САВЕЛИЙ ОРЛОВ

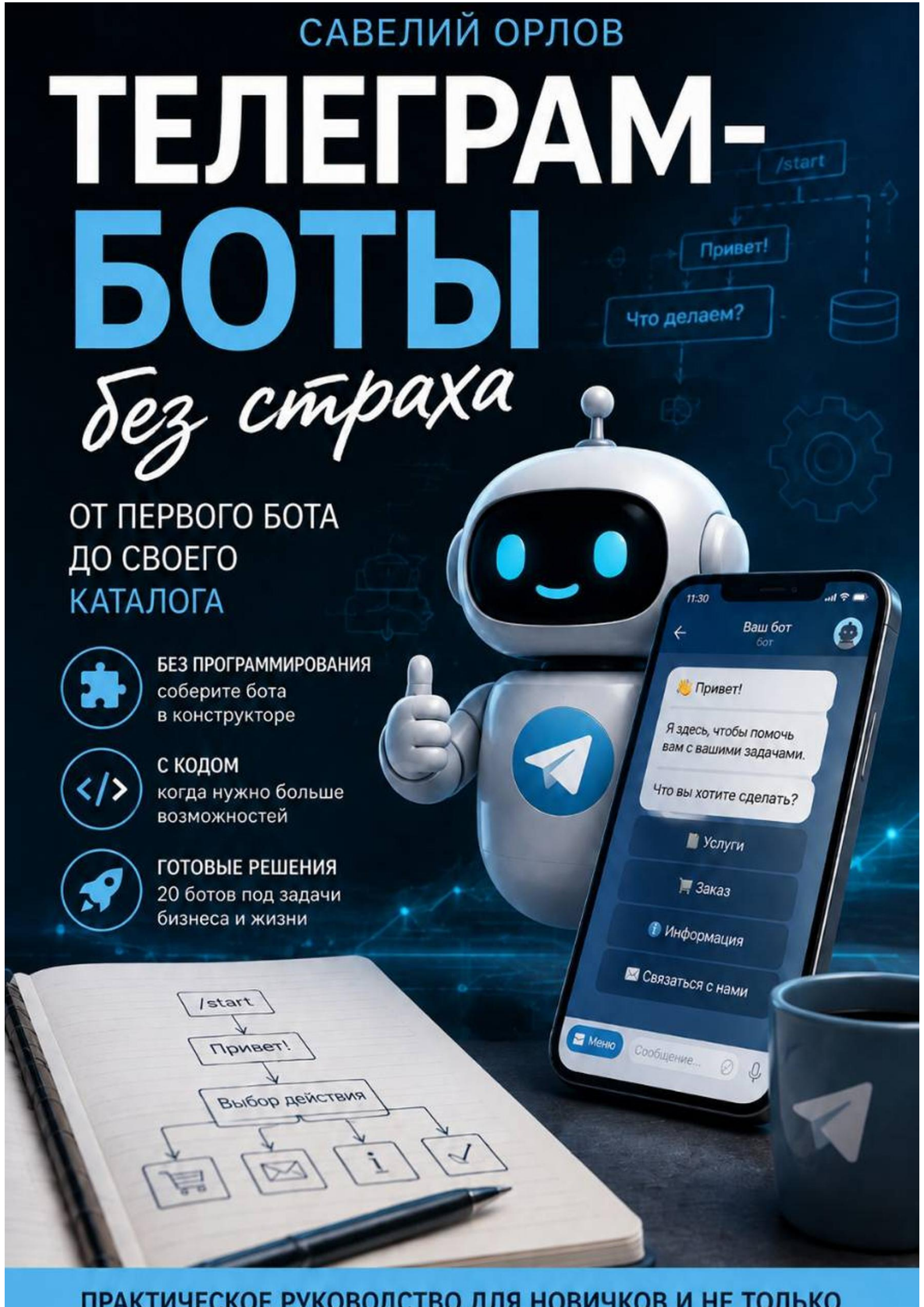
ТЕЛЕГРАМ-БОТЫ *без страха*

ОТ ПЕРВОГО БОТА
ДО СВОЕГО
КАТАЛОГА

БЕЗ ПРОГРАММИРОВАНИЯ
соберите бота
в конструкторе

С КОДОМ
когда нужно больше
возможностей

ГОТОВЫЕ РЕШЕНИЯ
20 ботов под задачи
бизнеса и жизни



ПРАКТИЧЕСКОЕ РУКОВОДСТВО ДЛЯ НОВИЧКОВ И НЕ ТОЛЬКО

Савелий Орлов

Телеграм-боты без страха. От первого бота до своего каталога

«Автор»
2026

Орлов С.

Телеграм-боты без страха. От первого бота до своего каталога / С. Орлов — «Автор», 2026

Сделать Telegram-бота можно, даже если вы никогда не программировали.Эта книга проведёт вас от простой схемы на бумаге до работающего бота. Сначала — без единой строки кода: регистрация, конструктор, меню, сбор заявок, напоминания и подключение нейросети. Затем, если возможностей конструктора станет мало, — Python, собственная логика и размещение бота на сервере.Один пример проходит через всю книгу: бот записи на услугу постепенно превращается из простой схемы в полноценный проект. А в финале вас ждёт каталог из двадцати ботов — от меню и приёма заявок до трекера привычек, учёта расходов и личного помощника с нейросетью.Вам не нужно становиться программистом. Достаточно понять логику бота — остальное вы освоите шаг за шагом.

Содержание

Телеграм-боты без страха. От первого бота до своего каталога

https://t.me/saveliyorlov/50

Вечер, половина десятого. Мастер маникюра отвечает на сообщение в директе. «Здравствуйте, а сколько стоит покрытие?» Она пишет цену. Следующее сообщение: «А когда можно?» Она открывает записную книжку, смотрит, отвечает. Ещё сообщение — от другого человека: «Здравствуйте, а сколько стоит покрытие?»

Это третий раз за сегодня и, наверное, сотый за месяц. Один и тот же вопрос, один и тот же ответ, набранный заново.

Где-то на этом месте появляется мысль: должен же быть бот, который это делает. Мысль появляется — и тут же упирается. Бот — это программирование. Программирование — это не ко мне.

Эта книга написана ровно про промежуток между «должен быть бот» и «я не программист».

Что такое бот

Одним предложением, без терминов:

Бот — это переписка, в которой за одну сторону заранее написали ответы.

Человек пишет или нажимает кнопку — бот отвечает тем, что для этого случая приготовлено. Больше в нём ничего нет. Ни экрана, который нужно рисовать, ни установки, ни версий для разных телефонов. Только переписка.

Из этого следует главное практическое обстоятельство, ради которого стоит читать дальше: **чтобы завести бота, нужно не программирование, а список ответов.** Кто составил список, тот соберёт бота. Кто не составил, не соберёт даже с программистом за деньги — потому что программисту нечего будет собирать.

Список составляется на бумаге. Инструмент выбирается потом.

Две части и кому какая

Частей три, но развилка в книге одна, и проходит она посередине. Читателей тоже двое — и это, скорее всего, один и тот же человек с разницей в две недели.

Часть первая — бот без кода. Собирается в конструкторе: вы соединяете готовые блоки, ничего не программируя. К концу части у вас работающий бот, который принимает заявки, задаёт вопросы, складывает ответы в таблицу, напоминает клиентам о визите и умеет обращаться к нейросети. Кода в этой части нет ни строчки — ни одной, это обещание.

