

JAVA DEVELOPMENT

FUNDAMENTOS JAVA



1

LISTA DE FIGURAS

Figura 1 – James Gosling, criador do Java em 2008	4
Figura 2 – Compilação e interpretação do código Java	8
Figura 3 – Site para o download da JDK	9
Figura 4 – Instalador do Java	9
Figura 5 – Versão do Java instalado na máquina	10
Figura 6 – Programa Hello World em Java	11
Figura 7 – Salvando a primeira classe Java	12
Figura 8 – Compilando a classe Java	12
Figura 9 – Classe Java compilada (bytecode)	12
Figura 10 – Interpretando o bytecode na JVM	13

SUMÁRIO

FUNDAMENTOS JAVA.....	4
1 INTRODUÇÃO	4
2 PLATAFORMA JAVA	6
3 EDIÇÕES DO JAVA.....	6
REFERÊNCIAS.....	14

EMSE

FUNDAMENTOS JAVA

1 INTRODUÇÃO

Nos dias atuais, tão conectados, provavelmente você já utilizou algum sistema escrito em Java. Pode ser em uma aplicação web, como internet banking, mobile nos aplicativos e até em TVs ou cartões inteligentes. Java está presente em todos os lugares!

Atualmente, Java é uma das linguagens mais utilizadas em todo o mundo e em amplo crescimento nas grandes empresas. Java não é somente uma linguagem de programação, mas uma plataforma de desenvolvimento que possui todas as ferramentas, como servidores, ambientes de execução e bibliotecas (*frameworks*) para construir aplicações de variados tipos, como desktop, web, APIs, mobile, sistemas embarcados etc.

Java começou a ser criado em 1991, na Sun Microsystems, por uma equipe de engenheiros liderados por Patrick Naughton, Sun Fellow e James Gosling. O Projeto Green consistia na criação de tecnologias para a comunicação entre aparelhos eletrônicos, como TVs, videocassetes, micro-ondas etc., uma visão de futuro, que hoje é uma realidade com IoT (Internet das Coisas).



Figura 1 – James Gosling, criador do Java em 2008
Fonte: Wikipedia (2022)

Dessa forma, surgiu a linguagem de programação denominada **Oak**, porém, naquela época, a tecnologia não estava preparada para essa conectividade entre os

aparelhos, mas com o advento da internet, a linguagem Oak foi adaptada para a internet e assim nasceu o Java, que foi lançado em 1995.

A Oracle adquiriu o Java em 2008 e, desde então, é uma referência no mercado de desenvolvimento de sistemas.

Mas por que o Java tem todo esse sucesso e é tão popular?

Seu grande diferencial é a portabilidade ou a independência de plataforma, ou seja, Java funciona em qualquer sistema operacional, assim, o desenvolvedor não precisa se preocupar em alterar o código caso a aplicação precise ser executada em outras plataformas.

Existem outros fatores representados por características extremamente relevantes:

- **Simples:** possui regras muito bem definidas que facilitam o entendimento do código, com uma sintaxe clara e sem a necessidade de manipular ponteiros (alocar memória da máquina para armazenar informações).
- **Robusto:** Java foi desenhado para construir sistemas confiáveis, o que reduz imprevistos em tempo de execução.
- **Seguro:** Java é uma plataforma que proporciona um ambiente seguro para desenvolvimento, no qual a memória é gerenciada automaticamente e fornece canais de comunicação seguros, protegendo a privacidade dos dados.
- **Alto desempenho:** o código Java é compilado para ser convertido em bytecodes, que são interpretados em um ambiente de execução chamado JVM (Java Virtual Machine). Se for necessário mais desempenho, esse ambiente de execução transforma os bytecodes em código de máquina nativo para ganhar desempenho, além de ser multi-thread, permitindo executar várias tarefas simultaneamente dentro do mesmo programa.
- **Orientado a objetos:** o paradigma da programação orientada a objetos (POO) é uma das mais difundidas e utilizadas atualmente. Ela permite trabalhar com um modelo similar ao mundo real, onde os “componentes” dos softwares são objetos que podem ser reutilizados em vários pontos da aplicação, assim como os objetos da vida real.

