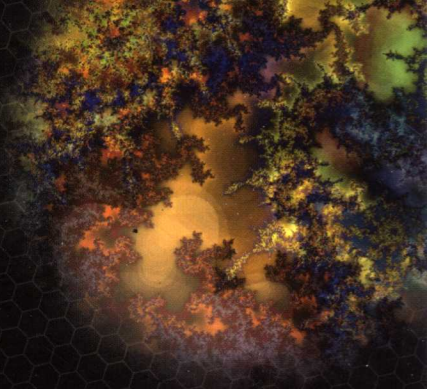
THE EXPERT'S VOICE® IN .NET



A Tester's Guide to .NET Programming

.NET 软件测试指南

(美) Randal Root
Mary Romero Sweeney 著
杨 浩 译

清华大学出版社

.NET 软件测试指南

(美) Randal Root
Mary Romero Sweeney 著
杨 浩 译

清华大学出版社
北 京

EISBN：1-59059-600-5
A Tester's Guide to .NET Programming
Randal Root, Mary Romero Sweeney

北京市版权局著作权合同登记号　图字：01-2006-4778

图书在版编目(CIP)数据

.NET 软件测试指南/(美)鲁特(Root, R.)，(美)斯维尼(Sweeney, M. R.)著；杨浩 译. —北京：清华大学出版社，2007.9
书名原文：A Tester' s Guide to .NET Programming
ISBN 978-7-302-15870-7
Ⅰ. N…　Ⅱ. ①鲁… ②斯… ③杨…　Ⅲ. 软件—测试—指南　Ⅳ. TP311.5-62

中国版本图书馆 CIP 数据核字(2007)第 118547 号

责任编辑：王　军　郑雪梅
装帧设计：孔祥丰
责任校对：成凤进
责任印制：何　芊
出版发行：清华大学出版社　　**地　　址**：北京清华大学学研大厦 A 座
http://www. tup. com. cn　**邮　　编**：100084
c - service@ tup. tsinghua. edu. cn
社 总 机：010-62770175　**邮购热线**：010-62786544
投稿咨询：010-62772015　**客户服务**：010-62776969
印 刷 者：清华大学印刷厂
装 订 者：三河市新茂装订有限公司
经　　销：全国新华书店
开　　本：185 × 260　**印　张**：33.75　**字　数**：780 千字
版　　次：2007 年 9 月第 1 版　　**印　次**：2007 年 9 月第1 次印刷
印　　数：1 ~ 4000
定　　价：59.99 元

序

多年来，我作为软件测试国际学院(IIST)的主席和 CEO，一直在尝试找出某种方式来帮助测试人员掌握必要的测试技术，以测试用现代开发技术编写的应用程序。我通过 Mary Sweeney 编著的 *Visual Basic for Testers* 一书(Apress 出版社，2001 年)，了解了 Mary 在这方面的能力，所以，让她也加入到我们的行列中。Mary 近两年来才成为 IIST 的一员，负责教授编程概念和数据库应用程序的测试。她在解决软件测试技术问题方面的能力是独一无二的，这已经通过她编写本书得到了证明。我一般没有时间为新书作序，但不想错过为本书写序的机会，因为本书的主题对于今天的专业测试人员非常重要。实际上，本书已经迟到得太久了。

对于需要掌握测试.NET 应用程序的专业测试人员来说，本书为他们带来了新的曙光。目前，.NET 应用程序的测试非常困难，因为这类应用程序可能非常复杂。在许多情况下，这种复杂性很可能使开发人员没有时间测试应用程序的各个方面，所以必须依赖专业测试人员对应用程序的所有内容进行全面彻底的测试。因此，本书是每位从事.NET 项目的专业测试人员的必备图书。测试.NET 应用程序对于软件开发人员和专业测试人员来说是一个技术挑战。本书介绍的方法和技巧肯定对开发人员和测试人员有所帮助，提高其.NET 应用程序的质量。本书非常便于专业测试人员学习，因为作者做了大量的工作，使高技术含量的术语易于为所有的专业测试人员理解。根据我对 Mary 教学风格的了解，这其实没有什么可惊讶的。我还特别欣赏每一章中的练习，它们是掌握相关技术主题的好帮手。

