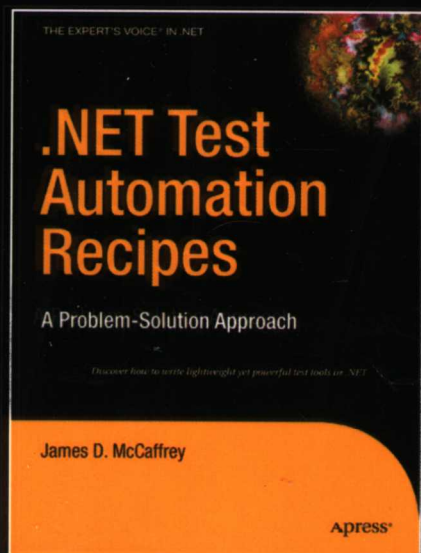


在.NET环境下编写轻量级的、功能强大的测试工具

.NET Test Automation Recipes: A Problem-Solution Approach

.NET软件测试 自动化之道



[美] James D. McCaffrey 著
刘晓伟 译



电子工业出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
<http://www.phei.com.cn>

.NET 软件测试自动化之道

.NET Test Automation Recipes: A Problem-Solution Approach

[美] James D. McCaffrey 著

刘晓伟 译

電子工業出版社

Publishing House of Electronics Industry

北京 • BEIJING

内 容 简 介

本书主要讲解在.NET环境下编写轻量级软件测试自动化程序的实用技术。全书内容由Windows UI测试、Web应用程序测试和存储过程及XML测试三部分组成,全面介绍了如何利用.NET平台提供的一系列技术(而不是依赖于第三方的商业自动化测试工具),采用C#语言编写轻量级的、功能强大的自动化测试程序。本书各个章节所提供的示例代码适用于单元测试(Unit Test)、集成测试(Integration Test)等软件开发过程中的各个测试环节。本书适合于所有在.NET平台上从事应用程序开发的程序员、测试人员以及自动化测试工具开发人员。

1-59059-663-3 .NET Test Automation Recipes: A Problem-Solution Approach by James D.McCaffrey.
Original English language edition published by Apress L. P., 2560 Ninth Street, Suite 219, Berkeley, CA
94710 USA. Copyright © 2006 by Apress L. P. Simplified Chinese-language edition copyright © 2007 by
Publishing House of Electronics Industry. All rights reserved.

本书简体中文专有翻译版权由 Apress L. P. 公司授予电子工业出版社,未经许可,不得以任何方式复制或抄袭本书的任何部分。

版权贸易合同登记号 图字:01-2006-6652

图书在版编目(CIP)数据

.NET 软件测试自动化之道 / (美) 麦克卡佛瑞 (McCaffrey, J.) 著; 刘晓伟译. —北京: 电子工业出版社, 2007.7

书名原文: .NET Test Automation Recipes: A Problem-Solution Approach

ISBN 978-7-121-04061-0

I. N… II. ①麦…②刘… III. 软件—测试—自动化 IV. TP311.5

中国版本图书馆 CIP 数据核字 (2007) 第 091774 号

责任编辑: 周筠 王继花

印 刷: 北京智力达印刷有限公司

装 订: 北京中新伟业印刷有限公司

出版发行: 电子工业出版社

北京市海淀区万寿路 173 信箱 邮编 100036

开 本: 787×1092 1/16 印张: 25.5 字数: 569 千字

印 次: 2007 年 7 月第 1 次印刷

定 价: 49.80 元

凡所购买电子工业出版社图书有缺损问题, 请向购买书店调换。若书店售缺, 请与本社发行部联系, 联系及邮购电话: (010) 88254888。

质量投诉请发邮件至 zlts@phei.com.cn, 盗版侵权举报请发邮件至 dbqq@phei.com.cn。服务热线: (010) 88258888。

译者序

与其他大多数软件测试书籍不同，这是一本重实践轻理论的测试书籍。本书的重点在于讲解如何针对.NET平台上不同类型的应用程序（Windows UI 程序，Web 程序和数据库应用程序）编写短小而高效的测试程序，这些测试程序适用于单元测试、集成测试等软件开发过程中的各个测试环节。本书所讲解的技术大多源自作者 James D. McCaffrey 在实际项目中所获得的宝贵经验。

