Fundamentos de programação

Métodos

Modularização de código

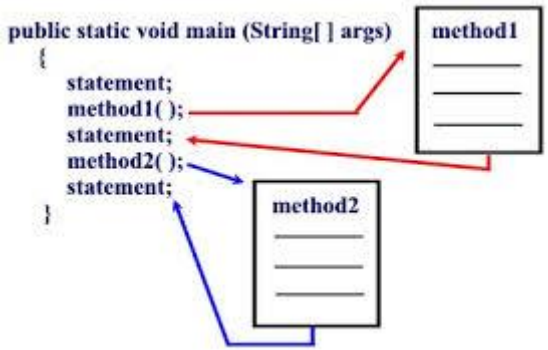
Edson Moreno

edson.moreno@pucrs.br

http://www.inf.pucrs.br/~emoreno

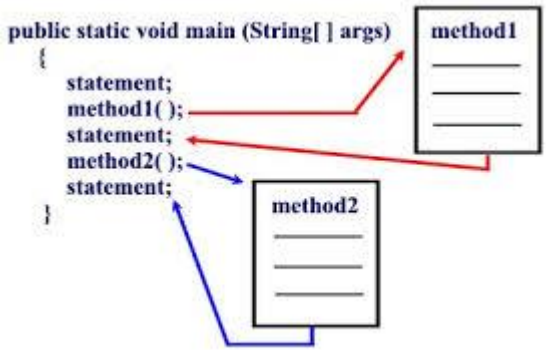
O que é um método?

- Um método é um trecho de código identificado
  - Possuem um nome
  - Tem um propósito específico (caixa preta)
  - Sempre pode ser invocado por outros métodos do mesmo objeto
  - Dependendo da forma como for implementado, pode retornar valores



Porque trabalha com métodos?

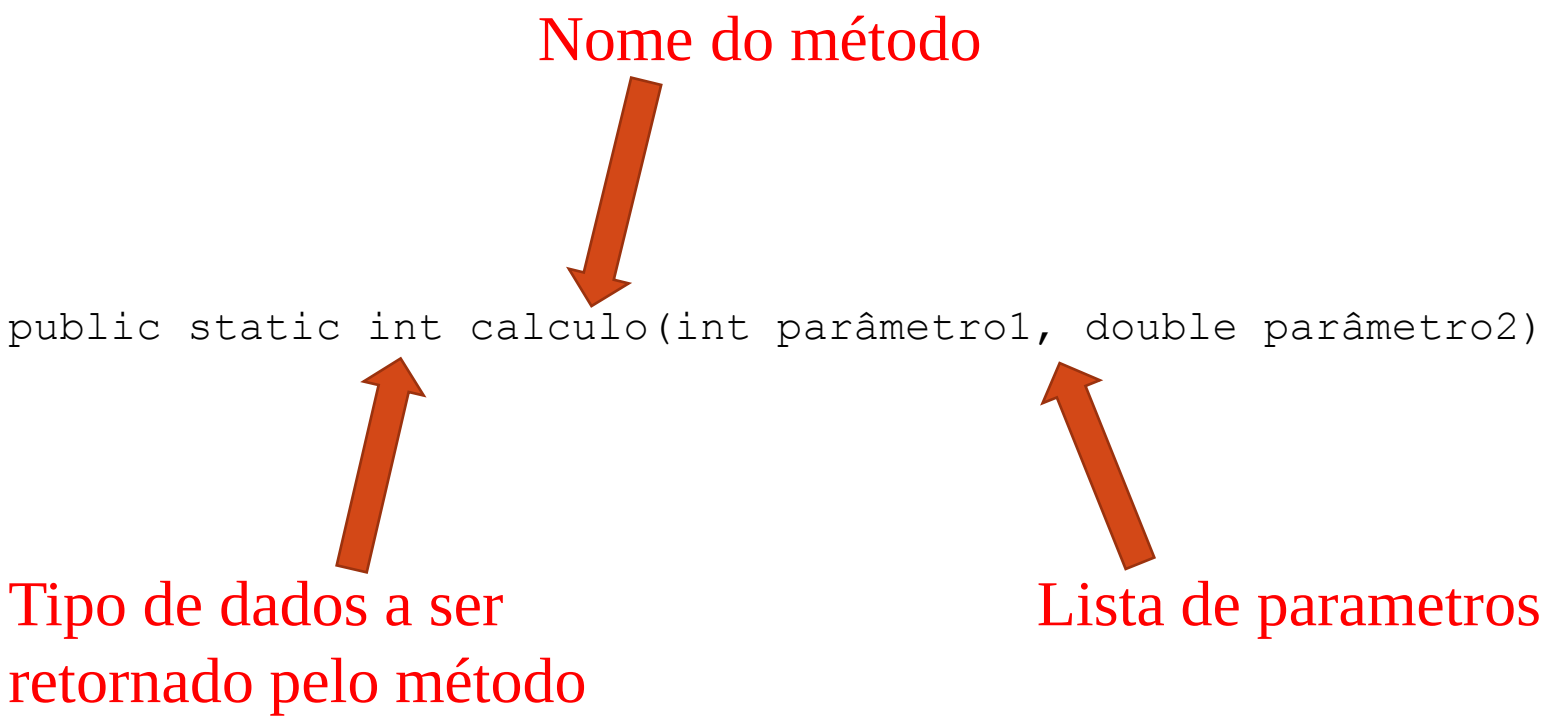
- A implementação de métodos permite
  - Reduzir quantidade de código redundante
  - Reduzir a quantidade de tempo de implementação
  - Concentrar pontos de avaliação/validação de código
  - Simplifica a construção de programas
  - Podem ou não retornar um resultado



