PROGRAMAÇÃO ORIENTADA A OBJETOS

POLIMORFISMO

AGENDA

1. Conceito de polimorfismo.
2. Tipo estático *versus* tipo dinâmico.
3. Tipos de polimorfismo.

POLIMORFISMO

- Habilidade pela qual uma única operação pode ser definida em mais de uma classe e assumir implementações diferentes em cada uma dessas classes.

POLIMORFISMO

- Significa **muitas formas**;
- Permite que um único nome de classe ou nome de método represente um código diferente, selecionado por algum mecanismo automático;
- Permite que um único nome expresse muitos comportamentos diferentes.

POLIMORFISMO

- Capacidade de um objeto em decidir que método aplicar a si mesmo, depende de onde ele está na hierarquia de heranças;
- A mesma chamada de método pode, em momentos diferentes, invocar diferentes métodos, dependendo do tipo dinâmico da variável utilizada para fazer esta chamada.

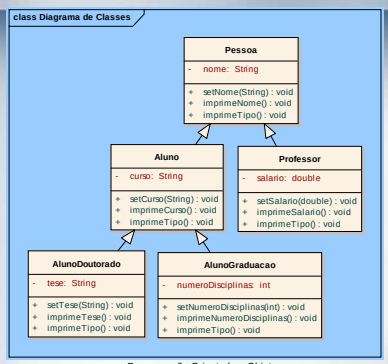
POLIMORFISMO

- Na programação orientada a objetos são enviadas mensagens para objetos, pedindo-lhes que realizem certas ações.

POLIMORFISMO

- Ao enviar uma mensagem que pede para uma subclasse aplicar um método:
  - A subclasse verifica se ela tem ou não um método com esse nome e com exatamente os mesmos parâmetros. Se tiver, usa-o;
  - Caso não, a classe progenitora torna-se responsável pelo processamento da mensagem e procura por um método com esse nome e esses parâmetros. Se encontrar, chama esse método;
  - Esse processo pode continuar subindo a seqüência de heranças;

POLIMORFISMO



POLIMORFISMO

- A chave para fazer o polimorfismo funcionar é a **ligação tardia** (late binding):
  - O compilador não gera o código para chamar um método em tempo de compilação;
  - Cada vez que um método é invocado para um objeto, o interpretador gera código para calcular que método deve ser chamado, usando informações do tipo do objeto;
- Este processo é também chamado de **ligação dinâmica** ou **despacho dinâmico**.

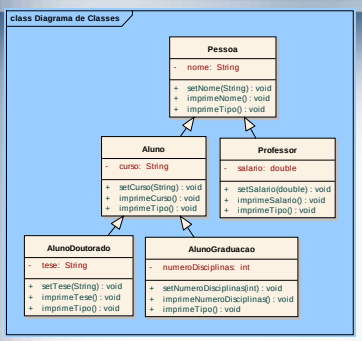
POLIMORFISMO

- O **tipo estático** de uma variável v é o tipo declarado no código fonte na instrução da declaração da variável;
- O **tipo dinâmico** de uma variável v é o tipo de objeto que está atualmente armazenado em v;
- A verificação de tipos utiliza o tipo estático, mas, em tempo de execução, os métodos do tipo dinâmico é que são executados.

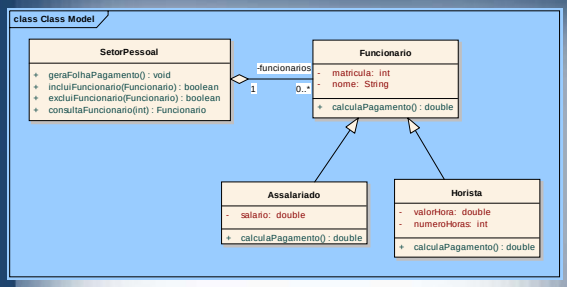
SOBRESCRITA DE MÉTODOS

- Uma subclasse pode sobrescrever a implementação de um método;
- A subclasse declara um método com a mesma assinatura que a superclasse, mas com um corpo diferente;
- O método que sobrescreve tem precedência em relação a chamadas de método nos objetos da subclasse.

SOBRESCRITA DE MÉTODOS



POLIMORFISMO



VARIÁVEIS POLIMÓRFICAS

- Variáveis em Java são chamadas de **variáveis polimórficas:**
  - A mesma variável pode referenciar objetos de seus tipo ou objetos de qualquer de seus subtipos;

MÉTODOS POLIMÓRFICOS

- Métodos em Java são chamados de **métodos polimórficos:**
  - A mesma chamada de método pode, em momentos diferentes, invocar métodos diferentes, dependendo do tipo dinâmico da variável utilizada para fazer a chamada.

TIPOS DE POLIMORFISMO

1. Polimorfismo de inclusão
2. Polimorfismo paramétrico
3. Coerção
4. Sobrecarga

TIPOS DE POLIMORFISMO

POLIMORFISMO DE INCLUSÃO

- Polimorfismo puro;
- Permite que se trate objetos relacionados genericamente;
- Diminui a quantidade de código escrito;
- Facilita a inclusão de novos subtipos;
- Gera novos comportamentos sem ter que alterar código;

POLIMOSFISMO PARAMÉTICO

- Permite a criação de tipos e métodos genéricos;
- **Problema:**
  - Várias operações semelhantes conceitualmente sendo reimplementadas;
  - int add(int a, int b)
  - matriz add(matrix a, matrix b)
- **Solução:**
  - Polimorfismo paramétrico do método
  - [T] add ([T] a, [T] b)
- Apenas a partir da versão 5 do Java este tipo de polimorfismo é permitido através do **Generics**.

SOBRECARGA

- Polimorfismo ad-hoc;
- Permite que se use o mesmo nome de método para muitos métodos diferentes;
- Ex:

public static int max(int a, int b);
public static int max(long a, long b);
public static int max(float a, float b);
public static int max(double a, double b);

- Útil quando o método é definido por um conceito e não por seus parâmetros.

COERSÃO

- A linguagem tem um mapeamento interno entre tipos;
- Forma limitada de polimorfismo;
- Se num particular contexto o tipo requerido é diferente do tipo dado, a linguagem verifica se existe uma coerção (i.e. conversão de tipos).
- Exemplo:
  - Se uma função soma() é definida como tendo 2 parâmetros reais, e um inteiro e um real são passados, o inteiro é convertido para real.