

Introdução ao AngularJS

Professor Anderson Henrique



O que é REST?

Conceitualmente falando, o modelo REST (*REpresentational State Transfer*) representa nada mais que uma “nova” possibilidade para a criação de web services, cujas principais diferenças em relação ao modelo tradicional (SOAP) estão na utilização semântica dos métodos HTTP (GET, POST, PUT e DELETE), na leveza dos pacotes de dados transmitidos na rede e na simplicidade, fazendo desnecessária a criação de camadas intermediárias (Ex.: Envelope SOAP) para encapsular os dados.

Portanto, saber REST nos dias atuais é de fundamental importância para qualquer desenvolvedor ou empresa. Vejamos então como essa coisa toda funciona.

No mundo REST, uma [requisição HTTP](#) é equivalente a uma chamada de um método (operação) em um objeto (recurso) residente no servidor.

Como principais características de uma requisição REST, podemos destacar:

- O método HTTP é utilizado para determinar a operação a ser realizada em um determinado recurso. Em geral, utiliza-se o GET para recuperar, POST para criar, PUT para alterar e DELETE para apagar;
- O recurso, por sua vez, é indicado na URL da requisição;
- Parâmetros podem ser passados na própria URL e/ou no corpo na requisição;
- Os tipos de dados utilizados na requisição e na resposta devem ser acordados entre o servidor e o(s) cliente(s). [JSON](#) e [XML](#) estão entre os tipos mais utilizados.

Java e REST

Até pouco tempo atrás não existia uma maneira “oficial” de **criar serviços REST em Java**, devido tanto ao fato de ser uma abordagem relativamente nova, quanto ao fato dela usar basicamente o HTTP, não sendo necessárias bibliotecas adicionais.

Entretanto, com o intuito de padronizar e simplificar a criação desse tipo de serviço em Java, foi criada a especificação JAX-RS (*Java API for RESTful Web Services*). Parte integrante da plataforma Java EE 6, essa especificação define um conjunto de anotações, classes e interfaces para a **criação de serviços REST**.

Como implementação de referência da JAX-RS, temos o projeto **Jersey**, o qual utilizaremos em nossos exemplos.



RESTful Web Services in Java.

Na prática, a facilidade que a JAX-RS nos oferece é a forma de expor um serviço. Seguindo a tendência do Java EE 6, conseguimos fazer praticamente tudo com o uso de anotações. Na **Tabela 1**, listamos as principais anotações e seus propósitos.

Anotação	Descrição
@Path	Define o caminho (relativo) para acessar um recurso (classe ou método). Ex.: @Path("/bookmarks");
@GET, @POST, @PUT, @DELETE e @HEAD	Indica a qual método HTTP o método da classe vai responder. Seria algo semelhante aos métodos doGet() e doPost() da classe HttpServlet, porém utilizando anotações.
@Consumes e @Produces	Indicam o(s) tipo(s) de dados que um método vai receber e produzir, respectivamente. Ex.: @GET @Consumes(MediaType.APPLICATION_JSON) @Produces(MediaType.APPLICATION_JSON) public void listAll() {}
@PathParam, @QueryParam, @FormParam, @HeaderParam, @MatrixParam e @CookieParam	Servem para mapear os parâmetros da requisição nos parâmetros do método.

Tabela 1

Dessa forma, é possível notar como fica fácil disponibilizar um serviço REST, como veremos no nosso exemplo prático.

Exemplo: Agenda de contatos

Nosso exemplo consiste numa agenda de contatos, onde o usuário poderá cadastrar seus contatos no servidor a partir de uma interface gráfica extremamente simples, construída utilizando apenas [HTML](#) e [JavaScript](#).

Antes de tudo, no entanto, precisamos definir como será a interface do nosso serviço, no que diz respeito ao formato das URLs e métodos HTTP. A **Tabela 2** mostra o funcionamento esperado para o serviço através de exemplos.

