Оглавление

Предисловие **15**

Благодарности **17**

О книге **19**

Кому следует прочитать эту книгу 19
Организация книги: дорожная карта 19
О программном коде 21
Дискуссионный форум liveBook 21
Другие онлайн-ресурсы 22
Об авторе 22
Об иллюстрации на обложке книги 22

Глава 1. Введение в Rust **23**

1.1. Где используется Rust? 24
1.2. С какой целью была написана книга «Rust в действии» 25
1.3. Вкус языка 26
1.3.1. Хитрый путь к «Hello, world!» 27
1.3.2. Ваша первая программа на Rust 29
1.4. Загрузка исходного кода книги 30
1.5. На что похож Rust? 31
1.6. В чем заключается особенность Rust? 34
1.6.1. Цель создания Rust: безопасность 36
1.6.2. Цель создания Rust: производительность 41
1.6.3. Цель создания Rust: управляемость 43
1.7. Особые возможности Rust 45
1.7.1. Достижение высокой производительности 45
1.7.2. Многопоточное выполнение программ 46
1.7.3. Достижение эффективной работы с памятью 46

1.8. Недостатки Rust ... 46
1.8.1. Циклические структуры данных ... 46
1.8.2. Время, затрачиваемое на компиляцию ... 46
1.8.3. Строгость ... 47
1.8.4. Объем языка ... 47
1.8.5. Излишний ажиотаж ... 47
1.9. Примеры использования TLS-безопасности ... 47
1.9.1. Heartbleed ... 48
1.9.2. Goto fail; ... 48
1.10. Для чего Rust подходит лучше всего? ... 50
1.10.1. В утилитах командной строки ... 50
1.10.2. В обработке данных ... 51
1.10.3. В расширяемых приложениях ... 51
1.10.4. В средах с ограниченными ресурсами ... 51
1.10.5. В серверных приложениях ... 52
1.10.6. В приложениях для ПК ... 52
1.10.7. В автономном режиме ... 52
1.10.8. В мобильных приложениях ... 53
1.10.9. В веб-режиме ... 53
1.10.10. В системном программировании ... 53
1.11. Скрытая фишка Rust: его сообщество ... 54
1.12. Разговорник по Rust ... 54
Резюме ... 54

Часть I. Особенности языка Rust ... 57

Глава 2. Основы языка ... 59
2.1. Создание работоспособной программы ... 60
2.1.1. Компиляция одиночных файлов с помощью утилиты rustc ... 60
2.1.2. Компиляция Rust-проектов с использованием cargo ... 61
2.2. Взгляд на синтаксис Rust ... 62
2.2.1. Определение переменных и вызов функций ... 63
2.3. Числа ... 64
2.3.1. Целые и десятичные (с плавающей точкой) числа ... 65
2.3.2. Записи целых чисел с основанием 2, 8 и 16 ... 66
2.3.3. Сравнение чисел ... 68
2.3.4. Рациональные, комплексные числа и другие числовые типы ... 73

2.4. Управление ходом выполнения программы ... 76
2.4.1. For: основной механизм итераций ... 76
2.4.2. Continue: пропуск оставшейся части текущей итерации ... 78
2.4.3. While: выполнение цикла, пока не изменится состояние условия ... 78
2.4.4. Loop: основа для циклических конструкций Rust ... 79
2.4.5. Break: прерывание цикла ... 80
2.4.6. If, if else и else: условное ветвление ... 81
2.4.7. Match: соответствие образцу с учетом типов ... 82
2.5. Определение функций ... 84
2.6. Использование указателей ... 85
2.7. Проект: визуализация множества Мандельброта ... 86
2.8. Расширенные определения функций ... 90
2.8.1. Явные аннотации времени жизни ... 90
2.8.2. Обобщенные функции ... 92
2.9. Создание grep-lite ... 95
2.10. Создание списков с использованием массивов, слайсов и векторов ... 99
2.10.1. Массивы ... 99
2.10.2. Слайсы ... 101
2.10.3. Векторы ... 102
2.11. Включение стороннего кода ... 104
2.11.2. Создание документации по сторонним контейнерам в локальной среде ... 107
2.11.3. Управление имеющимся в Rust набором инструментальных средств с помощью rustup ... 107
2.12. Поддержка аргументов командной строки ... 108
2.13. Чтение данных из файлов ... 110
2.14. Чтение из стандартного устройства ввода stdin ... 113
Резюме ... 114

Глава 3. Составные типы данных ... 117

3.1. Использование простых функций для экспериментов с API ... 117
3.2. Моделирование файлов с помощью struct ... 120
3.3. Добавление методов к структуре struct путем использования блока impl ... 125
3.3.1. Упрощение создания объектов за счет реализации метода new () ... 126
3.4. Возвращение сообщений об ошибках ... 129
3.4.1. Изменение значения известной глобальной переменной ... 129
3.4.2. Использование возвращаемого типа Result ... 135

