

AutoCAD 2006 VBA

A Programmer's Reference



Joe Sutphin

AutoCAD 2006 VBA: A Programmer's Reference

Copyright © 2005 by Joe Sutphin

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: 1-59059-579-3

Library of Congress Cataloging-in-Publication data is available upon request.

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: Tony Davis

Technical Reviewers: Phillip Ash, Steve Johnson

Editorial Board: Steve Anglin, Dan Appleman, Ewan Buckingham, Gary Cornell, Tony Davis,
Jason Gilmore, Jonathan Hassell, Chris Mills, Dominic Shakeshaft, Jim Sumser

Associate Publisher: Grace Wong

Project Manager: Kylie Johnston

Copy Edit Manager: Nicole LeClerc

Copy Editors: Candace English, Kim Wimpsett

Assistant Production Director: Kari Brooks-Copony

Production Editor: Janet Vail

Compositor: Linda Weidemann, Wolf Creek Press

Proofreaders: Linda Seifert and Sue Boshers

Indexer: Broccoli Information Management

Interior Designer: Van Winkle Design Group

Cover Designer: Kurt Krames

Manufacturing Manager: 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.

This book is dedicated to my wife, Grace.

*Without her, I would not be able to
accomplish the task of writing a book.*

Contents at a Glance

About the Author	xxv
About the Technical Reviewers	xxvii
Acknowledgments	xxix
Introduction	xxxix
■ CHAPTER 1 The VBA Integrated Development Environment (VBAIDE)	1
■ CHAPTER 2 Introduction to Visual Basic Programming	23
■ CHAPTER 3 Application Elements	55
■ CHAPTER 4 AutoCAD Events	67
■ CHAPTER 5 User Preferences	75
■ CHAPTER 6 Controlling Layers and Linetypes	87
■ CHAPTER 7 User Interaction and the Utility Object	111
■ CHAPTER 8 Drawing Objects	143
■ CHAPTER 9 Creating 3-D Objects	173
■ CHAPTER 10 Editing Objects	205
■ CHAPTER 11 Dimensions and Annotations	231
■ CHAPTER 12 Selection Sets and Groups	259
■ CHAPTER 13 Blocks, Attributes, and External References	285
■ CHAPTER 14 Views and Viewports	321
■ CHAPTER 15 Layout and Plot Configurations	337
■ CHAPTER 16 Controlling Menus and Toolbars	355
■ CHAPTER 17 Drawing Security	383
■ CHAPTER 18 Using the Windows API	391
■ CHAPTER 19 Connecting to External Applications	403
■ CHAPTER 20 Creating Tables	415
■ CHAPTER 21 The SummaryInfo Object	427
■ CHAPTER 22 An Illustrative VBA Application	437
■ APPENDIX A AutoCAD Object Summary	447
■ APPENDIX B AutoCAD Constants Reference	631
■ APPENDIX C System Variables	671
■ INDEX	697

Contents

About the Author xxv

About the Technical Reviewers xxvii

Acknowledgments xxix

Introduction xxxi

CHAPTER 1 The VBA Integrated Development Environment (VBAIDE) 1

 Visual Basic Concepts 1

 Windows, Events, and Messages 1

 Event-Driven vs. Procedural Programming 2

 Developing Your Applications Interactively 3

 Starting the Editor 3

 Exploring the User Interface 4

 The Project Explorer 5

 The Code Window 5

 The Properties Window 6

 The Object and Procedure Boxes 6

 The Immediate Window 7

 The Options Dialog Box 8

 Managing Projects 9

 Project Structure 9

 Creating, Opening, and Saving Projects 10

 Adding, Saving, and Removing Files 12

 Adding ActiveX Controls and Code Components 13

 The Object Browser 15

 VBARUN and the Macros Dialog Box 17

 Overview of AutoCAD VBA Commands 22

 Summary 22

CHAPTER 2	Introduction to Visual Basic Programming	23
Variables		23
Declaring Variables		23
Variable Scope and Lifetime		25
Constants		27
Data Types		27
Introduction to Arrays		29
Modules		31
UserForm		31
Procedures		32
Calling Procedures		34
Passing Arguments to Procedures		34
Control Structures		34
Decision Structures		34
If ... Then		35
If ... Then ... Else		35
Select Case		36
Loop Structures		37
Do While ... Loop		37
Do ... Loop While		38
Do Until ... Loop and Do Loop ... Until		38
For ... Next		39
For ... Each ... Next		40
Nested Control Structures		40
Exiting a Control Structure		41
Exiting a Sub or Function Procedure		42
With ... End With		42
Application Writing Techniques		43
Writing Statements on Multiple Lines		43
Combining Statements on a Single Line		43
Adding Comments to Your Code		44
Overview of Object-Oriented Programming		44
Objects and Classes		44
Object Data		45
Private Variables		45
Public Variables		45
An Object's Behavior		45
Debugging Basics		46
Summary		53

