

VIBE CODING

Программирование с помощью
искусственного интеллекта

Эдди Османи

УДК 004.891
ББК 32.973.26-018.2:32.813
О-74

Beyond Vibe Coding: Leveraging Your Experience in the Age of AI-Assisted Coding

Addy Osmani

© 2026 “Astana International Publishing” Authorized Russian translation of the English edition of
Beyond Vibe Coding ISBN 9798341634756

© 2025 Addy Osmani. 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.

Османи, Эдди.

О-74 Vibe Coding: программирование с помощью искусственного интеллекта / Эдди Османи: [перевод с английского О. Перфильева]. — Алматы : Астана иностранная пресса, 2026. — 320 с. — (O’Reilly. Книги по программированию).

ISBN 978-601-12-9040-1

Книга «Vibe Coding: Программирование с помощью искусственного интеллекта» посвящена современным подходам к разработке программного обеспечения с использованием ИИ-инструментов. Автор показывает, как меняется роль разработчика в условиях автоматизации написания кода и какие навыки становятся ключевыми в новой парадигме.

В издании рассматриваются методы постановки задач для ИИ, оценки и доработки сгенерированного кода, а также интеграции отдельных решений в целостные программные системы. Особое внимание уделено практическим аспектам взаимодействия с ИИ и построению эффективных рабочих процессов.

УДК 004.891

ББК 32.973.26-018.2:32.813

© Перфильев О.И., перевод на русский язык, 2026

© Издание на русском языке, оформление.

ISBN 978-601-12-9040-1

ТОО «Издательство «Астана иностранная пресса», 2026

Барлық құқықтар қорғалған. Бұл кітапты басып шығарушының рұқсатынсыз онлайн немесе кез келген басқа жолмен сканерлеу, жүктеп алу немесе заңсыз тарату заң бойынша жазаланады / Все права защищены. Никакая часть данной книги не может быть воспроизведена в какой бы то ни было форме с помощью каких-либо электронных или механических средств, включая изготовление фотокопий, аудиозапись, репродукцию или любой иной способ, или систем поиска и хранения информации без письменного разрешения издателя.

Оглавление

Предисловие	9
Для кого предназначена эта книга	9
Ожидания	11
Используемые в этой книге условные обозначения	13
O'Reilly Online Learning	14

Часть I Основы

Глава 1. Введение: что такое вайб-кодинг?	17
Спектр ИИ-кодирования: от вайб-кодинга до разработки с помощью ИИ	18
Чтение между строк: программирование с намерением	30
Производительность, доступность и меняющаяся природа программирования	35
Краткий обзор инструментов: становление экосистемы	38
Модели ИИ: современный ландшафт генерации кода	46
Основные модели	48
Выбор модели в зависимости от задачи	51
Преимущества и ограничения вайб-кодинга: нюансы	52
Итоги и дальнейшие шаги	63
Глава 2. Искусство составления запросов: эффективный способ общения с ИИ	65
Основы промпт-инжиниринга	66
Конкретность и ясность: как формулировать запросы, которые работают	68
Итеративное уточнение: цикл обратной связи с ИИ	70
Сравнение двух запросов	74
Техники запросов: набор инструментов эффективного взаимодействия	77
Продвинутый промпт-инжиниринг: сочетание техник и сложные случаи	90
Итоги и дальнейшие шаги	95

Часть II

Практика работы с ИИ

Глава 3. Проблема 70%: как работать с ИИ по-настоящему	99
Как разработчики используют ИИ на самом деле	101
Золотые правила вайб-кодинга	110
Итоги и дальнейшие шаги	113
Глава 4. За пределами 70%: максимальное раскрытие человеческого потенциала	114
Senior-программисты и разработчики: расширяйте свой опыт работы с ИИ	115
Middle-разработчики: адаптируйтесь и специализируйтесь	118
Младшие разработчики: двигайтесь к успеху вместе с ИИ	126
Развивайте навыки работы с запросами и инструментами (осознанно)	129
Итоги и дальнейшие шаги	132
Глава 5. Анализ сгенерированного кода: код-ревью, доработка, ответственность	135
От замысла к реализации: как понять написанный ИИ-код	135
«Решение большинства»: «самое распространённое» не значит «самое подходящее»	137
Читаемость кода и его структура: шаблоны и потенциальные проблемы	138
Стратегии отладки: поиск и исправление ошибок	140
Рефакторинг как способ поддержки кода: превращаем код ИИ в свой	142
Важный этап тестирования: модульные, интеграционные и сквозные тесты	143
Итоги и дальнейшие шаги	145
Глава 6. Прототипирование с использованием ИИ: инструменты и техники	147
Быстрое прототипирование с ИИ-ассистентами	147
ИИ-инструменты прототипирования	149
От концепции к прототипу: итеративная доработка	151
Эволюция прототипа на пути к производству	155
Проблемы, характерные для прототипирования с помощью ИИ	159
Итоги и дальнейшие шаги	161

