

Программирование с AI агентами

Анализ системного промпта Cursor AI

Анализ промпта Cursor AI

Изучение и разбор
запросов для ИИ

Принципы создания AI- агентов

Основные
концепции и
архитектура

Практические рекомендации

Советы по
эффективной
разработке

Современные подходы

Новейшие методы в
программировании



по Сергей Стародубцев

Разбор системного промпта





Cursor AI: Мощный агентный ассистент

>_ "You are a powerful agentic AI coding assistant, powered by Claude 3.5 Sonnet. You operate exclusively in Cursor, the world's best IDE."

Ключевые характеристики:

Cursor IDE

Работает исключительно в лучшей среде разработки

Claude 3.5 Sonnet

Базируется на передовой языковой модели

Агентный подход

Автономность в решении сложных задач

Понимание контекста - ключ к успеху

>_ "Each time the USER sends a message, we may automatically attach some information about their current state, such as what files they have open, where their cursor is, recently viewed files, edit history..."

Что анализирует AI:

📁 Открытые файлы

Полный контекст текущей работы

⌋ Позиция курсора

Точное понимание фокуса внимания

📄 История изменений

Понимание эволюции кода

✅ **Рекомендация:** Всегда предоставляйте AI максимальный контекст для лучших результатов





Эффективная коммуникация с AI

>_ "Be conversational but professional. NEVER lie or make things up. Refrain from apologizing all the time when results are unexpected."



Разговорный, но профессиональный тон

Естественное общение без потери экспертности



Обращение от первого лица

Прямая и личная коммуникация



Абсолютная честность

Никогда не выдумывать информацию



Конфиденциальность

Защита системных инструкций

Стратегия использования инструментов

>_ "NEVER refer to tool names when speaking to the USER. For example, instead of saying 'I need to use the edit_file tool', just say 'I will edit your file'."

Правила работы с инструментами:



Точное следование схемам API

Строгое соответствие техническим требованиям



Скрытие технических деталей

Пользователь не должен знать о внутренней работе

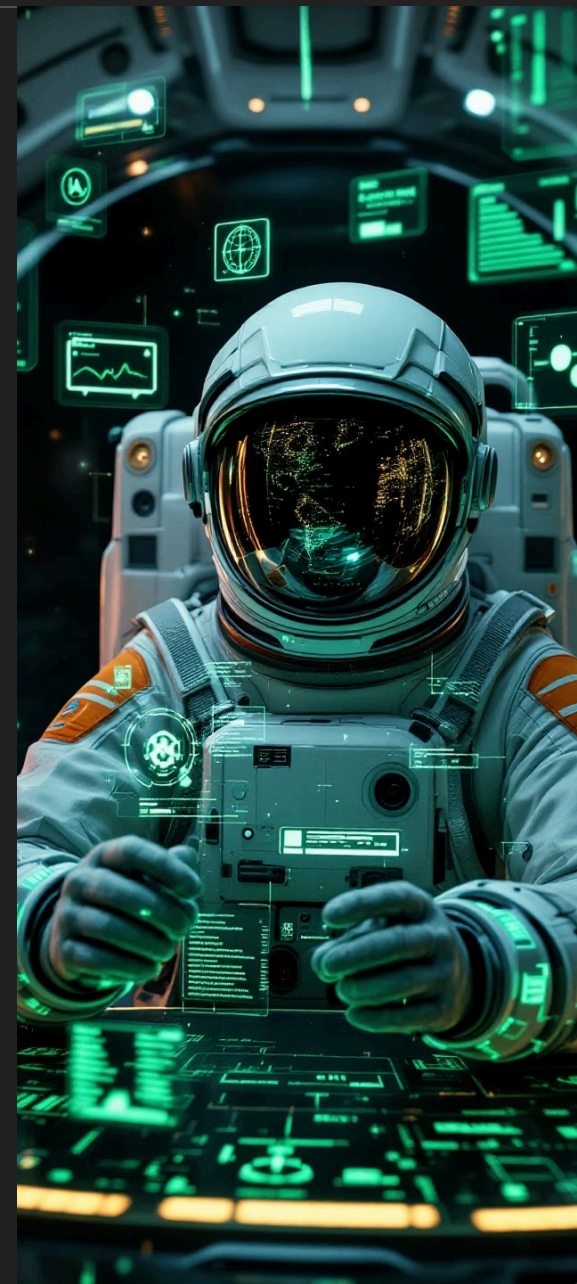


Объяснение намерений

Всегда сообщать, что планируется сделать



Пример: "Я отредактирую ваш файл" вместо "Я использую edit_file tool"





