



DOMAIN DRIVEN DESIGN

Prof. Rafael Desiderio

02 – INTRODUÇÃO AO JAVA

| O que é uma linguagem de programação?

- Uma **linguagem** é um **conjunto de regras sistemáticas** para a **comunicação de idéias**;
- Uma **linguagem de programação** é um conjunto de **símbolos, palavras e regras** utilizados na construção de sentenças que expressam e processam **instruções para computadores**;
- *“Uma linguagem de programação é uma linguagem a ser usada por uma pessoa para expressar um processo através do qual um computador pode resolver um problema.”*
(Dershem & Kipping, Programming Languages: Structure and Models);

| Tipos de linguagens

- **Linguagem de Máquina**

0010 0100

0001 1010

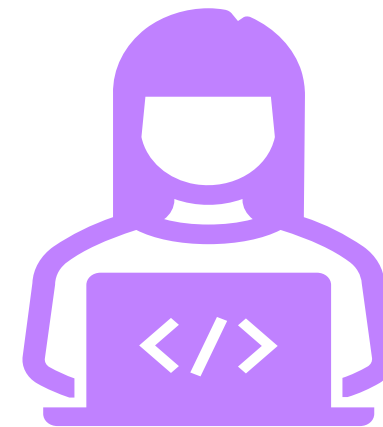
- **Linguagem de Baixo Nível**

LDA 4

STA A

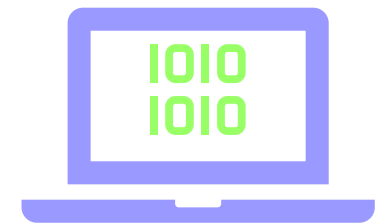
- **Linguagem de Alto Nível**

contador = 0;



| Linguagem de máquina

- São **linguagens** voltadas para a **máquina**;
- São baseadas no **código binário** utilizado diretamente pelo computador (0s e 1s);
- As instruções **variam de processador** para **processador**;
- Os **programas** são **escritos** utilizando as instruções da UCP;
- **Desvantagens: pouca portabilidade** (em geral, um programa para um processador não serve para outro), programas não são estruturados e de **difícil compreensão, manutenção e correção de erros** extremamente difíceis;



| Linguagem de baixo nível

- São **linguagens** voltadas para a **máquina** e para o **usuário**;
- São linguagens intermediárias entre a linguagem de máquina e a linguagem de alto nível;
- As instruções **são simplificações da linguagem** de máquina que usam código **mnemônicos**;
- Usualmente recebem a denominação de **Assembly**;
- **Vantagens**: comandos com sintaxe **mais inteligível** que as linguagens de máquina;
- **Desvantagens**: alguns mnemônicos ainda são de **difícil compreensão**; ainda depende da arquitetura do computador;



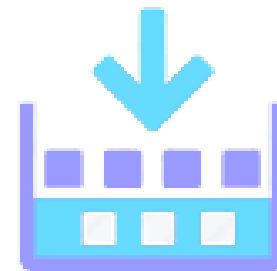
| Linguagem de alto nível

- São **linguagens** voltadas para o **usuário**;
- Os comandos das linguagens apresentam um **nível mais alto** de **abstração** e **próximos da linguagem humana**;
- Necessitam de **programas especiais** (compiladores ou interpretadores) para traduzir o código para linguagem de máquina;
- **Vantagens**: tem **maior portabilidade**, podendo ser executadas em várias plataformas com **pouquíssimas modificações**; não exigem conhecimento do código de máquina;
- **Desvantagens**: as rotinas geradas são mais genéricas e portanto mais **complexas** e por isso são **mais lentas** e ocupam mais memória;



| Metodos de implementação

- Os **programas escritos** em uma **linguagem de programação** de alto nível devem ser **traduzidos** para a **linguagem de máquina** para serem executados;
- Esse **programa tradutor** recebe como **entrada** o **código fonte** e **gera** o **código de máquina**;
- Exemplos:
 - **Compilação**;
 - **Interpretação**;
 - **Implementação**;



| Compilação

- Um **compilador traduz o programa fonte inteiro**, produzindo um outro programa equivalente, em **linguagem executável** (programa objeto);
- A vantagem é que o **compilador precisa traduzir um comando** apenas uma **única vez**, não importando quantas vezes ele será executado;

