



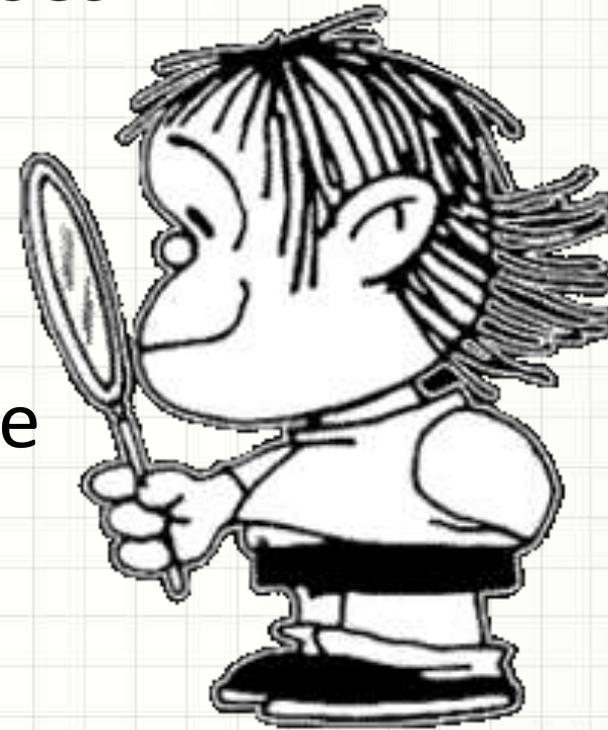
PROGRAMAÇÃO SERVIDOR EM SISTEMAS WEB INTRODUÇÃO À TECNOLOGIA SERVLETS

Prof. Dr. Daniel Caetano

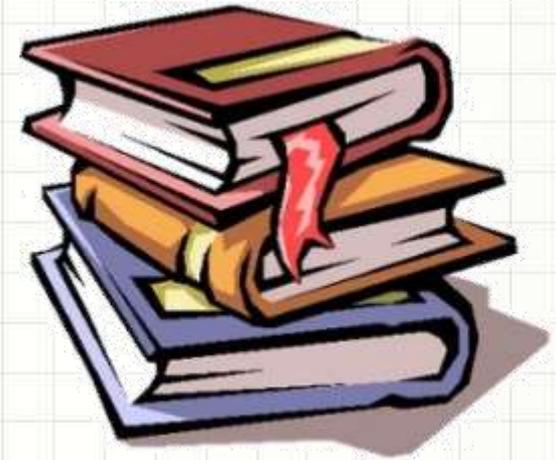
2012 - 1

Objetivos

- Apresentar o conceito aplicações orientada a serviços via web
- Apresentar o papel dos contentores Java
- Capacitar para a construção de Aplicações Web simples



Material de Estudo



Material

Acesso ao Material

Notas de Aula

<http://www.caetano.eng.br/aulas/psw/>
(Aula 4)

Apresentação

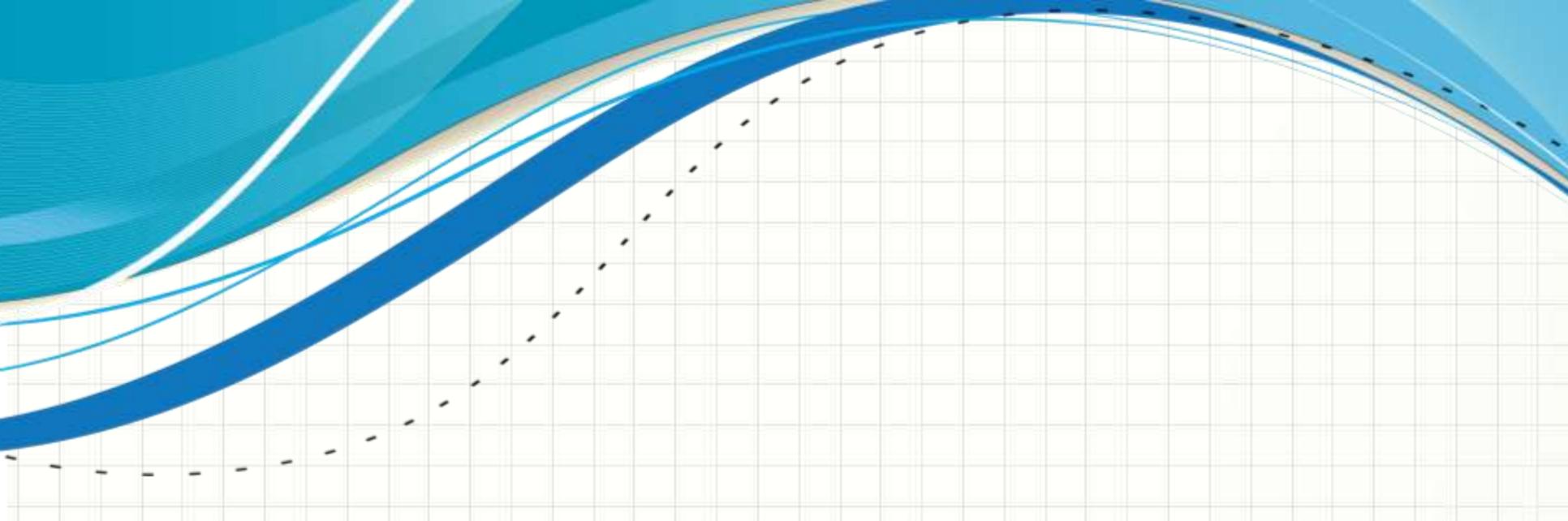
<http://www.caetano.eng.br/aulas/psw/>
(Aula 4)

Material Didático

-

Java: Como
Programar

(6ª Edição) Páginas 928 a 948



ARQUITETURA WEB E SERVIÇOS

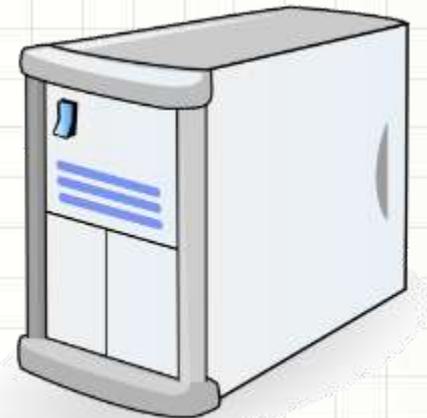
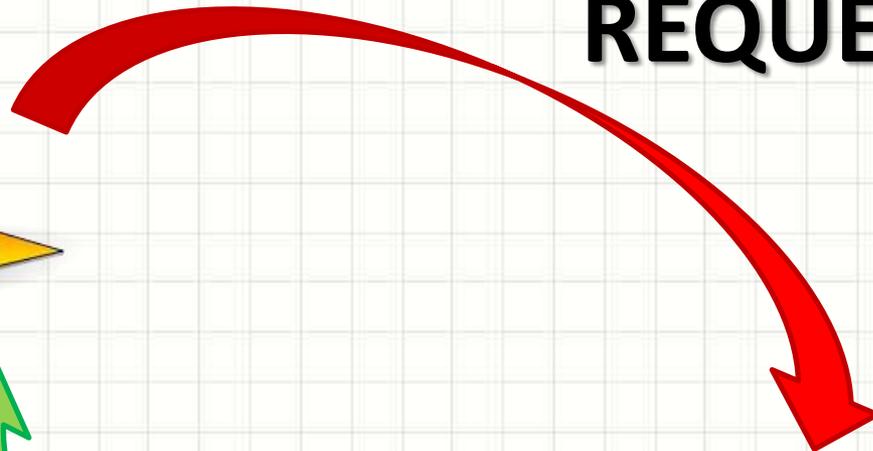
Arquitetura Web e Serviços

- Na primeira aula, vimos esse sistema:



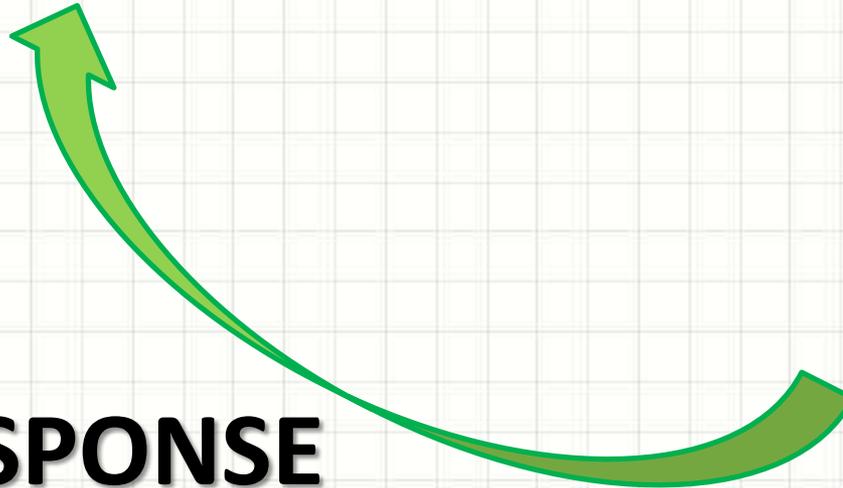
Cliente

REQUEST



Servidor

RESPONSE



Arquitetura Web e Serviços

- Quem é responsável por gerar a **request**?
- O computador **cliente**
 - Navegador
 - Outro programa qualquer
- Quem é responsável por gerar a **response**?
- O computador **servidor**
 - Programa servidor
- O **servidor** presta um **serviço** ao **cliente**

Arquitetura Web e Serviços

- **O cliente: navegador**
- Requisições são geradas quando
 - Digitamos uma nova URL
 - Clicamos em um link
 - Enviamos um formulário
- **O servidor: aplicativo** em computador 24/7
- Ele recebe **requests** e devolve **responses**
 - Apenas quando uma requisição chega ele atua
- É este tipo de aplicativo que iremos desenvolver!

Arquitetura Web e Serviços

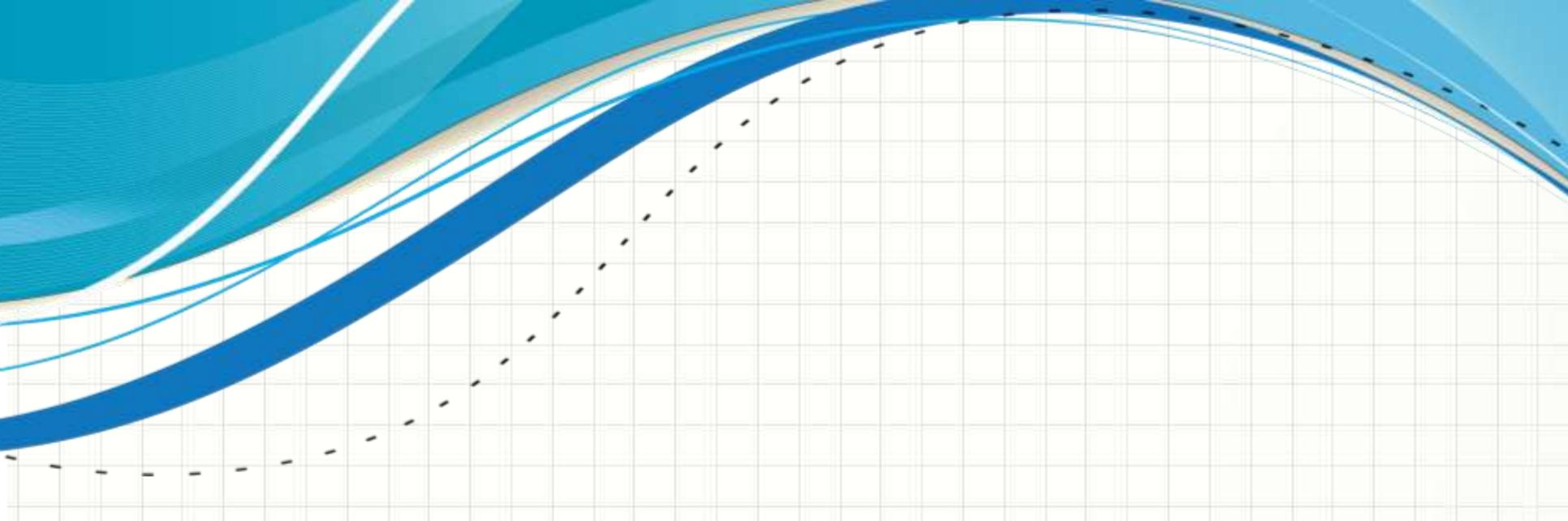
- Bem, faz sentido ter um servidor dedicado a um aplicativo, apenas?
- A resposta é: **depende!**
- Depende de quê?
 - Da demanda sobre estes serviços!
- É comum ter **servidores web** e **servidores de banco de dados** dedicados
- Se a demanda de um aplicativo web não é grande, não faz sentido ter toda uma infraestrutura só para ele...

Arquitetura Web e Serviços

- Vários aplicativos web na mesma máquina
 - **Comum**
- No caso do Java, estes pequenos Aplicativos Web são chamados de **servlets**
- Agora... que tal centralizar algumas funcionalidades, como o gerenciamento de conexões?
- Esse é o papel do **contentor Java**

Arquitetura Web e Serviços

- Existem vários contentores Java
 - GlassFish
 - TomCat
 - TomCat + JBoss
 - Dentre outros...
- Neste curso, usaremos o **GlassFish**
 - É o mais completo
 - É o padrão sugerido pela Oracle



O CONTENTOR JAVA E O CICLO DE VIDA

O Contentor Java

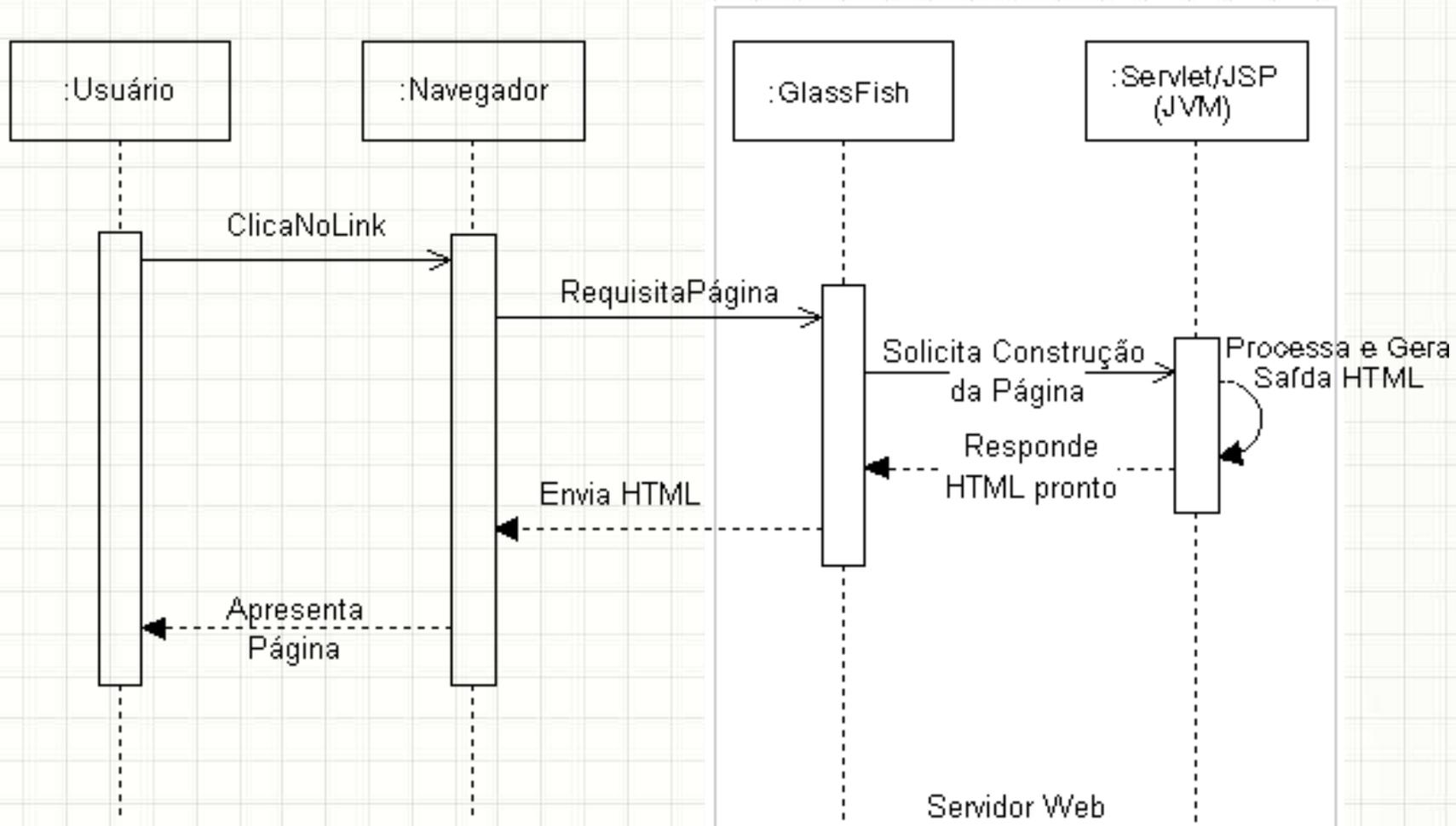
- O contentor Java também é conhecido como **servidor de aplicações Java**
- Gerencia a execução de serviços web Java
- Disponibilizar infraestrutura necessária para estes serviços
- Permite a comunicação com outros componentes Java e aplicações

O Contentor Java

- Servidor de Aplicações - Funcionamento básico:
 1. Aguarda pelas requests
 2. Repassa a request para o **servlet** correto
 3. Recebe a response do **servlet**
 4. Devolve a response para o cliente que enviou a request
- Ou seja, ele é responsável por gerenciar as conexões (dentre outras coisas)

O Contendor Java

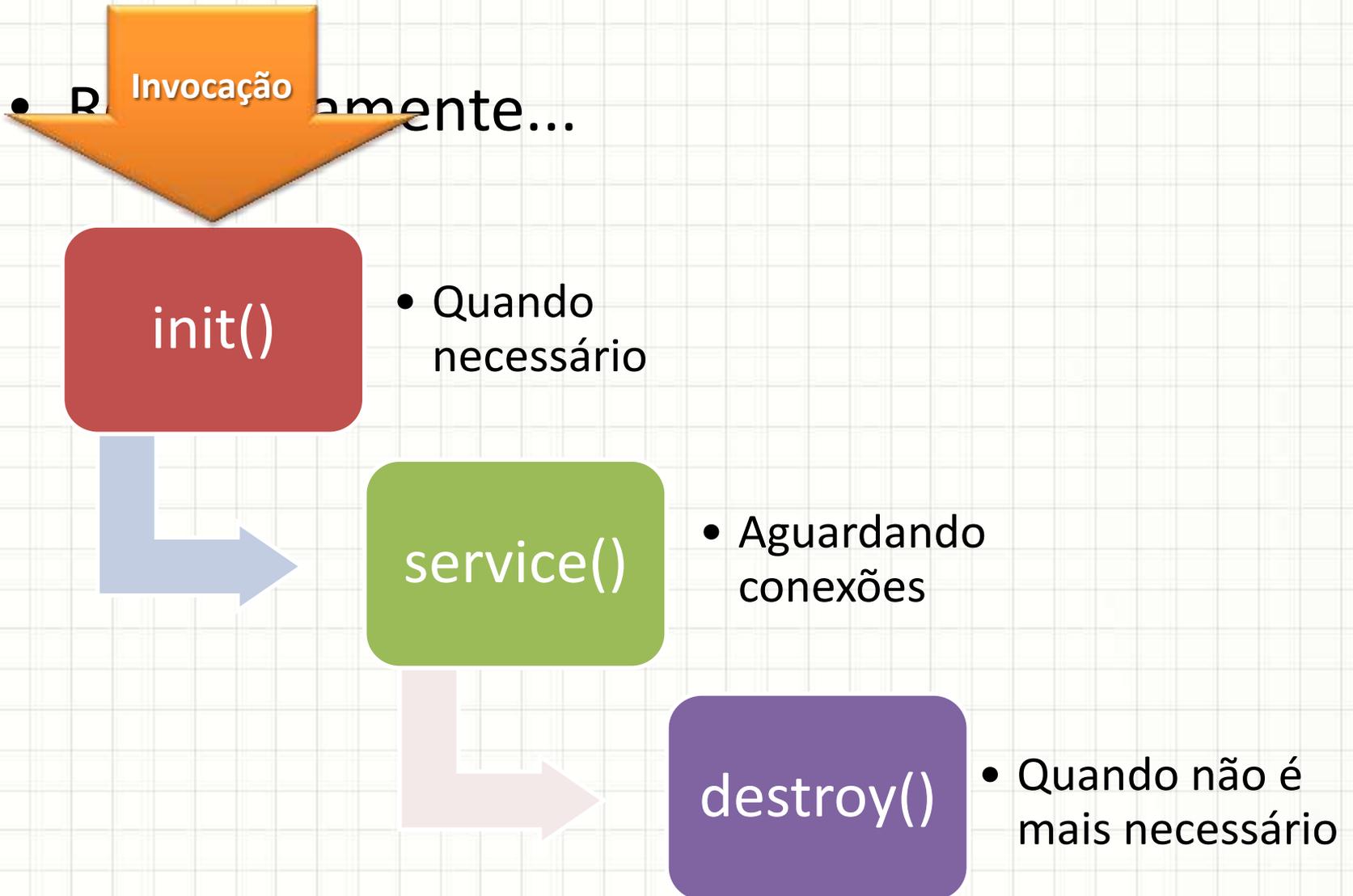
- Resumidamente...

