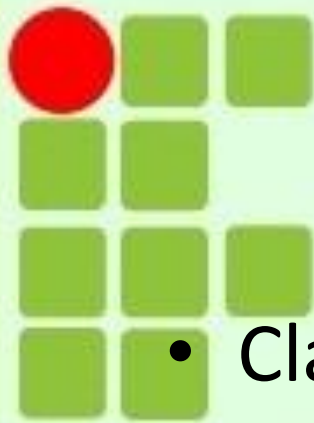


LPW-Servlet e JSP

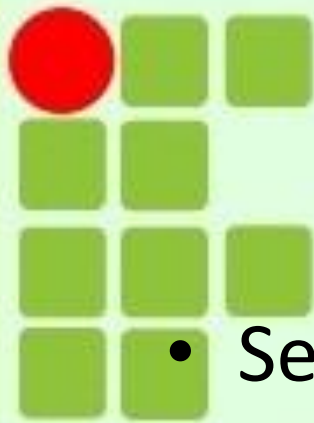
Eduardo Vasconcelos

eduardo.vasconcelos@garanhuns.ifp
e.edu.br



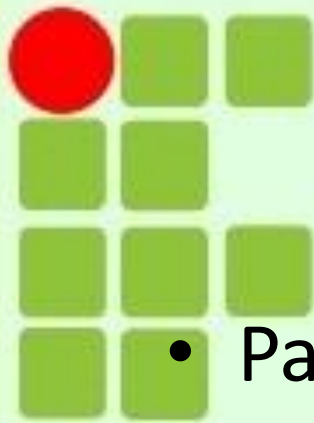
Servlet

- Classe java utilizada para processar requisições Web.
- Para tornar uma classe java um servlet, basta que esta estenda a classe `HTTPServlet`.



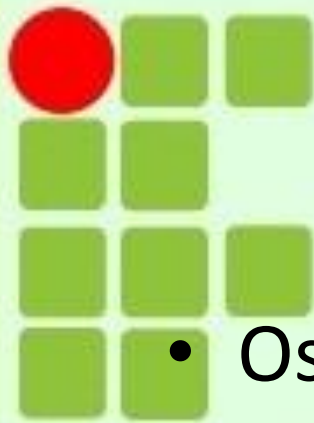
Servlet

- Se a IDE não possuir o recurso automático, basta baixar a biblioteca javax.servlet.
- <http://www.java2s.com/Code/Jar/j/Downloadjavaxservlet14jar.htm>



Servlet

- Para atender requisições Web com Servlet basta implementar um dos métodos abaixo:
 - doGet():
 - Atende requisições “get”;
 - doPost():
 - Atende requisições “post”;



Servlet

- Os métodos doGet e doPost possuem dois parâmetros utilizados para a comunicação com o cliente.
 - HttpServletRequest:
 - Contém todas as informações enviadas pelo cliente;
 - HttpServletResponse:
 - Contém todas as informações enviadas para o cliente;

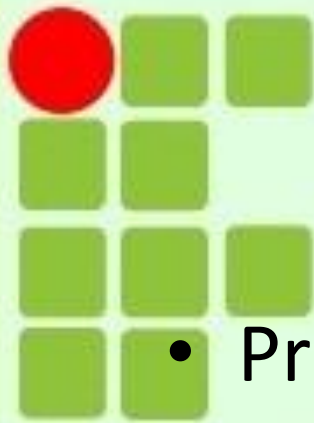
Servlet

- Principais métodos do HttpServletRequest:
 - `getParameter(String)`:
 - Usado para recuperar parâmetros passados nos campos de um formulário ou pela URL. Retorna uma String.

```
<form method="get" action="RecebeRequisicaoServlet">  
    .....  
    campo1: <input type="text" name="campo"/>  
</form>
```

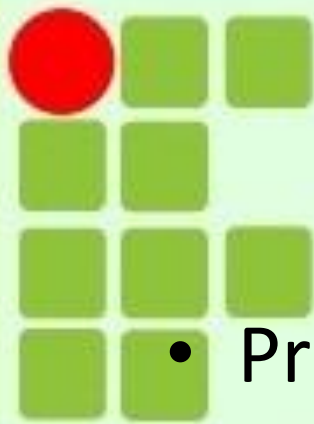
```
String campo1 = request.getParameter("campo");
```