Exemplo: programa em C



| Interpretação

- O interpretador “**executa**” diretamente as instruções do programa fonte, **sem** traduzir para linguagem de máquina;
- **Desvantagens**: execução mais **lenta**, devido ao passo de decodificação da instrução de alto nível, que é mais **complexa**;
- Tem **acesso ao programa fonte**, para depuração ou mesmo para alterar o código sendo executado;

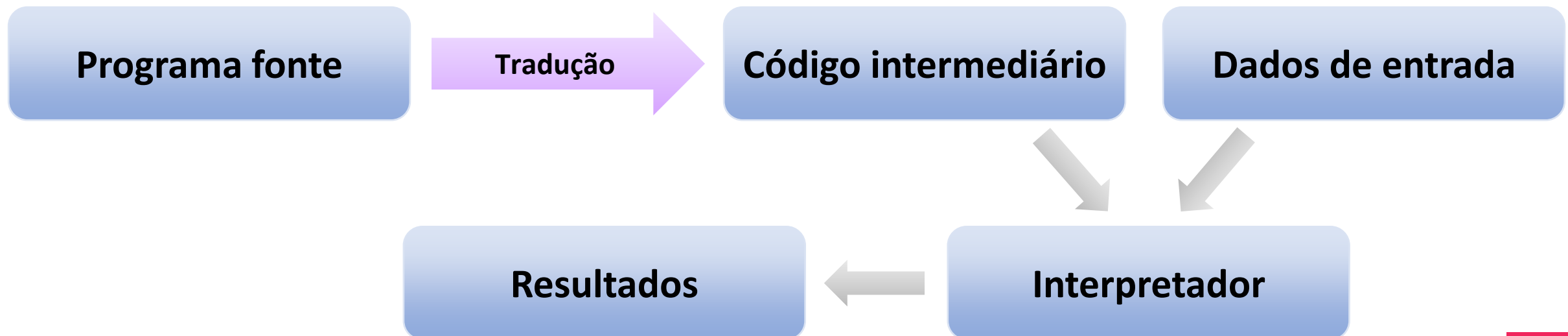
Exemplo: programa em **Basic, Python**



| Implementação Híbrida

- **Programas fonte** são traduzidos para uma **linguagem intermediária** que é interpretada para a execução;
- Tem **maior portabilidade** que uma linguagem compilada;
- São mais **rápidas** que uma linguagem interpretada, instruções intermediárias são projetadas para serem **interpretadas facilmente**;