CHAPTER 3	Application Elements	55
	Designing a UserForm	55
	Adding a UserForm to Your Application	55
	Setting UserForm Properties	56
	Adding a Control to a Form	58
	Visual Basic ActiveX Controls	60
	Label	60
	TextBox	61
	ComboBox	61
	ListBox	62
	CheckBox	62
	OptionButton	63
	ToggleButton	63
	Frame	64
	CommandButton	64
	Additional ActiveX Controls	64
	TabStrip	64
	MultiPage	65
	ScrollBar	65
	SpinButton	65
	Image	65
	Summary	66
CHAPTER 4	AutoCAD Events	67
	Application-Level Events	67
	Document-Level Events	70
	The BeginCommand and EndCommand Events	71
	The BeginOpen and EndOpen Events	72
	The BeginClose and BeginDocClose Events	72
	The Activate and Deactivate Events	73
	The BeginSave and EndSave Events	73
	Object-Level Events	73
	Summary	74
CHAPTER 5	User Preferences	75
	Getting and Setting Support Path(s)	77
	Controlling Cursor Size	78
	Getting and Setting the AutoSaveInterval Property	78
	Getting and Setting the Drawing Template File Path	79

Getting and Setting the Printer Support Path	79
Getting and Setting the File SaveAs Type	80
Enabling and Disabling the Startup Dialog Box	82
Saving and Retrieving Personal Preferences	83
User Preferences Changes in AutoCAD 2004	84
Summary	85

■ CHAPTER 6 **Controlling Layers and Linetypes**

Layers	87
Accessing Layers	88
Iterating Layers	88
Checking for Existing Layers	89
Creating a New Layer	91
Making a Layer Active	92
Turning a Layer On/Off	94
Setting a Layer to Be Frozen or Thawed	95
Locking/Unlocking a Layer	95
Making Layers Plottable	96
Renaming a Layer	96
Deleting a Layer	97
Getting a Layer's Handle	98
Layer Colors	98
Layer Linetypes	99
Layer Lineweights	100
Linetypes	100
Accessing Linetypes	101
Checking for Existing Linetypes	102
Loading a Linetype	103
Making a Linetype Active	105
Renaming a Linetype	106
Deleting a Linetype	107
Getting a Linetype's Handle	108
Changing a Linetype's Description	108
Scaling Linetypes	109
Summary	110

CHAPTER 7	User Interaction and the Utility Object	111
	Interface Methods	111
	Input Methods and Dialogs	111
	The Prompt Method	113
	The InitializeUserInput Method	113
	The GetXXX Methods	115
	Handling Errors in User Input	128
	Conversion Methods	129
	The AngleToReal Method	129
	The AngleToString Method	130
	The DistanceToReal Method	130
	The RealToString Method	131
	The AngleFromXAxis Method	132
	The PolarPoint Method	133
	The TranslateCoordinates Method	134
	Internet Methods	136
	The IsURL Method	136
	The LaunchBrowserDialog Method	137
	The GetRemoteFile Method	139
	The IsRemoteFile Method	140
	The PutRemoteFile Method	140
	Summary	141
CHAPTER 8	Drawing Objects	143
	Controlling the Drawing Space	143
	The ModelSpace and PaperSpace Collections	144
	Creating Objects	145
	Circular Objects	145
	Line Objects	150
	Other Objects of Interest	162
	Summary	172

CHAPTER 9	Creating 3-D Objects	173
	Understanding the 3DSolid Object	173
	Creating Simple Solid Objects	174
	The Box	174
	The Cone	176
	The Cylinder	177
	The Sphere	179
	The Torus	180
	The Wedge	181
	Creating Elliptical 3-D Objects	183
	The Elliptical Cone	183
	The Elliptical Cylinder	185
	Creating Extruded and Revolved Objects	186
	The Extruded Solid	187
	The Extruded Solid Along a Path	189
	The Revolved Solid	191
	Editing Solids	194
	Boolean Operations	194
	Interference Operation	196
	Slicing Solids	198
	Sectioning Solids	199
	Analyzing Solids: Mass Properties	201
	Summary	203
CHAPTER 10	Editing Objects	205
	Editing with Methods	205
	Copying Objects	206
	Deleting Objects	206
	Exploding Objects	207
	Highlighting Entities	208
	Mirroring Objects	209
	Moving Objects	212
	Offsetting Objects	213
	Rotating Objects	215
	Scaling Objects	218
	Object Arrays	219

