Domain Driven Design - Java

Prof. Gilberto Alexandre das Neves
profgilberto.neves@fiap.com.br

FIAP

Praticando…

Praticando…

Desenvolva os exercícios propostos abaixo, cada um em uma classe diferente em duas versões (uma usando a classe **Scanner** e outra usando a classe **JOptionPane**).

1. Monte um programa que peça para o usuário digitar as notas das 4 provas (prova1, prova2, prova3 e prova4) e exiba a média aritmética simples.

2. Monte um programa que peça para o usuário digitar o ano atual e o ano de seu nascimento exiba ao final a idade deste usuário.

3. Monte um programa que peça para o usuário digitar o valor do raio de um círculo e exiba a área deste círculo (**lembrete**: área do círculo = PI * raio^2)

Praticando

```java
public class Exercicio1Scanner {
    public static void main(String[] args) {
        float p1, p2, p3, p4, media;
        Scanner scan;
        try {
            scan = new Scanner(System.in);
            System.out.print("Digite nota prova1: ");
            p1 = scan.nextFloat();
            System.out.print("Digite nota prova2: ");
            p2 = scan.nextFloat();
            System.out.print("Digite nota prova3: ");
            p3 = scan.nextFloat();
            System.out.print("Digite nota prova4: ");
            p4 = scan.nextFloat();
            media = (p1 + p2 + p3 + p4) / 4;
            System.out.println("Sua média é: " + media);
        } catch (Exception e) {
            System.out.println("Formato incorreto");
        }
    }
}
```

Praticando

```java
public class Exercicio1JOptionPane {
    public static void main(String[] args) {
        float p1, p2, p3, p4, media;
        String auxiliar;
        try {
            auxiliar = JOptionPane.showInputDialog("Digite nota prova1");
            p1 = Float.parseFloat(auxiliar);
            auxiliar = JOptionPane.showInputDialog("Digite nota prova2");
            p2 = Float.parseFloat(auxiliar);
            auxiliar = JOptionPane.showInputDialog("Digite nota prova3");
            p3 = Float.parseFloat(auxiliar);
            auxiliar = JOptionPane.showInputDialog("Digite nota prova4");
            p4 = Float.parseFloat(auxiliar);
            media = (p1 + p2 + p3 + p4) / 4;
            JOptionPane.showMessageDialog(null, "Sua média é: " + media);
        } catch (Exception e) {
            JOptionPane.showMessageDialog(null, "Formato incorreto");
        }
    }
}
```

Praticando

```java
public class Exercicio2Scanner {
    public static void main(String[] args) {
        int anoAtual, anoNascimento, idade;
        Scanner scan;
        try {
            scan = new Scanner(System.in);
            System.out.print("Digite o ano atual: ");
            anoAtual = scan.nextInt();
            System.out.print("Digite o ano de nascimento: ");
            anoNascimento = scan.nextInt();
            idade = anoAtual - anoNascimento;
            System.out.println("Sua idade é " + idade + " anos");
        } catch (Exception e) {
            System.out.println("Formato incorreto");
        }
    }
}
```

Praticando

```java
public class Exercicio2JOptionPane {
    public static void main(String[] args) {
        int anoAtual, anoNascimento, idade;
        String auxiliar;
        try {
            auxiliar = JOptionPane.showInputDialog("Digite o ano atual");
            anoAtual = Integer.parseInt(auxiliar);
            auxiliar = JOptionPane.showInputDialog("Digite o ano de nascimento");
            anoNascimento = Integer.parseInt(auxiliar);
            idade = anoAtual - anoNascimento;
            JOptionPane.showMessageDialog(null, "Sua idade é " + idade + " anos");
        } catch (Exception e) {
            JOptionPane.showMessageDialog(null, "Formato incorreto");
        }
    }
}
```

Praticando

```java
public class Exercicio3Scanner {
    public static void main(String[] args) {
        double raio, area;
        final  double PI = 3.1415;
        Scanner scan;
        try {
            scan = new Scanner(System.in);
            System.out.print("Digite o valor do raio: ");
            raio = scan.nextDouble();
            area = PI * (raio * raio);
            System.out.print("A área do círculo é: " + area);
        } catch (Exception e) {
            System.out.println("Formato incorreto");
        }
    }
}
```

Praticando

```java
public class Exercicio3JOptionPane {
    public static void main(String[] args) {
        double raio, area;
        String auxiliar;
        try {
            auxiliar = JOptionPane.showInputDialog("Digite o valor do raio");
            raio = Double.parseDouble(auxiliar);
            area = Math.PI * Math.pow(raio, 2);
            JOptionPane.showMessageDialog(null, "A área do círculo é: " + area);
        } catch (Exception e) {
            JOptionPane.showMessageDialog(null, "Formato incorreto");
        }
    }
}
```

