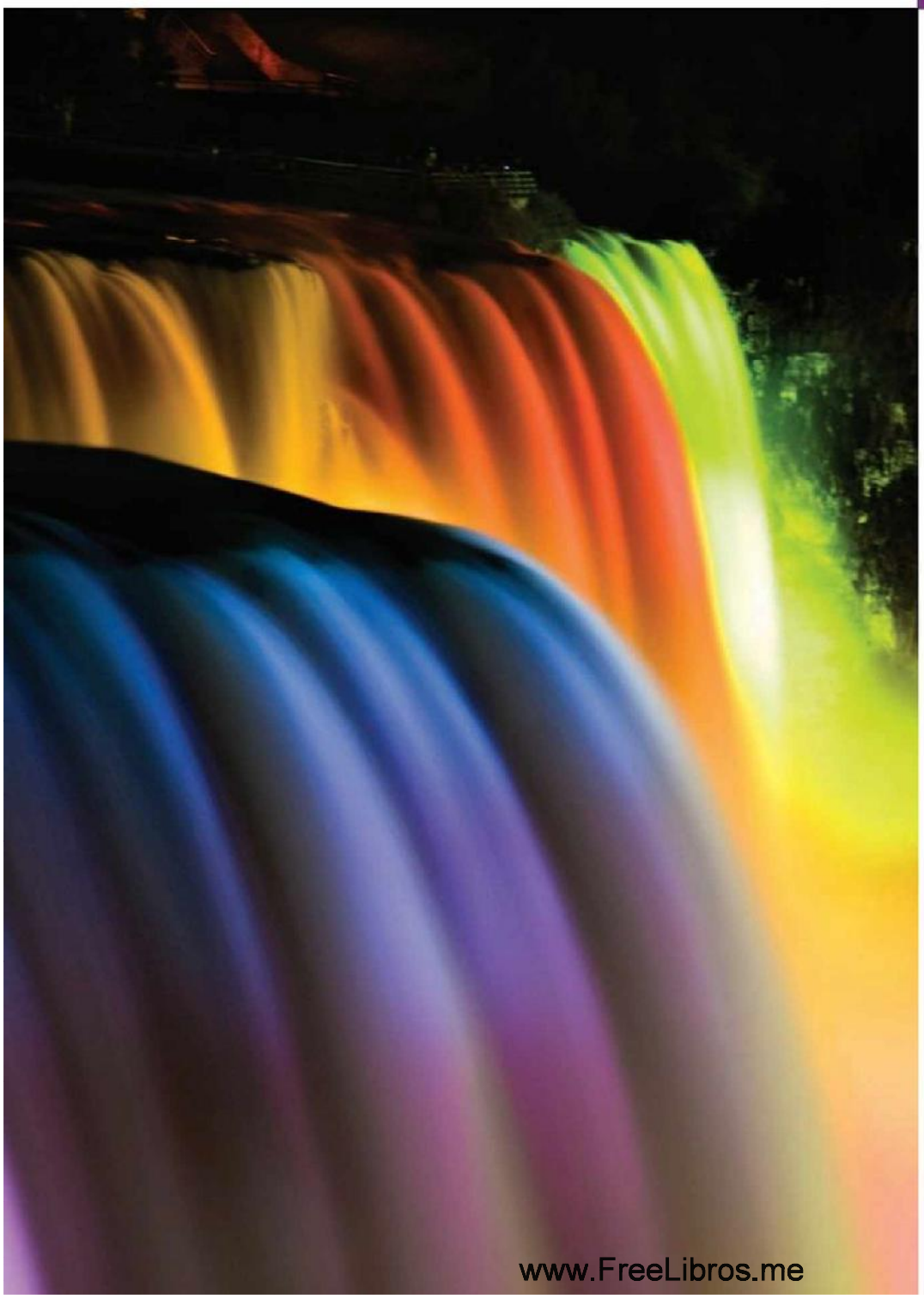
Introducción a las computadoras y a Java

1



El hombre sigue siendo la computadora más extraordinaria de todas.
—John F. Kennedy

Un buen diseño es un buen negocio.
—Thomas J. Watson, fundador de IBM

Qué maravilloso es que nadie necesite esperar un solo momento para empezar a mejorar el mundo.
—Anne Frank

Objetivos

En este capítulo aprenderá sobre:

- Los emocionantes y recientes acontecimientos en el campo de las computadoras.
- Los conceptos básicos de hardware, software y redes.
- La jerarquía de datos.
- Los distintos tipos de lenguajes de programación.
- Los conceptos básicos de la tecnología de objetos.
- La importancia de Internet y de la Web.
- Un típico entorno de desarrollo de programas en Java.
- Cómo probar una aplicación en Java.
- Algunas de las recientes tecnologías clave de software.
- La forma en que las computadoras le pueden ayudar a hacer una diferencia.

Plan general

1.1 Introducción

Bienvenido a Java: el lenguaje de programación de computadoras más utilizado en el mundo. Usted ya está familiarizado con las poderosas tareas que realizan las computadoras. Mediante este libro de texto, usted escribirá instrucciones para ordenar a las computadoras que realicen esos tipos de tareas. El *software* (las instrucciones que usted escribe) controla el *hardware* (las computadoras).

Aprenderá sobre la *programación orientada a objetos*: la metodología de programación clave de la actualidad. En este texto creará y trabajará con muchos *objetos de software*.

Java es el lenguaje preferido para satisfacer las necesidades de programación empresariales de muchas organizaciones. También se ha convertido en el lenguaje de elección para implementar aplicaciones basadas en Internet y software para dispositivos que se comunican a través de una red.

Hoy en día hay en uso más de mil millones de computadoras de propósito general, además de miles de millones de teléfonos celulares, teléfonos inteligentes (smartphones) y dispositivos portátiles (como las computadoras tipo tableta) habilitados para Java. De acuerdo con un estudio realizado por eMarketer, el número de usuarios móviles de Internet llegará a cerca de 134 millones para 2013.[1] Otros estudios han proyectado ventas de teléfonos inteligentes que sobrepasa a las ventas de computadoras personales en 2011[2] y ventas de tabletas que representarán cerca del 20% de todas las ventas de computadoras personales para 2015.[3] Para 2014, se espera que el mercado de las aplicaciones de teléfonos inteligentes exceda los $40 mil millones,[4] lo cual generará oportunidades importantes para la programación de aplicaciones móviles.

Ediciones de Java: SE, EE y ME

Java ha evolucionado con tanta rapidez que esta novena edición de *Cómo programar en Java* —basada en **Java Standard Edition 6 (Java SE 6)** con módulos opcionales sobre las nuevas características de **Java SE 7** —se publicó sólo 15 años después de la primera edición. Java se utiliza en un espectro tan amplio de aplicaciones, que cuenta con otras dos ediciones. **Java Enterprise Edition (Java EE)** está orientada hacia el desarrollo de aplicaciones de red distribuidas, de gran escala, y aplicaciones basadada en Web.

1 www.circleid.com/posts/mobile_internet_users_to_reach_134_million by_2013/.
2 www.pcworld.com/article/171380/more smartphones_than_desktop_pcs_by_2011.html.
3 www.forrester.com/ER/Press/Release/0,1769,1340,00.html.
4 *Inc.* diciembre de 2010/enero de 2011, páginas 116-123.

En el pasado, la mayoría de las aplicaciones de computadora se ejecutaban en computadoras "independientes" (que no estaban conectadas en red). En la actualidad, las aplicaciones se pueden escribir con miras a comunicarse entre computadoras en todo el mundo por medio de Internet y Web. Más adelante en este libro hablaremos sobre cómo crear dichas aplicaciones basadas en Web con Java.

Java Micro Edition (Java ME) está orientada hacia el desarrollo de aplicaciones para pequeños dispositivos con memoria restringida, como los teléfonos inteligentes BlackBerry. El sistema operativo Android de Google —que se utiliza en muchos teléfonos inteligentes, tabletas (pequeñas computadoras ligeras y móviles con pantallas táctiles), lectores electrónicos y otros dispositivos— utiliza una versión personalizada de Java que no se basa en Java ME.

La computación en la industria y la investigación

Éstos son tiempos emocionantes en el campo de la computación. Muchas de las empresas más influyentes y exitosas de las últimas dos décadas son compañías de tecnología, como Apple, IBM, Hewlett Packard, Dell, Intel, Motorola, Cisco, Microsoft, Google, Amazon, Facebook, Twitter, Groupon, Foursquare, Yahoo!, eBay y muchas más; que son fuentes de empleo importantes para las personas que estudian ciencias computacionales, sistemas de información o disciplinas relacionadas. Al momento de escribir este libro, Apple era la segunda compañía más valiosa del mundo, *con* la tecnología más preciada.[5] Las computadoras también se utilizan mucho en la investigación académica e industrial. La figura 1.1 provee unos cuantos ejemplos de las increíbles formas en que se utilizan las computadoras, tanto en la investigación como en la industria.

Nombre	Descripción
Internet	Internet —una red global de computadoras— se hizo posible gracias a la *convergencia de la computación y las comunicaciones*. Tiene sus raíces en la década de 1960; su patrocinio estuvo a cargo del Departamento de Defensa de Estados Unidos. Diseñada en un principio para conectar los sistemas de cómputo principales de alrededor de una docena de universidades y organizaciones de investigación, en la actualidad son miles de millones de computadoras y dispositivos controlados por computadora en todo el mundo los que utilizan Internet. Las computadoras descomponen las extensas transmisiones en paquetes en el extremo emisor, envían los paquetes a los receptores destinados y aseguran que se reciban en secuencia y sin errores en el extremo receptor. De acuerdo con un estudio de Forrester Research, el consumidor estadounidense promedio invierte en la actualidad la misma cantidad de tiempo en línea que el que pasa en la televisión (`forrester.com/rb/Research/understanding_changing_needs_of_us_online_consumer/g/id/57861/t/2`).
Proyecto Genoma Humano	El Proyecto Genoma Humano se fundó para identificar y analizar los más de 20,000 genes en el ADN humano. El proyecto utilizó programas de computadora para analizar datos genéticos complejos, determinar las secuencias de los miles de millones de pares de bases químicas que componen el ADN humano y almacenar la información en bases de datos que se han puesto a disposición de los investigadores en muchos campos. Esta investigación ha ocasionado una tremenda innovación y crecimiento en la industria de la biotecnología.

Fig. 1.1 | Unos cuantos usos para las computadoras (parte 1 de 3).

5 `www.zdnet.com/blog/apple/apple-becomes-worlds-second-most-valuable-company/9047`.

Nombre	Descripción
Red de la Comunidad Mundial	La Red de la Comunidad Mundial (`www.worldcommunitygrid.org`) es una red de computación sin fines de lucro. Las personas de todo el mundo donan el poder de procesamiento de cómputo que no utilicen, mediante la instalación de un programa de software seguro gratuito que permite a la Red de la Comunidad Mundial aprovechar ese poder sobrante cuando las computadoras están inactivas. El poder de cómputo se utiliza en lugar de las supercomputadoras para realizar proyectos de investigación científicos que están haciendo la diferencia, entre ellos: el desarrollo de energía solar a un precio asequible, el suministro de agua potable al mundo en desarrollo, la lucha contra el cáncer, la cura de la distrofia muscular, el hallazgo de medicamentos antivirales contra la influenza, el cultivo de arroz más nutritivo para las regiones que combaten la hambruna y otros más.
Imágenes para diagnóstico médico	Las exploraciones por tomografía computarizada (CT) con rayos X, también conocidas como CAT (tomografía axial computarizada), toman rayos X del cuerpo desde cientos de ángulos distintos. Se utilizan computadoras para ajustar la intensidad del rayo X, con lo cual se optimiza la exploración para cada tipo de tejido, para después combinar toda la información y crear una imagen tridimensional (3D).
GPS	Los dispositivos con Sistema de posicionamiento global (GPS) utilizan una red de satélites para obtener información con base en la ubicación. Varios satélites envían señales con etiquetas de tiempo al dispositivo GPS, el cual calcula la distancia hacia cada satélite con base en la hora en que la señal salió del satélite y se recibió. La ubicación de cada satélite y la distancia hacia cada uno de ellos se utilizan para determinar la ubicación exacta del dispositivo. Según la ubicación en la que usted se encuentre, los dispositivos GPS pueden proveer indicaciones paso a paso, ayudarle a encontrar con facilidad negocios cercanos (restaurantes, gasolineras, etcétera) y puntos de interés, o ayudarle a encontrar a sus amigos.
SYNC® de Microsoft	Ahora muchos autos Ford cuentan con la tecnología SYNC de Microsoft, la cual provee capacidades de reconocimiento y síntesis de voz (para leer mensajes de texto a los pasajeros) que le permiten usar comandos de voz para explorar música, solicitar alertas de tráfico y otras cosas más.
AMBER™ Alert	El Sistema de alerta AMBER (Desaparecidos en América: Sistema de Transmisión de Respuesta a Emergencias) se utiliza para buscar niños secuestrados. Las autoridades notifican tanto a las difusoras de TV y radio como a los funcionarios de carreteras estatales, quienes a su vez transmiten alertas en TV, radio, señales computarizadas en las carreteras, Internet y los dispositivos inalámbricos. AMBER Alert se asoció recientemente con Facebook. Cuyos usuarios pueden hacer clic en "Like" ("Me gusta") en las páginas de AMBER Alert según la ubicación, para recibir alertas en sus transmisiones de noticias.
Robots	Los robots son máquinas computarizadas que pueden realizar tareas (como; trabajos físicos), responder a los estímulos y otras cosas más. Se pueden utilizar para tareas diarias (por ejemplo, la aspiradora Roomba de iRobot), de entretenimiento (como las mascotas robóticas), combate militar, exploración espacial y en la profundidad del océano, manufactura y otras más. En 2004, el trotamundos marciano de la NASA a control remoto –que utilizaba tecnología Java—exploró la superficie para aprender sobre la historia del agua en el planeta.

Fig. 1.1 | Unos cuantos usos para las computadoras (parte 2 de 3).

