Curso de Java POO

Em 1991, a Sun Microsystems, com a visão de que computadores e aparelhos eletrodomésticos iriam trabalhar em conjunto no futuro, iniciou um projeto para desenvolver um dispositivo de controle para televisão interativa. Para isso, uma máquina virtual e uma linguagem de programação foram desenvolvidas. Essa linguagem, baseada em C/C++, recebeu, do seu criador James Gosling, o nome de OAk. Contudo, foi descoberto que esse nome já existia e o nome da linguagem passou a ser Java, em referência a uma especialidade de café vindo da ilha de Java, na Indonésia, teoricamente bastante consumido pelos participantes do projeto. Mudança de planos: o mercado de TV a cabo da época não aceitou bem a ideia de TV interativa (conceito que hoje é realidade). Ao perceber as possibilidades criadas pela WWW (World Wide Web, a Internet), os desenvolvedores do projeto mudaram de alvo e trabalharam para adaptar a Linguagem Java para a grande rede de computadores.

No ano de 1995, o Java foi disponibilizado ao público, na versão 1.0. Com a expansão da Internet, o Java se popularizou e a maioria dos browsers (navegadores de Internet) passou a fornecer suporte ao Java Applets (programas Java embutidos em páginas Web estáticas). A introdução dos Applets forneceu comportamento dinâmico às páginas Web estáticas, o que significa que tais páginas poderiam apresentar conteúdos diferentes para diversos acessos de visitantes, tornando a navegação muito mais interessante.

- O nome Java faz referência a um café vindo da ilha Java, na Indonésia.



No ano de **1998** , a nova versão Java 2 foi lançada com módulos específicos para o desenvolvimento de aplicações corporativas e para dispositivos móveis. Alguns anos depois, essas versões tiveram seus nomes definidos como J2EE (Enterprise Edition) e J2ME (Mobile Edition).

E em **2010**, a gigante do setor corporativo de software e hardware, Oracle Corporation, comprou a Sun Microsystems e assumiu o desenvolvimento da plataforma Java. Assim, a Oracle passou a ser responsável pela evolução do Java e coordenou eventos anuais sobre a linguagem, disponibilizou cursos profissionais de formação para desenvolvedores Java e controlou a emissão de certificados para desenvolvedores que se submeteram a provas sobre a linguagem.

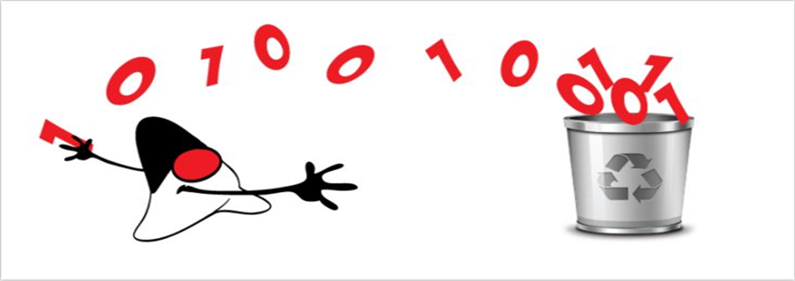
Atualmente, na versão 8 (Java SE 8, de 2014), a plataforma Java está presente em diversos dispositivos que utilizamos de forma cotidiana. Ela possui uma grande comunidade de desenvolvedores e entusiastas. Podemos citar alguns números sobre o alcance da plataforma Java no mercado de eletrônicos, como:

* Cerca de 5 milhões de estudantes Java;
* Mais de 10 milhões de desenvolvedores por todo o mundo;
* 15 bilhões de dispositivos executando Java;
* A plataforma número 1 para desenvolvimento na nuvem;
* 100% de aparelhos de leitura de Blu-ray executam Java;
* Java sendo utilizado em set-top boxes, impressoras, jogos, sistemas de navegação automotiva, casas lotéricas, dispositivos médicos, estações de pagamento de estacionamento e mais.

A [Máquina Virtual Java](https://leadfortaleza.com.br/ead2pcd/app/glossario?termo=M%C3%A1quina%20Virtual%20Java) é uma espécie de programa executado na máquina real que interpreta os bytecodes (resultados da compilação de Programas Java), tornando possível a portabilidade e a execução de programas em vários sistemas operacionais diferentes. Assim, um bytecode que foi gerado em uma máquina qualquer pode ser executado em máquinas diferentes, independentemente do [sistema operacional](https://leadfortaleza.com.br/ead2pcd/app/glossario?termo=sistema%20operacional) e da arquitetura, contanto que haja uma máquina virtual instalada.

