

JSF e PrimeFaces

Universidade Federal da Bahia
MATC84 - Laboratório de Programação Web
Renato Silva
Tiago Gonçalves

Agenda

❑ Parte teórica

❑ JSF

- ❑ O que é?
- ❑ Histórico
- ❑ Arquitetura

❑ PrimeFaces

- ❑ Por que usá-lo?
- ❑ Existe PrimeFaces sem JSF?
- ❑ Outros frameworks

Agenda

❑ Parte prática

- ❑ Configuração de ambiente
 - ❑ Criando projeto Java EE
 - ❑ Estrutura do projeto
 - ❑ Verificando se está tudo ok
- ❑ O primeiro contato com o JSF
- ❑ CRUD - Agenda de Contatos
 - ❑ Criando arquitetura MVC

Agenda

❑ Parte prática

- ❑ CRUD - Agenda de Contatos
 - ❑ Criando arquitetura MVC
 - ❑ Entendendo os escopos do JSF
 - ❑ Criando a primeira tela
 - ❑ Utilizando os componentes JSF
 - ❑ Enriquecendo a tela com o PrimeFaces

Parte téorica

JSF

O que é?

- ❑ É um framework MVC.
- ❑ Construção de UI baseadas em componentes.
- ❑ Programação dirigida a eventos.

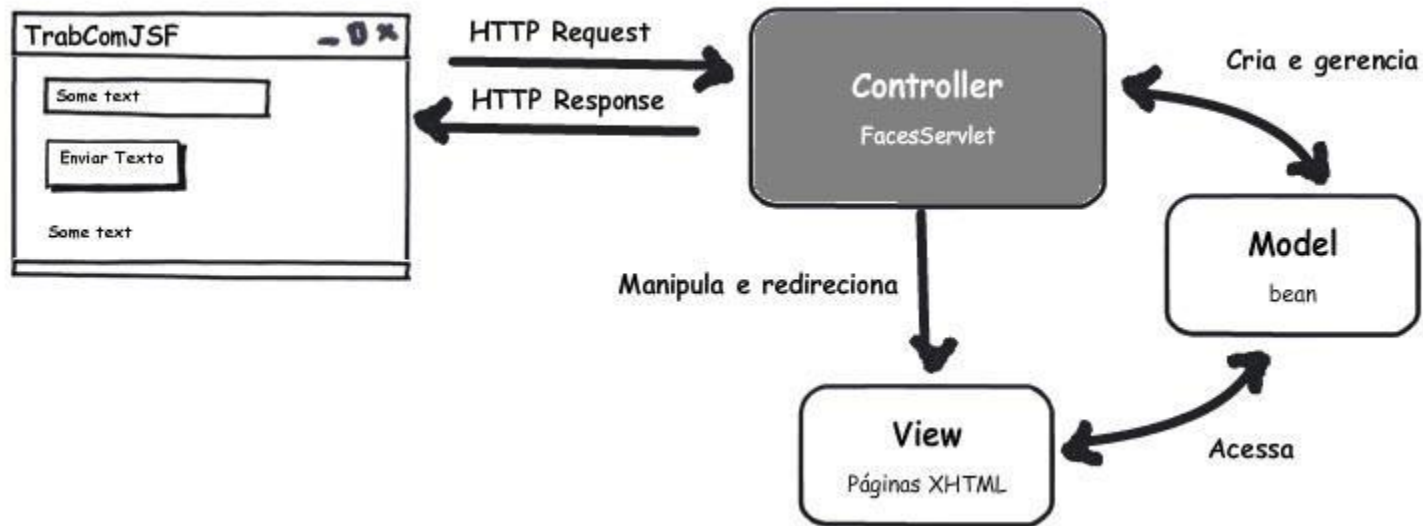
JSF

Histórico

- ❑ Foi criado através do Java Community Process
 - ❑ Sun Microsystems, Oracle, Borland, BEA, IBM
- ❑ O processo de especificação teve início em 2001
 - ❑ JSF 1.0(2004-03-11) — (DEPRECATED) – JSR 127
 - ❑ JSF 1.1(2004-05-27) — (DEPRECATED) – JSR 127
 - ❑ JSF 1.2(2006-05-11) – JSR 252
 - ❑ JSF 2.0(2009-06-28) – JSR 314
 - ❑ JSF 2.1(2010-10-22) – JSR 314
 - ❑ JSF 2.2(2013-05-21) – JSR 344

JSF

Arquitetura



PrimeFaces

Por que usá-lo?

❑ Simplicidade e produtividade

- ❑ Aplicações com comportamento **AJAX**.
- ❑ Sensação **desktop** em um ambiente WEB.

❑ Fácil uso

- ❑ "Um componente UI bom deve ocultar a complexidade, mas manter a flexibilidade".

❑ Comunidade forte

- ❑ Feedback, novas idéias, *bug reports* e *patches*.

PrimeFaces

Existe PrimeFaces sem JSF?