Nombre	Descripción
Una laptop por niño (OLPC)	Una Laptop Por Niño (OLPC) ofrece laptops económicas, habilitadas para Internet y de bajo consumo de energía para los niños pobres en todo el mundo; gracias a ello fomentan el aprendizaje y reducen la separación digital (`one.laptop.org`). Al proveer estos recursos educativos, OLPC aumenta las oportunidades de que los niños pobres aprendan y hagan la diferencia en sus comunidades.
Programación de juegos	El negocio de los juegos de computadora es más grande que el de las películas de estreno. El desarrollo de los videojuegos más sofisticados puede costar hasta $100 millones. El juego *Call of Duty 2: Modern Warfare* de Activision, lanzado al público en noviembre de 2009, obtuvo $310 millones en sólo un día en Norteamérica y el Reino Unido (`news.cnet.com/8301-13772_3-10396593-52.html?tag=mncol;txt`)! Los *juegos sociales* en línea, que permiten a usuarios de todo el mundo competir entre sí, están creciendo con rapidez. Zynga —creador de juegos en línea populares, como *Farmville* y *Mafia Wars*— se fundó en 2007 y ya cuenta con más de 215 millones de usuarios mensuales. Para dar cabida al aumento en el tráfico, ¡Zynga agrega casi 1,000 servidores por semana (`techcrunch.com/2010/09/22/zynga-moves-1-petabyte-of-data-daily-adds-1000-servers-a-week/`)! Las consolas de videojuegos también se están volviendo cada vez más sofisticadas. El control remoto del Wii utiliza un *acelerómetro* (para detectar la inclinación y la aceleración) junto con un sensor que determina hacia dónde apunta el dispositivo, lo cual le permite responder al movimiento. Al hacer ademanes con el control remoto del Wii en la mano, usted puede controlar el videojuego en la pantalla. Con Kinect para el Xbox 360 de Microsoft, usted —el jugador— se convierte en el controlador. Kinect utiliza una cámara, un sensor de profundidad y software sofisticado para seguir el movimiento de su cuerpo, lo cual le permite controlar el juego (`en.wikipedia.org/wiki/Kinect`). Con los juegos de Kinect puede bailar, hacer ejercicio, jugar deportes, entrenar animales virtuales y varias actividades más.
TV por Internet	Los receptores de TV por Internet (como Apple TV y Google TV) le dan acceso a diversos tipos de contenido —como juegos, noticias, películas, programas de televisión y más—, con lo cual usted puede acceder a una gran cantidad de contenido bajo demanda; ya no necesita depender de los proveedores de televisión por cable o vía satélite para recibir contenido.

Fig. 1.1 | Unos cuantos usos para las computadoras (parte 3 de 3).

1.2 Computadoras: hardware y software

Una computadora es un dispositivo capaz de realizar cálculos y tomar decisiones lógicas con una rapidez increíblemente mayor que los humanos. Muchas de las computadoras personales contemporáneas pueden realizar miles de millones de cálculos en un segundo —más de lo que un humano podría realizar en toda su vida. ¡*Las supercomputadoras* ya pueden realizar *miles de billones* de instrucciones por segundo! Dicho de otra forma, una computadora de mil billones de instrucciones por segundo puede realizar en un segundo más de 100,000 cálculos *¡para cada uno de los habitantes del planeta!* ¡Y estos "límites superiores" están aumentando con rapidez!

Las computadoras procesan datos bajo el control de conjuntos de instrucciones conocidas como **programas de computadora**. Los cuales guían a la computadora a través de conjuntos ordenados de acciones especificadas por gente conocida como **programadores** de computadoras. A los programas que se ejecutan en una computadora se les denomina **software**. En este libro aprenderá la metodología de programación clave de la actualidad que mejora la productividad del programador, con lo cual se reducen los costos de desarrollo del software: *programación orientada a objetos*.

Una computadora consiste en varios dispositivos conocidos como **hardware** (teclado, pantalla, ratón, discos duros, memoria, unidades de DVD y unidades de procesamiento). Los costos de las computadoras *han disminuido en forma espectacular*, debido a los rápidos desarrollos en las tecnologías de hardware y software. Las computadoras que ocupaban grandes habitaciones y que costaban millones de dólares hace algunas décadas, ahora pueden colocarse en las superficies de chips de silicio más pequeños que una uña, y con un costo de quizá unos cuantos dólares cada uno. Aunque suene irónico, el silicio es uno de los materiales más abundantes en el planeta (es uno de los ingredientes de la arena común). La tecnología de los chips de silicio ha vuelto tan económica a la tecnología de la computación que hay más de mil millones de computadoras de uso general funcionando a nivel mundial, y se espera que esta cifra se *duplique* en los próximos años.

Los chips de computadora (*microprocesadores*) controlan innumerables dispositivos. Entre estos **sistemas incrustados** están: frenos antibloqueo en los autos, sistemas de navegación, electrodomésticos inteligentes, sistemas de seguridad en el hogar, teléfonos celulares y teléfonos inteligentes, robots, intersecciones de tráfico inteligentes (collision avoidance systems), controles de videojuegos y más. La gran mayoría de los microprocesadores que se producen cada año están incrustados en dispositivos que no son computadoras de propósito general.[6]

Ley de Moore

Es probable que cada año, espere pagar por lo menos un poco más por la mayoría de los productos y servicios. En el caso de los campos de las computadoras y las comunicaciones se ha dado lo opuesto, en especial con relación a los costos del hardware que da soporte a estas tecnologías. Los costos del hardware han disminuido con rapidez durante varias décadas. Cada uno o dos años, las capacidades de las computadoras se *duplican* aproximadamente sin que el precio se incremente. Esta notable observación se conoce en el ámbito común como la **Ley de Moore**, y debe su nombre a la persona que identificó esta tendencia: Gordon Moore, cofundador de Intel, uno de los principales fabricantes de procesadores para las computadoras y los sistemas incrustados de la actualidad. La Ley de Moore y las observaciones relacionadas son especialmente ciertas en cuanto a la cantidad de memoria que tienen las computadoras para los programas, la cantidad de almacenamiento secundario (como el almacenamiento en disco) que tienen para guardar los programas y datos durante periodos extendidos de tiempo, y las velocidades de sus procesadores: las velocidades con que las computadoras ejecutan sus programas (realizan su trabajo). Se ha producido un crecimiento similar en el campo de las comunicaciones, en donde los costos se han desplomado a medida que la enorme demanda por el ancho de banda de las comunicaciones (la capacidad de transmisión de información) atrae una competencia intensa. No conocemos otros cambios en los que la tecnología mejore con tanta rapidez y los costos disminuyan de una manera tan drástica. Dicha mejora fenomenal está fomentando sin duda la *Revolución de la información*.

1.3 Jerarquía de datos

Los elementos de datos que procesan las computadoras forman una **jerarquía de datos** que se vuelve cada vez más grande y compleja en estructura, a medida que progresamos primero a bits, luego a caracteres, después a campos y así en lo sucesivo. La figura 1.2 ilustra una porción de la jerarquía de datos. La figura 1.3 sintetiza los niveles de la jerarquía de datos.

---

6 `www.eetimes.com/electronics-blogs/industrial-control-designline-blog/4027479/Real-men-program-in-C?pageNumber=1`.

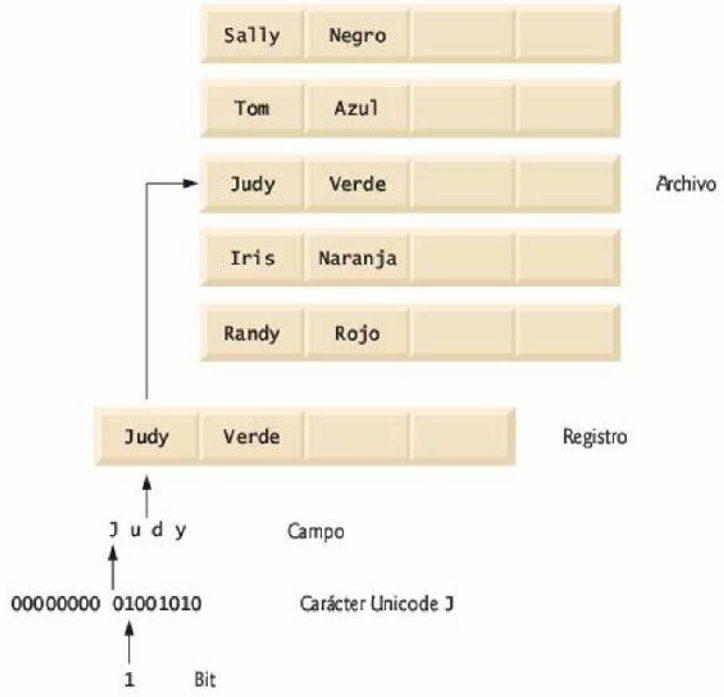


Fig. 1.2 | Jerarquía de datos.

Nivel	Descripción
Bits	El elemento de datos más pequeño en una computadora puede asumir el valor 0 o el valor 1. A dicho elemento de datos se le denomina **bit** (abreviación de "dígito binario": un dígito que puede asumir uno de dos valores). Es notable que las impresionantes funciones que realizan las computadoras sólo impliquen las manipulaciones más simples de 0s y 1s: *examinar el valor de un bit, establecer el valor de un bit e invertir el valor de un bit* (de 1 a 0 o de 0 a 1).
Caracteres	Es tedioso para las personas trabajar con datos en el formato de bajo nivel de los bits. En cambio, prefieren trabajar con *dígitos decimales* (0-9), *letras* (A-Z y a-z) y *símbolos especiales* (por ejemplo, $, @, %, &, *, (,), -, +, ", ? y /). Los dígitos, letras y símbolos especiales se conocen como **caracteres**. El **conjunto de caracteres** de la computadora es el conjunto de todos los que se utilizan para escribir programas y representar elementos de datos. Las computadoras sólo procesan los 1 y los 0, por lo que el conjunto de caracteres de una computadora representa a cada uno como un patrón de los 1 y los 0. Java usa caracteres **Unicode®** que están compuestos de dos **bytes**, cada uno de los cuales está formado a su vez de ocho bits. Unicode contiene caracteres para muchos de los idiomas en el mundo. En el apéndice L obtendrá más información sobre Unicode. En el apéndice B conocerá más información sobre el conjunto de caracteres **ASCII (Código estándar estadounidense para el intercambio de información)**: el popular subconjunto de Unicode que representa las letras mayúsculas y minúsculas, los dígitos y algunos caracteres especiales comunes.

Fig. 1.3 | Niveles de la jerarquía de datos (parte 1 de 2).

Nivel	Descripción
Campos	Así como los caracteres están compuestos de bits, los **campos** lo están por caracteres o bytes. Un campo es un grupo de caracteres o bytes que transmiten un significado. Por ejemplo, un campo compuesto de letras mayúsculas y minúsculas se puede usar para representar el nombre de una persona, y uno compuesto de dígitos decimales podría representar su edad.
Registros	Se pueden usar varios campos relacionados para componer un **registro** (el cual se implementa como una `clase` en Java). Por ejemplo, en un sistema de nómina, el registro de un empleado podría consistir en los siguientes campos (los posibles tipos para éstos se muestran entre paréntesis): • Número de identificación del empleado (un número entero) • Nombre (una cadena de caracteres) • Dirección (una cadena de caracteres) • Salario por horas (un número con punto decimal) • Ingresos del año a la fecha (un número con punto decimal) • Monto de impuestos retenidos (un número con punto decimal) Así, un registro es un grupo de campos relacionados. En el ejemplo anterior, todos los campos pertenecen al mismo empleado. Una compañía podría tener muchos empleados y un registro de nómina para cada uno.
Archivos	Un **archivo** es un grupo de registros relacionados. [*Nota:* Dicho en forma más general, un archivo contiene datos arbitrarios en formatos arbitrarios. En algunos sistemas operativos, un archivo se ve tan sólo como una *secuencia de bytes*: cualquier organización de esos bytes en un archivo, como cuando se organizan los datos en registros, es una vista creada por el programador de la aplicación]. Es muy común que una organización tenga muchos archivos, algunos de los cuales pueden contener miles de millones, o incluso billones de caracteres de información.

Fig. 1.3 | Niveles de la jerarquía de datos (parte 2 de 2).

1.4 Organización de una computadora

Sin importar las diferencias en la apariencia física, es posible percibir a las computadoras como si estuvieran divididas en varias **unidades lógicas** o secciones (figura 1.4).

Unidad lógica	Descripción
Unidad de entrada	Esta sección "receptora" obtiene información (datos y programas de cómputo) de los **dispositivos de entrada** y la pone a disposición de las otras unidades para que pueda procesarse. La mayor parte de la información se introduce a través de los teclados, pantallas táctiles y ratones. La información también puede introducirse de muchas otras formas, como hablar con su computadora, digitalizar imágenes y códigos de barras, leer dispositivos de almacenamiento secundario (como discos duros, unidades de DVD, Blu-ray Disc™ y Flash USB —también conocidas como "unidades de pulgar" o "memory sticks"), recibir video de una cámara Web e información en su computadora a través de Internet (como cuando descarga videos de YouTube™ o libros electrónicos de Amazon). Las formas más recientes de entrada son: leer los datos de la posición a través de un dispositivo GPS, y la información sobre el movimiento y la orientación mediante un acelerómetro en un teléfono inteligente o un controlador de juegos.

Fig. 1.4 | Unidades lógicas de una computadora (parte 1 de 2).

Unidad lógica	Descripción
Unidad de salida	Esta sección de "embarque" toma información que ya ha sido procesada por la computadora y la coloca en los diferentes **dispositivos de salida**, para que esté disponible fuera de la computadora. En la actualidad, la mayor parte de la información de salida de las computadoras se despliega en pantallas, se imprime en papel, se reproduce como audio o video en reproductores de medios portátiles (como los populares iPod de Apple) y pantallas gigantes en estadios deportivos, se transmite a través de Internet o se utiliza para controlar otros dispositivos, como robots y aparatos "inteligentes".
Unidad de memoria	Esta sección de "almacén" de acceso rápido, pero con relativa baja capacidad, retiene la información que se introduce a través de la unidad de entrada, para que esté disponible de manera inmediata y se pueda procesar cuando sea necesario. La unidad de memoria también retiene la información procesada hasta que la unidad de salida pueda colocarla en los dispositivos de salida. La información en la unidad de memoria es *volátil*: por lo general se pierde cuando se apaga la computadora. Con frecuencia, a la unidad de memoria se le conoce como **memoria** o **memoria principal**. Las típicas memorias principales en las computadoras de escritorio y portátiles contienen entre 1 GB y 8 GB (GB se refiere a gigabytes; un gigabyte equivale aproximadamente a mil millones de bytes).
Unidad aritmética y lógica (ALU)	Esta sección de "manufactura" realiza *cálculos* como suma, resta, multiplicación y división. También contiene los mecanismos de *decisión* que permiten a la computadora hacer cosas como, por ejemplo, comparar dos elementos de la unidad de memoria para determinar si son iguales o no. En los sistemas actuales, la ALU se implementa por lo general como parte de la siguiente unidad lógica, la CPU.
Unidad central de procesamiento (CPU)	Esta sección "administrativa" coordina y supervisa la operación de las demás. La CPU le indica a la unidad de entrada cuándo debe grabarse la información dentro de la unidad de memoria, a la ALU cuándo debe utilizarse la información de la unidad de memoria para los cálculos, y a la unidad de salida cuándo enviar la información desde la unidad de memoria hasta ciertos dispositivos de salida. Muchas de las computadoras actuales contienen múltiples CPU y, por lo tanto, pueden realizar muchas operaciones de manera simultánea. Un **procesador multinúcleo** implementa varios procesadores en un solo chip de circuitos integrados; un *procesador de doble núcleo (dual-core)* tiene dos CPU y un *procesador de cuádruple núcleo (quad-core)* tiene cuatro CPU. Las computadoras de escritorio de la actualidad tienen procesadores que pueden ejecutar miles de millones de instrucciones por segundo.
Unidad de almacenamiento secundario	Ésta es la sección de "almacén" de alta capacidad y de larga duración. Los programas o datos que no utilizan las demás unidades con frecuencia se colocan en dispositivos de almacenamiento secundario (por ejemplo, el *disco duro*) hasta que se requieran de nuevo, lo cual puede ser cuestión de horas, días, meses o incluso años después. La información en los dispositivos de almacenamiento secundario es *persistente*: se conserva aún y cuando se apaga la computadora. El tiempo para acceder a la información en almacenamiento secundario es mucho mayor que el necesario para acceder a la de la memoria principal, pero el costo por unidad de memoria secundaria es mucho menor que el correspondiente a la unidad de memoria principal. Las unidades de CD, DVD y Flash USB son ejemplos de dispositivos de almacenamiento secundario, los cuales pueden contener hasta 128 GB. Los discos duros típicos en las computadoras de escritorio y portátiles pueden contener hasta 2 TB (TB se refiere a terabytes; un terabyte equivale aproximadamente a un billón de bytes).

