

METODOLOGÍAS DE DISEÑO Y PROGRAMACIÓN ORIENTADA A OBJETOS

Basado en las clases de Alexandre Bergel

Ignacio Slater Muñoz



Departamento de Ciencias de la Computación
Universidad de Chile

Para mis alumnxs, de hoy y de siempre.

La parte del libro que nadie lee

La idea de este «apunte» nació como una *wiki* de *Github* creada por Juan-Pablo Silva como apoyo para el curso de *Metodologías de Diseño y Programación* dictado por el profesor Alexandre Bergel del Departamento de Ciencias de la Computación, Facultad de Ciencias Físicas y Matemáticas de la Universidad de Chile.

Lo que comenzó como unas notas para complementar las clases del profesor lentamente fue creciendo, motivado por esxs alumnxs que buscaban dónde encontrar soluciones para esas pequeñas dudas que no les dejaban avanzar.

El objetivo principal del texto sigue siendo el mismo, plantear explicaciones más detalladas, ejemplos alternativos a los vistos en clases y para dejar un documento al que lxs alumnxs puedan recurrir en cualquier momento.

Este libro no busca ser un reemplazo para las clases del curso, es y será siempre un complemento.

Esta obra va dirigida a los estudiantes de la facultad así como para cualquier persona que esté dando sus primeros pasos en programación. El libro presenta una introducción al diseño de software, la programación orientada a objetos y lo básico del lenguaje de programación *Java*. Se asume que los lectores tienen nociones básicas de programación, conocimiento básico de *Python* y, en menor medida, de *C*.

Antes de comenzar, debo agradecer a las personas que hicieron posible y motivaron la escritura de esto: Beatriz Graboloza, Dimitri Svandich y, por supuesto, Alexandre Bergel y Juan-Pablo Silva.

5 de febrero de 2021, Santiago, Chile

Índice general

1. Por algo se empieza	1
1.1. ¿Qué es un Java?	1

Capítulo 1

Por algo se empieza

Este libro no partía así, y si alguien leyó una versión anterior se dará cuenta. Solía comenzar con una descripción e instrucciones para instalar las herramientas necesarias para seguir este libro y eso no estaba mal, pero no me agradaba comenzar así, simplemente presentando las herramientas sin ningún contexto de por qué ni para qué las íbamos a utilizar.

Puede parecer ridículo cambiar todo lo que ya había escrito solamente por eso, pero cuando nos enfrentamos a problemas del mundo real esto comienza a cobrar más sentido. Reescribo estos capítulos por una razón simple pero sumamente importante y que será una de las principales motivaciones para las decisiones de diseño que tomaremos a medida que avancemos, en el desarrollo de software **lo único constante es el cambio**.¹ Una aplicación que no puede adaptarse a los cambios, sin importar que tan bien funcione, está destinada a morir.

¿Qué sucede entonces con las herramientas que vamos a utilizar? Las vamos a introducir, no podemos sacar una parte tan importante, pero no las vamos a presentar todas a la vez, en su lugar las iremos explicando a medida que las vayamos necesitando.

1.1. ¿Qué es un Java?

Java es uno de los lenguajes de programación más utilizados en el mundo (de ahí la necesidad de enseñarles éste y no otro lenguaje), se caracteriza por ser un lenguaje basado en clases, orientado a objetos, fuertemente tipado y (casi totalmente) independiente del sistema operativo.

¿Qué?

Tranquilos, vamos a ir de a poco. Comencemos por uno de los puntos que hizo que *Java* fuera adoptado tan ampliamente en la industria, la independencia del sistema operativo. Cuando *Sun Microsystems*² publicó la primera versión pública de *Java* (en 1996), los lenguajes de programación

¹Eric Freeman y Elizabeth Freeman, *Head First Design Patterns*.

²Actualmente *Java* es propiedad de *Oracle Corporation*

predominantes eran *C* y *C++* (y en menor medida *Visual Basic* y *Perl*). Estos lenguajes tenían en común que interactuaban directamente con la API del sistema operativo, lo que implicaba que un programa escrito para un sistema *Windows* no funcionaría de la misma manera en un sistema *UNIX*. *Java* por su parte planteó una alternativa distinta, delegando la tarea de compilar y ejecutar los programas a una máquina virtual (más adelante veremos en más detalle parte del funcionamiento de la *JVM* para entender sus beneficios y desventajas). Esto último hizo que, en vez de cambiar el código del programa para crear una aplicación para uno u otro sistema operativo, lo que cambiaba era la versión de la *JVM* permitiendo así que un mismo código funcionara de la misma forma en cualquier plataforma capaz de correr la máquina virtual.³ Pasarían varios años antes de que surgieran otros lenguajes que compartieran esa característica (destacando entre ellos *Python 2*, publicado el año 2000).

³ Actualmente casi todos los sistemas operativos son capaces de usar la *JVM*, en particular el sistema *Android* está implementado casi en su totalidad para usar esta máquina virtual.

Bibliografía

Evans y col.: Java in a Nutshell

java-nutshell-intro

Benjamin J. Evans y David Flanagan. *Java in a Nutshell*. Ed. por Mike Loukides y Meghan Blanchette. 6th ed. O'Reilly Media, Inc, 2015. Cap. 1. Introduction to the Java Environment, págs. 3-15.

Eric Freeman y col.: Head First Design Patterns

head-first-intro

Eric Freeman y Elizabeth Freeman. *Head First Design Patterns*. 1st ed. O'Reilly Media, Inc, 2004. Cap. Chapter 1: Intro to Design Patterns, págs. 1-35.