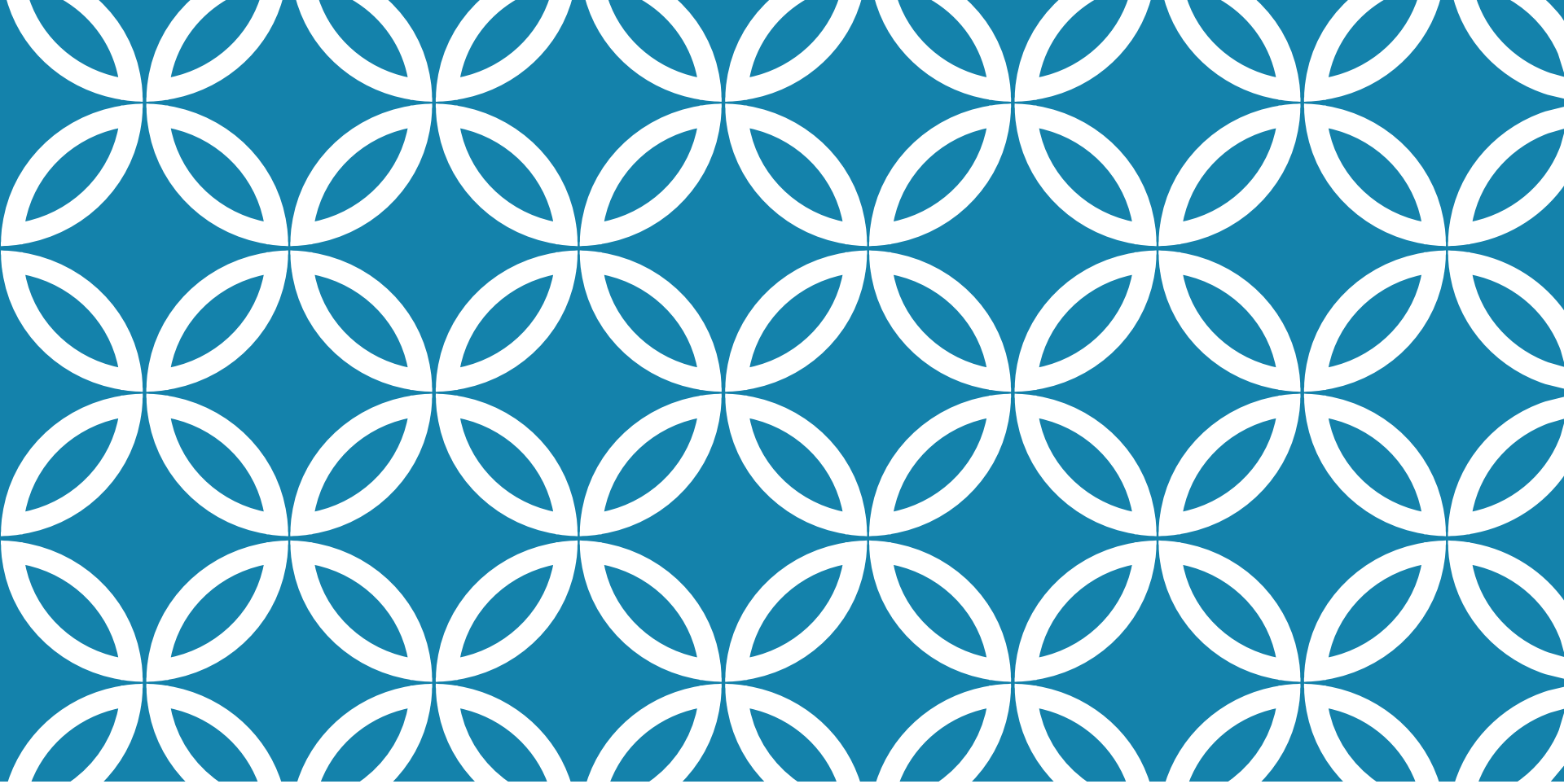


Programação II

Rafael Vieira Coelho

Métodos em Java

- métodos: Uma função definida dentro de um arquivo.
- As variáveis de entrada para um método são chamados de *Parâmetros*.
- A variável de saída de um métodos é chamada de seu *valor de retorno* (*return value*).
- Se um método não tiver valor de retorno em Java, declaramos como tendo tipo de retorno *void* (ex: método `main`).

