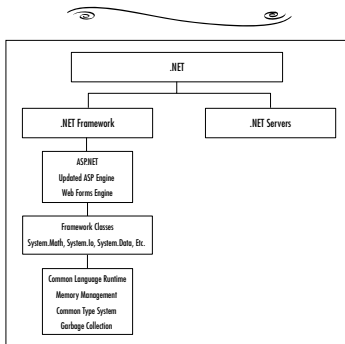


Contents

.NET Architecture



From the Series Editor **xxxi**

Chapter 1 New Features in Visual Basic .NET 1

Introduction 2

Examining the New IDE 3

Cosmetic Improvements 3

Development Accelerators 5

.NET Framework 6

A Very Brief and Simplified History 6

.NET Architecture 7

ASP.NET 7

Framework Classes 8

.NET Servers 8

Common Language Runtime 8

History 8

Convergence 9

Object-Oriented Language 10

Object-Oriented Concepts 10

Advantages of Object-Oriented Design 11

History of Object Orientation and VB 13

Namespaces 13

Web Applications 13

Web Applications Overview 13

Web Forms 14

Web Services 15

HyperText Transport Protocol 16

Simple Object Access Protocol 17

Security	17
Type Safety	18
Casting	18
Data Conversion	19
Bitwise Operations	20
New Compiler	20
Compiling an Executable	20
Architecture	21
File Management in Previous Versions of VB	21
File Management	22
Changes from Visual Basic 6.0	23
Variants	23
Variable Lower Bounds	23
Fixed Length Strings	23
NULL Propagation	23
Other Items Removed	24
Function Values	24
Short Circuits	25
Properties and Variables	25
Variable Lengths	25
Get and Set	26
Date Type	26
Default Properties	27
Summary	28
Solutions Fast Track	28
Frequently Asked Questions	31
Chapter 2 The Microsoft .NET Framework	33
Introduction	34
What Is the .NET Framework?	34
Introduction to the Common Language Runtime	35
Using .NET-Compliant Programming Languages	37
Creating Assemblies	39
Using the Manifest	42
Compiling Assemblies	45
Assembly Cache	45
Locating an Assembly	45

NOTE

Visualization is still key! Die-hard VB programmers may find themselves having a hard time visualizing all the new concepts in VB.NET (and we all know that proper logic visualization plays a big role in what we do). Something that may help is to think about VB.NET as a completely flexible language that can accommodate Web, console, and desktop use.

Private Assembly Files	51
Shared Assembly Files	51
Understanding Metadata	51
The Benefits of Metadata	52
Identifying an Assembly with Metadata	53
Types	53
Defining Members	54
Using Contracts	54
Assembly Dependencies	55
Unmanaged Assembly Code	55
Reflection	56
Attributes	57
Ending DLL Hell	58
Side-by-Side Deployment	58
Versioning Support	59
Using System Services	60
Exception Handling	60
StackTrace	61
InnerException	61
Message	61
HelpLink	62
Garbage Collection	62
Console I/O	62
Microsoft Intermediate Language	63
The Just-In-Time Compiler	63
Using the Namespace System to Organize Classes	64
The Common Type System	65
Type Safety	68
Relying on Automatic Resource Management	68
The Managed Heap	69
Garbage Collection and the Managed Heap	71
Assigning Generations	77
Utilizing Weak References	77
Security Services	79
Framework Security	80
Granting Permissions	81

Gaining Representation through a Principal	82
Security Policy	83
Summary	85
Solutions Fast Track	85
Frequently Asked Questions	88

Chapter 3 Installing and Configuring VB.NET

91

Installing Visual Studio .NET

- Phase 1: Installing
Windows components
- Phase 2: Installing
Visual Studio .NET
- Phase 3: Checking for
service releases

