


[< VOLTAR](#)

Organização em Pacotes

Organizar o código e a estrutura do projeto é primordial para lhe auxiliar na hora de dar manutenção ao seu programa ou para colocarmos as coisas certas nos lugares certos. Muitas vezes trabalhamos em equipe no mesmo projeto e manter a organização é muito importante, para que ninguém se perca ou não crie coisas em locais incorretos, dificultando sua busca. Nesta aula você aprenderá a como organizar seu projeto utilizando pacotes de código-fonte. Estamos falando, aqui, do famoso "package".

NESTE TÓPICO

- > Pacotes de código-fonte
- > Boas práticas para nomeação de pacotes
- > Classes em pacotes diferentes podem conversar 
- > Exemplo em Java
- > Pacotes de bibliotecas do Java
- > Resumo desta aula
- > Referências



Pacotes de código-fonte



O conceito de pacote é bastante simples. Segundo Teruel, “pacotes são contêineres (ou pastas) que se cria em um projeto para agrupar classes que possuem alguma característica comum ou afinidade”.

Isso quer dizer que se você tem, em seu projeto, um conjunto de classes que são responsáveis por coisas parecidas ou fazer parte de um mesmo domínio de conhecimento dentro da sua aplicação, elas podem ser agrupadas em um pacote único.

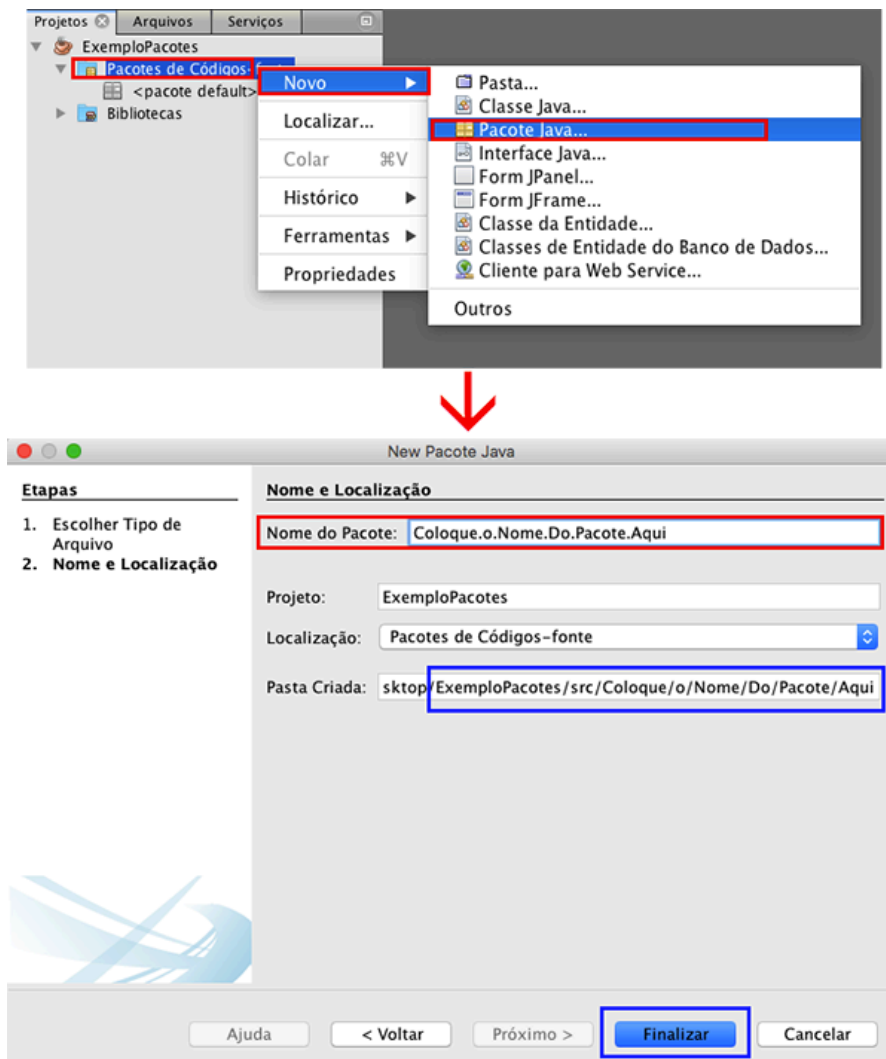
Por exemplo, se você está desenvolvendo um projeto de um sistema de vendas, você poderá ter um pacote para classes que se relacionam com produtos, um pacote de classes que se relacionam com clientes, outro pacote para classes que se relacionam com os funcionários etc.

