



# Programação WEB

## *Java Server Pages e Servlets*

Douglas Nassif Roma Junior

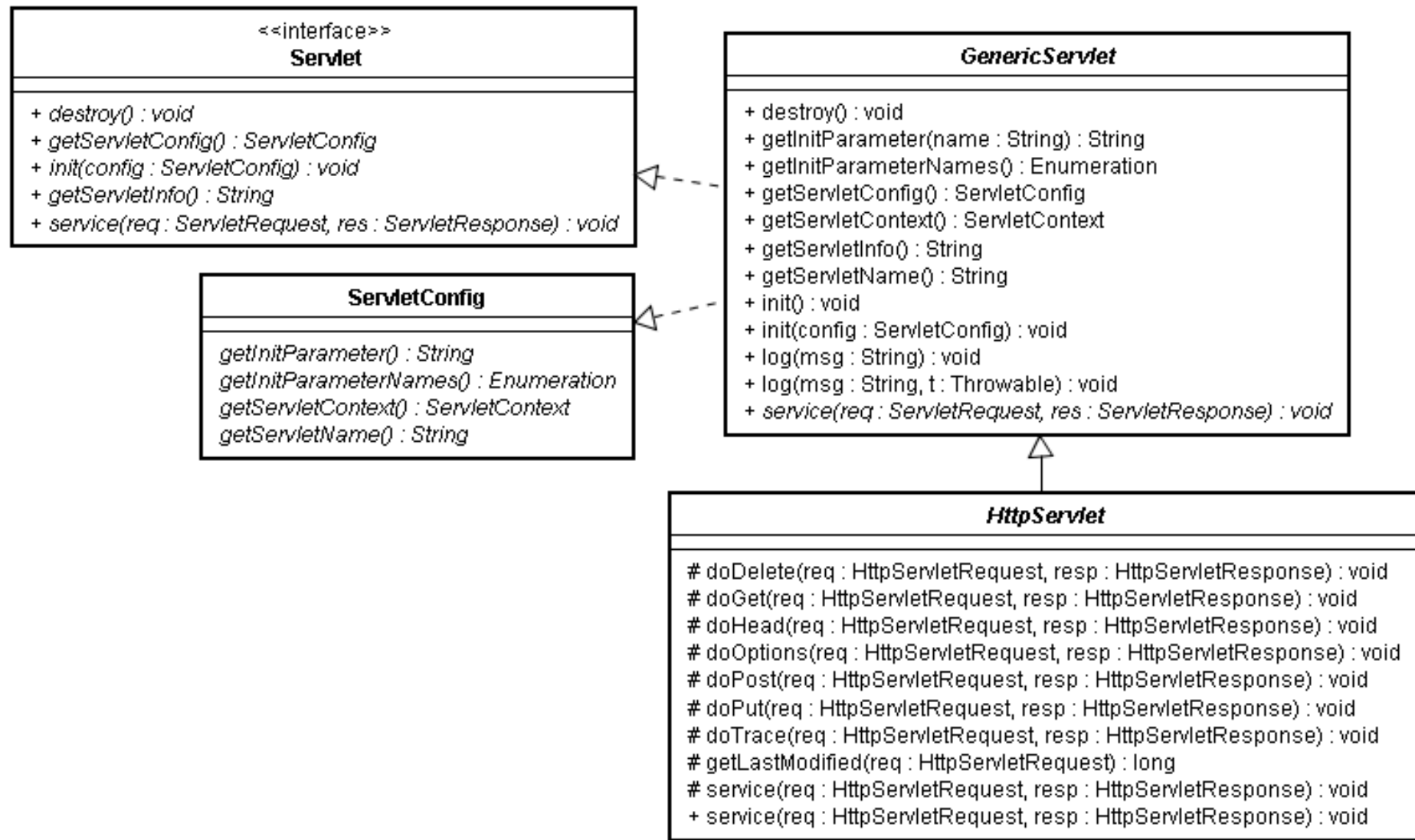
[douglas.junior@grupointegrado.br](mailto:douglas.junior@grupointegrado.br)

