



Curso de **Java8** para **Web**

Professor
Antonio Benedito Coimbra Sampaio Jr

abc  | Treinamentos

www.abctreinamentos.com.br

Primeira Disciplina

JAVA 8 - Fundamentos Teóricos e Orientação a Objetos

- **UNIDADE 1: Introdução à Tecnologia Java**
- **UNIDADE 2:** Introdução à Sintaxe Java
- **UNIDADE 3:** Programação Orientada a Objetos em Java (Parte I)
- **UNIDADE 4:** Programação Orientada a Objetos em Java (Parte II)

UNIDADE 1

INTRODUÇÃO À TECNOLOGIA JAVA


Configuração do Ambiente Java


Obtendo o JDK 8

Oracle Technology Network > Java > Java SE > Downloads

Overview Downloads Documentation Community Technologies Training

Java SE Downloads


Java Platform (JDK) 8u11
[DOWNLOAD](#)


JDK 8u11 & NetBeans 8.0
[DOWNLOAD](#)

Java Platform, Standard Edition

Java SE 8u11

This release includes important security fixes. Oracle strongly recommends that all Java SE 8 users upgrade to this release.
[Learn more](#)

- Installation Instructions
- Release Notes
- Oracle License
- Java SE Products
- Third Party Licenses
- Certified System Configurations
- Readme Files
 - JDK ReadMe

JDK
[DOWNLOAD](#)

Server JRE
[DOWNLOAD](#)


JRE
[DOWNLOAD](#)

Java SDKs and Tools

- [Java SE](#)
- [Java EE and Glassfish](#)
- [Java ME](#)
- [Java Card](#)
- [NetBeans IDE](#)
- [Java Mission Control](#)

Java Resources

- [Java APIs](#)
- [Technical Articles](#)
- [Demos and Videos](#)
- [Forums](#)
- [Java Magazine](#)
- [Java.net](#)
- [Developer Training](#)
- [Tutorials](#)
- [Java.com](#)


NEW!
Get it now for FREE!
[Subscribe Today](#)

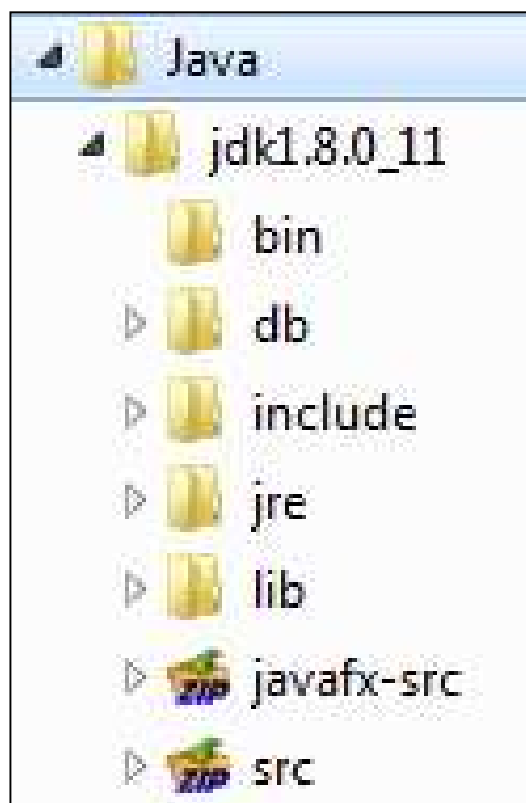
<http://www.oracle.com/technetwork/java/javase/downloads/index.html>

Instalando o JDK 8

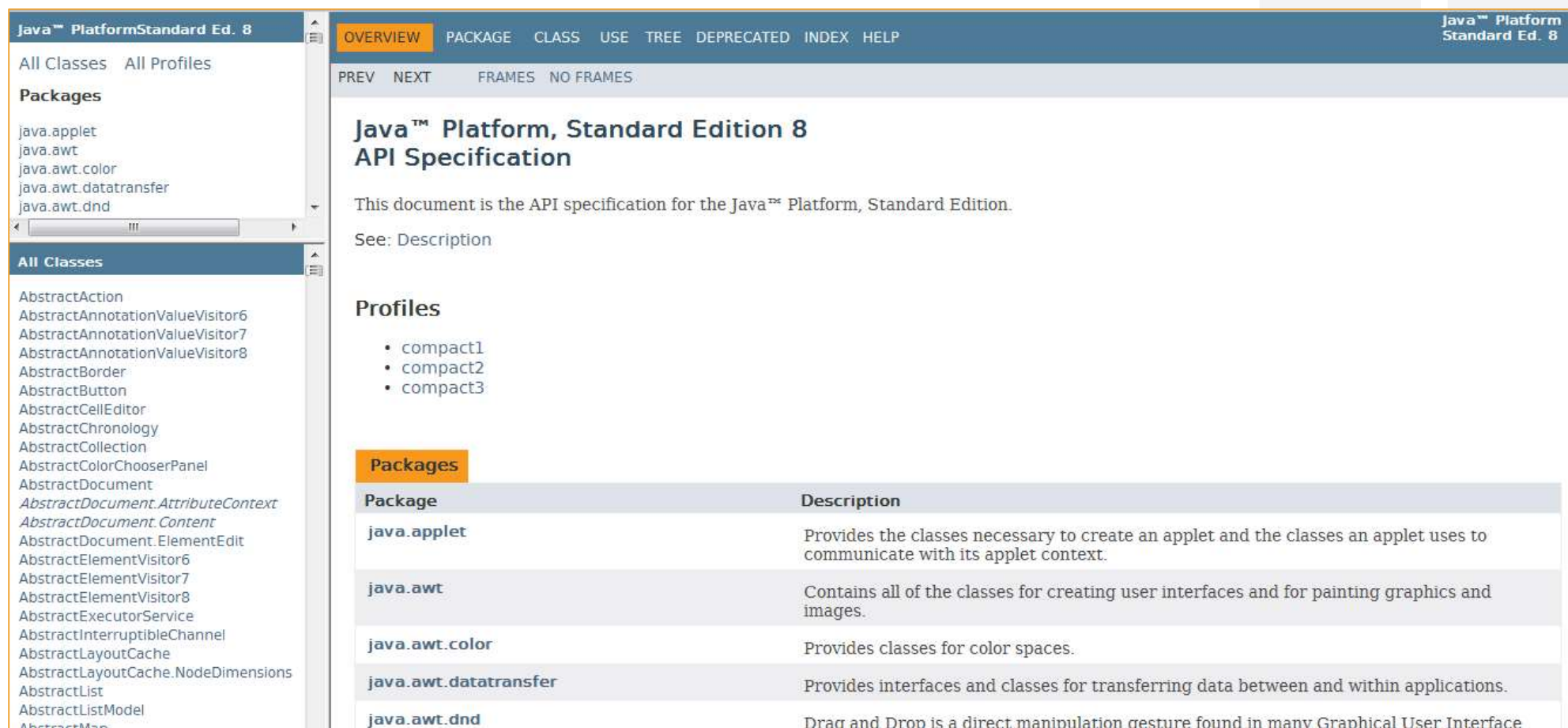


Checando a Instalação

```
C:\Users\Antonio Sampaio>java -version  
java version "1.8.0_11"  
Java(TM) SE Runtime Environment (build 1.8.0_11-b12)  
Java HotSpot(TM) 64-Bit Server VM (build 25.11-b03, mixed mode)
```



Documentação da API



Java™ Platform Standard Ed. 8

OVERVIEW PACKAGE CLASS USE TREE DEPRECATED INDEX HELP

PREV NEXT FRAMES NO FRAMES

Java™ Platform, Standard Edition 8 API Specification

This document is the API specification for the Java™ Platform, Standard Edition.

See: Description

Profiles

- compact1
- compact2
- compact3

Packages

Package	Description
java.applet	Provides the classes necessary to create an applet and the classes an applet uses to communicate with its applet context.
java.awt	Contains all of the classes for creating user interfaces and for painting graphics and images.
java.awt.color	Provides classes for color spaces.
java.awt.datatransfer	Provides interfaces and classes for transferring data between and within applications.
java.awt.dnd	Drag and Drop is a direct manipulation gesture found in many Graphical User Interface

<http://docs.oracle.com/javase/8/docs/api/>

- Essa documentação descreve todas as classes Java e como elas devem ser usadas.

Ambiente de Desenvolvimento

DEFINIÇÃO

- Ambiente Integrado de Desenvolvimento ou IDE (*Integrated Development Environment*) é um programa de computador que reúne ferramentas de apoio (compilador, interpretador, debug, etc.) para o desenvolvimento de software.

- **ECLIPSE**

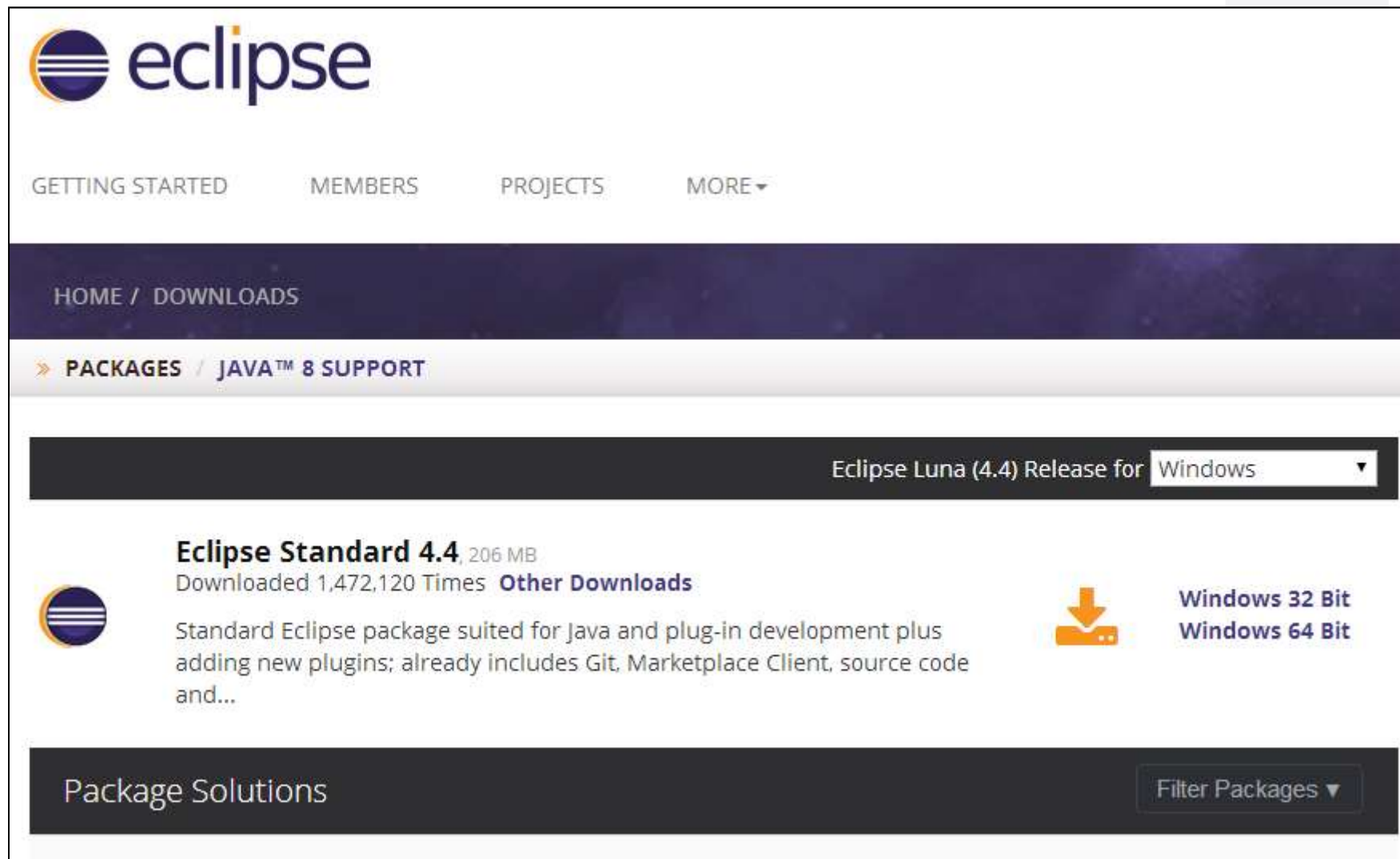
NOSSA ESCOLHA

- NETBEANS
- INTELLIJ
- JCREATOR

- O Eclipse possui suporte para o Java 8 a partir da versão Luna (4.4).

