POO
Programação Orientada a Objetos

Material 005

Professor Maromo

Agenda

- Construtores
- Sobrecarga
- Variável this
- Classe Math
- Comando package / Pacotes da API do Java
- Enumeradores
- Exercícios / Desafio

Construtores

Quando usamos a palavra reservada **new**, estamos **construindo um objeto**.

O **construtor é um bloco que possui o mesmo nome da classe**.

É importante **observar que quando não criamos explicitamente um construtor o compilador o fará automaticamente.**

Toda classe necessita de pelo menos um construtor (**seja implícito ou não**).

Pode-se ter mais do que um método construtor (overload), com assinaturas diferentes.

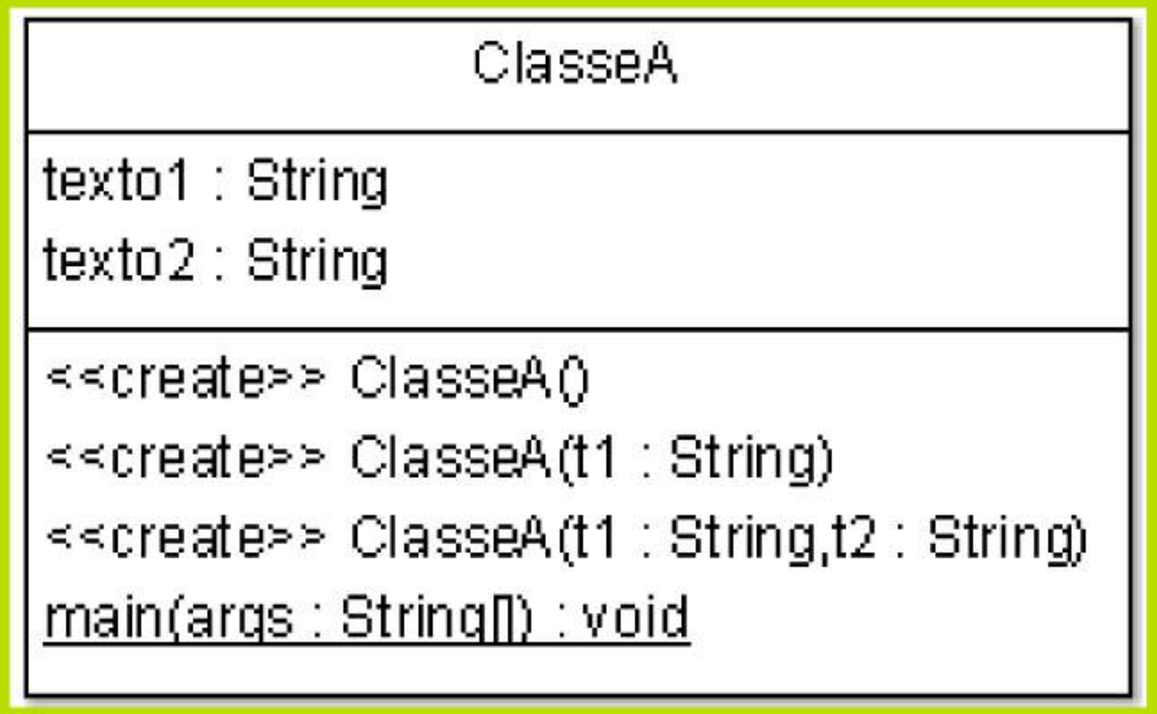
Sobrecarga (overload)

É a capacidade de definir mais de um método com o mesmo nome. No entanto sua assinatura deve ser diferente. Ex:

```
public class ExemploOverload {
    void imprimir(){
        System.out.println("Método imprimir - void");
    }
    int imprimir(int a){
        int num=a;
        return num;
    }
    int imprimir(int a, int b){
        return a+b;
    }
    public static void main(String[] args) {
        //Exemplo de chamada de métodos sobrecarregados.
        ExemploOverload obj = new ExemploOverload();
        System.out.println("Conceituando Overload (sobrecarga)");
        int a = 3;
        int b = 7;
        obj.imprimir();
        System.out.println("Valor de A: " + obj.imprimir(a));
        System.out.println("Valor da soma de A e B: " + obj.imprimir(a, b));
    }
```