PrimeFaces

Outros frameworks



PrimeFaces

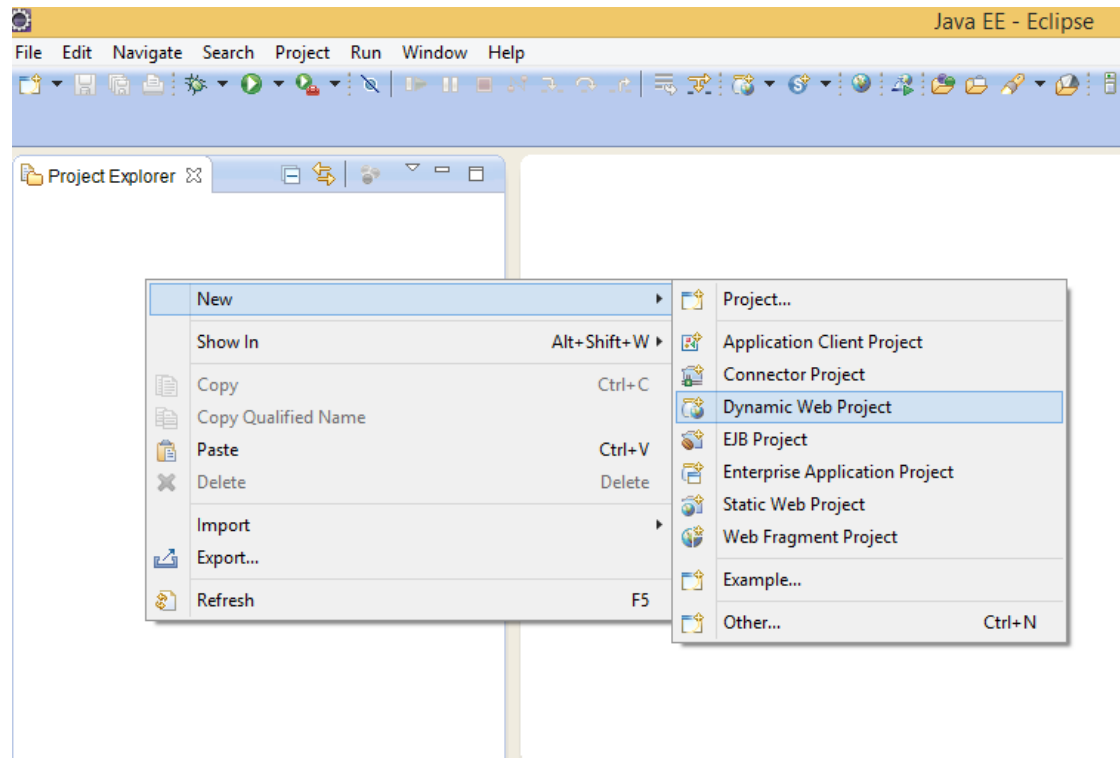
Outros frameworks - comparativo



Parte prática

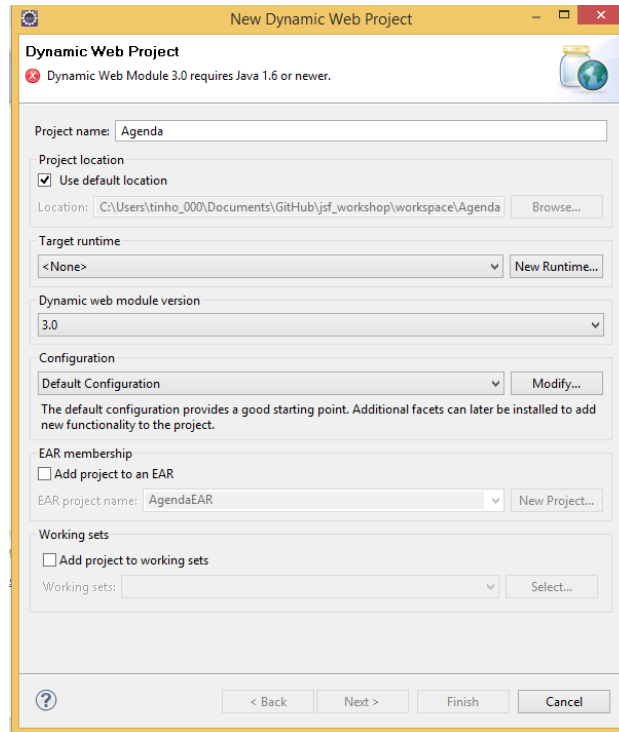
Configuração de ambiente

Criando projeto Java EE



Configuração de ambiente

Criando projeto Java EE



The screenshot shows the 'New Dynamic Web Project' dialog box in the Eclipse IDE. The dialog has a title bar 'New Dynamic Web Project' and a subtitle 'Dynamic Web Project'. A warning icon and message state: 'Dynamic Web Module 3.0 requires Java 1.6 or newer.' The 'Project name' field is set to 'Agenda'. The 'Project location' section has the 'Use default location' checkbox checked, and the 'Location' field shows the path 'C:\Users\tinho_000\Documents\GitHub\jsf_workshop\workspace\Agenda'. The 'Target runtime' is set to '<None>' with a 'New Runtime...' button. The 'Dynamic web module version' is set to '3.0'. The 'Configuration' section shows 'Default Configuration' with a 'Modify...' button. Below this, a note states: 'The default configuration provides a good starting point. Additional facets can later be installed to add new functionality to the project.' The 'EAR membership' section has the 'Add project to an EAR' checkbox unchecked, and the 'EAR project name' field is set to 'AgendaEAR' with a 'New Project...' button. The 'Working sets' section has the 'Add project to working sets' checkbox unchecked, and the 'Working sets' field is empty with a 'Select...' button. At the bottom, there are navigation buttons: '< Back', 'Next >', 'Finish', and 'Cancel'.

New Dynamic Web Project

Dynamic Web Project

Dynamic Web Module 3.0 requires Java 1.6 or newer.

Project name:

Project location

☒ Use default location

Location:

Target runtime

Dynamic web module version

Configuration

The default configuration provides a good starting point. Additional facets can later be installed to add new functionality to the project.

EAR membership

☐ Add project to an EAR

EAR project name:

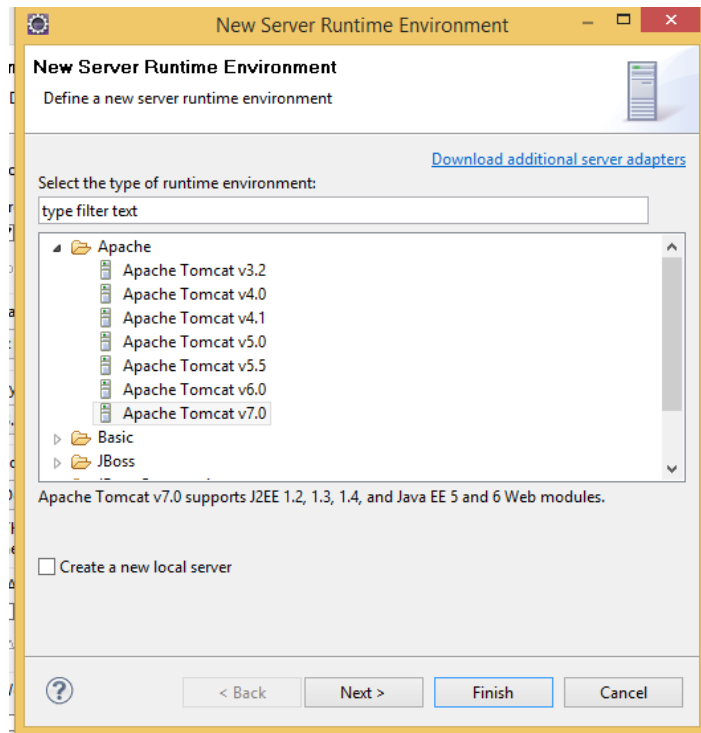
Working sets

☐ Add project to working sets

Working sets:

