuaLogin' />" method="post">
15      Login: <input type="text" name="login" /> <br><br>
16      Senha: <input type="password" name="senha" /> <br><br>
17      <input type="submit" value="Entrar" />
18      <input type="button" value="Voltar" onclick="history.back()" />
19    </form>
20  </body>
21 </html>
```

formularioLogin.jsp

## Frameworks

---

- ▶ Projeto SPRING – Login da Aplicação
  - ▶ Banco de Dados

```
CREATE TABLE `usuarios` (  
  `id` int(11) NOT NULL,  
  `login` varchar(15) NOT NULL,  
  `senha` varchar(8) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

## Frameworks

---

### ► Projeto SPRING – Login da Aplicação

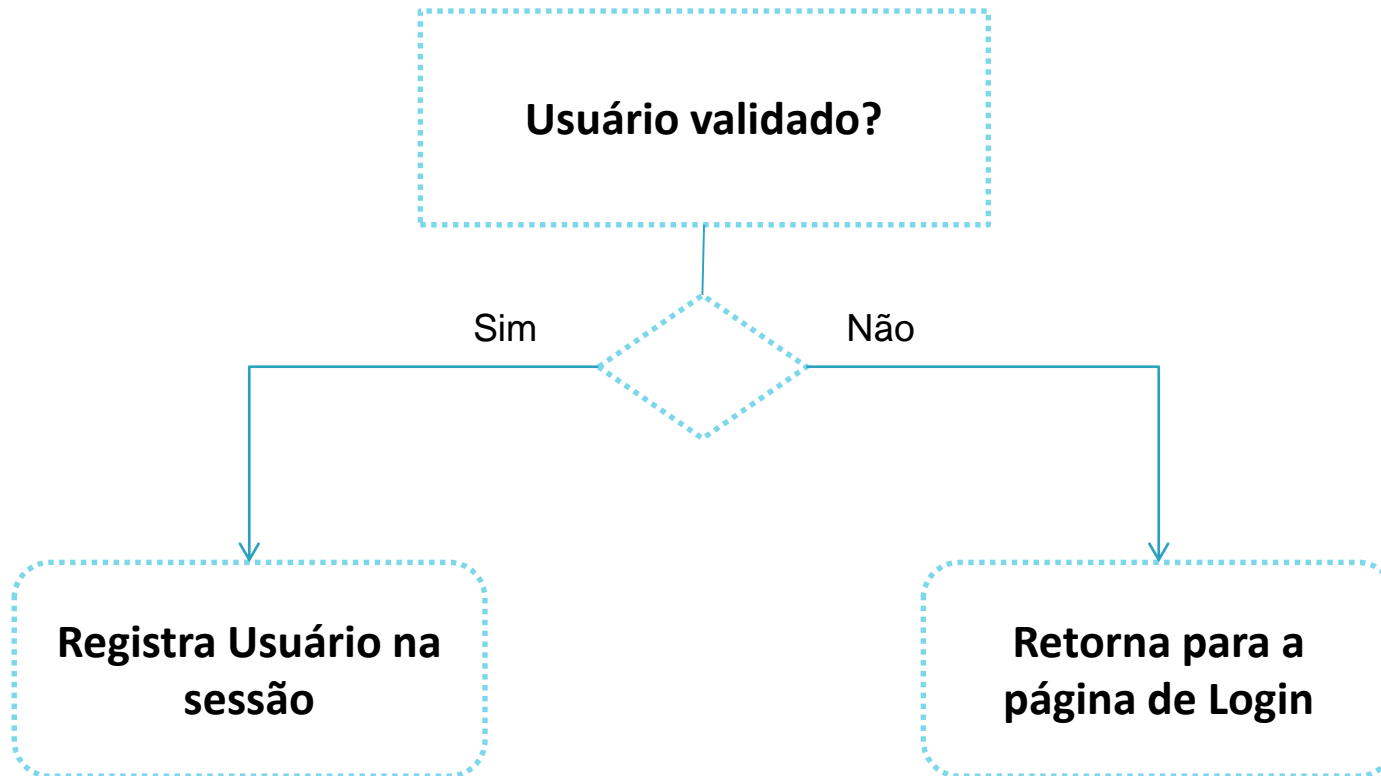
#### ► Serialização

- Salvar, Gravar e Capturar o estado atual de um objeto
- Alguns *servlet containers* como o Glassfish só permitem gravar na sessão objetos serializados

```
public class Usuario implements Serializable{  
  
    private int id;  
    private String login;  
    private String senha;  
  
    public Usuario() {}  
}
```

## Frameworks

### ► Projeto SPRING – Login da Aplicação





## Frameworks

---

### ► Projeto SPRING – Login da Aplicação

#### ► HttpSession

- Objeto referente à sessão
- Map<String, Object>
- É possível capturar a sessão em qualquer método
  - Para isso, coloca-se o HttpSession como parâmetro

```
@Controller
public class LoginController {

    public String efetuaLogin(HttpSession session) {
        //....
    }
}
```

## Frameworks

---

### ► Projeto SPRING – Login da Aplicação

#### ► Validação de Usuário

```
34      @RequestMapping("/efetuaLogin")
35      public String efetuaLogin(Usuario user, HttpSession session){
36          if(dao.validaUsuario(user)){
37              session.setAttribute("usuarioLogado", user);
38              session.removeAttribute("msgLoginInvalido");
39              return "menuAdm";
40          }else{
41              session.setAttribute("msgLoginInvalido", "O login não foi validado!");
42              return "redirect:formLogin";
43          }
44      }
45  }
```

LoginController

## Frameworks

---

- ▶ Projeto SPRING – Login da Aplicação
  - ▶ Validação de Usuário

```
<div class="msgErro">${msgLoginInvalido}</div><br>
```

### Página de Login

O login não foi validado!

