Plano Integrado de Estudos: Desenvolvedor Java Completo + Arquitetura e Desenvolvimento Java

Objetivo: Consolidar uma trilha de estudos pratica e progressiva, unificando fundamentos de Java com arquitetura backend moderna e ferramentas profissionais de desenvolvimento.

---

Fase 1 Fundamentos Essenciais e POO com Java

Estudos:

Curso: Java Foundations: The Complete Course with Java 21 Updates (Udemy)

* Sintaxe basica, tipos primitivos, operadores.
* Controle de fluxo (if, for, switch, while).
* Funcoes e escopo de variaveis.
* Arrays e manipulacao basica.
* Orientacao a Objetos: classes, objetos, heranca, polimorfismo, encapsulamento, abstracao.
* Tratamento de excecoes, enums e generics.
* Principios SOLID e boas praticas de Clean Code.

Projetos praticos:

* Calculadora, conversor de moedas, sistema de login simples (console).
* Refatoracao de codigo aplicando OOP.
* Sistema de biblioteca com POO e SOLID (console).

---

Fase 2 Java Intermediario + Colecoes + Streams

Estudos:

Curso: Java Foundations: The Complete Course with Java 21 Updates (Udemy)

* Java Collections (List, Set, Map, Queue).
* Java Time API.
* Introducao a Streams e Lambdas.
* Java IO e NIO.

Projetos praticos:

* Agenda de contatos com leitura/escrita em arquivos.
* Filtros com Streams sobre listas de objetos (ex: busca de contatos).

---

Fase 3 Spring Boot, API REST, JPA e Banco de Dados

Estudos:

Cursos recomendados: Spring Boot Essentials (YouTube - DevDojo, Loiane), Alura Spring Boot e Spring Data JPA

* Spring Boot: injecao de dependencia, configuracao inicial.
* Spring MVC e criacao de APIs RESTful.
* Spring Data JPA com MySQL e H2.
* Documentacao com Swagger.
* Introducao ao MongoDB e Cassandra.
* Teorema CAP e bancos NoSQL.

Projetos praticos:

* Sistema de gerenciamento de tarefas (ToDo CRUD REST).
* Sistema financeiro pessoal (receitas, despesas, filtro por data).
* Versao com SQL e NoSQL.
* Versionamento de API e documentacao com Swagger.

---

Fase 4 Testes, Arquitetura Limpa e Boas Praticas

Estudos:

Cursos recomendados: JUnit + Mockito (Loiane, DevDojo), Alura Testes com Java, livros Clean Architecture

/ Clean Code

* JUnit 5, Mockito.
* Testes de integracao com H2.
* Testcontainers (opcional).
* Arquitetura Limpa (Clean Architecture).
* Logging com SLF4J, Logback.

Projetos praticos:

* Criar testes unitarios e de integracao para o projeto de tarefas ou financeiro.
* Refatoracao com Clean Architecture e cobertura minima de 80% nos testes.

---

Fase 5 Seguranca, APIs Modernas e Comunicacao entre Sistemas

Estudos:

Cursos recomendados: Spring Security (Loiane/DevDojo), GraphQL Java (graphql-java.com), gRPC com Java (Tech Primers)

* Spring Security: autenticacao, autorizacao, JWT, OAuth2.
* GraphQL com Spring Boot.
* gRPC com Java.
* Padroes de projeto: Factory, Singleton, Strategy.

Projetos praticos:

* Sistema de autenticacao com perfis (ADMIN/USER).
* Endpoint GraphQL para listagem de dados.
* Endpoint gRPC simulando outro sistema (promocoes, por exemplo).

---

Fase 6 Mensageria, Resiliencia e Concorrencia

Estudos:

Cursos recomendados: Kafka para Iniciantes (Confluent/Alura), Resilience4j com Spring (YouTube), concorrencia em Java (Alura/Loiane)

* Kafka e RabbitMQ.
* Resilience4j: Retry, Circuit Breaker, Fallback.
* Threads, Runnable, ExecutorService.
* CompletableFuture e programacao assincrona.

Projetos praticos:

* Simulador de pedidos com dois microsservicos (envio via Kafka e consumo com RabbitMQ).
* Aplicacao REST assincrona.

---

Fase 7 Deploy, Docker, Cloud e Serverless

Estudos:

Cursos recomendados: Docker para Devs (Alura, YouTube), Deploy com Azure/GCP, AWS Lambda com Java (YouTube + AWS Docs)

* Conceitos de cloud computing: IaaS, PaaS, SaaS.
* Docker: criacao de containers, Dockerfile, volumes, redes.
* Azure, GCP e AWS: App Services, bancos em nuvem.
* AWS Lambda e Azure Functions com Java.
* API Gateway.
* Design patterns para funcoes serverless.

Projetos praticos:

* Criar imagem Docker da aplicacao.
* Deploy da aplicacao containerizada na nuvem.
* Funcao serverless de notificacao por e-mail via API Gateway.

---

Fase 8 Microsservicos, Observabilidade e Arquiteturas Avancadas

Estudos:

Cursos recomendados: Spring Cloud (DevDojo, Alura), observabilidade com Prometheus/Grafana (YouTube), CQRS com Java (Medium + GitHub)

* Microsservicos com Spring Cloud: Eureka, API Gateway, Config Server.
* Comunicacao com Feign.
* Padroes CQRS e Event Sourcing.
* Ferramentas de observabilidade: Prometheus, Grafana, ELK Stack.

Projetos praticos:

* Sistema com 3 microsservicos (usuarios, produtos, pedidos).
* Separacao de comandos e queries (CQRS).
* Monitoramento com Prometheus e logs estruturados com ELK ou Loki.

---

Extras recomendados:

* Leitura: "Clean Code" (Robert C. Martin) e "Effective Java" (Joshua Bloch).
* Praticar desafios no HackerRank, LeetCode, CodeWars.
* Contribuir em projetos open source (mesmo com documentacao).
* Consolidar aprendizados em repositorio GitHub com README e historico de evolucao.