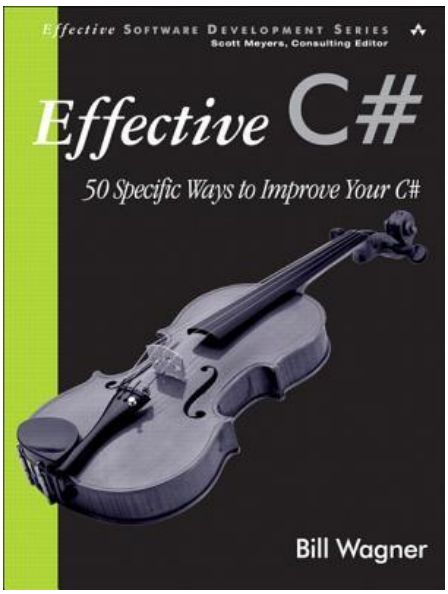


# Effective C#: 50 Specific Ways to Improve Your C#

*Bill Wagner*

# Effective C#: 50 Specific Ways to Improve Your C#

*Bill Wagner*

**Effective C#: 50 Specific Ways to Improve Your C#** Bill Wagner
"This book really demonstrates Bill's strengths as a writer and programmer. In a very short amount of time, he is able to present an issue, fix it and conclude it; each chapter is tight, succinct, and to the point." --Josh Holmes, Independent Contractor "The book provides a good introduction to the C# language elements from a pragmatic point of view, identifying best practices along the way, and following a clear and logical progression from the basic syntax to creating components to improving your code writing skills. Since each topic is covered in short entries, it is very easy to read and you'll quickly realize the benefits of the book." --Tomas Restrepo, Microsoft MVP "The book covers the basics well, especially with respect to the decisions needed when deriving classes from System.Object. It is easy to read with examples that are clear, concise and solid. I think it will bring good value to most readers." --Rob Steel, Central Region Integration COE & Lead Architect, Microsoft "*Effective C#* provides the C# developer with the tools they need to rapidly grow their experience in Visual C# 2003 while also providing insight into the many improvements to the language that will be hitting a desktop near you in the form of Visual C# 2005." --Doug Holland, Precision Objects "Part of the point of the .NET Framework--and the C# Language, in particular--is to let the developer focus solving customer problems and deliver product, rather than spending hours (or even weeks) writing plumbing code. Bill Wagner's *Effective C#*, not only shows you what's going on behind the scenes, but shows you how to take advantage of particular C# code constructs. Written in a dispassionate style that focuses on the facts--and just the facts--of writing effective C# code, Wagner's book drills down into practices that will let you write C# applications and components that are easier to maintain as well as faster to run. I'm recommending *Effective C#* to all students of my .NET BootCamp and other C#-related courses." --Richard Hale Shaw, www.RichardHaleShawGroup.com C#'s resemblances to C++, Java, and C make it easier to learn, but there's a downside: C# programmers often continue to use older techniques when far better alternatives are available. In Effective C#, respected .NET expert Bill Wagner identifies fifty ways you can start leveraging the full power of C# in order to write faster, more efficient, and more reliable software. *Effective C#* follows the format that made *Effective C++* (Addison-Wesley, 1998) and *Effective Java* (Addison-Wesley, 2001) indispensable to hundreds of thousands of developers: clear, practical explanations, expert tips, and plenty of realistic code examples. Drawing on his unsurpassed C# experience, Wagner addresses everything from value types to assemblies, exceptions to reflection. Along the way, he shows exactly how to avoid dozens of common C# performance and reliability pitfalls. You'll learn how to:
Use both types of C# constants for efficiency and maintainability, see item 2 Use immutable data types to eliminate unnecessary error checking, see item 7 Avoid the C# function that'll practically always get you in trouble, see item 10 Minimize garbage collection, boxing, and unboxing, see items 16 and 17 Take full advantage of interfaces and delegates, see items 19 though 22 Create CLS compliant assemblies that use noncompliant C# language features, see item 30 Improve reliability and maintainability by creating small, cohesive assemblies, see item 32 Leverage the full power of .NET's runtime diagnostics, see item 36 Know when--and when not--to use reflection, see items 42 and 43 Preview the major enhancements in C# 2.0, see item 49 You're already a successful C# programmer--this book can help you become an outstanding one. Bill Wagner is co-founder of and .NET consultant for SRT Solutions. A nationally recognized independent expert on .NET, he has been a regular contributor to ASP.NET Pro Magazine, Visual Studio Magazine, and the .NET Insight newsletter. In addition to being a Microsoft Regional Director, he is also active in the Southeast Michigan .NET User Group and the Ann Arbor Computing Society. He is author of *The C# Core Language Little Black Book* (The Coriolis Group, 2002).