Sobrecarga de Métodos Construtores

- Considere a classe ClasseA abaixo:



```java
import java.util.Scanner;
public class ClasseA {
    //Membros públicos
    public String texto1;
    public String texto2;
    //Construtor 1
    public ClasseA(){
        texto1 = "Primeiro Texto\n";
        texto2 = "Segundo Texto\n";
    }
    //Construtor 2 (overload)
    public ClasseA(String t1){
        texto1 = t1;
        texto2 = "";
    }
    //Construtor 3 (overload)
    public ClasseA(String t1, String t2){
        texto1 = t1;
        texto2 = t2;
    }
```

ClasseA

Método main

```java
public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    System.out.println("Digite o número de parâmetros [0,1,2]: ");
    int qtd = sc.nextInt();
    ClasseA obj=null;
    if (qtd==0){
        obj = new ClasseA();
    }else if(qtd==1) {
        String t;
        sc = new Scanner(System.in);
        System.out.printf("Digite um texto: \n");
        t = sc.nextLine();
        obj = new ClasseA(t);
    }else if(qtd==2){
        System.out.println("Digite dois textos: ");
        sc = new Scanner(System.in);
        String t = sc.nextLine();
        String t1 = sc.nextLine();
        obj = new ClasseA(t,t1);
    }
    System.out.println("Texto 1: " + obj.texto1);
    System.out.println("Texto 2: " + obj.texto2);
}
```

Nota sobre o exemplo anterior

- A ClasseA possui três métodos construtores 1 + (2 Overload).
- Quando um método sobrecarregado é chamado, o compilador Java seleciona o método adequado examinando a assinatura do mesmo, na chamada.

Variável this

É uma referência ao próprio objeto, usa-se em:

- Dentro de um construtor, podemos executar outro construtor com assinatura diferente.
- Resolver ambiguidade de nome entre um atributo e um parâmetro ou variável (caso mais comum de uso).
- Retornar a própria referência da instância em algum método.

Comando package

- Representa um grupo de elementos Java, como classes, interfaces, enumeradores e anotações.
- Quando agrupados permitem uma melhor organização dos inúmeros programas que podemos manipular.
- Regra de nomenclatura: usar letras minúsculas.

Comando import

- Comando import
  - Permite que um programa Java use elementos pertencentes a um pacote já existente.
  - Usado em conjunto com a palavra reservada static, ele pode ser usado sem o qualificador, ou seja, sem o nome da classe onde foi definido.
- Ex:

Exemplo: comando import
Programa: ExemploImport

```
import static java.lang.Math.*;
import static java.lang.System.*;
/**
 * Uso do método estático sqrt
 * para exibir a raiz quadrada da constante PI
 *
 */
public class Modulo03ExemploImport {

    public static void main(String[] args) {
        out.println(sqrt(PI));
    }
}
```

Classe Math

Possui vários métodos estáticos (**static)**, que permitem realizar cálculos matemáticos comuns.

Faz parte do pacote **java.lang**, que é implicitamente importado pelo compilador. Não é necessário importar a **classe Math** para utilizar os seus métodos.

Métodos

Método	Descrição	Exemplo
abs	Retorna o valor absoluto de um número.	Math.abs(-10) retorna 10
ceil	Arredonda o número para o menor inteiro maior ou igual.	Math.ceil(10.2) retorna 11.0
floor	Arredonda o número para o maior inteiro menor ou igual.	Math.floor(10.8) retorna 10.0
round	Arredonda o número para o inteiro mais próximo.	Math.round(10.5) retorna 11
sqrt	Retorna a raiz quadrada de um número.	Math.sqrt(16) retorna 4.0
pow	Eleva um número a potência de outro número.	Math.pow(2, 3) retorna 8.0
max	Retorna o maior de dois números.	Math.max(5, 10) retorna 10
min	Retorna o menor de dois números.	Math.min(5, 10) retorna 5
random	Retorna um número aleatório entre 0 (inclusivo) e 1.	Math.random() retorna um valor entre 0.0 e 1.0

