caelum

APOSTILA JAVA E ORIENTAÇÃO A OBJETOS

CAPÍTULO 13

# Ferramentas: jar e javadoc

*"Perder tempo em aprender coisas que não interessam, priva-nos de descobrir coisas interessantes"*
*— Carlos Drummond de Andrade*

Ao término desse capítulo, você será capaz de:

- criar o JAR do seu aplicativo;
- colocar um JAR no build path do seu projeto;
- ler um javadoc;
- criar o javadoc do seu aplicativo.

## 13.1 - ARQUIVOS, BIBLIOTECAS E VERSÕES

Assim que um programa fica pronto, é meio complicado enviar dezenas ou centenas de classes para cada cliente que quer utilizá-lo.

O jeito mais simples de trabalhar com um conjunto de classes é compactá-los em um arquivo só. O formato de compactação padrão é o `ZIP` com a extensão do arquivo compactado `JAR`.

**O arquivo .jar**

O arquivo **jar** ou **J**ava **AR**chive, possui um conjunto de classes (e arquivos de configurações) compactados, no estilo de um arquivo `zip`. O arquivo `jar` pode ser criado com qualquer compactador `zip` disponível no mercado, inclusive o programa `jar` que vem junto com o JDK.

Para criar um arquivo jar do nosso programa de banco, basta ir ao diretório onde estão contidas as classes e usar o comando a seguir para criar o arquivo **`banco.jar`** com todas as classes dos pacotes `br.com.caelum.util` e `br.com.caelum.banco`:

```
jar -cvf banco.jar br/com/caelum/util/*.class br/com/caelum/banco/*.class
```

Para usar esse arquivo `banco.jar` para rodar o `TesteDoBanco` basta rodar o `java` com o arquivo `jar` como argumento:

```
java -classpath banco.jar br.com.caelum.util.TesteDoBanco
```

Para adicionar mais arquivos **`.jar`**, que podem ser bibliotecas, ao programa basta rodar o java da seguinte maneira:

```
java -classpath biblioteca1.jar;biblioteca2.jar NomeDaClasse
```

Vale lembrar que o ponto e vírgula utilizado só é válido em ambiente Windows. Em Linux, Mac e outros Unix, é o dois pontos (varia de acordo com o sistema operacional).

Há também um arquivo de manifesto que contém informações do seu jar como, por exemplo, qual classe ele vai rodar quando o jar for chamado. Mas não se preocupe pois, com o Eclipse, esse arquivo é gerado automaticamente.

**Bibliotecas**

Diversas bibliotecas podem ser controladas de acordo com a versão por estarem sempre compactadas em um arquivo `.jar`. Basta verificar o nome da biblioteca (por exemplo `log4j-1.2.13.jar`) para descobrir a versão dela.

Então é possível rodar dois programas ao mesmo tempo, cada um utilizando uma versão da biblioteca através do parâmetro **`-classpath`** do java.

