

Оглавление



Предисловие	15
Вступление	20
Благодарности	22
О книге	23
Об авторе	25
От издательства	26
Глава 1. Добро пожаловать в мир функционального мышления	27
Что такое функциональное программирование	28
Недостатки определения при практическом применении	30
Определение ФП сбивает с толку руководителей	31
Функциональное программирование рассматривается как совокупность навыков и концепций	32
Действия, вычисления и данные	33
Функциональные программисты особо выделяют код, для которого важен момент вызова	34
Функциональное программирование отличает инертные данные от работающего кода	35
Функциональные программисты разделяют действия, вычисления и данные	36
Три категории кода в ФП	37
Как нам помогают различия между действиями, вычислениями и данными	38
Чем эта книга отличается от других книг о ФП	39
Что такое функциональное мышление	40
Основные правила для идей и навыков, представленных в книге	41

6 Оглавление

Итоги главы	44
Резюме	44
Что дальше?	44
Глава 2. Функциональное мышление в действии	45
Добро пожаловать в пиццерию Тони!	46
Часть 1. Проведение различий между действиями, вычислениями и данными	47
Организация кода по частоте изменений	48
Часть 2. Использование первоклассных абстракций	49
Временные линии наглядно представляют работу распределенных систем	50
Действия на временных линиях могут выполняться в разном порядке	51
Особенности распределенных систем: урок, полученный дорогой ценой	52
Сегментация временной линии: заставляем роботов ожидать друг друга	54
Положительные уроки	55
Итоги главы	56
Резюме	56
Что дальше?	56
ЧАСТЬ I. ДЕЙСТВИЯ, ВЫЧИСЛЕНИЯ И ДАННЫЕ	
Глава 3. Действия, вычисления и данные	58
Действия, вычисления и данные	59
Действия, вычисления и данные применимы в любых ситуациях	60
Что мы узнали при моделировании процесса покупки	63
Применение функционального мышления в новом коде	68
Наглядное представление процесса рассылки купонов по электронной почте	71
Реализация процесса отправки купонов	75
Применение функционального мышления в существующем коде	84
Распространение действий в коде	86
Действия могут принимать разные формы	87
Итоги главы	91
Резюме	91
Что дальше?	91
Глава 4. Извлечение вычислений из действий	92
Добро пожаловать в MegaMart.com!	93
Вычисление бесплатной доставки	94
Вычисление налога	95
Необходимо упростить тестирование	96
Необходимо улучшить возможности повторного использования кода	97
Различия между действиями, вычислениями и данными	98
У функций есть ввод и вывод	99

Тестирование и повторное использование связаны с вводом и выводом	100
Извлечение вычислений из действий	102
Извлечение другого вычисления из действия	105
Весь код в одном месте	117
Итоги главы	118
Резюме	118
Что дальше?	118
Глава 5. Улучшение структуры действий	119
Согласование структуры с бизнес-требованиями	120
Приведение функции в соответствие с бизнес-требованиями	121
Принцип: минимизация неявного ввода и вывода	123
Сокращение неявного ввода и вывода	124
Проверим код еще раз	127
Классификация наших расчетов	129
Принцип: суть проектирования в разделении	130
Улучшение структуры за счет разделения <code>add_item()</code>	131
Выделение паттерна копирования при записи	132
Использование функции <code>add_item()</code>	133
Классификация вычислений	134
Уменьшение функций и новые вычисления	138
Итоги главы	139
Резюме	139
Что дальше?	139
Глава 6. Неизменяемость в изменяемых языках	140
Может ли неизменяемость применяться повсеместно	141
Классификация операций чтения и/или записи	142
Три этапа выполнения копирования при записи	143
Преобразование записи в чтение с использованием копирования при записи	144
Сравнение двух версий	148
Операции копирования при записи обобщаются	149
Знакомство с массивами в JavaScript	150
Что делать с операциями чтения/записи	154
Разделение функции, выполняющей чтение и запись	154
Возвращение двух значений одной функцией	155
Операции чтения неизменяемых структур данных являются вычислениями	163
Приложения обладают состоянием, которое изменяется во времени	164
Неизменяемые структуры данных достаточно быстры	165
Операции с копированием при записи для объектов	166
Кратко об объектах JavaScript	167
Преобразование вложенных операций записи в чтение	172

8 Оглавление

Что же копируется?	173
Наглядное представление поверхностного копирования и структурного совместного использования	174
Итоги главы	178
Резюме	178
Что дальше?	178
Глава 7. Сохранение неизменяемости при взаимодействии с ненадежным кодом	179
Неизменяемость при работе с унаследованным кодом	180
Наш код копирования при записи должен взаимодействовать с ненадежным кодом	181
Защитное копирование позволяет сохранить неизменяемый оригинал	182
Реализация защитного копирования	184
Правила защитного копирования	185
Упаковка ненадежного кода	187
Защитное копирование, которое вам может быть знакомо	190
Сравнение копирования при записи с защитным копированием	192
Глубокое копирование затратнее поверхностного	193
Трудности реализации глубокого копирования в JavaScript	194
Диалог между копированием при записи и защитным копированием	196
Итоги главы	199
Резюме	199
Что дальше?	199
Глава 8. Многоуровневое проектирование: часть 1	200
Что такое проектирование программной системы	201
Что такое многоуровневое проектирование	202
Развитие чувства проектирования	203
Паттерны многоуровневого проектирования	204
Паттерн 1. Прямолинейная реализация	205
Три уровня детализации	219
Выделение цикла for	223
Обзор паттерна 1. Прямолинейная реализация	232
Итоги главы	234
Резюме	234
Что дальше?	234
Глава 9. Многоуровневое проектирование: часть 2	235
Паттерны многоуровневого проектирования	236
Паттерн 2. Абстрактный барьер	237
Абстрактные барьеры скрывают реализацию	238

