Оглавление

Предисловие от издательства 9

Об авторах 10

Предисловие 11

Для кого предназначена эта книга 12

Введение 13

Глава 1. Предпосылки 15

1.1. Введение в вайб-программирование 15

1.1.1. Воплощение идей 17

1.1.2. Креативность превыше технологий 18

1.2. Переход от командного стиля к целевому 20

1.2.1. Обзор императивного программирования 21

1.2.2. От команд к намерениям 22

1.2.3. Сравнение методов разработки 23

1.3. Основы вайб-программирования 25

1.3.1. Становление технологий больших языковых моделей 25

1.3.2. Стимулирующие факторы рыночного спроса 26

1.3.3. Ключевые проблемы разработчиков 28

1.4. Заключение 29

Глава 2. Эволюция способов программирования 30

2.1. Эволюция языков программирования 30

2.1.1. Общение с машиной 31

2.1.2. От двоичного кода к символьному языку 32

2.1.3. Возникновение языков программирования высокого уровня 36

2.1.4. Прорыв в области структурного и объектно-ориентированного программирования 39

2.1.5. Разнообразие современных моделей программирования 42

2.2. Эволюция взаимодействия при программировании 49

2.2.1. Зачатки программирования на физических носителях 49

2.2.2. Переход от строчного редактирования к полноэкранному взаимодействию 50

2.2.3. Эпоха интеграции: объединение редактирования, компиляции и отладки 52

2.2.4. Кросс-платформенная экосистема и реформа архитектуры плагинов 53

2.3. Возникновение разработки с минимальным кодом и без кода 54
2.3.1. Теоретическая система полноты по Тьюрингу 55
2.3.2. Ранние технологические исследования 56
2.3.3. Формирование концепции и первые практические шаги 58
2.3.4. Этапы развития рынка 58
2.3.5. Ограничения традиционных платформ с минимальным кодом и без кода 59
2.3.6. Использование ИИ в платформах с низким и нулевым уровнями кодирования 61
2.4. Заключение 62

Глава 3. Экосистема приложений вайб-программирования 63

3.1. Программирование с универсальными большими языковыми моделями 63
3.1.1. Вопросы и ответы с использованием больших языковых моделей 63
3.1.2. Большие языковые модели и их потенциал в программировании 65
3.1.3. Преимущества и недостатки диалогового режима программирования 66
3.2. Программирование с помощью IDE 66
3.2.1. Интеграция ИИ в плагины IDE 66
3.2.2. IDE с нативной интеграцией ИИ 68
3.2.3. Сравнение ИИ-интеграции в IDE 69
3.3. Сквозное программирование с помощью агентов 70
3.3.1. Концепция сквозного программирования с помощью агентов 70
3.3.2. Решения для сквозного программирования агентов 71
3.3.3. Механизм работы и системная архитектура 73
3.4. Будущее прикладных форм 75
3.4.1. Новые формы применения 75
3.4.2. Формы использования и сегментация пользователей 76
3.5. Заключение 78

Глава 4. Вайб-программирование: сценарии и практические примеры 79

4.1. Анализ сценариев применения 80
4.1.1. Быстрое создание прототипов продуктов 80
4.1.2. Возникновение «массового программирования» 82
4.1.3. Инструмент для новичков и препятствия для продвинутых 82
4.1.4. Автоматизация внутренних процессов компании: повышение эффективности и проблемы интеграции 83
4.2. Подробный анализ практических примеров 84
4.2.1. Примеры успеха независимых разработчиков 84
4.2.2. Опыт практического использования стартапами 88
4.2.3. Трансформация крупных предприятий 89
4.2.4. Адаптация и инновации сообщества Open Source 90
4.3. Заключение 92