2 PLATAFORMA JAVA

- Plataforma é um hardware ou um ambiente de software no qual um programa é executado. A maioria das plataformas pode ser descrita como uma combinação do sistema operacional com o hardware suportado. Como exemplos, temos: Microsoft Windows, Linux, Mac OS, Android etc.
- A Plataforma Java é um ambiente de software no qual os programas escritos na linguagem Java são executados. Essa plataforma é composta por dois elementos:
 - Biblioteca de classes para o desenvolvimento de aplicações Java.
 - Java Virtual Machine ou Máquina Virtual Java, que é o responsável por “executar” os programas Java.

3 EDIÇÕES DO JAVA

A plataforma Java é composta por três edições principais para construir aplicações específicas:

- **Java Standard Edition (JSE):** quando se pensa em Java, a maioria das pessoas pensa no JSE, que é a base da plataforma e possui as principais bibliotecas da linguagem, que são utilizadas nas outras edições do Java. Essa edição contempla os tipos de dados e coleções de classes responsáveis por criar aplicações desktop, conectar com banco de dados, trabalhar com segurança e redes além de manipular XML.
- **Java Enterprise Edition (JEE):** foi criado a partir do JSE, é a versão Enterprise da plataforma Java para desenvolvimento de sistemas corporativos, web, escaláveis, distribuídos que são executados dentro de servidores de aplicações.
- **Java Micro Edition (JME):** é a versão para sistemas embarcados e dispositivos móveis, uma versão enxuta do Java para rodar em dispositivos com menos recursos.

Além dessas três edições principais, o Java possui outras duas edições específicas:

- **Java Card:** permite o desenvolvimento de pequenos aplicativos para serem executados em smart cards e dispositivos similares, que possuem limitações de processamento.
- **JavaFX:** é utilizada para criar aplicações *Rich Internet Application (RIA)*, permite criar interfaces gráficas de usuário para qualquer plataforma, como desktop, web, mobile etc.

Siglas e frameworks

A plataforma Java possui vários componentes que possuem siglas, vamos conversar sobre cada uma delas:

Java Virtual Machine – JVM

A JVM é responsável por interpretar (executar) os arquivos Java (outras linguagens como Scala e Kotlin) compilados. É uma peça fundamental para a portabilidade do Java, o famoso “escreva uma vez e execute em qualquer lugar”.

A linguagem Java é **compilada** e **interpretada**. Compilação é o processo de “tradução” do programa escrito em uma linguagem de programação para a linguagem de máquina, para que as instruções possam ser executadas pelo processador.

Linguagens como C, C++ e Pascal são compiladas para um sistema operacional e arquitetura de hardware específicos, ou seja, depois do programa compilado, o código executável (binário) só funciona para aquele tipo de sistema operacional e arquitetura de hardware. Por exemplo, se um programa em C foi compilado para Windows 64-bit, se for necessário rodar em um ambiente diferente desse, como Windows 32-bit ou Linux, será necessário compilar novamente o programa para esse sistema específico.

Os arquivos Java (extensão .java) são compilados para gerar os bytecodes (extensão .class). Esses arquivos compilados são interpretados (executados) na

JVM. E aqui está o pulo do gato, pois cada sistema operacional possui uma máquina virtual Java.

