Lógica de Programação

Unidade 2.1 - Introdução à Programação Orientada a Objetos

QI ESCOLAS E FACULDADES
Curso Técnico em Informática

SUMÁRIO

LINGUAGEM DE PROGRAMAÇÃO

Uma linguagem de programação é um conjunto de comandos e regras de sintaxe, que servem para um programador dar instruções para a máquina, para que ela realize uma tarefa específica.

> *"Podemos imaginar o computador como uma super calculadora, capaz de fazer cálculos muito mais rápido que nós, mas para isso devemos dizer para o computador o que deve ser calculado e como deve ser calculado. A função das linguagens de programação é exatamente essa, ou seja, servir de meio de comunicação entre computadores e humanos." (ANDRADE, 2007)*

Existem dois tipos de linguagens de programação: Linguagens de Baixo Nível e Linguagem de Alto Nível.

Linguagens de Baixo Nível

O computador só entende códigos binários, ou seja, zeros e uns, qualquer coisa a ser executada é codificada por 0 e 1.

As linguagens de baixo nível são mais próximas à máquina (hardware), são linguagens interpretadas diretamente pelo computador, porém é uma linguagem mais difícil e incômoda de se trabalhar.

Exemplos de Linguagens de baixo nível: linguagem binária e linguagem Assembly.

Linguagens de Alto Nível

As linguagens de alto nível são linguagens mais fáceis de se trabalhar e entender. Todas as ações são representadas por comandos em forma de ordem, como: faça, imprima... Geralmente em inglês. São linguagens que facilitam a memorização e a compreensão da lógica.

Exemplos de Linguagens de alto nível: linguagem C, PHP, Java...

PARADIGMAS DA PROGRAMAÇÃO

Um **paradigma** é um padrão, uma maneira, um modelo a ser seguido como referência para fazer algo.

Na programação temos alguns paradigmas que nos auxiliam a escolher a melhor forma de resolver um algoritmo.

Paradigma Imperativo / Estruturado

Este paradigma é o primeiro paradigma a surgir e até hoje dominante. É um paradigma baseado na arquitetura Von Neumann[1]. Segue o conceito de estruturação, onde tudo possui uma sequência passo a passo a ser seguida, modificando os dados a fim de chegar ao resultado esperado.

Exemplos: C, Pascal, Basic...

Paradigma Declarativo / Lógico

Este paradigma caracteriza-se por descrever um problema, sem preocupar-se com o que resolver. A ideia é criar um algoritmo universal.

Exemplos: Prolog, Conniver...

Paradigma Funcional

Este paradigma trata a computação como uma avaliação de funções matemáticas, na prática subdivide o problema em outras funções e resolve cada uma separadamente.

Exemplos: LISP...

Paradigma Orientado a Objetos

Este paradigma trata praticamente tudo como objeto, cada qual com estrutura e comportamento próprio. São classificados em classes.

Exemplos: C#, C++, Java, SmallTalk...

PROGRAMAÇÃO ORIENTADA A OBJETOS

Objetos

Os objetos são a chave para a compreensão da tecnologia orientada a objeto. Olhe em volta agora e você vai encontrar muitos exemplos de objetos do mundo real: seu cachorro, sua mesa, sua televisão.

*Os objetos do mundo real partilham duas características: eles possuem **estado** e **comportamento**. Os cães têm estado (nome, cor, raça) e comportamento (latidos, cheirando, abanando o rabo). Bicicletas também*

[1] John Von Neumann foi um matemático considerado o criador dos computadores da forma como são projetados até hoje.

têm estado (marcha atual, velocidade atual) e comportamento (mudança de velocidade, alterando a marcha, freando). Identificar o estado e o comportamento de objetos do mundo real é uma ótima maneira de começar a pensar em termos de programação orientada a objeto.

Observe agora os objetos que estão ao seu redor. Para cada objeto que você vê, faça duas perguntas: "Que possíveis estados este objeto pode ter?" e "Que possíveis comportamentos que este objeto pode executar?".

Certifique-se de anotar suas observações. Ao fazer isso, você notará que os objetos do mundo real variam em complexidade. Uma lâmpada pode ter apenas dois estados possíveis (ligada ou desligada) e dois comportamentos possíveis (ligar, desligar), mas o rádio pode ter mais estados (ligado, desligado, volume, a estação atual) e comportamento (ligar, desligar, aumentar volume, diminuir volume, procurar, digitalizar, salvar estação).