Introduction	92
Editions	92
Installing Visual Studio .NET	93
Exercise 3.1: Installing Visual Studio .NET	94
Installing on Windows 2000	99
The New IDE	100
Integrated Development Environment	
Automation Model	100
Add-Ins	104
Exercise 3.2 Creating an Add-In Using the Add-In Wizard	105
Wizards	109
Macros	109
Home Page	110
Project Options	112
Toolbox	116
Child Windows	120
Window Types	122
Arranging Windows	123
Task List	123
Exercise 3.3 Setting Up a Custom Token	124
TaskList Views	124
Locating Code	126
Annotating Code	126
Solution Explorer	127
Properties Window	129
Form Layout Toolbar	130
Hide/Show Code Elements	132

Web Forms	133
Intellisense	134
Customizing the IDE	135
Customizing the Code Editor	135
Customizing Shortcut Keys	135
Customizing the Toolbars	136
Exercise 3.4 Adding a New Toolbar to the Existing Set	136
Exercise 3.5 Adding Commands to Toolbars	137
Customizing Built-In Commands	137
Exercise 3.6 Creating an Alias	138
Customizing the Start Page	139
Accessibility Options	141
Summary	142
Solutions Fast Track	142
Frequently Asked Questions	143

Developing & Deploying...

Embrace Your Parameters

VB.NET is insistent upon enclosing parameters of function calls within parentheses regardless of whether we are returning a value or whether we are using the Call statement. It makes the code much more readable and is a new standard for VB programmers that is consistent with the standard that nearly all other languages adopted long ago.

Chapter 4 Common Language Runtime 145

Introduction	146
Component Architecture	148
Managed Code versus Unmanaged Code	150
Interoperability with Managed Code	152
System Namespace	153
File I/O	155
Drawing	156
Printing	157
Common Type System	158
Type Casting	160
Garbage Collection	163
Object Allocation/Deallocation	164
Close/Dispose	165
Summary	166
Solutions Fast Track	167
Frequently Asked Questions	168

Chapter 5 .NET Programming Fundamentals

171

Introduction	172
Variables	173
Constants	175
Structures	176
Program Flow Control	178
If...Then...Else	178
Select Case	182
While Loops	184
For Loops	186
Arrays	187
Declaring an Array	188
Multidimensional Arrays	189
Dynamic Arrays	191
Functions	192
Object Oriented Programming	196
Inheritance	196
Polymorphism	197
Encapsulation	197
Classes	198
Adding Properties	198
Adding Methods	200
System.Object	201
Constructors	201
Overloading	202
Overriding	203
Shared Members	205
String Handling	206
Error Handling	210
Summary	213
Solutions Fast Track	214
Frequently Asked Questions	217

NOTE

When porting Visual Basic applications to Visual Basic .NET, be careful of the lower bounds of arrays. If you are using a for loop to iterate through the array, and it is hard-coded to initialize the counter at 1, the first element will be skipped. Remember that all arrays start with the index of 0.

Chapter 6 Advanced Programming Concepts

219

What Are Collections?

Collections are groups of like objects. Collections are similar to arrays, but they don't have to be redimensioned. You can use the *Add* method to add objects to a collection. Collections take a little more code to create than arrays do, and sometimes accessing a collection can be a bit slower than an array, but they offer significant advantages because a collection is a group of objects whereby an array is a data type.

Introduction	220
Using Modules	221
Utilizing Namespaces	222
Creating Namespaces	222
Understanding the Imports Keyword	226
Implementing Interfaces	229
Delegates and Events	232
Simple Delegates	235
Multicast Delegates	236
Event Programming	236
Handles Keyword	236
Language Interoperability	237
File Operations	239
Directory Listing	239
Data Files	241
Text Files	243
Appending to Files	246
Collections	246
The Drawing Namespace	248
Images	253
Printing	256
Understanding Free Threading	262
SyncLock	263
Summary	265
Solutions Fast Track	265
Frequently Asked Questions	267

Chapter 7 Creating Windows Forms

269

Introduction	270
Application Model	270
Properties	271
Manipulating Windows Forms	275
Properties of Windows Forms	275
Methods of Windows Forms	276
Creating Windows Forms	287

Creating Dialog Boxes


1. Create a form.
2. Set the **BorderStyle** property of the form to **FixedDialog**.
3. Set the **ControlBox**, **MinimizeBox**, and **MaximizeBox** properties of the form to **False**.
4. Customize the appearance of the form appropriately.
5. Customize event handlers in the Code window appropriately.