Ambiente de Desenvolvimento

ECLIPSE



The screenshot shows the Eclipse website's download page. At the top is the Eclipse logo and a navigation bar with links: GETTING STARTED, MEMBERS, PROJECTS, and MORE. Below this is a breadcrumb trail: HOME / DOWNLOADS. The main heading is PACKAGES / JAVA™ 8 SUPPORT. A dropdown menu shows 'Eclipse Luna (4.4) Release for Windows'. The main content area features the Eclipse Standard 4.4 package, which is 206 MB and has been downloaded 1,472,120 times. It includes a description: 'Standard Eclipse package suited for Java and plug-in development plus adding new plugins; already includes Git, Marketplace Client, source code and...'. There is a download icon and links for 'Windows 32 Bit' and 'Windows 64 Bit'. At the bottom, there is a 'Package Solutions' section and a 'Filter Packages' dropdown.

eclipse


GETTING STARTED MEMBERS PROJECTS MORE ▾


HOME / DOWNLOADS

» PACKAGES / JAVA™ 8 SUPPORT

Eclipse Luna (4.4) Release for Windows ▾

Eclipse Standard 4.4, 206 MB
Downloaded 1,472,120 Times [Other Downloads](#)

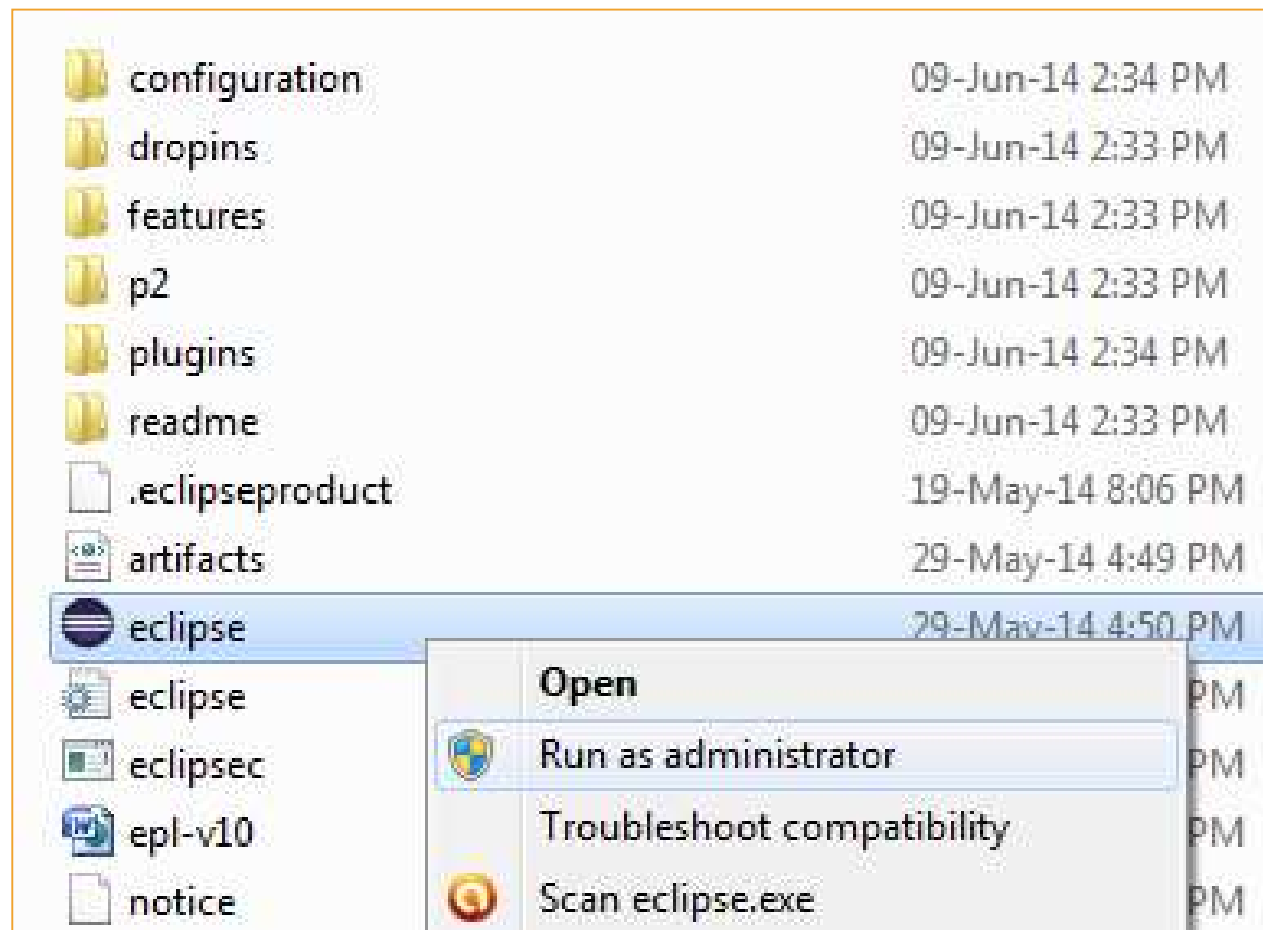
 Standard Eclipse package suited for Java and plug-in development plus adding new plugins; already includes Git, Marketplace Client, source code and...

 [Windows 32 Bit](#)
[Windows 64 Bit](#)

Package Solutions [Filter Packages ▾](#)

Ambiente de Desenvolvimento

ECLIPSE



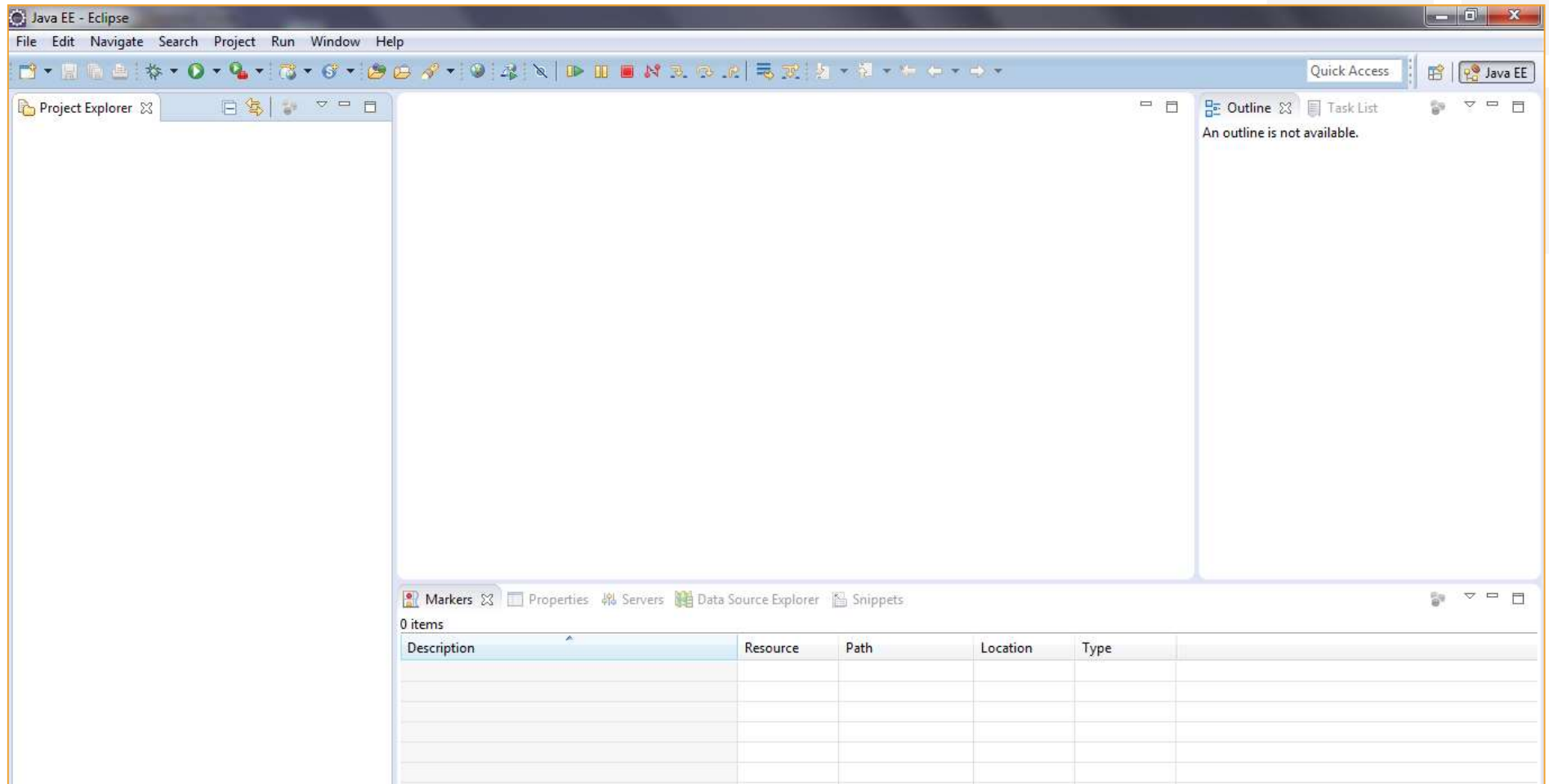
Ambiente de Desenvolvimento

ECLIPSE



Ambiente de Desenvolvimento

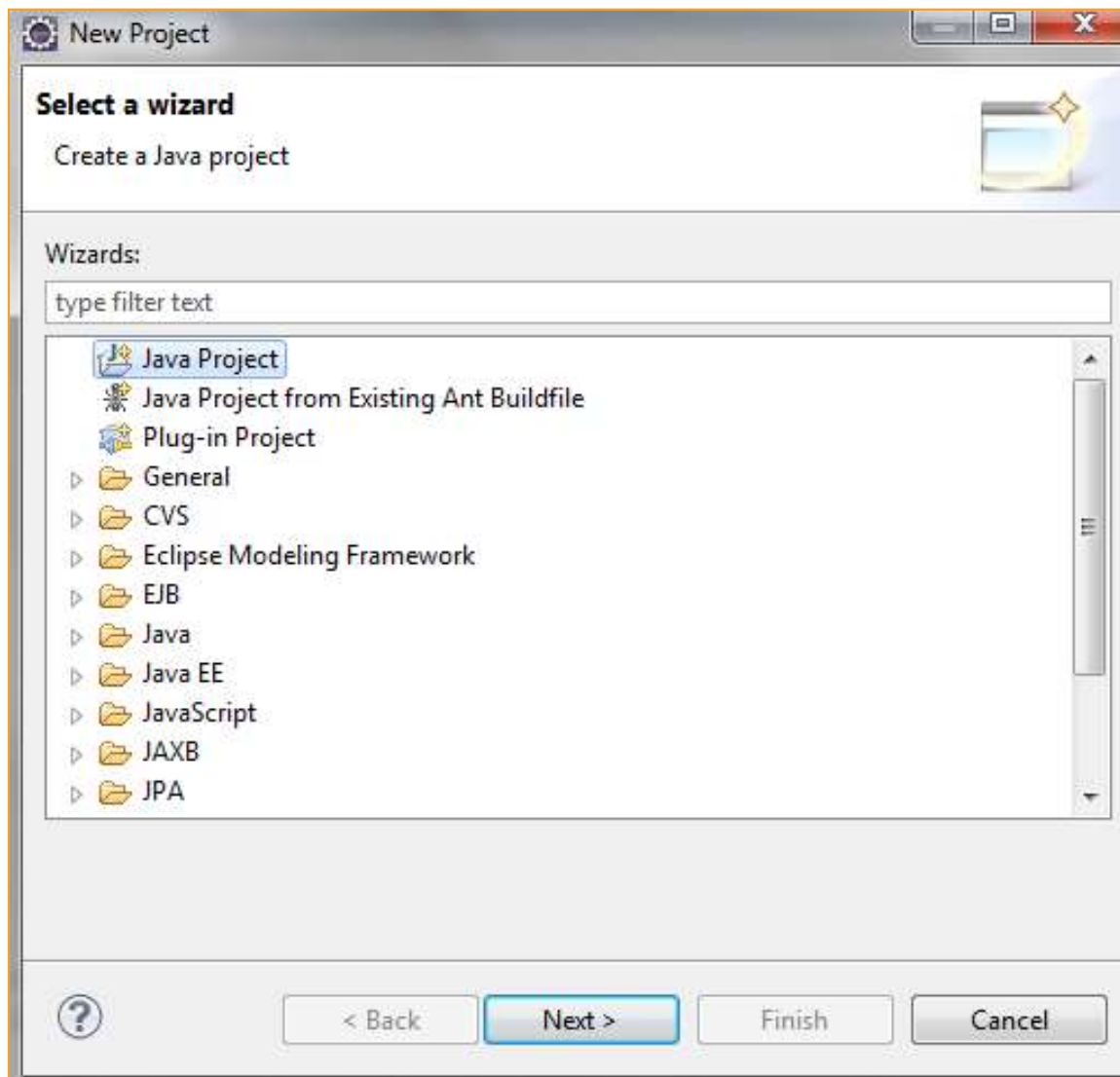
ECLIPSE



O meu Primeiro Código em Java

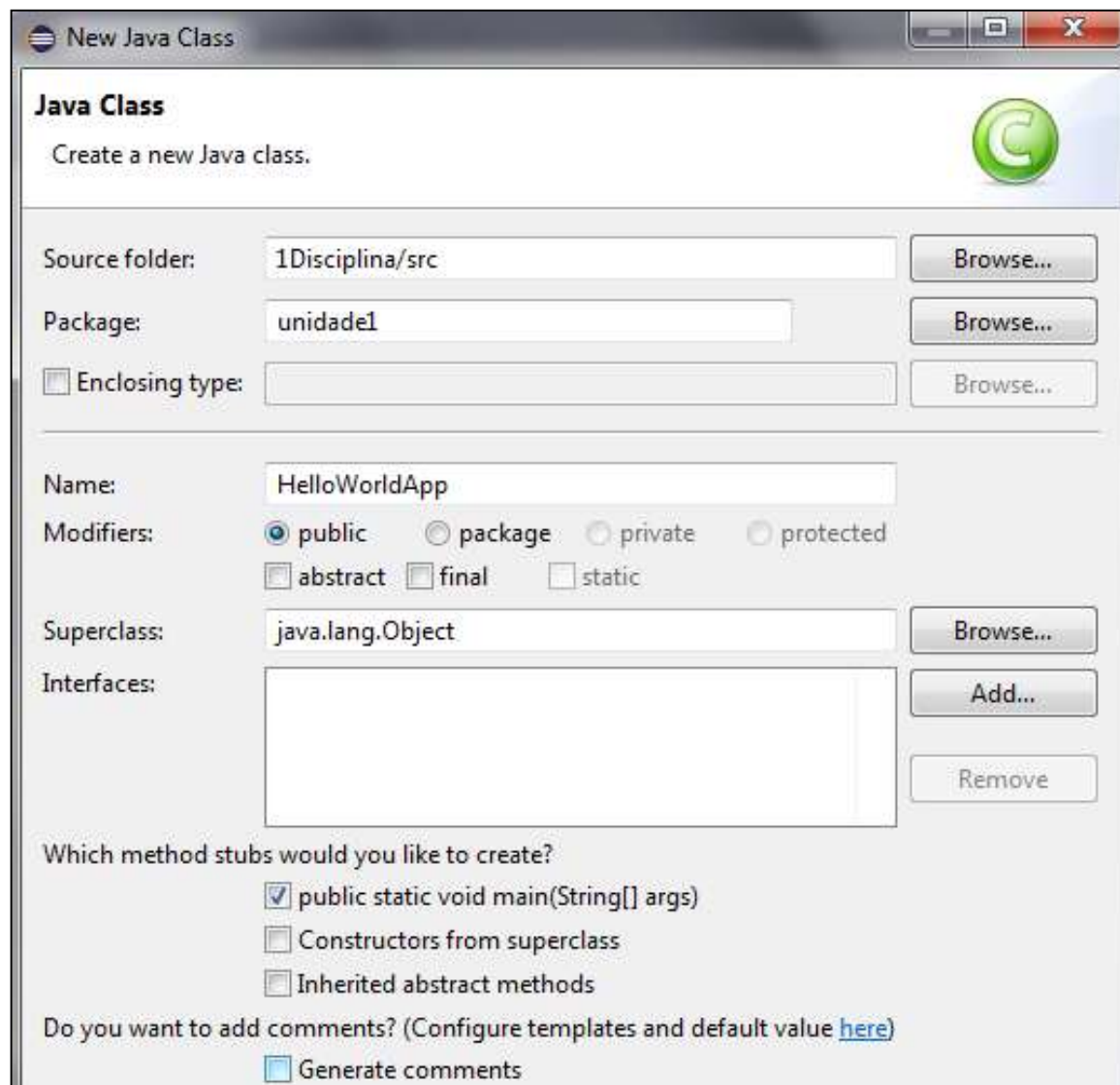
Primeiro Programa JAVA

- Inicialmente, é necessário criar um Projeto Java no Eclipse.