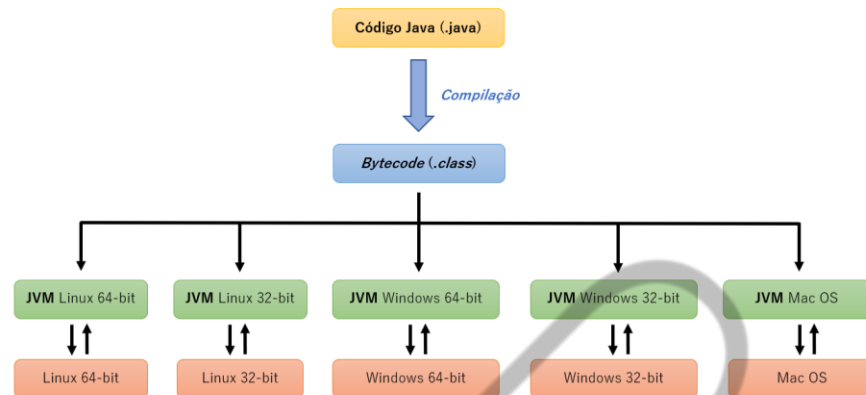


Figura 2 – Compilação e interpretação do código Java
Fonte: Elaborado pelo autor (2022)

O bytecode não roda diretamente na máquina e sim na JVM, dessa forma, é possível executar o programa independentemente da plataforma, não sendo preciso compilar para cada sistema e arquitetura de hardware. Uma vez compilado, o programa pode ser executado em qualquer JVM, não importando a plataforma que está sendo utilizada.

Java Development Kit – JDK e Java Runtime Environment – JRE

Qual a diferença do Java Runtime Environment (JRE) para o Java Development Kit (JDK)?

Se for para somente executar um programa Java, a JRE é o suficiente, pois ela possui tudo o que é necessário: a JVM e as bibliotecas. Para desenvolver programas Java, é preciso usar a JDK, que é formada pela JRE e diversas ferramentas, como o javac (compilador), javadoc (documentação), jdb (debug) etc.

Para realizar o download da JDK, acesse o link:

<https://www.oracle.com/java/technologies/downloads/>

Fundamentos Java

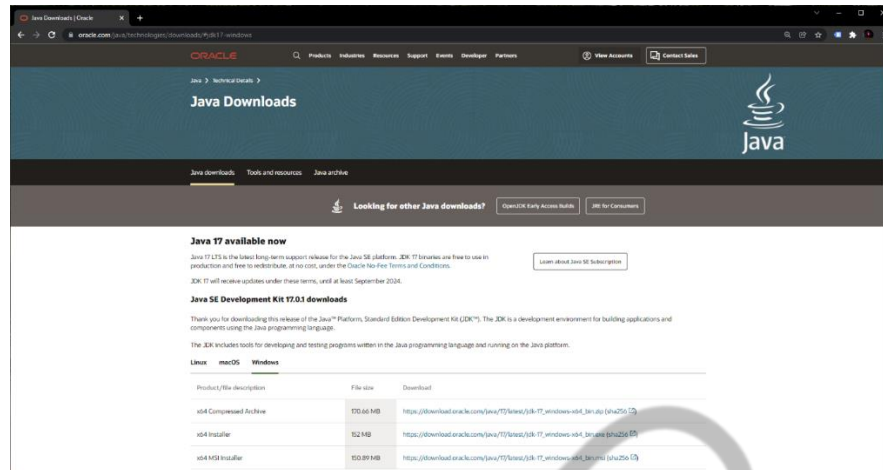


Figura 3 – Site para o download da JDK
Fonte: Elaborado pelo autor (2022)

Escolha o sistema operacional que está utilizando e realize o download. Se estiver utilizando o Windows, você pode escolher a segunda opção para baixar o instalador (Installer).

Depois de inicializar o instalador, basta clicar em Next e Next para instalar o Java na máquina.