**Criando um .jar automaticamente**

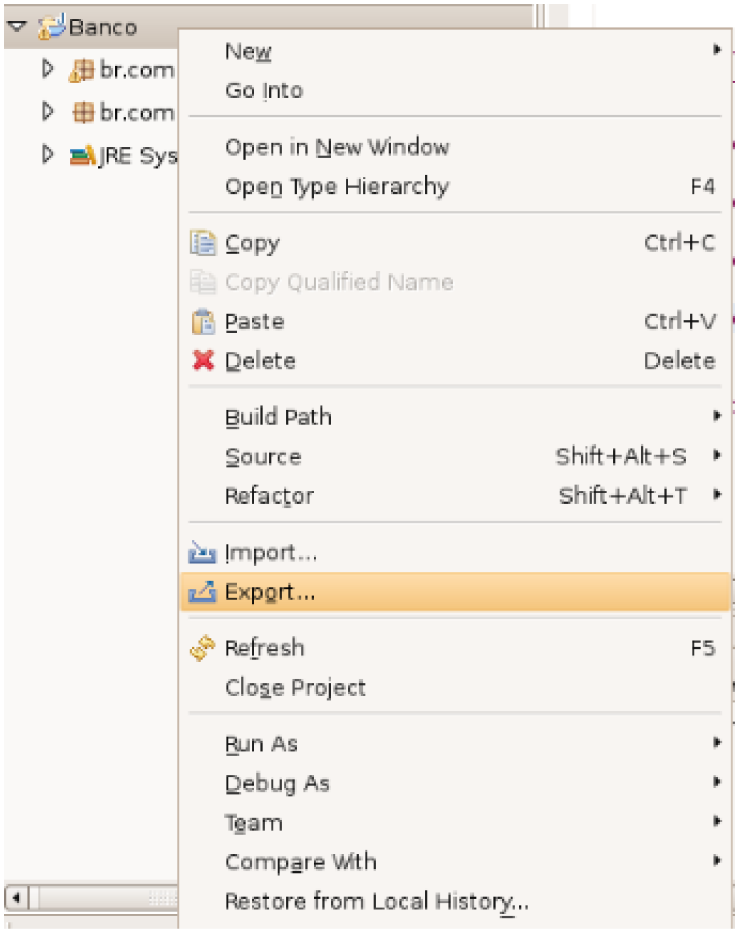
Existem diversas ferramentas que servem para automatizar o processo de deploy, que consiste em compilar, gerar documentação, bibliotecas etc. As duas mais famosas são o **ANT** e o **MAVEN**, ambos são projetos do grupo Apache.

O Eclipse pode gerar facilmente um jar, porém, se o seu build é complexo e precisa preparar e copiar uma série de recursos, as ferramentas indicadas acima possuem sofisticadas maneiras de rodar um script batch.

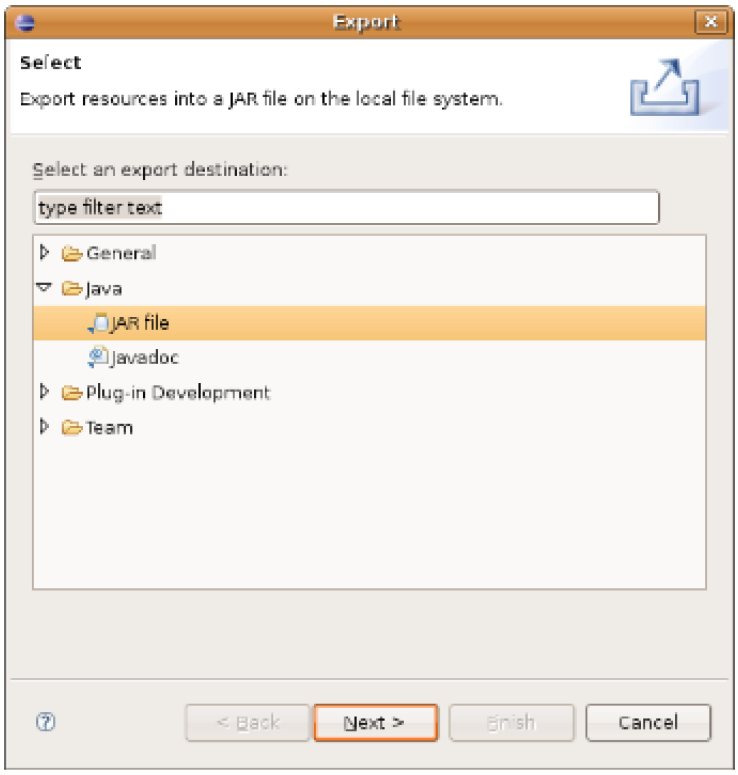
## 13.2 - Gerando o JAR pelo Eclipse

Neste exemplo, vamos gerar o arquivo JAR do nosso projeto a partir do Eclipse:

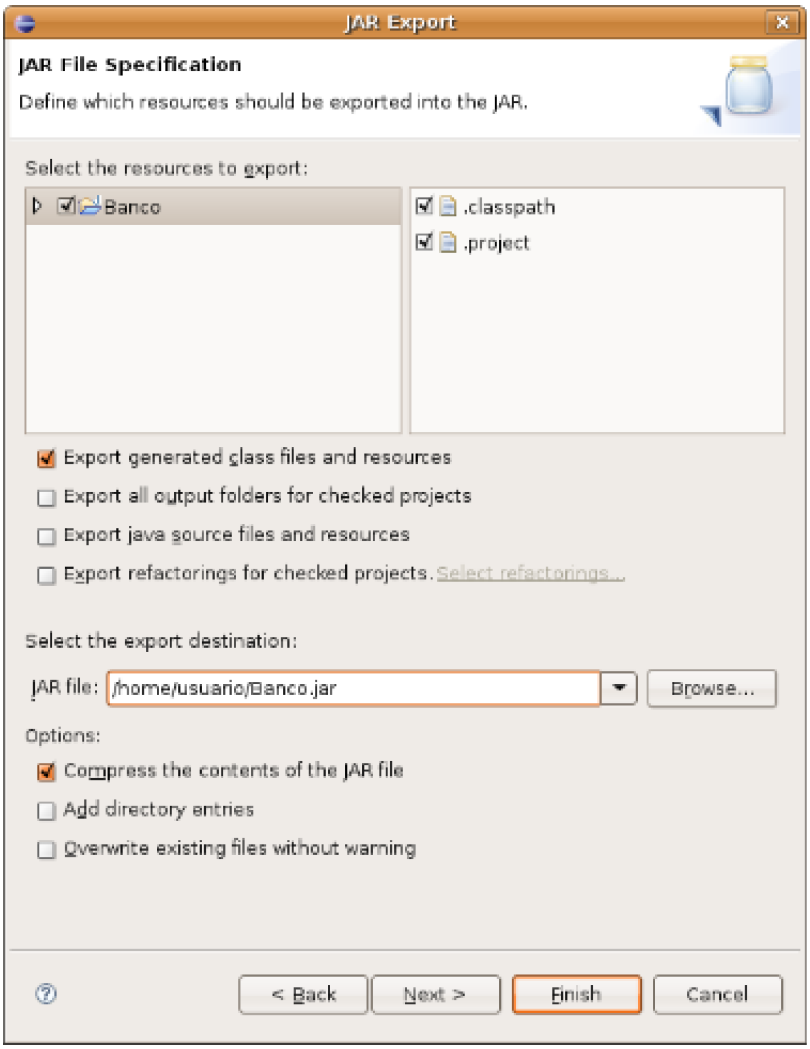
1. Clique com o botão direito em cima do nome do seu projeto e selecione a opção Export.