Editing with Properties	223
Changing an Object's Color	223
Changing an Object's <code>TrueColor</code> Property	225
Changing an Object's Color Properties	226
Changing an Object's Layer	226
Changing an Object's Linetype	227
Changing an Object's Visibility	228
The <code>Update</code> Method	229
Summary	229
 CHAPTER 11 Dimensions and Annotations	231
Working with Dimensions	231
Using the <code>DimStyle</code> Object	231
Setting Dimension Styles	232
Using the <code>CopyFrom</code> Method	232
Using Dimension Styles	233
Creating Dimensions	235
Using the <code>Tolerance</code> Object	245
Working with Annotations	248
Using the <code>TextStyle</code> Object	248
Using a <code>TextStyle</code>	254
Adding Annotations	256
Using the <code>Leader</code> Object	256
Summary	258
 CHAPTER 12 Selection Sets and Groups	259
Selection Sets	259
Adding a <code>SelectionSet</code> Object	259
Accessing and Iterating Selection Sets	260
Selecting Entities	261
Adding and Removing Items	273
The <code>Clear</code> , <code>Delete</code> , and <code>Erase</code> Methods	276
The <code>PickFirstSelectionSet</code> Property	277
Groups	278
Adding a Group Object	279
Accessing and Iterating Groups	279
Adding and Removing Items	280
The <code>Delete</code> Method	282
Summary	283

CHAPTER 13	Blocks, Attributes, and External References	285
	Blocks and Block References	286
	Accessing Block Objects	288
	Creating Blocks	289
	Defining and Manipulating Blocks	294
	Using MInsertBlock Objects	301
	Using External References	305
	Attributes	310
	Creating Attributes	310
	Manipulating Attribute References	315
	Summary	320
CHAPTER 14	Views and Viewports	321
	Views	321
	Creating a View	322
	Setting a View as Current	324
	Deleting a View	324
	Viewports	325
	The Model-Space Viewport	325
	The Paper-Space Viewport	331
	Summary	336
CHAPTER 15	Layout and Plot Configurations	337
	The Plot Object	337
	Plotting Your Drawing	338
	Plot Configurations	343
	Controlling Plot Parameters	347
	Summary	354
CHAPTER 16	Controlling Menus and Toolbars	355
	The MenuGroups Collection	356
	Loading Menu Groups	356

The MenuGroup Object	359
Saving Menu Groups	361
Unloading Menu Groups	362
Accelerator Keys	362
Changing the Menu Bar	362
Editing Menus	369
Editing Toolbars	376
Summary	382
CHAPTER 17 Drawing Security	383
Digital Signatures	383
The Action Property	384
The Algorithm Property	385
The Issuer Property	385
The ProviderName Property	386
The SerialNumber Property	386
The Subject Property	386
The TimeServer Property	387
Password Protection	387
Summary	389
CHAPTER 18 Using the Windows API	391
Declarations	391
Windows Data Structures	392
Visual Basic-to-DLL Calling Conventions	392
Specifying the Library	393
The Major Windows DLLs	394
Working with Windows API Procedures That Use Strings	394
Passing Arguments by Value or by Reference	395
Learning by Example	395
OpenFile Common Control Dialog Replacement for VBA	395
SaveAsFile Common Control Dialog Replacement for VBA	398
Retrieving the Status of the Caps Lock, Num Lock, and Scroll Lock Keys	400
Summary	402

CHAPTER 19	Connecting to External Applications	403
	Making the Connection	403
	Connecting to Microsoft Excel	405
	Creating a New Workbook	405
	Creating a New Worksheet	405
	Accessing a Worksheet	406
	Writing and Reading Cells	406
	Saving and Exiting Excel	406
	Connecting to Microsoft Word	407
	Creating a New Document	407
	Adding Text to the Document	407
	Setting Page Orientation	407
	Setting Margins	407
	Setting the Document Header and Footer	408
	Saving and Exiting Word	408
	Connecting to a Microsoft Access Database	408
	Connecting to a Database File	408
	Retrieving a Set of Records	409
	Writing Values to the Database File	410
	Closing the Connection	410
	Working with Other Databases	410
	Connectivity Automation Objects	411
	Advanced Database Issues	411
	Working with Services and Other APIs	411
	Summary	413
 CHAPTER 20	 Creating Tables	 415
	The AddTable Method	415
	The RegenerateTableSuppressed Property	416
	The GetText Method	416
	The SetText Method	417
	The GetTextHeight Method	418
	The SetTextHeight Method	419
	The InsertRows Method	420
	The InsertColumns Method	421
	Putting It All Together	423
	Summary	426