3.5. Определение и использование перечисления enum ... 138
3.5.1. Использование enum для управления внутренним состоянием ... 141
3.6. Определение общего поведения с помощью типажей ... 143
3.6.1. Создание типажа Read ... 143
3.6.2. Реализация std::fmt::Display для ваших собственных типов ... 145
3.7. Выставление своих типов на всеобщее обозрение ... 148
3.7.1. Protecting private data Защита личных данных ... 148
3.8. Создание встроенной документации ваших проектов ... 149
3.8.1. Использование rustdoc для визуализации документов, касающихся одного исходного файла ... 151
3.8.2. Использование cargo для визуализации документов для контейнера и его зависимостей ... 151
Резюме ... 153

Глава 4. Время жизни, владение и заимствование ... 155
4.1. Реализация имитации наземной станции CubeSat ... 156
4.1.1. Выявление первой проблемы, связанной со временем жизни ... 158
4.1.2. Особое поведение элементарных типов ... 161
4.2. Справочник по рисункам, используемым в этой главе ... 163
4.3. Кто такой владелец? Есть ли у него какие-либо обязанности? ... 164
4.4. Как происходит переход владения ... 165
4.5. Решение проблем, связанных с владением ... 167
4.5.1. Если полное владение не требуется, используйте ссылки ... 170
4.5.2. Сократите количество долгоживущих значений ... 173
4.5.3. Продублируйте значение ... 180
4.5.4. Заключите данные в специальные типы ... 184
Резюме ... 186

Часть II. Демистификация системного программирования ... 189

Глава 5. Углубленное изучение данных ... 191
5.1. Комбинации битов и типы ... 191
5.2. Жизнь целых чисел ... 194
5.2.1. Усвоение порядка следования байтов ... 197
5.3. Представление десятичных чисел ... 198
5.4. Числа с плавающей точкой ... 199
5.4.1. Взгляд на f32 изнутри ... 200
5.4.2. Выделение знакового бита ... 201

5.4.3. Выделение экспоненты ... 202
5.4.4. Выделение мантиссы ... 203
5.4.5. Разбиение числа с плавающей точкой на составные части ... 205
5.5. Форматы чисел с фиксированной точкой ... 207
5.6. Генерация случайных вероятностей из случайных байтов ... 213
5.7. Реализация центрального процессора (CPU), чтобы удостовериться, что функции также являются данными ... 215
5.7.1. CPU RIA/1: сумматор ... 215
5.7.2. Полный листинг кода для CPU RIA/1: сумматор ... 220
5.7.3. CPU RIA/2: мультипликатор ... 222
5.7.4. CPU RIA/3: блок вызова ... 226
5.7.5. CPU 4: добавление всего остального ... 233
Резюме ... 233

Глава 6. Память ... 235

6.1. Указатели ... 235
6.2. Исследование типов ссылок и указателей, имеющихся в Rust ... 238
6.2.1. Обычные указатели, используемые в Rust ... 243
6.2.2. Экосистема указателей Rust ... 246
6.2.3. Строительные блоки интеллектуальных указателей ... 248
6.3. Предоставление программам памяти для размещения их данных ... 249
6.3.1. Стек ... 250
6.3.2. Куча ... 252
6.3.3. Что такое динамическое распределение памяти? ... 256
6.3.4. Анализ влияния, оказываемого динамическим выделением памяти ... 264
6.4. Виртуальная память ... 266
6.4.1. История вопроса ... 266
6.4.2. Шаг 1. Сканирование процессом собственной памяти ... 267
6.4.3. Преобразование виртуальных адресов в физические ... 271
6.4.4. Шаг 2. Работа с операционной системой для сканирования адресного пространства ... 274
6.4.5. Шаг 3. Чтение и запись в память процесса ... 277
Резюме ... 277

Глава 7. Файлы и хранилища ... 279

7.1. Что такое формат файла? ... 279
7.2. Создание собственных форматов файлов для хранения данных ... 281
7.2.1. Запись данных на диск с помощью serde и формата bincode ... 281

7.3. Реализация клона hexdump ... 284
7.4. Файловые операции, проводимые в Rust ... 288
7.4.1. Открытие файла в Rust и управление его режимом доступности ... 288
7.4.2. Безопасное взаимодействие с файловой системой с помощью std::fs::Path ... 289
7.5. Реализация хранилища «ключ-значение» с архитектурой, структурированной по записям и доступной только для добавления ... 291
7.5.1. Модель «ключ-значение» ... 291
7.5.2. Представление actionkv v1: хранилище ключей и значений в памяти с интерфейсом командной строки ... 292
7.6. Actionkv v1: интерфейсный код ... 293
7.6.1. Настройка продукта условной компиляции ... 296
7.7. Понимание сути actionkv: контейнер libactionkv ... 298
7.7.1. Инициализация структуры ActionKV ... 298
7.7.2. Работа с отдельно взятой записью ... 302
7.7.3. Запись многобайтных двоичных данных на диск в гарантированном порядке следования байтов ... 304
7.7.4. Проверка ошибок ввода-вывода с помощью контрольных сумм ... 306
7.7.5. Вставка в существующую базу данных новой пары «ключ-значение» ... 309
7.7.6. Полный код листинга для actionkv ... 310
7.7.7. Работа с ключами и значениями с использованием HashMap и BTreeMap ... 315
7.7.8. Создание HashMap и ее заполнение значениями ... 318
7.7.9. Извлечение значений из HashMap и BTreeMap ... 319
7.7.10. Что выбрать: HashMap или BTreeMap? ... 320
7.7.11. Добавление к actionkv v2.0 индекса базы данных ... 322
Резюме ... 326

