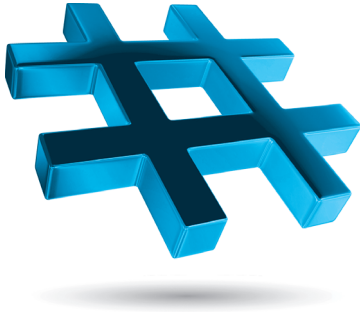

Preface

Welcome to the world of desktop, mobile and web app development with Microsoft's® Visual C#® programming language. *Visual C# How to Program, 6/e* is based on C# 6[1] and related Microsoft software technologies. You'll be using the .NET platform and the Visual Studio® Integrated Development Environment on which you'll conveniently write, test and debug your applications and run them on Windows® devices. The Windows operating system runs on desktop and notebook computers, mobile phones and tablets, game systems and a great variety of devices associated with the emerging "Internet of Things."

We believe that this book and its supplements for students and instructors will give you an informative, engaging, challenging and entertaining introduction to Visual C#. The book presents leading-edge computing technologies in a friendly manner appropriate for introductory college course sequences, based on the curriculum recommendations of two key professional organizations—the ACM and the IEEE.[2]

You'll study four of today's most popular programming paradigms:

- object-oriented programming,
- structured programming,
- generic programming and
- functional programming (new in this edition).

If you haven't already done so, please read the back cover and check out the additional reviewer comments on the inside back cover and the facing page—these capture the essence of the book concisely. In this Preface we provide more detail for students, instructors and professionals.

At the heart of the book is the Deitel signature *live-code approach*—rather than using code snippets, we generally present concepts in the context of complete working programs followed by sample executions. We include a broad range of example programs and exercises selected from computer science, business, education, social issues, personal utilities, sports, mathematics, puzzles, simulation, game playing, graphics, multimedia and many other areas. We also provide abundant tables, line drawings and UML diagrams for a more visual learning experience.

1. At the time of this writing, Microsoft has not yet released the official C# 6 Specification. To view an unofficial copy, visit `https://github.com/ljw1004/csharpspec/blob/gh-pages/README.md`
2. These recommendations include *Computer Science Curricula 2013 Curriculum Guidelines for Undergraduate Degree Programs in Computer Science*, December 20, 2013, The Joint Task Force on Computing Curricula, Association for Computing Machinery (ACM), IEEE Computer Society.

Read the Before You Begin section after this Preface for instructions on setting up your computer to run the hundreds of code examples and to enable you to develop your own C# apps. The source code for all of the book's examples is available at

```
http://www.deitel.com/books/VCSharpHTP6
```

and

```
http://www.pearsonhighered.com/deitel
```

Use the source code we provide to compile and run each program as you study it—this will help you master Visual C# and related Microsoft technologies faster and at a deeper level. Most of the book's examples work in Visual Studio on Windows 7, 8 or 10 (there is no 9). The code examples for the online presentation of the Universal Windows Platform (UWP) and XAML specifically require Windows 10.

Contacting the Authors

As you read the book, if you have a question, we're easy to reach at

```
deitel@deitel.com
```

We'll respond promptly.

Join the Deitel & Associates, Inc. Social Media Communities

For book updates, visit

```
http://www.deitel.com/books/VCSharpHTP6
```

subscribe to the *Deitel® Buzz Online* newsletter

```
http://www.deitel.com/newsletter/subscribe.html
```

and join the Deitel social media communities on

- Facebook®—`http://facebook.com/DeitelFan`
- LinkedIn®—`http://linkedin.com/company/deitel-&-associates`
- YouTube®—`http://youtube.com/DeitelTV`
- Twitter®—`http://twitter.com/Deitel`
- Instagram®—`http://instagram.com/DeitelFan`
- Google+™—`http://google.com/+DeitelFan`

Object-Oriented Programming with an Early Objects Approach

The book introduces the basic concepts and terminology of object-oriented programming in Chapter 1. In Chapter 2, you'll *visually* manipulate objects, such as labels and images. In Chapter 3, Introduction to C# App Programming, you'll write Visual C# program code that manipulates preexisting objects. You'll develop your first customized classes and objects in Chapter 4. Presenting objects and classes early gets you "thinking about objects" immediately and mastering these concepts more thoroughly.