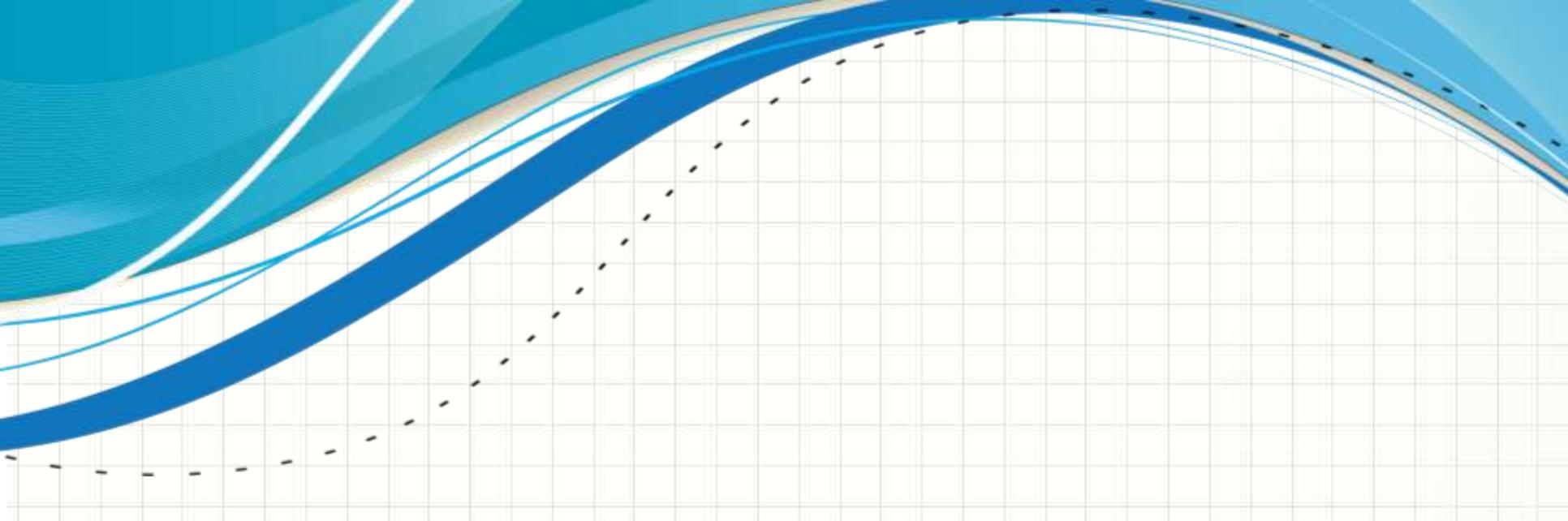


O Contentor Java

- Mas o servidor de aplicativos não faz apenas isso... Dentre outras coisas, ele também cuida do **ciclo de vida do servlet**
 1. Quando necessário, inicia o Servlet
 2. Coloca este Servlet ativo
 3. Finaliza o Servlet quando não é mais necessário