# Conteúdo

- A classe `HttpServlet`;
- A interface `HttpServletRequest`
  - Obter cabeçalhos;
  - Obter string de consulta;
  - Obter parâmetros;
  - Obter múltiplos valores de parâmetro;
- A interface `HttpServletResponse`;
  - Enviando código de erro;
  - Tratando caracteres especiais;
  - Armazenagem temporária;
  - Preenchimento do HTML;
  - Solicitação de despacho;
  - Inclusão de outros recursos;
  - Encaminhamento de controle de processamento



# HttpServlet

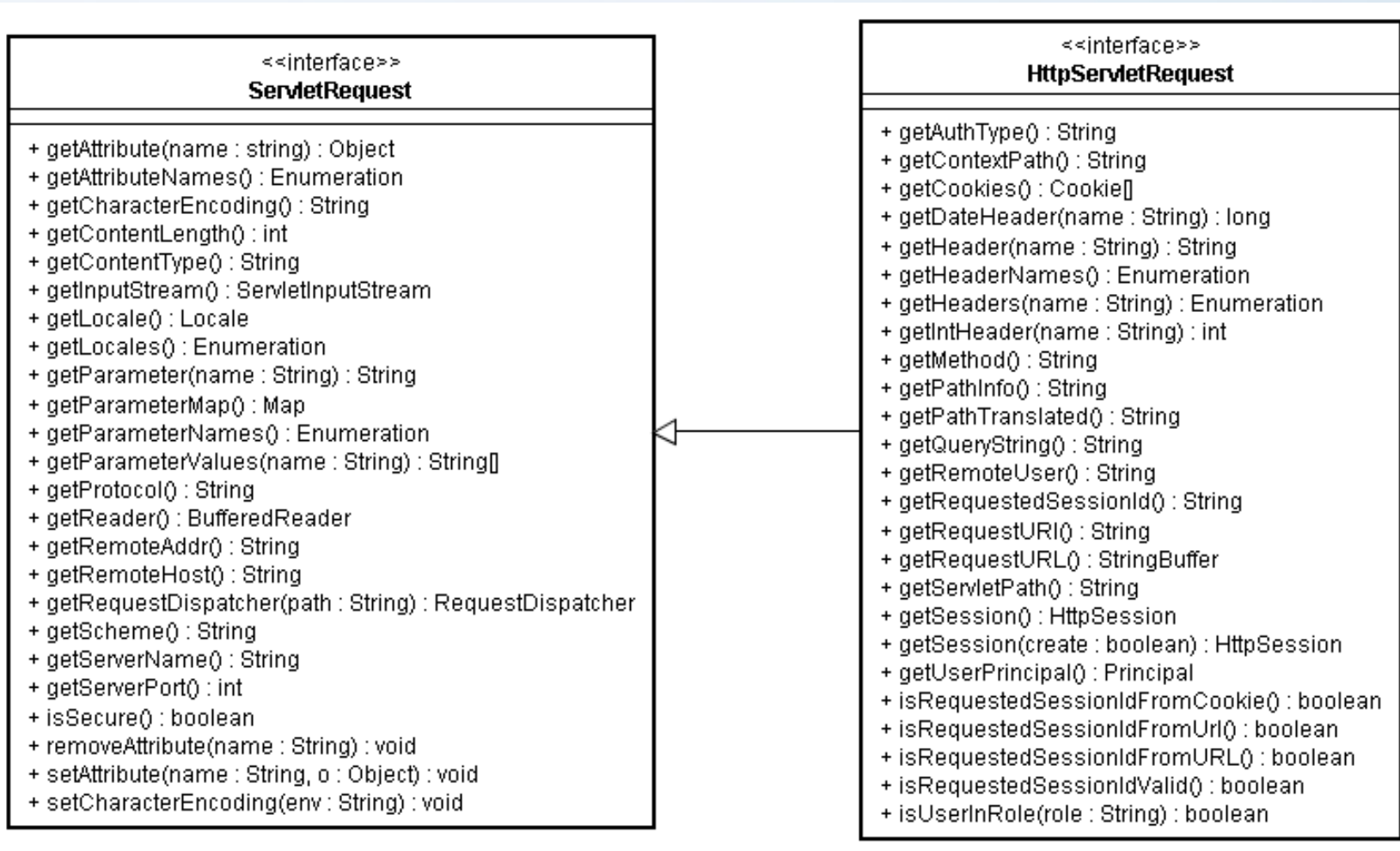


```

import java.io.*; import javax.servlet.*; import javax.servlet.http.*;
public class RegisterServlet extends HttpServlet {
protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException,
IOException {
    response.setContentType("text/html");
    PrintWriter out = response.getWriter();
    out.println("<html>");
    out.println("<head> <title> The Get Method </title> </head>");
    out.println("<body>");
    out.println("The servlet has received a GET. Now, click the button below.");
    out.println("<form method=\"post\">");
    out.println("<input type=\"submit\" value=\"Submit\" />");
    out.println("</form>");
    out.println("</body>");
    out.println("</html>");
}
protected void doPost(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    response.setContentType("text/html");
    PrintWriter out = response.getWriter();
    out.println("<html>");
    out.println("<head>");
    out.println("<title> The Post Method </title>");
    out.println("</head>");
    out.println("<body>");
    out.println("The servlet has received a Post.Thank you!");
    out.println("</body>");
    out.println("</html>");
}
}

