

Apostila do curso FJ-22

Lab. Java com Testes, JSF e Design Patterns

Um bom programador não conhece apenas a linguagem que usa, mas foca nas **boas práticas** e nas **ferramentas certas**. Este material gratuito é o que usamos no [curso de Java](#) da Caelum e esperamos que seja útil no seu aprendizado. Não deixe também de [compartilhar](#) essa apostila com seus amigos.

A **Caelum** oferece [cursos de TI](#) desde 2004 em todo o Brasil. É conhecida por seus cursos nas áreas de [Java](#), [Ruby](#), [mobile](#), [front-end](#), [.NET](#) e [agile](#), além de [cursos online](#). Temos diversas [apostilas abertas](#) para download e consulta gratuita. E, se estiver interessado em nossos cursos, não deixe de [entrar em contato](#).

SUMÁRIO

1. [Tornando-se um desenvolvedor pragmático](#)
 1. [O que é realmente importante?](#)
 2. [A importância dos exercícios](#)
 3. [Tirando dúvidas e referências](#)
 4. [Para onde ir depois?](#)
2. [O modelo da bolsa de valores, datas e objetos imutáveis](#)
 1. [A bolsa de valores](#)
 2. [Candlesticks: O Japão e o arroz](#)
 3. [O projeto Tail](#)
 4. [O projeto Argentum: modelando o sistema](#)
 5. [Trabalhando com dinheiro](#)
 6. [Palavra chave final](#)
 7. [Imutabilidade de objetos](#)
 8. [Trabalhando com datas: Date e Calendar](#)
 9. [Exercícios: o modelo do Argentum](#)
 10. [Resumo diário das Negociações](#)
 11. [Exercícios: fábrica de Candlestick](#)

12. [Exercícios opcionais](#)

3. [Testes Automatizados](#)

1. [Nosso código está funcionando corretamente?](#)
2. [Exercícios: testando nosso modelo sem frameworks](#)
3. [Definindo melhor o sistema e descobrindo mais bugs](#)
4. [Testes de Unidade](#)
5. [JUnit](#)
6. [Anotações](#)
7. [JUnit4, convenções e anotação](#)
8. [Exercícios: migrando os testes do main para JUnit](#)
9. [Vale a pena testar classes de modelo?](#)
10. [Exercícios: novos testes](#)
11. [Para saber mais: Import Estático](#)
12. [Mais exercícios opcionais](#)
13. [Discussão em aula: testes são importantes?](#)

4. [Trabalhando com XML](#)

1. [Os dados da bolsa de valores](#)
2. [O formato XML](#)
3. [Lendo XML com Java de maneira difícil, o SAX](#)
4. [XStream](#)
5. [Exercícios: Lendo o XML](#)
6. [Discussão em aula: Onde usar XML e o abuso do mesmo](#)

5. [Test Driven Design - TDD](#)

1. [Separando as candles](#)
2. [Vantagens do TDD](#)
3. [Exercícios: Identificando negociações do mesmo dia](#)
4. [Exercícios: Separando os candles](#)
5. [Exercícios opcionais](#)

6. [Acessando um Web Service](#)

1. [Integração entre sistemas](#)
2. [Consumindo dados de um Web Service](#)
3. [Criando o cliente Java](#)
4. [Exercícios: Nosso cliente Web Service](#)
5. [Discussão em aula: Como testar o cliente do web service?](#)

7. [Introdução ao JSF e Primefaces](#)

1. [Desenvolvimento desktop ou web?](#)
2. [Características do JSF](#)
3. [Exercícios: Instalando o Tomcat e criando o projeto](#)
4. [A primeira página com JSF](#)
5. [Interagindo com o modelo: Managed Beans](#)
6. [Recebendo informações do usuário](#)

7. [Exercícios: Os primeiros componentes JSF](#)
 8. [A lista de negociações](#)
 9. [Formatação de Data com JSF](#)
 10. [Exercícios: p:dataTable para listar as Negociações do Web Service](#)
 11. [Para saber mais: paginação e ordenação](#)
 12. [Exercício opcional: adicione paginação e ordenação à tabela](#)
-
8. [Refatoração: os Indicadores da bolsa](#)
 1. [Análise Técnica da bolsa de valores](#)
 2. [Indicadores Técnicos](#)
 3. [As médias móveis](#)
 4. [Exercícios: criando indicadores](#)
 5. [Refatoração](#)
 6. [Exercícios: Primeiras refatorações](#)
 7. [Refatorações maiores](#)
 8. [Discussão em aula: quando refatorar?](#)
-
9. [Gráficos interativos com Primefaces](#)
 1. [Por que usar gráficos?](#)
 2. [Gráficos com o Primefaces](#)
 3. [Documentação](#)
 4. [Definição do modelo do gráfico](#)
 5. [Isolando a API do Primefaces: baixo acoplamento](#)
 6. [Para saber mais: Design Patterns Factory Method e Builder](#)
 7. [Exercícios: Gráficos com Primefaces](#)
-
10. [Aplicando Padrões de projeto](#)
 1. [Nossos indicadores e o design pattern Strategy](#)
 2. [Exercícios: refatorando para uma interface e usando bem os testes](#)
 3. [Exercícios opcionais](#)
 4. [Indicadores mais elaborados e o Design Pattern Decorator](#)
 5. [Exercícios: Indicadores mais espertos e o Design Pattern Decorator](#)
-
11. [A API de Reflection](#)
 1. [Escolhendo qual gráfico plotar](#)
 2. [Exercícios: permitindo que o usuário escolha o gráfico](#)
 3. [Montando os indicadores dinamicamente](#)
 4. [Introdução a Reflection](#)
 5. [Por que reflection?](#)
 6. [Constructor, Field e Method](#)
 7. [Melhorando nosso ArgentumBean](#)
 8. [Exercícios: indicadores em tempo de execução](#)
 9. [Melhorando a orientação a objetos](#)
-
12. [Apêndice Testes de interface com Selenium](#)
 1. [Alterando o título do gráfico](#)

2. [Validação com JSF](#)
3. [Introdução aos testes de aceitação](#)
4. [Como funciona?](#)
5. [Trabalhando com diversos testes de aceitação](#)
6. [Para saber mais: Configurando o Selenium em casa](#)
7. [Exercícios: Testes de aceitação com Selenium](#)

Você encontra a Caelum também em:

Blog Caelum

Cursos Online

Facebook

Newsletter

Casa do Código

Twitter