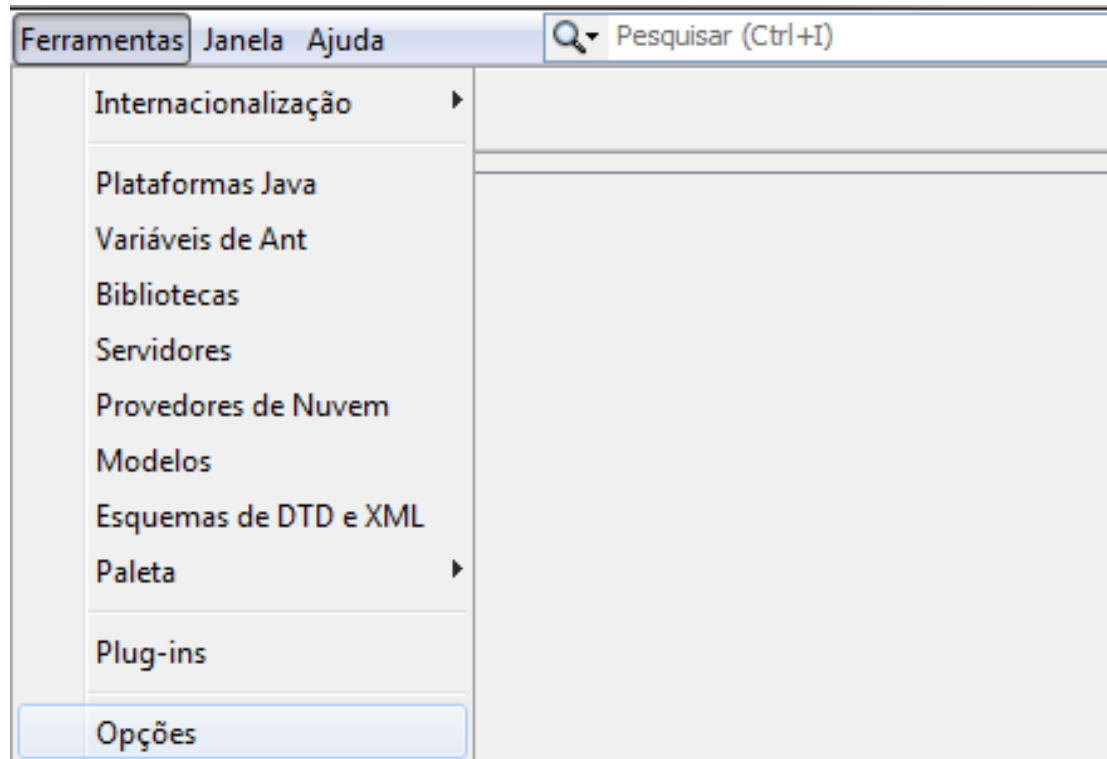
Método	URL	Descrição
GET	http://localhost:8080/contacts	Retorna uma lista com todos os contatos.
GET	http://localhost:8080/contacts/10	Retorna o contato de ID 10 ou erro 404 caso o contato não seja encontrado.
GET	http://localhost:8080/contacts/paulo	Retorna uma lista com os contatos que possuem a String "paulo" no seu nome.
POST	http://localhost:8080/contacts	Adiciona um novo contato. Os dados deverão ser passados no corpo da requisição no formato JSON. Um erro 400 (BAD REQUEST) deve ser retornado caso o nome não tenha sido informado.
PUT	http://localhost:8080/contacts/3	Atualiza o contato de ID 3. Os dados deverão ser passados no corpo da requisição no formato JSON.
DELETE	http://localhost:8080/contacts/15	Apaga o contato de ID 15.

Tabela 2

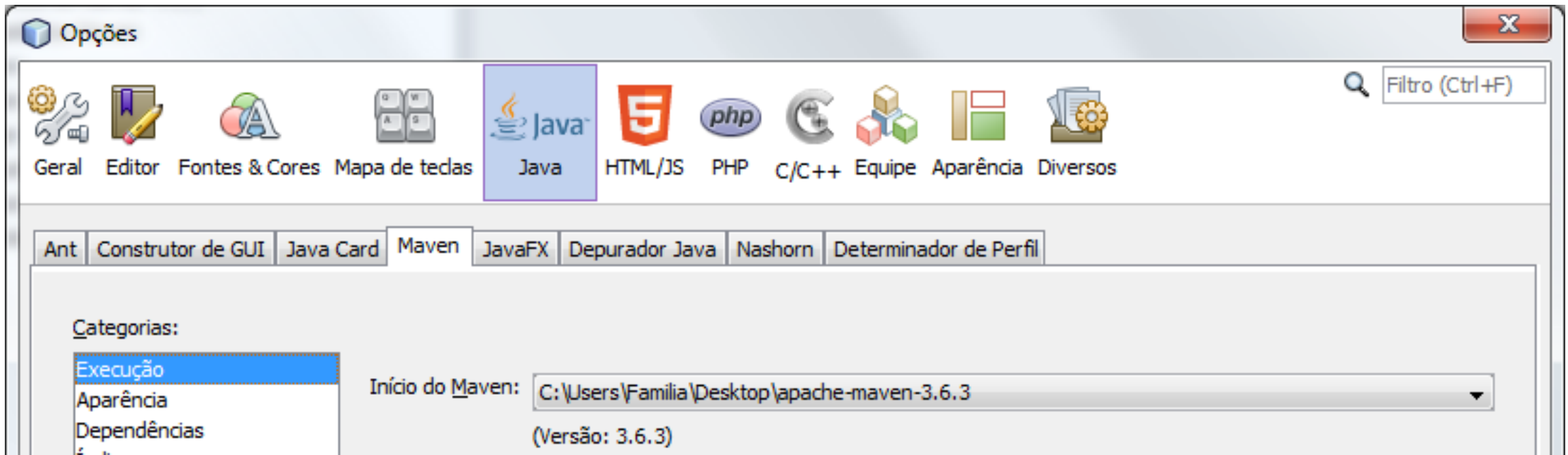
Antes de iniciar o nosso projeto na IDE Netbeans, vamos configurar o Apache Maven