Ciclo de Vida do Servlet

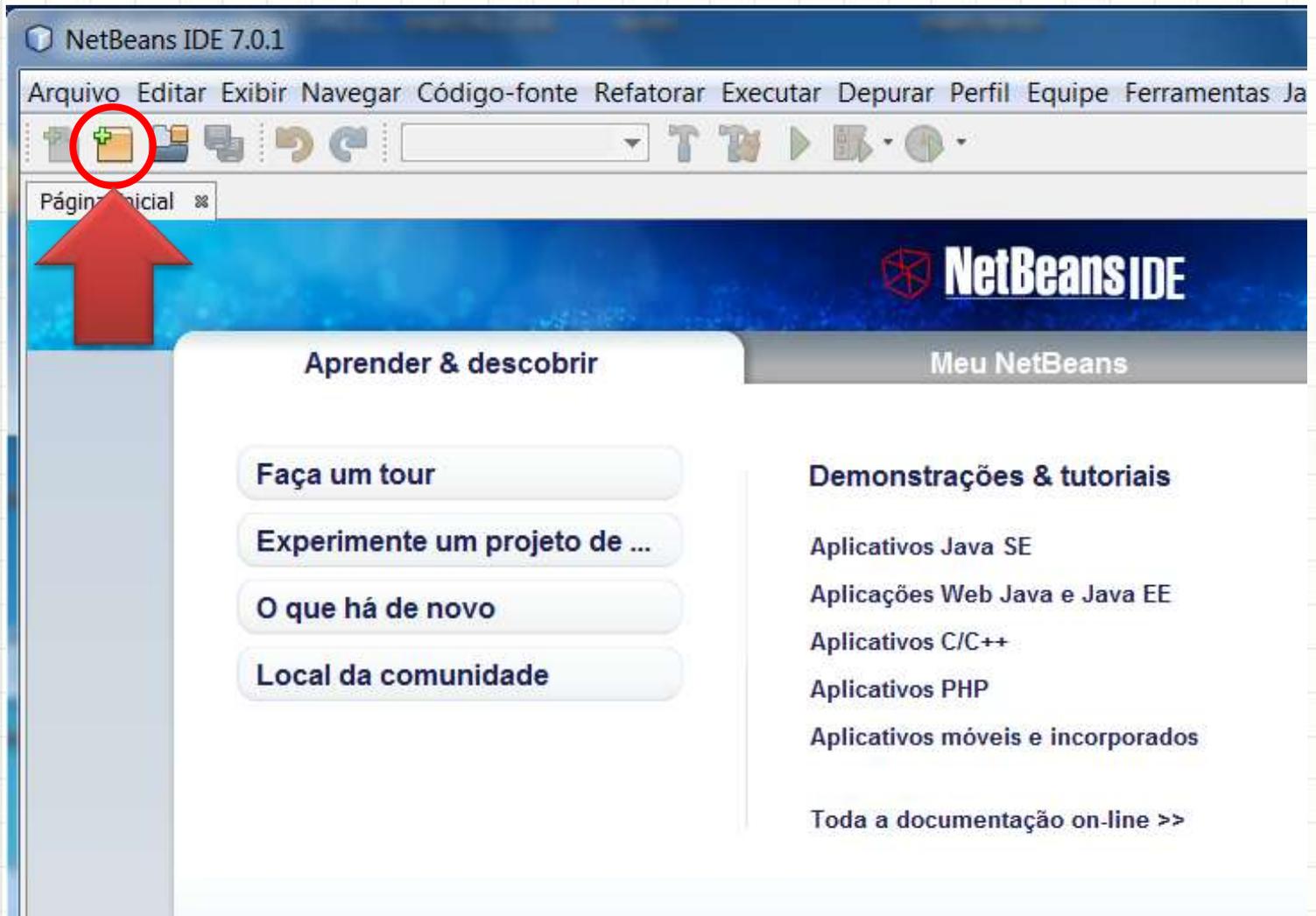




**SERVLETS NO
NETBEANS**

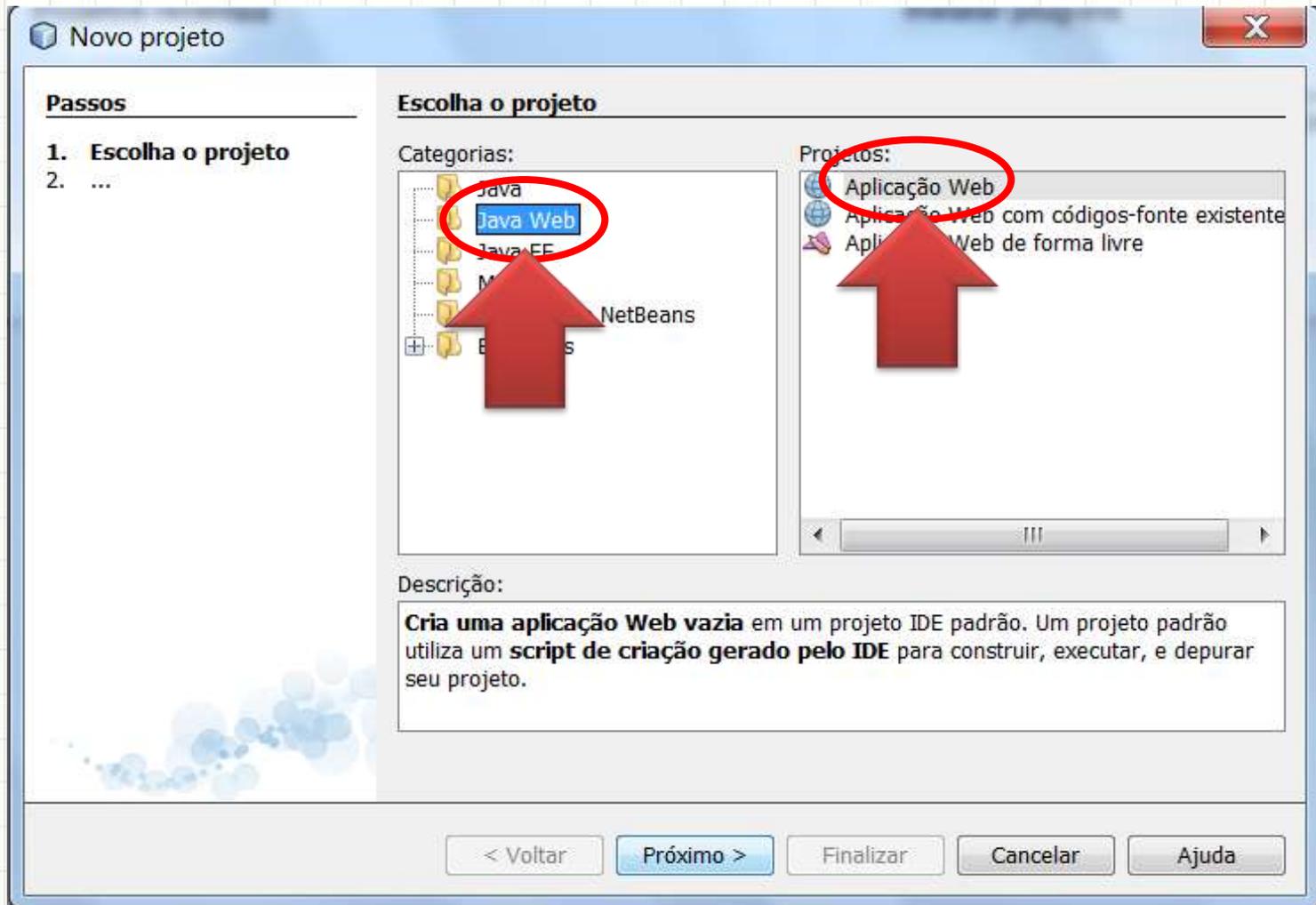
Criando um Servlet

- Iniciaremos criando um projeto



Criando um Servlet

- Escolha o tipo: Java Web e Aplicação Web



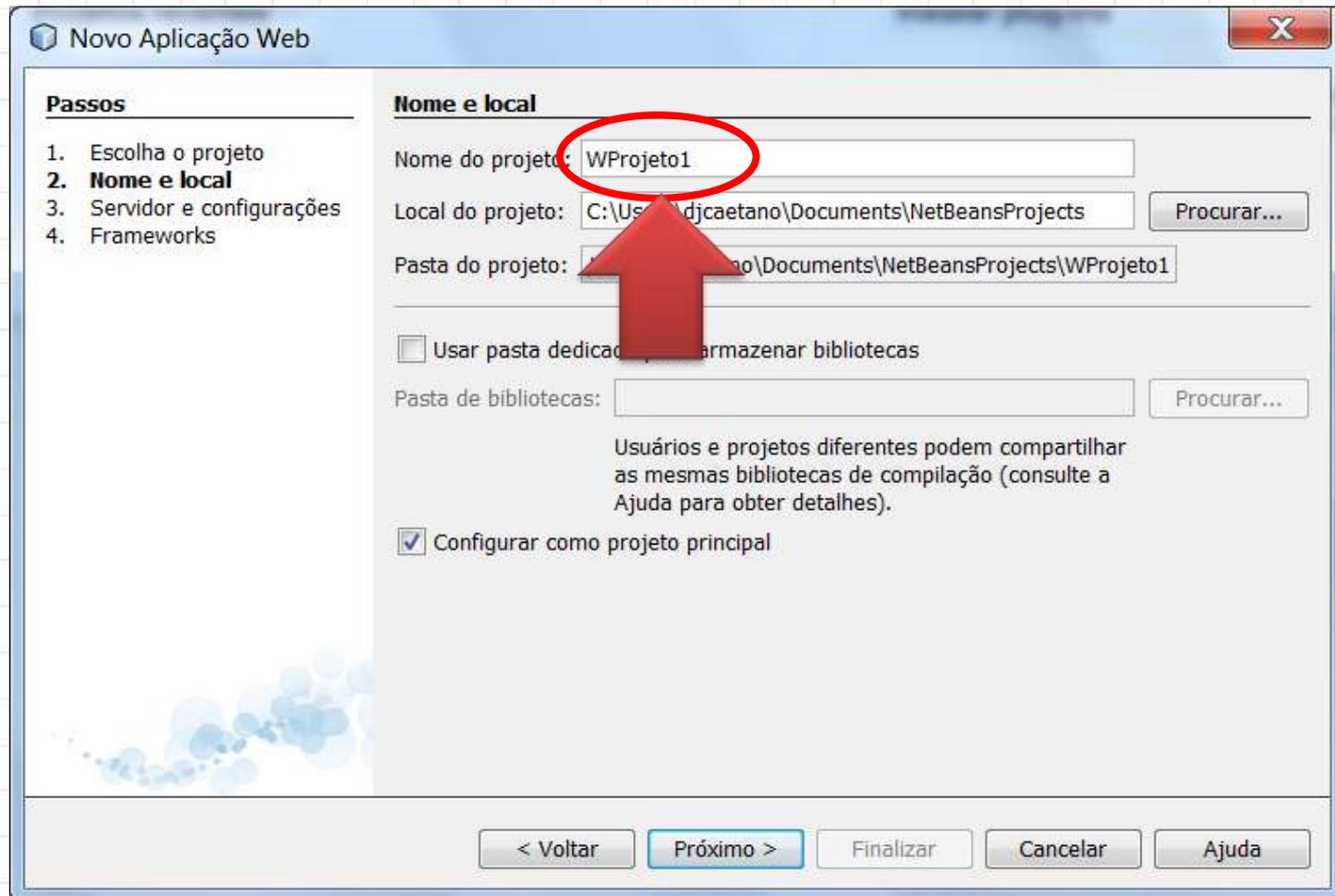
Criando um Servlet

- Escolha o tipo: Java Web e Aplicação Web



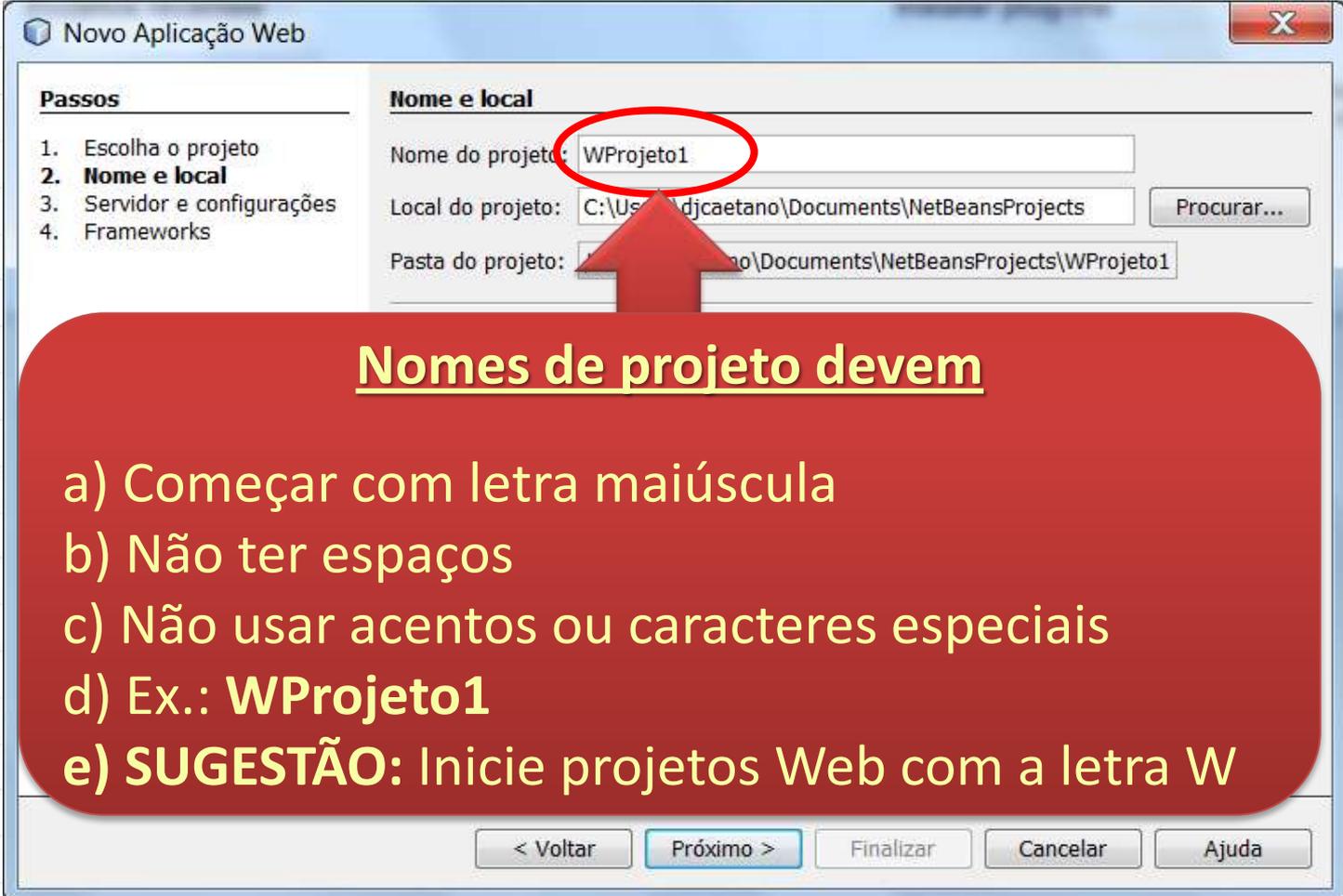
Criando um Servlet

- Dê um nome ao seu projeto: ex.: **WProjeto1**



Criando um Servlet

- Dê um nome ao seu projeto: ex.: **WProjeto1**

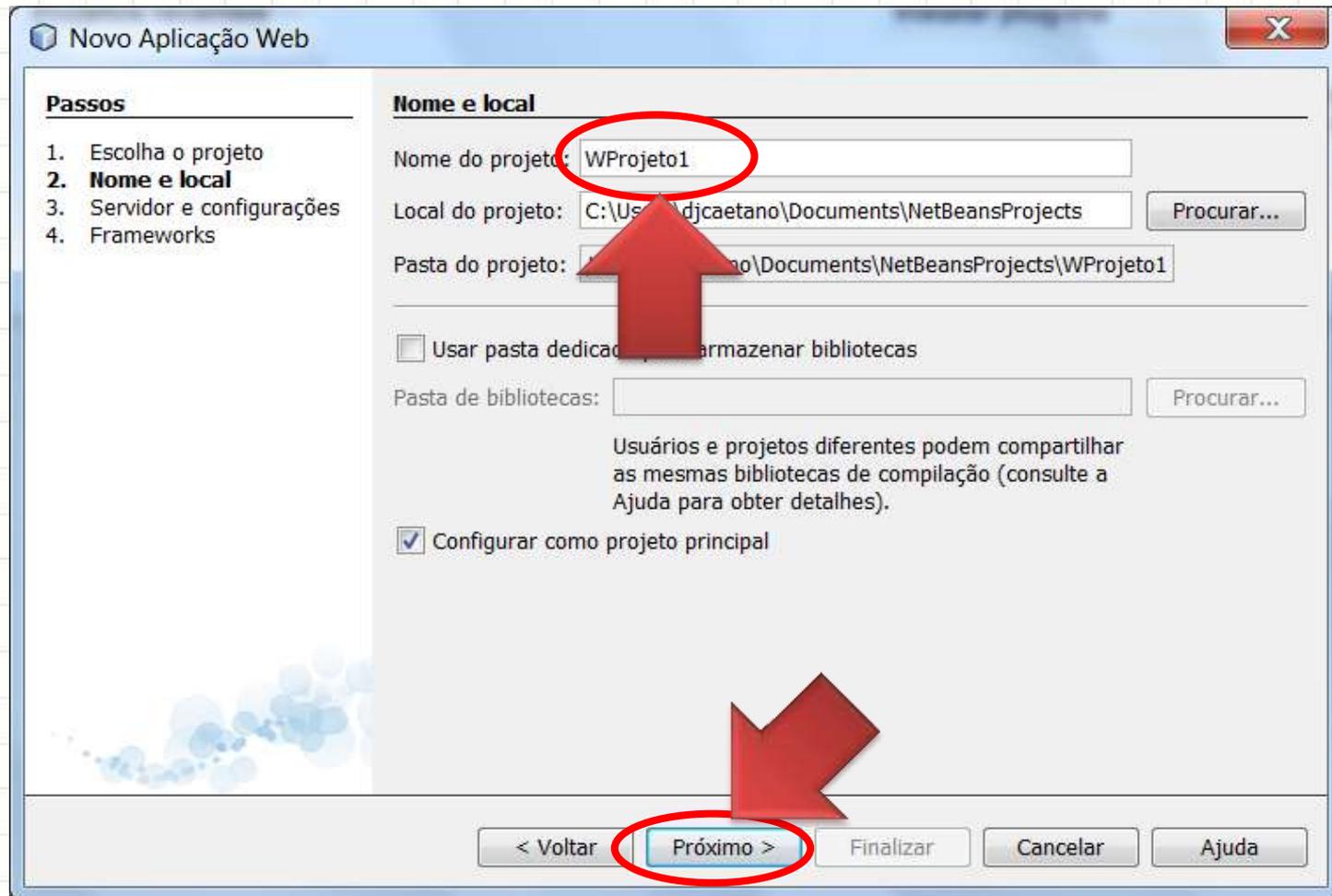


Nomes de projeto devem

- a) Começar com letra maiúscula
- b) Não ter espaços
- c) Não usar acentos ou caracteres especiais
- d) Ex.: **WProjeto1**
- e) **SUGESTÃO**: Inicie projetos Web com a letra W

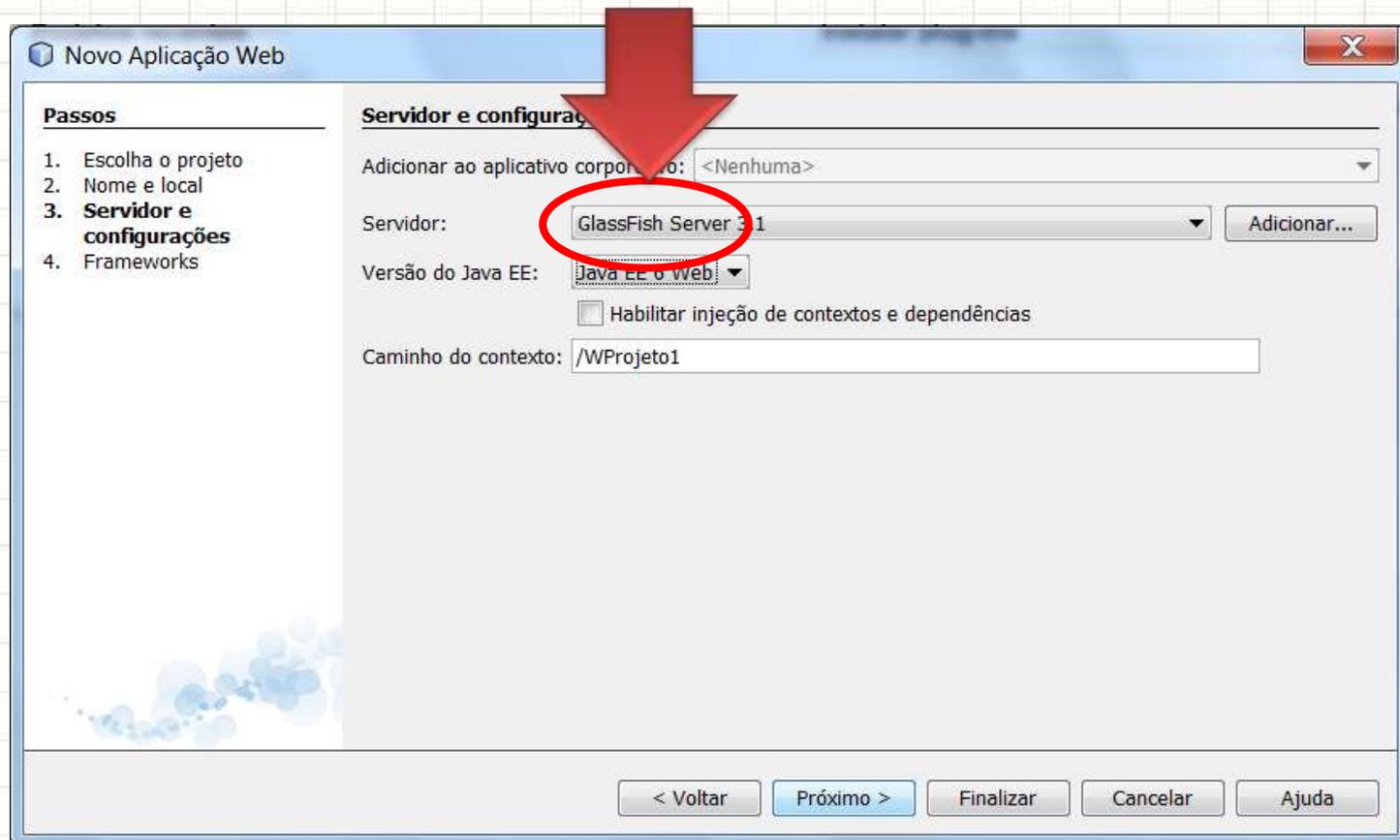
Criando um Servlet

- Dê um nome ao seu projeto: ex.: **WProjeto1**



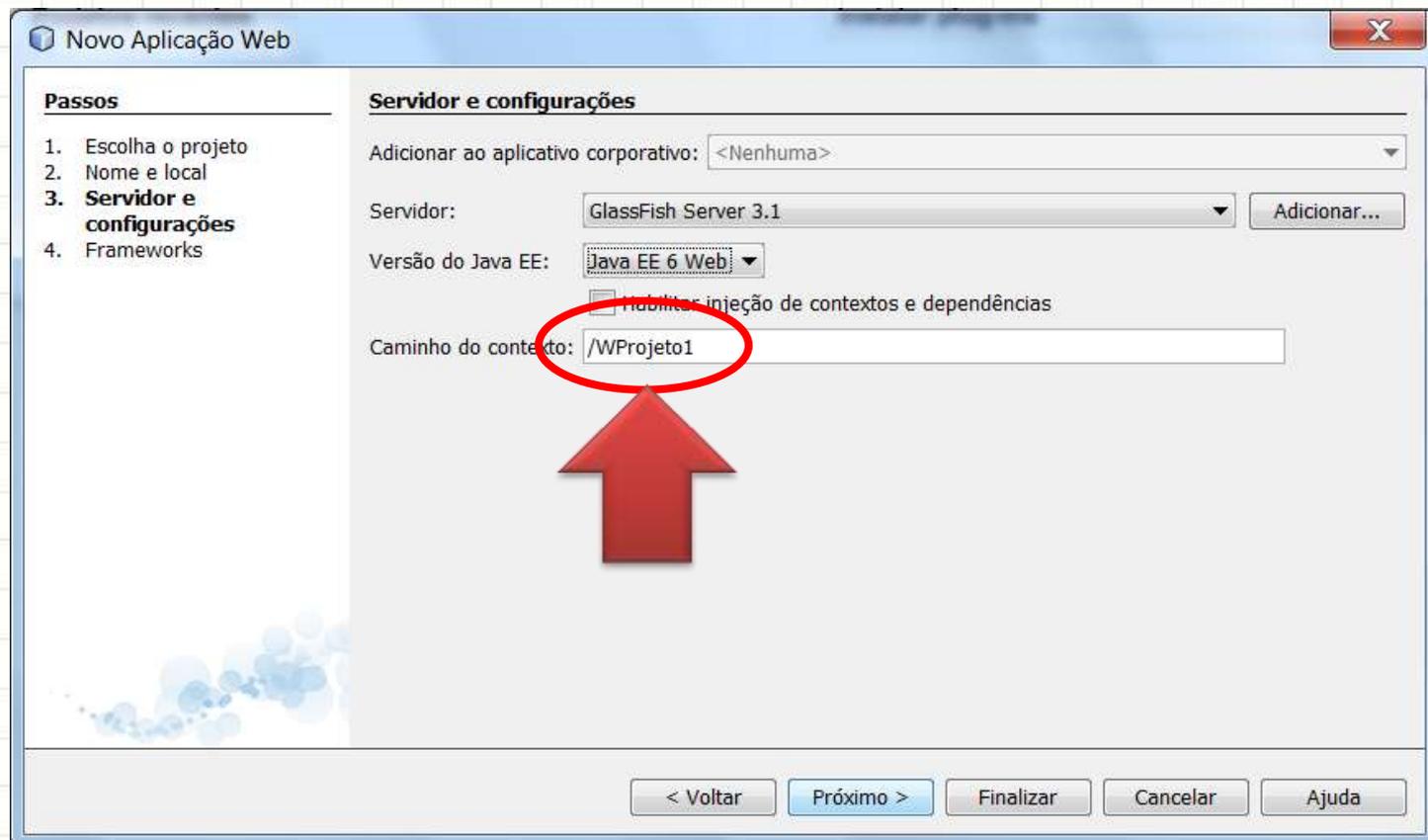
Criando um Servlet

- Selecione o contentor: **GlassFish**



Criando um Servlet

- Este é o endereço da aplicação: **/WProjeto1**



Criando um Servlet

- Clique em **Finalizar**

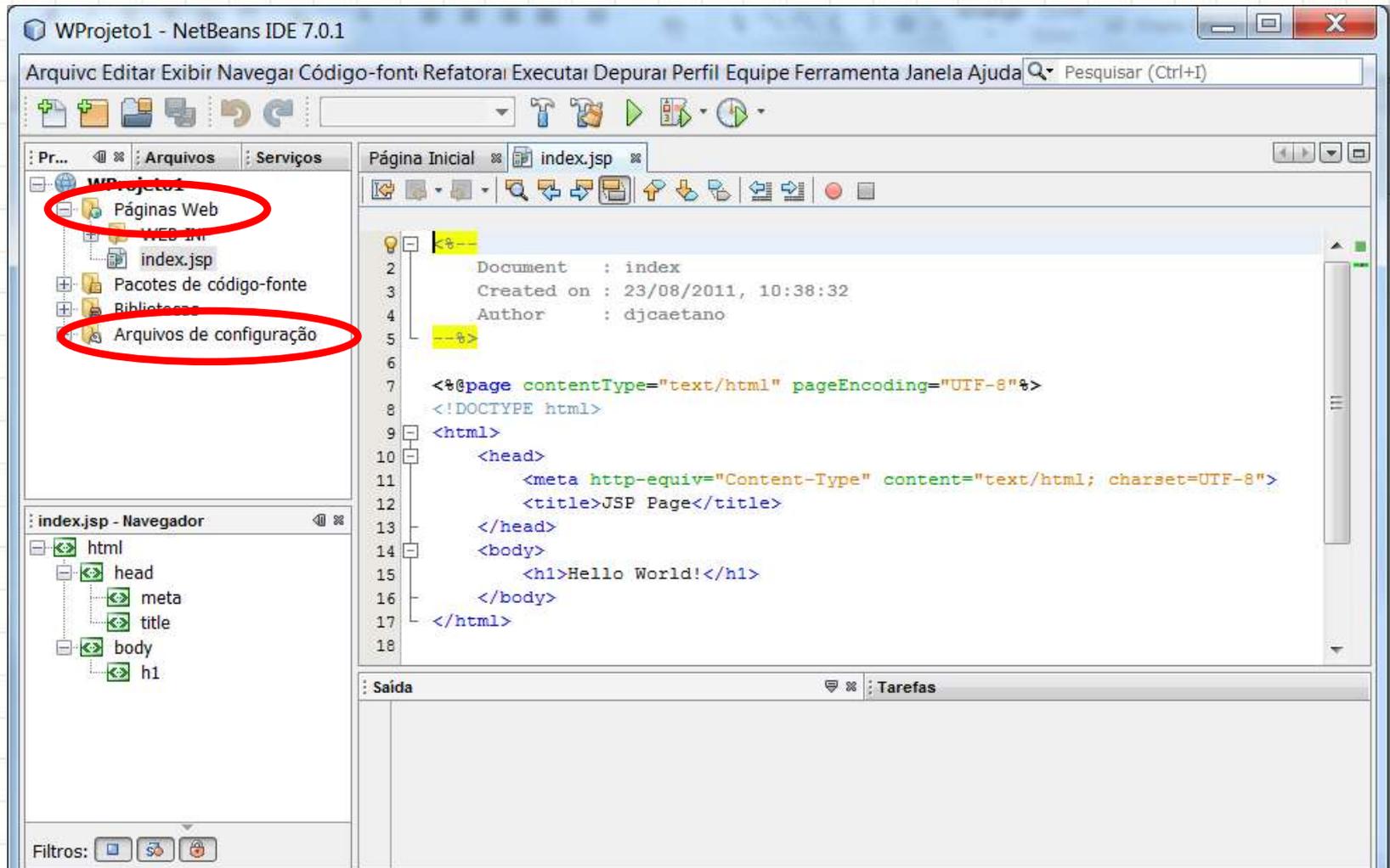
ATENÇÃO

Caso apareça uma janela solicitando **autorização** para acesso, responda **Permitir**



Criando um Servlet

- Esta tela vai aparecer... Há campos novos!



WProjeto1 - NetBeans IDE 7.0.1

Arquivar Editar Exibir Navegar Código-fonte Refatorar Executar Depurar Perfil Equipe Ferramenta Janela Ajuda Pesquisar (Ctrl+I)

Pr... Arquivos Serviços

WProjeto1

- Páginas Web
- WEB-INF
- index.jsp
- Pacotes de código-fonte
- Bibliotecas
- Arquivos de configuração

index.jsp - Navegador

- html
- head
- meta
- title
- body
- h1

Página Inicial index.jsp

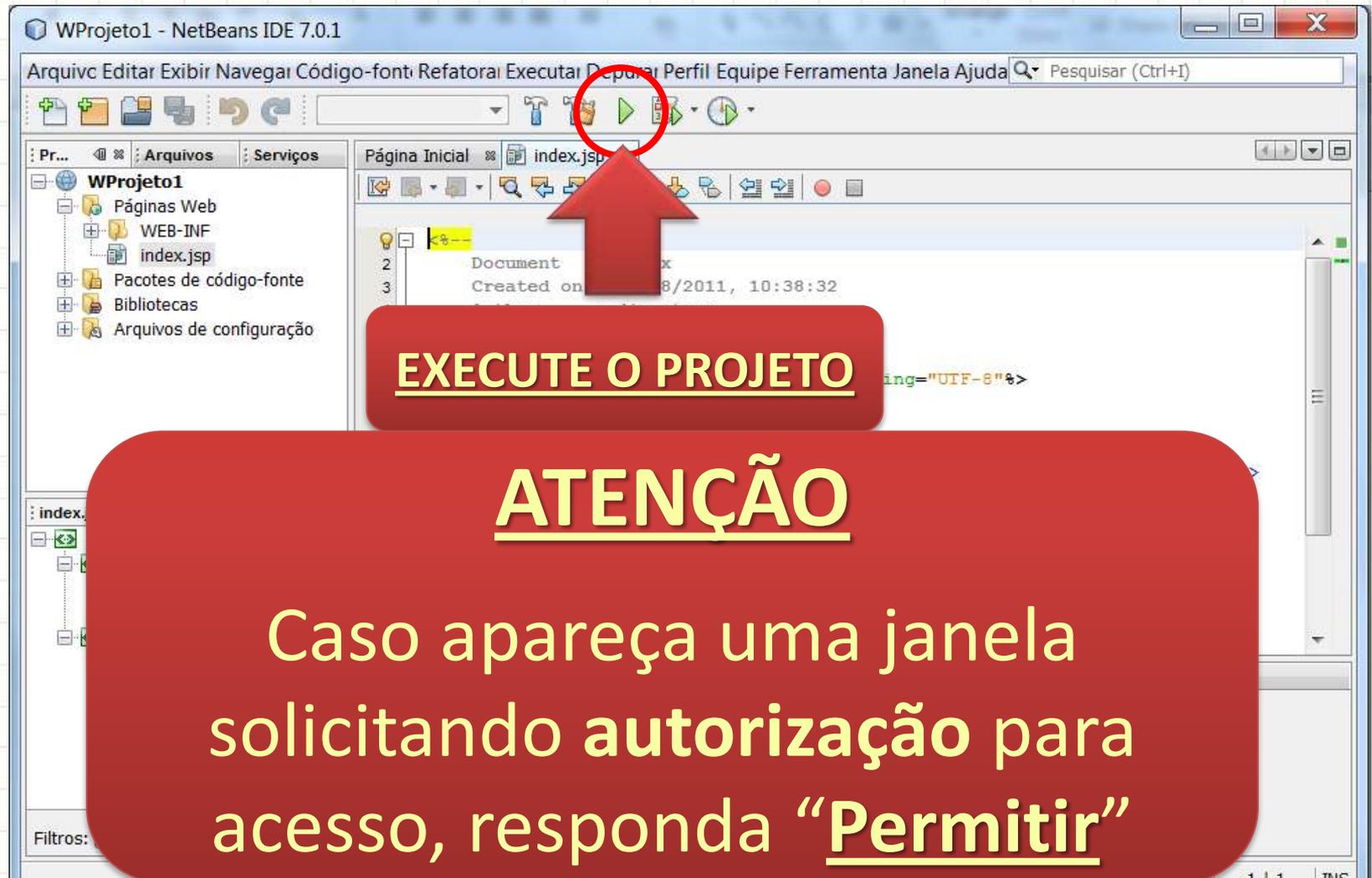
```
1 <!--
2 Document : index
3 Created on : 23/08/2011, 10:38:32
4 Author : djcaetano
5 -->
6
7 <%@page contentType="text/html" pageEncoding="UTF-8"%>
8 <!DOCTYPE html>
9 <html>
10 <head>
11 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
12 <title>JSP Page</title>
13 </head>
14 <body>
15 <h1>Hello World!</h1>
16 </body>
17 </html>
18
```

Saída Tarefas

Filtros: [] [] []

Criando um Servlet

- Execute o projeto e veja o que ocorre!