Часть вторая — бот на своём коде. Она нужна не всем и не сразу. Конструктор рано или поздно упирается в потолок: не хватает гибкости, вырастает абонентская плата, требуется что-то, чего в нём просто не предусмотрено. Вторая часть — про то, что делать дальше. Python, библиотека для ботов, размещение бота на сервере, чтобы он работал без вашего ноутбука.

Часть третья — каталог. Двадцать ботов с описанием: что делает, кому нужен, собирается в конструкторе или требует кода, за сколько собирается, где ломается. Читается не подряд, а по своей задаче.

Порядок чтения такой: **первая часть — всем, вторая — по необходимости, третья — под свою задачу.**

Если вы уже пишете код, первую часть можно пролистать. Но две главы из неё прочитайте: про то, из чего состоит бот, и про схему разговора. Они не про инструменты, они про устройство, и во второй части всё стоит на них.

Где лежит код и почему его нет здесь

Во второй части книги вы не найдёте программ, напечатанных целиком. Это сделано намеренно, и причину стоит объяснить сразу, чтобы она не выглядела экономией на читателе.

Электронная книга переверстывается под экран. Строки переносятся там, где решит читалка, отступы съезжают, моноширинный шрифт заменяется обычным. Для обычного текста это неважно. Для Python — смертельно: в нём отступы несут смысл, и программа с поехавшими отступами не запустится. Читатель на телефоне получил бы код, который гарантированно не работает, и был бы прав, поставив книге единицу.

Поэтому сделано так:

Весь код лежит по ссылке, в открытом хранилище (в приложении к книге есть ссылки на зеркала, а также qr-код с ссылкой на эти зеркала). Скачивается целиком или по одному файлу.

В книге — фрагменты по три-семь строк с разбором: что эта часть делает, что в ней меняют под себя, что трогать не нужно.

Каждый файл подписан так же, как сценарий в книге. Читаете про бота записи на услугу — в хранилище лежит файл с этим названием.

Побочная выгода общая: код в хранилище обновляется, когда меняются версии библиотек. Книга при этом не устаревает, потому что в ней объяснена логика, а логика не меняется.

Ссылка повторена в начале второй части — искать её по всей книге не придётся.

Что вы получите

Работающий бот под вашу задачу — сегодня в конструкторе, через неделю на своём коде.

Один бот, который делает одно дело и делает его до конца. Не десять ботов, не система, не «инфраструктура».

Реалистичные сроки, чтобы вы могли планировать: первый бот, отвечающий на сообщение, — полчаса. Бот записи на услугу, собранный в конструкторе по схеме, — вечер. Тот же бот на своём коде, размещённый на сервере, — две-три недели занятий по вечерам, если раньше вы кода не писали.

Последняя цифра не преувеличена и не преуменьшена. Обещания «за выходные» относятся к тем, кто уже умеет.

Чего в этой книге нет

Обещаний заработка. Ни слова про то, сколько можно получать, собирая ботов на заказ. Такие деньги существуют, но это другая профессия и другая книга.

Автоворонок, прогревов и всего, что к ним прилагается. Не потому, что это плохо, а потому что это про продажи, а не про ботов. Бот там — инструмент, и научиться ему можно отдельно.

Уверений, что это просто. Местами будет неприятно. Установка Python на чужой машине умеет портить вечер, а первое размещение бота на сервере — тем более. В таких местах в книге стоит блок «если не получилось» с точным текстом ошибок и объяснением, что они означают.

Историй успеха с цифрами. Ни одного «Марина собрала бота и увеличила выручку втрое».

Обещания, что бот заменит человека. Не заменит там, где нужен человек. Бот хорош на повторяющемся: одни и те же вопросы, один и тот же порядок действий. Переговоры, оценка ситуации, ответственность за чужие деньги остаются вам.

Предупреждение об устаревании

Часть того, что здесь написано, устареет, и лучше знать заранее, какая именно.

Устареет быстро: названия конструкторов, их тарифы, доступность оплаты, лимиты бесплатных версий. Версии библиотек. Цены на размещение кода. Всё это вынесено в приложение в конце книги — там оно собрано в одном месте и переписывается при переиздании, не трогая остальной текст.

Устареет медленно: расположение пунктов в интерфейсах, порядок действий при регистрации. Поэтому в книге нет описаний вида «нажмите кнопку справа вверху» — описывается логика: что нужно сделать, а не куда нажать. Логика переживает переделку интерфейса, а описание кнопки — нет.

Не устареет: устройство бота, схема разговора, места, где боты ломаются. Первые две главы и восьмая написаны так, что будут верны и через пять лет с любым инструментом.

Что пригодится из других книг серии

Если вы никогда не пользовались нейросетями — начните с книги **«Нейросети для работы»**. Не потому, что здесь без неё нельзя, а потому, что здесь она сильно пригодится. Код удобно разбирать и чинить с помощью модели: скопировали текст ошибки, попросили объяснить — и получили ответ вместо часа поисков. Эта книга такому приёму учит, а «Нейросети для работы» учит ставить задачу так, чтобы ответ был полезным.

Если вы читали **«ИИ-агенты без кода»** — вы узнаете ход мысли. Там задача раскладывалась на шаги. Здесь раскладывается разговор. Разница между агентом и ботом простая: агент работает без человека, бот разговаривает с человеком. Читать ту книгу до этой не обязательно.

Порядковых номеров у книг серии нет намеренно — порядок написания разошёлся с нумерацией, и номера путают. Все ссылки здесь по названиям.

Как устроена каждая глава

Одинаково, чтобы не приходилось искать.

Сначала — что мы делаем и зачем. Потом разбор. В конце — **«что должно получиться»** и **«если не получилось»** с типовыми ошибками. В главах второй части добавляется отдельный разбор ситуации, когда код не запускается: с точным текстом сообщения об ошибке и объяснением, что оно значит. Последняя глава устроена иначе — она подводит итог, а не учит новому.

Через всю книгу проходит **один сквозной пример — бот записи на услугу.** Мы нарисуем его схему во второй главе и соберём в конструкторе в четвёртой. В пятой он научится собирать данные, в шестой — напоминать о визите. В восьмой посмотрим, где он ломается,

в девятой упрёмся с ним в потолок конструктора, а в одиннадцатой и двенадцатой соберём заново на коде. В каталоге он идёт первым — в обеих версиях.

Пример выбран не потому, что запись на услугу — самая частая задача. Потому что в ней есть всё: кнопки, пошаговый опрос, проверка ввода, хранение данных, уведомление владельцу, событие по времени и работа с чужим временем. Кто собрал бота записи, соберёт что угодно.

Если ваша задача другая, ничего страшного: в каждой главе есть второй, короткий пример из другой области. Но следить советую за первым — сквозная задача, которую вы ведёте от схемы на бумаге до бота на сервере, даёт больше, чем десять разрозненных.

С чего начать прямо сейчас

Не с конструктора. Возьмите неделю своих переписок с клиентами и выпишите вопросы, которые повторились больше трёх раз.

Этот список и есть техническое задание на вашего первого бота. Всё остальное в книге — про то, как превратить его в работающую переписку.

Начинаем с устройства: из чего вообще состоит бот.

Часть I. Боты без кода

Глава 1. Из чего состоит бот

Человек, который собирается завести бота, обычно представляет его как маленькую программу — что-то вроде приложения, только внутри мессенджера. Из этого представления растёт весь страх: приложение пишут программисты, значит, и здесь без программиста никак.

Представление неверное, и это хорошая новость.

Бот устроен проще любого приложения. У него нет экрана, который нужно рисовать. Нет меню, которое нужно продумывать. Нет установки, обновлений, версий для разных телефонов. Всё, что у бота есть, — это переписка. Человек пишет или нажимает, бот отвечает. Больше ничего.

