Departamento de Engenharias e Computação
Colegiado de Ciência da Computação
Disciplina: Linguagem de Programação III

Cap 2 – Conceitos Básicos de Orientação a Objetos

Parte 4 – Métodos e Atributos de Classe Outros Tópicos

Professor: Otacílio José Pereira

Plano de Aula

- ***Objetivos***
  - Compreender a utilidade de métodos e atributos em classes
  - Implementar alguns casos e reconhecer o uso em outros casos

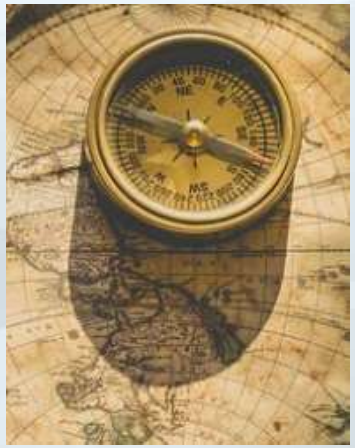
- ***Tópicos***
  - Contexto da disciplina
  - Situação problema
  - Métodos e atributos de classe
  - Reforçando ideia de Classe e Objeto
  - Outros elementos básicos

Contexto

- Onde estamos?
- Foco agora!
- Cenário de exemplo

Onde estamos?

- *Considerando nosso planejamento inicial*



- Capítulo 1 – Introdução
- Capítulo 2 - Conceitos básicos de Orientação a Objetos
  - Parte 1 : Classes, objetos, atributos e métodos
  - Parte 2 : Encapsulamento, modif. de acesso, getters e setters
  - Parte 3 : Construtores
  - Parte 4 : Métodos e Atributos de Classe

Foco

- Capítulo 3 – Herança e Polimorfismo
- Capítulo 4 – Classes abstratas e interfaces
- Capítulo 5 – Generics, collections e outros tópicos
- Capítulo 6 – Desenvolvimento de um projeto em OO

Onde estamos?

- Neste momento que estamos discutindo os conceitos básicos de OO, organizamos nossa trajetória nos seguintes tópicos
  - **(Ok) Parte 1**: Classes, objetos, atributos e métodos e discussões iniciais sobre orientação a objetos
  - **(Ok) Parte 2**: Encapsulamento, modificadores de acesso, getters e setters e sobrecarga de métodos
  - **Parte 3**: Construtores e instanciação de objetos
  - **Parte 4**: Atributos e métodos de classe
  - **Parte 5**: Outros elementos básicos

```
public class Conta {

    private     int Numero;
    private     int Agencia;
    private     String Titular;
    protected   double Saldo = 0.0;

    public static int QtdeContas = 0;
```

Métodos e atributos de Classes

- Situação para análise
- Compreensão
- Exemplos

Situação problema

- Vamos retomar nosso cenário de Instituição Bancária
- Imagine que por algum motivo precisássemos de saber quantas instâncias do objeto conta já foram criadas
  - Por exemplo, para atribuir um novo número a uma Conta
  - Ou para análise da criação de contas

- Neste caso, é interessante guardar a informação em um objeto?
- Como um objeto poderia ter informação de outros objetos?
- Faça uma figura com uma classe e seus vários objetos. Faz sentido guardar a informação nos objetos?

Atributos e métodos de classe

- Os métodos e atributos de classe pertencem à classe
- Eles são criados com o modificador **“static”**
- Eles são acessados através do nome da classe
- Perceba que com isso é possível concluir que as classes acabam tendo seus “espaços de memória” também

```
public class Conta {

    private     int Numero;
    private     int Agencia;
    private     String Titular;
    protected   double Saldo = 0.0;

    public static int QtdeContas = 0;
```

```
Conta()
{
    this.Numero  = 1;
    this.Agencia = 1;
    this.Titular = "Novo cliente";
    this.Saldo   = 0;

    QtdeContas++;
}
```