Vamos configurar o **Apache Maven** na IDE **Netbeans**

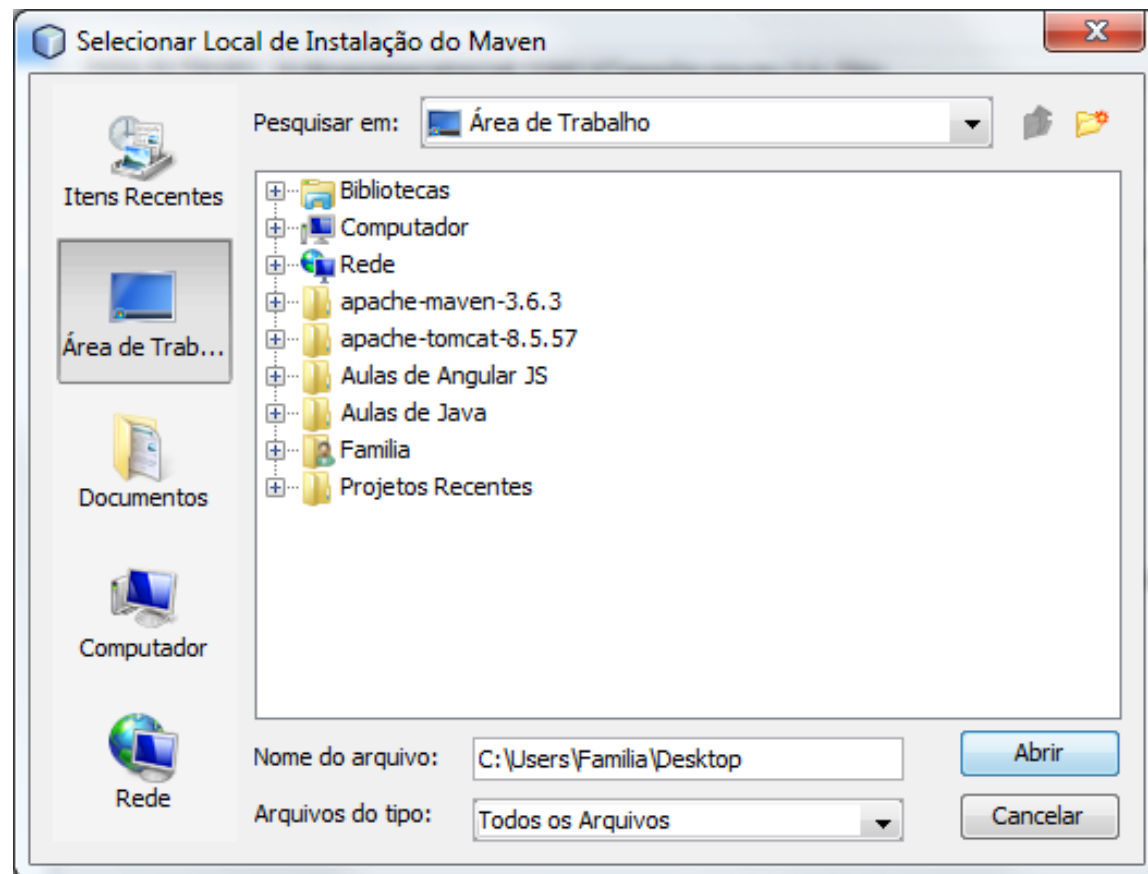
Vá no menu **Ferramentas** em **Opções**



Na janela Opções em **Categorias** selecione Execução e na Opção Início do Maven clique em cima e vá na opção Procurar...














Na próxima janela vá em Localização do servidor e clique no botão procurar:



Vá até a pasta onde em que foi descompactado o arquivo do Maven e clique em **abrir** e clique em **OK**

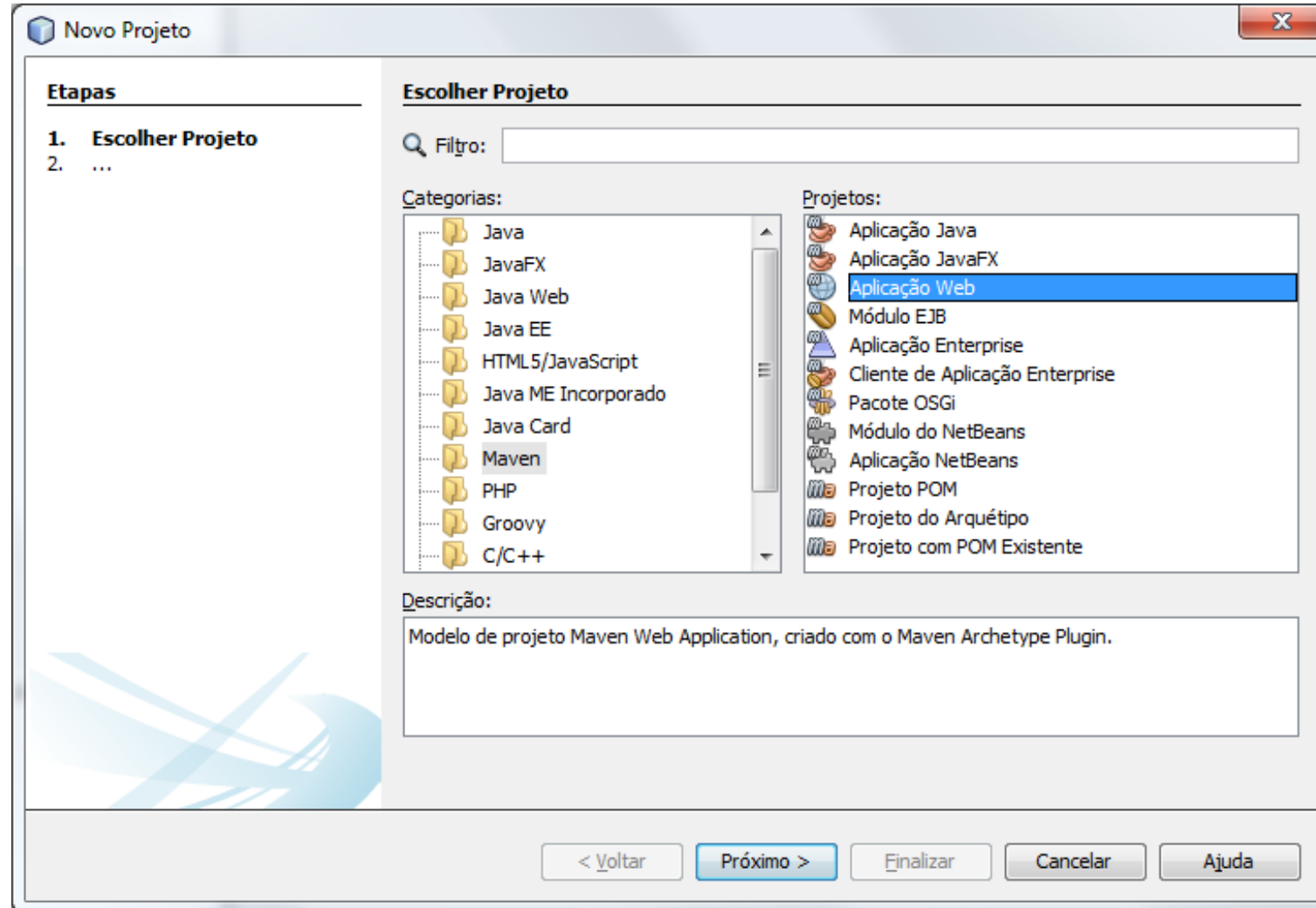
Vamos para a última configuração, na pasta do Apache Tomcat acesse a pasta conf e na pasta abra o arquivo Context.xml em um editor

