开发人员专业技术丛书

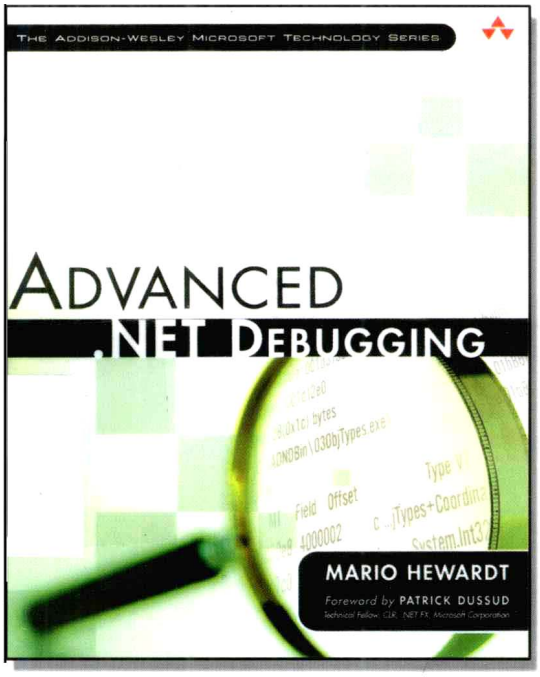
HZ BOOKS 华章科技

PEARSON

.NET高级调试

Advanced .NET Debugging

- 畅销书《Windows高级调试》姊妹篇
- .NET调试权威参考书
- 涉及.NET CLR 4.0最新调试功能



（美） **Mario Hewardt** 著

聂雪军 等译

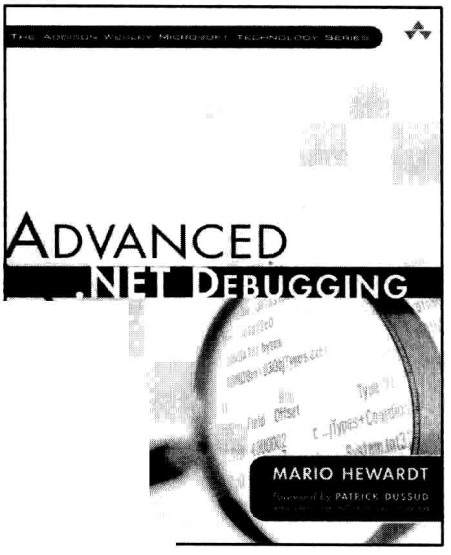
机械工业出版社
China Machine Press

PEARSON Education

开发人员专业技术丛书

.NET高级调试

Advanced .NET Debugging



Mario Hewardt 著

聂雪军 等译

机械工业出版社
China Machine Press

这是一本介绍如何通过非托管调试器（包括 WinDBG、NTSD 和 CDB 等）来调试 .NET 应用程序的书籍。本书内容主要包括：调试工具简介、CLR 基础、基本调试任务、程序集加载器、托管堆与垃圾收集、同步、互用性以及一些高级主题，如事后调试、一些功能强大的调试工具和 .NET 4.0 中的新功能等。

