Зависимости задач в /task: практика блокировок для команды

25.11.2026 · PDF-версия статьи с сайта cms.dolinais.ru

Зависимости задач — это формальные связи между задачами, при которых одна работа не может считаться выполненной, пока не закрыта другая. В /task такие связи выражены тремя элементами: списком блокирующих задач, парой встречных списков «Ждёт завершения» и «Блокирует», а также замком на карточке, который физически не даёт поставить финальный статус. Ниже — практический разбор: как это устроено, как настроить под свою команду и какие ошибки чаще всего встречаются при внедрении.

Живое демо /task

Ниже — интерактивная демонстрация /task: можно переключать виды, открывать задачи и пробовать действия с блокировками.

ДЕМО /TASK — КАНБАН: ПРИМЕР ДАННЫХ

Задачи в блоке не заданы.

Три сущности, которые нужно различать

Прежде чем настраивать, разведите три близких понятия — от этого зависит, куда смотреть при конфликте.

- **Блокирующая задача** — задача-предшественник, без закрытия которой текущая не может быть завершена. Это роль, а не отдельный объект: любая задача может быть блокиратором для другой.
- **«Ждёт завершения»** — список на карточке текущей задачи: какие задачи должны закрыться раньше неё. Читается сверху вниз: «я жду эти».
- **«Блокирует»** — обратный список: какие задачи не могут закрыться, пока не закроется текущая. Читается так: «эти ждут меня».

Оба списка — это две стороны одной связи. Если вы добавили зависимость из задачи А в задачу В, то в А появится В в «Ждёт завершения», а в В — А в «Блокирует». Дублировать связь с другой стороны не нужно: система отражает её автоматически.

Замок на карточке: что именно он запрещает

Замок — это визуальный и логический индикатор того, что задача заблокирована. Пока хотя бы один блокиратор открыт, кнопка завершения на карточке неактивна, а попытка сменить статус на финальный возвращает предупреждение со ссылкой на незакрытые задачи.

Важно понимать границы запрета, иначе команда начнёт обходить механику:

- Замок блокирует **завершение**, а не работу. Задачу можно брать в работу, комментировать, прикладывать файлы, менять исполнителя — это осознанное решение: параллельная подготовка часто полезна.

- Запрет действует на финальный статус, а не на промежуточные. Перевод в «В работе» или «На проверке» остаётся доступным.
- Замок снимается автоматически, как только закрывается последний блокиратор — ручных действий не требуется.

  Практическое правило: если задачу приходится «закрывать в обход» больше двух раз в неделю, проблема не в замке, а в модели зависимостей — скорее всего, связи выставлены слишком грубо.

Как настроить зависимости: пошагово

1. **Определите цепочку.** Выпишите задачи в порядке фактического выполнения. Зависимость ставится только там, где результат одной действительно является входом для другой.
2. **Добавьте блокиратор.** На карточке задачи-потомка откройте список «Ждёт завершения» и укажите задачу-предшественника. Убедитесь, что в карточке предшественника появилась обратная запись в «Блокирует».
3. **Проверьте замок.** Откройте задачу-потомка: кнопка завершения должна быть недоступна, а рядом — подсказка с перечнем открытых блокираторов.
4. **Закройте блокиратор.** Переведите предшественника в финальный статус и вернитесь к потомку: замок должен исчезнуть без дополнительных действий.
5. **Зафиксируйте договорённость.** Опишите в регламенте проекта, какие типы работ всегда связываются зависимостью (например, «дизайн → вёрстка», «API → интеграция»), а какие — нет.

Типичные сценарии использования

Последовательные этапы производства

Классика: макет → вёрстка → проверка на устройствах. Каждый следующий шаг ждёт предыдущего, и замок не даёт «закрыть» вёрстку до утверждения макета. Это защищает от ситуации, когда задача формально закрыта, а работа фактически переделывается.

Задачи, которые ждут внешнего решения

Если релиз зависит от согласования с заказчиком, создайте отдельную задачу «Согласование», а релизную поставьте в зависимость от неё. Тогда на планёрке видно не абстрактное «ждём», а конкретную задачу с ответственным и сроком.

Параллельные ветки с общей точкой сборки

Несколько задач могут блокировать одну общую — например, интеграционное тестирование ждёт и бэкенд, и фронтенд. В списке «Ждёт завершения» будет две записи, и замок снимется только после закрытия обеих. Это удобно для контроля готовности релиза, но требует дисциплины: слишком широкая точка сборки превращается в узкое место.

Плюсы и ограничения

Что даёт механика на практике:

- **Прозрачность ожиданий.** Видно, кто кого ждёт, без устных уточнений в чатах.
- **Защита от ложного закрытия.** Нельзя отметить готовой работу, которая опирается на незавершённый результат.
- **Честная картина сроков.** При планировании видно критические цепочки, а не только отдельные задачи.
- **Меньше ручных проверок.** Обратные списки формируются автоматически, не нужно вести таблицу связей отдельно.

Ограничения тоже стоит проговорить заранее. Зависимости не заменяют планирование: они фиксируют уже принятое решение о порядке работ. Циклические связи (А ждёт В, В ждёт А) делают цепочку неразрешимой — такие ситуации нужно разрывать, разделяя задачу на части. И, наконец, чрезмерная детализация связей замедляет работу: если каждая мелочь блокирует следующую, команда тратит больше времени на перестановку статусов, чем на саму работу.

Как поддерживать модель в порядке

Зависимости быстро устаревают, если их не пересматривать. Рабочая привычка: раз в неделю просматривать задачи с активным замком и проверять, что блокираторы действительно актуальны. Если блокиратор потерял смысл — связь нужно удалить, а не оставлять «на всякий случай». Полезно также перед стартом спринта пробегать списки «Блокирует» у ключевых задач: так заранее видно, что окажется на критическом пути.

Начните с малого: свяжите зависимости на одном участке проекта, где чаще всего возникают переделки, и оцените эффект через две-три недели. Если замок помогает ловить реальные проблемы, расширяйте практику на остальные цепочки — но не превращайте её в самоцель.

Источник: https://cms.dolinais.ru/zavisimosti-zadach-v-task-praktika-blokirovok-dlya-komandy