```

# A interface HttpServletRequest



# Obter cabeçalhos

- Métodos:
  - `Enumeration getHeaderNames()`
    - Retorna um `enumeration` com todos os nome de cabeçalhos em uma requisição. Se não houver cabeçalhos retorna uma `enumeration` vazia.
  - `String getHeader(String name)`
    - Retorna o valor de um cabeçalho da requisição como uma `String`. Se a requisição não incluir o cabeçalho especificado por `name`, este método retorna `null`.
    - O nome do cabeçalho é *case insensitive*.

# Obter cabeçalhos

Obtenção de cabeçalhos HTTP a partir de HttpServletRequest:

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.util.Enumeration;

public class HeaderListServlet extends HttpServlet {

    protected void doGet(HttpServletRequest request, HttpServletResponse
response)
        throws ServletException, IOException {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();

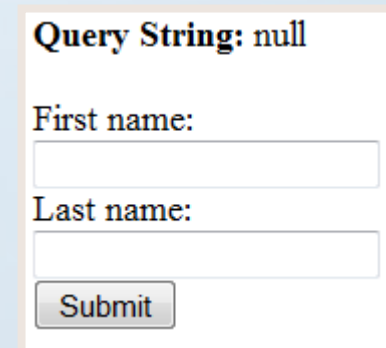
        Enumeration enumeration = request.getHeaderNames();

        while (enumeration.hasMoreElements()){
            String header = (String) enumeration.nextElement();
            out.println(header+": "+request.getHeader(header)+"<br>");
        }
    }
}
```



# Obter string de consulta

- Métodos:
  - `getQueryString()`
    - Retorna uma string de consulta que esta contida na requisição depois do caminho. Este método retorna `null` se a URL não contiver strings de busca.
- Crie um pequeno formulário que envie parâmetros;
- Imprima a string de busca.



**Query String: null**

First name:

Last name:



# Obter string de consulta

## Obtendo a string de consulta...

