

Contents

| | |
|--|-------------|
| Preface | xv |
| Acknowledgments | xvii |
| Introduction | 1 |
| Chapter 1: Accustoming Yourself to C++ | 11 |
| Item 1: View C++ as a federation of languages. | 11 |
| Item 2: Prefer consts, enums, and inlines to #defines. | 13 |
| Item 3: Use const whenever possible. | 17 |
| Item 4: Make sure that objects are initialized before they're used. | 26 |
| Chapter 2: Constructors, Destructors, and Assignment Operators | 34 |
| Item 5: Know what functions C++ silently writes and calls. | 34 |
| Item 6: Explicitly disallow the use of compiler-generated functions you do not want. | 37 |
| Item 7: Declare destructors virtual in polymorphic base classes. | 40 |
| Item 8: Prevent exceptions from leaving destructors. | 44 |
| Item 9: Never call virtual functions during construction or destruction. | 48 |
| Item 10: Have assignment operators return a reference to *this. | 52 |
| Item 11: Handle assignment to self in operator=. | 53 |
| Item 12: Copy all parts of an object. | 57 |
| Chapter 3: Resource Management | 61 |
| Item 13: Use objects to manage resources. | 61 |

| | |
|---|----|
| Item 14: Think carefully about copying behavior in resource-managing classes. | 66 |
| Item 15: Provide access to raw resources in resource-managing classes. | 69 |
| Item 16: Use the same form in corresponding uses of new and delete. | 73 |
| Item 17: Store newed objects in smart pointers in standalone statements. | 75 |

Chapter 4: Designs and Declarations 78

| | |
|---|-----|
| Item 18: Make interfaces easy to use correctly and hard to use incorrectly. | 78 |
| Item 19: Treat class design as type design. | 84 |
| Item 20: Prefer pass-by-reference-to-const to pass-by-value. | 86 |
| Item 21: Don't try to return a reference when you must return an object. | 90 |
| Item 22: Declare data members private. | 94 |
| Item 23: Prefer non-member non-friend functions to member functions. | 98 |
| Item 24: Declare non-member functions when type conversions should apply to all parameters. | 102 |
| Item 25: Consider support for a non-throwing swap. | 106 |

Chapter 5: Implementations 113

| | |
|---|-----|
| Item 26: Postpone variable definitions as long as possible. | 113 |
| Item 27: Minimize casting. | 116 |
| Item 28: Avoid returning "handles" to object internals. | 123 |
| Item 29: Strive for exception-safe code. | 127 |
| Item 30: Understand the ins and outs of inlining. | 134 |
| Item 31: Minimize compilation dependencies between files. | 140 |

Chapter 6: Inheritance and Object-Oriented Design 149

| | |
|--|-----|
| Item 32: Make sure public inheritance models "is-a." | 150 |
| Item 33: Avoid hiding inherited names. | 156 |
| Item 34: Differentiate between inheritance of interface and inheritance of implementation. | 161 |
| Item 35: Consider alternatives to virtual functions. | 169 |
| Item 36: Never redefine an inherited non-virtual function. | 178 |

| | |
|--|------------|
| Item 37: Never redefine a function's inherited default parameter value. | 180 |
| Item 38: Model "has-a" or "is-implemented-in-terms-of" through composition. | 184 |
| Item 39: Use private inheritance judiciously. | 187 |
| Item 40: Use multiple inheritance judiciously. | 192 |
| Chapter 7: Templates and Generic Programming | 199 |
| Item 41: Understand implicit interfaces and compile-time polymorphism. | 199 |
| Item 42: Understand the two meanings of typename. | 203 |
| Item 43: Know how to access names in templated base classes. | 207 |
| Item 44: Factor parameter-independent code out of templates. | 212 |
| Item 45: Use member function templates to accept "all compatible types." | 218 |
| Item 46: Define non-member functions inside templates when type conversions are desired. | 222 |
| Item 47: Use traits classes for information about types. | 226 |
| Item 48: Be aware of template metaprogramming. | 233 |
| Chapter 8: Customizing new and delete | 239 |
| Item 49: Understand the behavior of the new-handler. | 240 |
| Item 50: Understand when it makes sense to replace new and delete. | 247 |
| Item 51: Adhere to convention when writing new and delete. | 252 |
| Item 52: Write placement delete if you write placement new. | 256 |
| Chapter 9: Miscellany | 262 |
| Item 53: Pay attention to compiler warnings. | 262 |
| Item 54: Familiarize yourself with the standard library, including TR1. | 263 |
| Item 55: Familiarize yourself with Boost. | 269 |
| Appendix A: Beyond <i>Effective C++</i> | 273 |
| Appendix B: Item Mappings Between Second and Third Editions | 277 |
| Index | 280 |