Глава 5. Лучшие практические методики ... 93
5.1. Методики разработки промптов ... 93
5.1.1. Почему важны хорошие промпты ... 94
5.1.2. Основные принципы разработки промптов ... 95
5.1.3. Пример оптимизации промпта ... 98
5.1.4. Набор практических шаблонов подсказок ... 99
5.1.5. Настройка подсказок для Cursor ... 100
5.2. Планирование требований ... 108
5.2.1. Анализ требований ... 108
5.2.2. Составление документа с требованиями к продукту ... 111
5.2.3. Выбор ИИ-ориентированного технологического стека ... 114
5.2.4. Использование ИИ для генерации технических требований ... 117
5.3. Рецензирование и оптимизация кода ... 119
5.3.1. Ограничения ИИ ... 120
5.3.2. Распространенные дефекты качества ... 120
5.3.3. Некачественный код может привести к провалу проекта ... 125
5.3.4. Руководство по рецензированию кода в эпоху ИИ ... 126
5.3.5. Создание системы рецензирования кода ... 143
5.4. Инженерный подход ... 149
5.4.1. Введение в инженерию ... 150
5.4.2. Легкая инженерная система для вайб-программирования ... 151
5.5. Заключение ... 160

Глава 6. Практические примеры ... 161
6.1. Настройка среды ... 161
6.1.1. Подготовка инструментов ... 161
6.1.2. Структура проекта ... 169
6.2. Анализ требований проекта ... 171
6.2.1. Систематизация документации по требованиям ... 171
6.2.2. Составление документации по техническому проектированию ... 176
6.2.3. Составление документации плана реализации проекта ... 179
6.3. Разработка бэкенда ... 183
6.3.1. Принципы реализации ... 183
6.3.2. Разработка программы бэкенд-сервиса ... 185
6.3.3. Рецензирование кода ... 187
6.3.4. Тестирование интерфейсов ... 188
6.3.5. Доработка и дополнение функциональных возможностей ... 189
6.4. Разработка веб-систем ... 190
6.4.1. Подход к реализации ... 191
6.4.2. Разработка веб-страниц ... 193
6.4.3. Рецензирование кода ... 195
6.4.4. Вывод веб-страницы с помощью ИИ в соответствии с ожиданиями ... 196
6.4.5. Вызов реальных серверных интерфейсов ... 203
6.5. Развертывание приложения ... 206
6.5.1. Понимание логики развертывания кода ... 207

6.5.2. Развертывание приложения на Vercel ... 211
6.5.3. Реализация непрерывного развертывания с помощью GitHub Actions ... 217
6.6. Заключение ... 222

Глава 7. Ограничения и проблемы ... 224
7.1. Точка зрения потребителя ... 225
7.1.1. Обычный потребитель ... 225
7.1.2. Профессиональные разработчики ... 227
7.2. Точка зрения разработчика ... 229
7.2.1. Неудобное позиционирование продукта ... 229
7.2.2. Проблема издержек ... 230
7.2.3. Различные сценарии взаимодействия с потребителем ... 231
7.3. Революция мышления разработчиков и доступность технологий для всех ... 232
7.3.1. Смена мышления разработчиков старого поколения ... 232
7.3.2. Ключевые навыки разработчика новой эпохи ... 233
7.3.3. Глубина в одной области и охват в других: расцвет междисциплинарных навыков ... 235
7.3.4. Технологическое равенство для всех ... 236
7.3.5. Корректировка карьерного роста и образовательных траекторий ... 238
7.4. Заключение ... 239

Послесловие ... 241

Предметный указатель ... 242

Предисловие от издательства

Отзывы и пожелания

Мы всегда рады отзывам наших читателей. Расскажите нам, что вы думаете об этой книге – что понравилось или, может быть, не понравилось. Отзывы важны для нас, чтобы выпускать книги, которые будут для вас максимально полезны.

Вы можете написать отзыв на нашем сайте www.dmkpress.com, зайдя на страницу книги и оставив комментарий в разделе «Отзывы и рецензии». Также можно послать письмо главному редактору по адресу dmkpress@gmail.com; при этом укажите название книги в теме письма.

Если вы являетесь экспертом в какой-либо области и заинтересованы в написании новой книги, заполните форму на нашем сайте по адресу http://dmkpress.com/authors/publish_book/ или напишите в издательство по адресу dmkpress@gmail.com.

Список опечаток

Хотя мы приняли все возможные меры для того, чтобы обеспечить высокое качество наших текстов, ошибки все равно случаются. Если вы найдете ошибку в одной из наших книг – возможно, ошибку в основном тексте или программном коде, – мы будем очень благодарны, если вы сообщите нам о ней. Сделав это, вы избавите других читателей от недопонимания и поможете нам улучшить последующие издания этой книги.