关于书中部分术语的中文翻译，说明如下：test harness 统一译为“测试套件”，property 译为“属性”，attribute 保留英文不译（因为 property 和 attribute 中文意思相近而作为.NET 术语则含义完全不同）。另外，第 10 章“combinadic”和“factoradic”似是作者为了便于描述算法而自造的两个词，不见于其他数学文献或词典，故保留不译。

译者要特别感谢博文视点的周筠老师和编辑晓菲给予我的耐心和信任。感谢刘未鹏（<http://blog.csdn.net/pongba>）帮忙确定第 10 章部分数学术语的翻译。另外，还要感谢贾静香同学帮忙找出译文初稿中的部分错别字。

最后，译者希望本书中文版能够为开发人员和测试人员在软件测试的实际工作中提供帮助。

刘晓伟

2007 年 4 月于上海

关于作者

About the Author

JAMES MCCAFFREY 博士任职于 Volt Information Sciences 公司。他拥有南加州大学的博士学位、夏威夷太平洋大学信息系统专业的硕士学位、加州州立大学 Fullerton 分校数学专业的学士学位，以及加州大学 Irvine 分校心理学专业的学士学位。他曾是夏威夷太平洋大学的教授，并在微软公司作为主程序员参与过 IE 浏览器和 MSN Search 等关键产品的开发。

关于技术审校

About the Technical Reviewer

JOSH KELLING 是行业软件领域的私人顾问。他主修的专业是物理学，并通过自学成为软件开发人员，他有着近 10 年使用 Microsoft 技术开发行业软件和商业软件的经验。自从 .NET 推出 beta 版以来，他就主要专注于 .NET 平台上的开发工作。**JOSH KELLING** 的其他爱好包括：教书、滑雪、徒步旅行、采野蘑菇，以及打台球。

致谢

Acknowledgments

本书得以出版，得益于许多人的帮助。首先，我最应该感谢的是 Apress 出版社的 Jonathan Hassell 和 Elizabeth Seymour，在本书从概念到写作、编辑以及出版的整个过程中，他们都在不断地督促引导。Volt Information Sciences 公司的两位副总裁 Patrick Walker 和 Christina Harris 最早提出了本书的构想，并且为本书的写作提供了支持。本书的技术主审，Josh Kelling（供职于 Kelling Consulting）认真细致地找出并纠正了本书代码中的错误。我还要感谢 Doug Walter（供职于 Microsoft），他帮助我使得本书在技术细节上更为准确。本书的许多章节是基于我为 Microsoft 的 MSDN 杂志撰写的每月一次的专栏文章。MSDN 的编辑 Joshua Trupin 和 Stephen Toub 在写作方面给了我非常多的好建议，如果没有他们的这些建议，我不可能开始本书的写作。最后，我在 Volt 公司的职员——Shirley Lin, Lisa Vo Carlson, 以及 Grace Son——为我提供了很多行政方面的帮助。许多在 Microsoft 工作的 Volt 公司的软件工程师担当了本书辅助的技术和文字审校的角色。主要的技术审校者包括：Evan Kaplan、Steven Fusco、Bruce Ritter、Peter Yan、Ron Starr、Gordon Lippa、Kirk Slota、Joanna Tao、Walter Wittel、Jay Gray、Robert Hopkins、Sam Abolrous、Rich Bixby、Max Guernsey、Larry Briones、Kristin Jaeger、Joe Davis、Andrew Lee、Clint Kreider、Craig Green、Daniel Bedassa、Paul Kwiatkowski、Mark Wilcox、David Blais、Mustafa Al-Hasnawi、David Grossberg、Vladimir Abashyn、Mitchell Harter、Michael Svob、Brandon Lake、David Reynolds、Rob Gilmore、Cyrus Jamula、Ravichandhiran Kolandaiswamy 和 Rajkumar Ramasamy。