2. Na tela Export (como mostra a figura abaixo), selecione a opção "JAR file" e aperte o botão "Next".



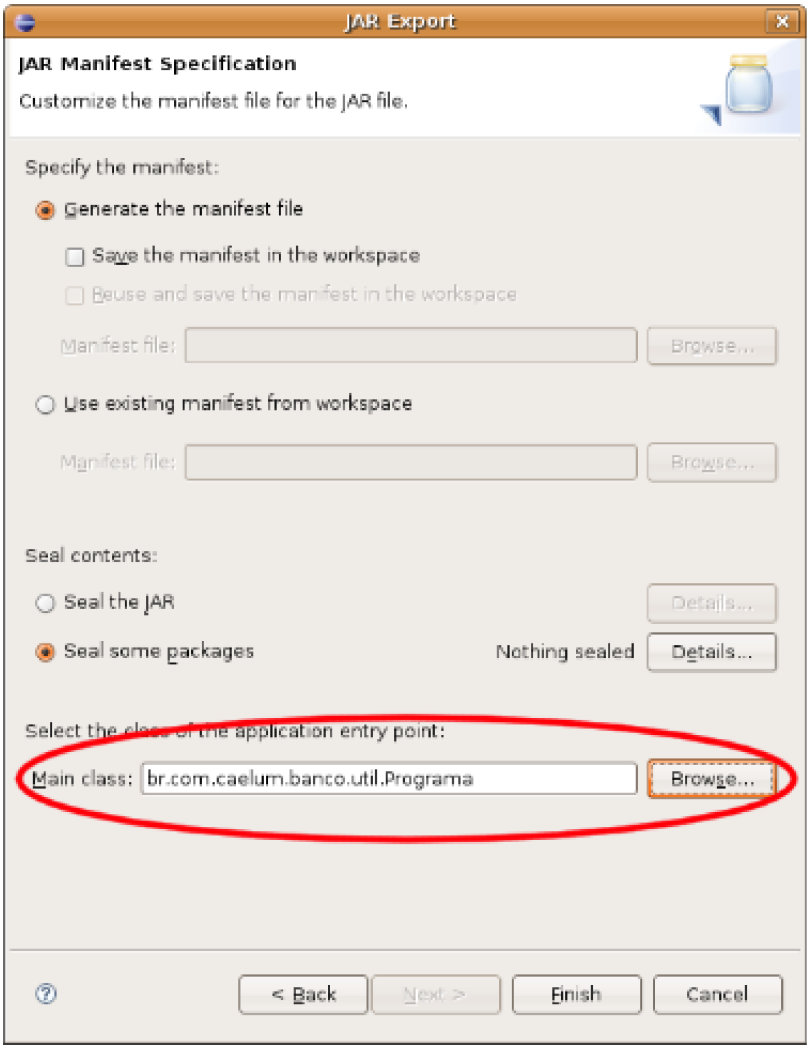

3. Na opção "JAR file:", selecione o local que você deseja salvar o arquivo JAR. E aperte "Next".




4. Na próxima tela, simplesmente clique em next, pois não há nenhuma configuração a ser

feita.

5. Na tela abaixo, na opção "select the class of the application entry point", você deve escolher qual classe será a classe que vai rodar automaticamente quando você executar o JAR.




6. Entre na linha de comando: `java -jar banco.jar`

É comum dar um nome mais significativo aos JARs, incluindo nome da empresa, do projeto e versão, como `caelum-banco-1.0.jar`.

## 13.3 - JAVADOC

Como vamos saber o que cada classe tem no Java? Quais são seus métodos, o que eles fazem?

Na sala de aula da Caelum, você pode acessar a documentação digitando na barra de endereço do Browser:

***/caelum/docs/api/index.html***

E, a partir da Internet, você pode acessar através do link:
http://download.java.net/jdk8/docs/api/index.html

No site da Oracle, você pode (e deve) baixar a documentação das bibliotecas do Java, frequentemente referida como "**javadoc**" ou API (sendo na verdade a documentação da API).


Nesta documentação, no quadro superior esquerdo, você encontra os pacotes e, no inferior esquerdo, está a listagem das classes e interfaces do respectivo pacote (ou de todos, caso nenhum tenha sido especificado). Clicando-se em uma classe ou interface, o quadro da direita passa a detalhar todos atributos e métodos.

Repare que métodos e atributos privados não estão aí. O importante é documentar o que sua classe faz, e não como ela faz: detalhes de implementação, como atributos e métodos privados, não interessam ao desenvolvedor que usará a sua biblioteca (ou, ao menos, não