Если вы найдете какие-либо ошибки в коде, пожалуйста, сообщите о них главному редактору по адресу dmkpress@gmail.com, и мы исправим это в следующих тиражах.

Нарушение авторских прав

Пиратство в интернете по-прежнему остается насущной проблемой. Издательство «ДМК Пресс» очень серьезно относится к вопросам защиты авторских прав и лицензирования. Если вы столкнетесь в интернете с незаконной публикацией какой-либо из наших книг, пожалуйста, пришлите нам ссылку на интернет-ресурс, чтобы мы могли применить санкции.

Ссылку на подозрительные материалы можно прислать по адресу dmkpress@gmail.com.

Мы высоко ценим любую помощь по защите наших авторов, благодаря которой мы можем предоставлять вам качественные материалы.

Об авторах

Фан Вэньцзе (Tecvan). Ведущий фронтенд-архитектор одной из крупных компаний, руководитель официального аккаунта Tecvan, автор руководства «Основные принципы и практическое применение Webpack 5» для платформы Juejin, ведущий подкаста Langshuo. Специализируется в области инженерных аспектов фронтенд-разработки и практического применения ИИ в программировании. На протяжении многих лет выступает в качестве лектора на мероприятиях D2 Terminal Technology Conference, GIAC (Global Internet Architecture Conference) и Frontend Zaozao Chat.

Ли Цзяньчао (Isboyjc). Опытный разработчик полного цикла, вайб-программист. Автор канала «Несерьезный фронтенд» на стриминговой платформе Bilibili. Специализируется на веб-разработке, проектировании системных архитектур, больших языковых моделях и создании приложений с ИИ-агентами. Активно занимается просветительской работой в области технологий искусственного интеллекта.

Ю И (Янь Сяофань). Старший фронтенд-архитектор команды по трансграничной электронной коммерции. Автор пособий «Проекты NestJS на практике» и «Практики DevOps на основе Node» на платформе Juejin. Специализируется на построении систем фронтенд-инжиниринга и проектировании архитектуры клиентских приложений. Выступал в качестве приглашенного эксперта на технологическом салоне ByteDance и в качестве лектора на конференции Rare Earth Juejin.

Чжан Бинь (Captain). Опытный координатор сообщества разработчиков, ведущий подкаста Langshuo. Основатель базы знаний AGI Juejin. Активный популяризатор вайб-программирования.

Предисловие

Программирование никогда не стоит на месте и постоянно развивается на волне технического прогресса. От механических перфокарт до структурного программирования, от объектно-ориентированного подхода до функциональной парадигмы – каждая из этих трансформаций изменяла способ взаимодействия человека с машиной. Сегодня перед нами предстала совершенно новая революция. **Вайб-программирование** – это не просто технологическая тенденция, а скорее переход к новому способу мышления, который превращает программирование из «императивного» набора точных инструкций в «интенциональное» выражение вдохновения. Эта концепция, впервые предложенная Андреем Карпатым (Andrej Karpathy) и сопровождаемая расцветом технологий генеративного ИИ, с невиданной скоростью меняет подход разработчиков к созданию ПО наряду с самой структурой их образа мышления.

Суть вайб-программирования заключается в раскрепощении и расширении потенциала разработчиков. Благодаря устойчивому прогрессу в области технологий больших языковых моделей разработчикам больше не нужно вводить код строка за строкой – вместо этого они могут выражать свои намерения на естественном языке и вместе с ИИ проходить весь путь от идеи до ее воплощения. Появление таких инструментов, как ChatGPT, Claude и Cursor, делает программирование доступным не только для избранных, но и для широкой аудитории, снижая технический порог вхождения и стимулируя креативность в различных дисциплинах. От независимых разработчиков, быстро создающих прототипы продуктов с помощью ИИ, до предприятий, использующих интеллектуальные системы для автоматизации сложных рабочих процессов, вайб-программирование меняет экосистему разработки ПО благодаря своей эффективности и простоте.

Путь этой трансформации не будет легким. Проблемы отладки кода, сгенерированного ИИ, ограничения производительности, угрозы безопасности, а также сложности адаптации к сложным проектам требуют от нас осознания того, что за технологическими удобствами скрывается необходимость тщательного анализа и освоения новых профессиональных навыков. Разработчики будущего будут заниматься не столько написанием кода, сколько постановкой задач, проектированием систем и интеграцией различных областей знаний. Вайб-программирование трансформирует инструменты и процессы программирования, но также подталкивает к «революции мышления» с приоритетом всестороннего подхода, способствуя развитию широкого кругозора и открывая эпоху технологического равенства.