Our early objects presentation continues in Chapters 5–9 with a variety of straightforward case studies. In Chapters 10–12, we take a deeper look at classes and objects,

present inheritance, interfaces and polymorphism, then use those concepts throughout the remainder of the book.

New C# 6 Features

6 We introduce key new C# 6 language features throughout the book (Fig. 1)—each defining occurrence is marked with a "6" margin icon as shown next to this paragraph.

C# 6 new language feature	First introduced in
`string` interpolation	Section 3.5
expression bodied methods and `get` accessors	Section 7.15
auto-implemented property initializers	Section 8.6.1
getter-only auto-implemented properties	Section 8.6.1
`nameof` operator	Section 10.5.1
null-conditional operator (`?.`)	Section 13.9.1
`when` clause for exception filtering	Section 13.10
`using static` directive	Section 21.3.1
null conditional operator (`?[]`)	Section 21.6
collection initializers for any collection with an `Add` extension method	Section 21.7
index initializers	Section 21.7

Fig. 1 | C# 6 new language features.

Interesting, Entertaining and Challenging Exercises

The book contains hundreds of exercises to practice the skills you learn. Extensive self-review exercises and answers are included for self-study. Also, each chapter concludes with a substantial set of exercises, which generally include

- simple recall of important terminology and concepts,
- identifying the errors in code samples,
- writing individual program statements,
- writing methods to perform specific tasks,
- writing C# classes,
- writing complete programs and
- implementing major projects.

Figure 2 lists only a sample of the book's hundreds of exercises, including selections from our *Making-a-Difference* exercises set, which encourage you to use computers and the Internet to research and work on significant social problems. We hope you'll approach these exercises in the context of your own values, politics and beliefs. The solutions to most of the book's exercises are available only to college instructors who have adopted the book for their courses. See "Instructor Supplements" later in this Preface.

A sampling of the book's exercises

Carbon Footprint Calculator
Body-Mass-Index Calculator
Attributes of Hybrid Vehicles
Gender Neutrality
Notepad GUI
Calendar and Appointments GUI
Calculator GUI
Alarm Clock GUI
Radio GUI
Displaying Shapes
Odd or Even?
Is a Number a Multiple of Another?
Separating an Integer's Digits
Table of Squares and Cubes
Account Class
Invoice Class
Employee Class
Date Class
Removing Duplicated Code
Target-Heart-Rate Calculator
Computerizing Health Records
Credit-Limit Calculator
Sales-Commission Calculator
Salary Calculator
Find the Two Largest Numbers
Dangling-else Problem
Palindromes
Decimal Equivalent of a Binary Number
Sides of a Right Triangle
Factorials
Infinite Series: Mathematical Constant *e*
World Population Growth
Enforcing Privacy with Cryptography
Bar Chart Display
Prime Numbers
Calculating Sales
Car-Pool Savings Calculator
Gas Mileage Calculator
Calculating the Value of π
Pythagorean Triples
Global Warming Facts Quiz
Tax Plan Alternative: The "FairTax"
Rounding to a Specific Decimal Place
Hypotenuse of a Right Triangle
Displaying a Square of Any Character
Separating Digits
Temperature Conversions
Perfect Numbers
Prime Numbers
Reversing Digits
Letter Grades to a Four-Point Scale
Coin Tossing
Guess-the-Number Game
Distance Between Two Points
Craps Game with Betting
Towers of Hanoi
Computer-Assisted Instruction
Sales Commissions
Duplicate Elimination
Dice Game of Craps
Airline Reservations System
Knight's Tour Chess Puzzle
Eight Queens Chess Puzzle
Sieve of Eratosthenes
Tortoise and the Hare
Card Shuffling and Dealing
Building Your Own Computer (Virtual Machine)
Polling
Querying an Array of Invoice Objects
Duplicate Word Removal
Rectangle Class
Savings-Account Class
Set of Integers
RationalNumber Class
HugeInteger Class
Tic-Tac-Toe
ComplexNumber Class
Shape Inheritance Hierarchy
Payroll System
Accounts Payable System
Polymorphic Banking Program
CarbonFootprint Interface: Polymorphism
Temperature Conversions
Painter
Guess the Number Game
Ecofont
Typing Tutor
Restaurant Bill Calculator
Story Writer
Pig Latin
Cooking with Healthier Ingredients
Spam Elimination
SMS Language
File of Student Grades
Telephone-Number Words
Student Poll
Phishing Scanner
Bucket Sort
Palindromes
Evaluating Expressions with a Stack
Building Your Own Compiler
Generic Linear Search
SortedDictionary of Colors
Prime Factorization
Bucket Sort with LinkedList<int>
Sieve of Eratosthenes with BitArray
Credit-Inquiry Program
Rolling a Die 60,000,000 Times
Baseball Database App
Parsing with LINQ to XML
I/O-Bound vs. Compute-Bound Apps
Recursive Fibonacci

