Практичний посібник зі створення Telegram-бота

Від карти діалогу й безпечної реєстрації до тестування, передачі доступів і підтримки

У ЦЬОМУ ГАЙДІ

Telegram-бот — це не просто набір кнопок у чаті, а керований процес: користувач виконує дію, бот обробляє її за правилами та передає результат менеджеру, CRM, каталогу або іншій системі. Щоб запуск не перетворився на нескінченне виправлення повідомлень і втрачені заявки, спочатку спроєктуйте маршрут користувача, а потім обирайте спосіб реалізації.

---

01 **1. Визначте результат, до якого бот веде користувача**

02 **2. Побудуйте карту переходів до написання текстів**

03 **3. Оберіть реалізацію та безпечно зареєструйте бота**

04 **4. Налаштуйте обробку подій і передачу даних**

05 **5. Прийміть роботу перед запуском і передайте її на підтримку**

Q **Часті питання**

01 · 1. ВИЗНАЧТЕ РЕЗУЛЬТАТ, ДО ЯКОГО БОТ ВЕДЕ КОРИСТУВАЧА

Робота починається з конкретної дії, яку людина має виконати в чаті. Якщо не визначити результат до створення меню, бот перетвориться на довідник із кнопками, що не приводить ані до заявки, ані до запису, ані до потрібної сторінки.

Сформулюйте один стартовий запит користувача: що саме він хоче отримати після відкриття бота. Результатом може бути вибір послуги, передавання контакту, опис запиту, перехід до товару, створення заявки або передавання звернення менеджеру.

Відокремте інтерфейс Telegram від правил бізнесу. Повідомлення, кнопки й команди лише показують користувачу доступні дії. Правила перевірки даних, зберігання вибору, створення заявки та передавання інформації мають бути визначені окремо.

Якщо бот веде до каталогу чи оформлення замовлення, розподіліть процес між ботом і сайтом. Не дублюйте в чаті ті самі дії, які вже зрозуміло реалізовані в інтернет-магазині.

01 **Сформулюйте цільову дію**
Запишіть, чим має завершуватися основний маршрут: заявкою, записом, вибором товару, переходом на сайт або передаванням менеджеру. Для кожного маршруту має бути одна зрозуміла умова завершення.

02 **Опишіть стартове повідомлення**
Зафіксуйте, що людина бачить після запуску, який вибір має зробити та які команди або кнопки доступні одразу.

03 **Визначте дані для кожного результату**
Позначте, чи потрібні контакт, вибір послуги, опис запиту або інші відомості. Одразу вкажіть, де вони зберігаються та хто їх отримує.

04 **Призначте момент передачі людині або системі**
Встановіть, на якому кроці бот завершує свою частину роботи: створює запис у системі, надсилає дані менеджеру або відкриває зовнішній ресурс.

> ДЕ ВТРАЧАЮТЬ
> Не збирайте персональні дані лише тому, що їх зручно запитати в чаті. Кожне поле має бути пов'язане з конкретною дією та відповідальним отримувачем.

ЧЕКЛІСТ РОЗДІЛУ

- [ ] Для основного маршруту зафіксовано конкретний результат.
  Без цього кнопки й повідомлення не утворюють керованого процесу.

- [ ] Для кожного поля даних визначено місце зберігання та отримувача.
  Без цього заявки можуть залишатися в чаті без обробки.
- [ ] Визначено межу між роботою бота, менеджера та підключеної системи.
  Без меж відповідальності звернення губляться між учасниками процесу.

Карта сценарію показує не красиві формулювання, а рішення, які бот має прийняти після кожної дії користувача. Вона дозволяє знайти тупики, дублікати та неописані помилки до того, як логіку почнуть збирати в конструкторі або коді.

Сценарій діалогу — це послідовність переходів між подіями. Для кожної кнопки, команди, введеного тексту або зовнішньої відповіді треба знати наступний екран, дію або повідомлення.

Не проєктуйте лише ідеальний маршрут. Користувач може написати довільний текст замість натискання кнопки, залишити поле порожнім, повернутися назад або повторно обрати вже відкритий пункт. Усі ці дії потребують окремої реакції.