Поэтому определение, с которого начинается эта книга, звучит так:

Бот — это набор ответов на события.

Не программа. Не сотрудник. Не воронка. Набор заранее описанных ответов на то, что может произойти в переписке.

Вся первая часть книги — про то, как этот набор составить. Вторая — про то, как записать его на языке, который понимает компьютер, когда конструктора перестаёт хватать. Но набор ответов остаётся тем же самым. Меняется только способ записи.

Эта глава даёт модель, по которой дальше разбирается любой бот: и ваш будущий, и чужой, который вы захотите повторить.

Три типа событий

События, на которые бот отвечает, бывают трёх видов. Их действительно только три, и это не упрощение для новичков — за пределами этих трёх у бота ничего не происходит.

Первое: человек написал сообщение. Текст, картинка, файл, геопозиция, голосовое. Для бота это одно событие с разным содержимым: пришло сообщение, надо на него отреагировать.

Отдельный случай — команды. Это те слова со слешем впереди, вроде /start, которые видно в меню бота. Технически команда — то же сообщение, просто платформа подсвечивает его и складывает в отдельный список, чтобы человеку было удобно. Для бота разница небольшая.

Второе: человек нажал кнопку. Кнопки бывают двух видов. Одни живут под полем ввода и заменяют клавиатуру — нажатие на такую превращается в обычное сообщение с текстом кнопки. Другие живут прямо под сообщением бота и при нажатии передают ему служебный сигнал, а само сообщение при этом можно переписать на месте. Разницу мы разберём в четвёртой главе, когда она начнёт иметь практическое значение. Сейчас достаточно: нажатие — событие.

Третье: наступило время. Ровно в девять утра. Через час после записи. За сутки до визита. Каждый понедельник. Здесь никто ничего не делал — событие наступило само, по часам.

Третий тип — единственный, где бот действует первым. Всё остальное — реакция.

Отсюда простая проверка, которая пригодится вам ещё много раз. Если вы хотите, чтобы бот что-то сделал, спросите себя: **после чего?** Если ответа нет — бот этого не сделает. Не потому, что не умеет, а потому, что ему нечем начать.

«Пусть бот сам напишет клиенту, который давно не приходил» — после чего? После того, как наступила дата на месяц позже последнего визита. Это третий тип, и такое возможно.

«Пусть бот сам поймёт, что клиент недоволен, и предложит скидку» — после чего? После сообщения, в котором есть недовольство. Событие есть, а вот с распознаванием недовольства всё сложнее — к этому вернёмся в главе о подключении нейросети.

Четыре составляющие ответа

Событие произошло. Дальше бот выполняет ответ, и любой ответ раскладывается на четыре части. Эти четыре слова — рабочий инструмент, с которым вы будете жить до конца книги, поэтому запомнить их стоит сразу.

Триггер — что произошло. Пришло сообщение «Записаться». Нажата кнопка «Понедельник». Наступило восемь вечера накануне визита.

Условие — в каком состоянии находится разговор. Одно и то же сообщение в разные моменты означает разное. «Иванов» в начале разговора — это, скорее всего, попытка что-то спросить. «Иванов» после вопроса «Как вас записать?» — это фамилия клиента. Триггер одинаковый, реакция должна быть разной, и различает их именно условие.

Действие — что бот делает внутри. Записывает строку в таблицу. Ставит будильник на завтра. Отправляет вам уведомление о новой заявке. Обращается к нейросети. Ничего из этого человек не видит.

Ответ — что человек видит на экране. Текст. Кнопки. Картинка. Иногда — правка предыдущего сообщения бота вместо нового.

Действие и ответ разделены не для красоты. Это разделение спасает от самой частой ошибки начинающих: бот делает всё правильно, но человек не понимает, что произошло. Заявка ушла в таблицу, уведомление пришло владельцу, а клиент смотрит в экран, где ничего не изменилось, и через минуту нажимает «Записаться» второй раз. Теперь заявок две.

Любое действие должно сопровождаться ответом. Даже коротким. Даже «Готово».

Обратное неверно: ответ без действия — нормальная ситуация. «Мы работаем с десяти до восьми» — просто текст, внутри ничего не происходит.

Состояние разговора

Из четырёх составляющих условие обычно даётся тяжелее всего, потому что за ним стоит понятие, которое звучит технически: состояние разговора.

На деле оно означает вот что: **бот помнит, на каком шаге вы находитесь.**

Больше ничего. Не помнит вас как личность, не помнит прошлый разговор, не делает выводов. Помнит шаг.

Сравнение, которое работает лучше всего, — анкета на бумаге. Администратор задаёт вопросы по списку и отмечает галочкой, до какой строки дошёл. Имя записано. Телефон записан. Стоим на строке «дата». Что бы вы сейчас ни сказали, администратор попробует понять это как дату — потому что галочка стоит здесь.

Бот делает то же самое. Состояние — это галочка на строке.

Отсюда следуют три вещи, которые сэкономят вам много времени.

Первая: **пока состояние не меняется, бот застревает.** Если вы забыли перевести галочку на следующую строку, бот будет задавать один и тот же вопрос сколько угодно раз. Это не поломка, а ровно то, что ему велели.

Вторая: **состояние может потеряться.** Бот перезапустился, конструктор обновился, прошло много времени — галочка стёрлась. Человек оказывается посреди анкеты, а бот не

знает, где он. Что с этим делать, разберём в восьмой главе; пока достаточно знать, что такое случается, и это не ваша вина.

Третья: **у каждого человека галочка своя.** Сто человек в одном боте — сто анкет, каждая на своей строке. Бот их не путает. Это одна из немногих вещей, которую и конструктор, и код делают за вас бесплатно.

Три знакомых бота, разобранных по схеме

Модель проверяется на чужих ботах. Возьмём три, с которыми вы наверняка сталкивались, и разложим каждый на события и составляющие. Заодно вы получите привычку, которая пригодится в третьей части: увидел бота — разобрал.

Бот доставки еды

Событие: человек нажал «Меню». Условие: неважно, из любого места. Действие: подтягивает список блюд из таблицы. Ответ: сообщение с кнопками по числу позиций.

Событие: нажата кнопка с блюдом. Условие: корзина открыта. Действие: добавляет позицию в корзину. Ответ: переписывает то же сообщение, показывая обновлённый состав и сумму.

Событие: нажата «Оформить». Условие: в корзине что-то есть. Действие: сохраняет заказ, отправляет уведомление на кухню. Ответ: «Заказ принят, ждите звонка».

Обратите внимание на условие в последнем случае. Если корзина пуста, а кнопка «Оформить» всё равно нажата, нужен отдельный ответ. Хорошо сделанные боты отвечают «Сначала выберите блюдо». Плохо сделанные — оформляют пустой заказ и присылают его на кухню.

Бот службы поддержки

Событие: человек написал что угодно. Условие: разговор в начале. Действие: нет. Ответ: список тем кнопками.

Событие: выбрана тема. Условие: разговор в начале. Действие: нет. Ответ: готовый текст по теме плюс кнопка «Не помогло».

Событие: нажато «Не помогло». Действие: создаёт обращение, передаёт его человеку из поддержки. Ответ: «Специалист ответит здесь же».

Здесь виден приём, к которому мы вернёмся в главе о том, где боты ломаются: у бота обязан быть выход к живому человеку. Бот, из которого нельзя выйти, злит сильнее, чем его отсутствие.

Бот записи на услугу

Это наш сквозной пример — с ним мы пройдём всю книгу до последней главы. Разберём его подробнее.

Событие: команда «старт». Условие: любое. Действие: нет. Ответ: приветствие и три кнопки — «Записаться», «Мои записи», «Как нас найти».