Métodos em Java

- **Reduzem a complexidade**
  - ✓ **Abstração**
  - ✓ **Encapsulam a informação**
  - ✓ **Minimizam o tamanho do código**

Aumentam a facilidade!

- **Aumentam a manutenibilidade e a correção**
  - ✓ **Evitam duplicação do código**
  - ✓ **Limitam o efeito das mudanças**
  - ✓ **Promovem a reutilização do código**

Diminuem o custo!

Métodos

- **Um métodos é apenas um bloco de código nomeado que tem zero ou mais entradas e gera zero ou uma saída.**

- **Exemplos:**

```
public static void main(String args[]);
public static double sqrt(double pVal);
```

Assinatura de métodos

- **A assinatura de um método é o que o identifica univocamente.**
- **A assinatura consiste do nome do método e da lista de parâmetros**
- **Exemplos:**

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

Assinatura

```
public static double sqrt(double pVal);
```

Assinatura de métodos

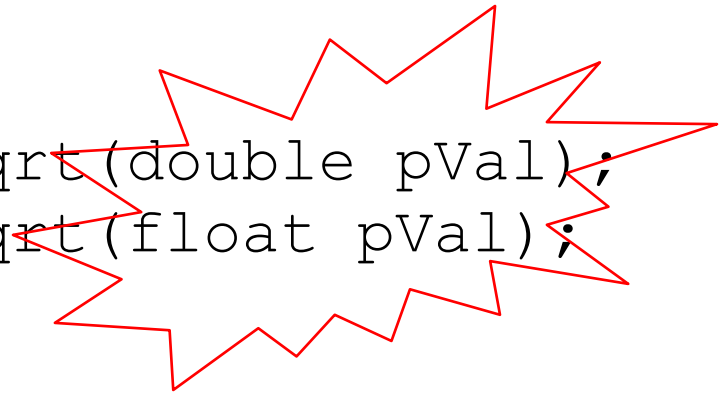
- **Nós podemos sobrecarregar um método, colocando duas definições diferentes para ele.**

- **Isto significa que, dependendo dos parâmetros que ele receber, ele vai ter um comportamento diferente.**

- **Para tanto, é importante que as assinaturas das duas ou mais definições sejam 100% distinguíveis pelo compilador**

- **Exemplo:**

```
public static double sqrt(double pVal);
public static float  sqrt(float pVal);
```



Assinatura de métodos

- **Logo, se dois métodos têm o mesmo nome, o método correto sera chamado com base nos argumentos que lhe são passados.**

