Hortolândia, 28 de agosto de 2013.

Aulas 17 e 18 – Programação Orientada a Objetos

Introdução

A presente aula apresenta os conceitos que levaram ao surgimento da **programação orientada a objetos**.

Iniciaremos explicando conceitos preliminares de programação, tais como **programas de computador**, **modelos**, **processos** e operações de **abstração**. Tais conceitos são comuns às várias linguagens de programação, porém no caso da programação orientada a objetos, são ainda mais relevantes.

Em seguida, serão apresentados os seguintes conceitos de orientação a objetos: **classes**, **objetos**, **métodos** e **atributos**.

1. O que é um Programa de Computador?

Um **programa** são conjuntos de comandos e regras que um programador deve conhecer para poder manipular os recursos de um computador.

Os **programas** são escritos usando **linguagens de programação**, que definem regras específicas e bem determinadas e um conjunto de operadores e comandos que podem ser usados.

O conteúdo dos programas é chamado de **código-fonte**, sendo que este último é traduzido para linguagem de máquina através de um programa chamado de **compilador**.

Os **programas**, por sua vez, processam **dados**, como por exemplo, valores em uma conta bancária, caracteres entrados por um teclado, pontos em uma imagem, etc.

O **paradigma da orientação a objetos** considera que os **dados** a serem processados e os **mecanismos de processamento** destes dados devem ser considerados em **conjunto**.

Existem pelo menos três estratégias para a compilação e interpretação de programas, que implicam nas seguintes classificações das **linguagens de programação**: **compiladas**, **interpretadas** e **compiladas e interpretadas**.

As linguagens compiladas e interpretadas – como é o caso de Java – produzem, por intermédio do compilador, e a partir do código-fonte, um código intermediário, ainda não otimizado para a máquina em que será executado. Esse código intermediário, por sua vez, é posteriormente interpretado, o que corresponde à sua execução, na máquina a que se destina.

A Figura 1 ao lado ilustra o processo de compilação de programas, com o uso de **código intermediário** – que é a abordagem utilizada pela **Linguagem Java**.

2. O que são modelos?

Modelos são representações simplificadas de objetos pessoais, pessoas, itens, tarefas, processos, conceitos, idéias etc.

Como exemplo de modelo, tomemos o caso de um **Restaurante Caseiro**, que serve refeições por quilo, e onde o gerente é ao mesmo tempo a pessoa que fica na balança e no caixa.

Este restaurante pode ter, por exemplo, uma planilha, controlada pelo gerente, onde são anotados os valores dos pesos dos pratos dos clientes, além de outros itens consumidos por cada mesa.

Esta planilha pode ter o formato descrito na Figura 2 a seguir.

Restaurante Caseiro Hipotético		
Mesa 1 [ ] Kg refeição [ ] Outros	**Mesa 2** [ ] Kg refeição [ ] Outros	**Mesa 3** [ ] Kg refeição [ ] Outros
Mesa 4 [ ] Kg refeição [ ] Outros	**Mesa 5** [ ] Kg refeição [ ] Outros	**Mesa 6** [ ] Kg refeição [ ] Outros

Figura 2: Exemplo de Modelo Simplificado de Restaurante.

A planilha, ou quadro-branco, é um possível modelo deste restaurante, representando de forma simplificada as informações do restaurante que são necessárias para a contabilização dos pedidos feitos para os garçons e o gerente.

O modelo do restaurante representa certos dados ou informações, que no caso são os itens e a quantidade dos pedidos por mesa. Como o modelo é uma simplificação do mundo real, os dados contidos nele são somente os relevantes à abstração do mundo real sendo feita.

Um modelo em geral contém também operações ou procedimentos associados a ele. Essas operações são listas de comandos que processarão os dados contidos no próprio modelo, bem como dados adicionais, caso necessário.

Algumas operações que poderiam fazer parte do modelo hipotético seriam: a inclusão de um determinado pedido por mesa, a modificação do estado de um pedido de uma mesa (servido ou não), o encerramento dos pedidos dos clientes de uma mesa e a apresentação da conta para os clientes.

2.1. Outro exemplo de modelo

Além do exemplo do restaurante, podemos também tomar o caso de uma **lâmpada** incandescente comum. Esta lâmpada pode ser acesa ou apagada, o que determina o seu estado – **ligada** ou **desligada**.