Exercício

- O que será impresso na última linha do código a seguir?

```
public static void main(String[] args)
{
    Conta c1, c2;

    c1 = new Conta();
    c1.setNumero(10);
    c1.setAgencia(25);
    c1.setTitular("Juvenildo Teobaldo Junior");
    c1.setSaldo(100.0);
    c1.Imprimir();

    c2 = new Conta();
    c2.Imprimir();

    c1 = new Conta(10, 20, "Ota");
    c1.Imprimir();

    System.out.println("Qtde contas : " + Conta.QtdeContas);
}
```

Atributos e métodos de classe

- Até o momento o enfoque foi no atributo da classe
- Caso quiséssemos encapsular o atributo de classe, como faríamos?
- Poderíamos neste caso, usar um método de classe, getter e setter

```
protected   double Saldo = 0.0;

public static int QtdeContas = 0;

public static int getQtdeContas()
{
    return QtdeContas;
}
```

Outros exemplos e concluindo

- Perceba que já atributos e métodos de classe, mas não nos aprofundamos nos detalhes.
  - System.out.print

```
public static void main(String [] args)
{
    Scanner sc = new Scanner(System.in);
    Sensor s_umidade = new Sensor();
```

- Outros exemplos
  - Existe a biblioteca Math
    - Math.PI Math.pow(3,2) Math.sin(10)

```java
public class Conta {

    private     final int Numero;
    private     final int Agencia = 2351;
    private     String Titular;
    protected   double Saldo = 0.0;

    public static int QtdeContas = 0;

    Conta()
    {
        this.Numero  = QtdeContas + 1;
        QtdeContas++;
        // this.Agencia = 1; Não pode ser alter
                             // Valor final
        this.Titular = "Novo cliente";
        this.Saldo   = 0;
    }
```

Outros tópicos

- Final
- Tipo enumerado

Final

- Modificador Final
  - Ele indica que uma vez uma variável inicializada, ela não pode ser mais alterada
  - Exemplos:
    - Perceba que o atributo agência, uma vez inicializado não pode ser alterado no construtor
    - Perceba que número uma vez iniciado não poderá ser alterado em outras partes
    - Outros possíveis campos
      - Data de criação

```
public class Conta {

    private     final int Numero;
    private     final int Agencia = 2351;
    private     String Titular;
    protected   double Saldo = 0.0;

    public static int QtdeContas = 0;

    Conta()
    {
        this.Numero  = QtdeContas + 1;
        QtdeContas++;
        // this.Agencia = 1; Não pode ser alter
                                // Valor final
        this.Titular = "Novo cliente";
        this.Saldo   = 0;
    }
```

- Princípio de projeto
  - O uso de final tem relação com um princípio que diz que é interessante dar o menor nível de privilégio para o programador, neste caso, se o programador se esquecer e tetnar alterar uma variável “final”, o compilador não permite

Tipo Enumerado

- Tipo enumerado
  - Um tipo enumerado funciona como uma lista de valores constantes
  - No nosso exemplo, imagine que fosse necessário estabelecer o status de uma conta e dentre alguns valores válidos (Ativa, Cancelada, Suspensa)

```
public enum StatusConta {
    ATIVA,
    CANCELADA,
    SUSPENSA;
}
```

```
public class Conta {

    private     int Numero;
    private     int Agencia;
    private     String Titular;
    protected   double Saldo = 0.0;

    public static int QtdeContas = 0;

    StatusConta status = StatusConta.ATIVA;
```