CHAPTER 21 The SummaryInfo Object	427
Overview	428
Properties	428
The Author Property	428
The RevisionNumber Property	429
The Subject Property	429
The Title Property	430
Adding Custom SummaryInfo	430
The AddCustomInfo Method	430
The GetCustomByKey Method	431
The NumCustomInfo Method	432
The GetCustomByIndex Method	432
The RemoveCustomByKey Method	433
The RemoveCustomByIndex Method	433
The SetCustomByKey Method	434
The SetCustomByIndex Method	434
Summary	435
 CHAPTER 22 An Illustrative VBA Application	 437
Start Building the Application	438
Writing the Code	438
Initial Declarations	438
Create a Session of Microsoft Word	439
Create a New Document in Word	440
Create a Table with the Word Document	440
Create Column Headings	440
Populate the Table with AutoCAD Layer Data	441
Helper Functions	442
Adjust the Fonts	444
Sort the Table	444
Autofit Column Text	445
Add Page Header and Footer	445
Print the Report	446
Summary	446

■ APPENDIX A AutoCAD Object Summary	447
AutoCAD Collections	447
The Application Property	448
The AcadObject Object	448
The AcadEntity Object	451
AcadDimension Object	454
AutoCAD Object Reference	457
Acad3DFace Object	457
Acad3DPolyline Object	459
Acad3DSolid Object	460
AcadApplication Object	461
AcadArc Object	465
AcadAttribute Object	466
AcadAttributeReference Object	468
AcadBlock Object	469
AcadBlockReference Object	476
AcadBlocks Collection	477
AcadCircle Object	478
AcadDatabase Object	479
AcadDatabasePreferences Object	480
AcadDictionaries Collection	482
AcadDictionary Object	483
AcadDim3PointAngular Object	484
AcadDimAligned Object	486
AcadDimAngular Object	491
AcadDimArcLength Object	492
AcadDimDiametric Object	497
AcadDimOrdinate Object	501
AcadDimRadial Object	504
AcadDimRadialLarge Object	508
AcadDimRotated Object	513
AcadDimStyle Object	517
AcadDimStyles Collection	518
AcadDocument Object	519
AcadDocuments Collection	525
AcadDynamicBlockReferenceProperty Object	526
AcadEllipse Object	527
AcadExternalReference Object	528
AcadFileDependency Object	529
AcadFileDependencies Object	529

AcadGroup Object	530
AcadGroups Collection	531
AcadHatch Object	531
AcadHyperlink Object	534
AcadHyperlinks Collection	535
AcadIDPair Object	535
AcadLayer Object	536
AcadLayers Collection	537
AcadLayout and AcadPlotConfiguration Objects	538
AcadLayouts and AcadPlotConfigurations Collections	541
AcadLeader Object	542
AcadLine Object	544
AcadLineType Object	545
AcadLineTypes Collection	546
AcadLWPolyline Object	547
AcadMenuBar Collection	548
AcadMenuGroup Object	549
AcadMenuGroups Collection	550
AcadMInsertBlock Object	551
AcadMLine Object	552
AcadModelSpace Collection	553
AcadMText Object	553
AcadPaperSpace Collection	555
AcadPlot Object	555
AcadPlotConfiguration Object	557
AcadPlotConfigurations Collection	557
AcadPoint Object	557
AcadPolyfaceMesh Object	558
AcadPolygonMesh Object	559
AcadPolyline Object	561
AcadPopupMenu Object	563
AcadPopupMenuItem Object	565
AcadPopupMenuCollection	566
AcadPreferences Object	567
AcadPreferencesDisplay Object	568
AcadPreferencesDrafting Object	570
AcadPreferencesFiles Object	572
AcadPreferencesOpenSave Object	574
AcadPreferencesOutput Object	575
AcadPreferencesProfiles Object	576

AcadPreferencesSelection Object	577
AcadPreferencesSystem Object.....	578
AcadPreferencesUser Object.....	579
AcadPViewport Object	580
AcadRasterImage Object	583
AcadRay Object.....	585
AcadRegion Object.....	586
AcadRegisteredApplication Object.....	587
AcadRegisteredApplications Collection.....	587
AcadSelectionSet Object	588
AcadSelectionSets Collection	590
AcadShape Object	590
AcadSolid Object	591
AcadSortEntsTable Object	592
AcadSpline Object.....	593
AcadState Object	595
AcadSummaryInfo Object	596
AcadTable Object	597
AcadTableStyle Object	604
AcadText Object	606
AcadTextStyle Object	607
AcadTextStyles Collection	609
AcadTolerance Object	609
AcadToolbar Object.....	611
AcadToolbarItem Object.....	613
AcadToolbars Collection	614
AcadTrace Object.....	615
AcadUCS Object.....	616
AcadUCSs Collection	617
AcadUtility Object	618
AcadView Object	621
AcadViewport Object	622
AcadViewports Collection	624
AcadViews Collection	625
AcadXline Object	626
AcadXRecord Object.....	626
SecurityParams Object.....	627
AcCmColor Object	628
LayerStateManager Object	629