Fig. 2 | A sampling of the book's exercises.

A Tour of the Book

This section discusses the book's modular organization to help instructors plan their syllabi.

Introduction to Computing, Visual C# and Visual Studio 2015 Community Edition
The chapters in this module of the book

- Chapter 1, Introduction to Computers, the Internet and Visual C#
- Chapter 2, Introduction to Visual Studio and Visual Programming

introduce hardware and software fundamentals, Microsoft's .NET platform and Visual Programming. The vast majority of the book's examples will run on Windows 7, 8 and 10 using the *Visual Studio 2015 Community* edition with which we test-drive a fun **Painter** app in Section 1.12. Chapter 1's introduction to object-oriented programming defines key terminology and discusses important concepts on which the rest of the book depends.

Introduction to C# Fundamentals
The chapters in this module of the book

- Chapter 3, Introduction to C# App Programming
- Chapter 4, Introduction to Classes, Objects, Methods and Strings
- Chapter 5, Algorithm Development and Control Statements: Part 1
- Chapter 6, Control Statements: Part 2
- Chapter 7, Methods: A Deeper Look
- Chapter 8, Arrays; Introduction to Exception Handling

present rich coverage of C# programming fundamentals (data types, operators, control statements, methods and arrays) and introduce object-oriented programming through a series of case studies. These chapters should be covered in order. Chapters 5 and 6 present a friendly treatment of control statements and problem solving. Chapters 7 and 8 present rich treatments of methods and arrays, respectively. Chapter 8 briefly introduces exception handling with an example that demonstrates attempting to access an element outside an array's bounds.

Object-Oriented Programming: A Deeper Look
The chapters in this module of the book

- Chapter 9, Introduction to LINQ and the `List` Collection
- Chapter 10, Classes and Objects: A Deeper Look
- Chapter 11, Object-Oriented Programming: Inheritance
- Chapter 12, OOP: Polymorphism and Interfaces
- Chapter 13, Exception Handling: A Deeper Look

provide a deeper look at object-oriented programming, including classes, objects, inheritance, polymorphism, interfaces and exception handling. An optional online two-chapter case study on designing and implementing the object-oriented software for a simple ATM is described later in this preface.

Chapter 9 introduces Microsoft's Language Integrated Query (LINQ) technology, which provides a uniform syntax for manipulating data from various data sources, such as

arrays, collections and, as you'll see in later chapters, databases and XML. Chapter 9 is intentionally simple and brief to encourage instructors to begin covering LINQ technology early. Section 9.4 introduces the `List` collection, which we use in Chapter 12. Later in the book, we take a deeper look at LINQ, using LINQ to Entities (for querying databases) and LINQ to XML. Chapter 9's LINQ coverage can be deferred if you're in a course which either skips LINQ or defers coverage until later in the book—it's required for one example in Chapter 17 (Fig. 17.6) and many of the later chapters starting with Chapter 22, Databases and LINQ.