相信本书将填补专业测试人员和开发人员的知识结构之间的鸿沟，确保高质量的.NET 应用程序的传送。

Magdy Hanna 博士
软件测试国际学院(IIST)的主席和 CEO
www.iist.org

前　言

当今的软件测试环境已经发生了较大的变化，人们对软件开发人员和测试人员的要求在实质上是一样的，这是大势所趋。公司要求软件测试人员深入理解编程语言的知识，并具备深邃的数据库技巧。测试人员还必须不断学习，补充新的知识，才能更高效地测试复杂的项目。

测试工程师需要了解所有相关的知识，如操作系统、网络、数据库等，才能找出错误，并清晰明白地报告错误。我们总是对测试新手说，这是一个很不错的职业，需要“活到老，学到老”，就好像我们从来没有离开过学校——必须一直不断地学习(当然，如果我们总是不满足，也会不停地学习，因为我们不可能掌握所有的知识)。因而，本书适合于自我激励的测试工程师，他们会一直努力地补充新知识，学习更多用.NET 自动测试软件的知识。

本书还面向非编程人员，如从事网络和 IT 行业的人。这些人都是技术人员，需要了解.NET 编程的许多知识，才能提高自己的技能。有关测试的其他信息仅用于指导读者如何发现和处理系统中的问题。

本书也适合于测试主管和经理，了解.NET 可以对测试项目起哪些作用。自动测试软件的一个公开的秘密是，能通过商业渠道获得的主要工具尽管广告宣传得很好，但并不能完成我们需要完成的所有工作。期望某个工具完全支持自动测试功能，可以测试各种各样的应用程序是不切实际的，包括新添加到 Visual Studio's Team Edition 软件中的 Team Test 软件(详见第 11 章)。这个富有革命性的新软件对于中大型公司而言是一件不可多得的好东西，但仍旧需要测试人员的努力，才能在技术上使该软件更适合本公司的具体项目。

这并不是说，自己编写工具总是最佳答案。但是，使用自动测试工具的同时，辅之以测试人员用传统语言编写的一些脚本，可以使公司从自动测试项目中获益良多。这是我们的希望，在本书相关的章节中将介绍如何做到这一点。

关于本书

具体而言，本书有三个目标，主要给软件测试人员介绍如下内容：

- 如何把.NET 用作测试工具，包括如何创建简单的测试实用程序，以及编写测试程序代码的基本方式。

- 在写好的.NET 程序中查找什么内容。
- 理解软件开发过程，了解软件开发人员付出的努力。

本书将从.NET 的高级主题开始介绍，主要关注可用于软件测试的方面。

本书不介绍的内容

本书的重点是用.NET 测试软件，所以不探讨 Visual Studio 开发环境的开发功能，目前有许多图书介绍这方面的技术。

这不是一本软件测试入门图书，这方面也有许多图书(详见附录 C)。本书旨在为这两类图书搭一座桥梁，让读者学习如何用.NET 编写代码，以支持项目的自动测试。即使读者没有测试方面的背景，也应能阅读和理解本书的内容；但是，可能不清楚一些术语和测试概念的引用。附录 C 包含了这些读者需要的信息。

本书读者对象

本书适合于希望提高.NET 环境中测试水平的软件测试人员(通常简称为测试人员)，指导测试人员如何使用.NET 进行软件的自动测试和小程序的软件开发。本书没有介绍软件测试的基础知识，因为我们假定读者熟悉基本的测试概念。具备测试经验对于阅读本书有一定的帮助，但不是必需的。因此，从事 IT 和网络的人员也可以通过本书更熟悉.NET 编程。软件测试经理和主管也应能从本书中获得有用的信息，以理解如何把.NET 插入他们的测试项目。其他技术人员也可以从本书中获得有益的信息，但本书主要面向致力于确保软件质量的测试工程师。