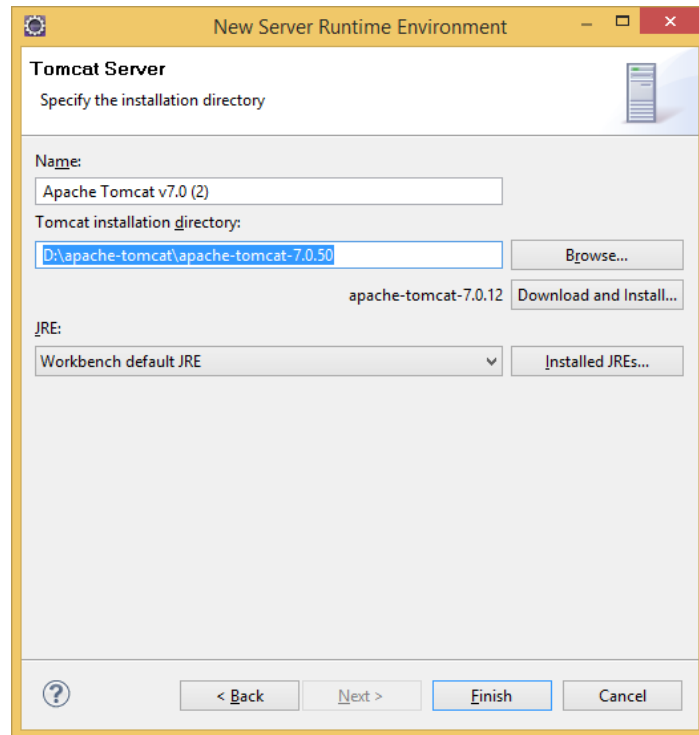
Configuração de ambiente

Criando projeto Java EE



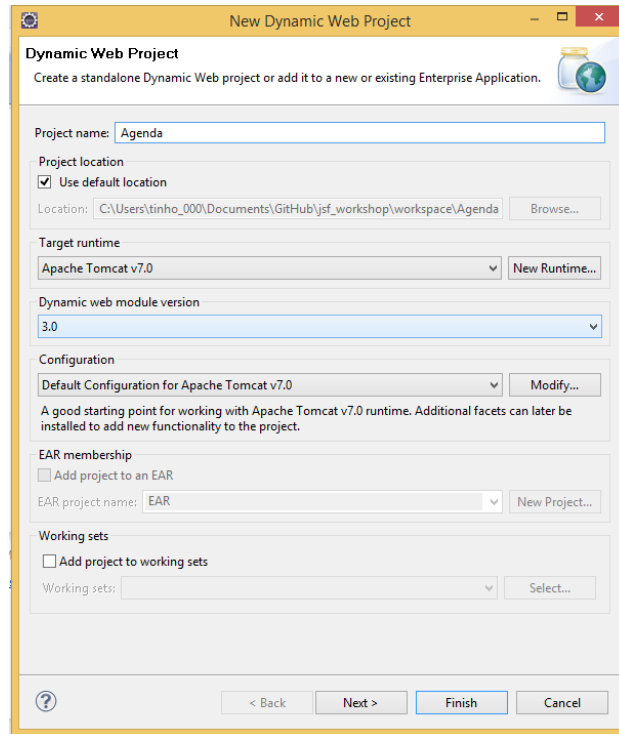
Configuração de ambiente

Criando projeto Java EE



Configuração de ambiente

Criando projeto Java EE



The screenshot shows the 'New Dynamic Web Project' dialog box. The title bar reads 'New Dynamic Web Project'. The main heading is 'Dynamic Web Project' with a subtitle 'Create a standalone Dynamic Web project or add it to a new or existing Enterprise Application.' and a small icon of a jar and globe. The dialog is divided into several sections: 'Project name' with a text field containing 'Agenda'; 'Project location' with a checked 'Use default location' checkbox and a text field showing the path 'C:\Users\tinho_000\Documents\GitHub\jsf_workshop\workspace\Agenda'; 'Target runtime' with a dropdown menu set to 'Apache Tomcat v7.0' and a 'New Runtime...' button; 'Dynamic web module version' with a dropdown menu set to '3.0'; 'Configuration' with a dropdown menu set to 'Default Configuration for Apache Tomcat v7.0' and a 'Modify...' button; 'EAR membership' with an unchecked 'Add project to an EAR' checkbox and an 'EAR project name' field set to 'EAR'; and 'Working sets' with an unchecked 'Add project to working sets' checkbox. At the bottom, there are four buttons: '< Back', 'Next >', 'Finish', and 'Cancel'.

New Dynamic Web Project

Dynamic Web Project
Create a standalone Dynamic Web project or add it to a new or existing Enterprise Application.

Project name:

Project location
☒ Use default location
Location:

Target runtime

Dynamic web module version

Configuration

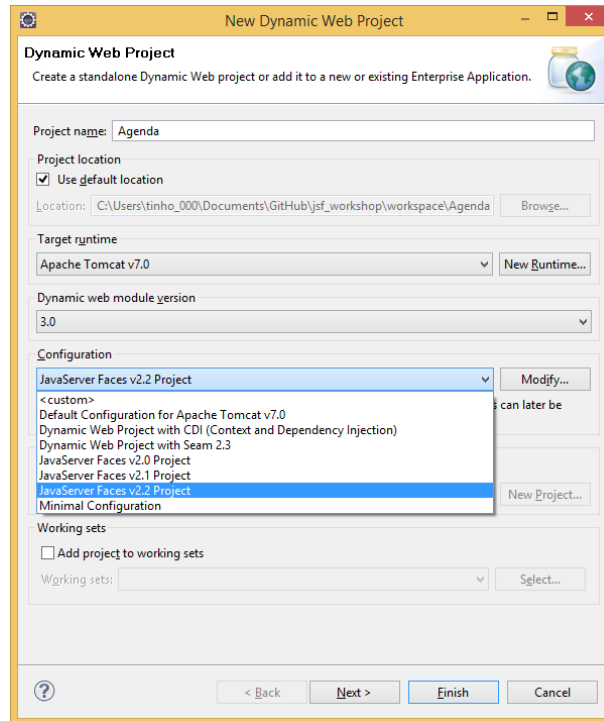
A good starting point for working with Apache Tomcat v7.0 runtime. Additional facets can later be installed to add new functionality to the project.