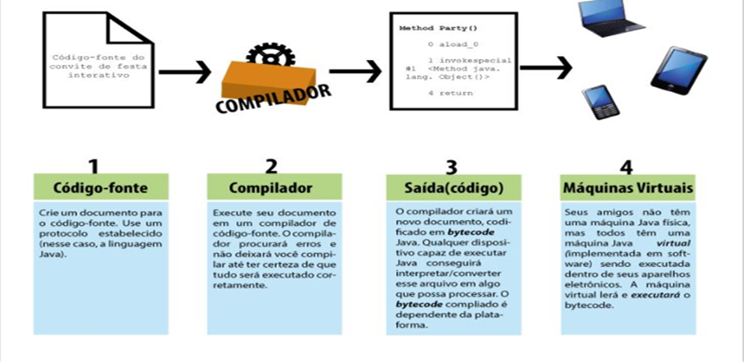
Figura 2: [Máquina Virtual Java](https://leadfortaleza.com.br/ead2pcd/app/glossario?termo=M%C3%A1quina%20Virtual%20Java) para diferentes plataformas.

A coleta de lixo é um recurso do Java que garante o bom uso da memória do computador. O coletor de lixo funciona como um vigilante, procurando, de tempos em tempos, por memória alocada desnecessariamente, liberando trechos de memória que não são mais usados. Em outras linguagens, esse trabalho deve ser feito manualmente pelo programador.

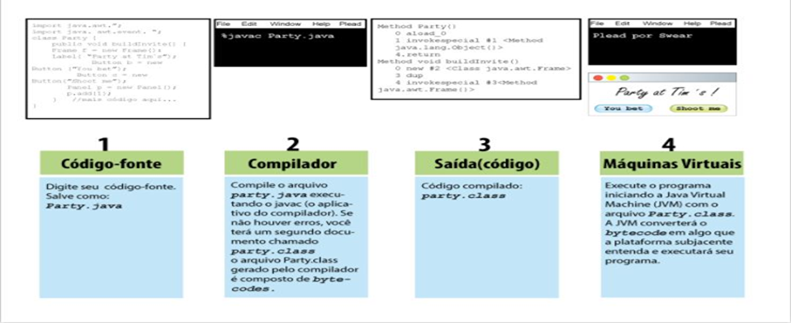
Figura 3: Coletor de lixo Java ([Garbage Collector](https://leadfortaleza.com.br/ead2pcd/app/glossario?termo=Garbage%20Collector)).

A compilação é o processo de tradução de um código-fonte escrito em linguagem de alto nível para um outro código de baixo nível que pode ser executado pelo processador. Em Java, este processo também é chamado de tradução, pois consiste em traduzir o código escrito em alto nível para bytecodes, ou seja, um código intermediário que será interpretado pela [Máquina Virtual Java](https://leadfortaleza.com.br/ead2pcd/app/glossario?termo=M%C3%A1quina%20Virtual%20Java) em vez de ser disponibilizado diretamente ao processador. Na tela a seguir, acompanhe como funciona o processo de compilação.

Observe o exemplo. O objetivo é escrever um aplicativo de convite interativo de festa que possa funcionar em qualquer dispositivo utilizado pelos seus amigos.

Figura 4: Etapas e componentes da criação de um programa Java.

Você criará um arquivo de código-fonte, compilará usando o [compilador](https://leadfortaleza.com.br/ead2pcd/app/glossario?termo=compilador) javac e, em seguida, executará o bytecode compilado em uma [Máquina Virtual Java](https://leadfortaleza.com.br/ead2pcd/app/glossario?termo=M%C3%A1quina%20Virtual%20Java).

Figura 5: Sequência de criação até a execução de programa Java.

Agora, vamos conhecer nosso primeiro código em Java! Um exemplo clássico, usado no ensino de linguagens de programação, é o “Hello World” (que significa "Olá Mundo"). Neste exemplo, é dado ao aluno um código-fonte bem pequeno, contendo apenas a estrutura suficiente para mostrar a mensagem "Hello World!" em tela. Observe o código do programa escrito em Java:

1. public class HelloWorld {

2. public static void main(String[] args) {

3. System.out.println("Hello World");

4. }

5. }

Em breve, estudaremos a fundo as características dos programas Java. Por enquanto, considere apenas que programas Java são formados por unidades de código-fonte chamadas de classes. Estas, por sua vez, são formadas por variáveis e métodos (funções). No código, a linha 1 define o nome da classe com a palavra-chave “class” seguida de “HelloWorld”. Assim, nossa classe se chamará HelloWorld. A palavra-chave “public”, no início da linha 1, está definindo que nossa classe é pública. Estudaremos mais sobre isso nas próximas aulas, mas, por enquanto, é importante que você saiba que classes públicas devem estar dentro de arquivos com o mesmo nome que elas. Assim, o arquivo que contém o código-fonte do exemplo mostrado deve, necessariamente, se chamar HelloWorld.java.

