

Память проекта: продолжать, а не вспоминать

Репозиторий контекста, правила и передача работы между сессиями



Я не хочу объяснять всё заново

Новая сессия полезна, если она продолжает работу с актуального состояния.



Анна Федина

CIO · Автор канала

У меня одновременно идут разработка, документы, личный бренд и рутина. В длинной переписке легко найти удачную идею и трудно понять, какая версия уже согласована. Поэтому я храню устойчивый контекст отдельно от чата.

В моей системе есть общий индекс, карточки направлений и собственные папки проектов. В карточку попадает решение и ссылка на результат. Код, исходные фото и большие документы остаются там, где с ними действительно работают.

Четыре разных вида информации

Правила отвечают на вопрос «как работать». Состояние - «что уже сделано». Источник - «на чём основан вывод». История - «почему решение изменилось». Если смешать всё в один файл, каждое продолжение начинается с раскопок.

Реальный пример: для AI-серии важно знать, что выпуск 1 согласован, где лежит финальный PDF и какие части ещё требуют проверки. Фраза «делали руководство» не помогает выбрать файл.

В практикуме я использую учебный проект demo-registry. Персональные данные и рабочие секреты ему не нужны. Понадобятся папка, текстовый редактор и, по желанию, Git для истории изменений.

Результат: компактная память одного проекта и проверка, что следующая сессия выбирает нужный файл без повторного пересказа всей истории.

Карта вместо одной огромной папки

У каждого файла должна быть понятная роль. Начать можно локально, без публикации репозитория.

```
demo-registry/
  AGENTS.md
  CONTEXT_INDEX.md
  projects/registry/CONTEXT.md
  decisions/001-number-format.md
  evidence/acceptance-demo.txt
  src/
  tests/
```

AGENTS.md содержит рабочие соглашения. Индекс даёт маршрут к нужному проекту. CONTEXT.md хранит текущий статус и следующий шаг. Решение о формате номера живёт отдельно: у него есть причина, дата и затронутые проверки.

Заполненный фрагмент индекса

Задача: регистрация учебного объекта → projects/registry/CONTEXT.md.
 Правило номера → decisions/001-number-format.md.
 Проверка текущей версии → evidence/acceptance-demo.txt.
 Исходный код → src/; проверочные примеры → tests/.

Индекс не должен автоматически затягивать всё дерево в контекст. Для правки подписи поля не нужны история камеры и планы личного бренда. Агент сначала определяет задачу, затем открывает относящиеся к ней материалы.

Что оставить за пределами общей памяти

Исходники частной переписки, ключи, личные фото и большие логи не нужны в общем репозитории. В карточке достаточно безопасного описания результата и адреса локального отчёта. Доступность этого адреса на другом устройстве проверяется отдельно.

Проверка: по каждой ссылке открывается существующий файл. Если результат ещё не создан, рядом стоит «план», а не «готово».

Правило и факт не заменяют друг друга

Инструкция «проверь код» не доказывает, что сегодняшняя версия прошла проверку.

Пример AGENTS.md

Работай только с учебными данными. Перед изменением прочитай контекст текущего проекта. Сохраняй чужие правки. После изменения правила номера запусти соответствующие тесты. В отчёте укажи проверку и ограничения. Не публикуй данные из этой папки.

Это пример соглашений, а не настройка технических разрешений. Запреты доступа обеспечиваются возможностями среды и правами, а текст помогает агенту работать последовательно. Секрет в инструкции всё равно остаётся секретом.

Как файлы находят разные инструменты

Codex загружает AGENTS.md по своей иерархии. Claude Code использует CLAUDE.md; в актуальных версиях также есть условная поддержка AGENTS.md. Если проекту нужны оба инструмента, я проверяю фактически загруженные инструкции, а не предполагаю, что одинаковые имена гарантируют одинаковое поведение.

```
# CLAUDE.md: вариант явного импорта
@AGENTS.md
```

Так можно держать общие соглашения в одном месте. Импорт - возможность Claude Code; переносить этот синтаксис в любой другой инструмент без проверки нельзя. Правила загрузки зависят от версии и настроек.

Промпт для проверки входа

Прочитай инструкции demo-registry и назови файлы, которые фактически использовал. Затем найди источник правила серийного номера. Не меняй файлы. Если видишь противоречие между правилом и кодом, покажи оба места и объясни, какого решения не хватает.

Ответ модели сверяю с доступным журналом загрузки. В Claude Code импорт CLAUDE.md можно проверить через /context → Memory files. Это проверка конкретного инструмента; один устный отчёт агента не доказывает автоматическую загрузку.

OpenAI: загрузка инструкций проекта

Claude Code: память и импорт инструкций

Карточка, по которой можно продолжить

Я фиксирую состояние после завершённого этапа и связываю его с проверенной версией.