## Effective C#: 50 Specific Ways to Improve Your C# Details

| | | |
|---|---|---|
| Date | : | Published December 13th 2004 by Addison-Wesley Professional (first published December 3rd 2004) |
| ISBN | : | 9780321245663 |
| Author | : | Bill Wagner |
| Format | : | Paperback 307 pages |
| Genre | : | Computer Science, Programming, Science, Technology, Software, Technical |

# From Reader Review Effective C#: 50 Specific Ways to Improve Your C# for online ebook

### Arun Mahendrakar says

Although I knew some of the tricks/tips mentioned in this book, majority of them were new and intriguing.

Simple and easy to understand examples even on advanced concepts make this an asset for any C# developer. For any issues on C# (up until version 4), I'll use this book as a constant reference guide.

---

### Aman Aujla says

want to read it

---

### Gishu Pillai says

Just a bunch of gotchas and light bulb moments strung together..

Get it if you want to kill some time.

---

### Andrew Dalgleish says

This is one of the best .NET programming books every written. It's a topical treatment of do's and dont's to follow when writing a .NET program in C#, and highlights some of the strengths of the language and underlying .NET Framework.

UPDATE:

There is a new second edition of this book that covers .NET 4, with a blue bar on the cover rather than a green one. Make sure you get the later edition. There is also a companion book called 'More Effective C#' by the same author (Wagner) - both are pretty much required reading for any .NET programmer so far as I'm concerned.

---

### Jonathan says

Some good ideas, but a ton of crap on LINQ and meta-programming that I hope I never have to use

---

### Ien Cheng says

Very good. C# is a beautiful language. Pity it's so tied up with Microsoft and the Microsoft stack. Thankfully there is Mono, the open-source alternative to all that.

---

### gimpf says

While the advice is generally sound, I would not recommend this book.

Disappointments require expectations, and I admit, what I describe as this book's failures are partially my own. But before I belabor my own mental state, I try to summarize the good parts:

If you are not (yet) an experienced programmer, but know the basics, and have some C# experience, but not much, this book might help you avoid some gotchas which would otherwise have taken painful experience to learn about.

If you are an experienced C# programmer and know "CLR via C#", and you have as bad a memory like I do, you will find one or two points you might already have forgotten.

I personally (re-)learned: a trick using side effects in exception filters, interpolated strings degrade to 'string' instead of 'formattable' left of a '.' operator (wtf), interpolated strings box all their arguments, and all lambda expressions within a method are translated to a single capturing closure (really big WTF?!). This list is complete.

Doesn't sound too bad; I'd say I'm still the target audience.

But. (Now comes the part where I become overly or even unfairly negative).

I would have to lie saying that the two to six hours required to read this book are well spent. If you are a fast or voracious reader, go for it, but keep in mind that the good nuggets are well hidden. Why? The big part is bad editing, a good deal are bad examples, and then there's still insufficient content.

Readership: What's the target audience? It contains pages over pages for obvious basics, but then mentions subtle perils in nothing more than a remark of half a paragraph, without further explanation. So we get almost seven pages arguing for using `var`, but only "the logic in the compiler that determines whether it should create a `string` or a `FormattableString` would create a `string` instead of a `FormattableString` when the result is the left side of the "." operator", plus some pseudo-explanation that doesn't explain the reasoning behind the language team's choice at all.

Structure: This information about the extension method is important; in an in-house tutorial I've seen code -- and implicitly acknowledged it as correct -- that is actually wrong because of this behavior. And I'd almost have missed this rule, because it was hidden within two pages of boring public API doc copy. Why not a rule "Do not use FormattableString as `this` parameter for an extension method". Of course, the number of pages for a single guidelines is a free parameter, but some choices can be counterproductive.

Editing: The example sentence above (about the formattable string, again) provides a glimpse. It is

notoriously difficult to be both precise and compact and have a fluent sentence structure, but I would have hoped that Addison-Wesley would provide support to make it work on their "Effective ... Series". Alas, no. We software engineers deserve good writing no less than detective story readers, or so I hope. A publishing house should take care of that.

