Linguagem de Programação Orientada a Objetos I

Introdução a Programação em Java

Prof. Tales Bitelo Viegas
https://fb.com/ProfessorTalesViegas

Estruturas Fundamentais de Java

- Sintaxe Geral
  - Distinção entre maiúsculas e minúsculas
  - Nomes de Classe: iniciam em maiúsculas
    - class Button, class NumberFormat
  - Nomes de variáveis: iniciam em minúsculas
    - int idade, float impostoDevido
  - Nomes de métodos: são verbos que iniciam em minúsculas e após usam maiúsculas
    - imprimirDados(), calcularImpostos()

Estruturas Fundamentais de Java

- Tipos de Dados Primitivos
  - Números Inteiros
    - int (32 bits)
    - short (16 bits)
    - long (64 bits)
    - byte (8 bits)
  - Números Reais
    - float (32 bits)
    - double (64 bits)
  - Caracter
    - char (Unicode – 16 bits)
  - Lógico
    - boolean (1 bit)

Estruturas Fundamentais de Java

- Operadores Aritméticos
  - Adição: +
    - op1 + op2
  - Substração: -
    - op1 - op2
  - Multiplicacão: *
    - op1 * op2
  - Divisão: /
    - op1 / op2
  - Resto da divisão: %
    - op1 % op2

Estruturas Fundamentais de Java

- Operações com atribuição
  - +=
    - Exemplo: x+=y;
    - Expressão equivalente: x = x + y;
  - -=
    - Exemplo: x-=y;
    - Expressão equivalente: x = x - y;
  - *=
    - Exemplo: x*=y;
    - Expressão equivalente: x = x * y;
  - /=
    - Exemplo: x/=y;
    - Expressão equivalente: x = x / y;
  - %=
    - Exemplo: x%=y;
    - Expressão equivalente: x = x % y;

Estruturas Fundamentais de Java

- Operadores de incremento e decremento
  - Incremento
    - Pré: ++op
    - Pós: op++
  - Decremento
    - Pré: --op
    - Pós: op—
  - Exemplos:
    - Considere m = 7 e n = 6
    - Ex: a = 2 * ++m → resultado: a = 16, m = 8
    - Ex: b = 2 * --n → resultado: b = 10, n = 5
    - Ex: a = 2 * m++ → resultado: a = 14, m = 8
    - Ex: b = 2 * n-- → resultado: b = 12, n = 5

Estruturas Fundamentais de Java

- Conversão entre Tipos Númericos
  - Conversões Implícitas
    - Maior precisão = Menor precisão
    - double > float > long > int > short > byte
    - Exemplo:
      int i = 10;
      double j = i;
    - Por padrão, um número com "." é um double (12.3 é double)
  - Conversões explícitas
    - Menor precisão = (cast) Maior precisão
    - Exemplo:
      double j = 14.89;
      int i = (int) j;

Estruturas Fundamentais de Java

- Operadores Relacionais
  - Maior: >
    - op1 > op2
  - Menor: <
    - op1 < op2
  - Igual: ==
    - op1 == op2
  - Maior Igual: >=
    - op1 >= op2
  - Menor Igual: <=
    - op1 <= op2
  - Diferente: !=
    - op1 != op2

Estruturas Fundamentais de Java

- Operadores Condicionais
  - AND: &&
    - op1 && op2
  - OR: ||
    - op1 || op2
  - NOT
    - ! op
- Operadores Bit a Bit
  - AND: &
    - op1 & op2
  - OR: |
    - op1 | op2
  - XOR: ^
    - op1 ^ op2

Construções Fundamentais

- Estruturas de Decisão
  - if-else

```
if (expressaoBooleana) {
    // codigo para a expressaoBooleana verdadeira
}
else {
    // codigo para a expressaoBooleana falsa
}
```

Construções Fundamentais

- Estruturas de Decisão
  - Operador ?
    - Operador ternário que funciona como uma abreviatura de um if-else

      expressao ? valor1 : valor2

    - O valor da expressão será valor1 caso a expressão seja verdadeira ou valor2 caso contrário
    - Exemplo:

      y = x >= 0 ? x : -x;

