Inatel
Instituto Nacional de Telecomunicações

Programação Orientada a Objetos

Prof. Renzo P. Mesquita

Bem-vindos ao curso de Programação Orientada a Objetos

ONBOARDING

Bem-vindos ao curso de Programação Orientada a Objetos

Critérios de Avaliação

*NPA = NPT * 0.60 + NPL * 0.40;*

NPT = (NP1 + NP2)/2;

*NP1 = PV1*0.60 + EP1*0.40*

*NP2 = PV2*0.60 + EP2*0.40*

*NPL = (RP * 0.50 + PF * 0,50);*

Resumo:

- 2 (duas) Provas (PV);
- Pelo menos 4 Exercícios Práticos Avaliativos;
- Relatórios Práticos (RP);
- Projeto Final (PF);

Bem-vindos ao curso de Programação Orientada a Objetos

Critérios de Aprovação

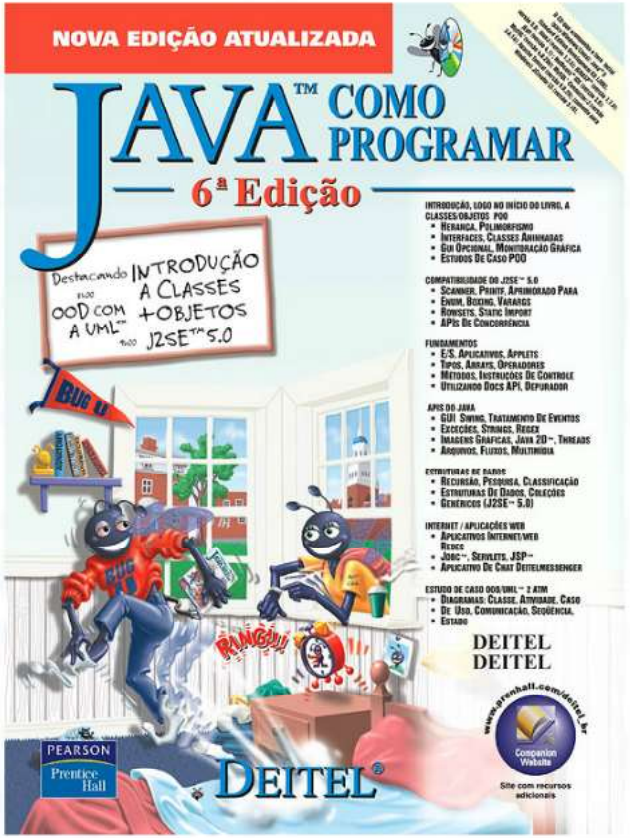
```
...
if(NPT >= 60 && NPL >= 60)
{
   escreve("APROVADO!"); NFA=NPA;
}
else if(NPA < 30)
{
   escreve("REPROVADO!"); NFA=NPA;
}
else
{
   calculaNP3(NPA);
}
...
```

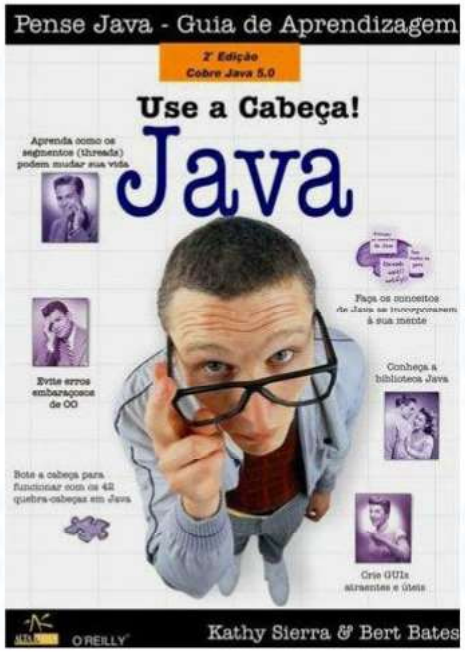
```
calculaNP3(int npa)
{
    NFA = (npa + NP3)/2;
    if(NFA >= 50)
    {
       escreve("APROVADO!");
    }
    else
    {
       escreve("REPROVADO!");
    }
}
```

Bem-vindos ao curso de Programação Orientada a Objetos

Material de Apoio

Livros, Apostilas, Fóruns etc.