本书内容翔实、条理清晰，适合软件开发人员、软件测试人员、质量保证人员和产品技术支持人员等参考。

本书版权登记号：图字：01-2009-7722

图书在版编目（CIP）数据

.NET 高级调试/（美）赫瓦特（Hewardt,M.）著；聂雪军等译. —北京：机械工业出版社，2010.11

（开发人员专业技术丛书）

书名原文：Advanced .NET Debugging

ISBN 978-7-111-32085-2

Ⅰ.N…　Ⅱ.①赫…　②聂…　Ⅲ.计算机网络－程序设计　Ⅳ.TP393

中国版本图书馆 CIP 数据核字（2010）第 192357 号

机械工业出版社（北京市西城区百万庄大街 22 号　邮政编码　100037）

责任编辑：秦　健

北京市荣盛彩色印刷有限公司印刷

2011 年 1 月第 1 版第 1 次印刷

186mm × 240mm · 25 印张

标准书号：ISBN 978-7-111-32085-2

定价：59.00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换

客服热线：（010）88378991；88361066

购书热线：（010）68326294；88379649；68995259

投稿热线：（010）88379604

读者信箱：hzjsj@ hzbook.com

对本书的赞誉

“虽然. NET 环境为开发人员提供了一组强大的工具来编写软件，但是对开发过程中出现的各种问题进行调试仍然是一项困难的任务。本书介绍了 CLR 的内部工作原理，这对于分析. NET 程序中出现的各种类型的错误是非常有帮助的。此外，本书还详细介绍了如何解决一些常见的复杂问题。对于所有. NET 开发人员来说，本书都值得一读。”

——Lee Culver，微软公司 CLR 快速响应团队

“你是否碰到过. NET 程序失去响应？或者周期性地出现极高的 CPU 使用率？或者程序崩溃？当程序出现问题时，你需要使用正确的知识和工具，并深入程序内部进行分析。本书包含了丰富的知识，可以帮助开发人员迅速找出各种软件问题。欢迎进入调试领域！”

——Roberto A Farah，微软公司高级专案领域工程师

“对于. NET 开发人员来说，无论是新手还是高手，本书都具有极高的参考价值。本书包含了许多调试技巧以及 CLR 内部工作机制的细节，这对于设计软件架构的开发人员来说非常有用。我个人强烈推荐 Mario 的这本书。”

——Jeffrey Richter，Wintellect 公司顾问，培训师以及作者

“这是 Mario 推出的又一本好书。他之前编写的《Windows 高级调试》（与 Daniel Pravat 合著）对于非托管代码的调试来说是一本不可多得的参考书，而本书同样有着极高的质量，清晰的阐述以及深入的探讨，因此对于. NET 调试来说同样很有帮助。本书详细介绍了如何观察托管堆的使用情况、理解垃圾收集器的行为以及如何跟踪同步错误等，这些内容将使你在查找和修复托管代码中的错误时更加高效。”

——Mark Russinovich，微软公司技术顾问

“本书深入介绍了一些调试工具，例如 SOS，这是在其他的书中还不曾见过的。这些内容对于理解和调试托管程序来说无疑是非常有帮助的。”

——Maoni Stephens，微软公司 GC 开发工程师

“对于想深入了解. NET 通用语言运行时（. NET Common Language Runtime）内部工作原理的人来说，本书绝对值得一读。书中清晰地阐述了. NET 系统的层次结构以及程序集的加载和组织。对于需要调试同步和内存破坏等复杂问题的开发人员来说，我建议阅读这本书。此外，本书还详细介绍了事后调试技术。”

——Pat Styles，微软公司员工

译 者 序

本书是 Mario Hewardt 继《Windows 高级调试》之后推出的又一部力作。.NET 框架为开发人员隐藏了底层系统的复杂性，例如自动化内存管理机制使得开发人员无须关心内存的释放与回收，从而极大地提升软件开发效率。然而，这种抽象也使得一些问题调试起来更为困难，因此需要了解的底层技术细节也变得更多。本书以非托管调试器（包括 WinDBG、NTSD 和 CDB）为基础，详细介绍了.NET 框架中关键组件的运作原理及其与.NET 应用程序中一些常见问题的关联，并通过丰富的示例来阐述在调试不同问题时所应采取的策略、步骤以及工具。