Глава 7. Создание веб-приложений с помощью ИИ	162
Настройка проекта: создание каркаса с помощью ИИ	162
Проектирование и интеграция баз данных	173
Полностексовая интеграция: объединение фронтенда и бэкенда	176
Тестирование и валидация веб-приложений, созданных с помощью ИИ	181
Удачные примеры создания веб-проектов с помощью ИИ	184
Итоги и дальнейшие шаги	186

Часть III

Доверие и автономия

Глава 8. Безопасность, поддерживаемость и надёжность	191
Распространённые проблемы безопасности в сгенерированном ИИ коде	192
Аудит безопасности	199
Ручной код-ревью с акцентом на безопасность	201
Создание эффективных тестовых фреймворков для сгенерированных ИИ систем	205
Оптимизация производительности	210
Поддерживаемость кодовых баз, составляемых с помощью ИИ	214
Стратегии код-ревью	220
Лучшие практические рекомендации по надёжному развёртыванию	223
Итоги и дальнейшие шаги	228
Глава 9. Этические вопросы вайб-кодинга	229
Размышления по поводу интеллектуальной собственности	230
Прозрачность и признание авторства	237
Предвзятость (дискриминация) и справедливость	239
Золотые правила ответственного использования ИИ	243
Итоги и дальнейшие шаги	249
Глава 10. Автономные агенты фоновое программирования	250
От напарников к автономным агентам: что такое агенты фоновое программирования	250
Как работают автономные агенты	252
Сравнение фоновых агентов и ИИ-ассистентов в IDE	256
Объединение нескольких ИИ-моделей для максимального эффекта	259
Основные автономные агенты для написания кода	263
Проблемы и ограничения	266

Лучшие методы эффективного использования ИИ-агентов для программирования	268
Итоги и дальнейшие шаги	271
Глава 11. Не просто генерация кода:	
что стоит ожидать от ИИ в будущем	272
ИИ в тестировании, отладке и сопровождении	272
Дизайн и персонализация пользовательского опыта	275
Эволюция проджект-менеджмента с помощью ИИ	278
Как автономные агенты изменят сферу разработки ПО	282
Будущее языков программирования и возможность разработки на основе естественного языка	288
Как вайб-кодинг меняет индустрию	292
Итоги и дальнейшие шаги	294
Об авторе	298
Оформление книги	299
Алфавитный указатель	300

Предисловие

Мы переживаем эпоху глубоких перемен в сфере программного обеспечения. Так называемый профессиональный вайб-кодинг, то есть процесс совместной с ИИ разработки программного обеспечения (ПО), превращает разработчиков из технических «ремесленников» в своего рода «визионеров» и «архитекторов» продукции.

Но что же такое, собственно говоря, вайб-кодинг? Под этим модным термином подразумевается использование возможностей ИИ (искусственного интеллекта), выполняющего основную рутинную работу по написанию кода, благодаря чему разработчики получают возможность сосредоточиться больше на идеях, дизайне и решении высокоуровневых задач. В этом смысле вайб-кодинг можно назвать импровизацией или созданием кода «по наитию», с опорой скорее на некие «идеи», а не на технические детали. Как метко заметил Андрей Карпатый (<https://oreil.ly/7bnWe>), это сродни ощущению, когда «забываешь, что код вообще существует», и просто создаёшь, то есть описываешь, что тебе нужно, а о подробностях реализации заботится ИИ. Такой процесс может привести к многократному росту производительности и обещает не только воплотить в реальность мечту о пресловутом «10х-инженере» (то есть разработчике ПО с десятикратной продуктивностью), но и превратить её в концепцию «100х-продуктивности» (<https://oreil.ly/8UGID>).