Windows `Forms` Graphical User Interfaces (GUIs)
The chapters in this module of the book

- Chapter 14, Graphical User Interfaces with Windows Forms: Part 1
- Chapter 15, Graphical User Interfaces with Windows Forms: Part 2

present a detailed introduction to building GUIs using Windows `Forms`—instructors teaching Visual C# still largely prefer Windows `Forms` for their classes. Many of the examples in GUI Chapters 14–15 can be presented after Chapter 4. We also use Windows `Forms` GUIs in several other print and online chapters.

There are two other GUI technologies in Windows—Windows Presentation Foundation (WPF) and Universal Windows Platform (UWP). We provide optional online treatments of both.[3]

Strings and Files
The chapters in this module of the book

- Chapter 16, Strings and Characters: A Deeper Look
- Chapter 17, Files and Streams

present `string` processing and file processing, respectively. We introduce `string`s beginning in Chapter 4 and use them throughout the book. Chapter 16 investigates `string`s in more detail. Most of Chapter 16's examples can be presented at any point after Chapter 4. Chapter 17 introduces text-file processing and object-serialization for inputting and outputting entire objects. Chapter 17 requires Windows `Forms` concepts presented in Chapter 14.

Searching, Sorting and Generic Data Structures
The chapters in this module of the book:

- Chapter 18, Searching and Sorting
- Chapter 19, Custom Linked Data Structures
- Chapter 20, Generics
- Chapter 21, Generic Collections; Functional Programming with LINQ/PLINQ

introduce searching, sorting and data structures. Most C# programmers should use .NET's *built-in* searching, sorting and generic collections (prepackaged data structures) capabilities, which are discussed in Chapter 21. For instructors who wish to present how to implement customized searching, sorting and data structures capabilities, we provide Chapters 18–20, which require the concepts presented in Chapters 3–8 and 10–13. Chapter 18 presents sev-

3. As of Summer 2016, Windows Forms, WPF and UWP apps all can be posted for distribution via the Windows Store. See `http://bit.ly/DesktopToUWP` for more information.

eral searching and sorting algorithms and uses Big O notation to help you compare how hard each algorithm works to do its job—the code examples use especially visual outputs to show how the algorithms work. In Chapter 19, we show how to implement your own custom data structures, including lists, stacks, queues and binary trees. The data structures in Chapter 19 store references to `objects`. Chapter 20 introduces C# generics and demonstrates how to create type-safe generic methods and a type-safe generic stack data structure.

Functional Programming with LINQ, PLINQ, Lambdas, Delegates and Immutability
In addition to generic collections, Chapter 21 now introduces functional programming, showing how to use it with LINQ to Objects to write code more concisely and with fewer bugs than programs written using previous techniques. In Section 21.12, with one additional method call, we'll demonstrate how PLINQ (Parallel LINQ) can improve LINQ to Objects performance substantially on multicore systems. The chapter's exercises also ask you to reimplement earlier examples using functional-programming techniques.

Database with LINQ to Entities and SQL Server
The chapter in this module

- Chapter 22, Databases and LINQ

presents a novice-friendly introduction to database programming with the ADO.NET Entity Framework, LINQ to Entities and Microsoft's free version of SQL Server that's installed with the Visual Studio 2015 Community edition. The chapter's examples require C#, object-oriented programming and Windows `Forms` concepts presented in Chapters 3–14. Several online chapters require the techniques presented in this chapter.

Asynchronous Programming
The chapter in this module

- Chapter 23, Asynchronous Programming with `async` and `await`

shows how to take advantage of multicore architectures by writing applications that can process tasks asynchronously, which can improve app performance and GUI responsiveness in apps with long-running or compute-intensive tasks. The `async` modifier and `await` operator greatly simplify asynchronous programming, reduce errors and enable your apps to take advantage of the processing power in today's multicore computers, smartphones and tablets. In this edition, we added a case study that uses the Task Parallel Library (TPL) `async` and `await` in a GUI app—we keep a progress bar moving along in the GUI thread in parallel with a lengthy, compute-intensive calculation in another thread.

