Programação Java SE

Tecnologias de Informação - Desenvolvimento / Programação

- **Nível:** Intermédio
- **Duração:** 35h

Sobre o curso

Este curso avançado em programação Java visa capacitar os participantes com conhecimentos práticos e técnicos essenciais para o desenvolvimento de aplicações modernas, eficientes e escaláveis.

Através de módulos focados em áreas como programação funcional, APIs modernas, concorrência, sistemas modulares e boas práticas em debugging, os formandos aprenderão a escrever código de alta qualidade, otimizar performance e aplicar padrões avançados de desenvolvimento.

Com exercícios práticos e projetos integrados, os participantes estarão preparados para enfrentar desafios do mercado de trabalho e a se proporem para exame 1Z0-830.

Destinatários

- O curso é destinado a programadores Java que já possuem experiência em desenvolvimento de aplicações e procuram aprofundar os seus conhecimentos em tópicos avançados para melhorar as suas práticas de desenvolvimento e se destacarem no mercado de trabalho.
- É ideal para programadores que ambicionam cargos mais avançados.
- Para aqueles que pretendem se preparar para certificações como o Oracle Certified Professional (OCP) – Java SE Developer (1Z0-830)

Objetivos

No final da ação de formação os participantes deverão estar aptos a:

- Desenvolver competências para aplicar a programação funcional com expressões lambda e Streams para otimizar o processamento de dados
- Capacitar os formandos a projetar e implementar sistemas modulares com a API Jigsaw
- Aplicar boas práticas de debugging em IDEs para a identificação e resolução de erros de forma eficiente
- Explorar a utilização de APIs modernas, como a API Date/Time e NIO.2, para criar aplicações robustas

- Aprofundar conhecimentos em concorrência e processamento paralelo para melhorar a performance de aplicações
- Aplicar padrões avançados de design e técnicas de desenvolvimento seguro em projetos reais
- Prepararem-se para o exame 1Z0-830. Este exame de certificação não está incluído no curso, mas todos os candidatos que pretendam a certificação internacional, ao concluir com sucesso o exame, irão obter a certificação Oracle Certified Professional: Java SE Developer.

Pré-requisitos

- Conhecimentos de programação orientada a objetos em Java
- Familiaridade com conceitos básicos de desenvolvimento de software e estruturas de dados.
- Experiência com o uso de IDEs como IntelliJ IDEA ou Eclipse.
- Compreensão de princípios de desenvolvimento de software e uso de bibliotecas padrão do Java

---

Programa

- Revisão de variáveis, tipos de dados e estruturas de controlo com boas práticas
- Algoritmos eficientes e resolução de problemas práticos
- Debugging avançado em IDEs como IntelliJ IDEA e Eclipse
- Generics e coleções avançadas
- Programação funcional
- APIs modernas
- Concorrência e processamento paralelo
- Projeto de sistemas modulares com Jigsaw

Revisão de variáveis, tipos de dados e estruturas de controlo com boas práticas

Este módulo tem como objetivo rever e reforçar os conceitos fundamentais de variáveis, tipos de dados e estruturas de controlo em Java, aplicando boas práticas de desenvolvimento para escrever código limpo e eficiente. Os participantes aprenderão a escolher e utilizar as melhores práticas para tornar o código mais legível e robusto, facilitando a manutenção e a colaboração em projetos de desenvolvimento.

Lições

- Revisão de variáveis e tipos de dados

Exploração dos diferentes tipos de variáveis e dados em Java (primitivos e objetos) e boas práticas para a sua escolha e utilização.

- Estruturas de controlo de fluxo

Aplicação das estruturas de controlo (if-else, switch, for e while) com foco na escrita de código eficiente e legível.

- Boas práticas em estruturas de controlo

Técnicas para evitar complexidade desnecessária e manter a clareza do código em estruturas de decisão e repetição.

