

Minicurso de Java

Introdução a Java

João Paulo Taylor Ienczak Zanette

7 de Fevereiro de 2017

O que é programação?

Introdução a Java

1. Deve ser "simples, orientada a objetos, e familiar";
2. Deve ser "robusta e segura";
3. Deve ser "de arquitetura neutra e portátil";
4. Deve executar com "alta performance";
5. Deve ser "interpretada, dinâmica e com suporte a threads ";

Características gerais da linguagem

Multiplataforma: O mesmo código pode rodar em mais de uma plataforma sem a necessidade de recompilar;

Biblioteca padrão extensa: Oferece diversos recursos já na biblioteca padrão, desde interfaces gráficas até manipulação de áudio/vídeo;

Alocação não-determinística: Recursos alocados não são liberados explicitamente pelo programador, isso é feito por um *Garbage-Collector*;

Suporte a JIT: Alguns trechos de código são otimizados em tempo de execução (*Just In Time Compilation*).

“Java é uma mãezona: não deixa você fazer algo errado e ainda limpa tudo para você.”

Onde é utilizada

- Aplicações Desktop no geral (com ou sem interface gráfica);
- Comunicação com Banco de Dados em servidores de aplicações Web (*back-end*);
- Aplicações para dispositivos móveis Android;
- Sistemas embarcados e de tempo-real.

Foi bastante utilizada também para aplicações de celulares antigos (JavaME).

A *Java Virtual Machine*

Toda aplicação Java roda na JVM a partir de uma sequência de instruções geradas de um código Java. Essas instruções se chamam ***Bytecode***.

Desenvolvendo uma aplicação Java simples


1. Crie um arquivo de código Java (.java);
2. Escreva o código (ver exemplo adiante), definindo o procedimento `main`;
3. Gere o *Bytecode* da aplicação, que será lido pela JVM. Isso é feito pelo processo de *Compilação*;
4. Execute o código chamando a JVM e indicando qual classe deve ser carregada. A JVM vai procurar e executar o procedimento `main` dessa classe.

Exemplo de código Java

```
public class Application {  
    public static void main(String... args) {  
        System.out.println("Hello, World!");  
    }  
}
```

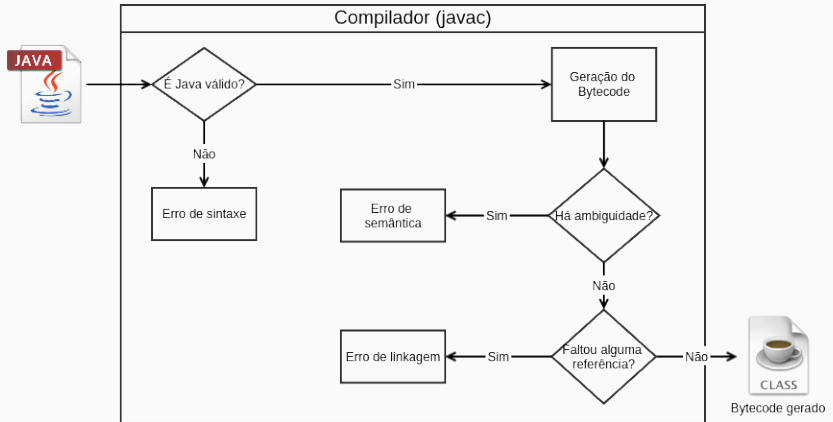
OBS: Toda aplicação Java inicia pelo main.

Utilizando o compilador de Java

☯ Eclipse: . Executar a aplicação também chama o compilador.