Событие: нажата «Записаться». Условие: любое. Действие: **ставит галочку на строку «услуга»**. Ответ: список услуг кнопками.

Событие: нажата кнопка с услугой. Условие: галочка на строке «услуга». Действие: запоминает выбор, переводит галочку на строку «дата». Ответ: ближайшие свободные дни кнопками.

Событие: нажата кнопка с днём. Условие: галочка на строке «дата». Действие: запоминает, переводит галочку на «время». Ответ: свободные часы кнопками.

Событие: нажата кнопка с часом. Условие: галочка на «время». Действие: записывает строку в таблицу, отправляет уведомление владельцу, ставит будильник на восемь вечера накануне. Ответ: «Записал: стрижка, четверг, 15:00» и кнопка «Отменить запись».

Событие: наступило восемь вечера накануне. Условие: запись не отменена. Действие: нет. Ответ: напоминание клиенту.

Пересчитайте: шесть ответов на события плюс один по времени. Это весь бот. Ничего сверх этого он не делает, и ничего сверх этого делать не должен.

Если сейчас вам кажется, что вы могли бы описать такой список для своей задачи, — вы правы, могли бы. Именно этим мы займёмся в следующей главе, и это главный навык книги.

Чего бот не умеет

Список ограничений идёт раньше списка возможностей, потому что почти все разочарования в ботах растут из ожиданий, которых бот не обещал.

Понимать намерение вне сценария. Бот отвечает на то, что вы предусмотрели. Человек, который вместо нажатия кнопки пишет «а можно я приду в среду вечером, часов в семь, если у вас там свободно», получит ответ ровно настолько осмысленный, насколько вы этот случай продумали. Нейросеть, подключённая к боту, сдвигает границу, но не убирает: она превратит фразу в дату и час, а решать, что с ними делать, всё равно будете вы. Об этом — глава о подключении модели.

Помнить вечно. Состояние живёт, пока живёт разговор. Данные живут, пока вы их куда-то сложили — в таблицу или в базу. То, что нигде не сохранено, исчезает при первом перезапуске. Это не недостаток конкретного инструмента, а свойство всех.

Отвечать за результат. Бот отправил напоминание — это не значит, что человек его прочитал. Бот принял заявку — это не значит, что клиент придёт. Бот исполняет действия, а не достигает целей.

Работать без вас. Токен отзывают, сервис меняет тарифы, площадка падает, таблица переполняется. Бот требует внимания — немного, но регулярно. Обещания «настроил и забыл» в этой книге не будет.

Заменить человека там, где нужен человек. Бот хорош на повторяющемся: одни и те же вопросы, один и тот же порядок действий, одна и та же форма. Всё, что требует переговоров, оценки ситуации или ответственности за чужие деньги, остаётся вам.

Чем бот отличается от чата с моделью и от агента

Три вещи легко перепутать, а разница между ними определяет, что вам вообще нужно строить.

Чат с моделью — это разговор с нейросетью в её собственном приложении. Инициатива всегда ваша, отвечает модель, каждый ответ рождается заново. Ничего не запускается по расписанию, никто, кроме вас, к этому разговору доступа не имеет. Как ставить задачи модели, разобрано в книге «Нейросети для работы»; здесь этот навык понадобится, но не в первой части.

Агент — записанная последовательность шагов, где один-два шага выполняет модель. Он запускается сам, делает работу и приносит вам результат. Смысл агента в том, чтобы человек в процесс не вмешивался. Об этом книга «ИИ-агенты без кода».

Бот — точка контакта с людьми. Смысл бота в том, что с ним разговаривают. Не вы, а ваши клиенты, читатели, ученики, сотрудники.

Разница практическая: **агент — про работу без человека, бот — про разговор с человеком.** Если ваша задача формулируется как «пусть это делается само, а мне приходит готовое» — вам нужен агент, и эта книга не самая полезная. Если как «пусть люди могут это сделать сами, не отвлекая меня» — вам нужен бот.

Часто нужно и то, и другое. Бот принимает заявки от клиентов, агент раз в неделю разбирает накопившееся и присылает вам сводку. Собираются они отдельно, соединяются через общую таблицу. Такой связкой мы займёмся ближе к концу книги.

Ещё одна пара, которую путают: бот и канал. Канал — это вещание, вы пишете, люди читают. Бот — переписка. У бота есть важное ограничение, которое стоит узнать сейчас, а не после сборки: **бот не может написать человеку первым, пока человек сам к нему не обратился.** Платформа устроена так намеренно, чтобы боты не превращались в источник непрошеных сообщений. Из этого следует, что рассылка возможна только по тем, кто хотя бы раз нажал «старт». Что из этого следует для рассылок и почему массовые сообщения — плохая идея даже когда технически возможны, разберём в шестой главе.

Что должно получиться

К концу этой главы у вас должна появиться привычка, а не знание. Проверить её просто: откройте любого бота, которым вы пользуетесь, и разберите три его шага по схеме — событие, условие, действие, ответ.

Если получилось, дальше будет легко: вся книга — это одна и та же схема, применённая к разным задачам и записанная разными способами.

Заодно попробуйте набросать список событий для своего бота. Пока без порядка и без деталей, просто перечислением: что человек может написать, что нажать, когда бот должен проснуться сам. Этот список станет черновиком схемы в следующей главе.

Если не получилось

«Я не могу понять, где у бота условие». Скорее всего, вы разбираете простого бота, у которого состояний нет вовсе — он отвечает одинаково откуда угодно. Так тоже бывает, и это нормальный бот. Условие появляется там, где начинается пошаговый опрос.

«Я не вижу действия — бот просто пишет текст». Значит, действия нет. Ответ без действия — половина всех шагов в любом боте.

«Мой список событий получился на сорок строк». Почти наверняка вы записали туда варианты одного события. «Клиент выбрал стрижку», «клиент выбрал окрашивание», «клиент выбрал укладку» — это одно событие «нажата кнопка с услугой» с разным содержимым. Схлопните такие строки в одну; правило простое: если реакция бота отличается только подставленным словом, событие одно.

«Я не знаю, чего хочу от бота». Тогда описывать события рано. Возьмите неделю переписок с клиентами и выпишите вопросы, которые повторились больше трёх раз. Бот начинается оттуда, а не с инструмента.

«Мне кажется, моя задача сложнее и в эту схему не влезет». Возможно. Проверить это можно только одним способом — попробовать разложить. Задачи, которые действительно не раскладываются на события, существуют, и в девятой главе мы разберём признаки, по которым видно, что вы упёрлись в потолок. Но упираются в него позже, чем боятся.

Глава 2. Схема разговора

Половина ботов, которые начинают собирать, не доживает до запуска. Причина почти всегда одна и та же, и она не техническая.

Человек открывает конструктор, видит поле для первого сообщения, пишет приветствие. Добавляет кнопку. Кнопка ведёт на второе сообщение — пишет второе. Оттуда нужны три ветки, он делает три. В каждой ветке ещё по паре шагов. Через час на экране паутина из тридцати прямоугольников, в которой уже не видно, куда что ведёт, а бот всё ещё не умеет главного. Человек закрывает вкладку и решает, что это сложнее, чем казалось.

Сложным было не это. Сложным было **придумывать разговор одновременно с его сборкой**. Две работы, каждая из которых требует головы целиком.

Поэтому правило, вокруг которого держится вся книга:

Сначала схема, потом инструмент.

Схема рисуется на бумаге, в заметках, в любом приложении для блок-схем — где угодно, кроме конструктора. Собирается за полчаса, переделывается за пять минут. Тот же бот, наполовину собранный в конструкторе, переделывается за час, и на третий раз вы бросите.

