Programação Orientada a Objetos

BACHARELADO EM ENGENHARIA DE COMPUTAÇÃO

PROF. LUCIO AGOSTINHO ROCHA

AULA 5: PROGRAMAÇÃO ORIENTADA A OBJETOS

2º.SEMESTRE 2022

Programação Orientada a Objetos

Introdução à Programação Orientada a Objetos

- Programação Orientada a Objetos (POO):
  - Encapsula dados (atributos) e métodos (comportamentos)
  - Objetos se comunicam através de interfaces.
  - Classes são as unidades de programação.
  - Classes encapsulam atributos e métodos.

Introdução à Programação Orientada a Objetos

- Em Java, toda Classe herda de outra Classe:
  - Se não explicitado, implicitamente a Classe herda da Classe java.lang.Object
- Construtor
  - É semelhante a um método, porém tem o mesmo nome da Classe
  - Construtor não possui retorno.
  - É utilizado para inicializar as variáveis de instância
  - Classe pode ter vários construtores (sobrecarga), mas com parâmetros diferentes

Introdução à Programação Orientada a Objetos

- Sobrecarga de Métodos:
  - Por padrão, todo objeto possui um método java.lang.Object.toString que identifica unicamente o objeto instanciado.
  - O método 'toString' pode ser invocado:
    - Implicitamente : System.out.println(objeto);
    - Explicitamente: System.out.println(objeto.toString);

Introdução à Programação Orientada a Objetos

- Escopo da Classe:
  - Variáveis de instância e métodos (chamados de Membros)
  - Membros são acessíveis para todos os métodos da Classe.
- Escopo dos Métodos:
  - Variáveis locais existem apenas dentro dos métodos
  - Variáveis locais não são acessíveis fora do escopo do método.

- Modificadores de Acesso:
  - Muitas vezes, o programador não deseja que todos os atributos e métodos (Membros) de uma classe estejam disponíveis para outros usuários.
  - Java fornece Ocultamento de Informação através de Modificadores de Acesso.
  - Um Modificador de Acesso indica se outras Classes podem ou não utilizar um Membro da Classe.
  - Nota: Modificador de Acesso é para visibilidade externa dos membros. Todos os membros de Classe são visíveis dentro da Classe, independente do modificador de acesso.

- Modificadores de Acesso:
  - Nível Superior (Classe):
    - public, ou private-package (sem modificador)
  - Nível Membro (Variáveis, Métodos ou Classes Internas):
    - public, private, protected, ou private-package (sem modificador)

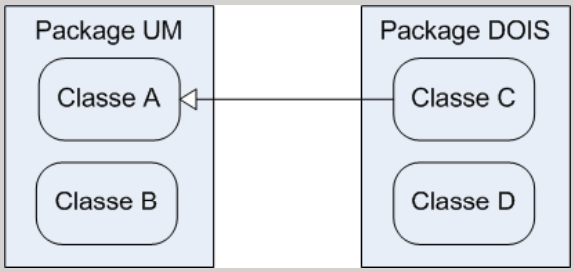
- Modificadores de Acesso: Nível Superior
  - public: classe visível para todas as outras classes de qualquer lugar
  - Sem modificador (package-private): classe visível apenas dentro do seu package.
  - Apenas public, abstract e final são permitidos para a Classe.

- Modificadores de Acesso: Nível Membro
  - public: membro é visível para todas as outras classes de qualquer lugar
  - Sem modificador (package-private (privado de pacote)): membro é visível apenas dentro do seu package.
  - protected: membro é visível apenas por classes dentro do seu package (mesmo que package-private) + visível por classes que herdam dessa classe (subclasses) em outros pacotes.
  - private: membro é acessível por classes apenas dentro do seu pacote.
    - Métodos Acessores: get
    - Métodos Mutadores: set

- Visibilidade dos membros da Classe:

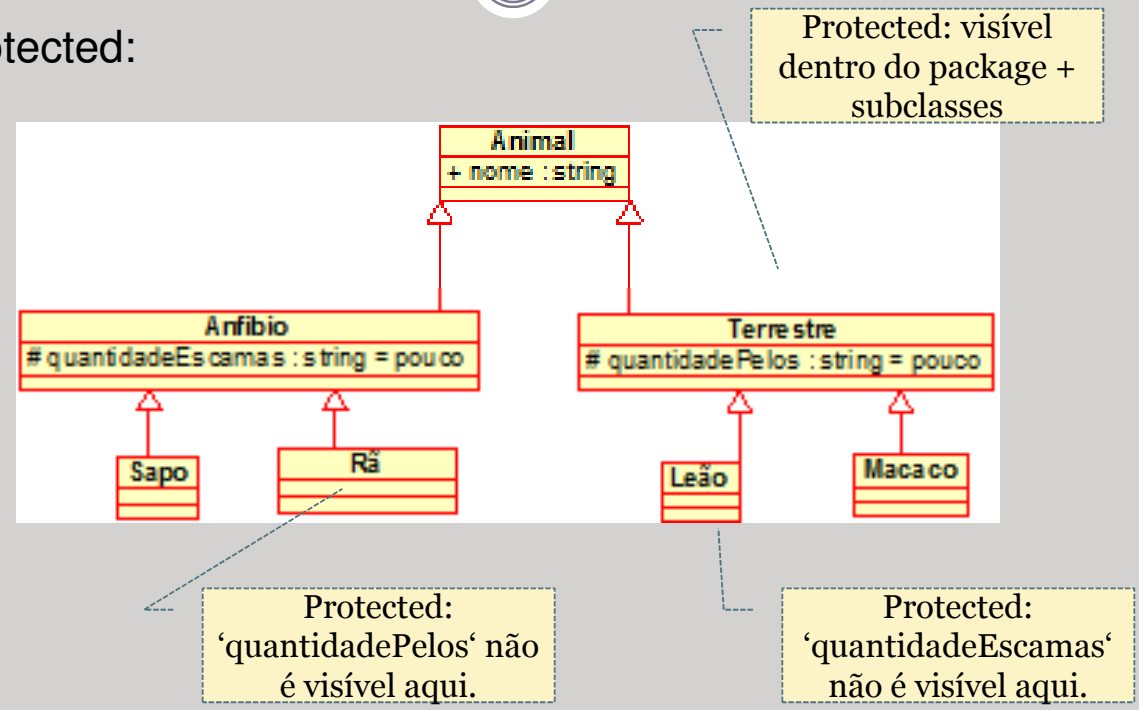
Modificador da Classe	Classe	Package	Subclasse	Fora do Pacote
public	●	●	●	●
protected	●	●	●	○
Sem modificador	●	●	○	○
private	●	○	○	○

- Visibilidade dos membros da Classe A:



Modificador da Classe A	Classe	Package	Subclasse	Fora do Pacote
public	●	●	●	●
protected	●	●	●	○
Sem modificador	●	●	○	○
private	●	○	○	○

- Protected:



- Protected:

```
package abc;

public class ABC {

  protected int abc;

}
```

```
package abc;

public class GHI {

  //Protected: acessível dentro do package
  new ABC().abc = 10;
  //Protected: ou acessivel com método 'get'
}
```

```
package abc;

public class DEF extends ABC{

  System.out.println(abc);

}
```

```
package def;
import abc;
public class DEF extends ABC{

  System.out.println(abc);

}
```

- Referência ‘this’:
  - É uma palavra reservada que referencia o próprio objeto.
  - ‘this’ também é utilizada para acessar os membros da classe (variáveis de instância e métodos)

- Encadeamento:
  - Utiliza a referência ‘this’ para acessar vários métodos em um única chamada.

```
public class ABC {
  ...
  public ABC metodo1( int a){
       return this;
  }
  public ABC  metodo2( int b){
     return this;
  }
}
```

```
public class DEF {
  ...
  ABC  abc = new ABC( );
  abc.metodo1( 123).metodo2( 456 );
}
```

return abc

abc.metodo2(456)

Modificador de Acesso Static

- Static
  - Variável de Classe
  - Método de Classe
    - Existem independente da criação de objetos.

Garbage Collection

Garbage Collection

- Garbage Collection
  - Retornar memória para o sistema
  - Java realiza automaticamente
    - Objetos são marcados para garbage collection se não há referência para eles
- Método Finalizador
  - Retorna recursos para o sistema
  - Método: java.lang.Object.finalize
  - Método não recebe parâmetros
  - Método não retorna valor (void)

Revisão

Revisão

- Programação Orientada a Objetos:
  - É uma forma de modelar o mundo real
  - Atributos: são as propriedades do objeto
  - Comportamentos: são as ações que o objeto realiza
  - Classe: é a unidade básica de programação em Java
    - Implementa métodos (ações) que o objeto realiza.
    - Define atributos: atributos que o objeto deve possuir.

Exercícios

<Ver conteúdo na plataforma de ensino>

Referências

- Referências bibliográficas da disciplina.