A Tour of the Online Content

The printed book contains the core content (Chapters 1–23) for introductory and intermediate course sequences. Several optional online topics for advanced courses and professionals are available on the book's password-protected Companion Website

```
http://www.pearsonhighered.com/deitel
```

New copies of this book come with a Companion Website access code that's located on the book's inside front cover. If the access code is already visible or there isn't an access code, you purchased a used book or an edition that does not come with an access code. In this case, you can purchase access directly from the Companion Website. Figure 3 lists the online topics, and Figure 4 lists a sample of the associated exercises.

Online topics
Web App Development with ASP.NET
XML and LINQ to XML
Universal Windows Platform (UWP) GUI, Graphics, Multimedia and XAML
REST Web Services
Cloud Computing with Microsoft Azure™
Using the Visual Studio Debugger
(Optional) Windows Presentation Foundation (WPF) GUI, Graphics, Multimedia and XAML
(Optional) ATM Case Study, Part 1: Object-Oriented Design with the UML
(Optional) ATM Case Study, Part 2: Implementing an OO Design in C#

Fig. 3 | Online topics on the *Visual C# How to Program, 6/e* Companion Website.

A sampling of the online chapters' exercises		
Guestbook App	College Loan Payoff Calculator App	Favorite Flickr Searches App
Web-Based Address Book	Car Payment Calculator App	Flag Quiz App
Enhanced Painter App	Mileage Calculator App	Phone Book App with Data Binding
PhotoViewer App	Body Mass Index Calculator App	Enhanced UsingGradients App
Data Bindings to LINQ queries	Target-Heart-Rate Calculator App	Enhanced DrawStars app
Snake PolyLine App	Phone-Book Web Service	Image Reflector App
Drawing App		Accessibility: Speech-Controlled Drawing App
Enhanced Tip Calculator App		
Mortgage Calculator App		

Fig. 4 | A sampling of the online chapters' exercises.

Web App Development with ASP.NET
Microsoft's .NET server-side technology, ASP.NET, enables you to create robust, scalable web-based apps. You'll build several apps, including a web-based guestbook that uses ASP.NET and the ADO .NET Entity Framework to store data in a database and display data in a web page.

Extensible Markup Language (XML)
The Extensible Markup Language (XML) is pervasive in the software-development industry, e-business and throughout the .NET platform. It's used in most of this book's online topics. XML is required to understand XAML—a Microsoft XML vocabulary that's used to describe graphical user interfaces, graphics and multimedia for Universal Windows Platform (UWP) GUI, graphics and multimedia apps, Windows 10 Mobile apps and Windows Presentation Foundation (WPF) apps. We present XML fundamentals, then discuss LINQ to XML, which allows you to query XML content using LINQ syntax.

Universal Windows Platform (UWP) for Desktop and Mobile Apps
The Universal Windows Platform (UWP) is designed to provide a common platform and user experience across all Windows devices, including personal computers, smartphones,

tablets, Xbox and even Microsoft's new HoloLens virtual reality and augmented reality holographic headset—all using nearly identical code. We present GUI, graphics and multimedia apps, and demonstrate them on both personal computers and the smartphone emulator that comes with Visual Studio 2015 Community edition.

REST Web Services
Web services enable you to package app functionality in a manner that turns the web into a library of *reusable* services. We include a case study on building a math question generator web service that's called by a math tutor app.

Building a Microsoft Azure™ Cloud Computing App
Microsoft Azure's web services enable you to develop, manage and distribute your apps in "the cloud." We'll demonstrate how to use Azure web services to store an app's data online.

Windows Presentation Foundation (WPF) GUI, Graphics and Multimedia
Windows Presentation Foundation (WPF)—created after Windows `Forms` and before UWP—is another Microsoft technology for building robust GUI, graphics and multimedia desktop apps. WPF provides you with complete control over all aspects of a GUI's look-and-feel and includes multimedia capabilities that are not available in Windows `Forms`. We discuss WPF in the context of a painting app, a text editor, a color chooser, a book-cover viewer, a television video player, various animations, and speech synthesis and recognition apps.

We're moving away from WPF in favor of UWP for creating apps that can run on desktop, mobile and other Windows devices. For this reason, the WPF introduction is provided *as is* from the previous edition—we will no longer evolve this material.