Fig. 1.4 | Unidades lógicas de una computadora (parte 2 de 2).

1.5 Lenguajes máquina, lenguajes ensambladores y lenguajes de alto nivel

Los programadores escriben instrucciones en diversos lenguajes de programación, algunos de los cuales los comprende directamente la computadora, mientras que otros requieren pasos intermedios de *traducción*. En la actualidad se utilizan cientos de lenguajes de computación. Éstos se dividen en tres tipos generales:

1. Lenguajes máquina
2. Lenguajes ensambladores
3. Lenguajes de alto nivel

Cualquier computadora puede entender de manera directa sólo su propio **lenguaje máquina**, el cual se define según su diseño de hardware. Por lo general, los lenguajes máquina consisten en cadenas de números (que finalmente se reducen a los 1 y 0) que instruyen a las computadoras para realizar sus operaciones más elementales, una a la vez. Los lenguajes máquina son *dependientes de la máquina* (es decir, un lenguaje máquina en particular puede usarse sólo en un tipo de computadora). Dichos lenguajes son difíciles de comprender para los humanos. Por ejemplo, he aquí la sección de uno de los primeros programas en lenguaje máquina, el cual suma el pago de las horas extras al sueldo base y almacena el resultado en el sueldo bruto:

```
+1300042774
+1400593419
+1200274027
```

La programación en lenguaje máquina era demasiado lenta y tediosa para la mayoría de los programadores. En vez de utilizar las cadenas de números que las computadoras podían entender de manera directa, los programadores empezaron a utilizar abreviaturas del inglés para representar las operaciones elementales. Estas abreviaturas formaron la base de los **lenguajes ensambladores**. Se desarrollaron *programas traductores* conocidos como **ensambladores** para convertir los primeros programas en lenguaje ensamblador a lenguaje máquina, a la velocidad de la computadora. La siguiente sección de un programa en lenguaje ensamblador también suma el pago de las horas extras al sueldo base y almacena el resultado en el sueldo bruto:

```
load    sueldobase
add     sueldoextra
store   sueldobruto
```

Aunque este código es más claro para los humanos, las computadoras no lo pueden entender sino hasta que se traduce en lenguaje máquina.

El uso de las computadoras se incrementó rápidamente con la llegada de los lenguajes ensambladores, pero los programadores aún requerían de muchas instrucciones para llevar a cabo incluso hasta las tareas más simples. Para agilizar el proceso de programación se desarrollaron los **lenguajes de alto nivel**, en donde podían escribirse instrucciones individuales para realizar tareas importantes. Los programas traductores, denominados **compiladores**, convierten programas en lenguaje de alto nivel a lenguaje máquina. Los lenguajes de alto nivel permiten a los programadores escribir instrucciones que son muy similares al inglés común, y contienen la notación matemática común. Un programa de nómina escrito en un lenguaje de alto nivel podría contener *una* instrucción como la siguiente:

```
sueldoBruto = sueldoBase + sueldoExtra
```

Desde el punto de vista del programador, los lenguajes de alto nivel son mucho más recomendables que los lenguajes máquina o ensamblador. Java es, por mucho, el lenguaje de alto nivel más utilizado.

El proceso de compilación de un programa escrito en lenguaje de alto nivel a un lenguaje máquina puede tardar un tiempo considerable en la computadora. Los programas *intérpretes* se desarrollaron para ejecutar programas en lenguaje de alto nivel de manera directa (sin el retraso de la compilación), aunque con más lentitud de la que se ejecutan en los programas compilados. Hablaremos más sobre la forma en que trabajan los intérpretes en la sección 1.9, en donde aprenderá que Java utiliza una astuta mezcla de compilación e interpretación, optimizada con base en el rendimiento, para ejecutar los programas. Los ejercicios 7.35-7.37 (en la Sección especial: Cree su propia computadora) le guiarán a través del proceso de creación de un programa intérprete.

1.6 Introducción a la tecnología de los objetos

Crear software en forma rápida, correcta y económica sigue siendo un objetivo difícil de alcanzar en una época en que la demanda de software nuevo y más poderoso va en aumento. Los *objetos*, o dicho en forma más precisa —como veremos en el capítulo 3— las *clases* de las que provienen los objetos, son en esencia componentes de software *reutilizables*. Existen objetos de fecha, objetos de hora, objetos de audio, objetos de video, objetos de automóviles, objetos de personas, etcétera. Casi *cualquier* sustantivo se puede representar de manera razonable como un objeto de software en términos de sus *atributos* (como el nombre, color y tamaño) y *comportamientos* (por ejemplo, calcular, moverse y comunicarse). Los desarrolladores de software han descubierto que al usar una metodología de diseño e implementación orientada a objetos y modular, pueden crea grupos de desarrollo de software más productivos de lo que era posible con las técnicas populares anteriores, como la "programación estructurada"; por lo general los programas orientados a objetos son más fáciles de comprender, corregir y modificar.

El automóvil como un objeto

Para ayudarle a comprender los objetos y su contenido, empecemos con una analogía simple. Suponga que desea *conducir un auto y hacer que vaya más rápido al oprimir el pedal del acelerador*. ¿Qué debe ocurrir para que usted pueda hacer esto? Bueno, antes de que pueda conducir un auto, alguien tiene que *diseñarlo*. Por lo general, un auto empieza en forma de dibujos de ingeniería, similares a los *planos de construcción* que describen el diseño de una casa. Estos dibujos de ingeniería incluyen el diseño del pedal del acelerador. El pedal *oculta* los complejos mecanismos que se encargan de que el auto aumente su velocidad, de igual forma que el pedal del freno *oculta* los mecanismos que disminuyen la velocidad del auto y el volante "oculta" los mecanismos que hacen que el auto de vuelta. Esto permite que las personas con poco o nada de conocimiento acerca de cómo funcionan los motores, los frenos y los mecanismos de la dirección puedan conducir un auto con facilidad.

Por desgracia, así como no es posible cocinar en la cocina de un plano de construcción, tampoco es posible conducir los dibujos de ingeniería de un auto. Antes de poder conducir un auto, éste debe *construirse* a partir de los dibujos de ingeniería que lo describen. Un auto completo tendrá un pedal acelerador *verdadero* para hacer que aumente su velocidad, pero aún así no es suficiente; el auto no acelerará por su propia cuenta (¡esperemos que así sea!), así que el conductor debe *oprimir* el pedal del acelerador para aumentar la velocidad del auto.

Métodos y clases

Ahora vamos a utilizar nuestro ejemplo del auto para introducir algunos conceptos clave de la programación orientada a objetos. Para realizar una tarea en una aplicación se requiere un **método**, el cual aloja las instrucciones del programa que se encargan de realizar sus tareas. El método oculta al usuario estas tareas, de la misma forma que el pedal del acelerador de un auto oculta al conductor los mecanismos para hacer que el auto vaya más rápido. En Java, creamos una unidad de programa llamada **clase** para alojar el conjunto de métodos que realizan las tareas de esa clase. Por ejemplo, una

clase que representa a una cuenta bancaria podría contener un método para *depositar* dinero en una cuenta, otro para *retirar* y un tercero para *solicitar* el saldo actual. Una clase es similar en concepto a los dibujos de ingeniería de un auto, que contienen el diseño de un pedal acelerador, volante de dirección, etcétera.

Instanciamiento

Así como alguien tiene que *construir un auto* a partir de sus dibujos de ingeniería para que otra persona lo pueda conducir después, también es necesario *crear un objeto* de una clase para que un programa pueda realizar las tareas definidas por los métodos de esa clase. Al proceso de hacer esto se le denomina *instanciamiento*. Entonces, un objeto viene siendo una **instancia** de su clase.

Reutilización

Así como los dibujos de ingeniería de un auto se pueden *reutilizar* muchas veces para construir muchos autos, también es posible *reutilizar* una clase muchas veces para crear muchos objetos. Al reutilizar las clases existentes para crear nuevas clases y programas, ahorramos tiempo y esfuerzo. La reutilización también nos ayuda a crear sistemas más confiables y efectivos, debido a que con frecuencia las clases y los componentes existentes pasan por un extenso proceso de *prueba*, *depuración* y optimización del *desempeño*. De la misma manera en que la noción de *piezas intercambiables* fue crucial para la Revolución Industrial, las clases reutilizables son cruciales para la revolución de software incitada por la tecnología de objetos.

Observación de ingeniería de software 1.1

Use un método de construcción en bloques para crear programas. Evite reinventar la rueda: use piezas existentes siempre que sea posible. Esta reutilización de software es un beneficio clave de la programación orientada a objetos.

Mensajes y llamadas a métodos

Cuando conduce un auto, al oprimir el pedal del acelerador envía un *mensaje* al auto para que realice una tarea: aumentar la velocidad. De manera similar, es posible *enviar mensajes a un objeto*. Cada mensaje se implementa como **llamada a método**, para indicar a un método del objeto que realice su tarea. Por ejemplo, un programa podría llamar al método *depositar* de un objeto cuenta de banco específico para aumentar el saldo de esa cuenta.

Atributos y variables de instancia

Además de tener capacidades para realizar tareas, un auto también tiene *atributos*: color, número de puertas, cantidad de gasolina en el tanque, velocidad actual y registro del total de kilómetros recorridos (es decir, la lectura de su velocímetro). Al igual que sus capacidades, los atributos del auto se representan como parte de su diseño en sus diagramas de ingeniería (que, por ejemplo, agregan un velocímetro y un indicador de combustible). Al conducir un auto real, estos atributos van incluidos. Cada auto conserva sus *propios* atributos. Por ejemplo, cada uno sabe cuánta gasolina hay en su tanque, pero *no* cuánta hay en los tanques de *otros* autos.

De manera similar, un objeto tiene atributos que lleva consigo a medida que se utiliza en un programa. Estos atributos se especifican como parte de la clase del objeto. Por ejemplo, un objeto cuenta bancaria tiene un *atributo saldo* que representa la cantidad de dinero en la cuenta. Cada objeto cuenta bancaria conoce el saldo de la cuenta que representa, pero *no* los saldos de las *otras* cuentas en el banco. Los atributos se especifican mediante las **variables de instancia** de la clase.

Encapsulamiento

Las clases **encapsulan** (envuelven) los atributos y métodos en objetos; los atributos y métodos de un objeto están muy relacionados entre sí. Los objetos se pueden comunicar entre sí, pero por lo general no se les permite saber cómo están implementados otros objetos; los detalles de implementación están *ocultos* dentro de los mismos objetos. Este **ocultamiento de información**, como veremos más adelante, es crucial para la buena ingeniería de software.

Herencia

Es posible crear una nueva clase de objetos con rapidez y de manera conveniente mediante la **herencia**: la nueva clase absorbe las características de una clase existente, con la posibilidad de personalizarlas y agregar características únicas propias. En nuestra analogía del auto, sin duda un objeto de la clase "convertible" *es un* objeto de la clase más *general* llamada "automóvil" pero, de manera más *específica*, el techo puede ponerse o quitarse.

Análisis y diseño orientado a objetos (A/DOO)

Pronto escribirá programas en Java. ¿Cómo creará el **código** (es decir, las instrucciones) para sus programas? Tal vez, al igual que muchos programadores, sólo encenderá su computadora y empezará a escribir. Quizás este método funcione para pequeños programas (como los que presentamos en los primeros capítulos del libro), pero ¿qué tal si le pidieran crear un sistema de software para controlar miles de cajeros automáticos para un banco importante? O ¿si le piden que trabaje con un equipo de 1,000 desarrolladores de software para crear el nuevo sistema de control de tráfico aéreo en Estados Unidos? Para proyectos tan grandes y complejos, no es conveniente tan sólo sentarse y empezar a escribir programas.

Para crear las mejores soluciones, debe seguir un proceso de **análisis** detallado para determinar los **requerimientos** de su proyecto (definir *qué* se supone que debe hacer el sistema) y desarrollar un **diseño** que los satisfaga (decidir *cómo* debe hacerlo el sistema). Lo ideal sería pasar por este proceso y revisar el diseño con cuidado (además de pedir a otros profesionales de software que lo revisen) antes de escribir cualquier código. Si este proceso implica analizar y diseñar su sistema desde un punto de vista orientado a objetos, se denomina proceso de **análisis y diseño orientado a objetos (A/DOO)**. Los lenguajes como Java son orientados a objetos. La programación en un lenguaje de este tipo, conocida como **programación orientada a objetos (POO)**, le permite implementar un diseño orientado a objetos como un sistema funcional.

El UML (Lenguaje unificado de modelado)

Aunque existen muchos procesos de A/DOO distintos, hay un solo lenguaje gráfico para comunicar los resultados de *cualquier* proceso de A/DOO que se utiliza en la mayoría de los casos. Este lenguaje, conocido como Lenguaje unificado de modelado (UML), es en la actualidad el esquema gráfico más utilizado para modelar sistemas orientados a objetos. Presentamos nuestros primeros diagramas de UML en los capítulos 3 y 4; después los utilizamos en nuestro análisis más detallado de la programación orientada a objetos en el capítulo 11. En nuestro ejemplo práctico *opcional* de ingeniería de software del ATM en los capítulos 12 y 13 presentamos un subconjunto simple de las características del UML, mientras lo guiamos por una experiencia de diseño orientada a objetos.