Traduzido de: Learning the Java Language - Object-Oriented Programming Concepts: What Is an Object? (ORACLE: THE JAVA TUTORIALS)
http://docs.oracle.com/javase/tutorial/java/concepts/object.html

Classes

No mundo real, muitas vezes você vai encontrar objetos que são parecidos, ou que pertencem à mesma espécie. Por exemplo, um cão da raça labrador e um cão da raça *bulldog* são diferentes na aparência, mas ambos são cães. Podemos ter uma bicicleta de corrida, e outra infantil, mas ambas são bicicletas e partilham características comuns.

> *Na programação orientada a objetos, uma classe é um modelo que contém a especificação de um objeto, ou seja, toda a lista de características e ações possíveis desse objeto.*

"*Uma classe para Orientação a Objetos (O.O) é como uma receita de bolo, você não come uma receita de bolo, mas a usa como referência para preparar o bolo. A função da classe é semelhante à da receita do bolo: utilizamos para criar um objeto bolo a partir de suas especificações*" (LUDWIG, 2007). Com a mesma receita, podemos fazer diversos bolos!

Um *software* orientado a objetos funciona através de relacionamentos e trocas de mensagens entre objetos. Este tipo de programação é uma forma especial de programar, mais próxima de como expressaríamos na vida real.

Nos dias de hoje, POO (Programação Orientada a Objetos) vem sendo cada vez mais utilizada, pois a criação de seu código segue uma séria de normas e maneiras de realizar as tarefas, nas quais mais tarde podem ser reaproveitadas em outros algoritmos ou sistemas.

Como utilizar POO

Para utilizarmos programação orientada a objetos temos que pensar em objetos. **Quase tudo é um objeto no sistema**.

Pensamos como faríamos no dia a dia. Imaginemos um carro. O carro é o elemento principal, o qual possui uma série de características, como cor, ano, placa, e funções executadas ou sofridas, como ligar, desligar, acelerar. Para que se fabrique um carro, precisamos de um molde, forma, modelo para determinar como será cada carro fabricado, isso em POO chamamos de **CLASSE**.

> *A **Classe** determina os atributos e métodos dos objetos que serão instanciados (criados).*

Os **atributos** são as características que os objetos terão. Ex.: modelo, marca, ano, cor, etc.

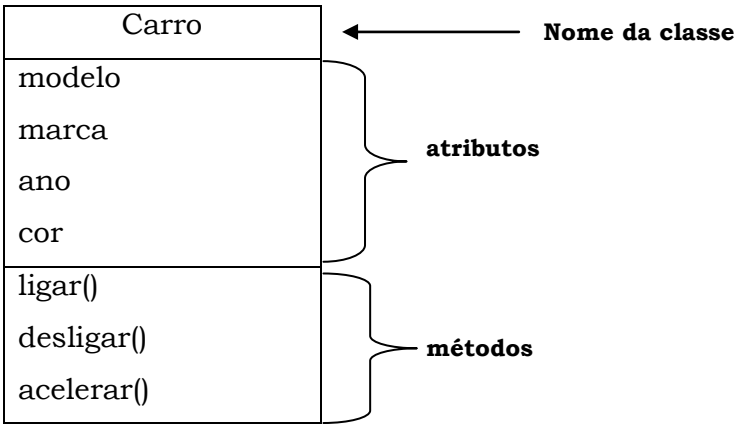
Os **métodos** são as ações que os objetos irão executar ou sofrer. Ex.: A pessoa tem a ação de andar, falar, comer, o carro sofre a ação de ser ligado, desligado, acelerado. Portanto, a pessoa teria os métodos: andar(), falar(), comer(), ligar(), desligar().

> ***Objeto**: representa entidades individuais*
> ***Classe**: representação genérica dos objetos*

Diagramas UML (Unified Modeling Language)

Modelo UML é um modelo universal da linguagem, serve como um projeto, escopo do nosso programa, através dele se representa como serão os objetos do sistema.

Exemplo: uma classe Carro, sendo que todo carro possui um modelo, marca, ano e cor, o mesmo pode ser ligado, desligado, acelerado e freado.