EAR membership
☐ Add project to an EAR
EAR project name:

Working sets
☐ Add project to working sets
Working sets:

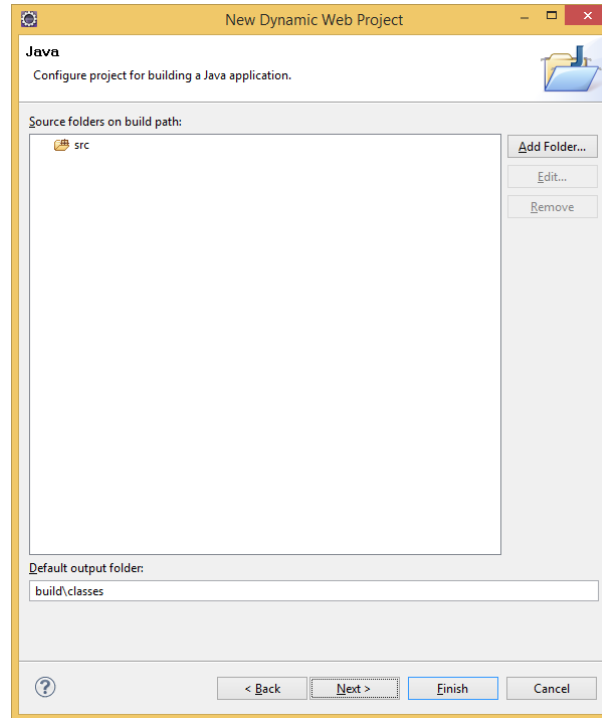
Configuração de ambiente

Criando projeto Java EE



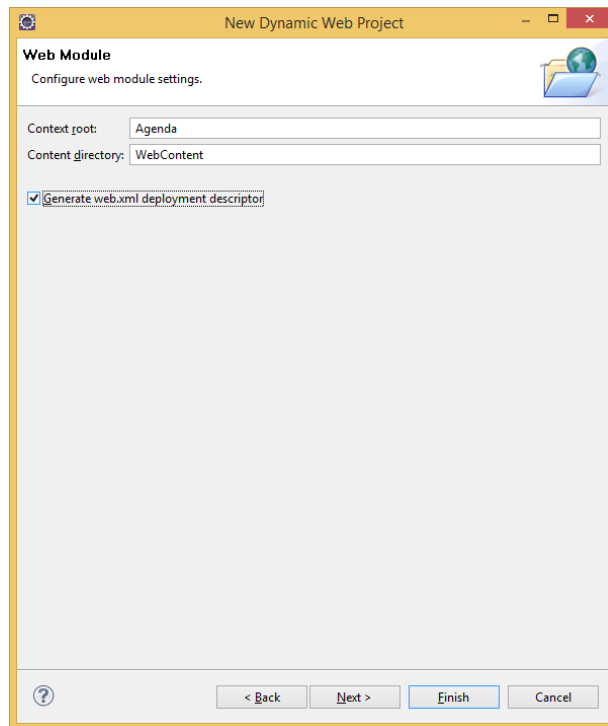
Configuração de ambiente

Criando projeto Java EE



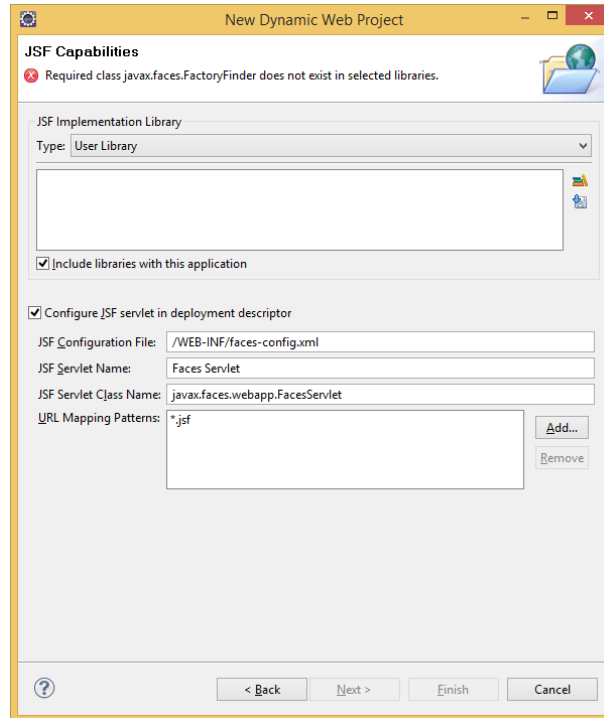
Configuração de ambiente

Criando projeto Java EE



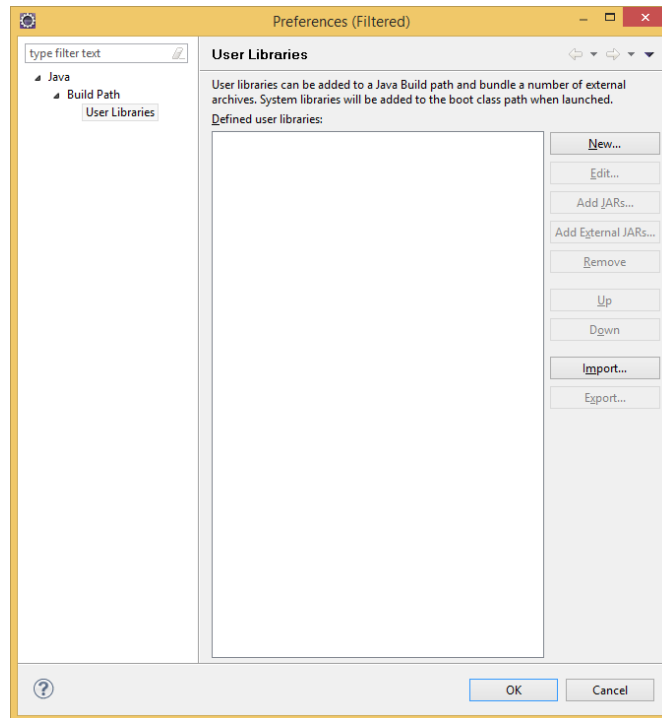
Configuração de ambiente

Criando projeto Java EE