Игнорирование подробностей симметрично	239
Замена структуры данных корзины	240
Повторная реализация корзины в виде объекта	242
Абстрактный барьер позволяет игнорировать подробности	243
Когда следует (или не следует!) использовать абстрактные барьеры	244
Обзор паттерна 2. Абстрактный барьер	245
Код становится более прямолинейным	246
Паттерн 3. Минимальный интерфейс	247
Обзор паттерна 3. Минимальный интерфейс	253
Паттерн 4. Удобные уровни	254
Паттерны многоуровневой архитектуры	255
Что можно узнать из графа о коде?	256
Код в верхней части графа проще изменять	257
Важность тестирования кода нижних уровней	259
Код нижних уровней лучше подходит для повторного использования	262
Итоги: что можно узнать о коде по графу вызовов	263
Итоги главы	264
Резюме	264
Что дальше?	264

ЧАСТЬ II. ПЕРВОКЛАССНЫЕ АБСТРАКЦИИ

Глава 10. Первоклассные функции: часть 1	266
Отдел маркетинга все еще должен согласовываться с разработчиками	268
Признак «кода с душком»: неявный аргумент в имени функции	269
Рефакторинг: явное выражение неявного аргумента	271
Определение того, что является и что не является первоклассным значением	273
Не приведут ли строки с именами полей к новым ошибкам?	274
Усложнят ли первоклассные поля изменения API?	276
Мы будем использовать множество объектов и массивов	281
Первоклассные функции могут заменить любой синтаксис	284
Пример цикла for: еда и уборка	287
Рефакторинг: замена тела обратным вызовом	293
Что это за синтаксис	296
Почему мы упаковали код в функцию	297
Итоги главы	300
Резюме	300
Что дальше?	300
Глава 11. Первоклассные функции: часть 2	301
Одна проблема, два метода рефакторинга	302
Рефакторинг копирования при записи	303

Рефакторинг копирования при записи для массивов	304
Возвращение функций функциями	314
Итоги главы	324
Резюме	324
Что дальше?	324
Глава 12. Функциональные итерации	325
Один признак «кода с душком» и два рефакторинга	326
MegaMart создает группу взаимодействия с клиентами	327
map() в примерах	331
Инструмент функционального программирования: map()	332
Три способа передачи функций	334
Пример: адреса всех клиентов	336
filter() в примерах	339
Инструмент функционального программирования: filter()	340
Пример: клиенты без покупок	341
reduce() в примерах	344
Инструмент функционального программирования: reduce()	345
Пример: конкатенация строк	346
Что можно сделать с reduce()	350
Сравнение трех инструментов функционального программирования	352
Итоги главы	353
Резюме	353
Что дальше?	353
Глава 13. Сцепление функциональных инструментов	354
Группа взаимодействия с клиентами продолжает работу	355
Улучшение цепочек, способ 1: присваивание имен шагам	361
Улучшение цепочек, способ 2: присваивание имен обратным вызовам	362
Улучшение цепочек: сравнение двух способов	363
Пример: адреса клиентов с одной покупкой	364
Преобразование существующих циклов for в инструменты функционального программирования	369
Совет 1. Создавайте данные	370
Совет 2. Работайте с целыми массивами	371
Совет 3. Используйте много мелких шагов	372
Совет 3. Используйте много мелких шагов	373
Сравнение функционального кода с императивным	375
Советы по сцеплению	376
Советы по отладке	378
Другие функциональные инструменты	378
reduce() для построения значений	383

Творческий подход к представлению данных	385
О выравнивании точек	390
Итоги главы	391
Резюме	392
Что дальше?	392
Глава 14. Функциональные инструменты для работы с вложенными данными ..	393
Функции высшего порядка для значений в объектах	394
Явное выражение имени поля	395
Построение update()	396
Использование update() для изменения значений	397
Рефакторинг: замена схемы «чтение — изменение — запись» функцией update() ..	398
Функциональный инструмент: update()	399
Наглядное представление значений в объектах	400
Наглядное представление обновлений вложенных данных	406
Применение update() к вложенным данным	407
Построение updateOption()	408
Построение update2()	409
Наглядное представление update2() с вложенными объектами	410
Четыре варианта реализации incrementSizeByName()	413
Построение update3()	414
Построение nestedUpdate()	416
Анатомия безопасной рекурсии	421
Наглядное представление nestedUpdate()	422
Сила рекурсии	423
Конструктивные особенности при глубоком вложении	425
Абстрактные барьеры для глубоко вложенных данных	426
Сводка использования функций высшего порядка	427
Итоги главы	428
Резюме	428
Что дальше?	429
Глава 15. Изоляция временных линий	430
Осторожно, ошибка!	431
Пробуем кликать вдвое чаще	432
Временные диаграммы показывают, что происходит с течением времени	434
Два фундаментальных принципа временных диаграмм	435
Две неочевидные детали порядка действий	439
Построение временной линии добавления товара в корзину: шаг 1	440
Асинхронным вызовам необходимы новые временные линии	441
Разные языки, разные потоковые модели	442
Поэтапное построение временной линии	443

