

# Fundamentos del lenguaje Java

El lenguaje Java se parece al lenguaje C++ de modo que un programador que conozca este lenguaje ha dado un gran paso adelante.

Sin embargo, existen también grandes diferencias entre ambos lenguajes. Un programador puede haber usado el lenguaje C++ como un lenguaje C mejorado sin haber usado para nada la Programación Orientada a Objetos. Sin embargo, Java es un lenguaje plenamente orientado a objetos y para escribir el programa más simple hemos de definir una clase. Los tipos básicos de datos son similares, pero los arrays son distintos y las cadenas de caracteres en Java son objetos de la clase *String*.

## Introducción

Muchos lectores que hayan programado en algún lenguaje, comprenderán sin dificultad este capítulo. Sin embargo, se recomienda leerlo con atención ya que incluso en los aspectos básicos hay diferencias entre unos lenguajes y otros. En esta introducción aprenderemos aspectos básicos del lenguaje Java: la primera aplicación, los comentarios, los tipos básicos de datos, los operadores, las sentencias condicionales e iterativas.

## Clases y objetos

Este capítulo es fundamental para entender la Programación Orientada a Objetos. Aprenderemos el concepto de clase, a crear objetos de una determinada clase, a acceder desde dichos objetos a los miembros dato y a los miembros función, a distinguir entre miembros estáticos y no estáticos y muchas otras cosas más. Estudiaremos dos entidades muy importantes en cualquier lenguaje de programación: los arrays y las cadenas de caracteres o strings, además de otras clases importantes en lenguaje Java.

## La herencia y el polimorfismo

Este es otro de los aspectos fundamentales de la Programación Orientada a Objetos. Trataremos de la herencia, de la reutilización del código, del concepto de clase abstracta y su diferencia con el concepto de *interface*. El significado de polimorfismo y de enlace dinámico. Aprenderemos que todas las clases en Java descienden de la clase base *Object*, que proporciona una funcionalidad mínima. La utilidad de las clases y de los métodos finales.

La parte más difícil de entender es el polimorfismo, es decir, la técnica que permite pasar un objeto de una clase derivada a funciones que conocen el objeto solamente por su clase base.