PRAGUE MEDIA SCHOOL

*Учебное занятие*

# Агенты

## О чём это занятие

Что меняется, когда большая языковая модель получает инструменты, доступы и право действовать в цикле. Как устроен агент, как им управлять и где для журналиста проходит граница безопасной автономии.

## Задачи для агента

Начинать работу лучше с простой и безопасной рутины. Можно поручить агенту навести порядок в папке, сделать таблицу расходов по фотографиям чеков, регулярно проверять обновления на нужных сайтах, переносить данные в таблицу или готовить однотипные документы по заданному образцу.

Агент не думает и не делает за вас — он берёт на себя последовательность технических действий: открыть, прочитать, скопировать, сверить, разложить, переименовать, оформить, сохранить версию.

Курс **AI-Journalist** · Раздел **Агенты** · Версия **май 2026**

Prague School media

www.pragueschool.media

СОДЕРЖАНИЕ

# Учебное занятие «Агенты»



дополнительно

# Цели занятия

После этого занятия вы сможете объяснить, чем агент отличается от чат-бота, разобрать любую агентскую конфигурацию по слоям, выбрать безопасный режим работы и сформулировать правила, по которым агент будет действовать в вашей редакции.

**ЧЕМУ ВЫ НАУЧИТЕСЬ**

- Различать части *тела* агента: LLM + инструменты + доступы + автономный цикл
- Контролировать доступы агента к данным — разрешать, запрещать и подтверждать решение
- Создавать память (memory), worklog и инструкции (skills)
- Читать и писать skill в формате .md и отличать low-level skill от high-level
- Подключать к агенту коннекторы и понимать, где нужен computer use
- Оценивать уровень риска и контролировать свои автоматизации

# Что делает агент на вашем ПК

Агенты автоматизируют рабочую «возню»: открыть, найти, сравнить, сверстать, сохранить версию и подготовить следующий шаг.

Например, это занятие сначала было написано человеком в обычном Google Doc, а затем агент сверстал его по нашим дизайн-гайдлайнам: разложил текст по страницам, оформил заголовки, блоки, подписи и структуру, применил наши цвета и шрифты.

## Установка

Два приложения для работы агентов. Если у вас есть подписка на Claude или ChatGPT — можете установить их и настраивать своего агента.

**Cowork** *Anthropic* — Сайт · Скачать

**Codex** *OpenAI* — Сайт · Скачать · Docs

# Агентность модели

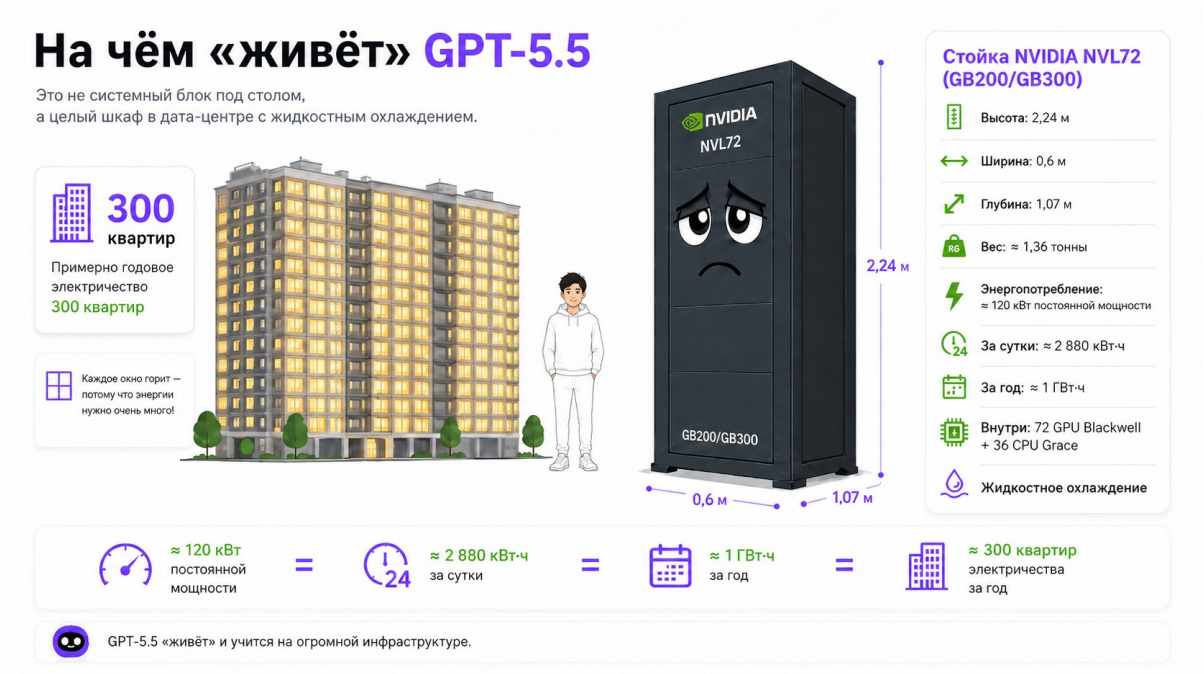
*Возможность действовать.*

## LLM — мозг без рук

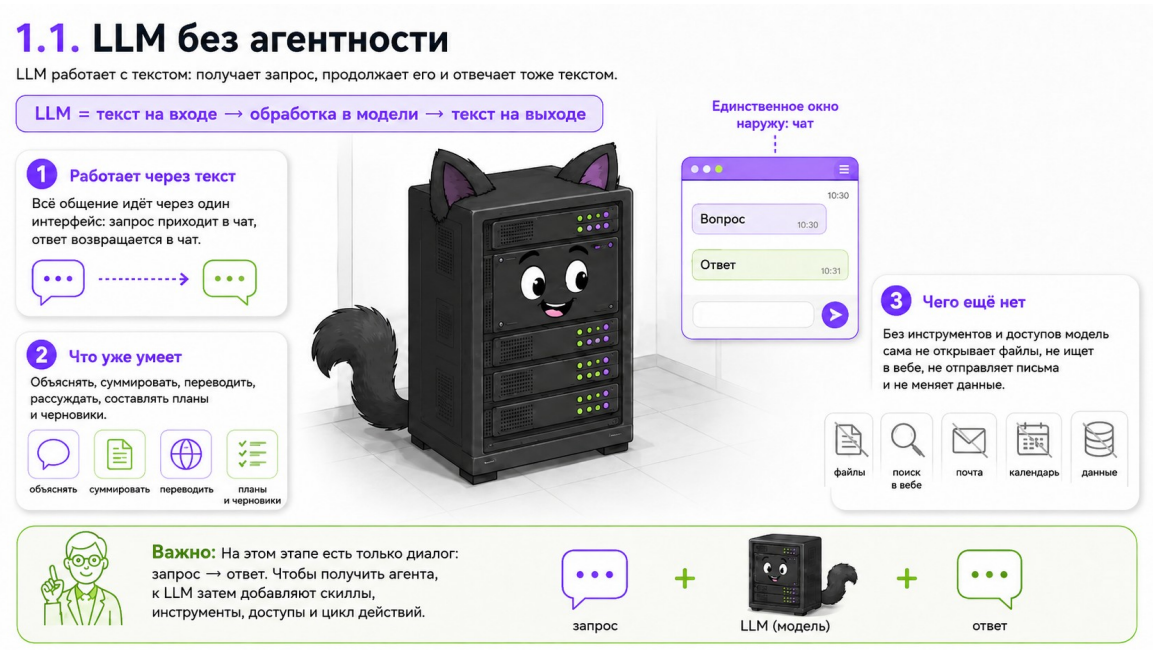
Большая языковая модель (LLM, Large Language Model) — это очень мощный предсказатель следующего слова, обученный на огромном объёме текста. Её суперспособность одна: получив текст на входе, выдать правдоподобное продолжение.

Модели, с которыми мы работаем на этом курсе — ChatGPT, Claude, Gemini, — настолько большие, что их нельзя запустить на собственном компьютере. Обученная модель работает на огромных серверах. В ней — практически всё, что произвела письменная культура человечества.

*Современные модели — например Claude Opus и GPT-5.5 — не помещаются на обычный компьютер: они живут в шкафу и могут одновременно обрабатывать до 50 000 запросов*




*На чём «живёт» GPT-5.5 — масштаб одной рабочей системы*




*LLM без агентности — только язык, без рук и доступов*

# Что добавляет «агентность»

Агент — это та же LLM, но которой дали три вещи:

> **Tool calling** — возможность не только говорить, но и действовать: открыть файл, отправить письмо, выполнить поиск, запустить код.
>
> **Доступы** — ключи к вашим приложениям и данным: почте, дискам, календарю, базам данных.
>
> **Право на цикл (автономию)** — разрешение действовать в несколько шагов самостоятельно: попробовать → посмотреть результат → исправить → продолжить, пока задача не решена.

Агент = LLM + инструменты + доступы + автономный цикл

# 1.3. Что добавляет tool calling

Tool calling — это механизм, при котором модель может вызывать внешние инструменты: передавать им параметры, получать результат и продолжать работу.

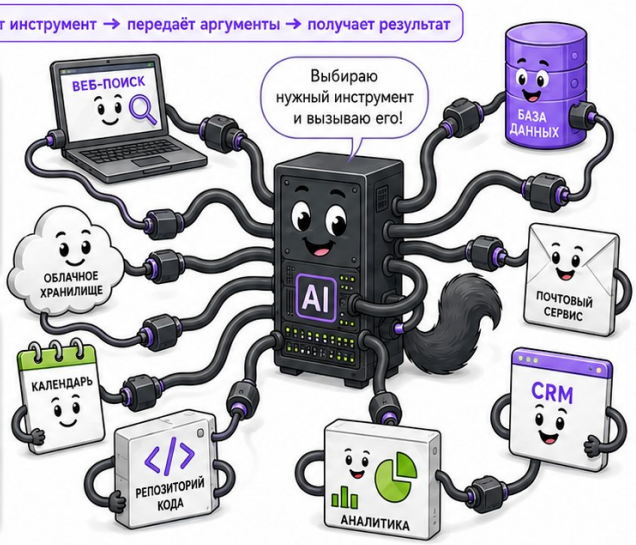
**Tool calling = модель выбирает инструмент → передаёт аргументы → получает результат**

### 1 Вызов по описанию

Модель видит, какие инструменты доступны, что они делают и какие аргументы принимают.

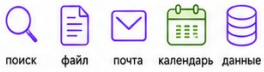
### 2 Передача аргументов

При вызове инструмента модель отправляет структурированные параметры: запрос, дату, текст письма, путь к файлу и т.п.


### 3 Результат возвращается модели

Инструмент выполняет действие или возвращает данные, а модель использует результат в следующем шаге или ответе.




### 4 Один интерфейс — много действий

Через tool calling модель может искать, читать, отправлять, считать и обновлять данные, не ограничиваясь только текстом.

**Аналогия:**

Без tool calling модель только говорит.
С tool calling она может не только отвечать, но и вызывать внешние инструменты, получать результат и продолжать работу.


Tool calling даёт модели способ обращаться к внешним функциям, но права доступа и допустимые действия всё равно нужно контролировать.

# Разрешения и периметр

*«Ключи и красные линии». Тренд — не «спрашивать на каждый чих», а настраиваемая автономия.*

В процессе установки вы создадите агенту его личную папку — где он сможет сохранять и изменять файлы. А разрешения на вызов инструментов, чтение других папок, логины в облачные сервисы от вашего имени, запуск программ и библиотек агент будет просить у вас отдельно.

