JAVA

GUIA DO PROGRAMADOR

ATUALIZADO PARA JAVA 7

2ª Edição

Peter Jandl Junior

Novatec

Editor: Rubens Prates
Revisão gramatical: Marta Almeida de Sá
Editoração eletrônica: Carolina Kuwabata
Capa: Victor Bittow

ISBN: 978-85-7522-370-3

Histórico de impressões:

Dezembro/2013 Segunda edição
Março/2007 Primeira edição (ISBN: 978-85-7522-109-9)

Novatec Editora Ltda.
Rua Luís Antônio dos Santos 110
02460-000 – São Paulo, SP – Brasil
Tel.: +55 11 2959-6529
Fax: +55 11 2950-8869
E-mail: novatec@novatec.com.br
Site: novatec.com.br
Twitter: twitter.com/novateceditora
Facebook: facebook.com/novatec
LinkedIn: linkedin.com/in/novatec
MP20131204

CAPÍTULO 1

Introdução

Este capítulo apresenta a plataforma Java, seu histórico resumido, suas principais características e aplicações, além dos recursos de programação necessários.

1.1 A Linguagem Java

Java é uma plataforma de programação apresentada ao mercado em 1995 pela Sun Microsystems que ainda provoca entusiasmo em programadores, analistas e projetistas de software, pois é o resultado de um enorme trabalho de pesquisa científica e tecnológica. Esta plataforma é um ambiente completo de desenvolvimento e execução de programas que reúne um conjunto ímpar de facilidades: uma linguagem completamente orientada a objetos, robusta, muito portátil, que permite operação em rede (com destaque à internet), a distribuição de aplicações e que incorpora diversas características voltadas à segurança.

Segundo sua especificação, o Java poderia ser assim caracterizado:

> *O Java é uma linguagem de programação de propósito geral, concorrente, baseada em classes e orientada a objetos. Projetada para ser simples o bastante para que a maioria dos programadores se torne fluente na linguagem. Java tem relação com C e C++, porém é organizada de forma diferente, com vários aspectos de C e C++ omitidos e algumas ideias de outras linguagens incluídas.*

1.2 Breve histórico

Tudo começou, em 1991, com o Green, um pequeno grupo de projeto da Sun Microsystems que pretendia criar uma nova geração de computadores portáteis, inteligentes e de comunicação avançada para ampliar suas potencialidades de uso.

Para isso decidiu-se criar também uma nova plataforma para o desenvolvimento de equipamentos de modo que seu software pudesse ser portado para os mais diferentes tipos de dispositivos. A linguagem de programação C++, devido a suas características e à experiência do grupo, foi a primeira escolha para condução do projeto, mas não permitia realizar com facilidade tudo aquilo que se vislumbrava.

James Gosling, um dos líderes da equipe, propôs criar uma nova linguagem que pudesse conter tudo aquilo considerado importante e que ainda fosse simples, portátil e fácil de programar. Surge a linguagem interpretada Oak (carvalho em inglês), batizada assim em razão da existência de uma dessas árvores em frente ao escritório de Gosling, o Green OS, e uma interface gráfica padronizada.

Após dois anos de trabalho, o grupo da Sun finalizou o Star7 (ou *7), um avançado PDA (Personal Digital Assistant) e, em 1993, surgiu a primeira oportunidade de aplicação dessa solução numa concorrência pública da Time-Warner para criação de uma tecnologia interativa para TV a cabo, vencida pela Silicon Graphics.

Problemas de copyright fazem o nome Oak ser trocado por Java, que continuava sem uso definido até 1994, quando, estimulados pelo grande crescimento da internet, Patrick Naughton e Jonathan Payne desenvolveram o programa navegador WebRunner, capaz de efetuar o download e a execução de código Java via internet. Esse navegador, com o nome HotJava, e a linguagem Java foram então apresentados formalmente pela Sun em maio de 1995 no SunWorld'95, onde o interesse pela solução se mostrou explosivo. Já no início de 1996, a Netscape Corp. lançou a versão 2.0 de seu browser Navigator incorporando as capacidades de download e execução de pequenas aplicações Java denominadas applets.