Displaying Modal Forms	288
Displaying Modeless Forms	289
Displaying Top-Most Forms	289
Changing the Borders of a Form	289
Resizing Forms	291
Setting Location of Forms	292
Form Events	294
Creating Multiple Document Interface Applications	297
Creating an MDI Parent Form	297
Creating MDI Child Forms	298
Exercise 7.1 Creating an MDI Child Form	298
Determining the Active MDI Child Form	299
Arranging MDI Child Forms	299
Adding Controls to Forms	300
Anchoring Controls on Forms	301
Docking Controls on Forms	303
Layering Objects on Forms	304
Positioning Controls on Forms	304
Dialog Boxes	305
Displaying Message Boxes	306
Common Dialog Boxes	306
The OpenFileDialog Control	306
The SaveFileDialog Control	309
The FontDialog Control	311
The ColorDialog Control	313
The PrintDialog Control	315
The PrintPreviewDialog Control	316
The PageSetupDialog Control	321
Creating Dialog Boxes	322
Creating and Working with Menus	323
Adding Menus to a Form	323
Exercise 7.2 Adding a Menu to a Form at Design Time	323

Dynamically Creating Menus	326
Exercise 7.3 Adding a Menu to a Form at Design Time	326
Adding Status Bars to Forms	328
Adding Toolbars to Forms	330
Data Binding	332
Simple Data Binding	332
Complex Data Binding	333
Data Sources for Data Binding	333
Using the Data Form Wizard	334
Using the Windows Forms Class Viewer	338
Using the Windows Forms ActiveX Control Importer	338
Summary	340
Solutions Fast Track	340
Frequently Asked Questions	344

Chapter 8 Windows Forms Components and Controls 347

Adding Items to a Combo Box at Design-Time

- 
1. Select the **ComboBox** control on the form.
 2. If necessary, use the **View** menu to open the **Properties** window.
 3. In the **Properties** window, click the **Items** property, then click the ellipsis.
 4. In **String Collection Editor**, type the first item, then press **Enter**.
 5. Type the next items, pressing **Enter** after each item.
 6. Click **OK**.

Introduction	348
Built-In Controls	348
Label Control	351
LinkLabel Control	354
TextBox Control	357
Button Control	361
CheckBox Control	364
RadioButton Control	365
RichTextBox Control	367
TreeView Control	369
ListBox Control	371
CheckedListBox Control	374
ListView Control	376
ComboBox Control	381
DomainUpDown Control	384
NumericUpDown Control	386
PictureBox Control	388
TrackBar Control	389

DateTimePicker Control	391
Panel Control	394
GroupBox Control	396
TabControl Control	397
Creating Custom Windows Components	399
Exercise 8.1: Creating a Custom Windows Component	399
Creating Custom Windows Controls	403
Exercise 8.2: Creating a Custom Windows Control	404
Summary	407
Solutions Fast Track	407
Frequently Asked Questions	408

Chapter 9 Using ADO.NET 409

XML Documents

XML documents are the heart of the XML standard. An XML document has at least one element that is delimited with one start tag and one end tag. XML documents are similar to HTML, except that the tags are made up by the author.