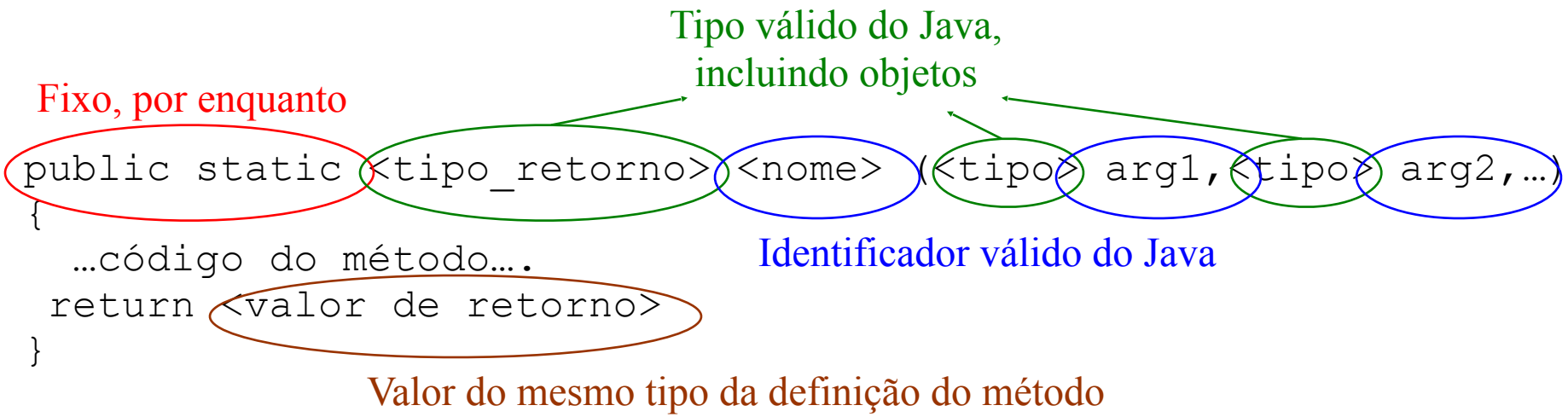
- **Por exemplo, o código abaixo chama os diferentes métodos definidos anteriormente.**

```
float  hypotenuse = 10.0f;
double longLeg = 5.0;

MyClass.sqrt(hypotenuse);
MyClass.sqrt(longLeg);
```

Definindo métodos

Para declarar métodos, usamos a seguinte sintaxe:



Chamando métodos

- **Como nós, por enquanto, só estamos usando métodos estáticos, eles pertencem às classes.**

- **Logo, para chamá-los, temos que dizer "o nome completo" do método, isto é, a classe a que ele pertence e o nome dele, da seguinte forma:**

```
MyClass.métodosName(arg1, arg2…)
```

Métodos

- **Métodos são blocos de código nomeados e auto-contidos.**
- **Sua única preocupação é com suas entradas e saídas.**
- **Todos os parâmetros em Java são passados por valor. Não existe passagem por referência.**
- **Entretanto, lembre-se que um objeto na verdade é um ponteiro e, apesar do ponteiro ser passado por valor, modificações na área apontada por ele são efetivadas globalmente.**

Métodos

- Variáveis definidas dentro de métodos têm escopo local, limitado àquele método, e não são vistas em outros trechos da classe.

- Você pode declarar variáveis com o mesmo nome em métodos diferentes.

- Nomenclatura usual:
  - Use um verbo seguido de um objeto (no sentido gramatical)
  - Não use nomes extremamente grandes.
  - A primeira letra é minúscula, mas outras palavras componentes do nome são iniciadas por maiúsculas.
  - Ex. : printReport, updatePosition, etc

Guia para definir métodos

- **Métodos são funções ➲ Use-os com o mesmo senso de organização que usávamos funções nas linguagens imperativas.**

- **Crie métodos para**
  - ✓ Encapsular complexidade e fazer seu código mais legível
  - ✓ Evitar código duplicado. Exemplo: O que é mais interessante:

```
inches = centimeters / 2.54;
inches = Metric.centimetersToInches(centimeters);
```

  - ✓Promover reutilização de código
  - ✓Isolar operações e estruturas de dados complexas

Guia para definir métodos

- **Métodos devem ter forte coesão: tudo dentro de um método deve ser relacionado ao seu propósito central. Se há dois propósitos, deve haver dois métodos.**

- **Outro método de pensar em métodos é colocar dentro deles passos relacionados que são executados em seqüência. Uma boa idéia é ter um método central que chama vários outros em seqüência**

Variáveis de Classe

- Java não permite que usemos variáveis globais (toda variável deve existir dentro de uma classe).

- Java usa a palavra chave `static` para indicar uma *variável de classe* versus uma *variável de instância*.

