

JSTL JavaServer Pages Standard Tag Library

O JSTL JavaServer Pages Standard Tag Library (JSTL) faz parte das especificações do J2EE que define um conjunto de extensões das tags JSP

O objetivo destas tags é substituir os scripts na página, facilitando a manutenção e o entendimento de uma página JSP

Por exemplo, para receber os parâmetros de uma página, ao invés de utilizarmos programação em Java, poderíamos receber as informações utilizando, como por exemplo o código abaixo.

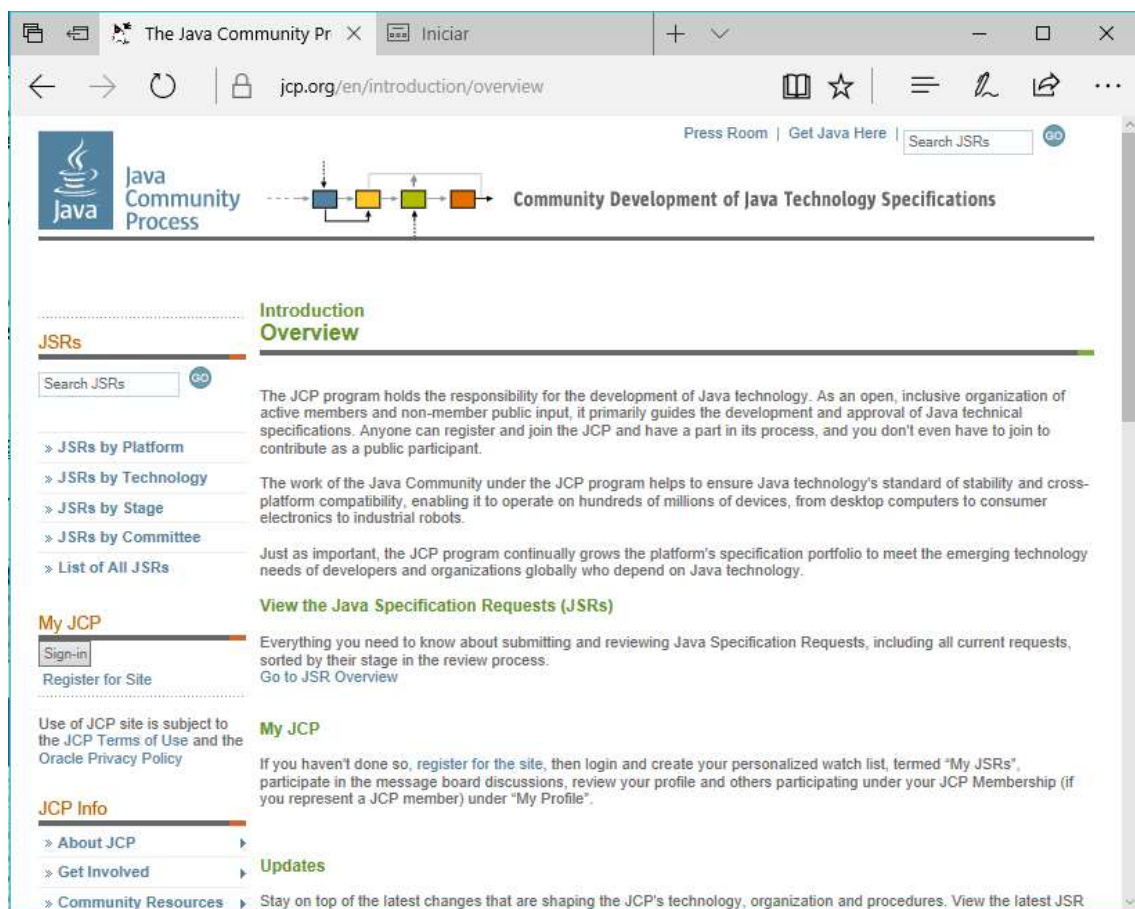
```
<c:set var="login" value="${param.login}"/>
```

Este código além de receber o parâmetro login, já está atribuindo o valor do mesmo para a variável login.