Introduction	410
Overview of XML	411
XML Documents	411
XSL	411
XDR	412
XPath	412
Understanding ADO.NET Architecture	412
Differences between ADO and ADO.NET	414
XML Support	414
ADO.NET Configuration	415
Remoting in ADO.NET	415
Maintaining State	415
Using the XML Schema Definition Tool	416
Connected Layer	417
Data Providers	418
Connection Strings	418
Exercise 9.1 Creating a Connection String	419
Command Objects	421
DataReader	425
DataSet	426

Disconnected Layer	427
Using DataSet	428
Relational Schema	428
Collection of Tables	430
Data States	431
Populating with the DataSet Command	432
Populating with XML	433
Populating Programmatically	434
Using the SQL Server Data Provider	435
TDS	436
Exercise 9.2 Using TypedDataSet	437
Remoting	439
Data Controls	440
DataGrid	440
Exercise 9.3 Using TypedDataSet and DataRelation	441
DataList	446
Repeater	450
Summary	454
Solutions Fast Track	454
Frequently Asked Questions	457
Chapter 10 Developing Web Applications	459
Introduction	460
Web Forms	461
A Simple Web Form	462
Exercise 10.1 Creating a Simple Web Form	462
How Web Forms Differ from Windows Forms	464
Why Web Forms Are Better Than Classic ASP	465
Adding Controls to Web Forms	467
Exercise 10.2 Adding Web Controls to a Web Form	468
Code Behind	473

NOTE

Web form controls not only detect browsers such as Internet Explorer and Netscape, but they also detect devices such as Palm Pilots and cell phones and generate appropriate HTML accordingly.

How Web Form Controls Differ from Windows Form Controls	476
ASP.NET Server Controls	476
Intrinsic Controls	476
Bound Controls	478
Exercise 10.3 Using the DataGrid Control	478
Exercise 10.4 Customizing DataGrid Control	482
Custom Controls	487
Validation Controls	488
Exercise 10.5 Using the Validation Controls	489
Creating Custom Web Form Controls	492
Exercise 10.6 A Simple Custom Control	493
Exercise 10.7 Creating a Composite Custom Control	497
Web Services	504
How Web Services Work	505
Developing Web Services	505
Exercise 10.8 Developing Web Services	507
Web Service Utilities	509
Service Description Language	509
Discovery	510
Proxy Class	510
Consuming Web Services from Web Forms	511
Exercise 10.9 Consuming Web Services from Web Forms	511
Using Windows Forms in Distributed Applications	513
Exercise 10.10 Consuming Web Services from Windows Forms	514
Exercise 10.11 Developing a Sample Application	516
Summary	519
Solutions Fast Track	519
Frequently Asked Questions	521

Chapter 11 Optimizing, Debugging, and Testing	523
Introduction	524
Debugging Concepts	524
Debug Menu	528
Watches	529
Breakpoints	531
Exceptions Window	532
Command Window	534
Conditional Compilation	536
Trace	538
Assertions	540
Code Optimization	541
Finalization	542
Transitions	542
Parameter Passing Methods	542
Strings	543
Garbage Collection	544
Compiler Options	544
Optimization Options	544
Output File Options	544
.NET Assembly Options	545
Preprocessor Options	546
Miscellaneous Options	546
Testing Phases and Strategies	546
Unit Testing	547
Integration Testing	547
Beta Testing	547
Regression Testing	548
Stress Testing	548
Monitoring Performance	548
Summary	550
Solutions Fast Track	551
Frequently Asked Questions	552

What Are Watches?

Watches provide us with a mechanism where we can interact with the actual data that is stored in our programs at runtime. They allow us to see the values of variables and the values of properties on objects. In addition to being able to view these values, you can also assign new values.

