

AlgaWorks

BLACK

CONFIRA O CONTEÚDO COMPLETO DE CURSOS DA ASSINATURA DA ALGAWORKS

SUMÁRIO

Lógica de Programação com Java para Iniciantes	3
Banco de Dados e SQL para Iniciantes	6
Ignição Java	7
Ignição Spring REST	7
Especialista Java	9
Testes Unitários com JUnit	22
Principais Design Patterns Aplicados com Java	24
TDD Essencial	25
Especialista Spring REST	26
Especialista JPA	45
Java e Orientação a Objetos	51
Web Design Responsivo com HTML5, CSS3 e BEM	55
Mergulhando no JavaScript	56
Fullstack Angular e Spring	58
Especialista React	66
Desenvolvimento Web com JSF 2	77
Sistemas Comerciais Java EE com CDI, JPA e PrimeFaces	80
PrimeFaces Responsivo	85
Explorando a Linguagem JavaScript	86
Spring Framework Expert	88

INICIAÇÃO EM PROGRAMAÇÃO

Lógica de Programação com Java para Iniciantes

1. Introdução

- 1.1. Introdução ao curso de Lógica de Programação
 - 1.2. Introdução a computação
 - 1.3. O que é um algoritmo
 - 1.4. Linguagens de programação
 - 1.5. Como começar e terminar o curso
- ### **2. Começando**

2.1. Como o Java funciona?

- 2.2. Instalando o Eclipse
- 2.3. Criando o primeiro projeto no Eclipse
- 2.4. Executando um programa simples no Eclipse
- 2.5. Exercício 01: Alterando a mensagem de Olá
- 2.6. Elaborando um algoritmo no Eclipse
- 2.7. Exercício 02: Cálculo do índice de massa corporal

- 2.8. Como importar um projeto no Eclipse
- 2.9. Palavras reservadas do Java

3. Variáveis e constantes

- 3.1. Introdução a variáveis
- 3.2. Tipos de dados existentes
- 3.3. Tipos numéricos (Byte, Short, Integer, Long)
- 3.4. Exercício 01: Calculando o quadrado de um número
- 3.5. Tipo texto (String)
- 3.6. Exercício 02: Faça a impressão do nome e do sobrenome da pessoa
- 3.7. Tipo lógico (Boolean)
- 3.8. Exercício 03: Crie um programa que informa se o aluno passou de ano
- 3.9. Tipos primitivos
- 3.10. Alterando o valor de uma variável
- 3.11. Exercício 04: Encontre o valor total baseado na quantidade e no desconto
- 3.12. Constantes
- 3.13. Exercício 05: Constantes
- 3.14. Escolhendo bons nomes para variáveis e constantes

4. Operadores

- 4.1. Introdução a operadores
- 4.2. Operadores aritméticos
- 4.3. Exercício 01: Calculadora simples

- 4.4. Operadores relacionais
- 4.5. Exercício 02: Desconto de frete em compras acima de R\$100,00
- 4.6. Operadores de atribuição
- 4.7. Exercício 03: Cálculo de gastos familiar
- 4.8. Operadores lógicos
- 4.9. Exercício 04: Verificação da possibilidade de aposentadoria
- 4.10. Concatenação de texto
- 4.11. Operadores de incremento e decremento

5. Estruturas de decisão

- 5.1. Introdução a estruturas de decisão
- 5.2. Estrutura "if"
- 5.3. Exercício 01: Verificar se a pessoa passou no concurso público
- 5.4. Utilizando o "if" encadeado
- 5.5. Exercício 02: Calcule o bônus do funcionário(a)
- 5.6. Estrutura "switch"
- 5.7. Exercício 03: Imprima o nome do dia da semana (domingo, segunda-feira, etc.)

6. Iteração

- 6.1. Introdução a iteração
- 6.2. Iterando com o "for"
- 6.3. Exercício 01: Encontre os números divisíveis por 3
- 6.4. Iterando com o "while"
- 6.5. Exercício 02: Repita o exercício anterior com o "while"

7. Vetores

- 7.1. Introdução a vetores
- 7.2. Declarando vetores de 1 dimensão
- 7.3. Exercício 01: Cadastrando as tarefas do dia
- 7.4. Declarando vetores de 2 dimensões
- 7.5. Exercício 02: Encontre a semana de maior faturamento
- 7.6. Declarando vetores com mais de 2 dimensões

8. Métodos

- 8.1. Introdução a métodos
- 8.2. Criando o primeiro método
- 8.3. Exercício 01: Crie um segundo método no algoritmo da aula passada
- 8.4. Recebendo parâmetros
- 8.5. Exercício 02: Identifique um novo padrão no algoritmo da aula passada
- 8.6. Retornando valores
- 8.7. Exercício 03: Crie uma pequena calculadora de subtração e adição
- 8.8. Recursividade
- 8.9. Exercício 04: Exibir a tabuada de um número qualquer
- 8.10. Escolhendo bons nomes para métodos

9. Programação orientada a objetos

- 9.1. Introdução à programação orientada a objetos

- 9.2. Criando a primeira classe
- 9.3. Exercício 01: Método que informa necessidade de repor estoque
- 9.4. Métodos de instância
- 9.5. Exercício 02: Método de instância que informa necessidade de repor estoque
- 9.6. Encapsulamento
- 9.7. Exercício 03: Exibindo os dados de um pedido
- 9.8. Diferença entre classe e instância

10. Leitura e escrita de dados em arquivos

- 10.1. Introdução à leitura e escrita
- 10.2. Escrevendo informações em arquivos de texto
- 10.3. Exercício 01: Crie uma lista de tarefas e salve a lista em um arquivo
- 10.4. Lendo informações de um arquivo de texto
- 10.5. Exercício 02: Mostre a lista de tarefas para o usuário

11. Utilizando código de outros programadores

- 11.1. Introdução ao uso de códigos de terceiros
- 11.2. Bibliotecas x Frameworks
- 11.3. Utilizando uma biblioteca de terceiro para envio de e-mails
- 11.4. Exercício 01: Crie uma lista de tarefas e envie a lista por email
- 11.5. Criando o sua própria biblioteca

- 11.6. Exercício 02: Use nossa biblioteca

12. Algoritmos avançados

- 12.1. Introdução aos algoritmos avançados
- 12.2. Criando uma lista dinâmica
- 12.3. Removendo elementos da lista
- 12.4. Ordenando a lista
- 12.5. Exercício 01: Ordene a lista de alunos do arquivo

13. Pré-requisitos para nossa primeira aplicação web

- 13.1. Por onde começar a criar uma aplicação web
- 13.2. Evoluindo para o Spring Tool Suite (STS)
- 13.3. O que é HTML?
- 13.4. O que é CSS?
- 13.5. Criando um projeto com o STS
- 13.6. Incluindo as páginas HTML no projeto
- 13.7. Fluxo de uma requisição web

14. Criando sua primeira aplicação

- 14.1. Importando o projeto de base
- 14.2. Criando a classe Contato
- 14.3. Configurando o armazenamento de contatos
- 14.4. Listando os contatos na página

- 14.5. O padrão JavaBean
- 14.6. Fazendo a página de cadastro funcionar
- 14.7. Verbos HTTP
- 14.8. Cadastrando um contato
- 14.9. Preparando para edição
- 14.10. Atualizando um contato
- 14.11. Removendo contatos
- 14.12. Conclusão e próximos passos

Banco de Dados e SQL para Iniciantes

1. Iniciando com banco de dados

- 1.1. O que é banco de dados?
- 1.2. Instalando o MySQL
- 1.3. Tipos de dados
- 1.4. A famosa chave primária
- 1.5. Schemas, inserindo e consultando dados
- 1.6. Desafio: Iniciando um novo projeto

2. Relacionamentos

- 2.1. Por que mais de uma tabela?
- 2.2. A chave estrangeira
- 2.3. Relacionamento muitos-para-um

- 2.4. Relacionamento um-para-muitos
- 2.5. Desafio: Relacionando tabelas
- 2.6. Relacionamento muitos-para-muitos
- 2.7. Desafio: Relacionamento muitos-para-muitos

3. Consultas

- 3.1. Cláusula where
- 3.2. Relacionando várias tabelas em uma consultas
- 3.3. Trabalhando com datas
- 3.4. Desafio: Relacionando tabelas

4. Alterações

- 4.1. Alterando sua tabela
- 4.2. Trabalhando com índices
- 4.3. Deletando dados
- 4.4. Atualizando dados
- 4.5. Desafio: Alterando suas tabelas

5. Consultas avançadas

- 5.1. Função sum
- 5.2. Agrupando dados com group by
- 5.3. Função avg
- 5.4. Desafio: Funções e group by
- 5.5. Ordenando com order by
- 5.6. Filtrando com like

- 5.7. Resultados únicos com distinct
- 5.8. Filtrando com in
- 5.9. Consultas dentro de consultas - subselect
- 5.10. Desafio: subselect

Ignição Java

- 1. Mergulhando em Java e OO
 - 1.1. Introdução e preparação de ambiente
 - 1.2. Fundamentos da linguagem Java
 - 1.3. Orientação a objetos, classes e objetos
 - 1.4. Métodos, construtores e sobrecarga
 - 1.5. Encapsulamento e JavaBeans
 - 1.6. Pacotes
- 2. Orientação a objetos avançada
 - 2.1. Herança e sobrescrita de métodos
 - 2.2. Upcasting e polimorfismo
 - 2.3. Classes abstratas
 - 2.4. Interfaces
 - 2.5. Exceções
 - 2.6. Boxing, enumerações, datas e valores monetários
- 3. Coleções, Lambdas e Streams API
 - 3.1. Collections Framework

- 3.2. Interfaces funcionais e expressões lambda
- 3.3. Streams API
- 3.4. Optional
- 3.5. Conclusão

Ignição Spring REST

- 1. Fundamentos de REST e Spring
 - 1.1. Boas vindas e as oportunidades do mercado
 - 1.2. Quem é você? Quem sou eu?
 - 1.3. Alguns combinados antes de continuar
 - 1.4. O que é uma API?
 - 1.5. O que é REST?
 - 1.6. Conhecendo o protocolo HTTP
 - 1.7. Entendendo os Recursos REST
 - 1.8. Identificando recursos REST
 - 1.9. Por que Spring?
 - 1.10. Conhecendo o ecossistema de projetos Spring
 - 1.11. Estudos de caso
- 2. Construindo uma REST API
 - 2.1. Preparando o ambiente de desenvolvimento
 - 2.2. Conhecendo o modelo de domínio do projeto
 - 2.3. Alternativas para criar projetos Spring Boot

- 2.4. Criando o projeto com Spring Initializr
- 2.5. Entendendo a estrutura do projeto Maven
- 2.6. Gerando o FatJAR e iniciando a aplicação
- 2.7. Implementando e testando a requisição de um recurso
- 2.8. Implementando uma Collection Resource
- 2.9. Configurando e usando o Lombok
- 2.10. Métodos e códigos de status HTTP
- 2.11. Content Negotiation
- 2.12. Turbinando a produtividade com DevTools

3. Persistindo os dados

- 3.1. Configurando a conexão com o banco de dados no projeto
- 3.2. Conhecendo e adicionando o Flyway no projeto
- 3.3. Criando a primeira migration com Flyway
- 3.4. Conhecendo o Jakarta Persistence (JPA)
- 3.5. Mapeando entidades com Jakarta Persistence
- 3.6. Implementando uma consulta com JPQL
- 3.7. Conhecendo o Spring Data JPA (SDJ) e criando um repositório
- 3.8. Injetando e usando o repositório do SDJ
- 3.9. Implementando Query Methods no repositório
- 3.10. Implementando endpoint de busca de recurso
- 3.11. Implementando endpoint de inclusão de recurso
- 3.12. Implementando endpoint de atualização de recurso
- 3.13. Implementando endpoint de exclusão de recurso

4. Evoluindo a API

- 4.1. Conhecendo e adicionando Jakarta Bean Validation no projeto
- 4.2. Validando entrada de dados com Jakarta Bean Validation
- 4.3. Implementando Domain Services
- 4.4. Implementando regra de negócio para restringir e-mails duplicados
- 4.5. Capturando exceções do controlador com `@ExceptionHandler`
- 4.6. Adicionando migration para criação da tabela de veículos
- 4.7. Criando e mapeando a entidade de veículo
- 4.8. Implementando os endpoints de consulta de veículos
- 4.9. Implementando o endpoint de inclusão de veículos
- 4.10. Implementando regras de negócio no cadastro de veículos
- 4.11. Protegendo propriedades somente-leitura
- 4.12. Validando em cascata
- 4.13. Validando com Validation Groups

5. Aplicando as boas práticas

- 5.1. Capturando exceções globais com `@RestControllerAdvice`
- 5.2. Usando a RFC 7807 (Problem Details for HTTP APIs)
- 5.3. Customizando as informações do ProblemDetail
- 5.4. Adicionando campos customizados no ProblemDetail
- 5.5. Customizando as mensagens de validação
- 5.6. Tratando exceções customizadas de forma global
- 5.7. Boas práticas para trabalhar com data/hora
- 5.8. Isolando o Domain Model do Representation Model
- 5.9. Criando o Representation Model do recurso de veículo

- 5.10. Transformando objetos com ModelMapper
- 5.11. Implementando assembler de Representation Model
- 5.12. Compondo objetos no Representation Model
- 5.13. Criando um Representation Model para entrada de dados

6. Modelando sub-recursos e ações não-CRUD

- 6.1. Criando e mapeando a entidade de autuação
- 6.2. Implementando serviço de autuação
- 6.3. Modelando sub-recursos
- 6.4. Implementando o endpoint de cadastro do recurso de autuação
- 6.5. Especializando a exceção de entidade não encontrada
- 6.6. Implementando recurso de coleção de autuações
- 6.7. Implementando regras de negócio de apreensão de veículo
- 6.8. Modelando ações não-CRUD
- 6.9. Implementando endpoints de ações não-CRUD
- 7. Alcançando o próximo nível
 - 7.1. Traçando um plano

ESPECIALIZAÇÃO EM JAVA E PADRÕES

Especialista Java

1. Plataforma Java e ambiente de desenvolvimento

- 1.1. Introdução ao curso
- 1.2. Como aprender Java e pedir ajuda
- 1.3. Por que aprender Java?
- 1.4. Um pouco sobre a história do Java
- 1.5. Conhecendo as plataformas Java
- 1.6. Conhecendo a Máquina Virtual Java (JVM)
- 1.7. JRE e JDK: qual é a diferença?
- 1.8. Conhecendo as versões do Java
- 1.9. Conhecendo as distribuições de JDKs e licenças de uso
- 1.10. Instalando o JDK no Ubuntu e macOS com SDKMan!
- 1.11. Instalando o JDK no Windows

1.12. Escolhendo um editor de código simples

2. Fundamentos da linguagem Java

- 2.1. Criando o primeiro programa Java
- 2.2. Compilando e executando um programa Java
- 2.3. Desafio: correção de erros
- 2.4. Escrevendo comentários no código
- 2.5. Conhecendo e usando convenções de código
- 2.6. Palavras reservadas
- 2.7. Trabalhando com variáveis
- 2.8. Operadores aritméticos
- 2.9. Desafio: variáveis e operadores aritméticos
- 2.10. Abreviando operadores aritméticos
- 2.11. Operadores de incremento e decremento
- 2.12. Tipos primitivos: boolean, char, byte e short
- 2.13. Tipos primitivos: int e long
- 2.14. Tipos primitivos: float e double
- 2.15. Conversão de tipos primitivos
- 2.16. Desafio: tipos primitivos e conversão
- 2.17. Promoção aritmética
- 2.18. Desafio: promoção aritmética
- 2.19. Trabalhando com String
- 2.20. Usando sequências de escape
- 2.21. Formatando a saída com printf
- 2.22. Recebendo entrada de dados
- 2.23. Desafio: String, entrada de dados, printf, etc
- 2.24. Usando JShell: o REPL do Java

3. Estruturas de controle e operadores

- 3.1. Operadores de igualdade e de negação (unário)
- 3.2. Operadores de comparação
- 3.3. Operadores lógicos
- 3.4. Desafio: operadores de igualdade e lógicos
- 3.5. Curto-circuito de operadores lógicos
- 3.6. Precedência de operadores lógicos
- 3.7. Estrutura condicional if
- 3.8. Estruturas condicionais else e else if
- 3.9. Desafio: calculadora complexa de IMC
- 3.10. Escopos e inicialização de variáveis
- 3.11. Estrutura condicional switch
- 3.12. Cláusula break
- 3.13. Switch Expressions
- 3.14. Operador ternário
- 3.15. Desafio: estrutura switch e operador ternário
- 3.16. Estrutura de repetição for
- 3.17. Estrutura de repetição while
- 3.18. Estrutura de repetição do/while
- 3.19. Cláusulas break e continue
- 3.20. Desafio: estruturas de repetição

4. Produtividade com a IDE IntelliJ IDEA

- 4.1. Conhecendo as IDEs mais populares
- 4.2. Instalando e conhecendo a IntelliJ IDEA

ALGAWORKS BLACK

- 4.3. Mais da IntelliJ IDEA: build, run, plugins, terminal e shared index
- 4.4. Usando Code Completion, Live Templates e Postfix Completion
- 4.5. Conhecendo os principais atalhos
- 4.6. Mais atalhos do IntelliJ IDEA
- 4.7. Usando o Debugger para depurar o seu código
- 4.8. Debugger: silenciamento, condição e desativação de breakpoints
- 4.9. Debugger: gerenciando variáveis e avaliando expressões
- 4.10. Debugger: watches e logging
- 4.11. Rascunhando e testando código com Scratch Files
- 4.12. Testando código com JShell Console da IDE
- 4.13. Consistência no estilo de codificação com EditorConfig
- 4.14. Importando um projeto existente na IDE

5. Mergulhando em orientação a objetos

- 5.1. O paradigma da Programação Orientada a Objetos (POO)
- 5.2. Entendendo o conceito de classes e objetos
- 5.3. Criando uma classe com atributos
- 5.4. Instanciando objetos
- 5.5. Acessando atributos de objetos
- 5.6. Conhecendo o diagrama de classes da UML
- 5.7. Desafio: instanciando objetos e acessando os atributos
- 5.8. Composição de objetos
- 5.9. Atribuindo o objeto na composição
- 5.10. Diagrama de classes: associação, agregação e composição
- 5.11. Valores padrão e inicialização de variáveis de instância

- 5.12. Inicialização de objetos em variáveis de instância
- 5.13. Caindo a ficha: variáveis referenciam objetos
- 5.14. Criando e invocando um método
- 5.15. Implementando a lógica do método
- 5.16. IntelliJ IDEA: debug de chamadas de métodos
- 5.17. Métodos com retorno
- 5.18. Implementando métodos menores e evitando duplicação de código
- 5.19. Saindo do método com a cláusula return
- 5.20. Métodos que retornam objetos
- 5.21. Refatorando para tornar uma classe mais rica
- 5.22. Discutindo nome e responsabilidade da classe
- 5.23. Métodos com argumentos
- 5.24. Passando objetos como argumentos de métodos
- 5.25. Desafio: composição de objetos e métodos
- 5.26. Diagrama de classes: métodos e dependências
- 5.27. Métodos que alteram variável de instância
- 5.28. Métodos que alteram o valor de parâmetro do tipo primitivo
- 5.29. Métodos que alteram o estado de objeto recebido como parâmetro
- 5.30. Usando a palavra-chave this
- 5.31. Atributos de classe com o modificador static
- 5.32. Método de instância alterando variável estática
- 5.33. Métodos de classe (estáticos)
- 5.34. Método estático acessando membro de instância
- 5.35. Desafio: membros estáticos
- 5.36. Declarando constantes com static e final
- 5.37. Modificador final em variáveis locais

- 5.38. Sobrecarga de métodos
- 5.39. Inferência de tipo de variável local
- 5.40. Desafio: modificador final em variáveis locais
- 5.41. Desafio: sobrecarga de métodos
- 5.42. Desafio: inferência de tipo de variável local

6. Começando com boas práticas e código limpo

- 6.1. Boas práticas com Effective Java e Clean Code
- 6.2. Código Limpo: escolha bons nomes
- 6.3. Código Limpo: tamanho e organização de classes
- 6.4. Código Limpo: comentários no código
- 6.5. Código Limpo: métodos pequenos e que fazem só uma coisa
- 6.6. Código Limpo: pensando melhor nos argumentos de métodos
- 6.7. Boas práticas: valide os argumentos

7. Wrappers e boxing

- 7.1. Usando classes wrapper
- 7.2. Métodos de conversão
- 7.3. Autoboxing e unboxing
- 7.4. Comparando wrappers
- 7.5. Desafio: wrappers e boxing
- 7.6. Boas práticas: prefira tipos primitivos a wrappers

8. Trabalhando com arrays

- 8.1. Declarando e inicializando arrays
- 8.2. Acessando e atribuindo elementos de arrays
- 8.3. Iterando em arrays
- 8.4. Transformando arrays em representações em string
- 8.5. Ordenando arrays em ordem natural e reversa
- 8.6. Criando arrays de objetos
- 8.7. Iterando em arrays de objetos
- 8.8. Copiando e expandindo arrays
- 8.9. Removendo elementos de arrays
- 8.10. Desafio: arrays
- 8.11. Um pouco da ArrayList da Collections API
- 8.12. Desafio: ArrayList
- 8.13. Diagrama de classes: multiplicidade para arrays
- 8.14. Boas práticas: retorne arrays ou coleções vazias no lugar de null
- 8.15. Criando arrays multidimensionais
- 8.16. Iterando em arrays multidimensionais
- 8.17. Lendo os parâmetros da linha de comando
- 8.18. Criando métodos com argumentos variáveis com Varargs
- 8.19. Boas práticas: use varargs com cuidado
- 8.20. Desafio: varargs

9. Gerenciamento de memória do Java

- 9.1. Estrutura da memória da JVM
- 9.2. Call Stack, Stack Memory e Heap Memory

- 9.3. Informações da Memória Heap com a Runtime API
- 9.4. Configurando a Memória Heap da JVM
- 9.5. Garbage Collector
- 9.6. Inalcançabilidade de objetos
- 9.7. Quando esgota a Memória Heap: OutOfMemoryError
- 9.8. Boas práticas: remova referências de objetos não usados

10. Construtores, pacotes e visibilidade

- 10.1. Criando e chamando construtores
 - 10.2. Construtores com parâmetros
 - 10.3. Sobrecarga de construtores
 - 10.4. Boas práticas: valide os argumentos de construtores também
 - 10.5. Encadeamento de chamadas de construtores
 - 10.6. Diagrama de classes: construtores
 - 10.7. Desafio: construtores
 - 10.8. Modificador final em variáveis de instância
 - 10.9. Organizando as classes em pacotes
 - 10.10. Importando classes de pacotes
 - 10.11. Modificador de acesso public e default
 - 10.12. Modificador de acesso private
 - 10.13. Diagrama de classes: visibilidade public, package e private
 - 10.14. Desafio: pacotes e modificadores de acesso
 - 10.15. Importando membros estáticos (static import)
 - 10.16. Múltiplas classes não-públicas em um único arquivo
 - 10.17. Conhecendo a documentação JavaDoc do Java SE11.
- Encapsulamento, JavaBeans e Records
- 11.1. O problema da falta de encapsulamento

- 11.2. Implementando encapsulamento
- 11.3. JavaBeans e métodos getters e setters
- 11.4. IntelliJ IDEA: gerando getters e setters
- 11.5. Desafio: encapsulamento e JavaBeans
- 11.6. Boas práticas: use métodos de acesso em classes públicas (incluindo Tell Don't Ask)
- 11.7. Código limpo: Lei de Demeter (incluindo Train Wreck)
- 11.8. Boas práticas: não permita instanciação com construtor privado
- 11.9. Boas práticas: crie cópias defensivas
- 11.10. Boas práticas: minimize a mutabilidade (incluindo Value Object)
- 11.11. Records
- 11.12. Diagrama de classes: Records

12. Herança

- 12.1. Conhecendo o projeto deste módulo
- 12.2. Criando classes etiquetadas (tagged classes)
- 12.3. Duplicando classes e isolando os comportamentos
- 12.4. Conhecendo herança e o relacionamento no diagrama de classes
- 12.5. Implementando herança
- 12.6. Sobrescrita de métodos
- 12.7. Modificador de acesso protected
- 12.8. Anotação @Override
- 12.9. Chamando método da superclasse com super
- 12.10. A classe Object

- 12.11. Invocando construtores da superclasse
- 12.12. Criando construtores com parâmetros na superclasse e subclasses
- 12.13. Boas práticas: sempre sobrescreva o método Object.toString
- 12.14. Modificador final em classes e métodos
- 12.15. Desafio: implementando herança
- 12.16. Sobrescrevendo o método Object.equals

13. Polimorfismo e classes abstratas

- 13.1. Upcasting de referências
- 13.2. O problema que polimorfismo resolve
- 13.3. Entendendo o polimorfismo
- 13.4. Downcasting de referências
- 13.5. Operador instanceof
- 13.6. Pattern Matching para o operador instanceof
- 13.7. Evitando o uso de instanceof
- 13.8. Criando um projeto de faturamento
- 13.9. Classes abstratas
- 13.10. Métodos abstratos
- 13.11. Desafio: polimorfismo e classes abstratas

14. Interfaces

- 14.1. Entendendo as interfaces
- 14.2. Criando a primeira interface
- 14.3. Implementando a primeira interface
- 14.4. Nova interface e injeção de dependências

