



.NET Development (General) Technical Articles

Garbage Collector Basics and Performance Hints

Rico Mariani
Microsoft Corporation

April 2003

Summary: The .NET garbage collector provides a high-speed allocation service with good use of memory and no long-term fragmentation problems. This article explains how garbage collectors work, then goes on to discuss some of the performance problems that might be encountered in a garbage-collected environment. (10 printed pages)

Applies to:
Microsoft® .NET Framework

Contents

[Introduction](#)
[Simplified Model](#)
[Collecting the Garbage](#)
[Performance](#)
[Finalization](#)
[Conclusion](#)

Introduction

In order to understand how to make good use of the garbage collector and what performance problems you might run into when running in a garbage-collected environment, it's important to understand the basics of how garbage collectors work and how those inner workings affect running programs.

This article is broken down into two parts: First I will discuss the nature of the common language runtime (CLR)'s garbage collector in general terms using a simplified model, and then I will discuss some performance implications of that structure.

Simplified Model

For explanatory purposes, consider the following simplified model of the managed heap. Note that this is *not* what is actually implemented.

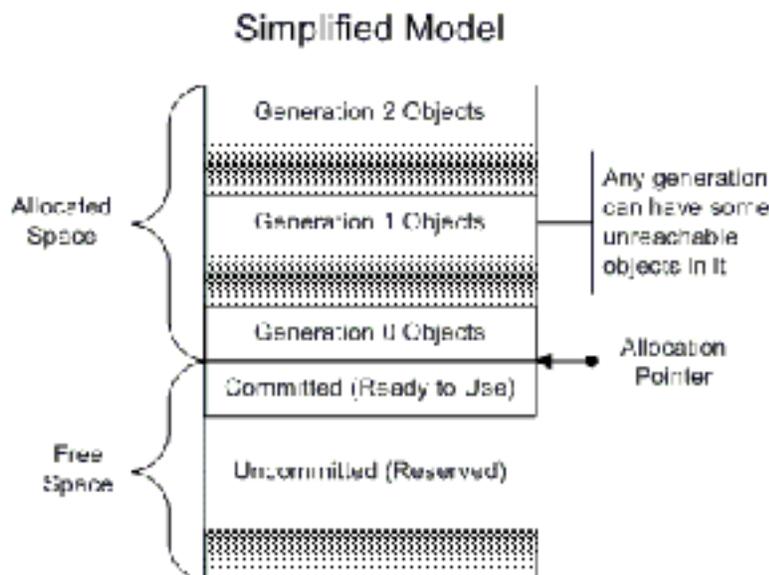


Figure 1. Simplified model of the managed heap