    - Poderia ser escrita da seguinte forma:

```
if (x>=0) {
    y = x
} else {
    y = -x;
}
```

Construções Fundamentais

- Estruturas de Decisão
  - switch-case

```
switch (expressao)
{
     case valor1:
    // codigo caso expressao seja igual valor1
    break;
     case valor2:
    // codigo caso expressao seja igual valor2
    break;
     . . .
     default:
    // codigo caso a expressao seja diferente de todos os
    // outros valores anteriores
}
```

Construções Fundamentais

- Estruturas de Repetição
  - for

```
for (inicializacao; expressaoBooleana; incremento) {
    // codigo executado enquanto expressaoBooleana
    // for verdadeira
}
```

  - while

```
while (expressaoBooleana) {
    // codigo executado enquanto expressaoBooleana
    // for verdadeira
}
```

Construções Fundamentais

- Estruturas de Repetição
  - do-while

```
do {
    // codigo executado enquanto expressaoBooleana
    // for verdadeira
}
while (expressaoBooleana);
```

Construções Fundamentais

- Controle de Fluxo
  - break
    - Interrompe estruturas while, for, do/while ou switch. A A execução continua com a primeira instrução depois da estrutura.
  - continue
    - Quando executada em uma estrutura while, for ou do/while, pula as instruções restantes no corpo dessa estrutura e prossegue com a próxima iteração do laço.

Estruturas Fundamentais de Java

- Comentários
  - Comentário de Linha

    // Este é um comentário

  - Comentário de Múltiplas Linhas

    /* Este é um comentário que pode ser utlizado em várias linhas */

  - Comentário de Documentação

    /** Este é um comentário para gerar documentação */

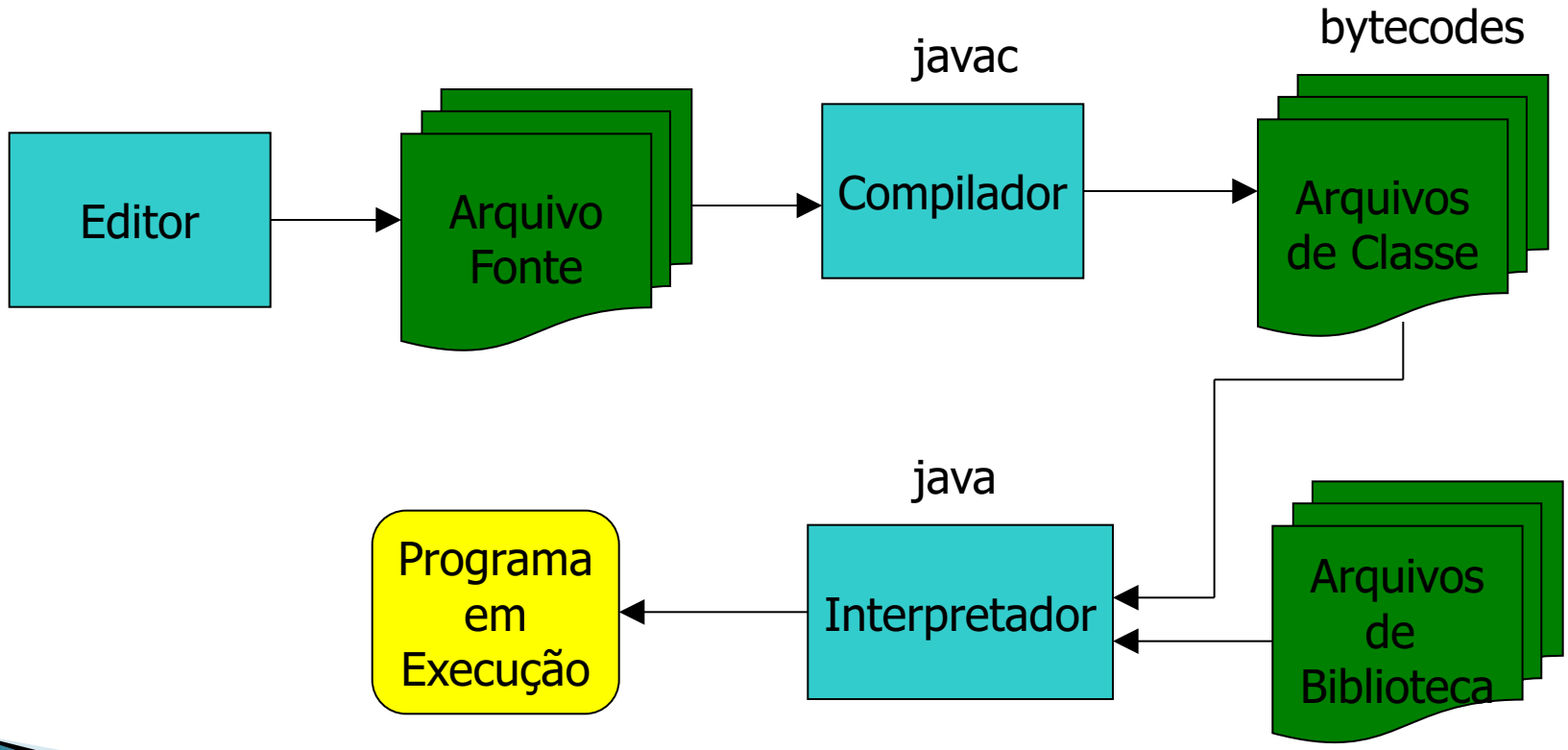
Para compilar e executar um programa Java