Цель: принимать номер с пробелами по краям.
Решение: trim + upper до проверки уникальности.
Изменено: src/serial.py; tests/test_serial.py.
Проверка: учебный набор normal, duplicate, empty.
Доказательство: evidence/acceptance-demo.txt, версия demo-v2.
Не проверено: конкурентные запросы и рабочая база.
Следующий шаг: проверить дубль после нормализации.

Здесь вымышленные пути и имена тестов: это образец заполнения. В реальной карточке указываю существующий файл, команду, дату и версию. После изменения кода старый успешный тест относится к старому состоянию.

Промпт: завершить этап

Сохрани карточку результата для demo-registry. Отдели сделанное от предложенного. Для каждого утверждения о готовности укажи файл и выполненную проверку. Запиши не более пяти устойчивых решений и один следующий шаг. Не копируй переписку, большой вывод инструментов и секреты. Если проверки не было, напиши это явно.

Проверка другой сессией

Откройте новый диалог и дайте только индекс, карточку и задачу. Попросите назвать актуальную версию и продолжить следующий шаг. Если приходится пересылать десять прежних сообщений, в карточке не хватает опорного факта или ссылки.

Короткая запись полезна, пока по ней можно восстановить доказательство. Удалить все подробности и оставить «всё работает» - потерять контекст, а не сжать его.

Сжатие, которое можно проверить

Из исследования SoL-Pi я беру идею возвращаться к полному доказательству по адресу, а не пересылать его каждый раз.

Большой журнал испытаний можно сохранить целиком, а в рабочую карточку вынести сбой, входной пример и место в отчёте. Следующий агент получает компактный вход и при необходимости открывает нужный фрагмент. Само расширение SoL-Pi для этого учебного приёма не требуется.

Ошибка: подробный, но устаревший контекст

В карточке: «номер состоит только из цифр». В текущем решении: «разрешены буквы и дефис». Агент честно следует старому файлу и ломает новый сценарий. Исправление: пометить отменённое решение, указать заменяющее и проверить тесты, зависевшие от старого правила.

Промпт: найти противоречия

Сверь CONTEXT.md, решение 001 и реализацию проверки номера. Верни только расхождения: утверждение / источник / действующее поведение / затронутый сценарий. Не выбирай победителя только по дате файла. Если статус решения неясен, сформулируй один вопрос владельцу; независимые проверки выполни сразу.

Дата изменения файла не равна дате принятия решения: документ могли отформатировать позже. Поэтому важны явный статус и причина замены. Ссылку на старое решение сохраняю для объяснения истории, но убираю его из активного маршрута.

Как оценить пользу

Сравните время восстановления задачи, ошибочно выбранные версии и число повторных чтений на нескольких похожих продолжениях. Экономии токенов можно заявлять только при наличии измерений, а не по длине карточки.

Исследование: SoL-Pi, механизмы работы с наблюдениями

Практикум: продолжить без пересказа

За один подход соберите память небольшой задачи и попробуйте передать её новому исполнителю.

Шаги задания

1. Выберите один действующий результат: документ, функцию или публикационный пакет. 2. Отделите правила от текущего состояния. 3. Запишите индекс из трёх-пяти ссылок. 4. Создайте карточку с подтверждением проверки. 5. Откройте новый сеанс и поручите следующий шаг.

Шаблон карточки
Цель и граница: ...
Принятое решение и основание: ...
Актуальный файл / версия: ...
Что проверено / где доказательство: ...
Что осталось неизвестным: ...
Следующее действие: ...

Промпт: принять передачу

По приложенному индексу восстанови состояние задачи. Назови актуальный результат и его статус. Проверь доступность источника проверки. Затем предложи минимальный следующий шаг и укажи, какие файлы нужны именно для него. Не считай недоступный отчёт доказательством успеха и не загружай все направления проекта.

Готово, если исполнитель выбрал правильную версию, нашёл подтверждение, заметил ограничения и не повторил выполненную работу. Если ссылки доступны только вам, подготовьте разрешённые рабочие копии; весь личный диск передавать не нужно.

Мой следующий шаг - выбрать исполнителя для этой задачи. В выпуске 5 разбираю модели, независимые подзадачи и стоимость принятого результата.

Сохраните карточку рядом с результатом. В общий контекст переносите только устойчивые сведения, которые действительно понадобятся повторно.

ИТ-Коктель с Анной Феединой

Про ИТ, управление, людей и жизнь вокруг работы.
Выберите площадку, на которой вам удобно.

Telegram

Заметки, обсуждения и новые материалы

[Открыть >](#)

ВК Видео

Выпуски и короткие фрагменты

[Открыть >](#)

Дзен

Тексты и видео

[Открыть >](#)

RuTube

Разговоры и выпуски подкаста

[Открыть >](#)

Сетка

Профессиональные темы и общение

[Открыть >](#)

Яндекс Музыка

ИТ-коктейль в аудиоформате

[Открыть >](#)

Названия площадок и карточки кликабельны.