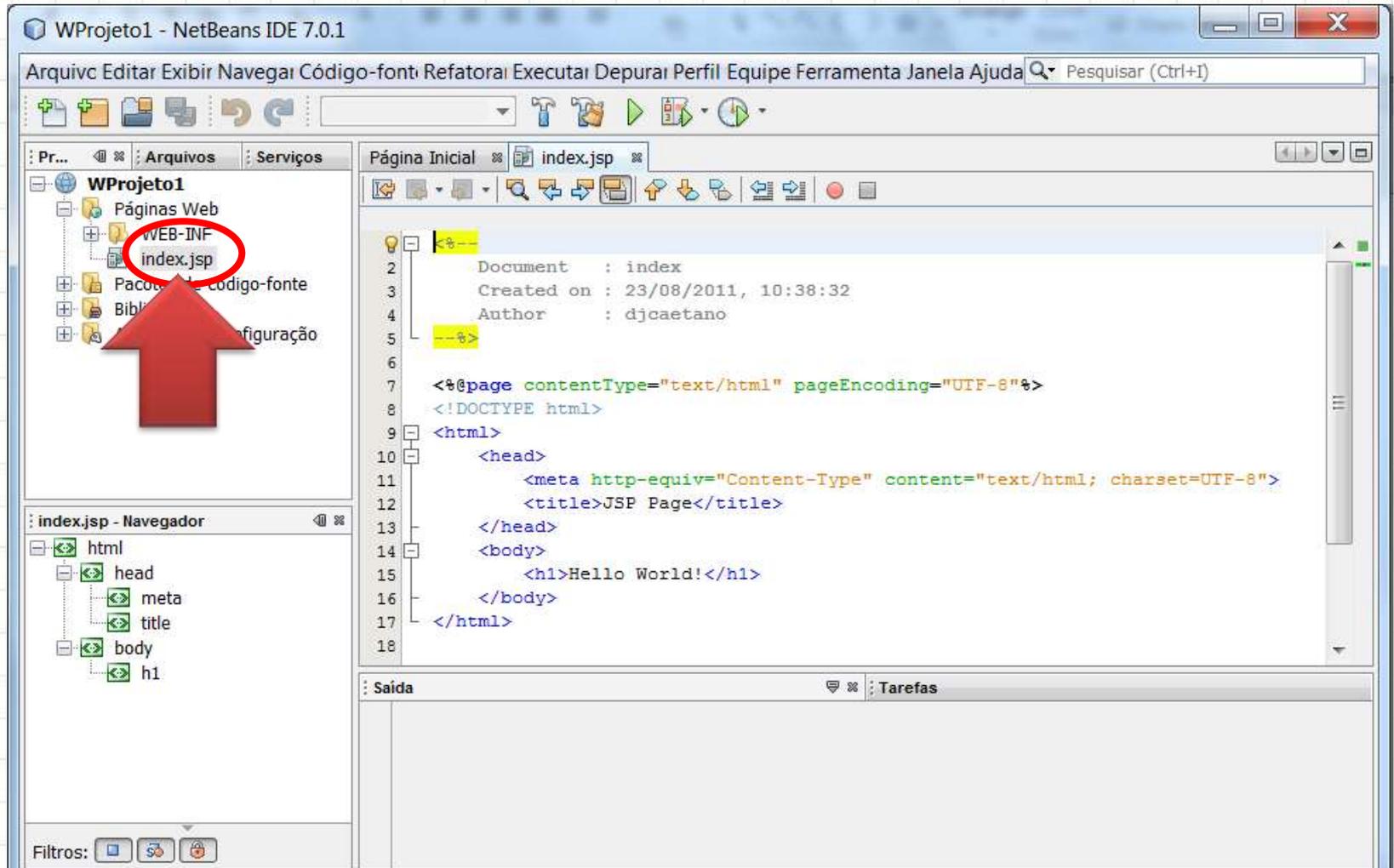
EXECUTE O PROJETO

ATENÇÃO

Caso apareça uma janela solicitando autorização para acesso, responda **Permitir**

Criando um Servlet

- Por padrão, o **index.jsp** é executado



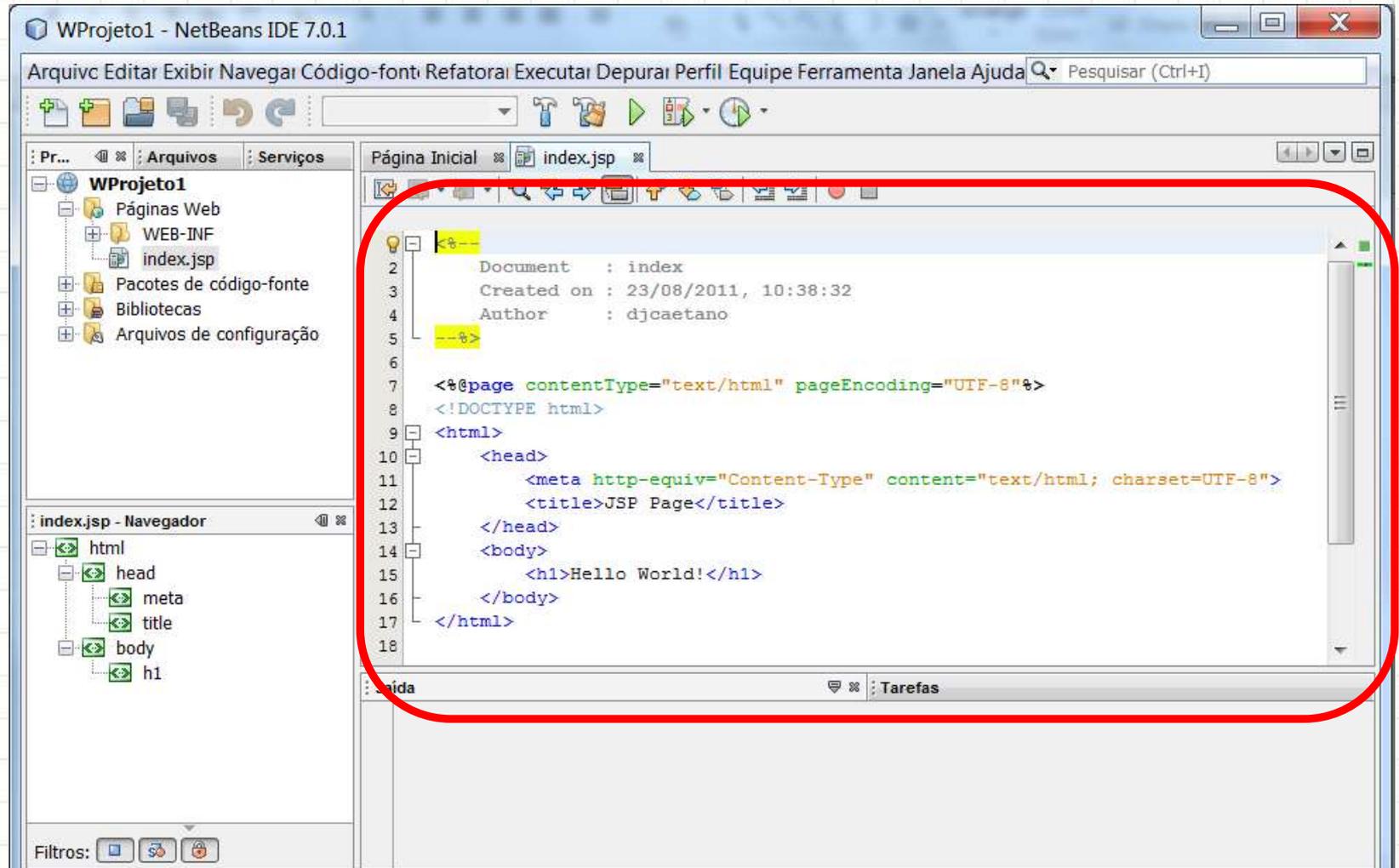
The screenshot shows the NetBeans IDE 7.0.1 interface. The main window is titled "WProjeto1 - NetBeans IDE 7.0.1". The menu bar includes "Arquivar", "Editar", "Exibir", "Navegar", "Código-fonte", "Refatorar", "Executar", "Depurar", "Perfil", "Equipe", "Ferramenta", "Janela", and "Ajuda". The toolbar contains various icons for file operations and development. The "Arquivos" (Files) view on the left shows the project structure for "WProjeto1", including "Páginas Web", "WEB-INF", "index.jsp", "Pacotes de código-fonte", "Bibliotecas", and "Configuração". The "index.jsp" file is highlighted with a red circle, and a large red arrow points to it. The "index.jsp - Navegador" (index.jsp - Navigator) view shows the HTML structure of the file, including "html", "head", "meta", "title", "body", and "h1". The main editor window displays the code for "index.jsp", which is a simple JSP page that outputs "Hello World!". The code is as follows:

```
1 <!--
2 Document : index
3 Created on : 23/08/2011, 10:38:32
4 Author : djcaetano
5 -->
6
7 <%@page contentType="text/html" pageEncoding="UTF-8"%>
8 <!DOCTYPE html>
9 <html>
10 <head>
11 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
12 <title>JSP Page</title>
13 </head>
14 <body>
15 <h1>Hello World!</h1>
16 </body>
17 </html>
18
```

The "Saída" (Output) and "Tarefas" (Tasks) views are visible at the bottom of the IDE.

Criando um Servlet

- Vamos construir um formulário...



The screenshot shows the NetBeans IDE 7.0.1 interface. The main editor window displays the code for `index.jsp`. The code is as follows:

```
1 <!--
2 Document : index
3 Created on : 23/08/2011, 10:38:32
4 Author : djcaetano
5 -->
6
7 <%@page contentType="text/html" pageEncoding="UTF-8"%>
8 <!DOCTYPE html>
9 <html>
10 <head>
11 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
12 <title>JSP Page</title>
13 </head>
14 <body>
15 <h1>Hello World!</h1>
16 </body>
17 </html>
18
```

The IDE interface includes a Project Explorer on the left showing the project structure for `WProjeto1`, with `index.jsp` selected under `WEB-INF`. A Navigator window at the bottom left shows the DOM tree for the rendered page, including `html`, `head`, `meta`, `title`, `body`, and `h1`. The status bar at the bottom indicates the current task is `Tarefas`.

Criando um Servlet

- Vamos construir um formulário...

```
WProjeto1 - NetBeans IDE 7.0.1
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Calculadora de Índice de Massa Corporal</title>
  </head>
  <body>
    <h1>Digite seus dados:</h1>
    <form method="post" action="Imc">
      <p>Peso: <input type="text" name="peso" /></p>
      <p>Altura: <input type="text" name="altura" /></p>
      <p><input type="submit" value="Enviar!" /></p>
    </form>
  </body>
</html>
```

Criando um Servlet

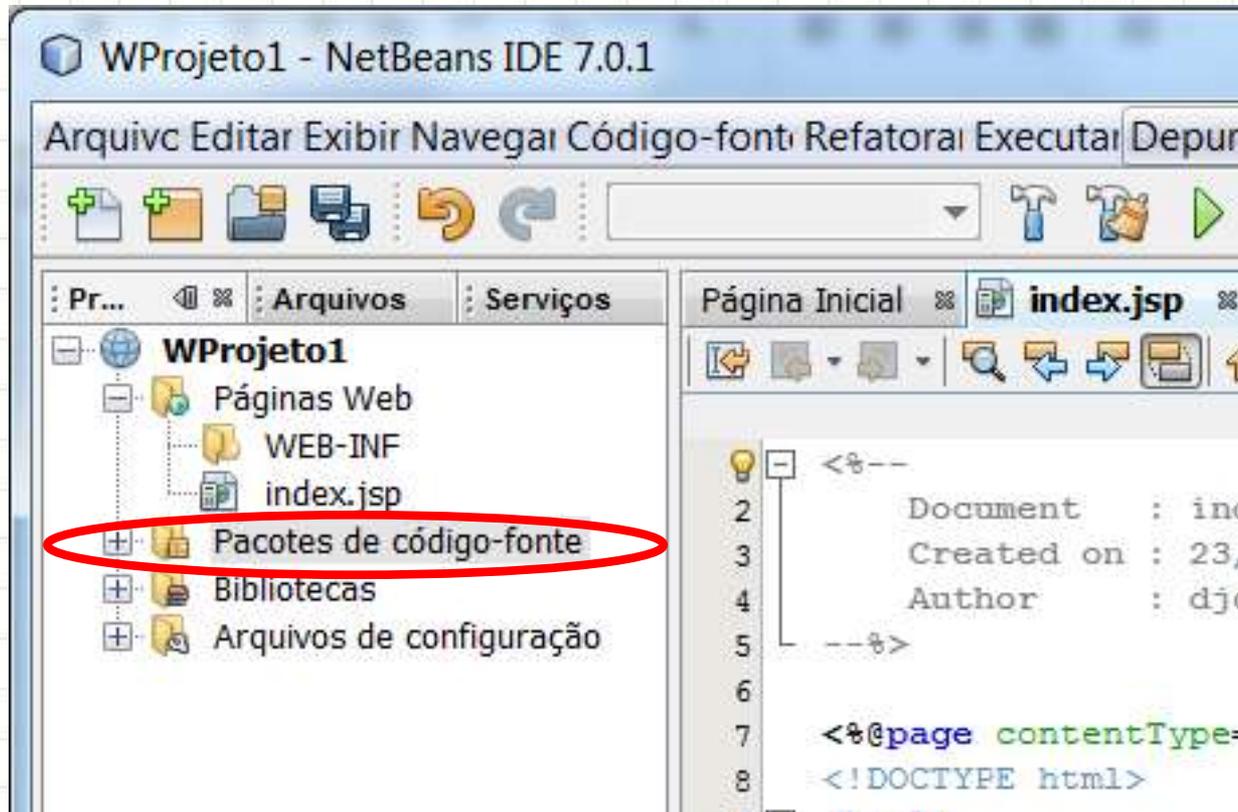
- Va

Isso irá gerar uma **request** para um aplicativo web chamado **Imc**.
Precisamos criá-lo!

```
<%@page
<!DOCTYPE
<html>
<head
</head>
<body>
  <h1>Digite seus dados:</h1>
  <form method="post" action="Imc">
    <p>Peso: <input type="text" name="peso" /></p>
    <p>Altura: <input type="text" name="altura" /></p>
    <p><input type="submit" value="Enviar!" /></p>
  </form>
</body>
</html>
```

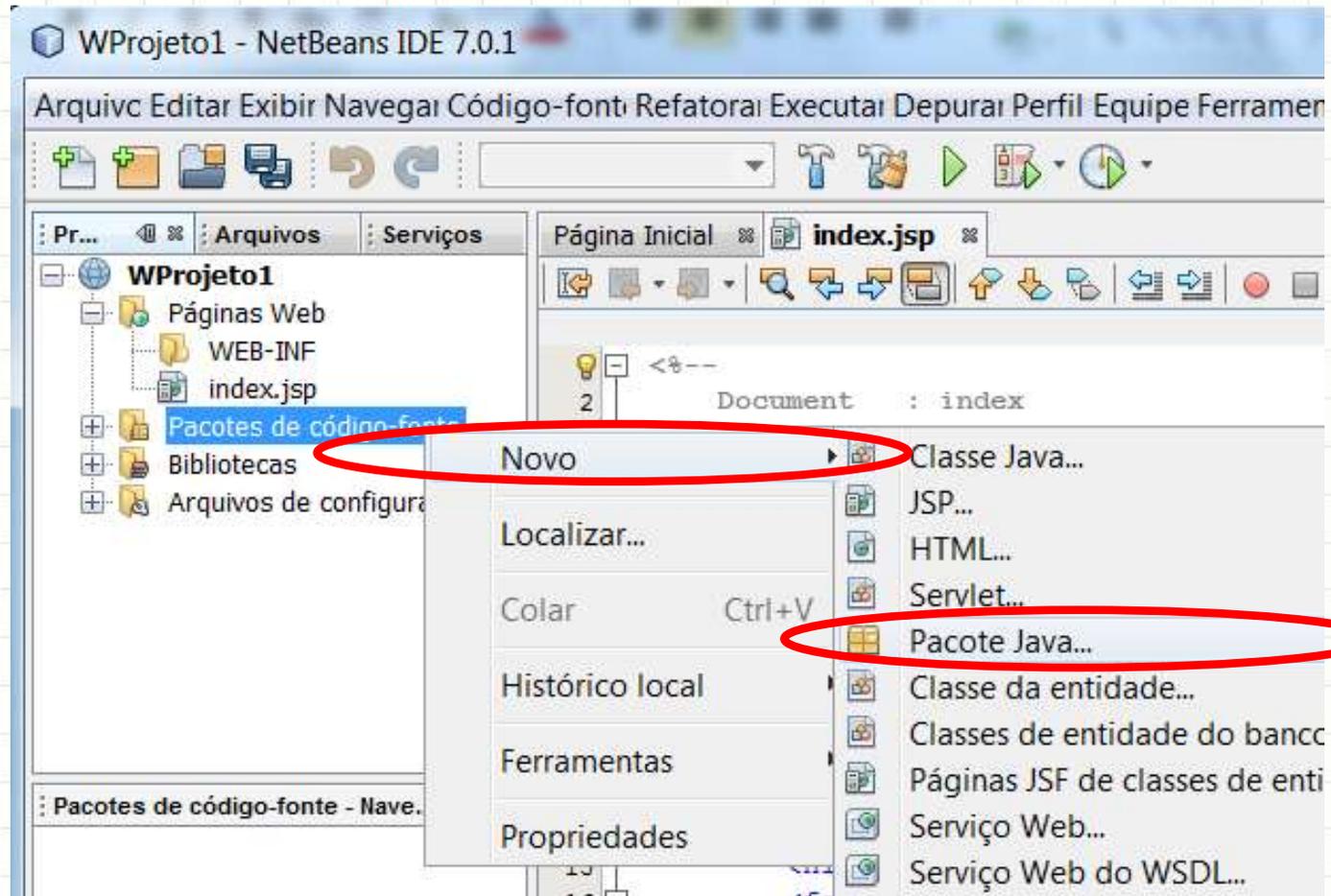
Criando um Servlet

- Clique com o botão direito em “Pacotes de Código Fonte”