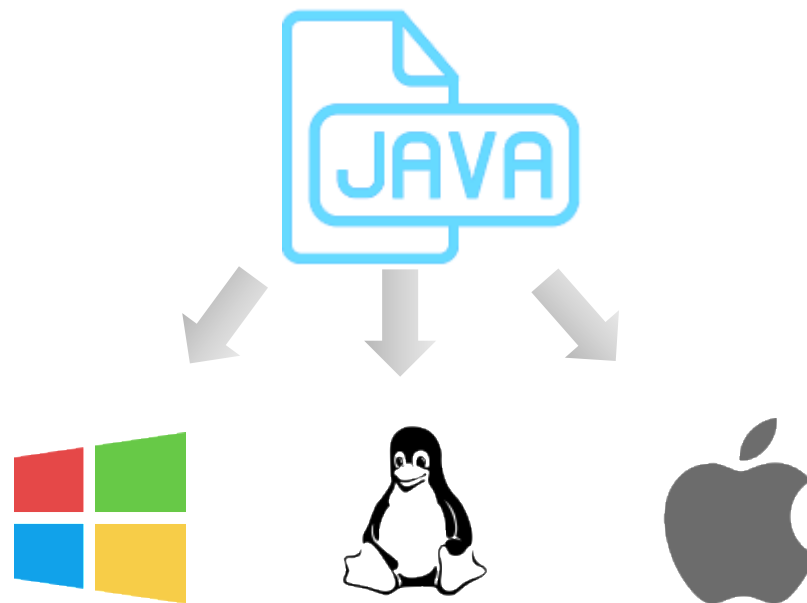
Exemplo: programa em **Java e C#**



JAVA

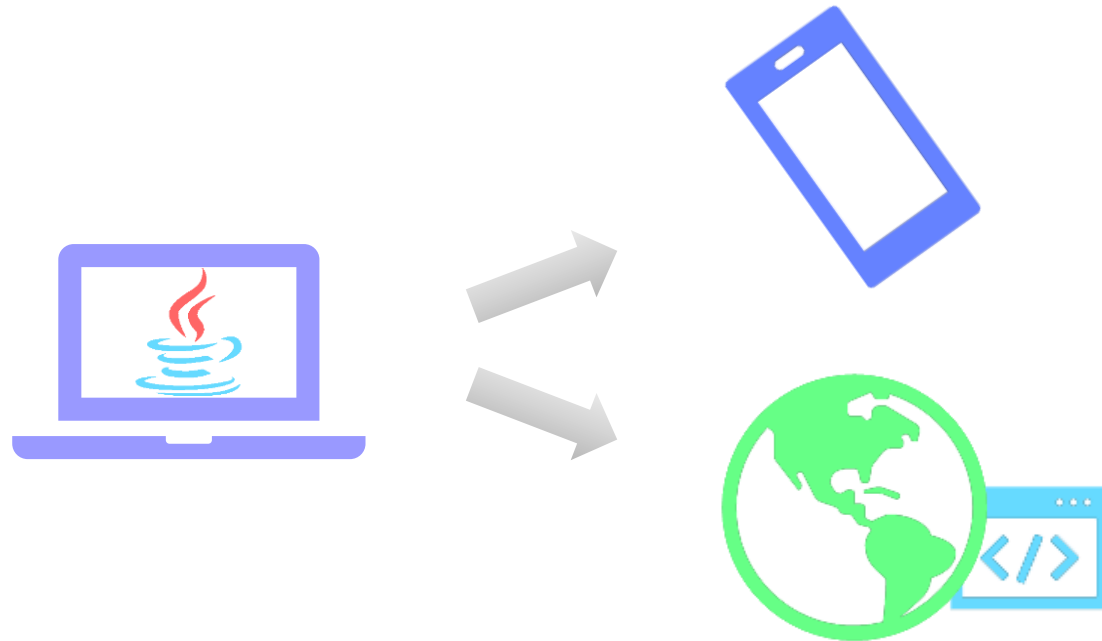
| Características do JAVA

- A **razão principal** para a criação do **Java** foi a necessidade de uma **linguagem independente de plataforma** que pudesse ser usada para a criação de **software** que iriam ser embutidos em **geladeiras, micro-ondas e controles remotos**, por exemplo;



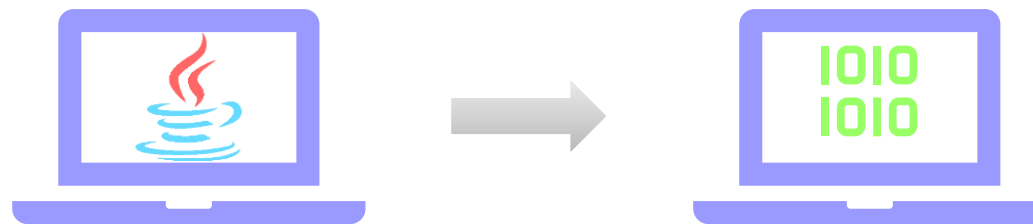
| Características do JAVA

- Em poucos anos os projetistas do **Java** perceberam que a **criação de aplicações para Internet** estavam tendo problemas de **portabilidade**;
- Assim, o foco da **Sun** passou a ser a **internet** em vez dos aparelhos eletrônicos;



