
Aprenda Programação Funcional

AMOSTRA

AMOSTRA

Aprenda Programação Funcional

*Como Pensar Funcionalmente
para Trabalhar com Códigos Complexos*

Jack Widman, Ph.D.



ALTA BOOKS

GRUPO EDITORIAL

Rio de Janeiro, 2024

Aprenda Programação Funcional

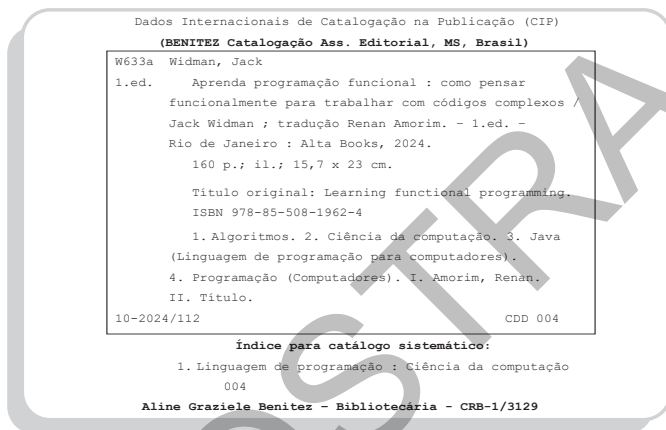
Copyright © 2024 ALTA BOOKS.

Copyright © 2022 Jack Widman.

ISBN: 978-85-508-1962-4

Authorized Portuguese translation of the English edition of Learning Functional Programming ISBN 9781098111755 © 2022 Jack Widman. This translation is published and sold by permission of O'Reilly Media, Inc., which owns or controls all rights to publish and sell the same. PORTUGUESE language edition published by Grupo Editorial Alta Books Ltda., Copyright © 2024 by STARLIN ALTA EDITORA E CONSULTORIA LTDA.

Impresso no Brasil – 1ª Edição, 2024 – Edição revisada conforme o Acordo Ortográfico da Língua Portuguesa de 2009.



Todos os direitos estão reservados e protegidos por Lei. Nenhuma parte deste livro, sem autorização prévia por escrito da editora, poderá ser reproduzida ou transmitida. A violação dos Direitos Autorais é crime estabelecido na Lei nº 9.610/98 e com punição de acordo com o artigo 184 do Código Penal.

O conteúdo desta obra fora formulado exclusivamente pelo(s) autor(es).

Marcas Registradas: Todos os termos mencionados e reconhecidos como Marca Registrada e/ou Comercial são de responsabilidade de seus proprietários. A editora informa não estar associada a nenhum produto e/ou fornecedor apresentado no livro.

Material de apoio e erratas: Se parte integrante da obra e/ou por real necessidade, no site da editora o leitor encontrará os materiais de apoio (download), errata e/ou quaisquer outros conteúdos aplicáveis à obra. Acesse o site www.altabooks.com.br e procure pelo título do livro desejado para ter acesso ao conteúdo.

Suporte Técnico: A obra é comercializada na forma em que está, sem direito a suporte técnico ou orientação pessoal/exclusiva ao leitor.

A editora não se responsabiliza pela manutenção, atualização e idioma dos sites, programas, materiais complementares ou similares referidos pelos autores nesta obra.

Produção Editorial: Grupo Editorial Alta Books
Diretor Editorial: Anderson Vieira
Vendas Governamentais: Cristiane Mutús
Gerência Comercial: Claudio Lima

Produtora Editorial: Isabella Gibara
Tradução: Renan Amorim
Copidesque: Aline Amaral
Revisão: André Cavanha; Alessandro Thomé
Diagramação: Joyce Matos
Revisão Técnica: Mario Adaniya
(Professor de Ciência da Computação no Centro Universitário Filadélfia - UniFil)



Rua Viúva Cláudio, 291 – Bairro Industrial do Jacaré
CEP: 20.970-031 – Rio de Janeiro (RJ)
Tels.: (21) 3278-8069 / 3278-8419
www.altabooks.com.br – altabooks@altabooks.com.br
Ouvidoria: ouvidoria@altabooks.com.br



