

Effective C++

par Matthieu Brucher (<http://matthieu-brucher.developpez.com/>) (Blog)

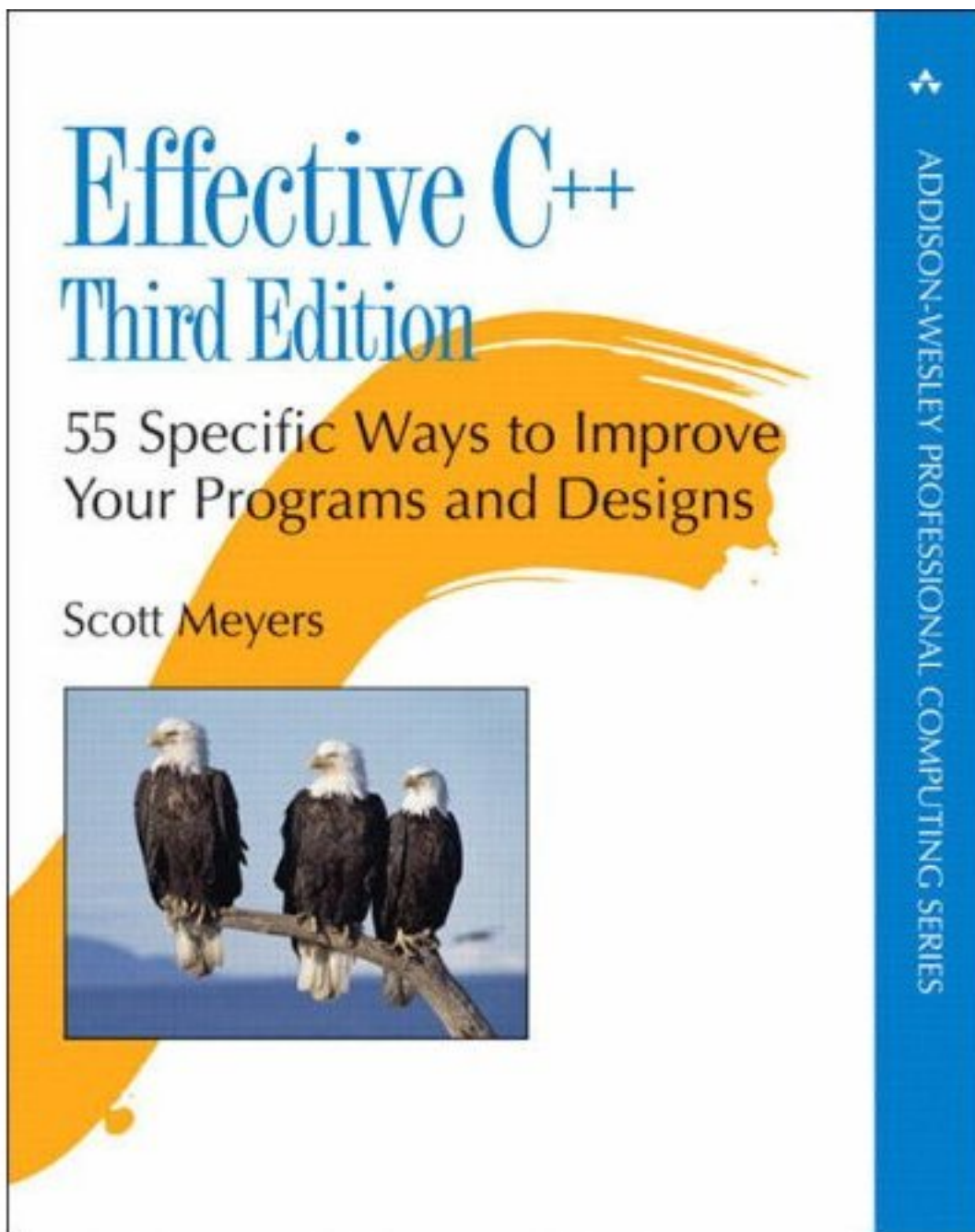
Date de publication : 10/05/2006

Dernière mise à jour : 27/03/2008

Critique d'Effective C++ de *Scott Meyers*

- I - Description de l'éditeur
- II - Table des matières
- III - Critique : Indispensable
- IV - Liens annexes

I - Description de l'éditeur



Une nouvelle édition d'un best-seller. C'est le livre à la pointe sur le marché, le second sur la programmation, un livre que tous devraient posséder.

Scott Meyers est l'un des experts de premier plan sur le développement logiciel en C++. C'est un excellent promoteur de ses livres.

Plus de 50% de matériel neuf et des mises à jour complètes de matériel vrai et essayé afin de l'amener au standard 2005.

Il n'y a pas de meilleur moyen d'amener le programmeur C++ à la pointe des pratiques que ce livre. C'est un classique - le numéro 1 dans son genre.