Не исключено, что в будущем вайб-программирование полностью устранит границы между техническими и нетехническими сферами, давая возможность стать творцами всем желающим. Это не просто обновление инструментов, а новая глава в истории сотрудничества человека и машины. Данная книга

призвана познакомить вас с сутью этой трансформации, исследовать ее технические основы, сценарии применения и практические способы реализации, а также честно рассказать о ее ограничениях и вызовах. Надеюсь, вы найдете в ней вдохновение и в симфонии кода и замыслов обретете свой собственный ритм программирования.

Чжан Бинь (Captain)
Июнь 2025 года

Для кого предназначена эта книга

Эта книга подходит как для профессиональных разработчиков с определенным техническим багажом, так и для менеджеров по продукту, UI/UX-дизайнеров и даже обычных пользователей, которые хотят разобраться в тенденциях программирования с использованием ИИ и заниматься инновациями с низким порогом входа.

Если вы профессиональный разработчик и хотите научиться эффективно и системно использовать инструменты программирования с ИИ, эта книга поможет вам сформировать концептуальную основу и избежать распространенных ошибок. Если вы являетесь менеджером по продукту или UI/UX-дизайнером, при этом имеете некоторое представление о программировании, но ваши базовые знания в этой области оставляют желать лучшего, эта книга поможет вам открыть для себя новые возможности в области разработки продуктов и повысить эффективность работы с помощью вайб-программирования. Даже если вы не занимаетесь технической работой, а просто интересуетесь влиянием ИИ на методы работы и творчества, эта книга поможет вам понять базовую логику происходящих изменений.

Эпоха вайб-программирования только начинается, и вне зависимости от ваших предпочтений – принять участие, выждать, критиковать или просто разобраться – вам обязательно нужно ознакомиться с реальной сутью этой технологической революции.

Введение

Цифровая экономика набирает обороты по всему миру. Начиная с 2023 года инновации в области технологий программирования с использованием искусственного интеллекта (ИИ) стали бесспорным движущим фактором модернизации промышленности. Вайб-программирование является новой парадигмой кодирования, в основе которой лежит модель разработки «что видишь, то и получаешь». Благодаря диалоговому взаимодействию разработчики могут быстро преобразовывать свои идеи в рабочий код, что значительно снижает барьер технической сложности для начинающих. Все больше предприятий, профессиональных разработчиков и даже обычных пользователей начинают знакомиться и использовать такие ИИ-сервисы для программирования, как GitHub Copilot, Cursor, Claude Code и Devin. Вайб-программирование постепенно выходит за пределы технического сообщества и формирует новую коммерческую экосистему.

Ярким свидетельством стремительного развития вайб-программирования являются данные статистики. По прогнозам Polaris Research, к 2032 году рынок ИИ-инструментов для программирования вырастет до $271,7 млрд и станет важнейшей инфраструктурой в сфере разработки ПО. Ожидается, что уже в 2025 году объем мирового рынка программных средств с ИИ-поддержкой будет превышать $250 млрд, а годовой совокупный темп роста сохранится на уровне более 35 %. С прикладной точки зрения, до 75 % пользователей платформы Replit никогда не писали программ до знакомства с ней, но с помощью ее ИИ-функции ghostwriter смогли успешно воплотить свои идеи в реальные продукты.

Кроме того, многие ведущие компании и стартап-команды проявляют острую прозорливость и активно переходят на вайб-программирование. Генеральный директор Google как-то сообщил, что более 25 % нового кода в компании генерируется искусственным интеллектом. Это свидетельствует о значительной роли ИИ в обширной кодовой базе Google, который обеспечивает ее эффективную работу. В известном стартап-инкубаторе из Кремниевой долины Y Combinator также сообщили, что среди участников мероприятия Winter Demo Day 2025 (W25 Demo Day) четверть стартап-команд используют ИИ для генерации 95 % своего кода. Благодаря ИИ они выходят на рынок с минимальными затратами, обеспечивают быструю итерацию продуктов и занимают выгодные позиции в условиях жесткой конкуренции.