*Para as minhas três filhas incríveis: Katherine, Annie e Victoria.
E para Andrea, cujo apoio e amor me incentivaram de tantas formas.*

AMOSTRA

AMOSTRA

Prefácio	xiii
Quem Deveria Usar Este Livro?	xiii
Como Este Livro Está Organizado	xiv
Convenções Utilizadas Neste Livro	xiv
1. O que É Programação Funcional?	1
Imutabilidade	3
Transparência Referencial	5
Funções de Alta Ordem	8
Avaliação Preguiçosa (Lazy Evaluation)	9
Pensando como um Programador Funcional	10
Os Benefícios da Programação Funcional	11
A PF Pode Aumentar a Produtividade	12
A PF É Divertida!	13
Scala	14
Conclusão	15

2. Preliminares Matemáticas	17
Teoria dos Conjuntos	17
Funções	19
Domínio e intervalo	19
Tipos de Funções	21
Função injetora	21
Função sobrejetora	22
Ciência da Computação Básica	22
Funções Anônimas	23
Funções como Objetos de Primeira Classe	24
Conclusão	24
3. Teoria das Categorias e Padrões.....	25
Padrões Baseados na Teoria das Categorias	27
Uma Breve História	28
Objetos e Morfismos	29
Um Exemplo de Categoria	30
A Categoria Scala	34
Morfismos	35
Funtores	36
O que um functor faz com um morfismo?	38
Formulação de um Functor na Linguagem de Programação	39
Os Padrões	42
O Padrão Functor	42
Monoides	43
Como os monoides podem ser úteis na programação funcional?	44

Transformações Naturais	46
Monads	48
flatMap e unit	48
Conclusão	51
4. Estrutura de Dados Funcional.....	53
A Estrutura de Dados Option	54
A Estrutura de Dados Try	59
A Estrutura de Dados Either	60
Funções de Alta Ordem	62
Monads em Compreensões de Sequência em Scala	63
Estruturas de Dados Tradicionais	65
Imutabilidade e Histórico	65
Preguiça (Laziness)	66
Resumo	66
5. Mais sobre Imutabilidade.....	67
Variáveis Mutáveis e Imutáveis	67
Recursão	68
Um Exemplo de Lista Encadeada	69
Recursão de Cauda	76
Mais Exemplos do Poder de fold em Scala	80
Uma Conexão entre fold e Monoides	81
Mais com Funções de Alta Ordem	84
De map para flatMap	86
Conclusão	88

6. Questões de Simultaneidade	91
Fluxos (<i>Streams</i>)	95
Akka Streams	96
Source	97
Flow	97
Sink	98
Mais sobre Fluxos	99
FS2: Functional Streams for Scala	100
Conclusão	102
7. Para Onde Ir a Partir Daqui	103
Adotando a Raiz Pura	103
A IO Monad	105
Mais sobre a IO Monad	106
E a Haskell?	107
Trilhando o Caminho Intermediário	107
Linguagens da JVM	108
Kotlin	108
Clojure	108
Linguagens .NET	109
Classes Type	109
Outro exemplo de uma classe type: uma biblioteca Json	113
Conclusão	115
Apêndice: Scala	117
Suposições	118
Visão Geral	118
var e val	120

Classes e Objetos	120
Funções	121
Funções que Retornam Funções	122
Classes de Caso	122
Declarando Funções	123
Currying	123
Funções Anônimas	124
Funções de Alta Ordem	124
Correspondência de Padrões	126
Traits	128
Distinguindo Classes Abstratas e Traits	129
Avaliação Preguiçosa	130
Parâmetros de Tipos	130
O Tipo Option	130
Future	131
Algumas Funções de Alta Ordem Fundamentais	133
map	133
flatten	133
flatMap	134
Outras Importantes Funções de Alta Ordem	134
foldLeft	134
filter	135
Conclusão	135
Índice	137
Sobre o Autor	143
Colofão	145