Lab: Best Practices in Variables and Control Structures

- Exercise 1: Criar um programa que utilize diferentes tipos de variáveis e dados para resolver um problema simples (e.g., cálculo de média de notas de alunos) e aplicar boas práticas na escolha dos tipos de dados.
- Exercise 2: Implementar uma aplicação que utilize estruturas de controlo de fluxo para resolver um problema de decisão (e.g., sistema de recomendação de produtos baseado em preferências do usuário).
- Exercise 3: Refatorar um código existente com estruturas de controlo complexas para aplicar boas práticas de legibilidade e simplificação.
- Exercise 4: Criar um programa que utilize loops e estruturas condicionais e de iteração para calcular a sequência de Fibonacci e aplicar técnicas de otimização para evitar repetições desnecessárias.

Algoritmos eficientes e resolução de problemas práticos

Este módulo tem como objetivo desenvolver a capacidade dos formandos de criar algoritmos eficientes para resolver problemas reais, utilizando as melhores práticas de programação em Java. Os participantes aprenderão a analisar a complexidade de algoritmos, otimizar soluções e aplicar estruturas de dados apropriadas para alcançar desempenho elevado em cenários empresariais.

Lições

- Análise de complexidade e boas práticas de otimização

Introdução à análise de complexidade de algoritmos (Big-O) e identificação de soluções ineficientes.

- Estruturas de dados para algoritmos eficientes

Seleção e utilização de estruturas de dados apropriadas, como listas, mapas e filas de prioridade, para resolver problemas específicos.

- Abordagens práticas para resolução de problemas

Aplicação de estratégias como divisão e conquista, backtracking e algoritmos gulosos em problemas empresariais.

Lab: Designing and Optimizing Algorithms

- Exercise 1: Analisar a complexidade de diferentes algoritmos para um problema simples (e.g., busca de um elemento em uma lista) e propor otimizações.
- Exercise 2: Implementar um algoritmo eficiente para resolver um problema prático, como ordenar e filtrar uma lista de dados (e.g., transações financeiras ou informações de clientes).
- Exercise 3: Aplicar a abordagem de divisão e conquista para resolver um problema mais complexo (e.g., busca binária em um conjunto de dados ordenados).
- Exercise 4: Resolver um problema empresarial utilizando algoritmos, como a seleção de tarefas com base em recursos limitados.

Debugging avançado em IDEs como IntelliJ IDEA e Eclipse

Este módulo tem como objetivo ensinar aos formandos a identificar e corrigir erros de forma eficiente em aplicações Java, utilizando as ferramentas avançadas de debugging disponíveis nas IDEs IntelliJ IDEA e Eclipse. Os participantes desenvolverão competências práticas para investigar comportamentos inesperados no código, analisar fluxos de execução e otimizar o desempenho nas suas aplicações.

Lições

- Ferramentas e funcionalidades de debugging

Exploração dos recursos de debugging avançado das IDEs, como breakpoints condicionais, inspeção de variáveis e execução passo a passo.

- Análise de fluxos de execução e chamadas de métodos

Investigação detalhada de pilhas de chamadas, navegação entre métodos e análise de fluxos lógicos em tempo de execução.

- Debugging de aplicações multithreaded

Identificação e solução de problemas em ambientes concorrentes, incluindo inspeção de threads e gestão de condições de corrida.

Lab: Debugging Applications in Java

- Exercise 1: Utilizar breakpoints condicionais e inspeção de variáveis para encontrar e corrigir erros lógicos em um algoritmo simples (e.g., cálculo de impostos)
- Exercise 2: Analisar o fluxo de execução de uma aplicação complexa para identificar o método responsável por comportamentos inesperados (e.g., erros em um sistema de reservas)
- Exercise 3: Diagnosticar e corrigir problemas de concorrência em uma aplicação multithreaded (e.g., condições de corrida em uma aplicação de cálculo financeiro)
- Exercise 4: Usar ferramentas de profiling das IDEs para identificar problemas de desempenho em uma aplicação Java

Generics e coleções avançadas

Este módulo tem como objetivo capacitar os formandos a trabalhar com coleções de dados complexas e a implementar soluções genéricas reutilizáveis, fundamentais para o desenvolvimento de aplicações modernas. Os participantes aprenderão a selecionar e utilizar estruturas de dados eficientes, personalizar ordenações e aplicar generics de forma avançada para escrever código seguro e flexível.