Человек, который расписал, что бот отвечает на каждое действие, соберёт его в любом инструменте. Человек, который не расписал, не соберёт нигде — ни в конструкторе, ни на коде, ни с программистом за деньги.

Эта глава — про то, как расписать. Навык из неё вы будете применять до последней страницы книги: в третьей части каждый из двадцати ботов описан именно такой схемой.

Если вы читали книгу «ИИ-агенты без кода», вы узнаете ход мысли. Там задача раскладывалась на шаги, и от качества раскладки зависело всё остальное. Здесь то же самое, только шаги идут не подряд, а ветвятся.

Что такое схема

Схема разговора — это список всех сообщений бота и всех переходов между ними.

Рисуется двумя элементами:

Прямоугольник — сообщение бота. Внутри пишется текст, который увидит человек, и перечисляются кнопки под ним.

Стрелка — действие человека. На стрелке подписывается, что именно он сделал: нажал такую-то кнопку, написал текст, ничего не сделал в течение суток.

Всё. Никаких других фигур не нужно, никакой нотации учить не надо. Кто пробовал рисовать блок-схемы по правилам, знает, сколько времени уходит на выбор фигуры вместо работы. Здесь выбирать не из чего.

Вот кусок нашего сквозного бота записи на услугу, записанный текстом. Рисовать удобнее, но текстом можно передать в книге, а вы перенесёте на бумагу.

[1] «Здравствуйте! Чем помочь?»

Кнопки: Записаться · Мои записи · Как нас найти

→ «Записаться» ведёт в [2]

→ «Мои записи» ведёт в [10]

→ «Как нас найти» ведёт в [11]

→ любой текст ведёт в [90]

[2] «Выберите услугу»

Кнопки: Стрижка · Окрашивание · Укладка · Назад

→ услуга ведёт в [3]

→ «Назад» ведёт в [1]

→ любой текст ведёт в [90]

[3] «На какой день?»

Кнопки: ближайшие свободные дни · Другая дата · Назад

→ день ведёт в [4]

→ «Другая дата» ведёт в [3a]

→ «Назад» ведёт в [2]

[3a] «Напишите дату числом: 25.08»

Кнопки: Назад к ближайшим дням

→ разобранная дата ведёт в [4]

→ неразобранный текст ведёт в [3a] с пояснением

→ «Назад» ведёт в [3]

[4] «Во сколько?»

Кнопки: свободные часы · Назад

→ час ведёт в [5]

→ «Назад» ведёт в [3]

[5] «Записал: стрижка, четверг, 15:00. Всё верно?»

Кнопки: Да, записать · Изменить

→ «Да» ведёт в [6]

→ «Изменить» ведёт в [2]

[6] «Готово. Напомню за день до визита.»

Кнопки: Отменить запись · В начало

Номера в квадратных скобках нужны, чтобы ссылаться на шаги, не переписывая их текст. В конструкторе они превратятся в названия блоков, в коде — в названия состояний. Пока это просто способ не запутаться.

Шаг [3а] с буквой — ответвление от третьего шага, а не самостоятельный этап. Так удобно помечать боковые ветки: видно, откуда человек пришёл и куда вернётся.

Обратите внимание на шаг [90] в конце первых двух блоков. Про него — отдельный раздел ниже, и он важнее половины остальной схемы.

Главное правило

У каждого сообщения бота должен быть предусмотрен ответ на всё, что человек может нажать или написать.

Правило звучит очевидно ровно до момента, когда вы начнёте проверять по нему свою схему. Тогда выяснится, что необработанных вариантов в ней больше, чем обработанных.

Проверяется каждый прямоугольник по отдельности. Смотрите на сообщение бота и спрашиваете: что человек может сделать прямо сейчас?

Он может нажать любую из показанных кнопок. Для каждой должна быть стрелка — с этим обычно всё в порядке.

Он может написать текст. Любой. «Здравствуйте», «а сколько стоит», «алло», случайную букву, потому что телефон в кармане. Стрелка на этот случай есть?

Он может нажать кнопку под старым сообщением бота, прокрутив переписку вверх. Такое бывает чаще, чем кажется: человек вернулся в чат через два дня, увидел последнее, что помнит, и нажал.

Он может ничего не делать. Открыл, отвлёкся, забыл. Через сутки вспомнил.

Он может прислать картинку, голосовое, файл, стикер.

Он может нажать «старт» посреди разговора.

Полностью закрыть все варианты в первой версии бота не нужно. Нужно **знать, какие вы не закрыли**, и сознательно решить, что с ними будет. Непредусмотренный вариант не исчезает от того, что вы о нём не подумали, — он просто превращается в молчащего бота на глазах у клиента.

Практический минимум, ниже которого опускаться нельзя, — три вещи из следующих трёх разделов: тупики, ветка «не понял», команда «начать сначала».

Тупики

Тупик — это шаг, с которого нет выхода. Человек попал и остался.

Тупики появляются сами собой, никто не рисует их намеренно. Самые частые виды:

Сообщение без кнопок и без обработки текста. «Спасибо за заявку!» — и всё. Что теперь? Написать нельзя, ответа не будет. Нажать нечего. Человек сидит в переписке, где ничего не работает, и делает единственный доступный вывод: бот сломался.

Ветка, которая не возвращается. «Как нас найти» → адрес и карта. Кнопки «В начало» нет, потому что показалось, что и так понятно. Не понятно.

Ссылка вместо кнопки. Ссылка уводит в браузер и обратно не приводит. Как единственный выход со шага она делает шаг тупиком.

Ожидание того, чего человек не даст. Бот просит прислать фотографию, а у человека её нет. Кнопки «пропустить» не предусмотрено. Разговор кончился.

Ошибка в самом конце. Человек прошёл семь шагов, на восьмом что-то не сработало, бот молчит. Формально тупика в схеме нет, а на практике человек в нём стоит.

Ищутся тупики механически, и это лучшая новость этой главы. Пройдите по списку шагов и отметьте каждый, из которого **нет ни одной исходящей стрелки**. Каждый такой шаг — либо законный конец разговора, либо тупик.

Законных концов у бота, вообще говоря, не бывает. Даже финальное «Готово, ждём вас в четверг» должно нести кнопку «В начало» или «Отменить запись». Разговор с ботом не заканчивается — он возвращается в исходную точку.

Второй способ проверки, который ловит то, что не поймал первый: возьмите список шагов и для каждого спросите, **какие стрелки в него входят**. Шаг, в который не входит ни одна, — недостижимый: вы его придумали, но попасть туда невозможно. Такое случается после переделок схемы.

Ветка «не понял»

Человек напишет текст там, где вы ждали нажатия кнопки. Не иногда — регулярно. Люди пишут «да», «хорошо», «а можно завтра», пишут вопросы, здороваются в середине опроса. Кнопки не отменяют привычки разговаривать.

Значит, у бота должен быть ответ на неопознанный текст. Это и есть ветка «не понял» — шаг [90] из схемы выше.

Устроена она просто. Бот получил сообщение, которое не совпало ни с одним ожидаемым вариантом, отвечает объяснением и возвращает человека туда, где тот стоял.

[90] «Не разобрал сообщение. Выберите вариант кнопкой

или напишите /start, чтобы начать сначала.»

Кнопки: те же, что были на текущем шаге

Три требования к этой ветке, каждое из которых стоит того, чтобы его соблюсти.

Она повторяет кнопки текущего шага. Не отправляет в начало — это наказание за попытку заговорить. Человек стоял на выборе даты, написал «а в выходные?» — он должен снова увидеть дни, а не приветствие.