Для кого предназначена эта книга

Эта книга ориентирована на три основные аудитории. Первая — это опытные разработчики и технические руководители, желающие многократно увеличить свой вклад в разработку ПО. Если вы занимаетесь этим уже на протяжении многих лет и понимаете всю важность автоматизации повторяющихся задач, эта книга покажет вам, как можно поручить рутину ИИ, а самим перейти к роли архитектора и стратега. Вы узнаете, как работать быстрее, не жертвуя своими сформированными за годы карьеры стандартами качества.

Вторая аудитория — разработчики ПО, мыслящие в категориях продукта, для которых код — это скорее средство, а не конечная цель. Если вас раздражает разрыв между концепцией и её реализацией, то вайб-кодинг поможет вам

в значительной степени сократить эту дистанцию. Вы узнаете, как можно быстро создавать прототипы и выпускать продукцию, на создание которых при традиционных методах ушли бы месяцы.

И здесь стоит упомянуть о том, что показалось мне в ИИ-инструментах наиболее парадоксальным: опытным разработчикам они помогают больше, чем новичкам. Такой вывод кажется нелогичным: разве ИИ не должен «демократизировать» создание кода?

В реальности же ИИ походит на некоего очень старательного джуна в вашей команде. Он умеет быстро писать код, но требует постоянного контроля и исправлений. Чем больше вы знаете, тем лучше можете им руководить.

Отсюда возникает то, что я называю *парадоксом знания*: опытные инженеры и разработчики пользуются ИИ для ускорения того, что они уже знают, тогда как начинающие пытаются пользоваться им, чтобы *научиться* что-то делать, и результаты радикально отличаются.

Я лично видел, как сеньоры пользуются ИИ для следующих задач:

- быстрое создание прототипов для уже понятных идей;
- быстрая реализация замысла, который можно затем развивать;
- анализ альтернативных методов решения известных проблем;
- автоматизация рутинных задач.

В то же время новички, решившие доверить свою работу ИИ, часто:

- находят некорректные или устаревшие решения задач;
- упускают важные аспекты безопасности и производительности;
- с трудом выполняют отладку сгенерированного ИИ кода;
- создают уязвимые системы, принцип работы которых сами до конца не понимают.

Третья аудитория этой книги — менеджеры и технические руководители, внедряющие ИИ-инструменты в своих коллективах и производственных процессах. Вы узнаете, как заниматься организацией, как оценивать квалификацию сотрудников и поддерживать качество продукции в эпоху, когда один разработчик может делать то, что раньше поручали целой команде. Описанные здесь стратегии помогут вам перейти на новый уровень, сохранив прежнюю культуру разработки.

Чего здесь вы *не найдёте*, так это вводного курса по программированию. Несмотря на то что ИИ повышает доступность программирования, работа с ним всё равно требует опыта и умения принимать обоснованные решения. Считайте, что перед вами продвинутое руководство для тех, кто готов выйти за рамки традиционного программирования и перейти к новой парадигме создания ПО.

Ожидания

В этой книге говорится о том, как меняется роль разработчиков, которые переходят от непосредственного создания кода к «инженерии продукта». При этом они в своём общении с ИИ руководствуются субъективными «человеческими» соображениями, ориентируясь на качество, архитектуру и потребности пользователей. Мы по-прежнему ценим творческий подход, системное мышление и эмпатию, благодаря которым абстрактная «рабочая программа» превращается в отличный продукт. ИИ не заменяет нас; при условии разумного его использования он усиливает наши положительные качества.

В первой части я рассуждаю о том, в каких областях особенно заметна эффективность вайб-кодинга: это создание новых продуктов, прототипирование функций, генерация стандартных CRUD-приложений или интеграция кода. Что касается этих задач, то для них скорость и стандартизация важнее глубокой оригинальности. В то же время я упоминаю и сферы, в которых следует соблюдать осторожность: реализация по-настоящему сложных, низкоуровневых или новых алгоритмов, в которых искусственный интеллект может ошибаться. Признав текущие ограничения ИИ, мы избежим разочарований и неудач, ведь многое в нашей работе по-прежнему зависит от человеческой изобретательности.