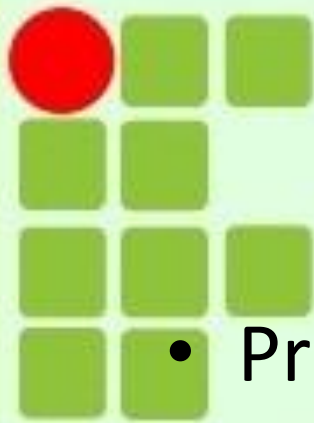
Servlet

- Principais métodos do `HttpServletRequest`:
 - `getParameterValues(String)`:
 - Funcionamento similar ao `getParameter()`;
 - Retorna um Array de String;
 - Utilizado para recuperar valores de listboxes e checkboxes.



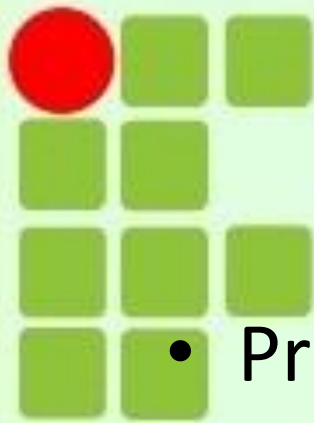
Servlet

- Principais métodos do HttpServletRequest:
 - `getParameterNames();`
 - Retorna um enumeration com os nomes dos parâmetros passados e null caso não haja parâmetros.
 - `getCookies()`
 - Retorna um array contendo todos os colocados por este programa.



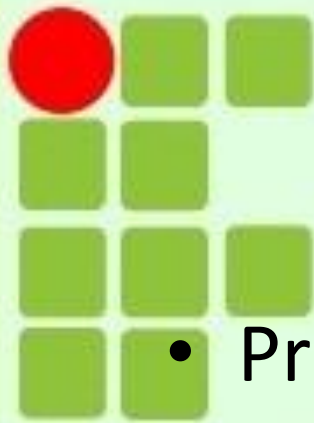
Servlet

- Principais métodos do `HttpServletRequest`:
 - `getSession(boolean)`:
 - Retorna a sessão corrente ou cria uma nova caso o método seja chamado sem argumento ou com argumento `true`;
 - Caso o método seja chamado com o argumento `false`, retorna `null`.



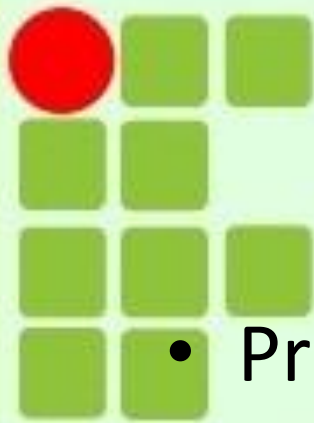
Servlet

- Principais métodos do `HttpServletRequest`:
 - `changeSessionID();`
 - Modifica o identificador da sessão corrente;
 - Pode ser utilizado para motivos de segurança.
 - `setAttribute(String, Object);`
 - Insere um objeto como atributo da requisição com o nome especificado no primeiro parâmetro.



Servlet

- Principais métodos do HttpServletResponse:
 - setContentType(String):
 - Especifica o tipo de conteúdo da saída;
 - “Application/pdf” para definir que a saída será um arquivo pdf.
 - getWrite():
 - Retorna um objeto do tipo PrintWriter;
 - Usado para escrever o html de saída.



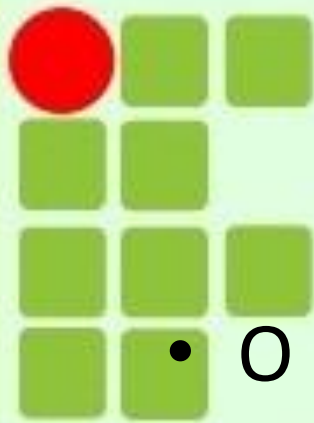
Servlet

- Principais métodos do HttpServletResponse:
 - getOutputStream:
 - Retorna um objeto do tipo ServletOutputStream;
 - Utilizado para enviar arquivos binários para o cliente.
 - Pode ser utilizado também para escrever arquivos de texto como html.



