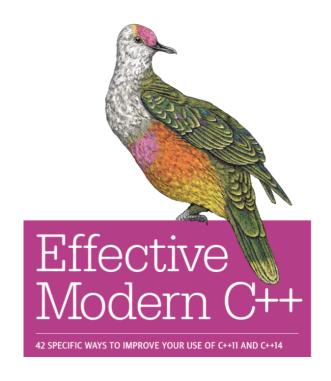
# 《Effective Modern C++》翻译



**Scott Meyers** 

backers 0 sponsors 0

- ! 2017.10开始更新
- ! 标注"已修订"的章节表示已经没有大致的错误 ! 我没有版权,我没有版权,我没有版权
- ! 本书要求读者具有C++基础
- ! 未翻译的条款名称现在直译, 翻译时可能适当修改
- !PDf版本下载,仅供翻译参考

## 景目

- 1. 类型推导
  - 1. Item 1:理解模板类型推导 已修订
  - 2. Item 2:理解auto类型推导
  - 3. Item 3:理解decltype
  - 4. Item 4:学会查看类型推导结果
- 2. auto
  - 1. Item 5:优先考虑auto而非显式类型声明
  - 2. Item 6:auto推导若非己愿,使用显式类型初始化惯用法
- 3. 移步现代C++
  - 1. <u>Item 7:区别使用()和{}创建对象</u>

- 2. Item 8:优先考虑nullptr而非0和NULL
- 3. Item 9:优先考虑别名声明而非typedefs
- 4. Item 10:优先考虑限域枚举而非未限域枚举 已修订
- 5. Item 11:优先考虑使用deleted函数而非使用未定义的私有声明
- 6. Item 12:使用override声明重载函数
- 7. <u>Item 13:优先考虑const iterator而非iterator</u>
- 8. Item 14:如果函数不抛出异常请使用noexcept
- 9. Item 15:尽可能的使用constexpr
- 10. Item 16:让const成员函数线程安全 由 @windski贡献
- 11. Item 17:理解特殊成员函数函数的生成

#### 4. 智能指针

- 1. <u>Item 18:对于独占资源使用std::unique\_ptr</u> @wendajiang更新完成
- 2. Item 19:对于共享资源使用std::shared\_ptr 已修订
- 3. <u>Item 20:像std::shared\_ptr一样使用std::weak\_ptr可能造成dangle</u> 更新完成
- 4. Item 21:优先考虑使用std::make unique和std::make shared而非new 由 @pusidun贡献
- 5. <u>Item 22:当使用Pimpl惯用法</u>,请在实现文件中定义特殊成员函数 由 @BlurryLight贡献
- 5. 右值引用,移动语意,完美转发
  - 1. Item 23:理解std::move和std::forward 由 @BlurryLight贡献
  - 2. Item 24:区别通用引用和右值引用 由 @BlurryLight贡献
  - 3. <u>Item 25:对于右值引用使用std::move、对于通用引用使用std::forward</u>由 @wendajiang贡献
  - 4. <u>Item 26:避免重载通用引用</u>由 @wendajiang贡献
  - 5. Item 27:熟悉重载通用引用的替代品由 @wendajiang贡献
  - 6. <u>Item 28:理解引用折叠</u>由 @wendajiang贡献
  - 7. Item 29:认识移动操作的缺点 由 @wendajiang贡献
  - 8. Item 30:熟悉完美转发失败的情况由 @wendajiang贡献

#### 6. Lambda表达式

- 1. Item 31:避免使用默认捕获模式 由 @LucienXian贡献
- 2. Item 32:使用初始化捕获来移动对象到闭包中 由 @LucienXian贡献
- 3. <u>Item 33:对于std::forward的auto&&形参使用decltype</u> 由 @LucienXian贡献
- 4. <u>Item 34:优先考虑lambda表达式而非std::bind</u> 由 @LucienXian贡献

#### 7. 并发API

- 1. Item 35:优先考虑基于任务的编程而非基于线程的编程 由 @wendajiang贡献
- 2. <u>Item 36:如果有异步的必要请指定std::launch::threads</u> 由 @wendajiang贡献
- 3. <u>Item 37:从各个方面使得std::threads unjoinable</u> 由 @wendajiang贡献
- 4. Item 38:关注不同线程句柄析构行为 由 @wendajiang贡献
- 5. <u>Item 39:考虑对于单次事件通信使用void</u> 由 @wendajiang贡献
- 6. <u>Item 40:对于并发使用std::atomic、volatile用于特殊内存区</u> 由 @wendajiang贡献

#### 8. 微调

- 1. Item 41:对于那些可移动总是被拷贝的形参使用传值方式 由 @wendajiang贡献
- 2. Item 42:考虑就地创建而非插入 由 @wendajiang贡献

### 贡献者

感谢所有参与翻译/勘误/建议的贡献者们~









### 赞助翻译