Глава 8. Работа в сети ... 327

8.1. Все о сетевой работе в семи абзацах ... 328
8.2. Создание HTTP GET-запроса с использованием reqwest ... 330
8.3. Типажные объекты ... 332
8.3.1. На что способны типажные объекты? ... 332
8.3.2. Что такое типажные объекты? ... 333
8.3.3. Создание небольшой ролевой игры: rpg-проект ... 333
8.4. TCP ... 337
8.4.1. Что такое номер порта? ... 339
8.4.2. Преобразование имени хоста в IP-адрес ... 339

8.5. Способы обработки ошибок, наиболее удобные для помещения в библиотеки ... 347
8.5.1. Проблема: невозможность возвращения нескольких типов ошибок ... 347
8.5.2. Заключение в оболочку нижестоящих ошибок путем определения нашего собственного типа ошибки ... 351
8.5.3. Фокусы с unwrap() и expect() ... 358
8.6. MAC-адреса ... 358
8.6.1. Создание MAC-адресов ... 360
8.7. Реализация конечных автоматов с помощью перечислений ... 362
8.8. Чистый TCP ... 363
8.9. Создание виртуального сетевого устройства ... 363
8.10. «Чистый» HTTP ... 365
Резюме ... 376

Глава 9. Время и хронометраж ... 377
9.1. Предыстория вопроса ... 378
9.2. Источники времени ... 380
9.3. Определения ... 380
9.4. Кодирование времени ... 382
9.4.1. Представление часовых поясов ... 383
9.5. clock v0.1.0: учим приложение сообщать о времени ... 383
9.6. clock v0.1.1: форматирование меток времени в соответствии с ISO 8601 и стандартами электронной почты ... 384
9.6.1. Реструктуризация кода clock v0.1.0 с целью более широкой архитектурной поддержки ... 385
9.6.2. Форматирование времени ... 386
9.6.3. Предоставление полноценного интерфейса командной строки ... 387
9.6.4. clock v0.1.1: полный проект ... 388
9.7. clock v0.1.2: установка времени ... 392
9.7.1. Общее поведение ... 392
9.7.2. Установка времени для операционных систем, использующих libc ... 392
9.7.3. Установка времени в MS Windows ... 395
9.7.4. clock v0.1.2: листинг полного кода ... 397
9.8. Более совершенные способы обработки ошибок ... 401
9.9. clock v0.1.3: вычисление разницы показания часов с показанием протокола сетевого времени — Network Time Protocol (NTP) ... 402
9.9.1. Отправка NTP-запросов и интерпретация ответов ... 402
9.9.2. Корректировка местного времени по ответу сервера ... 405

9.9.3. Преобразования между представлениями о времени, использующими различные степени точности и эпохи ... 407
9.9.4. clock v0.1.3: листинг полной версии кода ... 409
Резюме ... 417

Глава 10. Процессы, потоки и контейнеры ... 419
10.1. Безымянные функции ... 420
10.2. Порождение потоков ... 421
10.2.1. Введение в замыкания ... 421
10.2.2. Порождение потока ... 422
10.2.3. Эффект от порождения нескольких потоков ... 423
10.2.4. Эффект от порождения множества потоков ... 424
10.2.5. Воспроизведение результатов ... 426
10.2.6. Совместно используемые переменные ... 430
10.3. Отличие замыканий от функций ... 433
10.4. Аватары, процедурно генерируемые из многопоточного парсера и генератора кода ... 434
10.4.1. Как запустить проект render-hex, и как выглядит его предполагаемый вывод ... 434
10.4.2. Обзор однопоточной версии render-hex ... 435
10.4.3. Порождение потока для каждой логической задачи ... 446
10.4.4. Использование пула потоков и очереди задач ... 449
10.5. Конкурентные вычисления и виртуализация задач ... 457
10.5.1. Потоки ... 460
10.5.2. Что такое контекстное переключение? ... 460
10.5.3. Процессы ... 460
10.5.4. WebAssembly ... 461
10.5.5. Контейнеры ... 461
10.5.6. А зачем вообще использовать операционную систему? ... 461
Резюме ... 462

Глава 11. Ядро операционной системы ... 463
11.1. Оперяющаяся операционная система (FledgeOS) ... 463
11.1.1. Настройка среды разработки под создание ядра операционной системы ... 463
11.1.2. Проверка среды разработки ... 465
11.2. FledgeOS-0: получение хоть чего-то работоспособного ... 466
11.2.1. Первая загрузка ... 466
11.2.2. Инструкции по компиляции ... 468

