Módulos Físicos

Princípio do encapsulamento

- Tudo o que pode ser privado, deve ser privado!
- Regras gerais
  - Todos os atributos devem ser privados
  - Os construtores são usualmente públicos

As constantes, pelo contrário, podem e muitas vezes devem ser públicas.

Classe é um módulo

- Interface
  - Operações e métodos não privados
  - Constantes não privadas
- Implementação
  - Operações e métodos privados
  - Atributos privados
  - Corpos dos métodos
- Contrato
  - Pré e pós-condições das operações e métodos
- Manual de utilização
  - Comentários de documentação da classe
  - Comentários de documentação de cada característica pública

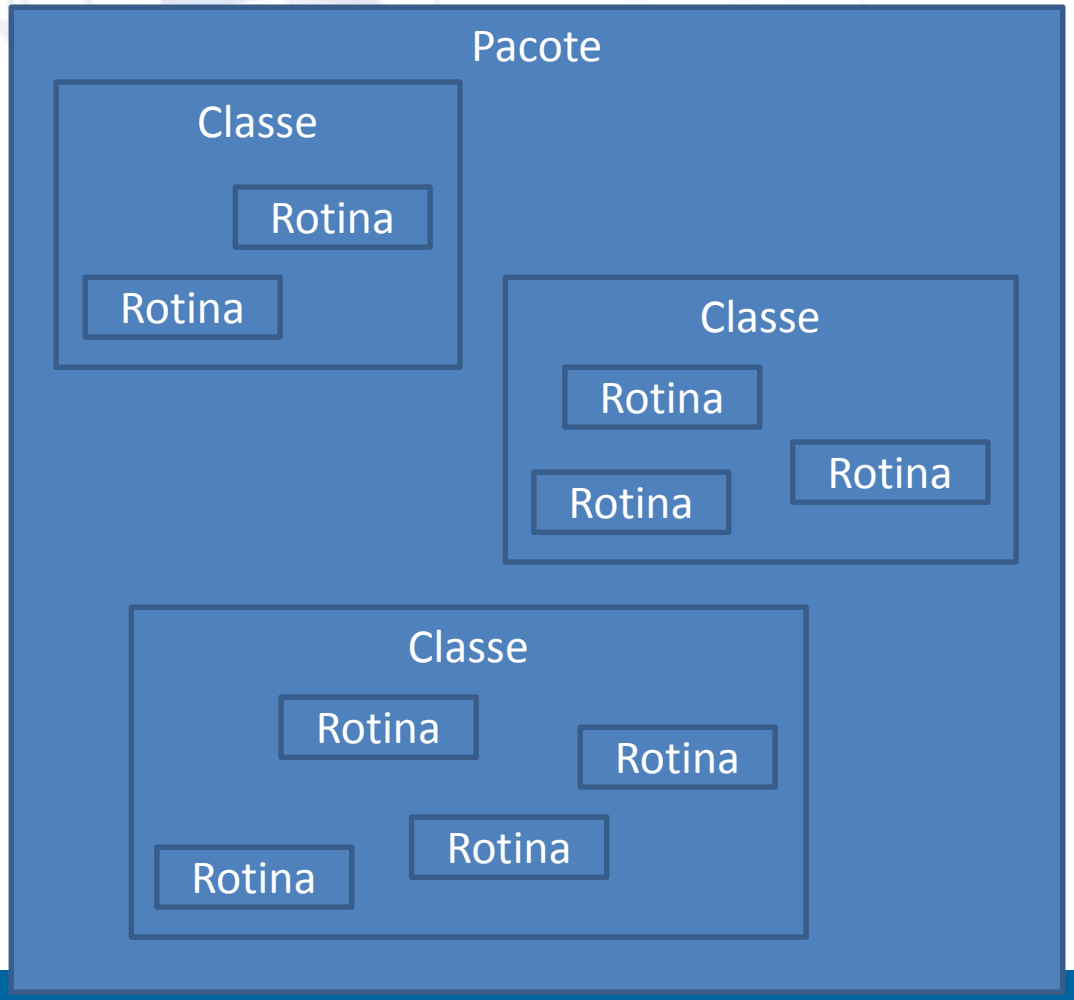
Categorias de acesso

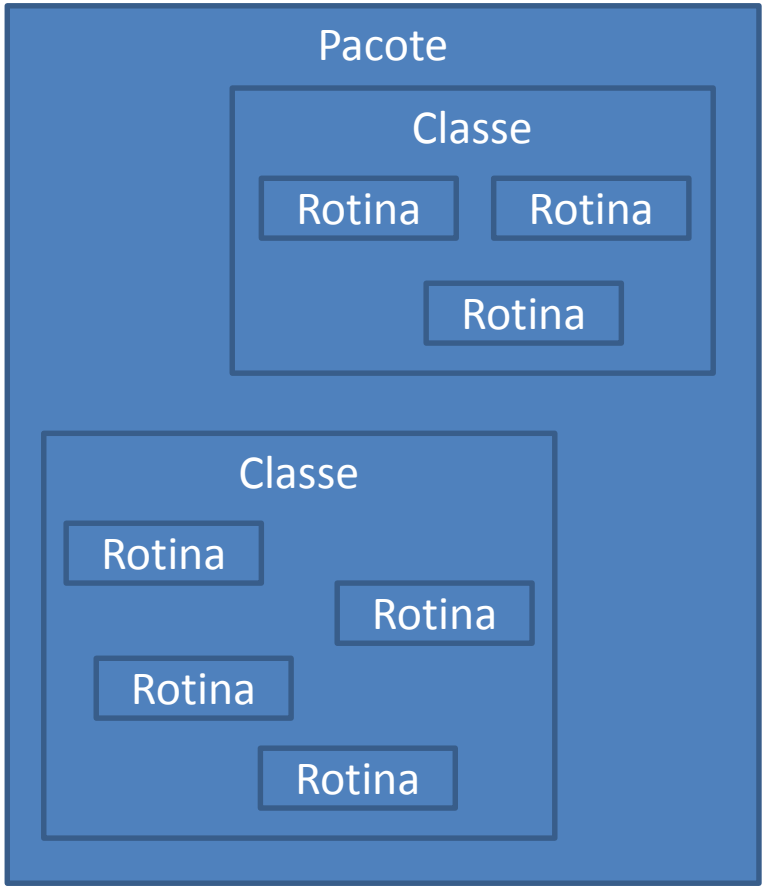
- Características ou membros podem ser
  - `private` – acesso apenas por outros membros da mesma classe
  - *package-private* (sem qualificador) – adicionalmente, acesso por membros de classes do mesmo pacote
  - `protected` – adicionalmente, acesso por membros de classes derivadas (a ver nas próximas aulas)
  - `public` – acesso universal

É possível acesso entre objectos da mesma classe! Cuidado!

Acessibilidade crescente

Unidades de modularização em Java





Pacotes

- Conjuntos de classes com relação lógica forte entre si
- Exemplos
  - `java.util`
  - `org.junit`
- Convenções de nomes
  - Só minúsculas
  - Sem separação entre palavras
  - Abreviaturas e siglas aceitáveis
  - Primeiros elementos são nome DNS invertido (e.g., `pt.iscte`)
  - Restantes elementos podem designar unidades organizacionais (e.g., `pt.iscte.dcti.poo`)

Pacotes como módulos

- Interface
  - Classes públicas
  - Membros não privados de classes públicas
- Implementação
  - Para além da implementação das classes…
  - … todas as classes privadas de pacote (*package-private*)

Pacotes: organização hierárquica

- java
  - lang
  - util
- org
  - junit
  - omg
- pt
  - iscte
    - dcti
      - ip
        - » games
      - poo

Game.java

```
package pt.iscte.dcti.poo

class Game {
    …
}
```

Player.java

```
package pt.iscte.dcti.poo

class Player {
    …
}
```

- Hierarquia aberta
  - Não têm declaração isolada