Pacotes da API do JAVA

Os pacotes de API em Java referem-se ao conjunto de bibliotecas padrão fornecidas pela Oracle para ajudar os desenvolvedores a criar aplicações Java. Estas bibliotecas contêm classes, interfaces e enumerações que fornecem funcionalidades básicas e avançadas para desenvolvimento em Java. Algumas delas:

java.lang, java.util, java.io, java.net, java.awt, java.sql, javax.Swing, java.Math, javax.servlet e javax.jsp, java.time

Constantes Math.PI e Math.E

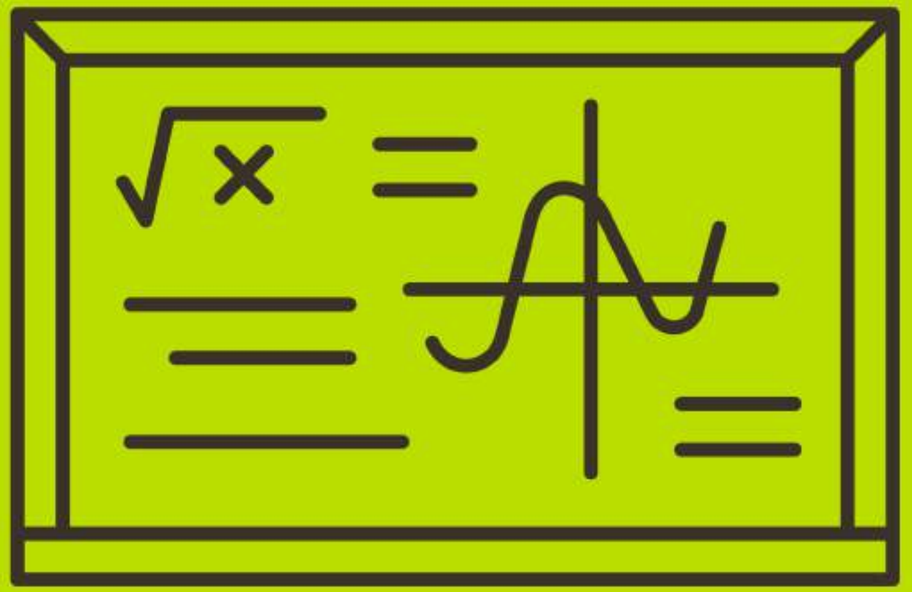
Constantes matemáticas

π

Math.PI
3,14159...

e

Math.E
2,718281...



Constante PI é a relação entre a circunferência de um círculo e o seu diâmetro.
Constante E é o valor da base para logaritmos naturais.