## Зоны автономии

Разрешения агенту лучше выдавать не по сервисам, а по типу действия. Чтение/просмотр считается безопасным действием.

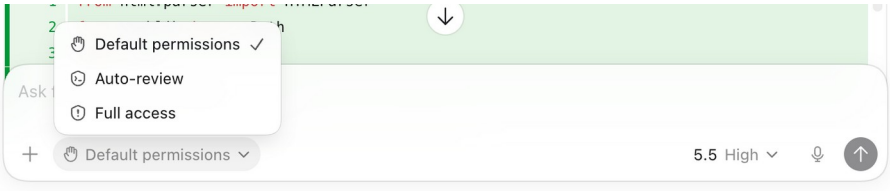
| | |
|---|---|
| 🟢 **Можно** | безопасный рабочий периметр — конкретная папка, которую вы создаёте для агента; в ней он может читать и записывать файлы, сравнивать и сохранять версии, создавать черновики, промежуточные версии, таблицы, отчёты и worklog. |
| 🟡 **С вашего разрешения** | запуск программ, установка библиотек, скачивание файлов. |
| 🔴 **Запрещено** | редактировать или удалять существующие документы на компьютере; совершать операции отправки, изменять настройки. |

ВАЖНО

**Итого**

Агент может подготовить новую версию, черновик или отчёт в своей папке, но не должен сам менять оригинал, отправлять сообщения, публиковать, удалять или расшаривать документы.

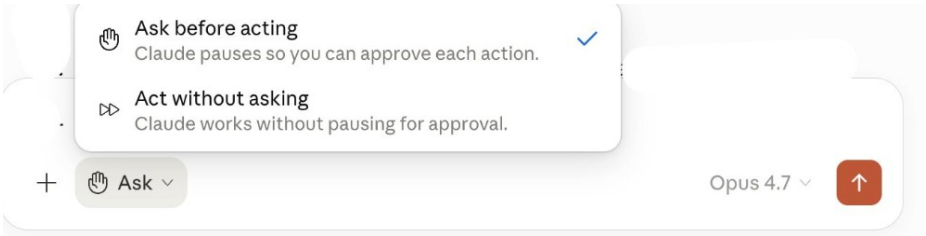
# Codex. Разрешения




В Codex есть три режима, переключаются прямо из чата:

- **Default permissions** — безопасный старт: агент свободно читает и анализирует, но перед записью, переименованием, публикацией или отправкой обязательно спрашивает.
- **Auto-review** — агент сам выполняет шаги, но финальные действия размечает как review: вы видите изменения и принимаете или отклоняете их одним кликом.
- **Full access** — полная свобода: агент действует без подтверждений. Имеет смысл только в изолированной песочнице или для проверенных рутин.

# Claude Cowork. Разрешения




В Cowork два режима, выбираются по кнопке внизу окна:

- **Ask before acting** — Claude останавливается перед каждым действием и ждёт подтверждения. Это режим по умолчанию и самый безопасный.
- **Act without asking** — Claude работает без пауз: проходит цикл «план → действие → проверка → следующий шаг» сам, не дожидаясь одобрения. Включайте, когда уверены в скилле и в периметре рабочей папки.

# Пример: default permissions в работе

*Что именно делает агент в самом безопасном режиме, где останавливается и где спрашивает разрешения. Зелёный лейбл — мой ввод, фиолетовый — ответ агента.*

**ЗАДАЧА**

### Рассортировать файлы в папке

На компьютере есть папка с видео разной продолжительности. Хочу отсортировать файлы по длине: от коротких к длинным, а затем переименовать так, чтобы длительность была видна прямо в названии.

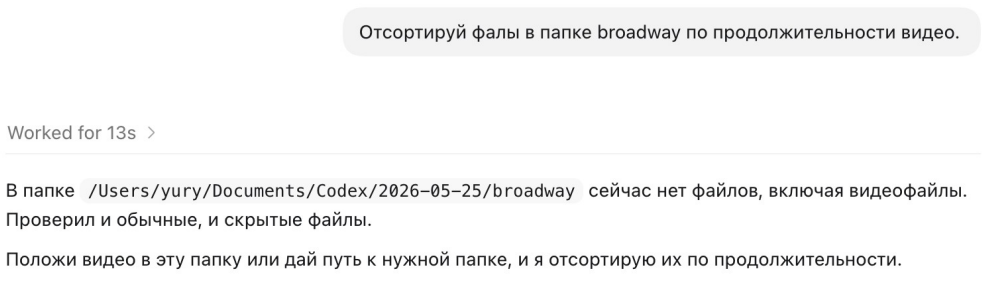
## 1. Первая попытка — без пути к папке

**ПРОМПТ**

Отсортируй файлы в папке по продолжительности — от коротких к длинным.

**ОТВЕТ**

Я не вижу никакой папки. Агент сидит в своей собственной директории и не имеет доступа к остальной файловой системе — нужно явно указать путь.



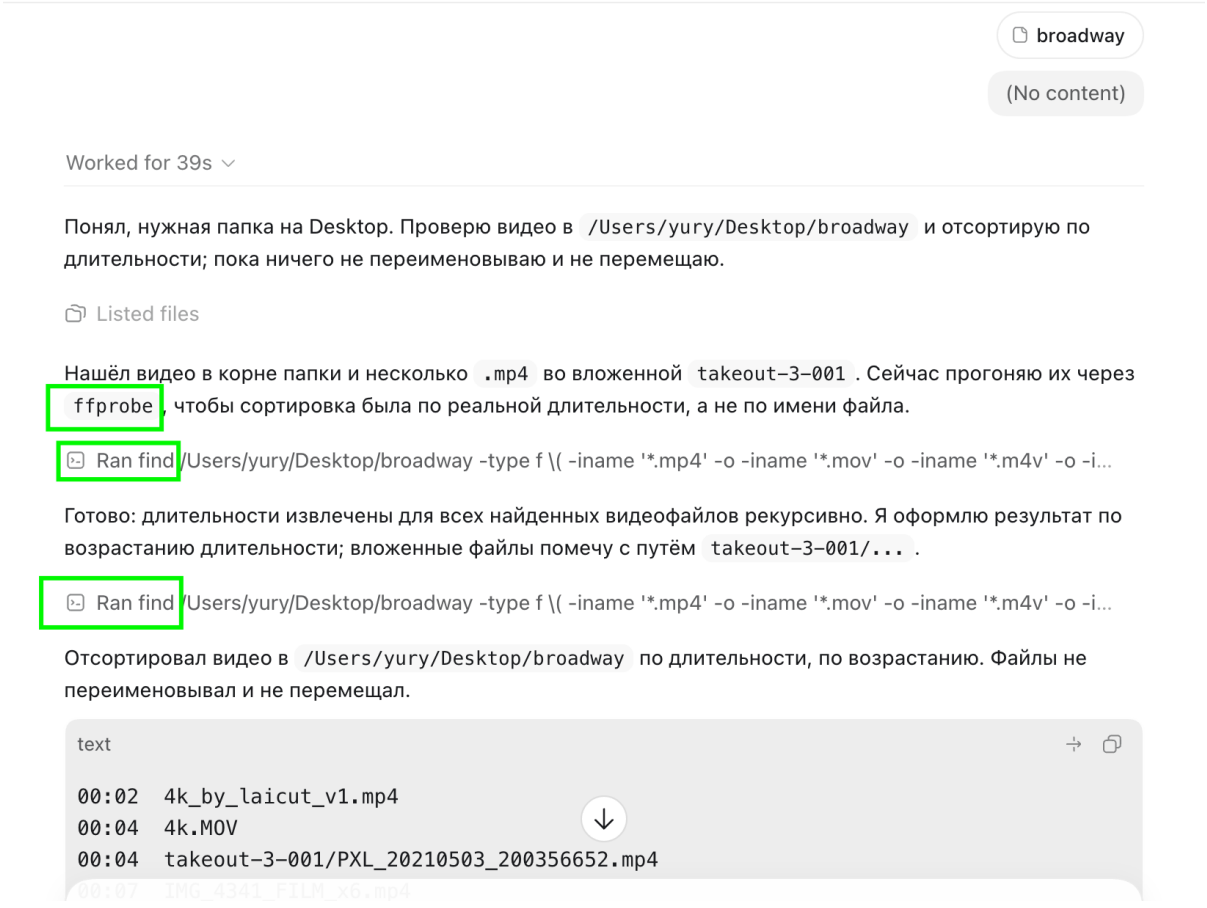

*Первый круг «промпт — ответ»: агент не видит папку, потому что сидит в своей собственной*

## 2. Даю путь — это и есть разрешение

Путь к папке = «разрешение» зайти в неё и прочесть содержимое. Для чтения метаданных видео агент применит специальную библиотеку — ffprobe, — она читает не только названия, но и длительность из заголовков медиафайла.

| ПРОМПТ | Путь к папке: /Users/yury/Desktop/broadway |
|---|---|

| ОТВЕТ | Запускаю инструмент для чтения и анализа (ffprobe) → сортирую как попросили в промпте 1. |
|---|---|



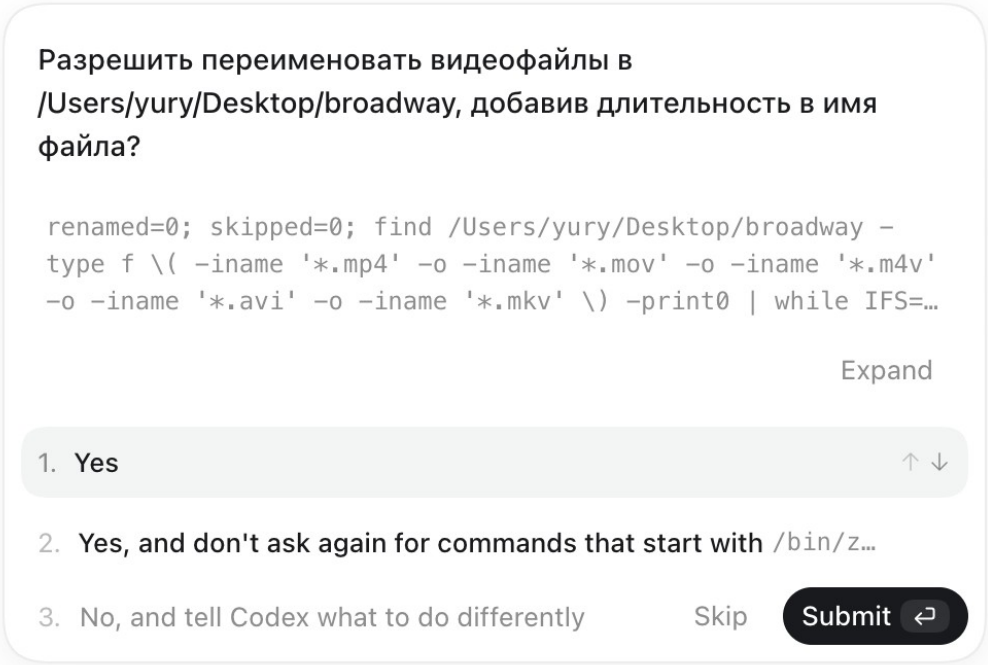

*ОТВЕТ: «Запускаю инструмент для чтения и анализа → сортирую»*

## 3. Хочу переименовать — агент останавливается

Несмотря на эксплицитную просьбу, агент не сможет произвести команду записи, пока я не выберу один из трёх вариантов продолжения:

| | |
|---|---|
| ПРОМПТ | Переименуй файлы так, чтобы в названиях учитывалась продолжительность. |

| | |
|---|---|
| ОТВЕТ | Это необратимое изменение. Подтвердите, что я могу переименовать файлы. Предлагаю формат: Lesson2 [13m53s].mp4. Варианты ответа:<br>1. Да.<br>2. Да, и в этой сессии можешь больше не спрашивать.<br>3. Нет (плюс возможность нового промпта). |




