Análise e Desenvolvimento de Sistemas

Programação Orientada a Objeto

Aula 1

Introdução a Programação Orientada a Objetos

Prof. Ivan Marcelo Pagnoncelli

Conversa inicial

Nesta aula, vamos iniciar o estudo da Programação Orientada a Objetos (POO), passando primeiramente por uma contextualização do que é e para que utilizamos a mesma e, posteriormente, apresentando a linguagem de programação que iremos utilizar como apoio, a linguagem Java.

Esta aula está dividida desta maneira:

1. O que é e para que serve a Programação Orientada a Objetos;
2. Classe e Objeto: elementos básicos da POO;
3. Utilizando o Java como linguagem de apoio;
4. Utilizando o Eclipse para construir o primeiro programa em POO.

Saiba mais sobre os conteúdos desta aula, assistindo ao vídeo de introdução que o professor Ivan preparou para você! Acesse o material *on-line*!

Contextualizando

A razão por trás da criação da Programação Orientada a Objetos (POO) é a similaridade dos conceitos de interação entre elementos da mesma com a interação que temos no dia a dia, tanto entre nós quanto entre nós e os objetos ao nosso redor.

Como exemplo, podemos citar que, ao utilizarmos nosso carro, sabemos que ele terá uma cor e um modelo, além de uma ação de acelerar e outra de frear. Utilizando esse exemplo, você saberia dizer se o seu carro, relacionando com a Programação Orientada a Objetos, é uma classe ou um objeto.

Entenda melhor essas ideias, assistindo ao vídeo que está disponível no material *on-line*.

O que é e para que serve a Programação Orientada a Objetos

Quando as linguagens de programação surgiram, grande parte delas utilizaram o modelo de programação estruturada ou procedural. Nesse modelo, os programas seguem um fluxo de execução sequencial, podendo também ter, em seu conteúdo, estruturas de decisões e repetições.

A seguir, você pode conferir mais sobre linguagens de programação estruturadas:

http://www.devmedia.com.br/introducao-a-programacao-estruturada/24951

A programação orientada a objetos teve sua origem quando da criação da linguagem de programação Simula 67, nos anos de 1960. Esta foi a primeira linguagem que utilizou os conceitos de **classe** e **objeto**.

Algumas linguagens de programação estruturadas adotaram o conceito de orientação a objeto, tornando-se, assim, linguagens híbridas, que funcionam com os dois métodos de programação. Um exemplo clássico é o C++ e um exemplo mais atual é o PHP.

Ainda nos anos de 1960, nos laboratórios da Xerox foi criada a linguagem de programação Smalltalk. Alan Kay, o líder do projeto criou o termo “programação orientada a objetos”, pois na linguagem Smalltalk, tudo é um objeto. Essas linguagens são chamadas de linguagens de programação puramente orientada a objetos, pois só se pode utilizar o modelo de programação orientada a objetos nela. Outro exemplo deste tipo de linguagem é a linguagem Java, que utilizaremos como apoio.

Alan Kay acreditava que o computador deveria funcionar como um organismo vivo, no qual cada célula funciona de forma independente, relacionando-se com outras células para manter o funcionamento de todo o organismo.

Utilizando essa ideia, Alan Kay começou a pensar em *softwares* como agrupamentos de objetos que se relacionam entre si, através de mensagens, para construir sistemas complexos, e percebeu que esta aproximação se dava

de forma cognitiva, pois no mundo real as pessoas se relacionam com objetos durante todo o tempo.

Conforme a teoria da orientação a objetos foi tomando forma, Alan Kay percebeu que sua utilização tornava o desenvolvimento dos sistemas menos complexos, pois um substantivo faz mais sentido que um verbo na mente das pessoas. Por exemplo, “gatos miam” faz mais sentido do que “miar”, pois une o objeto e a ação ao invés de termos apenas a ação sendo executada.

Com isso, Alan Kay conseguiu atingir a proposta inicial, que era de aproximar os sistemas computacionais do dia a dia dos desenvolvedores, pois utilizando os conceitos mostrados anteriormente, conseguimos modelar um sistema com vários objetos que são semelhantes aos que encontramos no dia a dia.

Aliando a isso alguns paradigmas da programação orientada a objetos, como herança e polimorfismo, temos a capacidade de modelar e construir sistemas computacionais complexos com menos esforço, visto que, assim como no mundo real, objetos computacionais também podem ser reutilizados.