Tipos de métodos

- Existem basicamente 2 tipos
  - Previamente definidos
    - E.g. System.out.println();
  - Desenvolvidos pelo programador (Objetivo desta aula)
    - E.g. Circulo.print();
- Métodos que já trabalhamos
  - Math.pow()
  - Math.sqrt()
  - Scanner.nextInt()
  - main()

Estrutura de um método



O método principal

- Main[1]
  - Todo programa em java requer este método
  - Main é a entrada do programa
  - Os parâmetros são as entradas aceitas

- Exemplo

```
public class Echo {
    public static void main (String[] args) {
        for (String s: args) {
            System.out.println(s);
        }
    }
}
```

1 - https://www.cs.princeton.edu/courses/archive/spr96/cs333/java/tutorial/java/anatomy/main.html

O método principal

- Exemplo

```
public class Echo {
    public static void main (String[] args) {
        for (String s: args) {
            System.out.println(s);
        }
    }
}
```

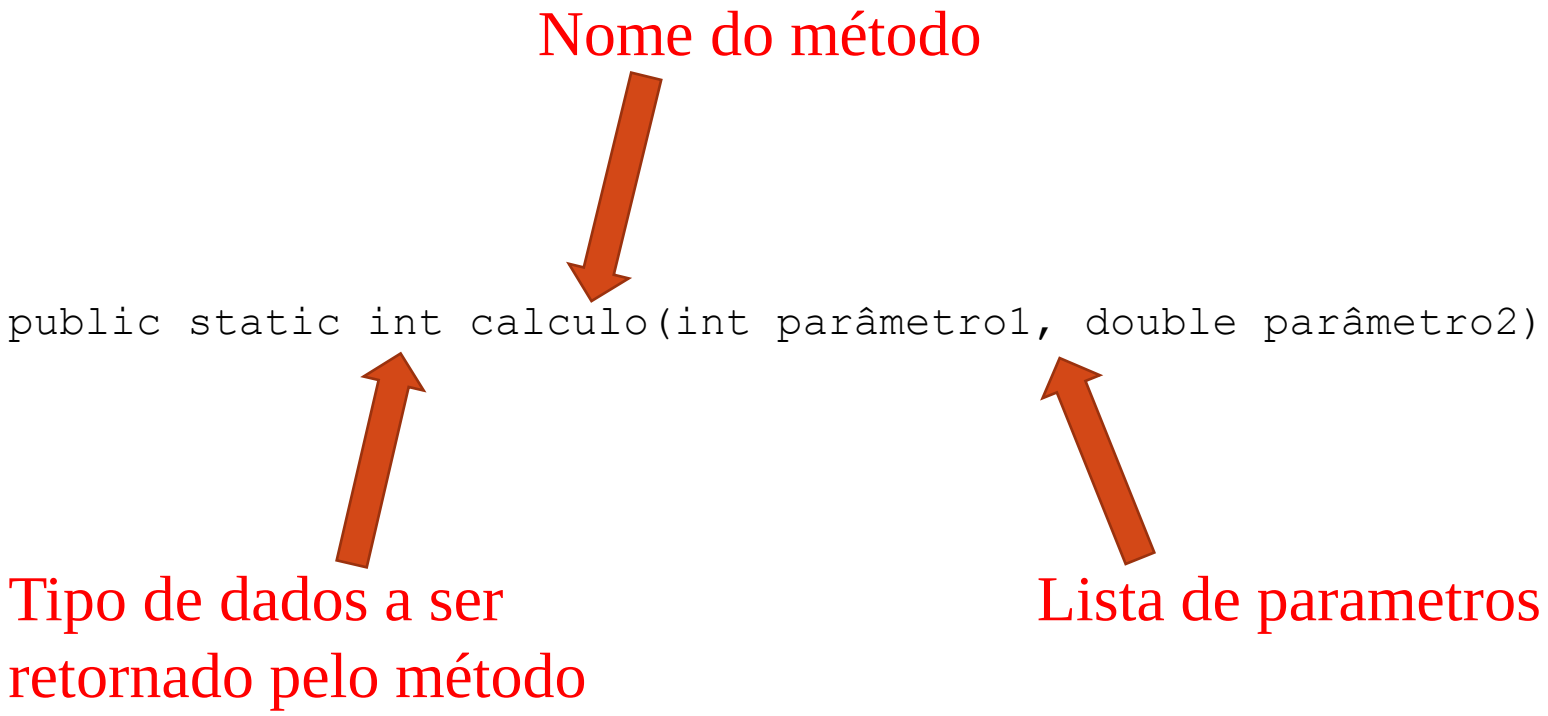
- Faço o teste
  - Abra um terminal
    - Windows+R
    - Digite o comando cmd
  - Salve o arquivo como *Echo.java*
  - Compile usando o comando *javac -cp . Echo.java*
  - Execute usando o comando *java Echo*
  - Experimente o comando *java Echo Teste de args*