A linha 2 do programa corresponde a um método chamado Main. Ele é o método principal de uma classe Java. Quando a [JVM](https://leadfortaleza.com.br/ead2pcd/app/glossario?termo=JVM) tentar executar um programa Java, irá procurar o método Main para executar as instruções contidas nele. No nosso caso, temos apenas a instrução System.out.println. Considere que System.out.println faça o mesmo trabalho que a função Escrever, que estudamos em algoritmos, ou seja, ela exibe uma mensagem em tela. Assim, quando essa instrução for executada, o texto Hello World! será exibido.

Você deve ter percebido que, antes do método Main, temos diversas outras palavras-chave: public, static, void e String args[] entre parênteses. O conjunto dessas palavras-chave juntamente com o nome “main" constitui o que chamamos de assinatura do método. A partir da assinatura de um método, podemos saber seu nome, que tipos de dados ele retorna como resultado do seu trabalho e que variáveis ele precisa receber para trabalhar. Você aprenderá mais sobre a assinatura de um método durante o curso.

Obs: Nem todas as classes Java possuem método Main. As classes que não possuem esse método não podem ser executadas, tendo outras funcionalidades, como prover aplicabilidades auxiliares a classes executáveis.

Um detalhe interessante que já podemos analisar na assinatura do método Main é o que chamamos de parâmetros. Um método pode receber uma lista de parâmetros, variáveis que contêm valores necessários para efetuar o trabalho do método. Esses parâmetros sempre vêm entre parênteses, depois do nome do método. No caso do método Main cujo conceito discutimos nesta aula, a variável String args[] é o único parâmetro, sendo “args” uma variável do tipo vetor (por isso o símbolo "[]") de String. String é um tipo de dado de texto em Java. Na nossa classe HelloWorld, não usamos a variável args[], mas perceberemos sua utilidade nos exemplos futuros.

Para finalizar, analisaremos como os programas em Java marcam o limite entre blocos de código. Percebeu os pares de chaves, { e }, dentro do código? Eles são delimitadores de bloco. Isso significa que o código que está dentro de um mesmo par de chaves compõe um bloco. Blocos podem estar um dentro dos outros. Por exemplo, o método Main é um bloco de código (está entre chaves), que, por sua vez, está dentro do bloco da classe HelloWorld, que também possui seu par de chaves. Diferentemente dos pseudocódigos em Portugol, que usamos até então, em Java, é necessário usar esses pares de chaves para delimitar os limites de cada bloco de código. Estudaremos isso com mais detalhes futuramente.

### **CRIAÇÃO E EXECUÇÃO DE UM PROGRAMA JAVA**

Como foi apresentado no tópico anterior, os códigos-fonte escritos em Java são compilados usando o programa javac, que está presente na instalação do JDK, tornando-se bytecodes. Quando executamos um programa Java, estamos, na verdade, submetendo esses bytecodes para serem interpretados pela [Máquina Virtual Java](https://leadfortaleza.com.br/ead2pcd/app/glossario?termo=M%C3%A1quina%20Virtual%20Java) ([JVM](https://leadfortaleza.com.br/ead2pcd/app/glossario?termo=JVM)) que, por sua vez, se comunica com o [sistema operacional](https://leadfortaleza.com.br/ead2pcd/app/glossario?termo=sistema%20operacional) do dispositivo para executar as instruções do programa Java de acordo com as particularidades do dispositivo. Esse processo pode ser dividido em 3 partes:

## **Edição**

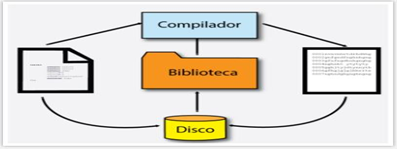
Para criar ou editar um código-fonte Java, você precisa apenas de um editor de texto simples. Porém, devido ao nível de complexidade que alguns códigos podem alcançar, é comum que os desenvolvedores utilizem um programa dedicado para escrever código-fonte Java. Esses programas são chamados de [IDE](https://leadfortaleza.com.br/ead2pcd/app/glossario?termo=IDE) (Integrated Development Environment) ou Ambientes Integrados de Desenvolvimento; eles facilitam o processo de desenvolvimento por automatizarem alguns passos da escrita de código, execução, testes e empacotamento (que significa juntar todas as partes do programa em uma versão de entrega). Inicialmente, as atividades de programação Java da Oficina serão realizadas aqui mesmo, na plataforma, e quando você passar a estudar sobre POO II (Programação Orientada a Objetos II) com Java, aprenderá a instalar o Java e conhecerá algumas [IDE](https://leadfortaleza.com.br/ead2pcd/app/glossario?termo=IDE)s, como NetBeans e Eclipse.

Os arquivos de código-fonte Java são salvos no disco do computador com a extensão .java. Por exemplo, o programa apresentado como exemplo no tópico anterior deveria ser armazenado em um arquivo chamado HelloWorld.java.

## **Compilação**

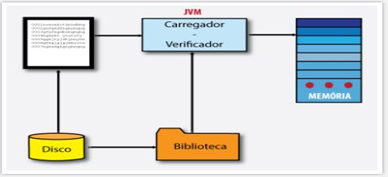
Durante a compilação de um código-fonte Java, o javac ([compilador](https://leadfortaleza.com.br/ead2pcd/app/glossario?termo=compilador) da plataforma Java) lê o código que você escreveu e consulta bibliotecas da própria linguagem. Isso ocorre porque muito do que você usa dentro do seu código é baseado em classes (unidade básica de código Java) próprias do Java. Nesse momento, também ocorre a verificação de escrita do código para evitar que ele possua algum erro de sintaxe, ou seja, não esteja seguindo as regras básicas de "gramática" da Linguagem Java. Se o código estiver sintaticamente correto, são gerados os bytecodes que poderão ser executados pela [JVM](https://leadfortaleza.com.br/ead2pcd/app/glossario?termo=JVM).

Assim como o código fonte original, os bytecodes também são salvos na forma de arquivos no disco. Estes arquivos têm extensão .class. Se compilarmos um arquivo chamado HelloWorld.java, o bytecode gerado deverá se chamar HelloWorld.class.



## **Execução**

Durante a fase de execução do programa Java, a [JVM](https://leadfortaleza.com.br/ead2pcd/app/glossario?termo=JVM) carrega os bytecodes do disco rígido e o coloca na memória do computador (como qualquer outro programa). A [JVM](https://leadfortaleza.com.br/ead2pcd/app/glossario?termo=JVM) acessará as bibliotecas da linguagem novamente para verificar a autenticidade de determinadas partes do programa. Se o código for identificado como seguro, após a verificação, o interpretador dentro da [JVM](https://leadfortaleza.com.br/ead2pcd/app/glossario?termo=JVM) inicia a execução das instruções desse programa. Nesta fase, a [JVM](https://leadfortaleza.com.br/ead2pcd/app/glossario?termo=JVM) coordena todas as operações, principalmente o acesso à memória e ao disco por parte do programa, sempre obedecendo às regras do [sistema operacional](https://leadfortaleza.com.br/ead2pcd/app/glossario?termo=sistema%20operacional) local.



### **PLATAFORMA JAVA**

A figura a seguir resume a relação entre cada um dos componentes presentes na execução de um programa Java.

Vamos analisá-la de cima para baixo. No topo da figura, temos o programa Java que desejamos executar. Por exemplo, o programa que apresentamos como exemplo, HelloWorld.java, está apoiado, ou seja, ele depende da API Java e da [Máquina Virtual Java](https://leadfortaleza.com.br/ead2pcd/app/glossario?termo=M%C3%A1quina%20Virtual%20Java). A API Java é a biblioteca da linguagem. As instruções do nosso programa “HelloWorld” e as suas dependências da API Java são executadas na [JVM](https://leadfortaleza.com.br/ead2pcd/app/glossario?termo=JVM), que funciona como uma ponte para os recursos do computador, representado na figura pela porção [Sistema Operacional](https://leadfortaleza.com.br/ead2pcd/app/glossario?termo=Sistema%20Operacional) + Hardware. A seguir, você aprenderá a realizar atividades de programação Java da Oficina. Vamos lá!

Figura 8: Componentes presentes na execução de um programa Java.