1.7 Sistemas operativos

Los **sistemas operativos** son sistemas de software que se encargan de hacer más conveniente el uso de las computadoras para los usuarios, desarrolladores de aplicaciones y administradores de sistemas. Los sistemas operativos proveen servicios que permiten a cada aplicación ejecutarse en forma segura, eficien-

te y *concurrente* (es decir, en paralelo) con otras aplicaciones. El software que contiene los componentes básicos del sistema operativo se denomina **kernel**. Los sistemas operativos de escritorio populares son: Linux, Windows 7 y Mac OS X. Los sistemas operativos móviles populares que se utilizan en teléfonos inteligentes y tabletas son: Android de Google, BlackBerry OS y Apple iOS (para sus dispositivos iPhone, iPad e iPod Touch).

Windows: un sistema operativo propietario

A mediados de la década de 1980 Microsoft desarrolló el **sistema operativo Windows**, el cual consiste en una interfaz gráfica de usuario creada sobre DOS: un sistema operativo de computadora personal muy popular en la época en que, para interactuar con él, los usuarios tecleaban comandos. Windows tomó prestados muchos conceptos (como los iconos, menús y ventanas) que se hicieron populares gracias a los primeros sistemas operativos Apple Macintosh, desarrollados en un principio por Xerox PARC. Windows 7 es el sistema operativo más reciente de Microsoft; algunas de sus características son; mejoras en la interfaz de usuario, un arranque más veloz, un mayor grado de refinamiento en cuanto a las características de seguridad, soporte para pantalla táctil y multitáctil, y otras más. Windows es un sistema operativo *propietario*; está bajo el control exclusivo de una compañía. Windows es por mucho el sistema operativo más utilizado en el mundo.

Linux: un sistema operativo de código fuente abierto

El sistema operativo Linux es tal vez el más grande éxito del movimiento de *código fuente abierto*. El **código fuente abierto** es un estilo de desarrollo de software que se desvía del desarrollo *propietario*, el cual predominó durante los primeros años del software. Con el desarrollo de código fuente abierto, individuos y compañías unen sus esfuerzos para desarrollar, mantener y evolucionar el software a cambio del derecho de usarlo para sus propios fines, por lo general sin costo. Por lo general el código fuente abierto es escudriñado por una audiencia mucho mayor que la del software propietario, de modo que casi siempre los errores se eliminan con más rapidez. El código fuente abierto también fomenta una mayor innovación. Sun abrió el código de su implementación del Kit de desarrollo de Java y de muchas de sus tecnologías de Java relacionadas.

Algunas organizaciones en la comunidad de código fuente abierto son: la fundación Eclipse (el Entorno integrado de desarrollo Eclipse ayuda a los programadores de Java a desarrollar software de manera conveniente), la fundación Mozilla (creadores del navegador Web Firefox), la fundación de software Apache (creadores del servidor Web Apache que se utiliza para desarrollar aplicaciones basadas en Web) y SourceForge (quien proporciona las herramientas para administrar proyectos de código fuente abierto; tiene más de 260,000 de estos proyectos en desarrollo). Las rápidas mejoras en la computación y las comunicaciones, la reducción en costos y el software de código fuente abierto han logrado que sea mucho más fácil y económico crear un negocio basado en software en la actualidad de lo que era hace unas cuantas décadas. Facebook es un gran ejemplo de ello; este sitio se inició desde un dormitorio universitario y se creó con software de código fuente abierto.[7]

El kernel de **Linux** es el núcleo del sistema operativo de código fuente abierto más popular y lleno de funcionalidades, que se distribuye en forma gratuita. Es desarrollado por un equipo de voluntarios organizados de manera informal; es popular en servidores, computadoras personales y sistemas incrustados. A diferencia de los sistemas operativos propietarios como Windows de Microsoft y Mac OS X de Apple, el código fuente de Linux (el código del programa) está disponible al público para que lo examinen y modifiquen; además se puede descargar e instalar sin costo. Como resultado, los usuarios del sistema operativo se benefician; de una comunidad de desarrolladores que depuran y mejoran el kernel de

7 `developers.facebook.com/opensource/`.

manera continua, de la ausencia de cuotas y restricciones de licencias, y de la habilidad de poder personalizar por completo el sistema operativo para cumplir necesidades específicas.

En 1991, Linus Torvalds, un estudiante de 21 años en la Universidad de Helsinki en Finlandia, empezó a desarrollar el kernel de Linux como un pasatiempo (El nombre Linux se deriva de "Linus" y "UNIX": un sistema operativo desarrollado por los Laboratorios Bell en 1969). Torvalds quería mejorar el diseño de Minix, un sistema operativo académico creado por el profesor Andrew Tanenbaum de la Vrije Universiteit en Amsterdam. El código fuente de Minix estaba disponible al público para que los profesores pudieran demostrar los conceptos básicos de la implementación de sistemas operativos a sus estudiantes.

Torvalds liberó la primera versión de Linux en 1991. La respuesta favorable condujo a la creación de una comunidad que ha continuado con el desarrollo y soporte de Linux. Los desarrolladores descargaron, probaron y modificaron el código de Linux; después enviaron correcciones de errores y retroalimentación a Torvalds, quien revisó esa información y aplicó las mejoras al código.

La liberación de Linux en 1994 integró muchas características que se encontraban por lo general en un sistema operativo maduro, con lo cual Linux se convirtió en una alternativa viable con respecto a UNIX. Las compañías de sistemas empresariales como IBM y Oracle se interesaron cada vez más en Linux, a medida que éste se volvía más estable y se extendía a nuevas plataformas.

Son varias cuestiones —el poder de mercado de Microsoft, el pequeño número de aplicaciones Linux amigables para los usuarios y la diversidad de distribuciones de Linux, tales como Red Hat Linux, Ubuntu Linux y muchas más— las que han impedido que se popularice el uso de Linux en las computadoras de escritorio. Sin embargo, este sistema operativo se ha vuelto muy popular en servidores y sistemas incrustados, como los teléfonos inteligentes basados en Android de Google.

Android

Android —el sistema operativo para dispositivos móviles y teléfonos inteligentes, cuyo crecimiento ha sido el más rápido hasta ahora— está basado en el kernel de Linux y en Java. Los programadores experimentados de Java no tienen problemas para entrar y participar en el desarrollo de aplicaciones para Android. Un beneficio de desarrollar este tipo de aplicaciones es el grado de apertura de la plataforma. El sistema operativo es gratuito y de código fuente abierto.

El sistema operativo Android fue desarrollado por Android, Inc., compañía que adquirió Google en 2005. En 2007 se formó la Alianza para los dispositivos móviles abiertos™ (OHA) —un consorcio de 34 compañías en un principio, y de 79 para el año 2010—, para continuar con el desarrollo de Android. Al mes de diciembre de 2010, ¡se activaban más de 300,000 teléfonos inteligentes con Android a diario![8] Ahora los teléfonos Android se venden más que los iPhone.[9] El sistema operativo Android se utiliza en varios teléfonos inteligentes (Motorola Droid, HTC EVO™ 4G, Samsung Vibrant™ y muchos más), dispositivos lectores electrónicos (como el Noble Nook™ de Barnes and Noble), computadoras tipo tableta (Dell Streak, Samsung Galaxy Tab y otras más), quioscos con pantallas táctiles dentro de las tiendas, autos, robots y reproductores multimedia.

Los teléfonos inteligentes Android tienen la funcionalidad de un teléfono móvil, cliente de Internet (para navegar en Web y comunicarse a través de Internet), reproductor de MP3, consola de juegos, cámara digital y demás, todo envuelto en dispositivos portátiles con *pantallas multitáctiles* a todo color —éstas pantallas le permiten controlar el dispositivo con *ademanes* en los que se requieren uno o varios toques simultáneos. Puede descargar aplicaciones de manera directa a su dispositivo Android, a través del Android Market y de otros mercados de aplicaciones. Al mes de diciembre de 2010 había cerca de 200,000 aplicaciones en el Android Market de Google.

8 www.pcmag.com/article2/0,2817,2374076,00.asp.

9 mashable.com/2010/08/02/android-outselling-iphone-2/.

Capítulos de desarrollo de aplicaciones Android en el sitio Web complementario

Debido al enorme interés en los dispositivos y aplicaciones basadas en Android, hemos integrado en el sitio Web complementario del libro una introducción de tres capítulos al desarrollo de aplicaciones Android, los cuales pertenecen a nuestro nuevo libro, *Android for Programmers: An App-Driven Approach*. Después de que aprenda Java, descubrirá que no es tan complicado empezar a desarrollar y ejecutar aplicaciones Android. Puede colocar sus aplicaciones en el Android Market en línea (`www.market.android.com`) y, si se vuelven populares, tal vez hasta pueda iniciar su propio negocio. Sólo recuerde: Facebook, Microsoft y Dell se iniciaron desde un dormitorio.

1.8 Lenguajes de programación

En esta sección veremos unos cuantos comentarios breves sobre varios lenguajes de programación populares (figura 1.5). En la siguiente sección veremos una introducción a Java.

Lenguaje de programación	Descripción
Fortran	Fortran (FORmula TRANslator, Traductor de fórmulas) fue desarrollado por IBM Corporation a mediados de la década de 1950 para utilizarse en aplicaciones científicas y de ingeniería que requerían cálculos matemáticos complejos. Aún se utiliza mucho y sus versiones más recientes son orientadas a objetos.
COBOL	COBOL (COmmon Business Oriented Language, Lenguaje común orientado a negocios) fue desarrollado a finales de la década de 1950 por fabricantes de computadoras, el gobierno estadounidense y usuarios de computadoras de la industria, con base en un lenguaje desarrollado por Grace Hopper, un oficial de la Marina de Estados Unidos y científico informático. COBOL aún se utiliza mucho en aplicaciones comerciales que requieren de una manipulación precisa y eficiente de grandes volúmenes de datos. Su versión más reciente soporta la programación orientada a objetos.
Pascal	Las actividades de investigación en la década de 1960 dieron como resultado la *programación estructurada*: un método disciplinado para escribir programas que sean más claros y fáciles de probar, depurar, y de modificar que los programas extensos producidos con técnicas anteriores. Uno de los resultados más tangibles de esta investigación fue el desarrollo del lenguaje de programación Pascal por el profesor Niklaus Wirth en 1971. Se diseñó para la enseñanza de la programación estructurada y fue popular en los cursos universitarios durante varias décadas.
Ada	Ada, un lenguaje basado en Pascal, se desarrolló bajo el patrocinio del Departamento de Defensa (DOD) de los Estados Unidos durante la década de 1970 y a principios de la década de 1980. El DOD quería un solo lenguaje que pudiera satisfacer la mayoría de sus necesidades. El nombre de este lenguaje basado en Pascal es en honor de Lady Ada Lovelace, hija del poeta Lord Byron. A ella se le atribuye el haber escrito el primer programa para computadoras en el mundo, a principios de la década de 1800 (para la Máquina Analítica, un dispositivo de cómputo mecánico diseñado por Charles Babbage). Su versión más reciente soporta la programación orientada a objetos.
Basic	Basic se desarrolló en la década de 1960 en el Dartmouth College, para introducir a los principiantes a la programación. Muchas de sus versiones más recientes son orientadas a objetos.
C	C fue implementado en 1972 por Dennis Ritchie en los Laboratorios Bell. En un principio se hizo muy popular como el lenguaje de desarrollo del sistema operativo UNIX. En la actualidad, la mayoría del código para los sistemas operativos de propósito general se escribe en C o C++.

Fig. 1.5 | Otros lenguajes de programación (parte 1 de 2).

Lenguaje de programación	Descripción
C++	C++, una extensión de C, fue desarrollado por Bjarne Stroustrup a principios de la década de 1980 en los Laboratorios Bell. C++ proporciona varias características que "pulen" al lenguaje C, pero lo más importante es que proporciona la capacidades de una programación orientada a objetos.
Objective-C	Objective-C es un lenguaje orientado a objetos basado en C. Se desarrolló a principios de la década de 1980 y después fue adquirido por la empresa Next, que a su vez fue comprada por Apple. Se ha convertido en el lenguaje de programación clave para el sistema operativo Mac OS X y todos los dispositivos operados por el iOS (como los dispositivos iPod, iPhone e iPad).
Visual Basic	El lenguaje Visual Basic de Microsoft se introdujo a principios de la década de 1990 para simplificar el desarrollo de aplicaciones para Microsoft Windows. Sus versiones más recientes soportan la programación orientada a objetos.
Visual C#	Los tres principales lenguajes de programación de Microsoft son Visual Basic, Visual C++ (basado en C++) y C# (basado en C++ y Java; desarrollado para integrar Internet y Web en las aplicaciones de computadora).
PHP	PHP es un lenguaje orientado a objetos de "secuencias de comandos" y "código fuente abierto" (vea la sección 1.7), el cual recibe soporte por medio de una comunidad de usuarios y desarrolladores; se utiliza en numerosos sitios Web, entre ellos Wikipedia y Facebook. PHP es independiente de la plataforma: existen implementaciones para todos los principales sistemas operativos UNIX, Linux, Mac y Windows. PHP también soporta muchas bases de datos, como MySQL.
Python	Python, otro lenguaje orientado a objetos de secuencias de comandos, se liberó al público en 1991. Fue desarrollado por Guido van Rossum del Instituto Nacional de Investigación para las Matemáticas y Ciencias Computacionales en Amsterdam (CWI); la mayor parte de Python se basa en Modula-3: un lenguaje de programación de sistemas. Python es "extensible": puede extenderse a través de clases e interfaces de programación.
JavaScript	JavaScript es el lenguaje de secuencias de comandos más utilizado en el mundo. Su principal uso es para agregar capacidad de programación a las páginas Web; por ejemplo, animaciones e interactividad con el usuario. Los principales navegadores Web cuentan con él.
Ruby on Rails	Ruby fue creado a mediados de la década de 1990 por Yukihiro Matsumoto; es un lenguaje de programación orientado a objetos de código fuente abierto, con una sintaxis simple que es similar a Python. Ruby on Rails combina el lenguaje de secuencias de comandos Ruby con el marco de trabajo de aplicaciones Web Rails, desarrollado por 37Signals. Su libro, *Getting Real* (`gettingreal.37signals.com/toc.php`), es una lectura obligatoria para los desarrolladores Web. Muchos desarrolladores de Ruby on Rails han reportado ganancias de productividad superiores a las de otros lenguajes, al utilizar aplicaciones Web que trabajan de manera intensiva con bases de datos. Ruby on Rails se utilizó para crear la interfaz de usuario de Twitter.
Scala	Scala (`www.scala-lang.org/node/273`), abreviación en inglés de "lenguaje escalable", fue diseñado por Martin Odersky, un profesor en la École Polytechnique Fédérale de Lausanne (EPFL) en Suiza. Se lanzó al público en 2003; utiliza los paradigmas de orientación a objetos y de programación funcional, y está diseñado para integrarse con Java. Si programa en Scala, podrá reducir de manera considerable la cantidad de código en sus aplicaciones. Twitter y Foursquare usan Scala.