O NetBeans é uma IDE que nos auxilia muito na programação Java pois ele organiza de forma visual todas as informações necessárias para o projeto. De qualquer forma, nunca se esqueça que o NetBeans está, na verdade,

mostrando uma outra forma de visualização de seu sistema de arquivos, ou seja, o seu projeto armazena diversos arquivos locais em seu computador e o NetBeans apenas os mostra e administra de forma organizada.

Quando uma classe está dentro de um pacote é preciso acrescentar, no início dela (antes da definição da classe), informações sobre o pacote, com a palavra reservada **package**. Os exemplos a seguir mostrarão como fazer isso.

Para criar um pacote no NetBeans, basta clicar com o botão direito sobre a pasta “Pacotes de código-fonte” e, em seguida, clicar em Novo → Pacote de código fonte. Um assistente abrirá para que você digite o nome do seu pacote, conforme a imagem abaixo:



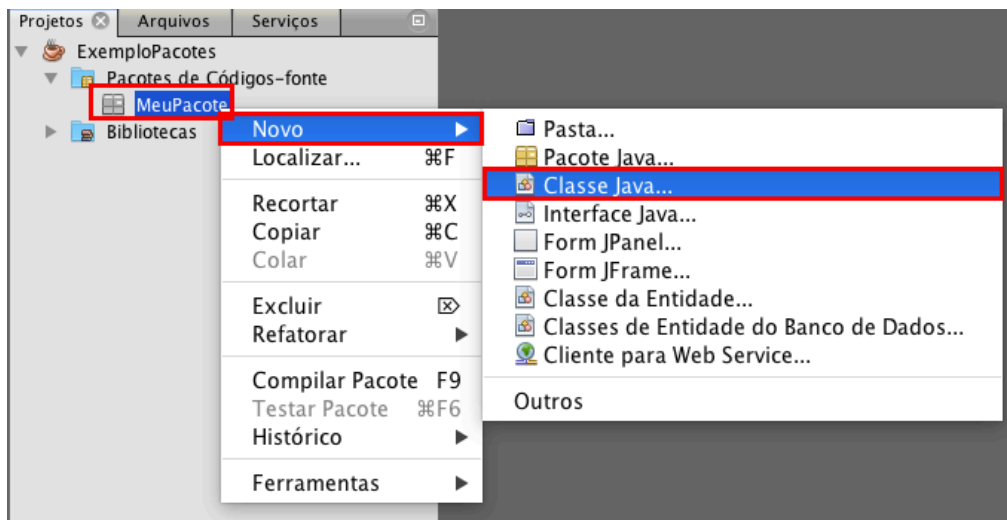
Criando um pacote no NetBeans

Repare no destaque em azul da imagem acima. O destaque está mostrando o caminho real no computador da localização do pacote. Veja que o pacote foi nomeado com pontos entre as palavras (você verá que isso é uma recomendação). Veja, agora, que o NetBeans irá criar uma subpasta para cada ponto acrescentado.

NETBEANS X ESTRUTURA REAL DE ARQUIVOS

Lembre-se que, fisicamente, todo o seu código fonte fica dentro da pasta “src” (de source em inglês, que quer dizer “origem”). Os pacotes e classes são criados dentro desta pasta. O programa compilado fica dentro da pasta “bin” (de binários).

Logo em seguida, para se criar classes dentro destes pacotes, é preciso clicar com o botão direito no pacote desejado e, em seguida, clicar em Novo → Classe Java.



Criando uma classe dentro de um pacote, no NetBeans

Mas antes de sairmos criando pacotes deliberadamente, é preciso aprender uma coisa **muito** importante antes: Boas práticas para nomear pacotes! Isso mesmo, existem boas práticas internacionais de programação Java que definem como o nome dos pacotes devem ser.