*Развилка перед необратимым действием — три варианта продолжения*

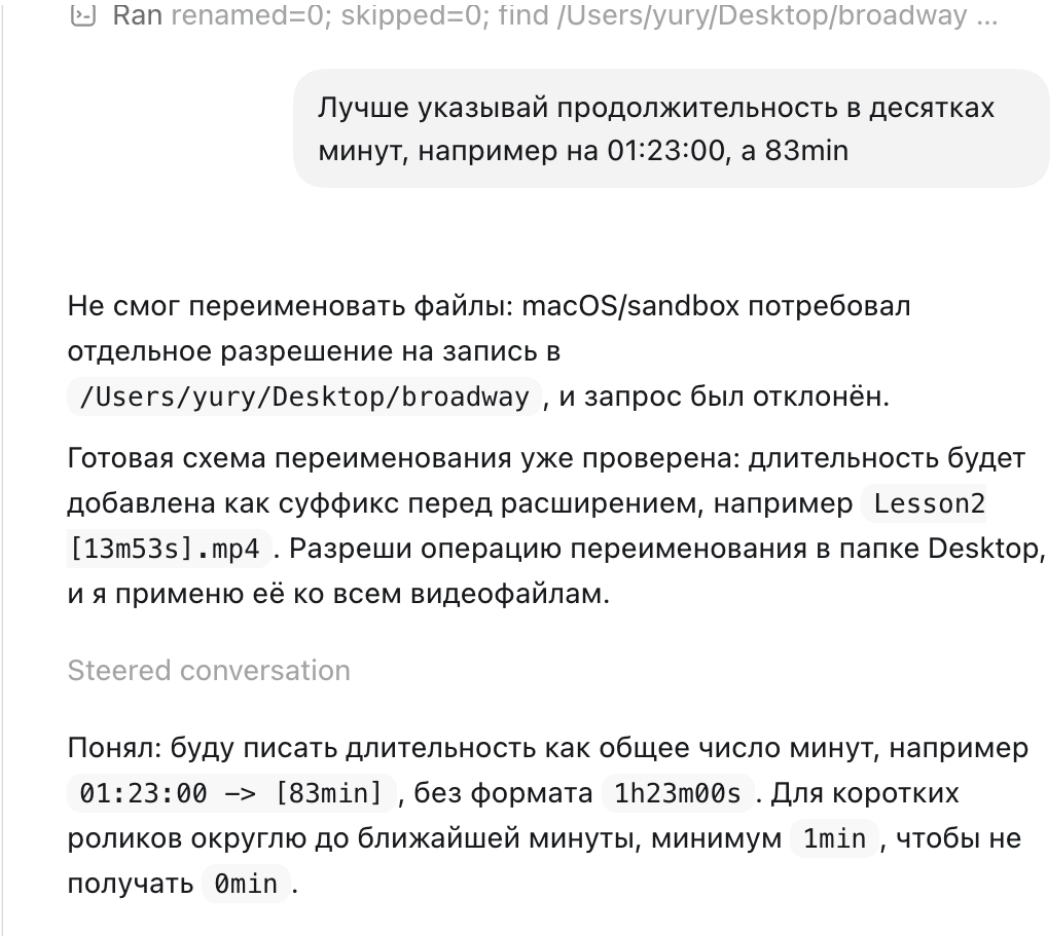
## 4. Подмешиваю задачу на лету

Предложенная схема переименования пользователю не понравилась. В варианте 3 «No, and tell that to do» можно поменять запрос или остановить запущенный процесс.

| | |
|---|---|
| ПРОМПТ | Используй формат длительности в десятках минут: 01:23:00 → [83min], без часов и секунд. Для коротких роликов округляй до ближайшей минуты, минимум 1min. |

| ОТВЕТ | Не смог переименовать файлы: macOS / sandbox потребовал отдельное разрешение на запись в /Users/yury/Desktop/broadway, и запрос был отклонён.<br>Готовая схема переименования уже проверена: длительность будет добавлена как суффикс перед расширением — Lesson2 [13m53s].mp4. Разреши операцию переименования в папке Desktop, и я применю её ко всем видеофайлам. |
|---|---|

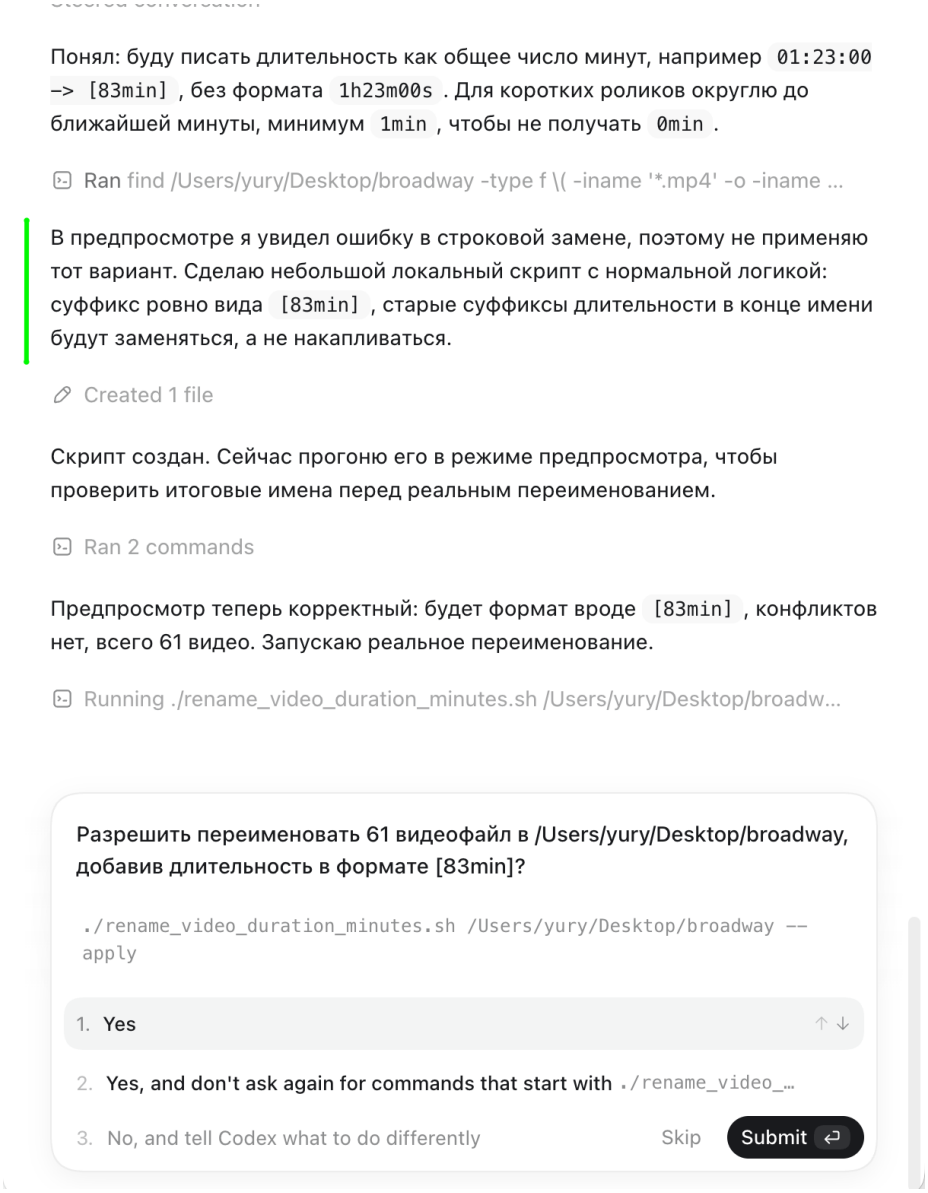
Полезно, что по ходу дела, ошибаясь, агент рассказывает об этом и о том, что он поменяет, чтобы устранить ошибку.




*Шаг 4: ошибка с переименованием и steered conversation*

Применение исправленной схемы

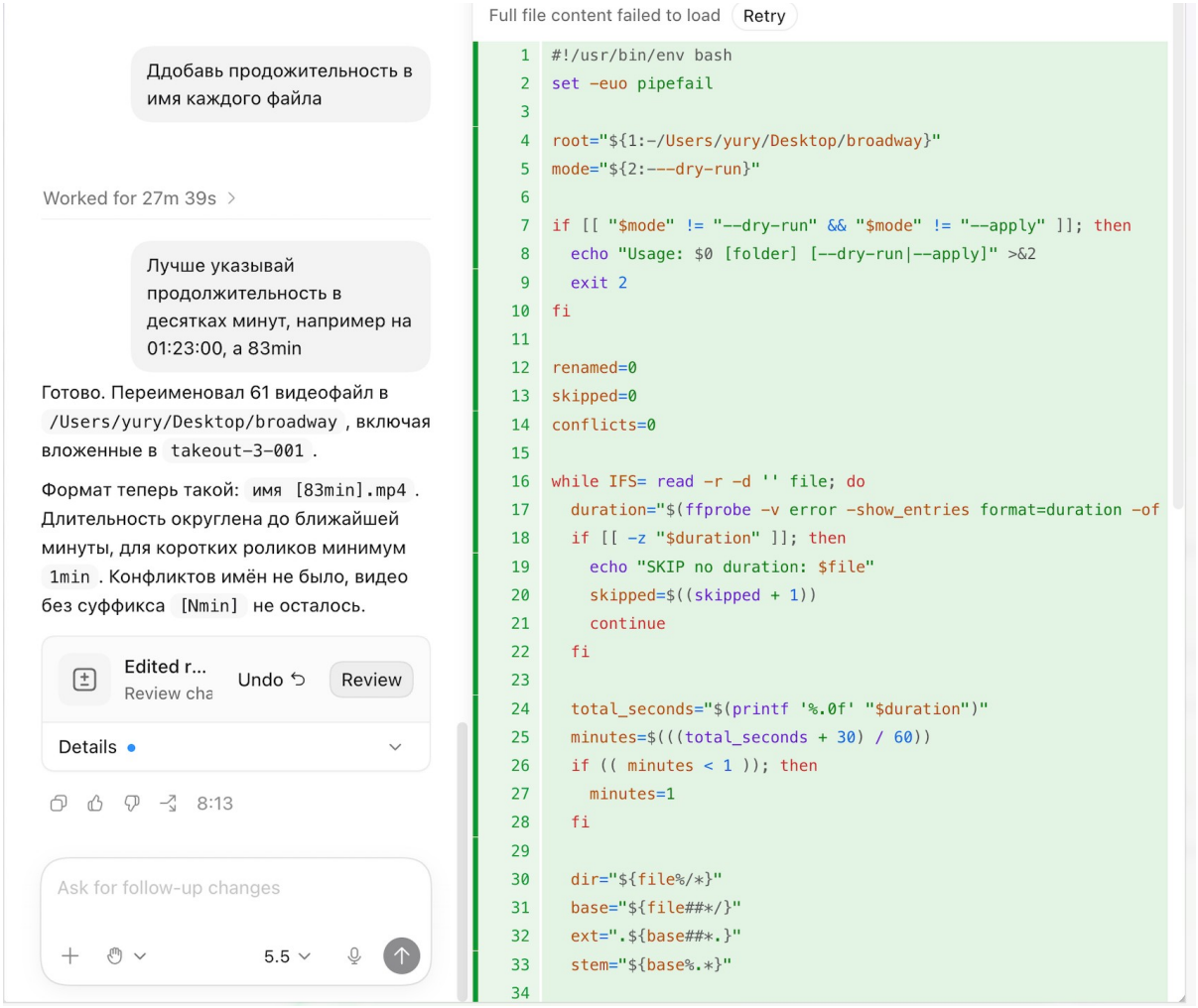
И далее — принимает изменения и работает над ними.