Especificações da JSTL 1.0

A especificação JSTL 1.0 é um JSR (Java Specification Request) criado através do JCP (Java Community Process)

Para conhecer mais visite <https://jcp.org/en/introduction/overview>

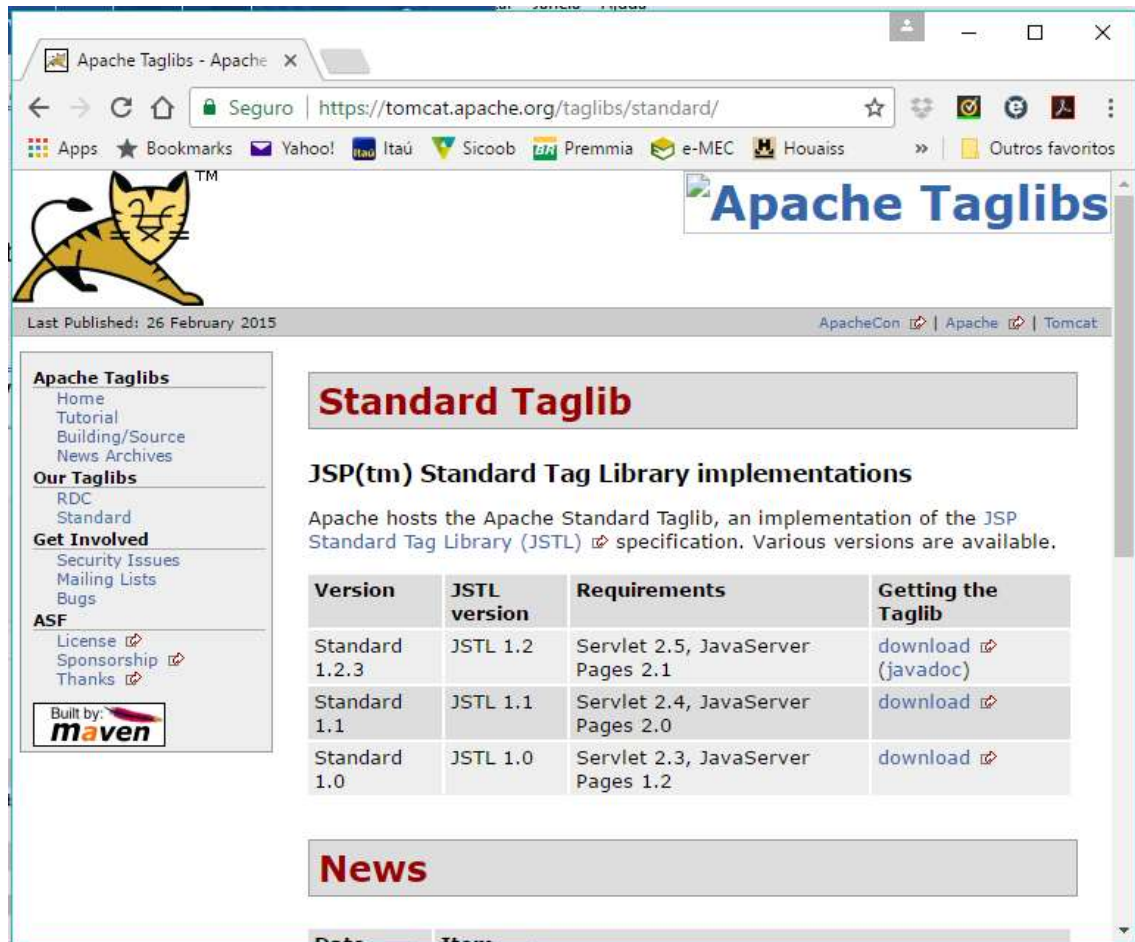


Instalando o JSTL

Para utilizar o JSTL é necessário a instalação de uma implementação da mesma.

Para obter uma cópia visite:

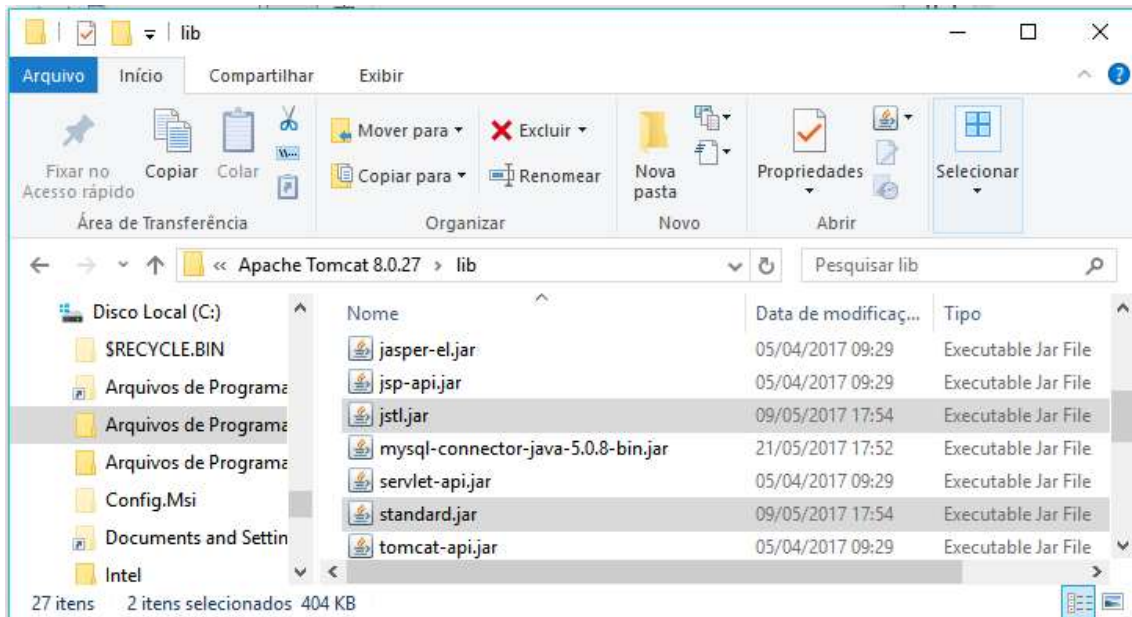
<https://tomcat.apache.org/taglibs/standard/>



Obs: utilize o mesmo da aula anterior que está no Moodle.

Observação sobre a instalação

Se preferir pode copiar estes arquivos para /lib do Tomcat, assim estará disponível para todas as suas aplicações web



Visão Geral do JSTL

O JSTL é composto pelos seguintes elementos:

Expression Language

Bibliotecas de tags

- Core

- Format

- SQL

- XML

Taglib Validators

API

Bibliotecas de tags

Biblioteca Core

Esta biblioteca contém tags de propósito geral

Manipulação de variáveis

- out
- set
- remove

Tratamento de Erros

- catch

Condicionais

- if
- choose
- when
- otherwise

Manipulação de URLs

- import
- url

- redirect
 - param
- Iteração
- forEach
 - forTokens

Manipulação de urls e Iteração serão abordados na próxima aula

Tags de Finalidades Gerais: out

<c:out>

Finalidade

Imprimir uma string em uma página JSP

Sintaxe

A sintaxe da ação **<c:out>** é:

Sem corpo

```
<c:out value="value" [escapeXml="{true|false}"]  
[default="defaultValue"] />
```

Com corpo

```
<c:out value="value" [escapeXml="{true|false}"]>  
default value  
</c:out>
```

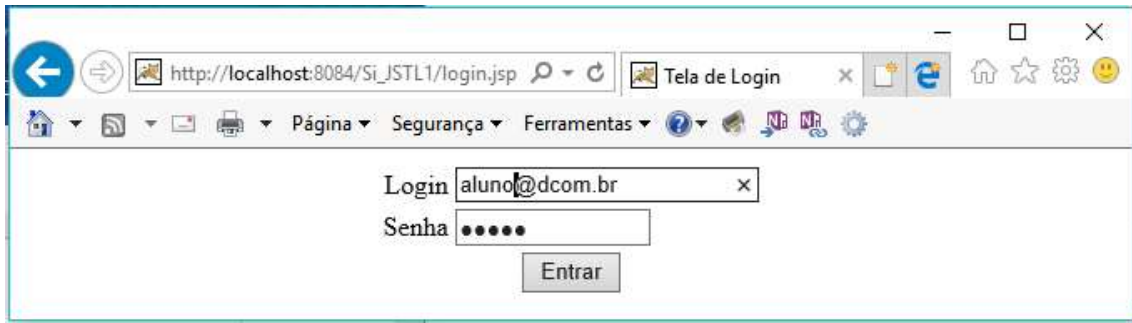
Atributos

Nome	Tipo	Descrição
value	Object	Expressão à ser avaliada
escapeXml	boolean	Determina como certos caracteres de XML que são convertidos em referências de entidade ou em referências de caractere. Por padrão, essa conversão acontece para os seguintes caracteres: Um caractere < é substituído por <. Um caractere > é substituído por >. Um caractere & é substituído por &. Um caractere ' é substituído por '. Um caractere " é substituído por ".
default	Object	O atributo <i>default</i> (opcional) especifica um valor padrão que deve ser utilizado quando a expressão é avaliada como <i>null</i>