12 Оглавление

Изображение временной линии добавления товара в корзину: шаг 2	445
Временные диаграммы отражают две разновидности последовательного кода ...	447
Временные диаграммы отражают неопределенность в упорядочении параллельного кода	448
Принципы работы с временными линиями	449
Однопоточная модель в JavaScript	450
Асинхронная очередь JavaScript	452
AJAX и очередь событий	453
Полный пример с асинхронными вызовами	454
Упрощение временной линии	455
Чтение завершенной временной линии	461
Упрощение временной диаграммы добавления товара в корзину: шаг 3	463
Рисование временной линии (шаги 1–3)	465
Резюме: построение временных диаграмм	468
Сопоставление временных диаграмм помогает выявить проблемы	469
Два медленных клика приводят к правильному результату	470
Два быстрых клика приводят к неправильному результату	471
Временные линии с совместным использованием ресурсов создают проблемы ...	472
Преобразование глобальной переменной в локальную	473
Преобразование глобальной переменной в аргумент	474
Расширение возможностей повторного использования кода	477
Принцип: в асинхронном контексте в качестве явного вывода вместо возвращаемого значения используется обратный вызов	478
Итоги главы	483
Резюме	483
Что дальше?	483
Глава 16. Совместное использование ресурсов между временными линиями ...	484
Принципы работы с временными линиями	485
Корзина все еще содержит ошибку	486
Необходимо гарантировать порядок обновлений DOM	489
Реализация очереди в JavaScript	492
Принцип: берите за образец решения по совместному использованию из реального мира	501
Совместное использование очереди	502
Анализ временной линии	507
Принцип: чтобы узнать о возможных проблемах, проанализируйте временную диаграмму	510
Пропуск задач в очереди	511
Итоги главы	515
Резюме	515
Что дальше?	515

Глава 17. Координация временных линий	516
Принципы работы с временными линиями	517
Ошибка!	518
Как изменился код	520
Идентификация действий: шаг 1	521
Представление каждого действия: шаг 2	522
Упрощение диаграммы: шаг 3	526
Анализ возможных вариантов упорядочения	529
Почему эта временная линия выполняется быстрее	530
Ожидание двух параллельных обратных вызовов	532
Примитив синхронизации для нарезки временных линий	533
Использование Cut() в коде	535
Анализ неопределенных упорядочений	537
Анализ параллельного выполнения	538
Анализ для нескольких кликов	539
Примитив для однократного вызова	546
Неявная и явная модель времени	548
Резюме: манипулирование временными линиями	553
Итоги главы	553
Резюме	554
Что дальше?	554
Глава 18. Реактивные и многослойные архитектуры	555
Два архитектурных паттерна	556
Связывание причин и эффектов изменений	557
Что такое реактивная архитектура	558
Плюсы и минусы реактивной архитектуры	559
Ячейки как первоклассное состояние	560
Переменную ValueCell можно сделать реактивной	561
Обновление значков доставки при изменении ячейки	562
FormulaCell и вычисление производных значений	563
Изменяемое состояние в функциональном программировании	564
Как реактивная архитектура изменяет конфигурацию систем	565
Отделение эффектов от причин	566
Центр связей между причинами и эффектами	568
Интерпретация последовательности шагов как конвейера	569
Гибкость временной линии	570
Второй архитектурный паттерн	574
Что такое многослойная архитектура	575
Краткий обзор: действия, вычисления и данные	576
Краткий обзор: многоуровневое проектирование	577

Традиционная многоуровневая архитектура	578
Функциональная архитектура	579
Упрощение изменений и повторного использования	580
Понятия, используемые для размещения правила в слое	583
Анализ удобочитаемости и громоздкости решения	584
Итоги главы	588
Резюме	588
Что дальше?	588

Глава 19. Путешествие в мир функционального программирования

продолжается	589
План главы	590
Полученные профессиональные навыки	591
Главные выводы	592
Приобретение навыков: победы и разочарования	593
Параллельные пути к мастерству	594
Песочница: создание побочного проекта	596
Песочница: практические упражнения	597
Реальный код: устранение ошибок	598
Реальный код: постепенное улучшение проекта	599
Популярные функциональные языки	600
Функциональные языки с наибольшим количеством вакансий	601
Функциональные языки на разных платформах	602
Возможность получения знаний	602
Математическая основа	603
Литература	605
Итоги главы	606
Резюме	606
Что дальше?	606