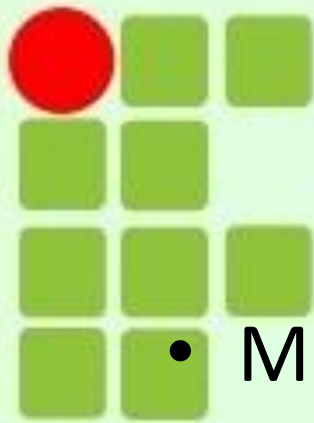
Servlet

- Principais métodos do HttpServletResponse:
 - addCookie(Cookie);
 - Insere um cookie no navegador do cliente.
 - sendRedirect(String);
 - Redireciona o fluxo de apresentação para a URL especificada como parâmetro.



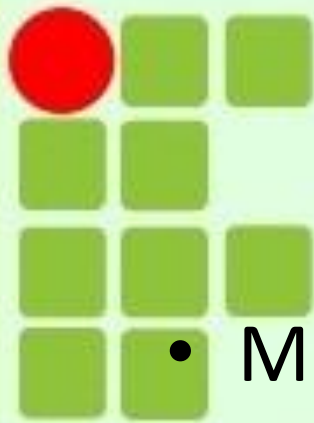
Session

- O protocolo HTTP não mantém estado entre as requisições.
 - Session é utilizado para manter informações entre diferentes requisições.
 - Pode ser utilizado tanto no servlet como no JSP.
 - Existe uma por sessão do navegador.



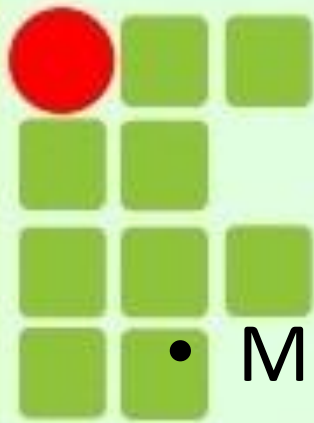
Session

- Métodos:
 - SetAttribute(String, object)
 - Insere um objeto no escopo da sessão com o nome especificado no primeiro parâmetro.
 - getAttribute(String)
 - Retorna o objeto especificado no parâmetro.
 - removeAttribute(String)
 - Remove o objeto da sessão especificado pelo parâmetro.



Session

- Métodos:
 - `getAttributeNames()`:
 - Retorna um `enum<String>` contendo os nomes dos atributos existentes na sessão;
 - `getCreationTime()`:
 - Retorna um `long` contendo o momento específico em que a sessão foi criada.
 - `getLastAcessedTime()`:
 - Retorna um `long` contendo o momento do último acesso a sessão.



Session

- Métodos:
 - `setMaxInactiveInterval(int)`:
 - Define o tempo em que a sessão deve expirar caso nenhum nova requisição seja feito.
 - `invalidate()`:
 - Remove todos os atributos da sessão para aquele cliente.



Cookies

- São arquivos de texto que ficam armazenados no cliente:
 - Em java.
 - `Cookie cookie = new Cookie("nome","valor");`
 - `response.addCookie(cookie);`
 - Para recuperar os cookies utiliza-se o método `getCookies()` do `HttpServletRequest`.



Cookies

- Métodos:
 - getName()
 - Retorna o nome do cookie;
 - getValue()
 - Retorna o valor armazenado;
 - setMaxAge(int)
 - Especifica o tempo de existência do cookie.
 - getMaxAge()
 - Retorna o tempo de existência do cookie.



Request Dispatch

- Quando usamos o método `sendRedirect()` para mudar o fluxo da apresentação para outra página o que acontece?
 - O sistema cria, internamente, uma nova requisição.
 - Todos os atributos e parâmetros existentes na requisição são perdidos.



Request Dispatch

- Request Dispatch é utilizado para transferir o estado da requisição corrente para outra página.
 - Métodos utilizados:
 - `forward(HttpServletRequest, HttpServletResponse);`
 - `Include(HttpServletRequest, HttpServletResponse).`



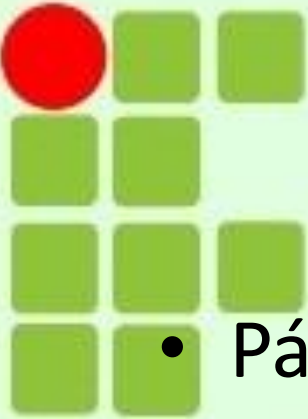
Request Dispatch

- Há duas formas de chamar o Request Dispatcher.
 - `Request.getRequestDispatcher(String);`
 - `getServletContext.getRequestDispatcher(String);`
 - O parâmetro String especifica a página para qual o fluxo da apresentação deverá ser mudada.