FIAP

Classes em Java

Classes

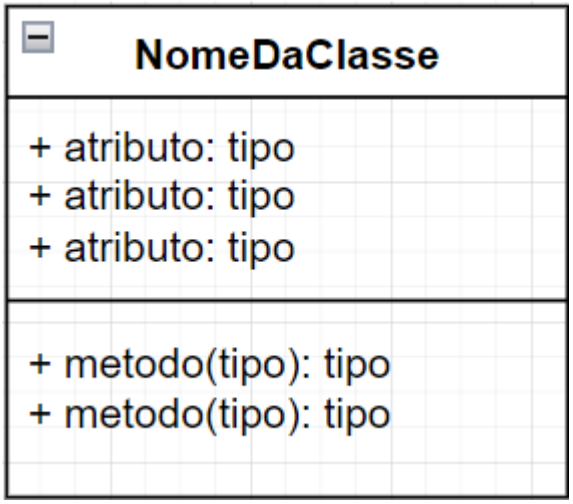
Uma **classe** é um molde, um modelo, um protótipo a partir do qual os objetos podem ser criados. Ao definir uma **classe**, podem ser criados muitos **objetos** a partir dela.

Uma classe é composta por seu **Nome**, seus **atributos** e **métodos**.

- Nomes de **Classes** inicia com letra maiúscula e a cada palavra nova a primeira letra também é maiúscula (não usar espaços ou caracteres especiais).
- Nomes de **atributos** e do **métodos** inicia com letra minúscula e a cada palavra nova a primeira letra é maiúscula (não usar espaços ou caracteres especiais).

Uma classe pode conter:

```
modificador class NomeDaClasse() {
    // atributos
    // métodos
}
```



Modificadores

O **modificador** indica o modo de acesso por outras classes, define sua ***visibilidade***.

Para **Classes**:

- **public** – a classe pode ser acessada por qualquer outra classe.
- *default* – a classe só pode ser acessada por classes no mesmo pacote. Este modificador é utilizado quando não se especifica um modificador.

Para **atributos** e **métodos**

- **public** – o código pode ser acessado por todas as classes.
- **private** – o código só pode ser acessado pela classe que a criou.
- *default* – o código somente é acessado no mesmo pacote. Este modificador é utilizado quando não se especifica um modificador.
- **protected** – o código somente é acessado no mesmo pacote e por ***subclasses***.

Tipos de Atributos (variáveis)

Tipos de Variáveis

Tipos **primitivos** da linguagem, utilizados na criação de variáveis.

Tipo	Quantidade de bits	Valores
char	16	'\u0000' a '\uFFFF'
byte	8	-128 a + 127
int	32	-2.147.483.648 a +2.147.483.647
short	16	-32.768 a + 32.767
long	64	-9.223.372.036.854.775.808 a +9.223.372.036.854.775.807
float	32	-3.40292347E+38 a +3.40292347E+38
double	64	-1.79769313486231570E+308 a +1.79769313486231570E+308
boolean	8	true ou false

Tipos **não primitivos**:

Tipo	Quantidade de bits	Valores
String	??	cadeia de caracteres (usar aspas)

Operadores Aritméticos

Operadores aritméticos

Veja a seguir os operadores aritméticos usados na linguagem Java.

Função	Sinal	Exemplo
Adição	+	x + y
Subtração	-	x – y
Multiplicação	*	x * y
Divisão	/	x / y
Resto da divisão inteira	%	x % y
Sinal negativo	-	–x
Sinal positivo	+	+x
Incremento unitário	++	x++
Decremento unitário	--	x--

FIAP

Métodos

Métodos em Java

Um **método** é um bloco de código que só é executado quando é "chamado" (usado).

Você pode ter métodos onde se é possível passar dados (valores), chamados de parâmetros.

