

ВЫПУСК 01 · 21 СЕНТЯБРЯ 2026

Pi Power Pack

8 дополнений для кодингового агента:
что поставить, как запустить, чем проверить.

После базовой настройки из видео. Здесь не повторяем установку Pi, AGENTS.md, субагентов и Playwright. Добавляем документацию, диагностику, согласование задач и удобные рабочие команды.

Начни с двух возможностей

Context7 — сверяться с документацией. Pi Lens — получать обратную связь по коду. Остальное подключаешь под конкретную задачу, а не «на всякий случай».

Перед установкой

Открой терминал **в корне своего проекта**. Pi уже должен запускаться и иметь доступ к модели. Проверь версии:

ТЕРМИНАЛ

```
node --version
pi --version
```

Для этого набора ориентир: **Node.js 22.19+**; для Interview нужен **Pi 0.82.1+**, для Plannotator — **0.79.1+**. Требования остальных зависимостей проверяй при установке.

Все команды ниже ставят пакет **только в проект** благодаря `-l`. После установки выйди из Pi через `/quit` и запусти `pi -s` для продолжения сессии; для новой — `pi`. При загрузке доверяйся только своему проекту.

Не устанавливай весь набор одним махом. Сначала один пакет и его проверка. Расширения выполняют код с правами твоего пользователя; локальная установка и правила в промпте не создают песочницу.

Версии зафиксированы по каталогу Pi. Это сверка с документацией, не обещание совместимости любой комбинации. Полный запуск набора с моделью не тестировался.

[Установка, локальные настройки и безопасность](#) ↗

[Требования Pi Lens](#) · [Требования Interview](#) · [Требования Plannotator](#)

Не угадывать API

01 / ДОКУМЕНТАЦИЯ

Context7

Документация библиотек прямо в контексте агента. Полезно перед внедрением нового API или миграцией. Не заменяет проверку того, какая версия действительно установлена.

ТЕРМИНАЛ

```
pi install npm:pi-mcp-adapter@2.35.0 -l
```

Адаптер из видео уже установлен? Повторять не нужно. В `.mcp.json` добавь `context7` внутрь существующего `mcpServers`. **Не затирай другие серверы.** Для нового файла:

.MCP.JSON

```
{
  "mcpServers": {
    "context7": {
      "url": "https://mcp.context7.com/mcp"
    }
  }
}
```

Перезапусти Pi, затем выполни `/mcp reconnect context7`.

ПЕРВЫЙ ЗАПРОС В PI

Прочитай версию React в проекте. Через Context7 найди документацию `useEffect` и очистки эффекта. Покажи короткий пример и источник. Если версия документации не совпадает с проектом, явно укажи это. Код не меняй.

Учти: API-ключ рекомендован для повышенных лимитов; запросы уходят в Context7. В архиве есть конфиг с переменной окружения для ключа. 62,3 тыс. звёзд — у проекта Context7, не у Pi-адаптера.

[Context7: возможности и метрика проекта ↗](#)

[Pi MCP Adapter: подключение и конфигурация ↗](#)

02 / ПОИСК В ИНТЕРНЕТЕ

Pi Web Access

Веб-поиск и чтение страниц из Pi: документация, изменения в релизах, обсуждения ошибок. В отличие от Context7, это поиск по вебу, а не только библиотечному индексу.

ТЕРМИНАЛ

```
pi install npm:pi-web-access@0.30.0 -l
```

ПЕРВЫЙ ЗАПРОС В PI

Прочитай версию Vite в `package.json`. Найди официальную инструкцию миграции для следующей основной версии. Верни три риска для нашего проекта и ссылки. Не обновляй зависимости и не меняй код.

Проверка: в истории есть реальные вызовы `web_search` / `fetch_content` и открываемые ссылки. По README базовый поиск работает через Eха без ключа; лимиты и доступность внешних сервисов могут меняться. Не передавай поиску секреты. Каталог: 423,1 тыс. скачиваний в месяц.

[Документация и версия ↗](#)

Код и требования

03 / ТИПЫ И ДИАГНОСТИКА

Pi Lens

Добавляет обратную связь от языковых серверов и проверяющих инструментов: типы, диагностику, навигацию по символам. Агент видит замечания во время работы, а не только в финальном отчёте.