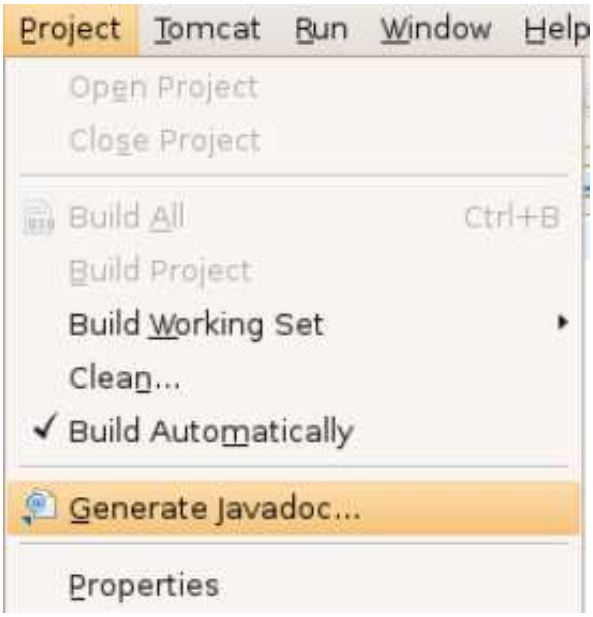
deveriam interessar).

Você também consegue gerar esse javadoc a partir da linha de comando, com o comando: `javadoc`.