```
public class HttpRequestDemoServlet extends HttpServlet {  
  
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException,  
        IOException {  
        response.setContentType("text/html");  
        PrintWriter out = response.getWriter();  
        out.println("<html>");  
        out.println("<head>");  
        out.println("<title> Obtaining the Query String </title>");  
        out.println("</head>");  
        out.println("<body>");  
        out.println("<b>Query String: </b>" + request.getQueryString() + "<br/><br/>");  
        out.println("<form method=\"GET\">");  
        out.println("First name: <br/>");  
        out.println("<input type=\"text\" name=\"FirstName\"><br/>");  
        out.println("Last name: <br/>");  
        out.println("<input type=\"text\" name=\"LastName\"><br/>");  
        out.println("<input type=\"submit\" value=\"Submit\">");  
        out.println("</form>");  
        out.println("</body>");  
        out.println("</html>");  
    }  
}
```

# Obtenção dos parâmetros

- Métodos:
  - Enumeration `getParameterNames()`
    - Retorna um Enumeration de objetos String contendo os nomes dos parâmetros da requisição. Se a requisição não apresentar parâmetros, o método retorna um Enumeration vazio.
  - String `getParameter(String name)`
    - Retorna o valor do parâmetro como uma string, ou null se o parâmetro não existir.
- Crie um pequeno formulário que envie parâmetros;
- Imprima os parâmetros enviados.

**The request's parameters are:**

**FirstName:** Andre

**LastName:** Luis

First name:

Last name:

Submit

# Obtenção dos parâmetros

Obtenção de parâmetros a partir de HttpServletRequest...