- É necessário o kit de desenvolvimento
  - The Java SE Developement Kit

- Conjunto de ferramentas de desenvolvimento de aplicações
  - compilador
  - ambiente de execução de programas: máquina virtual, bibliotecas de classes e outros arquivos
  - classes de demonstração
  - depurador
  - documentação de classes
  - ...

Para compilar e executar um programa Java

- Desenvolvimento de Programas



Meu Primeiro Programa

- Programa bem simples
  - Hello World

```
class HelloWorld {
        public static void main(String[] args) {
          System.out.println("Bem-vindo a Java.");
        }
}
```

  - main(): inicia a execução

Para compilar e executar um programa Java

- Nome do Arquivo Fonte
  - O nome do arquivo fonte java deve ser igual ao nome da classe
  - Neste exemplo: HelloWorld.java

- Para Compilar
  - javac <nome_arquivo.java>
  - Neste exemplo: javac HelloWorld.java
  - O compilador irá gerar um arquivo de bytecodes
  - Neste exemplo: HelloWorld.class

Para compilar e executar um programa Java

- Para Executar
  - Executar a aplicação através da invocação da JVM
  - java <nome_arquivo>
  - Não coloque a extensão .class
  - No exemplo: java HelloWorld

Meu Segundo Programa

- Segundo programa
  - Utilizando argumento na execução do programa

```
class HelloWorldComArgs{
    public static void main (String[] args) {
       // Nao esquecer o argumento
       System.out.println("Caro aluno(a): " + args[0] + " ... bem vindo a Java." );
    }
}
```

  - Para executar: java HelloWorldComArgs Tales

Meu Terceiro Programa

- Terceiro programa
  - Utilizando mais de um método

```
class HelloWorld2 {

        public static void imprimir() {
           System.out.println("Bem-vindo a Java.");
        }

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

Entrada de Dados

- Necessário utilizar um “Scanner”

```
class HelloName {

        public static void main(String[] args) {
           Scanner teclado = new Scanner(System.in);
           System.out.print("Informe sua idade: ");
           int idade = teclado.nextInt();
           System.out.println("Você tem " + idade + " anos");
        }
}
```

- nextInt(); nextFloat(); nextDouble(); nextLine();

Onde aprender mais ?

- BORGES, Karen Selbach. Caderno Universitário - Java : uma linguagem de programação orientada a objetos. Ed. ULBRA, 2005.

- SANTOS, Rafael. Introdução à Programação Orientada a Objetos Usando Java. Rio de Janeiro: Campus, 2003.

- HORSTMAN, Cay. Big Java. Ed. Bookman, 2004.