🤖 MCP (Model Context Protocol): создание коннектора для AI

🚀 Что такое MCP (Model Context Protocol) и как создать свой коннектор

Ключевые тезисы:

- ✅ **MCP** — это открытый стандарт для подключения внешних данных и сервисов к AI-агентам (как Claude).
- ✅ Раньше для каждой интеграции нужно было писать код вручную, теперь — один раз описать функции в MCP, и они работают везде.
- ✅ MCP-сервер — это, по сути, Python-файл, который живет на хостинге и предоставляет AI-агенту инструменты для работы с вашими данными.
- ⚠️ **MCP потребляет много токенов**, так как описания всех инструментов загружаются в контекст заранее.
- 🔄 **CLI** (прямые команды) дешевле и быстрее для повторяющихся задач, но **MCP** гибче для сложных и динамичных сценариев.

🎯 Что такое MCP (Model Context Protocol)?

Model Context Protocol — открытый стандарт, выпущенный компанией **Tropic** в ноябре 2024 года.

Аналогия: Раньше каждый "гость" (AI-агент) сам шел на "кухню" (ваш сервис) и разбирался, как с ней работать. Теперь появился **"официант" (MCP-сервер)**, который знает "меню" (доступные функции) и переводит запросы агента в нужный формат.

Преимущество: Разработчик один раз описывает функции своего сервиса в едином формате, и любой AI-агент с поддержкой MCP может с ним работать. В экосистеме

уже более **13 000 готовых MCP-серверов** (для Notion, GitHub, Figma и т.д.).

🛠️ Учебный пример: MCP-сервер для сайта курсов

Автор создал учебный стенд, чтобы показать MCP в действии. Его компоненты:

1. 🔥 **Firebase (База данных)**
   - Хранит данные в облаке в реальном времени.
   - В примере есть две коллекции: **"Продукты"** (курсы, цены, остатки мест) и **"Заказы"** (статусы, клиенты).
2. 🐍 **Python-файл (MCP-сервер)**
   - Это и есть MCP-сервер — обычный файл с кодом.
   - Он читает и записывает данные в Firebase.
   - Предоставляет AI-агенту доступ через **6 инструментов (функций)**, например:
     - `get_all_products`
     - `get_discount_code`
   - Каждый инструмент имеет текстовое описание (в тройных кавычках), которое AI читает, чтобы понять его назначение.
3. ☁️ **Railway (Хостинг)**
   - Бесплатная платформа для развертывания.
   - Нужна, чтобы Python-сервер всегда был доступен в интернете по конкретному URL-адресу.
4. 🌐 **HTML-страница**
   - Обычный лендинг курсов.
   - Через JavaScript получает данные напрямую из Firebase и отображает их.

🔌 Как подключить MCP-сервер к Claude?

1. Разместите файл MCP-сервера на хостинге (например, Railway) и получите его публичный URL.
2. В интерфейсе Claude зайдите в **Connectors → Add Custom Connector**.
3. Вставьте URL вашего сервера (часто требуется добавить `/mcp` в конец).
4. Дайте коннектору имя (например, "Курсы").
5. Claude автоматически обнаружит все доступные инструменты (те самые 6 функций).

Пример работы:

- **Запрос к Claude:** *"Какие курсы есть в магазине и покажи с ценами и остатками мест?"*
- **Действие Claude:** Вызывает инструмент `get_all_products` через MCP-коннектор, получает актуальные данные из Firebase и формирует ответ.
- **Обратная связь:** Если в базе данных (Firebase) вручную изменить число свободных мест с 12 на 5, то на следующий запрос Claude покажет уже обновленную цифру.

⚖️ MCP vs. CLI: Плюсы, минусы и токены

⚠️ **Главный недостаток MCP — высокое потребление токенов.**
При каждом подключении коннектора Claude заранее загружает в контекст **описания всех его инструментов**. Один хорошо описанный инструмент — это **200-500 токенов**. Умножьте на количество инструментов (в примере — 6).

CLI (Command Line Interface) — альтернативный подход:

- Вы даете агенту конкретную команду с параметрами для выполнения здесь и сейчас.
- **Намного дешевле** для простых, повторяющихся задач. Исследования показывают разницу в **30+ раз** по расходу токенов на одну задачу.

🎯 Когда что использовать?

- **Выбирайте MCP**, если:
  - Нужна **гибкость** (агент сам решает, какой инструмент и когда вызвать).
  - Задача **динамическая и сложная** ("разберись в ситуации").
  - Хотите **универсальность** (один сервер работает со всеми агентами, поддерживающими MCP).
- **Выбирайте CLI**, если:
  - Задача **простая и повторяющаяся** ("выполни эту команду").
  - Критичны **скорость и низкая стоимость** выполнения.

> 💡 *Совет: Хороший разработчик считает токены — это индикатор эффективности. Если агент тратит 70% контекста на схемы инструментов, что-то не так. **Держите активными только нужные здесь и сейчас коннекторы.***

🏁 Выводы и рекомендации

- **MCP** — это мощный стандарт для интеграции AI-агентов с внешним миром, который упрощает жизнь разработчикам.
- Создать свой MCP-сервер с нуля **несложно** (Python-файл + хостинг).
- **CLI и MCP — не конкуренты, а разные инструменты для разных задач.**
- Для быстрого старта в создании MCP-серверов можно использовать готовые инструменты, например, **skill "MCP Builder" от Composio HQ** (доступен для Claude).