```
public class HttpRequestParameterServlet extends HttpServlet {
protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    response.setContentType("text/html");
    PrintWriter out = response.getWriter();
    out.println("<html>");
    out.println("<head>");
    out.println("<title> Obtaining the Query String </title>");
    out.println("</head>");
    out.println("<body>");
    out.println("<h3>The request's parameters are: </h3>");
    Enumeration parameters = request.getParameterNames();
    while(parameters.hasMoreElements()){
        String parameter = (String)parameters.nextElement();
        out.println("<b>"+parameter+"</b>: "+request.getParameter(parameter)+"<br>");
    }
    out.println("<form method=\"GET\">");
    out.println("First name: <br/>");
    out.println("<input type=\"text\" name=\"FirstName\"><br/>");
    out.println("Last name: <br/>");
    out.println("<input type=\"text\" name=\"LastName\"><br/>");
    out.println("<input type=\"submit\" value=\"Submit\">");
    out.println("</form></body></html>");
}
}
```

# Obtenção de múltiplos parâmetros

- Método
  - `String[] getParameterValues(String name)`
    - Retorna um array de objetos String contendo todos os valores dados por um parâmetro em uma requisição, ou `null` se o parâmetro não existir. Se o parâmetro tem um único valor, o array terá uma única posição.
- Crie no doGet um formulário que envia múltiplos valores por POST que são capturados e impressos no método doPost.

**Select your favorite music:**

☐ Rock

☐ Jazz

☐ Classical

☐ Country

# Obtenção de múltiplos parâmetros

Obtenção de múltiplos parâmetros a partir de HttpServletRequest...

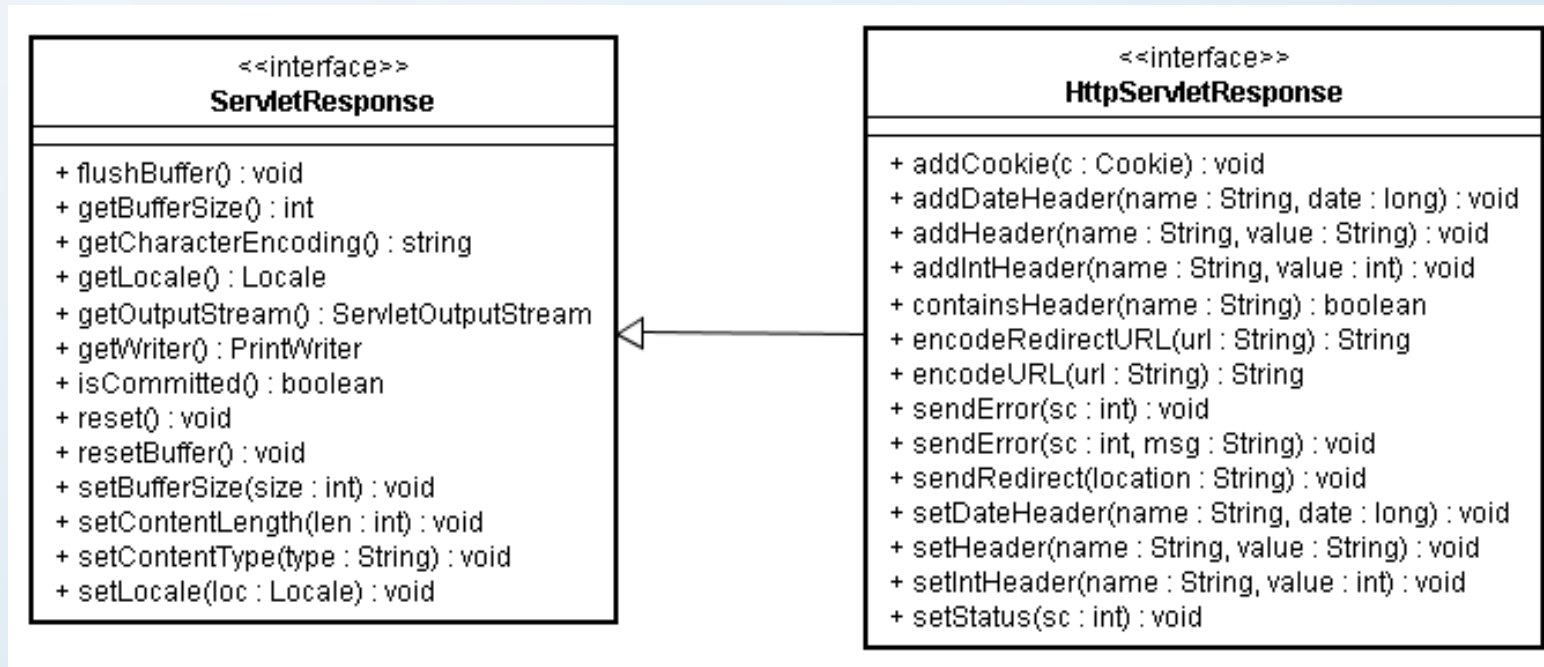
```
public class HttpResquestMultipleParameterServlet extends HttpServlet {
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException
    {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        out.println("<html>");
        out.println("<head>");
        out.println("<title> Obtaining Multi-value Parameters </title>");
        out.println("</head>");
        out.println("<body>");
        out.println("<h3>Select your favorite music: </h3>");
        out.println("<form method=\"post\">");
        out.println("<input type=\"checkbox\" name=\"favoriteMusic\" value=\"Rock\">Rock<br/>");
        out.println("<input type=\"checkbox\" name=\"favoriteMusic\" value=\"Jazz\">Jazz<br/>");
        out.println("<input type=\"checkbox\" name=\"favoriteMusic\" value=\"Class\">Classical<br/>");
        out.println("<input type=\"checkbox\" name=\"favoriteMusic\" value=\"Country\"> Country <br/>");
        out.println("<input type=\"submit\" name=\"Submit\">");
        out.println("</form>");
        out.println("</body>");
        out.println("</html>");
    }
    (...)
}
```

# Obtenção de múltiplos parâmetros

Continuação...