Lições

- Estruturas de dados eficientes: Maps, Sets e Lists:

Exploração das principais estruturas de dados da biblioteca Java Collections Framework, com foco na escolha adequada para diferentes cenários empresariais

- Técnicas avançadas de ordenação e bounded wildcards;

Personalização de ordenações com Comparator e Comparable, e aplicação de generics com bounded wildcards

para maior flexibilidade no código

- Integração de coleções com APIs comuns

Aplicação prática de coleções em conjunto com APIs externas

Lab: Managing Advanced Collections and Generics

- Exercise 1: Selecionar e implementar a estrutura de dados mais adequada para resolver problemas reais (e.g., gestão de inventário ou cálculo de estatísticas)
- Exercise 2: Personalizar a ordenação de uma lista de objetos complexos (e.g., lista de clientes com base em critérios como idade ou volume de compras) utilizando Comparator e Comparable.
- Exercise 3: Criar uma aplicação que utilize bounded wildcards para manipular coleções de diferentes tipos com segurança.
- Exercise 4: Integrar coleções com APIs externas

Programação funcional

Este módulo foca-se em ensinar aos formandos a utilizarem a programação funcional para processamento de coleções de dados de forma eficiente e com menor complexidade de código. Através do uso de expressões lambda e da Stream API, os participantes aprenderão a resolver problemas típicos do desenvolvimento empresarial, como a filtragem, transformação e agregação de dados, e a implementar soluções paralelas para otimizar a performance em grandes volumes de informação.

Lições:

- Lambdas e Streams em contextos empresariais: exploração das expressões lambda e da Stream API para simplificar a manipulação de dados em aplicações reais.
- Paralelismo com Streams para otimizar performance: implementação de Streams paralelos para processar grandes volumes de dados de forma eficiente.
- Integração de Streams com bibliotecas populares: aplicação prática de Streams em conjunto com ferramentas amplamente utilizadas.

Lab: Creating and Optimizing Functional Code

- Exercise 1: Criar um pipeline funcional para processar uma lista de objetos (e.g., catálogo de produtos ou lista de clientes), aplicando operações como filtragem e ordenação.
- Exercise 2: Implementar uma aplicação que processe um grande volume de dados (e.g., registos de vendas) utilizando Streams paralelos e analisar os ganhos de performance
- Exercise 3: Desenvolver uma solução prática para manipulação de coleções avançadas, integrando a Stream API com ferramentas externas.

APIs modernas

Este módulo tem como objetivo capacitar os formandos a trabalhar com as APIs mais recentes do Java para desenvolver aplicações robustas e eficientes. Os participantes aprenderão a utilizar a API Date/Time para operações com datas, manipular ficheiros e diretórios com NIO.2 e criar soluções de logging eficientes. Com o uso dessas APIs, os formandos serão capazes de construir aplicações que atendam às exigências modernas de

desenvolvimento e manutenção

Lições:

- API Date/Time para operações com datas complexas

Exploração da API Date/Time para manipulação de datas, horas e fusos horários, incluindo formatação e operações aritméticas.

- Manipulação de ficheiros e diretórios com NIO.2

Utilização da API NIO.2 para ler e escrever ficheiros, criar diretórios e gerir caminhos de forma eficiente e segura.