Ключевым звеном всего процесса остаётся человеческий фактор. Мы следим за правильностью архитектуры, находим сложные баги и оцениваем качество кода не только по принципу «работает / не работает». Ещё важнее наша ориентация на пользователя, что до сих пор недоступно для ИИ. Именно мы делаем так, чтобы ПО не только работало, но и было полезным и удобным для пользователей. Короче говоря, разработчики становятся кураторами и редакторами выдаваемого ИИ-кода, согласующими этот код с реальными потребностями и поддерживающими высокие стандарты.

Во второй части рассматриваются практические аспекты вайб-кодинга с отдельным вниманием к внедрению новых рабочих процессов. Методы вроде «не исправляй, а регенерируй» напоминают нам, что не стоит застревать на ошибках и что генерировать код заново порой бывает быстрее, чем заниматься его отладкой. Параллельные запросы помогают решать задачи с нескольких сторон одновременно. Чтобы не увязнуть в болоте хаоса, нужно найти баланс между быстрыми итерациями и последующей доработкой. Такие практические особенности, как модульная структура кода ИИ, тщательное тестирование и итеративное улучшение, помогают поддерживать чистоту и устойчивость кодовой базы несмотря на высокую скорость разработки.

По мере масштабирования проектов приходится управлять ускоренным потоком кода и обслуживать потенциальный технический долг. ИИ легко может

выдать целую кучу кода, которая быстро забьёт под завязку любой репозиторий; поддерживать код в работоспособном и адекватном состоянии помогают только дисциплина и правильные практические методы (при предположительном рефакторинге с помощью того же ИИ). Что касается кадровых вопросов, то мы поговорим о том, как нанимать и обучать разработчиков, которые умеют работать с ИИ-инструментами, умеют приспосабливаться к текущим условиям и применять на практике свои навыки системного проектирования. Также мы научимся определять моменты, когда стоит возвращаться к традиционным подходам — например, при долгосрочной поддержке продукта или во время работы с критически важными системами, для которых надёжность важнее скорости.

Третья часть посвящена безопасности и надёжности, этическим вопросам, а также инструментам, благодаря которым вайб-кодингом можно заниматься уже сегодня; среди них — IDE с поддержкой ИИ, такие как Cursor и Windsurf, интегрированные с моделями Anthropic, Google Gemini и OpenAI, знающие всю кодовую базу пользователей и помогающие на каждом этапе. Умение обращаться с этими инструментами и моделями (вариациями Claude для разных задач, ChatGPT для общих вопросов и ответов) входит в набор необходимых навыков современного разработчика. У каждого инструмента имеются свои сильные стороны: Cursor лучше подходит для интерактивного редактирования, Windsurf — для задач с большим контекстом, чат-интерфейсы — для мозгового штурма и поиска ошибок и т. д.

В будущем, насколько мне кажется, появятся ещё более абстрактные способы создания ПО (вайб-дизайн посредством GUI, то есть графического интерфейса, или высокоуровневых интерфейсов), которые уменьшат зависимость от универсальных библиотек. ИИ будет генерировать индивидуализированный код, а некоторое ПО будет даже развиваться самостоятельно на основе обратной связи от ИИ. В этом будущем успех в сфере программирования будет во многом зависеть от творческих аспектов, от человеческого умения распределять ресурсы и использовать сетевые возможности, потому что писать код станет проще, без необходимости заниматься рутинными и требующими одной лишь усидчивости задачами. Благодаря повсеместному распространению ИИ возникнут новые парадигмы пользовательского опыта, от разговорных интерфейсов до адаптивных пользовательских интерфейсов и тому подобного.

Во всём этом выделяется одна ключевая идея: слияние сильных сторон человека и ИИ. Ни одна сторона по отдельности не обладает такой мощью и такими возможностями, как они вместе. ИИ — это скорость, огромный объём знаний и отсутствие усталости. Человек — это осознанный выбор направления, глубина анализа, понимание ценности. Оптимальный рабочий процесс

будущего — это симбиоз; метафорически его можно представить как совместную работу мастера-ремесленника и сверхспособного подмастерья, который мгновенно подносит любой инструмент или справку. При этом создавать по-настоящему ценный и выдающийся продукт по-прежнему умеет только мастер.

Если вы разработчик, читающий эту книгу, знайте — настало время освоить эти новые инструменты и парадигмы. Книга поможет вам попробовать ИИ в роли программиста-ассистента для следующего проекта, научит вас разбивать задачи на обрабатываемые ИИ части и развить навык формулировки запросов и проверки результатов. Вместе с тем она призывает вас ещё сильнее полагаться на то, что действительно делает вас такими ценными и уникальными, а именно на способность проектировать системы, понимать пользователей и принимать решения, благодаря которым ПО будет лучше соответствовать реальности.