- 14.5. Conhecendo o projeto da financeira
- 14.6. Quando herança de classes se torna um problema
- 14.7. Código mais flexível: refatorando para usar interfaces
- 14.8. Métodos default em interfaces
- 14.9. Classes abstratas com interfaces
- 14.10. Métodos privados em interfaces
- 14.11. Métodos estáticos em interfaces
- 14.12. Variáveis são estáticas e finais em interfaces
- 14.13. Implementando múltiplas interfaces
- 14.14. Herança de interfaces
- 14.15. Desafio: interfaces

15. Boas práticas de herança e interfaces

- 15.1. Rigidez do código com herança
- 15.2. Boas práticas: prefira composição em vez de herança de classes
- 15.3. Código frágil: alto acoplamento com herança
- 15.4. Boas práticas: reduzindo acoplamento com composição
- 15.5. Decorator Pattern: consolidando os conhecimentos
- 15.6. Boas práticas: projete interfaces com cuidado
- 15.7. Boas práticas: use interfaces apenas para definir tipos
- 15.8. Boas práticas: referencie objetos por suas interfaces

16. Exceções

- 16.1. Introdução às exceções

ALGAWORKS BLACK

- 16.2. Lançando exceções
- 16.3. Stack Trace: interpretando e analisando exceções
- 16.4. Capturando exceções com try/catch
- 16.5. Relançando exceções e printStackTrace
- 16.6. Capturando exceções com múltiplos blocos catch
- 16.7. Hierarquia das exceções, checked e unchecked exceptions
- 16.8. Capturando checked exceptions
- 16.9. Criando exceções customizadas
- 16.10. Variáveis de instância em exceções customizadas
- 16.11. Lançando e propagando checked exceptions
- 16.12. Capturando exceções menos específicas (upcasting)
- 16.13. Capturando e lançando nova exceção
- 16.14. Boa prática: embrulhe a causa raiz
- 16.15. Capturando exceções com multi-catch
- 16.16. Usando a cláusula finally
- 16.17. IntelliJ IDEA: lançando exceções na ferramenta de debug
- 16.18. IntelliJ IDEA: adicionando Java Exception Breakpoints
- 16.19. Boas práticas: lance exceções ao invés de retornar null
- 16.20. Boas práticas: não engula exceções
- 16.21. Desafio: exceções

17. Generics

- 17.1. Introdução aos Generics
- 17.2. Implementando métodos genéricos
- 17.3. Delimitando tipos genéricos
- 17.4. Criando classes genéricas
- 17.5. Criando interfaces genéricas

- 17.6. Usando curingas para tipos desconhecidos
- 17.7. Desafio: Generics

18. Collections Framework

- 18.1. Por que mais uma API?
- 18.2. Introdução às listas e ao tipo List
- 18.3. Como funciona a ArrayList
- 18.4. Usando listas do tipo ArrayList
- 18.5. Iterando em lista com for tradicional
- 18.6. Usando listas com Generics
- 18.7. Localizando objetos em listas
- 18.8. Manipulando objetos da lista
- 18.9. Percorrendo a lista com Iterator
- 18.10. Percorrendo a lista com ListIterator
- 18.11. Percorrendo Iterables com enhanced for
- 18.12. LinkedList: mais performance na adição e remoção
- 18.13. Usando listas do tipo LinkedList
- 18.14. Vector: a lista thread-safe
- 18.15. Boas práticas: reduza o acoplamento usando o tipo da interface
- 18.16. Convertendo de lista para array
- 18.17. Ordenando listas pela ordem natural
- 18.18. Boas práticas: considere implementar a interface Comparable
- 18.19. Comparators: ordenando listas com outros critérios
- 18.20. Desafio: listas
- 18.21. Introdução aos conjuntos e ao tipo Set

- 18.22. Usando conjuntos do tipo HashSet
- 18.23. Tabelas de espalhamento (hash tables)
- 18.24. Implementando o método hashCode
- 18.25. Testando a implementação de hashCode com HashSet
- 18.26. Usando conjuntos do tipo TreeSet
- 18.27. Usando conjuntos do tipo LinkedHashMap
- 18.28. Desafio: conjuntos
- 18.29. Introdução aos mapas e ao tipo Map
- 18.30. Usando mapas dos tipos HashMap e Hashtable
- 18.31. Linked HashTable e TreeMap: outras implementações de mapas
- 18.32. Desafio: mapas
- 18.33. Boas práticas: encapsulamento com coleções não-modificáveis
- 18.34. Coleções imutáveis
- 18.35. Usando a API de List para manipular arrays

19. Enumerações

- 19.1. Usando enumerações à moda antiga
- 19.2. Criando tipos Enum
- 19.3. Diagrama de classes: enumeração
- 19.4. Usando os métodos do tipo Enum
- 19.5. Declarando e inicializando propriedades e construtores
- 19.6. Implementando métodos
- 19.7. Implementando métodos abstratos
- 19.8. Boas práticas: substitua parâmetros booleanos por enums
- 19.9. Desafio: enumerações

20. Trabalhando com strings

- 20.1. Comparação de strings
- 20.2. Pool de strings
- 20.3. Validando o conteúdo de strings
- 20.4. Extraindo trechos da String com indexOf e substring
- 20.5. Extraindo trechos da String com lastIndexOf e substring
- 20.6. Transformando strings
- 20.7. Implementando algoritmos usando os métodos da classe String
- 20.8. Desafio: validação de e-mail
- 20.9. Testando correspondências de strings com expressões regulares
- 20.10. Web Scraping: buscando correspondências com Pattern e Matcher
- 20.11. Manipulando strings com expressões regulares
- 20.12. Boas práticas: use StringBuilder para mais performance
- 20.13. Código mais limpo com Text blocks
- 20.14. Desafio: Text blocks e expressões regulares

21. Trabalhando com números

- 21.1. Comparando números da forma correta
- 21.2. Caching de classes wrapper
- 21.3. Operações matemáticas com java.lang.Math
- 21.4. Boas práticas: evite float e double se precisão é importante
- 21.5. Precisão nos cálculos com BigDecimal
- 21.6. Divisão de BigDecimal e formas de arredondamento

- 21.7. Formatação decimal com DecimalFormat
- 21.8. Localizando a formatação de números com Locale
- 21.9. Formatação numérica compacta
- 21.10. Transformando String em números com DecimalFormat
- 21.11. Desafio: formatação numérica

22. Date e Calendar: as APIs legadas

- 22.1. Entendendo os fusos horários
- 22.2. Instanciando datas com o tipo Date
- 22.3. Calculando e comparando datas com Date
- 22.4. Formatando Date para String
- 22.5. Convertendo de String para Date
- 22.6. Conhecendo o tipo Calendar
- 22.7. Obtendo unidades de tempo e atribuindo uma Date em Calendar
- 22.8. Operações de datas com o tipo Calendar
- 22.9. Comparando datas com o tipo Calendar
- 22.10. Desafio: calculando datas com Calendar

23. Date/Time API: mais nova e moderna

- 23.1. Introdução à Date and Time API e ao padrão ISO-8601
- 23.2. Instant: representando um momento na linha do tempo
- 23.3. LocalDate: representando apenas a data
- 23.4. LocalTime: representando apenas o horário
- 23.5. LocalDateTime: representando data e hora

- 23.6. Outras formas de obter instâncias de LocalDate, LocalTime e LocalDateTime
- 23.7. Formatando data/hora da nova API
- 23.8. Convertendo de String para objetos temporais
- 23.9. Desafio: LocalDate, LocalTime e LocalDateTime
- 23.10. Obtendo campos de objetos temporais e a enum ChronoField
- 23.11. Alterando campos de objetos temporais com métodos with
- 23.12. Adicionando e subtraindo objetos temporais com métodos de conveniência
- 23.13. Calculando objetos temporais com Chrono Unit
- 23.14. Desafio: calculadora de parcelas
- 23.15. Representando períodos com a classe Period
- 23.16. Calculando períodos de objetos temporais
- 23.17. Representando durações com a classe Duration
- 23.18. Calculando durações de objetos temporais
- 23.19. Desafio: calculando período
- 23.20. DayOfWeek: representando o dia da semana
- 23.21. Year: representando o ano
- 23.22. Month: representando o mês
- 23.23. Month Day: representando o dia do mês
- 23.24. Year Month: representando o mês do ano
- 23.25. Alterando campos de objetos temporais com TemporalAdjusters
- 23.26. Comparando objetos temporais
- 23.27. Desafio: TemporalAdjuster
- 23.28. Identificando fusos com ZoneId e ZoneOffset

- 23.29. Instanciando objetos temporais em um fuso horário específico
- 23.30. ZonedDateTime: data/hora com fuso horário
- 23.31. Calculando e convertendo de/para ZonedDateTime
- 23.32. Offset DateTime em Offset Time: data e hora com deslocamento do UTC
- 23.33. Desafio: trabalhando com fuso horário

24. Classes aninhadas

- 24.1. Introdução às classes aninhadas
- 24.2. Classes aninhadas estáticas
- 24.3. Modificadores de acesso de classes aninhadas
- 24.4. Enum como membro estático de uma classe
- 24.5. Classes aninhadas não-estáticas
- 24.6. Shadowing em classes aninhadas
- 24.7. Classes locais
- 24.8. Classes anônimas
- 24.9. Desafio: classes aninhadas

25. Expressões Lambda e Method Reference

- 25.1. Introdução ao módulo
- 25.2. Implementando Expressões Lambda
- 25.3. Entendendo as interfaces funcionais
- 25.4. Usando a interface @FunctionalInterface
- 25.5. Boas práticas: prefira lambdas a classes anônimas
- 25.6. Boas práticas: torne as lambdas mais concisas

- 25.7. Implementando Comparator com lambda
- 25.8. Boas práticas: prefira interfaces funcionais padrão
- 25.9. As 4 principais interfaces funcionais
- 25.10. Interface funcional Predicate: removendo elementos de coleções
- 25.11. Interface funcional Consumer: iterando em coleções com forEach
- 25.12. Interface funcional Function: ordenando lista com Comparator.comparing
- 25.13. Usando Method References
- 25.14. Referenciando métodos de uma instância particular
- 25.15. Referenciando métodos estáticos
- 25.16. Referenciando construtores
- 25.17. Desafio: expressões lambda e method reference

26. Optional

- 26.1. O jeito tradicional de evitar NPE
- 26.2. Evoluindo seu código com Optional
- 26.3. Testando valor do Optional com isPresent
- 26.4. Obtendo valor e lançando exceções com orElseThrow
- 26.5. Obtendo valor alternativo com orElse e orElseGet
- 26.6. Obtendo e testando valor com ifPresent e ifPresentOrElse
- 26.7. Testando e filtrando valor com Predicate
- 26.8. Aplicando transformações com map
- 26.9. Aplicando transformações com flatMap
- 26.10. Tipos especiais de Optional para tipos primitivos
- 26.11. Boas práticas ao usar Optional

26.12. Desafio: Optional

27. Streams API

- 27.1. Introdução à Streams API e operações básicas
- 27.2. Operação intermediária: Stream.filter
- 27.3. Operação terminal: Stream.forEach
- 27.4. Criando o pipeline com encadeamento de operações
- 27.5. Executando ações intermediárias com o método Stream.peek
- 27.6. Operações terminais de curto-circuito: findFirst e findAny
- 27.7. Testando predicados com Stream.anyMatch, Stream.allMatch e Stream.noneMatch
- 27.8. Ordenando elementos de Streams
- 27.9. Entendendo o que é uma operação intermediária com estado (stateful)
- 27.10. Aplicando transformações com Stream.map
- 27.11. Obtendo um Stream de elementos distintos
- 27.12. Achatando um Stream com Stream.flatMap
- 27.13. Usando as especializações de Stream para tipos primitivos
- 27.14. Entendendo as operações de redução com Stream.reduce
- 27.15. Reduzindo em BigDecimal e usando a função de combinação
- 27.16. Operações de redução que retornam Optional
- 27.17. Operações de redução especiais: sum, average e count
- 27.18. Operações de redução especiais: min e max
- 27.19. Coletando elementos do Stream em lista com Stream.collect
- 27.20. Usando coletores padrão da classe Collectors
- 27.21. Usando coletores de listas não-modificáveis

- 27.22. Coletando elementos do Stream em mapas
- 27.23. Gerando mapas agrupados com Collectors.groupingBy
- 27.24. Gerando mapas agrupados com valores agregados
- 27.25. Gerando mapas particionados com Collectors.partitioningBy
- 27.26. Outras formas de obter instâncias de Stream
- 27.27. Métodos Objects.isNull e Objects.nonNull
- 27.28. Boas práticas: prefira funções em streams sem efeito colateral
- 27.29. Desafio: Streams

28. Manipulando arquivos com a API clássica de I/O

- 28.1. Introdução à API clássica de I/O
- 28.2. Instanciando e criando arquivos e pastas com a classe File
- 28.3. Obtendo o caminho absoluto e canônico de File
- 28.4. Excluindo, renomeando e movendo arquivos e pastas
- 28.5. Obtendo informações de arquivos e diretórios
- 28.6. Listando arquivos e diretórios
- 28.7. Entendendo I/O streams e Byte-oriented streams
- 28.8. Lendo arquivos com FileInputStream
- 28.9. Boa prática: tratando IOException com try-with-resources
- 28.10. Escrevendo arquivos com FileOutputStream
- 28.11. Conhecendo Character-oriented streams
- 28.12. Lendo arquivos de texto com FileReader
- 28.13. Escrevendo arquivos de texto com FileWriter
- 28.14. Lendo arquivos texto de forma otimizada com BufferedReader

- 28.15. Escrevendo arquivos texto de forma otimizada com `BufferedWriter`
- 28.16. Reconhecendo a API de I/O em `System.in` e `Scanner`
- 28.17. Reconhecendo a API de I/O em `System.out` e a classe `PrintStream`
- 28.18. Desafio: API clássica de I/O

29. Manipulando arquivos com NIO.2

- 29.1. Introdução ao NIO é NIO.2
- 29.2. Representando arquivos e pastas com a classe `Path`
- 29.3. Trabalhando com caminhos absolutos e relativos
- 29.4. Operações básicas com a classe `Files`
- 29.5. Copiando arquivos e diretórios
- 29.6. Movendo arquivos e diretórios
- 29.7. Excluindo arquivos e diretórios
- 29.8. Realizando operações com `Files.walkFileTree`
- 29.9. Obtendo informações de arquivos e diretórios
- 29.10. Listando conteúdo de diretórios
- 29.11. Pesquisando arquivos em uma pasta e subpastas
- 29.12. Entendendo os buffers e usando `ByteBuffer`
- 29.13. Usando `CharBuffer`
- 29.14. Decodificando `ByteBuffer` em `CharBuffer`
- 29.15. Lendo arquivos com `ByteChannel`
- 29.16. Lendo arquivos com buffers menores
- 29.17. Escrevendo arquivos com `ByteChannel`
- 29.18. Usando a API de I/O clássica com implementações da NIO
- 29.19. Simplificando a leitura de arquivos com a classe `Files`

- 29.20. Simplificando a escrita de arquivos com a classe `Files`
- 29.21. Desafio: NIO.2

30. Serialização de objetos

- 30.1. Introdução à serialização de objetos
- 30.2. Tornando classes serializáveis com a interface `Serializable`
- 30.3. Serializando objetos com `ObjectOutputStream`
- 30.4. Desserializar objetos com `ObjectInputStream`
- 30.5. Ignorando propriedades com `transient`
- 30.6. Entendendo e gerando `serialVersionUID`
- 30.7. Boas práticas de serialização e `serialVersionUID`
- 30.8. Desafio: serialização de objetos

31. Arquivos JAR e Apache Maven

- 31.1. O que são os arquivos JAR
- 31.2. Gerando arquivos JAR como bibliotecas
- 31.3. Importando arquivos JAR no projeto
- 31.4. Usando bibliotecas parceiras
- 31.5. Gerando arquivos JAR executável
- 31.6. O que é Apache Maven?
- 31.7. Instalando o Apache Maven no Windows
- 31.8. Instalando o Apache Maven no macOS e Linux
- 31.9. Criando um projeto Maven com IntelliJ IDEA
- 31.10. Arquivo `pom.xml` e Maven Coordinates
- 31.11. Entendendo o Standard Directory Layout
- 31.12. Compilando e empacotando com Maven

- 31.13. Conhecendo os tipos de repositórios Maven
- 31.14. Instalando e adicionando dependências com Maven
- 31.15. Usando dependência do Maven Central Repository
- 31.16. Entendendo as dependências transitivas
- 31.17. Entendendo os plugins do Maven e o Super POM
- 31.18. Gerando Fat JAR com Maven Assembly Plugin

32. Java Logging API, Logback e SLF4J

- 32.1. Por que fazer logging?
- 32.2. Principais frameworks de logging
- 32.3. Usando o Java Logging API (JUL)
- 32.4. Conhecendo os níveis de log do JUL
- 32.5. Registrando um log de exceção com JUL
- 32.6. Criando logging.properties e configurando nível de log
- 32.7. Salvando logs em arquivos com FileHandler do JUL
- 32.8. Usando a Java Logging API com SLF4J
- 32.9. Usando o Logback com SLF4J
- 32.10. Conhecendo os níveis de log do Logback e SLF4J
- 32.11. Configurando a saída de logs com logback.xml
- 32.12. Customizando mais o padrão de layout do Encoder
- 32.13. Salvando logs em arquivos com FileAppender do Logback
- 32.14. Desafio: Logback e SLF4J

33. Banco de dados e JDBC

- 33.1. Introdução ao MySQL
- 33.2. Instalando o MySQL Server

- 33.3. Instalando o MySQL Workbench
- 33.4. Criando uma tabela no banco de dados
- 33.5. Introdução ao JDBC
- 33.6. Adicionando driver JDBC ao projeto e criando uma conexão
- 33.7. Executando consultas SQL com Statement
- 33.8. Realizando consultas SQL com a cláusula where
- 33.9. Executando consultas SQL com PreparedStatement
- 33.10. Executando comandos que alteram dados (DML)
- 33.11. Obtendo código de autoincremento gerado
- 33.12. Gerenciando transações com o banco de dados

34. Padrão de projeto: Repository

- 34.1. Introdução ao design de software
- 34.2. Criando entidade e camada de serviço
- 34.3. Conhecendo o pattern Repository
- 34.4. Implementando uma classe Repository
- 34.5. Implementando uma consulta no repositório
- 34.6. Desacoplando a conexão com o banco de dados
- 34.7. Implementando uma fábrica de Repository
- 34.8. Desacoplando a implementação de Repository
- 34.9. Alternando a implementação do Repository
- 34.10. Abstraindo a fábrica de repositórios
- 34.11. Criando um arquivo de configuração (properties)

35. Reflection API, anotações e classes seladas

- 35.1. Conhecendo o projeto de gerador de CSV
- 35.2. Lendo propriedades dinamicamente com Reflection API
- 35.3. Invocando a leitura de valores de propriedades
- 35.4. Introdução às anotações do Java
- 35.5. Criando uma anotação customizada
- 35.6. Processando anotações com Reflection API
- 35.7. Adicionando e processando atributos na anotação
- 35.8. Classes seladas
- 35.9. Conclusão e próximos passos

Testes Unitários com JUnit

1. Introdução

- 1.1. Introdução ao treinamento
- 1.2. Como usar o suporte
- 1.3. Pirâmide de testes
- 1.4. A importância dos testes unitários
- 1.5. O princípio FIRST
- 1.6. Introdução ao Maven
- 1.7. Instalando a IntelliJ no Windows
- 1.8. Instalando a IntelliJ no Linux
- 1.9. Instalando o JDK do Java no Windows
- 1.10. Instalando o JDK do Java no Linux
- 1.11. Instalando o Maven no Windows
- 1.12. Instalando o Maven no Linux
- 1.13. Baixando o projeto inicial do Github
- 1.14. Comandos básicos do Maven
- 1.15. Apresentação do projeto do curso

2. Explorando o JUnit

- 2.1. Introdução ao JUnit
- 2.2. Escrevendo o seu primeiro teste unitário
- 2.3. Explorando as asserções
- 2.4. Asserções de Exceptions
- 2.5. Asserções em listas
- 2.6. Asserções de Timeout
- 2.7. Asserções agrupadas com AssertAll
- 2.8. Executando do teste com Debug
- 2.9. Desabilitando testes unitários
- 2.10. Ignorando execução dos testes condicionalmente com Assumptions
- 2.11. Executando testes via Maven
- 2.12. Desafio - Escrevendo testes faltantes do SaudacaoUtil
- 2.13. Refatorando classe SaudacaoUtil
- 2.14. Desafio - Implementando conta bancária com testes unitários

3. Organizando testes unitários

- 3.1. Organizando testes com o padrão Triple A
- 3.2. Alterando nome de exibição dos testes com @DisplayName
- 3.3. Formatando nomes de testes com @DisplayNameGeneration
- 3.4. Aplicando a nomenclatura do BDD para nomear métodos de teste
- 3.5. Organizando classe de testes com sub-classes e @Nested
- 3.6. Preparando o cenário de testes com @BeforeEach e @BeforeAll

- 3.7. Um teste deve ter uma única asserção?
- 3.8. Combinando @Nested e @BeforeEach com a nomenclatura do BDD
- 3.9. Desafio - Implemente a lógica e testes de um carrinho de compras

4. Stub, Mock e Spy

- 4.1. Implementações falsas com Stub
- 4.2. Introdução ao Mock
- 4.3. Simulando classes com Mockito
- 4.4. Mock usando annotations
- 4.5. Alterando estado dos parâmetros passados no mock
- 4.6. Parâmetros dinâmicos
- 4.7. Verificando chamada de métodos com mock usando Mockito verify
- 4.8. Forçando uma Exception com mock
- 4.9. Capturando parâmetros enviados aos mocks com Argument Captor
- 4.10. Espionando um objeto real com Mockito
- 4.11. Alterando retorno de um mock após chamadas consecutivas
- 4.12. Verificando ordem de chamada de métodos
- 4.13. Usando mock em métodos estáticos
- 4.14. Entendendo problema de mocks não utilizados
- 4.15. Implementando testes no CadastroEditor no método de edição
- 4.16. Desafio - Criar testes do CadastroPost

5. Trabalhando com dados fictícios

- 5.1. Eliminando repetições de código com o Design Pattern Object Mother
- 5.2. Combinando Design Pattern Object Mother com Builder
- 5.3. Desafio - Refatorar testes antigos

6. Testes parametrizados

- 6.1. Testes parametrizados
- 6.2. Carregados dados de um CSV em um método de teste
- 6.3. Carregando valores de um Enum

7. Análise de cobertura de testes

- 7.1. Introdução a análise de cobertura de testes
- 7.2. Descobrimo cenário de testes não cobertos com ajuda da IntelliJ
- 7.3. Lacunas da análise de cobertura automática
- 7.4. Implementando o JaCoCo - Java Code Coverage Library

8. Asserções fluentes com AssertJ

- 8.1. Introdução ao AssertJ?
- 8.2. Asserções básicas
- 8.3. Asserções de Exceptions
- 8.4. Asserções com mensagens descritivas
- 8.5. Asserções customizadas

- 8.6. Múltiplas asserções em listas

Principais Design Patterns Aplicados com Java

1. Design patterns

- 1.1. Introdução
- 1.2. Factory method
- 1.3. Desafio: Factory method
- 1.4. Builder
- 1.5. Desafio: Builder
- 1.6. Abstract factory
- 1.7. Desafio: Abstract factory
- 1.8. Proxy
- 1.9. Desafio: Proxy
- 1.10. Observer
- 1.11. Desafio: Observer
- 1.12. Dependency Injection
- 1.13. Desafio: Dependency Injection
- 1.14. Decorator
- 1.15. Desafio: Decorator
- 1.16. Strategy
- 1.17. Desafio: Strategy

TDD Essencial

1. Introdução

- 1.1. Introdução
- 1.2. Eu já faço testes... ou será que não?
- 1.3. Tipos de testes automatizados
- 1.4. A importância do Teste de Unidade
- 1.5. Usando o JUnit para seu primeiro teste

2. Test-Driven Development - TDD - Parte 1

- 2.1. Como começou o TDD
- 2.2. Aprenda o ciclo vermelho-verde-refatora
- 2.3. Um passo de cada vez - baby steps
- 2.4. Melhorando o Design de Classe
- 2.5. O que testar agora?
- 2.6. E se criarmos os testes depois do código?