Alan Kay também formulou alguns princípios da programação orientada a objetos:

1. Qualquer coisa é um objeto, o que significa que, quando vamos transportar nosso sistema para um modelo computacional, tudo que for um “objeto” no mundo real será o mesmo em nosso sistema. Aqui cabe uma dica: substantivos normalmente podem ser convertidos em objetos em sistemas;
2. Objetos realizam tarefas a partir da requisição de serviços, ou seja, assim como objetos do mundo real, “o gato que mia”, os objetos computacionais disponibilizam serviços, mensagens ou ações, cujo significado é o mesmo, para que outros objetos possam acessar;
3. Cada objeto pertence a uma determinada classe, ou seja, os objetos são criados a partir de um modelo, ou projeto, chamado classe, que os agrupa. Por exemplo, temos os gatos Persas,

Sphynx, SRD (sem raça definida), que são todos tipos de gato e seguem um modelo de gato;

4. Uma classe agrupa objetos diferentes, como vimos no item anterior, quando temos 3 objetos do tipo gato;
5. Uma classe possui comportamentos associados ao objeto, como o gato miar, por exemplo. A classe, ou o modelo, que irá definir as ações que o objeto irá executar, como no exemplo de miar. Como a ação miar faz parte do tipo **gato**, todos os gatos que pertencem a essa classe miam;
6. Classes são organizadas em hierarquias. Aqui temos o conceito de herança, em que podemos ter especializações da nossa classe definidas. Essas especializações fazem parte de uma hierarquia de classes.

Assim, em comparação com a programação estruturada, a Programação Orientada a Objetos, ou POO, tem algumas vantagens, por exemplo:

1. Facilidade na reutilização de código e, consequentemente, codificação acelerada de sistemas e menos custo de desenvolvimento;
2. Facilidade na construção de sistemas mais complexos, pela integração de classes externas;
3. O projeto do sistema pode ser feito de forma mais abstrata;
4. Tanto o projeto quanto o desenvolvimento podem utilizar o mesmo padrão de construção;
5. Uma maior adequação à estrutura de cliente-servidor.

Mas como nem tudo são flores, também temos desvantagens na aplicação da POO para desenvolvimento de sistemas. Veja algumas delas:

1. Embora o conceito por trás da POO seja baseado em nosso dia a dia, o aprendizado desse modelo se torna mais complexo quando as pessoas aprendem primeiramente o modelo de programação estruturada;

2. Existe a necessidade de um *hardware* mais robusto para a execução de sistemas construídos utilizando a POO;
3. Devido aos sistemas crescerem em complexidade, a elaboração dos mesmos denota um maior esforço para criação do projeto e posterior construção.

Algumas linguagens que utilizam a POO e que são bastante comuns hoje em dia são:

- Java, que será nossa linguagem de apoio para os exemplos práticos;
- Javascript;
- C#;
- C++.

Tire suas dúvidas sobre o assunto, assistindo ao vídeo que está disponível no material *on-line*.

Classe e objeto: elementos básicos da POO

A POO possui dois elementos básicos em sua constituição: a classe e o objeto. Vamos dar uma olhada básica nesses dois conceitos, visto que teremos uma aula específica sobre os mesmos.

A classe

Embora o nome deste método de desenvolvimento de sistemas seja programação orientada a objetos, o elemento básico da mesma é a classe, pois é através dela que os Objetos são criados.

Isso porque, como vimos anteriormente, uma classe é uma representação de um objeto que fará parte do sistema de *software*. Essa representação, ou modelo, tipo, projeto, será utilizada para a criação de objetos semelhantes, ou seja, que possuem características e ações comuns, como os gatos definidos anteriormente.

Isso nos diz que, por ser a classe uma representação, ela não existe “fisicamente”. Ou seja, uma classe não ocupará espaço computacional. Assim sendo, um gato não existe, é um conceito. A classe irá definir as características e as ações que os objetos criados a partir dela possuirão.

Continuando no mundo dos felinos, nossa classe Gato pode ter como características:

- Raça;
- Nome;
- Peso;
- Idade.

E como ações:

- Miar;
- Andar;
- Comer.

Todos os objetos que criarmos baseados na classe Gato terão essas mesmas características e possuirão essas mesmas ações.

Temos ainda, segundo os princípios de POO, a possibilidade de generalizarmos nossas classes criando um sistema hierárquico de classes, ou seja, criando subclasses a partir de nossas classes. Assim poderíamos pensar que temos uma Classe Felinos e a partir dela criamos uma nova hierarquia chamada Gatos. Outras especializações de Felinos poderiam ser Leões, Tigres e Onças.

Objetos

Os objetos serão elementos que irão representar uma determinada classe em um determinado instante, ou, um objeto é uma instância de uma classe.

Podemos ter inúmeros objetos instanciados a partir de uma classe, mas, embora sejam do mesmo tipo, ou sigam o mesmo modelo, eles serão diferentes entre si, como por exemplo, se tivermos dois gatos persas. Eles podem ter as mesmas características, mas serão dois indivíduos diferentes, como as linguagens que implementam essa regra podem ser diferentes.

O conjunto dos valores atribuídos às características de um objeto, em um determinado instante, são considerados o estado deste objeto.

Para se aprofundar mais nessas questões, assista ao vídeo a seguir com as explicações do professor Ivan.

A linguagem Java

Para demonstrar a teoria por trás da POO vamos utilizar uma das linguagens de programação mais utilizadas na atualidade, o Java.

Podemos ver a seguir um *ranking* com as linguagens de programação mais comentadas e, consequentemente, mais utilizadas na atualidade. O Java encontra-se em segundo lugar:

http://www.tecmundo.com.br/programacao/82480-linguagens-programacao-usadas-atualmente-infografico.htm

Como o Javascript é uma linguagem que executa embarcado em uma página HTML, sua utilização fica restrita quando se pensa em utilizá-la para assuntos relacionados à prática da teoria de POO, mas ela implementa todos os conceitos de POO que veremos aqui.