Nome	Data de modificaç...	Tipo
 Catalina	23/08/2020 16:43	Pasta de arquivos
 catalina.policy	30/06/2020 22:50	Arquivo POLICY
 catalina.properties	30/06/2020 22:50	Arquivo PROPERT..
 context	30/06/2020 22:50	Documento XML
 jaspic-providers	30/06/2020 22:50	Documento XML
 jaspic-providers	30/06/2020 22:50	XML Schema File
 logging.properties	30/06/2020 22:50	Arquivo PROPERT..
 server	30/06/2020 22:50	Documento XML
 tomcat-users	23/08/2020 16:39	Documento XML
 tomcat-users	30/06/2020 22:50	XML Schema File
 web	23/08/2020 16:43	Documento XML

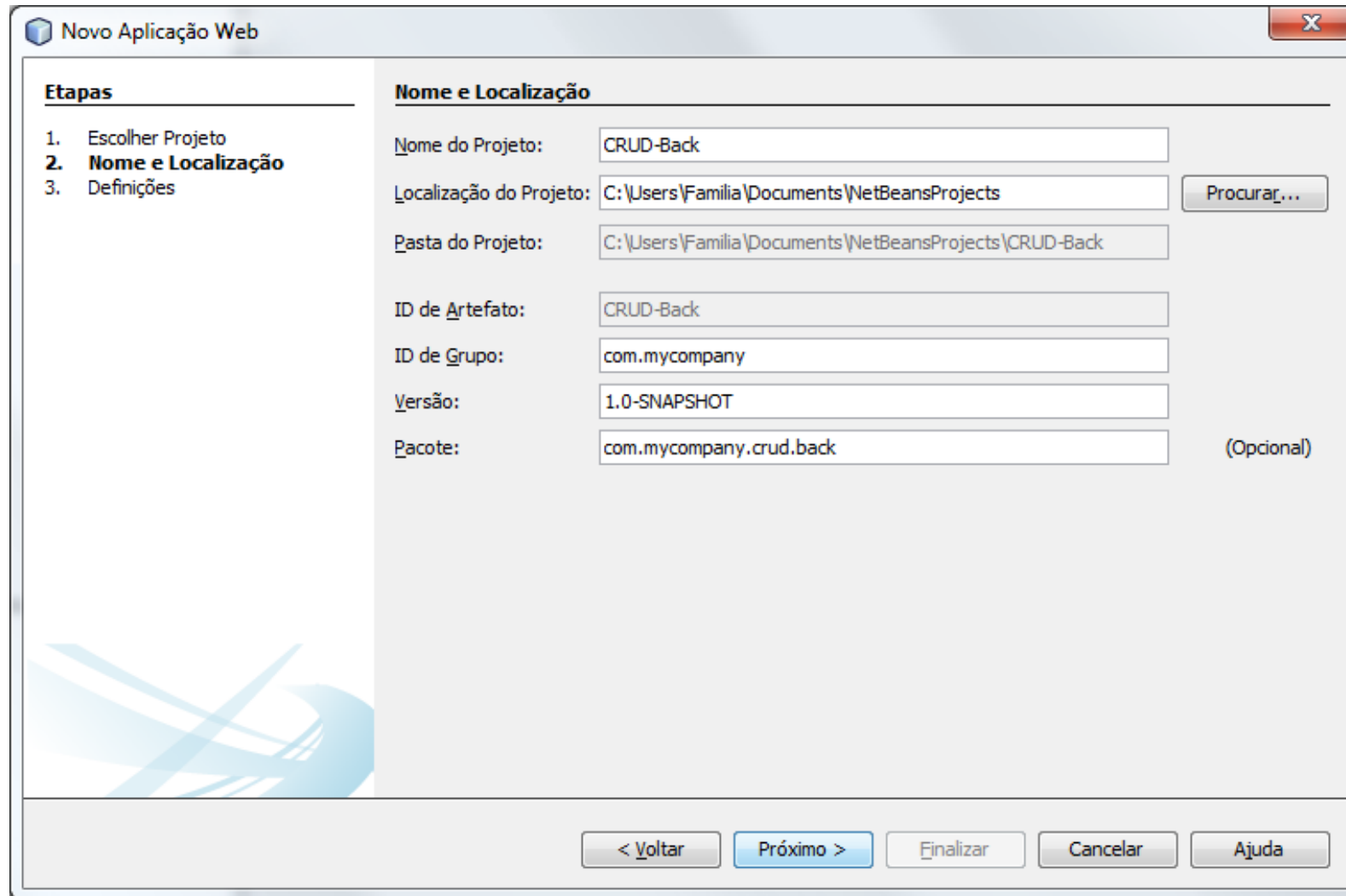
Na tag <Context> vamos adicionar um atributo **reloadable="true"**
salve o arquivo e o Tomcat está pronto para o desenvolvimento no
Ambiente IDE