APPENDIX B AutoCAD Constants Reference	631
Ac3DPolylineType	631
AcActiveSpace	631
ACAD_LWEIGHT	631
AcAlignment	632
AcAlignmentPointAcquisition	633
AcAngleUnits	633
AcARXDemandLoad	634
AcAttachmentPoint	634
AcAttributeMode	634
AcBlockScaling	635
AcBooleanType	635
AcCellAlignment	635
AcCellEdgeMask	636
AcCellType	636
AcColor	637
AcColorMethod	638
AcCoordinateSystem	638
AcDimArcLengthSymbol	639
AcDimArrowheadType	639
AcDimCenterType	640
AcDimFit	640
AcDimFractionType	640
AcDimHorizontalJustification	641
AcDimLUnits	641
AcDimPrecision	641
AcDimTextMovement	642
AcDimToleranceJustify	642
AcDimToleranceMethod	642
AcDimUnits	643
AcDimVerticalJustification	643
AcDrawingAreaSCMCommand	644
AcDrawingAreaSCMDefault	644
AcDrawingAreaSCMEdit	644
AcDrawingDirection	645
AcDynamicBlockReferenceProperty	645
AcEntityName	645
AcExtendOption	646
AcGradientPatternType	647
AcGridLineType	647

AcHatchObjectType.....	647
AcHatchStyle.....	648
AcHorizontalAlignment.....	648
AcInsertUnits	648
AcISOPenWidth	649
AcKeyboardAccelerator.....	650
AcKeyboardPriority	650
AcLeaderType.....	650
AcLineSpacingStyle	650
AcLineWeight.....	651
AcLoopType	652
AcMenuFileType	652
AcMenuGroupType.....	652
AcMenuItemType	652
AcMLineJustification.....	653
AcOleQuality.....	653
AcPatternType	653
AcPlotPaperUnits.....	654
AcPlotPolicy.....	654
AcPlotRotation	654
AcPlotScale.....	654
AcPlotType	656
AcPolylineType	656
AcPolymeshType	656
AcPreviewMode	657
AcPrinterSpoolAlert	657
AcProxyImage.....	657
AcRegenType.....	657
AcRotationAngle.....	658
AcRowType	658
AcSaveAsType.....	659
AcSelect	659
AcSelectType.....	660
AcShadePlot.....	660
AcTableDirection.....	661
AcTableStyleOverrides.....	661
AcTextFontStyle.....	665
AcTextGenerationFlag	665
AcToolbarDockStatus	666
AcToolbarItemType.....	666

AcUnits.....	666
AcVerticalAlignment.....	667
AcViewportScale.....	667
AcViewportSplitType.....	668
AcWindowState.....	669
AcXRefDemandLoad.....	669
AcZoomScaleType.....	669
APPENDIX C System Variables.....	671
The GetVariable Method.....	671
The SetVariable Method.....	672
INDEX	697

About the Author



■ **JOE SUTPHIN**'s background includes more than 25 years in the machinery manufacturing industry. He has more than 18 years of CAD experience with 12+ years of AutoCAD-specific experience. Joe is an Autodesk-registered developer, and his work has appeared in the pages of *Cadence* and *Cadalyst* magazines. He has been programming for nearly 20 years, with the last 15 years being Visual Basic-specific experience. In 1998 he collaborated with Microsoft on a Visual Basic application case study. He is the author of the best-selling book *AutoCAD 2000 VBA Programmer's Reference*.

About the Technical Reviewers

■ **PHILLIP ASH** began programming in the early 80s when public schools had either Apple IIs or TRS-80s. DOS was loaded from a cassette player, and BASIC was the *lingua franca* of the tyro programming set. He got involved in computer-aided design as a hobby in the late 80s, and after a few semesters in an architectural design and drafting curriculum, he took a job as an AutoCAD drafter/designer at a naval architecture firm in Portsmouth, Virginia. It was there he was introduced to AutoLISP.

For the next few years, Phill wrote code exclusively in AutoLISP. He began dabbling with Visual Lisp shortly after its inception and, at about the same time, started writing macros for Excel in Visual Basic for Applications. When Autodesk included VBA in AutoCAD, everything came together.

Eight years later, Phill is a senior developer of ShipWorks, the only shipbuilding AutoCAD add-on used by Northrop Grumman Newport News in the design of next-generation aircraft carriers.

Phill lives in Virginia Beach, Virginia, with his wife, Amy; sons, Alex and Rowan; and daughter, Morgan.