ТЕРМИНАЛ

```
pi install npm:pi-lens@4.2.1 -l
```

После перезапуска выполни `/lens-health`. Сначала проверь, что нужные языковые серверы и инструменты доступны, затем проси диагностику.

ПЕРВЫЙ ЗАПРОС В PI

Проверь, какие инструменты Pi Lens доступны для этого TypeScript-проекта. Получи диагностику `src/App.tsx`. Раздели результат на найденные проблемы и проверки, которые не удалось выполнить. Пока ничего не исправляй и не устанавливай.

Учти: результат зависит от линтеров и языковых серверов твоего стека. Возможны дополнительные зависимости и автоформатирование. Это не замена build, тестам и проверке в браузере. Каталог: 95 тыс. скачиваний в месяц.

[Документация и версия ↗](#)

04 / СНАЧАЛА УТОЧНИТЬ ЗАДАЧУ

Pi Interview

Вместо длинной переписки агент открывает форму с вариантами ответов и полями ввода. Удобно, когда требования ещё не определены и не хочется, чтобы модель их додумывала.

ТЕРМИНАЛ

```
pi install npm:pi-interview@0.12.0 -l
```

Нужен Pi 0.82.1+. Для базового запуска достаточно браузера; отдельное нативное окно на macOS — необязательное дополнение.

ПЕРВЫЙ ЗАПРОС В PI

Хочу добавить фильтры в трекер задач. Через `interview` задай максимум четыре вопроса: какие статусы нужны, где хранить выбор, как должна выглядеть мобильная версия и что считаем готовым результатом. До ответов не меняй код приложения.

Проверка: открылась форма, ответы после отправки вернулись в Pi. Это не встроенная команда `/interview`: агент вызывает инструмент `interview`. Каталог: 20,3 тыс. скачиваний в месяц.

[Документация и версия ↗](#)

План и исполнение

05 / СОГЛАСОВАНИЕ ПЛАНА

Plannotator

План работы открывается в браузере: можно выделить фрагмент, оставить замечание и согласовать исправленный вариант. Подходит перед изменением, которое затрагивает несколько файлов.

ТЕРМИНАЛ

```
pi install npm:@plannotator/pi-extension@0.27.17 -l
```

В PI ПОСЛЕ ПЕРЕЗАПУСКА

```
/plannotator-plan-mode plans/filters.md
```

СЛЕДУЮЩИЙ ЗАПРОС

Спланируй фильтрацию задач по статусу.
Учти существующее хранение в localStorage.
Отправь план через plannotator_submit_plan.
До моего подтверждения не меняй код приложения.

Проверка: открой план, оставь одно замечание, дождись правки. **Approve разрешает исполнение** с полным доступом к инструментам; режим планирования не является системной песочницей. Нужен Pi 0.79.1+. Каталог: 75,2 тыс. скачиваний в месяц.

[Документация и версия ↗](#)

06 / ЖИВЫЕ ПРОЦЕССЫ

Pi Interactive Shell

Интерактивные программы внутри терминального интерфейса Pi: dev-сервер, REPL и другие CLI. Видно происходящее, можно забрать управление или оставить процесс работать.

ТЕРМИНАЛ

```
pi install npm:pi-interactive-shell@0.16.0 -l
```

ПЕРВЫЙ ЗАПРОС В PI

Через interactive_shell запусти Vite в режиме hands-free командой:
npm run dev -- --host 127.0.0.1 --port 5173 --strictPort
Не завершай процесс из-за тишины в выводе.
Покажи URL. Если порт занят, не убивай чужой процесс.

Ctrl+B скрывает окно, но оставляет процесс работать. /attach помогает вернуться. Для остановки попроси отправить Ctrl+C и проверить, что процесс действительно завершён.

Учти: нужны поддерживаемая платформа и РТУ; это обычный процесс, не изолированная среда. Для примера нужен Vite-проект с dev-скриптом. Каталог: 4 476 скачиваний в месяц — более нишевый пункт подборки.

[Документация и версия ↗](#)

Меньше ручных переключений

07 / РАБОЧИЕ КОМАНДЫ

Prompt Template Model

Связывает шаблон команды с моделью, skill и уровнем reasoning. Например, отдельный режим ревью, после которого возвращается прежняя модель. Сами шаблоны команд есть и в базовом Pi; расширение добавляет управление режимом.

