POO

Programação Orientada a Objeto

Programação Estruturada

- Um programa é tipicamente escrito em uma única rotina (ou função) podendo, é claro, ser quebrado em subrotinas. Mas o fluxo do programa continua o mesmo, como se pudéssemos copiar e colar o código das subrotinas diretamente nas rotinas que as chamam, de tal forma que, no final, só haja uma grande rotina que execute todo o programa.

- Além disso, o acesso às variáveis não possuem muitas restrições na programação estruturada.

Programação Orientada a Objetos

- Surgiu como uma alternativa a essas características da programação estruturada. O intuito da sua criação também foi o de aproximar o manuseio das estruturas de um programa ao manuseio das coisas do mundo real, daí o nome "objeto" como uma algo genérico, que pode representar qualquer coisa tangível.

- Esse novo paradigma se baseia principalmente em dois conceitos chave: **classes e objetos.** Todos os outros conceitos, igualmente importantes, são construídos em cima desses dois.

Classe

- Uma classe define os atributos e comportamentos (métodos) comuns compartilhados por um tipo de objeto.
- As classes são usadas para criar ou instanciar objetos.
- Os objetos de certo tipo ou classificação compartilham os mesmos comportamentos e atributos.

Objeto

- Instâncias de classes, que determinam qual informação um objeto contém e como ele pode manipulá-la.

- É uma entidade capaz de reter um estado (informação) e que oferece uma série de informações (comportamento) ou para examinar ou para afetar este estado.

- É através deles que praticamente todo o processamento ocorre em sistemas implementados com linguagens de programação orientada a objetos.

Objeto

- Os objetos podem ser classificados em “Concreto” aquilo que eu posso tocar ou “Abstrato” aquilo que não consigo tocar.
- Exemplo de “Objeto Concreto”, um CARRO.

- Exemplo de “Objeto Abstrato”, uma REUNIÃO.

Abstração

- Consiste em um dos pontos mais importantes dentro de qualquer linguagem Orientada a Objetos.

- Como estamos lidando com uma representação de um objeto real (o que dá nome ao paradigma), temos que imaginar o que esse objeto irá realizar dentro de nosso sistema.

PROPRIEDADES:

- Também chamados de características, forma ou atributo. são as características de uma classe. A altura, o peso, a cor dos olhos ou a cor dos cabelos são exemplos de atributos.

MÉTODOS:

- São os comportamentos, é uma ação executada por um objeto quando passada uma mensagem ou em resposta a uma mudança de estado: é algo que o objeto faz.

Encapsulamento

- É ocultar partes independentes da implementação, permitindo construir partes invisíveis ao mundo exterior. Ficando disponível ao usuário somente a interface.

- Dessa forma, clientes conhecem somente a interface e não dependem da implementação interna.

Exemplo:

Encapsulamento

- Para realizar o encapsulamento são utilizados modificadores de acessibilidade

(+) Public ou Público: Indica que a propriedade ou método, pode ser acessado por qualquer outra classe. Ex.Telefone Público.

(#) Protected ou Protegido: Indica que a propriedade ou método, pode ser acessado pela classe e pelas classes derivadas. Ex. Telefone Fixo da casa.

(-) Private ou Privado: Indica que a propriedade ou método, pode ser acessado apenas pela classe. Ex. Celular.

Herança

- Permite basear uma nova classe na definição de uma outra classe previamente existente. A Herança será aplicada tanto para características quanto para os comportamentos.

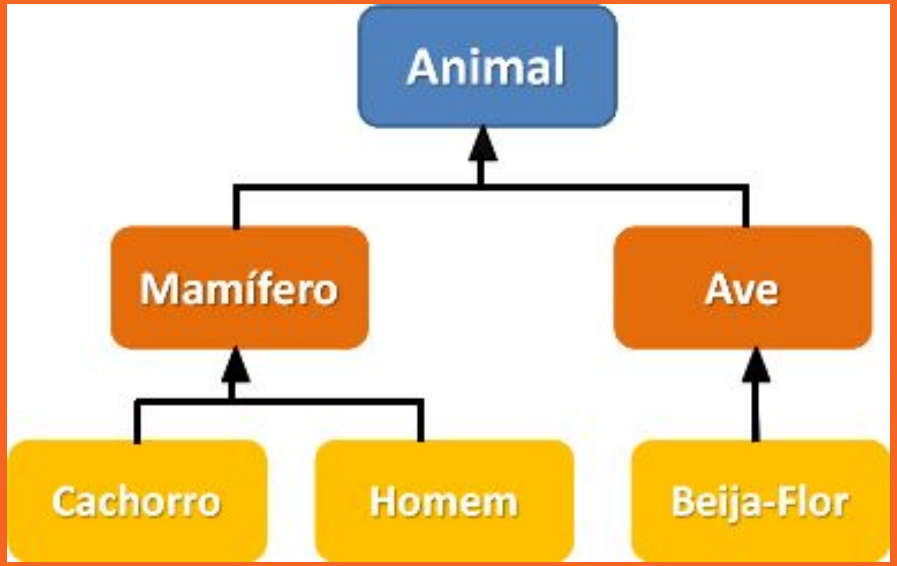
  - Herança de Implementação: É basicamente para implementar uma classe abstrata eu também conhecida como herança pobre, quando uma classe herda todas as características da superclasse não tendo nenhum atributo ou método a mais.

  - Herança para Diferença: É basicamente quando uma classe além de herdar todas as características da superclasse, ela ainda possui atributos particulares.

Herança

SUPER CLASSE

ESPECIALIZAÇÃO



GENERALIZAÇÃO

SUB CLASSE

Pollmorfismo

- É o princípio pelo qual duas ou mais classes derivadas de uma mesma superclasse podem invocar métodos que têm a mesma identificação (assinatura) mas comportamentos distintos, especializados para cada classe derivada, usando para tanto uma referência a um objeto do tipo da superclasse. A decisão sobre qual o método que deve ser selecionado, de acordo com o tipo da classe derivada, é tomada em tempo de execução, através do mecanismo de ligação tardia.

Pollmorfismo

- **SOBREPOSIÇÃO (Override):** Permite reescrever um método, ou seja, podemos reescrever nas Classes Filhas métodos criados inicialmente na Classe Mãe, os métodos que serão sobrepostos, diferentemente dos sobrecarregados, devem possuir o mesmo nome, tipo de retorno e quantidade de parâmetros do método inicial, porém o mesmo será implementado com especificações da classe atual, podendo adicionar um algo a mais ou não.

- **SOBRECARGA (Overload):** Consiste basicamente em criar variações de um mesmo método, ou seja, a criação de dois ou mais métodos com nomes totalmente iguais em uma classe. A Sobrecarga permite que utilizemos o mesmo nome em mais de um método contanto que suas listas de argumentos sejam diferentes para que seja feita a separação dos mesmos.