

**Username:** Pralay Patoria **Book:** Pro .NET Performance. No part of any chapter or book may be reproduced or transmitted in any form by any means without the prior written permission for reprints and excerpts from the publisher of the book or chapter. Redistribution or other use that violates the fair use privilege under U.S. copyright laws (see 17 USC107) or that otherwise violates these Terms of Service is strictly prohibited. Violators will be prosecuted to the full extent of U.S. Federal and Massachusetts laws.

---

## CHAPTER 4

# Garbage Collection

---

In this chapter, we will examine the .NET garbage collector (GC), one of the primary mechanisms affecting the performance of .NET applications. While freeing developers from worrying about memory deallocation, the GC introduces new challenges for constructing deterministically well-behaving programs in which performance is paramount. First, we will review the types of GC available in the CLR, and see how adapting an application to the GC can be very beneficial in terms of overall GC performance and pause times. Next, we'll see how generations affect GC performance and how to tune applications accordingly. Toward the end of the chapter we will examine the APIs available for controlling the GC directly, as well as the subtleties involved in correctly using non-deterministic finalization.

Many of the examples in this chapter are based on the authors' personal experience with real-world systems. Where possible, we tried to point you to case studies and even sample applications that you can work on while reading this chapter, illustrating the primary performance pain points. The "Best Practices" section toward the end of the chapter is full of such case studies and examples. However, you should be aware that some of these points are difficult to demonstrate with short code snippets or even a sample program because the situations where performance differences arise are typically within large projects that have thousands of types and millions of objects in memory.