Exemplo out

Página de Login

Esta página vai chamar incluirlogin.jsp



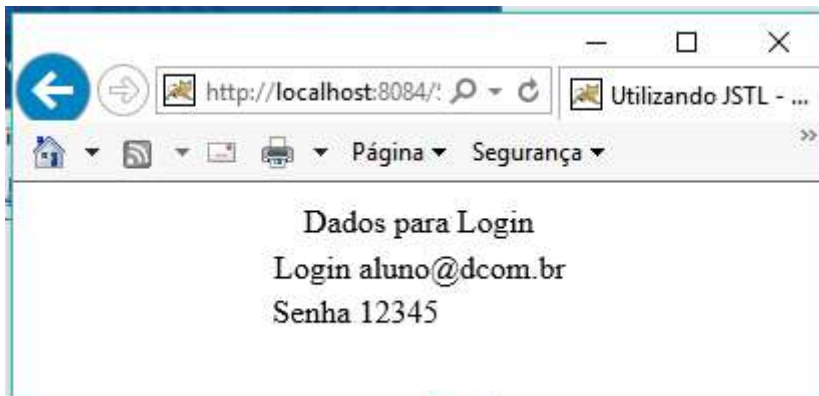
Código fonte da página

```
login.jsp
1 <%@page contentType="text/html" pageEncoding="UTF-8"%>
2 <!DOCTYPE html>
3 <html>
4 <head>
5 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
6 <title>Tela de Login</title>
7 </head>
8 <body>
9 <form action="incluirlogin.jsp">
10 <table border="0" align="center">
11 <tr>
12 <td align="right">Login</td>
13 <td align="left"><input type="text" name="login" size="25"></td>
14 </tr>
15 <tr>
16 <td align="right">Senha</td>
17 <td align="left"><input type="password" name="senha" size="15"></td>
18 </tr>
19 <tr>
20 <td colspan="2" align="center"><input type="submit" value="Entrar"></td>
21 </tr>
22 </table>
23 </form>
24 </body>
25 </html>
```

Página para receber os dados de Login

Página de incluirlogin.jsp

Esta página vai receber os parâmetros enviados pela página login.jsp e apresenta-los na tela



Código fonte da página

```
induirlogin.jsp
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Utilizando JSTL - out</title>
<LINK href="estilo.css" type="text/css" rel="stylesheet">
</head>
<body>
<table border="0" align="center">
<tr>
<td colspan="2" align="center"><c:out value="Dados para Login"/></td>
</tr>
<tr>
<td align="right">Login</td>
<td align="left"><c:out value="${param.login}"/></td>
</tr>
<tr>
<td align="right">Senha</td>
<td align="left"><c:out value="${param.senha}"/></td>
</tr>
</table>
</body>
</html>
```

Tags de Finalidades Gerais: set

Finalidades

Criar e atribuir valor à uma variável

Atribuir valor à um Bean

Atribuir e remover entradas em um tipo map

Sintaxe

A sintaxe da ação **<c:set>** é:

A ação **<c:set>** suporta quatro sintaxes, duas dessas sintaxes permitem você manipular variáveis de escopo e as outras duas, permitem você manipular Beans e MAPs.

Sintaxe 1: atribuir valor à uma variável e definir o seu escopo

<c:set value="value"

var="varName" [scope="{page|request|session|application}"]/>

Sintaxe 2: atribuir o valor do conteúdo do corpo à uma variável e definir o seu escopo

<c:set var="varName" [scope="{page|request|session|application}"]>

body content

</c:set>

Sintaxe 3: atribuir valor à uma variável de objeto

<c:set value="value"

target="target" property="propertyName"/>

Sintaxe 4: atribuir o valor do conteúdo do corpo à uma variável de objeto

```
<c:set target="target" property="propertyName">
body content
</c:set>
```

Atributos

Nome	Tipo	Descrição
value	Object	Expressão à ser avaliada
var	String	Nome da variável
scope	String	Escopo da variável
target	Object	Objeto, cuja propriedade sera configurada
property	String	Nome da propriedade.