```
(...)  
protected void doPost(HttpServletRequest request, HttpServletResponse  
response)  
    throws ServletException, IOException {  
    String[] values = request.getParameterValues("favoriteMusic");  
    response.setContentType("text/html");  
    PrintWriter out = response.getWriter();  
    if (values!=null){  
        int length = values.length;  
        out.println("You have selected:");  
        for (int i=0; i<length; i++){  
            out.println("<br>" + values[i]);  
        }  
    }  
}
```

# A interface HttpServletResponse





# LoginServlet

- Métodos:
  - `void sendRedirect(String location)`
    - Envia uma resposta ao cliente usando a URL especificada por `location`.
    - Este método pode aceitar caminhos relativos. O servlet container (tomcat) deve converter o caminho relativo em absoluto antes de enviar ao cliente.
  - `PrintWriter getWriter()`
    - Retorna um objeto `PrintWriter` que pode enviar texto para o cliente. O `PrintWriter` usa o *character encoding* retornado pelo `getCharacterEncoding()`.
    - Default ISO-8859-1.

# LoginServlet

- Vamos criar um servlet que construa um formulário para receber o login e senha de um usuário.
- Se o usuário informar os dados corretamente redirecione para index.html (também deve ser criado).
- Caso os dados não estejam corretos exiba o formulário novamente com uma mensagem de erro.
- Note que os dados do login e senha devem ser tratados pelo método `doPost`.
- Dica: use um método auxiliar.

# LoginServlet

```
public class LoginServlet extends HttpServlet {  
private void sendLoginForm(HttpServletResponse response, boolean withError) throws ServletException, IOException{  
    response.setContentType("text/html");  
    PrintWriter out = response.getWriter();  
    out.println("<html>");  
    out.println("<head>");  
    out.println("<title> Obtaining the Query String </title>");  
    out.println("</head>");  
    out.println("<body>");  
    if (withErrorMessage){  
        out.println("Login failed. Please try again.<br/>");  
    }  
    out.println("<h3> Please enter you user name and password </h3>");  
    out.println("<form method=\"POST\">");  
    out.println("User name: <br/>");  
    out.println("<input type=\"text\" name=\"userName\"><br/>");  
    out.println("Password: <br/>");  
    out.println("<input type=\"password\" name=\"password\"><br/>");  
    out.println("<input type=\"submit\" value=\"Submit\">");  
    out.println("</form>");  
    out.println("</body>");  
    out.println("</html>");  
}  
(...)
```

# LoginServlet

```
(...)  
protected void doGet(HttpServletRequest request, HttpServletResponse response)  
    throws ServletException, IOException {  
    sendLoginForm(response, false);  
}  
  
protected void doPost(HttpServletRequest request, HttpServletResponse response)  
    throws ServletException, IOException {  
    String userName = request.getParameter("userName");  
    String password = request.getParameter("password");  
    if (userName!=null && password!=null &&  
        userName.equals("admin") && password.equals("admin")) {  
        response.sendRedirect("/index.html");  
    } else {  
        sendLoginForm(response, true);  
    }  
}  
}
```

# LoginServletWithSendError

- Método:
  - void **sendError**(int sc, String msg)
  - Envia uma resposta para o cliente usando um status específico.

# LoginServletWithSendError

```
public class LoginServletWithSendError extends HttpServlet {
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException,
IOException {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        out.println("<html><head><title> Obtaining the Query String </title></head>");
        out.println("<body>");
        out.println("<h3> Please enter you user name and password </h3>");
        out.println("<form method=\"POST\">");
        out.println("User name: <br/> <input type=\"text\" name=\"userName\"><br/>");
        out.println("Password: <br/><input type=\"password\" name=\"password\"><br/>");
        out.println("<input type=\"submit\" value=\"Submit\">");
        out.println("</form></body></html>");
    }
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
        String userName = request.getParameter("userName");
        String password = request.getParameter("password");
        if (userName!=null && password!=null && userName.equals("admin") && password.equals("admin")){
            response.sendRedirect("/Aula04/");
        } else{
            response.sendError(HttpServletResponse.SC_FORBIDDEN, "Login failed.");
        }
    }
}
```

# SpecialCharacterServlet

In HTML, you use  
to change line.

In HTML, you use `<br/ >` to change line.