Optional Case Study: Using the UML to Develop an Object-Oriented Design and C# Implementation of an ATM (Automated Teller Machine)
The UML™ (Unified Modeling Language™) is the industry-standard graphical language for visually modeling object-oriented systems. We introduce the UML in the early chapters and provide an optional online object-oriented design case study in which we use the UML to design and implement the software for a simple ATM. We analyze a typical *requirements document* that specifies the details of the system to be built. We determine the *classes* needed to implement that system, the *attributes* the classes need to have, the *behaviors* the classes' methods need to exhibit and we specify how the classes must *interact* with one another to meet the system requirements. From the design, we produce a complete working C# implementation. Students often report a "light bulb moment"—the case study helps them "tie it all together" and truly understand object orientation.

Teaching Approach

Visual C# How to Program, 6/e contains a rich collection of examples. We concentrate on building well-engineered software and stress program clarity.

Live-Code Approach. The book is loaded with "live-code" examples—most new concepts are presented in the context of complete working Visual C# apps, followed by one or more executions showing program inputs and outputs. In the few cases where we show a code snippet, to ensure correctness first we tested it in a complete working program then copied the code from the program and pasted it into the book.

Syntax Shading. For readability, we syntax shade the code, similar to the way Visual Studio colors the code. Our syntax-shading conventions are:

```
comments appear like this
keywords appear like this
constants and literal values appear like this
all other code appears in black
```

Code Highlighting. We emphasize key code segments by placing them in gray rectangles.

Using Fonts for Emphasis. We place the key terms and the index's page reference for each defining occurrence in colored **bold** text for easy reference. We show on-screen components in the **bold Helvetica** font (for example, the **File** menu) and Visual C# program text in the `Lucida` font (for example, `int count = 5;`). We use *italics* for emphasis.

Objectives. The chapter objectives preview the topics covered in the chapter.

Programming Tips. We include programming tips to help you focus on important aspects of program development. These tips and practices represent the best we've gleaned from a combined nine decades of programming and teaching experience.

Good Programming Practices

The Good Programming Practices *call attention to techniques that will help you produce programs that are clearer, more understandable and more maintainable.*

Common Programming Errors

Pointing out these Common Programming Errors *reduces the likelihood that you'll make them.*

Error-Prevention Tips

These tips contain suggestions for exposing and removing bugs from your programs; many of the tips describe aspects of Visual C# that prevent bugs from getting into programs.

Performance Tips

These tips highlight opportunities for making your programs run faster or minimizing the amount of memory that they occupy.

Portability Tips

These tips help you write code that will run on a variety of platforms.

Software Engineering Observations

The Software Engineering Observations *highlight architectural and design issues that affect the construction of software systems, especially large-scale systems.*

Look-and-Feel Observation 3.1

These observations help you design attractive, user-friendly graphical user interfaces that conform to industry norms.

Summary Bullets. We present a detailed bullet-list summary of each chapter.

Terminology. We include an alphabetized list of the important terms defined in each chapter.

Index. We've included an extensive index for reference. Defining occurrences of key terms in the index are highlighted with a colored **bold** page number. The printed book index covers only the printed material. The online index includes the content from the printed book and the online content.

Obtaining the Software Used in *Visual C# How to Program, 6/e*

We wrote the code examples in *Visual C# How to Program, 6/e* using Microsoft's free Visual Studio 2015 Community edition. See the Before You Begin section that follows this preface for download and installation instructions.

Instructor Supplements

The following supplements are available to ***qualified instructors only*** through Pearson Education's online Instructor Resource Center (`www.pearsonhighered.com/irc`):

- ***Solutions Manual*** contains solutions to *most* of the end-of-chapter exercises. We've included many *Making-a-Difference* exercises, most with solutions. **Please do not write to us requesting access to the Pearson Instructor's Resource Center. Access is restricted to college instructors who have adopted the book for their courses. Instructors can obtain access through their Pearson representatives.** If you're not a registered faculty member, contact your Pearson representative or visit `http://www.pearsonhighered.com/educator/replocator/`. Exercise Solutions are *not* provided for "project" exercises. Check out our Programming Projects Resource Center for lots of additional exercise and project possibilities:

```
http://www.deitel.com/ProgrammingProjects
```

- ***Test Item File*** of multiple-choice questions (approximately two per top-level book section)
- ***Customizable PowerPoint® slides*** containing all the code and figures in the text, plus bulleted items that summarize the key points in the text.