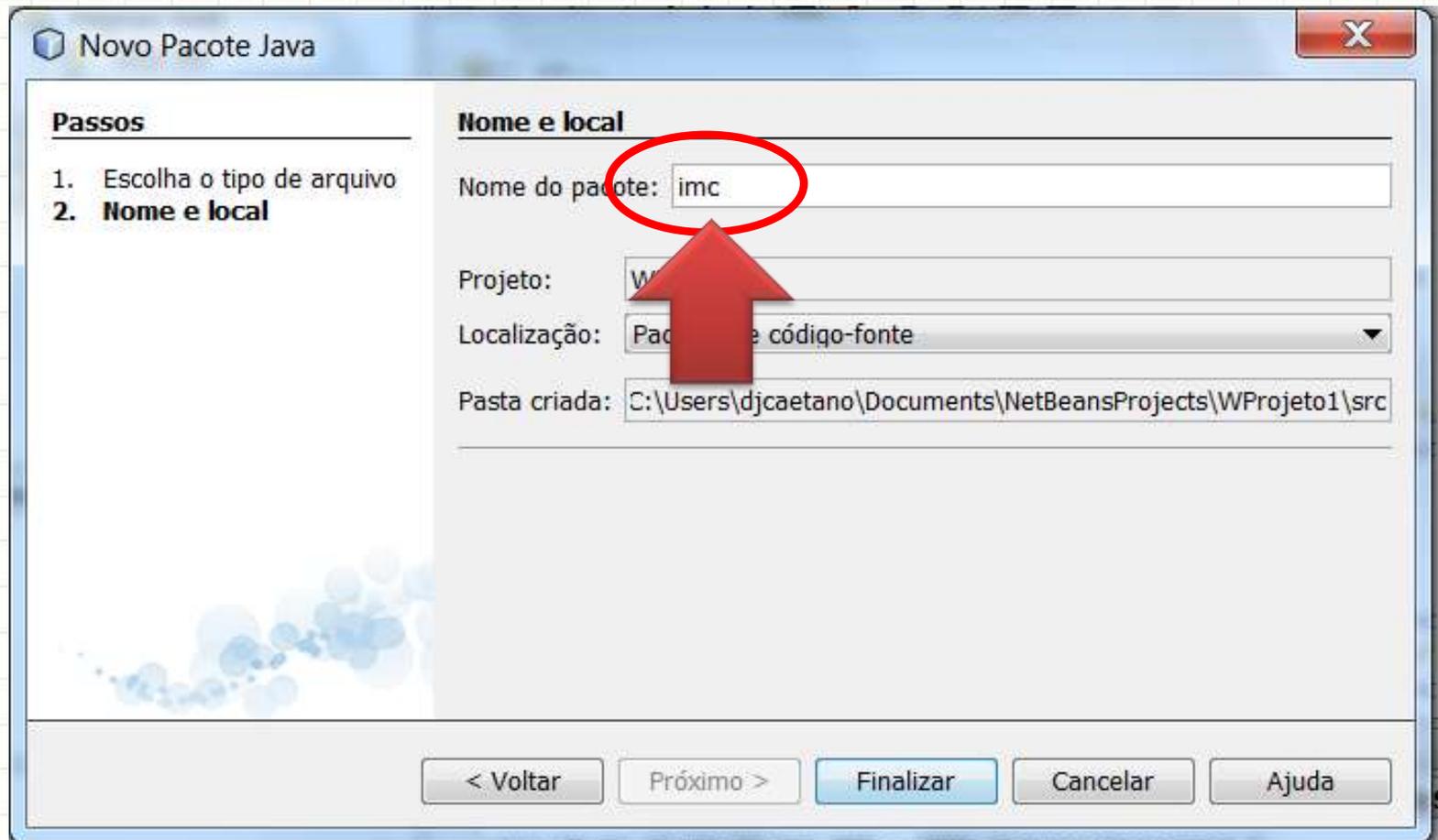
Criando um Servlet

- E selecione **Novo > Pacote Java...**



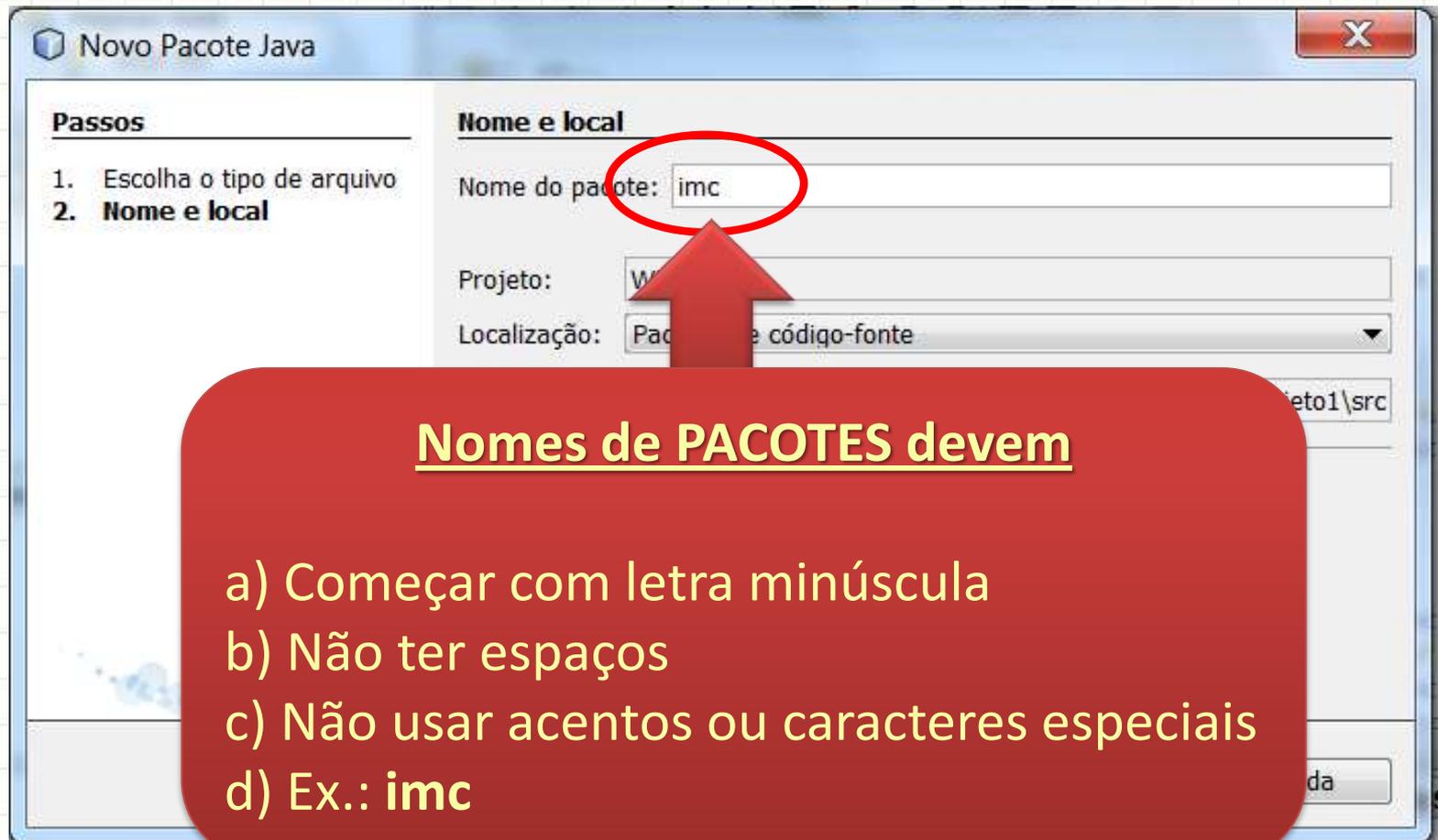
Criando um Servlet

- Agora dê um nome ao pacote: **imc**



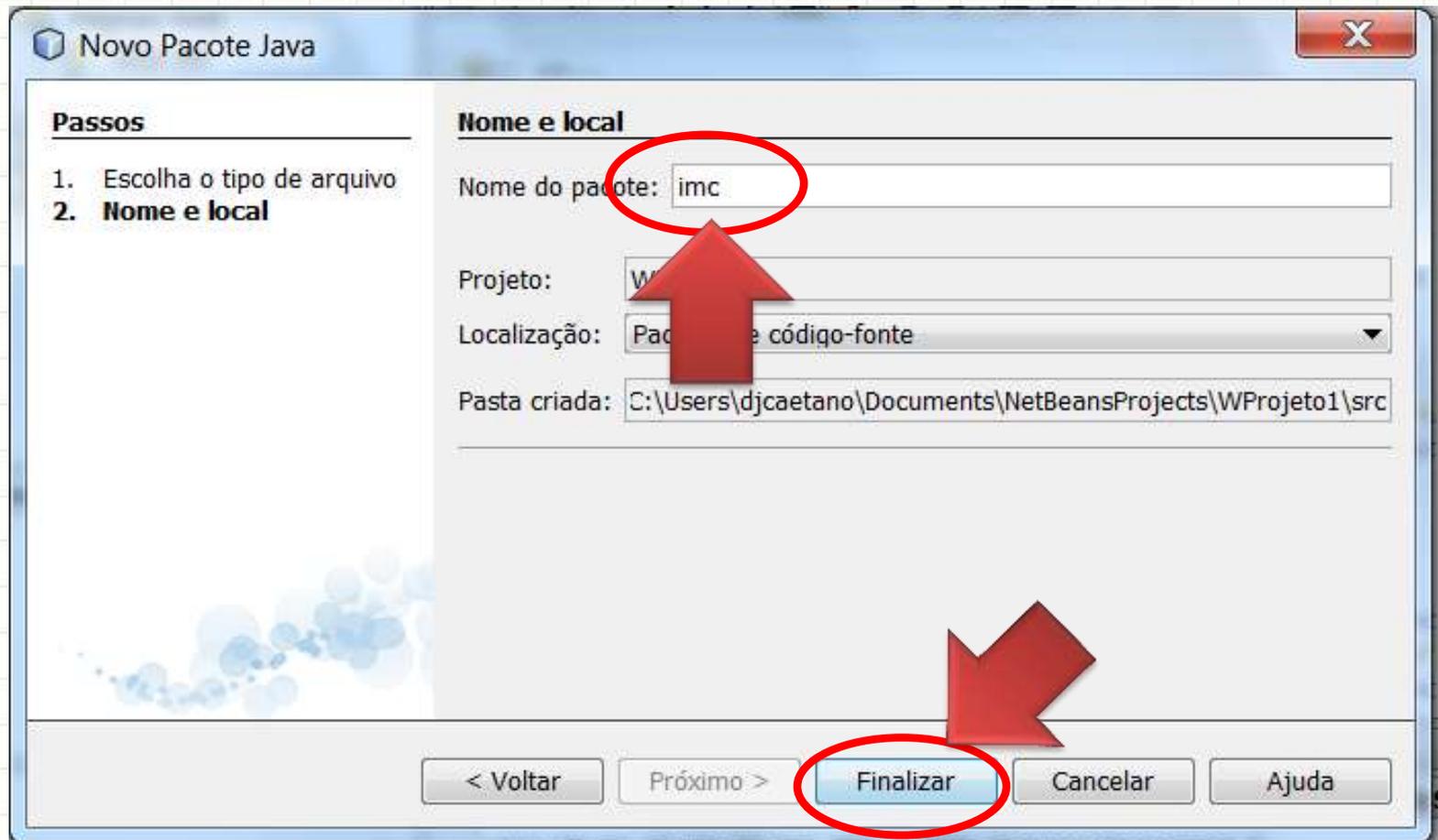
Criando um Servlet

- Agora dê um nome ao pacote: **imc**



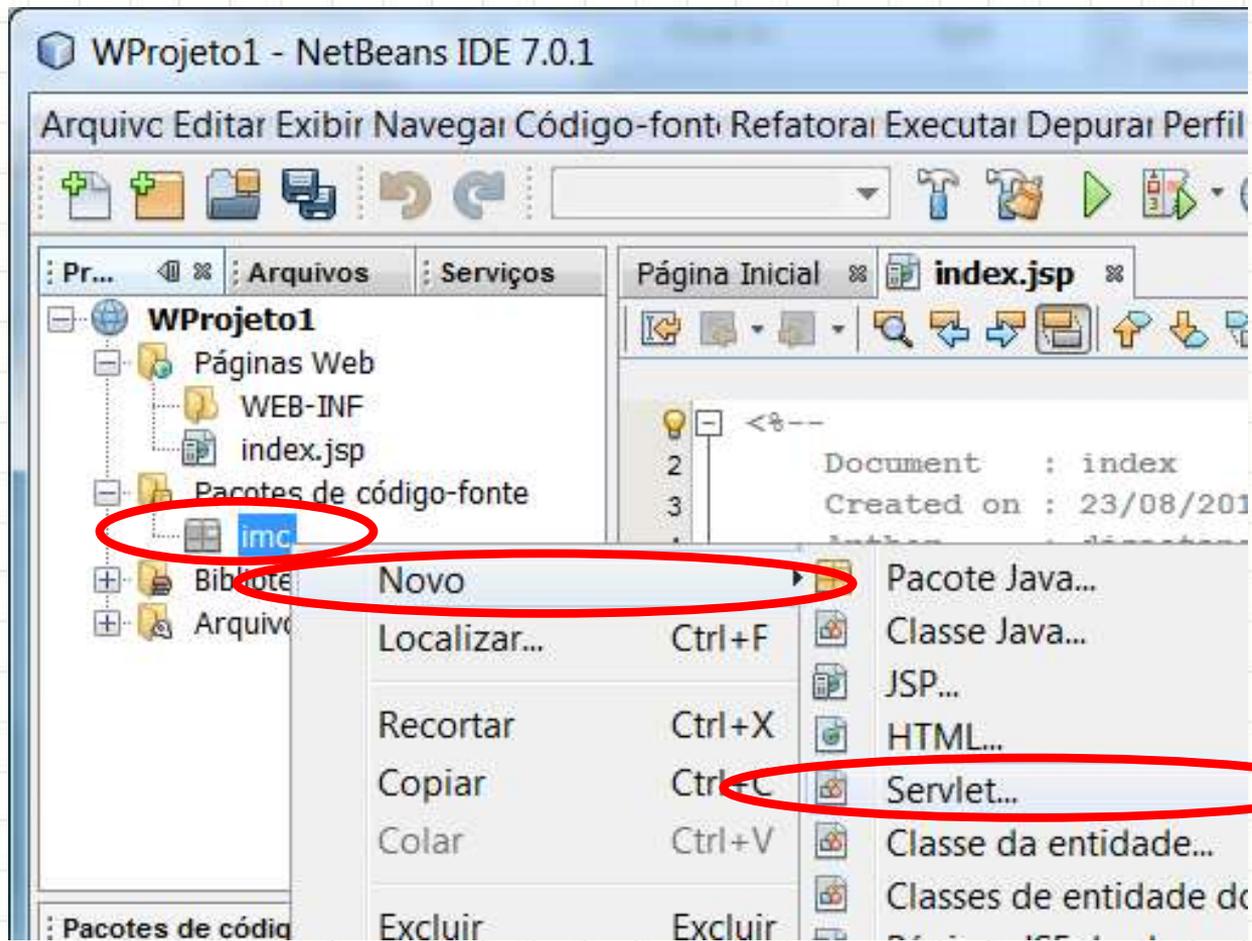
Criando um Servlet

- E clique em **Finalizar**



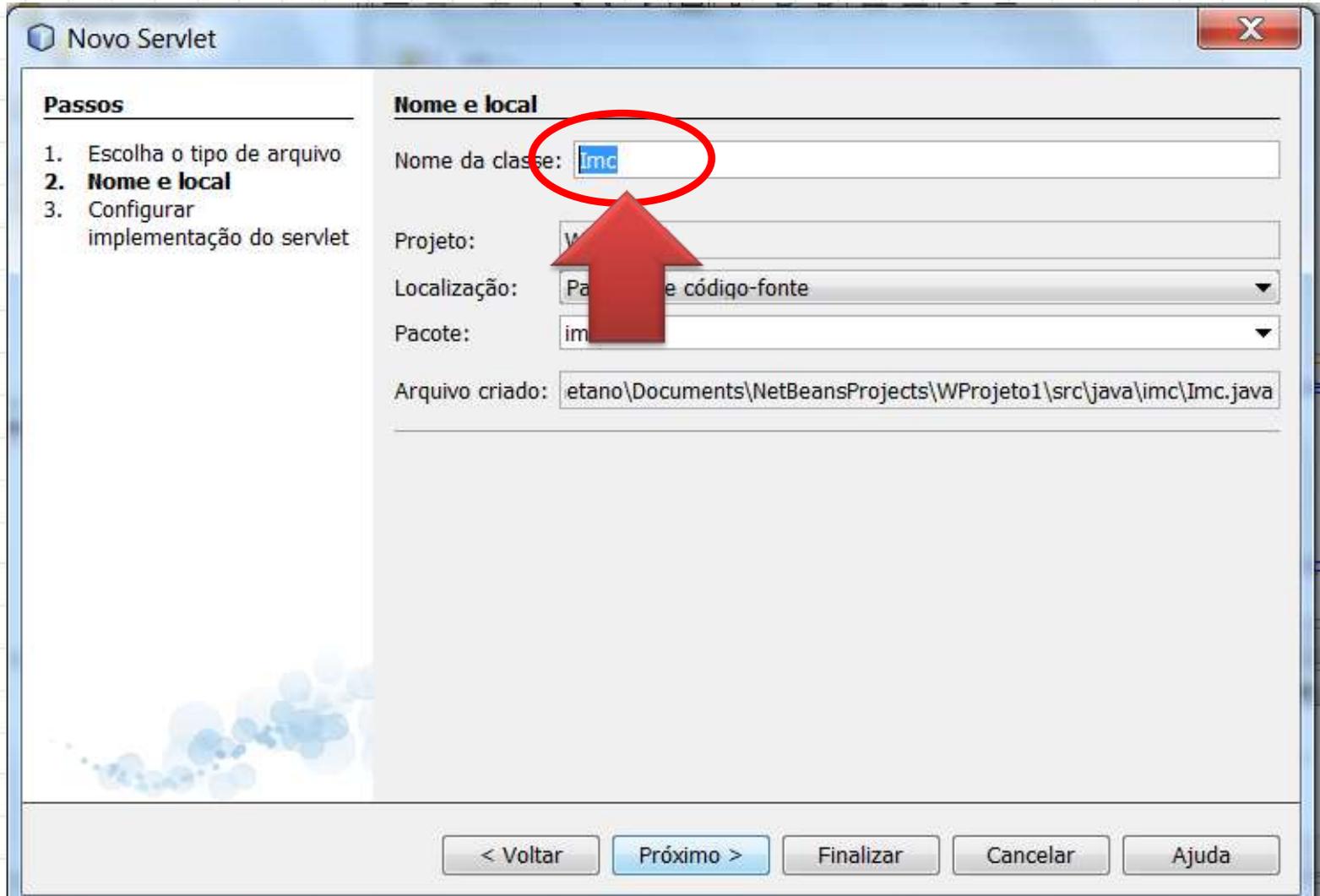
Criando um Servlet

- Agora clique com o botão direito no pacote **imc** e selecione **Novo > Servlet...**