POO e Java, preparem que aqui vamos nós!

Inatel
Instituto Nacional de Telecomunicações

Programação Orientada a Objetos

Cap.1 - Introdução ao Java

Prof. Me. Renzo P. Mesquita

Capítulo 1
Introdução ao Java

1.1. Por que usar o Java?
1.2. Uma breve história;
1.3. Máquina Virtual;
1.4. JVM, JRE e JDK;
1.5. Objetivos do Java (Onde podemos usá-lo?);
1.6. IDE's de Desenvolvimento;
1.7. Meu primeiro programa: "Hello Java!";
1.8. Conhecendo alguns shortcuts do IntelliJ;

Objetivos

1. Entender como funciona a Linguagem de Programação Java assim como suas Vantagens e Desvantagens;
2. Entender o que é uma Máquina Virtual;
3. Familiarizar-se com o IntelliJ IDEA, Compilar e executar nosso primeiro programa;

1.1. Por que usar o Java?

Quais eram os maiores problemas dos programadores na década de 90?

- Ponteiros?
- Gerenciamento de Memória?
- Rodar o programa em diferentes dispositivos e Sistemas Operacionais?
- Organização?
- Bibliotecas?
- Custo da tecnologia?
- ...

1.1. Por que usar o Java?

- A linguagem Java resolve bem os problemas citados anteriormente. Eles apareciam com frequência em outras linguagens de programação;
- O principal motivo de tentar evitar estes problemas era permitir que o Java rodasse em dispositivos pervasivos em geral como TV's, celulares, cafeteiras, aparelhos de DVD etc..

1.2. Uma Breve História

- Desde seu lançamento, em meados de 1995, a plataforma Java foi adotada mais rapidamente do que qualquer outra linguagem de programação na história;
- Em 2004 já possuía mais de 3 milhões de desenvolvedores em todo o mundo;
- Em 2006 a Sun anunciou que o Java faria parte do movimento de Software Livre, permitindo que desenvolvedores do mundo todo contribuíssem com mudanças para a linguagem;
- Em 2009 a empresa Oracle Corporation comprou a Sun Microsystems, e hoje, é a atual mantenedora do Java;

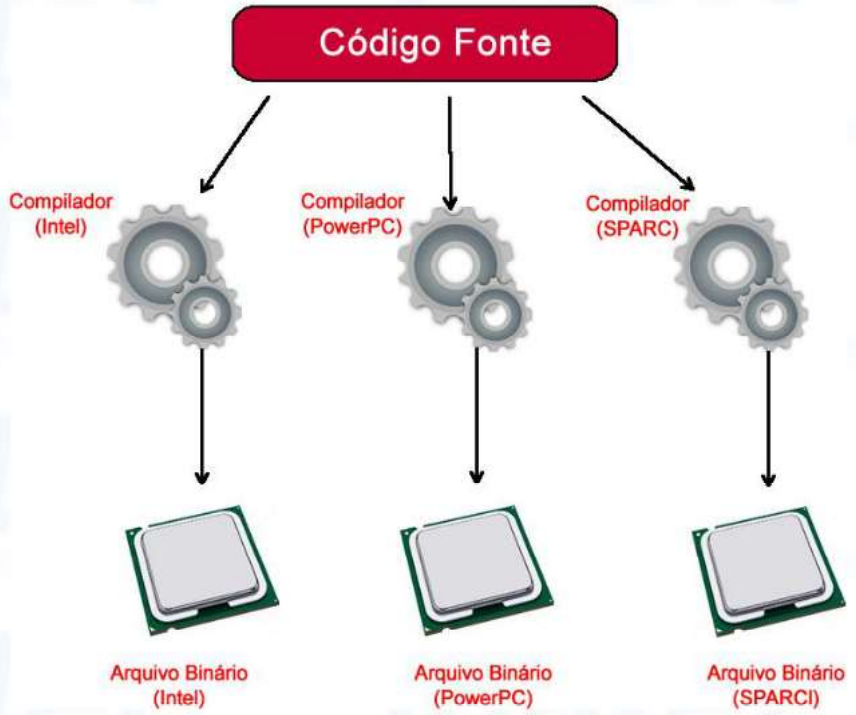
ORACLE®

1.3. Máquina Virtual

Joia! Mas qual foi e ainda é um dos fatores que faz o Java ser tão popular assim?