Microsoft DreamSpark™

Microsoft provides many of its professional developer tools to students for free via a program called DreamSpark (`http://www.dreamspark.com`). See the website for details on verifying your student status so you take advantage of this program. To compile, test, debug and run this book's examples, you need only Windows 10 and the free Visual Studio 2015 Community edition. With the exception of the online UWP examples, the book's examples also will compile and run on Windows 7 and higher.

Acknowledgments

We'd like to thank Barbara Deitel of Deitel & Associates, Inc. She painstakingly researched the new capabilities of Visual C#, Visual Studio, .NET and other key technologies. We'd also like to acknowledge Frank McCown, Ph.D., Associate Professor of Computer Science, Harding University for his suggestion to include an example that used a `ProgressBar` with `async` and `await` in Chapter 23—so we ported to C# a similar example from our textbook *Java How to Program, 10/e*.

We're fortunate to have worked with the dedicated team of publishing professionals at Pearson Higher Education. We appreciate the guidance, wisdom, energy and mentorship of Tracy Johnson, Executive Editor, Computer Science. Kristy Alaura did an extraordinary job recruiting the book's reviewers and managing the review process. Bob Engelhardt did a wonderful job bringing the book to publication.

Reviewers

The book was scrutinized by academics teaching C# courses and industry C# experts. They provided countless suggestions for improving the presentation. Any remaining flaws in the book are our own.

Sixth Edition Reviewers: Qian Chen (Department of Engineering Technology: Computer Science Technology Program, Savannah State University), Octavio Hernandez (Microsoft Certified Solutions Developer, Principal Software Engineer at Advanced Bionics), José Antonio González Seco (Parliament of Andalusia, Spain), Bradley Sward (College of Dupage) and Lucian Wischik (Microsoft Visual C# Team).

Fifth Edition Post-Publication Reviewers: To help us prepare to write 6/e, the following academics reviewed 5/e and provided many helpful suggestions: Qian Chen (Savannah State University), Hongmei Chi (Florida A&M University), Kui Du (University of Massachusetts, Boston), James Leasure (Cuyahoga Community College West), Victor Miller (Ramapo College), Gary Savard (Champlain College) and Mohammad Yusuf (New Hampshire Technical Institute).

Other recent edition reviewers: Douglas B. Bock (MCSD.NET, Southern Illinois University Edwardsville), Dan Crevier (Microsoft), Shay Friedman (Microsoft Visual C# MVP), Amit K. Ghosh (University of Texas at El Paso), Marcelo Guerra Hahn (Microsoft), Kim Hamilton (Software Design Engineer at Microsoft and co-author of *Learning UML 2.0*), Huanhui Hu (Microsoft Corporation), Stephen Hustedde (South Mountain College), James Edward Keysor (Florida Institute of Technology), Narges Kasiri (Oklahoma State University), Helena Kotas (Microsoft), Charles Liu (University of Texas at San Antonio), Chris Lovett (Software Architect at Microsoft), Bashar Lulu (INETA Country Leader, Arabian Gulf), John McIlhinney (Spatial Intelligence; Microsoft MVP Visual Developer, Visual Basic), Ged Mead (Microsoft Visual Basic MVP, DevCity.net), Anand Mukundan (Architect, Polaris Software Lab Ltd.), Dr. Hamid R. Nemati (The University of North Carolina at Greensboro), Timothy Ng (Microsoft), Akira Onishi (Microsoft), Jeffrey P. Scott (Blackhawk Technical College), Joe Stagner (Senior Program Manager, Developer Tools & Platforms, Microsoft), Erick Thompson (Microsoft), Jesús Ubaldo Quevedo-Torrero (University of Wisconsin–Parkside, Department of Computer Science), Shawn Weisfeld (Microsoft MVP and President and Founder of UserGroup.tv) and Zijiang Yang (Western Michigan University).