11.2.3. Листинги исходного кода ... 469
11.2.4. Способы справиться с паникой ... 474
11.2.5. Вывод информации на экран с использованием VGA-совместимого текстового режима ... 475
11.2.6 _start (): функция main () для FledgeOS ... 477
11.3. fledgeos-1: избавление от цикла занятости ... 477
11.3.1. Экономия ресурсов за счет прямого взаимодействия с центральным процессором ... 477
11.3.2. Исходный код fledgeos-1 ... 478
11.4. fledgeos-2: самостоятельная обработка исключений ... 479
11.4.1. Почти что правильная обработка исключений ... 479
11.4.2. Исходный код fledgeos-2 ... 480
11.5. fledgeos-3: текстовый вывод ... 481
11.5.1. Вывод на экран цветного текста ... 482
11.5.2. Управление представлением перечислений в памяти ... 482
11.5.3. Зачем использовать перечисления? ... 483
11.5.4. Создание шрифта для вывода информации в буфер кадра VGA ... 483
11.5.5. Вывод на экран ... 484
11.5.6. Исходный код fledgeos-3 ... 485
11.6. fledgeos-4: специализированная обработка паники ... 487
11.6.1. Реализация обработчика паники, сообщающего пользователю об ошибке. ... 487
11.6.2. Повторная реализация panic() с использованием core::fmt::Write ... 487
11.6.3. Реализация core::fmt::Write ... 488
11.6.4. Исходный код fledge-4 ... 489
Резюме ... 491

Глава 12. Сигналы, прерывания и исключения ... 493

12.1. Глоссарий ... 493
12.1.1. Сравнение сигналов и прерываний ... 495
12.2. Влияние прерываний на приложения ... 496
12.3. Программные прерывания ... 498
12.4. Аппаратные прерывания ... 499
12.5. Обработка сигналов ... 499
12.5.1. Поведение по умолчанию ... 499
12.5.2. Приостановка и возобновление работы программы ... 500
12.5.3. Перечень всех сигналов, поддерживаемых операционной системой ... 503

12.6. Обработка сигналов с помощью настраиваемых действий 504
12.6.1. Применение в Rust глобальных переменных 505
12.6.2. Использование глобальной переменной для указания на инициирование завершения выполнения программы 507
12.7. Отправка сигналов, определяемых в приложении 510
12.7.1. Общие сведения об указателях на функции и их синтаксисе 511
12.8. Игнорирование сигналов 512
12.9. Завершение работы из глубокой вложенности в стеке вызовов 514
12.9.1. Представление проекта sjlj 516
12.9.2. Настройка встроенных функций для их использования в программе 517
12.9.3. Приведение указателя к другому типу 519
12.9.4. Компиляция кода проекта sjlj 520
12.9.5. Исходный код проекта sjlj 521
12.10. Заметка о применении этих методов на платформах, не использующих сигналы. 524
12.11. Пересмотр исключений 524
Резюме 525

Предисловие

Кто знает, стоит ли вообще читать техническую литературу. Она может быть дорогой, скучной и сомнительной по содержанию. Хуже того, велика вероятность, что с ней вы так ничему и не научитесь. К счастью, эта книга написана автором, который все это прекрасно понимает.

Главная цель книги — обучить вас программированию на языке Rust. В книге представлены довольно крупные и способствующие обучению рабочие проекты. По ходу изучения материала будут созданы база данных, эмулятор процессора, ядро операционной системы и разработано несколько других интересных проектов. Предстоит даже заняться процедуральным искусством. Каждый проект разработан с целью изучения языка программирования Rust в удобном для вас темпе. Для тех, кто еще не освоился в Rust-программировании, есть множество возможностей по расширению проектов в любом выбранном направлении.

Но изучение языка программирования — это не только освоение его синтаксиса и семантики. Это также вступление в сообщество. К сожалению, уже устоявшиеся сообщества могут создавать для новичков незримые препятствия из-за совместно приобретенных знаний, применения специальных терминов и уже наработанной практики.

Для многих новичков Rust-программирования один из таких барьеров — концепция системного программирования. Редко у кого из программистов, переходящих на Rust, имеется опыт в данной области. В качестве компенсации перед книгой поставлена еще одна задача: научить вас системному программированию. А кроме других тем, в двенадцати главах книги вы почерпнете сведения о памяти, цифровом хронометраже и о работе драйверов устройств. Надеюсь, что после этого вы, став участником Rust-сообщества, сможете почувствовать себя в нем вполне уютно. И вы нам в нем нужны!

Наше общество зависит от применения программных средств, а критические дыры в безопасности считаются вполне нормальным и, возможно, даже неизбежным явлением. Rust показывает нам, что это уже не соответствует истине. А кроме того, наши компьютеры забиты под завязку раздутыми энергоемкими приложениями. Rust представляет собой жизнеспособную альтернативу, предназначенную для разработки программного обеспечения, менее требовательного к имеющимся ограниченным ресурсам.

Эта книга посвящена расширению возможностей. Ее конечная цель — убедить вас в этом. Rust не предназначен для какой-то избранной группы экспертов. Он является инструментом, доступным каждому. Читатели, проделавшие столь длинный путь в своем обучающем путешествии, удостоятся моей похвалы, и я с удовольствием сделаю для вас еще несколько шагов.