Em linguagens de programação como C++, Visual Basic e Pascal, temos por exemplo a seguinte situação quando vamos compilar um programa:

Para essas linguagens, o Código-fonte é compilado para o código de máquina específico de uma plataforma e sistema operacional.

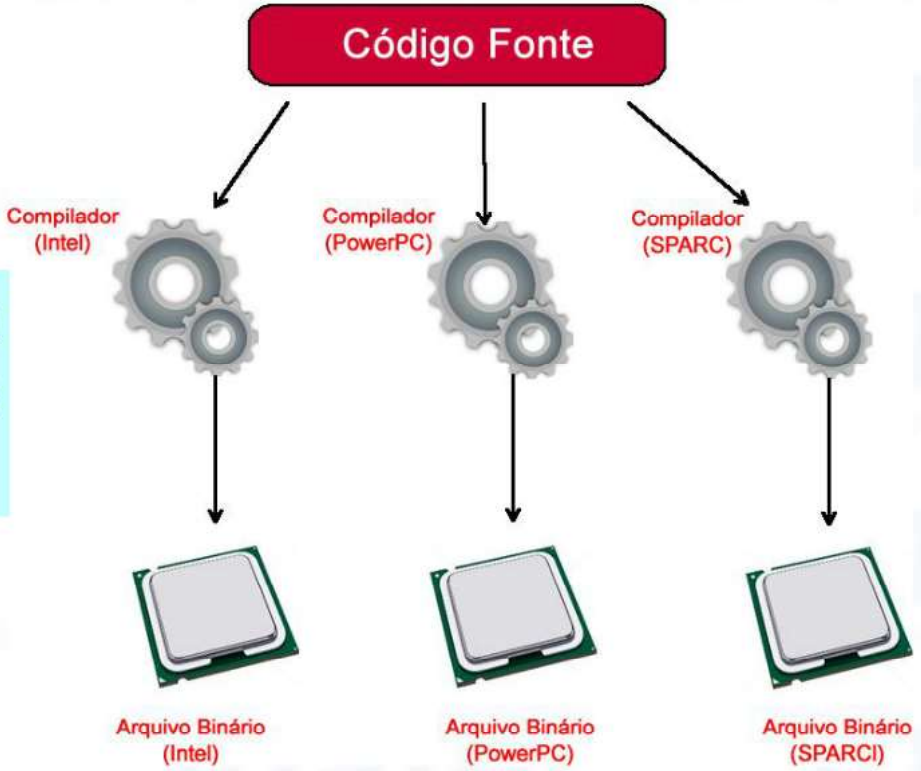


Tudo bem, mas qual o problema até aí?

1.3. Máquina Virtual

- Deve-se gerar um código executável para cada arquitetura/sistema operacional;
- Deveremos escrever um mesmo pedaço da aplicação para diferentes sistemas operacionais, já que eles não são compatíveis;