```
<html>
  <head>
    <title> HTML Tutorial - Changing Line </title>
  </head>
  <body>
    In HTML, you use <br/ > to change line.
  </body>
</html>
```



# SpecialCharacterServlet

```
public class SpecialCharacterServlet extends HttpServlet {  
protected void doGet(HttpServletRequest request, HttpServletResponse response)  
    throws ServletException, IOException {  
    response.setContentType("text/html");  
    PrintWriter out = response.getWriter();  
    out.println("<html>");  
    out.println("<head>");  
    out.println("<title> HTML Tutorial - Changing Line </title>");  
    out.println("</head>");  
    out.println("<body>");  
    out.println(encodeHtmlTag("In HTML, you use <br/ > to change line."));  
    out.println("</body>");  
    out.println("</html>");  
}  
(...)
```

# SpecialCharacterServlet

```
(...)  
public static String encodeHtmlTag(String tag){  
    if (tag == null)  
        return null;  
    StringBuffer encodingTag = new StringBuffer();  
    int length = tag.length();  
    for (int i=0; i<length; i++){  
        char c = tag.charAt(i);  
        if (c == '<'){  
            encodingTag.append("&lt;");  
        }else if (c == '>'){  
            encodingTag.append("&gt;");  
        }else if (c == '&'){  
            encodingTag.append("&amp;");  
        }else if (c == '\"'){  
            encodingTag.append("&quot;");  
        }else if (c == ' '){  
            encodingTag.append("&nbsp;");  
        }else{  
            encodingTag.append(c);  
        }  
    }  
    return encodingTag.toString();  
}  
}
```

# PopulateValueServlet