As you read the book, we'd sincerely appreciate your comments, criticisms, corrections and suggestions for improving the text. Please address all correspondence to:

```
deitel@deitel.com
```

We'll respond promptly. It was fun writing *Visual C# How to Program, 6/e*—we hope you enjoy reading it!

Paul Deitel
Harvey Deitel

About the Authors

Paul Deitel, CEO and Chief Technical Officer of Deitel & Associates, Inc., has over 35 years of experience in computing. He is a graduate of MIT, where he studied Information Technology. Through Deitel & Associates, Inc., he has delivered hundreds of programming courses worldwide to clients, including Cisco, IBM, Boeing, Siemens, Sun Microsystems (now Oracle), Dell, Fidelity, NASA at the Kennedy Space Center, the National Severe Storm Laboratory, NOAA (National Oceanic and Atmospheric Administration), White Sands Missile Range, Rogue Wave Software, SunGard, Nortel Networks, Puma, iRobot, Invensys and many more. He and his co-author, Dr. Harvey Deitel, are the world's best-selling programming-language textbook/professional book/video authors.

Paul was named as a Microsoft® Most Valuable Professional (MVP) for C# in 2012–2014. According to Microsoft, "the Microsoft MVP Award is an annual award that recognizes exceptional technology community leaders worldwide who actively share their high quality, real-world expertise with users and Microsoft." He also holds the Java Certified Programmer and Java Certified Developer designations and is an Oracle Java Champion.

C# MVP 2012–2014

Dr. Harvey Deitel, Chairman and Chief Strategy Officer of Deitel & Associates, Inc., has over 55 years of experience in the computer field. Dr. Deitel earned B.S. and M.S. degrees in Electrical Engineering from MIT and a Ph.D. in Mathematics from Boston University—he studied computing in each of these programs before they spun off Computer Science programs. He has extensive college teaching experience, including earning tenure and serving as the Chairman of the Computer Science Department at Boston College before founding Deitel & Associates, Inc., in 1991 with his son, Paul. The Deitels' publications have earned international recognition, with translations published in Japanese, German, Russian, Spanish, French, Polish, Italian, Simplified Chinese, Traditional Chinese, Korean, Portuguese, Greek, Urdu and Turkish. Dr. Deitel has delivered hundreds of programming courses to academic, corporate, government and military clients.

About Deitel & Associates, Inc.

Deitel & Associates, Inc., founded by Paul Deitel and Harvey Deitel, is an internationally recognized authoring and corporate training organization, specializing in computer programming languages, object technology, Internet and web software technology, and Android and iOS app development. The company's clients include academic institutions, many of the world's largest corporations, government agencies and branches of the military. The company offers instructor-led training courses delivered at client sites worldwide on major programming languages and platforms, including Visual C#®, C++, C, Java™, Android app development, iOS app development, Swift™, Visual Basic® and Internet and web programming.

Through its 40-year publishing partnership with Prentice Hall/Pearson, Deitel & Associates, Inc., creates leading-edge programming college textbooks, professional books, *LiveLessons* video products, e-books and REVEL™ interactive multimedia courses with integrated labs and assessment (`http://revel.pearson.com`). Deitel & Associates, Inc. and the authors can be reached at:

```
deitel@deitel.com
```

To learn more about Deitel's corporate training curriculum, visit

```
http://www.deitel.com/training
```

To request a proposal for worldwide on-site, instructor-led training at your organization, send an e-mail to `deitel@deitel.com`.

Individuals wishing to purchase Deitel books can do so via

```
http://bit.ly/DeitelOnAmazon
```

Individuals wishing to purchase Deitel *LiveLessons* video training can do so at:

```
http://bit.ly/DeitelOnInformit
```

Deitel books and *LiveLessons* videos are generally available electronically to Safari Books Online subscribers at:

```
http://SafariBooksOnline.com
```

You can get a free 10-day Safari Books Online trial at:

```
https://www.safaribooksonline.com/register/
```

Bulk orders by corporations, the government, the military and academic institutions should be placed directly with Pearson. For more information, visit

```
http://www.informit.com/store/sales.aspx
```