	Chapter 12 Security	553
	Introduction	554
	Security Concepts	555
	Permissions	555
	Principal	556
	Authentication	557
	Authorization	557
	Security Policy	558
	Type Safety	558
	Code Access Security	558
	.NET Code Access Security Model	559
	Stack Walking	559
	Code Identity	561
	Code Groups	562
	Declarative and Imperative Security	564
	Requesting Permissions	565
	Demanding Permissions	570
	Overriding Security Checks	572
	Custom Permissions	576
	Role-Based Security	578
	Principals	578
	WindowsPrincipal	579
	GenericPrincipal	580
	Manipulating Identity	581
	Role-Based Security Checks	583
	Security Policies	585
	Creating a New Permission Set	588
	Modifying the Code Group Structure	593
	Remoting Security	600
	Cryptography	600
	Security Tools	603
	Summary	606
	Solutions Fast Track	607
	Frequently Asked Questions	611
Within the .NET Framework, Three Namespaces Involve Cryptography		
1. <i>System.Security.Cryptography</i>	The most important one; resembles the CryptoAPI functionalities.	
2. <i>System.Security.Cryptography.X509Certificates</i>	Relates only to the X509 v3 certificate used with Authenticode.	
3. <i>System.Security.Cryptography.Xml</i>	For exclusive use within the .NET Framework security system.	

WARNING

You should under no circumstance edit the Security.config and Enterprise.config files directly. It is very easy to compromise the integrity of these files. Always use the Code Access Security Policy utility (caspol.exe) or the .NET Configuration tool; these will guard the integrity of the files and will also make a backup copy of the last saved version.

Chapter 13 Application Deployment 615

Introduction	616
Packaging Code	617
Configuring the .NET Framework	622
Creating Configuration Files	622
Machine/Administrator Configuration Files	623
Application Configuration Files	625
Security Configuration Files	626
Deploying the Application	629
Common Language Runtime	629
Windows Installer	630
CAB Files	631
Internet Explorer 5.5	632
Resource Files	633
Deploying Controls	637
Summary	639
Solutions Fast Track	640
Frequently Asked Questions	642

Chapter 14 Upgrading Visual Basic Applications to .NET 647

Introduction	648
Considerations Before Upgrading	648
Early Binding of Variables	649
Avoiding Null Propagation	650
Using ADO	651
Using Date Data Type	652
Using Constants	652
Considering Architecture Before Migration	653
Intranet/Internet Applications	653
Internet Information Server (IIS)	
Applications	654
DHTML Applications	655
ActiveX Documents	655
Client/Server and Multi-Tier Applications	655
Single-Tier Applications	656
Data Access Applications	656

Data Types	657
Variants	657
Integers	658
Dates	658
Boolean	659
Arrays	659
Fixed-Length Strings	660
Windows API Data Types	661
Converting VB Forms to Windows Forms	662
Control Anchoring	664
Keyword Changes	665
Goto	666
GoSub	666
Option Base	666
AND/OR	666
Lset	666
VarPtr	667
StrPtr	667
Def	667
Programming Differences	668
Method Implementation	668
Optional Parameters	668
Static Modifier	669
Return Statement	669
Procedure Calls	670
External Procedure Declaration	671
Passing Parameters	672
ParamArray	672
Overloading	674
References to Unmanaged Libraries	677
Metadata	679
Runtime Callable Wrapper	681
COM Callable Wrapper	682
Properties	684
Working with Property Procedures	684
Control Property Name Changes	685
Default Property	687

Avoiding Null Propagation

Null propagation means that if *Null* is used in an expression, the resulting expression is always *Null*. In previous versions of Visual Basic, the *Null* value disseminated throughout the expression.

Null Usage	690
Understanding Error Handling	690
Exercise 14.1: Using Error Handling	692
Data Access Changes in Visual Basic .NET	693
Dataset and Recordset	694
Application Interoperability	694
Cursor Location	695
Disconnected Access	695
Data Navigation	695
Lock Implementation	696
Upgrading Interfaces	696
Upgrading Interfaces from Visual Basic 6.0	699
Using the Upgrade Tool	703
Exercise 14.2 Using the Upgrade Wizard	703
Summary	708
Solutions Fast Track	709
Frequently Asked Questions	712
Index	713