Definição de um método

- O que é necessário
  - Definir se o método deve retornado algo
    - Se sim, defina o tipo
    - Se não, o tipo é void
  - Defina um nome
    - A escolha é sua, mas um nome contextualizado é sempre bom
  - Defina os parâmetros necessários
    - Quais são os parâmetros necessários
      - Quantidade de parâmetros
      - Tipos dos parâmetros

Definição de um método

- Resultado é como apresentado anteriormente



Implementação de um método

- Deve funcionar como uma caixa preta
  - Não importa como ocorre a computação
  - Importa
    - Os parâmetros a serem passados
    - O resultado esperado

Implementação de um método

- Um termostato como exemplo de caixa preta
  - Para implementar o método
    - Defina a temperatura desejada
    - O termostato liga o aquecimento conforme necessário
    - Não é preciso saber como ele realmente funciona!
      - Como ele sabe a temperatura corrente?
      - Quais sinais/comandos ele deve enviar ao aquecedor para ligá-lo?
  - Use métodos como “caixas pretas”
    - Passe para o método o que ele precisa para fazer o seu trabalho
    - Receba a resposta

Implementação de um método

- Um método para calcular o volume de um cubo
  - O que ele precisa para fazer os seu processamento?
  - Qual resposta ele retorna?
- Quando se escreve um método:
  - Escolha um nome para o método (cubeVolume)
  - Declare uma variável para cada argumento de entrada (double sideLength)
  - Determine o tipo do valor retornado (double)
  - Adicione modificadores tais como public static

```
public static double cubeVolume(double sideLength)
```

Dentro do método

- Escreva o corpo do método
  - O corpo do método é colocado entre chaves
  - O corpo contém declarações de variáveis e comandos que são executados quando o método é chamado
  - O método também deve retornar o valor calculado

```
public static double cubeVolume(double sideLength) {
   double volume = sideLength * sideLength * sideLength;
   return volume;
}
```

Invocação/chamada de um método

- Os valores retornados por cubeVolume são armazenados em variáveis locais dentro de main
- Os resultados são mostrados

```
public static void main(String[] args) {
    double result1 = cubeVolume(2);
    double result2 = cubeVolume(10);
    System.out.println(
      "A cube of side length 2 has volume " +
      result1);
    System.out.println(
     "A cube of side length 10 has volume " +
     result2);
}
```