Primeiro Programa JAVA

- Depois, deve-se criar a Classe Java.

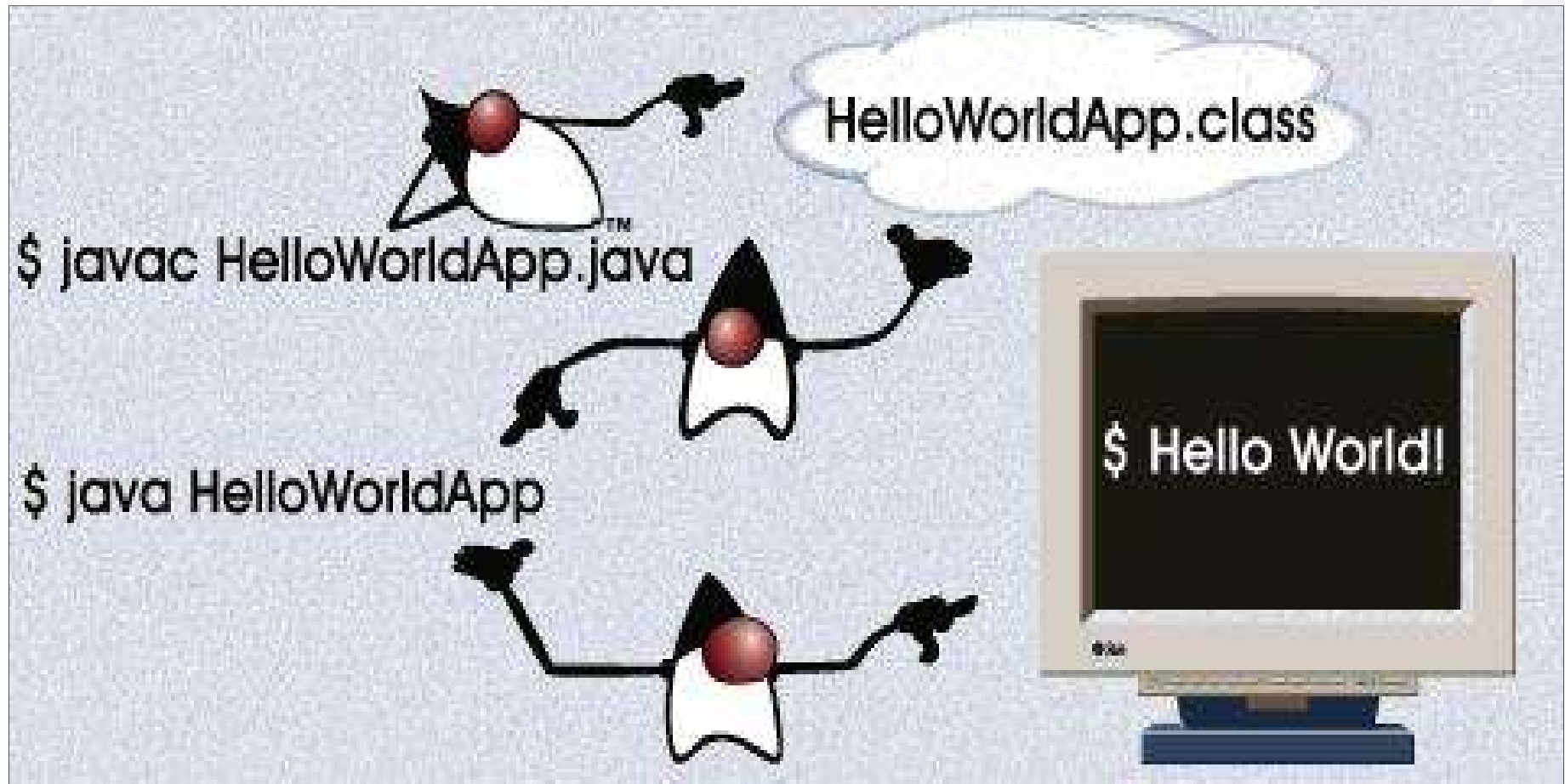


Primeiro Programa JAVA

- Digite o programa “HelloWorldApp.java” apresentado abaixo no editor do Eclipse.

```
/** Primeiro Programa Java */  
package unidade1;  
  
class HelloWorldApp  
{  
    public static void main(String arg[])  
    {  
        System.out.println("Hello World!");  
    }  
}
```

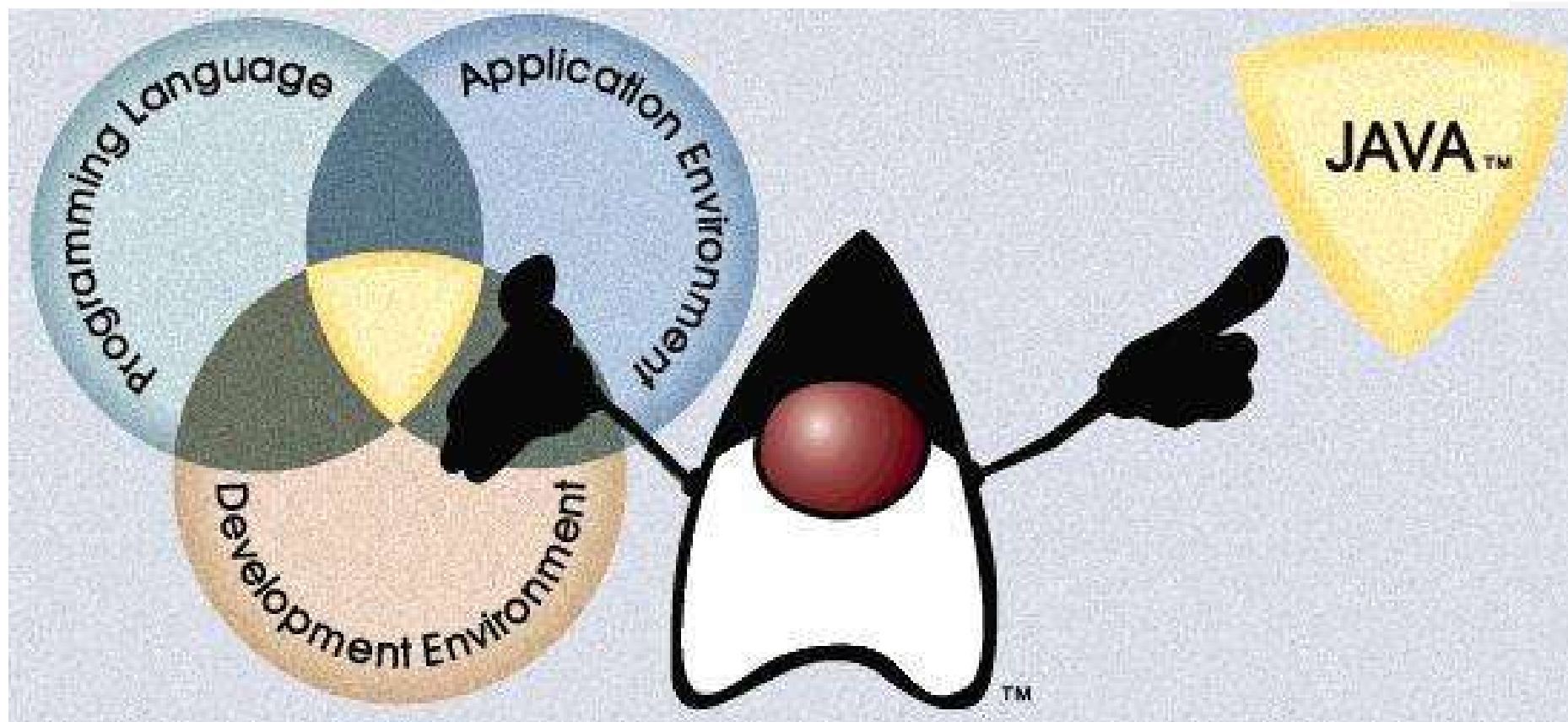
Primeiro Programa JAVA



O que é o Java?

Conceito do JAVA

- Java é uma Plataforma Tecnológica para a Criação de Software.



Conceito do JAVA

JAVA é:

- Uma **linguagem de programação** orientada a objetos;
- Uma **coleção de APIs** (classes, componentes, frameworks) para o desenvolvimento de aplicações multiplataforma;
- Um **ambiente de execução** presente em browsers, mainframes, SOs, dispositivos móveis, cartões inteligentes, eletrodomésticos.

JAVA APIs

Java EE



**Enterprise
Software/
Servers**

Java SE



**Applications/
Desktop
Software**

Java ME & Card



**Client
Deployed
Services**

JAVA APIs



Java Platform, Standard Edition (Java SE)

O Java SE foi projetado para permitir o desenvolvimento de aplicativos seguros, portáteis e de alto desempenho para a maior variedade possível de plataformas de computação. Disponibilizando aplicativos em ambientes heterogêneos, as empresas podem agilizar a produtividade do usuário final, a comunicação e a colaboração—além de reduzir drasticamente o custo de propriedade de aplicativos tanto de empresas quanto de clientes.

- Java Platform, Standard Edition (Java SE)
- Oracle Java SE Support
- Oracle Java SE Advanced & Oracle Java SE Suite
- Oracle JRockit
- Java FX



Java Platform, Enterprise Edition (Java EE)

O Java Platform, Enterprise Edition (Java EE) é o padrão do setor para computação Java empresarial. Com novos recursos que melhoram o suporte a HTML5, aumentam a produtividade do desenvolvedor e melhora ainda mais como as demandas corporativas podem ser atendidas, o Java EE 7 permite que os desenvolvedores escrevam menos códigos, tenham suporte melhor para os mais recentes aplicativos da Web e estruturas e recebam acesso à escalabilidade melhorada e funcionalidades mais avançadas e simples.

- Java Platform, Enterprise Edition (Java EE)

<http://www.oracle.com/br/technologies/java/overview/index.html?ssSourceSiteId>

JAVA APIs



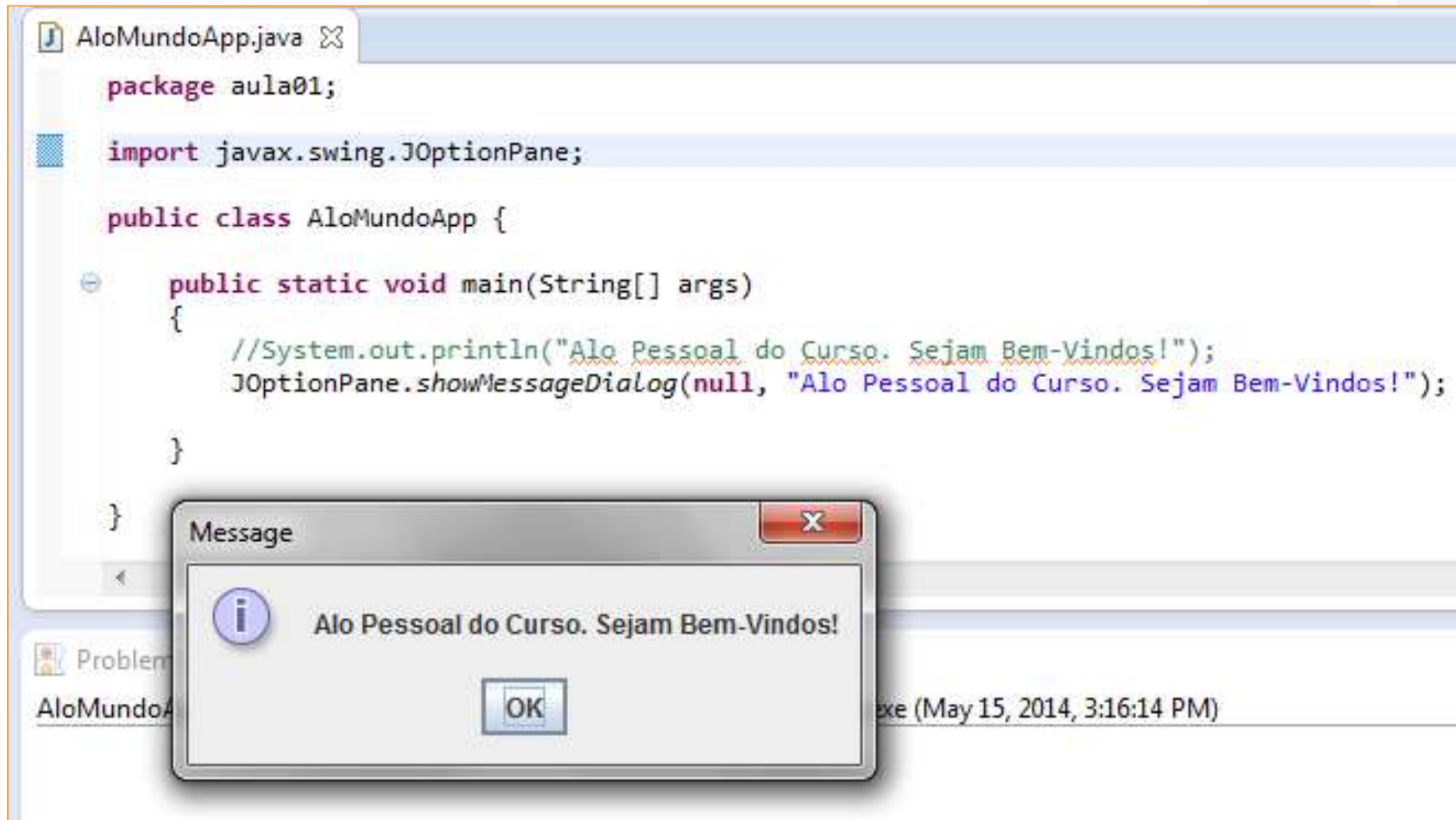
Java for Mobile Devices

O Java Platform, Micro Edition (Java ME) é usado por um grande número de desenvolvedores de telefone celular Java, operadoras e OEMs para criar produtos telefônicos de destaque ao redor do mundo. A Oracle é líder no oferecimento de tecnologia para telefone celular encontrada em mais de três bilhões de dispositivos, por enquanto.