Boas práticas para nomeação de pacotes

Quando você cria um novo pacote, se você colocar qualquer nome, mesmo que fora das boas práticas de programação Java, não dará erro. Contudo, essas boas práticas existem e é muito importante segui-las para sempre termos uma programação limpa, robusta e de fácil manutenção.

Alguns pontos são importantes na hora de nomear os pacotes (Teruel):

- Não utilizar espaços (o NetBeans já não lhe permite fazer isso pelo assistente de criação de pacotes)
- Utilizar apenas letras minúsculas
- Se o nome do pacote tiver mais de uma palavra, utilizar pontos para separá-las. Por exemplo: cadastro.clientes

Embora o Java, hoje, pertença a Oracle, a Sun Microsystems, criadora da linguagem, usava um padrão de nomenclatura muito interessante, que muitos programadores seguem até hoje e existem recomendações de boas práticas de programação para que este padrão seja seguido. Utilizar o *nome relativo* da empresa para nomear o pacote, mas de trás para frente. Para facilitar, veja alguns exemplos de nomes:

- br.uninove.poo.sistema
- br.uninove.poo.exemplos
- br.uninove.poo.exemplos.polimorfismo
- br.com.josefinocorp.erp.modulos.clientes
- etc...

Pode parecer estranho, mas os desenvolvedores da Sun diziam que isso ajuda a identificar quando você está trabalhando com classe criadas pela sua empresa (ou por você) e quando se está trabalhando com classes de terceiros, pois o nome do pacote iria ser diferente. De qualquer forma, isso é apenas uma recomendação. No mercado, essa prática é muito comum.

Classes em pacotes diferentes podem conversar



Isso mesmo, todas as classes em Java podem “enxergar” umas às outras (dependendo da visibilidade das classes, claro, mas isso será tratado noutro tópico).

Por padrão, as classes que estão no mesmo pacote “se enxergam” diretamente, ou seja, basta referencia-las.

Contudo, se você está trabalhando com classes que estão em pacotes diferentes, é preciso importar as classes ou o pacote inteiro dentro da classe a qual você está referenciando a outra.

Neste caso, usamos a palavra reservada: **import**.

Se você quer importar apenas uma classe daquele pacote, é preciso acrescentar no início de seu código a instrução **import {nome completo do pacote}.{nome da classe};**

Contudo, se você quer importar todas as classes daquele pacote, pode-se utilizar o “*” (asterisco) para importa-las, com a instrução: **import {nome do pacote completo}.*;**

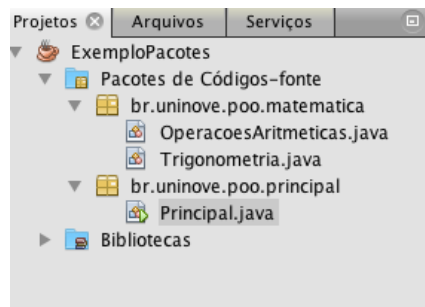
Em seguida, alguns exemplos serão mostrados para cada um destes cenários.

Exemplo em Java

Para exemplificar e demonstrar a utilização dos pacotes, vamos criar uma aplicação matemática com três classes para calcular a área de um triângulo: Duas para operações matemáticas (uma de operações simples e outra de operações de trigonometria) e outra classe para a execução do código.

Neste cenário, as duas classes com funções matemáticas ficarão no pacote “br.uninove.poo.matematica” e a classe principal ficará em outro pacote, no br.uninove.poo.principal.

Assim:



Organização do projeto em pacotes

E a codificação das classes, deste cenário, pode ser vista a seguir:



Classe OperacoesAritmeticas:

```
1. //instruções de pacote dessa classe
2.
3. package br.uninove.poo.matematica;
4.
5. public class OperacoesAritmeticas {
6.
7.     public float multiplica(float a, float b) {
8.         return a * b;
9.     }
10.
11.     public float soma(float a, float b) {
12.         return a + b;
13.     }
14.
15.     public float subtrai(float a, float b) {
16.         return a - b;
17.     }
18.
19.     public float divide(float a, float b){
20.         try{
21.             return a / b;
22.
23.         }catch (Exception ex){
24.             System.out.println("Erro" + ex.getMessage());
25.             return 0;
26.         }
27.     }
28. }
```