Tipo Enumerado

- Alguns comentários
  - Ela é declarada usando a palavra “enum” em vez de “class”
  - Ela pode conter outros elementos de classe, por exemplo atributos e construtores, mas com algumas restrições.
  - Mas daí existem alguns pontos a destacar
    - As constantes são implicitamente static (Veja acesso ao valor)
    - As constantes são implicitamente final

```
public enum StatusConta {
    ATIVA,
    CANCELADA,
    SUSPENSA;
}
```

```
public class Conta {

    private     int Numero;
    private     int Agencia;
    private     String Titular;
    protected   double Saldo = 0.0;

    public static int QtdeContas = 0;

    StatusConta status = StatusConta.ATIVA;
```

```
public enum Book
{
    // declara constantes do tipo enum
    JHTP("Java How to Program", "2015"),
    CHTP("C How to Pr
```

```
// campos de instância
private final String title; // título de livro
private final String copyrightYear; // ano dos direi
// construtor enum
Book(String title, String copyrightYear)
{
   this.title = title;
   this.copyrightYear = copyrightYear;
}
// acessor para título de campo
public String getTitle()
{
   return title;
}
```

Tipo Enumerado

- Esta é uma implementação no livro do Deitel, tópico 8.9.
- Uma reflexão sobre a implementação é que ela mantém um cadastro de livros, e em geral enumerados são usados para constantes.

```
public enum Book
{
   // declara constantes do tipo enum
   JHTP("Java How to Program", "2015"),
   CHTP("C How to Program", "2013"),
   IW3HTP("Internet & World Wide Web How to Program", "20
   CPPHTP("C++ How to Program", "2014"),
   VBHTP("Visual Basic How to Program", "2014"),
   CSHARPHTP("Visual C# How to Program", "2014");

   // campos de instância
   private final String title; // titulo de livro
   private final String copyrightYear; // ano dos direito

   // construtor enum
   Book(String title, String copyrightYear)
   {
      this.title = title;
      this.copyrightYear = copyrightYear;
   }

   // acessor para titulo de campo
   public String getTitle()
   {
      return title;
   }

   // acessor para o campo copyrightYear
   public String getCopyrightYear()
   {
      return copyrightYear;
   }
} // fim do enum Book
```

```
import java.util.EnumSet;

public class EnumTest
{
   public static void main(String[] args)
   {
      System.out.println("All books:");

      // imprime todos os livros em enum Book
      for (Book book : Book.values())
         System.out.printf("%-10s%-45s%s%n", book,
            book.getTitle(),book.getCopyrightYear());

      System.out.printf("%nDisplay a range of enum constants:%n");

      // imprime os primeiros quatro livros
      for (Book book : EnumSet.range(Book.JHTP, Book.CPPHTP))
         System.out.printf("%-10s%-45s%s%n", book,
            book.getTitle(),book.getCopyrightYear());
   }
} // fim da classe EnumTest
```

Tipo Enumerado

```
public enum OpcoesMenu {
SALVAR(1), IMPRMIR(2), ABRIR(3),
VISUALIZAR(4), FECHAR(5);

private final int valor;
OpcoesMenu(int valorOpcao){
valor = valorOpcao;
}
public int getValor(){
return valor;
}
}
```

- Para quem quiser saber mais, esta é uma boa leitura. E veja lá como o enum foi usado para opções de Menu, parece uma solução e uso bem interessante para o tipo enumerado.

- Leitura recomendada:

Tipos Enum no Java

Veja neste artigo como criar enumerações em Java, estruturas de dados que armazenam uma coleção de valores fixos predefinidos e imutáveis. Serão apresentadas as caraterísticas, as declarações e os métodos.

https://www.devmedia.com.br/tipos-enum-no-java/25729

Conclusões

- Retomando Plano de Aula
- Revisão
- Para saber mais

Conquistamos a aula?

- ***Objetivos***
  - Compreender a utilidade de métodos e atributos em classes
  - Implementar alguns casos e reconhecer o uso em outros casos

- ***Tópicos***
  - Contexto da disciplina
  - Situação problema
  - Métodos e atributos de classe
  - Reforçando ideia de Classe e Objeto
  - Outros elementos básicos

Departamento de Engenharias e Computação
Colegiado de Ciência da Computação
Disciplina: Linguagem de Programação III

Cap 2 – Conceitos Básicos de Orientação a Objetos

Parte 4 – Métodos e Atributos de Classe Outros Tópicos

Professor: Otacílio José Pereira