- Java for Mobile Devices
- Oracle Java Wireless Client
- Kit de Ferramentas de Interface do Usuário Leve
- Java ME SDK

<http://www.oracle.com/br/technologies/java/overview/index.html?ssSourceSiteId>

Exemplo JAVA SE



Exemplo JAVA EE

```
formFaleConoscoEnviar.jsp
<jsp:useBean id="mensagem" scope="page" cla
<%
String path = request.getContextPath();
String basePath = request.getScheme()+"://"

    try {
        String Email = (String)request.
        String Nome = (String)request.g
        String Assunto = (String)reques
        String Mensagem = (String)reque

        // Conteúdo, Assunto do Email,
        try {
            mensagem.Envial(Nome, Email
            session.setAttribute("statu
            response.sendRedirect("resu
        }
        catch (Exception e) {
            session.setAttribute("statu
            response.sendRedirect("resu
        }
    }
%>

<html>
<head>
    <base href="<%=basePath%>">

    <title>Enviando a sua mensagem...</titl
    <meta http-equiv="pragma" content="no-c
    <meta http-equiv="cache-control" conten
    <meta http-equiv="expires" content="0">
```

Fale Conosco

Nome: E-mail: Assunto: Mensagem:

Exemplo JAVA ME

```
private SampleBarCodeItem bc;
public void startApp()
{
    if (midletPaused)
    {
        resumeMIDlet();
    }
    else
    {
        initialize();
        startMIDlet();

        // Initialize the custom item
        bc = new SampleBarCodeItem("Aspose-
this.form.append(bc);
        bc.setPreferredSize(300, 200);
    }
    midletPaused = false;
}
```



Histórico

Histórico

INÍCIO:

- Projeto “Green”:
 - 1991 - James Gosling e outros engenheiros da Sun
 - Objetivo: criar aplicações para controlar produtos eletrônicos
 - Linguagem “Oak” baseada em C++
- As tentativas frustradas:
 - “*7” - controle remoto inteligente - sem patrocínio
- História Completa no site
- <http://www.oracle.com/technetwork/java/javase/overview/javahistory-index-198355.html>

EM 1995 A SUN LANÇA O JAVA:

- Programação na Web com Java Applet

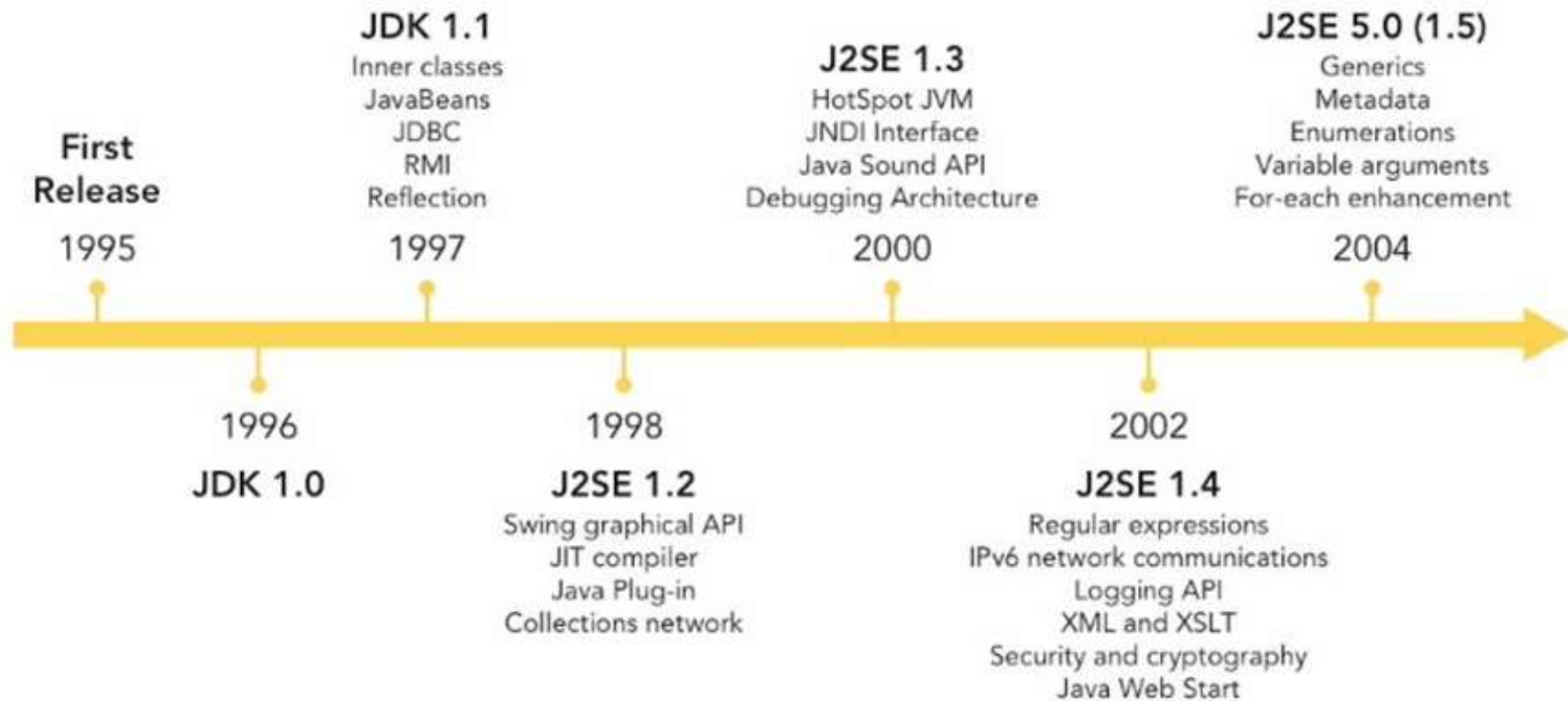
“Since 1995, Java has changed our world . . . and our expectations..”

Histórico

1995 – A INFLUÊNCIA DA WEB: *“Write Once, Run Anywhere”*

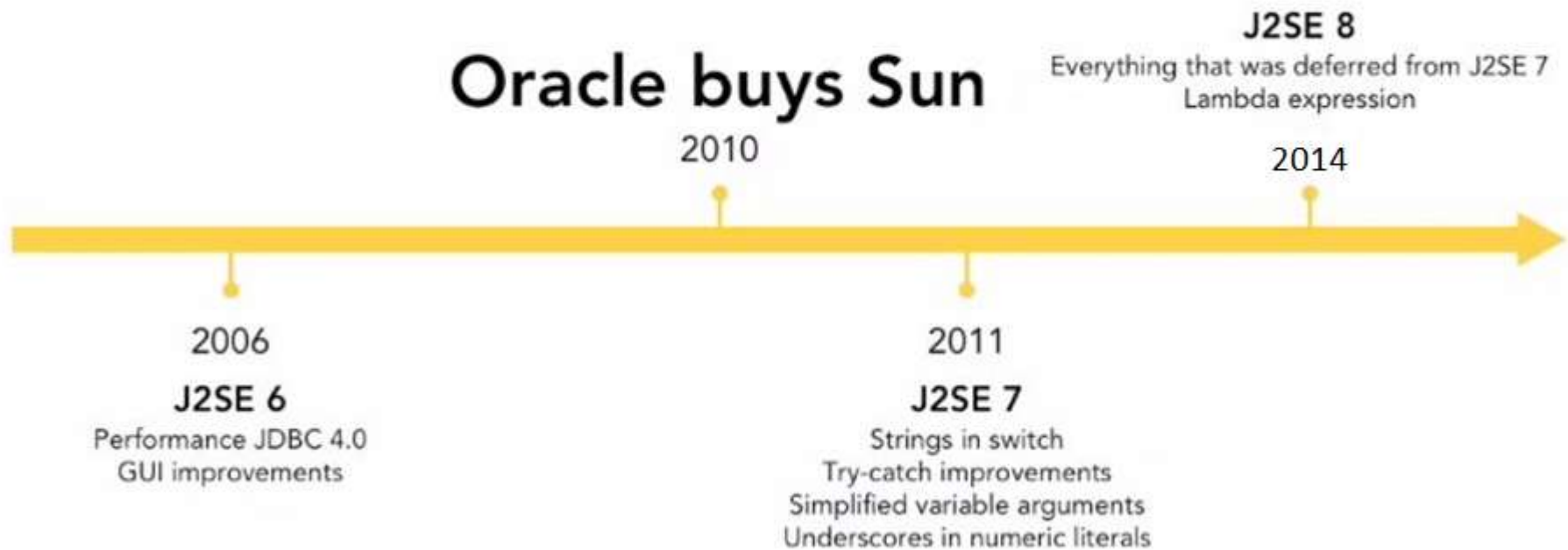
- Netscape Navigator 2.0 compatível com Java 1.0 (independência de plataforma, confiabilidade, segurança, processamento em tempo real, etc.)
- Desde então, cinco grandes revisões:
 - Java Development Kit (JDK) 1.0/1.0.2
 - Java Development Kit (JDK) 1.1/1.1.8
 - Java 2 Platform (Java 2 SDK e JRE 1.2, 1.3, 1.4)
 - Java Tiger (Java SDK 1.5)
 - **Java SDK 8 (março 2014)**

Histórico



<http://javatutorial-for-beginners.blogspot.com.br/>

Histórico



<http://javatutorial-for-beginners.blogspot.com.br/>

Histórico

Java 8 não pode ser instalado no Windows XP

No Windows XP support on Java 8



As of April 8, 2014 Microsoft stopped supporting Windows XP and therefore it is no longer a supported platform. XP users will be unable to install Java 8. Windows users must upgrade to Windows Vista or later to install Java 8.

Últimas Versões do Java

JRE Family Version	JRE Security Baseline (Full Version String)
8	1.8.0_11
7	1.7.0_55
6	1.6.0_75
5.0	1.5.0_65

← Versão do Curso

JDK 7 x JDK 8

- A implementação da **JSR 335: *Lambda Expressions for the Java™ Programming Language*** provocou uma das maiores atualizações em termos de código fonte, tanto na JVM como no JDK (compilador e bibliotecas).
- Para se ter uma idéia, ao se enumerar as novas classes dos pacotes **java.util** e **java.util.concurrent** que dão suporte à JSR 335 no JDK 8, chega-se ao expressivo número de 283 classes. Se também for somado as classes dos novos pacotes **java.util.function** e **java.util.stream**, chega-se a um total de 896 classes a mais no JDK 8 em relação ao JDK 7, como pode ser visto na Tabela abaixo.