从哪里开始

我们编写本书是为了提供各种知识，并告诉读者应从哪里开始。

首先，每位读者都应看看附录 A，其中提供了下载本书练习文件的信息，并帮助读者选择所需软件的版本和安装——包括启动系统，以运行 Web 页面。

其次，

- 没有编程经验的读者应从附录 B 开始，附录 B 介绍了大量的编程知识，之后阅读第 1 章，依次阅读其后的各章。试着完成每个练习，可以掌握每一章的主要内容。
- 有编程经验，但没有测试经验的读者应先阅读第 1 章，了解测试的基本知识，再查看目录，以确定从哪里开始，这取决于读者的兴趣。如果读者使用其他语言进行编程，至少可以跳过前面的内容，从第 2 章开始。快速浏览附录 B 也是有益的。

- 有编程经验和测试经验的读者可以根据需要，直接阅读自己感兴趣的章节，看看在测试项目中使用.NET 能做些什么。本书前面几章的内容比较基础，后面的章节将逐渐变难。

培训机构和教师要注意的事项

在软件测试课程中，本书可用作软件测试的教材，适合于软件自动测试中的初级和中级课程。本书的作者以本书为教材，利用 Sammamish 软件在课堂上教授软件测试。更多的信息以及本书的其他相关材料，请联系作者，作者的电子邮箱是 msweeney@sammamishsoftware.com 或 rroot@rootsource4training.com。

实践文件：练习的答案和演示代码

从第 2 章开始，每一章都有练习。这些练习的答案和各章的演示代码可以从 Apress 网站 www.apress.com 的 Source Code 部分或者 www.tupwk.com.cn/downpage 下载。

学习和使用.NET 的测试人员还可以在 www.sammamishsoftware.com 网站上下载其他相关的主题。有关本书的评论、问题或发现本书存在的错误，请联系作者，作者的电子邮箱是 msweeney@sammamishsoftware.com 或 rroot@rootsource4training.com。

目　录

第 1 章

用.NET 自动测试软件

软件测试并不是什么新东西，但成长速度非常快。在美国和世界范围内，软件测试正在以各种方式进行。在许多公司，传统的“黑盒”测试方式仍根深蒂固，但目前已出现了 Agile 和 Extreme 软件测试、其他各种术语和方法论。多年来，我们一直强调用昂贵的商用工具来测试软件，但现在对商用工具的追捧已渐趋平淡，因为人们对这些商用工具日益增长的功能要求已使工具的经销商不堪重负。现在，许多测试机构都转而生产自己的测试软件，通常是使用目前颇为流行的开放源代码的工具和软件。

2001 年，*Visual Basic for Testers*(Mary Romero Sweeney，Apress 出版社)一书出版，许多专业测试人员发现，在进行手动或自动测试时，需要辅之以自己的软件工具。自那以后，软件测试团体迫切需要掌握编写代码以生成测试结果，或编写自己的软件来创建测试实用工具的技术。每次会议和培训讨论会上，都有给测试人员讲授更多技术主题的课程或讲座，包括编程、网络和数据库。

我们总是需要快速、高效地使软件通过测试。在使软件测试自动化的过程中，有许多需要考虑的地方。如果思虑周详，对测试项目将非常有帮助，但如果处理不妥，可能会减慢测试的进度，甚至超出预算，无法完成任务。本书将探讨如何使用测试项目的.NET 应用程序软件，介绍这类软件可以用什么方式完成测试目标。

本书的内容教会读者用.NET 进行成功的项目测试。首先，需要讨论一般的软件自动测试。本章将介绍开始自动测试时涉及到的一些重要的管理问题，例如何时应进行自动测试，何时不应进行自动测试，需要哪些人员，以及如何建立自动测试团队。我们还将探讨创建测试软件的基本规则，以及给测试项目和实用工具使用 Visual Basic (VB) .NET 和 C#这两种.NET 语言的优缺点。