3. O Código do Teste

- 3.1. O código do teste também é importante
- 3.2. Removendo duplicações
- 3.3. Defina bons nomes
- 3.4. O padrão Test Data Builders
- 3.5. Atenção as asserções
- 3.6. Testando exceções

4. Test-Driven Development - TDD - Parte 2

- 4.1. Identificando responsabilidades de uma classe
- 4.2. Refatorando para aumentar coesão
- 4.3. Usando Mock Objects
- 4.4. Facilite a evolução do software
- 4.5. Próximos passos - mais Mock
- 4.6. Conclusão

ESPECIALIZAÇÃO EM REST, SPRING E JAKARTA PERSISTENCE

Especialista Spring REST

1. Introdução

- 1.1. Introdução ao treinamento
- 1.2. Como usar o suporte da AlgaWorks
- 1.3. Por que desenvolver REST APIs?
- 1.4. Conhecendo o modelo de domínio do projeto do curso
- 1.5. Preparando o ambiente de desenvolvimento: JDK e STS for Eclipse

2. Spring e Injeção de Dependências

- 2.1. Por que aprender e usar Spring?
- 2.2. Conhecendo o ecossistema Spring
- 2.3. Spring vs Jakarta EE (Java EE)
- 2.4. Conhecendo o Spring Boot
- 2.5. Criando um projeto Spring Boot com Spring Initialize
- 2.6. Conhecendo o Maven e o pom.xml de um projeto Spring Boot

- 2.7. Criando um controller com Spring MVC
- 2.8. Restart mais rápido da aplicação com DevTools
- 2.9. O que é injeção de dependências?
- 2.10. Conhecendo o IoC Container do Spring
- 2.11. Definindo beans com @Component
- 2.12. Injetando dependências (beans Spring)
- 2.13. Usando @Configuration e @Bean para definir beans
- 2.14. Conhecendo os pontos de injeção e a anotação @Autowired
- 2.15. Dependência opcional com @Autowired
- 2.16. Ambiguidade de beans e injeção de lista de beans
- 2.17. Desambiguação de beans com @Primary
- 2.18. Desambiguação de beans com @Qualifier
- 2.19. Desambiguação de beans com anotação customizada
- 2.20. Mudando o comportamento da aplicação com Spring Profiles
- 2.21. Criando métodos de callback do ciclo de vida dos beans
- 2.22. Publicando e consumindo eventos customizados
- 2.23. Configurando projetos Spring Boot com o application.properties
- 2.24. Substituindo propriedades via linha de comando e variáveis de ambiente
- 2.25. Criando e acessando propriedades customizadas com @Value
- 2.26. Acessando propriedades com @ConfigurationProperties
- 2.27. Alterando a configuração do projeto dependendo do ambiente (com Spring Profiles)
- 2.28. Ativando o Spring Profile por linha de comando e variável de ambiente

3. Introdução ao JPA e Hibernate

- 3.1. Instalando o MySQL Server e MySQL Workbench
- 3.2. O que é JPA e Hibernate?
- 3.3. Adicionando JPA e configurando o Data Source
- 3.4. Mapeando entidades com JPA
- 3.5. Criando as tabelas do banco a partir das entidades
- 3.6. Mapeando o id da entidade para autoincremento
- 3.7. Importando dados de teste com import.sql
- 3.8. Consultando objetos do banco de dados
- 3.9. Adicionando um objeto no banco de dados
- 3.10. Buscando um objeto pelo id no banco de dados
- 3.11. Atualizando um objeto no banco de dados
- 3.12. Excluindo um objeto do banco de dados
- 3.13. Conhecendo o padrão Aggregate do DDD
- 3.14. Conhecendo e implementando o padrão Repository
- 3.15. Conhecendo e usando o Lombok
- 3.16. Desafio: Lombok e repositório de restaurantes
- 3.17. Mapeando relacionamento com @ManyToOne
- 3.18. A anotação @JoinColumn
- 3.19. Propriedade nullable de @Column e @JoinColumn
- 3.20. Desafio: mapeando entidades

4. REST com Spring

- 4.1. O que é REST?
- 4.2. Conhecendo as constraints do REST
- 4.3. Diferença entre REST e RESTful

ALGAWORKS BLACK

- 4.4. Desenvolvedores de REST APIs puristas e pragmáticos
 - 4.5. Conhecendo o protocolo HTTP
 - 4.6. Usando o protocolo HTTP
 - 4.7. Instalando e testando o Postman
 - 4.8. Entendendo o que são Recursos REST
 - 4.9. Identificando recursos REST
 - 4.10. Modelando e requisitando um Collection Resource com GET
 - 4.11. Desafio: collection resource de estados
 - 4.12. Representações de recursos e content negotiation
 - 4.13. Implementando content negotiation para retornar JSON ou XML
 - 4.14. Consultando Singleton Resource com GET e @PathVariable
 - 4.15. Customizando as representações XML e JSON com @JsonIgnore, @JsonProperty e @JsonRootName
 - 4.16. Customizando a representação em XML com Wrapper e anotações do Jackson
 - 4.17. Conhecendo os métodos HTTP
 - 4.18. Conhecendo os códigos de status HTTP
 - 4.19. Definindo o status da resposta HTTP com @ResponseStatus
 - 4.20. Manipulando a resposta HTTP com ResponseEntity
 - 4.21. Corrigindo o Status HTTP para resource inexistente
 - 4.22. Status HTTP para collection resource vazia: qual usar?
 - 4.23. Modelando e implementando a inclusão de recursos com POST
 - 4.24. Negociando o media type do payload do POST com Content-Type
 - 4.25. Modelando e implementando a atualização de recursos com PUT
 - 4.26. Modelando e implementando a exclusão de recursos com DELETE
 - 4.27. Implementando a camada de domain services (e a importância da linguagem ubíqua)
 - 4.28. Refatorando a exclusão de cozinhas para usar domain services
 - 4.29. Desafio: modelando e implementando a consulta de recursos de restaurantes
 - 4.30. Modelando e implementando a inclusão de recursos de restaurantes
 - 4.31. Desafio: Modelando e implementando a atualização de recursos de restaurantes
 - 4.32. Desafio: implementando serviços REST de cidades e estados
 - 4.33. Analisando solução para atualização parcial de recursos com PATCH
 - 4.34. Finalizando a atualização parcial com a API de Reflections do Spring
 - 4.35. Introdução ao Modelo de Maturidade de Richardson (RMM)
 - 4.36. Conhecendo o nível 0 do RMM
 - 4.37. Conhecendo o nível 1 do RMM
 - 4.38. Conhecendo o nível 2 do RMM
 - 4.39. Conhecendo o nível 3 do RMM
- ## 5. Super poderes do Spring Data JPA
- 5.1. Implementando consultas JPQL em repositórios
 - 5.2. Conhecendo o projeto Spring Data JPA (SDJ)

ALGAWORKS BLACK

- 5.3. Criando um repositório com Spring Data JPA (SDJ)
- 5.4. Refatorando o código do projeto para usar o repositório do SDJ
- 5.5. Desafio: refatorando todos os repositórios para usar SDJ
- 5.6. Criando consultas com query methods
- 5.7. Usando as keywords para definir critérios de query methods
- 5.8. Conhecendo os prefixos de query methods
- 5.9. Usando queries JPQL customizadas com @Query
- 5.10. Externalizando consultas JPQL para um arquivo XML
- 5.11. Implementando um repositório SDJ customizado
- 5.12. Implementando uma consulta dinâmica com JPQL
- 5.13. Implementando uma consulta simples com Criteria API
- 5.14. Adicionando restrições na cláusula where com Criteria API
- 5.15. Tornando a consulta com Criteria API com filtros dinâmicos
- 5.16. Conhecendo o uso do padrão Specifications (DDD) com SDJ
- 5.17. Implementando Specifications com SDJ
- 5.18. Criando uma fábrica de Specifications
- 5.19. Injetando o próprio repositório na implementação customizada e a anotação @Lazy
- 5.20. Estendendo o JpaRepository para customizar o repositório base

6. Explorando mais do JPA e Hibernate

- 6.1. Mapeando relacionamento bidirecional com @OneToMany
- 6.2. Mapeando relacionamento muitos-para-muitos com @ManyToMany
- 6.3. Analisando o impacto do relacionamento muitos-para-muitos na REST API

- 6.4. Mapeando classes incorporáveis com @Embedded e @Embeddable
- 6.5. Testando e analisando o impacto da incorporação de classe na REST API
- 6.6. Mapeando propriedades com @CreationTimestamp e @UpdateTimestamp
- 6.7. Desafio: mapeando relacionamento muitos-para-um
- 6.8. Desafio: mapeando relacionamento um-para-muitos
- 6.9. Desafio: mapeando relacionamentos muitos-para-muitos
- 6.10. Entendendo o Eager Loading
- 6.11. Entendendo o Lazy Loading
- 6.12. Alterando a estratégia de fetching para Lazy Loading
- 6.13. Alterando a estratégia de fetching para Eager Loading
- 6.14. Resolvendo o Problema do N+1 com fetch join na JPQL

7. Pool de conexões e Flyway

- 7.1. Entendendo o funcionamento de um pool de conexões
- 7.2. Conhecendo o Hikari: a solução padrão de pool de conexões no Spring Boot
- 7.3. Configurando o pool de conexões do Hikari
- 7.4. Schema generation em produção não é uma boa prática
- 7.5. Flyway: ferramenta de versionamento de schemas de banco de dados
- 7.6. Adicionando o Flyway no projeto e criando a primeira migração
- 7.7. Evoluindo o banco de dados com novas migrações
- 7.8. Criando migrações complexas com remanejamento de dados

- 7.9. Criando migração a partir de DDL gerado por schema generation
- 7.10. Adicionando dados de testes com callback do Flyway
- 7.11. Reparando migrações com erros
- 7.12. Desafio: Criando migrações e mapeando as entidades Pedido e ItemPedido

8. Tratamento e modelagem de erros da API

- 8.1. Introdução ao tratamento e modelagem de erros
- 8.2. Lançando exceções customizadas anotadas com `@ResponseStatus`
- 8.3. Lançando exceções do tipo `ResponseStatusException`
- 8.4. Estendendo `ResponseStatusException`
- 8.5. Simplificando o código com o uso de `@ResponseStatus` em exceptions
- 8.6. Desafio: refatorando os serviços REST
- 8.7. Analisando os impactos da refatoração
- 8.8. Criando a exception `NegocioException`
- 8.9. Desafio: usando a exception `NegocioException`
- 8.10. Afinando a granularidade e definindo a hierarquia das exceptions de negócios
- 8.11. Desafio: lançando exceptions de granularidade fina
- 8.12. Tratando exceções em nível de controlador com `@ExceptionHandler`
- 8.13. Tratando exceções globais com `@ExceptionHandler` e `@ControllerAdvice`
- 8.14. Desafio: implementando exception handler
- 8.15. Criando um exception handler global com `ResponseEntityExceptionHandler`
- 8.16. Customizando o corpo da resposta padrão de `ResponseEntityExceptionHandler`
- 8.17. Conhecendo a RFC 7807 (Problem Details for HTTP APIs)
- 8.18. Padronizando o formato de problemas no corpo de respostas com a RFC 7807
- 8.19. Desafio: usando o formato de problemas no corpo de respostas
- 8.20. Customizando exception handlers de `ResponseEntityExceptionHandler`
- 8.21. Tratando a exception `InvalidFormatException` na desserialização
- 8.22. Habilitando erros na desserialização de propriedades inexistentes ou ignoradas
- 8.23. Desafio - tratando `PropertyBindingException` na desserialização
- 8.24. Lançando exception de desserialização na atualização parcial (PATCH)
- 8.25. Desafio: tratando exception de parâmetro de URL inválido
- 8.26. Desafio: tratando a exceção `NoHandlerFoundException`
- 8.27. Desafio: tratando outras exceções não capturadas
- 8.28. Estendendo o formato do problema para adicionar novas propriedades
- 8.29. Desafio: estendendo o formato do problema

9. Validações com Bean Validation

- 9.1. Validação do modelo com Bean Validation
- 9.2. Adicionando constraints e validando no controller com `@Valid`
- 9.3. Desafio: tratando exception de violação de constraints de validação
- 9.4. Estendendo o Problem Details para adicionar as propriedades com constraints violadas
- 9.5. Conhecendo e adicionando mais constraints de validação no modelo
- 9.6. Validando as associações de uma entidade em cascata
- 9.7. Agrupando e restringindo constraints que devem ser usadas na validação
- 9.8. Convertendo grupos de constraints para validação em cascata com `@ConvertGroup`
- 9.9. Desafio: adicionando constraints de validação no modelo
- 9.10. Customizando mensagens de validação na anotação da constraint
- 9.11. Customizando e resolvendo mensagens de validação globais em Resource Bundle
- 9.12. Desafio: customizando mensagens de validação
- 9.13. Resolvendo mensagens de validação com Resource Bundle do Bean Validation
- 9.14. Usando o Resource Bundle do Spring como Resource Bundle do Bean Validation
- 9.15. Criando constraints de validação customizadas usando composição

- 9.16. Criando constraints de validação customizadas com implementação de `ConstraintValidator`
- 9.17. Criando constraints de validação customizadas em nível de classe
- 9.18. Ajustando Exception Handler para adicionar mensagens de validação em nível de classe
- 9.19. Executando processo de validação programaticamente
- 9.20. Desafio: tratando a exception customizada de validações programáticas

10. Testes de integração

- 10.1. Introdução aos Testes de Integração e Testes de APIs
- 10.2. Preparando o projeto para testes de integração
- 10.3. Criando e rodando um teste de integração com Spring Boot, JUnit e AssertJ
- 10.4. Escrevendo bons nomes de testes
- 10.5. Desafio: escrevendo testes de integração
- 10.6. Rodando os testes pelo Maven
- 10.7. Configurando Maven Failsafe Plugin no projeto
- 10.8. Implementando Testes de API com REST Assured e validando o código de status HTTP
- 10.9. Validando o corpo da resposta HTTP
- 10.10. Criando um método para fazer setup dos testes
- 10.11. Entendendo o problema da ordem de execução dos testes
- 10.12. Voltando o estado inicial do banco de dados para cada execução de teste com callback do Flyway

- 10.13. Configurando um banco de testes e usando `@TestPropertySource`
- 10.14. Limpando e populando o banco de dados de teste
- 10.15. Testando endpoint passando parâmetro de URL
- 10.16. Desafio: refatorando o código de testes
- 10.17. Desafio: escrevendo testes de API

11. Boas práticas e técnicas para APIs

- 11.1. Analisando e definindo melhor o escopo das transações
- 11.2. Refinando o payload de cadastro com `@JsonIgnoreProperties`
- 11.3. Criando classes de mixin para usar as anotações do Jackson
- 11.4. Desafio: usando `@JsonIgnoreProperties` e Jackson Mixin
- 11.5. Antes de estudar sobre data/hora: lembrando as aulas de geografia e entendendo os fusos horários
- 11.6. Boas práticas para trabalhar com data e hora em REST APIs
- 11.7. Configurando e refatorando o projeto para usar UTC
- 11.8. Desafio - refatorando o código para usar `OffsetDateTime`
- 11.9. Isolando o Domain Model do Representation Model com o padrão DTO
- 11.10. Implementando a conversão de entidade para DTO
- 11.11. Criando DTOs para entrada de dados na API
- 11.12. Refatorando e criando um assembler de DTO
- 11.13. Desafio: Refatorando e criando um disassembler do DTO
- 11.14. Adicionando e usando o `ModelMapper`
- 11.15. Entendendo a estratégia padrão do `ModelMapper` para correspondência de propriedades

- 11.16. Customizando o mapeamento de propriedades com `ModelMapper`
- 11.17. Mapeando para uma instância destino (e não um tipo) com `ModelMapper`
- 11.18. Revisando e ajustando as mensagens de validação com o uso de DTOs
- 11.19. Estratégias de nomes de propriedades - snake case vs lower camel case
- 11.20. Desafio: usando DTOs como representation model
- 11.21. Corrigindo bug de tratamento de exception de integridade de dados com flush do JPA

12. Modelagem avançada e implementação da API

- 12.1. Modelando sub-recursos para relacionamentos
- 12.2. Granularidade de recursos: Chatty vs Chunky APIs
- 12.3. Modelando conceitos abstratos de negócio e ações não-CRUD como recursos
- 12.4. Implementando os endpoints de ativação e inativação de restaurantes
- 12.5. Desafio: implementando os endpoints de formas de pagamento
- 12.6. Adicionando endereço no modelo da representação do recurso de restaurante
- 12.7. Refatorando serviço de cadastro de restaurante para incluir endereço
- 12.8. Desafio: implementando os endpoints de grupos

- 12.9. Desafio: implementando os endpoints de usuarios
- 12.10. Um pouco mais sobre JPA: objeto alterado fora da transação é sincronizado com o banco de dados
- 12.11. Implementando regra de negócio para evitar usuários com e-mails duplicados
- 12.12. Implementando os endpoints de associação de formas de pagamento em restaurantes
- 12.13. Desafio: implementando os endpoints de produtos
- 12.14. Desafio: Implementando os endpoints de abertura e fechamento de restaurantes
- 12.15. Desafio: implementando os endpoints de associação de grupos com permissões
- 12.16. Desafio: implementando os endpoints de associação de usuários com grupos
- 12.17. Desafio: implementando endpoints de associação de usuários responsáveis com restaurantes
- 12.18. Implementando ativação e inativação em massa de restaurantes
- 12.19. Desafio: Implementando os endpoints de consulta de pedidos
- 12.20. Otimizando a query de pedidos e retornando model resumido na listagem
- 12.21. Desafio: Implementando o endpoint de emissão de pedidos
- 12.22. Implementando endpoint de transição de status de pedidos
- 12.23. Desafio: implementando endpoints de transição de status de pedidos
- 12.24. Refatorando o código de regras para transição de status de pedidos

- 12.25. Usando IDs vs UUIDs nas URIs de recursos

13. Modelagem de projeções, pesquisas e relatórios

- 13.1. Fazendo projeção de recursos com @JsonView do Jackson
- 13.2. Limitando os campos retornados pela API com @JsonFilter do Jackson
- 13.3. Limitando os campos retornados pela API com Squiggly
- 13.4. Implementando pesquisas simples na API
- 13.5. Modelando pesquisas complexas na API
- 13.6. Implementando pesquisas complexas na API
- 13.7. Tratando BindException ao enviar parâmetros de URL inválidos
- 13.8. Implementando paginação e ordenação em recursos de coleção da API
- 13.9. Desafio: implementando paginação e ordenação de pedidos
- 13.10. Implementando JsonSerializer para customizar representação de paginação
- 13.11. Implementando um conversor de propriedades de ordenação
- 13.12. Modelando endpoints de consultas com dados agregados (ideal para gráficos e dashboards)
- 13.13. Discutindo sobre onde implementar as consultas com dados agregados
- 13.14. Implementando consulta com dados agregados de vendas diárias
- 13.15. Desafio: adicionando os filtros na consulta de vendas diárias

- 13.16. Tratando time offset na agregação de vendas diárias por data
- 13.17. Conhecendo o JasperSoft Studio
- 13.18. Criando um layout do relatório JasperReports de vendas diárias
- 13.19. Estruturando endpoint e serviço de emissão de relatório em PDF
- 13.20. Preenchendo um relatório JasperReports com JavaBeans e gerando bytes do PDF

14. Upload e download de arquivos

- 14.1. Conhecendo soluções para upload de arquivos em REST APIs
- 14.2. Implementando upload de arquivo com multipart/form-data
- 14.3. Validando o tamanho máximo do arquivo
- 14.4. Desafio: Validando o content type do arquivo
- 14.5. Mapeando entidade FotoProduto e relacionamento um-para-um
- 14.6. Implementando serviço de cadastro de foto de produto
- 14.7. Excluindo e substituindo cadastro de foto de produto
- 14.8. Implementando o serviço de armazenagem de fotos no disco local
- 14.9. Integrando o serviço de catálogo de fotos com o serviço de armazenagem
- 14.10. Implementando a remoção e substituição de arquivos de fotos no serviço de armazenagem
- 14.11. Desafio: Implementando recuperação de foto no serviço de armazenagem
- 14.12. Desafio: Implementando endpoint de consulta de foto de produto
- 14.13. Servindo arquivos de fotos pela API
- 14.14. Checando media type ao servir arquivos de fotos
- 14.15. Desafio: implementando endpoint de exclusão de foto de produto
- 14.16. Corrigindo exception handler de media type não aceita
- 14.17. Amazon S3: conhecendo o serviço de storage da AWS
- 14.18. Criando usuário com permissões para adicionar objetos na Amazon S3
- 14.19. Criando chaves de acesso para a API da AWS
- 14.20. Criando bean de propriedades de configuração dos serviços de storage
- 14.21. Adicionando o SDK Java da Amazon S3 no projeto e criando a classe da implementação do serviço de storage
- 14.22. Definindo bean do client da Amazon S3 e configurando credenciais
- 14.23. Implementando a inclusão de objetos no bucket da Amazon S3
- 14.24. Desafio: Implementando a exclusão de objetos do bucket da Amazon S3
- 14.25. Implementando a recuperação de foto no serviço de storage do S3
- 14.26. Selecionando a implementação do serviço de storage de fotos

15. E-mails transacionais e Domain Events

- 15.1. Conhecendo soluções para envio de e-mails transacionais
- 15.2. Configurando o projeto para envio de e-mails usando servidor SMTP
- 15.3. Implementando o serviço de infraestrutura de envio de e-mails com Spring
- 15.4. Usando o serviço de envio de e-mails na confirmação de pedidos
- 15.5. Processando template do corpo de e-mails com Apache FreeMarker
- 15.6. Melhorando o texto do e-mail com FTL (FreeMarker Template Language)
- 15.7. Formatando valores monetários com FTL
- 15.8. Desafio: implementando serviço de envio de e-mail fake
- 15.9. Desafio: Implementando serviço de envio de e-mail sandbox
- 15.10. Conhecendo o padrão Domain Events do DDD
- 15.11. Publicando Domain Events a partir do Aggregate Root
- 15.12. Observando e reagindo a Domain Events
- 15.13. Reagindo a Domain Events em fases específicas da transação
- 15.14. Desafio: enviando e-mails no cancelamento de pedidos

16. CORS e consumo da API com JavaScript e Java

- 16.1. Implementando uma chamada na REST API com JavaScript

- 16.2. Testando a requisição na API com JavaScript e entendendo a Same Origin Policy
- 16.3. Entendendo o funcionamento básico de CORS e habilitando na API
- 16.4. Habilitando CORS em controladores e métodos com @CrossOrigin
- 16.5. Entendendo o funcionamento de preflight do CORS
- 16.6. Habilitando CORS globalmente no projeto da API
- 16.7. Desafio: implementando uma requisição GET com JavaScript
- 16.8. Implementando um formulário de cadastro e fazendo requisição POST com JavaScript
- 16.9. Desafio: implementando uma requisição DELETE com JavaScript
- 16.10. Implementando um client da REST API com Java e Spring (RestTemplate)
- 16.11. Tratando respostas com código de erro no client Java
- 16.12. Desafio: Implementando uma requisição POST no client Java

17. Cache de HTTP

- 17.1. Introdução ao Cache de HTTP
- 17.2. Habilitando o cache com o cabeçalho Cache-Control e a diretiva max-age
- 17.3. Desafio: adicionando o cabeçalho Cache-Control na resposta
- 17.4. Entendendo a validação de representações em cache com ETags
- 17.5. Implementando requisições condicionais com Shallow ETags

- 17.6. Adicionando outras diretivas de Cache-Control na resposta HTTP
- 17.7. Usando a diretiva no-cache no cabeçalho Cache-Control da requisição
- 17.8. Entendendo e preparando a implementação de Deep ETags
- 17.9. Implementando requisições condicionais com Deep ETags
- 17.10. Desafio: implementando requisições condicionais com Deep ETags

18. Documentação da API com OpenAPI, Swagger UI e SpringFox

- 18.1. Introdução à documentação de REST APIs
- 18.2. Conhecendo a OpenAPI (antes conhecida como Swagger)
- 18.3. Gerando a definição OpenAPI em JSON com SpringFox
- 18.4. Gerando a documentação da API em HTML com Swagger UI e SpringFox
- 18.5. Selecionando os endpoints da API para gerar a documentação
- 18.6. Descrevendo informações da API na documentação
- 18.7. Descrevendo tags na documentação e associando com controllers
- 18.8. Descrevendo as operações de endpoints na documentação
- 18.9. Descrevendo parâmetros de entrada na documentação
- 18.10. Descrevendo modelos de representações e suas propriedades
- 18.11. Descrevendo restrições de validação de propriedades do modelo
- 18.12. Descrevendo códigos de status de respostas de forma global
- 18.13. Desafio: descrevendo códigos de status de respostas de forma global
- 18.14. Descrevendo o modelo de representação de problema
- 18.15. Referenciando modelo de representação de problema com códigos de status de erro
- 18.16. Descrevendo códigos de status de respostas em endpoints específicos
- 18.17. Desacoplando anotações do Swagger dos controladores
- 18.18. Desafio: descrevendo documentação de endpoints de grupos
- 18.19. Descrevendo media type da resposta nos endpoints
- 18.20. Corrigindo documentação com substituição de Pageable
- 18.21. Corrigindo documentação com substituição Page
- 18.22. Desafio: descrevendo documentação de endpoints de cozinhas
- 18.23. Ignorando tipos de parâmetros de operações na documentação
- 18.24. Desafio: descrevendo documentação de endpoints de formas de pagamento
- 18.25. Descrevendo parâmetros globais em operações
- 18.26. Descrevendo parâmetros implícitos em operações
- 18.27. Desafio: descrevendo documentação de endpoints de pedidos
- 18.28. Descrevendo parâmetros de projeções em endpoints de consultas

- 18.29. Desafio: descrevendo documentação de endpoints de restaurantes
- 18.30. Desafio: descrevendo documentação de endpoints de estados
- 18.31. Desafio: descrevendo documentação de endpoints de fluxo de pedidos
- 18.32. Desafio: descrevendo documentação de endpoints de associação de restaurantes com formas de pagamento
- 18.33. Desafio: descrevendo documentação de endpoints de associação de restaurantes com usuários
- 18.34. Desafio: descrevendo documentação de endpoints de produtos
- 18.35. Desafio: descrevendo documentação de endpoints de fotos de produtos
- 18.36. Corrigindo documentação no Swagger UI para upload de arquivos
- 18.37. Desafio: descrevendo documentação de endpoints de associação de permissões com grupos
- 18.38. Desafio: descrevendo documentação de endpoints de usuários
- 18.39. Desafio: descrevendo documentação de endpoints de associação de grupos com usuários
- 18.40. Desafio: descrevendo documentação de endpoint de estatísticas