As operações que podem ser realizadas por uma lâmpada são simples: pode-se ligá-la e/ou desligá-la. O ato de ligar a lâmpada corresponde à modificação de seu estado para ligada, enquanto que o ato de desligar significa mudar o seu estado para desligada.

Para saber se uma lâmpada está ligada ou desligada, pode-se pedir que uma operação mostre o valor de seu estado.

De posse dos valores de estado possíveis para a lâmpada, e também das operações que podem ser realizadas por um objeto do tipo lâmpada, pode-se chegar a um modelo simples da mesma, o qual pode ser representado conforme o diagrama da Figura 3.

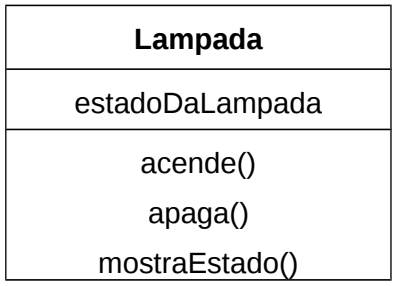

Figura 3: Modelo de uma Lâmpada Incandescente.

3. A abstração e os Processos de Abstração

A **abstração** é o processo utilizado na análise de uma situação real, através do qual se observa uma realidade, tendo-se por objetivo a determinação dos aspectos e fenômenos considerados essenciais, excluindo-se todos os aspectos considerados irrelevantes ou secundários.

O processo de abstração consiste basicamente em identificar, para um determinado objeto ou problema do mundo real, os seus principais aspectos e características, bem como quais são os seus principais comportamentos.

No caso particular da **programação orientada a objetos**, tais abstrações do mundo real são representadas na forma de **classes de objetos**. Uma **classe** de objetos é, portanto, um modelo de determinados objetos do mundo real, os quais possuem características em comum.

No modelo de classes, cada classe é representada por um nome de classe, um conjunto de atributos e um conjunto de métodos (operações) que um objeto desta classe pode executar.

Como exemplos de classes, podem-se citar: a classe Lampada (apresentada na seção anterior), uma classe Automovel, uma classe Estudante, uma classe Matricula, uma classe Funcionario, e assim por diante.

4. Classes de Objetos

Na **programação orientada a objetos**, as **classes** são estruturas das linguagens de programação para conter, para determinado modelo, os dados que devem ser representados e as operações que devem ser efetuadas para esses dados.

As **classes** representam, portanto, modelos de objetos do mundo real. Uma classe corresponde a um programa estático, que ainda não está em execução. Para o uso efetivo das classes, é necessária a instanciação das mesmas, que corresponde à criação de objetos (ou instâncias) destas classes.

Os **objetos** são, portanto, na orientação a objetos, as classes em execução e podem efetivamente representar e manipular dados, o que é feito por meio de seus métodos (ou operações).

A representação das classes segue o padrão definido pela Linguagem Unificada de Modelagem (UML). Uma classe é representada por meio de uma caixa, que pode ter uma, duas ou três divisões. Um exemplo de representação de classe em UML pode ser visto na Figura 4.

A primeira divisão é obrigatória, e deve conter o nome da classe. A segunda divisão contém os atributos da classe, ou seja, os nomes das variáveis que conterão os dados manipuláveis pela classe. A terceira divisão contém os nomes dos métodos, que são as operações que podem ser executadas pela classe.

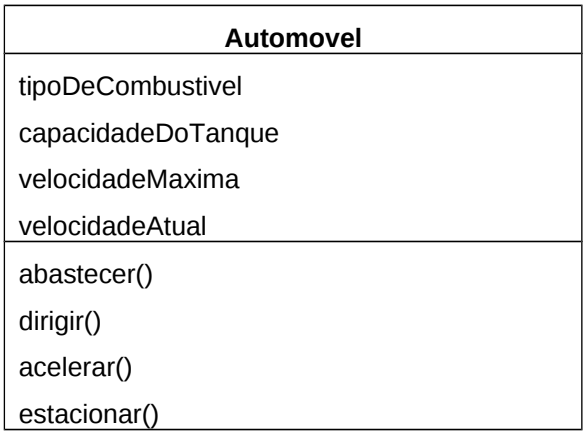

Figura 4: Representação em UML da Classe "**Automovel**".