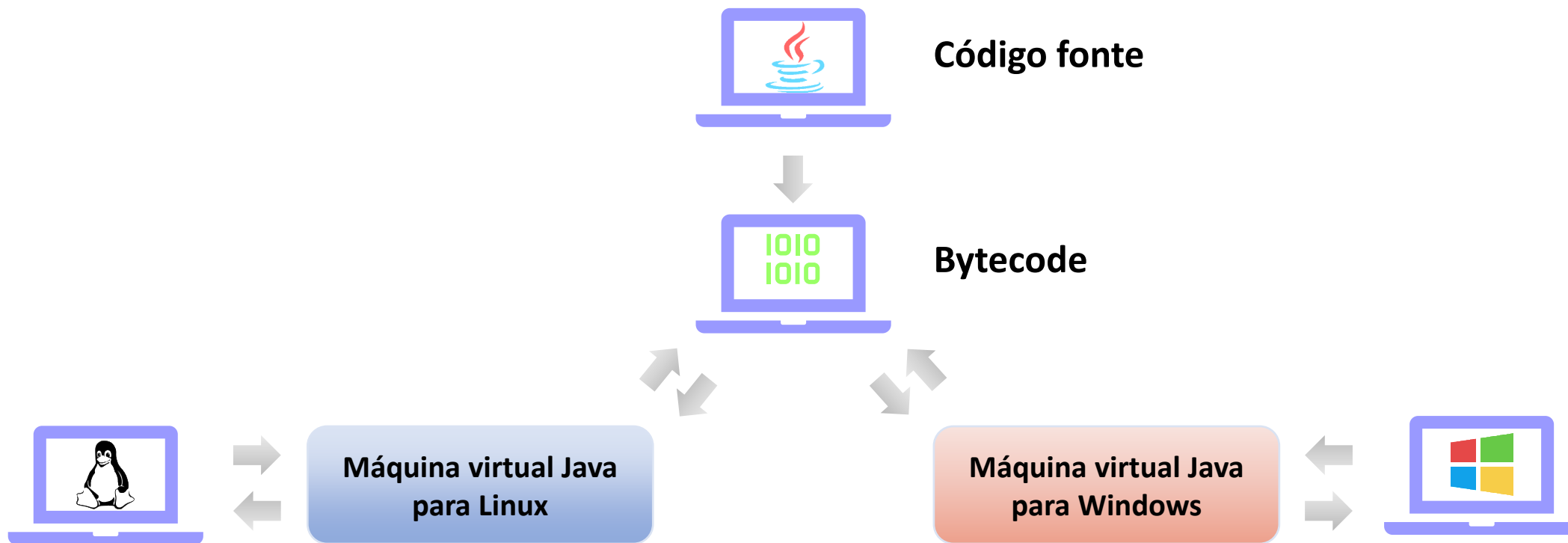
| Bytecodes

- O **código compilado** de um programa **Java** pode ser **executado** em **diferentes plataformas** sem nenhuma alteração;
- Isto é possível graças ao **bytecode** que é gerado durante a compilação do código fonte;



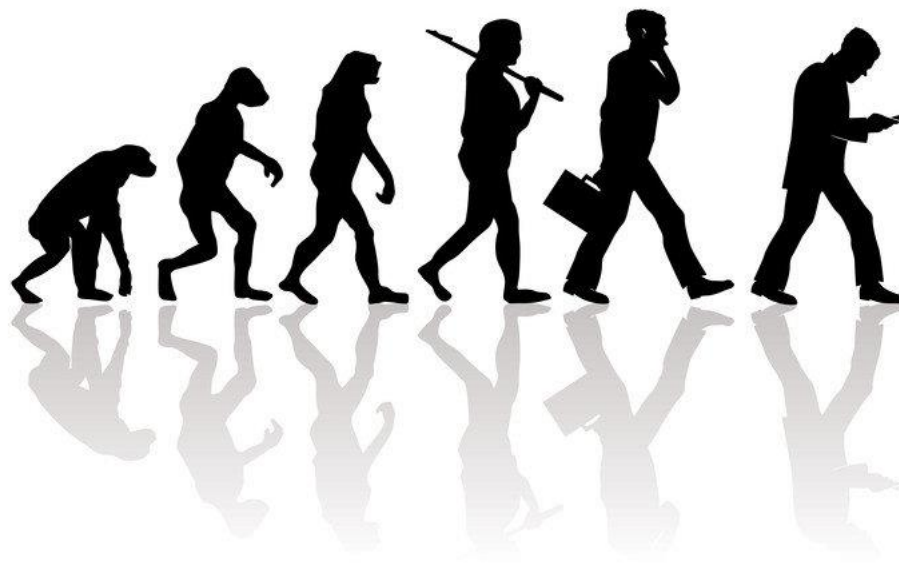
| Portabilidade

- O **mesmo código fonte Java** pode ser executada em diferentes sistemas operacionais, sem precisar de modificações, graças a **JVM**;



| Ciclo de vida

- Para **desenvolver uma aplicação** usando **Java**, deve-se seguir algumas **etapas específicas** que constituem o **ciclo de vida** do desenvolvimento de um programa Java;
- As **fases do ciclo** de vida são:
 - Criação;
 - Compilação;
 - Interpretação;