■ **STEVE JOHNSON** was born in England but has lived in Perth, Western Australia, since 1984. He is married with two children and has a bachelor's degree in applied science (information science). Steve has been an AutoCAD specialist since 1985. His company, cad nauseam, provides AutoCAD-based consulting, development, support, training, and technical writing services. His software for AutoCAD is used by hundreds of clients around the world.

Steve is a contributing editor of *Cadalyst* magazine and has written the monthly column *Bug Watch* since 1995. He is also the vice president, Asia Pacific, of Cadlock and a past president of the Western Australian AutoCAD User Group (WAAUG). As a former committee member on the Applications Programming Special Interest Group of the North American Autodesk User Group, he contributed to the NAAUG newsletter. NAAUG is now known as Autodesk User Group International (AUGI). Steve was part of the AUGI Benchmark Committee, which assisted in the development of the AUGI Gauge, a comprehensive AutoCAD benchmark.

Acknowledgments

I have many, many people to thank on this project. Please forgive me if I miss mentioning your name. First, the people at Apress: without them I would not have had the opportunity to create this work. Thanks to Gary Cornell for his initial acceptance and input, Kylie Johnston for coordinating things, and Tony Davis for getting it all started.

A great big thank you to the editorial and production staff: Candace English and Kim Wimpsett, my copy editors; Janet Vail, production editor; Nancy Wright, formatter; Linda Weidemann, compositor; Linda Seifert, proofreader; and Kevin Broccoli, indexer. I enjoyed working with each of you.

Second, a special thanks to my technical reviewers, Phillip Ash and Steve Johnson.

A special thank you to my children—Stephen, Clair, David, and Emily—for understanding when Daddy had to work.

Lastly, thanks to those who helped and whose names I forgot to include here. To everyone in the AutoCAD community with whom I've had the pleasure of crossing paths, from the bottom of my heart, thanks!

Introduction

This book provides a concise guide to the kind of customization programmers can achieve with AutoCAD 2006. It demonstrates how to use AutoCAD through short code examples written in Visual Basic for Applications (VBA). It also includes a complete quick reference that lists all the events, methods, and properties available with AutoCAD. Finally, it describes all the constants and system variables.

What Is AutoCAD?

So, what is AutoCAD? First released in 1982 under the name MicroCAD, AutoCAD has become a powerful tool for drafting and design purposes. AutoCAD 2006 incorporates many new features to enhance flexibility and drawing control. To reflect this extra functionality, many new ActiveX objects, properties, methods, and events have been included for improved programmability.

What Is This Book About?

This book is about AutoCAD 2006 and how to use AutoCAD VBA in your applications to handle all your drawing tasks more efficiently. It shows you how to programmatically control the creation and editing of individual drawing objects, manipulate linetypes and layers, control text and dimension styles, and do much more. As you encounter each of these topics, you'll learn all about the associated objects, including their properties, methods, and events.

By interfacing with AutoCAD, you can exploit all of AutoCAD's functionality that would have taken you a long time to write yourself. This book will first help you learn how to use this functionality. Then it will become a handy reference later, when you have a question that you just can't answer.

This book splits topics into neat and intuitive segments and makes it easy to find specific information when you need it (that is, when you're coding real-world applications).

This book is divided into three main parts:

- Chapters 1 through 3 provide a rapid introduction to Visual Basic and explain the notation and commands particular to AutoCAD VBA projects.
- Chapters 4 through 22 supply a detailed breakdown of most of the AutoCAD object model, covering common tasks complete with several varied code examples demonstrating how to use the relevant objects' methods and properties.
- Finally, the quick-reference appendixes describe all the members of all the AutoCAD objects at a glance. Appendix D, on Object Model Cross-Reference, can be found on the Apress website (www.apress.com).

Who Is This Book For?

The book is a reference guide for AutoCAD programmers, and it's primarily designed to explain and demonstrate the features of AutoCAD 2006. As such, this isn't a beginner's guide; however, if you've programmed in any language that can interface with other COM objects, you should be able to easily understand and use this book.

In particular, the book is aimed at programmers who use AutoCAD for daily tasks and can see the benefits of customizing and automating these tasks. I present programming techniques needed to create and modify AutoCAD drawings, customize preferences, query and set system variables, and so on, using the built-in VBA.

You can customize AutoCAD to any degree of sophistication. If you can think it up, then I bet you can use AutoCAD VBA and this book to help you achieve your goal.

Tell Us What You Think

I've worked hard on this book to make it enjoyable and useful. My best reward would be to hear from you that you liked it and that it was worth the money you paid for it. I've done my best to try to understand and match your expectations.

Apress and I would like to know what you think about it. Tell us what you liked best, and what we could have done better. If you think this is just a marketing gimmick, then test us—drop us a line! We'll answer, and we'll take whatever you say under consideration for future editions. The easiest way to do so is to send e-mail to feedback@apress.com.