Эта книга написана на основе личного опыта авторов, которые стали свидетелями данной трансформации. За последние два года, работая в качестве разработчиков, технических исследователей и активных пользователей продуктов ИИ, мы постоянно испытывали восторг от появления новых инструментов, идей и моделей. В то же время мы обнаружили, что многие люди по-прежнему имеют неправильное представление о вайб-программировании: кто-то счита-

ет его просто «интеллектуальным автозаполнением», кто-то теряется в многообразии терминов, а кто-то хочет начать, но не знает с чего.

Эта книга является попыткой систематически, объективно и доступно проследить эволюцию и основную логику развития данной технологии, а также продемонстрировать практические сценарии применения вайб-программирования в работе отдельных специалистов, команд и предприятий на примере реальных проектов в сфере ИИ. **Глава 1** начинается с истории появления вайб-программирования и помогает читателю по-новому взглянуть на роль искусственного интеллекта в трансформации парадигмы программирования. **Глава 2** посвящена развитию языков и инструментов программирования. **Глава 3** рассказывает о развитии экосистемы приложений вайб-программирования и ее перспективах. **Глава 4** посвящена сценариям применения и практическим примерам вайб-программирования. **Глава 5** представляет собой обзор научно обоснованных и технически проработанных методов вайб-программирования. **Глава 6** описывает создание полнофункционального ИИ-приложения с нуля и помогает читателю эффективно использовать соответствующие инструменты и методы программирования. **Глава 7** посвящена ограничениям и вызовам, с которыми сталкивается современная экосистема вайб-программирования, чтобы помочь читателю всесторонне оценить возможности этой новой парадигмы программирования.

Глава 1

Предпосылки

На протяжении последних десятилетий программирование строилось на командах и логике, а выполнение задач на компьютере осуществлялось с помощью построчных последовательностей кода. Однако в настоящее время в этой области наблюдается постепенный процесс изменения парадигмы. Слияние таких новых технологий, как большие языковые модели (Large Language Model, LLM), обработка естественного языка (Natural Language Processing, NLP) и интерактивное программирование (Interactive Programming), привело к появлению совершенно нового подхода к программированию, который в большей степени похож на человеческое мышление, а также более эффективен и гибок. Этим подходом стало **вайб-программирование** (Vibe Coding)[1].

Вайб-программирование – это не просто новая разновидность технологии, но и глубокая трансформация самой концепции разработки и взаимодействия между человеком и машиной. Переход от командного взаимодействия к программированию на основании замыслов, от приспособления человека к машине до понимания человеческого мышления самим компьютером – все это переопределяет границы возможностей современной программной отрасли.

В этой главе читатель шаг за шагом приобщится к сути и идеям вайб-программирования. Особое внимание будет уделено ключевым концепциям этой парадигмы, истории ее развития и предпосылкам ее роста, а также раскрытию ее потенциала и ценности как новой нормы в индустрии разработки.

1.1. Введение в вайб-программирование

В ноябре 2022 года ChatGPT, созданный компанией OpenAI, кардинально изменил устоявшиеся модели взаимодействия человека и машины, ворвавшись в массовое сознание с совершенно новой концепцией. В то время как люди все еще находились в плену стереотипов, связанных с сенсорными интерфейсами, ChatGPT впервые предложил использовать естественный язык в качестве основного средства общения между человеком и машиной, беспрецедентно сократив дистанцию между ними. Теперь человек может общаться с ИИ так же, как со своим приятелем.

---

[1] В настоящее время термин Vibe Coding переводится на русский язык как «программирование по настроению», «интуитивное программирование» «вайб-программирование» или просто «вайб-кодинг». Для обеспечения однозначности и единообразия терминологии в данной книге используется термин «вайб-программирование». – *Прим. перев*.

На момент появления ChatGPT у него все еще наблюдались явные технические недоработки, такие как ограниченный объем обучающих данных, отсутствие возможности подключения к интернету в режиме реального времени и пробелы в базе знаний, вследствие чего его функциональность в целом была еще далека от совершенства. Несмотря на это, даже в таком виде он уже обладал впечатляющими способностями восприятия и синтеза речи, вполне достаточными для работы с большинством повседневных запросов. На удивление многих экспертов, эволюция экосистемы больших языковых моделей происходила гораздо быстрее, чем ожидалось: менее чем за 3 года она трансформировалась из системы для ответов на вопросы на естественном языке в более комплексную интеллектуальную структуру, включающую мультимодальную интеграцию и агентный ИИ (agentic AI).

