Effective Software Development Series

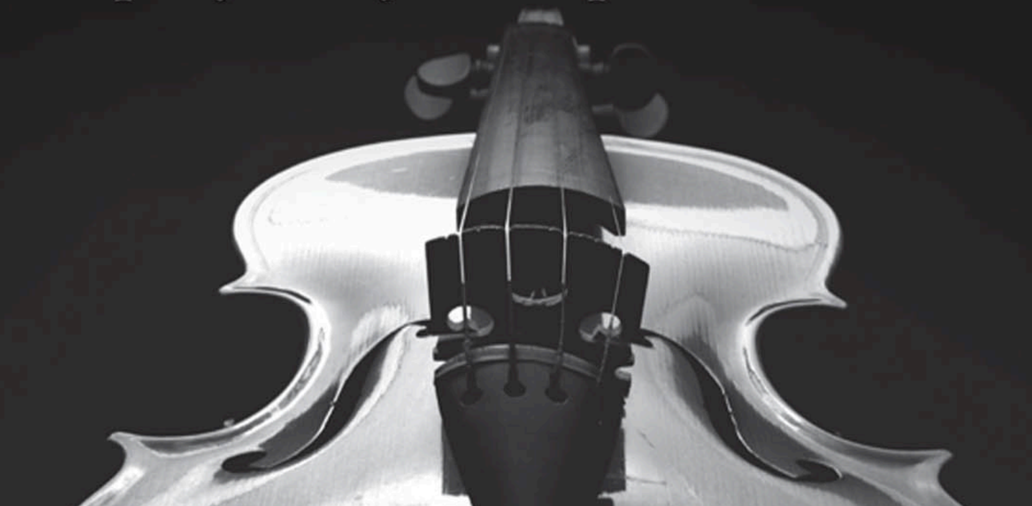
Scott Meyers, Consulting Editor

Effective C#

Third Edition

covers C# 6.0

50 Specific Ways to Improve Your C#



Content Update Program

FREE…See Details Inside

Bill Wagner

Effective C#

Third Edition

This book is part of InformIT's exciting new Content Update Program, which provides automatic content updates for major technology improvements!

- ► As significant updates are made to C#, this book will be updated or new sections will be added to match the updates to the technologies.
- ► The updates will be delivered to you via a free Web Edition of this book, which can be accessed with any Internet connection.
- ► This means your purchase is protected from immediately outdated information!

For more information on InformIT's Content Update program, see the inside back cover or go to **informit.com/cup**

Content Update Program

*If you have additional questions, please email our Customer Service department at **informit@custhelp.com**.*

Effective C# (Covers C# 6.0),: 50 Specific Ways to Improve Your C#

Table of Contents

Cover

Title Page

Copyright Page

Contents

Introduction

Chapter 1 C# Language Idioms

- Item 1: Prefer Implicitly Typed Local Variables
- Item 2: Prefer readonly to const
- Item 3: Prefer the is or as Operators to Casts
- Item 4: Replace string.Format() with Interpolated Strings
- Item 5: Prefer FormattableString for Culture-Specific Strings
- Item 6: Avoid String-ly Typed APIs
- Item 7: Express Callbacks with Delegates
- Item 8: Use the Null Conditional Operator for Event Invocations
- Item 9: Minimize Boxing and Unboxing
- Item 10: Use the new Modifier Only to React to Base Class Updates

Chapter 2 .NET Resource Management

- Item 11: Understand .NET Resource Management
- Item 12: Prefer Member Initializers to Assignment Statements
- Item 13: Use Proper Initialization for Static Class Members

Table of Contents

Item 14: Minimize Duplicate Initialization Logic

Item 15: Avoid Creating Unnecessary Objects

Item 16: Never Call Virtual Functions in Constructors

Item 17: Implement the Standard Dispose Pattern

Chapter 3 Working with Generics

Item 18: Always Define Constraints That Are Minimal and Sufficient

Item 19: Specialize Generic Algorithms Using Runtime Type Checking

Item 20: Implement Ordering Relations with IComparable<T> and IComparer<T>

Item 21: Always Create Generic Classes That Support Disposable Type Parameters

Item 22: Support Generic Covariance and Contravariance

Item 23: Use Delegates to Define Method Constraints on Type Parameters

Item 24: Do Not Create Generic Specialization on Base Classes or Interfaces

Item 25: Prefer Generic Methods Unless Type Parameters Are Instance Fields

Item 26: Implement Classic Interfaces in Addition to Generic Interfaces

Item 27: Augment Minimal Interface Contracts with Extension Methods

Item 28: Consider Enhancing Constructed Types with Extension Methods

Chapter 4 Working with LINQ

Item 29: Prefer Iterator Methods to Returning Collections

Pearson

Table of Contents

Item 30: Prefer Query Syntax to Loops

Item 31: Create Composable APIs for Sequences

Item 32: Decouple Iterations from Actions, Predicates, and Functions

Item 33: Generate Sequence Items as Requested

Item 34: Loosen Coupling by Using Function Parameters

Item 35: Never Overload Extension Methods

Item 36: Understand How Query Expressions Map to Method Calls

Item 37: Prefer Lazy Evaluation to Eager Evaluation in Queries

Item 38: Prefer Lambda Expressions to Methods

Item 39: Avoid Throwing Exceptions in Functions and Actions

Item 40: Distinguish Early from Deferred Execution

Item 41: Avoid Capturing Expensive Resources

Item 42: Distinguish between IEnumerable and IQueryable Data Sources

Item 43: Use Single() and First() to Enforce Semantic Expectations on Queries

Item 44: Avoid Modifying Bound Variables

Chapter 5 Exception Practices

Item 45: Use Exceptions to Report Method Contract Failures

Item 46: Utilize using and try/finally for Resource Cleanup

Item 47: Create Complete Application-Specific Exception Classes

Item 48: Prefer the Strong Exception Guarantee

Item 49: Prefer Exception Filters to catch and re-throw

Item 50: Leverage Side Effects in Exception Filters

Table of Contents

Index