  - Cada ficheiro `.java` declara o pacote a que pertence

Pacotes: organização hierárquica

- Relevante quanto a nomes
  - Organização lógica (como directórios)
  - Menor colisão de nomes

- Irrelevante quanto a categorias acesso
  - Pacote e subpacote são independentes
  - Membros do subpacote não o são do pacote
  - Membros do pacote não o são do subpacote

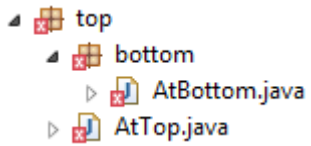
Ficheiros

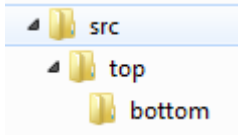
- Uma só classe pública por ficheiro
- Classe pública e seu ficheiro têm de ter o mesmo nome
- Número arbitrário de classes privadas de pacote (*package-private*) por ficheiro, mas…
- Boa prática: Uma só classe por ficheiro!

Directórios

Depende da implementação do Java.

- Usualmente, à hierarquia de pacotes corresponde uma hierarquia de directórios com os ficheiros correspondentes





No Eclipse (pacotes)

No Explorer do Windows (directórios)

Classe vs. Classe pacote

Classe

- Representa um "molde" para a criação de objectos (instâncias da classe)

```
public class Calculator {
   private int value;

   public Calculator() {
      ...
   }

   public int value() {
      ...
   }

   public void clear() {
      ...
   }
   ...
}
```

Classe pacote

- Representa um conjunto de métodos de classe relacionados empacotados num módulo

```
public class CharUtils {
   public static char nextLetter(char c) {
      ...
   }

   public static char previousLetter(char c) {
      ...
   }

   public static int distance(char x, char y) {
      ...
   }
   ...
}
```

Métodos e atributos de classe (`static`)

- A palavra-chave `static` significa que o atributo ou método é da classe e não das instâncias
- Para usar um método de classe basta importar o pacote
- Métodos static não têm acesso aos atributos de um objecto (não `static`)

Método de classe (`static`)

Utilização de método de classe

```
char c = CharUtils.nextLetter('a');
```

diferente dos métodos de instância

```
Calculator myCalculator = new Calculator();

myCalculator.clear();
```

Referências

- Y. Daniel Liang, "Introduction to Java Programming" 7th Ed. Prentice-Hall, 2010.

Sumário

- Módulos Físicos
- Pacotes