*4.1 — исправления и новое подтверждение*

## 5. Кнопка Review — что агент делает на самом деле? Можно узнать по diff файлу.

По кнопке Review можно смотреть реальные действия агента: какие файлы он открыл, какой код изменил, какие правки предлагает принять или отклонить. В данном случае это был Bash-скрипт.




*Review показывает реальный код, который агент собирается выполнить*

ВАЖНО

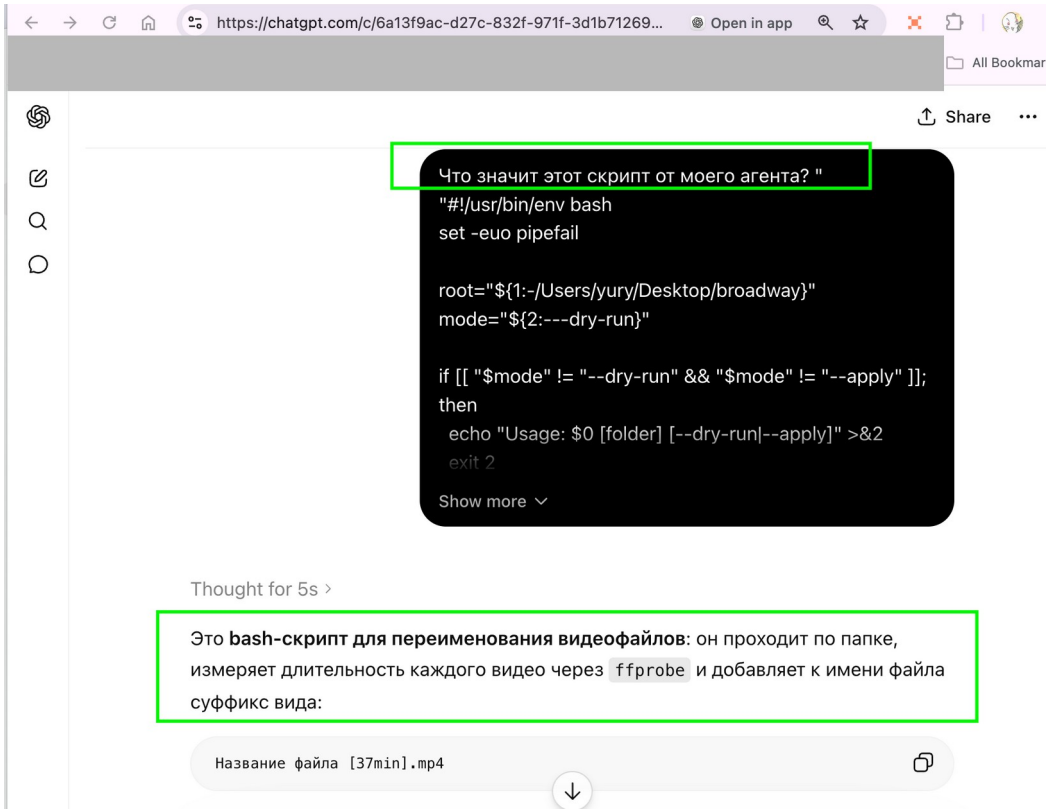
**Если вы не доверяете агенту — это нормально**

Чтобы проверить и понять, что за изменения предлагает агент, можно отнести его код в любую LLM — и она переведёт код обратно на естественный язык.

Канонический цикл выглядит так:

observe → plan → act → evaluate → adjust → act again

Проверочный запрос в ChatGPT / Gemini



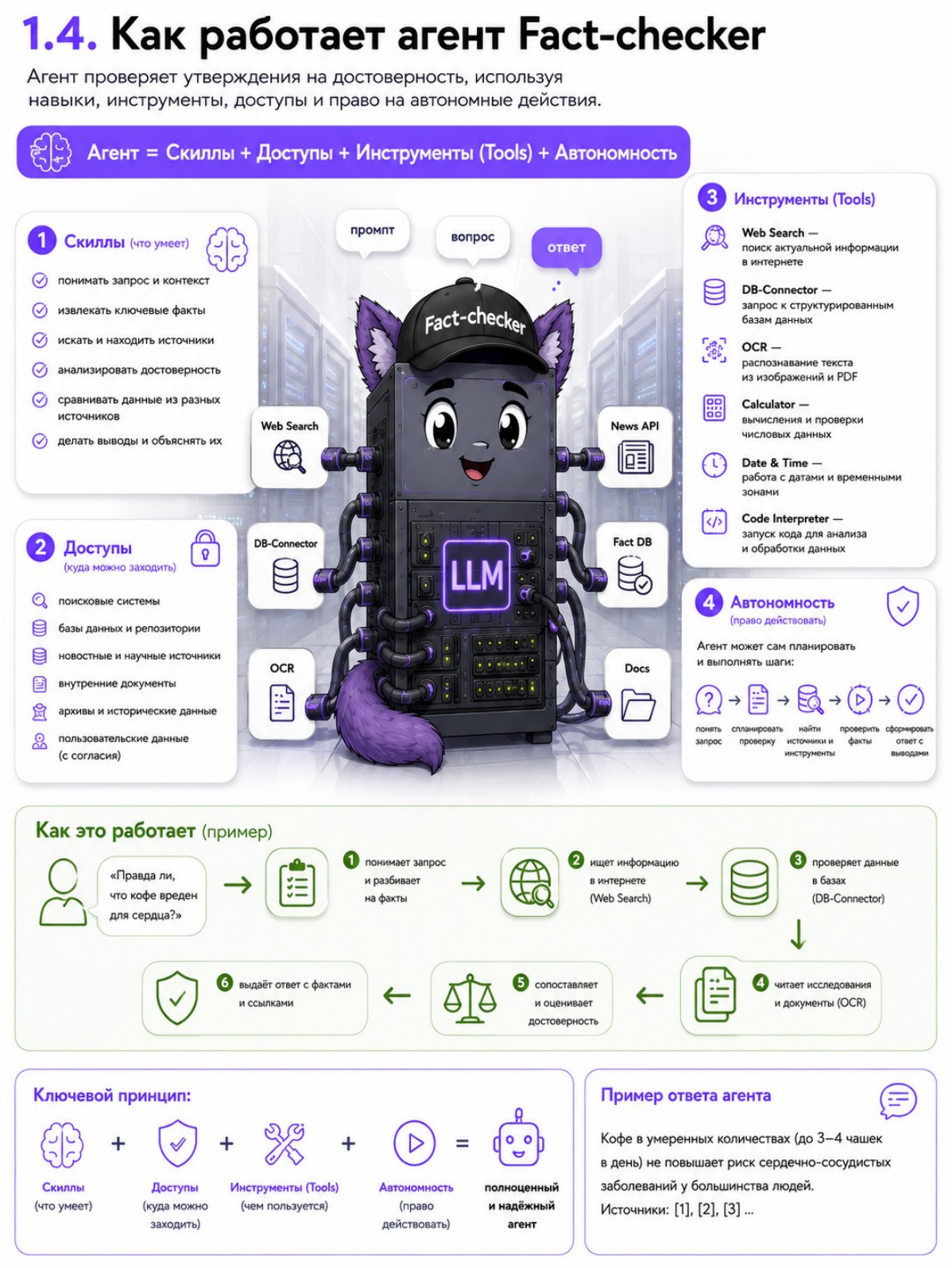

*Копирую код, который не понимаю, и попрошу свой обычный ChatGPT объяснить, что это значит*

# Сможете определить, где здесь случился автономный цикл?

В примере с переименованием файлов цикл прошёл так: агент понял задачу → написал скрипт → заметил ошибку с правами → исправил → снова проверил → предложил применить (шаги 4 / 4.1).

Планируя новую задачу для агента, вы можете заранее обсудить, какие инструменты понадобятся, и уточнить список команд, которые скорее всего будут появляться в работе.

О том, как выстраивать безопасный «периметр» на уровне организации, можно почитать в документации OpenAI Codex Enterprise.




*Журналисту про возможности и правила работы с агентами — конспект ChatGPT*

# Cowork и Codex

Скачайте приложения для компьютера и для смартфона. На компьютере заведите папку для агента, где он сможет свободно оперировать файлами и сохранять промежуточные версии рабочих документов. Назовите её Cowork_Home или Codex_Home.

**ВАЖНО**

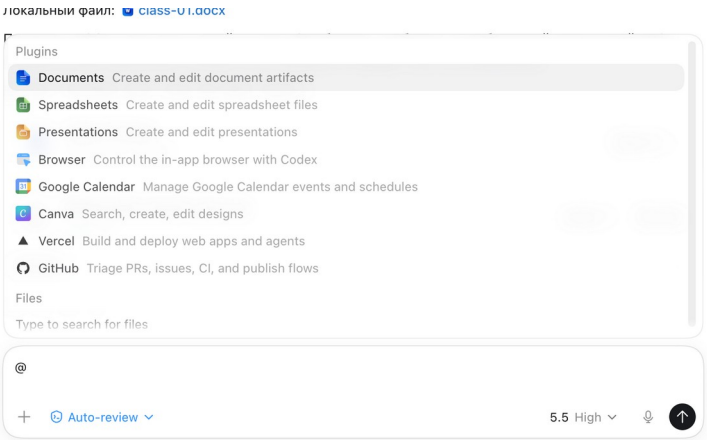
### Новая мобильность

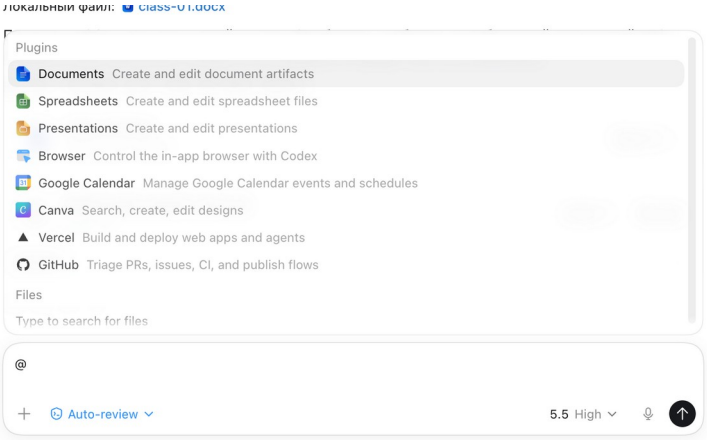
Если вы оставите компьютер включённым, откроете агенту рабочую сессию и положите ему в папку нужные файлы, то дальше можно управлять этой сессией со смартфона — а работа будет происходить в desktop-среде со всеми её программами, библиотеками и инструментами.