Зручний робочий артефакт — журнал перевірки сценарію. Він дає команді одне місце, де видно стан діалогу, очікувану дію, результат, дані та відповідального за наступний крок.

01 **Намалюйте стартову точку**
Почніть із запуску бота: зафіксуйте стартове повідомлення, меню та доступні команди. Позначте стан діалогу, у якому опиняється користувач.

02 **Розпишіть кожну кнопку як перехід**
Навпроти кожної кнопки вкажіть не її назву, а результат натискання: наступне питання, нове меню, зовнішню дію, створення заявки або повернення назад.

03 **Додайте очікувані дані**
Там, де бот запитує інформацію, вкажіть формат відповіді, дію після отримання та поведінку при незаповненому або некоректному значенні.

04 **Опишіть неочікувані дії**
Для кожного стану додайте реакцію на довільний текст, повторний вибір і повернення до попереднього пункту. Бот має пояснити доступний крок або повернути до меню, а не обривати діалог.

05 **Зафіксуйте фінальні стани**
Для кожного маршруту позначте, яке повідомлення бачить користувач після завершення та що в цей момент отримує менеджер або зовнішня система.

> ДЕ ВТРАЧАЮТЬ
> Кнопка з нечітким призначенням, наприклад «Опис послуги», створює невизначеність для команди й користувача. До початку реалізації має бути відомо, що саме вона відкриває та яка дія доступна далі.

ЧЕКЛІСТ РОЗДІЛУ

- [ ] Для кожної кнопки визначено наступний екран або зовнішню дію.
  Без цього під час реалізації з'являються суперечливі переходи.
- [ ] Описано реакцію на довільний текст, порожнє поле та повернення назад.
  Без цього бот працює лише за ідеальним сценарієм.
- [ ] Для кожної гілки зазначено фінальний стан і відповідального.
  Без цього неможливо перевірити, чи маршрут справді завершується результатом.
- [ ] Створено журнал перевірки з полями: стан, дія користувача, очікуваний результат, дані, відповідальний і статус перевірки.
  Без спільного журналу команда втрачає контроль над змінами сценарію.

03 · 3. ОБЕРІТЬ РЕАЛІЗАЦІЮ ТА БЕЗПЕЧНО ЗАРЕЄСТРУЙТЕ БОТА

Платформа не компенсує відсутню логіку, але правильний спосіб реалізації зменшує складність підтримки. Вибір роблять за сценарієм, даними та правилами обробки, а не за популярністю конкретного сервісу.

Конструктор доречний для меню, готових відповідей і послідовних форм. Власна серверна частина потрібна, коли є нестандартні правила, ролі користувачів, власне сховище даних або складні переходи. Інтеграція потрібна, коли бот читає або записує дані в CRM, каталог, систему запису чи інший бекенд.

Реєстрацію виконуйте лише через офіційний акаунт BotFather у Telegram. Він створює бота, задає назву та username, а також видає токен доступу для підключення сервера або конструктора.

Токен дає право працювати від імені бота через Telegram API. Його не можна вставляти у фронтенд, публікувати в репозиторії або надсилати у відкритих чатах.

01 **Зіставте сценарій із трьома способами реалізації**
Оберіть конструктор для типового меню й лінійних форм. Плануйте власний код, якщо логіка залежить від ролей, нестандартних правил або власного зберігання даних. Підключайте інтеграцію, коли потрібен обмін із наявною системою.

02 **Перевірте межі обраного рішення**
До старту з'ясуйте, чи можна експортувати сценарій і дані, де зберігатиметься код або налаштування та хто підтримуватиме рішення після запуску.

03 **Створіть бота через BotFather**
Відкрийте офіційний чат BotFather, надішліть команду /newbot і виконайте підказки: задайте назву та зрозумілий username, що відповідає продукту.

04 **Збережіть токен у контрольованому місці**
Передайте токен лише серверу або конструктору, який оброблятиме події. Визначте власника доступу та порядок оновлення токена.

