

# Introdução à linguagem Java

Prof. Hugo de Paula



PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS  
Departamento de Ciência da Computação

# Sumário

- 1 Linguagem Java
  - Histórico da linguagem Java
  - Ranking do Java
  - Características do Java
  - Estrutura de uma aplicação Java



# Histórico do Java

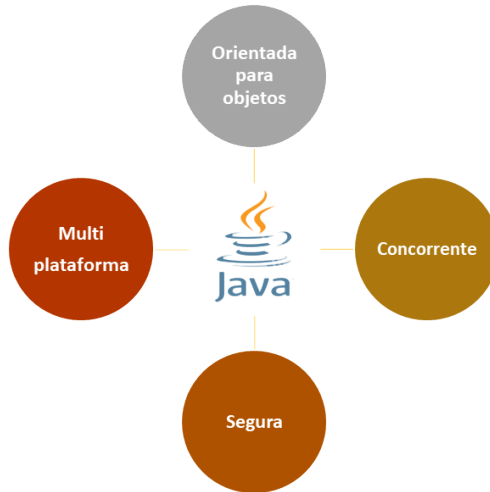
- Linguagem orientada a objetos.
- Desenvolvida em 1991 por James Gosling.
- 1a versão lançada 1995.
- Código fonte e objeto portáveis para diversas arquiteturas e sistemas operacionais.
- Linguagem de propósito geral derivada do C++.
- Linguagem compilada e depois interpretada:
  - *Bytecode* – ling. assembly de uma máquina hipotética.

*“Linguagem simples, orientada por objetos, distribuída, interpretada, robusta, segura, independente de arquitetura, portátil, de alta performance, multi-threaded e dinâmica.”*

SUN Microsystems, Maio de 1995



# Características do Java





# Ranking Java

**Tabela:** Resumo do Índice Tiobe

<http://www.tiobe.com/tiobe-index/>

fev/20	Ling. Programação	Ranking
1	Java	17,358%
2	C	16,766%
3	Python	9,345%
4	C++	6,164%
5	C#	5,927%

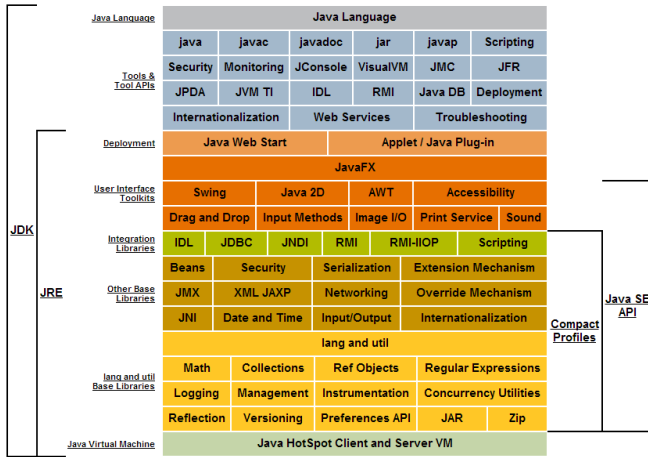


# Java é mais simples que a linguagem C++

- Semântica de referência ao invés de ponteiros.
- Não permite variáveis globais, apenas atributos estáticos.
- Coletor de lixo gerencia memória (sem DELETE).
- Não possui arquivos de cabeçalho (*header*).
- Sem sobrecarga de operadores, apenas de métodos.
- Sem herança múltipla de classes, mas com herança múltipla de interfaces.



# Diagrama conceitual do Java



<https://docs.oracle.com/javase/8/docs/>



# Compilando e executando Java

- Código-fonte: `.java`
- Compilador:
  - `javac <nomearq.java>`
- Arquivos objeto compilados: `.class`
- Para executar uma aplicação:
  - `java <nomearq>`
- Para executar um *applet*:
  - `appletviewer <nomearq.html>`
  - ou em algum navegador compatível.





# Estrutura de uma aplicação Java

- Classes são escritas em arquivos `.java`.
- Um arquivo `.java` pode conter diversas classes, mas apenas uma será pública e estará visível ao resto da aplicação.
- A classe pública de um arquivo `.java` deve ter o mesmo nome do arquivo `.java`.
- As classes compiladas (arquivos `.class`) devem estar em diretórios conhecidos do Java (variável de ambiente `CLASSPATH`).
- O diretório `<DIRETORIO_JDK>/JRE/CLASSES` é o local padrão para localização de classes.