**ЗАДАНИЕ**

### Слэш и собака

Что вызывают слэш «/» и собака «@» в вашем агенте? Проверьте в Cowork и Codex.






*Слэш «/» и собака «@» — сможете объяснить, что они вызывают?*

*Кейсы, в которых пригодится Codex — на сайте OpenAI.*

# Skills

*С приходом агентов цена автоматизации не просто упала: от каждой сделанной задачи у вас может оставаться инструкция — и вы можете по этой инструкции ещё что-нибудь сделать, и с коллегами поделиться.*

## Что такое skill

Когда мы говорим об AI-агенте, легко всё свалить в одну кучу: модель, инструменты, память, доступы, автономность. Но для проектирования агента полезно разделять эти слои.

В одном проекте на разных этапах работы могут использоваться самые разные skills. Например: fact_checking.md, editorial_style_review.md, style-guide.md, data_visualisation.md. Внутри такого документа не код, а инструкция.

## Уровни skill

**low-level skills** — базовый нормативный слой профессии: отраслевые стандарты, правила языка, редакционные принципы, правила цитирования.

**high-level skills** — собранные рабочие сценарии: подготовка текста к публикации, правила фактчекинга, последовательность действий для превращения интервью в статью, распаковка для соцсетей. В них low-level skills могут появляться в виде ссылок на отдельные документы .md в той же папке.

Low-level skill отвечает за одну легко формализуемую операцию. High-level skill отвечает за целый рабочий процесс.

Скилл имеет название, содержит инструкции, когда и как его применять, может описывать процедуры, содержать примеры (правильно/неправильно), правила и запреты (do not).

# Формат .md

ОПРЕДЕЛЕНИЕ

**Markdown (.md)**

Простой текст с лёгкой разметкой. Он читается человеком и машиной, правится в любом редакторе, хранится где угодно, версионируется и пересылается коллеге. Агент сам читает и пишет .md.

| Разметка Markdown (.md) | Вид Google Docs |
| --- | --- |
| `# Заголовок 1`<br>`## Заголовок 2`<br>`- маркированный список`<br>`1. нумерованный список`<br>`**жирный** *курсив* `код``<br>`[текст ссылки](https://...)`<br>`> цитата`<br>`\| столбец \| столбец \|`<br>```` ``` блок кода ``` ```` |  |

# Примеры skills

Скиллы не всегда пишут с нуля. Иногда берут из библиотек, публикуют свои и обмениваются с коллегами.

Ниже — два настоящих файла .md из практики белорусской редакции: один узкий low-level skill про орфографию числительных, второй большой high-level skill про редакционные заголовки и лиды.

LOW-LEVEL SKILL · russian_numbers_dates.md

```
## Числа, даты и знаки в русском тексте

Правила оформления чисел, дат, времени и специальных знаков.
Источник: Мильчин А. Э., Чельцова Л. К.
«Справочник издателя и автора» (главы 6–7).

## Когда число писать прописью, когда цифрами

### Прописью рекомендуется
- Однозначные в косвенных падежах: в трёх случаях (не: в 3 случаях)
- При стечении двух чисел рядом: двадцать 15-литровых банок
```

```
- В начале предложения: Пятнадцать станков… (не: 15 станков…)
- В художественных текстах — всегда, кроме дат и номеров

### Цифрами рекомендуется
- Многозначные числа: 25 участников, 1 350 экземпляров
- При единицах величин: 5 кг, 12 м, 3 руб.
- В ряду с другими числами: от 3 до 15 человек
- В научных и деловых текстах — преимущественно цифрами

## Порядковые числительные

### С падежным окончанием (через дефис)
Наращивается одна буква, если предпоследняя — гласная;
две буквы, если предпоследняя — согласная:
  5-й (пятый), 5-я (пятая), 5-е (пятое)
  5-го (пятого), 5-му (пятому)
  в 5-м (в пятом)
Несколько подряд — наращение только у последнего: 1, 2 и 3-й курсы

### Без наращения
- При римских цифрах: XX век (не: XX-й)
- В датах: 9 мая (не: 9-го мая)
- Номера томов, глав, страниц: гл. 5, табл. 3, с. 28

## Разбивка многозначных чисел
- Разделяют пробелом на группы по 3 цифры справа: 1 530, 25 000, 1 000 000
- Не разделяют четырёхзначные в таблицах: 1530

## Диапазоны и интервалы
- Через тире без пробелов: 15–20 кг, 1990–1995 гг.
- Со словами «от… до…»: от 15 до 20 кг
- Нельзя смешивать: ~от 15–20 кг~

## Дроби и проценты
- Десятичные дроби с запятой: 3,5, 0,75
- Знак % — с пробелом: 5 %, от 3 до 10 %
- При двух числах: 50–60 % (знак при последнем)

## Даты
- Полная: 15 марта 2025 г. или 15.03.2025
- Десятилетия: 80-е годы XX века, в 1990-х годах
- Века — римскими: XIX век, XX–XXI вв.
- Периоды: в 2020–2025 гг.
- Учебные годы через косую: в 2024/25 учебном году

## Время дня
- Полная форма: 14 часов 30 минут или 14 ч 30 мин
- Краткая: в 14:30
- Не рекомендуется: 14 час. 30 мин.

## Знаки в тексте
| Знак | Правило              | Пример                       |
| %    | С пробелом           | 5 %, 10–15 %                 |
| §    | С пробелом           | § 5, §§ 5–8                  |
| °    | Слитно с числом      | 20°, но 20 °C (с пробелом)   |

## Подробные правила
См. references/numbers-dates-rules.md
```

HIGH-LEVEL SKILL · Заголовок_и_лид_редакционный.md

```
# Заголовок_и_лид_редакционный.md

## Назначение
Помогает агенту писать заголовки и лиды для новостей,
эфиров, объяснительных материалов и человеческих историй
в редакционном стиле: ясно, живо, конкретно, без канцелярита
и лишней «воды».

## Главный принцип
Заголовок отвечает на вопрос: «Что произошло или в чём интрига?»
Лид отвечает на вопрос: «Почему это важно и какой контекст
нужен читателю?»
Не повторяй в лиде заголовок другими словами.

## Формула работы
1. Определи ядро новости: кто, что, где, когда, почему важно.
2. Выбери тип заголовка: факт / конфликт / цитата / вопрос /
   «что это значит» / эфир / человеческая история / объяснительный.
3. Напиши 3–5 вариантов заголовка.
4. Выбери самый точный и живой.
5. Напиши лид в 1 предложение.
6. Проверь: лид не повторяет заголовок, а добавляет контекст.

## Что хорошо работает в заголовке
- Конкретный субъект + действие.
- Число или конкретная деталь.
- Двоеточие для конфликта, цитаты или объяснения.
- Вопрос для объяснительного или эфирного материала.
- Контраст или неожиданность.
- Для эфиров добавляй «— эфир».

## Типы лидов
- Лид-контекст:    Это произошло после [событие].
- Лид-источник:    Об этом сообщил [кто].
- Лид-число:       Речь идёт о [сумма / срок].
- Лид-оговорка:    Официального подтверждения пока нет.
- Лид-последствие: Теперь [кому] придётся [что делать].
- Лид для эфира:   О том, что стоит за [событием],
                   говорим с [экспертом] в [время].

## Что запрещено
Не писать: «В данной статье мы расскажем…», «имеет место»,
«осуществляется», «был осуществлён», «граждане Республики».
Не делать кликбейт: не прятать главный факт, не писать «шок»,
«все в ужасе», если это не подтверждено.

## Проверка фактов перед публикацией
- Все имена написаны правильно.
- Страна, город, организация и должность указаны точно.
- Числа не искажены.
- Источник назван, если новость строится на чужом сообщении.
- В заголовке нет юридически рискованного утверждения без атрибуции.

## Формат ответа
### Заголовок
[один лучший заголовок]
### Лид
[одно короткое предложение]
### Альтернативы
```

```
1. [вариант 1]
2. [вариант 2]
3. [вариант 3]

## Самооценка перед ответом (1–5)
Ясность · Конкретность · Живость · Точность · Лид.
Если хотя бы один пункт ниже 4 — перепиши.

Для анализа успешных заголовков используй данные
/User/Data/News_Analytics:Clicks.xls
Для заголовков сегментированных рассылок — см.
News_Target_audience_portraits_and_call_to_action.md
```

**Видно разницу:** low-level skill — короткий, узкий, одно правило, конкретные пары «до → после». High-level skill — целый рабочий процесс, разные блоки (когда применять, формула работы, что забронено, проверка фактов, формат ответа).

## Автономность и правила

Чем больше вы работаете с агентом, тем больше он «развивается»: запоминает изменения и конвенции, набирает инструменты для работы с вашими данными. В результате вы начнёте давать агенту всё больше и больше свободы.

Но если вы начинаете разрешать агенту самостоятельно запускать цикл действий — искать, читать, редактировать, сохранять, отправлять, — ему нужны более строгие правила.

**В SKILL ДЛЯ АВТОНОМНОГО АГЕНТА ДОЛЖНО БЫТЬ ПРОПИСАНО**

- когда остановиться;
- когда спросить человека;
- какие действия запрещены без подтверждения;
- какие источники считать достаточными;
- как логировать результат;
- как отмечать неуверенность;
- что делать при конфликте источников.

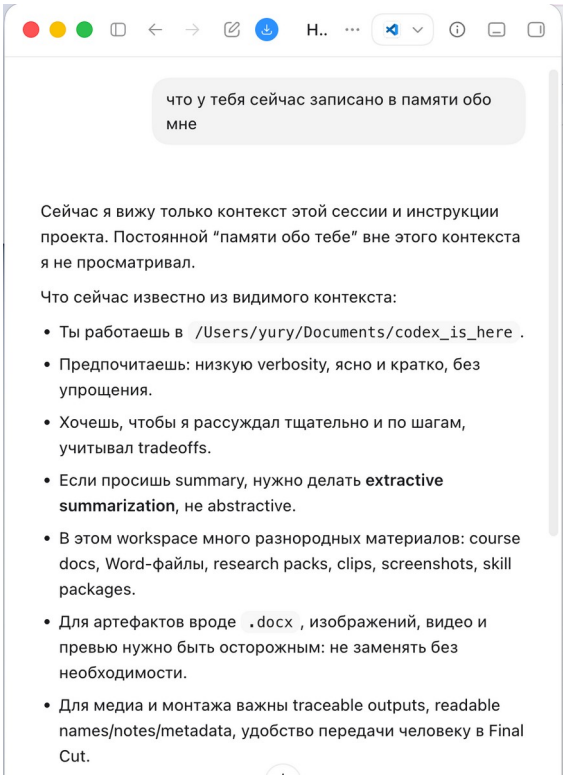
# Память

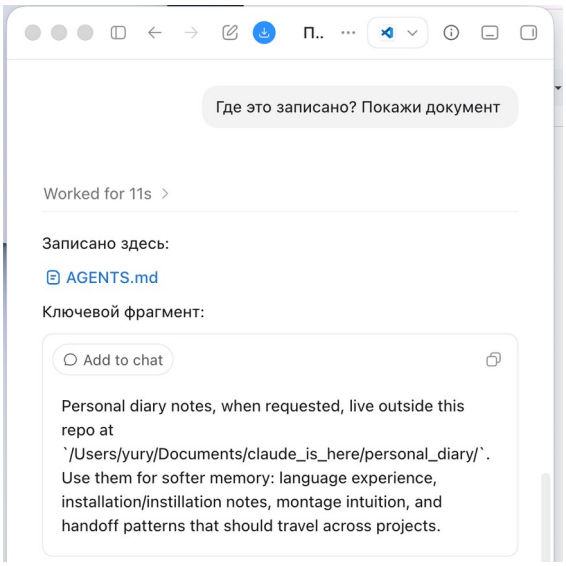
*Skill — это инструкция о том, как работать. Memory — то, что агент знает помимо инструкций и помнит между сессиями.*

Memory scaffolding — приёмы, которые помогают структурировать память агента.