➤ Terminal: `javac Application.java`

Processo de compilação de Java



Bytecode gerado

```
public class Application {
```

```
    public Application();
```

```
        Code:
```

```
        0: aload_0
```

```
        1: invokespecial #1 // java/lang/Object."<init>"
```

```
        4: return
```

```
public static void main(java.lang.String...);
```

```
    Code:
```

```
        0: getstatic    #2 // java/lang/System.out
```

```
        3: ldc         #3 // Hello, World!
```

```
        5: invokevirtual #4 // java/io/PrintStream.println
```

```
        8: return
```

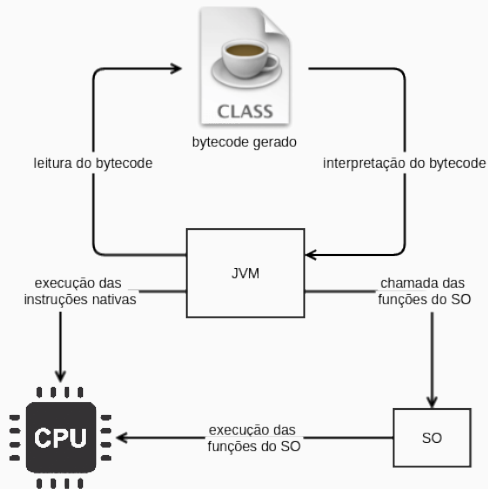
```
}
```

Executando uma aplicação Java

 Eclipse: Pressionar  +  ( Run).

 Terminal: `java Application`

Execução do Bytecode Java



Execução do *Bytecode* Java

Importante: A execução do código é sequencial, na ordem em que o código foi escrito. Portanto o seguinte trecho de código:

```
System.out.println("Linha 1");  
System.out.println("Linha 2");  
System.out.println("Linha 3");  
System.out.println("Linha 4");
```

Produzirá a saída:

```
Linha 1  
Linha 2  
Linha 3  
Linha 4
```

Trabalhando com programação

Problema: Como guardar valores?

Não queremos fazer programas que apenas mostrem textos na tela. Porém, se quisermos programas mais complexos, precisamos de mais recursos. Para fazer uma loja, por exemplo, precisamos:

- Guardar as informações dos produtos (nomes, preços, etc.);
- Guardar a lista de compras do cliente;
- Calcular o total da compra para que o cliente possa pagar.

Variáveis são valores guardados na memória (geralmente RAM).
Toda variável possui:

Identificador: Nome para referenciá-la no código. Deve ser único dentro de um mesmo escopo.

Tipo: Define o tipo de dado que ela guarda.

Há também, em programação, o conceito de *constantes*, que funcionam semelhante às variáveis, porém não podem ser alteradas.

Criando uma variável

Para criar uma variável, basta *declarar* ela, ou seja, escrever:

```
Tipo nome;
```

É possível ainda criar uma variável e já atribuir um valor a ela, da forma:

```
Tipo nome = valor;
```

Os tipos de dados que podem ser guardados são divididos em:

Tipos Primitivos: Valores indivisíveis, geralmente números;

Tipos Definidos por Usuário: Tipos personalizados, compostos por diferentes valores internos.

Tipos primitivos

Numéros inteiros:

Tipo	Tamanho	Intervalo
byte	1 Byte	-128 a 127
short	2 Bytes	-2^{15} a $2^{15} - 1$
int	4 Bytes	-2^{31} a $2^{31} - 1$
long	8 Bytes	-2^{63} a $2^{63} - 1$

Numéros reais:

Tipo	Tamanho	Intervalo
float	4 Bytes	2^{-149} a $(2 - 2^{-23}) \cdot 2^{127}$
double	8 Bytes	2^{-1074} a $(2 - 2^{-52}) \cdot 2^{1023}$

Para caracteres há o tipo char (2 Bytes).

Operações: Soma

```
int a = 5;  
int b = 7;  
int sum = a + b;
```

Após executar o trecho acima, o valor de `sum` será 12.

Operações: Subtração

```
int a = 5;  
int b = 7;  
int sub = a - b;
```

Após executar o trecho acima, o valor de sub será -2 .

Operações: Multiplicação

```
int a = 5;  
int b = 7;  
int mul = a * b;
```

Após executar o trecho acima, o valor de `mul` será 35.

Operações: Divisão

```
int a = 5;  
int b = 7;  
int div = a / b;
```

Esse é um caso especial em que, após executar o trecho acima, o valor de `div` será 0. Isso se dá porque estamos utilizando números inteiros.