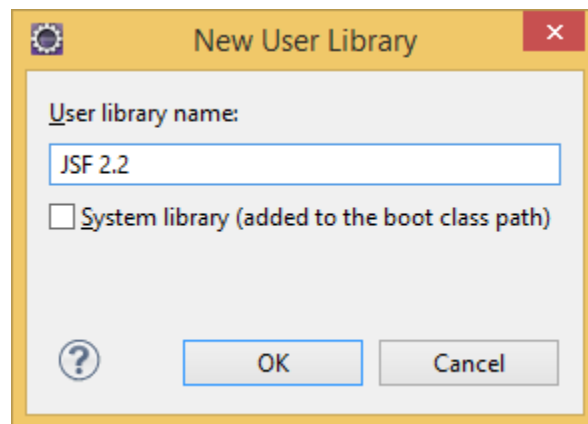
Configuração de ambiente

Criando projeto Java EE



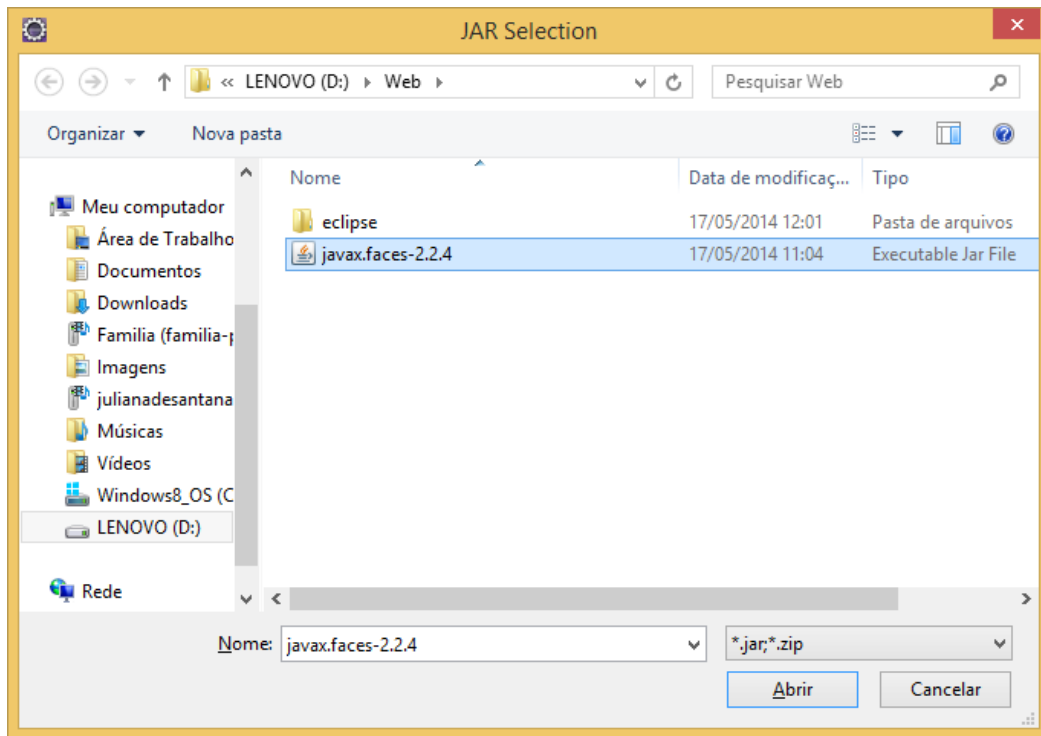
Configuração de ambiente

Criando projeto Java EE



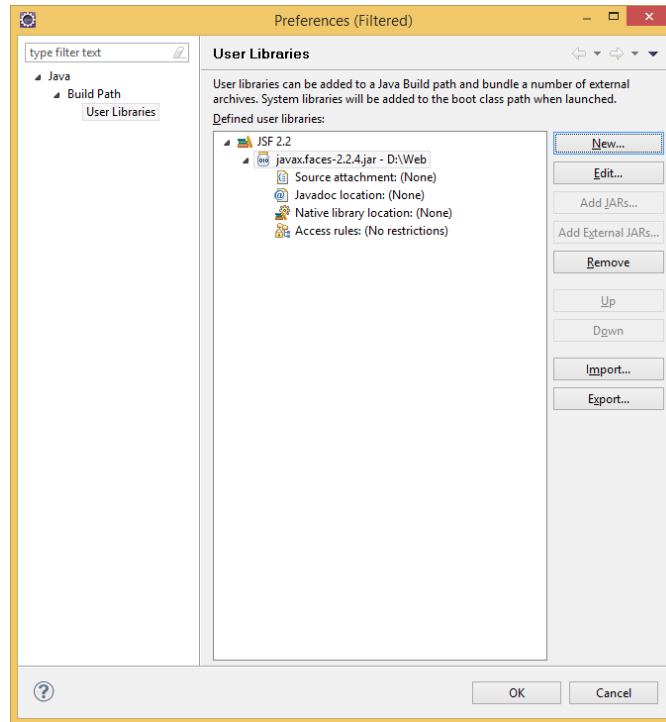
Configuração de ambiente

Criando projeto Java EE



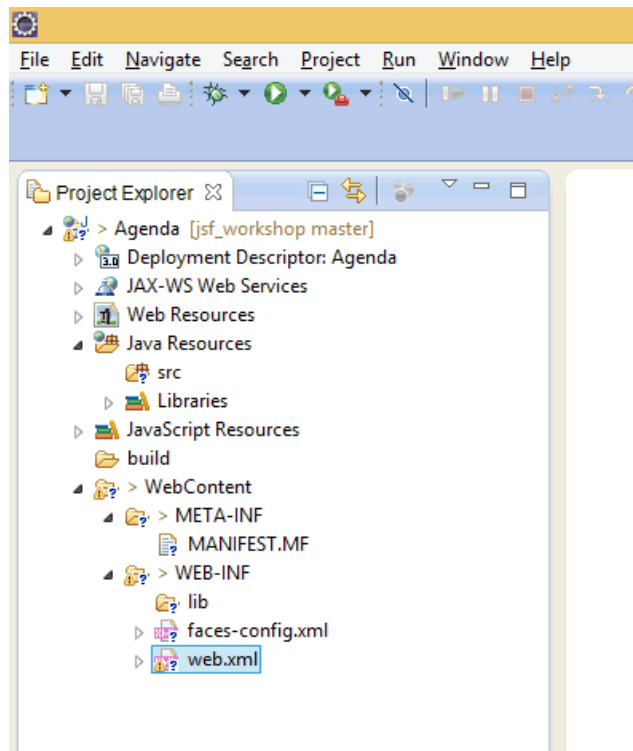
Configuração de ambiente

Criando projeto Java EE



Configuração de ambiente

Estrutura do projeto



Configuração de ambiente

Verificando se está tudo ok

Executemos o projeto pela primeira vez!