Все более широкое распространение диалоговых моделей на основе естественного языка постепенно меняет привязанность людей к поисковым системам, основанным на сопоставлении ключевых слов и частичном совпадении. Все больше людей предпочитают задавать вопросы непосредственно большим языковым моделям: от ранней ChatGPT от OpenAI до более поздних Doubao от ByteDance, Tongyi от Alibaba Cloud и Yuanbao от Tencent. Все эти ИИ-инструменты в корне меняют процесс получения информации. Эти ИИ-инструменты не только способны понимать приблизительные и нестандартные человеческие формулировки, но также могут создавать структурированные и релевантные ответы благодаря глубокому пониманию семантики. Такое преимущество существенно повышает эффективность поиска.

Такая беспрецедентная эффективность взаимодействия и способность к пониманию дают разработчикам возможность возлагать на ИИ более высокие ожидания: не просто получать идеи для разрешения задач или образцы кода, но и реально участвовать в креативном процессе и программировании. В связи с развитием и продвижением больших языковых моделей для программирования многие задаются вопросом: если машина сможет понять такое повседневное выражение, как «страница товара с корзиной», то можно ли будет отказаться от традиционного способа написания кода, предполагающего ввод каждой строки вручную?

Из этой идеи постепенно сформировалась совершенно новая парадигма – вайб-программирование. В феврале 2025 года Андрей Карпатый (Andrej Karpathy), один из основателей OpenAI, впервые представил эту концепцию в социальных сетях. Ее суть заключается в использовании больших языковых моделей для максимального приближения процесса разработки к формулировкам естественного языка и создания нового типа программирования, основанного на совместном творчестве человека и машины.

Говоря проще, вайб-программирование можно сравнить с тем, как будто у каждого разработчика появился универсальный «технический помощник» – что-то вроде Джарвиса (J.A.R.V.I.S.) из фильма «Железный человек» (Iron Man), который создает незримый, но чрезвычайно эффективный мост между разработчиком и требованиями к конечному продукту. Программистам больше не нужно тратить силы на разбор синтаксиса кода, выбор фреймворков или тонкости низкоуровневой реализации. Они могут сосредоточиться на ключевых вопросах: что нужно пользователям и какими функциями должен обладать продукт, доверив остальные задачи искусственному интеллекту.

Благодаря глубокому обучению на огромных массивах открытого исходного кода, системных архитектур и моделей разработки большие языковые модели не только освоили различные технологические стеки и лучшие отраслевые практики, но также приобрели способность трансформировать приблизительные пожелания в высококачественный код. С помощью вайб-программирования абстрактные творческие идеи могут быть оперативно преобразованы в практичные, работоспособные программные продукты, что значительно сокращает путь от зарождения идеи до ее воплощения. Традиционное программирование подобно исполнению конкретных команд, тогда как вайб-программирование больше напоминает общение с большой языковой моделью: разработчик задает направление и цели, модель понимает контекст и дополняет детали, чтобы в итоге создать высококачественный код совместными усилиями.

1.1.1. Воплощение идей

При использовании традиционной модели разработки ПО полный цикл создания продукта обычно включает в себя несколько этапов. Для начала команды по исследованию рынка и анализу потребностей пользователей изучают запросы и выявляют проблемные моменты, формируя предварительную концепцию продукта. Затем, на основе результатов исследования, менеджер по продукту составляет четко структурированный **документ с требованиями к продукту** (Product Requirement Document, PRD). Далее на основе этого документа специалисты по дизайну разрабатывают прототипы интерфейса, сценарии взаимодействия и визуальный дизайн, обеспечивая высокое качество взаимодействия с пользователем в дополнение к функциональности продукта. Наконец, на основе проектной документации и документа с требованиями к продукту команда разработчиков корректно интерпретирует замысел и воплощает его в виде исполняемого кода.