05 **Налаштуйте базове представлення**
Через BotFather додайте опис, аватар, команди та інші доступні параметри, щоб користувач розумів призначення бота ще до першої дії.

> ДЕ ВТРАЧАЮТЬ
> Не залишайте токен або доступ до акаунта бота в особистому акаунті виконавця. Втрата контролю над цими доступами блокує зміну логіки та підтримку.

ЧЕКЛІСТ РОЗДІЛУ

- [ ] Спосіб реалізації відповідає складності правил і даних у сценарії.
  Без цього простий конструктор може стати обмеженням для критичної логіки.
- [ ] Перевірено, де зберігаються сценарії, дані, код і доступи.
  Без цього бізнес залежатиме від одного сервісу або виконавця.
- [ ] Токен не розміщено у фронтенді, репозиторії чи відкритому чаті.
  Без цього стороння людина може отримати доступ до бота.

04 · 4. НАЛАШТУЙТЕ ОБРОБКУ ПОДІЙ І ПЕРЕДАЧУ ДАНИХ

Серверний бот має обробляти події, а не намагатися вгадати намір користувача з видимого тексту. Чітке розділення команд, натискань кнопок, станів діалогу та інтеграцій робить сценарій передбачуваним.

Telegram передає нові повідомлення й натискання кнопок як події. Логіка визначає їхній тип, читає команду або callback, перевіряє поточний стан діалогу, за потреби зберігає дані та повертає наступну відповідь.

Стан діалогу потрібно зберігати окремо від історії повідомлень. Якщо користувач обрав розділ каталогу або почав форму, сервер має пам'ятати цей вибір під час наступної події.

Для інтеграцій заздалегідь визначте, які дані передаються, у який момент це відбувається, де менеджер бачить звернення та що бот повідомляє користувачу, якщо відповідь від іншої системи не надійшла.

01 **Розділіть типи вхідних подій**
Створіть окремі гілки для команд, звичайного тексту та натискань кнопок. Не обробляйте всі повідомлення однаково, інакше дія, прив'язана до кнопки, буде втрачена.

02 **Прив'яжіть callback до конкретної дії**
Для /start створіть новий діалог і покажіть меню. Для callback кнопки виконуйте закладену дію, а не визначайте її за видимим текстом кнопки.

03 **Зберігайте стан діалогу**
Після важливого вибору записуйте поточний стан у сховище, доступне серверу. Це дає змогу коректно продовжити сценарій після наступної події.

04 **Обробіть некоректні та невідомі дії**
Додайте маршрут, який пояснює доступні дії або повертає користувача до меню. Некоректне введення не має руйнувати поточний діалог.

05 **Опишіть помилки обміну з іншими системами**
Для кожної інтеграції визначте, де переглядати помилку, хто її опрацьовує та яке повідомлення отримує користувач, якщо передавання даних не відбулося.

> ДЕ ВТРАЧАЮТЬ
> Не зберігайте контекст діалогу лише в тексті листування. Після наступної події бот може не зрозуміти, який вибір уже зробила людина, і надіслати відповідь з іншої гілки.

ЧЕКЛІСТ РОЗДІЛУ

- [ ] Команди, текстові повідомлення та callback кнопок мають окрему обробку.
  Без цього бот плутає різні типи дій.
- [ ] Стан діалогу зберігається в окремому доступному серверу сховищі.
  Без цього неможливо надійно продовжувати багатокрокові маршрути.
- [ ] Для кожної інтеграції визначено дані, момент передавання, місце перевірки та відповідального.
  Без цього помилки обміну виявляються лише після втрати звернення.
- [ ] Є зрозуміла відповідь на невідоме повідомлення або некоректну дію.
  Без цього користувач отримує обірваний діалог замість наступного кроку.

05 · 5. ПРИЙМІТЬ РОБОТУ ПЕРЕД ЗАПУСКОМ І ПЕРЕДАЙТЕ ЇЇ НА ПІДТРИМКУ

Запускати бота можна лише після перевірки всіх переходів, а не після успішного проходження одного маршруту. Після запуску логіка, токени, підключені системи та відповідальність мають бути передані бізнесу в зрозумілому вигляді.