其他的技术审校者还包括：Jerry Frost、Michael Wansley、Vanarasi Antony Swamy、Ted Keith、Chad Fairbanks、Chris Trevino、David Moy、Fuhan Tian、C.J. Eichholz、Stuart Martin、Justice Chang、Funmi Bolonduro、Alemeshet Alemu、Lori Shih、Eric Mattoon、Luke Burtis、Aaron Rodriguez、Ajay Bhat、Carol Snyder、Qiusheng Gao、Haik Babaian、Jonathan Collins、Dinesh Ravva、Josh Silveria、Brian Miller、Gary Roehl、Kender Talyor、Ahlee Ly、Conan Callen、Kathy Davis 和 Florentin Ionescu。

文字上的审校者包括：Christina Zubelli、Joey Gonzales、Tony Chu、Alan Vanderwarka、Matt Carson、Tim Garner、Michael Klevitsky、Mark Soth、Michael Roshak、Robert Hawkins、Mark McGee、Grace Lou、Reza Sorasi、Abhijeet Shah、April McCready、Creede Lambard、Sean McCallum、Dawn Zhao、Mike Agranov、Victor Araya Cantuarias、Jason Olsan、Igor Bodi、Aldon Schwimmer、Andrea Borning、Norm Warren、Dale Dey、Chad Long、Thom Hokama、Ying Guo、Yong Wang、David Shockley、Allan Lockridge、Prashant Patil、Sunitha Mutnuri、Ping Du、Mark Camp、Abdul Khan、Moss Willow、Madhavi Kandibanda、John Mooney、Filiz Kurban、Jesse Larsen、Jeni Jordan、Chris Rosson、Dean Thomas、Brandon Barela 和 Scott Lanphear。

导 读

Introduction

本书讲的是什么

本书讲解的是在.NET 环境下编写轻量级软件测试自动化程序的实用技术。如果你从事的是.NET 软件的开发、测试或管理工作，那么这本书就是为你而写的。在.NET 之前，编写自动化测试程序往往和编写待测程序一样困难。有了.NET，只需要以前几分之一的时间就可以编写出轻量级、定制的自动化测试程序。说到轻量级的自动化测试程序，我的意思是源代码通常不超过两页或者编写时间不超过两个小时、小型、专用的测试套件。本书的重点在于介绍可以直接应用到日常工作的那些实用技术。

本书适合哪些人

本书的读者定位是采用.NET 技术的软件开发者、测试人员和管理人员。阅读本书的前提是要基本熟悉.NET，但是对具体的熟练程度并没有特殊的要求。书中的例子已经在一些讲座中成功地得到应用，这些讲座的听众既有初级程序员，又有高级的系统程序员，背景千差万别。另外，本书的内容也曾针对一些学习中级.NET 编程的学生进行过讲授，实践证明这些内容是非常有效的。

轻量级自动化测试的优点

本书所讲述的自动化测试技术是为了补充其他测试的不足，比如手工测试、测试驱动开发（Test-Driven Development）、基于模型的测试（Model-Based Testing）、开源测试框架、商业测试框架，等等，而不是为了替代这些测试。软件测试自动化（包括本书所讲解的技术）相比于手工测试有 5 个优点。有时候我们把自动化测试相比于手工测试的这些优点概括为 SAPES：更快的速度（Speed），更高的准确性（Accuracy）、精度（Precision）以及效率（Efficiency），并且更有利于测试者提升自身技能（Skill-Building）。此外，与开源测试框架和商业框架相比，轻量级的自动化测试没有前者那么陡峭的学习曲线（有些框架甚至要求你学习一门专用的脚本语言）。与商业自动化测试框架相比，轻量级的自动化测试花费相对便宜并且很容易进行定制。与开源测试框架相比，轻量级的自动化测试不会有太多版本更新和 bug 修复，从而更为稳定。轻量级的自动化测