Pacote	Classes (JDK8/JDK7)	EL (JDK 8)	Total (JDK8/JDK 7)
java.util	428/299	43	471/299
java.util.concurrent	310/202	3	313/202
java.util.function	43/0	38	81/0
java.util.stream	332/0	200	532/0

Evolução

Evolução

JCPs (Java Community Process - www.jcp.org)

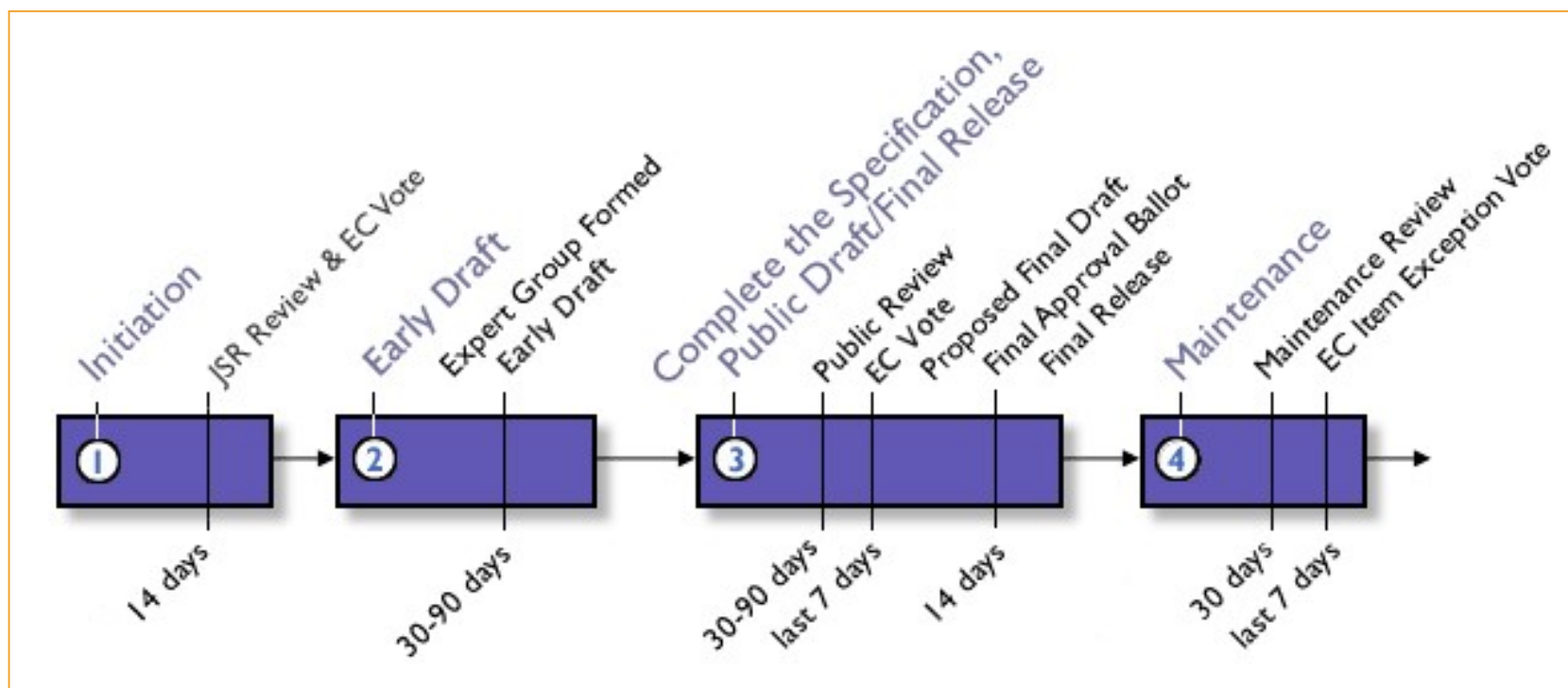
- Processo criado pela Sun e pelos primeiros licenciados da tecnologia Java como forma de evoluir e manter a tecnologia aberta e disponível para todos;
- Mais de 600 empresas participam;
- Definem as JSRs (Java Specification Request) que especificam novas tecnologias Java.



Evolução

JSRs (Java Specification Requests)

- 927 especificações (jcp.org/en/jsr/all);
- Exemplos:
 - JSR-174 (Monitoring and Management Specification for the Java Virtual Machine)
 - JSR-335 (Lambda Expressions for the Java Programming Language)



Presente e Futuro

Presente e Futuro

INTERNET DAS COISAS (IoT – Internet of Things)

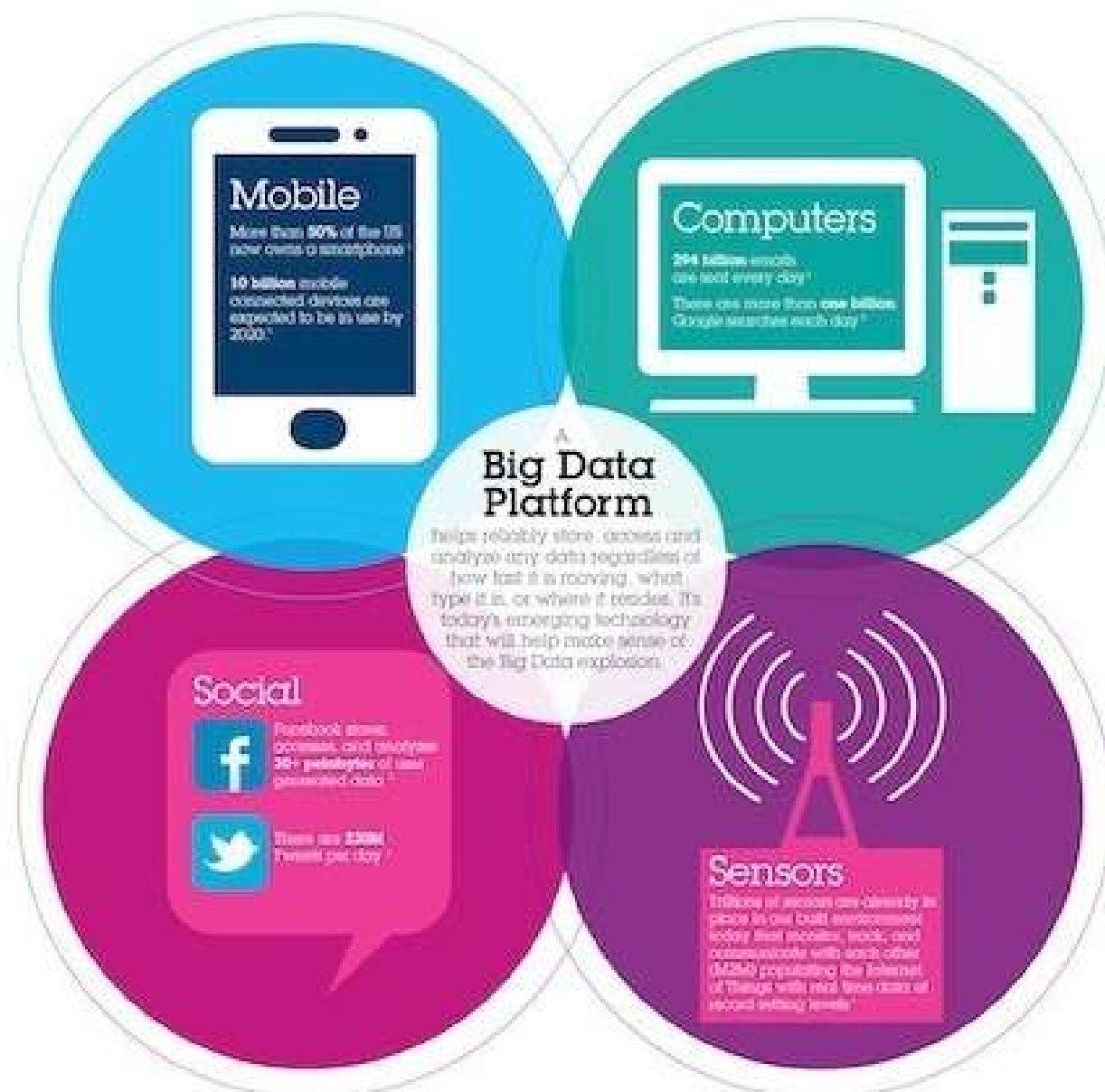
- “A Internet das coisas é uma revolução tecnológica que representa o futuro da computação e da comunicação e cujo desenvolvimento depende da inovação técnica dinâmica em campos tão importantes como os sensores wireless e a nanotecnologia.” [Wikipedia]

BIG DATA

- É o conjunto de soluções tecnológicas capaz de lidar com dados digitais em volume, variedade e velocidade inéditos até hoje. Na prática, a tecnologia permite analisar qualquer tipo de informação digital em tempo real, sendo fundamental para a tomada de decisões.

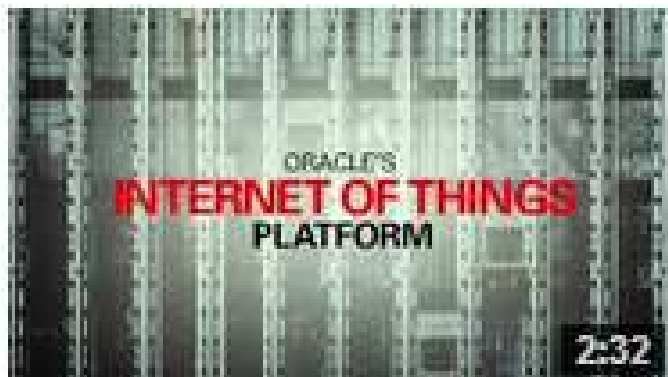
Presente e Futuro

BIG DATA



<http://www.business2community.com/big-data/big-data-big-insights-for-social-media-with-ibm-0501158#!FUJl6>

Vídeos IoT e BIGDATA



The Internet of Things: Managing the Complexity

https://www.youtube.com/watch?v=oFRyl7w_zD8



Big Data at Work

<https://www.youtube.com/watch?v=rCRi6dl4hZo>

Características do Java

Introdução ao JAVA

VANTAGENS COMO LINGUAGEM DE PROGRAMAÇÃO:

- Orientada a Objetos (O.O)
- código sem bugs: mais fácil em Java que em C++
- desalocação manual de memória não existe
- arrays “verdadeiros” + ausência de aritmética de ponteiro
- substituição de herança múltipla por interface
- independência de plataforma!

Introdução ao JAVA

SIMPLES

- Sintaxe similar a C / C++
- Não possui os recursos “perigosos”, desnecessários ou pouco usados:
 - Aritmética de ponteiros (`*--pt = vet+5`)
 - Estruturas (**struct**)
 - Definição de tipos (**typedef**)
 - Pré-processamento (**#define**)
 - Liberação explícita de memória (**free**)
- Eliminação de 50% dos erros mais comuns
- Interpretadores pequenos (256 Kb)

Introdução ao JAVA

ORIENTADA A OBJETOS

- Utiliza tecnologia de objetos similar a de C++, com alguns acréscimos e simplificações.

Introdução ao JAVA

DISTRIBUÍDA

- Implementa os protocolos da arquitetura TCP/IP, tais como: HTTP, SMTP, TCP, UDP, RTP, FTP, etc.

Introdução ao JAVA

ROBUSTA

- Possui checagem em tempo de compilação e execução.
- Gerenciamento automático de memória (coletor de lixo ou “garbage collector”).
- Ausência de recursos “perigosos”.
- Extensiva verificação do código.

Introdução ao JAVA

ROBUSTA – COLETOR DE LIXO

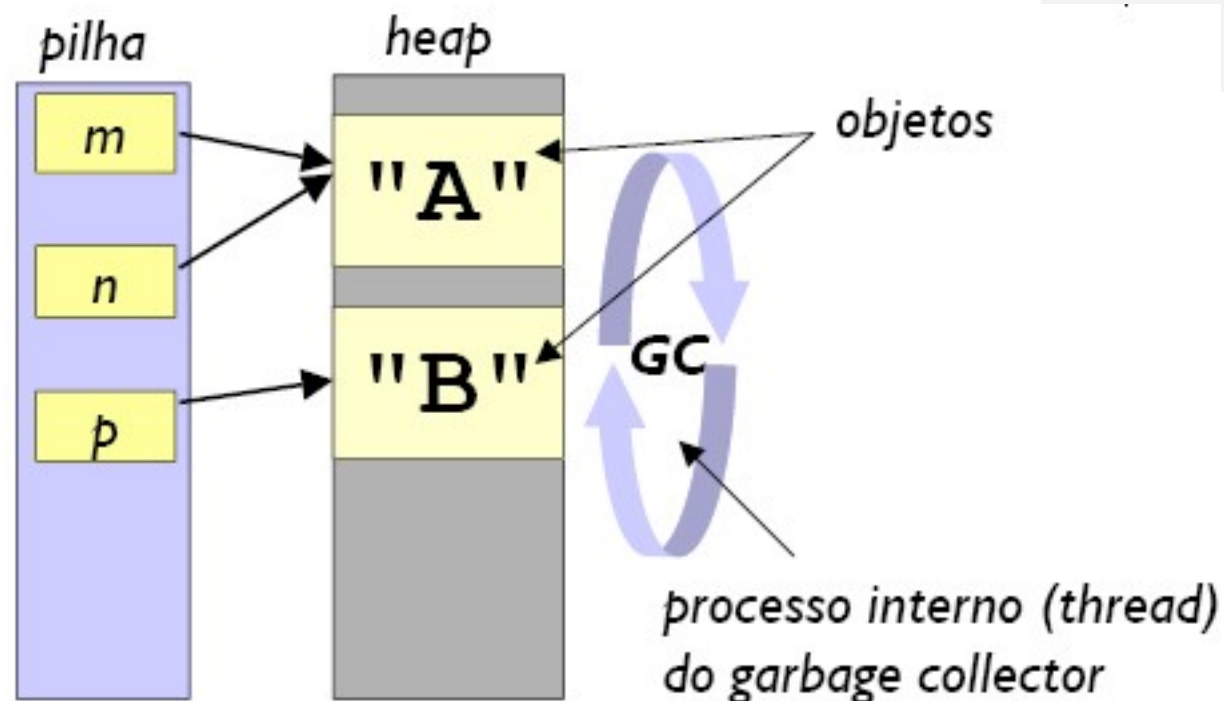
- A memória utilizada pela aplicação Java não é liberada pelo programador, ou seja, objetos criados não são destruídos pelo programador.

```
Mensagem m, n, p;
```

```
m = new Mensagem("A");
```

```
n = m;
```

```
p = new Mensagem("B");
```



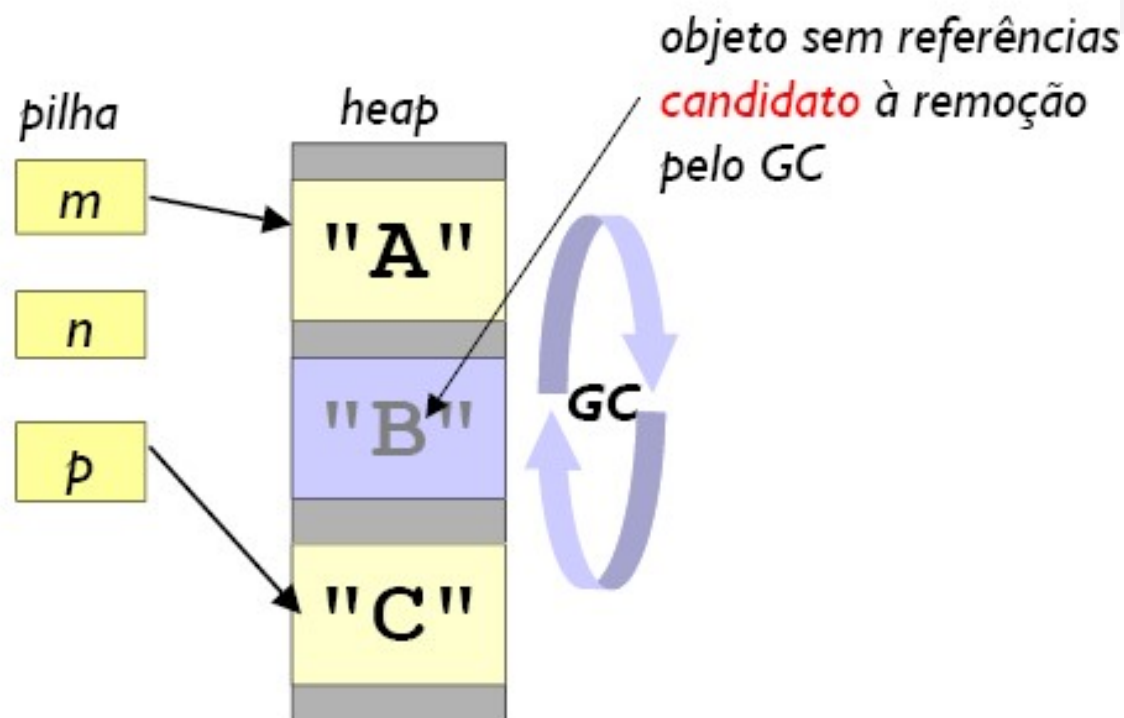
© Helder da Rocha

Introdução ao JAVA

ROBUSTA – COLETOR DE LIXO

- Quando um objeto não tem mais referências apontando para ele, seus dados não mais podem ser usados, e a memória deve ser liberada;
- O coletor de lixo irá liberar a memória na primeira oportunidade.

```
n = null;  
p = new Mensagem("C");
```



© Helder da Rocha

Introdução ao JAVA

INDEPENDENTE DE PLATAFORMA

- O fonte é compilado para um código intermediário (“bytecode”).
- O interpretador funciona como uma Máquina Virtual Java (JVM).