- Criação de sistemas de logs eficientes

Aplicação de práticas para integrar sistemas de logging em aplicações Java, utilizando APIs como e bibliotecas externas

Lab: Working with Modern Java APIs

- Exercise 1: Desenvolver um programa que utilize a API Date/Time para calcular e exibir a diferença entre duas datas, levando em consideração fusos horários diferentes.
- Exercise 2: Criar uma aplicação que leia e escreva ficheiros utilizando NIO.2, demonstrando operações de manipulação de diretórios e gestão de caminhos.
- Exercise 3: Implementar um sistema de logging em uma aplicação Java, configurando diferentes níveis de log e salvando logs em ficheiros de forma estruturada.
- Exercise 4: Criar um programa de exemplo que combine a manipulação de datas e a escrita de logs para monitorizar a execução de um processo (e.g., processamento de dados em lote)

Concorrência e processamento paralelo

Este módulo tem como objetivo ensinar os formandos a desenvolver aplicações Java que aproveitem o processamento paralelo para melhorar a performance e a escalabilidade. Os participantes aprenderão a utilizar as ferramentas e técnicas avançadas da API de concorrência do Java, além de aplicar padrões concorrentes para resolver problemas complexos em ambientes multithreaded.

Lições:

- Introdução à API de concorrência

Exploração da API de concorrência do Java, incluindo threads, Runnable e ExecutorService para gerenciar a execução de tarefas.

- Fork/Join Framework e CompletableFuture

Uso do Fork/Join Framework para dividir e conquistar tarefas complexas e CompletableFuture para programação assíncrona e encadeamento de tarefas.

- Padrões concorrentes em aplicações empresariais

Implementação de padrões como Producer-Consumer, Executor, e Thread Pool para resolver problemas comuns de concorrência e melhorar a performance de aplicações.

Lab: Developing Concurrent and Parallel Applications

- Exercise 1: Criar uma aplicação que utilize threads para executar tarefas simples de forma paralela (e.g., cálculos matemáticos em paralelo).
- Exercise 2: Implementar um programa que utilize o Fork/Join Framework para dividir uma tarefa de processamento em sub-tarefas e otimizar a execução.
- Exercise 3: Desenvolver uma aplicação que utilize CompletableFuture para encadear tarefas assíncronas e gerenciar exceções de forma eficiente.
- Exercise 4: Resolver um problema real com o uso de padrões concorrentes, como a implementação de um sistema de Producer-Consumer para processamento de filas de tarefas.

Projeto de sistemas modulares com Jigsaw

Este módulo tem como objetivo ensinar os formandos a projetar e implementar sistemas modulares em Java utilizando o Jigsaw que implementa o Java Plataform Module System (JPMS). Os participantes aprenderão a criar módulos, gerir dependências e integrar sistemas modulares com frameworks empresariais, tornando as aplicações mais organizadas, escaláveis e fáceis de manter.

Lições:

- Introdução ao sistema de módulos Java (Jigsaw)

Exploração da arquitetura modular do Java, compreendendo como o Jigsaw permite a criação de aplicações modulares para maior eficiência e controlo de dependências.

- Criação de módulos e gestão de dependências

Aprender a definir módulos com arquivos module-info.java e gerir dependências entre módulos de forma clara e eficiente.

- Integração modular com frameworks empresariais

Aplicação de conceitos de modularização em projetos reais, integrando com frameworks como Spring e Hibernate para construir aplicações robustas e modulares.

Lab: Building and Managing Modular Applications

- Exercise 1: Criar um projeto modular simples com o uso de module-info.java para definir um módulo e suas dependências.
- Exercise 2: Dividir uma aplicação existente em múltiplos módulos e reconfigurar as dependências para aproveitar os benefícios da modularização.
- Exercise 3: Integrar um projeto modular com um framework empresarial, como Spring, para criar uma aplicação modular e escalável.
- Exercise 4: Resolver problemas comuns na migração de aplicações não modulares para um formato modular, como conflito de versões e gerenciamento de acessos entre módulos.