Тестуйте бота в окремому тестовому доступі, не змішуючи перевірку з реальними зверненнями. Пройдіть кожну гілку від старту до завершення, включно з повторними натисканнями, поверненням назад, довільним текстом і незаповненими полями.

Для ботів з інтеграціями окремо перевіряйте весь ланцюг: бот передає дані, вони з'являються у потрібному місці, менеджер отримує звернення, а помилки обміну можна знайти й опрацювати.

Підтримка можлива, коли карта переходів, список доступів, опис інтеграцій, місце зберігання коду або налаштувань та контакт відповідального передані разом. Логіка не повинна залишатися лише в редакторі конструктора або в голові виконавця.

01 **Складіть перелік тестових маршрутів**
Перенесіть усі гілки з карти сценарію в журнал перевірки. Для кожної зазначте стартову дію, очікуваний результат і статус проходження.

02 **Перевірте основні маршрути**
Запустіть бота, пройдіть кожен сценарій до завершення та звірте повідомлення, кнопки й результати з картою переходів.

03 **Перевірте відхилення від сценарію**
Натисніть усі кнопки повторно, поверніться назад, введіть довільний текст замість очікуваної дії, залиште поле порожнім і змініть відповідь посеред маршруту.

04 **Прийміть інтеграції**
Перевірте, що звернення передається в потрібну систему, дані з'являються в потрібному місці, а менеджер бачить результат. Зафіксуйте порядок дій при помилці.

05 **Передайте пакет підтримки**
Передайте бізнесу карту сценарію, список доступів із власниками, опис інтеграцій, місце зберігання коду або налаштувань, порядок оновлення токена та контакт відповідального.

> ДЕ ВТРАЧАЮТЬ
> Не тестуйте лише правильний маршрут. Найбільше часу після запуску забирають помилки в поверненні назад, повторних виборах, некоректних введеннях і передаванні даних у зовнішні системи.

ЧЕКЛІСТ РОЗДІЛУ

- [ ] Усі гілки сценарію мають статус перевірки в журналі тестування.
  Без цього запуск відбувається на припущеннях, а не на результатах перевірки.
- [ ] Перевірено неправильний текст, порожні поля, повторні натискання та повернення назад.
  Без цього реальна поведінка користувачів швидко виявить необроблені помилки.
- [ ] Інтеграції перевірено від повідомлення користувача до появи даних у потрібній системі.
  Без цього бот може повідомляти про успіх, хоча заявка не надійшла.
- [ ] Бізнес отримав карту сценарію, доступи, опис інтеграцій і відповідальних за підтримку.
  Без цього будь-яка зміна залежатиме від попереднього виконавця.

ЧАСТІ ПИТАННЯ

Коли достатньо конструктора Telegram-ботів?

Конструктор підходить для меню, готових відповідей і послідовних форм. Якщо сценарій складається з кнопок, повідомлень і простого збору заявок, з нього можна почати.

Коли потрібна власна серверна частина?

Вона потрібна, коли бот має працювати за нестандартними правилами, враховувати ролі, зберігати дані у власній базі або виконувати складні переходи. Також серверна логіка потрібна, коли правила не вкладаються в механізми конструктора.

Як безпечно працювати з токеном бота?

Токен отримують через офіційний BotFather після створення бота та використовують для підключення сервера або конструктора. Його не публікують у фронтенді, репозиторіях і відкритих чатах; власником доступу має бути бізнес.

Що обов'язково перевірити перед запуском?

Потрібно пройти всі гілки від старту до завершення, натиснути кожну кнопку, перевірити повернення назад, повторний вибір, довільний текст і незаповнені поля. Якщо є інтеграція, перевіряють також появу даних у потрібній системі та маршрут помилки.

Які матеріали потрібні для підтримки після запуску?

Потрібні карта сценарію, список доступів і їхніх власників, опис інтеграцій, місце зберігання коду або налаштувань, порядок оновлення токена та контакт відповідального. Це дає змогу змінювати бот без залежності від одного виконавця.

ЩО ДАЛІ

Обговорити з Apricode

apri-code.com/kontakty/