

.NET 2.0 Interoperability Recipes

A Problem-Solution Approach



Bruce Bukovics

.NET 2.0 Interoperability Recipes: A Problem-Solution Approach

Copyright © 2006 by Bruce Bukovics

All rights reserved. No part of this work may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording, or by any information storage or retrieval system, without the prior written permission of the copyright owner and the publisher.

ISBN-13: 978-1-59059-669-2

ISBN-10: 1-59059-669-2

Printed and bound in the United States of America 9 8 7 6 5 4 3 2 1

Trademarked names may appear in this book. Rather than use a trademark symbol with every occurrence of a trademarked name, we use the names only in an editorial fashion and to the benefit of the trademark owner, with no intention of infringement of the trademark.

Lead Editor: Ewan Buckingham

Development Editor: Ralph Davis

Technical Reviewers: Christophe Nasarre, Nicholas Paldino

Editorial Board: Steve Anglin, Dan Appleman, Ewan Buckingham, Gary Cornell, Jason Gilmore,

Jonathan Hassell, James Huddleston, Chris Mills, Matthew Moodie, Dominic Shakeshaft, Jim Sumser, Matt Wade

Project Manager: Sofia Marchant

Copy Edit Manager: Nicole LeClerc

Assistant Production Director: Kari Brooks-Copony

Production Editor: Ellie Fountain

Compositor: Kinetic Publishing Services, LLC

Proofreader: Elizabeth Berry

Indexer: John Collin

Cover Designer: Kurt Krames

Manufacturing Director: Tom Debolski

Distributed to the book trade worldwide by Springer-Verlag New York, Inc., 233 Spring Street, 6th Floor, New York, NY 10013. Phone 1-800-SPRINGER, fax 201-348-4505, e-mail orders-ny@springer-sbm.com, or visit <http://www.springeronline.com>.

For information on translations, please contact Apress directly at 2560 Ninth Street, Suite 219, Berkeley, CA 94710. Phone 510-549-5930, fax 510-549-5939, e-mail info@apress.com, or visit <http://www.apress.com>.

The information in this book is distributed on an “as is” basis, without warranty. Although every precaution has been taken in the preparation of this work, neither the author(s) nor Apress shall have any liability to any person or entity with respect to any loss or damage caused or alleged to be caused directly or indirectly by the information contained in this work.

The source code for this book is available to readers at <http://www.apress.com> in the Source Code section.

Contents

About the Author	xi
About the Technical Reviewers	xiii
Acknowledgments	xv
Introduction	xvii
■ CHAPTER 1 Using C-Style APIs	1
1-1. Identifying the Unmanaged Function	2
1-2. Using the Function from Managed Code	5
1-3. Simplifying Reuse of Unmanaged Functions	8
1-4. Changing the Calling Convention	14
1-5. Renaming a Function	16
1-6. Changing the Character Set Used for Strings	21
1-7. Using Data Types That Improve Performance	25
1-8. Handling Errors from Unmanaged Functions	32
1-9. Using C++ Interop As a Managed Wrapper	41
1-10. Catching Unmanaged Exceptions with C++ Interop	50
1-11. Freeing Unmanaged Memory	58
1-12. Requesting Permission to Access Unmanaged Code	65
1-13. Securing Access to Unmanaged Code	72
1-14. Calling Functions Dynamically	83
■ CHAPTER 2 C-Style APIs: Structures, Classes, and Arrays	87
2-1. Passing Structures	88
2-2. Returning a Structure from Unmanaged Code	92
2-3. Specifying the Exact Layout of a Structure	97
2-4. Controlling Field-Level Marshaling Within Structures	104
2-5. Allocating Memory Within Structures	110
2-6. Passing Classes to Unmanaged Code	116
2-7. Passing Simple Arrays	126
2-8. Handling String Arrays	130
2-9. Passing Arrays of Structures	137

CHAPTER 3	Win32 API	141
3-1.	Accessing ANSI or Wide Functions	141
3-2.	Retrieving the Win32 Error Code	147
3-3.	Handling Callbacks	152
3-4.	Using Windows Constants	156
3-5.	Handling Handles	167
3-6.	Passing Managed Objects	173
3-7.	Marshaling Win32 Types	177
3-8.	Replacing Win32 Calls with .NET	179
CHAPTER 4	Using C++ Interop	191
4-1.	Using C++ Classes	193
4-2.	Mixing Managed and Unmanaged Code	199
4-3.	Detecting Compile-Time Traits	205
4-4.	Using Managed Objects from Unmanaged Code	209
4-5.	Marshaling Strings	216
4-6.	Marshaling Structures and Embedded Pointers	220
4-7.	Handling Callbacks with C++ Interop	224
4-8.	Using C++ As a Custom COM Wrapper	230
CHAPTER 5	Using COM	235
5-1.	Using COM Components from .NET	236
5-2.	Importing a Type Library	242
5-3.	Handling COM Events	248
5-4.	Marshaling COM Data Types	256
5-5.	Marshaling COM Variants	263
5-6.	Marshaling COM Arrays	272
5-7.	Extending COM Classes	281
5-8.	Changing the Apartment Model	290
5-9.	Refactoring for Performance	296
5-10.	Creating a Late-Bound COM Object	298
5-11.	Sharing an Interop Assembly	303
5-12.	Deploying Your Application	306
5-13.	Converting HRESULTs to Exceptions	308
5-14.	Refactoring HRESULTs	314
5-15.	Retrieving the HRESULT	319
5-16.	Providing Additional Error Information	322

CHAPTER 6	Exposing Managed Code to COM	327
6-1.	Exposing .NET Classes Using Late Binding	328
6-2.	Exposing .NET Classes Using Early Binding	332
6-3.	Exposing .NET Classes with Interfaces	338
6-4.	Managing COM Identity	346
6-5.	Controlling COM Visibility	353
6-6.	Preparing Assemblies for COM Interop	361
6-7.	Exposing Managed Events to COM	364
6-8.	Providing HRESULTs for Exceptions	369
6-9.	Preserving Success HRESULTs	374
CHAPTER 7	Marshaling to COM Clients	379
7-1.	Controlling Parameter Direction	379
7-2.	Marshaling Strings	392
7-3.	Marshaling Arrays	402
7-4.	Marshaling Variants	414
7-5.	Marshaling Currency	418
7-6.	Marshaling Null Variant Strings	423
7-7.	Marshaling Classes and Structures	426
7-8.	Passing Optional Parameters	436
CHAPTER 8	COM+ Enterprise Services	441
8-1.	Exposing Managed Code to COM+	442
8-2.	Implementing a Server Application	448
8-3.	Installing a Serviced Component	453
8-4.	Registering Components Dynamically	456
8-5.	Activating Components Just-in-Time	459
8-6.	Using Object Pooling	467
8-7.	Implementing Private Components	475
8-8.	Using Role-Based Security	479
8-9.	Performing Manual Security Checks	488
8-10.	Writing Managed Queued Components	493

■ **CHAPTER 9 COM+ Enterprise Services Transactions** 503

 9-1. Monitoring Transaction Status..... 504

 9-2. Enabling Automatic Transactions 515

 9-3. Placing an Automatic Vote 528

 9-4. Placing a Manual Vote 535

 9-5. Defining a Unit of Work..... 544

 9-6. Controlling the Transaction Isolation Level 556

 9-7. Implementing Transactional Code Blocks 558

 9-8. Building Your Own Resource Manager..... 564

 9-9. Using Services Without Components..... 577

■ **INDEX** 583