试套件与商业测试框架和开源测试框架相比,最大的优点是可以调动测试者的主观能动性——轻量级的自动化测试积极鼓励编写有创造性的测试程序,而商业框架和开源框架常常限制你只须编写框架支持良好的那些测试类型。轻量级的自动化测试唯一的缺点是不太容易维护。因为轻量级的测试套件编写起来非常容易,如果不小心处理,所创建的大量测试套件、测试用例数据和测试用例结果文件可能使你不堪重负。测试过程管理超出了本书的范围,但是在编写轻量级的自动化测试程序时,这却是一个不容低估的有挑战性的问题。

关于代码

本书所有代码都采用 C# 语言编写。因为底层的 .NET 框架为各种编程语言提供了统一的平台,如果有需要,不用费太大力气就可以把书中的这些代码改写成 Visual Basic .NET 代码。书中的所有代码都经过了测试,并且在 Windows XP Professional (SP2) 和 Windows Server 2003 上采用 Visual Studio .NET 2003(with Framework 1.1)和 SQL Server 2000 成功运行。这些代码也针对 Visual Studio 2005 (with Framework 2.0) 和 SQL Server 2005 进行了测试。但是如果你的工作环境是后者,则需要对这些代码进行一些细微的改动。这些代码针对的是 32 位的操作系统,并没有针对 64 位的系统进行过测试。

如果你对软件测试自动化还不是很熟悉,你可能很快就会发现,作为一个测试人员编写代码与作为开发人员编写代码有着明显的不同。本书中的大部分代码都采用传统、脚本风格的编码方式,而没有采用面向对象的风格。我觉得采用脚本风格编写的自动化测试代码更容易理解,但这只是我个人的看法。

另外,很多示例代码并没有引入很多参数,也没有封装成方法。这么做是为了让代码更为简洁明了。大多数通常的错误检查代码(比如针对方法的输入参数进行检查)都被省略掉了。错误捕获对于产品级的自动化测试代码来说是至关重要的(毕竟,我们是在通过代码找出错误),但是用于错误检查的代码经常是核心代码的三四倍。你可以很容易地对本书中的代码进行修改,比如封装成方法、添加错误检查、集成到其他测试框架以及封装到工具类或者程序库。本书大部分章节所测试的都是出于演示目的的小程序。从设计上来说,这些小程序并不是好的编程风格的范例,而且这些待测程序经常还包含故意制造的一些错误。这么做是为了保持演示程序不至于过大并且也模拟了在开发过程中出于尚待完善状态的应用程序。例如,我通常使用 `textBox1` 这样的默认控件名称,而没有采用更具说明性的名称,局部变量名称我也取得很短(比如用 `s` 表示一个字符串变量),有时候我还会把多个语句写在同一行上,等等,不一而足。我在本书的插图中还留下一些“严重等级为 4”的 bug(印刷错误);试试看你能不能找到它们。大多数时候,我都尽量让自己使用的术语尽可能地准确。例如,当某个子程序是 C# 类的成员函数时,我使用术语方法(method),当指代 Win32 API 库的 C++ 子程序时,我使用术语函数。但是,有时候当我觉得使用意思相近的术语可能更容易理解或者更容易让别人读懂,我也会使用那些稍有偏差的术语。例

如，有时候我使用术语字符串变量（string variable）而不是更为准确的字符串对象（string object）来指代 C# 的字符串。

本书采用了提出问题并给出解决方案（problem-solution）的组织结构。这种方法的优点是方便地组织各种各样的自动化测试任务。但是为了让本书的篇幅限制在合理的范围内，大多数解决方案并不是完整并且独立的代码段。我的意思是说，我经常省略掉变量声明，或者不会一再地讨论解决方案中所用到的命名空间和工程引用，等等。许多解决方案经常会用到同一章中的其他解决方案，所以你应该仔细考虑一下它们之间的依赖关系，以及如何才能利用它们写出完整的测试套件。为了帮助读者理解一章的各个小节是如何一起工作的，每章的最后一个小节我都给出了完整的独立的程序代码。