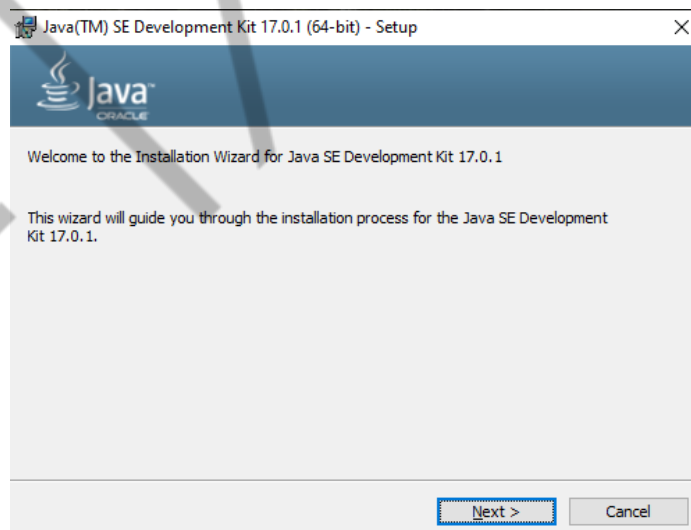
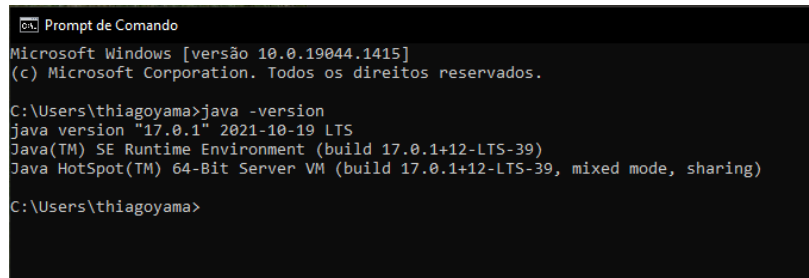


Figura 4 – Instalador do Java
Fonte: Elaborado pelo autor (2022)

Abra o prompt de comando e verifique se o Java foi instalado corretamente, para isso, utilize o comando `java -version`.



```
Prompt de Comando
Microsoft Windows [versão 10.0.19044.1415]
(c) Microsoft Corporation. Todos os direitos reservados.

C:\Users\thiagoyama>java -version
java version "17.0.1" 2021-10-19 LTS
Java(TM) SE Runtime Environment (build 17.0.1+12-LTS-39)
Java HotSpot(TM) 64-Bit Server VM (build 17.0.1+12-LTS-39, mixed mode, sharing)

C:\Users\thiagoyama>
```

Figura 5 – Versão do Java instalado na máquina
Fonte: Elaborado pelo autor (2022)

Frameworks do JEE

O Java Enterprise Edition (JEE) possui várias especificações para adicionar funcionalidades comuns às aplicações corporativas, como persistência de dados, segurança, mensageria, aplicação web, API etc. A lista a seguir enumera os frameworks mais utilizados e importantes do JEE:

- **Java Persistence API (JPA)** – especificação para acessar, persistir e manipular dados entre os objetos Java e as tabelas de um banco de dados relacional.
- **JavaServer Faces (JSF)** – especificação para construir interfaces web baseadas em componentes.
- **JavaServer Pages (JSP)** – tecnologia que ajuda os desenvolvedores a construir páginas HTML de forma dinâmica.
- **Java API for RESTful Web Services (JAX-RS)** – especificação para a construção de API Rest ou Web Services Restful.
- **Enterprise Java Beans (EJB)** – especificação para desenvolver componentes de negócio da aplicação.
- **Java Authentication and Authorization Service (JAAS)** – especificação para a implementação de segurança em aplicações corporativas.
- **Java Message Services (JMS)** – troca de mensagens assíncronas entre aplicações.

Note que falamos sobre especificações dos frameworks que formam o JEE, ou seja, é uma documentação que diz quais classes e arquivos devem conter e como devem funcionar.

Dessa forma, uma empresa ou um desenvolvedor pode utilizar essa especificação para implementar as bibliotecas seguindo os requisitos apresentados. Por exemplo, o JPA possui várias implementações, como o Hibernate (uma das mais famosas) e o OpenJPA da Apache. Qualquer uma dessas duas implementações deve funcionar da mesma forma, já que seguem a especificação do JEE da JPA.