Numa iniciativa inédita em meados de 1996, a Sun disponibilizou gratuitamente para a comunidade de software um conjunto de ferramentas de desenvolvimento Java denominado JDK 1.02 (Java Development Kit) para atender inicialmente as plataformas Sun Solaris e Microsoft Windows 95/NT. Embora retendo os direitos legais de posse dessa tecnologia, as especificações da linguagem e do ambiente de execução foram disponibilizadas publicamente, permitindo aparecer progressivamente outros kits para IBM OS/2, Unix/Linux e Apple Mac OS.

Um ano após seu anúncio, ocorreu o primeiro grande evento da linguagem Java, o JavaOne Conference, que, desde então, vem sendo usado para apresentar as novas características, os casos de sucesso, produtos e tecnologias associadas. Foi assim que se iniciou a história de sucesso do Java.

Atualmente a plataforma está organizada em três segmentos:

- **JavaME (Java Micro Edition)** – Destinado aos dispositivos computacionais móveis, tais como celulares, PDAs e set-top boxes. É composto de máquinas virtuais otimizadas para ambientes mais restritos, especificações de funcionalidades e uma API mais compacta.
- **JavaSE (Java Standard Edition)** – Integra os elementos padrão da plataforma e permite o desenvolvimento de aplicações de pequeno e médio porte. Inclui todas as APIs consideradas de base, além da máquina virtual padrão.
- **JavaEE (Java Enterprise Edition)** – Voltada para a construção de aplicações corporativas complexas. Adiciona APIs específicas aos elementos padrão da plataforma.

Desde o lançamento do Java, ocorreram poucas mudanças na linguagem (Tabela 1.1), embora sua API tenha sido largamente ampliada. A partir da versão 5, cujo projeto foi denominado Tiger, a plataforma recebeu nova numeração para refletir a maturidade e estabilidade da plataforma, sendo então chamada de Java 5. Esta foi a versão que recebeu o maior número de grandes adições na linguagem e API. A versão 6, conhecida como Mustang, também incorporou diversas melhorias na API.

Tabela 1.1 – Histórico das versões da plataforma Java

Ano	Versão	Versão interna/Major	Nome Versão	Novas características da linguagem Java
1996	1.0	1.0/44	Oak	Versão inicial
1997	1.1	1.1/45		Classes internas
1998	1.2	1.2/46	Playground	Declaração strictfp. Reflexão. Compilação JIT
2000	1.3	1.3/47	Kestrel	Tecnologia HotSpot para JVM
2002	1.4	1.4/48	Merlin	Diretiva assert. Encadeamento de exceções.
2004	5.0	1.5/49	Tiger	Autoboxing. Enumerações. Genéricos. Metadata (Anotações). Lista variável de argumentos.
2006	6.0	1.6/50	Mustang	Suporte para scripting
2011	7.0	1.7/51	Dolphin	Strings em switch. Automatic resource management (ARM) ou try-with-resources. Multi-catch. Inferência de tipos genéricos. Varargs melhorado. Novos literais.

A compra da Sun pela Oracle, em 2009, não alterou a trajetória das versões do Java. A última versão, 7, denominada Dolphin, foi liberada em 2011 e trouxe algumas novidades, como aperfeiçoamentos sintáticos, modularização da API, suporte

multilinguagem da JVM (por exemplo, para Ruby e Python), melhorias nas APIs de coleções, concorrência, anotações, além de um novo garbage collector.

Em todas essas versões o kit básico de desenvolvimento de software é denominado simplesmente JDK (Java Standard Edition Development Kit). O download da versão corrente pode ser feito em `http://www.oracle.com/technetwork/java/javase/downloads/index.html`. Tudo isso indica o sucesso e as perspectivas de longo prazo dessa tecnologia.