作为《Windows 高级调试》的姊妹篇，本书的目的同样是为了将一些有价值的调试思路和调试工具推荐给软件开发人员，以提高开发人员的调试效率。本书的特点主要有：

- 深入介绍了.NET 中关键组件的内部运作原理，包括 CLR 垃圾收集器、程序集加载器、应用程序域以及类型元数据等。在调试.NET 应用程序时，对.NET 内部运行机制了解得越清楚，就越能控制程序的运行状态，调试效率也就越高，这与调试 Windows 系统应用程序或者其他任何类型程序的道理是相通的。
- 通过丰富的示例来讲解调试过程。本书分析了.NET 应用程序在程序集加载、托管堆、同步以及互用性等方面的一些常见问题。对于每类问题，作者首先给出了问题的外在表现，然后介绍如何做出合理假设以及采用相应的调试策略来进行验证，并最终找出问题的根源。对调试过程中的每个步骤，作者都给出了非常清晰的讲解和分析，使得读者更容易掌握书中内容。
- 重点介绍了两个调试器扩展 SOS 和 SOSEX。在本书各章的内容中穿插对 SOS 与 SOSEX 中各个命令的介绍，包括命令的语法格式、应用场合以及使用技巧等。这些内容在其他书籍中是很少出现的。

此外，本书还给出了一些高级调试主题，包括崩溃转储文件的生成、事后调试，并介绍了一些辅助调试工具，例如 PowerDbg 和 CLR 分析器等。这些内容都为开发人员调试复杂问题提供了必要的基础知识。在最后一章还介绍了在.NET 4.0 中引入的一些新功能。

本书保持了《Windows 高级调试》的行文风格与内容组织方式，围绕核心的调试思路，采用了由浅入深、由表及里的讲解方式，并辅以丰富的示例和详细的分析，使读者逐步掌握.NET应用程序的调试技巧。本书的技术性较强，需要读者对.NET 框架的底层架构和组件有一定的了解，并且具备一定的 C#编程基础。作者 Mario Hewardt 是微软公司的资深工程师，他在调试领域已经工作了十余年。本书汇聚了作者多年来的调试实践经验，对于.NET 开发

人员来说是一本不可多得的参考手册。

参与本书翻译工作的还有李杨、吴汉平、徐光景、童胜汉、陈军、胡凯、刘红、张玮、陈红、李斌、李勇涛、王海涛、周云波、彭敏才和张世锋等。由于译者的水平和时间有限，翻译中的疏漏和错误在所难免，还望读者和同行不吝指正。

聂雪军

2010 年 8 月于武汉

序

去年，我们在微软公司庆祝了 CLR 发布十周年。CLR 的目的是通过提供一种安全的和稳定的环境来提高开发人员的生产效率。目前，CLR 在各种环境中都得到了广泛应用，例如，在性能和可伸缩性上有着极高要求的大型服务器程序，以及日常使用的桌面程序等。随着 CLR 的日益普及，基于 CLR 来开发软件的人们同样面临着越来越多的挑战，因为他们的产品必须能够在不同的机器配置和网络环境中运行；此外，随着硬件的高速发展，人们正在构建的软件功能越来越强，同时复杂性也越来越高。所有这些情形都意味着，当程序没有按照预期方式运行时，你就需要负责分析和修复程序中的问题，因此了解一些调试知识和工具就显得尤为重要。

为了提高工作效率，CLR 为开发人员实现了许多基础的辅助机制，从而使开发人员能将主要精力放在关键逻辑上。事实上，人们无需花太多的时间来理解完整的 CLR 内部细节，而只需知道一些有助于分析问题的重要概念，这一点非常重要。然而，要想知道哪些概念是重要的却并不容易。许多人都是通过反复摸索之后才掌握这些知识，而这需要长时间的积累过程并且有时候可能得不到准确的答案。

本书对运行时的阐述恰到好处，它能帮助你理解在分析问题时遵循的思考过程以及在解决问题时采用的各种技术，此外书中还给出了从调试实际应用程序中提炼出的许多实用技术。因此，如果你希望提高调试 CLR 应用程序的速度，那么应该仔细阅读本书。本书涵盖了托管程序调试的许多方面——特别是对于一些难以诊断的领域，例如线程同步问题，本书给出了深入而细致的讲解。此外，本书在说明调试技术时使用了大量的示例，使得读者更容易掌握这些技术。