Она не срабатывает бесконечно. Если один и тот же человек попал в «не понял» три раза подряд, дальше повторять бессмысленно — сценарий не сходится с тем, чего он хочет. На третий раз показывается кнопка «Написать человеку» или контакт. Считать повторы умеет и конструктор, и код; это ровно то самое состояние из первой главы, только хранит оно не шаг, а число.

Она не оправдывается. «Извините, я всего лишь бот и не понимаю сложных фраз» — лишние слова. Короткое «Не разобрал» плюс кнопки работает лучше.

Отдельно про формулировку. Не пишите «Такой команды не существует» и «Ошибка ввода». Человек не вводил команду и не ошибался — он просто написал предложение. Виноват сценарий, который этого не предусмотрел, и незачем сообщать человеку, что он что-то сделал не так.

Команда «начать сначала»

Обязательный элемент, без исключений.

У человека должен быть способ выйти из любого места разговора в исходную точку одним действием. Обычно это команда «старт», которую платформа и так показывает в меню бота. Достаточно, чтобы она работала **отовсюду**, а не только с чистого листа.

Проверка простая: зайдите на середину любого опроса и напишите «старт». Если бот воспринял это как ответ на текущий вопрос и записал слово «старт» в графу «имя» — команда не обработана.

Здесь нужна одна тонкость, о которой почти все забывают. «Начать сначала» должно **стирать состояние**, а не просто показывать первое сообщение. Иначе человек видит приветствие, нажимает «Записаться», а бот продолжает с той строки анкеты, на которой стоял. Галочка осталась на месте.

Рядом с «начать сначала» стоит кнопка «Назад» — она не обязательна, но сильно улучшает бота. Разница между ними в том, что «Назад» отматывает один шаг и сохраняет всё остальное, а «начать сначала» сбрасывает всё.

Кнопки против свободного текста

Правило: кнопки везде, где можно.

Обоснование чисто практическое. Каждое место, где человек пишет текст, — это место, где он может написать что угодно. Каждый такой ввод нужно проверять, а на неподходящий отвечать. Кнопка не может быть неправильной: нажать можно только то, что показано.

Свободный текст оправдан в трёх случаях.

Данные, которых вы не знаете заранее. Имя. Телефон. Адрес. Кнопками не покажешь.

Список, который слишком длинный для кнопок. Больше десяти вариантов на экране — уже неудобно, больше двадцати — бессмысленно. Здесь либо ввод с поиском, либо разбивка на два шага: сначала категория кнопками, потом варианты внутри неё.

Вопрос, ответ на который заранее не сузить. «Опишите проблему» в поддержке. Ответ уйдёт человеку, а не боту, и разбирать его будет человек. Такой шаг честнее любых кнопок.

Даже здесь ввод стоит сопровождать кнопкой «Пропустить» или «Назад». Иначе шаг превращается в тупик для того, кто не хочет отвечать.

Отдельно: **дату и время всегда давайте кнопками**. Строчка, которую человек напишет руками, — это «завтра», «14 числа», «в след пн», «в пятницу после обеда». Разбирать такое без нейросети мучительно, а с нейросетью — дорого и всё равно с ошибками. Показать восемь свободных окон кнопками проще для обеих сторон, и заодно вы никогда не получите заявку на день, когда закрыты.

Приём: попросить модель найти дыры

Схема, которую вы нарисовали сами, проверяется вами же — и вы проверяете её по той логике, по которой рисовали. Непокрытые ветки остаются невидимыми ровно потому, что вы о них не подумали дважды.

Здесь помогает нейросеть, и это первое место в книге, где она пригодится. Механика запроса разобрана в книге «Нейросети для работы»; здесь достаточно готовой формулировки.

Скопируйте схему текстом — в том виде, в каком она записана выше, с номерами шагов, — и попросите примерно следующее:

Вот схема разговора телеграм-бота. Проверь её по трём пунктам и ответь списком, без вступления.

Шаги, из которых нет выхода.

Шаги, в которые нельзя попасть.

Ситуации, на которые не предусмотрен ответ: человек написал текст вместо нажатия кнопки, нажал кнопку под старым сообщением, вернулся через сутки, прислал файл или голосовое. По каждому пункту укажи номер шага. Если по пункту замечаний нет, так и напиши.

Работает это хорошо по одной причине: задача формальная. Найти шаг без исходящих стрелок — механическая проверка списка, а не творчество. Здесь модель не выдумывает, а считает.

Чего от неё не стоит ждать: она не знает вашего дела. Совет «добавьте оплату онлайн» может быть уместным, а может быть бессмысленным, потому что у вас оплата на месте. Замечания по структуре берите, замечания по существу отбрасывайте.

Второй полезный вариант того же приёма: попросите модель сыграть за клиента, который ведёт себя неудобно.

Ты клиент, который хочет записаться, но плохо читает и пишет вместо нажатия кнопок. Пройди по этой схеме и покажи, где ты застрянешь.

Ответ обычно неприятный и полезный.

Как схема ложится в инструмент

Схема одинаковая, дальнейшая судьба у неё разная, и полезно понимать это заранее — тогда во второй части книги ничего не придётся переучивать.

В конструкторе прямоугольник становится блоком, стрелка — соединением между блоками. Схема переносится почти буквально: то, что нарисовано на бумаге, вы повторяете мышкой. Именно поэтому перерисовать схему на бумаге дёшево, а перерисовать её в конструкторе — час работы.

В коде прямоугольник становится функцией, которая отправляет сообщение, стрелка — записью «после этого шага перейти в такой-то». Номера шагов из схемы превращаются в названия состояний, и от того, насколько внятно вы их назвали, зависит, разберётесь ли вы в собственном коде через месяц.

Отсюда рекомендация: **давайте шагам осмысленные имена сразу**. Не «шаг 4», а «выбор времени». Номера удобны, пока схема на одном листе, и перестают быть удобными на тридцати шагах.

Упражнение

Возьмите свою задачу и нарисуйте схему. На бумаге, не в конструкторе.

Порядок, который работает быстрее всего:

Сначала главный путь. Человек пришёл, всё сделал правильно, дошёл до конца. Обычно это пять-семь прямоугольников. Не отвлекайтесь на исключения, пока не дорисовали главный путь до конца.

Потом возвраты. К каждому шагу — «Назад» и «В начало».

Потом отказы. На каждом шаге человек может передумать. Куда он попадает?

Потом непонятое. Ветка «не понял» с возвратом на текущий шаг.

Потом события по времени, если они есть. Напоминание, отмена по истечении срока, повторный вопрос через сутки.

Разбор реальной задачи по этому порядку занимает от получаса до двух часов. Это ровно то время, которое вы иначе потратите на переделки в конструкторе, только здесь оно потрачено с результатом.

Что должно получиться

Схема вашего бота на бумаге: все сообщения, все переходы, ветка «не понял», возвраты и события по времени.

Прогоните её по девяти пунктам. Каждый — вопрос, на который отвечают «да» или «нет», без «вроде бы».

Есть ли шаги, из которых нет ни одной стрелки?

Есть ли шаги, в которые не входит ни одна стрелка?

С каждого ли шага можно вернуться назад?

Работает ли «начать сначала» из любой точки и стирает ли оно состояние?

Есть ли ветка «не понял», и возвращает ли она на текущий шаг?

Что происходит, если человек нажал кнопку под старым сообщением?

Что происходит, если человек пропал на сутки посреди опроса?

Есть ли способ дозваться живого человека?

Указано ли для каждого шага, что бот делает внутри — записывает, уведомляет, ставит напоминание?