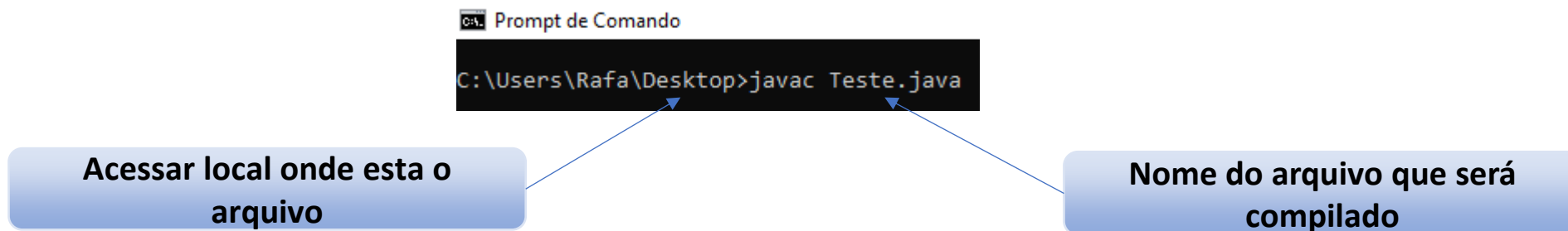
| Vamos à prática

- Abra o bloco de notas e digite o **código** conforme o exemplo abaixo:

```
Teste.java - Bloco de notas
Arquivo  Editar  Exibir

public class Teste {
    public static void main(String [] args){
        System.out.println("Java é fácil");
    }
}
```

- Agora precisamos **compilar** o programa, ou seja, transforma-lo em **bytecodes**. Para isso vamos acessar o local do arquivo (via prompt) e executar através de um programa chamado **javac**:



- Após a execução**, veja que o arquivo **Teste.class** foi criado no mesmo local onde esta o **Teste.java**;

| Vamos à prática

- Para **executar o programa**, utilize o **Java** e o nome do arquivo **Bytecode**;

cmd Prompt de Comando

```
C:\Users\Rafa\Desktop>java Teste
Java é fácil!
C:\Users\Rafa\Desktop>_
```

- Perceba os **passos** que foram realizados da **implementação** do código até a sua **execução**;
 - Para compilar o código foi utilizado um programa (javac.exe);
 - Para executar foi utilizado outro (java.exe);
- No dia a dia utilizaremos **ferramentas** (IDEs) que realizam todos esses processos de **forma automática**;

Plataforma Java

| Edições da plataforma Java

- O universo **Java** é composto por um vasto conjunto de tecnologias, que possui **três plataformas principais**, específicos para cada tipo de **aplicação**:



Micro Edition (ME)



Standard Edition (SE)



Enterprise Edition (EE)

| Java ME

- Ambiente de execução **altamente aperfeiçoado**;
- **Não** se destina à programação de **computadores** e sim à programação de **pequenos dispositivos eletrônicos**;
- **Exemplos**: cartões inteligentes, dispositivos móveis, equipamentos eletrônicos de consumo;



Micro Edition (ME)

- Solução para **desenvolvimento** de **aplicações Java**;
- **Inclui:**
 - Acesso a banco de dados;
 - Múltiplas linhas de execução;
 - Aplicações distribuídas;
 - Interfaces gráficas;
 - Redes e etc...;
- É distribuída de duas formas:
 - **JDK**, utilizada no desenvolvimento de aplicativos Java;
 - **JRE**, utilizada para a execução de aplicativos Java;



Standard Edition (SE)

