Revised Draft Preface

1 Java memory management

Do your Java applets halt with **`OutOfMemoryError`** exceptions? Does running Java cause your browser or Java virtual machine to bloat, hogging your system and slowing it down? Do your classes work fine during development and testing, but receive **`OutOfMemoryError`** exceptions when integrated into larger or longer-running programs?

If you have seen memory problems like these before, perhaps when writing C++ applications, you probably tackled them with a variety of explicit techniques such as maintaining reference counts for data, and using operators like **`delete`** and **`free`** to deallocate data when you were finished with it. In C++, the business of ensuring that a program uses the memory available to it as efficiently as possible — *memory management* — is left in your hands by the language.

The Java language, on the other hand, makes memory management its own responsibility — it provides automatic memory management, or garbage collection. There is no **`free`** or **`delete`** operator. So if your program uses too much memory at run time, what can you do get it under control?

The solution is to understand how garbage collection works — both in theory, and as implemented in the popular Java virtual machines — and to know which programming techniques trigger unnecessary memory allocation or make it difficult for the garbage collector to do its job. With a better understanding of these issues, you can start to reap the benefits of programming in a garbage-collected language like Java.

2 About this book

This book is a practical guide for improving the performance of Java programs by optimizing their memory usage. It will help you adapt your programming style to get the best from Java, by understanding how garbage collection works and discovering better techniques for programming in a dynamic, object-oriented language.

This book gives a straightforward, high-level account of how garbage collectors work, and explains how programs and garbage collectors influence each other. It offers a wealth of programming advice that will help you identify and fix code that is causing unnecessary memory consumption or poor memory utilization in your programs.

This book offers answers to such questions as:

- Why doesn't Java have **`delete`** and **`free`** operators like C++? What do I do if I *need* those operators?
- Why can't the garbage collector find more memory when my program needs it?
- How can I "uninitialize" automatic variables? How do I manage resources?
- What is "finalization"? When and how should I use it?
- How big is a Java object? How can I make sure my program is using efficient data structures?
- Why does my class keep getting reloaded? How can I stop that happening?
- Is there a difference in how different Java Virtual Machines (JVMs) handle garbage collection? Should I care?

3 Audience

This book is relevant to both new and experienced Java programmers. Whether you are coming to Java as a first language, or you are moving to Java from other languages such as C++, you will find useful information here.

4 What you need to know to understand this book

To understand this book, you should be a programmer familiar with Java or other object-oriented languages, such as C++, Common Lisp, or Dylan. The program examples in this book are written in Java, but they illustrate general principles relevant to other languages.

You do not need to know anything about garbage collection or memory management in general to understand this book. This book explains them both, showing how garbage collection approaches the general problem of managing program memory, and how it compares with manual memory management techniques found in languages like C++ and C.

5 Conventions used in this book

Yet to be written.

6 Where to go for more information

This book does not focus on garbage collection theory or algorithms. Nor does it explain how to implement a garbage collector. If you want to know more about those topics, the following sources are a good introduction:

> *Garbage Collection: Algorithms for Dynamic Memory Management*; Richard Jones, Rafael Lins; 1996, Wiley. ISBN 0-471-94148-4.
>
> *Uniprocessor Garbage Collection Techniques*; Paul R. Wilson; 1992, University of Texas at Austin. [need to add tech report number]

Harlequin's Memory Management Reference on the World Wide Web has an FAQ, bibliography, glossary and some introductory essays on memory management:

> **`<URL:http://www.harlequin.com/mm/reference/>`**

There are many books and other resources devoted to programming in Java. We have found the following useful:

> *The Java Programming Language*; Ken Arnold and James Gosling; 1997, Addison-Wesley. ISBN 0-201-63451-1.
>
> [others…]