19. Discoverability e HATEOAS: A Glória do REST

- 19.1. Introdução à Discoverability e HATEOAS
- 19.2. Adicionando a URI do recurso criado no header da resposta
- 19.3. Adicionando o Spring HATEOAS no projeto
- 19.4. Atualizando o projeto para Spring Boot 2.2 (Spring HATEOAS 1.0)
- 19.5. Resolvendo conflito de dependências com Spring HATEOAS e SpringFox
- 19.6. Conhecendo especificações para formatos Hypermedia
- 19.7. Adicionando hypermedia na representação de recurso único com HAL
- 19.8. Construindo links dinâmicos com WebMvcLinkBuilder
- 19.9. Construindo links que apontam para métodos
- 19.10. Adicionando hypermedia na representação de recursos de coleção
- 19.11. Montando modelo de representação com RepresentationModelAssembler
- 19.12. Desafio: adicionando hypermedia nos recursos de usuários
- 19.13. Corrigindo link de coleção de recurso de responsáveis por restaurante
- 19.14. Desafio: adicionando hypermedia nos recursos de estados
- 19.15. Adicionando hypermedia em recursos com paginação
- 19.16. Desafio: adicionando hypermedia em recursos de pedidos (paginação)
- 19.17. Corrigindo links de paginação com ordenação
- 19.18. Adicionando links com template variables

ALGAWORKS BLACK

- 19.19. Desafio: adicionando template variables do filtro de pedidos
- 19.20. Refatorando construção e inclusão de links em representation model
- 19.21. Desafio: refatorando construção e inclusão de links
- 19.22. Adicionando links de transições de status de pedidos
- 19.23. Adicionando links condicionalmente
- 19.24. Desafio: adicionando hypermedia nos recursos de restaurantes
- 19.25. Desafio: adicionando links condicionais no recurso de restaurante
- 19.26. Desafio: adicionando template variable de projeção de restaurantes
- 19.27. Desafio: adicionando hypermedia nos recursos de formas de pagamento
- 19.28. Adicionando links para desassociação de formas de pagamento com restaurante
- 19.29. Adicionando links com template variable de caminho para associação de formas de pagamento com restaurante
- 19.30. Desafio: adicionando links de associação de restaurantes com responsáveis
- 19.31. Desafio: adicionando hypermedia nos recursos de produtos
- 19.32. Desafio: adicionando links para recurso de foto de produto
- 19.33. Desafio: adicionando hypermedia nos recursos de grupos
- 19.34. Desafio: adicionando links de associação de grupos com permissões
- 19.35. Desafio: adicionando links de associação de usuários com grupos
- 19.36. Implementando o Root Entry Point da API

- 19.37. Desafio: implementando endpoint com links de recursos de estatísticas
- 19.38. Comprimindo as respostas HTTP com Gzip
- 19.39. Corrigindo as propriedades de links na documentação
- 19.40. Corrigindo a documentação dos endpoints de cidades
- 19.41. Corrigindo a paginação na documentação
- 19.42. Desafio: corrigindo a documentação dos endpoints de estados
- 19.43. Desafio: corrigindo a documentação dos endpoints de formas de pagamento
- 19.44. Desafio: corrigindo a documentação dos endpoints de grupos
- 19.45. Desafio: corrigindo a documentação dos endpoint de pedidos (paginação)
- 19.46. Desafio: corrigindo a documentação dos endpoints de produtos
- 19.47. Desafio: corrigindo a documentação dos endpoints de restaurantes e usuários
- 19.48. Removendo modelo de representação inutilizado da documentação

20. Evoluindo e versionando a API

- 20.1. Evoluindo a API com gestão de mudanças
- 20.2. Evitando quebrar os clientes: nova propriedade no modelo
- 20.3. Evitando quebrar os clientes: exclusão de propriedade do modelo

- 20.4. Evitando quebrar os clientes: alteração de tipo de propriedade do modelo
- 20.5. Evitando quebrar os clientes: alteração na estrutura de dados do modelo
- 20.6. Evitando quebrar os clientes: alteração de URL de recurso
- 20.7. O que é e quando versionar uma API?
- 20.8. As principais técnicas de versionamento de APIs
- 20.9. As principais abordagens para manter a base de código de APIs versionadas
- 20.10. Preparando o projeto para versionamento da API por Media Type
- 20.11. Implementando o versionamento da API por Media Type
- 20.12. Definindo a versão padrão da API quando o Media Type não é informado
- 20.13. Implementando o versionamento da API por URI
- 20.14. Desafio: Refatorando controladores para adicionar /v1 nas URIs
- 20.15. Desafio: adicionando os recursos de cozinhas na v2 da API
- 20.16. Gerando documentação das versões da API com SpringFox e Swagger UI
- 20.17. Desafio: revisando documentação da v2 da API
- 20.18. Depreciando uma versão da API
- 20.19. Desligando uma versão da API

21. Logging

- 21.1. Introdução ao Logback e SLF4J
- 21.2. Desafio: registrando logs de exceptions não tratadas

- 21.3. Criando uma conta no Loggly: serviço de gerenciamento de logs na nuvem
- 21.4. Configurando o appender do Loggly no Logback
- 21.5. Configurando o Logback para alternar as configurações por Spring Profiles

22. Segurança com Spring Security e OAuth2

- 22.1. Introdução à segurança de REST APIs
- 22.2. Adicionando segurança na API com Spring Security
- 22.3. Configurando Spring Security com HTTP Basic
- 22.4. Configurando autenticação de usuários em memória
- 22.5. Introdução ao OAuth2
- 22.6. Soluções para OAuth2: nova stack do Spring Security vs Spring Security OAuth
- 22.7. Conhecendo o fluxo Resource Owner Password Credentials
- 22.8. Criando o projeto do Authorization Server com Spring Security OAuth2
- 22.9. Configurando o fluxo Authorization Server com Password Credentials e Opaque Tokens
- 22.10. Configurando o endpoint de introspecção de tokens no Authorization Server
- 22.11. Configurando o Resource Server com a nova stack do Spring Security
- 22.12. Conhecendo o fluxo para emitir e usar Refresh Tokens
- 22.13. Configurando o Refresh Token Grant Type no Authorization Server

- 22.14. Configurando a validade e não reutilização de Refresh Tokens
- 22.15. Conhecendo o fluxo Client Credentials
- 22.16. Configurando o Client Credentials Grant Type no Authorization Server
- 22.17. Revisando o fluxo Authorization Code
- 22.18. Configurando o Authorization Code Grant Type
- 22.19. Testando o fluxo Authorization Code com um client JavaScript
- 22.20. Conhecendo o fluxo Implicit
- 22.21. Configurando o fluxo Implicit Grant Type
- 22.22. Mais segurança com PKCE e Authorization Code Grant
- 22.23. Implementando o suporte a PKCE com o fluxo Authorization Code
- 22.24. Testando o fluxo Authorization Code com PKCE com o método plain
- 22.25. Testando o fluxo Authorization Code com PKCE com o método SHA-256
- 22.26. Testando um client JavaScript com PKCE e Authorization Code
- 22.27. Decidindo qual fluxo OAuth2 usar

23. OAuth2 avançado com JWT e controle de acesso

- 23.1. Armazenando tokens no Redis: um banco de dados NoSQL
- 23.2. Configurando o RedisTokenStore

- 23.3. Entendendo a diferença entre Stateful e Stateless Authentication
- 23.4. Transparent Tokens: conhecendo o JSON Web Tokens (JWT)
- 23.5. Gerando JWT com chave simétrica (HMAC SHA-256) no Authorization Server
- 23.6. Configurando o Resource Server para JWT assinado com chave simétrica
- 23.7. Entendendo a diferença entre assinatura com chave simétrica e assimétrica
- 23.8. Gerando um par de chaves com keytool
- 23.9. Assinando o JWT com RSA SHA-256 (chave assimétrica)
- 23.10. Desafio: criando bean de propriedades de configuração do KeyStore
- 23.11. Extraindo a chave pública no formato PEM
- 23.12. Configurando a validação de JWT no Resource Server com a chave pública
- 23.13. Revisando o fluxo de aprovação do Authorization Code com JWT
- 23.14. Autenticando usuário com dados do banco de dados
- 23.15. Desafio: refatorando serviços de usuários para usar BCrypt
- 23.16. Adicionando claims customizadas no payload do JWT
- 23.17. Obtendo usuário autenticado no Resource Server
- 23.18. Definindo e criando as permissões de acesso
- 23.19. Carregando as permissões concedidas na autenticação
- 23.20. Carregando as Granted Authorities e restringindo acesso a endpoints na API
- 23.21. Method Security: Restringindo acesso com @PreAuthorize e SpEL

ALGAWORKS BLACK

- 23.22. Desafio: tratando AccessDeniedException no ExceptionHandler
- 23.23. Simplificando o controle de acesso em métodos com meta-annotações
- 23.24. Entendendo os escopos do OAuth2
- 23.25. Carregando Granted Authorities dos escopos do OAuth2 no Resource Server
- 23.26. Restringindo acesso a endpoints por escopos do OAuth2
- 23.27. Desafio: restringindo acesso dos endpoints de restaurantes
- 23.28. Restringindo acessos de forma contextual (sensível à informação)
- 23.29. Restringindo acessos com @PostAuthorize
- 23.30. Desafio: restringindo acessos ao endpoint de pesquisa de pedidos
- 23.31. Desafio: restringindo acessos aos endpoints de transição de status de pedidos
- 23.32. Desafio: restringindo acessos aos endpoints de formas de pagamentos
- 23.33. Desafio: restringindo acessos aos endpoints de cidades e estados
- 23.34. Desafio: restringindo acessos aos endpoints de usuários, grupos e permissões
- 23.35. Desafio: restringindo acessos aos endpoints de estatísticas
- 23.36. Configurando os clientes OAuth2 em um banco de dados SQL
- 23.37. Cadastrando clientes OAuth2 no banco de dados e testando a emissão de tokens

- 23.38. Corrigindo lógica de restrição de acessos para Client Credentials Flow
- 23.39. Gerando links do HAL dinamicamente de acordo com permissões do usuário
- 23.40. Desafio: gerando links do HAL dinamicamente de acordo com permissões
- 23.41. Juntando o Resource Server com o Authorization Server no mesmo projeto
- 23.42. Ajustando a documentação da API para suporte a OAuth2
- 23.43. Customizando a página de login
- 23.44. Customizando a página de OAuth2 Approval
- 23.45. Implementando o endpoint do JSON Web Key Set (JWKS)
- 23.46. Externalizando o KeyStore: criando um ProtocolResolver para Base64

24. Dockerizando a aplicação

- 24.1. Conhecendo o Docker
- 24.2. Instalando o Docker
- 24.3. Executando um container
- 24.4. Gerenciando melhor os containers
- 24.5. Conhecendo a arquitetura do Docker
- 24.6. Entendendo o que são as imagens e o Docker Hub
- 24.7. Gerenciando imagens
- 24.8. Executando um container do MySQL
- 24.9. Construindo a imagem da aplicação com Dockerfile
- 24.10. Criando uma network e conectando dois containers
- 24.11. Construindo a imagem Docker pelo Maven

ALGAWORKS BLACK

- 24.12. Disponibilizando a imagem da aplicação no Docker Hub
- 24.13. Conhecendo e usando Docker Compose
- 24.14. Controlando a ordem de inicialização com wait-for-it.sh
- 24.15. Escalando um serviço com Docker Compose
- 24.16. Entendendo o Poor Man's Load Balancer (DNS Round Robin)
- 24.17. Configurando um proxy reverso com Nginx
- 24.18. Configurando o Spring Session com Redis
- 24.19. Resolvendo problemas com storage de Authorization Codes
- 24.20. Configurando o Spring Session Data Redis
- 24.21. Resolvendo problemas com storage de Authorization Codes

25. Deploy em containers Docker na Amazon

- 25.1. Introdução ao deployment em produção
- 25.2. Mais organização das propriedades do projeto com Spring Profiles
- 25.3. Dependência de JavaMailSender não satisfeita: melhorando o uso da herança
- 25.4. Conhecendo a Amazon Web Services (AWS)
- 25.5. Entendendo alguns conceitos fundamentais da nuvem da AWS
- 25.6. Monitorando e criando um alerta de orçamento da AWS
- 25.7. Criando o bucket no Amazon S3
- 25.8. Criando uma instância do MySQL no Amazon RDS
- 25.9. Criando schema e usuário da aplicação
- 25.10. Conhecendo e criando uma conta no Redislabs
- 25.11. Criando uma instância do Redis na nuvem

- 25.12. Conhecendo o Amazon Elastic Container Service (ECS) e AWS Fargate
- 25.13. Publicando um container no Amazon ECS
- 25.14. Subindo a imagem Docker para o Amazon Elastic Container Registry (ECR)
- 25.15. Organizando as variáveis de ambiente do container da aplicação
- 25.16. Gerenciando as configurações com AWS Systems Manager Parameter Store
- 25.17. Configurando Amazon ECS para rodar nossa aplicação
- 25.18. Permitindo a leitura de parâmetros do Parameter Store pelo serviço do Amazon ECS
- 25.19. Permitindo o acesso ao MySQL pelo Security Group do serviço do Amazon ECS
- 25.20. Inserindo dados no banco de dados de produção
- 25.21. Conhecendo o Elastic Load Balancing da Amazon
- 25.22. Configurando e provisionando um Load Balancer na Amazon
- 25.23. Configurando o balanceamento de carga no serviço do Amazon ECS
- 25.24. Registrando um domínio de internet no Registro.br
- 25.25. Configurando o domínio para o Application Load Balancer
- 25.26. Configurando certificado TLS (HTTPS) com AWS Certificate Manager
- 25.27. Configurando o protocolo HTTPS nos links da API com HATEOAS
- 25.28. Testando a API em produção
- 25.29. Conclusão e próximos passos

26. Documentação da API com SpringDoc

- 26.1. Conhecendo o SpringDoc
- 26.2. Removendo o SpringFox do projeto
- 26.3. Adicionando o SpringDoc no projeto
- 26.4. Configurando múltiplas documentações em um só projeto
- 26.5. Ajustando a documentação da API para suporte a OAuth2
- 26.6. Descrevendo tags na documentação e associando com controllers
- 26.7. Descrevendo as operações de endpoints na documentação
- 26.8. Descrevendo parâmetros de entrada na documentação
- 26.9. Descrevendo modelos de representações e suas propriedades
- 26.10. Descrevendo restrições de validação de propriedades do modelo
- 26.11. Descrevendo códigos de status de respostas de forma global
- 26.12. Descrevendo códigos de status de respostas em endpoints específicos
- 26.13. Descrevendo códigos de status de respostas de forma global para cada tipo de método HTTP
- 26.14. Descrevendo o modelo de representação de problema
- 26.15. Referenciando modelo de representação de problema com códigos de status de erro
- 26.16. Desafio: descrevendo documentação de endpoints de grupos
- 26.17. Corrigindo documentação com substituição de Pageable
- 26.18. Desafio: descrevendo documentação de endpoints de cozinhas
- 26.19. Desafio: descrevendo documentação de endpoints de formas de pagamento
- 26.20. Desafio: descrevendo documentação de endpoints de pedidos
- 26.21. Descrevendo parâmetros de projeções em endpoints de consultas
- 26.22. Descrevendo media type da resposta nos endpoints
- 26.23. Corrigindo documentação no Swagger UI para upload de arquivos
- 26.24. Desafio: descrevendo documentação de endpoints de restaurantes
- 26.25. Desafio: descrevendo documentação de endpoints de estados
- 26.26. Desafio: descrevendo documentação de endpoints de fluxo de pedidos
- 26.27. Desafio: descrevendo documentação de endpoints de associação de restaurantes com formas de pagamento
- 26.28. Desafio: descrevendo documentação de endpoints de associação de restaurantes com usuários
- 26.29. Desafio: descrevendo documentação de endpoints de produtos
- 26.30. Desafio: descrevendo documentação de endpoints de fotos de produtos
- 26.31. Desafio: descrevendo documentação de endpoints de associação de permissões com grupos

- 26.32. Desafio: descrevendo documentação de endpoints de usuários
- 26.33. Desafio: descrevendo documentação de endpoints de associação de grupos com usuários
- 26.34. Desafio: descrevendo documentação de endpoint de estatísticas
- 26.35. Desafio: descrevendo documentação de endpoint de permissões
- 26.36. Corrigindo documentação ocultando o Root Entry Point

27. Spring Authorization Server

- 27.1. O que é o Spring Authorization Server?
- 27.2. Removendo o Authorization Server antigo do projeto
- 27.3. Configuração inicial do Authorization Server com Access Token opaco
- 27.4. Testando com fluxo Client Credentials com Postman
- 27.5. Inspeccionando token opaco usando o endpoint OAuth2 Introspect
- 27.6. Configurando o Resource Server com token opaco
- 27.7. Armazenando autorizações no banco de dados
- 27.8. Revogando o Access Token com OAuth2 Revoke
- 27.9. Configurando a geração de Access Token JWT no Authorization Server
- 27.10. Configurando o Resource Server com token JWT
- 27.11. Implementando um cliente com o fluxo Authorization Code
- 27.12. Testando o fluxo Authorization Code + PKCE + S256 e corrigindo problemas

- 27.13. Implementando um cliente com o fluxo Refresh Token
- 27.14. Customizando o token JWT com dados do usuário
- 27.15. Lendo informações customizadas do JWT no Resource Server
- 27.16. Implementado Repository de Clients do OAuth2 via JDBC
- 27.17. Customizando a página de login do Authorization Server
- 27.18. Customizando a página de consentimento do OAuth2
- 27.19. Armazenando autorizações de consentimento no banco de dados
- 27.20. Criando uma página de listagem dos clientes com consentimentos permitidos
- 27.21. Revogando consentimentos e autorizações dos clientes

28. Spring Boot 3

- 28.1. As principais mudanças do Spring Boot 3
- 28.2. Removendo componentes incompatíveis
- 28.3. Atualização a dependências e componentes do Spring
- 28.4. Alterações do Jakarta EE e Jakarta Persistence 3.0
- 28.5. Atualizando o Spring Doc
- 28.6. Atualizando o Spring Authorization Server

Es

1.1

1.1.1

1.2.

1.3.

1.4.

1.5.

1.6.

1.7.

1.8.

1.9.

1.10

1.11

1.12

1.13

2.1

2.2.

2.3.

2.4.

2.5.

2.6.

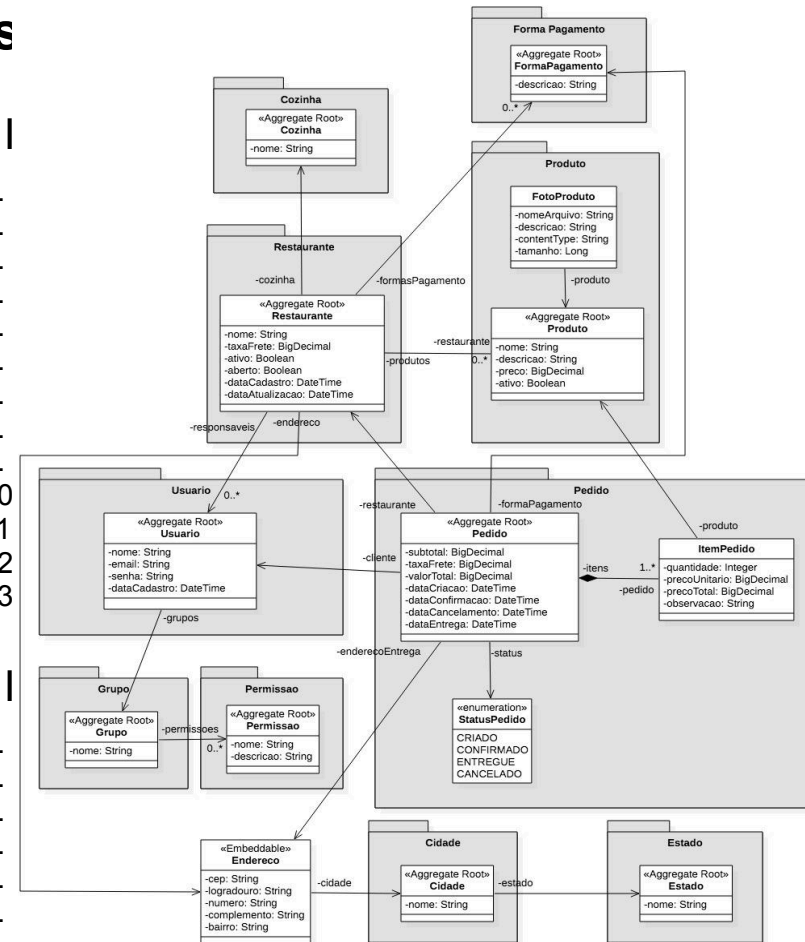
2.7.

2.8. Aditiindo e recriando uma transação

2.9. Inserindo o primeiro objeto com o método persist

2.10. Removendo objetos com o método remove

2.11. Atualizando objetos com o método merge



- 2.12. Atualizando objetos gerenciados
- 2.13. Adicionando objetos com o método merge
- 2.14. Entendendo a diferença entre os métodos persist e merge
- 2.15. Exercício: implementando um CRUD
- 2.16. Desanexando objetos do contexto de persistência com o método detach
- 2.17. Conhecendo e usando Lombok

3. Mapeamento básico

- 3.1. Conhecendo o modelo de domínio do projeto e criando as entidades
- 3.2. Mapeando as entidades e customizando os nomes das tabelas e colunas
- 3.3. Exercício: mapeando a classe Pedido
- 3.4. Entendendo a diferença entre mapear atributos ou métodos
- 3.5. Mapeando enumerações com @Enumerated
- 3.6. Mapeando objetos embutidos com @Embeddable
- 3.7. Conhecendo as estratégias para geração de identificador com @GeneratedValue
- 3.8. Configurando a geração de identificador com @SequenceGenerator
- 3.9. Configurando a geração de identificador com @TableGenerator
- 3.10. Configurando geração de identificador com a estratégia IDENTITY
- 3.11. Exercício: corrigindo classes de testes

4. Mapeamento de relacionamentos

- 4.1. Conhecendo os tipos de relacionamentos entre entidades
- 4.2. Mapeando relacionamentos muito-para-um com @ManyToOne
- 4.3. Exercício: mapeando relacionamentos muitos-para-um

- 4.4. Mapeando relacionamentos um-para-muitos com @OneToMany
- 4.5. Exercício: mapeando relacionamentos um-para-muitos
- 4.6. Mapeando autorelacionamentos com @ManyToOne e @OneToMany
- 4.7. Removendo objetos referenciados por outras entidades
- 4.8. Mapeando relacionamentos muitos-para-muitos com @ManyToMany e @JoinTable
- 4.9. Mapeamento relacionamentos um-para-um com @OneToOne
- 4.10. Exercício: mapeando relacionamentos um-para-um
- 4.11. Mapeando relacionamentos um-para-um com @JoinTable
- 4.12. Entendendo o funcionamento de Eager e Lazy Loading
- 4.13. Para o que serve o atributo optional?
- 4.14. Exercício: usando o atributo optional

5. Conhecendo o EntityManager

- 5.1. Estados e ciclo de vida dos objetos
- 5.2. Entendendo o cache de primeiro nível
- 5.3. Gerenciamento de transações
- 5.4. Funcionamento do método flush
- 5.5. Contexto de persistência e o dirty checking
- 5.6. Callbacks para eventos do ciclo de vida
- 5.7. Listeners para eventos do ciclo de vida

6. Mapeamento avançado

- 6.1. Conhecendo detalhes da anotação @Column
- 6.2. Exercício: anotação @Column
- 6.3. Mapeando chave composta com @IdClass
- 6.4. Exercício: usando @IdClass
- 6.5. Mapeando chave composta com @EmbeddedId

ALGAWORKS BLACK

- 6.6. Mapeando chave primária e estrangeira na mesma coluna com @MapsId
- 6.7. Exercício: usando @MapsId
- 6.8. Declarando propriedades transientes com @Transient
- 6.9. Mapeando coleções de tipos básicos com @ElementCollection
- 6.10. Mapeando coleções de objetos embutidos com @ElementCollection
- 6.11. Mapeando mapas com @ElementCollection
- 6.12. Mapeando e persistindo dados de arquivos com @Lob
- 6.13. Exercício: persistindo fotos de produtos
- 6.14. Mapeando tabela secundária com @SecondaryTable
- 6.15. Mapeando herança com @MappedSuperclass
- 6.16. Entendendo a diferença entre estender uma entidade abstrata e usar a anotação @MappedSuperclass
- 6.17. Mapeando herança com estratégia de tabela única (single table)
- 6.18. Mapeando herança com estratégia de uma tabela por classe (table per class)
- 6.19. Mapeando herança com a estratégia Joined Table
- 6.20. Exercício: voltando o mapeando de herança para tabela única

7. Mapeando entidades para geração do DDL

- 7.1. Quando criar o schema do banco usando JPA?
- 7.2. Configurando detalhes da tabela com @Table
- 7.3. Exercício: usando @Table
- 7.4. Configurando colunas com @Column
- 7.5. Exercício: usando @Column
- 7.6. Corrigindo os testes do JUnit
- 7.7. Usando a anotação @Lob em strings
- 7.8. Configurando chaves estrangeiras com @JoinColumn
- 7.9. Exercício: usando @JoinColumn

- 7.10. Entendendo alguns detalhes de @JoinTable
- 7.11. Configurando tabelas secundárias com @SecondaryTable
- 7.12. Conhecendo as estratégias de schema generation
- 7.13. Gerando o schema do banco com arquivos de scripts SQL
- 7.14. Gerando o schema do banco com metadados e scripts
- 7.15. Exportando os scripts de schema generation para arquivos externos
- 7.16. Configurando propriedades da unidade de persistência dinamicamente para schema generation

8. Operações em cascata

- 8.1. Configurando operações em cascata
- 8.2. Fazendo inserções de objetos em cascata
- 8.3. Exercício: fazendo inserções em cascata
- 8.4. Fazendo atualizações em cascata
- 8.5. Exercício: fazendo atualizações em cascata
- 8.6. Fazendo remoções em cascata
- 8.7. Entendendo a remoção em cascata com @ManyToMany
- 8.8. Removendo objetos órfãos com a propriedade orphanRemoval
- 8.9. Quando configurar operações em cascata?

