Introdução à Programação
Orientada a Objetos com Java

Pacotes

Paulo Borba
Centro de Informática
Universidade Federal de Pernambuco

Vamos melhorar a reusabilidade e extensibilidade...

Organizando as classes do sistema em **pacotes** que podem ser analisados, reusados e modificados isoladamente ou com o auxílio de outros poucos **pacotes**

Este é o objetivo desta aula

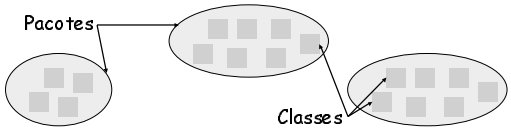
Tipos de **módulos** em Java

- Classes
  - agrupam definições de métodos, atributos e construtores
  - definem tipos
- Pacotes
  - agrupam definições de **classes** relacionadas
  - estruturam sistemas de médio e grande porte

Classes e pacotes

Pacotes facilitam a localização das classes pois oferecem um nível mais alto de abstração:

há mais classes do que pacotes



Pacotes e diretórios

- As classes de um pacote são definidas em arquivos com o mesmo cabeçalho:

```
package nomeDoPacote;
```

- Cada pacote é associado a um diretório do sistema operacional:
  - Os arquivos `.class` das classes do pacote são colocados neste diretório
  - É recomendável que o código fonte das classes do pacote também esteja neste diretório

Nomeando pacotes

- O nome de um pacote é parte do nome do seu diretório associado: o pacote

```
exemplos.banco
```

deve estar no diretório

```
/home/phmb/exemplos/banco
```

assumindo que o compilador Java foi informado para procurar classes em

```
/home/phmb/
```

Pacotes e subdiretórios

- Subdiretórios não correspondem a "subpacotes", são pacotes como outros quaisquer
- Não existe nenhuma relação, além de lógica, entre **`exemplos`** e **`exemplos.banco`**:

```
package exemplos;
public class /*...*/
```

```
package exemplos.banco;
public class /*...*/
```

Pacotes e visibilidade de declarações

- **`public`**
  - atributos, métodos, construtores e classes
  - declaração pode ser utilizada (é visível) em qualquer lugar
- **`private`**
  - atributos, métodos e construtores
  - declaração só pode ser utilizada na classe onde ela é introduzida

Outros modificadores de visibilidade

- **`protected`**
  - atributos, métodos e construtores
  - declaração só pode ser utilizada no pacote onde ela é introduzida, ou nas subclasses da classe onde ela é introduzida
- Ausência de modificador
  - atributos, métodos, construtores e classes
  - declaração só pode ser utilizada no pacote onde ela é introduzida

Reuso de declarações

- As declarações feitas em um arquivo são visíveis em qualquer outro arquivo do mesmo pacote, a menos que elas sejam **privadas**
- Qualquer arquivo de um pacote pode usar as definições **visíveis** de outros pacotes, através do mecanismo de importação de pacotes...

Importação de pacotes

- Importando definição de tipo específica:

```
package segundo.pacote;
import primeiro.pacote.NomeDoTipo;
```

- Importando todas definições de tipo públicas:

```
package segundo.pacote;
import primeiro.pacote.*;
```

Detalhes sobre importação de pacotes

- Tanto **`NomeDoTipo`** quanto **`primeiro.pacote.NomeDoTipo`** podem ser usados no corpo de **`segundo.pacote`**
- **`segundo.pacote`** não pode definir um tipo com nome **`NomeDoTipo`** caso a importação tenha sido específica

Mais detalhes sobre importação de pacotes

A importação de pacotes não é **transitiva** nem **distribui** sobre os arquivos de um pacote

Dicas para estruturar aplicações com pacotes

- Agrupar classes relacionadas, com dependência (de **implementação** ou **conceitual**) entre as mesmas
- Evitar dependência mútua entre pacotes:

```
package a;
import b.*;
/*...*/
```

```
package b;
import a.*;
/*...*/
```

Estruturação típica de um sistema de informação

- Vários pacotes para as classes da interface com o usuário, um para cada conjunto de telas associadas
- Um pacote para a classe fachada e exceções associadas
- Um pacote para cada coleção de negócio, incluindo as classes básicas, coleções de dados, interfaces, e exceções associadas
- Um pacote **`sistema.util`** contendo classes auxiliares de propósito geral

Pacotes da API de Java

- Definição de interfaces gráficas (**`java.awt`**)
- Suporte a objetos distribuídos (**`java.rmi`**)
- Interface com Banco de Dados (**`java.sql`**)
- Básicos: threads e manipulação de strings (**`java.lang`**), arquivos (**`java.io`**), utilitários de propósito geral (**`java.util`**)

Resumindo...

- Importância de estruturar um sistema com pacotes
- Cláusula `package`
- Cláusula `import`, importação específica e importação genérica
- Visibilidade de declarações
- Pacotes da biblioteca de Java