Благодарности

Спасибо Кэти за то, что удержала меня от свертывания проекта, и за то, что раз за разом поднимала мне настроение, когда я падал духом. Спасибо также Флоренс и Октавии за их обнимашки и улыбки, даже когда папа не мог с ними играть, потому что он писал эту книгу.

Я в долгу перед столькими людьми, что мне кажется несправедливым перечислить лишь немногих избранных. При работе над книгой я пользовался поддержкой многих участников Rust-сообщества. В ходе разработки материалов книги через liveBook поступили тысячи читательских исправлений, вопросов и предложений. И вклад каждого читателя помог мне усовершенствовать текст. Спасибо.

Особое чувство благодарности я испытываю к небольшому числу читателей, со многими из которых мы стали друзьями. Это Ай Майга (Aï Maiga), Ана Хобден (Ana Hobden), Эндрю Мередит (Andrew Meredith), Андрей Лесников (Andréy Lesnikóv), Энди Гроув (Andy Grove), Артуро Ж. Перес (Arturo J. Pérez), Брюс Митченер (Bruce Mitchener), Сесиль Тонглет (Cecile Tonglet), Дэниел Карозоне (Daniel Carosone), Эрик Ридж (Eric Ridge), Эстебан Кубер (Esteban Kuber), Флориан Гилчер (Florian Gilcher), Ян Бэттерсби (Ian Battersby), Джейн Ласби (Jane Lusby), Хавьер Виола (Javier Viola), Джонатан Тернер (Jonathan Turner), Лачезар Лечев (Lachezar Lechev), Лучано Маммино (Luciano Mammino), Люк Джонс (Luke Jones), Натали Блумфилд (Natalie Bloomfield), Олександр Каленюк (Oleksandr Kaleniuk), Оливия Ифрим (Olivia Ifrim), Пол Фариа (Paul Faria), Пол Дж. Саймондс (Paul J. Symonds), Филипп Гневош (Philipp Gniewosz), Род Элиас (Rod Elias), Стивен Оутс (Stephen Oates), Стив Клабник (Steve Klabnik), Таннер Аллард (Tanner Allard), Томас Локни (Thomas Lockney) и Уильям Браун (William Brown); для меня общение с вами на протяжении последних четырех лет было особой привилегией.

Я выражаю сердечную благодарность рецензентам книги, среди которых Афшин Мехрабани (Afshin Mehrabani), Аластер Смит (Alastair Smith), Брайс Дарлинг (Bryce Darling), Кристоффер Финк (Christoffer Fink), Кристофер Хаупт (Christopher Haupt), Дамиан Эстебан (Damian Esteban), Федерико Эрнандес (Federico Hernandez), Герт Ван Лаэтхем (Geert Van Laethem), Джефф Лим (Jeff Lim), Йохан Лизеборн (Johan Liseborn), Джош Коэн (Josh Cohen), Конарк Моди (Konark Modi), Марк Купер (Marc Cooper), Морган Нельсон (Morgan Nelson), Рамнивас Ладдад (Ramnivas Laddad), Риккардо Москетти (Riccardo Moschetti), Санкет Найк (Sanket Naik), Сумант Тамбе (Sumant Tambe), Тим ван Дерзен (Tim van Deurzen), Том Барбер (Tom Barber), Уэйд Джонсон (Wade Johnson), Уильям Браун (William Brown), Уильям Уиллер (William Wheeler) и Ив Дорфсман (Yves Dorfsman). Все ваши комментарии были прочитаны. А многие улучшения на последних этапах работы над книгой стали возможны благодаря вашим содержательным отзывам.

Особой похвалы за их терпение, профессионализм и позитивный настрой заслуживают два представителя команды Manning: Элеша Хайд (Elesha Hyde) и Фрэнсис Буран (Frances Buran), которые умело провели книгу через длинную череду черновых вариантов.

Также спасибо всем остальным редакторам разработки, в числе которых Берт Бейтс (Bert Bates), Джерри Куч (Jerry Kuch), Михаэла Батинич (Mihaela Batinić), Ребекка Райнхарт (Rebecca Rinehart), Рене ван ден Берг (René van den Berg) и Тим ван Дейрзен (Tim van Deurzen). Моя благодарность распространяется также на производственных редакторов, в числе которых Бенджамин Берг (Benjamin Berg), Дейрдре Хиам (Deirdre Hiam), Дженнифер Хоул (Jennifer Houle) и Пол Уэллс (Paul Wells).

В процессе выполнения принятой в Manning программы раннего доступа (MEAP-программы) книга претерпела 16 выпусков, что было бы невозможно без поддержки большой группы специалистов. Спасибо вам, Александар Драгосавлевич (Aleksandar Dragosavljević), Ана Ромак (Ana Romac), Элеонора Гарднер (Eleonor Gardner), Иван Мартинович (Ivan Martinović), Лори Вейдерт (Lori Weidert), Марко Райкович (Marko Rajkovic), Матко Хрватин (Matko Hrvatin), Мехмед Пашич (Mehmed Pasic), Мелисса Айс (Melissa Ice), Михаэла Батинич (Mihaela Batinic), Оуэн Робертс (Owen Roberts), Радмила Эрцеговац (Radmila Ercegovac) и Рейхана Марканович (Rejhana Markanovic).