O primeiro contato com o JSF

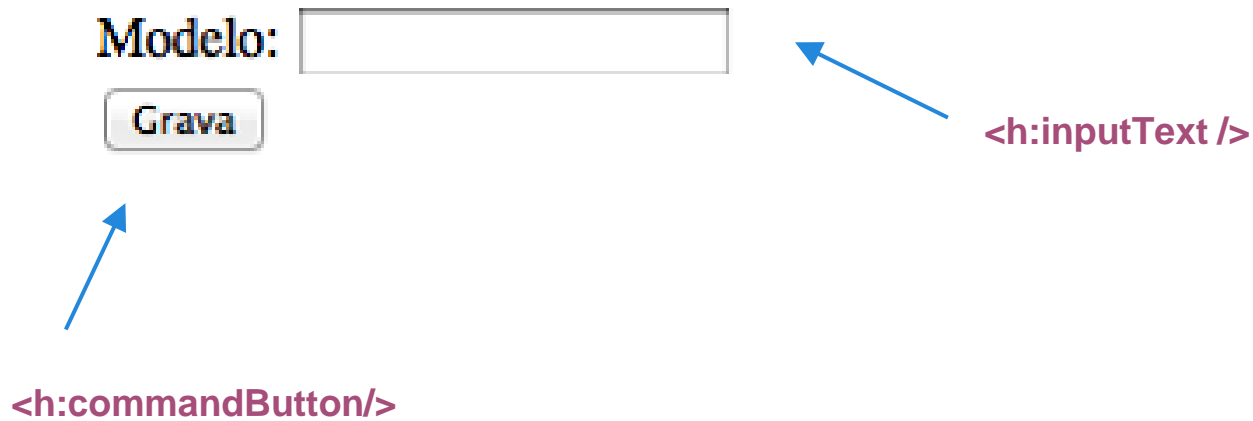
```
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:f="http://java.sun.com/jsf/core"
      xmlns:h="http://java.sun.com/jsf/html">

  <h:head>
    <title>Cadastro de automóvel</title>
  </h:head>

  <h:body>
    <h:form>
      Modelo: <h:inputText
        value="#{automovelBean.automovel.modelo}"/>
      <br/>
      <h:commandButton value="Grava"
        action="#{automovelBean.grava}" />
    </h:form>
  </h:body>
</html>
```

Exemplo de .xhtml com tags do jsf

O primeiro contato com o JSF



O primeiro contato com o JSF

```
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:f="http://java.sun.com/jsf/core"
      xmlns:h="http://java.sun.com/jsf/html">

  <h:head>
    <title>Cadastro de automóvel</title>
  </h:head>

  <h:body>
    <h:form>
      Modelo: <h:inputText
        value="#{automovelBean.automovel.modelo}"/>

      <br/>
      <h:commandButton value="Grava"
        action="#{automovelBean.grava}" />
    </h:form>
  </h:body>
</html>
```

Managed Bean



O primeiro contato com o JSF

```
@ManagedBean
public class AutomovelBean {
    private Automovel automovel = new Automovel();

    public void grava() {
        new AutomovelDAO().grava(automovel);
    }

    // getter e setter
}
```


O primeiro contato com o JSF

cadastro.xhtml

```
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:f="http://java.sun.com/jsf/core"
      xmlns:h="http://java.sun.com/jsf/html">

  <h:head>
    <title>Cadastro de automóvel</title>
  </h:head>

  <h:body>

    <h:form>
      Modelo: <h:inputText
        value="#{automovelBean.automovel.modelo}" />
      <br/>
      <h:commandButton value="Grava"
        action="#{automovelBean.grava}" />
    </h:form>
  </h:body>
</html>
```

O que aparece no navegador.

```
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title>Cadastro de Automóvel</title>
  </head>
  <body>
    <form id="j_idt6" name="j_idt6" method="post"
      action="/faces-motors/faces/automovel/cadastro.xhtml"
      enctype="application/x-www-form-urlencoded">

      <input type="hidden" name="j_idt6" value="j_idt6" />

      Modelo: <input type="text" name="j_idt6:j_idt9" />

      <input type="submit" name="j_idt6:j_idt18" value="Grava" />

      <input type="hidden" name="javax.faces.ViewState"
        id="javax.faces.ViewState"
        value="2892395016760917637:4875835637171130451"
        autocomplete="off" />

    </form>
  </body>
</html>
```

CRUD - Agenda de Contatos

Meta:

Nome:

Endereço:

Número:

Telefone:

Cidade:

Salvar

Limpar

Nome	Endereço	Número	Telefone	Cidade

CRUD - Agenda de Contatos

Primeiramente iremos criar a classe “Contato” no pacote “br.com.agenda.models”. Nesta classe teremos os seguintes atributos:

- ☐ nome
- ☐ endereco
- ☐ numero
- ☐ telefone
- ☐ cidade

Crie todos os atributos como privados, tipo String e seus respectivos get's e set's.

CRUD - Agenda de Contatos

A classe deve ficar da seguinte maneira:

```
package br.com.agenda.models;

public class Contato {

    private String nome;
    private String endereco;
    private String numero;
    private String telefone;
    private String cidade;

    /* gets e sets */
}
```

Use o atalho
“Alt+Shift+S+R” para
gerar os métodos get’s
e set’s.

CRUD - Agenda de Contatos

Agora iremos criar o Managed Bean no pacote
“br.com.agenda.controller” com o seguinte nome
“ContatoMBean”.

CRUD - Agenda de Contatos

```
package br.com.agenda.controller;

import javax.faces.bean.ManagedBean;
import javax.faces.bean.ViewScoped;

import br.com.agenda.models.Contato;

@ManagedBean
@ViewScoped
public class ContatoMBean implements Serializable{

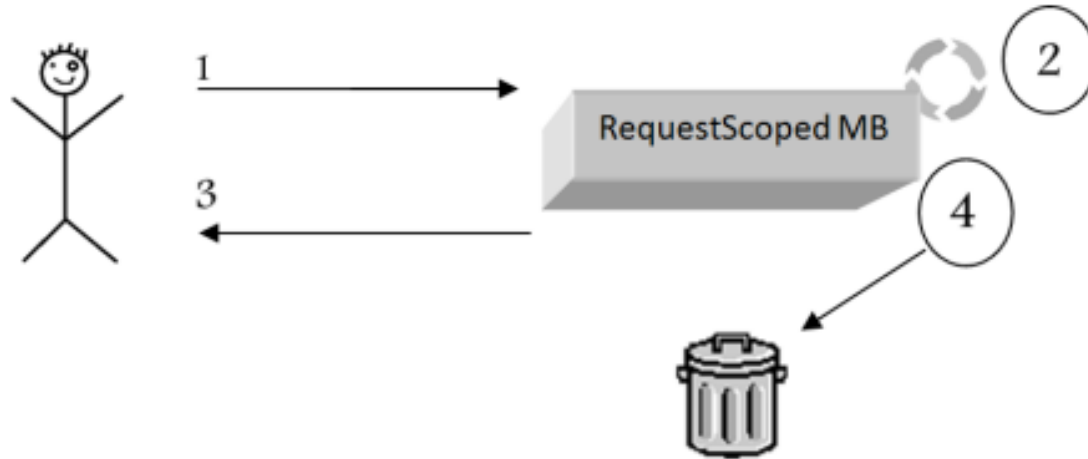
    private static final long serialVersionUID = 1L;

    private Contato contato = new Contato();
    private List<Contato> contatos = new ArrayList<Contato>();
    /*gets e sets*/
}
```