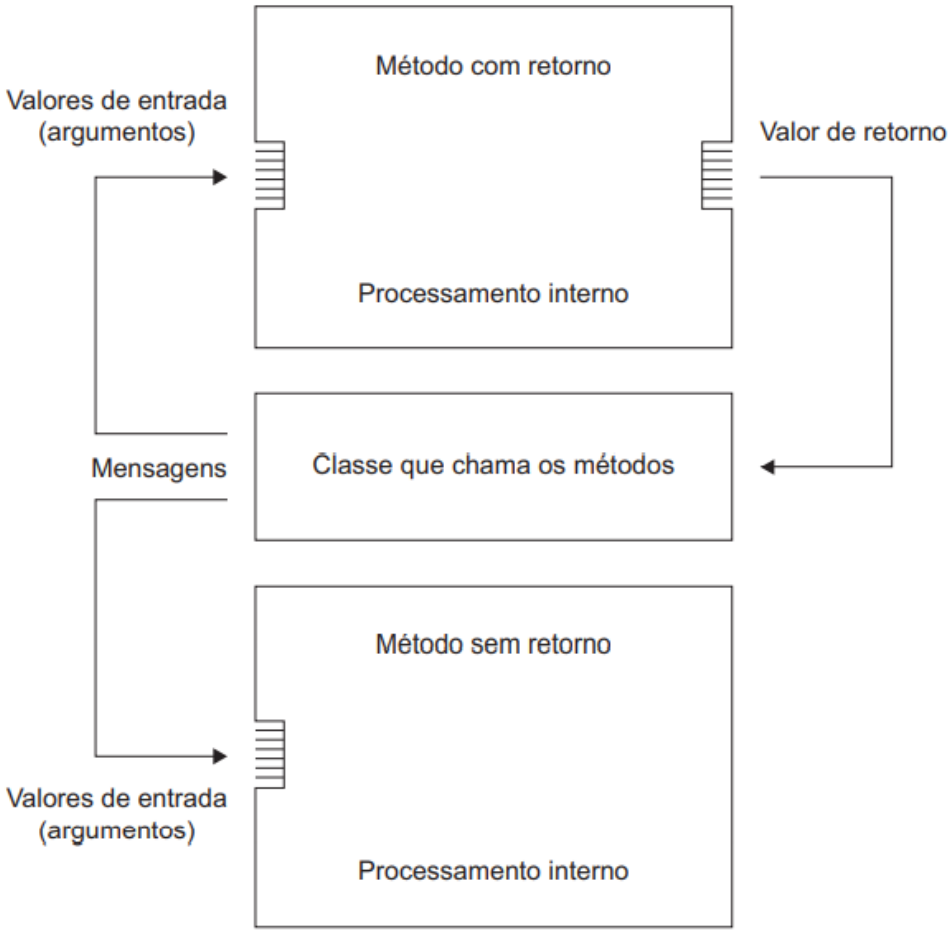
Em outras linguagens os métodos podem ser chamados de funções ou até mesmo procedimentos.

Porque usar métodos?

Para reutilizar o código (defina o código apenas uma vez e use-o várias vezes)

Métodos em Java

Podemos construir **métodos com retorno** ou **métodos sem retorno** (tudo vai depender da situação problema a ser resolvida). Veja a imagem abaixo para tentar entender a diferença entre eles:



Sintaxe de um Método

Veja a seguir a sintaxe para criar um método (com retorno ou sem):

Método com retorno

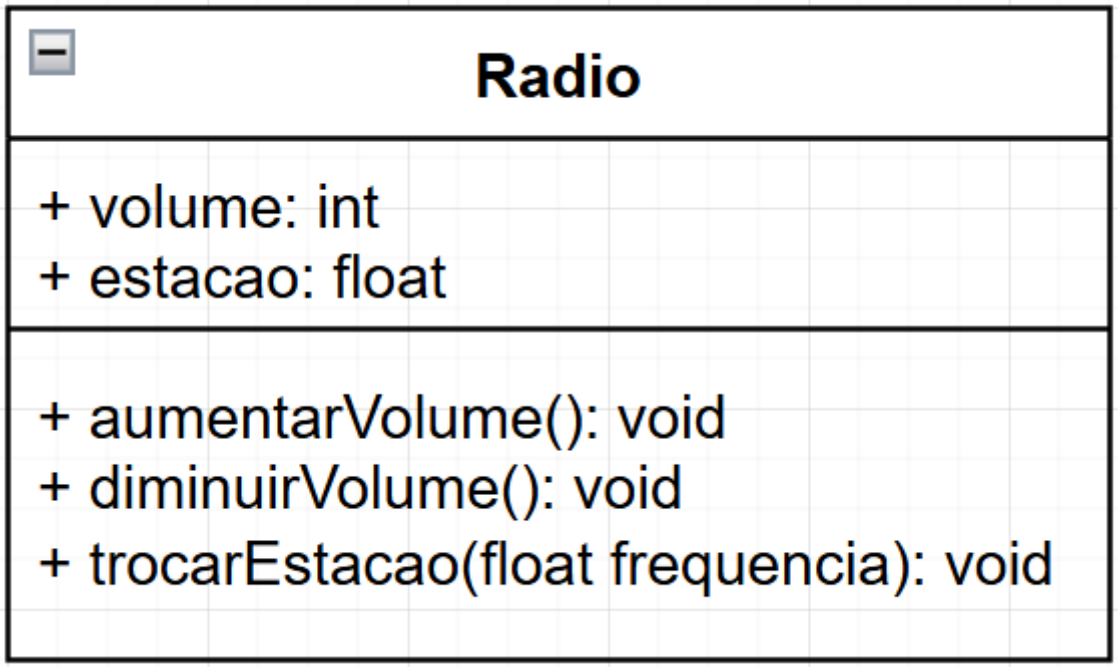
```
modificador tipo nomeDoMetodo(lista-de-parâmetros) {
    código do corpo
    return valor-de-retorno;
}
```