9. JPQL do básico ao avançado

- 9.1. Introdução à JPQL (Java Persistence Query Language)
- 9.2. Entendendo as diferenças entre TypedQuery e Query
- 9.3. Selecionando um atributo da entidade como retorno da consulta
- 9.4. Trabalhando com projeções
- 9.5. Trabalhando com projeções e DTO
- 9.6. Fazendo inner join entre as entidades
- 9.7. Usando left outer join

ALGAWORKS BLACK

- 9.8. Fazendo o join e usando o fetch
- 9.9. Entendendo as Path Expressions
- 9.10. Exercício: consultando pedidos com produto específico
- 9.11. Passando parâmetros para as consultas
- 9.12. Usando expressão condicional like
- 9.13. Usando expressões condicionais is null e is empty
- 9.14. Usando expressões condicionais de maior e menor
- 9.15. Exercício: usando expressões de maior e menor com datas
- 9.16. Usando expressão condicional between
- 9.17. Usando expressão de diferente
- 9.18. Usando operadores lógicos
- 9.19. Ordenando os resultados da consulta
- 9.20. Fazendo paginação de resultados
- 9.21. Limitando a quantidade de registros retornados
- 9.22. Usando funções para strings
- 9.23. Usando funções para datas
- 9.24. Usando funções para números
- 9.25. Usando funções para coleções
- 9.26. Usando funções nativas
- 9.27. Usando funções de agregação
- 9.28. Agrupando o registros com group by
- 9.29. Exercício: usando group by
- 9.30. Usando a cláusula where com group by
- 9.31. Usando o having para condicionar o agrupamento
- 9.32. Usando a expressão case
- 9.33. Usando a expressão in
- 9.34. Usando o distinct para evitar duplicações
- 9.35. Criando subqueries
- 9.36. Criando subqueries com a expressão in
- 9.37. Criando subqueries com a expressão exists
- 9.38. Exercício: usando a expressão in
- 9.39. Exercício: criando subqueries

- 9.40. Exercício: criando subqueries com exists
- 9.41. Criando subqueries com all
- 9.42. Criando subqueries com any
- 9.43. Exercício: criando subqueries com all
- 9.44. Fazendo operações em lote
- 9.45. Atualizando objetos em lote
- 9.46. Removendo objetos em lote
- 9.47. Configurando uma dynamic query
- 9.48. Configurando uma query nomeada com @NamedQuery
- 9.49. Externalizando queries em um arquivo XML
- 9.50. Abordagem híbrida para dynamic e named queries

10. Criteria API do básico ao avançado

- 10.1. Introdução à Criteria API do JPA
- 10.2. Selecionando um atributo da entidade como retorno da consulta
- 10.3. Exercício: retornando todos os produtos
- 10.4. Trabalhando com projeções
- 10.5. Usando tuple para uma projeção
- 10.6. Trabalhando com projeções e DTO
- 10.7. Fazendo inner join entre as entidades
- 10.8. Usando a cláusula on no join
- 10.9. Usando left outer join
- 10.10. Fazendo o join e usando o fetch
- 10.11. Consultando pedidos com um produto específico
- 10.12. Passando parâmetros para as consultas
- 10.13. Tipagem forte com metamodel
- 10.14. Usando expressão condicional like
- 10.15. Usando as expressões condicionais is null e is empty
- 10.16. Usando expressões condicionais de maior e menor
- 10.17. Exercício: usando expressões de maior e menor com datas

- 10.18. Usando expressão condicional between
- 10.19. Usando expressão de diferente
- 10.20. Usando operadores lógicos
- 10.21. Ordenando os resultados da consulta
- 10.22. Fazendo paginação e limitando resultados
- 10.23. Usando funções para string
- 10.24. Usando funções para datas
- 10.25. Usando funções para números
- 10.26. Usando funções para coleções
- 10.27. Usando funções nativas
- 10.28. Usando funções de agregação
- 10.29. Agrupando registros com o group by
- 10.30. Exercício: usando group by
- 10.31. Diferença entre expressions, paths e predicates
- 10.32. Exercício: consultando pedidos com produto específico
- 10.33. Agrupando registros com funções no group by
- 10.34. Usando o having para condicionar o agrupamento
- 10.35. Usando a expressão case
- 10.36. Usando a expressão in
- 10.37. Usando distinct para evitar duplicações
- 10.38. Criando subqueries
- 10.39. Relacionando a subquery com a query principal
- 10.40. Criando subquery com a expressão in
- 10.41. Criando subquery com a expressão exists
- 10.42. Exercício: criando subqueries
- 10.43. Exercício: criando subqueries com in
- 10.44. Exercício: criando subqueries com exists
- 10.45. Criando subqueries com all
- 10.46. Criando subqueries com any
- 10.47. Exercício: criando subqueries com all
- 10.48. Atualizando objetos em lote
- 10.49. Removendo objetos em lote

11. Consultas nativas

- 11.1. Por que usar query nativa?
- 11.2. Executando SQL e retornando uma lista de arrays
- 11.3. Executando SQL e retornando uma entidade
- 11.4. Passando parâmetros para consulta nativa
- 11.5. Mapeando resultado de queries nativas com @SqlResultSetMapping
- 11.6. Usando @SqlResultSetMapping com @FieldResult
- 11.7. Usando @SqlResultSetMapping com @ColumnResult e retornando DTO
- 11.8. Usando uma @NamedNativeQuery
- 11.9. Adicionando consultas no arquivo XML
- 11.10. Exercício: mapeando retorno para um DTO
- 11.11. Invocando stored procedures com parâmetros in e out
- 11.12. Recebendo uma lista de registros da procedure
- 11.13. Exercício: atualizando registros com procedures
- 11.14. Configurando uma procedure com a anotação @NamedStoredProcedureQuery
- 11.15. Invocando uma view do banco de dados

12. Bean Validation, pool de conexões, Entity Graph e detalhes avançados

- 12.1. Validando objetos com Bean Validation
- 12.2. Exercício: validando objetos
- 12.3. Analisando anotações utilizadas
- 12.4. Entendendo o Pool de Conexões
- 12.5. Usando o HikariCP como gerenciador do pool de conexões
- 12.6. Buscando conexões de um nome JNDI
- 12.7. Criando um conversor de atributo
- 12.8. O problema do @OneToOne com o lazy no Hibernate

- 12.9. Entendendo e configurando um Entity Graph
- 12.10. Adicionando um Subgraph na consulta
- 12.11. Utilizando metamodel com Entity Graph
- 12.12. Configurando o Entity Graph através da anotação @NamedEntityGraph
- 12.13. Ajustando a configuração da entidade Pedido
- 12.14. Resolvendo o problema do N+1

13. Second Level Cache (cache compartilhado)

- 13.1. Entendendo o cache de segundo nível (ou shared cache)
- 13.2. Incluindo as entidades no cache
- 13.3. Removendo entidades do cache
- 13.4. Verificando se uma entidade está no cache
- 13.5. Modos de cache e a anotação @Cacheable
- 13.6. Fazendo controle dinâmico do cache
- 13.7. Configurando o EhCache como provedor
- 13.8. Fechando as instância de EntityManager dos exemplos de cache

14. Concorrência e locking

- 14.1. O que é concorrência e início da configuração dos exemplos
- 14.2. Resolvendo problemas de concorrência com Lock Otimista
- 14.3. Tipos que o atributo com @Version pode ter
- 14.4. Entendendo a diferença entre Lock Otimista e Lock Pessimista
- 14.5. Fazendo Lock Pessimista com PESSIMISTIC_READ
- 14.6. Fazendo Lock Pessimista com PESSIMISTIC_WRITE

- 14.7. Entendendo o que acontece se misturarmos mais de um tipo de lock
- 14.8. Outros tipos de lock
- 14.9. Lock com JPQL e Criteria API

15. Multitenancy

- 15.1. O que é Multitenancy (ou Multitenant) e os tipos de abordagem
- 15.2. Alteração na classe EntityManagerTest para melhorar a organização dos testes
- 15.3. Criando um novo schema no banco de dados
- 15.4. Implementando multitenancy com abordagem por schema
- 15.5. Implementando multitenancy com abordagem por máquina
- 15.6. Analisando uma aplicação web com Multitenant
- 15.7. Implementando multitenancy por coluna em uma aplicação web

16. PostgreSQL e EclipseLink

- 16.1. Instalando o PostgreSQL
- 16.2. Alterando as configurações para usar JPA com PostgreSQL
- 16.3. Alterando as configurações para usar EclipseLink como implementação do JPA

17. JPA em aplicações web

- 17.1. Reconhecendo o que aprendemos de JPA dentro de uma aplicação web
- 17.2. Configurando um projeto web com Spring MVC
- 17.3. Entendendo o JPA em um projeto com Spring MVC

- 17.4. Publicando nosso projeto no JBoss Wildfly
- 17.5. Configurando um projeto web com KumuluzEE
- 17.6. Entendendo o JPA em um projeto com KumuluzEE
- 17.7. Transações com RESOURCE_LOCAL vs JTA
- 17.8. Baixando e configurando o JBoss Wildfly
- 17.9. Publicando o projeto no JBoss Wildfly
- 17.10. Conclusão do curso e próximos passos

CURSOS LEGADOS

JOO

Java e Orientação a Objetos

1. Introdução

- 1.1. Como aprender Java?
- 1.2. A história do Java
- 1.3. As plataformas Java e como elas evoluem
- 1.4. Máquina virtual Java
- 1.5. Baixando, instalando e configurando a JDK
- 1.6. Exercício: instalação da JDK

2. Fundamentos da linguagem

- 2.1. Codificando, compilando e executando o programa "oi mundo"
- 2.2. Exercício: codificando um primeiro programa
- 2.3. Comentários

ALGAWORKS BLACK

- 2.4. Sequências de escape
- 2.5. Palavras reservadas
- 2.6. Convenções de código
- 2.7. Trabalhando com variáveis
- 2.8. Nomeando variáveis
- 2.9. Operadores aritméticos
- 2.10. Exercício: variáveis e operadores aritméticos
- 2.11. Tipos primitivos
- 2.12. Outros operadores de atribuição
- 2.13. Conversão de tipos primitivos
- 2.14. Promoção aritmética
- 2.15. Exercício: tipos primitivos e outros operadores de atribuição
- 2.16. Trabalhando com strings
- 2.17. Recebendo entrada de dados
- 2.18. Operadores de comparação e igualdade
- 2.19. Estruturas de controle if, else if e else
- 2.20. Exercício: Strings, entrada de dados, operadores de comparação e if else
- 2.21. Escopo de variáveis
- 2.22. Operadores lógicos
- 2.23. Exercício: operadores lógicos
- 2.24. Estrutura de controle switch
- 2.25. Operador ternário
- 2.26. Operadores de incremento e decremento
- 2.27. Estrutura de controle while
- 2.28. Estrutura de controle do-while
- 2.29. Estrutura de controle for
- 2.30. Cláusulas break e continue

- 2.31. Exercício: operador ternário, decremento e estruturas de repetição
- 2.32. Introdução e instalação do Eclipse IDE
- 2.33. Depurando códigos com o Eclipse
- 2.34. Exercício: instalando o Eclipse IDE

3. Orientação a Objetos - parte 1

- 3.1. O que é POO?
- 3.2. Classes e objetos
- 3.3. Criando uma classe com atributos
- 3.4. Instanciando objetos
- 3.5. Acessando atributos de objetos
- 3.6. Exercício: instanciando e acessando atributos do objeto
- 3.7. Composição de objetos
- 3.8. Valores padrão
- 3.9. Variáveis referenciam objetos
- 3.10. Criando, nomeando e chamando métodos
- 3.11. Métodos com retorno
- 3.12. Passando argumentos para métodos
- 3.13. Argumentos por valor ou referência
- 3.14. Exercício: composição de objetos e chamada de métodos

4. Wrappers, boxing e arrays

- 4.1. Wrappers do java.lang
- 4.2. Boxing
- 4.3. Desafio: wrappers e boxing

- 4.4. Trabalhando com arrays
- 4.5. Exercício: arrays

5. Orientação a Objetos - parte 2

- 5.1. Introdução à UML e diagrama de classes
- 5.2. Desafio: diagrama de classes
- 5.3. O objeto this
- 5.4. Construtores
- 5.5. Encapsulamento e modificadores de acesso public e private
- 5.6. Criando JavaBeans
- 5.7. Desafio: objeto this, construtores e JavaBeans
- 5.8. Organizando os projetos em pacotes
- 5.9. Modificador de acesso default
- 5.10. Modificadores static e final
- 5.11. Desafio: static e final
- 5.12. Enumerações
- 5.13. Desafio: pacotes e enumerações
- 5.14. Herança e modificador protected
- 5.15. Classe java.lang.Object
- 5.16. Sobreposição
- 5.17. Desafio: herança e sobreposição
- 5.18. Sobrecarga
- 5.19. Exercício: sobrecarga
- 5.20. Polimorfismo, casting de objetos e instanceof
- 5.21. Classes abstratas
- 5.22. Desafio: polimorfismo e classes abstratas
- 5.23. Interfaces

- 5.24. Exercício: interfaces e polimorfismo

6. Tópicos avançados

- 6.1. Coleta de lixo
- 6.2. Classe java.lang.Math
- 6.3. Desafio: classe java.lang.Math
- 6.4. Tratando e lançando exceções
- 6.5. Desafio: exceções
- 6.6. Classes String, StringBuffer e StringBuilder
- 6.7. Trabalhando com datas
- 6.8. Desafio: datas
- 6.9. Trabalhando com números
- 6.10. Desafio: números
- 6.11. Collections Framework
- 6.12. Métodos equals e hashCode
- 6.13. Desafio: collections
- 6.14. Arquivos JAR
- 6.15. Exercício: arquivos JAR
- 6.16. Documentação javadoc
- 6.17. Desafio: javadoc
- 6.18. Ordenando objetos

7. Mais detalhes e frameworks

- 7.1. Métodos da classe String
- 7.2. StringBuilder e StringBuffer
- 7.3. JOptionPane

- 7.4. JDBC
- 7.5. Varargs
- 7.6. Introdução a XML
- 7.7. Introdução ao JSON
- 7.8. Arquivos properties
- 7.9. Geração de javadoc
- 7.10. Introdução a expressão regulares
- 7.11. Introdução a generics
- 7.12. Logging com log4j
- 7.13. Debug com Eclipse
- 7.14. Maven
- 7.15. Testes de unidade com JUnit
- 7.16. Parâmetros da JVM

8. Entrada e saída - I/O

- 8.1. Gravando arquivo
- 8.2. Lendo arquivo
- 8.3. A classe Scanner

9. Serialização de objetos

- 9.1. Salvando e lendo objetos em arquivo
- 9.2. Enviando objetos na rede
- 9.3. Criando uma aplicação de chat

10. Novidades do Java 7

- 10.1. Separador de dígitos em literais numéricos
- 10.2. Switch case com String
- 10.3. Diamond
- 10.4. try-with-resources e multi-catch

11. Novidades do Java 8

- 11.1. Introdução ao Lambda
- 11.2. Referência a métodos
- 11.3. Interfaces funcionais
- 11.4. Introdução a Stream
- 11.5. API de Data - Parte 1
- 11.6. API de Data - Parte 2
- 11.7. API de Data - Parte 3
- 11.8. API de Data - Parte 4
- 11.9. Próximos passos
- 11.10. Conclusão

APLICAÇÕES WEB COM ANGULAR E REACT

Web Design Responsivo com HTML5, CSS3 e BEM

1. Introdução

- 1.1. Introdução ao curso
- 1.2. Apresentando o projeto
- 1.3. Preparando o ambiente de desenvolvimento

2. Começando com HTML

- 2.1. O que é HTML?
- 2.2. Estrutura básica do documento
- 2.3. DOCTYPE e codificação
- 2.4. Primeiras tags: títulos, quebras de linhas e parágrafos
- 2.5. Comentários
- 2.6. Ênfase, importância e marcação
- 2.7. Imagens
- 2.8. Âncoras (links)
- 2.9. Elementos estruturais

3. Começando com CSS

- 3.1. O que é CSS
- 3.2. Estilos incorporados e Regras CSS
- 3.3. Estilos em arquivos externos
- 3.4. Seletores de tipo, classe e ID
- 3.5. Agrupando seletores
- 3.6. Seletores descendentes
- 3.7. Seletores de filhos diretos
- 3.8. Cores
- 3.9. Formatação de textos
- 3.10. Inspeccionando com Chrome DevTools
- 3.11. Entendendo a propriedade display
- 3.12. Adicionando bordas
- 3.13. Espaçamento interno (padding)
- 3.14. Margens de elementos
- 3.15. Box model e a propriedade box-sizing

4. Iniciando o projeto do curso

- 4.1. Preparando o projeto
- 4.2. Criando o cabeçalho da página
- 4.3. Listas ordenadas e não-ordenadas
- 4.4. Adicionando os planos
- 4.5. Flutuando elementos
- 4.6. Usando pseudo-elementos
- 4.7. Configurando os planos lado a lado com float
- 4.8. Criando um botão

- 4.9. As pseudo-classes :focus e :hover
- 4.10. Reset CSS e Normalize.css

5. Web Design Responsivo

- 5.1. O que é Responsive Web Design?
- 5.2. Unidade de medida: pixel
- 5.3. Unidade de medida: percentual
- 5.4. Meta tag viewport
- 5.5. Layout fixo e fluído
- 5.6. Layout responsivo e media queries
- 5.7. Como funciona um sistema de Grid CSS
- 5.8. Sistema de grid do Bootstrap
- 5.9. Ajustando o projeto para usar Grid CSS

6. Especificidade, BEM e boas práticas

- 6.1. Especificidade do CSS
- 6.2. Caos no CSS: porque uma metodologia é importante?
- 6.3. A Metodologia BEM: seu código escalável
- 6.4. Como usar BEM na prática - parte 1
- 6.5. Como usar BEM na prática - parte 2
- 6.6. Ajustando o projeto do curso com BEM
- 6.7. Mais organização: CSS com Guidelines

7. Encerrando o projeto e mais CSS

- 7.1. Unidades de medida: em e rem

- 7.2. Ajustando unidades de medida no projeto
- 7.3. Adicionando chamada principal
- 7.4. Adicionando depoimento
- 7.5. Adicionando rodapé
- 7.6. Posicionamento estático e fixo
- 7.7. Posicionamento relativo
- 7.8. Posicionamento absoluto
- 7.9. Adicionando rótulo no plano
- 7.10. Ajustando margem do plano
- 7.11. Adicionando aspas no depoimento
- 7.12. Adicionando o bloco de navegação
- 7.13. Adicionando o menu para telas pequenas
- 7.14. Ajustando o menu para telas médias e grandes
- 7.15. JavaScript Hook: chaveando o menu
- 7.16. Criando um formulário: assinatura de plano
- 7.17. Concluindo o formulário de assinatura de plano
- 7.18. Entendendo as tabelas do HTML
- 7.19. Aplicando estilos em tabelas com CSS
- 7.20. Conclusão e próximos passos

Mergulhando no JavaScript

1. Colocando os pés na água

- 1.1. Introdução ao curso
- 1.2. Por que devo aprender JavaScript moderno? (ES6+)

- 1.3. Configurando o ambiente de desenvolvimento
- 1.4. Obtendo o melhor do suporte

2. A superfície do JavaScript

- 2.1. Introdução ao módulo
- 2.2. Hello World com JavaScript
- 2.3. Criando objetos
- 2.4. Criando Array de objetos
- 2.5. O princípio de funções
- 2.6. Condicionais
- 2.7. Programação funcional vs Programação imperativa
- 2.8. Truthy e Falsy

3. Mergulhando nos fundamentos do JavaScript

- 3.1. Relembrando alguns tópicos
- 3.2. Tudo dentro do JavaScript é considerado um objeto
- 3.3. Entendendo o uso do var
- 3.4. Declarando constantes (const)
- 3.5. Declarando variáveis (let)
- 3.6. Criando funções com function
- 3.7. Function.call e Function.apply
- 3.8. Criando arrow functions
- 3.9. Argumentos de função
- 3.10. Quando não usar arrow functions (Hoisting)
- 3.11. Trabalhando com template strings

- 3.12. Timers com setInterval e setTimeout

4. Orientação a Objetos

- 4.1. Introdução ao módulo de OOP
- 4.2. Criando uma classe
- 4.3. Métodos e propriedades dentro de uma classe
- 4.4. Constructor
- 4.5. Heranças
- 4.6. Object.prototype e Arrow function

5. Desestruturação e Spread

- 5.1. O que é desestruturação?
- 5.2. Desestruturando objetos
- 5.3. Desestruturando arrays
- 5.4. Desestruturando funções
- 5.5. Introdução ao Spread operator
- 5.6. Clonando objetos com spread
- 5.7. Arrays e spread
- 5.8. Utilizando Spread dentro de parâmetros de funções
- 5.9. Recebendo argumentos com Rest operator
- 5.10. Desestruturando um array com Rest operator

6. Manipulação de listas (Arrays)

- 6.1. Introdução ao módulo
- 6.2. Criando arrays a partir de listas iteráveis com Array.from

ALGAWORKS BLACK

- 6.3. Criando arrays do zero com Array.of
- 6.4. Iterando sobre itens de um array com forEach
- 6.5. Remapeando arrays com Array.map
- 6.6. Filtrando arrays com Array.filter
- 6.7. Reduzindo uma lista à um único valor com Array.reduce
- 6.8. Encontrando um registro específico em uma lista com Array.find
- 6.9. Validando elementos de uma lista com Array.some e Array.every

7. Promises e código assíncrono

- 7.1. Introdução às Promises
- 7.2. Criando uma Promise
- 7.3. Recebendo o valor da promise com then e catch
- 7.4. Async e Await
- 7.5. Múltiplas promises em paralelo com Promise.all
- 7.6. Promise chaining

8. Módulos e pacotes

- 8.1. Introdução ao NPM
- 8.2. Instalando e utilizando um pacote
- 8.3. Gitignore e node_modules

9. TypeScript

- 9.1. Por que usar TypeScript?