Fig. 1.5 | Otros lenguajes de programación (parte 2 de 2).

1.9 Java y un típico entorno de desarrollo en Java

La contribución más importante a la fecha de la revolución del microprocesador es que hizo posible el desarrollo de las computadoras personales. Los microprocesadores están teniendo un profundo impacto en los dispositivos electrónicos inteligentes para uso doméstico. Al reconocer esto, Sun Microsystems patrocinó en 1991 un proyecto interno de investigación corporativa dirigido por James Gosling, que resultó en un lenguaje de programación orientado a objetos y basado en C++, al que Sun llamó Java.

Un objetivo clave de Java es poder escribir programas que se ejecuten en una gran variedad de sistemas computacionales y dispositivos controlados por computadora. A esto se le conoce algunas veces como "escribir una vez, ejecutar en cualquier parte".

La popularidad del servicio Web se intensificó en 1993; en ese entonces Sun vio el potencial de usar Java para agregar *contenido dinámico*, como interactividad y animaciones, a las páginas Web. Java generó la atención de la comunidad de negocios debido al fenomenal interés en Web. En la actualidad, Java se utiliza para desarrollar aplicaciones empresariales a gran escala, para mejorar la funcionalidad de los servidores Web (las computadoras que proporcionan el contenido que vemos en nuestros exploradores Web), para proporcionar aplicaciones para los dispositivos de uso doméstico (como teléfonos celulares, teléfonos inteligentes, receptores de televisión por Internet y mucho más) y para muchos otros propósitos. En 2009, Oracle adquirió Sun Microsystems. En la conferencia JavaOne 2010, Oracle anunció que el 97% de todas las computadoras de escritorio, tres mil millones de dispositivos portátiles y 80 millones de dispositivos de televisión ejecutan Java. En la actualidad hay cerca de 9 millones de desarrolladores de Java, en comparación con los 4.5 millones en 2005.[10] Ahora Java es el lenguaje de desarrollo de software más utilizado en todo el mundo.

Bibliotecas de clases de Java

Usted puede crear cada clase y método que necesite para formar sus programas de Java. Sin embargo, la mayoría de los programadores en Java aprovechan las ricas colecciones de clases existentes en las **bibliotecas de clases de Java**, que también se conocen como **API (Interfaces de programación de aplicaciones)** de Java.

Tip de rendimiento 1.1

Si utiliza las clases y métodos de las API de Java en vez de escribir sus propias versiones puede mejorar el rendimiento de sus programas, ya que estas clases y métodos están escritos de manera cuidadosa para funcionar con eficacia. Esta técnica también reduce el tiempo de desarrollo de los programas.

Tip de portabilidad 1.1

Aunque es más fácil escribir programas portables (programas que se puedan ejecutar en muchos tipos distintos de computadoras) en Java que en la mayoría de los otros lenguajes de programación, las diferencias entre los compiladores, las JVM y las computadoras pueden ocasionar que sea difícil lograr la portabilidad. El simple hecho de escribir programas en Java no garantiza la portabilidad.

Ahora explicaremos los pasos típicos utilizados para crear y ejecutar una aplicación en Java, mediante el uso de un entorno de desarrollo (el cual se ilustra en las figuras 1.6-1.10). Por lo general, los programas en Java pasan a través de cinco fases: edición, compilación, carga, verificación y ejecución. Hablaremos sobre estos conceptos en el contexto del Kit de desarrollo de Java SE (JDK). Puede descargar el JDK más actualizado y su documentación en `www.oracle.com/technetwork/java/javase/`

10 `jaxenter.com/how-many-java-developers-are-there-10462.html`.

`downloads/index.html`. *Lea la sección Antes de empezar este libro para asegurarse de configurar su computadora en forma apropiada para compilar y ejecutar programas en Java.* Tal vez también desee visitar el centro para principiantes de Java (New to Java Center) de Oracle en:

```
www.oracle.com/technetwork/topics/newtojava/overview/index.html
```

[*Nota*: este sitio Web proporciona las instrucciones de instalación para Windows, Linux y Mac OS X. Si no utiliza uno de estos sistemas operativos, consulte la documentación del entorno de Java de su sistema o pregunte a su instructor cómo puede realizar estas tareas con base en el sistema operativo de su computadora. Si encuentra un problema con éste o cualquier otro vínculo mencionado en este libro, visite el sitio `www.deitel.com/books/jhtp9/` para consultar la fe de erratas y notifíquenos su problema al correo electrónico `deitel@deitel.com`].

Fase 1: Creación de un programa

La fase 1 consiste en editar un archivo con un *programa de edición*, conocido comúnmente como *editor* (figura 1.6). Usted escribe un programa en Java (conocido por lo general como **código fuente**) por medio del editor, realiza las correcciones necesarias y guarda el programa en un dispositivo de almacenamiento secundario, como su disco duro. Un nombre de archivo que termina con la **extensión .java** indica que éste contiene código fuente en Java.

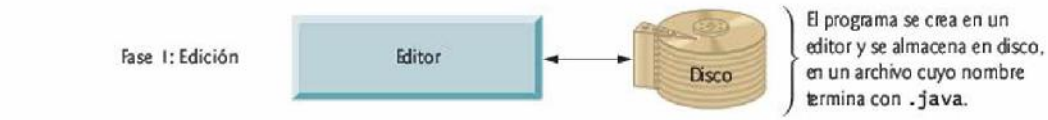


Fig. 1.6 | Entorno de desarrollo típico de Java: fase de edición.

Dos de los editores muy utilizados en sistemas Linux son `vi` y `emacs`. En Windows, basta con usar el Bloc de Notas. También hay muchos editores de freeware y shareware disponibles en línea, como EditPlus (`www.editplus.com`), TextPad (`www.textpad.com`) y jEdit (`www.jedit.org`).

Para las organizaciones que desarrollan sistemas de información extensos, hay **entornos de desarrollo integrados (IDE)** disponibles de la mayoría de los proveedores de software. Los IDE proporcionan herramientas que dan soporte al proceso de desarrollo del software, incluyendo editores para escribir y editar programas, y depuradores para localizar **errores lógicos**: errores que provocan que los programas se ejecuten en forma incorrecta. Los IDE populares son Eclipse (`www.eclipse.org`) y NetBeans (`www.netbeans.org`).

Fase 2: Compilación de un programa en Java para convertirlo en códigos de bytes

En la fase 2, el programador utiliza el comando **javac** (el **compilador de Java**) para **compilar** un programa (figura 1.7). Por ejemplo, para compilar un programa llamado `Bienvenido.java`, escriba

```
javac Bienvenido.java
```

en la ventana de comandos de su sistema (es decir, el **Símbolo del sistema** en Windows, el *indicador de shell* en Linux o la aplicación Terminal en Mac OS X). Si el programa se compila, el compilador produce un archivo **.class** llamado `Bienvenido.class` que contiene la versión compilada del programa.

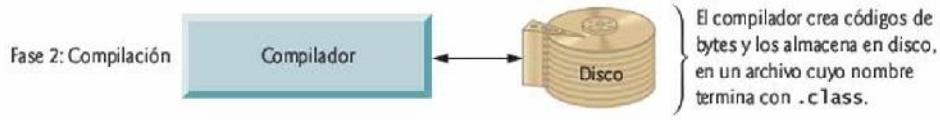


Fig. 1.7 | Entorno de desarrollo típico de Java: fase de compilación.

El compilador de Java traduce el código fuente de Java en **códigos de bytes** que representan las tareas a ejecutar en la fase de ejecución (fase 5). La **máquina virtual de Java (JVM)**, que forma parte del JDK y es la base de la plataforma Java, ejecuta los códigos de bytes. Una **máquina virtual (VM)** es una aplicación de software que simula a una computadora, pero oculta el sistema operativo y el hardware subyacentes de los programas que interactúan con ésta. Si se implementa la misma VM en muchas plataformas computacionales, las aplicaciones que ejecute se podrán utilizar en todas esas plataformas. La JVM es una de las máquinas virtuales más utilizadas en la actualidad. La plataforma NET de Microsoft utiliza una arquitectura de máquina virtual similar.

A diferencia del lenguaje máquina, que depende del hardware de una computadora específica, los códigos de bytes son instrucciones independientes de la plataforma; no dependen de una plataforma de hardware en especial. Entonces, los códigos de bytes de Java son **portables**: es decir, se pueden ejecutar los mismos códigos de bytes en cualquier plataforma que contenga una JVM que comprenda la versión de Java en la que se compilaron los códigos de bytes sin necesidad de volver a compilar el código fuente. La JVM se invoca mediante el comando `java`. Por ejemplo, para ejecutar una aplicación en Java llamada `Bienvenido`, debe escribir el comando

```
java Bienvenido
```

en una ventana de comandos para invocar la JVM, que a su vez inicia los pasos necesarios para ejecutar la aplicación. Esto comienza la fase 3.

Fase 3: Cargar un programa en memoria

En la fase 3, la JVM coloca el programa en memoria para ejecutarlo; a esto se le conoce como **cargar** (figura 1.8). El **cargador de clases** toma los archivos `.class` que contienen los códigos de bytes del programa y los transfiere a la memoria principal. El cargador de clases también carga cualquiera de los archivos `.class` que su programa utilice, y que sean proporcionados por Java: Puede cargar los archivos `.class` desde un disco en su sistema o a través de una red (como la de su universidad local o la red de la empresa, o incluso desde Internet).

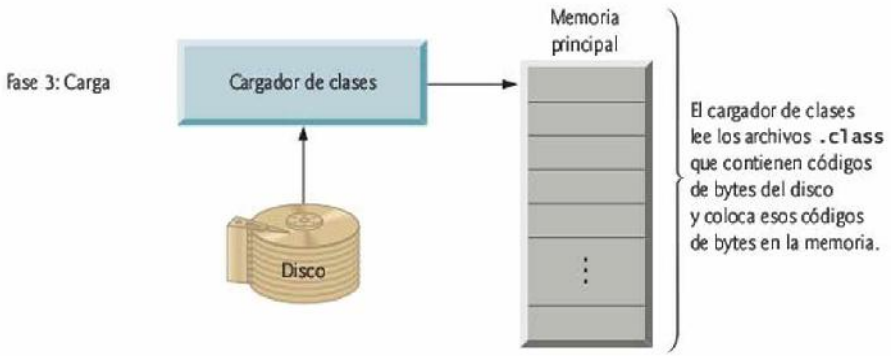


Fig. 1.8 | Entorno de desarrollo típico de Java: fase de carga.

Fase 4: Verificación del código de bytes

En la fase 4, a medida que se cargan las clases, el **verificador de códigos de bytes** examina sus códigos de bytes para asegurar que sean válidos y que no violen las restricciones de seguridad de Java (figura 1.9). Java implementa una estrecha seguridad para asegurar que los programas en Java que llegan a través de la red no dañen sus archivos o su sistema (como podrían hacerlo los virus de computadora y los gusanos).

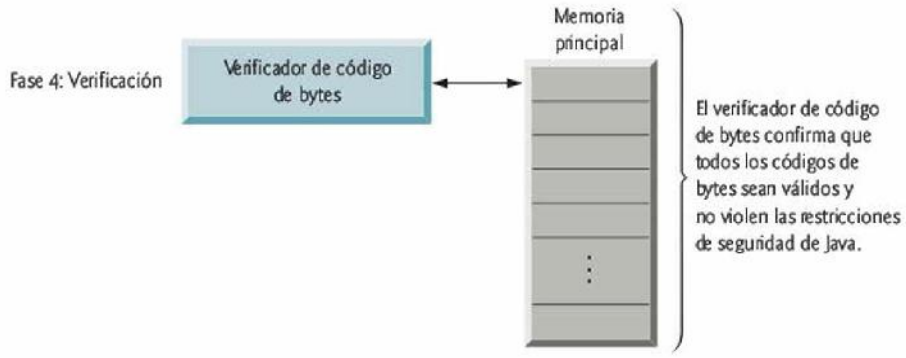


Fig. 1.9 | Entorno de desarrollo típico de Java: fase de verificación.

Fase 5: Ejecución

En la fase 5, la JVM **ejecuta** los códigos de bytes del programa, realizando así las acciones especificadas por el mismo (figura 1.10). En las primeras versiones de Java, la JVM era tan sólo un intérprete de códigos de bytes de Java. Esto hacía que la mayoría de los programas se ejecutaran con lentitud, ya que la JVM tenía que interpretar y ejecutar un código de byte a la vez. Algunas arquitecturas de computadoras modernas pueden ejecutar varias instrucciones en paralelo. Por lo general, las JVM actuales ejecutan códigos de bytes mediante una combinación de la interpretación y la denominada **compilación justo a tiempo (JIT)**. En este proceso, la JVM analiza los códigos de bytes a medida que se interpretan, en busca de *puntos activos*: partes de los códigos de bytes que se ejecutan con frecuencia. Para estas partes, un **compilador justo a tiempo (JIT)** (conocido como **compilador HotSpot de Java**) traduce los códigos de bytes al lenguaje máquina correspondiente a la computadora. Cuando la JVM encuentra estas partes compila-

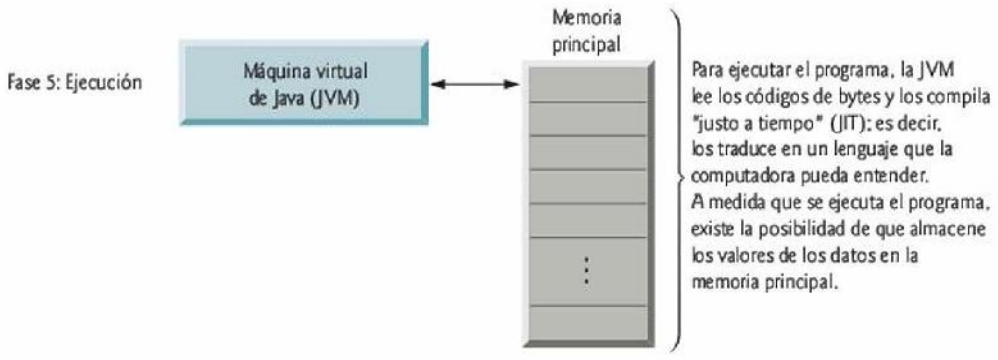


Fig. 1.10 | Entorno de desarrollo típico de Java: fase de ejecución.

das de nuevo, se ejecuta el código en lenguaje máquina, que es más rápido. Por ende, los programas en Java en realidad pasan por *dos* fases de compilación: una en la cual el código fuente se traduce a código de bytes (para tener portabilidad a través de las JVM en distintas plataformas computacionales) y otra en la que, durante la ejecución, los códigos de bytes se traducen en lenguaje máquina para la computadora actual en la que se ejecuta el programa.

