Mauro Bogliaccino

Corso Java

Programma del corso Java SE - Java SE 11 Programmer II

Java SE 11 Programmer II

obiettivi

- Creare applicazioni Java che sfruttano le funzionalità orientate agli oggetti del linguaggio Java, come incapsulamento, ereditarietà e polimorfismo
- Eseguire un'applicazione Java dalla riga di comando
- Creare applicazioni che utilizzano il Java Collections Framework
- Cercare e filtrare le raccolte utilizzando Lambda Expressions
- Implementare tecniche di gestione degli errori utilizzando la gestione delle eccezioni
- Implementare la funzionalità di input/output (I/O) per leggere e scrivere su file di dati e di testo

obiettivi II

- Manipolare file, directory e file system usando la specifica JDK NIO.2
- Eseguire più operazioni su tabelle di database, tra cui la creazione, la lettura, l'aggiornamento e l'eliminazione utilizzando sia la tecnologia JDBC che JPA
- Utilizzare le funzionalità di concorrenza di Lambda Expression
- Applicare pratiche e servizi di programmazione modulari alle applicazioni
- Creare applicazioni multi-thread ad alte prestazioni

Programmazione Funzionale

- Interfacce funzionali ed espressioni lambda
- Stream di raccolte e filtri
- Interfacce funzionali integrate

Stream e stream paralleli

- Concorrenza
- Flussi paralleli
- Operazioni terminal: Collezionisti
- Creazione di stream personalizzati

Programmazione API Java e concetti di codifica sicura

- I / O (Fundamentals e NIO2)
- Codifica sicura
- Applicazioni di database con JDBC
- Localizzazione

Collezioni e generici

- Cos'è il Java Collections Framework (JCF)?
- Iteratori
- Classi di tipo wrapper
- Cosa sono i generici?
- gestire le collezioni: List, Set e Map
- I metodi di utilità di Collections
- Vector, ArrayList
- LinkedList
- HashSet, TreeSet

Elementi Sintassi JAVA

- Java Advanced: enumerazioni
- Java Advanced: inizializzatore statico
- Creare Jar eseguibili, creare e includere Jar nel progetto
- Java advanced:
  - nested classes,
  - member classes,
  - anonymous inner classes
- Reflection, Class, Constructor classes

Advanced Class design

- Binding dinamico
- Casting con oggetti
- Cast e instanceof
- Classi astratte
- Classi innestate o interne
- Classi anonime
- Schede CRC e diagrammi UML.
- Classi anonime
- Classi astratte
- Lambda espressioni.

design patterns

- singleton
- decorator
- mvc
- dao
- factory

JAVA WEB

- introduzione JSP
- approfondimento JSP
- JSP e Servlet
- [JSP: le direttive]
- [JSP: le direttive]
- [JSP: le espressioni]
- [JSP: le dichiarazioni]
- GlassFish e TomCat
- JSTL: Jsp standard action
- Servlet REST controller: doGet, doPost
- RequestDispatcher

JDBC

- JDBC
  - connector-J
  - Connection
  - DriverManager
  - Statement, PreparedStatement
  - ResulSet
  - executeQuery()
  - executeUpdate()

Fondamenti di Java

- Creare e utilizzare le classi final
- Creare e utilizzare classi interne, nidificate e anonime
- Creare e utilizzare le enumerazioni

Gestione delle eccezioni e asserzioni

- Usa il costrutto try-with-resources
- Creare e utilizzare classi di eccezione personalizzate
- Metti alla prova gli invarianti usando le asserzioni

Eccezioni

- Meccanismo di gestione delle eccezioni
- Eccezioni controllate e non
- Errori ed eccezioni
- Il costrutto TRY-CATCH-FINALLY
- Tipi di eccezione ed il costrutto Throws
- Lanciare un'eccezione: il costrutto Throw
- Eccezioni personalizzate
- Cenni sull'uso di un debugger
- Il debugger di Eclipse
- Correzione di un programma Java.
- Test unitari

Interfacce Java

- Creare e utilizzare interfacce con metodi predefiniti (default)
- Creare e utilizzare interfacce con metodi privati

Generici e Collezioni

- Utilizzare le classi wrapper, il boxing automatico e l'unboxing automatico
- Creare e utilizzare classi generiche, metodi con notazione a diamante e caratteri jolly
- Descrivi il Framework di raccolta e usa le interfacce di raccolta chiavi
- Usa Comparator e interfacce comparabili
- Creare e utilizzare metodi di praticità per le raccolte

Framework Collections

- Foreach ed Iterator
- Implementazioni di Set e SortedSet
- Implementazioni di List
- Implementazioni di Queue
- Implementazioni di Map e SortedMap
- Tipi Generics

Interfaccia funzionale ed espressioni lambda

- Definire e scrivere interfacce funzionali
- Crea e usa espressioni lambda tra cui istruzione lambdas, variabile locale per i parametri lambda

API Java Stream

- Descrivere l'interfaccia Stream e le pipeline
- Usa espressioni lambda e riferimenti a metodi