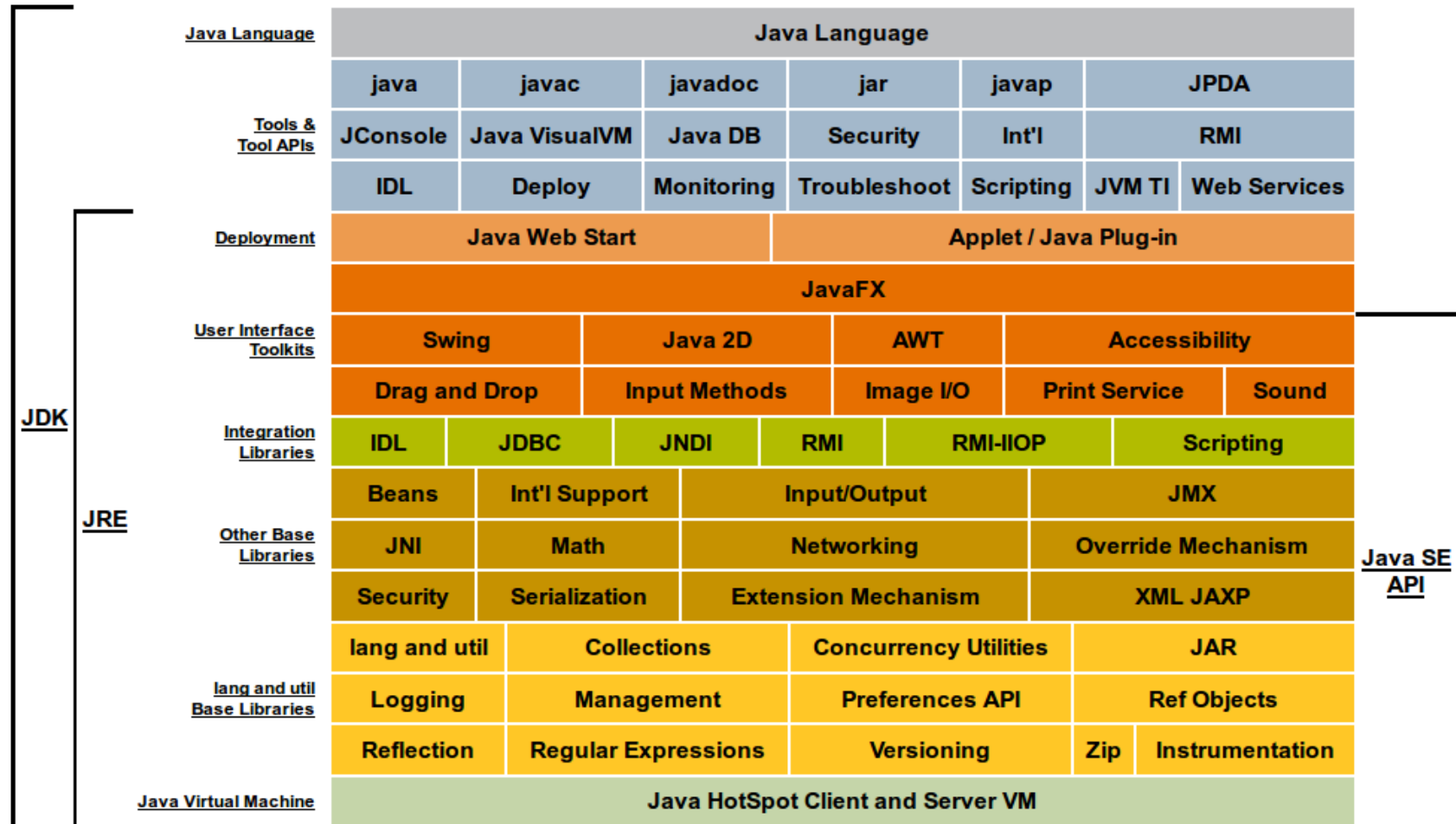
| Java EE

- Voltado para **desenvolvimento de aplicações de grande porte e aplicações web**;
- Seu **modelo de componentes** simplificam o **desenvolvimento** de aplicações;



Enterprise Edition (EE)

| Plataforma Java



| Java Development Kit - JDK

- Várias **ferramentas** que auxiliam um **desenvolvedor** de software a implementar e executar os **programas Java**;
- As ferramentas:
 - **JAVAC** (Compilador, converte o código-fonte em bytecode)
 - **JAVA** (Interpreta o bytecode)
 - **JDB** (Depura um programa Java)
 - **JAVAP** (Decompila um arquivo class do Java)
 - **JAVADOC** (Gera informação HTML das classes criadas)
 - **JAVAH** (Cria headers que estendem Java para C)
 - **APPLETVIEWER** (Executa Applets)
 - **JAR** (Empacota e compacta os arquivos da aplicação)



| Java Virtual Machine - JVM

- Segundo a Sun: *“uma máquina imaginária implementada via software ou hardware que executa instruções vindas de bytecodes”*;
- Linguagens como C são executadas pelo SO;
- **JVM** atua como **intermediária** entre o programa e o sistema;
- Responsável por gerenciar: **memória (garbage collector), erros, exceções, threads**;
- Garante as seguintes características da linguagem **Java**:
 - **Portabilidade**
 - **Eficiência (JIT)**
 - **Segurança**

| Resumindo

Programa em Java

Java Bytecode

JVM

Sistema Operacional

Hardware

Copyright © 2024 Prof. Rafael Desiderio

Todos os direitos reservados. Reprodução ou divulgação total ou parcial deste documento é expressamente proibido sem o consentimento formal, por escrito do Professor (autor).

“Para conquistar o sucesso, você precisa aceitar todos os desafios que vierem na sua frente. Você não pode apenas aceitar os que você preferir” - Mike Gafka