

Aula 13: O que é Java?

Professor(a): Virgínia Fernandes Mota

<http://www.dcc.ufmg.br/~virginiaferm>

ALGORITMOS E ESTRUTURAS DE DADOS - SETOR DE INFORMÁTICA



O que é Java?

- Ao final desta aula você será capaz de:
 - responder o que é Java;
 - mostrar as vantagens e desvantagens do Java;
 - entender bem o conceito de máquina virtual;
 - compilar e executar um programa simples.

- Vamos entender um pouco da história da plataforma Java.
- Quais eram os seus maiores "problemas" quando programava na década de 1990?
 - ponteiros?
 - gerenciamento de memória?
 - organização?
 - falta de bibliotecas?
 - ter de reescrever parte do código ao mudar de sistema operacional?
 - custo financeiro de usar a tecnologia?
- "Resolve", mas a linguagem teve seu lançamento focado no uso em clientes web (browsers) para rodar pequenas aplicações (applets).

- O Java foi criado pela antiga Sun Microsystems (1992) e mantida através de um comitê (<http://www.jcp.org>).
- Seu site principal era o java.sun.com, e java.com um site mais institucional, voltado ao consumidor de produtos e usuários leigos, não desenvolvedores.
- Com a compra da Sun pela Oracle em 2009, muitas URLs e nomes tem sido trocados para refletir a marca da Oracle. A página principal do Java é:
<http://www.oracle.com/technetwork/java/>

- Em uma linguagem de programação como C, temos a seguinte situação quando vamos compilar um programa: código → compila → código binário.
- Muitas vezes o próprio código fonte é desenvolvido visando uma única plataforma.
- Já o Java utiliza do conceito de **Máquina Virtual**, onde existe, entre o sistema operacional e a aplicação, uma camada extra responsável por "traduzir- mas não apenas isso - o que sua aplicação deseja fazer para as respectivas chamadas do sistema operacional onde ela está rodando no momento.
- *Independência de plataforma.*

- Máquina Virtual é um conceito bem mais amplo que o de um interpretador: "computador de mentira".
- Sua aplicação roda sem nenhum envolvimento com o sistema operacional: A Máquina Virtual é responsável por gerenciar memória, threads, a pilha de execução, etc.
- **Java Virtual Machine (JVM).**

- Essa camada, a Máquina Virtual, não entende código java, ela entende um código de máquina específico.
- Esse código de máquina é gerado por um compilador java, como o javac, e é conhecido por "bytecode", pois existem menos de 256 códigos de operação dessa linguagem, e cada "opcode"gasta um byte.
- O compilador Java gera esse bytecode que, diferente das linguagens sem Máquina Virtual, vai servir para diferentes sistemas operacionais, já que ele vai ser "traduzido"pela JVM.

- Hotspot é a tecnologia que a JVM utiliza para detectar "pontos quentes" da sua aplicação: código que é executado muito, provavelmente dentro de um ou mais loops.
- Quando a JVM julgar necessário, ela vai compilar estes códigos para instruções realmente nativas da plataforma, tendo em vista que isso vai provavelmente melhorar o desempenho da sua aplicação.
- Esse compilador é o JIT: Just inTime Compiler, o compilador que aparece "bem na hora" que você precisa.

Java lento? Hotspot e JIT

- A compilação é feita dinamicamente.
- Dessa forma é possível que algumas otimizações sejam feitas à medida que for necessário.
- Mas ainda assim, pode ser mais lento do que uma linguagem puramente estruturada, ou com mais recursos de hardware.
- Round 1: Fight → Java vs. C++



Compilando o primeiro programa

- Vamos para o nosso primeiro código! O programa que imprime uma linha simples.

```
1      System.out.println("Minha primeira aplicação Java!"  
    );
```

- Mas esse código não é aceito pelo compilador java: A burocracia do java.
- O mínimo que precisaríamos escrever é algo como:

```
1  class MeuPrograma {  
2      public static void main(String[] args) {  
3          System.out.println("Minha primeira aplicação  
4              Java!");  
5      }  
}
```

- Após digitar o código acima, grave-o como **MeuPrograma.java**.

Compilando o primeiro programa

- Para compilar, você deve pedir para que o compilador de Java da Oracle, chamado `javac`, gere o bytecode correspondente ao seu código Java.
 - `javac MeuPrograma.java`
- Depois de compilar, o bytecode foi gerado.
- Note que foi criado um arquivo `.class`.

Executando o primeiro programa

- O javac é o compilador Java, e o java é o responsável por invocar a Máquina Virtual para interpretar o seu programa.
 - java MeuPrograma

Como é o bytecode?

- O `MeuPrograma.class` gerado não é legível por seres humanos (não que seja impossível).
- Ele está escrito no formato que a virtual machine sabe entender e que foi especificado que ela entendesse.
- É como um assembly, escrito para esta máquina em específico. Podemos ler os mnemônicos utilizando a ferramenta `javap` que acompanha o JDK.
 - `javap -c MeuPrograma`

Como é o bytecode?

```
MeuPrograma();  
  Code:  
    0:   aload_0  
    1:   invokespecial   #1; //Method java/lang/Object."<init>":()V  
    4:   return  
  
public static void main(java.lang.String[]);  
  Code:  
    0:   getstatic   #2; //Field  
java/lang/System.out:Ljava/io/PrintStream;  
    3:   ldc         #3; //String Minha primeira aplicação Java!!  
    5:   invokevirtual #4; //Method java/io/PrintStream.println:  
        (Ljava/lang/String;)V  
    8:   return  
  
}
```

Como é o bytecode?



Exercícios: Modificando o Hello World

- 1 Altere seu programa para imprimir uma mensagem diferente.
- 2 Altere seu programa para imprimir duas linhas de texto usando duas linhas de código `System.out`.
- 3 Sabendo que os caracteres `\n` representam uma quebra de linha, imprima duas linhas de texto usando uma única linha de código `System.out`.

Na próxima aula...

Revisão e Prova