Interfacce funzionali integrate

- Utilizzare le interfacce dal pacchetto java.util.function
- Utilizzare le interfacce funzionali di base tra cui Predicato, Consumatore, Funzione e Fornitore
- Usa le variazioni primitive e binarie delle interfacce di base del pacchetto java.util.function

Operazioni Lambda in streaming

- Estrai i dati del flusso usando i metodi map, peek e flatMap
- Cerca i dati dello stream utilizzando i metodi di ricerca findFirst, findAny, anyMatch, allMatch e noneMatch
- Usa la classe opzionale
- Eseguire calcoli utilizzando le operazioni di conteggio, max, min, media e somma del flusso
- Ordina una raccolta usando le espressioni lambda
- Utilizzare i raccoglitori con stream, incluse le operazioni groupingBy e partitioningBy

I/O (Fundamentals e NIO2)

- Leggi i dati e scrivi i dati della console e dei file utilizzando il flusso I / O
- Utilizzare I / O Stream per leggere e scrivere file
- Leggere e scrivere oggetti utilizzando la serializzazione
- Utilizzare l'interfaccia Path per operare su percorsi di file e directory
- Utilizzare la classe Files per controllare, eliminare, copiare o spostare un file o una directory
- Usa l'API Stream con i file

Input/Output

- Introduzione all'I/O: input da tastiera
- Java.io: Leggere un file
- Java.io: Scrivere su un file
- Java.io:Operazioni su file
- Networking: Socket
- Flussi di byte e di caratteri
- Flussi di oggetti
- Accesso sequenziale e random.
- Files
- Serializzazione e deserializzazione.

Codifica sicura nell'applicazione Java SE

- Prevenzione della negazione del servizio nelle applicazioni Java
- Protezione delle informazioni riservate nell'applicazione Java
- Implementazione delle linee guida per l'integrità dei dati - iniezioni e convalida dell'inclusione e dell'input
- Prevenire l'attacco esterno del codice limitando l'accessibilità e l'estensibilità, gestendo correttamente la convalida dell'input e la mutabilità
- Protezione della costruzione di oggetti sensibili
- Protezione della serializzazione e della deserializzazione

Applicazioni di database con JDBC

- Collegati ai database utilizzando gli URL JDBC e DriverManager
- Utilizzare PreparedStatement per eseguire operazioni CRUD
- Utilizzare le API PreparedStatement e CallableStatement per eseguire operazioni sul database

Introduzione a JDBC

- Convenzioni JDBC URL Naming
- Gestione dei driver: il DriverManager
- Gestione degli errori
- Le SQLException
- Supporto per i tipi di dati
- Estensioni standard di JDBC
- Connection Pooling
- Gestione dei dati: JDBC
- Crud su DB

Localizzazione

- Usa la classe Locale
- Usa pacchetti di risorse
- Formatta messaggi, date e numeri con Java

annotazioni

- Descrivere lo scopo delle annotazioni e dei tipici schemi di utilizzo
- Applica annotazioni a classi e metodi
- Descrivi le annotazioni di uso comune nel JDK
- Dichiara annotazioni personalizzate

JSP e Servlet

- Primo approccio a JSP
- Installazione ed esecuzione della prima pagina JSP
- Elementi fondamentali di JSP
- Utilizzo degli elementi fondamentali
- Utilizzo di JavaBeans
- Lavorare con i database
- Elementi Avanzati di una pagina JSP
- Uso di Etichette personalizzate

Distribuire e mantenere un'applicazione

- Pacchetti, JAR, architettura
- Modifica e requisiti dell'applicazione

Compredere l'uso dei moduli

- Il sistema del modulo
- JARs
- Dichiarazioni del modulo
- JDK modulare

Programmazione modulare

- Introduzione alla programmazione modulare in Java
- Servizi in un'applicazione modulare
- Migrazione ad un'applicazione modulare

Comprensione dei moduli

- Descrivi il JDK modulare
- Dichiarare i moduli e abilitare l'accesso tra i moduli
- Descrivi come viene compilato ed eseguito un progetto modulare

Guida su baeldung.com

Migrazione ad un'applicazione modulare

- Migrare l'applicazione sviluppata utilizzando una versione Java precedente a SE 9 a SE 11 inclusa la migrazione top-down e bottom-up, suddividendo un'applicazione Java SE 8 in moduli per la migrazione
- Esegui un'applicazione modulaized su classpath e su modulepath
- Utilizzare jdeps per determinare le dipendenze e identificare il modo per affrontare le dipendenze cicliche

Servizi in un'applicazione modulare

- Descrivere i componenti dei servizi, comprese le direttive
- Progetta un tipo di servizio,
- carica i servizi utilizzando ServiceLoader,
- verificare le dipendenze dei servizi inclusi i moduli consumer e provider

Concorrenza

- Creare thread di lavoro utilizzando Runnable, Callable e utilizzare un ExecutorService per eseguire contemporaneamente attività
- Usa java util raccolte e classi simultanee tra cui CyclicBarrier e CopyOnWriteArrayList
- Scrivi codici thread-safe
- Identificare i problemi di threading come deadlock e livelock

Flusso parallelo

- Sviluppa il codice che utilizza il flusso parallelo
- Implementa decomposizione e riduzione con stream