A mesma classe pode ser representada também suprimindo-se a divisão que contém os atributos, ou a divisão que contém os métodos, ou ambas.

Isto é feito em função de que, em determinadas etapas do projeto de um sistema orientado a objetos, o projetista pode querer representar apenas alguns aspectos de uma classe.

As demais representações possíveis para a classe **Automóvel** são:

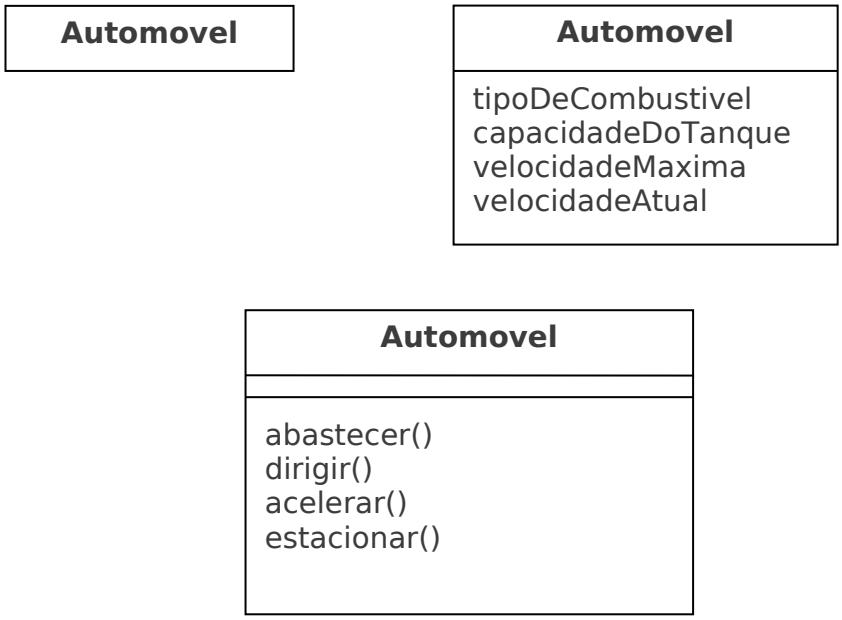

Figura 5: Representações alternativas da classe "**Automovel**".

Esta representação para as classes é independente da linguagem orientada a objetos que se for utilizar.

Os objetos são representados da mesma forma que as classes, apenas na divisão do nome da classe irá aparecer uma indicação de que se trata de uma instância da classe.

Por exemplo, a expressão "**meuCarro:Automovel**" indica que o objeto "**meuCarro**" é uma instância da classe **Automovel**.

5. Conclusões

O processo de abstração, quando aplicado na programação orientada a objetos, leva o desenvolvedor primeiramente a representar um determinado problema computacional a ser resolvido na forma de um modelo de classes de objetos.

Este modelo de classes obtido é então transformado, por meio da atividade de programação, em estruturas de classes, escritas de acordo com as regras de uma determinada linguagem de programação, escolhida pelo desenvolvedor. Tais classes são então compiladas (ou seja, transformadas em código executável) e então executadas pelo computador.

As classes, uma vez instanciadas pelo computador (ou seja, espaço de memória foi alocado para as mesmas), são chamadas de objetos. A representação dos objetos é semelhante à das classes, sendo que a única diferença é que os objetos possuem valores para seus atributos e podem ter seus métodos executados por outros objetos.

Na próxima aula veremos as principais operações de abstração aplicáveis às classes (classificação, instanciação, generalização, agregação e associação).

6. Referências Bibliográficas

BORATTI, Isaias Camilo. **Programação Orientada a Objetos usando Delphi.** Quarta Edição. Editora Visual Books. Florianópolis, 2007.

DEITEL, H.M., DEITEL, P.J. **Java – Como programar**. Terceira edição. Porto Alegre: Bookman Editora, 2001.

SANTOS, Rafael. **Introdução à Programação orientada a objetos usando Java**. 8ª Reimpressão. Rio de Janeiro: Campus - Elsevier, 2003.