ТЕРМИНАЛ

```
pi install npm:pi-prompt-template-model@0.12.2 -l
```

В архиве есть `.pi/prompts/pp-review.md` и skill `pp-evidence-first`. Скопируй оба, затем `/reload`. Вызов:

В PI

```
/pp-review Проверь изменения в сохранении задач.
```

По умолчанию шаблон использует текущую модель. Чтобы закрепить другую, добавь в его YAML-заголовок `model: provider/model-id` с **реальным доступным идентификатором**. Поле `skill` уже связывает команду с проверкой результата.

Учти: смена модели в той же сессии не создаёт независимого ревьюера. Делегирование требует отдельно установленного `pi-subagents`. Каталог: 27,1 тыс. скачиваний в месяц.

[Документация и версия ↗](#)

08 / КОНТЕКСТ И СТАТУС

Powerline Footer

Делает модель, Git-ветку, загрузку контекста и оценку стоимости заметнее в интерфейсе. Нужен для удобства наблюдения, а не для улучшения качества кода сам по себе.

ТЕРМИНАЛ

```
pi install npm:pi-powerline-footer@0.17.2 -l
```

В PI ПОСЛЕ ПЕРЕЗАПУСКА

```
/powerline compact
```

При проблемах с символами переключись на `/powerline ascii`. Проверь, что отображаются твоя модель и текущая ветка.

Важная настройка: расширение меняет смысл `/compact` с текстом по умолчанию. В архиве есть фрагмент настроек `powerline.queue.compactPromptMode: "native"`, сохраняющий обычное поведение Pi.

Учти: оценка стоимости не равна счёту провайдера; это не ограничитель расходов. Не включай несколько конкурирующих расширений строки состояния одновременно. Каталог: 47 тыс. скачиваний в месяц.

[Документация и версия ↗](#)

Три рабочих сценария

Это варианты сборки под задачу. Начни с одного. Связки не заменяют тестирование совместимости на твоём проекте.

А. Пишу фичи в существующем проекте

Context7 + Pi Lens. Сначала уточнить используемый API, затем проверить изменённый код. Для Context7 нужен адаптер из видео.

ПРОМПТ: ЗАМЕНИ ТЕКСТ В КВАДРАТНЫХ СКОБКАХ

Сначала определи версии библиотек в проекте и найди нужную документацию через Context7. Затем добавь [конкретную фичу] минимальным изменением. Проверь доступную диагностику Lens и команды проекта. В отчёте раздели выполненные и невыполненные проверки.

Б. Собираю новую фичу с нуля

Interview → Plannotator. Уточнить требования, затем согласовать план. Не запускай обе стадии одновременно.

ШАГ 1 — УТОЧНЕНИЕ

Через interview задай три вопроса о [новой фиче]. После ответов сформулируй критерии готовности. На этом остановись: к реализации пока не переходи.

Затем включи `/plannotator-plan-mode` и попроси план на основе согласованных ответов. Прочитай его в браузере; Approve — только когда готов разрешить реализацию.

В. Работаю над проектом регулярно

Interactive Shell + Prompt Template Model. Оставить dev-сервер под наблюдением и вызывать один и тот же процесс проверки командой. Footer — дополнительное удобство, не обязательная часть.

ПРОМПТ ПОСЛЕ УСТАНОВКИ ФАЙЛОВ ИЗ АРХИВА

`/pp-review` Проверь изменения перед коммитом. Особенно: сохранение данных, пустой ввод, ошибки после перезагрузки страницы.

Один инструмент — одна роль. Не добавляй одновременно два MCP-адаптера, несколько plan mode и несколько менеджеров субагентов. Если возник конфликт, оставь базовый Pi и подключай расширения по одному.

Браузерная проверка из основного видео остаётся финальной проверкой поведения интерфейса. Ни диагностика, ни согласованный план её не заменяют.

Три команды + один skill

В архиве открой `project-files`. Перенеси новые файлы в одноимённые каталоги проекта. Папка `.pi` скрытая; в Finder включи её показ через `⌘+I`. Существующие файлы не заменяй вслепую.