1.1 测试人员需要了解的.NET 编码知识

.NET 语言非常强大，可以完成一些测试任务，但需要有经验的测试人员和程序员编写代码。可惜，对于测试人员如何编写代码，以完成测试任务，并没有太多的资源。大多数资源都是面向软件开发人员的，而不是测试人员。

用.NET 语言进行测试需要在观念上有一个转变。测试人员可以学习标准的 Visual Basic 课程，但仍不知道如何把这些知识用于测试项目。这些课程和大多数图书都只介绍

控件的用法，以及创建界面友好的应用程序的方式。测试人员并不关心这些，而是关心如何快速开发实用程序，或如何使用代码获得系统信息和其他与测试相关的数据。本书和其他图书的一个区别是，本书不讨论控件或如何为应用程序开发漂亮的前端。这些都是测试人员应掌握的知识，但已有太多的图书介绍它们，所以本书主要介绍测试人员在测试项目上使用.NET 语言所必须尽快理解的内容：

- 如何访问.NET Framework 内部的库函数，返回平台、文件、注册表、操作系统等相关信息。
- 如何用基本控件创建一个前端，以查看测试信息和结果。
- 如何快速、方便地访问数据库。
- 如何访问 Windows 注册表，返回相关的应用程序信息。

当然，这些主题只是个开头，但它们代表了测试人员需要通过本书掌握的内容，以及用于完成测试任务的代码。本书将讨论这些主题和其他内容。

1.2 用.NET 语言进行测试的原因

.NET 语言并不是测试工具，它们是用于软件开发的编程语言。为什么用 Visual Basic 或 C#进行测试，而不使用 Perl、C 或 C++？像 Perl、Python、VBScript、Rexx 等脚本语言有许多用户，为什么不使用它们？实际上，这些语言都不是为测试软件而创建的。诚然，它们在测试项目上有很大的优势，尤其是对已安装了这些语言，并能熟悉使用它们的用户来说，优势就更明显。如果能很容易获得这些语言，而且员工对它们很熟悉，就可以选择它们完成许多测试编码工作。同样，如果很容易获得.NET 语言，员工能熟练使用它们，.NET 语言就是最佳选择。

如果所开发的项目本身就是用.NET 编写的，测试人员使用.NET 语言就是顺理成章的事。但在这方面存在一个误区：如果要测试.NET 应用程序，就需要用.NET 语言进行自动测试。这是不正确的，在 Windows 平台上使用.NET 语言，可以获得完成测试任务所需的所有功能。

既然.NET 语言并不是测试工具，怎么可能将它用于测试？.NET Framework 库有许多支持测试过程的功能。例如，有许多内部函数可以返回测试平台和被测试的应用程序的重要信息。.NET 的 Shell 函数和 SendKeys 类也可以用于运行应用程序，操作其图形用户界面(GUI)。Visual Studio Database Tools 可以连接数据库，检查其结构和数据。还可以编写自己的测试工具，例如负载测试程序。当然，较复杂的编程任务需要程序员花一定的时间编写。

.NET 语言还可以用于测试应用程序的许多后台操作。例如，可以编写脚本来访问系统环境变量和性能计数器。自动测试脚本可以检查正确的负载和文件信息的提取过程。.NET 语言是强大的开发工具，这也使得该语言能在测试方面大展拳脚。

1.3 为测试项目选择.NET 语言

.NET 平台包含许多语言选项，这是因为目前，语言只是.NET 公共语言运行库(CLR)上的一层，它们都编译为相同的中间语言(IL)。于是，选择哪种语言就是个人的喜好问题，而不是技术问题。我们可以根据语言的难易程度来选择语言。对于初学者，可以选择 Visual Basic。如果已经有另一种语言的基础了，如 C 或 Java，就可以选择这些语言的.NET 版本：C++或 J#。还可以选择 C#，它是专门为.NET 平台开发的一种新语言，以前使用流行语言编程的用户会很快熟悉并掌握它。