```
public class PopulateValueServlet extends HttpServlet {
protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    String userName = "Duncan \"The Great\" Young";
    String password = "lo&&lita";
    response.setContentType("text/html");
    PrintWriter out = response.getWriter();
    out.println("<html>");
    out.println("<head>");
    out.println("<title> Populate HTML Element </title>");
    out.println("</head>");
    out.println("<body>");
    out.println("<h3>Your user name and password.</h3>");
    out.println("<form method=\"post\">");
    out.println("<br> User name: <br>");
    out.println("<input type=\"text\" name=\"userName\" value=\"\"+userName+\"\"");
    out.println("<br> Password: <br>");
    out.println("<input type=\"text\" name=\"password\" value=\"\"+password+\"\"");
    out.println("</form>");
    out.println("</body>");
    out.println("</html>");
}
}
```

# RequestDispatcher



- Define um objeto que recebe requisições do cliente e envia qualquer recurso (tal como um servlet, arquivo HTML ou JSP). O container servlet cria o objeto `RequestDispatcher`, que é usado como um envoltório de um recurso localizado em um caminho particular ou dado por um nome.

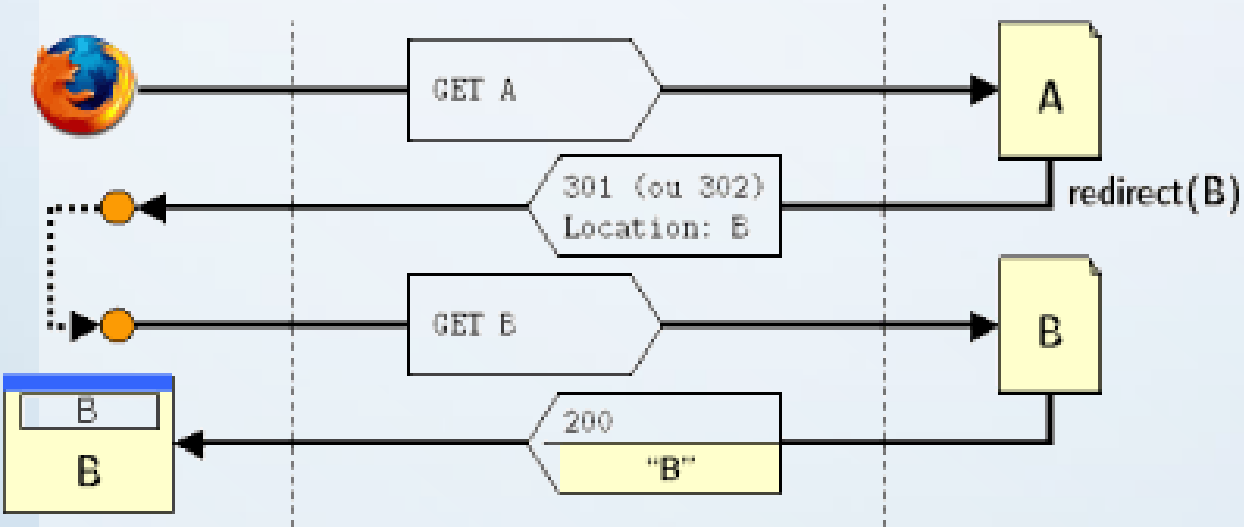
# RequestDispatcher

- Objetos RequestDispatcher servem para repassar requisições para outra página ou servlet.
- Seus dois principais métodos são:
  - `include(request, response)` – inclui a saída e processamento de um recurso no servlet.
  - `forward(request, response)` - repassa a requisição para um recurso.
- Para obter um RequestDispatcher use:
  - `RequestDispatcher rd=request.getRequestDispatcher("url");`
- Para repassar a requisição para outra máquina use:
  - `rd.forward(request, response);`

# sendRedirect vs rq.forward

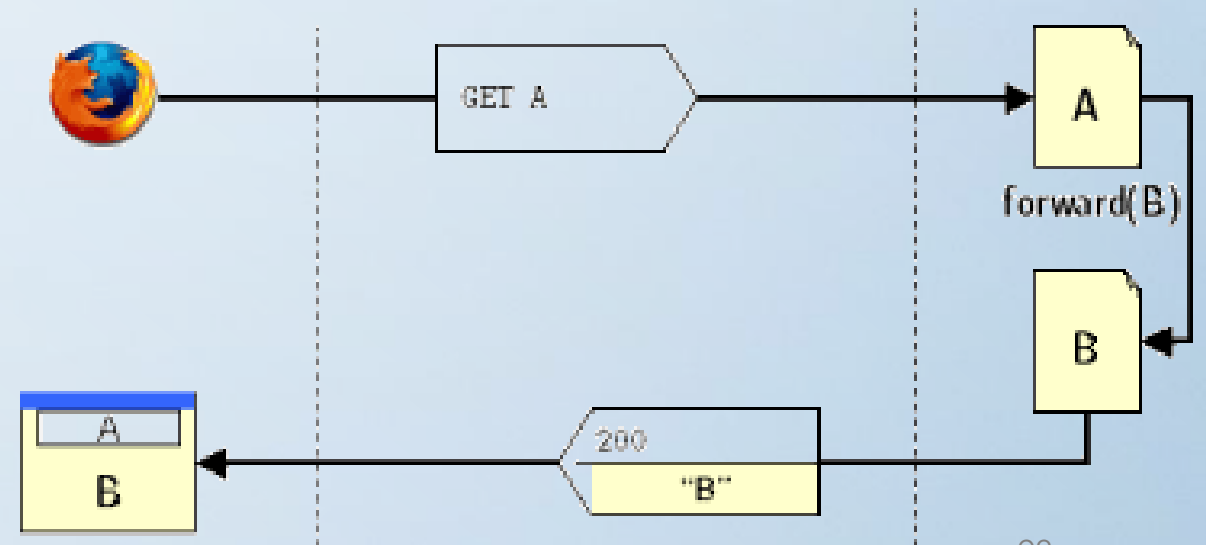
Cliente

Servidor



Cliente

Servidor



Dúvidas

