Зависимости задач в /task: как блокировки и списки ускоряют проект

13.11.2026 · PDF-версия статьи с сайта cms.dolinais.ru

В управлении проектами редко задачи существуют изолированно. Одна задача часто становится фундаментом для другой: пока не готов дизайн, нельзя верстать; пока не настроен сервер, нельзя тестировать деплой. В **/task** такие связи реализованы через **зависимости задач —** механизм, который автоматически отслеживает порядок выполнения и не даёт закрыть задачу, если её «блокираторы» ещё в работе. В этой статье разберём, как устроены блокирующие задачи, что показывают списки «Ждёт завершения» и «Блокирует», зачем нужен замок на карточке и как запрет завершения помогает избежать хаоса.

Живое демо /task

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

ДЕМО /TASK — КАНБАН: ПРИМЕР ДАННЫХ

Задачи в блоке не заданы.

Что такое зависимости задач и зачем они нужны

Зависимость задач — это связь между двумя задачами, при которой одна не может быть завершена (или начата) до выполнения другой. В /task такая связь устанавливается вручную или автоматически при создании подзадач и проявляется в трёх ключевых элементах:

- **Блокирующие задачи —** задачи, которые должны быть закрыты, чтобы текущая могла считаться выполненной.
- **Список «Ждёт завершения» —** перечень задач, которые блокируют текущую (то есть текущая ждёт их завершения).
- **Список «Блокирует» —** перечень задач, которые зависят от текущей (текущая блокирует их).

Такая двусторонняя видимость помогает каждому участнику понимать контекст: что мешает работе и на что влияет его собственный прогресс. Без подобной прозрачности легко получить ситуацию, когда задача закрыта, а следующий этап не может начаться из-за отсутствия нужного артефакта.

Как работают блокировки: списки «Ждёт завершения» и «Блокирует»

В карточке задачи появились два новых блока. В «Ждёт завершения» перечислены задачи, которые должны быть выполнены раньше. Пока хотя бы одна из них не в статусе «Готово», текущая задача считается заблокированной. В «Блокирует» — наоборот, список задач, которые ждут завершения этой. Это не просто информационные списки: они влияют на поведение интерфейса и логику работы.

Например, вы работаете над задачей «Настроить CI/CD». В «Ждёт завершения» у неё могут быть задачи «Создать репозиторий» и «Определить стейджи». Пока они не закрыты, система не позволит перевести

текущую задачу в статус «Готово». При этом в «Блокирует» у неё будет задача «Задеплоить на прод», которая, в свою очередь, ждёт завершения вашей.

Списки нестатичны: при изменении статуса блокирующей задачи (например, её закрытии) зависимая задача автоматически разблокируется, и в её карточке исчезает предупреждение. Это избавляет от ручного отслеживания и снижает риск ошибок.

Замок на карточке: визуальный индикатор блокировки

Чтобы сразу видеть, что задача не может быть выполнена, в /task используется **замок на карточке**. Он появляется, если у задачи есть хотя бы один незакрытый блокиратор. Замок отображается как в списке задач, так и на канбан-доске, и в карточке задачи. Это простой, но мощный сигнал: не нужно открывать каждую задачу, чтобы понять, почему она стоит на месте.

Замок исчезает автоматически, как только все блокирующие задачи переходят в статус «Готово». При этом история изменений сохраняется: в хронологии задачи можно увидеть, когда появился и снялся замок, какие задачи были блокираторами.

> Важно: замок — не просто иконка. Он напрямую связан с запретом завершения. Если вы попытаетесь закрыть задачу с активным замком, система выдаст предупреждение и не даст это сделать.

Запрет завершения до закрытия блокираторов

Это ключевое правило: задача не может быть переведена в статус «Готово», пока все её блокираторы не закрыты. Такой запрет действует на уровне бизнес-логики и не зависит от прав пользователя. Даже администратор не сможет обойти ограничение без предварительного закрытия или удаления зависимостей.

Зачем это нужно? Представим типичную ошибку: разработчик закрыл задачу «Написать API», хотя на самом деле не завершил интеграцию с базой данных (которая была отдельной блокирующей задачей). В результате тестировщик берёт задачу в работу и тратит время впустую. Запрет завершения заставляет команду явно фиксировать порядок и не пропускать этапы.

Если блокиратор нужно снять по объективным причинам (например, зависимость больше не актуальна), можно удалить связь вручную — но это осознанное действие, а не случайное закрытие.

Как работать с зависимостями: пошаговый пример

Рассмотрим создание задачи «Подготовить релиз» с двумя блокираторами: «Собрать сборку» и «Обновить документацию».

1. Откройте задачу «Подготовить релиз» и перейдите в блок зависимостей.
2. В списке «Ждёт завершения» нажмите «Добавить» и выберите из проекта задачи «Собрать сборку» и «Обновить документацию».
3. Система автоматически добавит в карточки этих задач обратную связь — они появятся в списке «Блокирует».
4. На карточке «Подготовить релиз» появится замок. Теперь при попытке закрыть задачу вы увидите

предупреждение.
5. Закройте «Собрать сборку» и «Обновить документацию». Как только обе задачи перейдут в «Готово», замок исчезнет, и «Подготовить релиз» можно будет завершить.

Такой подход особенно полезен в спринтах, где много параллельных работ. Он дисциплинирует и помогает точно оценивать готовность инкремента.

Преимущества механизма зависимостей

- **Прозрачность**: каждый видит, что именно тормозит задачу, без дополнительных обсуждений.
- **Защита от ошибок**: запрет завершения исключает случайное закрытие задачи при незавершённых блокираторах.
- **Автоматизация**: не нужно вручную отслеживать статусы — система сама снимает замки и обновляет списки.
- **Улучшение планирования**: анализируя цепочки зависимостей, можно выявлять узкие места и перераспределять ресурсы.
- **Единый источник правды**: списки «Ждёт завершения» и «Блокирует» всегда актуальны и синхронизированы.

В итоге зависимости задач в /task — это не просто дополнительная функция, а инструмент, который делает процесс разработки более предсказуемым и управляемым. Попробуйте настроить блокировки в своём следующем проекте — и вы увидите, как исчезают многие организационные проблемы.

Источник: https://cms.dolinais.ru/zavisimosti-zadach-v-task-kak-blokirovki-i-spiski-uskoryayut-proekt