Código completo

```
/**
This program computes the volumes of two cubes.
*/
public class Cubes {
  public static void main(String[] args) {
    double result1 = cubeVolume(2);
    double result2 = cubeVolume(10);
    System.out.println("A cube with side length 2 has volume " + result1);
    System.out.println("A cube with side length 10 has volume " + result2);
  }
/**
Computes the volume of a cube.
@param sideLength the side length of the cube
@return the volume
*/
  public static double cubeVolume(double sideLength) {
    double volume = sideLength * sideLength * sideLength;
    return volume;
  }
}
```

Implementação de um método

- Exemplo de métodos sem retorno

```
public static void asterisks( )       {
          int count;
          for(count = 1; count<=20; count++)
               System.out.print("*");
          System.out.println( );
     }
```

Implementação de um método

- Exemplo de métodos com retorno

```
public static int soma(int a, int b)
{
    int c=a+b;
    return(c);
}
```

Implementação de um método

- Exemplo de métodos com múltiplos retornos

```
public static int maior(int a, int b)
{
   if(a>b) return(a);
   else return(b);
}
```

Escopo

- Variáveis tem escopo de atuação
  - Variáveis declaradas dentro de uma classe
    - Podem ser acessada e modificadas por toda a classe
  - Variáveis declaradas dentro de um método
    - Podem ser acessadas e modificadas dentro do método
  - Parâmetros de um método
    - Podem ser acessado e modificados dentro do método
    - A variável passada por parâmetro não é modificada

Escopo

- Código exemplo

```
public class test
{
    static int variavelDaClasse; // variável declarada com escopo na classe
    public static void main(String [] args)
    {
        int variavelDoMetodo=0; // variável declarada com escopo no método
        System.out.println("O valor da variavelDoMetodo é " + variavelDoMetodo);
        variavelDaClasse=vaiSaber(variavelDoMetodo)+1;
        System.out.println("variavelDoMétodo após a chamada é " + variavelDoMetodo);
        System.out.println("A variávelDaClasse é " + variavelDaClasse);
    }

    public static int vaiSaber(int parametro) // parâmetro com escopo no método
    {
      parametro=parametro+1;
      System.out.println("O valor do parametro é = " + parametro);
      return (parametro);
    }
}
```

Documentação digital

- Java possui uma ferramenta para automatizar a geração de documentação
  - Javadoc[1]

- Geração da documentação vem da inclusão de comentários feitos pelo programador
  - Comentário incluído acima do método
  - Inicie o comentário com /**
  - Escreva o propósito do método
  - Descreva os argumentos iniciando a linha por @param
  - Descreva o que será retornado iniciando a linha por @return
  - Termine com */

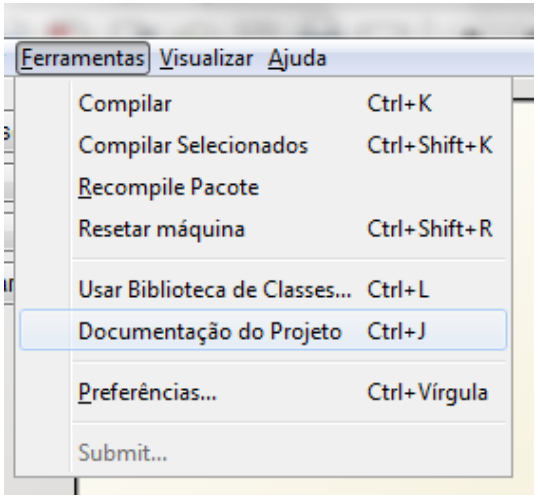
1 - http://docs.oracle.com/javase/7/docs/technotes/tools/windows/javadoc.html

Documentação digital

- Exemplo de comentários para documentação

```
/**
Computes the volume of a cube.
@param sideLength the side length of the cube
@return the volume
*/
public static double cubeVolume(double sideLength)
```

- Gerando a documentação no bluej



Exercícios

- Implemente um método ***multInt(int a, int b)*** para calcular ***a*** multiplicado por ***b*** usando apenas somas sucessivas.

- Implemente um método ***powInt(int a, int b)*** para calcular ***a*** elevado na potência ***b*** usando apenas multiplicações sucessivas. Use o método multInt.

- Implemente um método main para testar o método ***powInt*** (e indiretamente também testar o método multInt).

Exercícios

- Implemente métodos para os seguintes casos:
  - Calcular o fatorial de um número
  - Verificar se um número é primo ou não
  - calcular as áreas de:
    - quadrado,
    - retângulo,
    - círculo,
    - trapézio,
    - triângulo
  - Verificar se, dados 3 comprimentos, eles poderiam corresponder aos lados de um triângulo ou não