Login:

Senha:

## Frameworks

### ► Projeto SPRING – Login da Aplicação

#### ► Validação de Usuário

```
10 <html>
11   <body>
12     <h1>Funcionalidades restritas aos administradores apenas!!</h1>
13     Bem-Vindo(a), ${usuarioLogado.login}!
14     <br><br>
15     <a href="<c:url value='/logout'/'>">Sair</a>
16     <a href="<c:url value='/'/'>">Voltar</a>
17   </body>
18 </html>
```

**Funcionalidades restritas aos administradores apenas!!**

Bem-Vindo(a), jhoyce!

[Sair](#) [Voltar](#)

## Frameworks

---

### ► Projeto SPRING – Login da Aplicação

#### ► Bloqueando acesso indevido

##### ► Interceptadores

- ❑ Funcionam como Filtros
- ❑ Implementar a interface HandlerInterceptor
  - ❑ Métodos: preHandle, postHandle e afterCompletion
- ❑ SPRING MVC
  - ❑ HandlerInterceptorAdapter: estender e sobrescrever o método de interesse
- ❑ Recuperando sessão

```
HttpSession session = request.getSession();
```

```
session.getAttribute("usuarioLogado")
```

```
17 public class AutorizadorInterceptor extends HandlerInterceptorAdapter {
18
19     @Override
20     public boolean preHandle(HttpServletRequest request,
21                             HttpServletResponse response,
22                             Object controller) throws Exception {
23
24         String uri = request.getRequestURI();
25
26         if( uri.endsWith("/") ||
27            uri.endsWith("formLogin") ||
28            uri.endsWith("efetuaLogin") ||
29            uri.endsWith("listaTarefas") ||
30            uri.contains("resources")) {
31             return true;
32         }
33
34         HttpSession session = request.getSession();
35
36         if(session.getAttribute("usuarioLogado") != null) {
37             return true;
38         }
39
40         response.sendRedirect("formLogin");
41
42         return false;
43     }
44 }
```

## Frameworks

---

### ► Projeto SPRING – Login da Aplicação

#### ► Bloquando acesso indevido

##### ► Interceptadores

- ❑ Registrar o interceptador
- ❑ O Spring MVC não permite esse registro via anotações, então usaremos a configuração via XML

```
<!-- Filtro de autorização e autenticação-->  
<mvc:interceptors>  
    <bean class="mvc.interceptor.AutorizadorInterceptor" />  
</mvc:interceptors>
```

dispatcher\_servlet.xml

## *Frameworks*

---

# EXTRAS



## Frameworks

---

### ► Projeto SPRING

#### ► Internacionalização

```
<fmt:message key="tarefa.descricao" />
```

```
<!-- path dos properties de mensagens -->
<bean id="messageSource"
      class="org.springframework.context.support.ReloadableResourceBundleMessageSource">
  <property name="basename" value="/WEB-INF/mensagens" />
</bean>
```

dispatcher-servlet.xml

```
tarefa.descricao=Descrição
```

mensagens\_pt.properties

```
tarefa.descricao=Description
```

mensagens\_en.properties

## Frameworks

---

### ► Projeto SPRING

#### ► Internacionalização

```
<!-- linguagem padrão -->
<bean id="localeResolver" class="org.springframework.web.servlet.i18n.CookieLocaleResolver">
    <property name="defaultLocale" value="pt"/>
</bean>

<!-- Changes the locale when a 'lang' request parameter is sent; e.g. /?lang=pt -->
<mvc:interceptors>
    <bean class="org.springframework.web.servlet.i18n.LocaleChangeInterceptor"
        p:paramName="lang" />
</mvc:interceptors>
```

dispatcher-servlet.xml

# Frameworks

## ► Projeto SPRING

### ► Internacionalização

#### ► Validação da Camada de Visão : **form:errors**

```
5  <%@taglib uri="http://www.springframework.org/tags/form" prefix="form" %>
6  <html>
7  <body>
8      <h1>Formulário de Adição de Tarefas</h1>
9      <form action="<c:url value='/adicionaTarefa'/>" method="post">
10         <fmt:message key="tarefa.descricao"/>
11         <br>
12         <textarea name="descricao" rows="5" cols="100" ></textarea>
13         <br>
14         <form:errors path="tarefa.descricao" cssStyle="color:red"/>
15         <br>
16         <input type="submit" value="Adicionar">
17     </form>
18 </body>
19 </html>
```

## Frameworks

---

### ► Projeto SPRING

#### ► Listar Tarefas

<b>Id</b>	<b>Descrição</b>	<b>Finalizada?</b>	<b>Data de Finalização</b>
19	Ser dedicado!	Não finalizada	
21	Ser determinado!	Não finalizada	

## Frameworks

---

### ▶ EXTRAS

- ▶ Incluindo Recursos Externos [css, js, imgs]

```
<head>  
    <link rel="stylesheet" href="<c:url value='/resources/css/default.css'/>" >  
</head>
```

```
<!-- caminho de recursos [css, js, imgs] -->  
<mvc:resources mapping="/resources/**" location="/resources/" />
```

dispatcher-servlet.xml

## Frameworks

---

### ► Autenticação e Autorização

#### ► Sessão

- Configurando o tempo limite – web.xml

```
<session-config>
  <session-timeout>
    30
  </session-timeout>
</session-config>
```