Problemas que pueden ocurrir en tiempo de ejecución

Es probable que los programas no funcionen la primera vez. Cada una de las fases anteriores puede fallar, debido a diversos errores que describiremos en este texto. Por ejemplo, un programa en ejecución podría intentar una división entre cero (una operación ilegal para la aritmética con números enteros en Java). Esto haría que el programa de Java imprimiera un mensaje de error. Si esto ocurre, tendría que regresar a la fase de edición, hacer las correcciones necesarias y proseguir con las fases restantes de nuevo, para determinar que las correcciones hayan resuelto el(los) problema(s) [*Nota:* la mayoría de los programas en Java reciben o producen datos. Cuando decimos que un programa muestra un mensaje, por lo general queremos decir que aparece en la pantalla de su computadora. Los mensajes y otros datos pueden enviarse a otros dispositivos, como los discos y las impresoras, o incluso a una red para transmitirlos a otras computadoras].

Error común de programación 1.1

Los errores, como la división entre cero, ocurren a medida que se ejecuta un programa, de manera que a estos errores se les llama ***errores en tiempo de ejecución****. Los* ***errores fatales en tiempo de ejecución*** *hacen que los programas terminen de inmediato, sin haber realizado bien su trabajo. Los* ***errores no fatales en tiempo de ejecución*** *permiten a los programas ejecutarse hasta terminar su trabajo, lo que a menudo produce resultados incorrectos.*

1.10 Prueba de una aplicación en Java

En esta sección, ejecutará su primera aplicación en Java e interactuará con ella. Para empezar, ejecutará una aplicación de ATM, la cual simula las transacciones que se llevan a cabo al utilizar una máquina de cajero automático, o ATM (por ejemplo, retirar dinero, realizar depósitos y verificar los saldos de las cuentas). Aprenderá a crear esta aplicación en el ejemplo práctico *opcional* orientado a objetos que se incluye en los capítulos 12 y 13. Para los fines de esta sección vamos a suponer que está utilizando Microsoft Windows.[11]

En los siguientes pasos, ejecutará la aplicación y realizará varias transacciones. Los elementos y la funcionalidad que podemos ver en esta aplicación son típicos de lo que aprenderá a programar en este libro [*Nota:* utilizamos fuentes para diferenciar las características que se ven en una pantalla (por ejemplo, el **Símbolo del sistema**) y los elementos que no se relacionan de manera directa con una pantalla. Nuestra convención es enfatizar las características de la pantalla como los títulos y menús (por ejemplo, el menú **Archivo**) en una fuente **Helvetica sans-serif** en semi-negritas, y enfatizar los elementos que no son de la pantalla, como los nombres de archivo o los datos de entrada (como `NombrePrograma.java`) en una fuente `Lucida sans-serif`. Como tal vez ya se haya dado cuenta, la ocurrencia de definición de cada término en el texto se establece en **negritas**. En las figuras en esta sección, resaltamos en una pantalla gris claro la entrada del usuario requerida por cada paso y señalamos las partes importantes de la

11 En `www.deitel.com/books/jhtp9/`, ofrecemos una versión en Linux de esta prueba. También ofrecemos vínculos a videos que le ayudarán a empezar a trabajar con varios entornos de desarrollo integrados populares (IDE), como el Kit de desarrollo de Java SE 6 para Windows, el SKD de Eclipse para Windows, NetBeans, jGRASP, DrJava, BlueJ y el editor de texto TestPad para Windows.

aplicación. Para aumentar la visibilidad de estas características, modificamos el color de fondo de las ventanas del **Símbolo del sistema** a blanco y el color de las letras a negro]. Ésta es una versión simple que consiste de texto solamente. Más adelante en el libro, aprenderá las técnicas para rediseñar este ejemplo mediante el uso de las técnicas de GUI (interfaz gráfica de usuario).

1. ***Revise su configuración.*** Lea la sección Antes de empezar este libro para confirmar que haya instalado Java de manera apropiada en su computadora, y copiado los ejemplos del libro en su disco duro.

2. ***Localice la aplicación completa.*** Abra una ventana **Símbolo del sistema**. Para ello, puede seleccionar **Inicio | Todos los programas | Accesorios | Símbolo del sistema**. Para cambiar al directorio de la aplicación del ATM, escriba `cd C:\ejemplos\cap01\ATM` y después oprima *Intro* (figura 1.11). El comando `cd` se utiliza para cambiar de directorio.

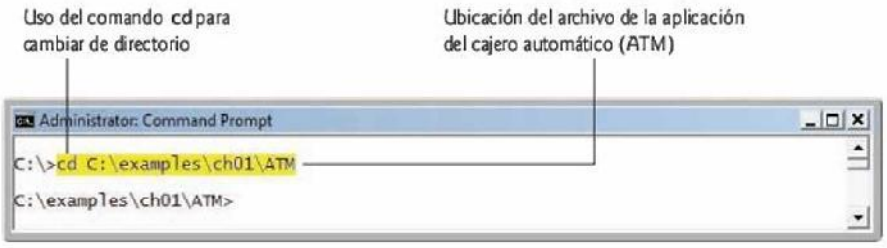


Fig. 1.11 | Abrir una ventana **Símbolo del sistema** en Windows XP y cambiar de directorio.

3. ***Ejecute la aplicación del ATM.*** Escriba el comando `java EjemploPracticoATM` y oprima *Intro* (figura 1.12). Recuerde que el comando `java`, seguido del nombre del archivo `.class` de la aplicación (en este caso, `EjemploPracticoATM`), ejecuta la aplicación. Si especificamos la extensión `.class` al usar el comando `java` se produce un error [*Nota:* los comandos en Java son sensibles a mayúsculas/minúsculas. Es importante escribir el nombre de esta aplicación con las letras A, T y M mayúsculas en "ATM", una letra E mayúscula en "Ejemplo" y una letra P mayúscula en "Practico". De lo contrario, la aplicación no se ejecutará.] Si recibe el mensaje de error "`Exception in thread "main" java.lang.NoClassDefFoundError:EjemploPracticoATM`", entonces su sistema tiene un problema con `CLASSPATH`. Consulte la sección Antes de empezar este libro para obtener instrucciones acerca de cómo corregir este problema.

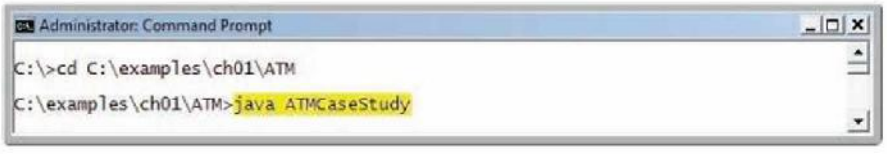


Fig. 1.12 | Uso del comando `java` para ejecutar la aplicación del ATM.

4. ***Escriba un número de cuenta.*** Cuando la aplicación se ejecuta por primera vez, muestra el mensaje "`¡Bienvenido!`" y le pide un número de cuenta. Escriba `12345` en el indicador "`Escriba su numero de cuenta:`" (figura 1.13) y oprima *Intro*.

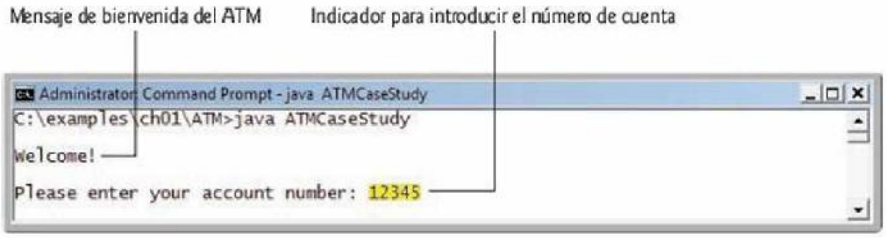


Fig. 1.13 | La aplicación pide al usuario un número de cuenta.

5. ***Escriba un NIP.*** Una vez que introduzca un número de cuenta válido, la aplicación mostrará el indicador "`Escriba su NIP:`". Escriba "`54321`" como su NIP (Número de Identificación Personal) válido y oprima *Intro*. A continuación aparecerá el menú principal del ATM, que contiene una lista de opciones (figura 1.14). En el capítulo 14 le mostraremos cómo puede introducir un NIP en forma privada mediante el uso de un objeto `JPasswordField`.

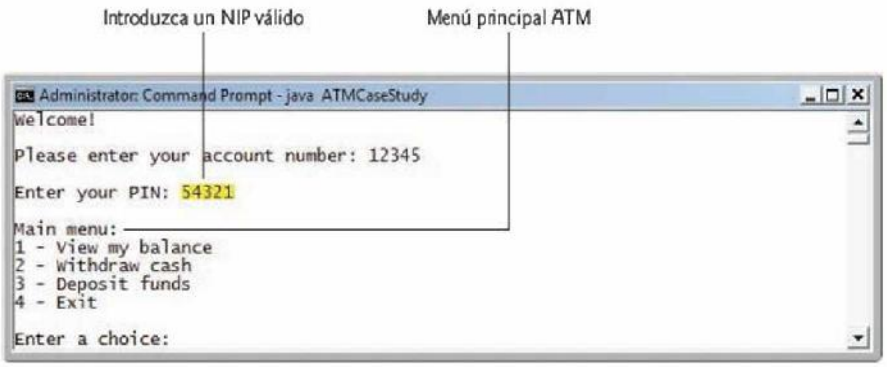


Fig. 1.14 | El usuario escribe un número NIP válido y aparece el menú principal de la aplicación del ATM.

6. ***Revise el saldo de la cuenta.*** Seleccione la opción 1, "`Ver mi saldo`" del menú del ATM (figura 1.15). A continuación la aplicación mostrará dos números: `Saldo disponible` (`$1,000.00`) y `Saldo total` (`$1,200.00`). El saldo disponible es la máxima cantidad de dinero en su cuenta, disponible para retirarla en un momento dado. En algunos casos, ciertos fondos como los depósitos recientes, no están disponibles de inmediato para que el usuario pueda retirarlos, por lo que el saldo disponible puede ser menor que el saldo total, como en este caso. Después de mostrar la información de los saldos de la cuenta, se vuelve a mostrar el menú principal de la aplicación.

7. ***Retire dinero de la cuenta.*** Seleccione la opción 2, "`Retirar efectivo`", del menú de la aplicación. A continuación aparecerá (figura 1.16) una lista de montos en dólares (por ejemplo: 20, 40, 60, 100 y 200). También tendrá la oportunidad de cancelar la transacción y regresar al menú principal. Retire $100 seleccionando la opción 4. La aplicación mostrará el mensaje "`Tome su efectivo ahora`" y regresará al menú principal. [*Nota:* por desgracia, esta aplicación sólo *simula* el comportamiento de un verdadero ATM, por lo cual no dispensa efectivo en realidad].

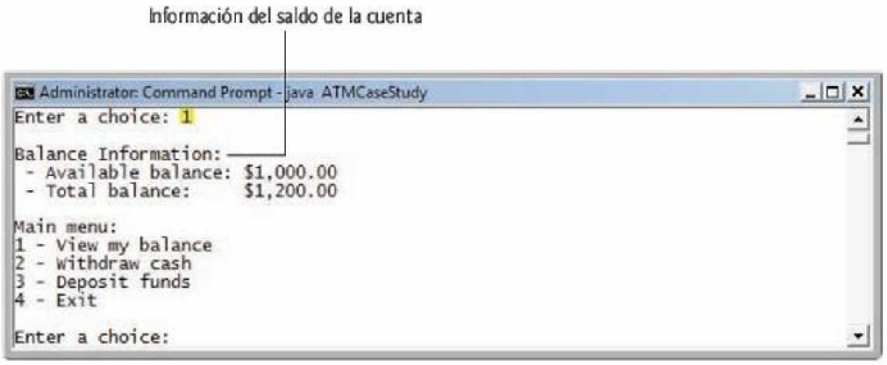


Fig. 1.15 | La aplicación del ATM muestra la información del saldo de la cuenta del usuario.

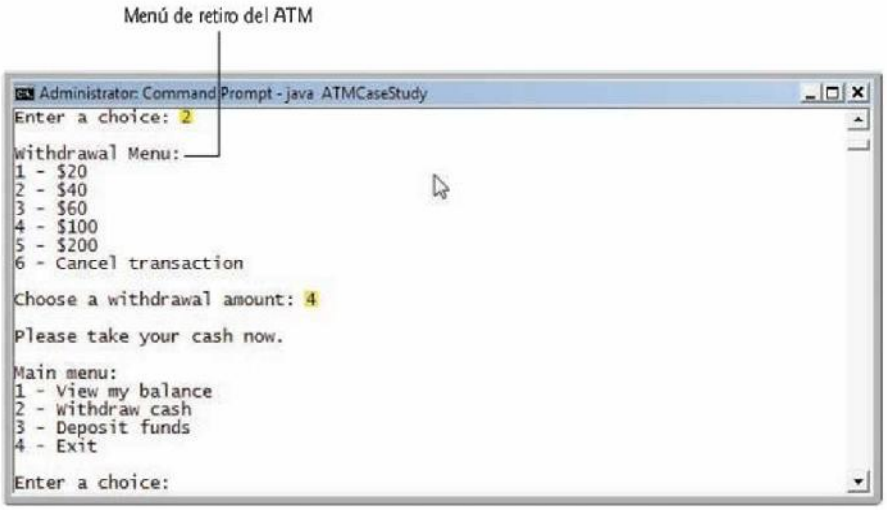


Fig. 1.16 | Se retira el dinero de la cuenta y la aplicación regresa al menú principal.

8. ***Confirme que la información de la cuenta se haya actualizado.*** En el menú principal, seleccione la opción 1 de nuevo para ver el saldo actual de su cuenta (figura 1.17). Observe que tanto el saldo disponible como el saldo total se han actualizado para reflejar su transacción de retiro.

9. ***Finalice la transacción.*** Para finalizar su sesión actual en el ATM, seleccione la opción 4, "Salir" del menú principal (figura 1.18.) El ATM saldrá del sistema y mostrará un mensaje de despedida al usuario. A continuación, la aplicación regresará a su indicador original, pidiendo el número de cuenta del siguiente usuario.