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!--
3 Licensed to the Apache Software Foundation (ASF) under one or more
4 contributor license agreements. See the NOTICE file distributed with
5 this work for additional information regarding copyright ownership.
6 The ASF licenses this file to You under the Apache License, Version 2.0
7 (the "License"); you may not use this file except in compliance with
8 the License. You may obtain a copy of the License at
9
10 http://www.apache.org/licenses/LICENSE-2.0
11
12 Unless required by applicable law or agreed to in writing, software
13 distributed under the License is distributed on an "AS IS" BASIS,
14 WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
15 See the License for the specific language governing permissions and
16 limitations under the License.
17 -->
18 <!-- The contents of this file will be loaded for each web application -->
19 <Context reloadable="true">
```

Vamos criar um Novo Projeto no Netbeans, Maven, Aplicação Web



Depois de clicar em Próximo, precisamos definir o **Nome do Projeto** para **CRUD-Back**

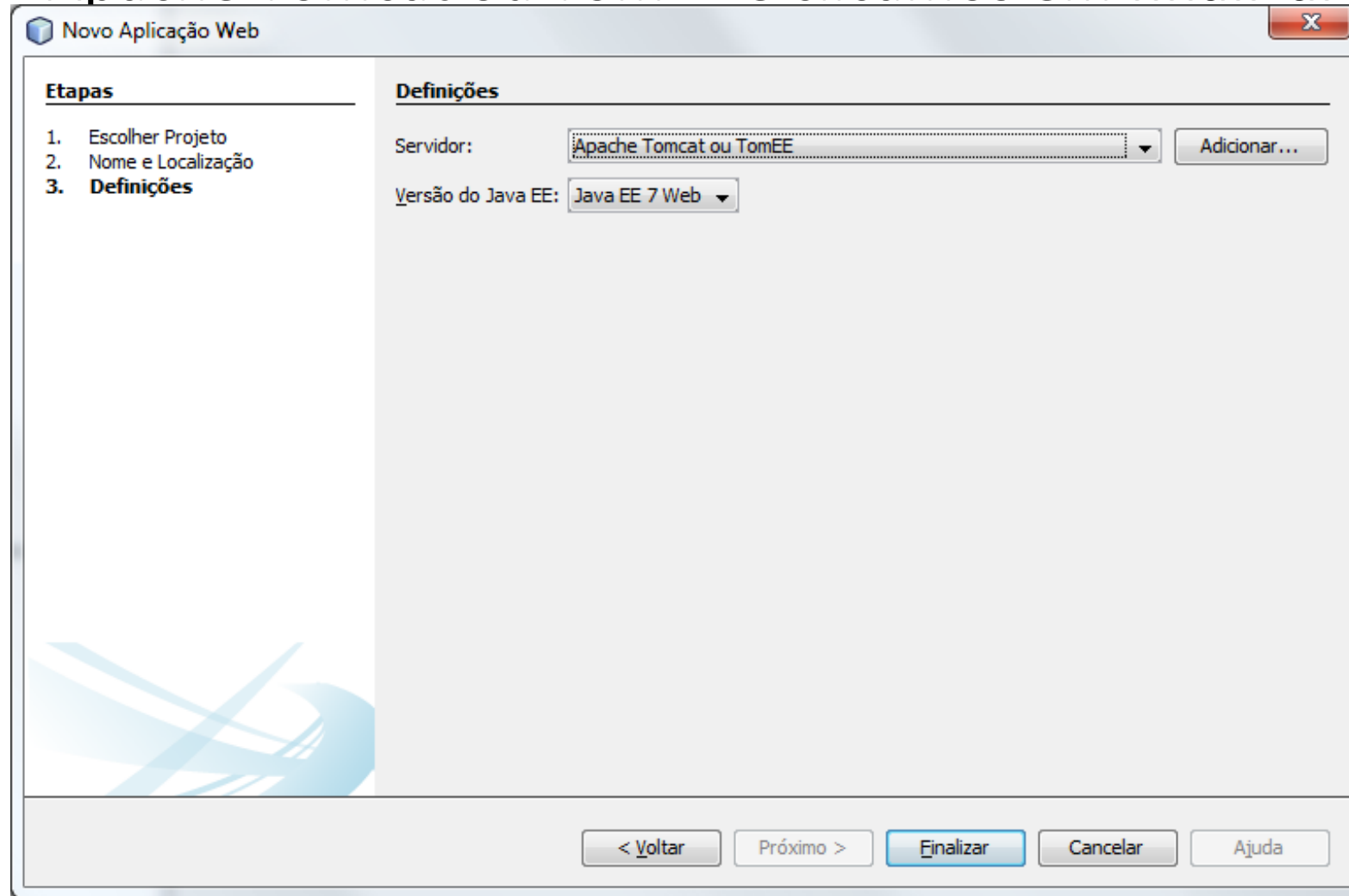


The screenshot shows the 'Novo Aplicação Web' (New Web Application) wizard in NetBeans IDE. The window has a title bar with a close button. On the left, there is a sidebar titled 'Etapas' (Steps) with three items: '1. Escolher Projeto', '2. Nome e Localização' (which is highlighted), and '3. Definições'. The main area is titled 'Nome e Localização' and contains several text input fields and buttons. The fields are: 'Nome do Projeto:' with the value 'CRUD-Back'; 'Localização do Projeto:' with the value 'C:\Users\Familia\Documents\NetBeansProjects' and a 'Procurar...' button to its right; 'Pasta do Projeto:' with the value 'C:\Users\Familia\Documents\NetBeansProjects\CRUD-Back'; 'ID de Artefato:' with