同一个团队的某些测试人员选择用 VB .NET 进行编码，而另一些测试人员选择 C#或其他语言，仍可以使软件很好地交互操作。我们必须摒弃某些对 VB .NET 的偏见，因为用 VB .NET 也可以编写大量代码，生产出好的产品。使用 Visual Basic 的一个主要优点是，它非常流行，很容易掌握，而且是使用广泛的 Microsoft Office 产品的宏语言和大多数 ASP 网页的脚本语言，许多其他软件公司也在其产品中使用 Basic。这说明，许多人都具备 Basic 的基础知识，使用或希望掌握它的人绝不会少。还有大量关于 Visual Basic 的图书和资源。尽管这些图书和资源不是专门为软件测试而编写的，但只要仔细阅读，就会在用户组和图书中找到许多可用于测试目的的代码。

C#的用户群一直在不断地扩大，因为它具备 VB .NET 的某些易用性，大量的编程结构又类似于 Java 和 Perl。C#的一个常见的误区是，它比 VB .NET 强大，但实际上 C# 和 VB .NET 的功能差不多。

那么，应选择哪种语言？两个都选可以吗？这两种语言都可编译为 IL，所以用一种语言编写的组件可以在另一种语言中使用。Visual Basic 和 C#都非常流行，也很容易掌握，所以本书的示例包含了用这两种语言编写的代码(为节省时间和幅面，本书没有包含 J#和 C++代码)。

1.4 软件自动测试的概念

下面回过头来看看自动测试的基本概念，并定义一些术语。首先，自动测试就是用软件完成所有的测试工作。换言之，就是编写代码，去测试另一些代码。自动测试包括所有他人编写的工具，因为他人编写的工具是测试软件的程序。编写自动测试脚本就是创建程序代码的过程，即编写用于测试软件的代码。自动测试脚本也称为测试软件，就是用于测试软件的程序代码。

以前，大多数测试工作都是手动完成的。也就是说，测试人员坐下来，用指定的过程运行应用程序，试图找出错误，并在发布产品前更正错误。软件自动测试则前进了一步。基本的软件测试比较严格，且有详细的说明，所以测试人员编写了一些软件，自动完成测试软件的一些工作。当然，许多成功的测试项目即使没有使用自动测试脚本也能完成。实际上，许多应用程序仍主要以手动测试为主。以用户体验的方式测试产品是无法替代的，经验丰富的测试人员的能力也是无法替代的。因而，自动测试永远不会(也不

应)完全替代应用程序的手动测试。但如果使用正确，自动测试可以大大加快测试进程。

自动测试越来越受到关注，是因为软件应用程序的复杂性和规模日益增加，需要更好更快的方式进行测试。自动测试并不能替代测试人员，虽然这是主管人员希望从自动软件测试中获得的一个好处，但自动测试可以用增强的功能改进测试过程。(实际上，只要新的自动测试项目一启动，常常需要更多的测试人员，而且是技术更高的测试人员。)测试过程有点乏味，也较费时间，但非常重要，不在项目中执行某些测试任务常常是因为时间限制或预算不足。例如，要验证大量文件或数据从一个系统传送到另一个系统，如果手动验证，就会被禁止；如果编写代码来验证，就是可以实现的。用自动测试脚本来改进测试过程有许多优点，下面列出自动测试的一些用途：

- 执行乏味或重复的手动测试任务，例如平台和应用程序的启动、退出和清理例程。
- 成批运行测试。
- 设置对 COM 对象或.NET 类的引用，测试其接口。
- 连接数据库，测试数据的有效性。
- 访问并检查 Windows 注册表。
- 创建支持测试过程的测试实用工具，如记录和启动脚本。