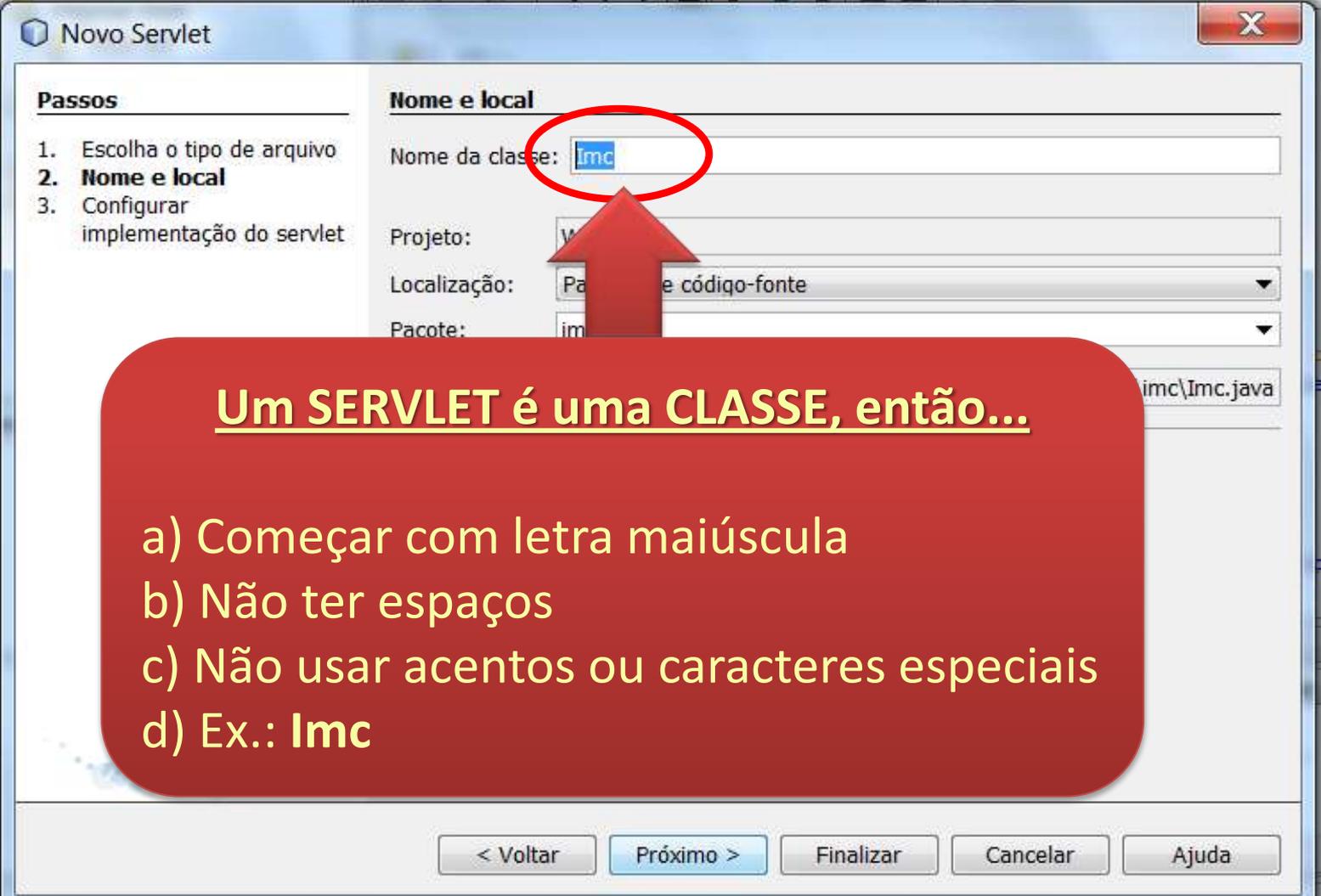
Criando um Servlet

- Agora dê um nome ao servlet: **Imc**



Criando um Servlet

- Agora dê um nome ao servlet: **Imc**

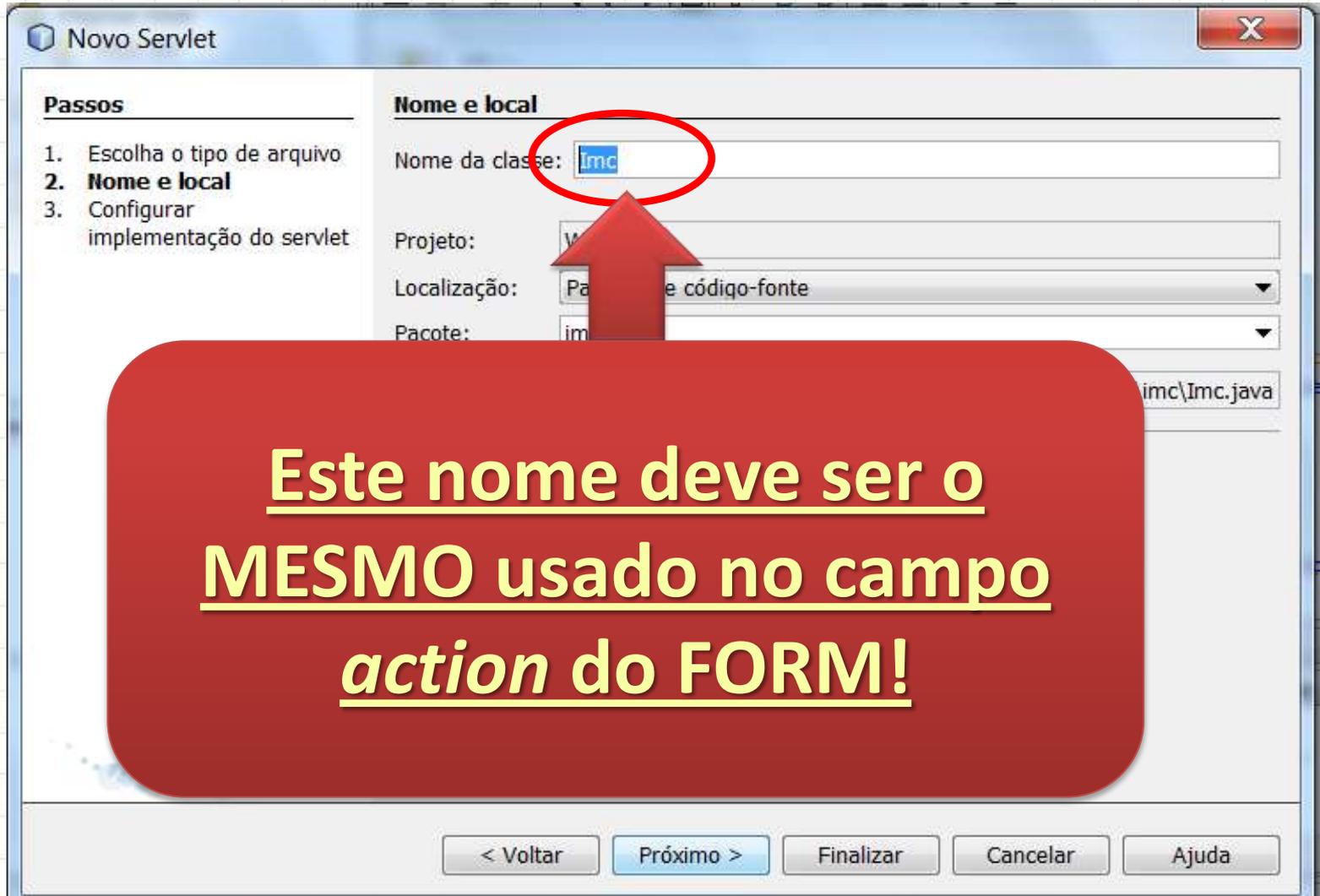


Um SERVLET é uma CLASSE, então...

- a) Começar com letra maiúscula
- b) Não ter espaços
- c) Não usar acentos ou caracteres especiais
- d) Ex.: **Imc**

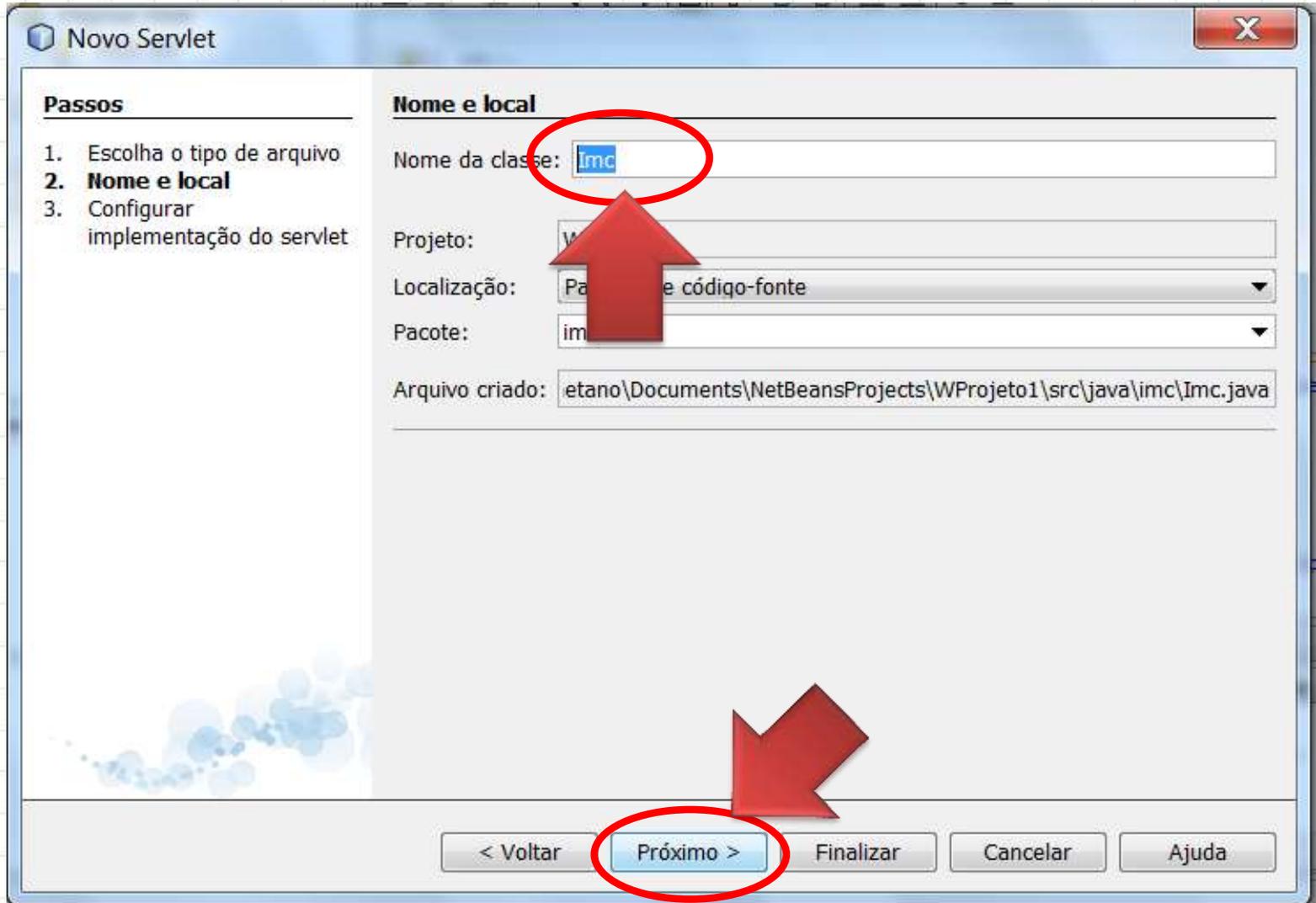
Criando um Servlet

- Agora dê um nome ao servlet: **Imc**



Criando um Servlet

- E clique em **Próximo**



Criando um Servlet

- **MARQUE** a opção **Adiciona informação ao descritor**

Novo Servlet

Passos

1. Escolha o tipo de arquivo
2. Nome e local
3. **Configurar implementação do servlet**

Configurar implementação do servlet

Registre o servlet com o aplicativo, dando ao servlet um nome interno (Nome do servlet). Depois especifique os padrões que identifiquem as URLs que invocam o servlet. Separar múltiplos padrões com vírgulas.

Adicionar informação ao descritor de implementação (web.xml)

Nome da classe:

Nome do servlet:

Padrão(ões) de URL:

Parâmetros de inicialização:

Nome	Valor
------	-------

Criando um Servlet

- **MARQUE** a opção **Adiciona informação ao descritor**

Novo Servlet

Passos

1. Escolha o tipo de arquivo
2. Nome e local
3. **Configurar implementação do servlet**

Configurar implementação do servlet

Registre o servlet com o aplicativo, dando ao servlet um nome interno (Nome do servlet). Depois especifique os padrões que identifiquem as URLs que invocam o servlet. Separar múltiplos padrões com vírgulas.

Adicionar informação ao descritor de implementação (web.xml)

Classe:

Nome:

URL:

Parâmetros de inicialização:

Nome	Valor
------	-------

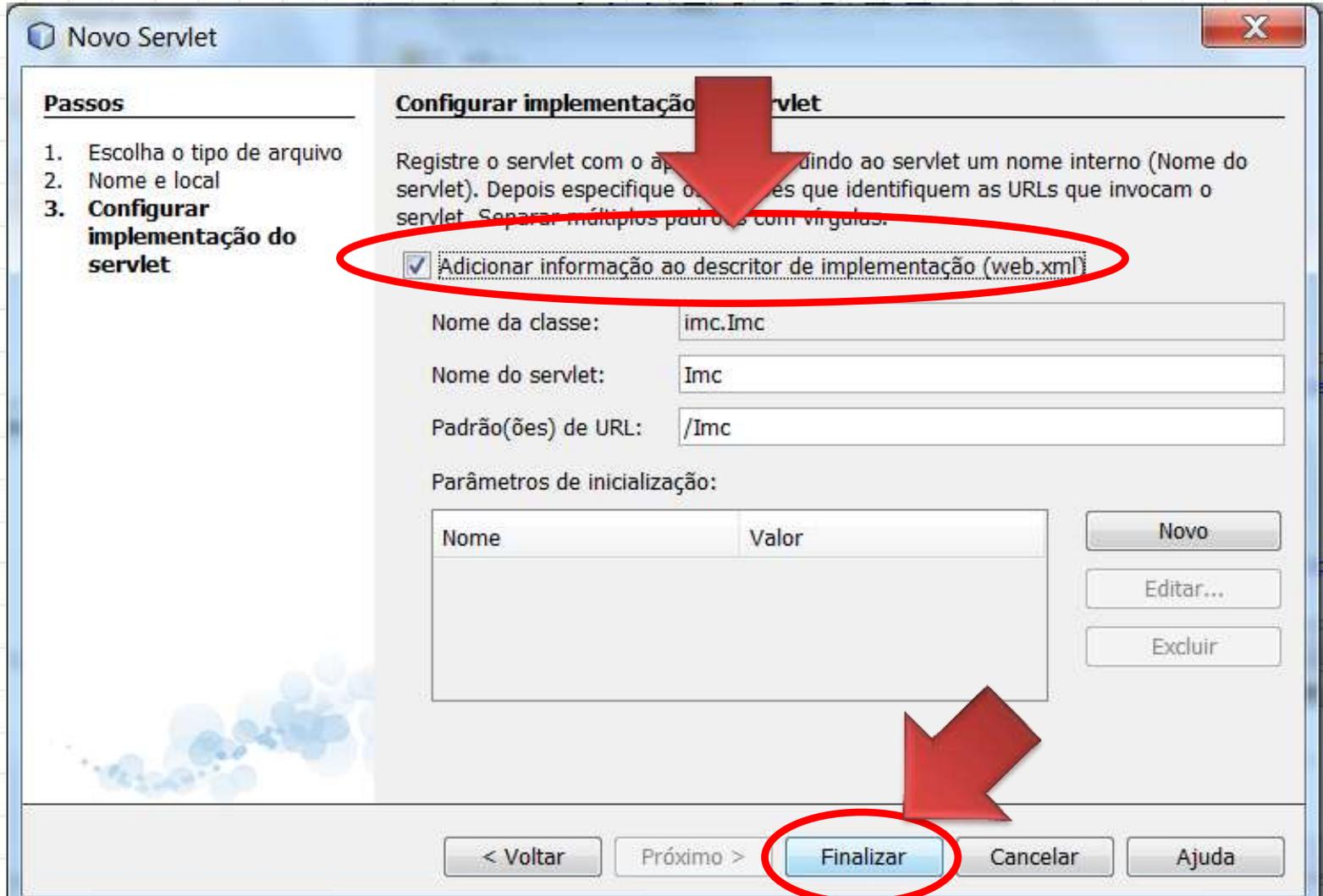
Novo
Editar...
Excluir

< Voltar Próximo > Finalizar Cancelar Ajuda

Nomes para
acesso ao Servlet

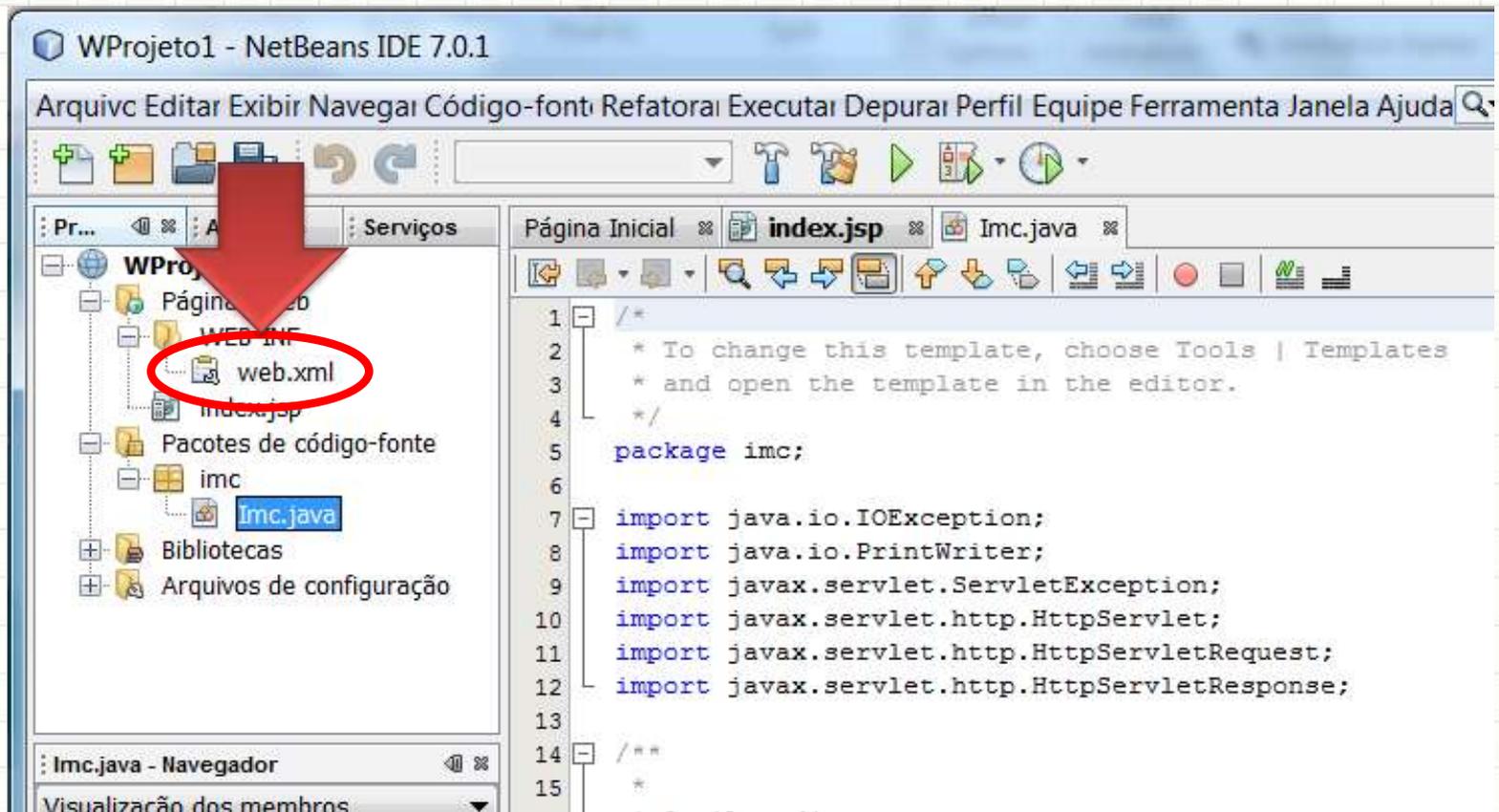
Criando um Servlet

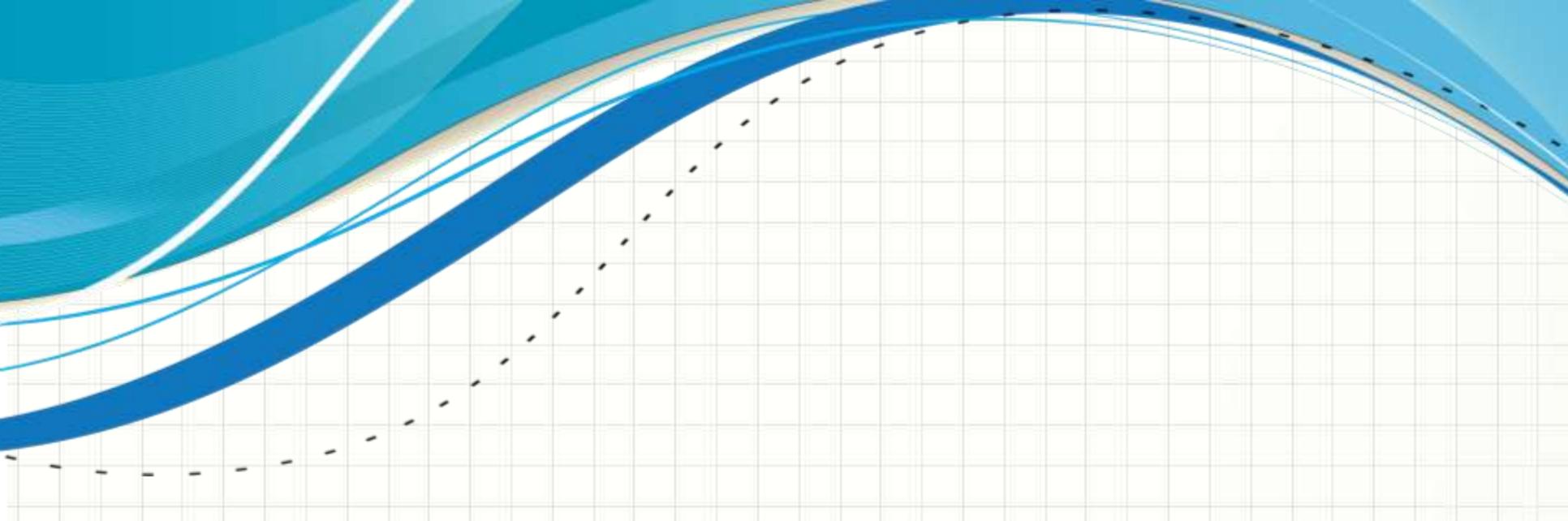
- E clique em **Finalizar**



Criando um Servlet

- Isso vai criar uma série de arquivos e abrir o servlet na área de edição... O primeiro é o de configuração:

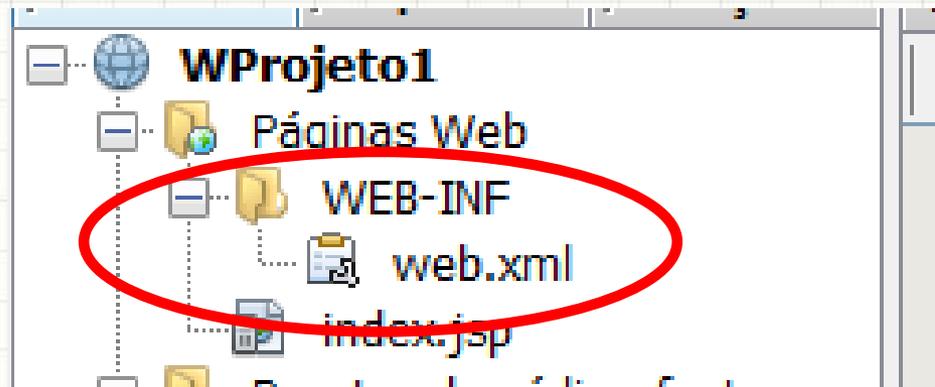




CONFIGURANDO O SERVLET

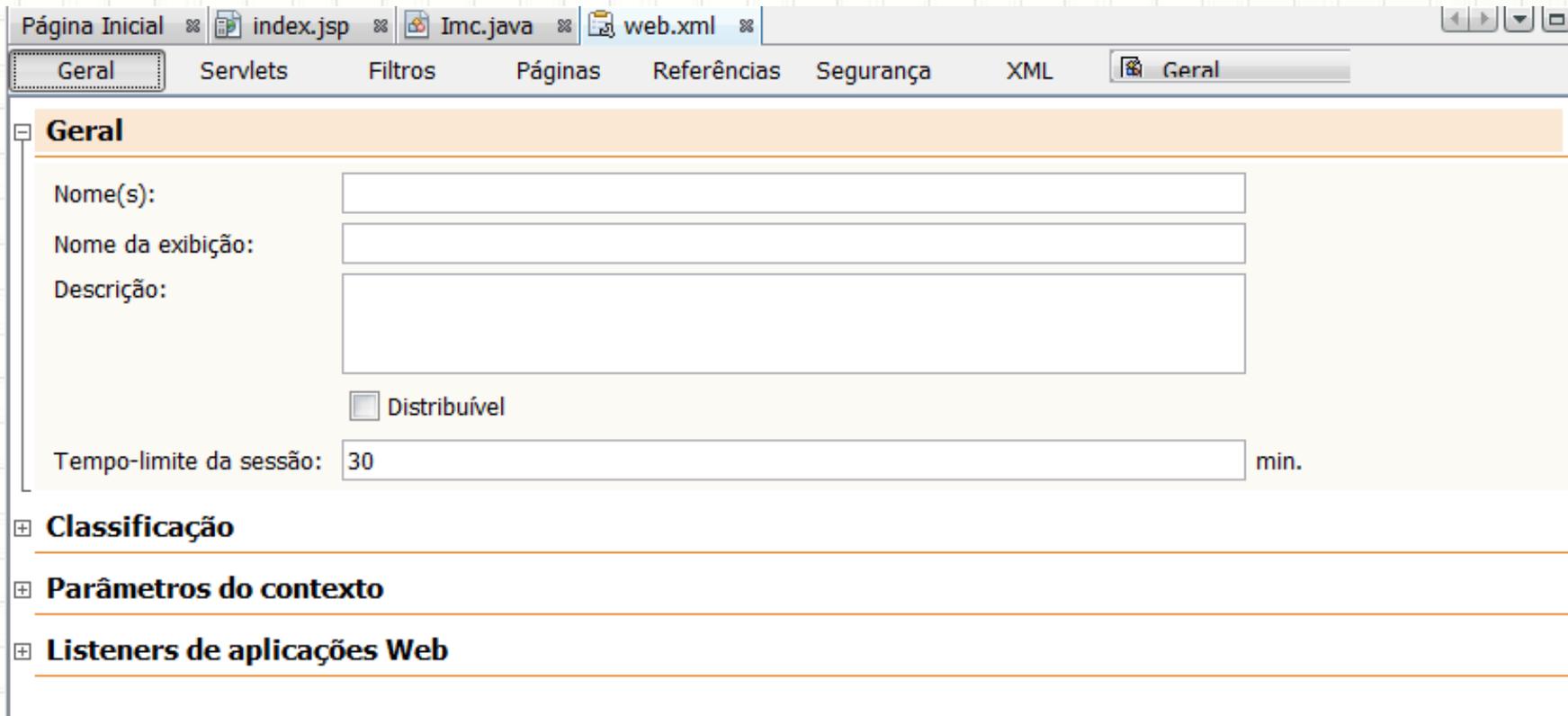
Configurações do Servlet

- O arquivo de configuração é chamado **web.xml...**
- Algumas vezes existem vários deles
- O que nos interessa está na pasta **web-inf**
- Clique duas vezes neste arquivo...



Configurações do Servlet

- A aba **Geral** serve para indicar a descrição do **SISTEMA**, além de indicar o **timeout** (veremos isso depois)



The image shows a screenshot of an IDE configuration window for a Servlet. The window has a title bar with tabs for 'Página Inicial', 'index.jsp', 'Imc.java', and 'web.xml'. Below the title bar, there are several tabs: 'Geral', 'Servlets', 'Filtros', 'Páginas', 'Referências', 'Segurança', 'XML', and 'Geral'. The 'Geral' tab is selected and active. The main content area is divided into sections:

- Geral**: This section contains several input fields and a checkbox:
 - Nome(s): [Empty text box]
 - Nome da exibição: [Empty text box]
 - Descrição: [Empty text box]
 - Distribuível
 - Tempo-limite da sessão: 30 min.
- Classificação**: A section with a plus sign icon and a horizontal line below it.
- Parâmetros do contexto**: A section with a plus sign icon and a horizontal line below it.
- Listeners de aplicações Web**: A section with a plus sign icon and a horizontal line below it.

Configurações do Servlet

- A aba **Servlets** configura itens importantes

The screenshot shows the 'Servlets' configuration window in an IDE. The 'Servlets' tab is selected, and the configuration for a servlet named 'Imc' is displayed. The configuration includes the following fields and options:

- Nome do servlet:** Imc
- Ordem de inicialização:**
- Descrição:** (empty text area)
- Classe do servlet:** imc.Imc [Ir para código-fonte](#)
- Arquivo JSP:** [Ir para o código-fonte](#)
- Padrão de URL:** /Imc
Utilize vírgula (,) para separar múltiplos padrões.
- Parâmetros de inicialização:**

Nome do parâmetro	Valor do parâmetro	Descrição
<input type="button" value="Adicionar..."/>	<input type="button" value="Editar..."/>	<input type="button" value="Remover"/>
- Referência de função de segurança:**

Nome Ref do papel	Vínculo Ref do papel	Descrição
<input type="button" value="Adicionar..."/>	<input type="button" value="Editar..."/>	<input type="button" value="Remover"/>
- Executar como:**

Configurações do Servlet

- A ordem de inicialização

The screenshot shows the 'Servlets' configuration window in an IDE. The 'Imc' servlet is selected, and its configuration is displayed. The 'Ordem de inicialização' (Startup Order) field is highlighted with a red circle, indicating its importance in the context of the slide's topic.

General: Geral, Servlets, Filtros, Páginas, Referências, Segurança, XML, Imc

Servlets: Adicionar elemento servlet...

Imc -> /Imc: Remove

Nome do servlet: Imc **Ordem de inicialização:**

Descrição:

Classe do servlet: imc.Imc Procurar... [Ir para código-fonte](#)

Arquivo JSP: Procurar... [Ir para o código-fonte](#)

Padrão de URL: /Imc
Utilize vírgula (,) para separar múltiplos padrões.

Parâmetros de inicialização:

Nome do parâmetro	Valor do parâmetro	Descrição
-------------------	--------------------	-----------

Adicionar... Editar... Remover

Referência de função de segurança:

Nome Ref do papel	Vínculo Ref do papel	Descrição
-------------------	----------------------	-----------

Adicionar... Editar... Remover

Executar como:

Configurações do Servlet

- O nome do Servlet

The screenshot shows the configuration window for a servlet named 'Imc'. The 'Nome do servlet' field is highlighted with a red circle and contains the text 'Imc'. Other fields include 'Descrição', 'Classe do servlet' (set to 'imc.Imc'), 'Arquivo JSP', and 'Padrão de URL' (set to '/Imc'). There are also sections for initialization parameters and security function references.

Nome do servlet: **Imc** Ordem de inicialização:

Descrição:

Classe do servlet: Procurar... [Ir para código-fonte](#)

Arquivo JSP: Procurar... [Ir para o código-fonte](#)

Padrão de URL:
Utilize vírgula (,) para separar múltiplos padrões.

Parâmetros de inicialização:

Nome do parâmetro	Valor do parâmetro	Descrição
-------------------	--------------------	-----------

Adicionar... Editar... Remover

Referência de função de segurança:

Nome Ref do papel	Vínculo Ref do papel	Descrição
-------------------	----------------------	-----------

Adicionar... Editar... Remover

Executar como:

Configurações do Servlet

- A classe do Servlet (pode ser trocada!)

Servlets

Adicionar elemento servlet...

Imc -> /Imc

Remove

Nome do servlet: Imc

Ordem de inicialização:

Descrição:

Classe do servlet: imc.Imc [Ir para código-fonte](#)

Arquivo JSP: [Ir para o código-fonte](#)

Padrão de URL: /Imc

Utilize vírgula (,) para separar múltiplos padrões.

Parâmetros de inicialização:

Nome do parâmetro	Valor do parâmetro	Descrição
-------------------	--------------------	-----------

Referência de função de segurança:

Nome Ref do papel	Vínculo Ref do papel	Descrição
-------------------	----------------------	-----------

Executar como:

Configurações do Servlet

- E o nome de acesso do Servlet

The screenshot shows the configuration window for a servlet named 'Imc'. The 'Padrão de URL' field is highlighted with a red circle and contains the value '/Imc'. Below this field, there is a note: 'Utilize vírgula (,) para separar múltiplos padrões.' The window also includes sections for initialization parameters and security function references, each with a table structure and buttons for adding, editing, and removing items.

Nome do servlet: Ordem de inicialização:

Descrição:

Classe do servlet: [Ir para código-fonte](#)

Arquivo JSP: [Ir para o código-fonte](#)

Padrão de URL: Utilize vírgula (,) para separar múltiplos padrões.

Parâmetros de inicialização:

Nome do parâmetro	Valor do parâmetro	Descrição
-------------------	--------------------	-----------

Referência de função de segurança:

Nome Ref do papel	Vínculo Ref do papel	Descrição
-------------------	----------------------	-----------

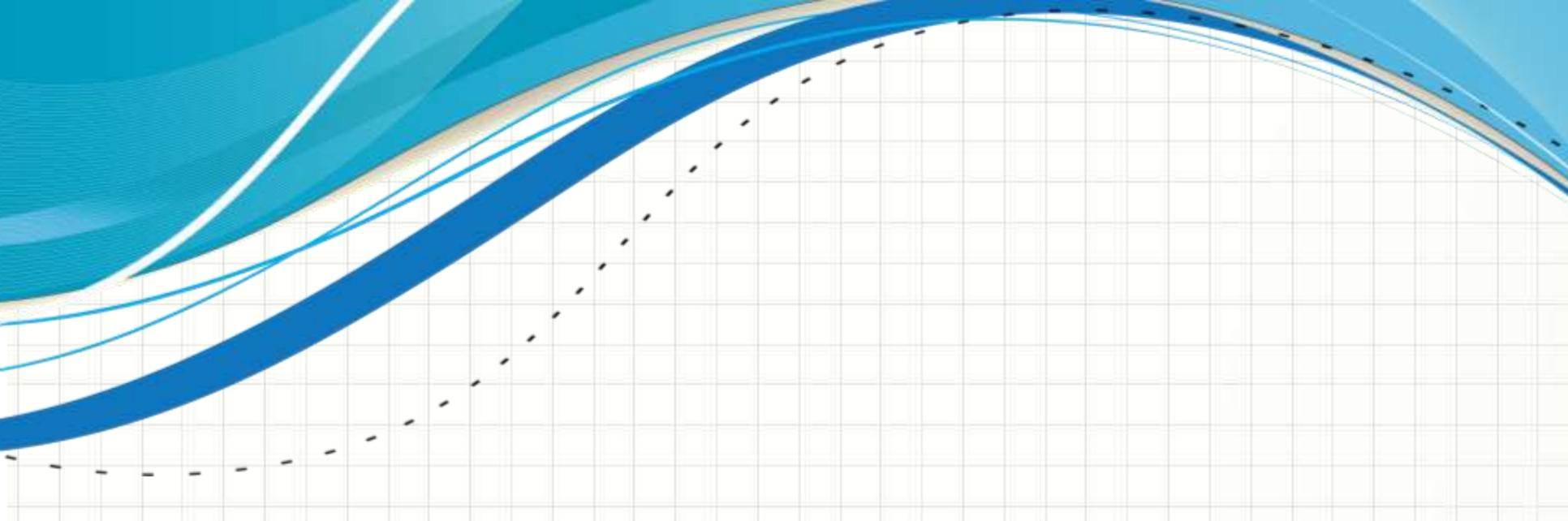
Executar como:

Configurações do Servlet

- Essas e outras configurações podem ser feitas diretamente pela aba XML...
- **Mas pelo NetBeans é mais conveniente!**



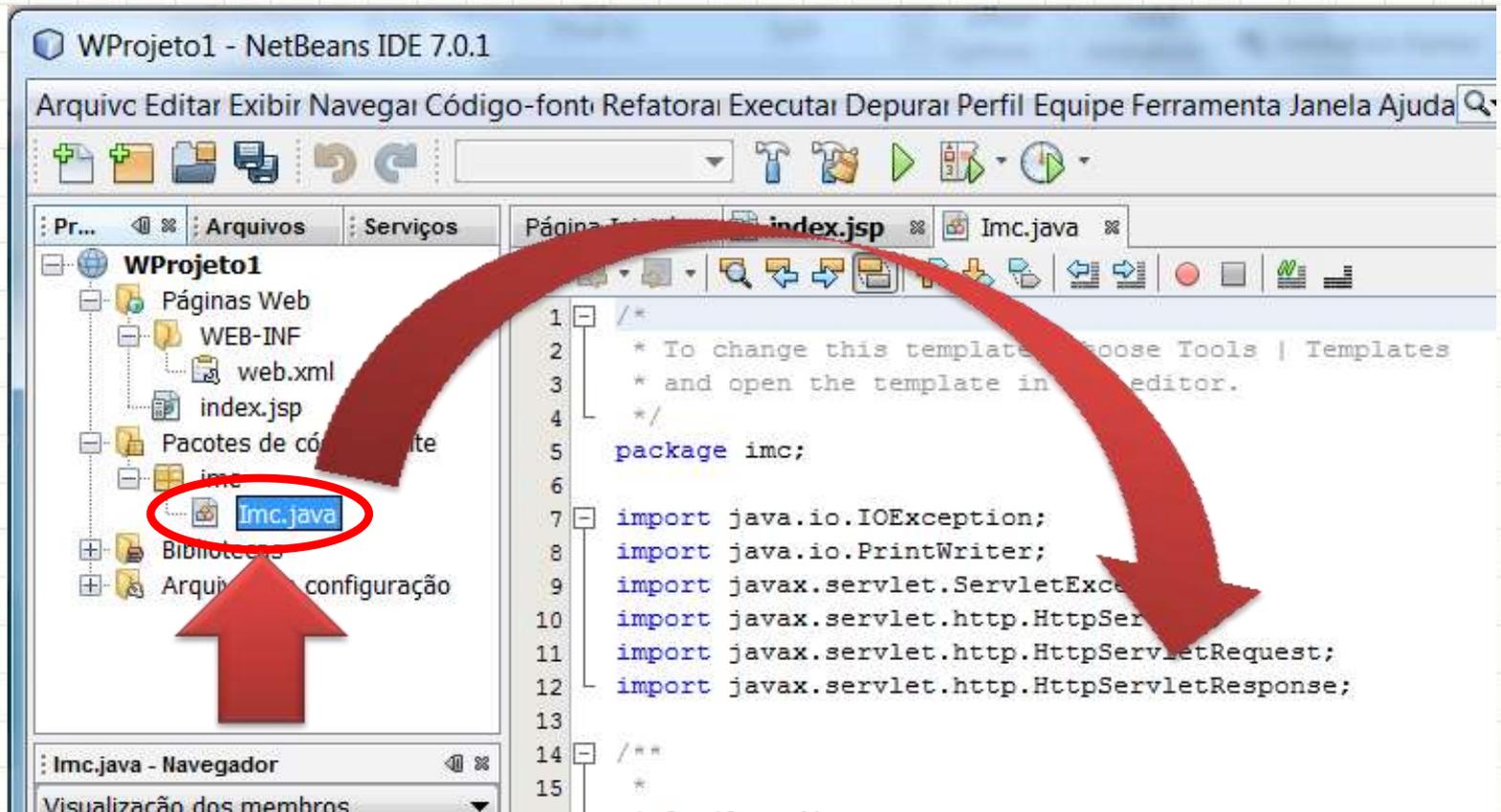
```
Página Inicial  index.jsp  Imc.java  web.xml
Geral  Servlets  Filtros  Páginas  Referências  Segurança  XML
1  <?xml version="1.0" encoding="UTF-8"?>
2  <web-app version="3.0" xmlns="http://java.sun.com/xml/ns/javaee" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3  <
4      <servlet>
5          <servlet-name>Imc</servlet-name>
6          <servlet-class>imc.Imc</servlet-class>
7      </servlet>
8      <servlet-mapping>
9          <servlet-name>Imc</servlet-name>
10         <url-pattern>/Imc</url-pattern>
11     </servlet-mapping>
12     <session-config>
13         <session-timeout>
14             30
15         </session-timeout>
16     </session-config>
17 </web-app>
```



VOLTANDO A EDITAR O SERVLET

Criando um Servlet

- O outro arquivo importante é o arquivo do Servlet, neste caso o **Imc.java**.



Limpando a área

- Vamos apagar os comentários do NetBeans

```
[-] /*
    * To change this template, choose Tools | Templates
    * and open the template in the editor.
    */
package imc;

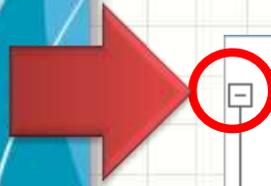
[-] import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

[-] /**
    *
    * @author djcaetano
    */
public class Imc extends HttpServlet {

[-] /**
    * Processes requests for both HTTP <code>GET</code> and <code>POST</code> method
    * @param request servlet request
    * @param response servlet response
    * @throws ServletException if a servlet-specific error occurs
    * @throws IOException if an I/O error occurs
    */
    protected void processRequest(HttpServletRequest request, HttpServletResponse res
```

Limpando a área

- Vamos esconder código “desnecessário”



```
package imc;
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class Imc extends HttpServlet {

    protected void processRequest(HttpServletRequest request, HttpServletResponse res
        throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        try {
            /* TODO output your page here
            out.println("<html>");
            out.println("<head>");
            out.println("<title>Servlet Imc</title>");
            out.println("</head>");
            out.println("<body>");
            out.println("<h1>Servlet Imc at " + request.getContextPath () + "</h1>");
            out.println("</body>");
            out.println("</html>");
            */
        } finally {
            out.close();
        }
    }
}
```

Entendendo o Servlet

- Observe que o Servlet **extends HttpServlet**

```
package imc;
import ...

public class Imc extends HttpServlet {

    protected void processRequest(HttpServletRequest request, HttpServletResponse res
        throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        try {
            /* TODO output your page here
            out.println("<html>");
            out.println("<head>");
            out.println("<title>Servlet Imc</title>");
            out.println("</head>");
            out.println("<body>");
            out.println("<h1>Servlet Imc at " + request.getContextPath () + "</h1>");
            out.println("</body>");
            out.println("</html>");
            */
        } finally {
            out.close();
        }
    }
}
```

HttpServlet methods. Click on the + sign on the left to edit the code.

Entend

- Vamos e

Este é o trecho que
mais nos interessa!

```
package imc;
import ...

public class Imc extends HttpServlet {

    protected void processRequest(HttpServletRequest request, HttpServletResponse res
        throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        try {
            /* TODO output your page here
            out.println("<html>");
            out.println("<head>");
            out.println("<title>Servlet Imc</title>");
            out.println("</head>");
            out.println("<body>");
            out.println("<h1>Servlet Imc at " + request.getContextPath () + "</h1>");
            out.println("</body>");
            out.println("</html>");
            */
        } finally {
            out.close();
        }
    }
}
```

HttpServlet methods. Click on the + sign on the left to edit the code.

Entendendo o Servlet

- Vamos entender a estrutura

```
package imc;
+ import ...

public class Imc extends HttpServlet {

    protected void processRequest(HttpServletRequest request, HttpServletResponse res
        throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        try {
            /* TODO output your page here
            out.println("<html>");
            out.println("<head>");

            + "</h1>");

        } finally {
            out.close();
        }
    }
}
```



+ HttpServlet methods. Click on the + sign on the left to edit the code.