Exemplo set

Página de Login

Baseado na página incluirlogin.jsp, ao invés de imprimirmos diretamente o resultado dos parâmetros, passamos a utilizar variáveis. Observe as linhas 10 e 11 do código abaixo.

```
incluirlogin2.jsp
1 <%@page contentType="text/html" pageEncoding="UTF-8"%>
2 <%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
3 <html>
4   <head>
5     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
6     <title>Utilizando JSTL - out</title>
7     <LINK href="estilo.css" type="text/css" rel="stylesheet">
8   </head>
9   <body>
10    <c:set var="login" value="${param.login}"/>
11    <c:set var="senha" value="${param.senha}"/>
12    <table border="0" align="center">
13      <tr>
14        <td colspan="2" align="center"><c:out value="Dados para Login"/></td>
15      </tr>
16      <tr>
17        <td align="right">Login</td>
18        <td align="left"><c:out value="${login}"/></td>
19      </tr>
20      <tr>
21        <td align="right">Senha</td>
22        <td align="left"><c:out value="${senha}"/></td>
23      </tr>
24    </table>
25  </body>
26 </html>
```

Tags de Finalidades Gerais: remove

<c: remove >

Finalidade

Remover uma variável

Sintaxe

A sintaxe da ação **<c:remove>** é:

```
<c:remove var="varName"  
[scope="{page|request|session|application}"]/>
```

Nome	Tipo	Descrição
var	String	Nome da variável
scope	String	Escopo da variável

Observação

Se você não especificar o atributo escopo, a ação **<c:remove>** irá buscar nos escopos por uma variável de escopo com o nome que você especificou no atributo var.

A ordem da busca nos escopos é:

- page,
- request
- session
- application

Exemplo

Código parcial

```
<c:set var="status" value="Login correto"/>  
<c:out value="${status}" />  
<c:remove var="status"/>  
<c:out value="${status}" />
```

Tags de Finalidades Gerais: catch

<c: catch >

Finalidade

Tratar erros

Sintaxe

A sintaxe da ação **<c:catch>** é:

<c:catch [var="varName"]>
nested actions
</c:catch>

Exemplo

Código parcial

```
<c:catch var="erro">
  <%
    out.println(5/0);
  %>
</c:catch>
${erro}
```

Ações Condicionais

As tags para ações condicionais foram criadas para facilitar a realização de programação condicional nas páginas JSP

A JSTL tem as seguintes tags condicionais:

- <c:if>
- <c:choose>
- <c:when>
- <c:otherwise>

Elas permitem realizar dois tipos diferentes de processamento

- Processamento condicional simples
- Processamento mutuamente exclusivo.

Tags de Finalidades Gerais: if

<c: if >

Finalidade

Realiza o processamento condicional simples.

Se o valor de test como um valor true, o conteúdo do corpo da ação <c:if> é processado.

Caso contrário o conteúdo do corpo é ignorado.

O conteúdo do corpo das tags pode ser qualquer código JSP válido!

Sintaxe

A sintaxe da ação <c:if> é:

Sintaxe 1: sem corpo

<c:if test="testCondition"
var="varName" [scope="{page|request|session|application}"]/>

Sintaxe 2: com corpo

```
<c:if test="testCondition"
[var="varName"] [scope="{page|request|session|application}"]>
body content
</c:if>
```

Exemplo

Código parcial

```
<c:catch var="erro">
  <%
    out.println(5/0);
  %>
</c:catch>

<c:if test="${erro != null}">
  <c:out value="${'Ocorreu uma exceção.'}" />
</c:if>

<c:if test="${erro != null}">
  <c:set var="mensagem" value="Ocorreu uma exceção" scope="page" />
</c:if>
<c:out value="${mensagem}" />
```

Observação

O atributo test, que é requerido, é uma expressão boolean que determina quando o conteúdo do corpo da tag será processado.

Os atributos opcionais var e scope especificam a variável de escopo que armazenará o valor boolean da expressão especificada com o atributo test.

Tags de Finalidades Gerais: choose

<c:choose >

Finalidade

Realiza o processamento mutuamente exclusivo.