“write once, run anywhere”

Introdução ao JAVA

INDEPENDENTE DE PLATAFORMA – JVM

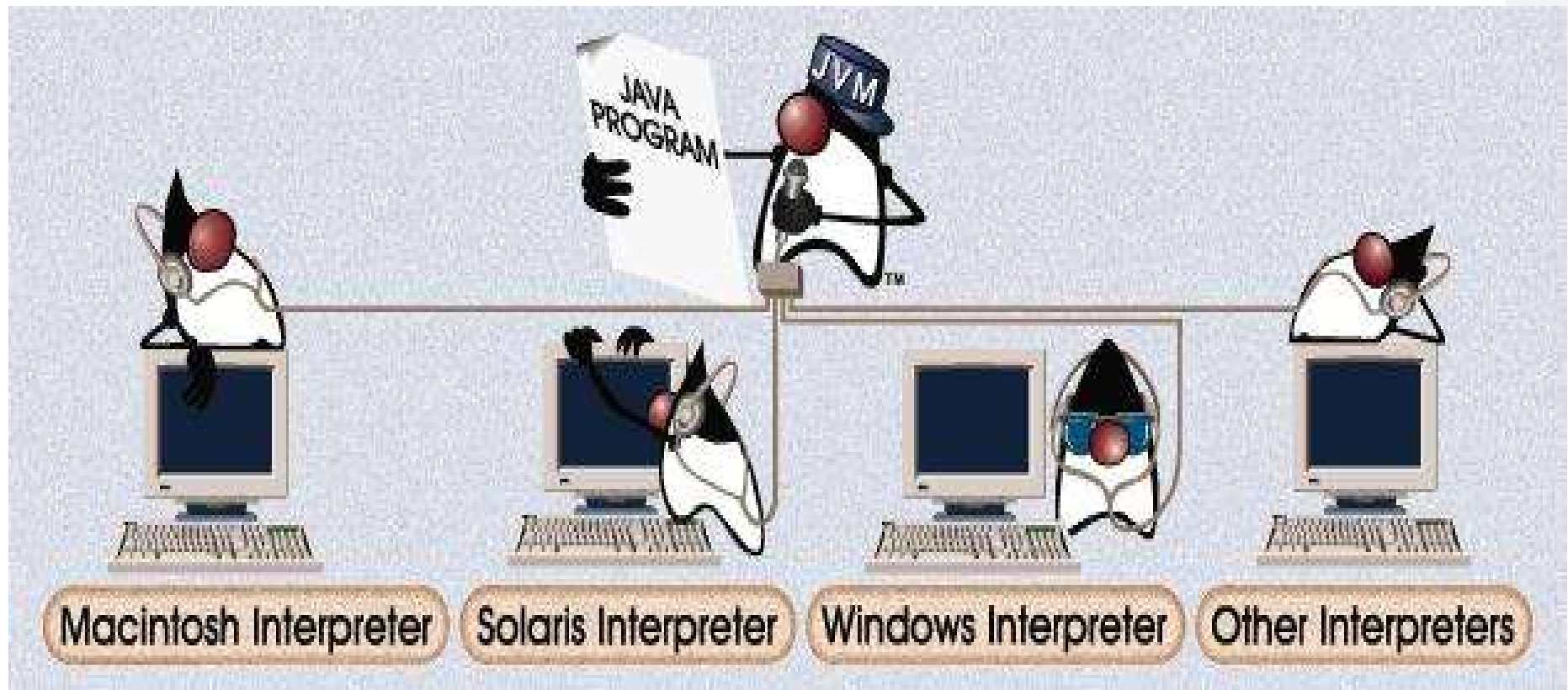


"write once, run anywhere"

Introdução ao JAVA

INDEPENDENTE DE PLATAFORMA – JVM

- Cabe ao interpretador Java de cada plataforma de hardware específica assegurar a execução do código compilado para a JVM.



"write once, run anywhere"

Introdução ao JAVA

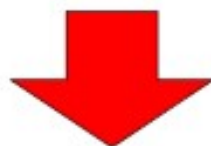
INDEPENDENTE DE PLATAFORMA – JVM

- Bytecode (*.class) é o código de máquina que roda em qualquer máquina através da Máquina Virtual Java (JVM).

Código
Java
(texto)

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

HelloWorld.java



compilação (javac)

HelloWorld.class

F4 D9 00 03 0A B2 FE FF FF 09 02 01 01 2E 2F 30 62 84 3D 29 3A C1

Bytecode Java (código de máquina virtual)

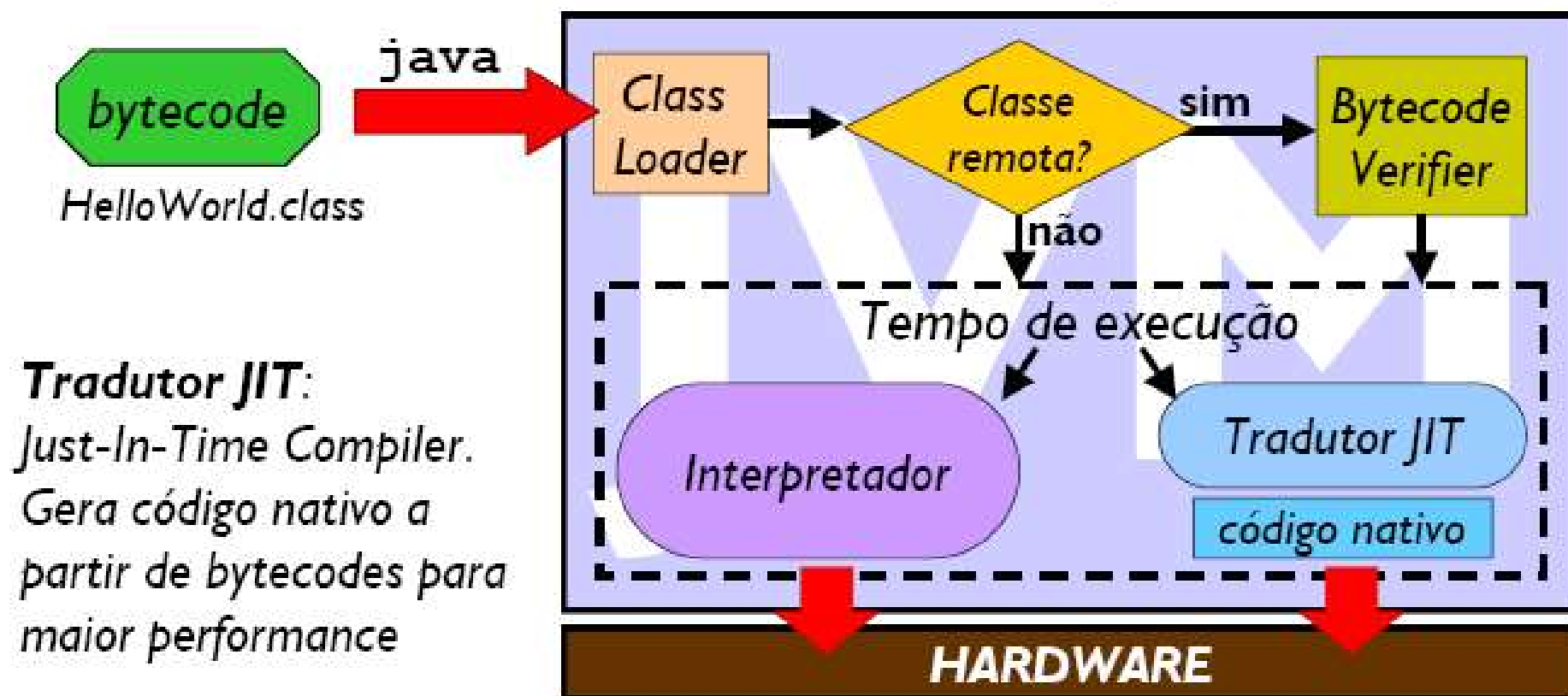
Uma "classe" Java

© Helder da Rocha

Introdução ao JAVA

INTERPRETADA

- Os bytecodes são executados por um interpretador (JVM) embora existam opções de compilação.



Introdução ao JAVA

PARALELIZÁVEL

- Suporta a programação baseada em “light weight processes” (*threads*).

Máquina Virtual Java (JVM)

JVM

DEFINIÇÃO

- Máquina imaginária implementada como uma aplicação de software em uma máquina real;
- A máquina virtual Java é hoje um sofisticado sistema de software, de alto grau de complexidade;
- HotSpot e JRockit são exemplos de tecnologias de máquina virtual imediatas comprovadas que fazem do Java um dos ambientes de programação mais rápidos;
- Otimizações incorporadas para ambientes multithread o deixam ainda mais rápido.


JVM

DEFINIÇÃO

- A JVM é uma especificação e não um software! Assim sendo, é possível escolher a JVM de um determinado fabricante, caso algum detalhe da Oracle JVM (padrão do SDK) não estiver atendendo a alguma necessidade em particular.
- Oferece vários algoritmos de Coleta de Lixo:
 - SGC (SerialGC) – Aplicação típica, um processador, < 2GB RAM
 - TGC (ParallelGC) – Muitas threads alocando objetos
 - CMS (ConcMarkSweepGC) - Pode compartilhar recursos do processador com o coletor de lixo enquanto a aplicação está executando
 - Train(TrainGC) - Requerimento de pausas mínimas; aceita eficiência baixa.

JVM

E O SUPORTE A LINGUAGENS DINÂMICAS

- As linguagens de programação LISP, Erlang, OCaml, Scala, Python/Django e Ruby on Rails podem rodar sobre a JVM (Máquina Virtual Java);
-  JRuby é a implementação 100% em puro Java da linguagem de programação Ruby, possibilitando que os scripts Ruby sejam executados na JVM.

Da Vinci Machine Project (JSR 292)

“A Java Virtual Machine não conhece NADA sobre a linguagem de programação. Ela apenas sabe executar um bytecode que é gerado pelo compilador.”