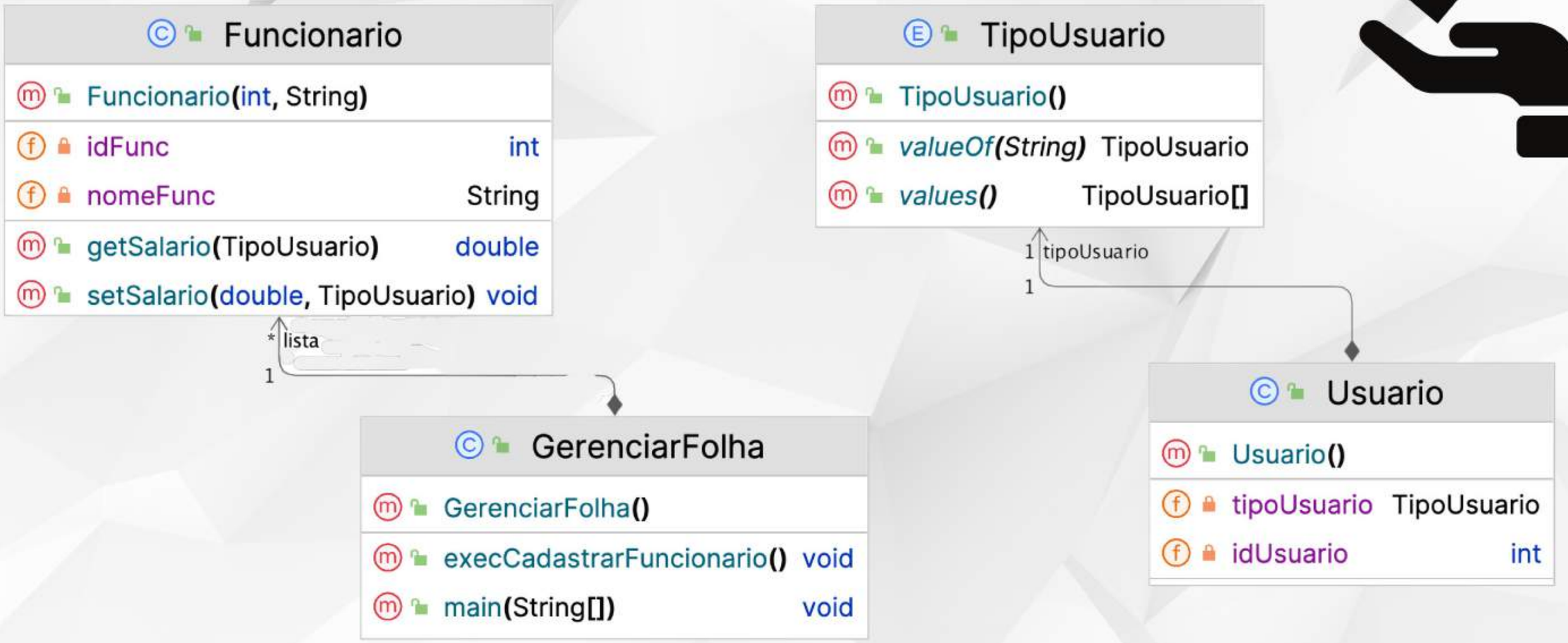
Pacotes da API do Java

Para ter uma visão geral dos pacotes da API, acesse:
https://docs.oracle.com/en/java/javase/11/docs/api/index.html

Enumeradores (enum)

- Java nos possibilita criar uma estrutura de dados enumarada.
- Essas estruturas de dados enumeradas são conjuntos de constantes organizados em ordem de declaração, ou seja, o que é declarado primeiro virá primeiro.
- A funcionalidade principal de **enum** é agrupar valores com o mesmo sentido dentro de uma única estrutura.
- No próximo exemplo, criou-se uma estrutura de enumerador para representar os tipos de usuários possíveis.

Aplicação: PrjFolhaPagamento



Enumerador Tipo

```
public enum TipoUsuario {
    OPERADOR,
    SUPERVISOR,
    ADMIN
}
```

Classe: Usuario

```java
public class Usuario {
    private int idUsuario;
    private TipoUsuario tipoUsuario;

    public int getIdUsuario() {
        return idUsuario;
    }

    public void setIdUsuario(int idUsuario) {
        this.idUsuario = idUsuario;
    }

    public TipoUsuario getTipoUsuario() {
        return tipoUsuario;
    }

    public void setTipoUsuario(TipoUsuario tipoUsuario) {
        this.tipoUsuario = tipoUsuario;
    }
}
```