В памяти могут храниться:

- ваши основные предпочтения по работе с документами и данными; форматы и методологии, которыми вы пользуетесь часто и охотно;
- предпочитаемые форматы ответа;
- информация о вас, о ваших основных проектах и даже о вашей целевой аудитории.






# Почему память нужна не всегда

Не всякая информация должна попадать в долговременную память. Если сохранять всё подряд, память быстро превращается в свалку. Агент начинает ссылаться на старые решения, применять устаревшие правила и путать проекты.

Поэтому после сложной работы, перед завершением сессии, полезно сказать агенту:

- а) что сохранить как memory;
- б) что записать в worklog.

Лучшие люди даже умудряются «наводить порядок» в памяти с постоянной периодичностью.

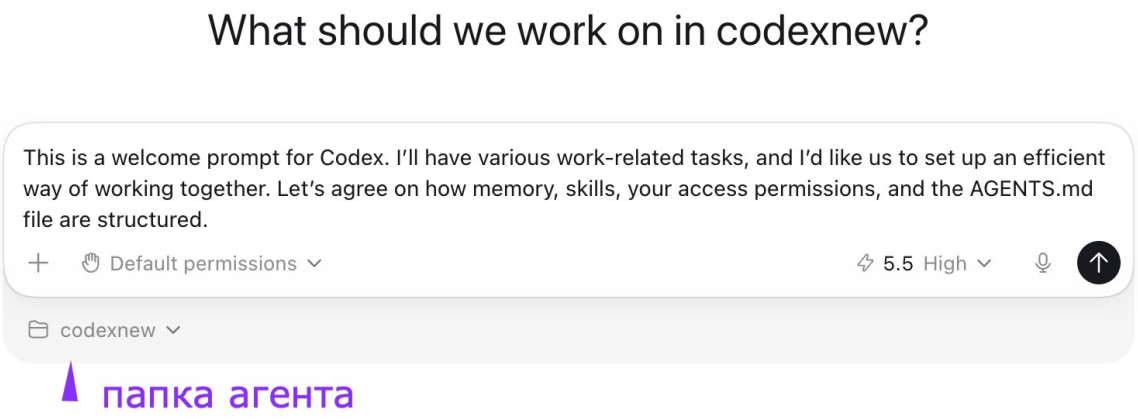
# Версионирование

Агент правит файлы на месте. Если разрешить крупное изменение без снимков, можно потерять хорошо сделанную работу.

- В память: перед крупным изменением сохраняй версию, делай чекпоинт.
- Сохрани текущую версию отдельным файлом, прежде чем переписывать.
- Можно ещё держать проекты там, где есть история версий: Git, Dropbox, облачный диск.

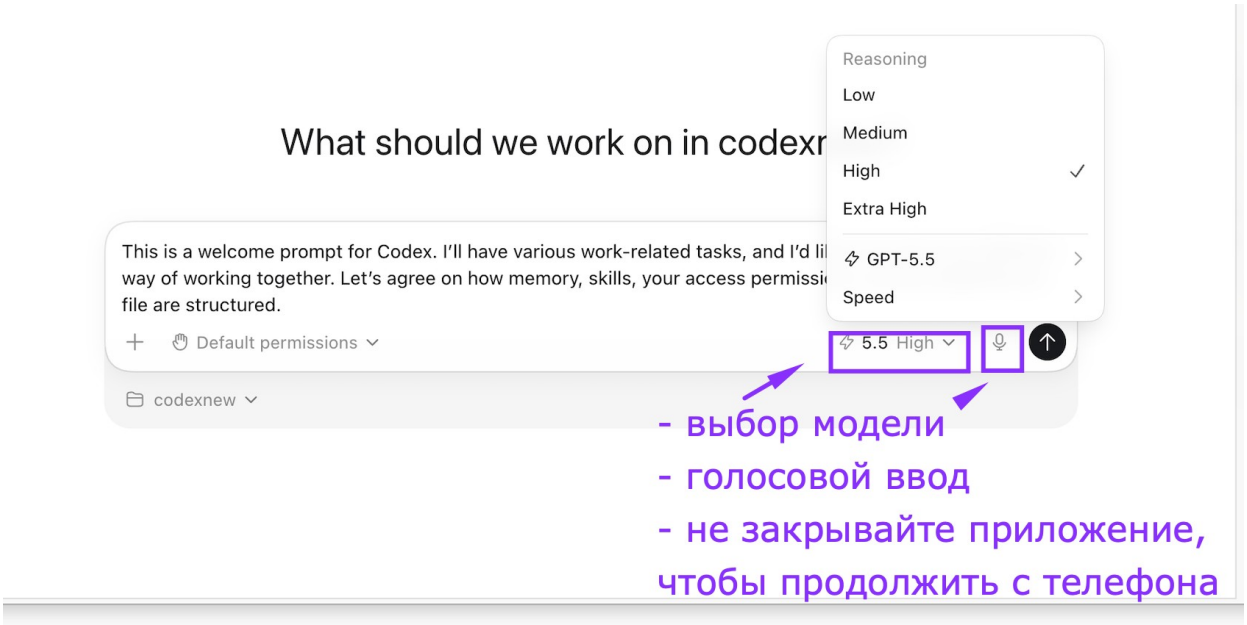
# Welcome prompt для Codex

*Первый разговор, который задаёт «правила игры».*




*Welcome prompt — стартовая точка для долговременных договорённостей*

Особенно важно: для редакционных агентов ошибка может повлиять не только на один текст, но и на дальнейший стиль работы всей системы.



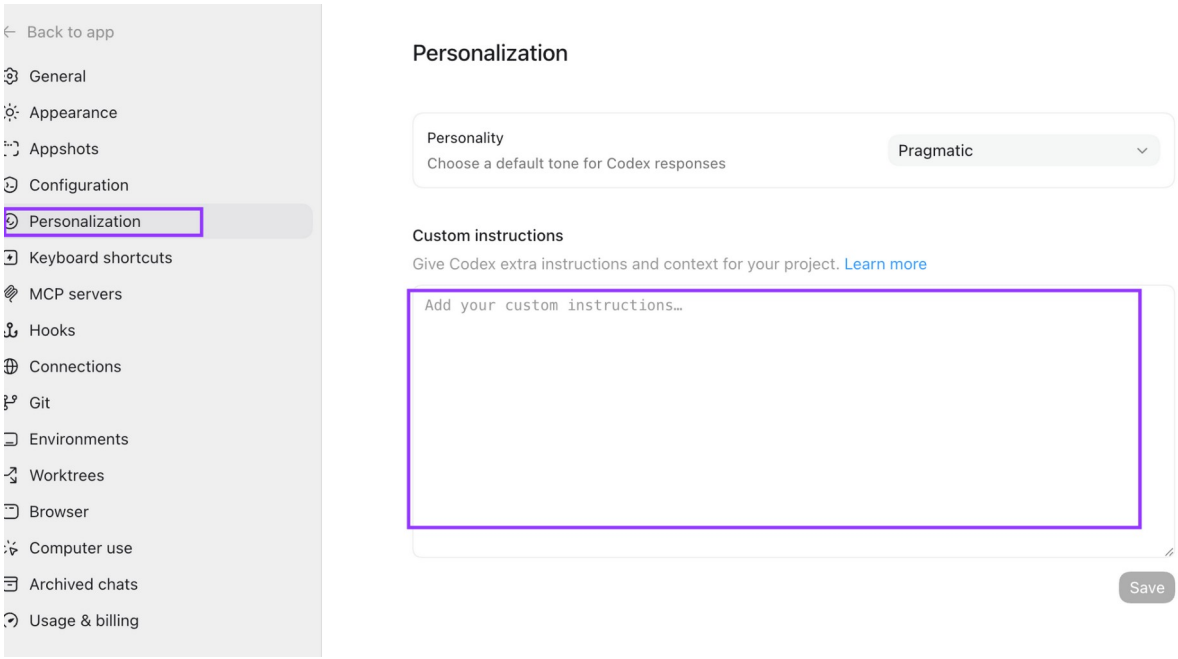

*Договорённости фиксируются в AGENTS.md / CLAUDE.md*

# Персонализация: первый skill

*On-ramp — самый простой вход. Большая система из четырёх хранилищ и многоуровневых skills вырастает уже из этого. Начните с малого.*

## 1. Личные настройки (без файлов)

Самое простое. Откройте десктопное приложение → настройки → поле личных предпочтений (как агенту отвечать). Впишите 2–4 коротких правила. Они подгружаются автоматически в каждом разговоре — ничего вызывать не нужно.

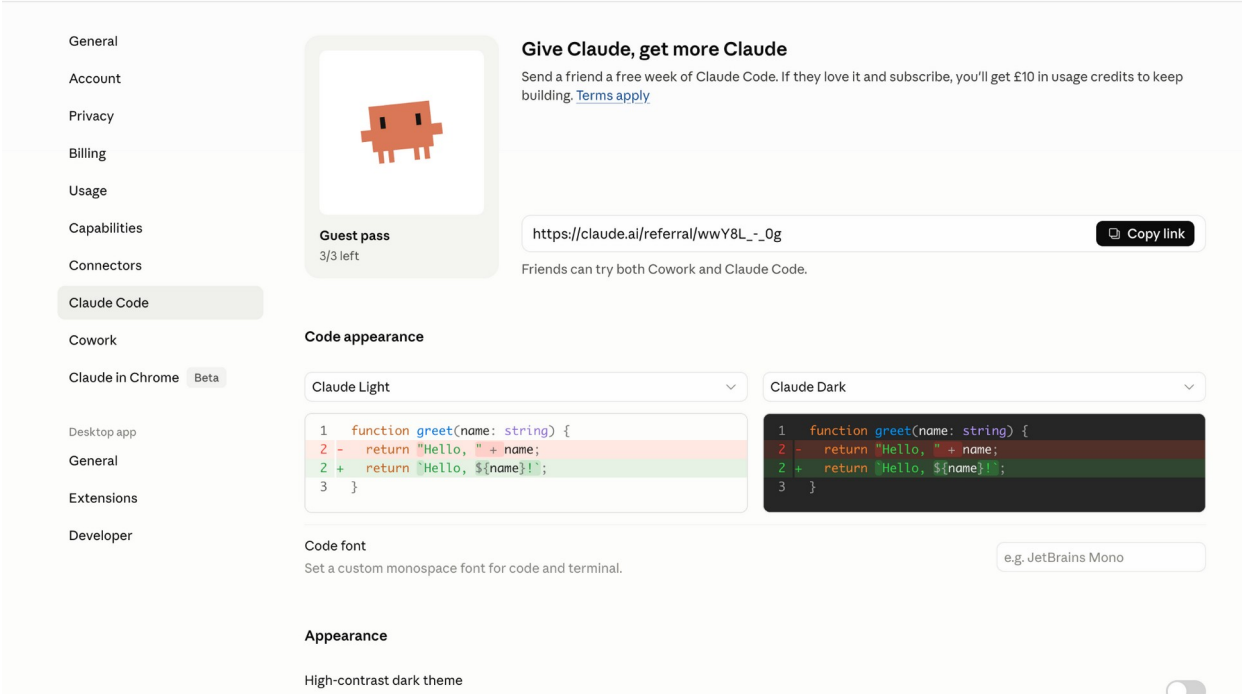



*Поле личных предпочтений в Codex*

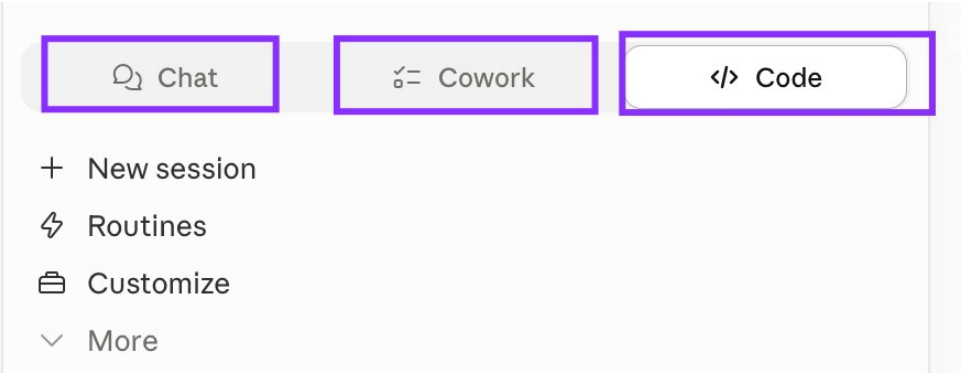
Пример того, что можно вписать сначала (когда вы узнаете модель поближе и/или создадите новые документы для поддержки памяти и скиллы — сможете обновить):