You can also find more details about Apress on the Web site at <http://www.apress.com>. There you'll find the code from the latest Apress books, sneak previews of forthcoming titles, and information about the authors and editors. You can order Apress titles directly from the site or find out where your nearest local bookstore with Apress titles is located.

Customer Support

If you find a mistake in the book, your first port of call should be the errata page for this book on the Web site at <http://www.apress.com>. You can see any errata already posted there or submit your own.

If you can't find an answer there, send an e-mail to support@apress.com telling us about the problem. We'll do everything we can to answer promptly. Please remember to let us know the book title your query relates to and, if possible, the page number. This will help us reply to you quickly.



The VBA Integrated Development Environment (VBAIDE)

Within AutoCAD, you develop VBA programs in the Visual Basic for Applications (VBA) Integrated Development Environment (IDE). As it does with the Visual LISP IDE, Autodesk provides the VBAIDE as an integral part of many of its products, including AutoCAD. Unlike the Visual LISP IDE, however, Microsoft licenses the VBAIDE to Autodesk for inclusion in its products. Therefore, its features are from Microsoft, not Autodesk.

This chapter explores the VBAIDE environment's facets and shows you how to take advantage of its tools. It covers these topics:

- Visual Basic concepts
- Starting the editor
- Exploring the user interface
- Managing projects
- Using the text editor
- The Object Browser

Visual Basic Concepts

Since VBA is a Microsoft Windows development environment, you'll find developing in VBA easiest if you have some knowledge of Windows. If you are new to Windows, you'll find fundamental differences between programming in Windows and programming in other environments, such as Visual LISP. The next sections outline concepts of Windows programming that may be new to you.

Windows, Events, and Messages

Explaining the inner workings of Windows requires much more space than is available in this book. But you don't need an extensive knowledge of the Windows workings to create useful

applications. The Windows operating system can be simplified to three basic concepts: windows, events, and messages.

A window is a rectangular region on the screen that has its own border. The AutoCAD drawing window, Notepad, a Word document, and the place where you compose an e-mail are all windows.

Windows can have hierarchy. A dialog form, which is a window within a distinct application, contains relevant ActiveX components and code. A drawing window is the parent of a dialog form window, and AutoCAD is the parent of the drawing windows within it. The operating system is the parent “window” of all applications running in it, including AutoCAD.

Each window recognizes and controls the programs that execute inside them or their subordinate windows. To manage windows, Windows assigns a unique ID known as a *handle*, or *hWnd* in programming jargon, to each window. Windows uses *events* to constantly monitor each window for signs of activity. Events change the application environment or *system state*. They occur when a user acts, such as by clicking the mouse or pressing a key; programmatically; or by another window through system processes.

Each time an event is triggered, Windows sends a message to the hosting application. Windows processes the message and broadcasts it to the windows. Then, based on its own instructions, each window can take appropriate action such as repainting itself when uncovered by another window. In the case of VBA in AutoCAD, the VBA work space intercepts event messages. VBA programs can then respond directly or pass the event up to AutoCAD or to Windows if necessary. VBA provides a controlling environment within AutoCAD in which to execute and respond to events, either directly or by allowing AutoCAD or Windows to respond.

Although this seems like a lot of work, VBA hides most of the low-level details from you and exposes *event procedures*, which are routines that execute when a particular event occurs, for your convenience. You can quickly create very powerful applications without being concerned with low-level details.

Event-Driven vs. Procedural Programming

When a traditional procedural application runs, it follows a predetermined path that controls the portions and sequence of code executed. It starts with the first line of code and progresses from the top down, calling each procedure when needed, until reaching the end of the code. This predetermined path is the major difference between procedural and event-driven applications.

Event-driven applications do not have a predetermined destiny. Different sections of code are executed based upon the events triggered in whatever order they occur. Depending on what events occur when the application runs, some sections of code may not get executed at all.

You can't predict the sequence of events, so you must make assumptions about the application's “state” at any moment. This seems like it would be difficult, but it's really not. Typically, you have a set of possibilities to work with, such as the Click, DblClick, KeyPress, and LostFocus events. For example, you might require the user to type a value in a TextBox before enabling a CommandButton that allows further processing. The TextBox control's Change event would contain code that enables the CommandButton control, as shown in this sample code:

```
Private Sub TextBox1_Change()  
    If Len(TextBox1.Text) > 0 Then  
        CommandButton1.Enabled = True  
  
    Else  
        CommandButton1.Enabled = False  
    End If  
End Sub
```

Each time you add or delete text from the TextBox control, the program executes this event procedure. The code checks the length of the text and if it is greater than zero, meaning there is something in the TextBox, it enables the CommandButton. Otherwise, it disables the CommandButton.