1.3 Características principais

A linguagem Java exibe importantes características que diferenciam-na de outras linguagens de programação, dentre as quais são destacadas: orientada a objetos, independência de plataforma, sem ponteiros, performance, segurança e multithreaded.

- **Orientada a objetos** – Java é uma linguagem puramente orientada a objetos e atende a todos os requisitos necessários para isso: oferece mecanismos de abstração, encapsulamento e hereditariedade. Com exceção de seus tipos primitivos de dados, tudo em Java são classes ou instâncias de classes.
- **Independencia de plataforma** – Java é independente de plataforma porque seus programas não são compilados para uma plataforma de hardware específica, mas, sim, como bytecodes, uma forma intermediária de código que funciona como uma linguagem de máquina para a JVM (Java Virtual Machine). A JVM é, na verdade, um interpretador de bytecodes para a plataforma na qual é executada. Como é possível implementar uma JVM para qualquer plataforma, é viável que um mesmo programa Java seja executado em qualquer dessas arquiteturas.
- **Sem ponteiros** – Java não possui ponteiros, i.e., não permite a manipulação direta de endereços de memória nem exige que os objetos criados sejam explicitamente destruídos, livrando os programadores de uma tarefa complexa. Toda a manipulação de variáveis e objetos se dá por meio de referências. A JVM inclui um mecanismo de gerenciamento de memória (o automatic garbage collector), que recupera a memória alocada para objetos não mais referenciados pelo programa.
- **Performance** – Java foi projetada para ser compacta, independente de plataforma e para utilização em rede, o que levou ao esquema de interpretação de bytecodes, o qual oferecia performance apenas razoável nas primeiras versões.

Essa limitação foi superada pela incorporação de um compilador JIT (Just In Time) na JVM que, durante a carga do programa, converte os bytecodes em código nativo e possibilita uma melhora significativa na performance dos programas Java, equiparando seu desempenho ao de programas nativos. Esta tecnologia deu origem à família de JVMs HotSpot da Sun.

- **Segurança** – Como é possível obter aplicações por meio de uma rede, Java inclui mecanismos de segurança que podem, no caso de applets, evitar, por exemplo, qualquer operação no sistema de arquivos da máquina-alvo, minimizando riscos. Esses mecanismos são flexíveis o suficiente para especificar diferentes níveis de acesso ao sistema-alvo e ainda determinar se um miniaplicativo (applet) é considerado seguro.
- **Multithreaded** – A execução concorrente de múltiplas rotinas, i.e., a utilização de fluxos de execução independentes, permite a construção de aplicações sofisticadas. Java oferece suporte para criação e uso de threads, possibilitando inclusive sua sincronização.

Além das características comentadas, Java é também uma linguagem robusta, que incentiva o controle de erros; usa tipos inteiros e ponto flutuante com aritmética compatível com os padrões IEEE; suporta caracteres Unicode; possui mecanismos de reflexão (que determinam em tempo de execução tipos e informações dos objetos em uso); dispõe de tipos genéricos; é extensível dinamicamente; além de naturalmente voltada para construção de aplicações em rede ou distribuídas.

Mesmo com tudo isso, Java ainda é uma linguagem sintática e estruturalmente simples, o que a torna uma linguagem de programação única. Exatamente por isso tornou-se a linguagem de programação orientada a objetos mais utilizada no mundo.

1.4 Ambiente Java

Java é uma linguagem independente de plataforma porque seus programas são compilados em um formato próprio denominado bytecodes, instruções de tamanho fixo que constituem a linguagem da JVM, armazenados em arquivos de classe `.class`. Em cada combinação específica de hardware e sistema operacional deve existir uma JVM apropriada, capaz de interpretar os bytecodes ou de transformá-los em código nativo que possa ser executado pelo processador do sistema. A JVM sempre utiliza os serviços oferecidos pelo sistema operacional em uso. Assim o ambiente Java é composto com a JVM, sua API e com as classes da aplicação, como mostra a Figura 1.1.