本书将探讨自动测试的上述用途和其他用途。

1.5 技术测试和非技术测试

很多人认为，所有的测试人员都是技术能人。毕竟，我们既是测试人员，也是开发人员。但是，所有的测试人员都需要编写代码，基本上是各自领域内的程序员，这种想法是错误的。在每个测试团队中，有一些精通编程的测试人员是有益的、明智的，但不需要团队中的所有测试人员都是编程高手。这是因为，程序员和测试人员的思维方式一般是不同的，而且也应该不同。精通编程的测试人员知道软件开发人员在什么情况下会犯什么样的错误，并可以据此改进测试项目。但是，这会使测试人员从分析的角度出发，而不是像一般的用户或没有这种编程技术的测试人员那样通盘考虑问题。因此，具备编程技术的测试人员可能会遗漏非技术测试人员不会遗漏的错误，反之亦然。另外，知识渊博、经验丰富的非技术测试人员是无可替代的，他的知识和技能不能仅由用户测试来替代，所以我们需要三种人：技术测试人员、非技术测试人员和用户测试人员。

1.6 进行自动测试的场合

并不是所有的测试任务都能从自己编写的测试代码中获益。事实上，在许多情况下，都不适宜进行自动测试。那么，该如何确定是否进行自动测试，以及何时进行自动测试？这需要分析和界定自动测试计划和手动测试计划的分界线。用 Visual Basic 还是 C#编程语言也需要仔细规划，因为编写测试脚本基本上是一个软件开发任务，可能会占用进度表中的大量时间。

如何确定哪些任务用手动测试，哪些任务用自动测试脚本来测试？经验固然重要，但在开始一个自动测试项目之前，要回答一些基本的问题。下面三小节将探讨如何确定项目是否应该使用自动测试。

1.6.1 项目和人员问题

测试主管常常在没有仔细考虑人手是否足够、人员是否有足够能力的情况下，就承担了自动测试项目。有得力的手下对于任何项目来说都是至关重要的。下面是一些必须考虑的重要事项：

- 自动测试的范围有多大？如果目标是自动完成所有的测试任务，这个范围就是不切实际的。如果试图把自动测试与已有的项目或新项目合并起来，最好从小一些的好管理的目标开始。例如，可以要求团队用 Visual Basic 或 C#编写一些简单的实用工具，以支持测试项目。这也有助于检查团队的经验水平，因为并不是所有的测试人员或程序员的能力都相同。
- 测试人员的自动测试水平如何？如果团队成员对自动测试很陌生，就需要花一定的时间和资金送他们去学习。在第一个项目开始之前，还需要招聘一些有经验的自动测试人员。经验水平决定了我们所能承担的自动测试级别。如果测试人员只具备编程的初级知识，就不能承担大项目。但他们可以用 Visual Studio 的一些工具或向导来完成测试项目，创建并使用一些简单的测试实用工具。
- 有多少高水平的测试人员？如果有高水平的测试人员，可以招聘到他们吗？许多项目在开始时有一些经验丰富的人员，但他们常常中途转向其他项目。这似乎非常不明智，但这种情况经常发生，我们都习以为常了。

1.6.2 产品问题

并不是所有的应用程序都能恰如其分地创建出来。在编写自动测试代码时，我们常常在 GUI 上工作，只考虑了软件的各个部分。但我们必须花一定的时间确定这是否适合于产品，还必须进行整体分析。下面就是应考虑的一些问题：

- 所测试的应用程序的功能集是否相对稳定？如果不稳定，在应用程序发生变化时，就需要修改所编写的测试代码。如果过早开始自动测试，就会陷入不断修改测试代码的泥潭，从而耗尽宝贵的预算。自动测试比较适合于结构和组件相对稳定的产品。
- 是否计划测试 UI？产品是基于 GUI 的吗？一些自动测试工具是专门面向 GUI 的。如果项目用于测试应用程序的 GUI，某些自动测试工具(包括商用和开发源代码的测试工具)就比较好。.NET 语言可以用于某种程度的 GUI 测试，但需要编写大量的代码。因此，在大多数情况下，不应该选择用.NET 进行大量基于 GUI 的测试。