本书内容

大多数计算机书籍，都会在导读部分概要性地介绍一下书中的内容。我不打算采取这种做法，我觉得要知道一本书的内容最好的方法是浏览一下它的目录；我自己通常就是这样做的。尽管如此，我还是想提一下本书的 4 个特定主题，这 4 个主题在我的同事中间激发了巨大的兴趣。第 1 章，“API 测试”从很多方面来说都是软件测试最基本的类型。如果你对软件测试还不是很熟悉，那么你不但可以从中学到有用的测试技术，而且还能了解到软件测试中的许多基本原则。第 3 章，“基于 Windows 的 UI 测试”介绍了通过用户界面对应用程序进行操纵的技术。即使有很多年经验的软件测试人员也会惊叹于通过 .NET 和第 3 章所讲述的技术，UI 测试能变得如此的容易。第 5 章“请求-响应测试”讲解了用于测试 Web 程序的基本技术。Web 开发人员和测试人员经常也会惊叹于在 .NET 环境下，这些技术是如此的强大。第 10 章“排列与组合”介绍了如何通过编写程序生成把输入值的所有排列或组合考虑在内的测试用例。有经验的测试人员和测试新手都觉得组合数学和 .NET 结合起来大大提高了测试用例的生成效率。

使用书中的代码

本书旨在为从事软件开发和测试的人员提供实际的帮助。很自然地，这就意味着你可以在你的系统或文档里使用本书给出的代码。但是这种授权并不包括在网站上或杂志文章中大量地复制本书中的代码，或者在会议讲演中使用本书给出的例子，等等。大多数作者（包括我在内）都会希望，如果您在书中或杂志中用到了他们书中给出的例子，那么最好能指明所援引例子的原作者。另外，作者对使用这些示例代码可能造成的后果不承担任何责任。

联系博文视点

您可以通过如下方式与本书的出版方取得联系。

读者信箱: sheguang@broadview.com.cn

投稿信箱: broadvieweditor@gmail.com

北京博文视点资讯有限公司 (武汉分部)

湖北省 武汉市 洪山区 吴家湾 邮科院路特 1 号 湖北信息产业科技大厦 1402 室

邮政编码: 430074

电话: (027)87690813 传真: (027)87690813 转 817

若您希望参加博文视点的有奖读者调查, 或对写作和翻译感兴趣, 欢迎您访问:

<http://bv.csdn.net>

关于本书的勘误、资源下载及博文视点的最新书讯, 欢迎您访问博文视点官方博客:

<http://blog.csdn.net/bvbook>

目录预览

Contents at a Glance

关于作者	I
关于技术审校	III
致谢	V
导读	VII
第 1 章 API 测试	3
第 2 章 基于反射的 UI 测试	33
第 3 章 基于 Windows 的 UI 测试	65
第 4 章 测试套件设计模式	97
第 5 章 请求-响应测试	135
第 6 章 基于脚本的 Web UI 测试	167
第 7 章 底层的 Web UI 测试	185
第 8 章 Web Service 测试	207
第 9 章 SQL 存储过程测试	237
第 10 章 排列与组合	265
第 11 章 ADO.NET 测试	301
第 12 章 XML 测试	335
索引	365