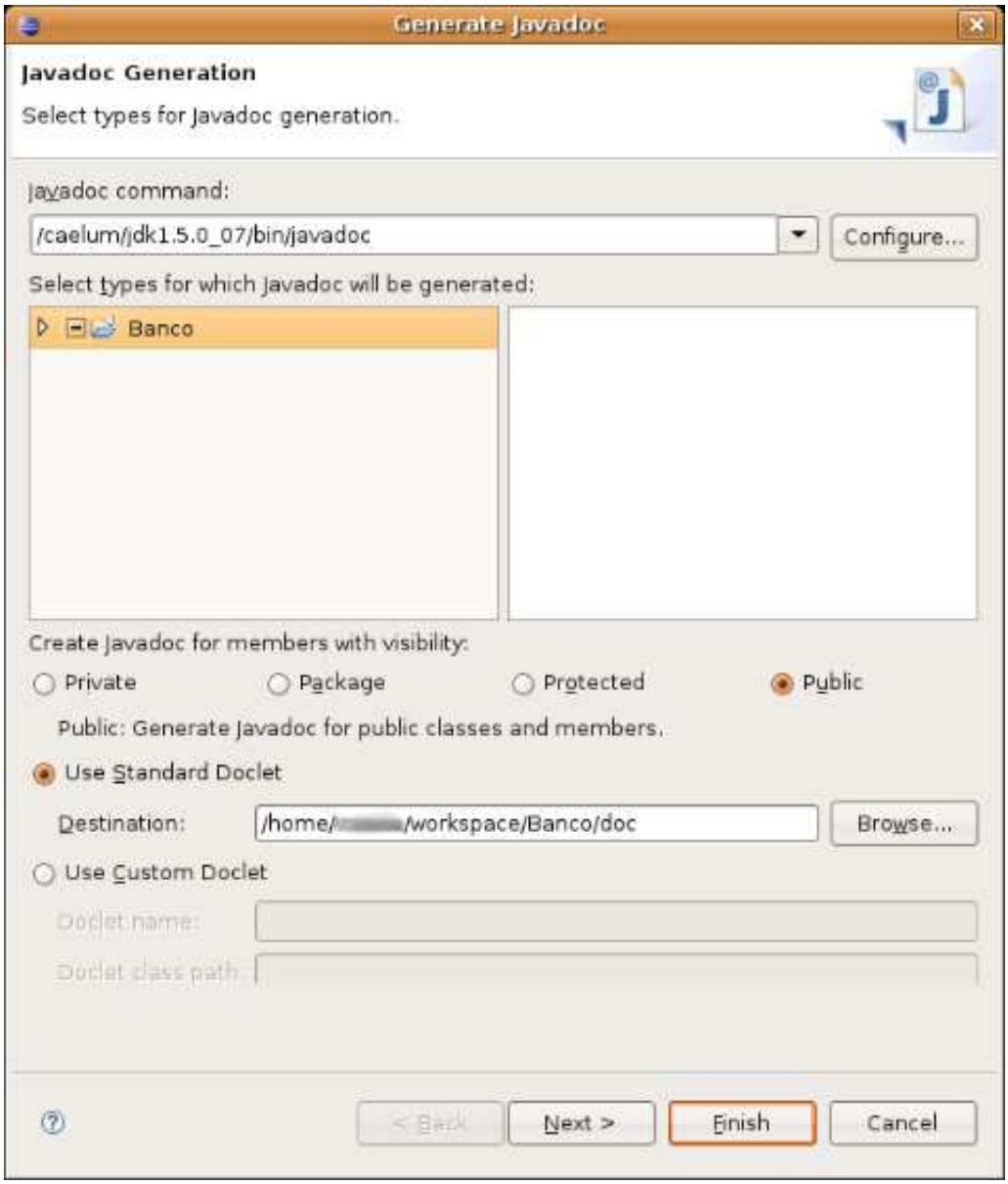
## 13.4 - Gerando o Javadoc

Para gerar o Javadoc a partir do Eclipse é muito simples, siga os passos abaixo:

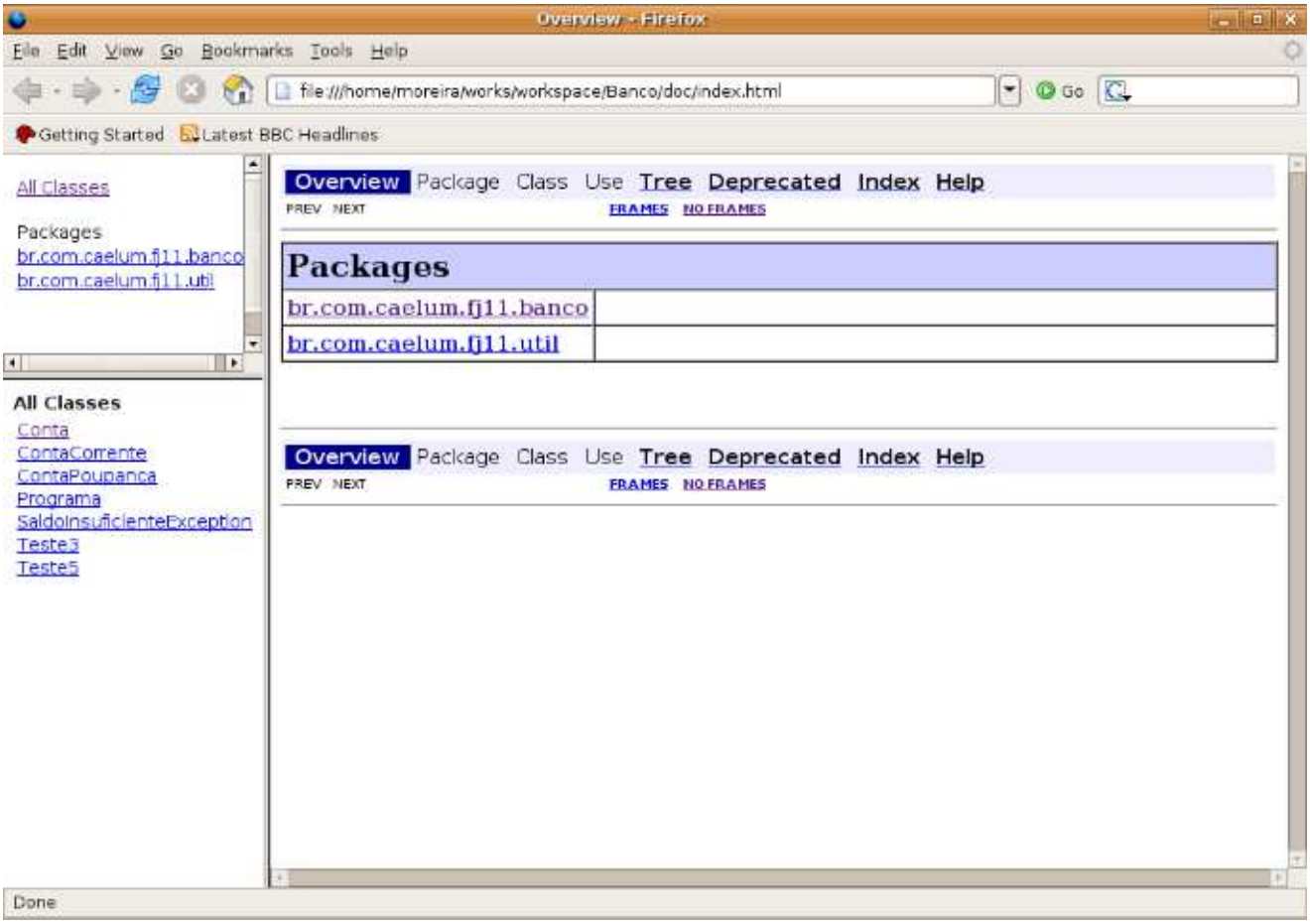
1. Na barra de menu, selecione o menu Project, depois a opção "Generate Javadoc...". (apenas disponível se estiver na perspectiva Java, mas você pode acessar o mesmo wizard pelo export do projeto).