Já que estamos falando de frameworks Java, talvez já tenha ouvido falar de outros, como Spring, GWT do Google, Struts, Log4J, JUnit etc. O mundo Java é extenso, com milhares de frameworks que solucionam problemas corriqueiros na construção de aplicações, alguns complementam o JEE e outros possuem as mesmas funcionalidades, como é o caso do Spring, um framework muito utilizado atualmente na construção de aplicações corporativas.

Para finalizar, vamos criar, compilar e executar o primeiro programa Java!

Para isso, abra o bloco de notas e escreva o código abaixo, não se preocupe em entender o código agora, vamos ver de forma detalhada mais à frente. Neste momento, o importante é compreender o processo: escrever o código Java, compilar com uma ferramenta que baixamos junto da JDK e depois interpretar o bytecode gerado pela compilação na JVM para executar o programa.

Tome cuidado para escrever tudo igual, inclusive as letras minúsculas e maiúsculas, pois fazem toda a diferença para o Java.

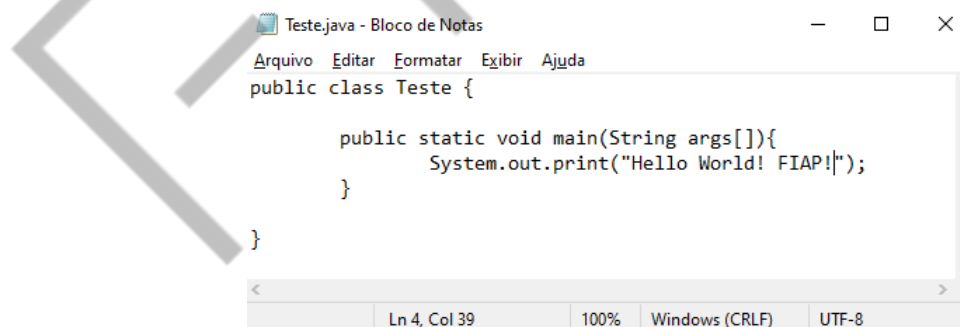


Figura 6 – Programa Hello World em Java
Fonte: Elaborado pelo autor (2022)

Agora, salve o arquivo em algum diretório, observe que a extensão deve ser .java e não .txt, já que é um código Java. No tipo, escolha "Todos os arquivos" e o nome do arquivo deve ser "Teste.java".

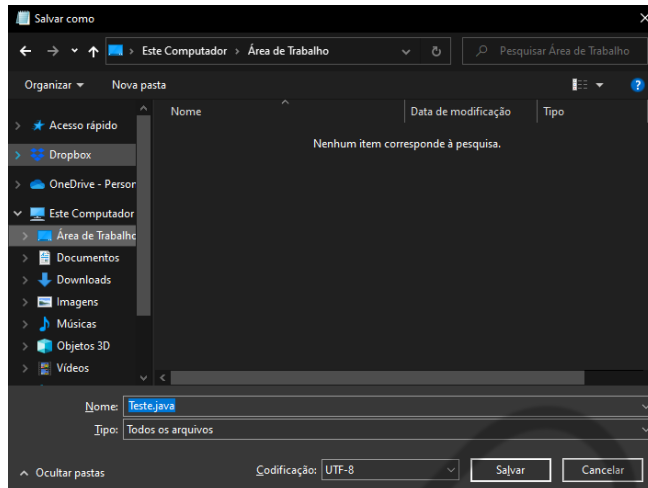


Figura 7 – Salvando a primeira classe Java
Fonte: Elaborado pelo autor (2022)

Abra o prompt de comando ou o terminal, se estiver em outro tipo de sistema operacional, e navegue até o diretório no qual você salvou o arquivo.