Entendendo o Servlet

- Vamos entender o funcionamento

```
protected void processRequest(HttpServletRequest request, HttpServletResponse res  
    throws ServletException, IOException {  
    response.setContentType("text/html;charset=UTF-8");  
    PrintWriter out = response.getWriter();  
    try {  
        /* TODO output your page here
```

Quando uma **request** chega ao contentor, ela é repassada para esse método, juntamente com uma **reponse** para armazenarmos a saída

```
}
```

```
}
```

Entendendo o Servlet

- Vamos entender o funcionamento

```
protected void processRequest(HttpServletRequest request, HttpServletResponse res
    throws ServletException, IOException {
    response.setContentType("text/html;charset=UTF-8");
    PrintWriter out = response.getWriter();
    try {
        /* TODO output your here
        out.println("<html>
        out.println("<head>
```

A primeira coisa feita é a configuração do objeto **out**, que usaremos para imprimir a saída

```
() + "</h1>");
```

Programando o Servlet

- Aqui é onde escreveremos o nosso código
- **Temos de imprimir um HTML!**

```
protected void processRequest(HttpServletRequest request, HttpServletResponse res
    throws ServletException, IOException {
    response.setContentType("text/html;charset=UTF-8");
    PrintWriter out = response.getWriter();
    try {
        /* TODO output your page here
        out.println("<html>");
        out.println("<head>");
        out.println("<title>Servlet Imc</title>");
        out.println("</head>");
        out.println("<body>");
        out.println("<h1>Servlet Imc at " + request.getContextPath () + "</h1>");
        out.println("</body>");
        out.println("</html>");
        */
    } finally {
        out.close();
    }
}
```

Programando o Servlet

- Acompanhe o professor!

```
protected void processRequest(HttpServletRequest request, HttpServletResponse res
    throws ServletException, IOException {
    response.setContentType("text/html;charset=UTF-8");
    PrintWriter out = response.getWriter();
    try {
        /* TODO output your page here
        out.println("<html>");
        out.println("<head>");
        out.println("<title>Servlet Imc</title>");
        out.println("</head>");
        out.println("<body>");
        out.println("<h1>Servlet Imc at " + request.getContextPath () + "</h1>");
        out.println("</body>");
        out.println("</html>");
        */
    } finally {
        out.close();
    }
}
```

Programando o Servlet

- Execute e veja o que acontece!

```
protected void processRequest(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    response.setContentType("text/html;charset=UTF-8");
    PrintWriter out = response.getWriter();
    try {
        out.println("<html>");
        out.println("<head>");
        out.println("<title>Calculadora de Imc: Resultados</title>");
        out.println("</head>");
        out.println("<body>");
        out.println("<h1>Índice de Massa Corporal: </h1>");
        out.println("</body>");
        out.println("</html>");
    } finally {
        out.close();
    }
}
```

Programando o Servlet

- Falta fazer o cálculo:
 - $IMC = PESO / (ALTURA * ALTURA)$
- Mas antes... É preciso pegar os parâmetros na **request**...
- Pegar... Parâmetro... Da request...
 - `request.getParameter("nome")`
- Os nomes são “peso” e “altura” (do form!)
- **NOTA:** as informações da **request** estão sempre no formato **String**!

Programando o Servlet

- Execute e veja o que acontece!

```
protected void processRequest(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    response.setContentType("text/html;charset=UTF-8");
    PrintWriter out = response.getWriter();
    try {
        out.println("<html>");
        out.println("<head>");
        out.println("<title>Calculadora de Imc: Resultados</title>");
        out.println("</head>");
        out.println("<body>");
        out.println("<h1>Índice de Massa Corporal: </h1>");

        String pesoT = request.getParameter("peso");
        String alturaT = request.getParameter("altura");

        out.println("Peso: " + pesoT);
        out.println("Altura: " + alturaT);

        out.println("</body>");
        out.println("</html>");
    } finally {
        out.close();
    }
}
```

Programando o Servlet

- Exec

```
protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
    try {  
        out.println("Pesos: ");  
        out.println("Alturas: ");  
        String pesoT = request.getParameter("peso");  
        String alturaT = request.getParameter("altura");  
        out.println("Peso: " + pesoT);  
        out.println("Altura: " + alturaT);  
        out.println("</body>");  
        out.println("</html>");  
    } finally {  
        out.close();  
    }  
}
```

OPA! Não pulou linha!

- a) Estamos imprimindo HTML
- b) Onde está o <p> e o </p> ???

```
String pesoT = request.getParameter("peso");  
String alturaT = request.getParameter("altura");  
  
out.println("Peso: " + pesoT);  
out.println("Altura: " + alturaT);
```

```
out.println("</body>");  
out.println("</html>");  
} finally {  
    out.close();  
}
```

Programando o Servlet

- Execute e veja o que acontece!

```
protected void processRequest(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    response.setContentType("text/html;charset=UTF-8");
    PrintWriter out = response.getWriter();
    try {
        out.println("<html>");
        out.println("<head>");
        out.println("<title>Calculadora de Imc: Resultados</title>");
        out.println("</head>");
        out.println("<body>");
        out.println("<h1>Índice de Massa Corporal: </h1>");

        String pesoT = request.getParameter("peso");
        String alturaT = request.getParameter("altura");

        out.println("<p>Peso: " + pesoT + "</p>");
        out.println("<p>Altura: " + alturaT + "</p>");

        out.println("</body>");
        out.println("</html>");
    } finally {
        out.close();
    }
}
```

Programando o Servlet

- Antes de fazer os cálculos, porém...
- Temos que converter estas **Strings** para números com vírgula do tipo **double**
- Existe um método pronto para isso, na classe **Double**... Ele se chama **valueOf**
- **NOTA:** Use PONTO para indicar peso e altura
- Certo: 1.70 Errado: 1,70

Programando o Servlet

- Execute e veja o que acontece!

```
protected void processRequest(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    response.setContentType("text/html;charset=UTF-8");
    PrintWriter out = response.getWriter();
    try {
        out.println("<html>");
        out.println("<head>");
        out.println("<title>Calculadora de Imc: Resultados</title>");
        out.println("</head>");
        out.println("<body>");
        out.println("<h1>Índice de Massa Corporal: </h1>");

        String pesoT = request.getParameter("peso");
        String alturaT = request.getParameter("altura");
        double peso = Double.valueOf(pesoT);
        double altura = Double.valueOf(alturaT);

        double imc = peso / (altura*altura);
        out.println("<p>IMC: " + imc + "</p>");

        out.println("</body>");
        out.println("</html>");
    } finally {
        out.close();
    }
}
```

Programando o Servlet

- Podemos melhorar um pouco mais...

```
protected void processRequest(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    response.setContentType("text/html;charset=UTF-8");
    PrintWriter out = response.getWriter();
    try {
        out.println("<html>");
        out.println("<head>");
        out.println("<title>Calculadora de Imc: Resultados</title>");
        out.println("</head>");
        out.println("<body>");
        out.println("<h1>Índice de Massa Corporal: </h1>");

        String pesoT = request.getParameter("peso");
        String alturaT = request.getParameter("altura");
        double peso = Double.valueOf(pesoT);
        double altura = Double.valueOf(alturaT);

        double imc = peso / (altura*altura);
        out.printf("<p>IMC: %3.1f</p>", imc);

        out.println("</body>");
        out.println("</html>");
    } finally {
        out.close();
    }
}
```

Programando o Servlet

- O programa tem problemas com “,” nos números? Experimente!
- Corrija substituindo:

```
String pesoT = request.getParameter("peso");  
String alturaT = request.getParameter("altura");  
double peso = Double.valueOf(pesoT);  
double altura = Double.valueOf(alturaT);
```

- Por isso:

```
String pesoT = request.getParameter("peso");  
pesoT = pesoT.replaceAll(",", ".", "");  
String alturaT = request.getParameter("altura");  
alturaT = alturaT.replaceAll(",", ".", "");  
double peso = Double.valueOf(pesoT);  
double altura = Double.valueOf(alturaT);
```

Programando o Servlet

- Para imprimir mensagens de acordo com o resultado, experimente acrescentar o código abaixo!

```
out.printf("<p>IMC: %3.1f</p>", imc);
```

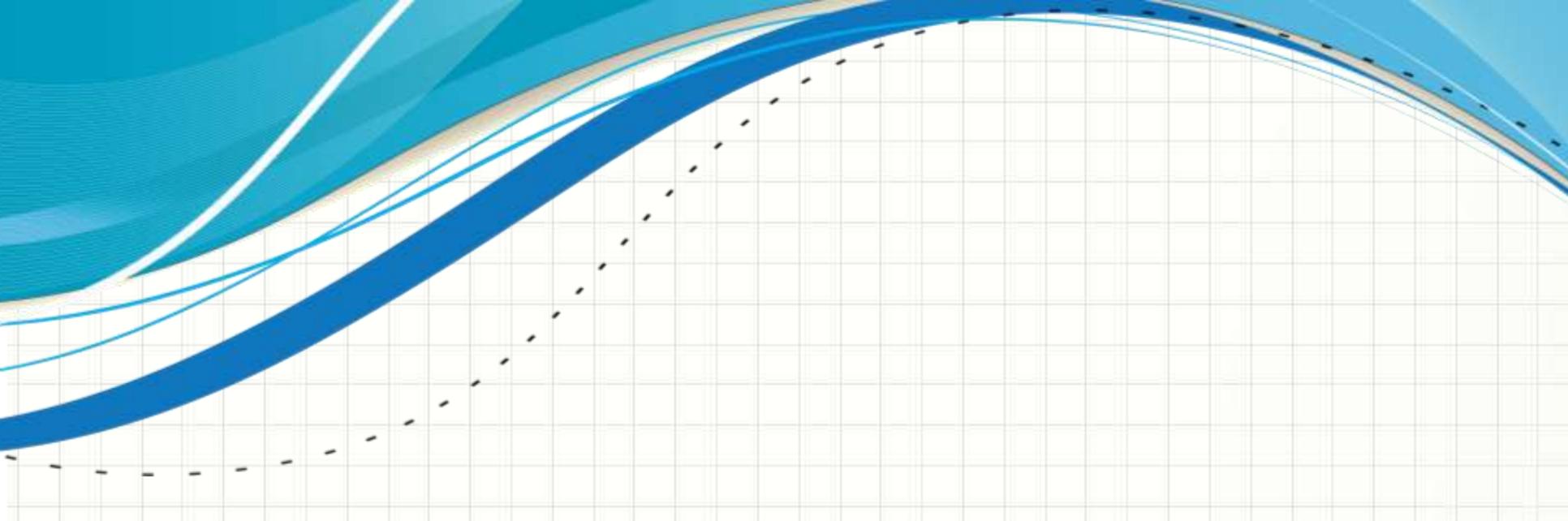
```
if (imc < 18.0) {  
    out.println("<p>Cuidado! Abaixo!</p>");  
} else if (imc < 25.0) {  
    out.println("<p>Parabéns! Peso ideal!</p>");  
} else {  
    out.println("<p>Cuidado! Acima!</p>");  
}
```

```
out.println("</body>");
```

Programando o Servlet

- O servlet ainda “capota” quando um “texto” é digitado nos campos numéricos! Experimente!
- Corrija isso acrescentando o “catch” abaixo!

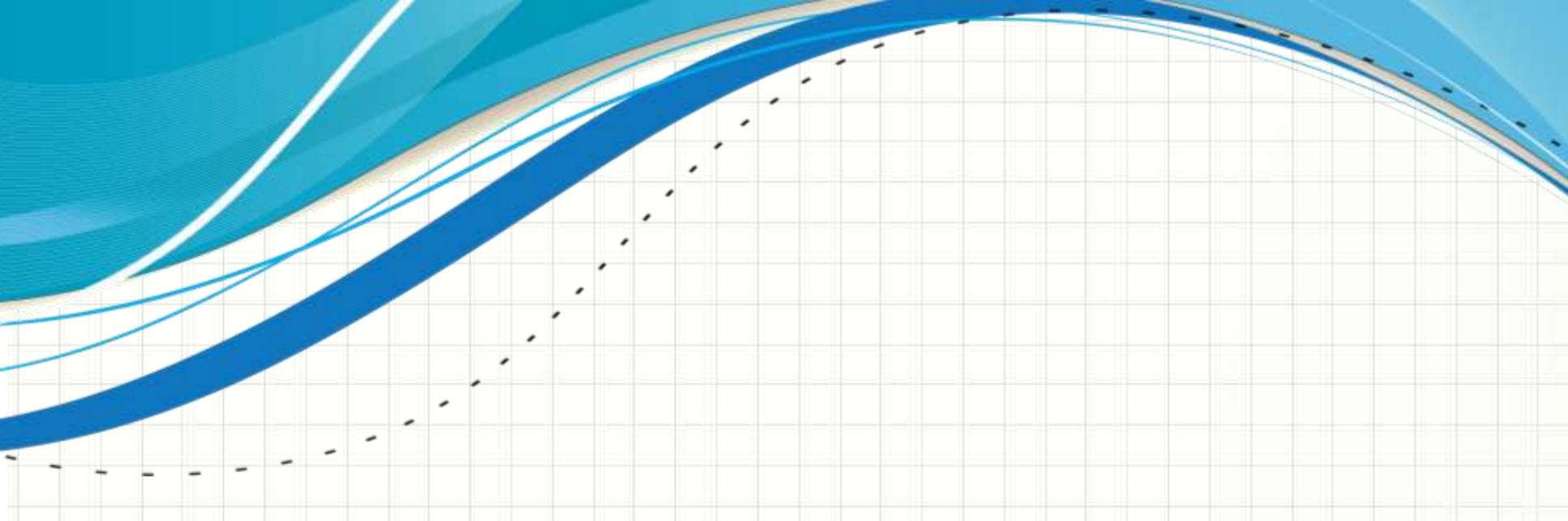
```
        out.println("</body>");
        out.println("</html>");
    } catch (Exception e) {
        out.println("<html>");
        out.println("<head>");
        out.println("<title>Resutado do IMC</title>");
        out.println("</head>");
        out.println("<body>");
        out.println("<p>Digite um número, mané!</p>");
        out.println("</body>");
        out.println("</html>");
    } finally {
        out.close();
    }
```



ATIVIDADE ESTRUTURADA

Orientação Atividades Estruturadas

- Esta disciplina possui Atividades Estruturadas
- Elas serão disponibilizadas no momento oportuno
- A primeira consiste em uma pesquisa (leitura e redação)
- A segunda consiste em compreender e modificar um sistema funcional
- Aguardem maiores informações!



CONCLUSÕES

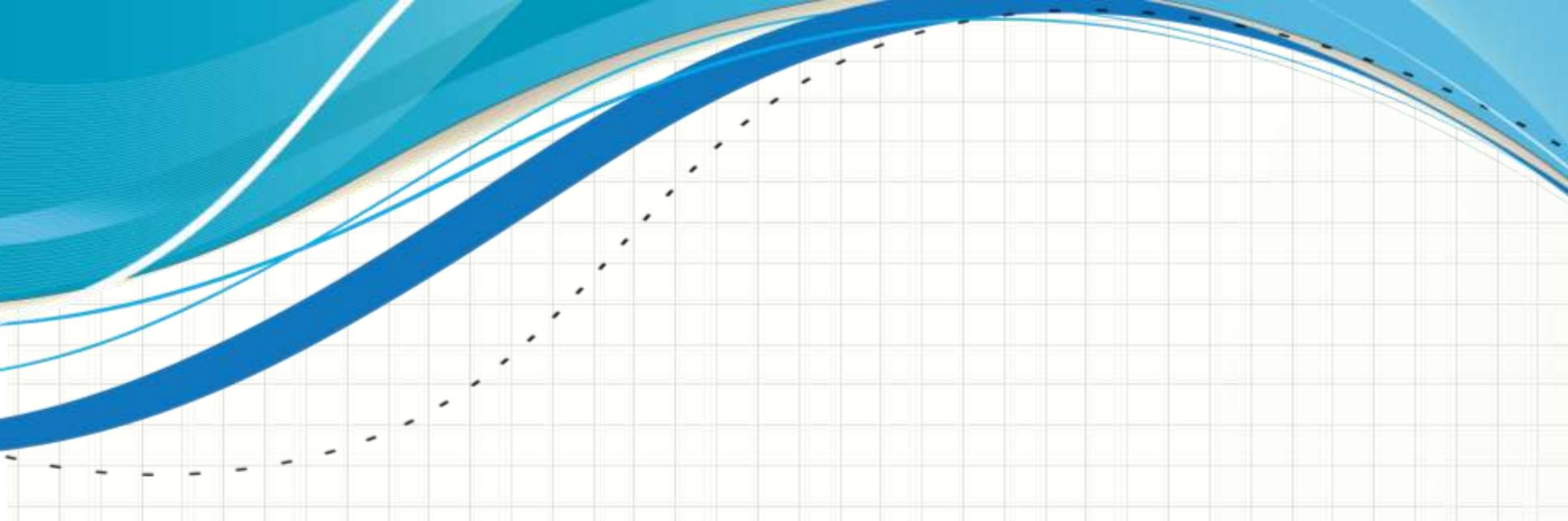
Resumo

- As aplicações Web funcionam como pequenos programas que rodam no servidor
- Estes programas, em Java, são feitos com a tecnologia Servlets
- Um Servlet basicamente recebe uma requisição (**request**) e coloca os resultados em uma resposta (**response**)
- **TAREFA**
 - Trabalho 2 Online!

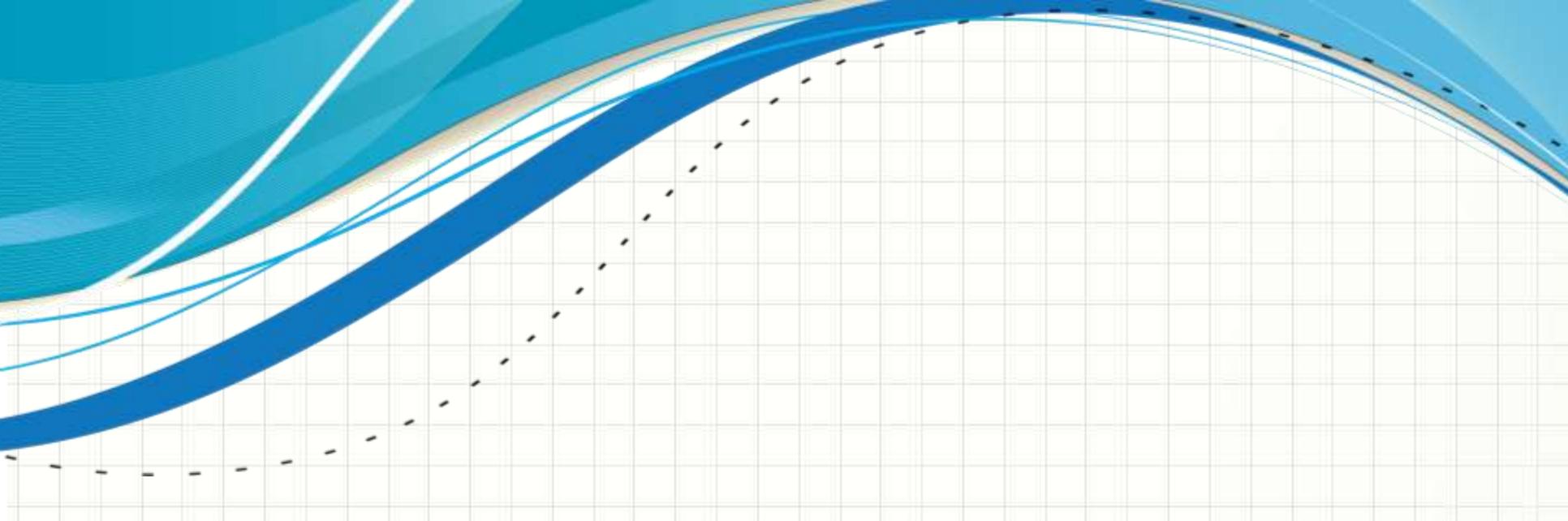
Próxima Aula



- Ainda não estou seguro com esses tais Servlets!
- Na próxima aula, exercitaremos mais...
- E veremos algumas novidades!



PERGUNTAS?



**BOM DESCANSO
A TODOS!**