Спасибо также маркетинговой команде в составе Бранко Латинчича (Branko Latincic), Кэндис Гиллхулли (Candace Gillhoolley), Коди Танкерсли (Cody Tankersley), Лукаса Вебера (Lucas Weber) и Степана Юрековича (Stjepan Jureković). Вы были для меня великим источником поддержки.

Отзывчивость и польза чувствовались и от расширенного состава команды Manning. Спасибо вам, Айра Дукжич (Aira Dučić), Эндрю Уолдрон (Andrew Waldron), Барбара Мирецки (Barbara Mirecki), Бранко Латинчич (Branko Latincic), Брекин Эли (Breckyn Ely), Кристофер Кауфманн (Christopher Kaufmann), Деннис Далинник (Dennis Dalinnik), Эрин Туи (Erin Twohey), Ян Хаф (Ian Hough), Иосип Марас (Josip Maras), Джулия Куинн (Julia Quinn), Лана Класик (Lana Klasic), Линда Котлярская (Linda Kotlyarsky), Лори Кервальд (Lori Kehrwald) и Мелоди Долаб (Melody Dolab) за помощь в работе над книгой. Спасибо и Майку Стивенсу (Mike Stephens) за то, что он положил начало этому изменяющему всю мою жизнь процессу. Он меня предупреждал, что будет сложно, и был абсолютно прав.

О книге

В первую очередь книга предназначена для людей, которые, возможно, изучали бесплатные материалы по Rust в Интернете, а затем спросили себя: «А что же дальше?». В книге содержатся десятки интересных примеров, поддающихся расширению при наличии соответствующего времени и творческого потенциала. Эти примеры позволяют двенадцати главам книги охватить продуктивное подмножество Rust и многие из наиболее важных сторонних библиотек экосистемы.

Основной упор в примерах кода делается на доступность для начинающих. В них нет особых излишеств, применяемых в элегантных, идиоматических приемах языка Rust. Читатели, неплохо разбирающиеся в программировании на Rust, могут не согласиться с некоторыми принятыми в примерах стилевыми решениями. Но я надеюсь, что они потерпят их ради тех, кто еще только учится.

Эта книга не задумывалась в качестве исчерпывающего справочника. Некоторые части языка и стандартной библиотеки были опущены. Как правило, здесь имеются в виду узкоспециализированные составляющие, изучению которых потребовалось бы уделить отдельное внимание. Вместо этого книга сосредоточена на предоставлении читателям достаточного объема базовых знаний и придания уверенности в том, что, в случае необходимости, им будет по силам изучить опущенные здесь специализированные области. Книга также уникальна в семействе книг, посвященных системному программированию, поскольку почти каждый приведенный в ней пример работает в Microsoft Windows.

Кому следует прочитать эту книгу

Эта книга должна понравиться тем, кто интересуется языком Rust, кто учится на практических примерах, или тем, кто напуган фактом принадлежности Rust к языкам системного программирования. Наибольшую пользу от книги получат читатели, уже имеющие опыт программирования, поскольку в ней будет изложен ряд концепций, используемых при программировании компьютерного оборудования.

Организация книги: дорожная карта

Книга состоит из двух частей. В первой части представлен синтаксис языка Rust и ряд его характерных особенностей. Во второй части те знания, что были получены при изучении первой, применяются при разработке ряда проектов. В каждой главе вводится одна-две новых концепций языка Rust.

Согласно вышесказанному, в первой части книги дается краткое введение в Rust:

- Глава 1, «Введение в Rust», объясняет причины появления Rust и рассказывает о том, как приступить к программированию на этом языке.
- Глава 2, «Особенности языка Rust», предоставляет фундаментальные сведения о синтаксисе языка. В качестве примеров используются визуализация множества Мандельброта и создание клона команды `grep`.
- Глава 3, «Составные типы данных», объясняет, как в Rust составляются типы данных и какими средствами обработки ошибок располагает этот язык.
- Глава 4, «Время жизни, владение и заимствование», рассматривает механизмы, гарантирующие неизменную корректность доступа к данным.

Во второй части книги Rust применяется для введения в область системного программирования:

- Глава 5, «Углубленное изучение данных», повествует о том, как информация представлена в цифровых компьютерах, при этом особое внимание уделяется приближению чисел. Примеры включают создание собственного числового формата и эмулятора центрального процессора.
- Глава 6, «Память», объясняет такие понятия, как ссылки, указатели, виртуальная память, стек и куча. Примеры включают создание сканера памяти и реализацию проекта процедурального искусства.
- Глава 7, «Файлы и хранилища», объясняет процессы сохранения структур данных на устройствах хранения информации. Примеры включают создание клона утилиты hex dump и разработку работоспособной базы данных.
- Глава 8, «Работа в сети», объясняет способы, применяемые компьютерами для обмена данными посредством многократной повторной реализации HTTP с избавлением при каждом шаге от очередного уровня абстракции.
- Глава 9, «Время и хронометраж», исследует процессы отслеживания времени в цифровом компьютере. Примеры включают создание работоспособного NTP-клиента.
- Глава 10, «Процессы, потоки и контейнеры», объясняет, что такое процессы, потоки и связанные с ними абстракции. Примеры включают создание приложения черепашьей графики и средства синтаксического анализа, работающего в режиме параллельных вычислений.
- Глава 11, «Ядро операционной системы», дает описание роли операционной системы и способов начальной загрузки компьютеров. Примеры включают компиляцию своего собственного загрузчика и ядра операционной системы.
- Глава 12, «Сигналы, прерывания и исключения», объясняет порядок связи внешнего мира с центральным процессором и операционными системами.

Книга предназначена для последовательного чтения. Предполагается, что во всех последующих главах используются знания, полученные в предыдущих. Но проекты из каждой главы не связаны друг с другом. Поэтому по всем проектам, касающимся интересующих вас тем, можно проходить в произвольном порядке.

О программном коде

Примеры кода, приведенные в этой книге, написаны на языке Rust в редакции 2018 года и протестированы под управлением Windows и Ubuntu Linux. Никакого специального программного обеспечения, кроме рабочей установки Rust, не требуется. Инструкции по установке изложены в главе 2.

Эта книга содержит множество примеров исходного кода как в пронумерованных листингах, так и в виде обычного текста. В обоих случаях, чтобы исходный код отличался от обычного текста, он отформатирован `таким вот шрифтом фиксированной ширины`. Иногда код также выделяется жирным шрифтом, чтобы отметить его фрагменты, изменившиеся по сравнению с предыдущими действиями, рассмотренными в главе, например когда к существующей строке кода добавляется новая функция.

Во многих случаях первоначальный исходный код был переформатирован: чтобы он уместился на книжной странице, в нем были переработаны отступы и добавлены разрывы строк. Но иногда и этого было недостаточно, поэтому в листингах попадаются маркеры продолжения строки (➥). Кроме того, при описании кода в тексте комментарии, имевшиеся в листингах исходного кода, зачастую удаляются. Для выделения важных понятий многие листинги сопровождаются примечаниями к коду.

Дискуссионный форум liveBook

Приобретение этой книги открывает свободный доступ к закрытому веб-форуму, запущенному издательством Manning Publications, где можно оставлять отзывы о книге, задавать вопросы технического плана и получать помощь от автора и от других пользователей:

- Чтобы попасть на форум, перейдите по адресу https://livebook.manning.com/book/rust-in-action/welcome/v-16/.
- Дополнительные сведения о форумах, запущенных издательством Manning, и о правилах поведения на них можно получить по адресу https://livebook.manning.com/#!/discussion.

Придерживаясь обязательств, принятых перед читателями, издательство Manning обеспечивает место, где может состояться содержательный диалог между отдельными читателями, а также между читателями и автором. При этом автор не обязан общаться с каким-то определенным количеством участников форума, поскольку его собственное участие остается добровольным (и неоплачиваемым). Чтобы не потерять авторский интерес к форуму, предлагаем попробовать задать автору несколько сложных вопросов! Пока книга находится в печати, форум и архивы предыдущих обсуждений будут доступны с веб-сайта издателя.

Другие онлайн-ресурсы

Тима Макнамару можно найти в социальной сети по метке @timClicks. Его основные каналы — Twitter (https://twitter.com/timclicks), YouTube (https://youtube.com/c/timclicks) и Twitch (https://twitch.tv/timclicks). Также можно свободно подключиться к его Discord-серверу по адресу https://discord.gg/vZBX2bDa7W.

Об авторе

Тим Макнамара (Tim McNamara) учился программировать, чтобы помогать осуществлению проектов гуманитарной помощи по всему миру прямо из своего дома в Новой Зеландии. За последние 15 лет Тим стал экспертом в области интеллектуального анализа текста, обработки естественного языка и обработки данных. Он организатор сообщества Rust Wellington и регулярно проводит учебные занятия по программированию на языке Rust не только в формате живого общения, но и по Интернету через Twitch и YouTube.

Об иллюстрации на обложке книги

Рисунок на обложке этой книги имел подпись «Le maitre de chausson» («Мастер тапочек») или «The boxer» («Боксер»). Иллюстрация взята из собрания работ множества художников под редакцией Луи Кармера (Louis Curmer), опубликованного в Париже в 1841 году. Коллекция называется «LesFrançais peints par eux-mêmes», что переводится как «Французы на собственных рисунках». Каждая иллюстрация четко прорисована и раскрашена вручную, а богатое разнообразие рисунков, представленных в коллекции, живо напоминает нам о том, насколько обособленными в культурном отношении были регионы мира, города́, деревни и кварталы всего 200 лет назад. Будучи разобщенными, люди говорили на разных диалектах и языках. На улицах или в сельской местности просто по тому, как люди одеты, было легко определить, где они живут и каково их ремесло или социальное положение.