Programmatically changing the text in the TextBox triggers the Change event. If you allow for this occurrence, you might get unexpected results. Using events is very powerful, but you must know what each event might trigger elsewhere in your application.

Developing Your Applications Interactively

In more-traditional development environments, most developers follow a distinct three-step process: writing, compiling, and testing. However, VBA uses a more interactive approach to development that makes it easier for both beginning and experienced developers.

Languages such as C++ require you to write all the code then compile it. During the compile, you may uncover numerous errors, from simple typing errors to more-complex syntax errors. The Visual Basic programming environment, on the other hand, interprets your code every line of the way, alerting you to potential problems now instead of during a lengthy compile cycle.

Because Visual Basic is partially compiling your code as you type it, it takes very little time to finish compiling the code and execute your application. Unlike with other languages, you will find that you are constantly writing, executing, and refining your application. In addition, the Visual Basic environment employs a graphical environment. You most often work in the graphical interface first and on the code second. This lets you spend more time creating and less time compiling and recompiling.

Starting the Editor

One of the first questions that you will face is “How do I enter source code (structured commands) or develop a user interface (forms or dialog boxes)?” The answer is the IDE, a graphical user interface you use to develop applications. It is similar to development environments provided in other applications, such as Microsoft Access and Microsoft Excel.

To display the VBAIDE, choose Tools ► Macro ► Visual Basic Editor or press Alt+F11. Alternatively, you can start the editor by typing VBAIDE at the AutoCAD command prompt, as shown in Figure 1-1.



Figure 1-1. *The AutoCAD command prompt*

Note AutoCAD includes a Visual Basic menu and toolbar. You can load it by copying `ACAD.DVB` from `\Sample\VBA\VBAIDEMenu` (in the AutoCAD folder) to a directory in your support path. The toolbar will then autoload when the first VB command is launched. If you already have an `ACAD.DVB`, you can cut and paste from the provided `ACAD.DVB` to your own. This provides an AutoCAD toolbar for the following commands: `VBAIDE`, `VBAPREF`, `VBAMAN`, `VBALOAD`, and `VBARUN`. This also adds New, Open, and Close to the File pull-down on the `VBAIDE` toolbar.

Exploring the User Interface

The editor is composed of several different windows. The first time you open it, it looks like Figure 1-2. Use the View menu to control which windows are visible. To get context-sensitive help on any window, click in it and press F1.

The rest of this section discusses the most frequently used windows.

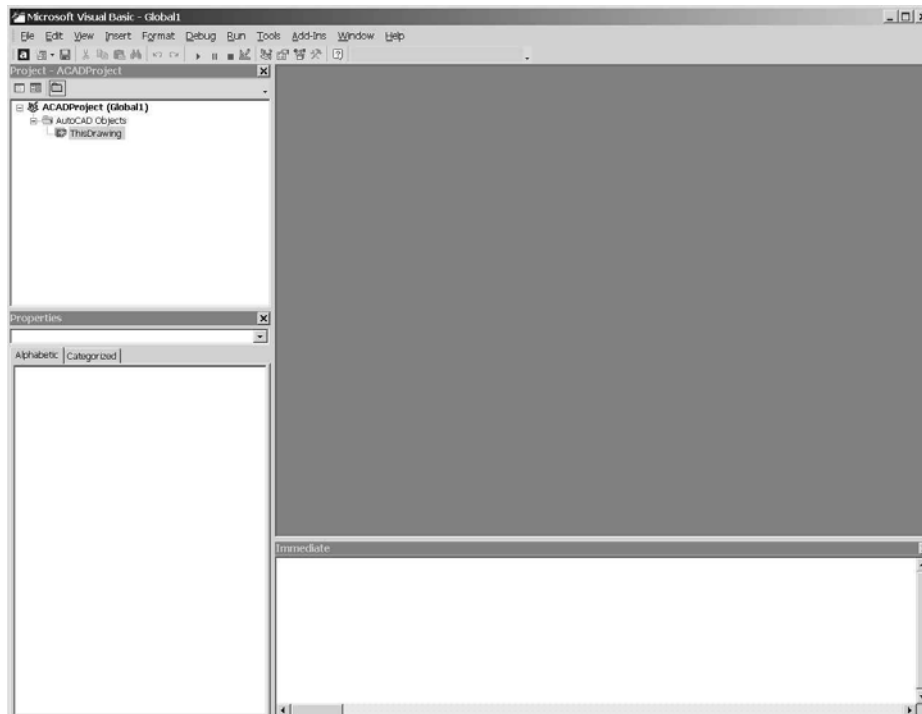


Figure 1-2. *The AutoCAD VBAIDE*