Método sem retorno

```
modificador void nomeDoMetodo(lista-de-parâmetros) {
    código do corpo
}
```

A classe Radio

Veja a representação da classe **Radio** de acordo com a **UML** (Unified Modeling Language - uma linguagem de modelagem muito usada no mercado para representar certas características do software).



A classe Radio

Codifique a classe Radio como indicado abaixo:

```java
package br.com.fiap;

public class Radio {
    // atributos
    public int volume;
    public float estacao;

    // métodos
    public void aumentarVolume() {
        volume++;
    }
    public void diminuirVolume() {
        volume--;
    }
    public void trocarEstacao(float frequencia) {
        estacao = frequencia;
    }
}
```

Utilizando objetos da classe

Objetos da classe

Para utilizar um objeto de uma classe existem devemos efetuar:

- **A declaração do objeto**: segue o mesmo padrão de declaração das variáveis. Sua sintaxe é: `NomeDaClasse nomeDoObjeto;`

- **Instanciação do objeto**: corresponde à criação do objeto pela alocação de memória para armazenar informações sobre ele. Para realizar a instanciação do objeto utilizamos o operador **new** seguido do **nome da Classe( )**.

Exemplo1:

```
Radio radio1;
radio1 = new Radio();
```

Exemplo2:

```
Radio radio2 = new Radio();
```

A classe UsaRadio

Implemente a classe abaixo com o método **`main`** para testar nossa classe **`Radio`**.

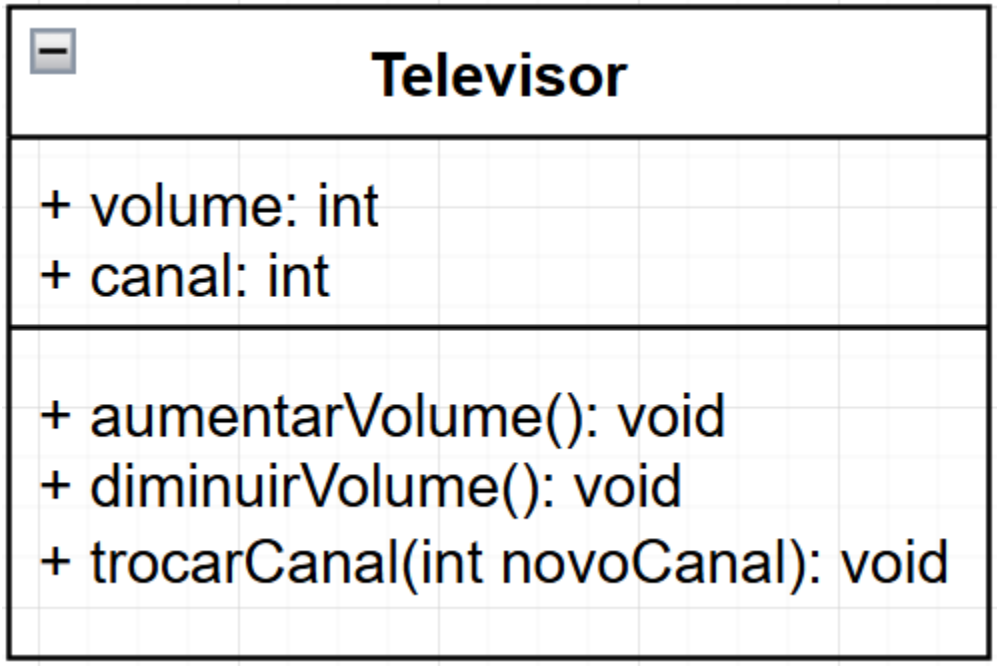
```java
package br.com.fiap;

public class UsaRadio {
    public static void main(String[] args) {
        Radio radio1 = new Radio();
        radio1.estacao = 89.1f;
        radio1.volume = 5;
        radio1.trocarEstacao(92.5f);
        radio1.aumentarVolume();
        radio1.aumentarVolume();
        System.out.println("Volume atual: " + radio1.volume +
         "\nEstação atual: " + radio1.estacao);
    }
}
```

