1. Record Nr. UNINA9910828643203321

Autore	Sturm Oliver
Titolo	Professional functional programming in C# : classic programming techniques for modern projects / / Oliver Sturm
Pubbl/distr/stampa	Chichester, West Sussex, U.K., : Wiley, 2011
ISBN	9786613157331 9781283157339 1283157330 9780470970287 0470970286
Edizione	[1st edition]
Descrizione fisica	1 online resource (290 p.)
Classificazione	COM051060
Disciplina	005.1/14 005.133
Soggetti	C# (Computer program language) Functional programming (Computer science)
Lingua di pubblicazione	Inglese
Formato	Materiale a stampa
Livello bibliografico	Monografia
Note generali	Includes index.
Nota di bibliografia	Includes index.
Nota di contenuto	Professional Functional Programming in C#: Classic Programming Techniques for Modern Projects; Contents; Introduction; Part I: Introduction to Functional Programming; Chapter 1: A Look at Functional Programming History; What Is Functional Programming?; Functional Languages; The Relationship to Object Oriented Programming; Summary; Chapter 2: Putting Functional Programming Into a Modern Context; Managing Side Effects; Agile Programming Methodologies; Declarative Programming; Functional Programming Is a Mindset; Is Functional Programming in C# a Good Idea?; Summary Part II: C# Foundations of Functional ProgrammingChapter 3: Functions, Delegates, and Lambda Expressions; Functions and Methods; Reusing Functions; Anonymous Functions and Lambda Expressions; Extension Methods; Referential Transparency; Summary; Chapter 4: Flexible Typing With Generics; Generic Functions; Generic Classes; Constraining Types; Other Generic Types; Covariance and Contravariance; Summary; Chapter 5: Lazy Listing With Iterators; The Meaning of Laziness; Enumerating Things with .NET; Implementing Iterator Functions;

Returning IEnumerator; Chaining Iterators; Summary
Chapter 6: Encapsulating Data in ClosuresConstructing Functions Dynamically; The Problem with Scope; How Closures Work; Summary; Chapter 7: Code is Data; Expression Trees in .NET; Analyzing Expressions; Generating Expressions; .NET 4.0 Specifics; Summary; Part III: Implementing Well-Known Functional Techniques in C#; Chapter 8: Currying and Partial Application; Decoupling Parameters; Manual Currying; Automatic Currying; Calling Curried Functions; The Class Context; What FCSlib Contains; Calling Parts of Functions; Why Parameter Order Matters; Summary; Chapter 9: Lazy Evaluation What's Good about Being Lazy?Passing Functions; Explicit Lazy Evaluation; Comparing the Lazy Evaluation Techniques; Usability; Efficiency; How Lazy Can You Be?; Summary; Chapter 10: Caching Techniques; The Need to Remember; Precomputation; Memoization; Deep Memoization; Considerations on Memoization; Summary; Chapter 11: Calling Yourself; Recursion in C#; Tail Recursion; Accumulator Passing Style; Continuation Passing Style; Indirect Recursion; Summary; Chapter 12: Standard Higher Order Functions; Applying Operations: Map; Using Criteria: Filter; Accumulating: Fold
Map, Filter, and Fold in LINQStandard Higher Order Functions; Summary; Chapter 13: Sequences; Understanding List Comprehensions; A Functional Approach to Iterators; Ranges; Restrictions; Summary; Chapter 14: Constructing Functions From Functions; Composing Functions; Advanced Partial Application; Combining Approaches; Summary; Chapter 15: Optional Values; The Meaning of Nothing; Implementing Option(al) Values; Summary; Chapter 16: Keeping Data From Changing; Change Is Good - Not!; False Assumptions; Being Static Is Good; A Matter of Depth; Cloning; Automatic Cloning Implementing Immutable Container Data Structures

Sommario/riassunto

Take advantage of the growing trend in functional programming. C# is the number-one language used by .NET developers and one of the most popular programming languages in the world. It has many built-in functional programming features, but most are complex and little understood. With the shift to functional programming increasing at a rapid pace, you need to know how to leverage your existing skills to take advantage of this trend. Functional Programming in C# leads you along a path that begins with the historic value of functional ideas. Inside, C# MVP and functional progra