- 9.2. Tipando variáveis
- 9.3. Type aliasing
- 9.4. Tipando argumentos de funções
- 9.5. Tipando objetos e propriedades opcionais
- 9.6. Union vs Intersection
- 9.7. Interfaces
- 9.8. Exportando e importando interfaces e types
- 9.9. Types vs Interfaces
- 9.10. Generics
- 9.11. Conclusão e próximos passos

Fullstack Angular e Spring

1. Introdução ao REST

- 1.1. Introdução ao curso
- 1.2. Como usar o suporte
- 1.3. O que é SOFEA?
- 1.4. O que é REST?
- 1.5. Conhecendo o projeto do curso
- 1.6. Ambiente de desenvolvimento REST
- 1.7. Testando APIs com Postman
- 1.8. Introdução ao protocolo HTTP

2. Fundamentos do REST

- 2.1. O que é um recurso?
- 2.2. Representações de um recurso
- 2.3. Modelo de maturidade Richardson - Nível 0
- 2.4. Modelo de maturidade Richardson - Nível 1
- 2.5. Modelo de maturidade Richardson - Nível 2
- 2.6. Modelo de maturidade Richardson - Nível 3
- 2.7. HATEOAS
- 2.8. Segurança de APIs REST
- 2.9. Idempotência

3. Primeiras consultas e cadastros na API

- 3.1. Criando o projeto da API
- 3.2. Conectando ao MySQL
- 3.3. Migração de dados com Flyway
- 3.4. Consultando primeiro recurso com GET
- 3.5. Coleção vazia, o que retornar?
- 3.6. Cadastrando nova categoria com POST
- 3.7. Desafio: Retornar 404 caso não exista a categoria
- 3.8. Validando atributos desconhecidos
- 3.9. Tratando erros com ExceptionHandler
- 3.10. Validando valores inválidos com Bean Validation
- 3.11. Desafio: Criando o cadastro de pessoa
- 3.12. Usando eventos para adicionar header Location

4. Atualização e remoção de recursos na API

- 4.1. Removendo pessoa com DELETE
- 4.2. Sobre atualização de recursos REST
- 4.3. Atualizando pessoa com PUT
- 4.4. Implementando atualização parcial com PUT

5. Relacionamentos entre recursos REST

- 5.1. Criando a migração e entidade de lançamento
- 5.2. Desafio: Lista e busca de lançamentos
- 5.3. Desafio: Cadastrando o primeiro lançamento
- 5.4. Validando inconsistências
- 5.5. Validando lançamento com Bean Validation
- 5.6. Regra para não salvar pessoa inativa
- 5.7. Implementando pesquisa de lançamento com Metamodel
- 5.8. Desafio: Removendo lançamentos
- 5.9. Implementando a paginação de lançamentos

6. Segurança da API

- 6.1. Implementando autenticação Basic
- 6.2. Fluxo básico do OAuth
- 6.3. Implementando segurança com OAuth 2 e Password Flow
- 6.4. JSON Web Tokens - JWT
- 6.5. Configurando JWT no projeto
- 6.6. Renovando o access token com o refresh token
- 6.7. Movendo o refresh token para o cookie

- 6.8. Movendo o refresh token do cookie para a requisição
- 6.9. O que é CORS?
- 6.10. Criando filtro para CORS
- 6.11. Movendo o usuário para o banco de dados
- 6.12. Adicionando permissões de acesso
- 6.13. Desafio: Finalizando permissões de acesso
- 6.14. Implementando o logout

7. Deploy da API em produção

- 7.1. Implementando projeção de lançamento
- 7.2. Profiles do Spring
- 7.3. Criando a conta no Heroku
- 7.4. Deploy da API na nuvem
- 7.5. Nome do usuário no token JWT
- 7.6. Alternando OAuth 2 e Basic Security com profiles
- 7.7. Desafio: Pesquisa de pessoa
- 7.8. Ajustando o CEP
- 7.9. Desafio: Atualização de lançamento

8. Introdução ao Angular

- 8.1. O que é Angular?
- 8.2. AngularJS vs Angular 2/4/X: a confusão das versões
- 8.3. Instalando o Visual Studio Code
- 8.4. Introdução ao HTML
- 8.5. Introdução ao CSS
- 8.6. Instalando o Node.js e NPM

- 8.7. Instalando e criando um projeto com Angular CLI
- 8.8. Abrindo o projeto no VS Code
- 8.9. Abrindo e executando um exemplo do curso

9. Fundamentos do Angular, componentes e data binding

- 9.1. Bootstrapping e AppModule
- 9.2. O que são componentes
- 9.3. Criando um componente
- 9.4. Instalando a biblioteca CSS do Bootstrap
- 9.5. Introdução a data binding
- 9.6. Usando interpolação
- 9.7. Usando event binding
- 9.8. Usando variável de referência
- 9.9. Usando property binding
- 9.10. Usando two-way data binding
- 9.11. Introdução às diretivas
- 9.12. Exibindo condicionalmente com as diretivas ngIf e hidden
- 9.13. Iterando com a diretiva ngFor
- 9.14. Binding de propriedades customizadas com @Input
- 9.15. Binding de eventos customizados com @Output e EventEmitter
- 9.16. Adicionando estilos CSS em componentes
- 9.17. Estilos CSS dinâmicos com ngStyle
- 9.18. Classes CSS dinâmicas com ngClass

10. Páginas de pesquisa

- 10.1. Instalando plugins úteis no Visual Studio Code
- 10.2. Escolhendo uma biblioteca de componentes
- 10.3. Criando o projeto do curso e instalando o PrimeNG
- 10.4. Adicionando o formulário de pesquisa de lançamentos
- 10.5. Adicionando uma tabela de dados
- 10.6. Customizando colunas com ng-template
- 10.7. Fazendo paginação de dados
- 10.8. Adicionando tooltip
- 10.9. Colocando a tabela de dados responsiva
- 10.10. Criando o componente de pesquisa de lançamentos
- 10.11. Criando o componente de barra de navegação
- 10.12. Adicionando menu intercambiável
- 10.13. Desafio: criando componente de pesquisa de pessoas

11. Diretivas e pipes

- 11.1. Criando diretivas customizadas
- 11.2. Respondendo a eventos do hospedeiro com @HostListener
- 11.3. Vinculando propriedades do hospedeiro com @HostBinding
- 11.4. Usando property binding em diretivas customizadas
- 11.5. Exportando a API da diretiva para o template
- 11.6. Conhecendo e usando pipes
- 11.7. Passando parâmetros para pipes
- 11.8. Desafio: usando pipes

12. Formulários e validação

- 12.1. Introdução aos formulários
- 12.2. Template-driven Forms: Criando um formulário
- 12.3. Registrando os controles do formulário
- 12.4. Adicionando opções dinâmicas no campo de seleção
- 12.5. Definindo o valor padrão em campos com ngModel
- 12.6. Two-way binding com ngModel
- 12.7. Adicionando validação em formulários
- 12.8. Exibindo erro de validação do formulário
- 12.9. Exibindo erro de validação de controles do formulário
- 12.10. Rastreamento do estado em controles do formulário
- 12.11. Estilizando os campos inválidos com classes CSS do Angular
- 12.12. Estilizando os campos inválidos com Bootstrap
- 12.13. Limpando formulários (reset)

13. Páginas de cadastro

- 13.1. Criando o protótipo do formulário de cadastro de lançamentos
- 13.2. Adicionando seletor de data (componente Calendar)
- 13.3. Adicionando botão de seleção
- 13.4. Adicionando caixa de seleção (componente Dropdown)
- 13.5. Adicionando máscara de dinheiro com ng2-mask-money
- 13.6. Desafio: criando o protótipo do formulário de cadastro de pessoa
- 13.7. Adicionando campo com máscara (componente InputMask)
- 13.8. Validando controles de formulário com PrimeNG

ALGAWORKS BLACK

- 13.9. Criando componente de mensagem de erro de validação
- 13.10. Desafio: controles, validações e mensagens de erro
- 13.11. Desafio: criando mais componentes

14. Módulos do Angular

- 14.1. Introdução aos módulos
- 14.2. Criando um módulo e exportando um componente
- 14.3. Reexportando um módulo
- 14.4. Criando um componente interno do módulo
- 14.5. O que são Feature Modules?
- 14.6. Criando um Feature Module
- 14.7. Desafio: criando o feature module de pessoas
- 14.8. O que são Shared Modules?
- 14.9. Criando um Shared Module
- 14.10. O que é Core Module?
- 14.11. Desafio: criando o Core Module

15. Serviços e injeção de dependências

- 15.1. Introdução aos serviços
- 15.2. Implementando um serviço
- 15.3. O que é injeção de dependências?
- 15.4. Configurando o injetor com provider por classe
- 15.5. Configurando o injetor com provider por fábrica
- 15.6. Configurando o injetor com provider por valor e o decorator @Inject

- 15.7. Injetando serviços dentro de serviços e o decorador @Injectable
- 15.8. Como funciona o Injetor Hierárquico

16. Requisições HTTP

- 16.1. Por que precisamos de requisições HTTP?
- 16.2. Instalando e testando o json-server
- 16.3. Fazendo requisição com GET e recebendo o retorno
- 16.4. Fazendo requisição com POST
- 16.5. Fazendo requisição com DELETE
- 16.6. Fazendo requisição com PUT
- 16.7. Tratando erros de requisições HTTP

17. Implementando os serviços do projeto

- 17.1. Revisando e iniciando o back-end do projeto do curso
- 17.2. Criando o serviço de consulta de lançamentos
- 17.3. Adicionando filtro por descrição na pesquisa de lançamentos
- 17.4. Adicionando filtro por datas na pesquisa de lançamentos
- 17.5. Implementando a paginação no serviço de lançamentos
- 17.6. Configurando a paginação lazy do PrimeNG
- 17.7. Desafio: criando a consulta e listagem de pessoas
- 17.8. Excluindo lançamentos e o decorador @ViewChild
- 17.9. Adicionando mensagem de sucesso com Angular Toasty
- 17.10. Adicionando diálogo de confirmação antes da exclusão
- 17.11. Alterando o locale da aplicação para pt-BR
- 17.12. Criando um serviço de tratamento de erros

ALGAWORKS BLACK

- 17.13. Desafio: implementando a exclusão de pessoas
- 17.14. Desafio: mensagem de erro de usuário na exclusão de pessoa
- 17.15. Desafio: implementando a mudança de status de pessoas
- 17.16. Desafio: implementando o serviço de listagem de categorias
- 17.17. Listando as categorias cadastradas no dropdown
- 17.18. Desafio: listando as pessoas cadastradas no dropdown
- 17.19. Criando classes de modelo e usando no cadastro de lançamentos
- 17.20. Implementando o serviço de cadastro de lançamentos
- 17.21. Desafio: implementando o cadastro de pessoas

18. Roteamento e navegação

- 18.1. Introdução a rotas
- 18.2. Configurando rotas na aplicação
- 18.3. Navegando com Router Link
- 18.4. Estilizando links da rota ativa
- 18.5. Recebendo parâmetros da rota
- 18.6. Desafio: implementando os serviços de atualização e busca por código
- 18.7. Preenchendo os campos na edição de lançamentos
- 18.8. Salvando lançamentos editados
- 18.9. Implementando navegação imperativa
- 18.10. Fazendo redirecionamento
- 18.11. Tratando rota não encontrada
- 18.12. Definindo o título da página dinamicamente
- 18.13. Refatorando as rotas para usar Routing Module

- 18.14. Criando um Routing Module para o módulo de funcionalidade
- 18.15. Desafio: roteamento e edição de pessoas

19. Segurança do front-end

- 19.1. Introdução à segurança do front-end
- 19.2. Revisando a segurança da API com OAuth 2 e JWT
- 19.3. Desafio: módulo de segurança e protótipo da tela de login
- 19.4. Implementando o serviço de autenticação com OAuth 2
- 19.5. Decodificando o JWT e armazenando no Local Storage
- 19.6. Tratando casos de erros e sucesso de autenticação
- 19.7. Adicionando o Access Token nas chamadas HTTP
- 19.8. Exibindo o nome do usuário logado
- 19.9. Exibindo o menu do sistema conforme permissões do usuário
- 19.10. Obtendo um novo access token
- 19.11. Interceptando chamadas HTTP para tratar a expiração do access token
- 19.12. Protegendo componentes
- 19.13. Protegendo rotas com guarda de rotas (CanActivate)
- 19.14. E se o Refresh Token expirar?
- 19.15. Tratando acessos de usuários deslogados na AuthGuard
- 19.16. Implementando o logout

20. Deploy em produção do cliente Angular

- 20.1. Configurando a aplicação com environment do Angular CLI
- 20.2. Fazendo build para o ambiente de produção

- 20.3. Respondendo requisições com Node.js e Express
- 20.4. Fazendo deploy em produção no Heroku
- 20.5. Conclusão

21. Apêndice: Recursos avançados

- 21.1. Carregamento tardio de módulos (Lazy loading)
- 21.2. Formulários reativos
- 21.3. Criando um formulário reativo
- 21.4. Usando a propriedade formGroup
- 21.5. Configurando o HTML do formulário reativo
- 21.6. Criando validações customizadas

22. Apêndice: Melhorando o back-end

- 22.1. Preparação do retorno dos dados para os gráficos
- 22.2. Criando consulta para dados por categoria
- 22.3. Retornando os dados estatísticos de lançamento por categoria
- 22.4. Criando consulta para dados por dia
- 22.5. Retornando os dados estatísticos de lançamento por dia
- 22.6. Instalando o Jaspersoft Studio
- 22.7. Ajustando o layout do relatório
- 22.8. Criando o DTO do relatório
- 22.9. Criando os campos e parâmetros do relatório
- 22.10. Ajustando o título e o rodapé do relatório
- 22.11. Usando os campos do DTO no relatório
- 22.12. Criando a consulta do relatório

- 22.13. Gerando os bytes do relatório
- 22.14. Retornando os bytes do relatório na requisição
- 22.15. Criando um agendamento de tarefa (Scheduler)
- 22.16. Configurando o envio de e-mail
- 22.17. Enviando um e-mail simples
- 22.18. Configurando o template para o envio do e-mail
- 22.19. Processando o template e enviando o e-mail
- 22.20. Buscando lançamentos vencidos com Spring Data JPA
- 22.21. Agendando o envio de e-mail
- 22.22. Incluindo logs no agendamento do envio de e-mail
- 22.23. Criando a entidade Contato para suportar mestre-detalle
- 22.24. Resolvendo o StackOverflowError com @JsonIgnoreProperties
- 22.25. Inserindo uma pessoa com contato
- 22.26. Usando a propriedade orphanRemoval
- 22.27. Ignorando contatos da pessoa na pesquisa de lançamento
- 22.28. Upload de arquivos para API
- 22.29. Criando conta na Amazon AWS
- 22.30. Configurando o serviço S3
- 22.31. Criando o bucket no S3 automaticamente
- 22.32. Implementando o envio do arquivo para o S3
- 22.33. Enviando arquivos para o S3
- 22.34. Anexando arquivo no lançamento
- 22.35. Atualizando e removendo anexo
- 22.36. Configurando URL do anexo

23. Apêndice: Melhorando o front-end

- 23.1. Criando o módulo Dashboard
- 23.2. Plotando gráficos com dados estáticos
- 23.3. Criando o serviço da Dashboard
- 23.4. Buscando dados do gráfico de pizza
- 23.5. Buscando dados do gráfico de linhas
- 23.6. Formatando labels no Chart.JS
- 23.7. Criando módulo de relatórios
- 23.8. Configurando formulário do relatório
- 23.9. Exibindo o PDF para o usuário
- 23.10. Exercício: Incluindo itens de menu: dashboard e lançamento
- 23.11. Listando contatos na tela mestre-detalle
- 23.12. Criando o diálogo de contato
- 23.13. Criando o formulário de contato
- 23.14. Incluindo um novo contato
- 23.15. Corrigindo estilo do botão "Novo"
- 23.16. Editando contato
- 23.17. Removendo contato
- 23.18. Criando componente de contatos
- 23.19. Upload com o componente FileUpload
- 23.20. Fazendo download do anexo
- 23.21. Tratando erro de upload
- 23.22. Utilizando componente ProgressSpinner
- 23.23. Desabilitando botão "Salvar" no upload
- 23.24. Salvando e removendo anexo

24. Apêndice: Combos dependentes

- 24.1. Criando entidades cidade e estado
- 24.2. Criando pesquisa de estados e cidades
- 24.3. Buscando estados e cidades
- 24.4. Preenchendo Dropdown de estados
- 24.5. Carregando Dropdown de cidades
- 24.6. Validando cidade e estado, e salvando pessoa
- 24.7. Ajustando estado e cidade na pesquisa de pessoas

25. Apêndice: Spring Authorization Server com Authorization Code e PKCE

- 25.1. Conhecendo o fluxo Authorization Code com PKCE
- 25.2. Implementando o Spring Authorization Server
- 25.3. Testando o fluxo Authorization Code com PKCE com o método plain
- 25.4. Testando o fluxo Authorization Code com PKCE com o método SHA256
- 25.5. Configurando um domínio local
- 25.6. Gerando chave RSA
- 25.7. Direcionando login para o Authorization Server
- 25.8. Gerando o Code Challenge e State no Client
- 25.9. Recebendo o Authorization Code no Client e solicitando o Access Token
- 25.10. Implementando logout no Authorization Server
- 25.11. Integrando logout no Client

Especialista React

1. Introdução ao curso

- 1.1. Bem-vindo ao curso
- 1.2. Por que React?
- 1.3. Metodologia de componentes com React
- 1.4. Hands on com Codepen
- 1.5. Obtendo o melhor do suporte
- 1.6. Preparando o ambiente de desenvolvimento
- 1.7. Prévia do projeto final do curso

2. Introdução ao React

- 2.1. Introdução ao React
- 2.2. Entendendo mais sobre SPAs
- 2.3. Criando uma aplicação React com CRA
- 2.4. Entendendo o código gerado pelo CRA
- 2.5. Habilitando o ESLint no VSCode
- 2.6. Criando o componente de Hello World
- 2.7. Entendendo um pouco mais sobre o JSX
- 2.8. Incorporando expressões dentro do JSX
- 2.9. JSX também é uma expressão
- 2.10. Atributos com JSX
- 2.11. A segurança do JSX
- 2.12. Como o JSX é Renderizado no DOM
- 2.13. Atualizando um componente em tempo real

ALGAWORKS BLACK

- 2.14. Os estados dentro de componentes React (com o hook `useState`)
- 2.15. Como funcionam as props dentro do React
- 2.16. Encapsulando componentes com `props.children`
- 2.17. Introdução à estilização de componentes (atributo `style`)
- 2.18. Estilizando componentes com CSS (de forma dinâmica)
- 2.19. Estilizando componentes com `Styled Components`
- 2.20. A importância histórica dos componentes baseados em classe
- 2.21. Criando um `Class Component` (Componente baseado em classe)
- 2.22. Acessando props dentro de um `Class Component`
- 2.23. Estados em `Class Components`
- 2.24. Introdução aos ciclos de vida com `componentDidMount`
- 2.25. Recuperando atualizações do estado com `componentDidUpdate`
- 2.26. Limpando efeitos colaterais com `componentWillUnmount`
- 2.27. Renderização condicional de várias formas
- 2.28. Manipulação de eventos
- 2.29. Interligação com formulários
- 2.30. Dica preciosa para lidar com formulários
- 2.31. Renderizando listas

3. Hooks avançados

- 3.1. Introdução aos Hooks
- 3.2. "Ciclos de vida" com o hook `useEffect`
- 3.3. Criando um hook personalizado

- 3.4. O estado local dos hooks
- 3.5. Passando parâmetros para os hooks
- 3.6. Como realmente funciona o hook `useRef`
- 3.7. Gerenciamento de estado complexo com `useReducer`
- 3.8. Gerenciando funções imperativas do DOM com `useImperativeHandle`
- 3.9. Memorizando funções com `useCallback`
- 3.10. Trocando tempo por espaço com `useMemo`

4. Componentes documentados com StoryBook

- 4.1. Briefing do CMS (Projeto 1)
- 4.2. Introdução ao Figma
- 4.3. Entendendo Melhor o Figma
- 4.4. Configurando o StoryBook no projeto
- 4.5. Entendendo o StoryBook
- 4.6. Criando o boilerplate do Botão
- 4.7. Criando as variantes do botão
- 4.8. Recebendo todas as props de um button dentro do componente personalizado
- 4.9. Finalizando o componente de botão com o Polished
- 4.10. Criando a API do Input
- 4.11. Estilizando o componente Input
- 4.12. Adicionando estilização global (fontes) no Storybook
- 4.13. Criando o componente do ValueDescriptor
- 4.14. Uma nova forma de renderizar componentes de forma condicional

ALGAWORKS BLACK

- 4.15. Criando componente de parágrafo com estilização condicional
- 4.16. Desafio - FieldDescriptor
- 4.17. Utilizando Icones dentro do React
- 4.18. Desafio - Componente Confirm
- 4.19. Apresentando o React Table
- 4.20. Criando a estrutura da Tabela
- 4.21. Estilizando a tabela
- 4.22. Alinhando as colunas da tabela
- 4.23. Reutilizando o componente de Tabela
- 4.24. Recuperando dados além do accessor
- 4.25. Alinhando os Headers
- 4.26. Criando componente NoData
- 4.27. Aplicando o NoData dentro da tabela
- 4.28. Apresentando e instalando o ChartJS
- 4.29. Configurando a altura do Chart
- 4.30. Downgrade do ChartJS
- 4.31. Configurando as Legendas no Chart
- 4.32. Configurando a tensão da linha no Chart
- 4.33. Labels do ChartJS
- 4.34. Entendendo como estruturar os dados no Chart
- 4.35. Estilizando os datasets
- 4.36. Entendendo um pouco mais sobre os eixos do Chart
- 4.37. Removendo gridlines do Chart
- 4.38. Finalizando o visual do Chart
- 4.39. Propificando o componente de Chart
- 4.40. Desafio - Progress Bar
- 4.41. Iniciando o componente CircleChart

- 4.42. Documentando o CircleChart
- 4.43. Setup do CircleChart
- 4.44. Criando os componente estilizados do CircleChart
- 4.45. Finalizando (e apreciando) o CircleChart
- 4.46. Componente Image Upload
- 4.47. Desafio profile
- 4.48. Desafio - SessionController
- 4.49. Criando o contador de palavras
- 4.50. Introdução ao TagInput
- 4.51. Estilizando o TagInput
- 4.52. Exemplo funcional no StoryBook (com o TagInput)
- 4.53. Desafio - ErrorDisplay
- 4.54. Editor de texto em Markdown

5. Como funcionam as rotas

- 5.1. O que esperar do módulo
- 5.2. Dica valiosa para seguir com o módulo
- 5.3. Sobre conflitos
- 5.4. Instalando e Configurando o React Router DOM
- 5.5. Quem é o BrowserRouter
- 5.6. Quem é o Switch
- 5.7. Route e a propriedade exact
- 5.8. As views são só componentes
- 5.9. A nomenclatura dos arquivos
- 5.10. Recuperando parâmetros na rota
- 5.11. Recuperando parâmetros da Query String
- 5.12. Manipulando o histórico com useHistory

- 5.13. A facilidade do componente Link
- 5.14. Alterando o título da página conforme as rotas
- 5.15. Code Splitting

6. Páginas e funcionalidades do CMS

- 6.1. Introdução ao módulo
- 6.2. Instalando as fontes no projeto
- 6.3. Aplicando estilos globais com StyledComponents
- 6.4. Criando o layout base do sistema
- 6.5. Aplicando os componentes no layout
- 6.6. Estilizando o menu lateral com base na página atual
- 6.7. Criando as primeiras Features
- 6.8. Feature UserTopTags
- 6.9. Feature UserEarnings
- 6.10. Adicionando espaçamento no final da página
- 6.11. Criando a rota de lista de editores
- 6.12. Feature EditorsList
- 6.13. Criando o formulário de post
- 6.14. Fazendo o contador de palavras funcional
- 6.15. Página 404
- 6.16. Criando componente de confirmação
- 6.17. Replicado a lógica do confirm no Info
- 6.18. Transformando um método em uma macrotask (event loop)
- 6.19. Estilizando um componente de uma biblioteca externa
- 6.20. Desafio - Grid do perfil do editor
- 6.21. Escondendo dados sensíveis do usuário
- 6.22. Blindando a view com ErrorBoundary

7. O Docker que o front-end precisa saber

- 7.1. Introdução ao módulo
- 7.2. Introdução ao Docker
- 7.3. Instalando o Docker no Windows
- 7.4. Instalando o Docker no Linux
- 7.5. O que são Imagens
- 7.6. Criando um Container
- 7.7. Criando um Dockerfile
- 7.8. Como transformar uma API REST em um Container
- 7.9. O que são Volumes
- 7.10. Rede no Docker
- 7.11. O que é o Docker Compose
- 7.12. Removendo dados não utilizados pelo Docker

8. Interligando a SPA com a API

- 8.1. Introdução ao módulo
- 8.2. Front-end vs Back-end (e requisições HTTP)
- 8.3. Requisições HTTP (verbos, uri, headers e body)
- 8.4. O retorno das requisições HTTP (status code)
- 8.5. Subindo a infra com Docker
- 8.6. O que é OpenAPI
- 8.7. Enviando uma requisição HTTP com fetch
- 8.8. Alterando propriedades da requisição
- 8.9. Como o fetch lida com status code
- 8.10. Por que o Axios é melhor que o fetch
- 8.11. Services Layer Pattern

ALGAWORKS BLACK

- 8.12. Gerando interfaces TypeScript automaticamente com OpenAPI
- 8.13. Criando a classe de Serviço
- 8.14. Criando o primeiro serviço
- 8.15. Enviando dados (método POST) com axios
- 8.16. Lidando com parâmetros da API
- 8.17. A Developer Experience do SDK
- 8.18. Upload de arquivos em cloud
- 8.19. Criando FileService
- 8.20. Enviando o arquivo para o Storage
- 8.21. Abstraindo a lógica do upload
- 8.22. Implementando o cadastro de posts
- 8.23. Fazendo a lista de editores funcionar
- 8.24. O primeiro contato com manipulação de datas
- 8.25. Perfil de editor
- 8.26. Top 3 Tags
- 8.27. Ganhos do usuário
- 8.28. Transformando dados para o ChartJs
- 8.29. Trocando de usuário na API
- 8.30. Buscando dados paginados da API
- 8.31. Resolvendo pequenos erros de depreciação
- 8.32. Aplicando error boundaries
- 8.33. Limitações do error boundary
- 8.34. Criando um HOC de Boundary
- 8.35. Adicionando Skeleton na Interface
- 8.36. Criando um Loading
- 8.37. Fazendo o Loading funcionar
- 8.38. Desafio - Loading animado com blur
- 8.39. Prevenindo erros de passarem em branco com onUnhandledRejection
- 8.40. Paginação Server-side vs Client-side
- 8.41. Aplicando paginação no react-table
- 8.42. Paginando no servidor
- 8.43. React Paginate
- 8.44. Desafio - Criar modal de Preview de Post
- 8.45. Renderizando markdown
- 8.46. Desafio - Layout do Post
- 8.47. Recuperando post da API
- 8.48. Abrindo links em uma nova aba
- 8.49. Estilizando o título do post
- 8.50. Upload de imagem no post
- 8.51. Desabilitando botões por status do post
- 8.52. Publicando um post
- 8.53. Melhorando a experiência do cadastro de post
- 8.54. O problema do BrowserRouter
- 8.55. Edição do post