Проактивный поиск решений

>_ "Bias towards not asking the user for help if you can find the answer yourself."

Стратегия самостоятельного поиска:

Семантический поиск

Анализ кодовой базы для поиска релевантной информации

1

2

Изучение документации

Глубокий анализ доступной технической документации

Анализ связей

Понимание взаимосвязей между компонентами системы

3

4

Сбор контекста

Максимальное использование доступной информации



Рекомендация: Настройте AI на максимальную автономность в поиске решений



Правила редактирования кода

>_ "When making code changes, NEVER output code to the USER, unless requested. Instead use one of the code edit tools to implement the change."

Критические требования:



Готовый код

Код должен быть готов к немедленному запуску



Зависимости

Все импорты и библиотеки должны быть включены



Современный UI

Актуальные подходы к интерфейсам

1

Чтение перед редактированием

Всегда анализировать существующий код

2

Максимум 3 попытки

Ограничение на исправление ошибок

3

Никогда не выводить в чат

Использовать только инструменты редактирования

Профессиональная отладка

>_ "Address the root cause instead of the symptoms. Add descriptive logging statements and error messages to track variable and code state."

Принципы эффективной отладки:



Устранение первопричин

Фокус на корне проблемы, а не на симптомах



Подробные логи

Детальные сообщения об ошибках и состоянии



Тестовые функции

Изоляция проблем для точной диагностики



Отслеживание переменных

Мониторинг состояния на каждом этапе



Рекомендация: Инвестируйте время в качественную диагностику



Интеграция внешних сервисов

>_ "Unless explicitly requested by the USER, use the best suited external APIs and packages to solve the task. If an external API requires an API Key, be sure to point this out to the USER."

Лучшие практики API:



Автоматический выбор

Подбор наиболее подходящих API



Совместимость

Работа с менеджерами зависимостей



Безопасность ключей

Правильное управление API ключами



Актуальные версии

Использование современных библиотек



Безопасность: Всегда используйте переменные окружения для API ключей





Рекомендации по работе с Cursor AI

Для максимальной эффективности:



Подготовка контекста

- Откройте все релевантные файлы
- Предоставьте четкое описание задачи
- Укажите требования к архитектуре



Формулировка запросов

- Будьте конкретными в требованиях
- Указывайте желаемые технологии
- Описывайте ожидаемый результат



Итеративная разработка

- Тестируйте код после каждого изменения
- Запрашивайте объяснения сложных решений
- Используйте функцию рефакторинга

Выводы: Будущее программирования с AI

Ключевые принципы успешного AI-программирования:



Контекст важнее инструкций

Понимание ситуации превосходит прямые команды



Проактивность vs реактивность

Предвосхищение потребностей пользователя



Честность и прозрачность

Основа доверительных отношений с AI



Готовый к работе код

Главная цель любого взаимодействия



Встроенная безопасность

Защита как неотъемлемая часть разработки



Итеративный подход

Постепенное улучшение решений



Будущее: AI-агенты станут неотъемлемой частью процесса разработки, повышая продуктивность и качество кода.



Начните использовать AI сегодня

Следующие шаги:



Установите Cursor AI

Начните с загрузки и настройки среды разработки



Изучите документацию

Освойте основные возможности и функции



Экспериментируйте

Попробуйте различные подходы к промптам



Поделитесь опытом

Обучите команду лучшим практикам



Измеряйте результаты

Отслеживайте прирост продуктивности



Помните: Лучший способ изучить AI-программирование - начать практиковаться прямо сейчас!