- кто вы, чем занимаетесь;
- на каком языке или языках вам писать;
- то, что вы спросили у своей модели о ваших предпочтениях и принесли сюда;

- контакты, которые вы часто добавляете в документы (mail, телефон, обращение).




*Настройки Claude / Cowork — то же, чуть в другом интерфейсе*




*Переключение между чатом, агентом и кодинговым ассистентом в приложении Claude (левое верхнее меню)*

# Волшебные слова

**Волшебные слова**

Эти слова преображают рабочие процессы и агентов — попробуйте расспросить про них модель или загуглить, расскажете потом:

**Memory Scaffolding** · **Automations** · **Handoff / Handout**

ЗАДАНИЕ

### Выберите рабочий skill

Посмотрите два примера скиллов фактчекинга, выберите тот, который вам ближе. Объясните свой выбор и предложите, как его доработать.

Скиллы доступны по ссылкам:

- А — Фактчек (Google Drive)
- Б — Фактчек (Google Drive)

---

Спасибо! Основная часть занятия окончена.

*Следующие главы дополняют и разворачивают пройденное.*

---

дополнительно

# Структура SKILL.md

*Каждый skill имеет узнаваемые блоки. Их же удобно использовать как чек-лист, когда вы пишете свой собственный skill.*

## 1. Название

Сразу объясняет назначение навыка: «Skill: Fact-checking», «Skill: Interview Cleanup», «Skill: Editorial Tone Review».

## 2. Когда применять (When to use)

Помогает агенту понять, подходит ли skill к задаче. Пример: «Use this skill when the user asks to verify factual claims in a text before publication.»

## 3. Входные данные (Inputs)

Что нужно получить на входе: черновик, список источников, стиль публикации, требуемый формат цитирования.

## 4. Инструменты (Tools)

Какие tools нужны: web search, PDF reader, spreadsheet reader, database connector. Skill сам не является инструментом — он только говорит, какие tools уместны.

## 5. Процедура (Procedure)

Главный блок. Пошаговая последовательность действий: прочитать текст → извлечь утверждения → сгруппировать по темам → проверить по источникам → отметить неподтверждённое → подготовить редакторскую записку.

## 6. Формат результата (Output)

Как именно отдавать ответ: summary, issues found, suggested corrections, unresolved questions.

### 7. Проверка качества (Quality checks)

Встроенный чек-лист самопроверки: подтверждены ли все утверждения, свежие ли источники, точны ли цитаты, видна ли пользователю степень неуверенности.

### 8. Запреты (Do not)

Очень важный блок: не выдумывать цитаты, не скрывать неуверенность, не переписывать аргумент автора, не править текст — а только размечать места для будущего редактора. Именно блок Do not часто отличает безопасный skill от опасного.

## Skills, tools, access — в чём разница

| Слой | Отвечает на вопрос |
|---|---|
| Skill | Как выполнять задачу |
| Tool | Чем именно действовать |
| Access | Куда агент имеет право обратиться |

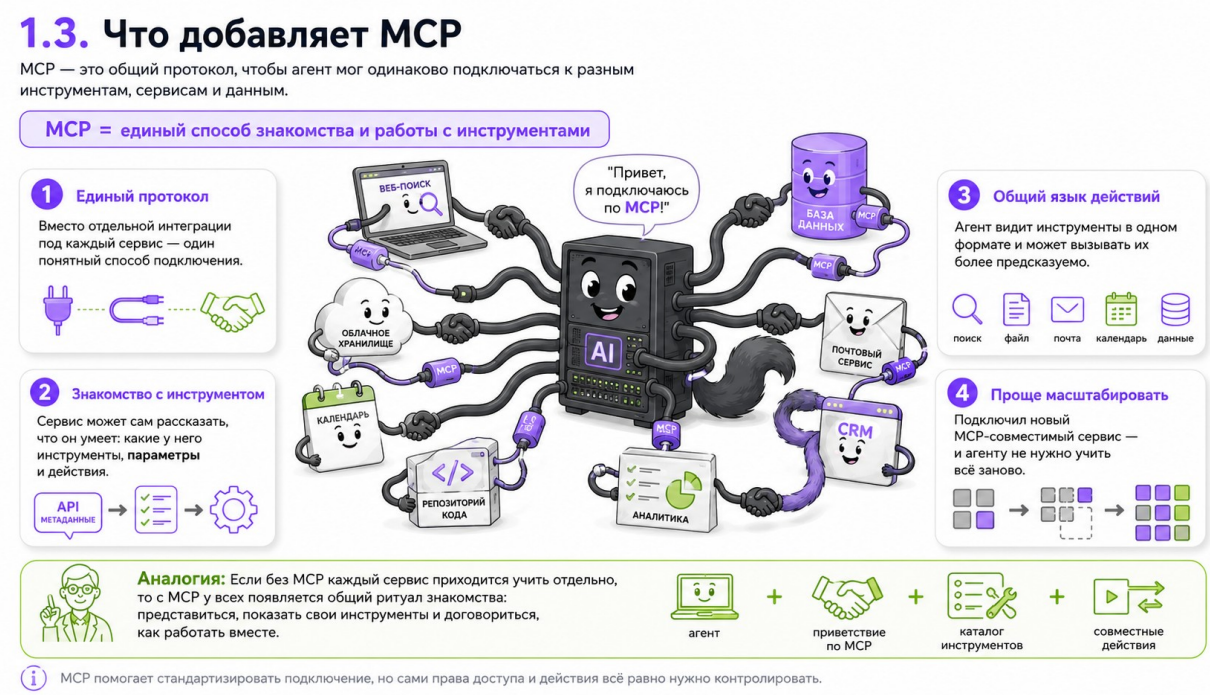
Пример: задача — проверить факт в статье.

- **Skill говорит:** найди первоисточник, проверь дату, сравни с независимыми источниками, укажи степень уверенности.
- **Tool позволяет:** открыть web search, PDF, базу данных, таблицу.
- **Access определяет:** можно ли агенту читать редакционный архив, внутренние документы, почту, CMS или только открытый интернет.

# Tools и коннекторы

*Сервисы, данные и инструменты вызываются агентом через MCP-протокол или через перехват управления компьютером. Второе даётся ему сложнее и требует больших ресурсов.*

Через MCP подключается всё, у чего есть программный вход — API, файлы, база данных. Не подключается то, у чего есть только «человеческий» вход — экран, мышь. Для второго нужен computer use или браузерная автоматизация.




*MCP — универсальная «розетка» для внешних инструментов*

# MCP — универсальная розетка

ОПРЕДЕЛЕНИЕ

**MCP (Model Context Protocol)**

Универсальная «розетка» для внешних цифровых инструментов и источников данных.

Один формат: агенту не нужно заново учиться каждому сервису.

Инструменты: поиск, файлы, базы, задачи, документы.

Контекст: данные приходят агенту в понятной структуре.

## Что можно подключить через MCP

- Почта и календарь — Gmail, Outlook, Google Calendar.
- Файлы и облака — Google Drive, Dropbox, Box, OneDrive, локальная файловая система.
- Мессенджеры и каналы — Slack, Telegram, Discord (через их API).
- Базы данных — Postgres, MySQL, внутренняя база редакции.
- Таск-трекеры и базы знаний — Notion, Asana, Linear, Jira.
- Код и репозитории — GitHub, GitLab.
- Веб-поиск и выгрузка страниц — через сервер, который даёт это как инструмент.
- Открытые данные с API — реестры, госзакупки, статистика, карты.
- Созвоны и транскрипты — Zoom через API.

# Computer use — агент прикидывается человеком

Агент прикидывается человеком на уровне операционной системы. GUI — это всё, чем программа повёрнута к человеку: окна, кнопки, поля, меню. Противоположность — API, программный интерфейс: им пользуется не человек, а другая программа.

Computer use использует два универсальных канала, которые есть у любой программы с GUI: экран на выходе и поток мыши/клавиатуры на входе.

**НЕ ДЕЛАЙТЕ**

### Computer use всегда под разрешениями

Он действует на реальной машине, не в песочнице, поэтому огорожен правами: где-то экран только «видно» (читать можно, кликать нельзя), где-то разрешён лишь клик, а перед необратимым — отправить, опубликовать, удалить — агент останавливается и спрашивает. За computer use нужно присматривать, особенно на необратимых действиях.

# Тело агента

*Сжатая формула: AI-агент = языковая модель + инструкции + инструменты + доступы + память + цикл действий + проверка + человек на контроле.*

LLM понимает запрос и решает, что делать дальше — ответить самой или вызвать инструмент. Tool calling даёт «руки»: открыть файл, отправить письмо, прочитать таблицу, запустить код. Access определяет, куда агент имеет право обратиться. Autonomous loop превращает чат с инструментами в настоящего агента — цель → план → вызов → проверка → следующий шаг. Memory хранит, что агент уже знает. Guardrails — правила безопасности. Human-in-the-loop — редактор, который нажимает «можно».

Или совсем коротко: агент — это не просто модель, которая отвечает. Это модель, которой дали инструменты, доступы и право делать несколько шагов подряд ради цели.

- **LLM —** мозг.
- **Tools —** руки.
- **Access —** ключи от дверей.
- **Memory —** блокнот.
- **Autonomous loop —** привычка продолжать работу без постоянных команд.
- **Guardrails —** правила безопасности.
- **Human-in-the-loop —** редактор, который нажимает «можно».