Com relação ao Java, não nos aprofundaremos nas APIs que compõem a linguagem, ou seja, não vamos nos aprofundar na linguagem Java, mas sim utilizá-la para verificação das teorias de POO.

Seguem algumas dicas para utilizarmos o Java para nossa verificação:

1. As classes, no Java, são definidas em arquivos ".java" e só pode haver uma classe pública dentro de um arquivo desses. Por exemplo, para utilizarmos no Java, a classe abaixo deve estar dentro de um arquivo chamado AloMundo.java:

```
public class AloMundo {
        public static void main(String[] args) {
                System.out.println("Alo mundo");
        }
}
```

2. Para a execução de uma classe no Java, devemos criar um método dentro da classe com a seguinte assinatura:

```
public static void main(String args[])
```

Não devemos confundir esse método com os métodos construtores da classe, que veremos mais adiante.

3. O Java possui itens chamados **pacotes** para organizar o código. É uma boa prática de programação utilizar pacotes, ou seja, não criar uma classe em ela estar em um pacote e iniciar o nome dos pacotes com letra minúscula. Para definir que uma classe pertence a determinado pacote no Java utilizamos o seguinte:

```
package laboratório;
```

Essa instrução indica ao Java que a classe que será definida na sequência pertence ao pacote "laboratório".

4. No Java as classes sempre começam com letra maiúscula e, se forem nomes compostos, os outros nomes também iniciarão com letra maiúscula, como podemos ver no item 1.

Lembrando que essa é uma boa prática de programação em linguagem Java, não uma regra da linguagem.

5. Os atributos das classes, na linguagem Java, são escritos em letra minúscula, sem "-" ou "_".

Esta também é uma boa prática de programação e não uma regra da linguagem.

6. Os métodos devem iniciar sempre com letras minúscula e as palavras subsequentes, caso existam, deverão ter a primeira letra maiúscula. Por exemplo, o método abaixo tem duas palavras em sua formação:

```
public void meuMetodo();
```

Outro exemplo é o método exemplificado no item 2.

Criando o primeiro programa Java

Para criar nossas classes de exemplo na linguagem Java vamos utilizar uma IDE, que consiste em uma ferramenta de desenvolvimento, própria para este fim.

A IDE que iremos utilizar é o Eclipse, que pode ser baixado do repositório de programas ou do *site* da ferramenta.

Neste *site* também temos toda a documentação referente à ferramenta, além de vários *plug-ins* para desenvolvimento de soluções mais específicas.

Vamos fazer uso do Eclipse para criarmos nosso primeiro programa em linguagem Java. Lembrando que, como tudo no Java é um objeto, precisamos criar uma classe para esse nosso exemplo.

Esta parte da aula é explicada no vídeo que está disponível no material *on-line*. Não deixe de acessar!

Trocando ideias

Para finalizar este encontro, vamos deixar uma reflexão para você: como vimos, Alan Kay idealizou a programação orientada a objetos para que tivéssemos nos meios computacionais a mesma relação entre objetos que temos em nosso dia a dia. Isso deveria nos bastar para que esse modelo de desenvolvimento seja de fácil e rápida assimilação a todos, pois todos nos relacionamos desta forma todos os dias.

Como então explicar a primeira desvantagem que apresentamos, segundo a qual este modelo é de difícil assimilação a pessoas acostumadas à programação estruturada?

Discuta a questão no fórum desta disciplina, disponível no Ambiente Virtual de Aprendizagem.

Na prática

Imagine a seguinte situação: você foi contratado por uma empresa de *software* que precisa desenvolver um sistema para Pet Shops que deva atender aos seguintes requisitos:

1. Deve ser possível o cadastramento do cliente, com as seguintes informações: nome, telefone e endereço;
2. Deve ser possível o cadastramento do animal, com as seguintes características associadas: nome, nome do tutor, tipo de animal, peso e observações.

Identifique quais as classes que o sistema deve ter, além das características dessas classes.

Síntese

Como vimos neste encontro, a POO foi elaborada com o intuito de aproximar os modelos computacionais ao nosso dia a dia, trazendo para dentro dos sistemas de informação elementos aos quais estamos acostumados, como objetos e seus modelos, as classes.

Apresentamos também a linguagem que será nosso apoio para exemplificar os conceitos e teorias por trás da POO, a linguagem Java, além da ferramenta que utilizaremos para tal.

Assista às considerações finais do professor Ivan no vídeo que está disponível no material *on-line*.

Referências

http://www.hardware.com.br/artigos/programacao-orientada-objetos/
http://www.devmedia.com.br/
https://pt.wikipedia.org/wiki/Orienta%C3%A7%C3%A3o_a_objetos
http://www.webgoal.com.br/origem-da-orientacao-a-objetos/
http://social.technet.microsoft.com/wiki/pt-br/contents/articles/9644.conceitos-de-orientacao-a-objetos.aspx