Depois, utilize a ferramenta “**javac**” seguida do nome e da extensão do arquivo que deseja compilar.

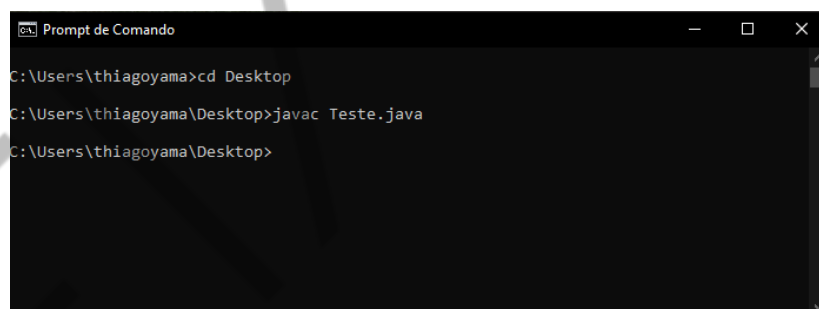


Figura 8 – Compilando a classe Java
Fonte: Elaborado pelo autor (2022)

Veja o resultado! Um novo arquivo foi criado, com a extensão .class, que é o bytecode, o código Java compilado (Figura “Classe Java compilada (bytecode)”).

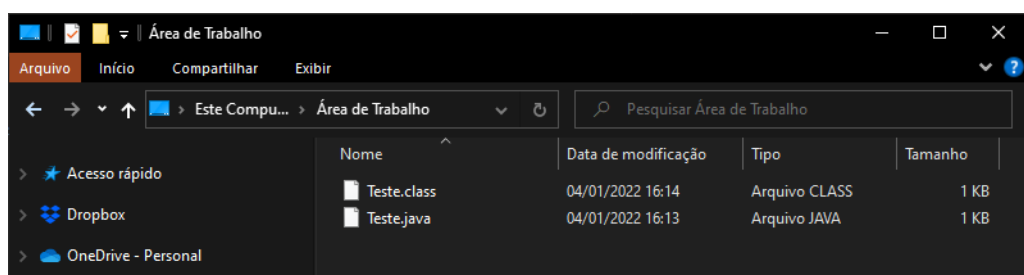
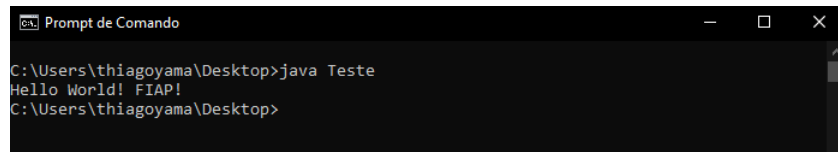


Figura 9 – Classe Java compilada (bytecode)
Fonte: Elaborado pelo autor (2022)

Para interpretar o bytecode na JVM, basta utilizar a ferramenta “**java**” seguida do nome do bytecode, assim o programa será executado!



```
Prompt de Comando
C:\Users\thiagoyama\Desktop>java Teste
Hello World! FIAP!
C:\Users\thiagoyama\Desktop>
```

Figura 10 – Interpretando o bytecode na JVM
Fonte: Elaborado pelo autor (2022)

REFERÊNCIAS

BARNES, D. J. **Programação orientada a objetos com Java**: uma introdução prática utilizando Blue J. São Paulo: Pearson, 2004.

COLHO, A. **Java com orientação a objetos**. Rio de Janeiro: Ciência Moderna, 2012.

DEITEL, P.; DEITEL, H. **Java como programar**. 10. ed. São Paulo: Pearson, 2016.

HORSTMANN, C.; CORNELL, G. **Core Java**. Volume I – Fundamentos. 8. ed. São Paulo: Pearson, 2009.

SIERRA, K.; BATES, B. **Use a cabeça! Java**. Rio de Janeiro: Alta Books, 2010.