С тех пор стиль одежды изменился, и разнообразие по регионам, столь богатое в то время, исчезло. Сейчас трудно отличить друг от друга жителей разных континентов, не говоря уже о разных городах или регионах. Возможно, мы обменяли культурное разнообразие на более яркую личную жизнь, проходящую, конечно же, в более разнообразном и быстро развивающемся техническом окружении.

В наше время, когда одну компьютерную книгу трудно отличить от другой, издательство Manning подчеркивает изобретательность и инициативу компьютерного бизнеса за счет обложек своих книг, показывающих богатое разнообразие региональной жизни двухсотлетней давности, оживленное изображениями из таких, как упомянутая здесь, коллекций.

1 Введение в Rust

В этой главе рассматриваются следующие вопросы:

- Введение в свойства Rust и о том, для чего он создавался.
- Разбор синтаксиса Rust.
- Где следует, а где не следует применять Rust.
- Создание вашей первой программы на Rust.
- Сравнение Rust с объектно-ориентированными языками и системами программирования более широкого спектра.

Знакомьтесь с Rust — это язык программирования, расширяющий ваши возможности. Углубленное знакомство с ним откроет вам не только язык программирования, обладающий невероятной скоростью и безопасностью, но и весьма приятный инструмент для программирования «на каждый день».

Приступив к программированию на Rust, вы вряд ли остановитесь. И эта книга, «Rust в действии», поможет вашему становлению в качестве Rust-программиста. Но она не станет учить программированию с нуля. Книга рассчитана на тех, кто рассматривает Rust в качестве своего следующего языка, и на тех, кому нравится освоение практических примеров. Вот наиболее яркие примеры, включенные в эту книгу:

- Визуализация множества Мандельброта.
- Grep-клон.
- Эмулятор центрального процессора.
- Искусство генерации.
- База данных.
- Клиенты HTTP, NTP и hexdumps.
- Интерпретатор языка LOGO.
- Ядро операционной системы.

Из списка ясно, что книга позволит не только освоить Rust, но и познакомит с *системным и низкоуровневым программированием*. Изучая книгу «Rust в действии», вы сможете узнать о роли операционной системы, о том, как работает процессор, как компьютеры отслеживают время, что такое указатели и что такое тип данных. Вы получите представление о взаимодействии внутренних систем компьютера. Выйдя за рамки изучения синтаксиса Rust, вы также поймете, для чего был создан этот язык и для решения каких задач он предназначен.

1.1. Где используется Rust?

Ежегодно, с 2016 по 2020 год, в опросе разработчиков в Stack Overflow Rust удостаивался звания «Самый любимый язык программирования». Видимо, именно поэтому Rust был принят ведущими разработчиками компьютерных технологий:

- Amazon Web Services (AWS) пользуется Rust с 2017 года для своих решений по бессерверным вычислениям AWS Lambda и AWS Fargate. Благодаря этому Rust развил свои успехи. Для выпуска своего сервиса Elastic Compute Cloud (EC2) компания Amazon создала операционную систему Bottlerocket OS и AWS Nitro System [1].
- Компания Cloudflare применяет Rust при разработке множества своих сервисов, включая общедоступную DNS, средства бессерверных вычислений и программы по проверке пакетов[2].
- Компания Dropbox применила Rust для перестройки своего внутреннего хранилища данных, управляющего эксабайтами данных[3].
- Компания Google применяет Rust при разработке таких компонентов Android, как модуль Bluetooth. Rust также используется для компонента Chrome OS под названием crosvm и играет важную роль в разработке новой операционной системы Google под названием Fuchsia[4].
- Компания Facebook использует Rust для повышения эффективности работы своих веб-, мобильных и API-сервисов, а также для разработки компонентов HHVM, виртуальной машины HipHop, используемой языком программирования Hack[5].
- Компания Microsoft пишет на Rust компоненты своей платформы Azure, включая демон безопасности для своей службы Интернета вещей (IoT)[6].
- В Mozilla язык Rust используется для совершенствования веб-браузера Firefox, база которого содержит 15 миллионов строк кода. Первые два проекта Mozilla Rust-в-Firefox — анализатор метаданных MP4 и система кодирования-декодирования текста, позволили повысить общую производительность и стабильность работы браузера.
- Входящая в GitHub компания npm Inc. использует Rust, чтобы справиться с «более чем 1,3 миллиардов загрузок пакетов в день»[7].

---

[1] См. «How our AWS Rust team will contribute to Rust's future successes», http://mng.bz/BR4J.

[2] См. «Rust at Cloudflare», https://news.ycombinator.com/item?id=17077358.

[3] См. «The Epic Story of Dropbox's Exodus From the Amazon Cloud Empire», http://mng.bz/d45Q.

[4] См. «Google joins the Rust Foundation», http://mng.bz/ryOX.

[5] См. «HHVM 4.20.0 and 4.20.1», https://hhvm.com/blog/2019/08/27/hhvm-4.20.0.html.

[6] См. https://github.com/Azure/iotedge/tree/master/edgelet.

[7] См. «Rust Case Study: Community makes Rust an easy choice for npm», http://mng.bz/xm9B.