Conceitos de Classes e Objetos

Rafael Silva Guimarães

Instituto Federal do Espírito Santo

rafaelg@ifes.edu.br
http://rafaelguimaraes.net
https://github.com/rafaelsilvag/ifesJava/

18 de Agosto de 2015

INSTITUTO FEDERAL
ESPÍRITO SANTO

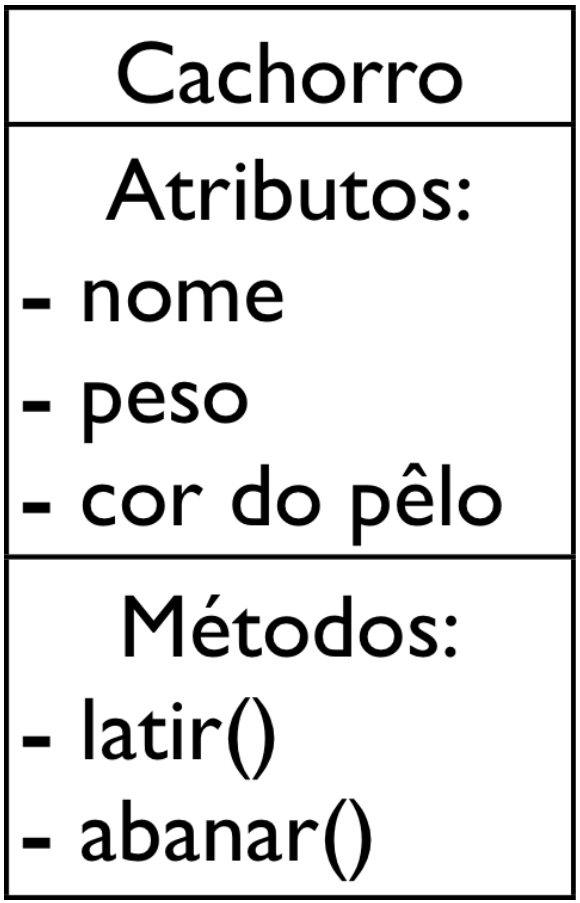
Glossário

INSTITUTO FEDERAL
ESPÍRITO SANTO

Objetos

- Na programação orientada a objetos, objeto é uma abstração dos objetos reais existentes.
- Em uma sala de aula, por exemplo, existem diversos objetos: alunos, cadeiras, mesas etc.
- Os objetos compartilham duas características principais:
  - Todos possuem um **estado**(conjunto de propriedades do objeto);
  - **Comportamentos**(as ações possíveis sobre o objeto);

Objetos


Cachorro
Atributos:
- nome
- peso
- cor do pêlo
Métodos:
- latir()
- abanar()

Objetos

- Na classe apresentada, teremos objetos com os atributis(estados) e métodos(comportamentos) bem definidos;
- Um conceito chamado de **encapsulamento**, é quando os atributos só podem ser alterados utilizando métodos do próprio objeto;
- Para isso aprenderemos que tanto os atributos quanto os métodos teremos quantificadores, que definem as permissões de acesso: **public, private, protected**;
- No Java, os métodos de acesso começam com as palavras **get** e **set**:

INSTITUTO FEDERAL
ESPÍRITO SANTO

Objetos

- Mensagem entre objetos: Em orientação a objetos é muito comum que um objeto necessite realizar uma tarefa que já está definida em outro objeto;
- Em outras palavras, um objeto X pode necessitar de um procedimento(método) já definido em um objeto Y;
  - ```
    boolean enviarEmail(String email){
            // Implementação
    }
    ```
  - ```
    Objeto01 obj = new Objeto01();
    obj.enviarEmail("Teste de envio");
    ```

Classes

- Os objetos são gerados a partir das classes, sendo **instâncias** em memória que mantém seus valores de maneira individual. A criação de uma classe deve anteceder o uso de um objeto, uma vez que este é criado a partir dela.

```
qualificador class NomeDaClasse {
        // Variáveis
        // Métodos
}
```

Classes

- O **qualificador** da classe indica como a classe pode ser usada por outras classes. Neste caso, os qualificadores utilizados são **public** e **private**.
- O qualificador **public** indica que o conteúdo do classe pode ser usada por outras classes do mesmo pacote e de outro pacote. Na prática, **pacotes** são diretórios.
- O qualificador **private** indica que o conteúdo da classe é privado e pode ser usado apenas por classes do mesmo pacote, isto é, por classes localizadas no mesmo diretório. Por padrão o qualificador utilizado é o **private**.

Criação de Classes

- Uma classe é composta basicamente por declaração de variáveis(atributos) e implementação dos métodos.

```
public class Aluno {
        int id;
        String nome;
        String endereco;
        String curso;

        void mostrarDados(){
                // Implementação do método
        }
}
```

Criação de Classes

- Se você não especificar o qualificador por padrão é utilizado no atributo ou método o qualificador **public**;
- As variáveis id, nome, endereço e curso são conhecidas como variáveis de instância, pois serão instânciadas pelos objetos;
- Essa classe não é executável, uma vez que não possui o método **main**. Por este motivo, não é possível executá-la;

Utilização de Objetos

- Conforme vimos anteriormente, uma classe permite criar objetos que serão utilizados em outras classes.

```
public class Principal {
        public static void main(String args[]){
                Aluno joao = new Aluno();

                joao.id = 1;
                joao.nome = "João Pedro";
                joao.endereco = "Rua teste";
                joao.curso = "Sistemas de Informação";
        }
}
```

Pacotes

- No desenvolvimento de pequenas aplicações Java, pode ser viável manter o código em um mesmo diretório. Entretanto, para aplicações maiores, colocar todos os arquivos em um mesmo local, sem organização, pode aumentar a possibilidade de colisão de classes (classes com o mesmo nome no mesmo escopo) e dificultar a localização de um determinado código.
- Em Java, a solução para esses problemas está na organização de classes em pacotes.

Pacotes

- No desenvolvimento de pequenas aplicações Java, pode ser viável manter o código em um mesmo diretório. Entretanto, para aplicações maiores, colocar todos os arquivos em um mesmo local, sem organização, pode aumentar a possibilidade de colisão de classes (classes com o mesmo nome no mesmo escopo) e dificultar a localização de um determinado código.
- Em Java, a solução para esses problemas está na organização de classes em pacotes.

Referências


Cornell, G. ; Horstmann, S. C.

Core Java 2: Fundamentos (vol.1.)

Pearson Makron Books São Paulo


Cornell, G. ; Horstmann, S. C.

Core Java 2: Recursos Avançados (vol.2.)

Pearson Makron Books São Paulo


Sebesta, R.W.

Conceitos de Linguagens de Programação

Bookman São Paulo


Deitel, Paul; Deitel, Harvey

Java - Como Programar

Pearson São Paulo