II - Table des matières

- **GENERIC PROGRAMMING AND THE C++ STANDARD LIBRARY.**
 - Item 1: Iterators.
 - Item 2: Case-Insensitive Strings-Part 1.
 - Item 3: Case-Insensitive Strings-Part 2.
 - Item 4: Maximally Reusable Generic Containers-Part 1.
 - Item 5: Maximally Reusable Generic Containers-Part 2.
 - Item 6: Temporary Objects.
 - Item 7: Using the Standard Library (or, Temporaries Revisited).
- **EXCEPTION-SAFETY ISSUES AND TECHNIQUES.**
 - Item 8: Writing Exception-Safe Code-Part 1.
 - Item 9: Writing Exception-Safe Code-Part 2.
 - Item 10: Writing Exception-Safe Code-Part 3.
 - Item 11: Writing Exception-Safe Code-Part 4.
 - Item 12: Writing Exception-Safe Code-Part 5.
 - Item 13: Writing Exception-Safe Code-Part 6.
 - Item 14: Writing Exception-Safe Code-Part 7.
 - Item 15: Writing Exception-Safe Code-Part 8.
 - Item 16: Writing Exception-Safe Code-Part 9.
 - Item 17: Writing Exception-Safe Code-Part 10.
 - Item 18: Code Complexity-Part 1.
 - Item 19: Code Complexity-Part 2.
- **CLASS DESIGN AND INHERITANCE.**
 - Item 20: Class Mechanics.
 - Item 21: Overriding Virtual Functions.
 - Item 22: Class Relationships-Part 1.
 - Item 23: Class Relationships-Part 2.
 - Item 24: Uses and Abuses of Inheritance.
 - Item 25: Object-Oriented Programming.
- **COMPILER FIREWALLS AND THE PIMPL IDIOM.**
 - Item 26: Minimizing Compile-time Dependencies-Part 1.
 - Item 27: Minimizing Compile-time Dependencies-Part 2.
 - Item 28: Minimizing Compile-time Dependencies-Part 3.
 - Item 29: Compilation Firewalls.
 - Item 30: The "Fast Pimpl" Idiom.
- **NAME LOOKUP, NAMESPACES, AND THE INTERFACE PRINCIPLE.**
 - Item 31: Name Lookup and the Interface Principle-Part 1.
 - Item 32: Name Lookup and the Interface Principle-Part 2.
 - Item 33: Name Lookup and the Interface Principle-Part 3.
 - Item 34: Name Lookup and the Interface Principle-Part 4.
- **MEMORY MANAGEMENT.**

- Item 35: Memory Management-Part 1.
- Item 36: Memory Management-Part 2.
- Item 37: auto_ptr.
- TRAPS, PITFALLS, AND ANIT-IDIOMS.
 - Item 38: Object Identity.
 - Item 39: Automatic Conversions.
 - Item 40: Object Lifetimes-Part 1.
 - Item 41: Object Lifetimes-Part 2.
- MISCELLANEOUS TOPICS.
 - Item 42: Variable Initialization - Or Is It?
 - Item 43: Const-Correctness.
 - Item 44: Casts.
 - Item 45: bool.
 - Item 46: Forwarding Functions.
 - Item 47: Control Flow.

III - Critique : Indispensable

Cette critique s'applique à la troisième édition de ce livre (**Le C++ efficace**), en anglais. Le sous titre est 55 Specific Ways to Improve Your Programs and Designs. 5 éléments ont été apparemment ajoutés.

En fait, le livre a été complètement remanié, par exemple la première partie n'est plus consacrée à la migration C vers C++, mais à l'adaptation au C++ à travers les mots-clés const, les initialisations, ... La difficulté est croissante, les premiers éléments étant très simples à appréhender et à mettre en oeuvre - logique, c'est l'adaptation au C++ - puis on avance vers des terrains de plus en plus minés - qu'est-ce que le compilateur crée comme fonction qu'on ne veut pas forcément, les opérateurs d'assignations, ... -. La partie sur la gestion de la mémoire gagne en complexité et s'applique à toutes les ressources, non plus seulement la mémoire. Les parties dédiées au design et déclaration des classes, leurs implémentations ainsi que l'héritage sont elles aussi pertinentes. Certains éléments des anciennes éditions ont été déplacés dans d'autres sections, d'autres éléments ont été incorporés judicieusement. Une partie dédiée à la programmation générique a été introduite, on va même jusqu'à la méta-programmation. La section consacrée à new et delete est aussi nouvelle et on parle de leur surcharge, ce qui est très intéressant.

Ce livre est maintenant vraiment dédié au C++ et non à la jonction C/C++ comme par le passé. Il est même orienté vers le futur, plusieurs allusions au nouveau standard en discussion sont faites dans tout le livre avec les nouvelles classes apparues dans TR1:: et Boost. Ce livre, dans cette dernière édition, est de nouveau à la pointe, comme il l'a été lors de la première édition, après avoir marqué le pas face aux livres plus récents et plus "C++". Encore un livre à avoir dans sa bibliothèque !

IV - Liens annexes

 ***Critique sur la page de livres C++***

 ***Achat sur Amazon.fr***

 ***Lien vers le site de l'éditeur***