在本书中重点讲解的调试工具之一就是 SOS 调试器扩展，这个工具是由 CLR 小组开发和维护的。每当发布新版本的 CLR 时，都会对 SOS 进行升级，使 SOS 包含更多的新功能。对于分析托管进程中的问题来说，SOS 是一种功能强大的工具。它提供的大部分功能都是无法从其他调试工具中获得的。例如，SOS 可以找出引用托管堆中某个对象的根对象，这是托管程序开发人员经常遇到的问题之一。在熟悉了这个工具的使用后，你将可以进一步理解程序的工作流程。我还从未见过有其他的书比这本书更详细地介绍 SOS。

当掌握本书介绍的知识后，在分析问题时可以付出更少的时间和精力。我希望读者在阅读这本书时获得的乐趣与我在审阅本书手稿时获得的乐趣是一样的。

Patrick Dussud

微软公司技术顾问和.NET CLR 首席架构师

前　言

自从2007年底写完《Windows高级调试》（由机械工业出版社于2009年5月出版——译者注）一书后，许多读者都来信希望我再写一本关于.NET调试的书籍。最初在《Windows高级调试》中确实包含了一章内容专门介绍.NET调试，但最终还是去掉了这章，因为我认为，仅有一章的篇幅无法涵盖.NET调试的所有内容，这给读者带来的只能是更多的疑问而不是启迪。.NET已经成为众多开发人员的首选平台。根据统计数据表明，使用C#的开发人员与使用C++的开发人员在数量上基本相当。要想在.NET开发中取得成功，那么就需要知道如何正确地应对在开发中遇到的各种问题和挑战。

为什么在一本介绍.NET调试的书中使用了WinDBG、NTSD以及CDB等这些调试器？显然还有其他的调试器可以使用（有些调试器甚至有着更高的用户友好性）。初看上去，使用非托管调试器似乎有些棘手，但如果将它们的功能完全发挥出来，那么在分析一些复杂的问题时将节约大量的时间。部分原因是当使用非托管调试器时，更容易收集CLR本身（即.NET运行时）的关键内部信息，并可以根据这些信息来分析问题的原因。这些信息包括垃圾收集器、互用性层、同步原语等。这种信息不仅会在许多调试任务中起到关键作用，而且还是一种很好的学习资料，因为它给出了对运行时架构的详细描述。最后，在某些情况下（随着目前“互联”解决方案越来越多），需要使用一种零痕迹（ZEOR foot print）调试器。这种“更友好的”调试器通常需要在本地进行安装，在这个过程中将把所需的二进制文件复制到目标机器上，并在系统的不同位置保存配置信息等。如果在某台机器上不允许修改配置信息（例如在用户机器或者数据中心的机器上），那么唯一可行的方法就是使用非托管调试器，因为这种调试器不需要修改配置信息。

本书弥补了调试领域中的这种缺陷，并重点介绍了如何在CLR环境中发挥非托管调试器的功能。本书采用了与实践紧密相结合的讲解方式，使用真实的调试任务示例，以确保读者不仅可以学习书中介绍的内容，而且还能获得实际的体验。我希望读者在阅读本书时获得的乐趣与我在写这本书时获得的乐趣一样。

本书的目标读者

如果在分析问题时知道该采用哪些工具，并且知道如何对不同的问题进行分类，那么通常可以开发出高质量的软件产品，并降低在技术支持上的开销。然而，尽管我们已经竭尽全力消除产品中的错误，但在客户机器上仍然会冒出一些意料之外的问题，只有知道如何在这种情况下分析这些问题，才能有效地避免给客户造成麻烦或者停机。所有的软件工程师都有必要了解在各种不同的环境中应该使用哪些工具来分析问题。如果掌握了调试的技巧，那么

