

Modern C++ Design

*Generic Programming
and Design Patterns Applied*

Andrei Alexandrescu

Foreword by Scott Meyers

Foreword by John Vlissides



C++ In-Depth Series ♦ Bjarne Stroustrup

Modern C++ Design

Modern C++ Design: Generic Programming and Design Patterns Applied

Table of Contents

Contents

Foreword by Scott Meyers

Foreword by John Vlissides

Preface

Acknowledgments

Part I: Techniques

Chapter 1 Policy-Based Class Design

1.1 The Multiplicity of Software Design

1.2 The Failure of the Do-It-All Interface

1.3 Multiple Inheritance to the Rescue?

1.4 The Benefit of Templates

1.5 Policies and Policy Classes

1.6 Enriched Policies

1.7 Destructors of Policy Classes

1.8 Optional Functionality Through Incomplete Instantiation

1.9 Combining Policy Classes

1.10 Customizing Structure with Policy Classes

1.11 Compatible and Incompatible Policies

1.12 Decomposing a Class into Policies

1.13 Summary

Chapter 2 Techniques

2.1 Compile-Time Assertions

Table of Contents

- 2.2 Partial Template Specialization
- 2.3 Local Classes
- 2.4 Mapping Integral Constants to Types
- 2.5 Type-to-Type Mapping
- 2.6 Type Selection
- 2.7 Detecting Convertibility and Inheritance at Compile Time
- 2.8 A Wrapper Around `type_info`
- 2.9 `NullType` and `EmptyType`
- 2.10 Type Traits
- 2.11 Summary

Chapter 3 Typelists

- 3.1 The Need for Typelists
- 3.2 Defining Typelists
- 3.3 Linearizing Typelist Creation
- 3.4 Calculating Length
- 3.5 Intermezzo
- 3.6 Indexed Access
- 3.7 Searching Typelists
- 3.8 Appending to Typelists
- 3.9 Erasing a Type from a Typelist
- 3.10 Erasing Duplicates
- 3.11 Replacing an Element in a Typelist
- 3.12 Partially Ordering Typelists
- 3.13 Class Generation with Typelists
- 3.14 Summary
- 3.15 Typelist Quick Facts

Chapter 4 Small-Object Allocation

- 4.1 The Default Free Store Allocator
- 4.2 The Workings of a Memory Allocator
- 4.3 A Small-Object Allocator

Table of Contents

- 4.4 Chunks
- 4.5 The Fixed-Size Allocator
- 4.6 The SmallObjAllocator Class
- 4.7 A Hat Trick
- 4.8 Simple, Complicated, Yet Simple in the End
- 4.9 Administrivia
- 4.10 Summary
- 4.11 Small-Object Allocator Quick Facts

Part II: Components

Chapter 5 Generalized Functors

- 5.1 The Command Design Pattern
- 5.2 Command in the Real World
- 5.3 C++ Callable Entities
- 5.4 The Functor Class Template Skeleton
- 5.5 Implementing the Forwarding Functor::operator()
- 5.6 Handling Functors
- 5.7 Build One, Get One Free
- 5.8 Argument and Return Type Conversions
- 5.9 Handling Pointers to Member Functions
- 5.10 Binding
- 5.11 Chaining Requests
- 5.12 Real-World Issues I: The Cost of Forwarding Functions
- 5.13 Real-World Issues II: Heap Allocation
- 5.14 Implementing Undo and Redo with Functor
- 5.15 Summary
- 5.16 Functor Quick Facts

Chapter 6 Implementing Singletons

- 6.1 Static Data + Static Functions != Singleton
- 6.2 The Basic C++ Idioms Supporting Singleton
- 6.3 Enforcing the Singletons Uniqueness

Table of Contents

- 6.4 Destroying the Singleton
- 6.5 The Dead Reference Problem
- 6.6 Addressing the Dead Reference Problem (I): The Phoenix Singleton
- 6.7 Addressing the Dead Reference Problem (II): Singletons with
Longevity
- 6.8 Implementing Singletons with Longevity
- 6.9 Living in a Multithreaded World
- 6.10 Putting It All Together
- 6.11 Working with SingletonHolder
- 6.12 Summary
- 6.13 SingletonHolder Class Template Quick Facts

Chapter 7 Smart Pointers

- 7.1 Smart Pointers 101
- 7.2 The Deal
- 7.3 Storage of Smart Pointers
- 7.4 Smart Pointer Member Functions
- 7.5 Ownership-Handling Strategies
- 7.6 The Address-of Operator
- 7.7 Implicit Conversion to Raw Pointer Types
- 7.8 Equality and Inequality
- 7.9 Ordering Comparisons
- 7.10 Checking and Error Reporting
- 7.11 Smart Pointers to const and const Smart Pointers
- 7.12 Arrays
- 7.13 Smart Pointers and Multithreading
- 7.14 Putting It All Together
- 7.15 Summary
- 7.16 SmartPtr Quick Facts

Chapter 8 Object Factories

- 8.1 The Need for Object Factories

Table of Contents

- 8.2 Object Factories in C++: Classes and Objects
- 8.3 Implementing an Object Factory
- 8.4 Type Identifiers
- 8.5 Generalization
- 8.6 Minutiae
- 8.7 Clone Factories
- 8.8 Using Object Factories with Other Generic Components
- 8.9 Summary
- 8.10 Factory Class Template Quick Facts
- 8.11 CloneFactory Class Template Quick Facts

Chapter 9 Abstract Factory

- 9.1 The Architectural Role of Abstract Factory
- 9.2 A Generic Abstract Factory Interface
- 9.3 Implementing AbstractFactory
- 9.4 A Prototype-Based Abstract Factory Implementation
- 9.5 Summary
- 9.6 AbstractFactory and ConcreteFactory Quick Facts

Chapter 10 Visitor

- 10.1 Visitor Basics
- 10.2 Overloading and the Catch-All Function
- 10.3 An Implementation Refinement: The Acyclic Visitor
- 10.4 A Generic Implementation of Visitor
- 10.5 Back to the Cyclic Visitor
- 10.6 Hooking Variations
- 10.7 Summary
- 10.8 Visitor Generic Component Quick Facts

Chapter 11 Multimethods

- 11.1 What Are Multimethods?
- 11.2 When Are Multimethods Needed?
- 11.3 Double Switch-on-Type: Brute Force

Table of Contents

- 11.4 The Brute-Force Approach Automated
- 11.5 Symmetry with the Brute-Force Dispatcher
- 11.6 The Logarithmic Double Dispatcher
- 11.7 FnDispatcher and Symmetry
- 11.8 Double Dispatch to Functors
- 11.9 Converting Arguments: `static_cast` or `dynamic_cast`?
- 11.10 Constant-Time Multimethods: Raw Speed
- 11.11 BasicDispatcher and BasicFastDispatcher as Policies
- 11.12 Looking Forward
- 11.13 Summary
- 11.14 Double Dispatcher Quick Facts

Appendix: A Minimalist Multithreading Library

- A.1 A Critique of Multithreading
- A.2 Lokis Approach
- A.3 Atomic Operations on Integral Types
- A.4 Mutexes
- A.5 Locking Semantics in Object-Oriented Programming
- A.6 Optional volatile Modifier
- A.7 Semaphores, Events, and Other Good Things
- A.8 Summary

Bibliography

Index