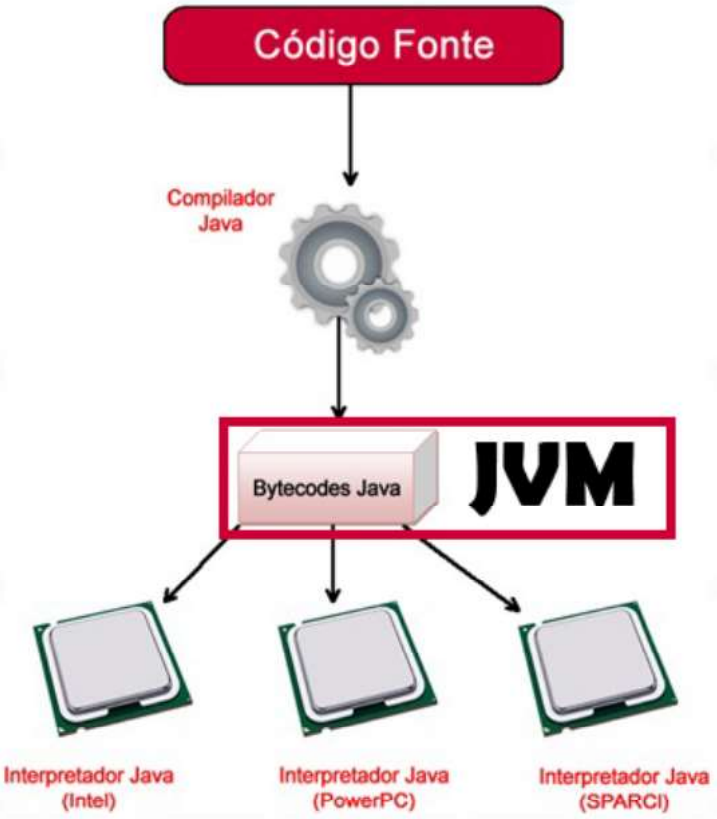
Mas isso é um problema? Tem jeito de deixar esse procedimento ainda mais fácil?



1.3. Máquina Virtual

O Java utiliza do conceito de Máquina Virtual, que é uma camada extra de Software responsável por rodar uma aplicação Java em qualquer arquitetura ou sistema operacional.

- Basta que o Hardware/SO seja capaz de executar uma máquina virtual Java.



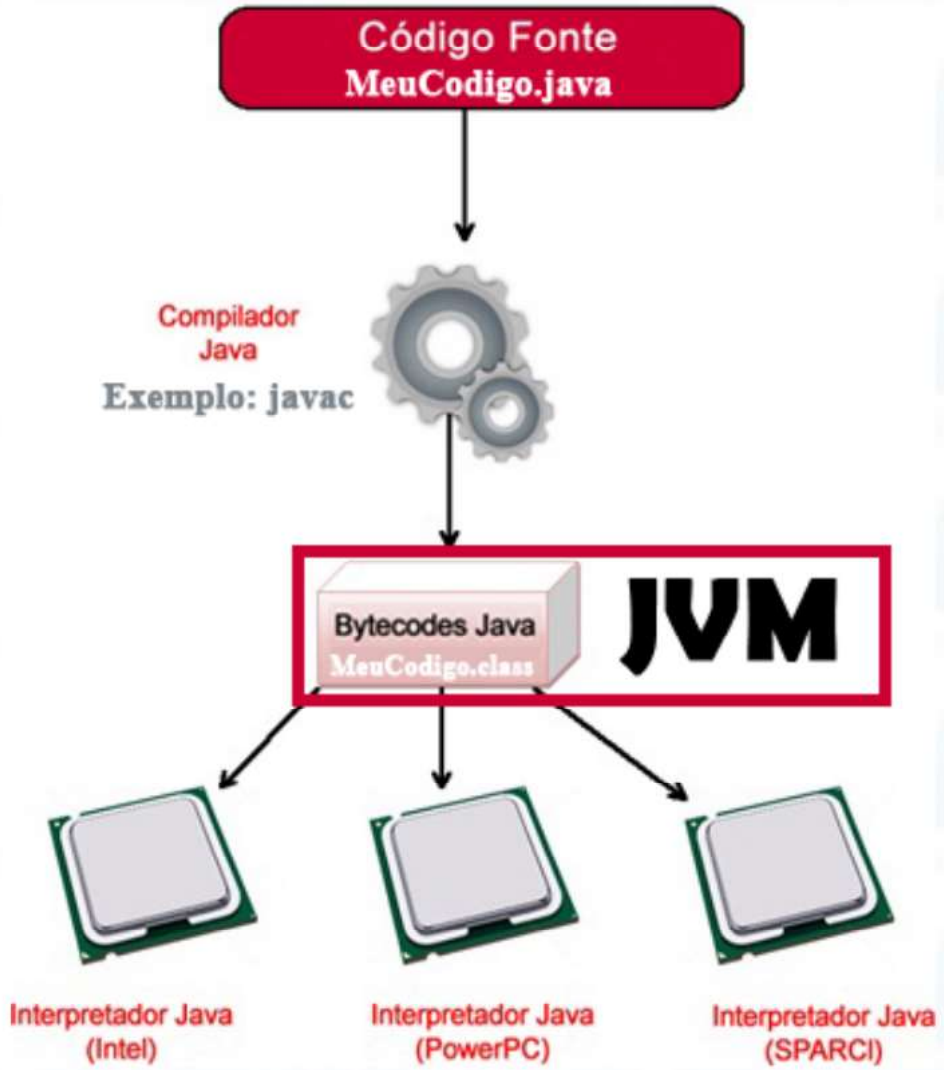
Vamos ver mais em detalhes como isso funciona?