产品质量将得到极大提升，而且公司的形象也会由于软件的高质量和高可靠性而得到提升。

软件开发工程师

我曾看到许多开发人员努力解决各种复杂的错误，并且最终花费数天（有时候甚至是数星期）时间来缩小问题的范围以及找到问题的根源。在大多数时候，我都会询问开发人员使用了哪些工具。得到的答案通常大同小异：反复进行代码审查和跟踪，直至最终找出问题。代码审查和有针对性的跟踪对于分析错误来说固然重要，但它们通常也是非常耗时的。面对现实吧！假使我们能够获得在分析代码问题时需要的所有信息，那么将不需要使用调试器。然而，实际情况是，在某些情况中仅凭代码跟踪不足以解决问题，而是需要将调试器附载到出现问题的进程上。在许多时候，当我告诉开发人员某个工具可以极大地降低某种问题的分析时间后，开发人员会吃惊地发现居然有这种工具存在。

本书的目标读者之一就是那些在.NET 平台上编写代码以及分析复杂问题的开发人员。能够深入地理解哪些工具可以帮助开发人员找出一些复杂且耗时的问题，对于产品的成功来说非常重要。在开发过程中，知道使用哪些工具以及启动哪些配置，对于获得成功来说是非常关键的。

质量保证工程师

质量保证（Quality Assurance，QA）的目的是找出开发人员在代码中出现的问题。详细的测试计划以及完全自动化的测试流程都能使质量保证工程师们以高效的方式来测试各个组件。正如了解调试工具和配置信息对开发人员非常重要，这对于质量保证工程师也同样重要。在测试过程中，他们可能会遇到各种问题，如果在测试过程中能使用正确的工具，那么将帮助他们以及开发人员在解决问题上节约大量的时间。如果在进行测试时没有使用正确的工具，那么在重新启动测试流程（并且打开这个工具）时可能最终发现这不是一个可重现的问题。在本书中介绍的调试器和工具能够提高质量保证工程师的工作效率，并且帮助整个产品团队更快地获得结果。

产品支持工程师

产品支持工程师面临的情况与软件开发工程师和质量保证工程师遇到的情况非常相似。关键的区别在于他们所处的环境不同。他们不仅要解决客户的问题，而且经常必须面对来自多个地方的代码（即不仅仅是产品的代码）。此外，产品支持工程师通常只能使用进程的静态快照，而无法进行实时调试，这就使得找到问题的根源更加困难。在这些情况下，如果知道如何使用调试器以及相应的工具，那么就不需要多次往返于公司与客户之间（这不仅会带来很高的成本开销，而且很容易令人沮丧），并且能够立即解决问题。

运行工程师

随着越来越多的软件移动到云中（一种基于服务的提供方式），越来越多的代码是在专门的

数据中心运行，而不是在客户的机器上运行。负责这些服务正常运行的工程师们就被称为运行工程师。运行工程师面临的关键挑战之一是，要解决所有使服务无法最优化运行的问题。通常，这意味着要尽快地解决问题。如果运行团队无法解决某个问题，那么产品团队会参与进来，这将是一个耗时的过程，因为产品团队可能要反复地指导运行团队分析这个问题。通过使用正确的工具，运行团队可以解决遇到的大部分问题，而无需将这些问题提交给产品团队，因此节约了双方的时间；并且最重要的是，客户在使用服务时遇到的停机时间也将更短。

预备知识

虽然本书介绍的是如何使用非托管调试器，但重点在于介绍如何调试.NET 代码，而不是非托管调试的一些基本步骤。第 3 章将会简要介绍一些主题，例如如何将调试附载到目标进程，设置符号路径以及设置断点等，但并没有给出详细介绍。关于非托管调试器以及非托管代码调试的进一步介绍可以参考我之前写的一本书：Mario Hewardt 和 Daniel Pravat，《Advanced Windows Debugging》，Boston，MA：Addison-Wesley，2007。[⊖]

此外，还需要深入理解 C#，因为所有的示例代码都是采用 C#编写的。在学习 C#时可以使用下面这本书：Mark Michaelis，《Essential C# 2.0》，Boston，MA：Addison-Wesley，2006。

虽然读者需要预先具备 C#的知识，但并不需要深入了解 CLR。本书不仅介绍了如何调试.NET 程序，还对.NET 平台中大量的核心代码给出了详细解释，这些信息是获得调试成功的重要基础。

