Java SE 8 Programming

Duración del curso

5 días (40 hrs)

Descripción general:
El curso de programación Java SE 8 cubre las características principales del lenguaje y las interfaces de programación de aplicaciones (API) que usará para diseñar aplicaciones orientadas a objetos con la plataforma Java Standard Edition 8 (Java SE 8).

Objetivos:

Al término del curso usted será capaz de:

- Diseñar aplicaciones en la plataforma Java usando las APIs nativas más conocidas.
- Poner en práctica los conocimientos sobre orientación a objetos.
- Identificar buenas prácticas para diseñar aplicaciones robustas.
- Utilizar y diseñar expresiones Lamba.
- Almacenar y manipular datos usando colecciones.
- Interactuar con archivos, directorios y el sistema de ficheros.
- Conectar con bases de datos utilizando consultas SQL estándar mediante JDBC.
- Diseñar aplicaciones multi-thread con alto rendimiento.

Prerrequisitos:
Java SE 8 Fundamentos

Esquema del curso

Descripción general de la plataforma Java

- Definición de cómo el lenguaje Java logra la independencia de la plataforma
- Diferenciación entre las plataformas Java ME, Java SE y Java EE
- Evaluación de las opciones de bibliotecas, middleware y bases de datos de Java
- Definiendo cómo sigue evolucionando el lenguaje Java

Revisión de la sintaxis y las clases de Java

- Creación de clases Java sencillas
- Creación de variables primitivas
- Uso de operadores
- Creación y manipulación de cadenas
- Uso de las instrucciones if-else y switch
- Iteración con bucles: while,do-while,for,enhanced for
- Creación de matrices
- Uso de campos, constructores y métodos Java

Encapsulación y subclases

- Uso de la encapsulación en el diseño de clases de Java
- Modelado de problemas empresariales mediante clases Java
- Hacer que las clases sean inmutables
- Creación y uso de subclases de Java
- Métodos de sobrecarga

Métodos de invalidación, polimorfismo y clases estáticas

- Uso de niveles de acceso: privado, protegido, predeterminado y público
- Métodos de anulación
- Uso de la invocación de métodos virtuales
- Uso de varargs para especificar argumentos de variables
- Uso del operador instanceof para comparar tipos de objetos
- Uso de lances hacia arriba y hacia abajo
- Modelado de problemas empresariales mediante la palabra clave static

Clases abstractas y anidadas

- Diseño de clases base de uso general mediante clases abstractas
- Construcción de clases y subclases abstractas de Java
- Aplicación de la palabra clave final en Java
- Distinguir entre clases de nivel superior y anidadas

Interfaces y expresiones lambda

- Definición de una interfaz Java
- Elegir entre la herencia de interfaz y la herencia de clases
- Ampliación de una interfaz
- Métodos predeterminados
- Clases internas anónimas
- Definición de una expresión lambda

Colecciones y Genéricos

- Creación de una clase genérica personalizada
- Uso del diamante de inferencia de tipos para crear un objeto
- Creación de una colección mediante el uso de genéricos
- Implementación de una ArrayList
- Implementación de un TreeSet
- Implementación de un HashMap
- Implementación de un Deque
- Ordenar cobros

Colecciones y Genéricos

- Creación de una clase genérica personalizada
- Uso del diamante de inferencia de tipos para crear un objeto
- Creación de una colección mediante el uso de genéricos
- Implementación de una ArrayList
- Implementación de un TreeSet
- Implementación de un HashMap
- Implementación de un Deque
- Ordenar cobros

Interfaces funcionales integradas de Lambda

- Enumeración de las interfaces integradas incluidas en java.util.function
- Interfaces principales: predicado, consumidor, función, proveedor
- Uso de versiones primitivas de interfaces base
- Uso de versiones binarias de interfaces base

Operaciones de Lambda

- Extracción de datos de un objeto mediante map
- Descripción de los tipos de operaciones de flujo
- Descripción de la clase Optional
- Descripción del procesamiento diferido
- Ordenar una secuencia
- Guardar los resultados en una colección mediante el método collect
- Agrupación y partición de datos mediante la clase Collectors

Excepciones y aserciones

- Definición del propósito de las excepciones de Java
- Uso de las cláusulas catch, multi-catch y finally
- Cierre automático de recursos con una instrucción try-with-resources
- Reconocimiento de clases y categorías de excepción comunes
- Creación de excepciones personalizadas
- Prueba de invariantes mediante aserciones

API de fecha y hora de Java

- Creación y administración de eventos basados en fechas
- Creación y administración de eventos basados en el tiempo
- Combinación de fecha y hora en un solo objeto
- Trabajar con fechas y horas en distintas zonas horarias
- Gestión de los cambios resultantes del horario de verano
- Definición y creación de marcas de tiempo, períodos y duraciones
- Aplicación de formato a fechas y horas locales y por zonas

Fundamentos de E/S

- Descripción de los conceptos básicos de entrada y salida en Java
- Lectura y escritura de datos desde la consola
- Uso de secuencias para leer y escribir archivos
- Escritura y lectura de objetos mediante la serialización

E/S de archivo (NIO.2)

- Uso de la interfaz Path para operar en rutas de acceso de archivos y directorios
- Uso de la clase Files para comprobar, eliminar, copiar o mover un archivo o directorio
- Uso de la API de Stream con NIO2

Concurrencia

- Descripción de la programación de tareas del sistema operativo
- Creación de subprocesos de trabajo mediante Runnable y Callable
- Uso de un ExecutorService para ejecutar tareas simultáneamente
- Identificación de posibles problemas de subprocesos
- Uso de la tecnología atómica sincronizada y concurrente para administrar la atomicidad
- Uso de bloqueos de monitor para controlar el orden de ejecución de los subprocesos
- Uso de las colecciones java.util.concurrent

El marco de unión de bifurcación

- Paralelismo
- La necesidad de Fork-Join
- Robo de trabajo
- RecursiveTask

Flujos paralelos

- Revisión de las características clave de las transmisiones
- Descripción de cómo hacer que una canalización de flujo se ejecute en paralelo
- Enumerar los supuestos clave necesarios para usar una canalización paralela
- Definición de reducción
- Describir por qué la reducción requiere una función asociativa
- Cálculo de un valor mediante reduce
- Describir el proceso de descomposición y posterior fusión del trabajo
- Enumeración de las consideraciones clave de rendimiento para secuencias paralelas

Aplicaciones de bases de datos con JDBC

- Definición del diseño de la API de JDBC
- Conexión a una base de datos mediante un controlador JDBC
- Envío de consultas y obtención de resultados de la base de datos
- Especificación externa de la información del controlador JDBC
- Realización de operaciones CRUD mediante la API de JDBC

Localización

- Descripción de las ventajas de localizar una aplicación
- Definición de lo que representa una configuración regional
- Leer y establecer la configuración regional mediante el objeto Locale
- Creación de un paquete de recursos para cada configuración regional
- Llamar a un paquete de recursos desde una aplicación
- Cambiar la configuración regional de un paquete de recursos