目 录

Contents

关于作者	I
关于技术审校	III
致谢	V
导读	VII

第 1 部分 Windows 应用程序测试

■ 第 1 章 API 测试	3
1.0 导读	3
1.1 存储用于测试用例的数据	6
1.2 读入测试用例数据	7
1.3 解析测试用例	8
1.4 把数据转换为合适的类型	9
1.5 判定测试用例通过与否	11
1.6 记录测试用例结果	13
1.7 给测试用例结果加上时间戳	16
1.8 通过计算对测试结果进行总结	17
1.9 获得测试运行的总时间	19
1.10 处理输入为空或期望值为空的情况	20
1.11 处理“方法抛出异常”的情况	22
1.12 处理输入参数为空字符串的情况	24
1.13 编写程序，在测试用例失败时发送警告邮件	26
1.14 自动运行测试套件	28
1.15 示例程序：ApiTest	29
■ 第 2 章 基于反射的 UI 测试	33
2.0 导读	33
2.1 运行待测程序	35
2.2 设置窗体的属性	39
2.3 获得窗体属性	44

2.4	设置控件的属性	47
2.5	访问控件的属性	50
2.6	方法调用	53
2.7	示例程序: ReflectionUITest	58
第 3 章	基于 Windows 的 UI 测试	65
3.0	导读	65
3.1	运行待测程序	66
3.2	获得待测程序主窗体的句柄	68
3.3	获得有名字控件的句柄	73
3.4	获得无名控件的句柄	75
3.5	发送字符给控件	78
3.6	鼠标单击一个控件	80
3.7	处理消息对话框	82
3.8	处理菜单	86
3.9	检查应用程序的状态	89
3.10	示例程序: WindowsUITest	91
第 4 章	测试套件设计模式	97
4.0	导读	97
4.1	创建使用文本文件存储数据并采用 Streaming 模型的测试套件	100
4.2	创建使用文本文件存储数据并采用 Buffered 模型的测试套件	104
4.3	创建使用 XML 文件存储数据并采用 Streaming 模型的测试套件	108
4.4	创建使用 XML 文件存储数据并采用 Buffered 模型的测试套件	113
4.5	创建用于轻量级自动化测试数据存储的 SQL 数据库	117
4.6	创建使用 SQL 存储数据并采用 Streaming 模型的测试套件	119
4.7	创建使用 SQL 数据并采用 Buffered 模型的测试套件	123
4.8	发掘关于待测系统的信息	126
4.9	示例程序: PokerLibTest	129

第 2 部分 Web 应用程序测试

第 5 章	请求-响应测试	135
5.0	导读	135
5.1	发送一个简单的 HTTP GET 请求并且取回响应	138
5.2	发送带有认证信息的 HTTP 请求并取回响应	139
5.3	发送复杂的 HTTP GET 请求并且取回响应	140

5.4	逐行返回 HTTP 响应的内容	141
5.5	发送简单的 HTTP POST 请求给传统的 ASP 网页	143
5.6	发送 HTTP POST 请求给 ASP.NET Web 应用程序	145
5.7	处理特殊的输入字符	150
5.8	通过编写程序确定 ViewState 和 EventValidation 的值	152
5.9	处理 CheckBox 和 RadioButtonList 控件	156
5.10	处理 DropDownList 控件	157
5.11	判断“请求-响应”的测试结果	159
5.12	示例程序: RequestResponseTest	162
第 6 章	基于脚本的 Web UI 测试	167
6.0	导读	167
6.1	创建一个基于脚本的 UI 测试套件结构	170
6.2	判断 Web 程序的状态	172
6.3	把进度信息记录在测试套件界面上	173
6.4	验证待测 Web 页面上 HTML 元素的值	174
6.5	操纵待测 Web 页面上 HTML 元素的值	176
6.6	把测试场景结果保存到客户端的文本文件	177
6.7	把测试场景的结果保存到服务器上的数据库表里	179
6.8	示例程序: ScriptBasedUITest	181
第 7 章	底层的 Web UI 测试	185
7.0	导读	185
7.1	启动 IE 并连接到这个实例	188
7.2	如何判断待测 Web 程序何时完全加载到浏览器	190
7.3	操纵并检查 IE Shell	192
7.4	操纵待测 Web 页面上的 HTML 元素的值	194
7.5	验证 Web 页面上 HTML 元素的值	195
7.6	创建一个 Excel 工作簿用来存储测试场景的运行结果	198
7.7	把测试场景的运行结果存入 Excel 工作簿	200
7.8	从 Excel 工作簿里读取已存储的测试结果	201
7.9	示例程序: LowLevelUITest	203
第 8 章	Web Service 测试	207
8.0	导读	207
8.1	使用 Proxy 机制测试 Web 方法	212
8.2	使用套接字测试 Web 方法	214