the value 'CRUD-Back'; 'ID de Grupo:' with the value 'com.mycompany'; 'Versão:' with the value '1.0-SNAPSHOT'; and 'Pacote:' with the value 'com.mycompany.crud.back'. To the right of the 'Pacote' field is the text '(Opcional)'. At the bottom of the dialog, there are five buttons: '< Voltar', 'Próximo >' (highlighted in blue), 'Finalizar', 'Cancelar', and 'Ajuda'.

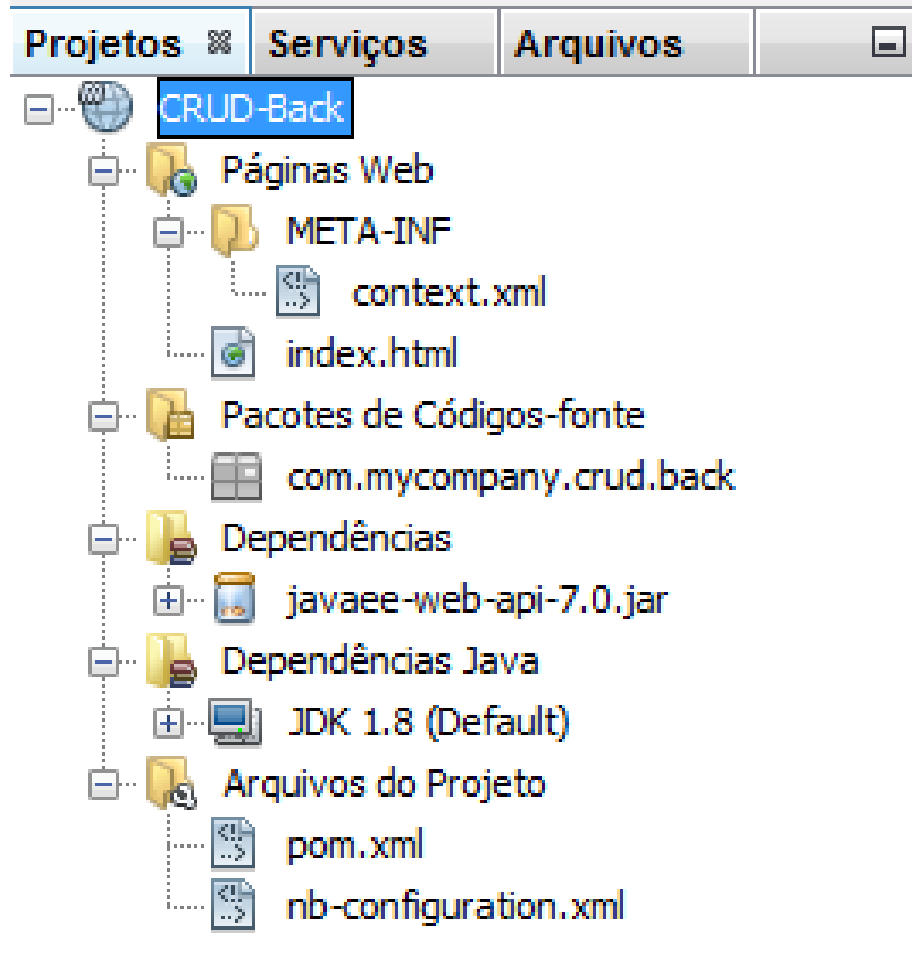
Nome e Localização	
Nome do Projeto:	CRUD-Back
Localização do Projeto:	C:\Users\Familia\Documents\NetBeansProjects Procurar...
Pasta do Projeto:	C:\Users\Familia\Documents\NetBeansProjects\CRUD-Back
ID de Artefato:	CRUD-Back
ID de Grupo:	com.mycompany
Versão:	1.0-SNAPSHOT
Pacote:	com.mycompany.crud.back (Opcional)

< Voltar Próximo > Finalizar Cancelar Ajuda

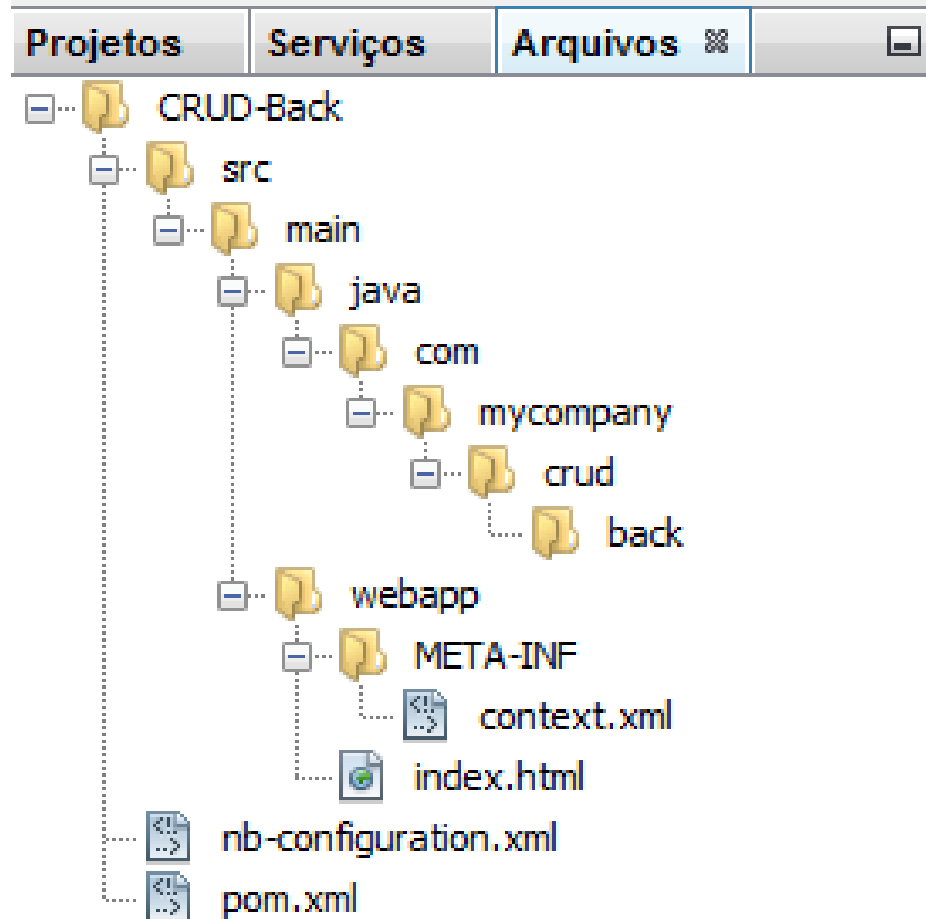
Depois de clicar em Próximo, precisamos definir o **servidor** vamos selecionar Apache Tomcat ou TomEE e clicamos em **finalizar**