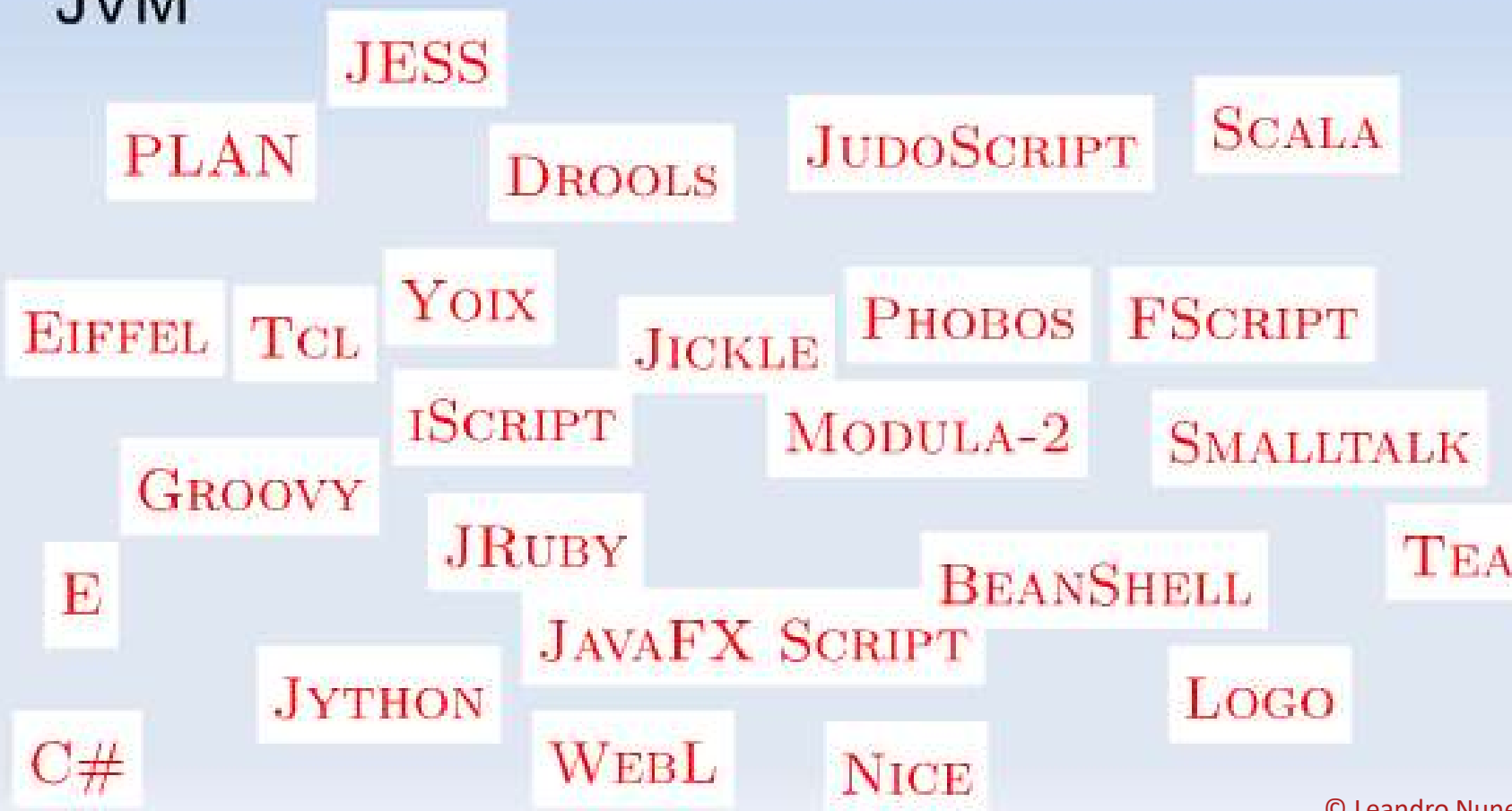


© Leandro Nunes

JVM

E O SUPORTE A LINGUAGENS DINÂMICAS

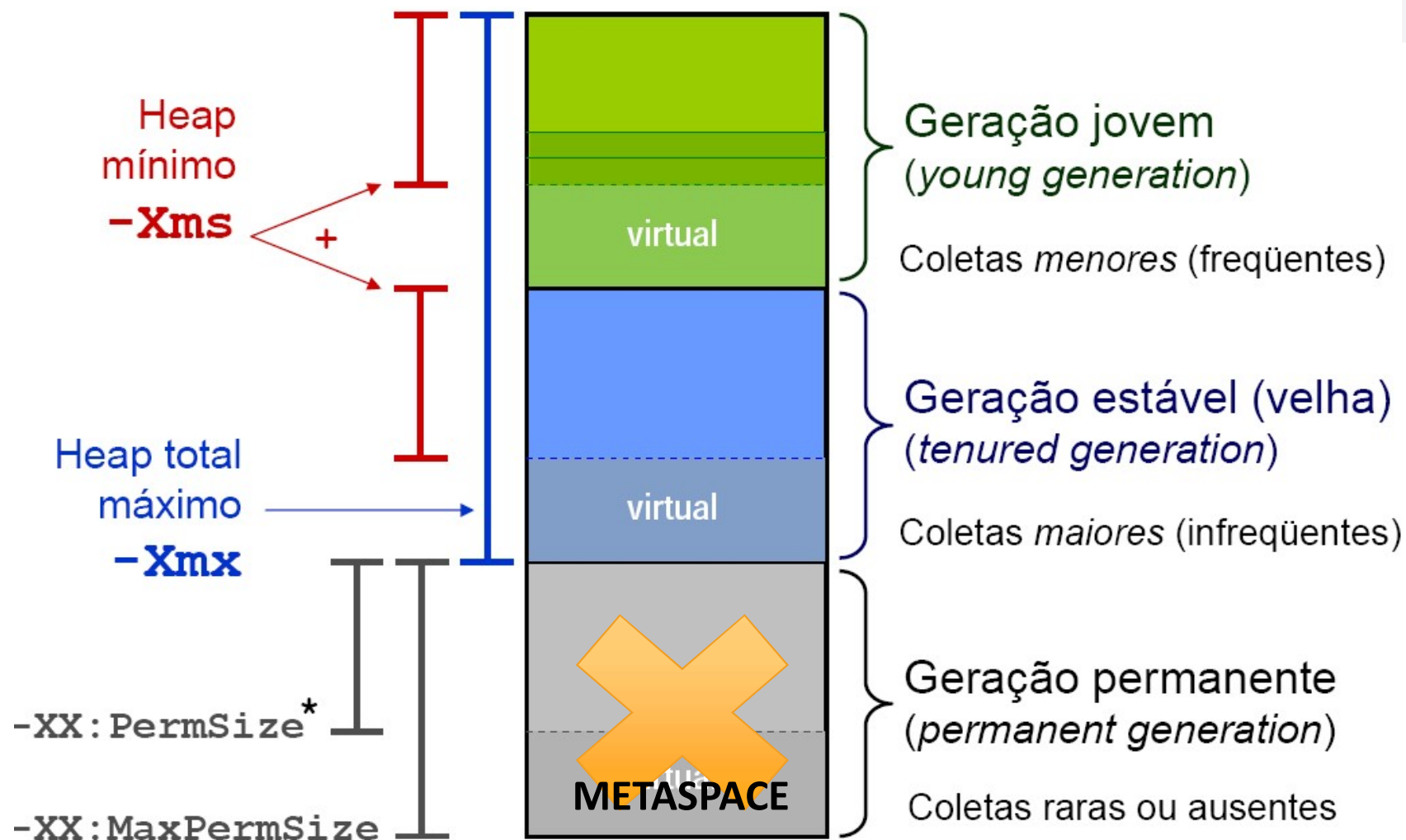
Hoje são dezenas de linguagens que rodam na JVM



© Leandro Nunes

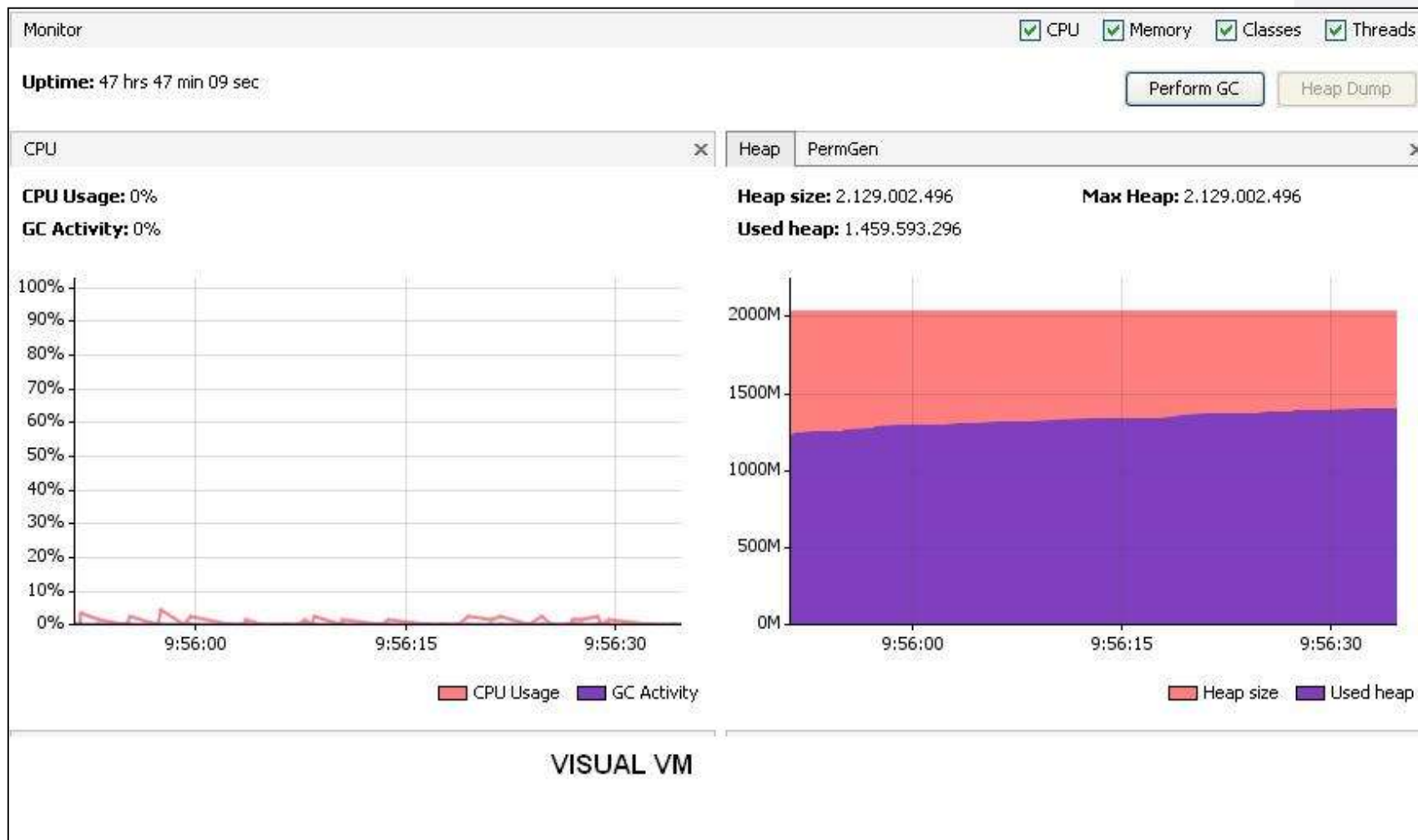
JVM

E O SEU GERENCIAMENTO DE MEMÓRIA



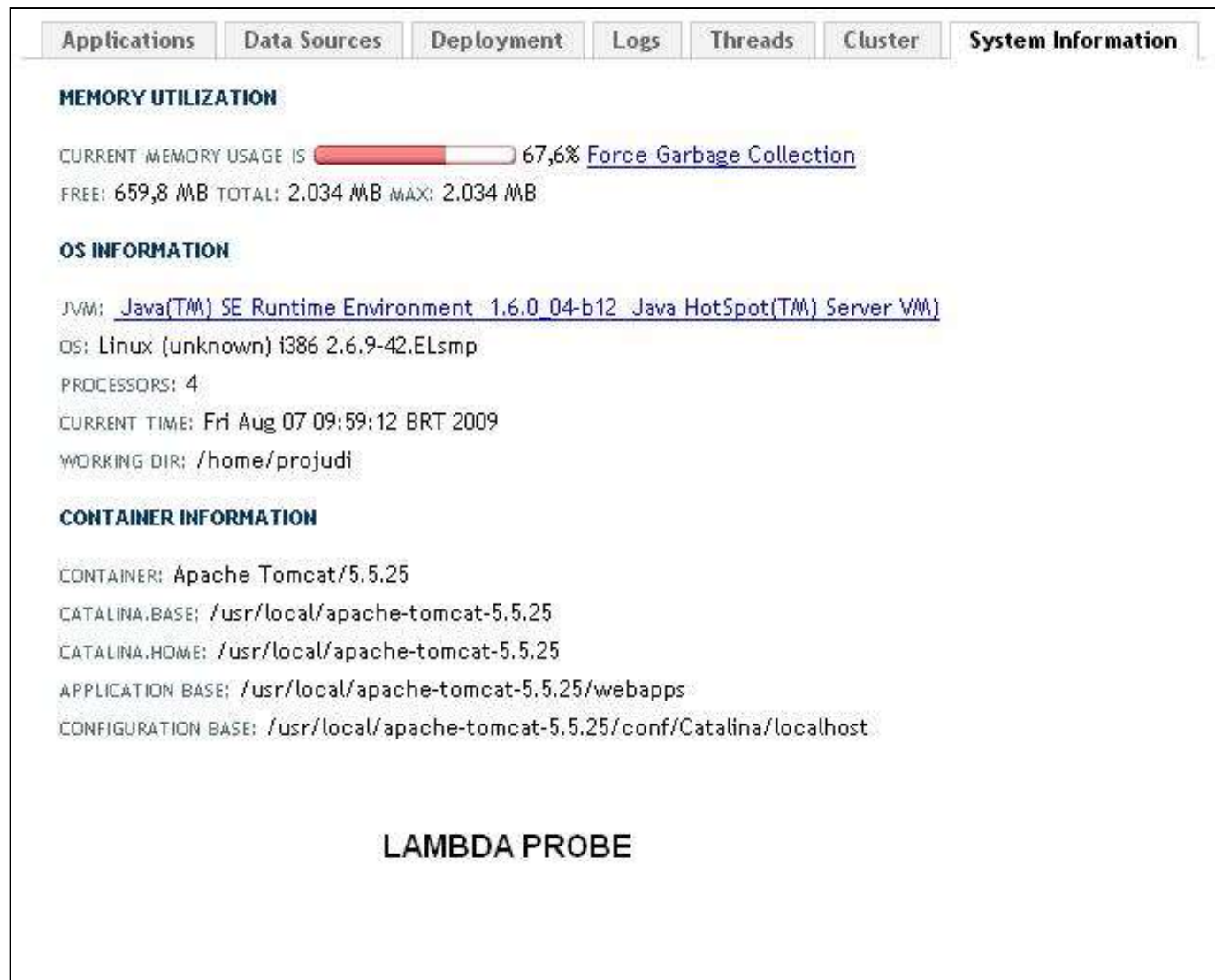
JVM

E O SEU GERENCIAMENTO DE MEMÓRIA



JVM

E O SEU GERENCIAMENTO DE MEMÓRIA



JVM

E O PROBLEMA DE 'MEMORY LEAK'

- “Ao contrário do que muitas pessoas pensam, uma aplicação escrita em Java pode sim apresentar problemas de vazamento de memória, termo comumente conhecido por *memory leak*. Infelizmente, um grande número de programadores Java acha que *memory leak* é coisa de C/C++ e que o *garbage collector* do Java resolve esse problema completamente”.
- “Elimine todas as referências a objetos desnecessários. Fazendo isso, o *garbage collector* terá condições de fazer o seu trabalho completamente e você estará livre dos *memory leaks* em suas aplicações”.