AMOSTRA

No decorrer dos últimos anos, a programação funcional (PF) passou por um período de renascença. Várias empresas estão procurando por programadores que tenham experiência em programação funcional. Muitas linguagens que não foram projetadas originalmente para serem funcionais vêm evoluindo com o passar do tempo para incluírem capacidades funcionais. Linguagens como Java, JavaScript e Python, para citar apenas algumas, vêm se tornando cada vez mais funcionais. O incentivo para os programadores adquirirem experiência funcional parece se dever, em parte, à aparente melhoria no processo de desenvolvimento, incluindo uma sensação de que, quando adotamos o método funcional, temos menos bugs e produzimos códigos mais expansivos e robustos. Se isso é verdade ou não e se a proporção da produção de códigos funcionais aumentará, só o tempo dirá. Por enquanto, consideraremos a programação funcional como um de vários paradigmas, cada um deles com suas próprias vantagens e desvantagens.

Quem Deveria Usar Este Livro?

Basicamente, todos os programadores. Se não tem experiência em PF, mas já ouvir falar nela e ficou curioso, ou até se comprou esta publicação sem a menor noção do que seja a PF, este livro será útil para você. Programadores de PF experientes também tirarão proveito dele. Este livro analisa as raízes da Teoria das Categorias da PF de uma forma jamais vista em outros livros sobre PF. Por fim, os programadores que têm certa experiência no uso da PF, mas que desejam ir além e se aprofundar nos conceitos e teorias que a compõem, encontrarão muitas coisas que poderão lhes ser úteis.

Como Este Livro Está Organizado

Queremos demonstrar, por meio de várias linguagens de programação, como os construtos funcionais podem melhorar nossos códigos. Mas, à medida que o livro avança, a linguagem de programação Scala é usada com mais frequência nos exemplos. O motivo disso é que o autor acredita que, devido à facilidade com que ideias funcionais conseguem ser expressas em Scala, o leitor terá muito mais facilidade de entender tais ideias se estas forem transmitidas de um modo mais natural, o que é possível fazer com a Scala. Para uma breve introdução à Scala, veja o apêndice.

Convenções Utilizadas Neste Livro

As seguintes convenções tipográficas são usadas neste livro:

Itálico

Indica novos termos, URLs, endereços de e-mail, nomes de arquivos e extensões de arquivos.

Fonte monoespaçada

Usada para a listagem de programas e para fazer referência a elementos dos programas em parágrafos, como nomes de variáveis ou funções, bancos de dados, tipos de dados, variáveis de ambiente, declarações e palavras-chave.



Este elemento representa uma dica ou sugestão.



Este elemento representa uma observação geral.



Este elemento representa um alerta ou atenção.

Temos um site para este livro, onde listamos erratas, exemplos e outras informações adicionais. Você pode acessar essa página utilizando o seguinte endereço: <https://oreil.ly/learning-fp> (conteúdo em inglês).

O que É Programação Funcional?

Programação funcional? Functores? Monoides, monads? “Eu não sou nenhum matemático!”, você talvez diga. “Como vou aprender esses conceitos esotéricos? E por que deveria querer fazer isso?” Essas preocupações são totalmente compreensíveis. Mas a verdade é que você não precisa ser um matemático para ser um programador funcional.

Os conceitos fundamentais da programação funcional são fáceis de entender quando apresentados de forma clara e direta. E é esse o objetivo deste livro: tornar a programação funcional compreensível e prática. Mais especificamente, vou ensiná-lo a *pensar* como um programador funcional. Mas por que você desejaria aprender programação funcional?

Imagine o seguinte: são 22h e você está totalmente travado, tentando resolver um bug de um programa que precisa entregar na manhã seguinte. O problema parece girar em torno de uma variável chamada *ratio*. O problema é que, dependendo das condições do sistema que está modelando, essa variável continua mudando. Sua frustração aumenta. Ou você tem um prazo no trabalho e seu microsserviço tem um bug elusivo que está lutando para encontrar. O problema parece estar em dois loops for aninhados nos quais as variáveis são modificadas de uma forma bastante intrincada. A lógica é complexa, e você não consegue encontrar a solução. Quem dera houvesse uma maneira de escrever programas de uma forma em que os valores das variáveis não mudassem! É aí que entra a programação funcional.



As variáveis cujos valores mudam com frequência são uma considerável fonte de bugs em programas. Pode ser difícil acompanhar os valores delas porque elas podem mudar a qualquer momento.