1.3. Máquina Virtual

1. O Código desenvolvido pelo programador tem extensão *.java;

2. Este código é Compilado e transformado em um arquivo com extensão *.class (Bytecodes);

3. O código *.class agora é interpretado pela JVM, permitindo executá-lo em qualquer Arquitetura/SO;



1.3. Máquina Virtual

"Write Once, Run Everywhere"

Este é o lema do Java "Escreva uma vez, rode em qualquer lugar".

Mas além de fornecer alta portabilidade, quais outros tipos de serviços são disponibilizados por uma Máquina Virtual?

- A JVM funciona como um computador, porém virtual;
- Realiza a gerência automática de:
  - Memória das aplicações Java (Garbage Collector);
  - Pilhas de execução;
  - Escalonamento de Threads;
  - Estatísticas das aplicações;
  - etc..

1.4. JVM, JRE e JDK

Mas afinal de contas, o que é necessário para executar ou construir aplicações Java?

1. JVM (Java Virtual Machine)

É apenas a máquina virtual Java, e não é possível fazer o download dela separadamente;

2. JRE (Java Runtime Environment)

Ambiente de execução Java formado pela JVM e por bibliotecas. É o bastante que um dispositivo precisa apenas para executar aplicações Java;

3. JDK (Java Development Kit)

Formada pela JRE e por ferramentas de desenvolvimento, como por exemplo, o compilador Java.

1.5. Objetivos do Java (Onde podemos usá-lo?)

Para termos uma idéia do poder do Java, vai aí alguns exemplos de projetos renomados que utilizam desta linguagem:

- A Google usa bastante o Java. Ela é a linguagem utilizada para criação de aplicativos para Android;
- Muitos jogos foram criados em Java, como por exemplo, o renomado Minecraft;
- A NASA utiliza do Java, por exemplo, para controlar seus robôs (Spirit e Opportunity) que estão explorando Marte;
- O OpenOffice, que é um pacote de aplicativos para escritório semelhante ao Microsoft Office, é feito em Java;
- E BILHÕES de dispositivos como câmeras, TV's, fornos microondas, entre outros usam Java;

1.6. IDE's de Desenvolvimento

Uma IDE (Integrated Development Environment) é uma ferramenta que facilita muito a vida do programador no desenvolvimento de uma aplicação em uma linguagem específica.

As principais IDE's para Java são:

eclipse

Apache NetBeans IDE

IJ *

** Usaremos neste curso o IntelliJ IDEA Community*
https://www.jetbrains.com/idea/download/

1.7. Meu primeiro programa "Hello Java!"

Todo programa em Java em uma IDE deve possuir 3 componentes básicos:

1. Um Método (que realiza uma função específica);
2. Uma Classe (que encapsula uma série de métodos);
3. Um Pacote (que guarda uma série de Classes);

```
package meuprimeiroprograma;

public class MeuPrimeiroPrograma {
    public static void main(String[] args) {
        System.out.println("Hello Java!");
    }
}
```

3

2

1

1.7. Meu primeiro programa "Hello Java!"

O Java é uma linguagem Case Sensitive, ou seja, diferencia letras maiúsculas de minúsculas. Algumas regrinhas importantes devem ser adotadas em relação a isso.

1. Constantes

Todas as letras são maiúsculas. Ex: MINHACONSTANTE, PI, etc.