本书的组织结构

本书分为三个部分，下面介绍每一部分包含的内容。

第一部分　简介

这一部分包含的内容主要是介绍如何使用非托管调试器来调试.NET 的基础知识。介绍的主题包括：需要使用的工具、对 MSIL 的介绍、基本的调试任务等，我对这些主题都进行了全面的分析和阐述。如果你是第一次了解这些调试器，那么建议你按照先后顺序阅读这些内容。

第 1 章　调试工具简介

这一章对本书中使用的工具给出了简要介绍，包括应用场合、下载位置以及安装方法等。所介绍的工具包括：Windows 调试工具集、SOS、SOSEX、CLR 分析器等。

第 2 章　CLR 基础

这一章介绍了.NET 运行时的核心基础。首先介绍了主要的运行时组件，然后是一些相关主题，包括程序集的加载、运行时元数据等。其中使用非托管调试器和工具来说明运行时

⊖ 中文版为《Windows 高级调试》，由机械工业出版社于 2009 年 5 月出版。——译者注

的内部工作机制。

第 3 章　基本调试任务

这一章介绍了在使用非托管调试器调试. NET 程序时的一些最常见调试任务，此外还介绍了一些其他主题，例如查看线程数据、垃圾收集器堆、. NET 异常以及事后调试的基本概念等。

第二部分　调试实践

第二部分是本书的核心内容，介绍了主要的 CLR 组件以及如何分析与这些组件相关的常见问题。在这部分每一章的开始，首先给出组件的概述，并通过调试器来说明关键概念。之后给出使用这个组件时会出现的常见编程错误。接下来详细介绍了在解决这些错误时该采取的思路，并给出了直观的调试会话。第二部分中的各章可以按照任意顺序来阅读，因为每一章都是针对某个特定组件的问题进行讨论。

第 4 章　程序集加载器

. NET 程序既可以是简单的命令行程序，也可以是复杂的多进程/多机器服务器程序，其中包含了大量程序集的相互协作。要想高效地调试. NET 程序中的问题，你必须仔细地理解各个. NET 程序集之间的依赖性。这一章介绍了 CLR 程序集加载器的工作原理，以及在使用这个组件时最常见的问题。

第 5 章　托管堆与垃圾收集

虽然. NET 开发人员可以坐享自动内存管理机制带来的好处，但仍然需要小心地避免一些高开销的错误。CLR 垃圾收集器是一个自动的内存管理器，它使开发人员可以更少关注内存管理，而更多关注程序逻辑。尽管 CLR 负责为开发人员管理内存，但仍然要小心地避免一些可能对程序造成严重破坏的陷阱。在这一章中，我们将看到垃圾收集器的工作机制，学会观察垃圾收集器的内部信息以及与自动垃圾收集操作相关的一些常见编程错误。

第 6 章　同步

多线程的环境包含了大量的灵活性和高效因素。然而，伴随这种灵活性的是线程管理中的复杂性。为了避免在程序中出现高开销错误，我们必须小心地确保线程以一种相互协作的方式来执行工作。这一章介绍了. NET 中的同步原语，并介绍了如何使用调试器和工具来分析常见的线程同步问题。此外还介绍了死锁以及线程池等问题。

第 7 章　互用性

. NET 非常依赖底层的 Windows 组件。为了调用非托管的 Windows 组件，CLR 公开了两种主要的互用性方法：

- 平台调用
- COM 互用性

由于. NET 和 Win32 编程模型非常不同，因此一些编程习惯经常会导致难以跟踪的问题。在这一章中，我们将看到一些在使用互用性时常见的错误，以及如何使用调试器和工具来分

析这些错误。

第三部分　高级主题

这一部分介绍的主题包括：事后调试、一些功能强大的调试工具，以及.NET 4.0 中的新功能。

第 8 章　事后调试

通常，我们可能无法获得对故障机器的完全访问权限，因此无法对客户产品机器上出现的错误进行实时调试。本章介绍了如何在不需要访问物理机器的情况下对问题进行调试。讨论的主题包括：崩溃转储的基本知识，如何生成以及分析崩溃转储文件等。

