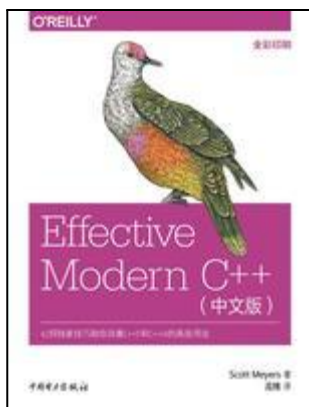
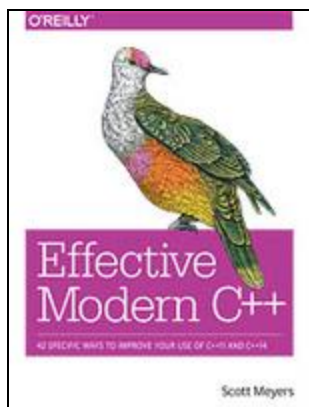


# 《Effective Modern C++ ——改善 C++11 和 C++14 的42个具体做法》



|      |                                                                                                                                                                                                                                                                 |
|------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 分类   | <a href="#">IT 类 / 软件开发 / 编程语言 / C++</a>                                                                                                                                                                                                                        |
| 作者   | (美) <a href="#">Scott Meyers</a> / 斯科特·梅耶<br>                                                                                                                                 |
| 英文书名 | 《Effective Modern C++——42 Specific Ways to Improve Your Use of C++11 and C++14》                                                                                                                                                                                 |
| 出版年份 | 2014(原著)<br>2018(简体中译本)                                                                                                                                                                                                                                         |
| 相关链接 | <a href="#">本书官网</a>                                                                                                                                                                                                                                            |
| 相关书籍 | 俺的网盘上分享过他的如下几本书： <ul style="list-style-type: none"><li>• 《<a href="#">Effective C++——改善程序与设计的55个具体做法</a>》</li><li>• 《<a href="#">More Effective C++——35个改善编程与设计的有效方法</a>》</li><li>• 《<a href="#">Effective Modern C++——改善 C++11 和 C++14 的42个具体做法</a>》</li></ul> |

## 简介

作者是知名的 C++ 写手，出版过好几本畅销的 C++ 读物。

他写的书，最有名的就是这个【Effective 系列】。此书是《[Effective C++](#)》和《[More Effective C++](#)》的升级版，针对新的 C++ 标准(C++11 和 C++14)。此书秉承 Effective 系列的一贯风格，列出 C++ 新标准的若干个注意事项。

## 英文目录

### 1 Deducing Types

- Item 1: Understand template type deduction
- Item 2: Understand auto type deduction
- Item 3: Understand decltype
- Item 4: Know how to view deduced types

### 2 Auto

- Item 5: Prefer auto to explicit type declarations
- Item 6: Use the explicitly typed initializer idiom when auto deduces undesired types

### 3 Moving to Modern C++

- Item 7: Distinguish between () and {} when creating objects
- Item 8: Prefer nullptr to 0 and NULL
- Item 9: Prefer alias declarations to typedefs
- Item 10: Prefer scoped enums to unscoped enums
- Item 11: Prefer deleted functions to private undefined ones
- Item 12: Declare overriding functions override
- Item 13: Prefer const\_iterators to iterators
- Item 14: Declare functions noexcept if they won't emit exceptions
- Item 15: Use constexpr whenever possible
- Item 16: Make const member functions thread safe
- Item 17: Understand special member function generation

### 4 Smart Pointers

- Item 18: Use std::unique\_ptr for exclusive-ownership resource management
- Item 19: Use std::shared\_ptr for shared-ownership resource management
- Item 20: Use std::weak\_ptr for std::shared\_ptr-like pointers that can dangle
- Item 21: Prefer std::make\_unique and std::make\_shared to direct use of new
- Item 22: When using the Pimpl Idiom, define special member functions in the implementation file

### 5 Rvalue References, Move Semantics, and Perfect Forwarding

- Item 23: Understand std::move and std::forward
- Item 24: Distinguish universal references from rvalue references
- Item 25: Use std::move on rvalue references, std::forward on universal references
- Item 26: Avoid overloading on universal references

- Item 27: Familiarize yourself with alternatives to overloading on universal references
- Item 28: Understand reference collapsing
- Item 29: Assume that move operations are not present, not cheap, and not used
- Item 30: Familiarize yourself with perfect forwarding failure cases

## **6 Lambda Expressions**

- Item 31: Avoid default capture modes
- Item 32: Use init capture to move objects into closures
- Item 33: Use decltype on auto&& parameters to std::forward them
- Item 34: Prefer lambdas to std::bind

## **7 The Concurrency API**

- Item 35: Prefer task-based programming to thread-based
- Item 36: Specify std::launch::async if asynchronicity is essential
- Item 37: Make std::threads unjoinable on all paths
- Item 38: Be aware of varying thread handle destructor behavior
- Item 39: Consider void futures for one-shot event communication
- Item 40: Use std::atomic for concurrency, volatile for special memory

## **8 Tweaks**

- Item 41: Consider pass by value for copyable parameters that are cheap to move and always copied
- Item 42: Consider emplacement instead of insertion

[【编程随想】收藏的电子书清单](#)

[【编程随想】的博客](#)