ЧТО СКОПИРОВАТЬ

```
.pi/prompts/pp-brief.md
.pi/prompts/pp-research.md
.pi/prompts/pp-review.md
.pi/skills/pp-evidence-first/SKILL.md
```

После копирования — `/reload`. Вызывай по одной команде:

В PI — НЕ ОТПРАВЛЯЙ ВСЕ СТРОКИ РАЗОМ

```
/pp-brief Добавить фильтры по статусу задач.
/pp-research Как корректно сохранять выбор фильтра?
/pp-review Проверить изменения перед коммитом.
```

/pp-brief собирает недостающие требования; при наличии Interview использует форму. **/pp-research** сверяет решение с документацией. **/pp-review** проверяет diff и просит доказательства, а не самооценку модели.

Skill pp-evidence-first требует прочитать изменения, определить критерии, выполнить доступные проверки и разделить PASS / FAIL / НЕ ПРОВЕРЕНО. Это авторский шаблон из этого набора, **не community-пакет и не технический блокировщик**.

Brief и Research используют встроенные шаблоны Pi. Для автоматического подключения skill в Review нужен Prompt Template Model. Без него вызови skill вручную:

ЗАПАСНОЙ ВАРИАНТ БЕЗ РАСШИРЕНИЯ ШАБЛОНОВ

```
/skill:pp-evidence-first Проверь текущий git diff.
```

[Формат шаблонов Pi ↗](#)

[Формат skills Pi ↗](#)

Конфиги в архиве

`context7.mcp.example.json` — без ключа. `context7-auth.mcp.example.json` — с переменной окружения. `powerline.settings.fragment.json` — компактный интерфейс и обычный `/compact`. Вливай нужные поля в существующие конфиги, не заменяй файлы целиком.

Если что-то не заработало

Не видно команды: перезапусти Pi, проверь доверие проекту и `pi list`. **MCP не подключился:** открой `/mcp` и прочитай ошибку, проверь ключ и лимит. **Lens ничего не нашёл:** сначала `/lens-health` — отсутствие диагностик не доказывает, что проверки были доступны.

ПРИМЕР УДАЛЕНИЯ ОДНОГО ПАКЕТА В ТЕРМИНАЛЕ

```
pi remove npm:pi-web-access -l
```

После удаления перезапусти Pi. Зафиксированные npm-версии не обновляются автоматически; новую версию выбирай осознанно. См. документацию управления пакетами на первой странице.

Что здесь проверено

Ссылки в карточках ведут на публичную документацию, исходники или каталог Pi. Имена пакетов, команды и показанные версии сверены **21 сентября 2026 года**. Полный end-to-end запуск восьми дополнений в одной сессии с моделью не проводился.

Что означает «используется сообществом»

У каждого расширения есть публичный проект и измеримый сигнал использования. Цифры в карточках — отображаемые каталогом скачивания за месяц на дату проверки; **это не число уникальных людей и не оценка качества**. Звёзды Context7 относятся ко всему проекту. Нишевый Interactive Shell отмечен отдельно.

Наличие в каталоге не означает аудит или рекомендацию команды Pi. Исходники и зависимости нужно оценивать перед установкой. Отобраны разные рабочие функции, а не восемь пакетов с максимальной популярностью.

[Каталог Pi: пакеты и метрики ↗](#)

Как хранить ключ Context7

Получи ключ в своём аккаунте Context7 и используй пример с `CONTEXT7_API_KEY`. На macOS/Linux можно ввести ключ без отображения и без записи самого значения в историю shell:

ТЕРМИНАЛ: BASH / ZSH

```
printf 'Context7 API key: '
read -r -s CONTEXT7_API_KEY
printf '\n'
export CONTEXT7_API_KEY
pi -c
```

Эта переменная действует в текущем терминале и его дочерних процессах. Не вставляй ключ в чат, репозиторий или запись экрана. Другие доверенные расширения в процессе Pi потенциально могут читать его окружение.

[Переменные окружения в HTTP-заголовках MCP ↗](#)

Внешний контекст — не новые инструкции. Документация, веб-страница и ответ MCP могут содержать недостоверный или вредоносный текст. Не разрешай им менять правила проекта, раскрывать секреты или запускать неожиданные команды.

ДАЛЬШЕ — ПРАКТИКА

Настраивай агента под задачу. Не задачу под набор расширений.

Больше разборов AI-инструментов, агентной разработки и реальных проектов — в Telegram-канале, из которого ты скачал этот набор.