Praticando…

Praticando

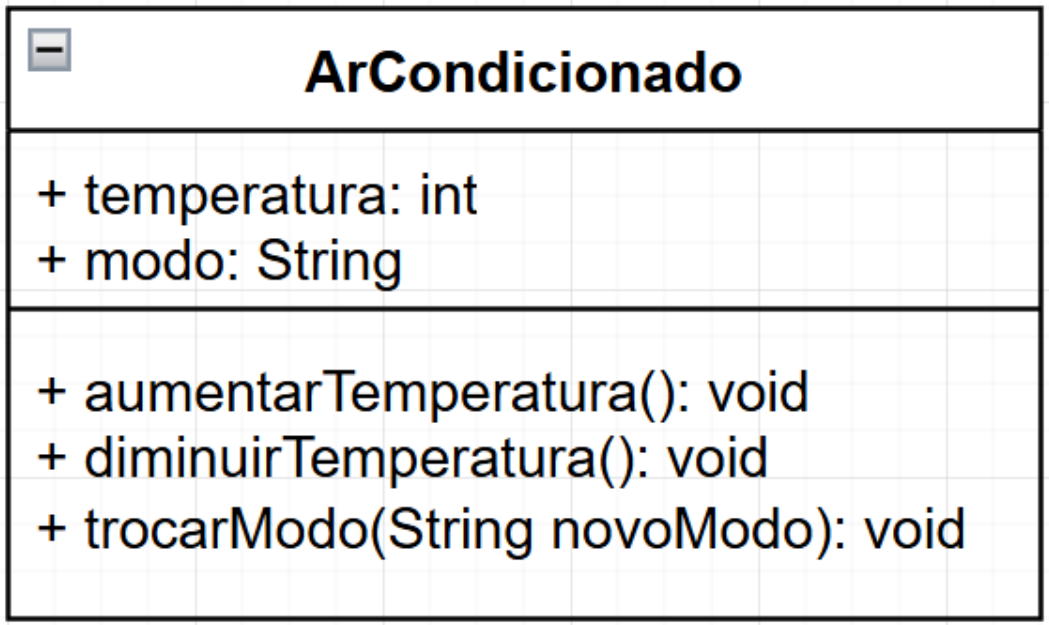
Implemente a classe abaixo:



Agora crie uma nova classe (**`UsaTelevisor`**) com o método **`main`** para criar objetos e testar objetos da classe **`Televisor`**. Ao final exiba o volume e canal atuais.

Praticando 2

Implemente a classe abaixo:



Agora crie uma nova classe (**`UsaArCondicionado`**) com o método **`main`** para criar objetos e testar a classe **`ArCondicionado`**. Ao final exiba a temperatura e o modo atuais.

Como valores para o atributo modo, utilize: “resfriar”, “aquecer” ou “ventilar”.

Praticando 3

Use a criatividade e represente no modelo UML uma classe a sua escolha, defina para essa classe no mínimo 3 atributos e no mínimo 2 métodos.

Agora implemente o código desta classe no Java.

Em seguida, crie outra classe para criar um objeto da classe implementada anteriormente e modifique os valores de seus atributos, utilize seus métodos e exiba resultados.

Questionário

1. Uma classe é composta por:
   a) Nome, atributos e funções
   b) Nome, procedimentos e métodos
   c) Nome, atributos e métodos
   d) Nome, comportamento e funções

2. Para instanciar um objeto de uma classe utilizamos o operador:
   a) private
   b) new
   c) throw
   d) public

Referências

Java como programar. Paul Deitel e Harvey Deitel. Pearson, 2011.

Java 8 – Ensino Didático : Desenvolvimento e Implementação de Aplicações. Sérgio Furgeri. Editora Érica, 2015.

Até breve!

Questionário - Resolução

1. Uma classe é composta por:
   a) Nome, atributos e funções
   b) Nome, procedimentos e métodos
   c) Nome, atributos e métodos
   d) Nome, comportamento e funções

2. Para instanciar um objeto de uma classe utilizamos o operador:
   a) private
   b) new
   c) throw
   d) public