Схема, прошедшая эти девять пунктов, собирается в конструкторе за вечер. Дальше — третья глава, где мы заводим бота и получаем от него первое сообщение.

Если не получилось

«Схема разрослась до сотни шагов». Скорее всего, вы нарисовали отдельную ветку для каждого варианта выбора: своя цепочка для стрижки, своя для окрашивания, своя для укладки. Это одна цепочка, в которой запомнена выбранная услуга. Правило то же, что в первой главе: если шаги отличаются только подставленным словом, шаг один.

«Не понимаю, где заканчивается один шаг и начинается другой». Шаг — это одно сообщение бота и ожидание ответа. Если бот отправил три сообщения подряд и ничего не ждёт — это один шаг с длинным текстом.

«Кнопок на шаге получилось пятнадцать». Разбейте на два шага: категория, потом вариант. Заодно на экране перестанет быть тесно.

«Не могу решить, кнопки или текст». Спросите, сколько вариантов ответа существует. Конечное число, которое влезает на экран, — кнопки. Бесконечное — текст с проверкой.

«Схема есть, но кажется, что бот получается тупой». Обычно так и есть, и обычно это правильно. Первый бот должен уметь одно дело целиком, а не пять дел наполовину. Всё лишнее из схемы вычёркивается до сборки, а не после.

«Клиент всё равно напишет что-то, чего я не предусмотрел». Напишет. Задача схемы не в том, чтобы предугадать всё, а в том, чтобы непредусмотренное не ломало разговор. Ветка «не понял» и выход к живому человеку закрывают этот случай целиком.

Глава 3. Регистрация бота и первый запуск

Задача этой главы — получить от собственного бота первое сообщение. Не собрать сценарий, не подключить таблицу, а просто написать боту «старт» и увидеть ответ, который вы сами придумали.

Занимает это около получаса, из которых двадцать минут уходит на придумывание имени.

Важно понимать, что в этой главе делается, а что нет. Здесь бот **заводится** — получает имя, адрес и ключ доступа. Ума ему это не добавляет: он умеет ровно одно сообщение. Сценарий из схемы, нарисованной во второй главе, мы будем собирать в четвёртой. Но заводится бот один раз и потом живёт годами, а сценарий внутри него можно переделывать сколько угодно.

Что вообще происходит при регистрации

Бот регистрируется у самой платформы, а не у конструктора. Это первое, что стоит понять, потому что из этого следует всё остальное.

Внутри мессенджера есть служебный бот, через который создаются все остальные боты. Он один на всех. Вы пишете ему, он заводит вам нового бота и выдаёт **токен** — длинную строку из цифр и букв.

Токен — это ключ. Всё, что потом происходит с ботом, происходит через него: конструктор с этим токеном управляет ботом, ваш код с этим токеном управляет ботом. Сам бот живёт на серверах платформы; конструктор и код лишь говорят ему, что отвечать.

Отсюда практическое следствие, которое избавляет от главного страха переезда: **бот не привязан к конструктору.** Надоел сервис, вырос тариф, пришло время своего кода — вы забираете токен и вставляете в другое место. Имя бота, адрес, подписчики остаются. Люди даже не заметят, что внутри что-то поменялось.

Именно поэтому во второй части книги мы не будем заводить нового бота. Возьмём того же самого.

Заводим бота

Порядок действий, без описания того, где что нарисовано на экране.

Находите служебного бота платформы. Он называется BotFather — ищется через обычный поиск по имени. Отличается синей галочкой; ботов с похожими именами много, и все они не те. Проверьте галочку, прежде чем что-то ему писать.

Начинаете разговор обычной командой «старт». В ответ приходит список команд — их там много, но нужна одна.

Даёте команду создания нового бота. Она называется /newbot.

Отвечаете на два вопроса. Первый — как бота зовут. Второй — какой у него адрес.

Дальше про эти два вопроса подробнее, потому что второй меняется тяжело, а первый — легко.

Имя

Имя — то, что человек видит вверху переписки. Русские буквы, пробелы, эмодзи — всё разрешено. Меняется в любой момент, без последствий.

Пишите то, что понятно вашему клиенту: «Запись — салон "Марта"», а не «Marta bot v2».

Адрес

Адрес — то, что начинается с собаки: @marta_zapis_bot. По нему бота находят в поиске и по нему дают ссылку.

Требования: только латинские буквы, цифры и подчёркивание, обязательное окончание на bot или _bot, длина от пяти до тридцати двух символов.

Адрес меняется, но со сложностями: старая ссылка перестаёт работать. Если вы уже раздали её клиентам, напечатали на визитке или поставили в шапку канала — переезд обойдётся дорого. Поэтому думайте над адресом дольше, чем над именем.

Три совета, которые экономят время:

Короче — лучше. Адрес диктуют по телефону.

Без года и без версии. salon_2026_bot через год выглядит заброшенным.

Занятые адреса не освобождаются. Хороший короткий адрес почти наверняка занят, и это нормально: добавьте название города или дела, а не цифру в конец.

Токен

После второго ответа приходит сообщение с токеном. Выглядит он как число, двоеточие и длинная строка символов.

Что с ним сделать прямо сейчас: скопировать и сохранить туда, где вы его найдёте через месяц и куда не залезут посторонние. Менеджер паролей, если он у вас есть. Заметка на устройстве — хуже, но приемлемо.

Чего с ним не делать никогда: не пересылать в переписке, не выкладывать на скриншотах, не вставлять в публичный код, не отправлять тому, кто предлагает «помочь настроить».

Если токен утёк — тот, кто его получил, управляет вашим ботом полностью: читает все переписки, пишет от его имени вашим клиентам, меняет настройки. Не «может причинить неудобство», а именно управляет.

Что делать, если утёк: у служебного бота есть команда перевыпуска токена. Старый мгновенно перестаёт работать, вы получаете новый и подставляете его в конструктор. Бот, подписчики, имя, адрес — всё остаётся. Потеря — несколько минут, пока бот не отвечает.

Перевыпускайте **при малейшем сомнении**. Это бесплатно и занимает минуту. Раздумья о том, точно ли утёк, стоят дороже перевыпуска.

Настройка облика

Всё, что дальше, — тоже через служебного бота, отдельными командами. Ни один пункт не обязателен для работы, но каждый влияет на то, поймёт ли человек, куда он попал.

Описание — текст, который человек видит **до** того, как нажал «старт», в пустой ещё переписке. Единственное место, где можно объяснить, зачем этот бот нужен. Одно-два предложения: «Записываю на стрижку и напоминаю о визите. Салон "Марта", улица Ленина, 5».

Информация о боте — короткая строка в профиле, видна рядом с именем. Тут хватает нескольких слов.

Аватар — картинка. Логотип, если он есть. Если нет — что угодно узнаваемое, кроме безликой иконки робота: их у человека в списке чатов уже десяток, и все на одно лицо.

Меню команд — список, который платформа показывает человеку, когда он набирает слеш. Сюда стоит положить две-три команды, не больше:

start — начать сначала

help — как пользоваться

Меню команд решает задачу, о которой мы говорили во второй главе: у человека всегда должен быть способ выйти из любого места разговора. Команда «старт» в меню — этот способ, и он работает даже тогда, когда все кнопки уехали за пределы экрана.

Про кнопку «Остановить бота» стоит знать заранее. Она есть у платформы, вы её не контролируете, и человек может нажать её в любой момент. После этого бот не может ему написать — ни напоминание, ни ответ. В таблице подписчиков он останется, а сообщения до него не дойдут. Это нормальное поведение платформы, и в главе про рассылки мы разберём, как с этим жить.