2. Em seguida, aparecerão as opções para gerar a documentação do seu sistema, selecione todas as classes do seu sistema e deixe as outras opções como estão. Não esqueça de marcar o caminho da opção "Destination", pois é lá que estará sua documentação.




3. Abra a documentação através do caminho que você marcou e abra o arquivo index.html, que vai chamar uma página semelhante a essa da figura abaixo.



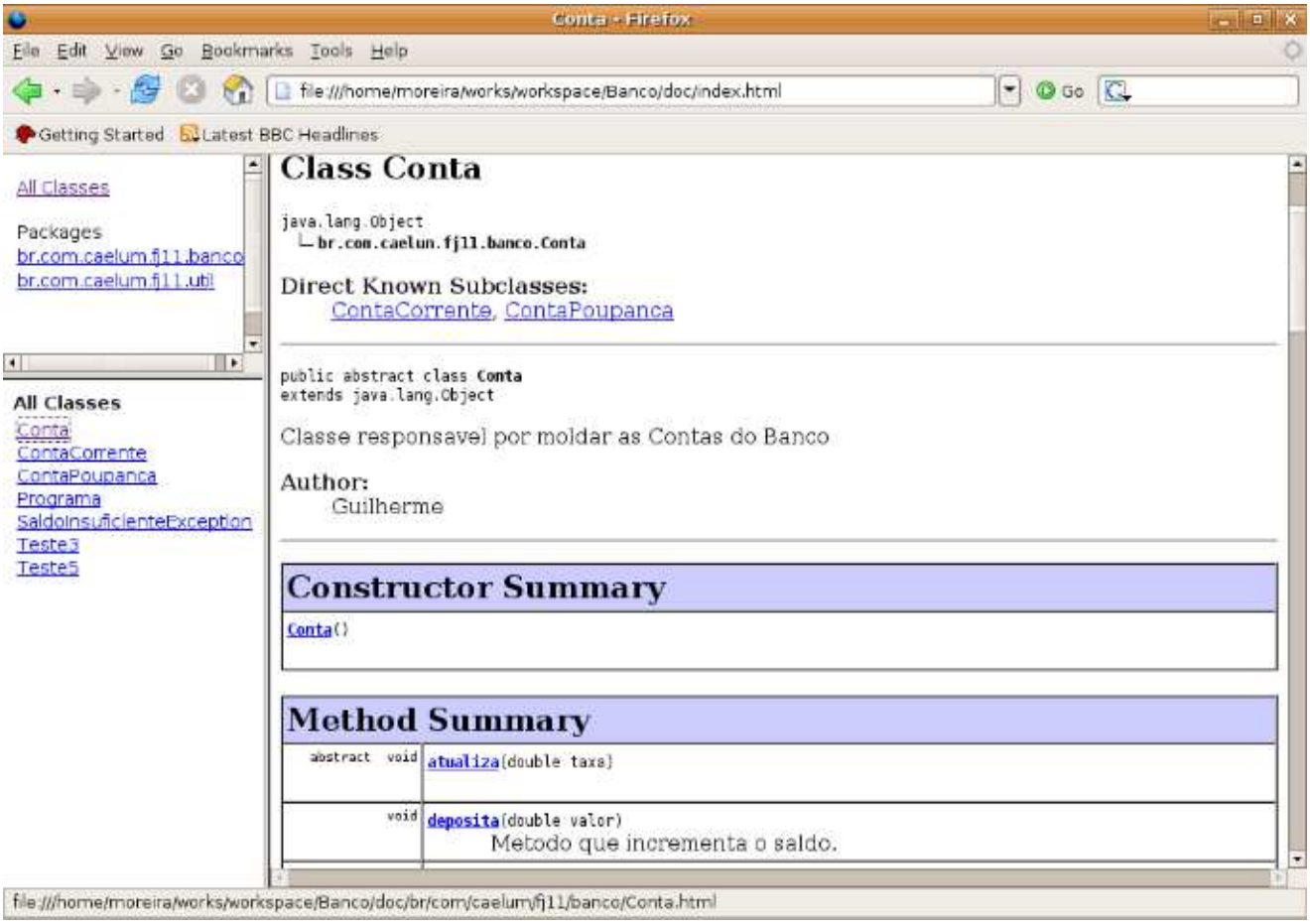

Para colocarmos comentários na documentação, devemos adicionar ao código, sob forma de comentário, abrindo o texto com `/**` e fechando com `*/` e, nas outras linhas, apenas colocando `*`. Também podemos definir outras informações neste texto, como: autor, versão, parâmetros, retorno, etc. Adicione alguns comentários ao seu projeto como abaixo:

```
/**
 * Classe responsável por moldar as Contas do Banco
 *
 * @author Manoel Santos da Silva
 */

public abstract class Conta{
  ...
}
```

Ou adicione alguns comentários em algum método seu:

```
/**
 * Metodo que incrementa o saldo.
 * @param valor
 */

public void deposita(double valor) {
  ...
}
```

Veja como ficou:




## 13.5 - Exercícios: Jar e Javadoc

1. Gere um jar do seu sistema com o arquivo de manifesto. Execute-o com `java -jar`:

```
java -jar caelum-banco-1.0.jar
```

Se o Windows ou o Linux foi configurado para trabalhar com a extensão *.jar*, basta você dar um duplo clique no arquivo, que ele será "executado": o arquivo `Manifest` será lido para que ele descubra qual é a classe com `main` que o Java deve processar.

2. Gere o Javadoc do seu sistema. Para isso, vá ao menu *Project*, depois à opção *Generate Javadoc*, se estiver na perspectiva Java. Se não, dê um clique com o botão direito no seu projeto, escolha *Export* e depois *javadoc* e siga o procedimento descrito na última seção deste capítulo.

Independente da perspectiva que você usa no Eclipse, você também pode usar o **ctrl + 3** e começar a escrever `JavaDoc`, até que a opção de exportar o JavaDoc apareça.

**Interface versus implementação novamente!**

Repare que a documentação gerada não mostra o conteúdo dos métodos, nem

atributos e métodos privados! Isso faz parte da implementação, e o que importa para quem usa uma biblioteca é a interface: o que ela faz.



CAPÍTULO ANTERIOR:

Pacotes - Organizando suas classes e bibliotecas

PRÓXIMO CAPÍTULO:

O pacote java.lang