Sintaxe

A sintaxe da ação **<c:if>** é:

```
<c:choose>
body content (<when> and <otherwise> subtags)
</c:choose>
```

O conteúdo do corpo da ação **<c:choose>** pode ser:

- Uma ou muitas ações **<c:when>**
- Nenhuma ação **<c:otherwise>**. esta ação deve aparecer depois as ações **<c:when>**
- Caracteres de espaço em branco entre as ações **<c:when>** e **<c:otherwise>**

A ação **<c:choose>** processa uma de suas ações aninhadas.

A regra é que a ação **<c:choose>** processe a primeira ação **<c:when>** cuja condição de teste é avaliada como true.

Entretanto, se nenhuma das ações **<c:when>** for aplicável, a ação **<c:otherwise>** é processada, se estiver presente.

Se fornecer uma ação **<c:otherwise>**, você deverá certificar-se de que ela é a última ação dentro da ação **<c:choose>**.

Exemplo

Código parcial

```
<c:choose>
  <c:when test="{erro != null}">
    Ocorreu uma exceção.
  </c:when>
  <c:otherwise>
    Não ocorreu uma exceção.
  </c:otherwise>
</c:choose>
```

Tags de Finalidades Gerais: when

<c: when >

Finalidade

Se você necessitar escolher entre mais de duas condições, você pode simular um switch simplesmente adicionando mais ações **<c:when>** no corpo da ação **<c:choose>**.

Sintaxe

A sintaxe da ação **<c:when>** é:

<c:when test="testCondition">

body content

</c:when>

Nome	Tipo	Descrição
test	boolean	A condição test que determina se o conteúdo deverá ou não ser processado

A ação **<c:when>** é similar a ação **<c:if>**, ambas as ações tem um atributo test que determina se o conteúdo do corpo da ação é avaliado.

Tags de Finalidades Gerais: otherwise

<c: otherwise >

Finalidade

Nenhuma das suas ações **<c:when>** foi avaliada como **true**, o corpo de **otherwise** será executado .

A sintaxe da ação é **<c:otherwise>** é:

<c:otherwise>

conditional block

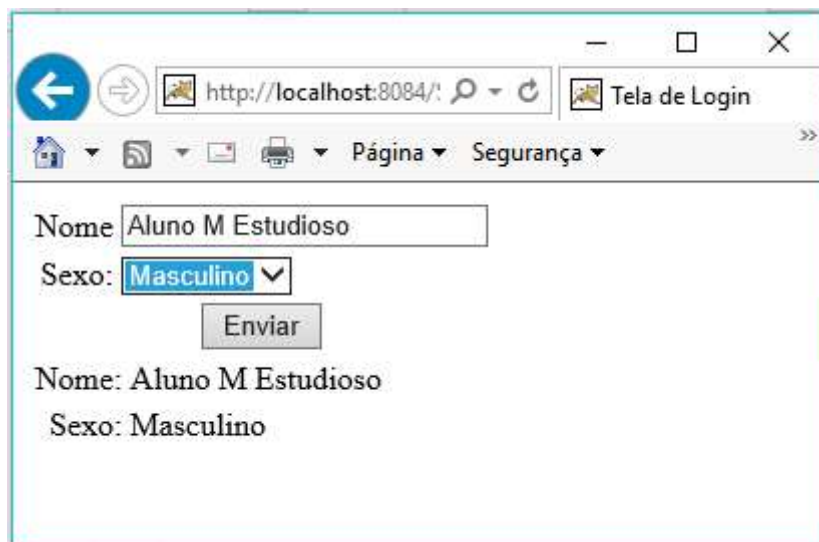
</c:otherwise>

Avaliação da Aula

1 - Criar a seguinte tela de cadastro.

Esta página chamará ela mesma

Não deverá ser utilizado nenhum scriptlet (códigos em Java)



Nome: Aluno M Estudioso

Sexo: Masculino

Enviar

Nome: Aluno M Estudioso

Sexo: Masculino

2- Deverá validar se os campos estão preenchidos

Utilize empty

Exemplo

http://localhost:8084/ Tela de Login

Nome

Sexo:

Enviar

Nome:

Sexo:

3 - Apresentar os valores que foram cadastrados.

http://localhost:8084/ Tela de Login

Nome

Sexo:

Enviar

Nome:

Sexo:

Por favor preencha o nome