1. Record Nr. UNINA9910300747603321

Autore	Nesteruk Dmitri
Titolo	Design Patterns in Modern C++ : Reusable Approaches for Object-Oriented Software Design / / by Dmitri Nesteruk
Pubbl/distr/stampa	Berkeley, CA : , : Apress : , : Imprint : Apress, , 2018
ISBN	9781484236031 1484236033
Edizione	[1st ed. 2018.]
Descrizione fisica	1 online resource (xiii, 314 pages) : illustrations
Disciplina	004
Soggetti	Programming languages (Electronic computers) Software engineering Computer programming Programming Languages, Compilers, Interpreters Software Engineering Programming Techniques
Lingua di pubblicazione	Inglese
Formato	Materiale a stampa
Livello bibliografico	Monografia
Note generali	Includes index.
Nota di contenuto	1. Introduction -- Pt I Creational Patterns -- 2. Builder -- 3. Factories -- 4. Prototype -- 5. Singleton -- Pt II Structural Patterns -- 6. Adapter -- 7. Bridge -- 8. Composite -- 9. Decorator -- 10. Façade -- 11. Flyweight -- 12. Proxy -- Pt III Behavioral Patterns -- 13. Chain of Responsibility -- 14. Command -- 15. Interpreter -- 16. Iterator -- 17. Mediator -- 18. Null Object -- 19. Observer -- 20. State -- 21. Strategy -- 22. Template Method -- 23. Visitor -- 24. Maybe Monad -- Pt IV Appendix -- 25. Appendix A: Functional Design Patterns.
Sommario/riassunto	Apply modern C++17 to the implementations of classic design patterns. As well as covering traditional design patterns, this book fleshes out new patterns and approaches that will be useful to C++ developers. The author presents concepts as a fun investigation of how problems can be solved in different ways, along the way using varying degrees of technical sophistication and explaining different sorts of trade-offs. Design Patterns in Modern C++ also provides a technology demo for modern C++, showcasing how some of its latest features (e. g., coroutines) make difficult problems a lot easier to solve. The

examples in this book are all suitable for putting into production, with only a few simplifications made in order to aid readability. You will: Apply design patterns to modern C++ programming Use creational patterns of builder, factories, prototype and singleton Implement structural patterns such as adapter, bridge, decorator, facade and more Work with the behavioral patterns such as chain of responsibility, command, iterator, mediator and more Apply functional design patterns such as Monad and more.