Чем больше этапов и участников в таком процессе, тем выше затраты на взаимодействие и тем сложнее становятся возможные сложности. Еще более серьезной проблемой является техническая ограниченность разработчиков и недостаточно глубокое понимание требований к продукту, что затрудняет полноценное и точное воплощение проектных требований в виде стабильной и работоспособной программной системы. Это зависит не только от технической зрелости разработчиков, но и от способности всей команды правильно интерпретировать, согласовывать и реализовывать задачи.

Возьмем, к примеру, такую простую функцию, как «изменение цвета при нажатии кнопки». При традиционном подходе к разработке ПО даже для реализации такой базовой функции потребуется написание нескольких строк кода вручную, что включает в себя работу со структурой HTML, стилями CSS и логикой JavaScript.

```
<button id="colorBtn">Нажмите, чтобы изменить цвет </button>
<script>
  var btn = document.getElementById("colorBtn");
  btn.addEventListener("click", function() {
    btn.style.backgroundColor = "#ff0000";
  });
</script>
```

Для технически неосведомленных или начинающих разработчиков этот простой фрагмент кода может показаться «загадочным текстом». Комплексная программная система состоит из бесчисленного множества подобных базовых функциональных модулей, которые послойно собираются и связываются в соответствии со строгой архитектурной концепцией, нормами кодирования и техническими стандартами. Именно по этой причине программисты с навыками системного программирования долгое время оставались дефицитным человеческим ресурсом.

Появление генеративного ИИ кардинально меняет эту ситуацию. Благодаря возможностям больших языковых моделей роль разработчика трансформируется из «создателя кода» в «проектировщика функциональности». Разработчику не требуется вникать в программные фреймворки, API-вызовы или детали низкоуровневой реализации, достаточно лишь четко сформулировать свои требования на естественном языке. Например, для реализации функции «изменение цвета при нажатии кнопки» разработчику не нужно понимать технические детали, такие как обработка событий или работа с **моделью объектов документа** (Document Object Model, DOM). Достаточно просто задать ИИ запрос: «Создай красную кнопку, которая при нажатии станет синей», – чтобы ИИ за несколько секунд сгенерировал высококачественный код с оптимальной структурой, соответствующий лучшим практикам, и даже автоматически добавил дополнительные функции, такие как анимационные эффекты, адаптивный макет и совместимость со средствами доступа. Эта революция полностью разрушила технические барьеры программирования, превратив разработку ПО из «технической игры для избранных» в «творческий инструмент для всех».

1.1.2. Креативность превыше технологий

Суть вайб-программирования вовсе не заключается в отрицании ценности программных технологий. Смысл в том, чтобы технологии стали «невидимой инфраструктурой». Так же, как при повседневном использовании мобильного телефона нам не нужно заботиться об архитектуре чипа и принципах его работы, в будущем при разработке ПО тоже не придется зацикливаться на деталях реализации кода. Постоянное обучение ИИ незаметно устраняет барьеры, связанные с базовыми технологиями, и превращает ИИ в эффективный механизм преобразования, соединяющий запросы и их реализацию.

Как только стоимость воплощения идей приблизится к нулю, ключевым конкурентным преимуществом разработчиков вновь станет сама креативность: умение определять потребности пользователей, проектировать интерфейс взаимодействия и планировать функциональную архитектуру. Именно в этом заключается основная трансформация, которую приносит ИИ: технологии работают на креативность, а не наоборот. Как опытные разработчики, так и обычные люди без навыков программирования смогут воплощать свои идеи в жизнь благодаря взаимодействию с ИИ на естественном языке.

Техническим специалистам ИИ может помочь в превращении из мастеров написания кода в системных архитекторов.

Специалисты в сфере технологий:
от мастеров написания кода к системным архитекторам

Опытный бэкенд-инженер Лао Чжан в прошлом затрачивал три дня на отладку интерфейса функции календаря посещаемости сотрудников. Сегодня благодаря вайб-программированию ему достаточно всего лишь сказать ИИ:

```
Создай календарь посещаемости с функцией переключения месяцев;
Обеспечь возможность просмотра записей о посещаемости при нажатии
на дату;
Отмечай опоздания сотрудников красным цветом, данные должны синхро-
низироваться с существующим бэкенд-интерфейсом.
```

ИИ не только оперативно генерирует фронтенд-код с библиотекой React-компонентов, но и автоматически согласовывает его с форматом Java-интерфейса, а также добавляет анимацию загрузки данных. Основная задача Лао Чжана теперь состоит не в построчной отладке стилей, а в оптимизации алгоритма учета рабочего времени (даже этот этап может быть реализован с помощью Vibe), что повышает эффективность разработки примерно на 60 %. Это яркий пример того, как технические специалисты совершают впечатляющий переход от рутинной работы к творческой деятельности.