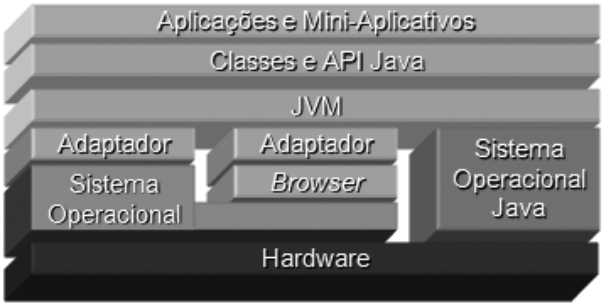

Figura 1.1 – Ambiente Java.

O desenvolvimento de programas em Java requer um editor que permita salvar o programa fonte como arquivos de extensão `.java`. Um compilador Java deve ser usado para transformar o programa fonte em bytecodes, salvos em arquivos de classe de extensão `.class`. Para executar uma aplicação Java é necessário uma JVM que interpretará ou conduzirá a execução dos arquivos de classe (Figura 1.2), usando direta ou indiretamente o código nativo do sistema.

Um ambiente mínimo é aquele que permite apenas a execução de aplicações Java, o que é possível com o uso do JavaRE (Java Runtime Environment), cujo download gratuito é possível em: `http://www.oracle.com/technetwork/java/javase/downloads/index.html`.

Figura 1.2 – Código-fonte, bytecodes e código nativo.

1.5 Recursos necessários

Um ambiente de desenvolvimento mínimo para a construção de aplicações Java requer um JDK (Java Development Kit), um conjunto útil de ferramentas de desenvolvimento considerado padrão e que inclui o JRE. Recomenda-se o uso do JDK na versão 7 (ou mais atual) de modo que todos os exemplos contidos neste material possam ser utilizados integralmente. O download gratuito do JDK pode ser feito em `http://www.oracle.com/technetwork/java/javase/downloads/index.html`.

Além do JDK, é útil dispor da documentação da API da mesma versão, a qual inclui informação sobre os pacotes de classes, as classes contidas e também as ferramentas do JDK. O download da documentação pode ser feito a partir do mesmo site de obtenção do JDK.

A edição dos programas Java pode ser feita com um editor de texto simples, tal como o notepad (bloco de notas) dos sistemas MS Windows ou gedit dos sistemas Linux, mas é convenivente uma ferramenta capaz de efetuar o destaque de sintaxe do Java. A melhor opção é o uso de um IDE (Integrated Development Environment) apropriado para Java, como o Eclipse ou NetBeans.

1.5.1 Instalando e configurando o JDK

A instalação do JDK é simples e pode ser feita seguindo as instruções do programa instalador. Para que o JDK funcione corretamente no prompt de comandos é necessário fazer alguns ajustes no sistema. O primeiro ajuste é criar uma variável de ambiente denominada `CLASSPATH` com conteúdo inicial . (ponto, caractere que indica o diretório atual). O segundo ajuste é editar a variável de ambiente `PATH`, que geralmente já existe, para adicionar o diretório em que estão instaladas as ferramentas do JDK (na plataforma Windows o usual é `C:\Arquivos de Programas\Java\jdk1.7.0_11\bin`).

Consulte a documentação do sistema operacional em uso para verificar como se deve proceder com os ajustes necessários.

1.6 Exercícios de revisão

1. Como se organiza atualmente a plataforma Java?
2. Por que os programas Java são considerados independentes de plataforma?
3. Descreva o ciclo básico de desenvolvimento de uma aplicação Java.
4. O que são o JavaRE e o JDK?
5. Quais recursos são necessários para constituir um ambiente de desenvolvimento Java?