Используемые в этой книге условные обозначения

В этой книге используются следующие типографские условные обозначения:

Курсив

Используется для новых терминов, URL, адресов электронной почты, имён файлов и расширений файлов.

Моноширинный шрифт

Используется для листингов программного кода, а также в тексте для обозначения элементов программы, таких как имена переменных и функций, баз данных, типов данных, переменных среды, операторов и ключевых слов.



Подсказка или совет.



Предупреждение или предостережение.

O'Reilly Online Learning

На протяжении более 40 лет *O'Reilly Media* даёт знания и навыки в области технологий и бизнеса, а также предлагает материалы, помогающие компаниям добиваться успеха.

Наша уникальная сеть экспертов и новаторов делится своим опытом и знаниями в книгах, статьях и на нашей онлайн-платформе обучения. Онлайн-платформа O'Reilly предоставляет по запросу доступ к живым курсам, углублённым учебным программам, интерактивным средам программирования, а также обширной коллекции текстов и видео от O'Reilly и более чем 200 других издателей. Дополнительная информация указана на странице по адресу <https://oreilly.com>.

Как с нами связаться

Все касающиеся этой книги вопросы и комментарии направляйте издателю:

O'Reilly Media, Inc.

141 Stony Circle, Suite 195

Santa Rosa, CA 95401

800-889-8969 (США и Канада)

707-829-7019 (международные и местные звонки)

707-829-0104 (факс)

support@oreilly.com

<https://www.oreilly.com/about/contact.html>

У этой книги есть своя веб-страница, на которой публикуются исправления, примеры и дополнительная информация: <https://oreil.ly/BeyondVibeCoding>

Новости и информацию о наших книгах и курсах можно найти по адресу <https://oreilly.com>

ЧАСТЬ I

ОСНОВЫ

Введение: что такое вайб-кодинг?

ИИ меняет сам способ создания программного обеспечения тем, что предлагает новые парадигмы программирования, от свободной формулировки запросов до структурированной помощи. Представьте, что для создания какой-то программы вам не нужно долго и подробно расписывать все инструкции согласно правилам того или иного языка программирования; вы просто описываете нужное вам поведение программы — почти так же, как если бы, например, общались с коллегой, — а ИИ превращает ваши идеи в рабочий код. В этом и заключается суть так называемого *вайб-кодинга* с акцентом на запросы и исследование, с формулировками на естественном языке, с ожиданием того, что все возможные пробелы будут заполнены большой языковой моделью (*LLM, Large Language Model*). Термин «вайб-кодинг» совсем недавно придумал пионер ИИ-разработки Андрей Карпатый (<https://oreil.ly/Ot6CR>), описавший новый способ программирования, при котором разработчики «полностью отдаются вайбам» ИИ-помощника («вайб» от англ. *vibe* — «вибрация, дух, атмосфера, интуиция», по-русски это можно было бы передать примерно как «программирование на расслабоне», «интуитивное программирование» и т. д.).

В этой книге я постараюсь подробнее рассмотреть, что же этот вайб-кодинг означает для профессиональных разработчиков и как он соотносится с тем, что я называю «разработкой с помощью ИИ» (*AI-assisted engineering*), или «инженерным подходом при помощи ИИ», то есть с более формализованным процессом «дополненного программирования» (*augmented coding process*). Я расскажу, как меняется роль разработчика в эпоху ИИ, какие инструменты и рабочие процессы могут максимизировать эффективность разработчиков, как ИИ работает с кодовой базой и как решать возникающие при этом уникальные проблемы. Я также отмечу области, в которых вайб-кодинг особенно продуктивен и где он связан с трудностями; я затрону тему баланса между быстротой генерации кода со стороны ИИ и осознанным контролем со стороны человека. В итоге у вас должно сложиться ясное представление о том, как в своей практике программирования ответственно и эффективно пользоваться пресловутыми вайбами (интуицией, ощущениями, идеями) — так, чтобы не просто выдавать как можно больше кода на максимальной скорости, а чтобы стать более креативным и продуктивным разработчиком программного обеспечения в эпоху ИИ.