9. Criando e publicando um SDK

- 9.1. Introdução ao módulo
- 9.2. O que é um SDK?
- 9.3. Inicializando o SDK
- 9.4. Versionamento Semântico
- 9.5. Publicando o SDK
- 9.6. Instalando o SDK no CMS
- 9.7. A importância do prepublish

- 9.8. Implementando os primeiros serviços
- 9.9. Desafio - Serviço de Pagamentos
- 9.10. Desafio - Serviço de Fluxo de Caixa
- 9.11. Desafio - Finalizar o serviço de Usuários
- 9.12. Desafio - Finaliza o serviço de Posts
- 9.13. Content-Type personalizado
- 9.14. Publicando a versão final do SDK com Types
- 9.15. Mapeando erros da API
- 9.16. Identificando os tipos de erro
- 9.17. Substituindo o SDK interno

10. Gerenciamento de Estado Global

- 10.1. Introdução ao módulo
- 10.2. Introdução ao Flux
- 10.3. O que é o Redux
- 10.4. Instalando os pacotes necessários
- 10.5. Configurando a store raiz
- 10.6. Criando o primeiro slice
- 10.7. Redux DevTools
- 10.8. Criando o primeiro reducer
- 10.9. Criando a primeira action
- 10.10. Disparando uma ação
- 10.11. Acessando a store dentro de um componente
- 10.12. Estado global vs Estado local
- 10.13. As Thunks
- 10.14. Reagindo às Thunks
- 10.15. isFulfilled, isPending e isRejected

- 10.16. Reduzindo boilerplate com createReducer
- 10.17. Abstraindo o Redux com Hooks
- 10.18. Evitando loops infinitos nos Hooks
- 10.19. Migrando uma feature para o Redux
- 10.20. Nem sempre vale a pena usar o Redux
- 10.21. Desafio - Conectar o PostList no Redux
- 10.22. Desafio - Migrar features para os Hooks
- 10.23. Não refatorar pode ser o melhor caminho
- 10.24. Paramos por aqui

11. Server-Side Rendering com Next.js

- 11.1. Introdução ao módulo
- 11.2. Briefing do projeto
- 11.3. O que é SSR
- 11.4. Como funciona o SSR com Next
- 11.5. Iniciando um projeto com Next CLI
- 11.6. A estrutura gerada pelo projeto
- 11.7. Entendendo o arquivo index.ts
- 11.8. Páginas vs Componentes
- 11.9. Styled Components no Next
- 11.10. Criando o tema
- 11.11. Criando estilos globais
- 11.12. Criando o Layout
- 11.13. Convertendo a logo em um componente
- 11.14. Contendo o conteúdo da página
- 11.15. Credits no Footer
- 11.16. Usando o Link no NavBar

ALGAWORKS BLACK

- 11.17. Post em destaque
- 11.18. Fallback no avatar
- 11.19. Background com transparência no CSS
- 11.20. Adicionando um link no post de destaque
- 11.21. Responsividade no Post em destaque
- 11.22. getServerSideProps vs useEffect
- 11.23. Recuperando parâmetros da Query String
- 11.24. Validando parâmetros
- 11.25. Listando os posts
- 11.26. Criando a base do PostCard
- 11.27. Adicionando animação
- 11.28. Deixando a lista responsiva
- 11.29. React Paginate com SSR
- 11.30. Estilizando a paginação com Pseudo Elementos do CSS
- 11.31. Parâmetros nas rotas
- 11.32. Introdução à tratativa de erros com Next
- 11.33. Recebendo informações do erro na página
- 11.34. Reutilizando a lógica de renderização de erro
- 11.35. Componente Error do next
- 11.36. Identificando erros com instanceof
- 11.37. Página de 404 personalizada (com ilustrações)
- 11.38. Adicionando o slug na URL
- 11.39. Corrigindo o Slug
- 11.40. URL Canônica
- 11.41. Desafio - PostHeader
- 11.42. Formatando datas com SSR
- 11.43. Renderizando post em Markdown
- 11.44. Instalando plugins no Markdown

- 11.45. Syntax Highlighting com Remark
- 11.46. Adicionando Open Graph no artigo
- 11.47. Corrigindo o Code
- 11.48. Adicionando o Disqus
- 11.49. Página de Erro 500
- 11.50. Barra de progresso
- 11.51. Definindo o page como parâmetro opcional
- 11.52. Header responsivo
- 11.53. Header fixo

12. Conhecendo o Ant Design

- 12.1. Introdução ao módulo
- 12.2. O que é o Ant Design
- 12.3. Ant Design vs Material UI
- 12.4. Instalando e configurando o Ant Design
- 12.5. Limpando o projeto
- 12.6. Configurando o Prettier
- 12.7. Introdução ao Grid (Row e Col)
- 12.8. Entendendo a documentação (+Row)
- 12.9. Responsividade com Breakpoints
- 12.10. Tipografia
- 12.11. Adicionando e entendendo o Layout
- 12.12. Introdução ao Table
- 12.13. Introdução ao Input
- 12.14. Introdução ao Form

13. Criando a base do Admin

- 13.1. Introdução ao módulo
- 13.2. Briefing do Admin
- 13.3. Protótipo de baixa fidelidade (Wireframe)
- 13.4. Criando e Limpando o projeto com CRA
- 13.5. Instalando o Ant Design
- 13.6. Configurando o Redux no Projeto
- 13.7. Instalando o SDK
- 13.8. Layouts do Ant Design
- 13.9. Descentralizando o Layout
- 13.10. Instalando a biblioteca de gráficos
- 13.11. Transformando os dados do gráfico
- 13.12. Configurando o AreaChart
- 13.13. Configurando as legendas do gráfico
- 13.14. Configurando a label do eixo horizontal
- 13.15. Configurando os Tooltips no gráfico
- 13.16. Desabilitando o eixo vertical no gráfico
- 13.17. Definindo o tamanho do gráfico
- 13.18. Últimos posts (componente Card)
- 13.19. Espaçamento entre Rows (componente Space)
- 13.20. Responsividade no Layout
- 13.21. Impedindo "shrink" no conteúdo da página
- 13.22. Criando o menu de navegação
- 13.23. Criando a primeira rota
- 13.24. Criando as rotas base
- 13.25. Acessando as rotas pelo menu lateral
- 13.26. "Abrir em uma nova guia" no menu

- 13.27. Localizando o sidebar ao carregar a aplicação
- 13.28. Adicionando a logo no header
- 13.29. Configurando o CRACO no projeto
- 13.30. Less, tema escuro e variáveis
- 13.31. Todas as variáveis do Ant Design
- 13.32. Delimitando e centralizando o layout

14. Módulo de usuários do Admin

- 14.1. Tabela de usuários
- 14.2. Customizando renderização das colunas da tabela
- 14.3. Recuperando dados além da coluna
- 14.4. Impedindo as tabelas de quebrarem
- 14.5. Definindo uma largura para a coluna
- 14.6. Usuários no Redux
- 14.7. Alterando o status do usuário no Redux
- 14.8. Acessando com o perfil de gerente
- 14.9. Remapeando dados locais ao disparar uma action
- 14.10. O dispatch retorna uma Promise
- 14.11. Filtro de busca personalizado e reutilizável
- 14.12. Controlando loading da tabela
- 14.13. Paginação local
- 14.14. Tabela responsiva com Scroll
- 14.15. Propriedade responsive
- 14.16. Renderizando cards na tabela (mobile)
- 14.17. Resolvendo erros no console
- 14.18. Iniciando o formulário de usuário
- 14.19. Componente de abas (Tabs)

ALGAWORKS BLACK

- 14.20. Dados pessoais
- 14.21. Substituindo diretivas ngFor e vFor com Array fill map
- 14.22. Desafio - Dados bancários
- 14.23. Fazendo o upload do avatar
- 14.24. Recortando e redimensionando imagens no front-end
- 14.25. Corrigindo uns bugs no upload
- 14.26. Montando o objeto do formulário
- 14.27. Forçando a renderização de uma aba
- 14.28. Definindo os campos como obrigatórios
- 14.29. Identificando erros em abas com Array reduce
- 14.30. Otimizando o algoritmo de busca de erros
- 14.31. Controlando as abas por meio de estado
- 14.32. Controlando o estado do formulário
- 14.33. Tentando cadastrar o usuário na API
- 14.34. Remapeando os erros com regex
- 14.35. Definindo máximo e mínimo de caracteres
- 14.36. Validando com enum
- 14.37. Validadores personalizados
- 14.38. Aplicando máscara nos inputs
- 14.39. Desformatando valores antes de enviar para a API
- 14.40. Criando DTOs para enviar dados
- 14.41. Prevenindo erros na rede
- 14.42. Capturando erros de forma global com onUnhandledRejection
- 14.43. Tratando erros do Redux
- 14.44. Observando actions com middlewares
- 14.45. Validando o cadastro do usuário
- 14.46. O Bug do Switch
- 14.47. Ordenando na tabela
- 14.48. Recebendo um usuário existente no formulário
- 14.49. Transformando datas em objetos Moment
- 14.50. Recebendo o avatar como prop
- 14.51. Criando a funcionalidade de edição
- 14.52. Ação para a página de edição de usuários
- 14.53. Recuperando o usuário para edição na URL
- 14.54. Interface alternativa para recurso não encontrado
- 14.55. Desafio - Label dinâmica no botão do formulário
- 14.56. Desafio - Formulário responsivo
- 14.57. Detalhes do usuário
- 14.58. Skills do usuário
- 14.59. Descriptions com uma coluna
- 14.60. Responsividade com hook useBreakpoint
- 14.61. Ação de editar usuário
- 14.62. Dupla confirmação com Popconfirm
- 14.63. Desafio - Lista de posts
- 14.64. Impedindo múltiplos cliques ao cadastrar usuário
- 14.65. Criando o input monetário
- 14.66. Desafio - Componente de não encontrado
- 14.67. Desafio - usePageTitle
- 14.68. Paginando do lado do servidor
- 14.69. Exibindo de forma condicional os elementos do formulário
- 14.70. Formatando o telefone
- 14.71. z-index nas colunas fixas
- 14.72. Definindo o estado inicial em caso de edições de usuários
- 14.73. Corrigindo o Initial Value do CurrencyInput
- 14.74. Renderizando skills e posts de forma condicional

14.75. Revisando o módulo de usuários

15. Módulo de pagamentos do Admin

- 15.1. Tabela de pagamentos
- 15.2. Ações de pagamentos
- 15.3. Seleção de elementos na tabela
- 15.4. Corrigindo um endpoint no SDK
- 15.5. Filtro de pagamentos por mês
- 15.6. Traduzindo o Ant Design inteiro
- 15.7. Responsividade na lista de agendamentos
- 15.8. Desabilitando o Popconfirm
- 15.9. Abstraindo o Popconfirm
- 15.10. Atalho para perfil do usuário
- 15.11. Paginação e Ordenação Server Side
- 15.12. Layout base do detalhamento de pagamento
- 15.13. Segregando o Detalhamento em Features
- 15.14. Desafio - Segregar posts e bônus
- 15.15. Responsividade "freestyle"
- 15.16. Trigger do menu no canto
- 15.17. Estado de carregamento
- 15.18. Desafio - Implementar busca com parâmetro no pagamento
- 15.19. Imprimindo o relatório da página
- 15.20. Desafio - Botão de aprovação
- 15.21. A ordem das rotas importa
- 15.22. A base do cadastro de pagamento
- 15.23. Busca com acentos e case no Select
- 15.24. Filtrando apenas os usuários

- 15.25. Desabilitando datas do Date Picker
- 15.26. Dica para montar objetos complexos
- 15.27. Campos dinâmicos
- 15.28. Corrigindo o campo de bônus
- 15.29. Desafio - Layout da Previsualização do pagamento
- 15.30. Identificando mudanças em campos específicos
- 15.31. Evitando requisições desnecessárias com debounce
- 15.32. Recuperando a prévia do pagamento da API
- 15.33. Corrigindo o agendamento
- 15.34. Limpando a prévia em certos casos
- 15.35. Desafio - Componente alternativo para prévia de pagamento
- 15.36. Exibindo erro no componente alternativo
- 15.37. Estado de Loading
- 15.38. Buscando editores na API
- 15.39. Implementando o cadastro
- 15.40. Aprovando pagamentos em batch
- 15.41. Migrando a solução para o Flux
- 15.42. Colhendo os frutos do Flux
- 15.43. Limpando a seleção
- 15.44. Desafio - Aprovação individual e remoção de agendamentos

16. Módulo de fluxo de caixa do Admin

- 16.1. Lista de entradas
- 16.2. Filtrando os dados por mês
- 16.3. Controlando estado de seleção pela View
- 16.4. Feature de remoção em batch
- 16.5. Criando os Slices

ALGAWORKS BLACK

- 16.6. Convertendo o hook useCashFlow para o Redux
- 16.7. Base do gerenciamento de categorias
- 16.8. Criando o slice de categorias (+ dica TodoTree)
- 16.9. Validando os filtros
- 16.10. Formulário de cadastro de categoria
- 16.11. Cadastro de categoria
- 16.12. dispatch().unwrap()
- 16.13. Removendo categorias
- 16.14. Desafio - Atualizar categorias com Loading
- 16.15. Estruturando o formulário de despesa
- 16.16. Criando o objeto de cadastro
- 16.17. Listando categorias
- 16.18. Finalizando o DTO do lançamento
- 16.19. Cadastro dinâmico
- 16.20. Tratando erros no Redux com RejectWithValue
- 16.21. Desabilitando datas inválidas
- 16.22. Recuperando entrada existente para atualizar
- 16.23. Finalizando a edição de despesa
- 16.24. Desafio - Remover entrada individualmente
- 16.25. Renderizando detalhes da entrada
- 16.26. Verificando atualizações no registro
- 16.27. Combinando filtros na Query String
- 16.28. Mostrando o mês correto no título da página
- 16.29. Traduzindo o módulo de CRUD
- 16.30. Desafio - Responsividade

17. Blindando a aplicação (Segurança)

- 17.1. Introdução ao módulo de segurança
- 17.2. Stateful vs Stateless Authentication
- 17.3. OAuth 2.0
- 17.4. JWT
- 17.5. Authorization Code + PKCE
- 17.6. Habilitando a segurança na API
- 17.7. Criando o Authorization Service
- 17.8. Enviando para a tela de login
- 17.9. Primeiro token de acesso
- 17.10. Adicionando token de acesso nas requisições
- 17.11. Renovando o token
- 17.12. E quando o refresh token vence?
- 17.13. Um pouco mais sobre JWT
- 17.14. Decodificando um JWT
- 17.15. UI alternativa para erros de permissão
- 17.16. UI alternativa para erros de permissão do Redux
- 17.17. Desafio - UI alternativa
- 17.18. Buscando dados do usuário
- 17.19. Hook useAuth
- 17.20. Dropdown no perfil
- 17.21. Logout programático + Ver perfil
- 17.22. Resolvendo o bug do usuário
- 17.23. Desabilitando funções por permissões
- 17.24. Desabilitando campos com base em permissões
- 17.25. Corrigindo bug no cadastro

- 17.26. Desabilitando itens com base em perfil (e não em permissões)
- 17.27. Desafio - Desabilitar funções diversas
- 17.28. Breadcrumb
- 17.29. Desafio - Nome na home
- 17.30. Detalhes - Gerenciamento de categorias
- 17.31. Detalhes - Ações do usuário
- 17.32. Detalhes - Menu com Drawer
- 17.33. Detalhes - Pre-loading
- 17.34. Quando fazer logout e quando fazer login?
- 17.35. Um sintoma da falta de testes
- 17.36. Repassando o fluxo de autenticação para o CMS
- 17.37. Session Controller
- 17.38. Detalhamento do usuário logado
- 17.39. Acessando posts no blog por meio do CMS
- 17.40. Gerando um token com escopo limitado
- 17.41. Usando o token com escopo limitado
- 17.42. Bloqueando acesso ao sistema com base no perfil

APLICAÇÕES WEB COM JAVASERVER FACES

Desenvolvimento Web com JSF 2

1. Aplicações web com Java

- 1.1. Introdução ao curso de JSF
- 1.2. Introdução ao desenvolvimento web
- 1.3. Containers Java EE
- 1.4. Instalando o Tomcat
- 1.5. Exercícios: instalação do Tomcat
- 1.6. Instalando e configurando o Eclipse
- 1.7. Exercícios: instalação e configuração do Eclipse
- 1.8. Uma aplicação Java web simples
- 1.9. Importando um projeto do Github no Eclipse
- 1.10. Distribuindo aplicações em WAR
- 1.11. Administrando o Apache Tomcat
- 1.12. Exercícios: aplicação web simples com o Eclipse e Tomcat

2. Desenvolvimento com JavaServer Faces

- 2.1. O que é JSF?
- 2.2. Baixando uma implementação JSF

ALGAWORKS BLACK

- 2.3. Codificando o primeiro projeto
- 2.4. O arquivo web.xml
- 2.5. Managed beans
- 2.6. Exercício: primeiro projeto em JSF
- 2.7. Backing beans
- 2.8. Escopos de aplicação e sessão
- 2.9. Outros escopos de managed beans
- 2.10. Exercício: managed beans e seus escopos
- 2.11. Usando navegação implícita
- 2.12. Manipulando eventos de ação
- 2.13. Manipulando eventos de mudança de valor
- 2.14. Exercício: navegação e eventos
- 2.15. Ciclo de vida

3. Principais componentes

- 3.1. Atributos comuns de componentes
- 3.2. Entradas, saídas de texto e imagens
- 3.3. Menus, caixas de listagem e itens de seleção
- 3.4. Campos de checagem e botões radio
- 3.5. Botões e links
- 3.6. Painéis
- 3.7. Mensagens
- 3.8. Tabelas de dados
- 3.9. Componentes dentro de células
- 3.10. Aplicando estilos em tabelas
- 3.11. Arquivos JavaScript e CSS
- 3.12. Exercícios: usando componentes e aplicando estilos

- 3.13. Projeto do curso: prototipando tela de consulta de lançamento
- 3.14. Projeto do curso: prototipando tela de novo lançamento
- 3.15. Exercícios: prototipando telas do projeto do curso

4. Conversão e validação

- 4.1. Conversores de números e datas
- 4.2. Customizando mensagens de erro de conversão
- 4.3. Exercício: usando conversores
- 4.4. Usando validadores
- 4.5. Customizando mensagens de erro de validação
- 4.6. Atributo immediate
- 4.7. Exercício: usando validadores
- 4.8. Criando conversores personalizados
- 4.9. Criando validadores personalizados
- 4.10. Exercício: criando conversores e validadores
- 4.11. Projeto do curso: usando conversores e validadores
- 4.12. Projeto do curso: conversor personalizado
- 4.13. Projeto do curso: validador de data futura
- 4.14. Projeto do curso: validador condicional
- 4.15. Exercício: usando conversores e validadores no projeto do curso

5. Persistência de dados

- 5.1. Preparando um banco de dados MySQL
- 5.2. Conhecendo e configurando JPA 2 com Hibernate
- 5.3. Mapeamento objeto-relacional

- 5.4. Testando o Hibernate
- 5.5. Carregando menu de pessoas do banco de dados
- 5.6. Integrando as telas com Hibernate
- 5.7. Exclusão de lançamentos do banco de dados
- 5.8. Transações e o pattern Open Session in View
- 5.9. Pattern Repository
- 5.10. Implementando regras de negócio
- 5.11. Exercício: implementando o pattern repository e a camada de regras de negócio

6. JavaServer Faces Avançado

- 6.1. Suporte ao método GET
- 6.2. Regras de navegação explícitas
- 6.3. Aplicações JSF com AJAX
- 6.4. Exercício: método GET e AJAX
- 6.5. Template de páginas com Facelets
- 6.6. Criando componentes customizados
- 6.7. Exercício: template e componentes customizados
- 6.8. Internacionalização de sistemas
- 6.9. Segurança da aplicação com JAAS
- 6.10. Página 403, proteção de componentes e MD5
- 6.11. Exercício: internacionalização e segurança da aplicação

7. Colocando em produção

- 7.1. Preparando o ambiente Java em um servidor cloud
- 7.2. Fazendo deploy da aplicação na nuvem

- 8. Migrando para Java EE 7
- 8.1. Introdução ao Java EE 7
- 8.2. Rodando o projeto no Tomcat 8
- 8.3. Configurando o JSF 2.2
- 8.4. Novos namespaces do JSF 2.2
- 8.5. Navegação explícita (from-view-id requerido)
- 8.6. Componente ViewAction
- 8.7. HTML5 Friendly Markup - Atributos Pass-through
- 8.8. HTML5 Friendly Markup - Elementos Pass-through
- 8.9. Resource Library Contracts
- 8.10. Upload de arquivos com FileUpload
- 8.11. Enviando arquivos via Ajax e direto para o banco de dados
- 8.12. Servindo arquivos para download com JSF
- 8.13. Atualizando para o Hibernate 4.3
- 8.14. Conclusão e próximos passos

Sistemas Comerciais Java EE com CDI, JPA e PrimeFaces

1. Introdução

- 1.1. Introdução ao curso
- 1.2. Instalando e configurando o Eclipse
- 1.3. Instalando o Apache Tomcat
- 1.4. Iniciando o Apache Tomcat dentro do Eclipse
- 1.5. Introdução ao Maven
- 1.6. Instalando o plugin do Maven no Eclipse
- 1.7. Criando um projeto web no Eclipse com Maven
- 1.8. Gerando arquivo WAR em um projeto Maven
- 1.9. Baixando os exemplos do GitHub e importando no Eclipse
- 1.10. Introdução ao PrimeFaces
- 1.11. Começando com PrimeFaces
- 1.12. Para quem usa NetBeans: criando um projeto com PrimeFaces

2. Formulários e Ajax com PrimeFaces

- 2.1. PanelGrid, InputText, OutputLabel e CommandButton
- 2.2. Ajax e renderização parcial
- 2.3. Processamento parcial
- 2.4. Submissão parcial
- 2.5. Notificações com AjaxStatus

- 2.6. Entrada de senhas com Password
- 2.7. Entrada de textos com InputTextarea
- 2.8. Entrada de data e hora com Calendar
- 2.9. Caixa de listagem com SelectOneListbox
- 2.10. Menu de seleção com SelectOneMenu
- 2.11. SelectOneMenu com conteúdo customizado e filtro
- 2.12. Menus de seleção dependentes com SelectOneMenu e Ajax
- 2.13. Botão de rádio com SelectOneRadio
- 2.14. Caixa de checagem múltipla com SelectManyCheckbox
- 2.15. Campo de autossugestão com AutoComplete
- 2.16. AutoComplete com suporte a POJO
- 2.17. Máscara de entrada com InputMask
- 2.18. Caixa de checagem com SelectBooleanCheckbox
- 2.19. Exercício: formulários com PrimeFaces

3. Prototipação com formulários

- 3.1. Criando o projeto do curso com Maven
- 3.2. Criando o template com Facelets
- 3.3. Prototipando a página de login
- 3.4. Prototipando o cadastro de produto
- 3.5. Adicionando o componente AjaxStatus no layout padrão
- 3.6. Configurando máscara de dinheiro com jQuery
- 3.7. Desafio: prototipando o cadastro de cliente
- 3.8. Desafio: prototipando o cadastro de usuário

4. Exibindo painéis e dados com PrimeFaces

- 4.1. Painel com abas com TabView
- 4.2. TabView dinâmica
- 4.3. Tabela de dados com DataTable
- 4.4. Ordenando dados da DataTable por coluna
- 4.5. Paginação de dados com DataTable
- 4.6. Exercício: painéis e dados com PrimeFaces

5. Prototipação com painéis e dados

- 5.1. Prototipando a pesquisa de produtos
- 5.2. Prototipando a pesquisa de pedidos
- 5.3. Prototipando o cadastro de pedido
- 5.4. Desafio: prototipando pesquisa de clientes
- 5.5. Desafio: prototipando pesquisa de usuários
- 5.6. Desafio: prototipando listagem de endereços de clientes
- 5.7. Desafio: prototipando inclusão de usuário em grupos

6. Menus, mensagens e diálogos com PrimeFaces

- 6.1. Barra de menu com Menubar
- 6.2. Mensagens com Messages e Growl
- 6.3. Diálogo suspenso com Dialog
- 6.4. Diálogo de confirmação com ConfirmDialog
- 6.5. Desafio: diálogos e mensagens

7. Prototipação com menus e diálogos

- 7.1. Incluindo um menu no sistema de pedidos
- 7.2. Incluindo diálogo de confirmação de exclusão de produto
- 7.3. Desafio: ajustando a barra de menu
- 7.4. Desafio: prototipando inclusão de endereços de clientes
- 7.5. Desafio: incluindo diálogos de confirmação

8. Injeção de dependências com CDI

- 8.1. O que é injeção de dependências?
- 8.2. A especificação CDI e a implementação Weld
- 8.3. Configurando um projeto com JSF e CDI
- 8.4. Escopos de beans
- 8.5. Pontos de injeção
- 8.6. Qualificadores
- 8.7. Métodos produtores
- 8.8. Usando o `@ViewScoped` com CDI
- 8.9. Desafio: projeto com JSF e CDI

9. Tratamento de exceções e logging

- 9.1. Tratando a exceção `ViewExpiredException`
- 9.2. Tratando outras exceções
- 9.3. Logging de mensagens e erros com Log4J