CRUD - Agenda de Contatos

Entendendo os escopos do JSF

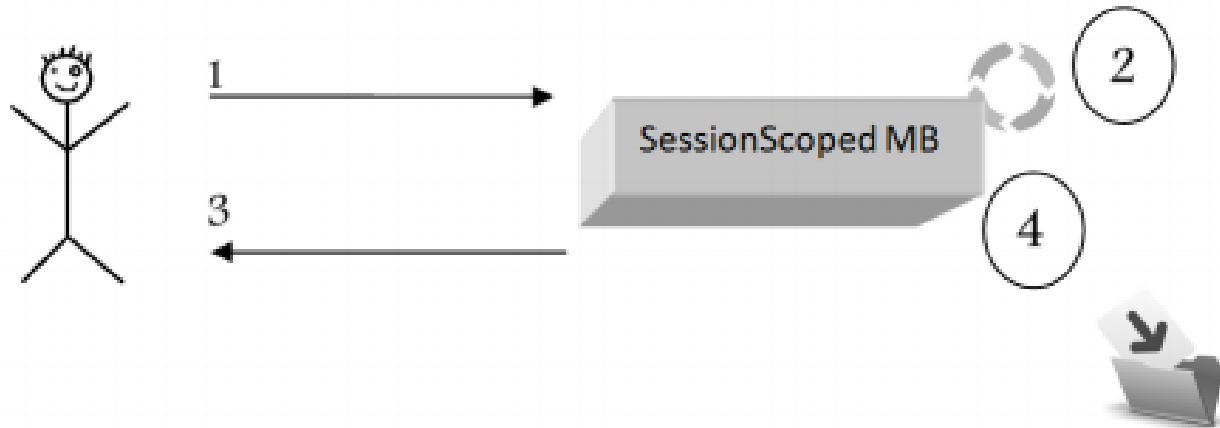
@RequestScoped para escopos curtos.



CRUD - Agenda de Contatos

Entendendo os escopos do JSF

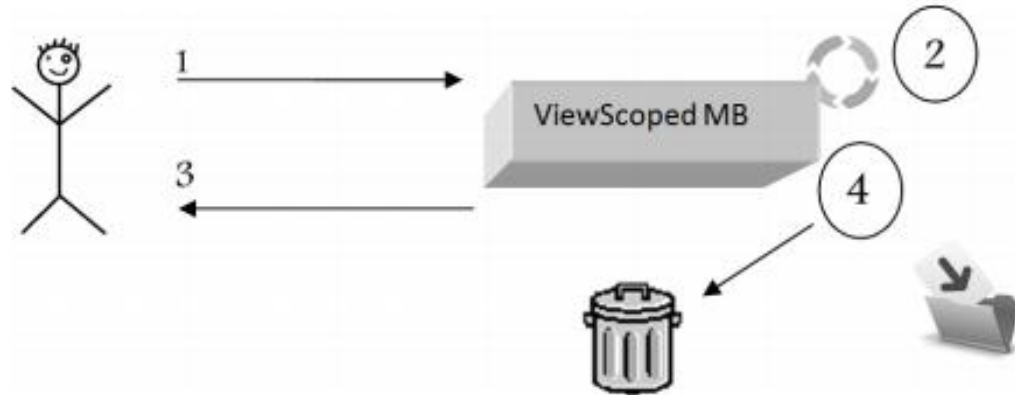
Mantenha o bean na sessão com `@SessionScoped`.



CRUD - Agenda de Contatos

Entendendo os escopos do JSF

Entenda o **@ViewScoped**.



CRUD - Agenda de Contatos

Criando a primeira tela

- ☐ Criando arquivo index.xhtml
- ☐ Escolhendo o template
- ☐ Configurando a página no web.xml

CRUD - Agenda de Contatos

Criando a primeira tela

Resultado em código:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
    xmlns:ui="http://java.sun.com/jsf/facelets"
    xmlns:f="http://java.sun.com/jsf/core"
    xmlns:h="http://java.sun.com/jsf/html">

<h:head></h:head>
<body>
</body>
</html>
```

CRUD - Agenda de Contatos

Utilizando os componentes JSF

Tags <h:form/> e <h:panelGrid/>.

```
<body>  
<h:form>  
<h:panelGrid>  
  
</h:panelGrid>  
</h:form>  
</body>
```

CRUD - Agenda de Contatos

Utilizando os componentes JSF

Entrada e Saída de dados com `<h:inputText>` e `<h:outputLabel>`.

Nome:	<input type="text"/>
Endereço:	<input type="text"/>
Número:	<input type="text"/>
Telefone:	<input type="text"/>
Cidade:	<input type="text"/>

CRUD - Agenda de Contatos

Utilizando os componentes JSF

Entrada e Saída de dados com <h:inputText> e <h:outputLabel>.

```
<h:outputLabel value="Nome:" />
```

```
<h:inputText id="nome" value="#{contatoMBean.contato.nome}" size="20"  
required="true" requiredMessage="Campo Obrigatório" />
```

```
<h:message for="nome" />
```

```
<h:outputLabel value="Endereço:" />
```

```
<h:inputText id="end" value="#{contatoMBean.contato.endereco}" size="20"  
required="true" requiredMessage="Campo Obrigatório" />
```

```
<h:message for="end" />
```

CRUD - Agenda de Contatos

Utilizando os componentes JSF

Entrada e Saída de dados com <h:inputText> e <h:outputLabel> (continuação).

```
<h:outputLabel value="Número:" />

<h:inputText id="nro" value="#{contatoMBean.contato.numero}" size="20" />

<h:message for="nro" />

<h:outputLabel value="Telefone:" />

<h:inputText id="tel" value="#{contatoMBean.contato.telefone}" size="20" />

<h:message for="tel" />

<h:outputLabel value="Cidade:" />

<h:inputText id="mun" value="#{contatoMBean.contato.cidade}" size="20"
required="true" requiredMessage="Campo Obrigatório" />

<h:message for="mun" />
```

CRUD - Agenda de Contatos

Utilizando os componentes JSF

Verificando preenchimento do campo com <h:message>.