第 9 章　一些功能强大的调试工具

在.NET 调试过程中，除了一些“标准的“工具外，还有其他一些功能强大的调试工具。这一章为读者介绍了这些强大的工具，例如 PowerDBG（通过 PowerShell 调试）以及其他工具。

第 10 章　CLR 4.0

随着 CLR 4.0 的即将发布，这一章将简要介绍 CLR 4.0 中的新增功能。此外，本章依次介绍了之前内容中每个主题在 CLR 4.0 中的不同之处。

排版约定

在本书的示例中给出了大量的调试输出。调试输出是指用户在执行某个动作后调试器显示的结果。通常，这种调试输出包含了以简洁形式给出的信息。为了引用这些数据并使读者更容易阅读，在调试输出中采用了粗斜体。此外，在调试输出中的粗体内容表示需要键入的命令。在接下来的示例中说明了这些字体。

```
0:000> ~*kb
.  0  Id: 924.a18 Suspend: 1 Teb: 7ffdf000 Unfrozen
ChildEBP RetAddr  Args to Child
0007fb1c 7c93edc0 7ffdf000 7ffd4000 00000000 ntdll!DbgBreakPoint
0007fc94 7c921639 0007fd30 7c900000 0007fce0 ntdll!LdrpInitializeProcess+0xffa
0007fd1c 7c90eac7 0007fd30 7c900000 00000000 ntdll!_LdrpInitialize+0x183
00000000 00000000 00000000 00000000 00000000 ntdll!KiUserApcDispatcher+0x7
0:000> dd 0007fd30
0007fd30  00010017 00000000 00000000 00000000
0007fd40  00000000 00000000 00000000 ffffffff
0007fd50  ffffffff f735533e f7368528 ffffffff
0007fd60  f73754c8 804eddf9 8674f020 85252550
0007fd70  86770f38 f73f4459 b2f3fad0 804eddf9
0007fd80  b30dccd1 852526bc b30e81c1 855be944
0007fd90  85252560 85668400 85116538 852526bc
0007fda0  852526bc 00000000 00000000 00000000
```

在这个示例中，需要在调试会话中键入 ~ * **kb**。在键入这个命令后，将输出几行信息，其中最关键的就是 **0007fd30**。接下来应该键入 **dd 0007fd30** 命令，这个命令将显示之前高亮

数值 **0007fd30** 的更多信息。在本书中使用的所有工具都是从安装文件夹中启动的。例如，如果 Windows 调试器安装到文件夹 C:\Program Files\Debugging Tools for Windows 中，那么启动 windbg.exe 的命令行如下所示：

```
C:\>windbg
```

所需工具

在本书中使用的所有工具都可以免费下载。在第 1 章中给出了在本书中使用的工具及其下载地址。

示例代码

要说明如何调试存在问题的代码，最有效的方式就是使用实际的示例。然而，在一本书中包含完整的示例代码是不现实的，因为这将使我们无法以简洁的方式说明问题。因此，对本书中的示例代码都进行了简化（但并不影响完整性）。所有的示例代码都是基于 C#和. NET 2.0 编写的。每个示例都可以从本书的 Web 站点下载，地址为 http://www.advanceddotnetdebugging.com。每个示例都包含了一个 MSBuild 项目文件。MSBuild 是一个功能完备的命令行构建环境，与. NET SDK 2.0 一起发行，并且与 Microsoft Visual Studio 是兼容的。所有的调试会话都是基于 32 位版本的. NET 框架。

技术支持

虽然我努力避免在本书中出现错误，但还是不可避免地存在一些疏漏。你可以将发现的错误提交到本书的网站 http://www.advanceddotnetdebugging.com 或者可以直接给我写信，电子邮件地址为 marioh@advanceddotnetdebugging.com。在这个网址上还包括一个勘误表以及相应的错误和修正。