Exemplos de dois objetos criados a partir da classe Carro:

Objeto 1:
Carro modelo: Ka, marca: Ford, ano: 2012, cor: cinza. O Ka pode ser ligado, desligado e acelerado.

Objeto 2:
Carro modelo: Idea, marca: Fiat, ano: 2012, cor: branco. O Idea pode ser ligado, desligado e acelerado.

Atributos

Características dos objetos, dados que estes objetos precisam armazenar. Como exemplo de dados, podemos citar nome, idade, endereço, cor, altura, peso, que são características de pessoas.

Métodos

Ações que nossos objetos poderão executar ou sofrer. Exemplos: andar, falar, ligar, acelerar.

Variável

Espaço alocado na memória para guardar informações temporariamente, que ao longo do uso do sistema pode sofrer alterações.

Exemplo: idade, peso, altura...

Constante

Espaço alocado na memória para guardar informações fixas, que não sofrerão alteração durante o uso do sistema.

Exemplo: Valor do PI na matemática.

PADRONIZAÇÃO DE NOMENCLATURA

No Java para termos uma escrita correta e um padrão de código, seguimos as regras e convenções da escrita.

Regras

As regras são normas que seguimos para que o código execute corretamente. Elas são aplicadas em Classes, atributos, métodos e qualquer variável ou constante que criarmos no programa. Vejamos as principais regras abaixo, baseadas na convenção internacional de código Java:

1. Não utilizar espaço entre as palavras.

Exemplo: ~~numero de filhos~~ → **numeroDeFilhos**

2. Não acentuar as palavras.

Exemplo: ~~salário~~ → **salario**

3. Não iniciar nenhuma palavra com números.

Exemplo: ~~1nome~~ → **nome1**

4. Não utilizar comandos existentes na linguagem.

Exemplo: if, while, for, ArrayList, System, public, private...

5. Não utilizar ç.

Exemplo: ~~valorRefeição~~ → **valorRefeicao**

6. Não utilizar caracteres especiais.

Exemplo: ~~valor&desconto~~ → **valorEDesconto**

Convenções

As convenções de nomenclatura servem para deixar nosso código o mais legível e documentável possível, assim podemos ter um reaproveitamento de código, seguindo o padrão apropriado. Elas são aplicadas em Classes, atributos, métodos e qualquer variável ou constante que criarmos no programa. Vejamos:

Convenções para nomes de CLASSES

Primeira letra de cada palavra em MAIÚSCULA, sempre um substantivo no singular.

Exemplos:

~~pessoa~~ → **Pessoa**

~~contabancaria~~ → **ContaBancaria**

Convenções para nomes de ATRIBUTOS

Primeira palavra toda minúscula e demais palavras com a primeira letra em MAIÚSCULA.

Exemplos:

~~nome~~ → **nome**

~~nomedopai~~ → **nomeDoPai**

Convenções para nomes de MÉTODOS

Primeira palavra toda minúscula e demais palavras com a primeira letra em MAIÚSCULA. E utilizar um verbo na primeira palavra, no infinitivo.

Exemplos:

~~ande()~~ → **andar()**

~~calcularmedia~~ → **calcularMedia()**

Variáveis

Segue a convenção dos atributos. Primeira palavra toda minúscula e demais palavras com a primeira letra em MAIÚSCULA.

Exemplos:

~~Nome~~ → **nome**

~~nomedopai~~ → **nomeDoPai**

Constantes

Toda a palavra em MAIÚSCULA e com *underline* (_) quando for mais de uma palavra.

Exemplos:

~~pi~~ → **PI**

~~paradeexecutar~~ → **PARA_DE_EXECUTAR**

Exemplo de Diagrama UML seguindo regras e convenções:

Modelo
nome endereco idade altura peso nomeDaMae nomeDoPai
desfilar() fotografar() promoverEventos()

REFERÊNCIAS

ANDRADE, Gabriel. **O que são linguagens de programação.** InfoEscola, 2007. Disponível em http://www.infoescola.com/informatica/o-que-sao-linguagens-de-programacao/. Acesso em setembro de 2012.

LUDWIG, Ricardo**. Orientação a Objetos: *Classes***. Blog do Ludwig, 2007. Disponível em http://bloglud.wordpress.com/2007/06/02/orientacao-a-objetos-classes/. Acesso em setembro de 2012.