- Para acessar variável de classe, use o nome da classe

- Ex:

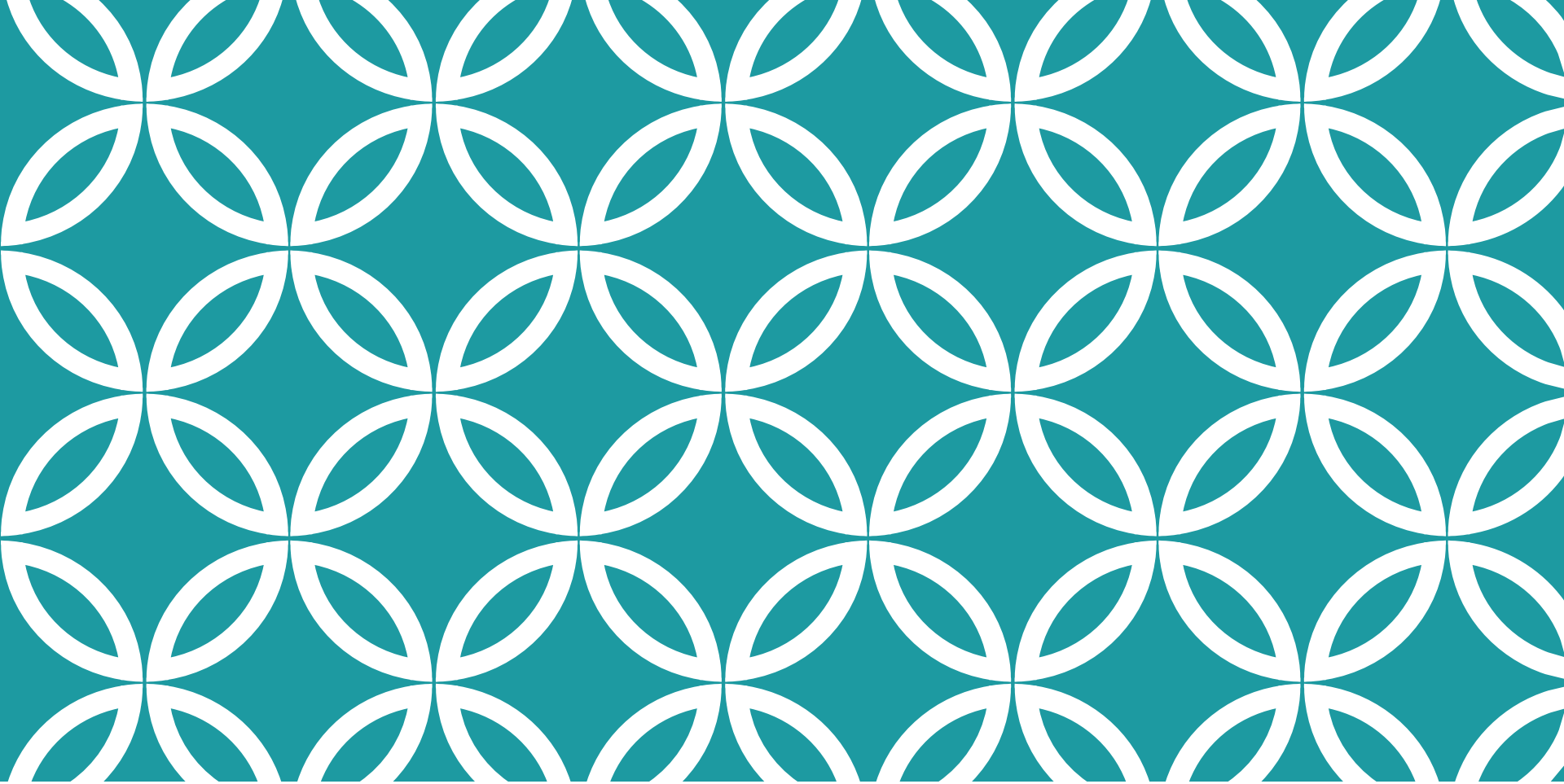
```
public static final double PI = 3.14159;
public static int numCircles;
```

Métodos de classe

- **Exemplo:** `Math.sqrt();`

- `Math` **é o nome da classe e** `sqrt()` **é o nome do método da classe(ou método estático).**

- **São equivalentes a métodos "globais", mas sem haver chance de conflitos com nomes.**



2. Exercícios