2. Atributos e Métodos

A primeira palavra começa com letra minúscula e as palavras seguintes com letra maiúscula. Ex: minhaVariavel, meuMetodo etc.

3. Classes e Interfaces

A primeira palavra começa com letra maiúscula e as palavras seguintes também começam com letra maiúscula. Ex: MinhaClasse.

4. Pacotes

Todas as letras são minúsculas. Ex: meupacote.

1.7. Meu primeiro programa "Hello Java!"

Quando executamos um programa em Java, o sistema tem que ter um ponto de início. A JVM está programada para executar o método main.

```
public static void main(String[] args) {
  // Codigo a ser executado
}
```

Mas o que significa cada uma das partes do método? *

1. public static: moderadores de acesso;
2. void: tipo de retorno do método;
3. main: nome do método;
4. String[] args: parâmetros passados na função;

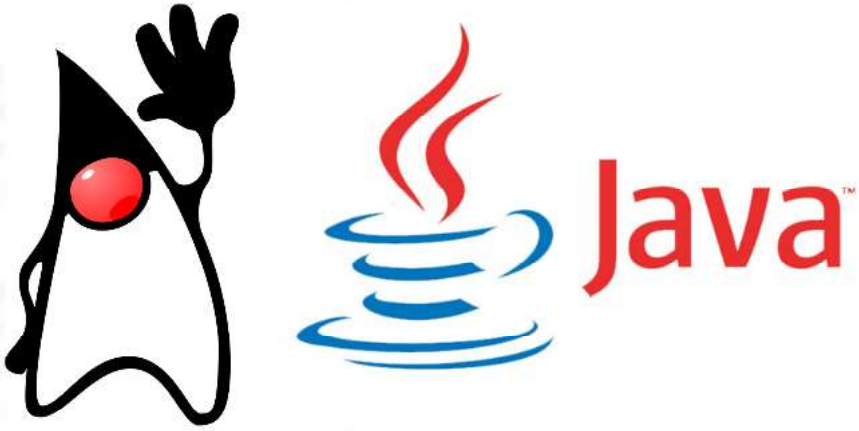
** Nas aulas futuras conseguiremos compreender mais em detalhes o significado de cada uma destas partes.*

1.7. Meu primeiro programa "Hello Java!"

Exercício 1 - HelloJava

1) Crie um programa para imprimir duas mensagens diferentes: uma usando o método print e outro usando o método println. Observe a diferença entre eles;

2) Sabendo que os caracteres \n na String representam quebras de linhas, imprima duas linhas de texto usando uma única linha com o método print;



1.8. Conhecendo alguns shortcuts do IntelliJ;

Como discutimos, o IntelliJ é uma IDE que facilita, e MUITO, a vida do programador. Por exemplo, ela fornece uma série de atalhos (ou shortcuts) que são bastante utéis.

1.psvm + enter
Cria o método main;

2.fori + enter
Cria a estrutura do for;

3.ifn + enter
Cria a estrutura do if;

4.do + enter
Cria a estrutura do do-while;

5. sout + enter
Comando para escrever algo;

6.Ctrl + f
Procura um texto no código;

7.Ctrl + d
Duplica uma linha selecionada;

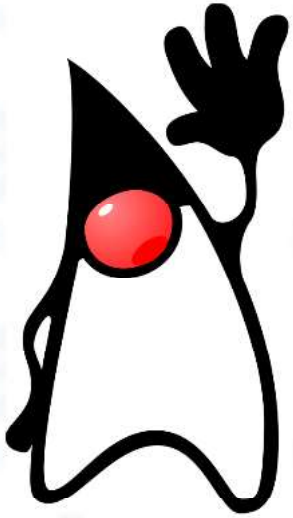
8.Ctrl + y
Exclui uma linha selecionada;

9.Ctrl + shift + F10
Executa a classe principal;

10.Ctrl + alt + L
Identa/Organiza o código fonte;

A lista completa de shortcuts oferecidos pode ser acessada em:
https://www.jetbrains.com/help/idea/mastering-keyboard-shortcuts.html

FIM
DO
CAPÍTULO 1



Próximo Capítulo

Tipos Primitivos e
Controle de Fluxo