



Orientação a Objetos em Java

Prof. Lucas Boaventura





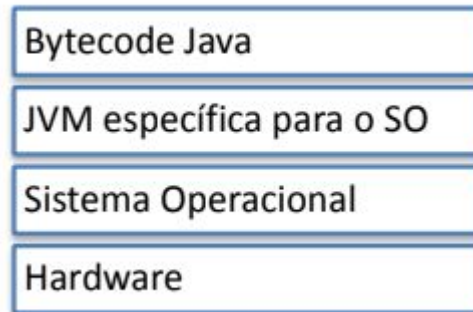
Histórico do Java

- Baseado no C++ com o intuito de ser uma linguagem Orientada a Objetos
- 1991 - Início do desenvolvimento pela Sun Microsystems (The Green Project)
- 1995 - Lançado ao público
- **“Write once, Run Anywhere”**
- 1996 - Java Developers Kit (JDK)
- 1998 - J2SE 1.2 (Standard Edition) incluindo a biblioteca Swing (API Gráfica). Compilador Just-in-Time (JIT) e o Plug-in para web browsers.
- 2010 - A Oracle comprou a Sun Microsystems e é agora a proprietária do Java



Características do Java

- Linguagem Orientada a Objetos
- Arquivos fonte **.java**
- **Bytecode** (arquivos “compilados”)
- Cada classe deve ser declarada em um arquivo separado
- Compilador **javac**
- Linguagem interpretada pela **JVM - Java Virtual Machine**





Java vs C++

- Java:

- Simplicidade
- Portabilidade
- Bytecode interpretado pela JVM ou compilado JIT
- Biblioteca Padrão mais rica em recursos
- Garbage Collection
- Herança simples

- C/C++:

- Acesso direto ao hardware (Chamadas de sistema)
- Controle direto da memória
- Código compilado para linguagem de máquina.
- Maior performance para aplicações de tempo-real
- Heranças múltiplas



Introdução ao Java

- Estrutura de um programa Básico (**MeuPrograma.java**)

o nome da classe começa com letra maiúscula. Para mais de uma palavra sempre se usa a primeira letra maiúscula.

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

- Para Compilar o bytecode: **javac MeuPrograma.java**
- Saída: **MeuPrograma.class**(bytecode)
- Executar: **java MeuPrograma**



Introdução ao Java

- Tipos Primitivos (Acesso rápido)

<i>TIPO</i>	<i>TAMANHO</i>
boolean	1 bit
byte	1 byte
short	2 bytes
char	2 bytes
int	4 bytes
float	4 bytes
long	8 bytes
double	8 bytes

- Objetos Complexos
 - String, Date, Integer, etc.
- Java é fortemente orientada a objetos
- As bibliotecas padrão do Java são todas bibliotecas de classe



Introdução ao Java

```
String fraseCompleta = "Olá!";  
String fraseCompleta = new String("Olá!");  
int tamanho = fraseCompleta.length();
```

- Ex.: Classe String

- Métodos:

- `charAt(int) : char`
 - `compareTo(object) : int`
 - `concat(String) : String`
 - `copyValueOf(char []) : String`
 - `length() : int`
 - etc

para variáveis, utiliza-se o primeiro nome começando com letra minúscula e os demais nomes começando com letra maiúscula.

Introdução ao Java



<i>Tipo de Dado</i>	<i>Classe</i>
byte	Byte
short	Short
int	Integer
long	Long
float	Float
double	Double

Introdução ao Java



```
double    valorDouble    = 130.4d;
Double    objetoDouble   = new Double(valorDouble);
byte      meuByte        = objetoDouble.byteValue();
int       meuInt          = objetoDouble.intValue();
float     meuFloat        = objetoDouble.floatValue();
String    minhaString     = objetoDouble.toString();
```



Introdução ao Java

- Java API Docs (<https://docs.oracle.com/en/java/javase/11/docs/api/index.html>)

OVERVIEW

MODULE

PACKAGE

CLASS

USE

TREE

DEPRECATED

INDEX

HELP

Java SE 11 & JDK 11

ALL CLASSES

SEARCH:

Java® Platform, Standard Edition & Java Development Kit Version 11 API Specification

This document is divided into two sections:

Java SE
The Java Platform, Standard Edition (Java SE) APIs define the core Java platform for general-purpose computing. These APIs are in modules whose names start with `java`.

JDK
The Java Development Kit (JDK) APIs are specific to the JDK and will not necessarily be available in all implementations of the Java SE Platform. These APIs are in modules whose names start with `jdk`.

All Modules

Java SE

JDK

Other Modules

Module	Description
<code>java.base</code>	Defines the foundational APIs of the Java SE Platform.
<code>java.compiler</code>	Defines the Language Model, Annotation Processing, and Java Compiler APIs.
<code>java.datatransfer</code>	Defines the API for transferring data between and within applications.
<code>java.desktop</code>	Defines the AWT and Swing user interface toolkits, plus APIs for accessibility, audio, imaging, printing, and JavaBeans.
<code>java.instrument</code>	Defines services that allow agents to instrument programs running on the JVM.
<code>java.logging</code>	Defines the Java Logging API.
<code>java.management</code>	Defines the Java Management Extensions (JMX) API.
<code>java.management.rmi</code>	Defines the RMI connector for the Java Management Extensions (JMX) Remote API.
<code>java.naming</code>	Defines the Java Naming and Directory Interface (JNDI) API.
<code>java.net.http</code>	Defines the HTTP Client and WebSocket APIs.
<code>java.prefs</code>	Defines the Preferences API.
<code>java.rmi</code>	Defines the Remote Method Invocation (RMI) API.
<code>java.scripting</code>	Defines the Scripting API.
<code>java.se</code>	Defines the API of the Java SE Platform.
<code>java.security.jgss</code>	Defines the Java bindings of the JGSS (Java Security Extensions) API (JGSS API).