```
<h:outputLabel value="Número:" />
```

```
<h:inputText id="nro" value="#{contatoMBean.contato.numero}" size="20" />
```

```
<h:message for="nro" />
```


CRUD - Agenda de Contatos

Utilizando os componentes JSF

Botões e Eventos.

A rectangular button with a light gray gradient and a thin black border. The text "Salvar" is centered in a blue, monospace-style font.A rectangular button with a light gray gradient and a thin black border. The text "Limpar" is centered in a blue, monospace-style font.

CRUD - Agenda de Contatos

Utilizando os componentes JSF

Botões e Eventos.

```
<h:commandButton id="btnE" value="Enviar" type="submit" action="#{contatoMBean.salvar}"/>
```

CRUD - Agenda de Contatos

Utilizando os componentes JSF

Botões e Eventos.

```
<h:commandButton id="btnE" value="Enviar" type="submit" action="#{contatoMBean.salvar}"/>
```

```
public void salvar() {  
    contatos.add(contato);  
    contato = new Contato();  
}
```

CRUD - Agenda de Contatos

Utilizando os componentes JSF

Botões e Eventos.

Agora é sua vez, crie o botão Limpar e associe ao mesmo a ação de limpar o formulário.

CRUD - Agenda de Contatos

Utilizando os componentes JSF

Botões e Eventos.

Agora é sua vez, crie o botão Limpar e associe ao mesmo a ação de limpar o formulário.

```
<h:commandButton id="btnLimpar" value="Limpar" type="reset"/>
```

CRUD - Agenda de Contatos

Utilizando os componentes JSF

Listando contatos salvos.

Nome	Endereço	Número	Telefone	Cidade

CRUD - Agenda de Contatos

Utilizando os componentes JSF

Listando contatos salvos.

```
<h:panelGrid>
<h:dataTable id="tableContato" value="#{contatoMBean.contatos}" var="contato"
title="Contatos Cadastrados" border="1" rows="10">
<h:column>
<f:facet name="header">
<h:outputText value="Nome" />
</f:facet>
<h:outputText value="#{contato.nome}" />
</h:column>
<h:column>
<f:facet name="header">
<h:outputText value="Endereço" />
</f:facet>
<h:outputText value="#{contato.endereco}" />
</h:column>
<h:column>
<f:facet name="header">
```

CRUD - Agenda de Contatos

Utilizando os componentes JSF

Listando contatos salvos (continuação).

```
<h:outputText value="Número" />
</f:facet>
<h:outputText value="#{contato.numero}" />
</h:column>
<h:column>
<f:facet name="header">
<h:outputText value="Telefone" />
</f:facet>
<h:outputText value="#{contato.telefone}" />
</h:column>
<h:column>
<f:facet name="header">
<h:outputText value="Cidade" />
</f:facet>
<h:outputText value="#{contato.cidade}" />
</h:column>
</h:dataTable>
</h:panelGrid>
```


CRUD - Agenda de Contatos

Enriquecendo a tela com o PrimeFaces

Com o primefaces-5.0.jar na pasta lib do seu projeto, basta adicionar mais um namespace no início do index.xhtml.

```
xmlns:p="http://primefaces.org/ui"
```

CRUD - Agenda de Contatos

Enriquecendo a tela com o PrimeFaces

Ficando da seguinte forma:

```
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:ui="http://java.sun.com/jsf/facelets"
      xmlns:f="http://java.sun.com/jsf/core"
      xmlns:h="http://java.sun.com/jsf/html"
      xmlns:p="http://primefaces.org/ui" >
```

CRUD - Agenda de Contatos

Enriquecendo a tela com o PrimeFaces

Resultado:

Nome:

Endereço:

Número:

Telefone:

Cidade:

Nome	Endereço	Número	Telefone	Cidade
No records found.				

CRUD - Agenda de Contatos

Enriquecendo a tela com o PrimeFaces

Preencha o formulário e clique no botão Salvar. O que aconteceu?

CRUD - Agenda de Contatos

Enriquecendo a tela com o PrimeFaces

Preencha o formulário e clique no botão Salvar. O que aconteceu?

Possivelmente nada aconteceu.

CRUD - Agenda de Contatos

Enriquecendo a tela com o PrimeFaces

Preencha o formulário e clique no botão Salvar. O que aconteceu?

Possivelmente nada aconteceu.

```
<p:commandButton id="btnSalvar" value="Salvar" type="submit"  
action="#{contatoMBean.salvar}"/>
```



Não precisaremos mais.

CRUD - Agenda de Contatos

Enriquecendo a tela com o PrimeFaces

Deixe o botão da seguinte forma, adicione a tag `ajax= " false"`.


```
<p:commandButton id="btnSalvar" value="Salvar" ajax="false"
action="#{contatoMBean.salvar}"/>
```

CRUD - Agenda de Contatos

Enriquecendo a tela com o PrimeFaces

Deixe o botão da seguinte forma, adicione a tag `ajax = "false"`.

```
<p:commandButton id="btnSalvar" value="Salvar" ajax="false"
action="#{contatoMBean.salvar}"/>
```



A tag `ajax` é exclusiva do PrimeFaces e por padrão é `true`.

CRUD - Agenda de Contatos

Enriquecendo a tela com o PrimeFaces

A nova tabela

Nome	Endereço	Número	Telefone	Cidade
No records found.				

CRUD - Agenda de Contatos

Enriquecendo a tela com o PrimeFaces

Action e ActionListener

- ❑ A tag “action” é usado normalmente em situação em que precisamos fazer um redirecionamento ou desejamos submeter a página por completo em uma requisição. Por padrão não é passada nada por parâmetro.
- ❑ A tag “actionListener” é usada quando queremos usar requisições que atualizam parte da página web utilizando ajax. Por padrão é preciso ter como parâmetro um “ActionEvent”.

CRUD - Agenda de Contatos

Enriquecendo a tela com o PrimeFaces

Action e ActionListener

Adicione como parâmetro um `ActionEvent` do pacote **`javax.faces.event.*`**.

```
public void salvar(ActionEvent e) {  
    contatos.add(contato);  
    contato = new Contato();  
}
```

CRUD - Agenda de Contatos

Enriquecendo a tela com o PrimeFaces

Action e ActionListener

No botão substitua a tag “action” para a “actionListener”, remova a tag ajax e adicione a tag update com o id da tabela. Como segue abaixo:

```
<p:commandButton id="btnSalvar" value="Salvar" update="tableContato"
actionListener="#{contatoMBean.salvar}"/>
```

Obrigado!