Effective C++ 阅读笔记

1. **让自己习惯C++**
2. **视C++为一个语言联邦**

视C++为一个由四个相关语言组成的联邦而非单一语言。在组成它的每个次语言中，各种守则和通例都倾向简单、直观易懂并且容易记住。当从一个次语言移往另一个次语言，守则可能改变。

四个次语言：C语言、面向对象的C++、C++模板、STL。

1. **尽量以const, enum, inline替代#define**
2. **尽可能使用const**
3. **确定对象被使用前已被初始化**
4. **构造、析构、赋值运算**
5. **了解C++默默编写并调用了哪些函数**
6. **若不想使用编译器自动生成的函数，就改明确拒绝**
7. **为多态基类声明virtual析构函数**
8. **别让异常逃离析构函数**
9. **绝不在构造和析构过程中调用virtual函数**
10. **令operate=返回一个reference to \* this**
11. **在operate=中处理“自我赋值”**
12. **复制对象时勿忘其每一个成分**
13. **资源管理**
14. **以对象管理资源**
15. **在资源管理类中小心copying行为**
16. **在资源管理类中提供对原始资源的访问**
17. **成对使用new和delete时要采取相同形式**
18. **以独立语句将newed对象置于智能指针**
19. **设计与声明**
20. **让接口容易被正确使用，不易被误用**
21. **设计class犹如设计type**
22. **宁以pass-by-reference-to-const替换pass-by-value**
23. **必须返回对象时，别妄想返回其reference**
24. **将成员变量声明为private**
25. **宁以non-member, non-friend替换member函数**
26. **若所有参数皆需类型转换，请为此采用non-member函数**
27. **考虑写出一个不抛异常的swap函数**
28. **实现**
29. **尽可能延后变量定义式的出现时间**
30. **尽量少做转型动作**
31. **避免返回handles指向对象内部成分**
32. **为“异常安全”而努力是值得的**
33. **透彻了解inlining的里里外外**
34. **将文件之间的编译依存关系将至最低**
35. **继承和面向对象设计**
36. **确定你的public继承塑模出is-a关系**
37. **避免遮掩继承而来的名称**
38. **区分接口继承和实现继承**
39. **考虑virtual函数以外的其他选择**
40. **绝不重新定义继承而来的non-virtual函数**
41. **绝不重新定义继承而来的缺省参数值**
42. **通过重合塑模出has-a或“根据某物实现出”**
43. **明智而审慎地提出private继承**
44. **明智而审慎地提出多重继承**
45. **模板和泛型编程**
46. **了解隐式接口和编译期多态**
47. **了解typename的双重意义**
48. **学习处理模板化基类内的名称**
49. **将于参数无关的代码抽离templates**
50. **运用成员函数模板接收所有兼容类型**
51. **需要类型转换时请为模板定义非成员函数**
52. **请使用traits classes表现类型信息**
53. **认识template元编程**
54. **定制new和delete**
55. **了解new-handler的行为**
56. **了解new和delete的合理替换时机**
57. **编写new和delete时需固守常规**
58. **写了placement new也要写placement delete**
59. **杂项讨论**
60. **不要轻忽编译器的警告**
61. **让自己熟悉包括TR1在内的标准程序库**
62. **让自己熟悉Boost**