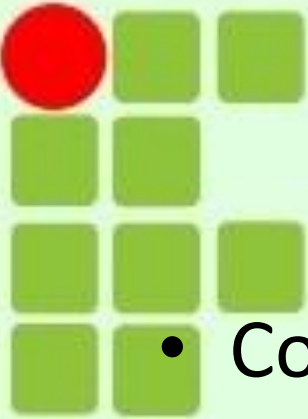
Request Dispatch

- Para mudar o fluxo para a página principal fazemos:
 - `getServletContext().getRequestDispatcher("index.jsp").forward(request,response);`
- Se utilizado o método `include()`:
 - O fluxo é enviado para a próxima página depois volta para a página corrente.
- Se utilizado o método `forward()`:
 - O fluxo é enviado em definitivo para a próxima página.



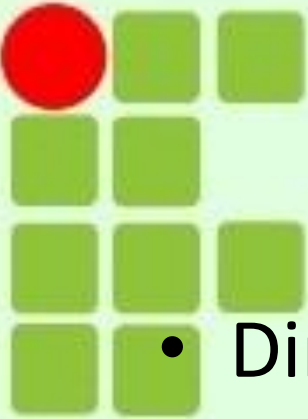
Java Server Pages (JSP)

- Página HTML que permite a inserção de elementos JAVA.
 - Possui vantagem na apresentação de informações.



Java Server Pages (JSP)

- Consiste em um conjunto de diretivas e scriptings.
 - Diretivas são utilizadas para configurar a página, incluir recursos externos ou referenciar bibliotecas de tags;
 - Scriptings para escrever código java.



Java Server Pages (JSP)

- Diretiva.

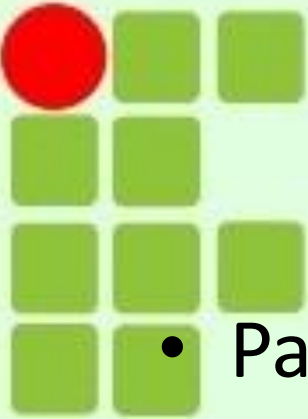
`<%@NomeDaDiretiva atributos %>`

– page, include, taglib.

- Page.

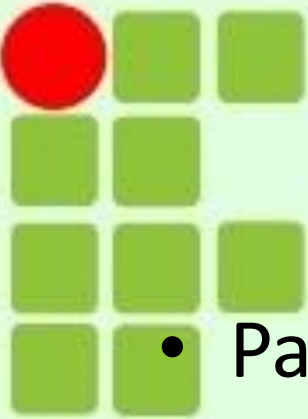
– Utilizado para configurar a página.

`<%@page atributo="valor"%>`



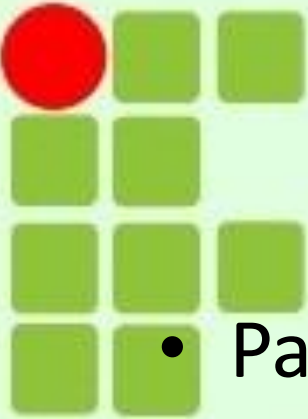
Java Server Pages (JSP)

- Page.
 - autoflush:
 - Especifica se o buffer de saída da página deve ser descarregado assim que estiver cheio.
 - Se definido como false, uma exceção é lançada.
 - buffer:
 - Especifica o tamanho do buffer de saída da página.



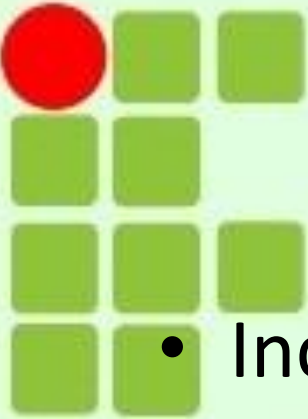
Java Server Pages (JSP)

- Page.
 - contentType:
 - Define o tipo de conteúdo da página.
 - isErrorPage:
 - Utilizadas para definir uma página de erro.
 - isErrorPage:
 - Aponta para qual página de erro a exceção levantada na página corrente deve ser enviada.