Первый запуск

Теперь у вас есть бот, который ничего не умеет. Написать ему можно, ответа не будет — потому что отвечать некому: токен есть, а никакого сценария за ним не стоит.

Дальше два пути, и они расходятся ровно здесь.

Путь первой части книги: взять конструктор, вставить туда токен, собрать сценарий из блоков. Этим займёмся в четвёртой главе.

Путь второй части: написать программу, которая с этим токеном работает. Глава одиннадцатая.

Чтобы закончить эту главу результатом, а не обещанием, сделаем минимальное: **одно сообщение в ответ на «старт»**. Это доступно в любом конструкторе, включая самые простые и бесплатные — те, что настраиваются прямо внутри мессенджера, без регистрации на стороннем сайте.

Порядок одинаковый везде, как бы ни выглядел конкретный сервис:

Вы заходите в конструктор и добавляете своего бота — он просит токен, вы вставляете тот, что сохранили.

Конструктор проверяет токен и показывает имя вашего бота. Если показал — связь есть, самое трудное позади.

Вы создаёте первый блок и привязываете его к команде «старт».

Внутрь блока пишете текст приветствия.

Сохраняете и включаете бота.

Дальше — открываете своего бота в мессенджере и пишете ему «старт».

Что должно получиться

Бот отвечает вашим текстом. Ровно один раз, ровно одним сообщением, на всё остальное молчит.

Этого достаточно. Работает связка, а не сценарий: платформа знает про бота, конструктор управляет ботом через токен, сообщение доходит до человека. Всё остальное в первой части книги — наращивание на этот каркас.

Заодно проверьте две вещи, пока они на виду:

Как бот выглядит для нового человека. Попросите кого-нибудь открыть его и не нажимать «старт» сразу — пусть скажет, понятно ли из описания, что это и зачем. Вам самим это уже не проверить: вы знаете, что там написано.

Работает ли команда «старт» второй раз. Напишите её ещё раз. Бот должен ответить так же. Если он молчит — блок настроен на первое обращение, а не на команду, и это первое, что придётся поправить в четвёртой главе.

Приветствие: короткое отступление

Раз уж текст приветствия у вас уже написан, стоит сказать, каким он должен быть. Это единственное сообщение, которое прочитают все без исключения.

Работающее приветствие отвечает на три вопроса за три строки: **кто это, что умеет, что делать дальше.**

Здравствуйте! Это бот салона «Марта».

Записываю на услуги и напоминаю о визите.

Выберите, что нужно:

Дальше кнопки. Кнопки появятся в следующей главе, но текст уже стоит написать так, будто они есть.

Чего в приветствии быть не должно: истории салона, перечисления всех возможностей, извинений за то, что вы бот, и просьб отнестись с пониманием. Человек пришёл записаться.

Проверка: прочитайте приветствие вслух. Если на середине хочется вздохнуть — оно длинное.

Если не получилось

«Служебный бот не отвечает на команду создания». Проверьте, что пишете именно ему, а не боту-двойнику: их много и все с похожими именами, отличается настоящий синей галочкой. Вторая причина — команда написана с опечаткой или с большой буквы.

«Адрес занят». Придумывайте другой, освободить занятый нельзя. Добавьте город, район, название дела — но не цифру в конец, такие адреса выглядят как второй заход после неудачи.

«Потерял токен». Не страшно: у служебного бота есть команда, показывающая список ваших ботов и их токены. Если и это не помогло — перевыпуск.

«Конструктор пишет, что токен неверный». В девяти случаях из десяти при копировании захвачен пробел в начале или в конце, или строка скопирована не целиком. Скопируйте заново, целиком, и посмотрите на конец строки — она длинная и обрывается незаметно.

«Конструктор пишет, что бот уже используется». Один бот в один момент управляется одним местом. Если вы раньше подключали этого бота к другому конструктору, отключите его там. Эта же ошибка встретится во второй части, когда бот окажется запущен и на ноутбуке, и на сервере одновременно.

«Бот молчит на "старт"». По порядку: включён ли бот в конструкторе (у многих сервисов есть отдельный переключатель), сохранён ли блок, привязан ли он именно к команде «старт». Если всё на месте — напишите боту с другого аккаунта: владельца некоторые конструкторы обслуживают отдельно, и у вас может работать то, чего нет у остальных.

«Ответ приходит, но с задержкой в минуту». Бесплатные тарифы части конструкторов работают с задержкой. Это не поломка, а особенность тарифа. Прежде чем что-то чинить, проверьте, не о ней ли речь.

«Аватар не ставится». Требования к размеру и формату у платформы свои. Проще всего взять квадратную картинку и уменьшить её до пятисот точек по стороне.

«Всё получилось, но бот кажется пустым». Так и есть. У него одно сообщение. Следующая глава — про то, как за один вечер превратить его в бота записи по схеме, которую вы нарисовали.

Глава 4. Конструкторы: выбор и сборка

Бот заведён, схема нарисована. Осталось соединить одно с другим.

Эта глава — самая практическая в первой части. К её концу у вас работает бот записи на услугу: приветствие, меню, выбор услуги, выбор дня, выбор времени, подтверждение, возврат в начало. Всё, что нарисовано во второй главе, кроме сбора данных и напоминаний — они в следующих двух главах.

Времени это занимает вечер. Больше всего его уходит на выбор конструктора, поэтому начнём с того, как выбирать быстро.

Названия сервисов, тарифы и лимиты вынесены в приложение в конце книги. Они меняются каждые несколько месяцев, и держать их в тексте главы бессмысленно. Здесь — типы, критерии и порядок работы, которые не меняются.

Три типа конструкторов

Различать их полезно до того, как начнёте сравнивать конкретные сервисы, — иначе вы сравниваете несравнимое.

Визуальные. Бот собирается из блоков на большом холсте, блоки соединяются стрелками. Выглядит ровно как схема из второй главы, только мышкой. Именно поэтому перенос схемы на бумаге в такой конструктор занимает минуты: вы уже знаете, что рисовать.

Плюс — видно всю логику целиком. Минус — на пятидесяти блоках холст превращается в паутину, и работать с ним тяжело.

Это основной тип, и первого бота проще всего собрать здесь.

Табличные. Сценарий описывается списком: шаг, что бот говорит, куда ведут ответы. Похоже на заполнение таблицы.

Плюс — не разъезжается на больших сценариях, легко искать нужный шаг. Минус — не видно картины целиком, и ошибку в переходах глазами не поймать.

Встроенные в платформы рассылок. Бот тут не главное: сервис заточен под работу с базой подписчиков, а сценарий — приложение к ней. Логика обычно проще, зато развитый инструмент для сообщений по базе.

Годится, если ваша задача — в основном рассылка, а бот нужен для приёма подписки.

Для сквозной задачи этой книги — записи на услугу — берём **визуальный**.

Пять критериев выбора

Сравнивать сервисы можно бесконечно, потому что у каждого свой набор возможностей и все они по-своему хороши. Чтобы не утонуть, проверьте пять вещей и на этом остановитесь.

Первое: оплата. Проходит ли карта, нужен ли статус ИП или юрлица для оплаты, что будет с ботом при просрочке — выключится сразу или проработает несколько дней. Последнее выясняется в оферте, а не на странице тарифов.

Второе: нужные подключения. Для бота записи это таблица и уведомления вам. Для магазина — приём платежей. Для бота с нейросетью — возможность обращаться к модели. Составьте список того, что нужно **вашей** задаче, и проверьте по нему. Не наоборот: возможности, которых у вас в списке нет, не должны влиять на выбор вообще.

Конец ознакомительного фрагмента.