# Estrutura de uma aplicação Java

- Classes são agrupadas em pacotes, definido pela cláusula **package**.
- Pacotes são conjuntos de classes, interface e subpacotes relacionados.
- Pacotes são úteis porque:
  - Agrupam interfaces e classes relacionadas
  - Criam espaço de nomes para Interfaces e classes.
  - outro nível de encapsulamento.
- Para utilizar um pacote, usa-se a cláusula **import**; semelhante ao **#include** da linguagem C.
- Pacotes também são encontrados através do **CLASSPATH**, e são considerados subdiretórios do caminho.



# Estrutura de uma aplicação Java

- Pacotes podem ser aninhados.
- Exemplo:
  - `package adaptativo.RedesNeurais;`
  - `package adaptativo.Genetico;`
- O aninhamento é puramente semântico.
- O pacote `adaptativo` é diferente do pacote `adaptativo.genetico` que é diferente do pacote `adaptativo.RedesNeurais`.
- As classes do pacote `Genetico` devem estar em um sub-diretório `Genetico` do diretório `adaptativo`.
- Pacotes podem ser referenciados diretamente sem utilizar a cláusula `import`, bastando chamar diretamente pelo nome do pacote.



# Aplicações e Applets

- Aplicações console:
  - Rodam sob a JVM, que faz a tradução direta para o sistema operacional.
  - Utilizam apenas as interfaces de entrada e saída padrão Java: `java.System`, `java.io`, `java.lang`, `java.util`.
- Aplicações gráficas (rodam em janelas):
  - Devem utilizar bibliotecas gráficas contidas em alguma interface da JFC (Java Foundation Classes): AWT, Swing, Java 2D, etc...
  - São orientadas a eventos.
- *Applets*:
  - Aplicações leves que rodam no contexto do navegador web.
  - *Applets* são sempre pequenas aplicações gráficas. Herdam de `java.Applet`



# Aplicações e Applets: Exemplo

- Aplicação console:

```
/* AloMundo.java */
```

```
class AloMundo {  
    public static void main(String args[]) {  
        System.out.println("Alo Mundo!");  
    }  
}
```

- Colocado no arquivo AloMundo.java
- Java é sensível a maiúsculas e minúsculas.
- Edite, compile e rode o arquivo AloMundo.java



# Aplicações e Applets: Exemplo

- Applet:

```
/* AloMundoApplet.java */
```

```
import java.awt.Graphics;
```

```
public class AloMundoApplet extends java.applet.Applet {
```

```
    public void paint(Graphics g) {  
        g.drawString("Alo Mundo!", 5, 25);  
    }
```

```
}
```

- Colocado no arquivo AloMundoApplet.java
- Edite, compile e rode o arquivo AloMundoApplet.java



# Aplicações e Applets: Exemplo

- Rodando Applets
- O compilador gera um arquivo `AloMundoApplet.class` que deve ser executado dentro de uma página HTML.
- Crie o arquivo `AloMundoApplet.html` e abra-o em um navegador:

```
<HTML>
<HEAD>
<TITLE>Alo todo mundo!!!</TITLE>
</HEAD>

<BODY>
  <P>O applet Java diz :
  <APPLET CODE="AloMundoApplet.class" WIDTH=150 HEIGHT=25>
  </APPLET>
</BODY>
</HTML>
```



# Aplicações e Applets: Exemplo

- Aplicações Gráficas:

```
/* AloMundoGraphic.java */  
import javax.swing.JOptionPane;  
  
public class AloMundoGraphic {  
    public static void main(String args[]) {  
        JOptionPane.showMessageDialog(null, "Alo Mundo!" );  
        System.exit(0); // encerra a aplicação  
    }  
}
```

- Edite, compile e rode o arquivo AloMundoGraphic.java.
- Para fixar o conceito de pacotes, adicione a linha
- **package** AloPkg;
- No início no arquivo e tente compilar e executar o programa.





# Lançando programas Java

- Em aplicações, tudo começa pelo método `main`.
- **`public static void main(String args[])`**
- `args[]` é correspondente ao `argv[]` do C, exceto que `args[0]` é equivalente ao `argv[1]`
- `main` não retorna um valor, apesar de que a JVM pode capturar códigos de saída: `System.exit(0);`
- Em *Applets* não há métodos `main`. Existem 3 métodos que rodam automaticamente (em ocasiões distintas) que constituem o disparo do programa: `start`, `init` e `paint`.