1. Record Nr. UNINA9910459611903321

Autore	Reddy Martin
Titolo	API design for C++ [[electronic resource] /] / Martin Reddy
Pubbl/distr/stampa	Boston, : Elsevier/Morgan Kaufmann, 2011
ISBN	1-282-99425-5 9786612994258 0-12-385004-5
Edizione	[1st edition]
Descrizione fisica	1 online resource (468 p.)
Disciplina	005.13/3 005.133
Soggetti	Application program interfaces (Computer software) C++ (Computer program language) Electronic books.
Lingua di pubblicazione	Inglese
Formato	Materiale a stampa
Livello bibliografico	Monografia
Note generali	Description based upon print version of record.
Nota di bibliografia	Includes bibliographical references (p. 409-412) and index.
Nota di contenuto	Front Cover; API Design for C++; Copyright; Contents; Foreword; Preface; Why You Should Read This Book; Who is the Target Audience; Focusing On C++; Conventions; Book Web Site; Acknowledgments; Author Biography; Chapter 1: Introduction; 1.1 What are Application Programming Interfaces?; 1.1.1 Contracts and Contractors; 1.1.2 APIs in; 1.2 What's Different About Api Design?; 1.3 Why Should you Use APIs?; 1.3.1 More Robust Code; 1.3.2 Code Reuse; 1.3.3 Parallel Development; 1.4 When Should you Avoid APIs?; 1.5 Api Examples; 1.5.1 Layers of APIs; 1.5.2 A Real-Life Example 1.6 File Formats and Network Protocols 1.7 About this Book; Chapter 2: Qualities; 2.1 Model the Problem Domain; 2.1.1 Provide a Good Abstraction; 2.1.2 Model the Key Objects; 2.2 Hide Implementation Details; 2.2.1 Physical Hiding: Declaration versus Definition; 2.2.2 Logical Hiding: Encapsulation; 2.2.3 Hide Member Variables; 2.2.4 Hide Implementation Methods; 2.2.5 Hide Implementation Classes; 2.3 Minimally Complete; 2.3.1 Don't Overpromise; 2.3.2 Add Virtual Functions Judiciously; 2.3.3 Convenience APIs; 2.4 Easy to Use; 2.4.1 Discoverable; 2.4.2 Difficult to Misuse; 2.4.3 Consistent 2.4.4 Orthogonal 2.4.5 Robust Resource Allocation; 2.4.6 Platform

Independent; 2.5 Loosely Coupled; 2.5.1 Coupling by Name Only; 2.5.2 Reducing Class Coupling; 2.5.3 Intentional Redundancy; 2.5.4 Manager Classes; 2.5.5 Callbacks, Observers, and Notifications; Callbacks; Observers; Notifications; 2.6 Stable, Documented, and Tested; Chapter 3: Patterns; 3.1 Pimpl Idiom; 3.1.1 Using Pimpl; 3.1.2 Copy Semantics; 3.1.3 Pimpl and Smart Pointers; 3.1.4 Advantages of Pimpl; 3.1.5 Disadvantages of Pimpl; 3.1.6 Opaque Pointers in C; 3.2 Singleton; 3.2.1 Implementing Singletons in
3.2.2 Making Singletons Thread Safe 3.2.3 Singleton versus Dependency Injection; 3.2.4 Singleton versus Monostate; 3.2.5 Singleton versus Session State; 3.3 Factory Methods; 3.3.1 Abstract Base Classes; 3.3.2 Simple Factory Example; 3.3.3 Extensible Factory Example; 3.4 API Wrapping Patterns; 3.4.1 The Proxy Pattern; 3.4.2 The Adapter Pattern; 3.4.3 The Facade Pattern; 3.5 Observer Pattern; 3.5.1 Model-View-Controller; 3.5.2 Implementing the Observer Pattern; 3.5.3 Push versus Pull Observers; Chapter 4: Design; 4.1 A Case for Good Design; 4.1.1 Accruing Technical Debt
4.1.2 Paying Back the Debt 4.1.3 Design for the Long Term; 4.2 Gathering Functional Requirements; 4.2.1 What Are Functional Requirements?; 4.2.2 Example Functional Requirements; 4.2.3 Maintaining the Requirements; 4.3 Creating Use Cases; 4.3.1 Developing Use Cases; 4.3.2 Use Case Templates; 4.3.3 Writing Good Use Cases; 4.3.4 Requirements and Agile Development; 4.4 Elements of Api Design; 4.5 Architecture Design; 4.5.1 Developing an Architecture; 4.5.2 Architecture Constraints; 4.5.3 Identifying Major Abstractions; 4.5.4 Inventing Key Objects; 4.5.5 Architectural Patterns
4.5.6 Communicating the Architecture

Sommario/riassunto

The design of application programming interfaces can affect the behavior, capabilities, stability, and ease of use of end-user applications. With this book, you will learn how to design a good API for large-scale long-term projects. With extensive C++ code to illustrate each concept, API Design for C++ covers all of the strategies of world-class API development. Martin Reddy draws on over fifteen years of experience in the software industry to offer in-depth discussions of interface design, documentation, testing, and the advanced topics of scripting and plug-in extensibility.