Code Examples: This book will get an errata addendum for its code samples as long as all of them taken together. I exaggerate, of course, but less than you might hope. Sometimes code is repeated, interjected too early, or does not show the differences that are explained in the text. I soon started ignoring the code examples **completely**. _If_ you write a book that will also be approached by novices (or tired professionals at night for their self-education), you better don't confuse them with incorrectly copy-pasted code.

Code Examples, part 2: Code formatting in books is always a challenge, but this book's result is a new low (for me). Code Conventions introduced early are not used. The choice of coding style is practically unseen, because it would not work with Visual Studio's IntelliSense at all, for example placing the '.' of Linq methods at the end of a line, and starting with the plain method name in the next.

Obsolete interjections: If a new edition has been created not only to incorporate new language versions, but also to adapt to "the new world" where a reference to C++ is obsolete, why assume the reader knows about COM and details of its reference counting?

Examples: A good example (easy to understand, illustrating the point, and seen in real-world code) of showing the difference (and only that) between returning error values vs. throwing exceptions would be `dict[key]` vs. `dict.TryGetValue()`, and explain the relationship between post-conditions and return values. A bad example would be comparing apples with oranges, like in `File.Exists()` vs. `File.Open()`.

Missing stuff: For some reason, Effective C++ explained C++ specific issues so well that it also made you a better software developer in any language. Here, the author obviously has all the knowledge in his brain, but for some reason, most expositions w.r.t. design and code style choices are not enlightening for the reader -- ok, well, me. Learning how to avoid the pitfalls of C#'s and .NET's history is all it is. This is written on the cover of the book, so it's not really a valid complaint. But on the other hand, other books manage, and this one isn't a GC implementation deep dive, so ... whatever.

I have more complaints (free-form jumps between "good code" and "bad code", for instance...), but I think I already overdid it with this review.

I have become a grumpy old man, complaining about everything, and talking about the good old times where "Exceptional C++" was ruling the world, followed by "Practical Common Lisp" and "CLR via C#". Maybe the books back then were just as bad as the books are now, but the hell I don't think I would have grokked "strong exception safety" from that kind of exposition that Effective C# tried to throw at me.

I hope that publisher and author re-evaluate the effort that good writing needs, bite the bullet, and provide a reworked edition.

But for this edition, as it is, verdict: Avoid.

---

## David says

**Buggy code samples**

There were a surprisingly large number of errors in the Kindle edition. The second edition was more accurate and I don't remember any errors in the code samples.

---

## Boris says

???????? ?????! ??????? ? ????? ????? ?? ????. ?????? ?????????? ?????????? ????????? ???? "??????????
??????????", ?? ????? ??? ?? ????? ??? ????? ???????. ? ????????? ??????? ???????? ????? ??????? ? ???????
???????????. ? ?? ????????????? ?? ???????, ?? ???? ?? ???? ???????? ?? ?????????????, ??? ??? "?????? ?
????????".
? ??????? ?????????? ????? ?????????? ???????????? ???????????? + ???????????? ????????, ??????? ??????? ?
?????. ??????? ????? ???????? ????????. ?? ??? ?? ??? - ??? ??? ?? ??????? ????.?? ???? ??? ??, ?? ? ?????
??? ?? ???????? ?? ?????? ??????? - ?? ??? ??? ?? ??????? ?? ???????, ?? ????????, ?? ??????. ? ?????
??????? ?? ???????? ???????, ??????? ???? ? ?????? ? ?????.
???? ??? ??????? ???? ????????? ????? C#, ?? ?????? ?? ?????? ????????????? ????????? ? ???????,
????????????? ???????? ????? ? ????? - ??? ????? ?? ??? ?????? ????????!
P.S.: ???? ?????? ??????? ? ??? ????? ???????, ?? ???? ????? ?????????? ?????? ? ?????????.

---

## Ivo Stoykov says

Effective C# is a must read book for C# developers. Wide range of problems are discussed in this book. They are exposed in a compact shape with samples and solutions.

---

## Kurt Kozina says

Like Code Complete 2 this is a great book for improving your c# coding standards. This book deals with specific c# programming issues and concepts pointing out the good and bad very clearly

---

## Aleksei says

Not as eye-opening as Meyers' Effective C++, but quite useful nevertheless.

---