Classe Trigonometria:

```
1. //instruções de pacote dessa classe
2. package br.uninove.poo.matematica;
3.
4. public class Trigonometria {
5.
6.     public float areaTriangulo(float base, float altura) {
7.         OperacoesAritmeticas oa = new OperacoesAritmeticas();
8.         //área do triangulo = (base X altura) / 2
9.         //Usa o objeto oa para multiplicar:
10.         float area = oa.multiplica(base, altura) / 2;
11.         return area;
12.     }
13. }
```



E, finalmente, a classe Principal:

No códigos-fonte acima, repare nas declarações de pacotes (package) e importações (import). Repare, também, que fizemos uma importação de uma classe pronta do Java. A seguir, discutiremos os pacotes de bibliotecas da linguagem.

Implemente o projeto acima e veja, em seu console de execução, o projeto acima em ação.

```
1. //instruções de pacote dessa classe
2. package br.uninove.poo.principal;
3.
4. //Importa a classe trigonometria. Se quiséssemos importar todas as
5. //classes do pacote, ficaria assim:
6. //      import br.uninove.poo.matematica.*
7. import br.uninove.poo.matematica.Trigonometria;
8.
9. //Importa uma classe do Java, para usar o Scanner do teclado
10. import java.util.Scanner;
11.
12. public class Principal {
13.
14.     public static void main(String args[]) {
15.         Trigonometria tri = new Trigonometria();
16.         Scanner sc = new Scanner(System.in);
17.         float base, altura;
18.
19.         System.out.print("Informe o valor da base do triângulo: ");
20.         base = sc.nextFloat();
21.
22.         System.out.print("Informe o valor da altura do triângulo: ");
23.         altura = sc.nextFloat();
24.
25.         System.out.print("\nA área deste triângulo é: ");
26.         System.out.println(tri.areaTriangulo(base, altura));
27.     }
28. }
```

Pacotes de bibliotecas do Java



Java possui diversas API (API–Application Programming Interface - Interface de Programação de Aplicativos) com diversas funções prontas, que vêm com a instalação da JDK.

Alguns exemplos de APIs do Java que podem ser utilizados, podem ser vistos abaixo:

java.lang	É o único pacote importado automaticamente pelo Java. Possui classes essenciais para o funcionamento de qualquer programa.
java.util.*;	Importação de todas as classes que pertencem ao pacote java.util, que possui várias classes importantes.
java.util.Date;	Indica que será usada a classe Date do pacote java.util que está no diretório de classes \java\util.
java.util.Calendar	Com a classe Calendar podemos realizar vários tipos de operações envolvendo data e hora.
java.awt.*;	API gráfica. criação de gráficos e imagens básicos além de interfaces com o usuário.
javax.swing.*	API gráfica. Ciação de componentes de interface mais sofisticada com o usuário.
java.net.URL;	Importa apenas a classe URL do pacote java.net.

Resumo desta aula

Pronto, chegamos ao fim de mais uma aula e, a partir de agora, além de poder programar coisas mais avançadas, você já sabe como organizar seu projeto em pacotes, importar classes de outros pacotes e utilizar APIs que podem ser importadas diretamente em seu código-fonte.

Para praticar, tente pegar alguns programas que você já desenvolveu e organiza-los em pacotes, dividindo as classes por responsabilidades. Boa programação e bons estudos.

Quiz

Exercício Final

Organização em Pacotes



INICIAR ➤

Referências

Deitei P. e Deitel H., 2010, Java : Como programar, 8ª Edição, Pearson Pretice Hall

Teruel, E. C., 2015, Programação Orientada a Objetos com Java - sem mistérios - 1ª Ed., Editora Uninove

Schildt, H., 2015, Schildt, Java para iniciantes : crie, compile e execute programas Java rapidamente, Bookman



Avalie este tópico



ANTERIOR

Encapsulamento



Índice

Biblioteca

(https://www.uninove.br/conheca-

a-

uninove/biblioteca/sobre-

a-

biblioteca/apresentacao/)

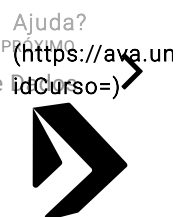
Portal Uninove

(http://www.uninove.br)

Mapa do Site

Conexão com Bases de IdCurso=)

© Todos os direitos reservados



Ajuda?

PRÓXIMO

(https://ava.un

IdCurso=)