小结

在当前复杂的软件解决方案中，无论是独立的命令行程序还是高度互连并且在全世界范围内通信的系统，都会存在代码问题。要确保在这些产品中不出现错误似乎是一项很艰难的任务，但只要有正确的工具以及知道如何使用这些工具，那么软件工程师的工作就会变得很轻松。这些强大的工具和正确的思路不仅有助于使分析问题的过程变得更高效，而且还为企业节省了大量的财力和减少了可能造成的客户流失。本书的目的是为了使软件工程师们获得. NET调试的知识和经验，从而避免一些严重的陷阱，并且使问题的分析过程变得更为高效和成功。

欢迎阅读。

致谢

基于之前编写《Windows 高级调试》一书的经验，我很清楚地认识到当一边在微软全职工作，一边在业余时间写书时需要付出多大的努力。在《Windows 高级调试》一书获得成功后，读者不断要求我再写一本关于. NET 和 CLR 调试的书，因此我决定再次迎接挑战。虽然与第一本书相比，这次的任务要更容易一些，但仍然付出了很大的努力——不仅仅是我自己，而且还包括本书中从构思到出版过程中参与的许多人。

首先，我要感谢我的家人。我的妻子 Pia 不仅毫无怨言地容忍我经常说“这个周末我要写作”，而且在 2008 年底为我带来了漂亮可爱的女儿 Gemma。如果没有 Pia 和 Gemma 的耐心、支持和鼓励，本书是无法完成的。

感谢 Addison-Wesley 团队，他们对一位在业余时间写作的作者再次给予了难以置信的宽容。我的团队成员经常不得不请求修改时间表，提供高质量的编辑过程并且加速出版计划，以便使读者能尽快看到这本书。特别要感谢 Joan Murray，感谢他使得整个过程极为顺畅，感谢 Chris Zahn 仔细审阅和纠正我的英语表述，感谢 Olivia Basegio 在 Joan 度假时负责接管编辑工作。此外，特别感谢 Curt Johnson 为本书所做的市场推广工作。

当然，无论作者如何小心，总会有些技术细节不能准确把握。因此，如果有一群优秀的工程师来审阅本书的内容，那么对于一本书的成功无疑是非常重要的。在本书的编写过程中，我很荣幸与一些优秀的工程师们共事（大多数都在. NET 部门工作），他们提出了极有见地的反馈和建议，并回答了许多常见的问题。真诚地感谢 Mark Russinovich、Maoni Stephens、Roberto Farah、Tess Ferrandez、Lee Culver、Pat Styles、Eric Eilebrecht、Steve Johnson 以及 Jon Langdon。

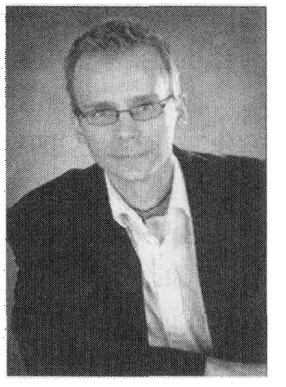
感谢 Patrick Dussud，他不仅非常细致地审阅了本书的手稿，而且还为本书作序。如果没有他的贡献，本书不会是大家现在看到的这样。

同样感谢 Easy Web Launch（www. easyweblaunch. com）的 Alexandra H. Anderson，感谢他为本书提供了网页。

最后，我想要感谢这么多年来为我提供反馈意见的所有读者。你们的支持使本书得以完成。

关 于 作 者

Mario Hewardt 是《Windows 高级调试》的作者之一，他是微软公司的资深开发经理。他拥有 11 年的工作经验，从 Windows 98 一直到 Windows Vista。在过去的几年中，Mario 主要从事 SaaS 领域的工作，开发了 Asset Inventory Service，这个服务用于帮助用户跟踪他们的资产清单。他目前正在领导一个团队，为下一代 Microsoft 在线管理服务开发核心支撑平台。



©www. BrookeClark. com

目　录

第二部分 调试实践