Людям без опыта программирования ИИ дает возможность воплотить свои творческие идеи в реальные продукты.

Люди без опыта программирования:
от авторов идей до владельцев продуктов

Вышедшая на пенсию учительница госпожа Ван мечтала разработать приложение для семейного фотоальбома, чтобы ее дети, проживающие в разных городах, могли в режиме реального времени обмениваться фотографиями из своей повседневной жизни. Она озвучила ИИ свои общие пожелания следующим образом:

```
Возможность загружать фотографии;
После входа в систему члены семьи должны видеть фотоальбомы друг
друга в хронологическом порядке;
Также должна быть возможность оставлять комментарии.
```

ИИ, как заботливый помощник, помог ей уточнить детали:

```
Входить через WeChat или регистрировать аккаунт;
Нужны ли теги для классификации фотографий;
Создавать ли видео с воспоминаниями за каждый год.
```

Госпожа Ван сделала следующие уточнения:

```
Вход через WeChat;
Классификация по тегам «Путешествия», «Праздники» и т. д.;
Создание видео с воспоминаниями за каждый год под фоновую музыку
«Дорогой путешественник».
```

С помощью ИИ госпожа Ван всего за три дня создала и доработала полнофункциональное приложение с интеллектуальной сортировкой и функцией монтажа видео. Более того: когда ей показалось, что видео генерируется слишком медленно, она просто попросила: «Ускорьте создание видео», после чего ИИ автоматически оптимизировал алгоритм сжатия изображений. Так, не имея ни малейшего представления о программировании, она обзавелась собственным семейным приложением для общения.

Несмотря на то что ИИ пока что не в состоянии решать все сложные задачи разработки, он уже с легкостью справляется с большинством повседневных задач. Будь то создание сайта электронной коммерции, разработка приложения для ведения дневника или дизайн небольшой развлекательной игры, ИИ способен точно определить и доработать требования, а также быстро сгенерировать готовый к использованию код.

Революционная сущность вайб-программирования заключается в том, что оно перестраивает систему координат мышления разработчика. Оно не преуменьшает важность мышления, а помогает разработчику перенести акцент с «мелочей технической реализации» на «основные задачи создания ценности». Когда генерация кода станет такой же простой, как набор текста, конкурентоспособность продукта будет зависеть не от программистских навыков разработчика, а от его способности анализировать потребности клиентов и раскрывать творческий потенциал.

Вполне возможно, что дальнейшая эволюция и распространение ИИ повлекут за собой новый цикл перемен в плане повышения производительности труда в цифровую эпоху. Так же, как пакет Office сделал обработку текстов и статистический анализ данных базовыми навыками для работников офиса, ИИ, в свою очередь, может стать цифровой инфраструктурой нового поколения, которая разрушит технологические барьеры и даст каждому человеку возможность свободно воплощать свои идеи в жизнь.

1.2. Переход от командного стиля к целевому

В 50-х годах XX века программирование представляло собой чрезвычайно сложную профессию. Разработчики того времени должны были не только в совершенстве владеть логикой работы ПО, но также глубоко понимать принципы работы аппаратной платформы. В те времена программирование больше напоминало своего рода искусство работы с языком аппаратного уровня, когда с машиной велся прямой диалог с помощью строк двоичных инструкций.

По мере развития компьютерных наук и появления высокоуровневых языков программирования эффективность разработки заметно возросла. Но даже несмотря на постоянное совершенствование технологических инструментов – от ассемблера до высокоуровневых языков, от командной строки до **интегрированных сред разработки** (Integrated Developer Environment, IDE), процесс создания ПО по-прежнему строился по принципу «человек пишет код, машина выполняет». Такая модель не только оказала глубокое влияние на процесс разработки, но и сформировала привычный рабочий режим программистов.

Конец ознакомительного фрагмента.
Приобрести книгу можно
в интернет-магазине
«Электронный универс»
e-Univers.ru