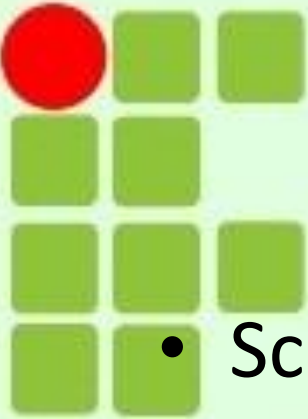
Java Server Pages (JSP)

- Page.
 - isThreadSafe:
 - Indica se a página aceita acesso concorrente.
 - session:
 - Define se a sessão será utilizada na página.
 - import:
 - Utilizada para importar classes para a página.
 - Pode ser declarada mais de uma vez.



Java Server Pages (JSP)

- Include:
 - Utilizada para inserir conteúdo externo à página.
- `<%@include file="página que deseja-se incluir"%>`



Java Server Pages (JSP)

- Scriptings

- Declaração:

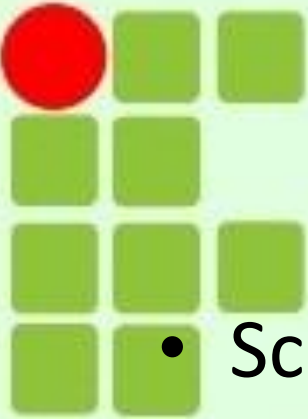
- Utilizado para declarar variáveis e métodos na página.

- ```
<%! int a = 0 %>
```

- Scriptlet:

- Utilizado para programar a página:

- ```
<% if(a>10){out.println("é maior que 10")} %>
```



Java Server Pages (JSP)

- Scriptings

- Expressão:

- Utilizado para escrever no HTML resultado de expressões ou valores de variáveis.

- ```
<%= a+20 %>
```

- Comentário:

- ```
<%-- isto é um comentário jsp --%>
```

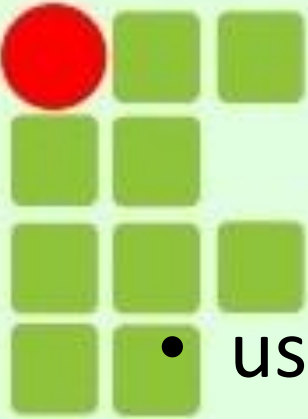



Objetos implícitos em JSP

- O JSP possui alguns objetos que podem ser utilizados livremente na página sem a necessidade de uma declaração formal.
 - Request;
 - Response;
 - Session;
 - Page;
 - Out;
 - Exception;
 - Application;
 - PageContext.

Tags em JSP

- O JSP também possui algumas tags utilizadas para reduzir a quantidade de código java.
 - Sintaxe:
`<jsp:nomeDaTag atributos="valor"/>`



Tags em JSP

- useBean

- Cria uma variável e a coloca em um contexto.

```
<jsp:useBean id="p"  
class="br.edu.garanhuns.ifpe.model.entidades.P  
essoa" scope="page"/>
```

- Cria uma variável p do tipo pessoa e a coloca no escopo da página corrente.
 - É possível colocar no escopo da sessão, requisição ou aplicação.

Tags em JSP

- **setProperty**

- Usado para inserir valores nas propriedades de um objeto.

```
<jsp:setProperty name="p" property="nome" value="Aluizio"/>
```

- **getProperty**

- Retorna o valor de uma propriedade do objeto;

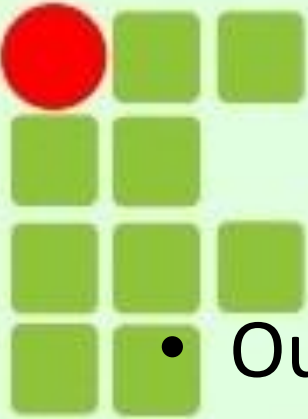
```
<jsp:getProperty name="p" property="nome"/>
```

Tags em JSP

- forward:
 - Funciona como o requestDispatcher forward:
`<jsp:forward page="página"/>`
- param:
 - Insere um parâmetro na requisição.
`<jsp:param name="nomedoparametro"
value="valor"/>`

Tags em JSP

- Outro tipo de expressão JSP.
 - Existe uma expressão JSP que permite o uso de forma direta de objetos inserido na sessão ou na requisição.
 - Sintaxe:
 - `${variável}`



Tags em JSP

- Outro tipo de expressão JSP.

```
<%  
    session.setAttribute("minhaVariavel", "Este é um valor inserido na sessão");  
%>  
  
<h1>${minhaVariavel}</h1>
```