10. ***Salga de la aplicación del ATM y cierre la ventana*** **Símbolo del sistema.** La mayoría de las aplicaciones cuentan con una opción para salir y regresar al directorio del **Símbolo del sistema** desde el cual se ejecutó la aplicación. Un ATM real no proporciona al usuario la opción de apagar la máquina ATM. En vez de ello, cuando el usuario ha completado todas las transacciones deseadas y elige la opción del menú para salir, el ATM se reinicia y muestra un indicador para el número de cuenta del siguiente usuario. Como se muestra en la figura 1.18, la

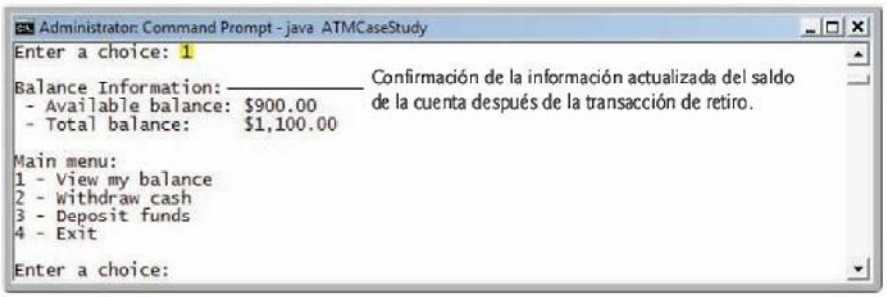


Fig. 1.17 | Verificación del nuevo saldo.

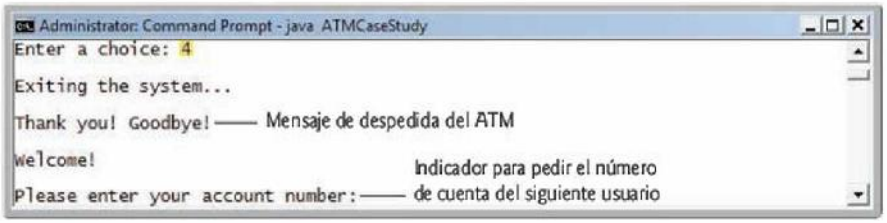


Fig. 1.18 | Finalización de una sesión de transacciones con el ATM.

aplicación del ATM se comporta de manera similar. Al elegir la opción del menú para salir sólo se termina la sesión del usuario actual con el ATM, no toda la aplicación completa. Para salir realmente de la aplicación del ATM, haga clic en el botón de cerrar (x) en la esquina superior derecha de la ventana **Símbolo del sistema**. Al cerrar la ventana, la aplicación termina su ejecución.

1.11 Web 2.0: Las redes sociales

Literalmente, la Web cobró fuerza a mediados de la década de 1990, pero surgieron tiempos difíciles a principios del año 2000, debido al desplome económico de "punto com". Al resurgimiento que empezó alrededor de 2004, se le conoce como **Web 2.0**. A Google se le considera en muchas partes como la compañía característica de Web 2.0. Algunas otras compañías con "características de Web 2.0" son: YouTube (sitio para compartir videos), FaceBook (red social), Twitter (microblogs), Groupon (comercio social), Foursquare (reportes o "check-ins" móviles), Salesforce (software de negocios que se ofrece como servicios en línea), Craigslist (listados gratuitos de anuncios clasificados), Flickr (sitio para compartir fotos), Second Life (un mundo virtual), Skype (telefonía por Internet) y Wikipedia (una enciclopedia en línea gratuita).

Google

En 1996, los candidatos a un doctorado en ciencias computacionales de Stanford, Larry Page y Sergey Brin, empezaron a colaborar en un nuevo motor de búsqueda. En 1997 le cambiaron el nombre a Google con base en el término matemático *gúgol* (en inglés, googol), una cantidad representada por el número "uno" seguido de 100 "ceros" (o 10^{100}): un número de un tamaño asombroso. La habilidad de Google para devolver resultados de búsquedas con extrema precisión le ayudó a convertirse con rapidez en el motor de búsqueda más utilizado, además de ser uno de los sitios Web más populares en el mundo.

Google continúa siendo un innovador en las tecnologías de búsqueda. Por ejemplo, Google Goggles es una fascinante aplicación móvil (disponible en Android e iPhone) que permite al usuario realizar una búsqueda, con la novedad de que utiliza una fotografía en vez de texto. Usted sólo tiene que tomar fotografías de puntos de referencia, libros (cubiertas o códigos de barras), logotipos, arte o etiquetas de botellas de vino, y Google Goggles escanea la fotografía para devolver los resultados de la búsqueda. También puede tomar una fotografía de texto (por ejemplo, el menú de un restaurante o un anuncio) y Google Goggles lo traducirá por usted.

Servicios Web y mashups

En este libro incluimos un tratamiento detallado sobre los servicios Web (capítulo 31) y presentamos la nueva metodología de desarrollo de aplicaciones conocida como *mashups*, en la que puede desarrollar con rapidez aplicaciones poderosas e intrigantes, al combinar servicios Web complementarios (a menudo gratuitos) y otras formas de fuentes de información (figura 1.19). Uno de los primeros mashups fue `www.housingmaps.com`, que combina al instante los listados de bienes raíces proporcionados por `www.craigslist.org` con las capacidades de generación de mapas de *Google Maps* para ofrecer mapas que muestren las ubicaciones de los apartamentos en renta dentro de cierta área.

Fuente de servicios Web	Cómo se utiizan
Google Maps	Servicios de mapas
Facebook	Redes sociales
Foursquare	Reportes (check-ins) móviles
LinkedIn	Redes sociales para negocios
YouTube	Búsquedas de video
Twitter	Microblogs
Groupon	Comercio social
Netflix	Renta de películas
eBay	Subastas en Internet
Wikipedia	Enciclopedia colaborativa
PayPal	Pagos
Last.fm	Radio por Internet
Amazon eCommerce	Compra de libros y otros artículos
Salesforce.com	Administración de las relaciones con el cliente (CRM)
Skype	Telefonía por Internet
Microsoft Bing	Búsqueda
Flickr	Compartir fotografías
Zillow	Precios de bienes raíces
Yahoo Search	Búsqueda
WeatherBug	Clima

Fig. 1.19 | Algunos servicios Web populares (`www.programmableweb.com/apis/directory/1?sort=mashups`).

Ajax

Ajax es una de las tecnologías de software más importantes de Web 2.0, ya que ayuda a las aplicaciones basadas en Internet a funcionar como las aplicaciones de escritorio; una tarea difícil, dado que dichas

aplicaciones sufren de retrasos en la transmisión, a medida que los datos se intercambian entre su computadora y las computadoras servidores en Internet. Mediante el uso de Ajax, las aplicaciones como Google Maps han logrado un desempeño excelente, además de que su apariencia visual se asemeja a las aplicaciones de escritorio. Aunque no hablaremos sobre la programación "pura" con Ajax en este libro (que es bastante compleja), en el capítulo 30 le mostraremos cómo crear aplicaciones habilitadas para Ajax mediante el uso de los componentes de JavaServer Faces (JSF) habilitados para Ajax.

Aplicaciones sociales

Durante los últimos años se ha producido un aumento considerable en el número de aplicaciones sociales en Web. Aún y cuando la industria de la computación ya alcanzó la madurez, estos sitios fueron capaces de tener un éxito fenomenal en un periodo de tiempo relativamente corto. La figura 1.20 analiza unas cuantas de las aplicaciones sociales que están generando un impacto.

Compañía	Descripción
Facebook	Facebook inició desde un dormitorio en Harvard en el año 2004, gracias a los alumnos Mark Zuckerberg, Chris Hughes, Dustin Moskovitz y Eduardo Saverin, y ahora tiene un valor estimado de 70 mil millones de dólares. Para enero de 2011, Facebook era el sitio más activo en Internet con más de 600 millones de usuarios —casi 9% de la población mundial—, quienes invierten 700 mil millones de minutos en Facebook al mes. Según su tasa de crecimiento actual (cerca del 5% mensual), en 2012 Facebook llegará a mil millones de usuarios ¡de los dos mil millones de Internet! La actividad en este sitio lo hace muy atractivo para los desarrolladores de aplicaciones. Cada día, los usuarios de Facebook instalan más de 20 millones de aplicaciones (`http://www.facebook.com/press/info.php?statistics`).
Twitter	Jack Dorsey, Evan Williams e Isaac "Biz" Stone fundaron Twitter en 2006: todo desde la compañía de podcasts, Odeo. Twitter revolucionó los *microblogs*. Los usuarios publican "tweets": mensajes de hasta 140 caracteres de longitud. Se publican cerca de 95 millones de tweets a diario (`twitter.com/about`). Usted puede seguir los tweets de amigos, artistas, negocios, representantes del gobierno (incluso el presidente de Estados Unidos, quien tiene 6.3 millones de seguidores), etcétera, o seguir tweets del tema para dar seguimiento a noticias, tendencias y mucho más. Al momento de escribir este libro, Lady Gaga tenía el mayor número de seguidores (más de 7.7 millones). Twitter se convirtió en el punto de origen para muchas noticias de última hora en todo el mundo.
Groupon	Groupon, un *sitio de comercio social*, fue lanzado por Andrew Mason in 2008. Para enero de 2011 la compañía estaba valuada alrededor de los $15 mil millones ¡con lo cual se convirtió en la compañía con más rápido crecimiento hasta esa fecha! Ahora está disponible en cientos de mercados en todo el mundo. Groupon muestra una oferta diaria en cada mercado para restaurantes, vendedores al detalle, servicios, atracciones y demás. Las ofertas se activan sólo hasta que se inscribe el mínimo número de personas requeridas para comprar el producto o servicio. Si usted se inscribe en una oferta y todavía no cumple con el mínimo, tal vez se vea tentado a dar aviso a otras personas sobre esa oferta por correo electrónico, Facebook, Twitter, etcétera. Si la oferta no cumple con el mínimo de ventas, se cancela. Una de las ofertas de Groupon más exitosas a nivel nacional a la fecha fue un certificado de $50 dólares en mercancía de una importante compañía de ropa a sólo $25. Se vendieron más de 440,000 cupones en un solo día.

Fig. 1.20 | Aplicaciones sociales (parte 1 de 2).

Compañía	Descripción
Foursquare	Foursquare —creada en 2009 por Dennis Crowley y Naveen Selvadurai— es una aplicación para realizar reportes (*check-ins*) móviles, la cual le permite notificar a sus amigos los lugares que visita. Puede descargar la aplicación en su teléfono inteligente y vincularla con sus cuentas de Facebook y Twitter, de modo que sus amigos puedan seguirlo desde varias plataformas. Si no tiene un teléfono inteligente, puede reportarse mediante un mensaje de texto. Foursquare utiliza el servicio GPS para determinar su ubicación exacta. Las empresas usan Foursquare para enviar ofertas a los usuarios que se encuentren cerca. Foursquare inició sus operaciones en marzo de 2009 y ya cuenta con más de 5 millones de usuarios en todo el mundo.
Skype	Skype es un producto de software que le permite realizar llamadas de voz y de video (la mayoría son gratuitas) a través de Internet, mediante el uso de una tecnología llamada *VoIP* (*Voz sobre IP*; IP se refiere a "Protocolo de Internet"). Niklas Zennström y Dane Janus Friiis fundaron Skype en 2003. Dos años después, vendieron la compañía a eBay por $2.6 mil millones.
YouTube	YouTube es un sitio para compartir videos que se fundó en 2005. Antes de que transcurriera un año, Google compró la compañía por $1.65 mil millones. En la actualidad, YouTube es responsable del 10% del tráfico total en Internet (`www.webpronews.com/topnews/2010/04/16/facebook-and-youtube-get-the-most-business-internet-traffic`). Menos de un año después de la liberación del iPhone 3GS de Apple —el primer modelo del iPhone en ofrecer video— las transferencias desde dispositivos móviles a YouTube aumentaron un 400% (`www.hypebot.com/hypebot/2009/06/youtube-reports-1700-jump-in-mobile-video.html`).

Fig. 1.20 | Aplicaciones sociales (parte 2 de 2).

1.12 Tecnologías de software

La figura 1.21 muestra una lista de palabras de moda que escuchará en la comunidad de desarrollo de software. Creamos Centros de Recursos sobre la mayoría de estos temas, y hay muchos por venir.

Tecnología	Descripción
Software ágil	El **desarrollo ágil de software** es un conjunto de metodologías que tratan de implementar software con más rapidez y menos recursos que las metodologías anteriores. Visite los sitios de Agile Alliance (`www.agilealliance.org`) y Agile Manifesto (`www.agilemanifesto.org`). También puede visitar el sitio en español `www.agile-spain.com`.
Refactorización	La **refactorización** implica reformular el código para hacerlo más claro y fácil de mantener, al tiempo que se preserva su funcionalidad. Es muy utilizado en las metodologías de desarrollo ágil. Muchos IDE contienen *herramientas de refactorización* integradas para realizar la mayor parte del proceso de refactorización de manera automática.
Patrones de diseño	Los **patrones de diseño** son arquitecturas probadas para construir software orientado a objetos flexible y que pueda mantenerse. El campo de los patrones de diseño trata de enumerar a los patrones recurrentes, y de alentar a los diseñadores de software para que los reutilicen y puedan desarrollar un software de mejor calidad con menos tiempo, dinero y esfuerzo. En el apéndice Q analizaremos los patrones de diseño de Java.

Fig. 1.21 | Tecnologías de software (parte 1 de 2).

