ACH 2003 - Computação Orientada a Objetos

Pacotes

Prof. Flávio Luiz Coutinho
flcoutinho@usp.br

Pacotes

Grupo de classes/interfaces relacionadas, que estão agrupadas em um mesmo diretório. Agrupamento que se dá por algum tipo de afinidade.

Pacotes

Grupo de classes/interfaces relacionadas, que estão agrupadas em um mesmo diretório. Agrupamento que se dá por algum tipo de afinidade.

Motivação: organização do código

Pacotes

Grupo de classes/interfaces relacionadas, que estão agrupadas em um mesmo diretório. Agrupamento que se dá por algum tipo de afinidade.

Motivação: organização do código (um nível a mais)

Pacotes

Organização do código:

- função / métodos
- classes
- arquivos

…

Pacotes

Organização do código:

- função / métodos
- classes
- arquivos

…

É suficiente parar por aqui? Imagine em um projeto com inúmeras classes.

Pacotes

Organização do código:

- função / métodos
- classes
- arquivos
- pacotes
- módulos / bibliotecas
- projetos

Pacotes

O mecanismo disponível na linguagem Java para organizar as classes em pacotes, vai além da mera criação de uma estrutura hierárquica de diretórios/subdiretórios:

Pacotes

O mecanismo disponível na linguagem Java para organizar as classes em pacotes, vai além da mera criação de uma estrutura hierárquica de diretórios/subdiretórios:

- É preciso declarar que um determinado arquivo fonte (que pode declarar uma ou mais classes) faz parte de um pacote.

Pacotes

O mecanismo disponível na linguagem Java para organizar as classes em pacotes, vai além da mera criação de uma estrutura hierárquica de diretórios/subdiretórios:

- É preciso declarar que um determinado arquivo fonte (que pode declarar uma ou mais classes) faz parte de um pacote.
- O caminho definido a partir da raiz do projeto, até o local onde um arquivo fonte se encontra, define o nome do pacote.

Pacotes

O mecanismo disponível na linguagem Java para organizar as classes em pacotes, vai além da mera criação de uma estrutura hierárquica de diretórios/subdiretórios:

- É preciso declarar que um determinado arquivo fonte (que pode declarar uma ou mais classes) faz parte de um pacote.
- O caminho definido a partir da raiz do projeto, até o local onde um arquivo fonte se encontra, define o nome do pacote.
- O nome do pacote passa a fazer parte integrante do nome completo das classes.

Pacotes

Exemplo:

- classes Principal, A e B
- todas no mesmo diretório (fazendo parte do chamado “pacote padrão”)

- atributos com modificador de acesso padrão.

- programa em execução.

Pacotes

Exemplo:

- Criação de uma estrutura de pacotes:
    - meupacote (pacote da classe Principal)
    - meupacote.moduloA (pacote da classe A)
    - meupacote.moduloB (pacote da classe B)

- Estrutura correspondente em disco
- Declarações “package” nos arquivos fonte

Pacotes

Exemplo:

- Compilando as classes, depois de organizadas em pacote:
  - que problemas surgem ao compilar as classes após a reorganização?
  - resolução: nome completo, import da classe, import *

- Ambiguidade de nomes e resolução.

Pacotes

Para compilar e rodar na linha de comando (sem o auxílio de uma IDE, ou de ferramentas para gerenciar a compilação de projetos) a dica é: trabalhe sempre na raiz do projeto.

Toda vez que o javac / java encontrar o nome de uma classe, e precisar encontrar fonte / .class correspondente, o arquivo será procurado a partir do diretório corrente, seguindo o caminho de diretórios correspondente ao nome do pacote.

Mas o diretório corrente não é o único ponto a partir do qual essa procura ocorre. Na realidade podem haver diversos pontos iniciais de procura.

Pacotes

Estes pontos iniciais de procura (na verdade uma lista de diretórios) são definidos (e podem ser redefinidos) na variável de ambiente CLASSPATH, ou especificando um classpath particular na hora de invocar o javac / java.

Exemplo: compilando e rodando nosso projeto a partir de um diretório que não corresponde à raiz do projeto:

```
javac/java -classpath /.../raiz_do_projeto/ /.../Fonte.java
```

Pacotes

Lembrete: implicações quanto à visibilidade dos recursos:

- public todo mundo
- protected classes do mesmo pacote + subclasses
- <default> classes do mesmo pacote
- private apenas a própria classe

A criação de uma estrutura de pacotes adequada pode ser útil para definir um nível de acesso intermediário aos recursos (atributos/métodos) de uma classe (visíveis para algumas classes, mas não visíveis globalmente).