Classe: Folha

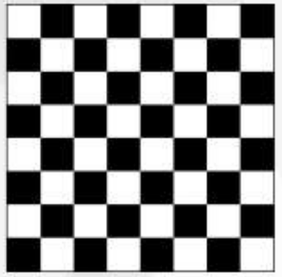
```
public class Funcionario {
    private int idFunc;
    private String nomeFunc;
    private double salario;

    .......

    public double getSalario(TipoUsuario tipoUsuario) {
        if(tipoUsuario==TipoUsuario.ADMIN){
            return salario;
        }
        throw new IllegalArgumentException("Sem permissão de acesso");
    }

    public void setSalario(double salario,TipoUsuario tipoUsuario) {
        if(tipoUsuario==TipoUsuario.ADMIN){
            this.salario = salario;
        }else{
            throw new IllegalArgumentException("Sem permissão para alterar salário");
        }
    }
}
```

Regra de acesso a leitura do campo. É passado o tipo de usuário que está acessando o campo para leitura. Caso seja tipo.admin, poderá ler o valor do Salário

Só pode definir o salário (atribuir valor) se o usuário for administrador.

DESAFIO

Estudo de Caso:
Jogo Tabuleiro da Sorte



A **loteca Boa Sorte** para entreter os seus clientes necessita de um programa em Java que enquanto os clientes aguardam na fila possam se distrair jogando no tabuleiro da sorte.

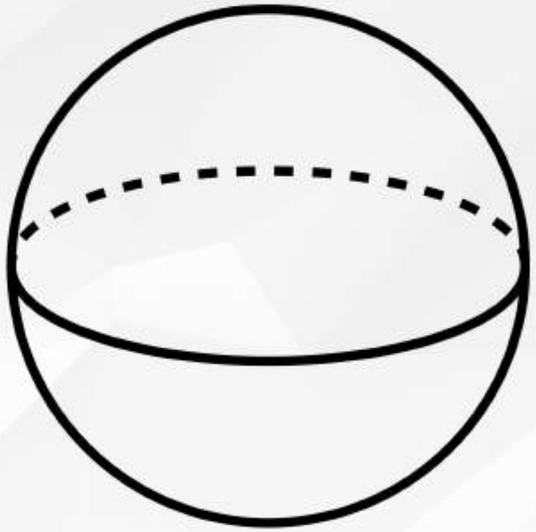
- O Tabuleiro da sorte é um jogo simples, que consistem em uma matriz de 10 x 10 [Bi dimensional] que cada elemento é um número inteiro entre 1 e 100.
- Esses elementos são gerados aleatoriamente e o usuário deve digitar dois números também entre 1 e 100.

  A regra para a vitória e prêmio:
- Caso um dos números digitados pelo usuário tenha aparecido 3 ou mais vezes no tabuleiro o usuário é um feliz ganhador.
- Prêmio: o valor é de R$ 1000,00 por número encontrado, supondo que no tabuleiro seja encontrado quatro números iguais a um dos que o usuário tenha digitado, o seu prêmio é de R$ 4.0000,00.

  Sua missão: desenvolver um programa em Java (Orientado a Objetos) que solucione este cenário.

Exercício prático

- Você deve criar um programa orientado a objetos que dado o valor de um raio pelo usuário, seja calculado e devolvido o valor do volume de uma esfera.
- Realize as abstrações necessárias e crie a classe de modelagem com atributos e métodos que ache pertinente.



O volume de uma esfera de raio r é igual a $\frac{4}{3}\pi r^3$.

$$V = \frac{4}{3}\pi r^3$$

Observe a figura

Crie um novo Projeto Chamado **ProjetoBaralho**

1. Modele conforme figura ao lado. Nela temos dois enumeradores Valor e Naipe, Uma Classe carta que é composta de Naipe e Valor, e um Classe Baralho que contém uma Lista de Cartas.

2. Seu desafio:
    1. Monte um baralho com 52 cartas;
    2. Embaralhe essas cartas;
    3. Exiba o baralho já embaralhado.

Suba no git hub com comentários.

Obrigado

fim

Até a próxima aula

Referências Bibliográficas

- Mendes; **Java com Ênfase em Orientação a Objetos,** Novatec.
- Deitel; **Java, como programar** – 10° edição. Java SE 7 e 8
- Arnold, Gosling, Holmes; **A linguagem de programação Java** – 4° edição.
- Apostilas da Caelum.