© [Carlos Tosin](#)

Ambiente Java

Ambiente JAVA

DISPONÍVEL GRATUITAMENTE NA INTERNET

- **Java 8 DK (Java 8 Development Kit)** é o ambiente padrão distribuído pela Oracle para desenvolvimento de aplicações Java.
- O **J8DK** consiste de:
 - **JRE (Java Runtime Environment)** - também distribuído separadamente: ambiente para execução de aplicações;
 - **Ferramentas para desenvolvimento**: compilador, debugger, gerador de documentação, empacotador JAR, etc;
 - **Código-fonte** das classes da API;
 - **Demonstrações de uso das APIs**, tais como Applets, interfaces gráficas com Swing e o uso de recursos de multimídia.
- A documentação das APIs é distribuída separadamente.

Ambiente JAVA

JDK

- O **Java Development Kit (JDK)** é o conjunto de ferramentas para o desenvolvedor Java. Fazem parte do JDK:
 - javac (compilador)
 - java (interpretador – máquina virtual)
 - javadoc (gerador automático de documentação)
 - applet viewer (visualizador de Applets)
 - jconsole (monitorador de aplicações)
 - javap (mostra o mneumônico de um bytecode)
 - javaws (permite fazer download e executar aplicativos Java via Internet)
 - Outros (/bin).

Ambiente JAVA

JCONSOLE

J2SE 5.0 Monitoring & Management Console: 1380@localhost

Connection

Summary Memory Threads Classes MBeans VM

Summary

Uptime: 15,719 seconds Process CPU time: 1,265 seconds
Total compile time: 0,100 seconds

Threads

Live Threads: 16 Peak: 16
Daemon threads: 13 Total started: 17

Memory

Current heap size: 1.196 kbytes Committed memory: 1.984 kbytes
Maximum heap size: 65.088 kbytes
Objects pending for finalization: 0
Garbage collector: Name = 'Copy', Collections = 10, Total time spent = 0,066 seconds
Garbage collector: Name = 'MarkSweepCompact', Collections = 0, Total time spent = 0,000 seconds

Classes

Current classes loaded: 1.837 Total classes unloaded: 0
Total classes loaded: 1.837

Operating System

Total physical memory: 1.046.704 kbytes Free physical memory: 254.088 kbytes
Committed virtual memory: 17.028 kbytes

Ambiente JAVA

VERSÕES DO JDK

- Desde o lançamento do primeiro JDK, em 1995, várias versões já surgiram:
- **1996: Versão 1.0 do (JDK) foi disponibilizado pela Sun.**
 - 8 pacotes com 212 classes
 - Netscape 2.0-4.0 incluía o Java 1.0.
 - Microsoft e outras empresas licenciaram o Java.
- **1997: Versão 1.1**
 - 23 pacotes - 504 classes
 - Microsoft desenvolveu a sua própria Java Virtual Machine para o Internet Explorer.
 - Pacote Swing melhorado.

Ambiente JAVA

VERSÕES DO JDK

- **1999: Versão 1.2, também conhecida como Java 2 Platform:**
 - 59 pacotes - 1520 classes
 - Collections API incluída para oferecer suporte a listas, conjuntos e hash tables.
 - Divisão do Java 2:
 - Java 2 Micro Edition (J2ME)
 - Java 2 Standard Edition (J2SE)
 - Java 2 Enterprise Edition (J2EE)
- **2000: Versão 1.3:**
 - 76 pacotes - 1842 classes;
 - Melhoria de performance da JVM.

Ambiente JAVA

VERSÕES DO JDK

- **2002: Versão 1.4:**
 - 135 pacotes - 2991 classes
 - Mudanças no framework Collections e nas classes de Rede
 - Melhorias - IO, suporte a XML, etc.
- **2004: Versão 1.5 (Java Tiger):**
 - Introduz novidades na sintaxe
 - Esta versão incluiu novas funcionalidades como *Generics*
 - *Annotations*
 - *Autoboxing e Auto-unboxing*
 - *Enumerations e Static import*
 - *For-each e Varargs*
 - *Entrada e Saída formatadas (java.util.Scanner e Formatter)*
 - *Novos estilos look-and-feel (GTK/Linux e Windows XP)*

Revolucionário

Ambiente JAVA

CÓDIGO ANTES DO JAVA 5

```
List lista = new ArrayList();  
// não restringe tipo dos elementos  
lista.add(new Integer(25));  
// boxing: converter int 25 em objeto Integer  
lista.add(new Integer(33));  
// boxing: converter 33 em Integer  
int total = 0;  
for (Iterator it = lista.iterator(); it.hasNext();) {  
    Integer elem = (Integer)it.next();  
    // iteração next() e cast explícito (Integer)  
    total += elem.intValue();  
    // unboxing: intValue()  
}
```

Ambiente JAVA

CÓDIGO APÓS O JAVA 5

```
List lista = new ArrayList();  
List<Integer> lista = new ArrayList<Integer>();  
// generics: define tipo do elemento Integer  
lista.add(25);  
// autoboxing  
lista.add(33);  
// autoboxing  
int total = 0;  
for (Integer elem : lista) {  
    // for() melhorado: iteração implícita, dispensa cast  
    total += elem;  
    // autoboxing: unboxing implícito  
}  

```

Ambiente JAVA

VERSÕES DO JDK

- **2006: Versão 1.6 (Java Mustang):**

- Suporte nativo a XML (API JAX-B)
- Novas APIs como JSR-223 (javax.script), JSR-269 (javax.annotation)

OPEN SOURCE



Ambiente JAVA

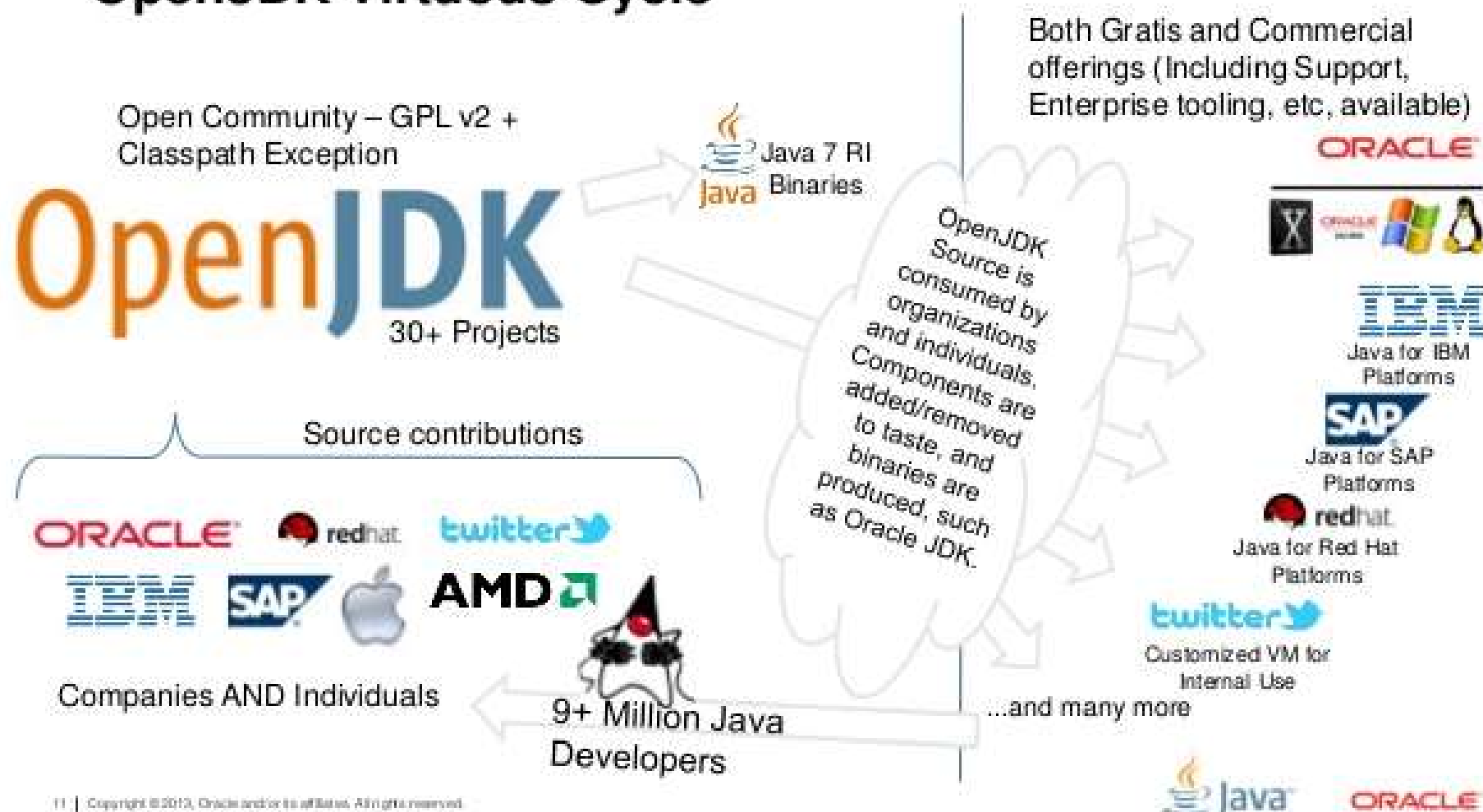
O QUE É O PROJETO OPENJDK?

- O OpenJDK é um projeto que foi iniciado pela Sun Microsystems para a criação de um JDK baseado totalmente em software livre e de código aberto. O projeto foi iniciado em 2006 e teve como base a JVM da Sun (HotSpot).
- O projeto OpenJDK funciona como a implementação de referência Open Source do Java Standard Edition. Empresas como a Oracle, IBM, e Azul Systems suportam e investem no projeto OpenJDK para continuar evoluindo a plataforma Java.
- O Oracle JDK é baseado no OpenJDK, mas traz outras ferramentas como o Mission Control, e a máquina virtual traz algumas features avançadas como por exemplo o Flight Recorder. Até a versão 6, a Oracle oferecia duas máquinas virtuais: JRockit (BEA) e HotSpot (Sun). A partir da versão 7 a Oracle unificou as máquinas virtuais, e levou as features avançadas do JRockit para dentro da VM HotSpot.

Ambiente JAVA

CÍRCULO VICIOSO

OpenJDK Virtuous Cycle



Ambiente JAVA

VERSÕES DO JDK

- **2011: Versão 1.7 (Java Dolphin):**

Diretriz ORACLE

- Switch com suporte a String
- Nova verificação de referência nula
- Suporte à representação de Binários
- Sufixos para byte e short
- Inicialização de Mapas de forma mais simples
- Uso mais elegante de tipos Genéricos
- Tratamento de várias exceções no mesmo bloco
- Variação do bloco try{}
- Indexação de arrays com 64b

Ambiente JAVA

VERSÕES DO JDK

- **2014: Versão 1.8 (Descontinuado codinome):**

- Expressões Lambda
- Nova API de datas e API de Stream
- Novos métodos nas APIs de Collections, Concurrency, IO/NIO
- Remoção do PermGen e a criação do Metaspace para ocupar o seu lugar
- Interfaces com métodos estáticos e métodos com implementação
- Interface com apenas um método (`@FunctionalInterface`)

Programação Funcional

```
// CÓDIGO ANTES DO JAVA 8 (SEM LAMBDA)
```

```
List<Integer> lista = Arrays.asList(1,2,3,4,5);  
For (Integer inteiro : lista)  
    System.out.println("Número: " + inteiro);
```

```
// CÓDIGO APÓS O JAVA 8 (COM LAMBDA)
```

```
List<Integer> lista = Arrays.asList(1,2,3,4,5);  
lista.forEach(valor->System.out.println(valor));
```

JAVA 8



- A versão Java SE 8 é o resultado do desenvolvimento que envolveu revisão aberta, compilações semanais e extensa colaboração entre engenheiros da Oracle e membros da comunidade de desenvolvedores Java em todo o mundo por meio da comunidade OpenJDK e do *Java Community Process* (JCP).
- O Java SE 8 contribui para maior produtividade do desenvolvedor e aumentos significativos de desempenho de aplicativos, melhores coleções e anotações, modelos de programação paralela mais simples e uso mais eficiente de processadores multicore modernos.
- Os principais recursos do JDK 8 são o Project Lambda (JSR 335), uma nova API de data e hora (JSR 310), um conjunto de perfis compactos e a remoção da “geração permanente” da HotSpot Java Virtual Machine (JVM).

JAVA 8

- O Oracle JDK 8 já alcançou desempenho recorde mundial em sistemas de quatro soquetes em servidores baseados em Intel da NEC e sistemas de dois soquetes em servidores SPARC T5 da Oracle, com uma melhoria de desempenho de 12% a 41% em comparação com o JDK 7 na mesma configuração Oracle .
- Outro benefício é que a compatibilidade do Java SE 8 com versões anteriores da plataforma preserva os conjuntos de habilidades dos atuais desenvolvedores de software em Java e ajuda a proteger os investimentos nesta tecnologia.

Vídeo JAVA 8



Oracle Java 8 Presentation

<https://www.youtube.com/watch?v=0q0LEf0lyi0>

RESUMO

TÓPICOS APRESENTADOS

- Nesta aula nós estudamos:
 - **Configuração do Ambiente Java**
 - **O meu Primeiro Código em Java**
 - **O que é o Java?**
 - **Histórico**
 - **Evolução**
 - **Presente e Futuro**
 - **Características do Java**
 - **Máquina Virtual Java (JVM)**
 - **Ambiente Java**

ATIVIDADES PARA SE APROFUNDAR

- 1) Pesquisar na Internet aplicações Java SE, EE e ME.
- 2) Pesquisar no site da Oracle a documentação da API Java 8. Encontre as classes **JOptionPane** e **String**.

<http://docs.oracle.com/javase/8/docs/api/>

- 3) Identificar uma situação específica em que um bytecode Java 'roda' no Windows e 'não roda' no Linux.
- 4) Identificar as principais APIs Java para Big Data e IoT.
- 5) Explicar para que serve o Java Mission Control.
- 6) Descobrir as principais novidades previstas para o Java 9.