Tecnología	Descripción
LAMP	MySQL es un sistema de administración de bases de datos de código fuente abierto. PHP es el lenguaje de "secuencias de comandos" del lado servidor de código fuente abierto más popular para el desarrollo de aplicaciones Web. **LAMP** es un acrónimo para el conjunto de tecnologías de código fuente abierto que usan muchos desarrolladores en la creación de aplicaciones Web: se refiere a Linux, Apache, MySQL y PHP (o Perl, o Python; otros dos lenguajes de secuencias de comandos).
Software como un servicio (SaaS)	Por lo general, el software siempre se ha visto como un producto; la mayoría aún se ofrece de esta forma. Para ejecutar una aplicación, hay que comprarla a un distribuidor de software. Después la instalamos en la computadora y la ejecutamos cuando sea necesario. A medida que aparecen nuevas versiones, actualizamos el software, lo cual genera con frecuencia un gasto considerable. Este proceso puede ser incómodo para las organizaciones con decenas de miles de sistemas, a los que se debe dar mantenimiento en una diversa selección de equipo de cómputo. En el **Software como un servicio (SaaS)**, éste ejecuta en servidores ubicados en cualquier parte de Internet. Que al ser actualizados, los clientes en todo el mundo ven las nuevas capacidades sin necesidad de una instalación local. Podemos acceder al servicio a través de un navegador. Los navegadores son bastante portables, por lo que podemos ver las mismas aplicaciones en una amplia variedad de computadoras desde cualquier parte del mundo. Salesforce.com, Google, Microsoft Office Live y Windows Live ofrecen SaaS.
Plataforma como un servicio (SaaS)	La **Plataforma como un servicio (PaaS)** provee una plataforma de cómputo para desarrollar y ejecutar aplicaciones como un servicio a través de Web, en vez de instalar las herramientas en su computadora. Los proveedores de PaaS más importantes son: Google App Engine, Amazon EC2, Bungee Labs, entre otros.
Computación en la nube	SaaS y PaaS son ejemplos de **computación en la nube** en donde el software, las plataformas y la infraestructura (por ejemplo, el poder de procesamiento y el almacenamiento) se alojan según la demanda a través de Internet. Esto ofrece a los usuarios flexibilidad, escalabilidad y un ahorro en los costos. Por ejemplo, considere las necesidades de almacenamiento de datos de una compañía, que pueden fluctuar de manera considerable en el transcurso de un año. En vez de invertir en hardware de almacenamiento de gran escala —cuyo costo de compra, mantenimiento y aseguramiento puede ser considerable, además de que no siempre es posible aprovechar su capacidad total—, la compañía podría comprar servicios basados en la nube (como Amazon S3, Google Storage, Microsoft Windows Azure™, Nirvanix™ y otros) según los fuera requiriendo.
Kit de desarrollo de software (SDK)	Los **Kits de desarrollo de software (SDK)** incluyen tanto las herramientas como la documentación que utilizan los desarrolladores para programar aplicaciones. Por ejemplo, usted usará el Kit de desarrollo de Java (JDK) para crear y ejecutar aplicaciones de Java.

Fig. 1.21 | Tecnologías de software (parte 2 de 2).

La figura 1.22 describe las categorías de liberación de versiones de los productos de software.

Versión	Descripción
Alfa	El software *alfa* es la primera versión de un producto de software cuyo desarrollo aún se encuentra activo. Por lo general las versiones alfa tienen muchos errores, son incompletas y estables; además se liberan a un pequeño número de desarrolladores para que evalúen las nuevas características, para obtener retroalimentación lo más pronto posible, etcétera.

Fig. 1.22 | Terminología de liberación de versiones de productos de software (parte 1 de 2).

Versión	Descripción
Beta	Las versiones *beta* se liberan a un número mayor de desarrolladores en una etapa posterior del proceso de desarrollo, una vez que se ha corregido la mayoría de los errores importantes y las nuevas características están casi completas. El software beta es más estable, pero todavía puede sufrir muchos cambios.
Candidatos para liberación (Release Candidates)	En general, los *candidatos para liberación* tienen todas sus *características completas*, están (supuestamente) libres de errores y listos para que la comunidad los utilice, con lo cual se logra un entorno de prueba diverso: el software se utiliza en distintos sistemas, con restricciones variables y para muchos fines diferentes. Cualquier error que aparezca se corrige y, en un momento dado, el producto final se libera al público en general. A menudo, las compañías de software distribuyen actualizaciones incrementales a través de Internet.
Beta permanente	El software que se desarrolla mediante este método por lo general no tiene números de versión (por ejemplo, la búsqueda de Google o Gmail). Este software, que se aloja en la nube (no se instala en su computadora), evoluciona de manera constante de modo que los usuarios siempre dispongan de la versión más reciente.

Fig. 1.22 | Terminología de liberación de versiones de productos de software (parte 2 de 2).

1.13 Cómo estar al día con las tecnologías de información

La figura 1.23 muestra una lista de las publicaciones técnicas y comerciales que le ayudarán a permanecer actualizado con la tecnología, las noticias y las tendencias más recientes. También encontrará una lista cada vez más grande de Centros de recursos relacionados con Internet y Web en `http://www.deitel.com/ResourceCenters.html`.

Publicación	URL
Bloomberg BusinessWeek	`www.businessweek.com`
CNET	`news.cnet.com`
Computer World	`www.computerworld.com`
Engadget	`www.engadget.com`
eWeek	`www.eweek.com`
Fast Company	`www.fastcompany.com/`
Fortune	`money.cnn.com/magazines/fortune/`
InfoWorld	`www.infoworld.com`
Mashable	`mashable.com`
PCWorld	`www.pcworld.com`
SD Times	`www.sdtimes.com`
Slashdot	`slashdot.org/`
Smarter Technology	`www.smartertechnology.com`
Technology Review	`technologyreview.com`
Techcrunch	`techcrunch.com`
Wired	`www.wired.com`

Fig. 1.23 | Publicaciones técnicas y comerciales.

1.14 Conclusión

En este capítulo analizamos el hardware y software de computadora, los lenguajes de programación y los sistemas operativos. Vimos las generalidades de un entorno típico de desarrollo de programas de Java y probamos una aplicación de Java. Introdujimos los fundamentos de la tecnología de objetos. Aprendió acerca de algunos de los emocionantes y nuevos acontecimientos en el campo de las computadoras. También analizamos cierta terminología clave del desarrollo de software.

En el capítulo 2 creará sus primeras aplicaciones de Java. Podrá ver cómo es que los programas muestran mensajes en la pantalla y obtienen información del usuario mediante el teclado para procesarla. Utilizará los tipos de datos primitivos y los operadores aritméticos de Java en cálculos que emplean los operadores de igualdad y relacionales de Java para escribir instrucciones simples de toma de decisiones.

Ejercicios de autoevaluación

1.1 Complete las siguientes oraciones:

a) La compañía que popularizó la computación personal fue _____ .
b) La computadora que legitimó la computación personal en los negocios y la industria fue _____ .
c) Las computadoras procesan datos bajo el control de conjuntos de instrucciones conocidas como _____ .
d) Las unidades lógicas clave de la computadora son _____, _____, _____, _____, _____ y _____ .
e) Los tres tipos de lenguajes descritos en este capítulo son _____, _____ y _____ .
f) Los programas que traducen programas en lenguaje de alto nivel a lenguaje máquina se denominan _____ .
g) _____ es un sistema operativo de teléfonos inteligentes, basado en el kernel de Linux y en Java.
h) En general, el software _____ tiene todas sus *características completas*, está (supuestamente) libre de errores y listo para que la comunidad lo utilice.
i) Al igual que muchos teléfonos inteligentes, el control remoto del Wii utiliza un _____ que permite al dispositivo responder al movimiento.

1.2 Complete las siguientes oraciones sobre el entorno de Java:

a) El comando _____ del JDK ejecuta una aplicación de Java.
b) El comando _____ del JDK compila un programa de Java.
c) Un archivo de programa de Java debe terminar con la extensión de archivo _____ .
d) Cuando se compila un programa en Java, el archivo producido por el compilador termina con la extensión _____ .

1.3 Complete las siguientes oraciones (con base en la sección 1.6):

a) Los objetos tienen una propiedad que se conoce como _____ ; aunque éstos pueden saber cómo comunicarse con los demás objetos a través de interfaces bien definidas, por lo general no se les permite saber cómo están implementados los otros objetos.
b) Los programadores de Java se concentran en crear _____ , que contienen campos y el conjunto de métodos que manipulan a esos campos y proporcionan servicios a los clientes.
c) El proceso de analizar y diseñar un sistema desde un punto de vista orientado a objetos se denomina _____ .
d) Mediante la _____ , se derivan nuevas clases de objetos al absorber las características de las clases existentes y luego agregar características únicas propias.

e) _____ es un lenguaje gráfico que permite a las personas que diseñan sistemas de software utilizar una notación estándar en la industria para representarlos.
f) El tamaño, forma, color y peso de un objeto se consideran _____ de su clase.

Respuestas a los ejercicios de autoevaluación

1.1 a) Apple. b) Computadora personal (PC) de IBM. c) programas. d) unidad de entrada, unidad de salida, unidad de memoria, unidad central de procesamiento, unidad aritmética y lógica, unidad de almacenamiento secundario. e) lenguajes máquina, lenguajes ensambladores, lenguajes de alto nivel. f) compiladores. g) Android. h) Candidato de liberación. i) acelerómetro.

1.2 a) `java`. b) `javac`. c) `.java`. d) `.class`. e) códigos de bytes.

1.3 a) ocultamiento de información. b) clases. c) análisis y diseño orientados a objetos (A/DOO). d) herencia. e) El Lenguaje unificado de modelado (UML). f) atributos.

Ejercicios

1.4 Complete las siguientes oraciones:
a) La unidad lógica de la computadora que recibe información desde el exterior de la computadora para que ésta la utilice se llama _____ .
b) El proceso de indicar a la computadora cómo resolver un problema se llama _____ .
c) _____ es un tipo de lenguaje computacional que utiliza abreviaturas del inglés para las instrucciones de lenguaje máquina.
d) _____ es una unidad lógica de la computadora que envía información que ya ha sido procesada por la computadora a varios dispositivos, de manera que pueda utilizarse fuera de la computadora.
e) _____ y _____ son unidades lógicas de la computadora que retienen información.
f) _____ es una unidad lógica de la computadora que realiza cálculos.
g) _____ es una unidad lógica de la computadora que toma decisiones lógicas.
h) Los lenguajes _____ son los más convenientes para que el programador pueda escribir programas con rapidez y facilidad.
i) Al único lenguaje que una computadora puede entender directamente se le conoce como el _____ de esa computadora.
j) _____ es una unidad lógica de la computadora que coordina las actividades de todas las demás unidades lógicas.

1.5 Complete las siguientes oraciones:
a) _____ se utiliza ahora para desarrollar aplicaciones empresariales de gran escala, para mejorar la funcionalidad de los servidores Web, para proporcionar aplicaciones para dispositivos domésticos y muchos otros fines más.
b) En un principio, _____ se hizo muy popular como lenguaje de desarrollo para el sistema operativo UNIX.
c) La compañía Web 2.0 _____ es la que tiene el crecimiento más rápido de la historia.
d) El lenguaje de programación _____ fue desarrollado por Bjarne Stroustrup a principios de la década de 1980 en los Laboratorios Bell.

1.6 Complete las siguientes oraciones:
a) Por lo general, los programas de Java pasan a través de cinco fases: _____, _____, _____, _____ y _____ .
b) Un _____ proporciona muchas herramientas que dan soporte al proceso de desarrollo de software, como los editores para escribir y editar programas, los depuradores para localizar los errores lógicos en los programas, y muchas otras características más.

c) El comando java invoca al _____, que ejecuta los programas de Java.
d) Una _____ es una aplicación de software que simula una computadora, pero oculta el sistema operativo y el hardware subyacentes de los programas que interactúan con la VM.
e) El _____ toma los archivos `.class` que contienen los códigos de bytes del programa y los transfiere a la memoria principal.
f) El _____ examina los códigos de bytes para asegurar que sean válidos.

1.7 Explique las dos fases de compilación de los programas de Java.

1.8 Es probable que usted lleve en su muñeca uno de los tipos de objetos más comunes en el mundo: un reloj. Analice cómo se aplica cada uno de los siguientes términos y conceptos a la noción de un reloj: objeto, atributos, comportamientos, clase, herencia (por ejemplo, considere un reloj despertador), abstracción, modelado, mensajes, encapsulamiento, interfaz y ocultamiento de información.

Marcar la diferencia

Hemos incluido en este libro ejercicios Marcar la diferencia, en los que le pediremos que trabaje con problemas que son de verdad importantes para los individuos, las comunidades, los países y el mundo. Para obtener más información sobre las organizaciones a nivel mundial que trabajan para marcar la diferencia, y para obtener ideas sobre proyectos de programación relacionados, visite nuestro Centro de recursos para marcar la diferencia en `www.deitel.com/makingadifference`.

1.9 ***(Prueba práctica: calculadora de impacto ambiental del carbono)*** Algunos científicos creen que las emisiones de carbono, sobre todo las que se producen al quemar combustibles fósiles, contribuyen de manera considerable al calentamiento global y que esto se puede combatir si las personas tomamos conciencia y limitamos el uso de los combustibles con base en el carbono. Las organizaciones y los individuos se preocupan cada vez más por el "impacto ambiental del carbono". Los sitios Web como Terra Pass

```
www.terrapass.com/carbon-footprint-calculator/
```

y Carbon Footprint

```
www.carbonfootprint.com/calculator.aspx
```

ofrecen calculadoras de impacto ambiental del carbono. Pruébelas para determinar el impacto que provoca usted en el ambiente debido al carbono. Los ejercicios en capítulos posteriores le pedirán que programe su propia calculadora de impacto ambiental del carbono. Como preparación, le sugerimos investigar las fórmulas para calcularlo.

1.10 ***(Prueba práctica: calculadora del índice de masa corporal)*** Según las estimaciones recientes, dos terceras partes de las personas que viven en Estados Unidos padecen de sobrepeso; la mitad de estas personas son obesas. Esto provoca aumentos considerables en el número de personas con enfermedades como la diabetes y las cardiopatías. Para determinar si una persona tiene sobrepeso o padece de obesidad, puede usar una medida conocida como índice de masa corporal (IMC). El Departamento de Salud y Servicios Humanos de Estados Unidos proporciona una calculadora del IMC en `www.nhlbisupport.com/bmi/`. Úsela para calcular su propio IMC. Un ejercicio del capítulo 2 le pedirá que programe su propia calculadora del IMC. Como preparación, le sugerimos investigar las fórmulas para calcular el IMC.

1.11 ***(Atributos de los vehículos híbridos)*** En este capítulo aprendió sobre los fundamentos de las clases. Ahora empezará a describir con detalle los aspectos de una clase conocida como "Vehículo híbrido". Los cuales se están volviendo cada vez más populares, puesto que por lo general pueden ofrecer mucho más kilometraje que los operados sólo por gasolina. Navegue en Web y estudie las características de cuatro o cinco de los autos híbridos populares en la actualidad; después haga una lista de todos los atributos relacionados con sus características de híbridos que pueda encontrar. Por ejemplo, algunos de los atributos comunes son los kilómetros por litro en ciudad y los kilómetros por litro en carretera. También puede hacer una lista de los atributos de las baterías (tipo, peso, etcétera).

1.12 ***(Neutralidad de género)*** Muchas personas desean eliminar el sexismo de todas las formas de comunicación. Usted ha recibido la tarea de crear un programa que pueda procesar un párrafo de texto y reemplazar palabras que tengan un género específico con palabras neutrales en cuanto al género. Suponiendo que recibió una lista de palabras con género específico y sus reemplazos con neutralidad de género (por ejemplo, reemplace "esposa" por "cónyuge", "hombre" por "persona", "hija" por "descendiente", y así en lo sucesivo), explique el procedimiento que utilizaría para leer un párrafo de texto y realizar estos reemplazos en forma manual. ¿Cómo podría su procedimiento generar un término extraño como