10. Persistência de dados com JPA 2 e Hibernate

- 10.1. O que é ORM, JPA e Hibernate?
- 10.2. Configurando um projeto com JPA e Hibernate
- 10.3. Criando entidades do cadastro de clientes
- 10.4. Desafio: criando entidades do cadastro de usuários
- 10.5. Desafio: criando entidades do cadastro de produtos
- 10.6. Criando entidades do pedido de venda
- 10.7. Mapeando entidades do cadastro de clientes
- 10.8. Detalhes físicos no mapeamento do cadastro de clientes
- 10.9. Desafio: mapeando entidades do cadastro de usuários
- 10.10. Desafio: mapeando entidades do cadastro de produtos
- 10.11. Mapeando entidades do pedido de venda

11. Validação do modelo com Bean Validation

- 11.1. Introdução ao Bean Validation e Hibernate Validator
- 11.2. Configurando Bean Validation
- 11.3. Adicionando restrições no cadastro de produto
- 11.4. Adicionando restrições no pedido
- 11.5. Desafio: adicionando restrição no modelo
- 11.6. Mensagens em português e com label do campo
- 11.7. Customizando mensagens de validação
- 11.8. Composto novas restrições

12. Integrando páginas, serviços e repositórios

- 12.1. Criando um produtor CDI para injetar EntityManager
- 12.2. Implementando o repositório de categorias
- 12.3. Listener preRenderView chamado muitas vezes
- 12.4. Criando um conversor de categorias
- 12.5. Combo boxes dependentes de categorias e subcategorias
- 12.6. Implementando o serviço de cadastro de produto
- 12.7. Controlando transações em beans CDI
- 12.8. Implementando a pesquisa de produtos
- 12.9. Implementando a edição de produtos
- 12.10. Implementando a exclusão de produtos
- 12.11. Desafio: cadastro de usuários
- 12.12. Desafio: pesquisa de usuários
- 12.13. Desafio: edição e exclusão de usuários
- 12.14. Desafio: cadastro de clientes
- 12.15. Desafio: pesquisa de clientes
- 12.16. Desafio: edição e exclusão de clientes

13. Página mestre-detalle: cadastro de pedidos

- 13.1. Implementando a pesquisa de pedidos
- 13.2. Implementando o cadastro de pedidos sem itens
- 13.3. Implementando a edição de pedidos
- 13.4. Calculando valores da capa do pedido

- 13.5. Adicionando itens ao pedido
- 13.6. Atualizando e removendo itens do pedido
- 13.7. Salvando os itens do pedido
- 13.8. Alertando a falta de produtos no estoque
- 13.9. Emitindo pedidos
- 13.10. Cancelando pedidos
- 13.11. Controlando status de pedidos
- 13.12. Desafio: inclusão de endereços de clientes

14. Envio de e-mail com JavaMail e CDI

- 14.1. Instalando o Simple-Mail no repositório local do Maven
- 14.2. Configurando o projeto para envio de e-mail com CDI
- 14.3. Enviando e-mail com dados do pedido
- 14.4. Template de e-mail com Apache Velocity
- 14.5. Serviços profissionais para envio de e-mails
- 14.6. Desafio: enviando e-mail

15. Segurança da aplicação com Spring Security

- 15.1. Introdução à autenticação e autorização
- 15.2. Configurando o Spring Security no projeto
- 15.3. Controlando acesso às páginas
- 15.4. Criando um provedor de autenticação customizado
- 15.5. Exibindo o nome do usuário logado
- 15.6. Criando uma página de login customizada
- 15.7. Resolvendo o problema com requisição Ajax

- 15.8. Protegendo componentes
- 15.9. Desafio: protegendo páginas e componentes

16. Gráficos com PrimeFaces

- 16.1. Incluindo um gráfico de linhas com dados aleatórios
- 16.2. Criando a consulta para o gráfico de linhas
- 16.3. Populando o gráfico com dados dinâmicos
- 16.4. Desafio: gráfico de pizza

17. Relatórios com JasperReports e iReport

- 17.1. Introdução ao JasperReports e iReport
- 17.2. Criando um relatório de pedidos emitidos
- 17.3. Melhorando a formatação do relatório
- 17.4. Chamando o relatório de uma página JSF
- 17.5. Desafio: criando um novo relatório

18. Fazendo deploy na nuvem da Amazon AWS

- 18.1. Criando uma instância no Amazon EC2
- 18.2. Criando uma instância de MySQL no Amazon RDS
- 18.3. Instalando o JDK e Tomcat no servidor de produção
- 18.4. Fazendo deploy da aplicação
- 18.5. Testando o sistema na nuvem
- 18.6. Conclusão e próximos passos

19. Apêndice: Migrando para Java EE 7

- 19.1. Introdução ao Java EE 7
- 19.2. Atualizando as bibliotecas do projeto
- 19.3. Migrando para o Tomcat 8
- 19.4. Fazendo o projeto rodar depois das atualizações
- 19.5. Resolvendo o erro no gráfico do PrimeFaces
- 19.6. Resolvendo o problema com diálogos que não abrem
- 19.7. Namespaces do Java EE 7
- 19.8. Usando DOCTYPE do HTML5
- 19.9. Nova anotação `@ViewScoped` para beans CDI

20. Apêndice: Mais recursos do PrimeFaces

- 20.1. Validando entradas em client-side
- 20.2. Customizando mensagens de validação client-side
- 20.3. Estendendo validação client-side com JSF
- 20.4. Usando Metadata na validação client-side
- 20.5. Estendendo validação client-side com Bean Validation
- 20.6. Configurando validação client-side com anotações terceiras
- 20.7. Implementando conversor client-side para Categoria
- 20.8. Hackeando o código de validação client-side do PrimeFaces
- 20.9. Abrindo páginas externas em diálogos com Dialog Framework
- 20.10. Implementando diálogo para seleção de clientes
- 20.11. Exception Handler do PrimeFaces
- 20.12. Controlando o foco de formulários
- 20.13. Exportando para Excel com DataExporter
- 20.14. Carregamento lazy em DataTable

- 20.15. Usando o tema do Bootstrap
- 20.16. Injeção de beans CDI em conversores JSF
- 20.17. Componente ViewAction
- 20.18. Configurando um pool de conexões com C3P0
- 20.19. Relatório com JasperSoft Studio
- 20.20. Escopo ConversationScoped

21. PrimeFaces 6, Spring Security 4, Hibernate 5 e Log4j 2

- 21.1. Atualizando para PrimeFaces 6
- 21.2. Atualizando para Spring Security 4 com configuração programática
- 21.3. Atualizando para Log4j 2
- 21.4. Atualizando para Hibernate 5
- 21.5. Criteria do Hibernate Depreciado: Usando Criteria API do JPA
- 21.6. Desafio: Usando Criteria API do JPA

PrimeFaces Responsivo

1. Introdução

- 1.1. Introdução ao workshop
- 1.2. O que é Responsive Web Design?
- 1.3. PrimeFaces é responsivo?
- 1.4. Apresentando o projeto
- 1.5. Ambiente de desenvolvimento do projeto

2. Layout das páginas

- 2.1. Meta tag viewport
- 2.2. Layout fixo
- 2.3. Layout fluido
- 2.4. Layout responsivo e media queries
- 2.5. Sistema de Grid CSS do PrimeFaces
- 2.6. Criando o topo
- 2.7. Criando a barra lateral
- 2.8. Criando a área de conteúdo
- 2.9. Intercambiando a barra lateral
- 2.10. Criando o menu do sistema
- 2.11. Criando o arquivo de template

3. Componentes responsivos

- 3.1. DataTable responsiva com prioridade de colunas

- 3.2. DataTable responsiva com reflow
- 3.3. Configurando o tema do Bootstrap
- 3.4. TabView responsivo
- 3.5. PanelGrid responsivo e componentes fluidos (ui-fluid)
- 3.6. Dialog responsivo

4. Finalizando o projeto

- 4.1. Adicionando endereços
- 4.2. Adicionando clientes
- 4.3. Editando clientes
- 5. Layouts e temas premium do PrimeFaces
 - 5.1. Introdução aos layouts e temas
 - 5.2. Usando o layout
 - 5.3. Configurando o tema
 - 5.4. Conclusão

APLICAÇÕES COM SPRING FRAMEWORK

Explorando a Linguagem JavaScript

1. Introdução

- 1.1. Introdução ao workshop
- 1.2. Como usar o suporte
- 1.3. Introdução ao JavaScript
- 1.4. Instalando o Node.js no Mac
- 1.5. Instalando o Node.js no Windows
- 1.6. Rodando JavaScript no Sublime Text no Mac
- 1.7. Rodando JavaScript no Sublime Text no Windows

2. Fundamentos do JavaScript

- 2.1. Variáveis e hoisting
- 2.2. Tipagem dinâmica
- 2.3. Tipo Number
- 2.4. Tipo String
- 2.5. Tipo Boolean

- 2.6. O operador typeof
- 2.7. Operadores aritméticos
- 2.8. Operadores de comparação
- 2.9. Operadores lógicos
- 2.10. Estrutura de controle if, else if, else
- 2.11. Operador ternário
- 2.12. Estrutura de controle for
- 2.13. Estrutura de controle while
- 2.14. Valores null e undefined

3. Funções

- 3.1. Introdução a funções
- 3.2. Alert e console.log
- 3.3. Funções matemáticas
- 3.4. Criando funções
- 3.5. Escopo de função e global
- 3.6. Closures

4. Introdução a objetos

- 4.1. Introdução a objetos
- 4.2. Criando objetos
- 4.3. Objetos dentro de objetos
- 4.4. Apagando propriedades do objeto
- 4.5. Iterando sobre propriedades do objeto
- 4.6. Comparando objetos

5. Arrays

- 5.1. Introdução a arrays
- 5.2. Criando arrays
- 5.3. Array vs Object
- 5.4. Adicionando e removendo elementos do array
- 5.5. Iterando nos elementos com forEach
- 5.6. Funções toString() e join()
- 5.7. Elementos do array
- 5.8. Mais da API de arrays

6. JavaScript no Browser

- 6.1. Conhecendo o Developer Tools
- 6.2. Debug com o Developer Tools
- 6.3. Boas práticas
- 6.4. Objeto window
- 6.5. CDN vs local hosting

7. jQuery

- 7.1. Introdução ao jQuery
- 7.2. Seletores
- 7.3. CSS
- 7.4. Introdução ao HTTP
- 7.5. Ajax e promises
- 7.6. Manipulação do DOM
- 7.7. Plugins jQuery

8. Trabalhando com eventos

- 8.1. Introdução a eventos
- 8.2. Eventos de manipulação do DOM
- 8.3. Disparando eventos
- 8.4. Temporizadores

9. Funções e objetos

- 9.1. Entendendo o protótipo do objeto
- 9.2. Como funciona o shadowing
- 9.3. Funções fábrica
- 9.4. Funções construtoras
- 9.5. Module Pattern
- 9.6. Propriedade prototype das funções
- 9.7. Usando a função bind
- 9.8. Disparando eventos customizados
- 9.9. Chamando funções através de call e apply
- 9.10. Namespaces
- 9.11. Conclusão

Spring Framework Expert

1. Introdução

- 1.1. Introdução ao curso
- 1.2. Conhecendo o projeto do curso
- 1.3. Como usar o suporte
- 1.4. Introdução ao protocolo HTTP
- 1.5. Introdução ao Maven
- 1.6. Instalando e configurando o Eclipse
- 1.7. Instalando o Apache Tomcat
- 1.8. Iniciando o Tomcat no Eclipse
- 1.9. Instalando e configurando o MySQL no Mac
- 1.10. Instalando e configurando o MySQL no Windows
- 1.11. Instalando o myJrebel no Eclipse

2. Introdução ao Spring

- 2.1. O Spring
- 2.2. Spring vs Java EE
- 2.3. O padrão MVC
- 2.4. Spring MVC vs JSF
- 2.5. Por dentro do Spring MVC

3. Spring MVC, Thymeleaf e Bean Validation

- 3.1. Criando o projeto

- 3.2. Conhecendo o Thymeleaf
- 3.3. Configurando o Spring
- 3.4. Configurando a ViewResolver com Thymeleaf
- 3.5. O que fazer primeiro do MVC? M, V ou C?
- 3.6. Começando o cadastro de cerveja
- 3.7. Introdução a validação do formulário
- 3.8. Forward e Redirect
- 3.9. Desafio: Validando a descrição
- 3.10. Resolvendo problemas de acentuação

4. O Thymeleaf

- 4.1. O que é uma template engine?
- 4.2. Thymeleaf vs JSP
- 4.3. Mantendo os dados no formulário
- 4.4. Thymeleaf e Spring
- 4.5. Framework extensível
- 4.6. Plugin do Eclipse para o Thymeleaf

5. Layout responsivo com Bootstrap e HTML5

- 5.1. Conhecendo os templates do Bootstrap
- 5.2. Conhecendo o layout do projeto
- 5.3. Baixando e configurando o layout no projeto
- 5.4. Prototipando o cadastro da cerveja
- 5.5. Melhorando a organização do Layout
- 5.6. Links com URL Expression
- 5.7. Configurando máscara de dinheiro com jQuery

- 5.8. Organizando o layout com fragmentos
- 5.9. Prototipando o cadastro rápido de estilo
- 5.10. Desafio: Prototipando o cadastro de cliente
- 5.11. Desafio: prototipando o cadastro de usuário
- 5.12. Bootstrap Switch no status do usuário
- 5.13. Desafio: prototipando o cadastro de cidade
- 5.14. Desafio: prototipando o cadastro de estilo
- 5.15. Adicionando o favicon

6. Logging

- 6.1. Introdução
- 6.2. Configurando o SLF4J com Log4j 2
- 6.3. Configurando níveis de log com Log4j2

7. Injeção de dependência com Spring IoC

- 7.1. Introdução
- 7.2. O ApplicationContext
- 7.3. Autowired
- 7.4. Escopo dos beans

8. JPA, Hibernate e Flyway

- 8.1. Introdução
- 8.2. Introdução ao JPQL e Hibernate Criteria
- 8.3. Mapeamento da entidade Cerveja
- 8.4. Migrando o banco de dados com Flyway

9. Spring Data JPA e Padrão Repository

- 9.1. Introdução
- 9.2. Configurando o projeto
- 9.3. O JpaRepository
- 9.4. Repository vs DAO
- 9.5. Montando os combos e radio do cadastro da cerveja
- 9.6. Usando o th:field
- 9.7. Fazendo o bind do estilo com a cerveja
- 9.8. Iniciando a transação para salvar a cerveja

10. Validação customizada e conversores

- 10.1. Mostrando mensagens do cadastro de cerveja
- 10.2. Validação customizada com Bean Validation
- 10.3. Convertendo valores numéricos
- 10.4. Desafio: Validando campos da cerveja
- 10.5. Formatando input com erro
- 10.6. Desafio: Implementar o cadastro do estilo

11. Tratando exceções, callbacks JPA e Ajax

- 11.1. Validando se existe um estilo persistido
- 11.2. Salvando o estilo com Ajax: diálogo de cadastro rápido
- 11.3. Mapeando o controller e mais sobre o ResponseEntity
- 11.4. Tratando exceções com ExceptionHandler
- 11.5. Callbacks JPA - Salvando SKU com caixa alta

12. Modularizando o JavaScript

- 12.1. Module Pattern e Namespaces
- 12.2. Modularizando a máscara de dinheiro
- 12.3. Modularizando o cadastro rápido de estilo

13. Estendendo o Thymeleaf

- 13.1. Dialeto e processadores
- 13.2. Novo atributo para classe de erro
- 13.3. Novo elemento para mensagens
- 13.4. Desafio: Removendo duplicação da tag html

14. Upload da foto e retorno assíncrono

- 14.1. Conhecendo o UIKit
- 14.2. Componente de upload da foto com drag and drop
- 14.3. Upload da foto com Ajax
- 14.4. Melhorando a disponibilidade da aplicação - retorno assíncrono
- 14.5. Salvando o nome da foto da cerveja
- 14.6. Criando pastas para salvar fotos
- 14.7. Salvando a imagem temporária
- 14.8. Mostrando a foto na tela
- 14.9. Refatorando script do upload da foto
- 14.10. Mantendo a foto na validação

15. Pesquisa, redimensionamento da foto, paginação e ordenação

- 15.1. Detalhe campos obrigatórios
- 15.2. Prototipando a pesquisa da cerveja
- 15.3. Redimensionando a imagem - salvando o thumbnail
- 15.4. Filtrando e resolvendo o problema do n+1
- 15.5. Paginação no cliente vs paginação no servidor
- 15.6. Prototipando a paginação
- 15.7. Paginação na pesquisa de cervejas
- 15.8. Mostrando total de páginas dinamicamente
- 15.9. Destacando a página selecionada
- 15.10. Implementando botões previous e next
- 15.11. Mantendo o filtro nas páginas
- 15.12. Prototipando a ordenação dinâmica
- 15.13. Parâmetros da ordenação
- 15.14. Ordenando a pesquisa por SKU ou nome
- 15.15. Componente Thymeleaf para ordenação
- 15.16. Desafio: Pesquisa de estilos
- 15.17. Desafio: Componente para paginação
- 15.18. Bean útil para paginação
- 15.19. Corrigindo bug na pesquisa

16. Cadastro com máscaras e combo dependente

- 16.1. Configurando máscara de telefone com jQuery

- 16.2. Atualização do Thymeleaf Layout Dialect
- 16.3. Configurando máscara de CPF/CNPJ com jQuery
- 16.4. Implementando radio TipoPessoa
- 16.5. Desafio: Configurando máscara do CEP com jQuery
- 16.6. Aplicando migração estado e cidade
- 16.7. Selecionando cidades pelo estado com Ajax
- 16.8. Aplicando migração de cliente
- 16.9. Agrupando validações para CPF ou CNPJ
- 16.10. Salvando CPF/CNPJ sem formatação
- 16.11. Validando cliente já cadastrado
- 16.12. Desafio: Começando a pesquisa de clientes
- 16.13. Finalizando pesquisa de clientes
- 16.14. Desafio: Finalizar cadastro de cidade
- 16.15. Desafio: Implementando pesquisa de cidades

17. Otimizando a performance com cache

- 17.1. Cacheando a busca de cidades por estado
- 17.2. Invalidando o cache no cadastro da cidade
- 17.3. Cache profissional com Guava do Google

18. Cadastro com relacionamento ManyToMany

- 18.1. Planejando o modelo de grupos e permissões
- 18.2. Criando a migração do usuário, grupo e permissão
- 18.3. Desafio: Começando o cadastro do usuário
- 18.4. Validador customizado para confirmação de senha

- 18.5. Input e componente para Data
- 18.6. Desafio: Salvando o usuário no banco de dados
- 18.7. Salvando o status do usuário
- 18.8. Salvando o usuário com grupos
- 18.9. Salvando a senha criptografada com BCrypt

19. Segurança com Spring Security

- 19.1. Introdução à autenticação e autorização
- 19.2. Configurando o Spring Security no projeto
- 19.3. Tela de login customizada
- 19.4. Autenticando usuário e senha no banco de dados
- 19.5. Restringindo o acesso às páginas
- 19.6. Carregando as permissões do usuário
- 19.7. Adicionando tela de acesso negado
- 19.8. Entendendo o CSRF
- 19.9. Configurando o CSRF
- 19.10. Problema de acentuação voltou?
- 19.11. Expirando sessão com novo login
- 19.12. Duração da sessão
- 19.13. Exibindo o usuário logado
- 19.14. Escondendo componentes de usuários sem permissão
- 19.15. Segurança nos métodos

20. Páginas de erros customizadas

- 20.1. Acessando a página 403 - Acesso negado
- 20.2. Criando página 404 - Não encontrado

- 20.3. Desafio: criando página 500 - Erro no servidor

21. Pesquisa com filtros avançados e multisseleção de linhas

- 21.1. Implementando menu lateral
- 21.2. Desafio: Protótipo da pesquisa de usuários
- 21.3. Trabalhando no filtro por grupo
- 21.4. Multisseleção de linhas - ativando/desativando vários usuários
- 21.5. Melhorando usabilidade da página de pesquisa de usuários
- 21.6. Paginação da pesquisa de usuários

22. Diálogo e autocomplete

- 22.1. Prototipando cadastro de nova venda
- 22.2. Prototipando pesquisa rápida de clientes
- 22.3. Implementando a pesquisa rápida de clientes
- 22.4. Selecionando o cliente na pesquisa rápida
- 22.5. Conhecendo o componente EasyAutocomplete
- 22.6. Pesquisando cervejas por sku ou nome
- 22.7. Renderizando cervejas no autocomplete
- 22.8. Prototipando itens da venda

23. Página mestre-detalhe, escopo de sessão e testes unitários

- 23.1. Como funciona o escopo de sessão?
- 23.2. Introdução aos testes unitários
- 23.3. Criando tabela de itens de venda
- 23.4. Testando a tabela de itens de venda
- 23.5. Mantendo os itens da venda no servidor
- 23.6. Renderizando HTML retornado do Ajax
- 23.7. Adicionando mesmos itens na venda
- 23.8. Implementando atualização da quantidade de itens
- 23.9. Prototipando remoção de itens da venda
- 23.10. Removendo itens da venda
- 23.11. Simulando um escopo de view
- 23.12. Atualizando valor total
- 23.13. Desafio: Box do valor total negativo
- 23.14. Desafio: Criando mapeamento para tabelas de venda
- 23.15. Salvando a venda no banco de dados
- 23.16. Criando validador customizado do Spring
- 23.17. Emitindo uma venda
- 23.18. Desafio: Implementando a pesquisa das vendas
- 23.19. Ajustando menu lateral

24. Envio de e-mails e chamadas assíncronas

- 24.1. Escolhendo o serviço de e-mail
- 24.2. Configuração do servidor de e-mail com arquivos externos
- 24.3. Configuração de chamadas assíncronas

- 24.4. Enviando e-mails simples
- 24.5. Criando template para e-mail
- 24.6. Enviando o e-mail em html com imagens
- 24.7. Formatando valores numéricos no e-mail

25. Implementando edição e exclusão

- 25.1. Implementando a exclusão de cerveja
- 25.2. Implementando a edição de cerveja
- 25.3. Implementando a edição de usuário
- 25.4. Implementando a edição da venda
- 25.5. Implementando o cancelamento da venda
- 25.6. Desafio: Implementando edições e exclusões

26. Dashboard - consultas avançadas e gráficos

- 26.1. Prototipando o dashboard
- 26.2. Box de vendas no ano, no mês e ticket médio
- 26.3. Conhecendo o Chartjs
- 26.4. Gráfico - Vendas por mês
- 26.5. Consultas em arquivos externos
- 26.6. Desafio: Box total de clientes, valor total e itens no estoque
- 26.7. Eventos para controle de estoque
- 26.8. Desafio: Gráfico - Vendas por origem

27. Internacionalização

- 27.1. O header "Accept-Language"
- 27.2. Configurando a internacionalização e o Thymeleaf
- 27.3. Internacionalizando as mensagens de validação
- 27.4. Desafio: internacionalizando sistema

28. Deploy em produção na nuvem

- 28.1. Novo plugin para formatação de moeda
- 28.2. Novas versões Thymeleaf e Layout
- 28.3. Deploy em um Tomcat externo
- 28.4. Criando conta na AWS e bucket no S3
- 28.5. Buscando imagens a partir da URL
- 28.6. Upload das fotos para o S3 da Amazon
- 28.7. Escolhendo um fornecedor na nuvem
- 28.8. Fazendo deploy na nuvem

29. Relatórios com JasperReports

- 29.1. Introdução ao JasperReports
- 29.2. Criando relatório de vendas emitidas
- 29.3. Melhorando a formatação do relatório
- 29.4. Emitindo relatório com Spring MVC

30. Spring Boot

- 30.1. Introdução

- 30.2. Criando sua primeira aplicação com o Spring Boot
- 30.3. Spring Boot na nuvem
- 30.4. Vale a pena fazer a migração?
- 30.5. Migração da app Brewer
- 30.6. Próximos passos e conclusão

31. Apêndice: Atualizando para o Spring Framework 5

- 31.1. Alterando o Spring Framework para versão 5
- 31.2. Utilizando a interface WebMvcConfigurer em WebConfig
- 31.3. Utilizando a API do JasperReports diretamente
- 31.4. Usando JCache com EhCache

32. Apêndice: Configurando o Spring Boot 2

- 32.1. Usando o STS
- 32.2. Reorganizando o pom.xml
- 32.3. Alterando classes de configuração
- 32.4. Corrigindo método findOne
- 32.5. Corrigindo importação da classe ClassForErrorAttributeTagProcessor
- 32.6. Alterar profile da classe FotoStorageLocal
- 32.7. Configurar conexão com o banco de dados
- 32.8. Criando classe BrewerApplication
- 32.9. Renomear arquivo consultas-nativas.xml
- 32.10. Ajustando classe PaginacaoUtil
- 32.11. Ajuste na classe CervejaEntityListener

ALGAWORKS BLACK

- 32.12. Criando classes conversoras para datas
- 32.13. Criando classes conversoras para moedas
- 32.14. Corrigir caminho da imagem da cerveja
- 32.15. Usar a propriedade user.home na classe FotoStorageLocal
- 32.16. Usando as novas anotações @NotBlank e @Email
- 32.17. Removendo método com @InitBinder
- 32.18. Substituindo o Log4j
- 32.19. Ajustar páginas de erro
- 32.20. Publicando no Heroku