Обзор разработки ИИ-агентов на Metabot

Введение

Metabot — это универсальная **low-code / full-code платформа**, объединяющая возможности чат-ботов, интеграций и backend-автоматизации.
Она служит коммуникационным и интеграционным ядром, где можно проектировать бизнес-процессы, собирать ассистентов и подключать внешние сервисы.

На этой основе создан **Metabot Agent Stack (MAS)** — фреймворк для разработки **интеллектуальных ассистентов и мультиагентных систем**, где можно:

- подключать любые LLM;
- интегрировать базы знаний;
- строить цепочки reasoning;
- и управлять всем этим без кода.

MAS использует компоненты платформы (скрипты, атрибуты, API-шлюзы, базу знаний, трассировку) и расширяет их возможностями работы с **LLM, RAG-поиском и reasoning-цепочками**.

Именно связка **Metabot + MAS** превращает платформу из конструктора чат-ботов в **инфраструктуру для создания ИИ-агентов**, способных взаимодействовать с данными, людьми и процессами.

Основные компоненты

Все компоненты вместе формируют Metabot Agent Stack (MAS):

Компонент	Назначение

Run asynchronous API-request	Команда для выполнения асинхронного API-запрос с замыканием состояния на текущей команде и ожиданием результата выполнения запроса. Через нее происходит обращение к LLM.
LLMClient	Универсальный компонент для работы с языковыми моделями (OpenAI, Claude, YandexGPT, GigaChat и др.) через API. Поддерживает синхронные и асинхронные запросы.
Snippets	Параметризуемые фрагменты кода и настроек, которые позволяют конфигурировать агента под конкретный проект.
Knowledge Base	База знаний на PostgreSQL с поддержкой PgVector. Позволяет хранить документы, разрезать их на чанки, векторизировать и выполнять семантический поиск.
KnowbaseSearch	Компонент для RAG-поиска (Retrieval Augmented Generation). Ищет релевантные фрагменты знаний и формирует контекст для LLM.
Custom Tables	Пользовательские таблицы для хранения данных, промптов и контекстов.
Txt Importer	Скрипт для импорта текстовых файлов, разрезки на чанки и векторизации (эмбеддинги).
Tracing	Встроенная трассировка запросов и ответов LLM — для отладки и анализа reasoning-процессов.
API/Webhooks	Встроенные средства интеграции с внешними системами и сервисами.

Как работает агент в Metabot

Создание ИИ-агента в **Metabot** строится вокруг **сценария (script)**. Сценарий описывает последовательность шагов взаимодействия с пользователем и внешними системами.

1. Пользователь пишет сообщение в чат (Telegram, WebChat, WhatsApp).
2. Скрипт получает это сообщение и вызывает **LLMClient** внутри команды **Выполнить асинхронный API-запрос** — формируется запрос (prompt) к языковой модели.
3. Ответ модели возвращается в нужную точку сценария и используется для следующего шага.

Пример логики:

> Пользователь задаёт вопрос → скрипт вызывает LLMClient → LLM обращается к базе знаний → формируется ответ → отправляется пользователю.

Таким образом, агент — это **цепочка вызовов** между пользователем, памятью, моделью и логикой сценария.

Работа с базой знаний

Metabot поддерживает **векторную базу знаний** с возможностью гибридного поиска.

- **Импорт данных:** файлы (PDF, TXT, DOCX) можно загружать через Txt Importer, который разбивает их на чанки и векторизирует.
- **Поиск:** компонент KnowbaseSearch позволяет выполнять семантический поиск (по смыслу), а компонент кастомной таблицы позволяет реализовать поиск по точному совпадению.
- **Контекст:** найденные фрагменты подаются в LLM, обеспечивая точные и осмысленные ответы.

Конфигурация и хранение настроек

Все параметры агента — ключи API, идентификаторы моделей, пути к данным — сохраняются в **атрибутах** бота. Это делает систему безопасной и удобной для тиражирования проектов.

Для каждого ассистента можно хранить:

- параметры LLM;
- промпты и конфигурации;
- параметры подключения к базам знаний;
- контексты для разных режимов общения.

Отладка и трассировка

Для разработки ИИ-агентов крайне важно видеть, **почему** модель дала тот или иной ответ. В **Metabot** встроена система трассировки:

- журнал запросов и ответов;
- сохранение контекста промптов;
- визуальная отладка reasoning-потока.

Это помогает быстро улучшать промпты и повышать точность ответов.

Интеграции и API

Metabot легко соединяется с внешними системами:

- CRM
- ERP и LMS
- DataLens, Google Sheets, SQL-базы
- API партнёров и внутренних сервисов

Встроенные Low-code возможности позволяют строить интеграции без необходимости писать сложный backend.

□□ Полезные уроки и материалы

□□ *Смотрите также:*

- Работа с LLMClient
- Сниппеты и конфигурация агентов
- База знаний и семантический поиск
- Трассировка и отладка
- Интеграция с внешними API

Версия #3
Artem Garashko создал 2025-11-12 11:53:27 UTC
Artem Garashko обновил 2025-11-12 14:03:47 UTC