Depois que criar o nosso projeto, esta será a estrutura encontrada na Guia Projetos

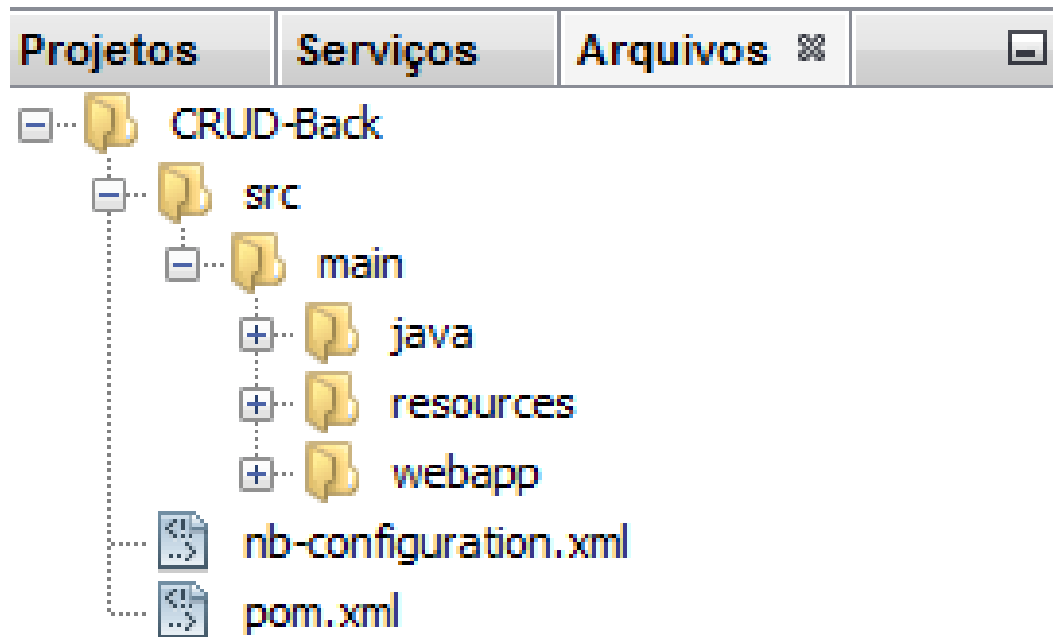


Depois que criar o nosso projeto, esta será a estrutura encontrada na Guia Arquivos



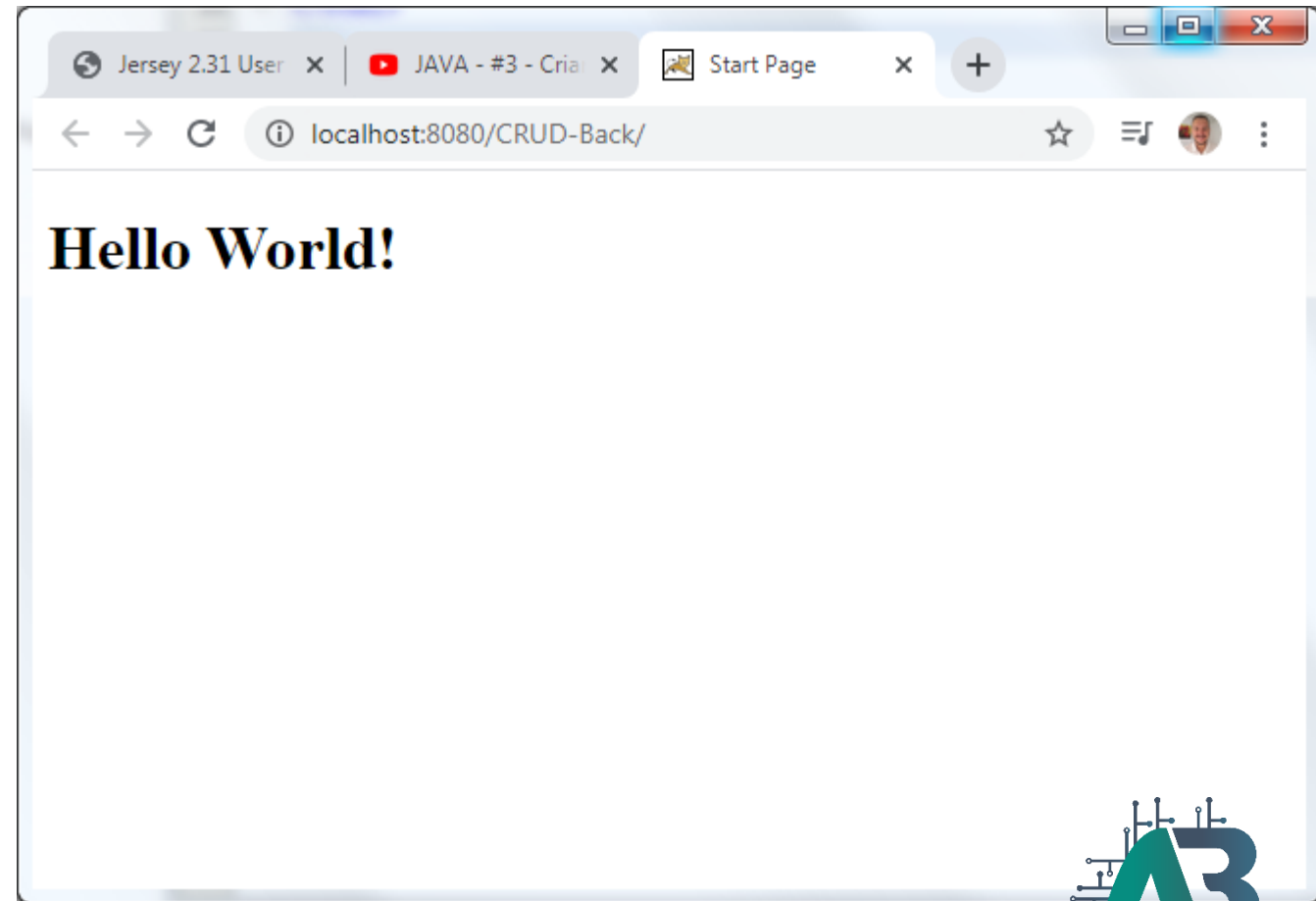
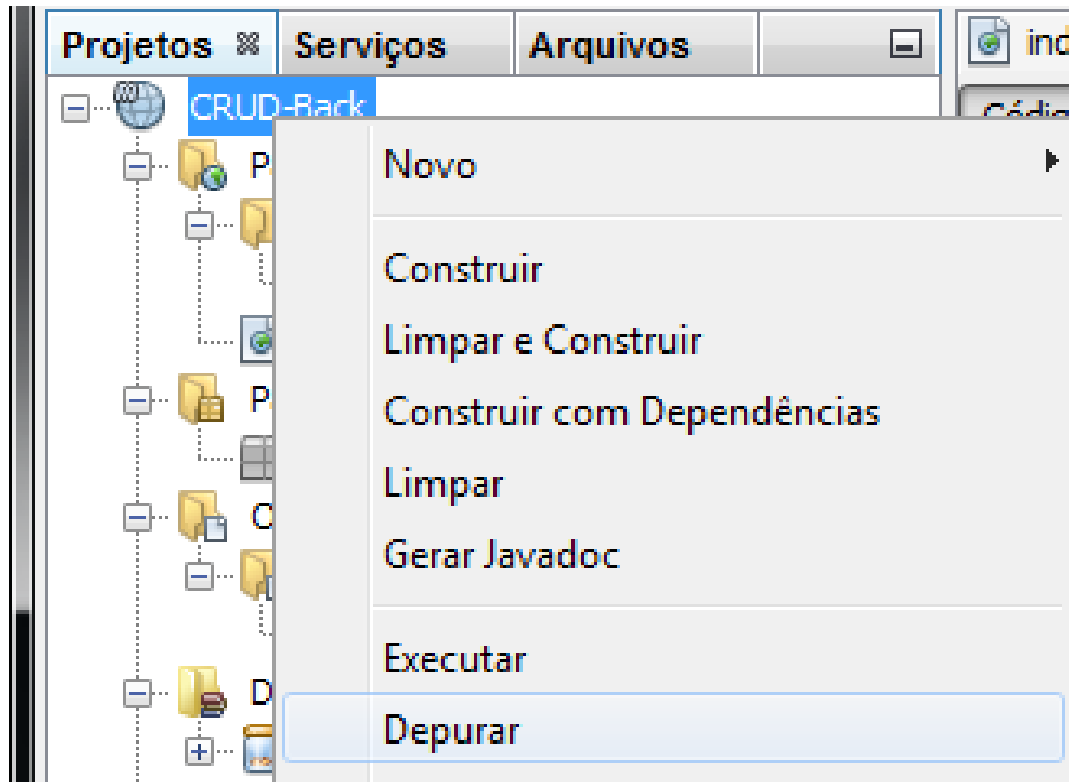
Dentro do diretório **main**, vamos criar um diretório chamado **resources**

Depois que criar o nosso projeto, esta será a estrutura encontrada na Guia Arquivos



Dentro do diretório **main**, vamos criar um diretório chamado **resources**

Agora com o nosso projeto pronto, vamos depurar (debug), clique com o botão auxiliar sobre o projeto CRUD-Back e na opção depurar



Dúvidas?

Professor Anderson Henrique



Para a próxima aula

01 – Aplicação Back-End em JAVA

02 – Criando nossa Classe, nosso Service e Configurando REST com Java utilizando MAVEN e JAX-RS

Professor Anderson Henrique