8.3 使用 HTTP 测试 Web 方法	220
8.4 使用 TCP 测试 Web 方法	222
8.5 使用存储在内存中的测试用例数据	226
8.6 把测试结果存储在内存中	229
8.7 示例程序: WebServiceTest	232

第 3 部分 部分数据测试

第 9 章 SQL 存储过程测试	237
9.0 导读	237
9.1 创建测试用例以及测试结果存储	239
9.2 执行 T-SQL 脚本	241
9.3 使用 BCP 工具导入测试用例数据	243
9.4 创建 T-SQL 测试套件	245
9.5 从 T-SQL 测试套件里把测试结果直接写入文本文件	249
9.6 当待测存储过程返回行集的时候, 如何判断测试结果是否通过	252
9.7 当待测存储过程返回 out 参数时, 如何判断测试结果是否通过	254
9.8 当待测存储过程没有返回值的时候, 如何判断测试结果是否通过	256
9.9 示例程序: SQLspTest	259
第 10 章 排列与组合	265
10.0 导读	265
10.1 创建数学组合对象	267
10.2 计算从 n 个数据中取出 k 个样本共有多少种取法	269
10.3 计算某个数学组合项的后继者	271
10.4 对于给定的 n 和 k , 生成所有的数学组合项	273
10.5 按照字典顺序找出某个数学组合的第 m 个项	275
10.6 把数学组合应用到字符串数组	278
10.7 创建数学排列对象	280
10.8 计算 n 阶排列的排列数	282
10.9 计算某个数学排列项的后继项	284
10.10 针对给定的 n 值, 产生所有的数学排列项	286
10.11 按照字典顺序, 找出某个数学排列的第 k 个项	287
10.12 把数学排列应用到字符串数组	291
10.13 示例程序: ComboPerm	293

第 11 章 ADO.NET 测试	301
11.0 导读	301
11.1 当期望值是 DataSet 的时候, 如何判断测试结果是否通过	303
11.2 测试有返回值的存储过程	306
11.3 测试返回值为行集 (Rowset) 的存储过程	309
11.4 测试把返回值放入 out 参数的存储过程	311
11.5 测试没有返回值的存储过程	314
11.6 测试不通过存储过程对数据库进行访问的程序	318
11.7 比较两个 DataSet 对象是否相等	321
11.8 从文本文件把测试用例数据读入到 SQL 数据库表	324
11.9 从 SQL 表把测试用例数据读入到文本文件	327
11.10 示例程序: ADOdotNETTest	329
第 12 章 XML 测试	335
12.0 导读	335
12.1 通过 XmlTextReader 解析 XML	337
12.2 通过 XmlDocument 解析 XML	339
12.3 通过 XPathDocument 解析 XML	341
12.4 通过 XmlSerializer 解析 XML	343
12.5 通过 DataSet 对象解析 XML 文件	347
12.6 通过 XSD Schema 对 XML 文件进行验证	350
12.7 通过 XSLT 对 XML 进行修改	353
12.8 通过 XmlTextWriter 对 XML 文件进行写操作	355
12.9 比较两个 XML 文件是否严格相等	356
12.10 不考虑编码方式, 比较两个 XML 文件是否严格相等	358
12.11 比较两个 XML 文件的规范等价性	359
12.12 示例程序: XmlTest	361
索引	365