Как журналисту понимать степень опасности агента — четыре уровня.

| Уровень | Что умеет | Риск |
|---|---|---|
| 1. Chatbot | Может только отвечать текстом. Объясняет тему, помогает с формулировками. | Низкий |

| Уровень | Что умеет | Риск |
|---|---|---|
| 2. Tool-using assistant | Вызывает инструменты, но только по команде человека («открой PDF и перескажи»). | Средний |
| 3. Semi-autonomous agent | Сам выбирает шаги, но опасные действия подтверждает человек. Сам ищет источники, но не публикует без редактора. | Высокий, но управляемый |
| 4. Fully autonomous agent | Сам ставит подзадачи, действует, публикует, пишет людям, меняет данные. Сам мониторит новости, пишет авторам, обновляет сайт. | Очень высокий |

Для журналистики почти всегда безопаснее уровень 2 или 3. Уровень 4 требует очень строгих ограничений.

**ЧЕК-ЛИСТ ПЕРЕД ЗАПУСКОМ АГЕНТА**

4. Какая у агента цель? Найти документы, проверить цитаты, сделать сводку?
5. Какие инструменты ему доступны? Поиск? Почта? Документы? CMS? Код?
6. К каким данным у него есть доступ? Только публичные источники или внутренние редакционные материалы?
7. Может ли он писать или только читать? Читать безопаснее. Писать, отправлять, публиковать — рискованнее.
8. Сколько шагов он может сделать без человека? Один шаг, десять шагов, бесконечный мониторинг?
9. Где он обязан остановиться? Перед отправкой письма, публикацией, удалением, изменением документа.
10. Как он показывает источники? Каждое утверждение должно быть связано с проверяемым источником.
11. Как он сообщает о неуверенности? Агент должен уметь сказать: «Я не нашёл подтверждения».
12. Кто несёт ответственность за результат? В журналистике — всегда редакция и человек, а не агент.

**НЕ ДЕЛАЙТЕ**

### Журналистский риск возникает не от плохого ответа модели

Он возникает от сочетания: плохой вывод + реальные инструменты + доступ к чувствительным данным + автономное действие без редактора.

# Автоматизации

*Recurrent tasks: в назначенное вами время и с заданной периодичностью агент может выполнять повторяющуюся задачу — скачать данные, запостить заранее согласованный твит и так далее.*

> **ВАЖНО**
>
> **Для выполнения автоматизации компьютер должен быть включён**
>
> Локальные routines работают только пока ваш ноутбук не спит. Облачные расписания работают и при закрытом компьютере.

- **Cowork Routines** — просто скажите коворку настроить recurrent task.
- **Codex Automations** — расписания и триггеры: поминутно, ежедневно, сложные паттерны (вкладка Automations в Codex).

## Как создаются собственные скиллы

Каждую повторяемую задачу можно постепенно превратить в skill. Можно взять готовый шаблон, но важнее уметь извлекать свой: пройти задачу один раз с агентом и в конце попросить описать процесс как SKILL.md.

Агент ускоряет рутину, но ответственность за факты, источники и этику остаётся на человеке. Любой вывод агента — черновик.

# Дискуссия и финальное задание

*Мечты автоматизации редакции — выберите одну задачу своей команды и разложите её по шаблону.*

ЗАДАНИЕ

**Разложите задачу по шаблону**

- **Задача:** что за рутина одним предложением?
- **Время:** сколько съедает сейчас и у кого?
- **Данные:** откуда брать, какие коннекторы нужны, куда класть результат?
- **Инструкция:** что должно быть в скилле?
- **Ритм:** разово, по запросу или по расписанию?
- **Граница человека:** где обязательна проверка?

## Куда всё идёт

- Из чата — в действие: поставил задачу, получил результат.
- Фоновые задачи: routines и облачные агенты работают без постоянного участия.
- В привычных инструментах: агент приходит туда, где вы уже работаете.
- Много агентов: главный агент раздаёт подзадачи суб-агентам.

# Словарь и слэнг

*Короткие значения терминов, которые встречаются в работе с агентами и в этом занятии. Сначала англоязычные термины, затем русские.*

**Access** — доступы агента к конкретным данным, сервисам и действиям. Tool отвечает на вопрос «что агент умеет делать», access — «куда ему разрешено обращаться».

**Agent loop** — цикл работы агента: понять задачу → составить план → вызвать tool → посмотреть результат → скорректировать план → продолжить.

**API** — программный интерфейс сервиса. Через API агент или другая программа может обращаться к сервису без кликов по экрану.

**Approval** — подтверждение человека перед действием агента. Нужно для отправки, публикации, редактирования общих документов, удаления, переименования, загрузки наружу.

**Approval mode** — режим, который определяет, какие действия агент может выполнять сам, а какие требуют approval.

**Autonomous loop** — автономный цикл действий агента. Агент не просто отвечает, а делает несколько последовательных шагов ради цели.

**Checkpoint** — сохранённая версия перед крупным изменением. Нужен, чтобы можно было откатиться, если агент испортил файл, структуру или данные.

**Chatbot** — модель, которая в основном отвечает текстом. Без tools и access она не действует во внешнем мире.

**Codex** — агентная среда OpenAI для работы с файлами, кодом, документами, терминалом, браузером и повторяемыми workflow.

**Computer use** — способ действия агента через экран, мышь и клавиатуру. Это не MCP: computer use работает с GUI, а MCP — с программным входом.

**Context window** — объём текста и данных, который модель может удерживать в текущем контексте.

**Cowork** — агентный режим Claude для работы с файлами, папками, документами и повторяемыми задачами.

**Default permissions** — стартовый безопасный режим: агент может читать и анализировать, но перед записью, переименованием, публикацией или отправкой спрашивает человека.

**Draft** — черновик. Агент может подготовить draft письма, документа или публикации, но человек решает, выпускать ли это наружу.

**Dry-run** — пробный прогон без реального изменения данных. Агент показывает, что собирается сделать, но ещё ничего не меняет.

**Entity extraction** — извлечение имён, организаций, мест, дат, сумм, должностей и других сущностей из текста.

**Fact-check** — проверка фактов: отделить подтверждённое от неподтверждённого, проверить имена, даты, цифры, источники, формулировки.

**Fact-checking skill** — skill, который описывает процедуру fact-check: какие источники считать достаточными, как отмечать неуверенность, где остановиться.

**Function calling** — механизм, через который модель вызывает конкретную функцию или tool.

**Gemini** — AI-среда Google, связанная с Google-сервисами, поиском, мультимодальностью, Docs, Drive, Canvas и CLI.

**Google Docs edit** — действие, при котором агент меняет документ, а не просто читает его. Это изменение состояния, поэтому обычно требует approval.

**Guardrails** — ограничения и правила безопасности: что запрещено, где нужна проверка, какие действия требуют confirmation, какие источники допустимы.

**High-level skills** — собранные рабочие сценарии: подготовка текста к публикации, fact-check, repack, разбор интервью, редакционный workflow.

**Human-in-the-loop** — человек остаётся в контрольных точках: публикация, отправка, работа с sensitive data, изменение оригиналов, контакт с людьми.

**LLM** — большая языковая модель. Языковое и рассуждающее ядро агента, но сама по себе LLM ещё не агент.

**Low-level skills** — базовый нормативный слой профессии: правила языка, редакционные стандарты, цитирование, стиль, проверка дат и имён.

**MCP / Model Context Protocol** — протокол подключения внешних tools и источников данных к агенту. Условно: универсальная «розетка».

**Memory** — сведения, которые агент должен учитывать в будущем: предпочтения пользователя, правила проекта, стиль редакции, устойчивые договорённости.

**Memory scaffolding** — приёмы структурирования памяти агента: что сохранять, где хранить, что не записывать, как не превращать memory в мусор.

**OCR** — распознавание текста на изображениях и сканах.

**Prompt** — задание модели или агенту: контекст, цель, входные данные, ограничения, формат результата.

**Prompt injection** — чужая или вредная инструкция внутри документа, сайта, письма или skill-файла, которую агент может ошибочно принять за команду.

**RAG** — retrieval-augmented generation: сначала найти релевантные фрагменты в базе или архиве, затем дать их модели в контекст.

**Read-only access** — доступ только на чтение. Агент может читать и анализировать, но не может менять, отправлять, удалять или публиковать.

**Recurrent task** — повторяющаяся задача по расписанию.

**Retrieval** — поиск нужных фрагментов в большом корпусе данных перед передачей модели.

**Routines** — повторяемые рабочие процедуры или регулярные задачи, которые можно поручить агенту: мониторинг, проверка ссылок, сортировка файлов, обновление таблицы.

**Scraping** — автоматическое извлечение данных со страниц сайтов. Может быть полезным, но требует осторожности и approval.

**Sensitive data** — чувствительные данные: контакты источников, персональные данные, неопубликованные материалы, переписка, документы расследований, credentials, source lists.

**Skill** — инструкция о том, как выполнять определённый тип задачи. Skill не является tool: он не делает действие сам, а описывает процедуру.

**SKILL.md** — markdown-файл со skill: когда применять, входные данные, процедура, tools, формат результата, критерии качества, запреты.

**System instruction** — базовая инструкция агента: роль, стиль работы, правила, ограничения, критерии качества.

**Timeline extraction** — извлечение хронологии: событий, дат, последовательности действий.

**Token** — единица текста для модели: слово, часть слова или знак.

**Tool** — конкретная возможность агента: поиск, чтение PDF, OCR, анализ таблицы, запуск кода, доступ к Drive, календарю, CMS.

**Tool calling** — способность модели выбрать и вызвать подходящий tool.

**Worklog** — рабочий журнал конкретной сессии: что сделали, какие решения приняли, что не сработало, что важно для продолжения.

**Workflow** — повторяемая последовательность действий: входные данные → шаги → проверка → результат.

---

**Автоматизация** — связка LLM, tools, access и расписания. Нужна для регулярных задач и повторяемых процедур, но не отменяет человеческий контроль.

**Агент** — LLM, которой дали tools, access и право действовать в цикле.

**Аудит действий** — возможность понять, что агент сделал: какие файлы открыл, какие tools вызвал, какие изменения предложил.

**Версионирование** — сохранение истории изменений и промежуточных версий.

**Граница человека** — место в workflow, где агент обязан остановиться и передать решение человеку.

**Инструменты записи** — tools, которые меняют состояние: редактируют документ, переименовывают файлы, отправляют письма, публикуют.

**Инструменты чтения** — tools, которые только читают и структурируют: поиск, чтение PDF/DOCX/CSV, OCR, расшифровка, анализ архива.

**Коннектор** — мост между агентом и внешним сервисом: Drive, календарём, GitHub, базой данных, почтой, CMS.

**Контур работы** — среда, где действует агент: локальная папка, облачная папка, репозиторий, Google Docs, CMS, терминал.

**Логирование** — запись хода работы агента: что проверено, какие источники использованы, где возникла ошибка.

**Минимальный доступ** — принцип: агент получает только те данные и права, которые нужны для конкретной задачи.

**Периметр безопасности** — заранее заданная область, внутри которой агент может работать: конкретная папка, read-only access, тестовая среда.

**Песочница** — ограниченная среда выполнения, где код или агент не должны затрагивать остальную систему.

**Публикация** — высокорисковое действие: отправка материала наружу, обновление сайта, пост в соцсетях, письмо источнику.

**Редакционный риск** — риск от сочетания: ошибочный вывод + реальные tools + access к sensitive data + автономное действие без редактора.

**Сценарий агента** — описание задачи как процесса: цель, данные, tools, шаги, результат, ограничения, точки проверки.

**Точка остановки** — условие, при котором агент обязан прекратить автономную работу и спросить человека.

**Шаг автономии** — одно действие внутри agent loop: найти источник, открыть файл, извлечь факты, сравнить версии, сохранить результат.

---

Prague School media

www.pragueschool.media

*Май, 2026*