1. Record Nr. UNINA9910300656603321

Autore	Fischer Robert
Titolo	Java Closures and Lambda / / by Robert Fischer
Pubbl/distr/stampa	Berkeley, CA : , : Apress : , : Imprint : Apress, , 2015
ISBN	9781430259992 143025999X
Edizione	[1st ed. 2015.]
Descrizione fisica	1 online resource (207 p.)
Collana	Expert's voice in Java
Disciplina	005.13/3
Soggetti	Java (Computer program language) Software engineering Java Software Engineering/Programming and Operating Systems
Lingua di pubblicazione	Inglese
Formato	Materiale a stampa
Livello bibliografico	Monografia
Note generali	Includes index.
Nota di contenuto	Contents at a Glance; Introduction; Chapter 1: Java 8: It's a Whole New Java; Java 8 Returns Java to the Forefront; Java Has Had Functional Programming All Along; Java 8 Is Not Just Syntactic Sugar; Is Java 8 a Functional Programming Language?; Enough with the Theory; onto the Implementation!; Chapter 2: Understanding Lambdas in Java 8; Java 8's Lambda Syntax; Lambdas as Closures; No-Argument and Multi-Argument Lambdas; Partial Function Application and Mr. Curry's Verb; Mr. Curry's Verb and Functional Shape; Lambdas with No Return Value; Lambdas with Complex Bodies Lambdas with Explicit TypingLambdas as Operators; Lambdas as Predicates; Lambdas with Primitive Arguments; Making Methods into Lambdas; Making Static Methods into Lambdas; Making Constructors into Lambdas; Making Instance Methods into Lambdas; Specifying a Method to Be Used Later; Lambdas as Interface Implementations; Def ault Methods; Static Methods; Functional Interface Helper Methods; Function.identity() and UnaryOperator.identity(); Function.compose and Function.andThen; Consumer.andThen; Predicate.and and Predicate.or; Predicate.isEqual; Predicate.negate BinaryOperator. minBy and BinaryOperator. maxByLambda Best Practices; Use Interfaces; Use Method References; Define Lambdas

Inline; Lambdas Should Always Be Threadsafe; Don't Use Null; Don't R elease Zalgo; Build Complexity from Simple Parts; Use Types and the Compiler to Your Advantage; Chapter 3: Lambda's Domain: Collections, Maps, and Streams; Lambdas and Functional Programming; Functional Iteration; Manipulating Collections and Maps with Lambdas; Filtering Collections and Maps; Mapping Collections and Maps; Map Computations; Streams; Stream Creation; Mapping and Filtering Streams
Collecting, Processing, or Reducing StreamsPrimitive Streams; Lambda's Domain in Review; Chapter 4: I/O with Lambdas; Temporary Files and the Hole in the Middle; Exception Handling via Input: Passing in an Exception Handler; Exception Handling via Output: Capturing Execution Results; Consuming Our Temp File Function; Reading All the Lines of Files in a Directory; Complex Stream Processing Using Creative Flattening; Streaming all the Lines of all the Files in a Directory; Summary; Chapter 5: Data Access with Lambdas; Representing the Intermediary Query Results; Printing Out the Results
Mapping the ResultSet to a StreamMethod One: Building a Stream Using the Stream Builder; Method Two: Building a Stream Using Stream.of and Stream.flatMap; Mapping the ResultSet with Result-Based Error Handling; Method Three: Building a Stream Using an AbstractSpliterator; Method Four: Building a Stream from an Iterator; Pulling It All Together; Chapter 6: Lambda Concurrency; Lambdas and Classic Java Threading; Lambdas and Executors; Lambdas and the ThreadPoolExecutor; Lambdas and Fork/Join; Stream Parallelism; Conclusion; Chapter 7: Lambdas and Legacy Code; Resources and Exceptions
Handling Resources by Throwing an Unchecked Exception

Sommario/riassunto

Java Closures and Lambda introduces you to significant new changes to the Java language coming out of what is termed Project Lambda. These new changes make their debut in Java 8, and their highlight is the long-awaited support for lambda expressions in the Java language. You’ll learn to write lambda expressions and use them to create functional interfaces and default methods for evolving APIs, among many other uses. The changes in Java 8 are significant. Syntax and usage of the language are changed considerably with the introduction of closures and lambda expressions. This book takes you through these important changes from introduction to mastery. Through a set of clear examples, you’ll learn to refactor existing code to take advantage of the new language features. You’ll learn what those features can do for you, and when they are best applied. You’ll learn to design and write new code having these important new features in mind from the very beginning. Clearly explains the fantastic benefits resulting from Project Lambda Explains the syntax and IDE support for the new features Shows how to streamline your code by bringing some of the benefits of functional programming to the Java language Illustrates parallelism in closures through Stream and Spliterator objects Explains API evolution by adding methods to existing interfaces without breaking existing interface implementations, a technique addressing potential multiple inheritance issues.