**1. Клонирование репозитория**

Проблемы:

- Нет доступа: `Permission denied`.

- Неправильный URL.

- Требуются SSH-ключи.

Решения:

- Проверьте URL репозитория.

- Настройте ssh-ключи через `ssh-keygen` и добавьте публичный ключ в свой профиль на GitHub/GitLab/Bitbucket.

- Для https — проверьте логин и пароль/токен доступа.

**2. Работа с ветками**

\*\*Проблемы:\*\*

- Надо создать новую ветку: `git checkout -b <branch>`.

- Переключение не работает — “changes not staged for commit”.

\*\*Решения:\*\*

- Сохраните изменения: `git add . && git commit -m "msg"`, либо используйте `git stash` для временного сохранения.

**3. Изменения и коммиты**

Проблемы:

- Нужно отменить изменения:

- в файлах: `git checkout -- <file>`

- в индексе (стадии): `git reset HEAD <file>`

- Случайно закоммитили не то.

- Использовать `git reset --soft HEAD~1` (удалить последний коммит, изменения оставить).

- `git commit --amend` (изменить последний коммит).

**4. Слияние веток (merge)**

Проблемы:

- Конфликт слияния: `Merge conflict in...`

- Многие файлы в конфликте.

Решения:

- Вручную разрешите конфликты: откройте файлы, выберите нужные изменения между `<<<<<<<`, `=======`, `>>>>>>>`.

- После разрешения — `git add .`, затем завершите слияние `git commit` (или `git merge --continue`).

**5. Ребейз (rebase)**

Проблемы:

- Конфликты при ребейзе.

- “You have unstaged changes”

Решения:

- Сохраните изменения (`stash`) перед ребейзом.

- В процессе разрешайте конфликты, после — используйте `git rebase --continue`.

- Если "запутались" — прервать: `git rebase --abort`.

**6. Push на сервер**

Проблемы:

- Коммиты ушли вперёд в origin: `! [rejected] ... -> ... (non-fast-forward)`

- Нет прав на репозиторий.

Решения:

- Обновите локальную ветку: `git pull --rebase`, убедитесь, что все конфликты решены.

- Для force-push: `git push --force-with-lease` (осторожно, опасно — можно переписать историю других!).

**7. Pull и обновление из origin**

Проблемы:

- Конфликты при pull.

- "Diverged branches: have and have not merged"

Решения:

- Разберите конфликты вручную (см. выше).

- Для сложных случаев используйте отдельную рабочую ветку для объединения изменений.

**8. Удаление файлов и коммитов**

Проблемы:

- Удалён не тот файл: восстановите через `git checkout HEAD <file>`.

- Нужно удалить последний коммит: `git reset --hard HEAD~1` (осторожно: удаляет изменения).

**9. Работа с удалёнными репозиториями**

Проблемы:

- Новый remote: `git remote add <name> <url>`

- Смена remote: `git remote set-url <name> <url>`

**10. Восстановление после ошибок**

Проблемы и решения:

- Запутались в коммитах: git reflog — показывает историю HEAD, можно восстановить утерянную ветку/коммит.

- Отменить merge: `git merge --abort`.

- Отменить локальные изменения: `git reset --hard HEAD`.

**11. Частые ошибки**

- “fatal: not a git repository” — не инициализирован git-репозиторий, сделайте `git init` или перейдите в нужную папку.

- “fatal: refusing to merge unrelated histories” — используйте `git pull --allow-unrelated-histories`.

- “Object not found” — возможно, репозиторий повреждён, попробуйте клонировать заново.

**12. Работа с подмножеством файлов (cherry-pick)**

Проблемы:

- Вытянуть только отдельные коммиты: `git cherry-pick <commit>`

**13. Сброс и откат**

Проблемы:

- Сброс ветки к состоянию origin: `git reset --hard origin/<branch>`

- Частичный откат файла: `git checkout <commit> <file>`

**14. Работа с большими файлами**

Проблемы:

- Ограничение размера (например, GitHub >100MB): подключите Git LFS или перепишите историю (`git filter-branch`).

**15. Удаление веток**

Проблемы:

- Локальная: `git branch -d <branch>`

- Удалённая: `git push origin --delete <branch>`

Краткий итог

- Всегда делайте backup перед опасными операциями (`reset`, `rebase`, `force-push`).

- Разделяйте коммиты по смыслу.

- Комментируйте изменения понятными сообщениями коммита.

- Применяйте `reflog` для восстановления проблемных ситуаций.

**1. Не могу отправить (push) коммиты: ошибка “rejected – non-fast-forward”**

Если\*\* вы видите ошибку

`! [rejected] ... -> ... (non-fast-forward)`

то выполните:

git pull --rebase

Решите конфликты, если есть, затем:

git push

**2. Случайно закоммитил лишний файл**

Если закоммитили файл по ошибке, то

- Удалите файл из индекса и закоммитьте изменение:

git rm --cached <имя\_файла>

git commit --amend

git push --force-with-lease

**3. Ошибка merge conflict при объединении веток**

Если видите конфликт:

- Откройте конфликтующий файл, выберите нужный вариант между

`<<<<<<<`, `=======`, `>>>>>>>`

- Добавьте файл:

git add <имя\_файла>

git commit или git merge --continue, если rebase

**4. Случайно удалили важный коммит/ветку**

Если пропал коммит/ветка, то

- Введите:

git reflog

Найдите хэш нужного состояния

git checkout <hash>

Или восстановите ветку:

git branch имя\_ветки <hash>

**5. Хотите “откатить” проект к определённому коммиту**

Если нужен откат ко всем изменениям до выбранного коммита, то

- Для локального отката:

git reset --hard <hash\_коммита>

- Чтобы откатить и на сервере:

git push --force-with-lease

**6. Запутались в незакоммиченных изменениях и не можете переключить ветку**

Если git ругается, что есть незакоммиченные изменения, то

- Зафиксируйте их:

git add .

git stash

git checkout <branch>

Верните изменения:

git stash pop

**7. Надо вернуть только определённый файл к предыдущему состоянию**

Если хотите откатить ОДИН файл, то

- Используйте:

git checkout HEAD <имя\_файла>

**8. Нужно перенести конкретный коммит из одной ветки в другую**

Если хотите “вытащить” отдельный коммит, то

- В ветке-получателе выполните:

git cherry-pick <hash\_коммита>

**9. Не могу клонировать репозиторий: permission denied**

Если ошибка «Permission denied» при клонировании,

- Проверьте ssh-ключи:

ssh-add -l

- Если нет ключа:

ssh-keygen -t ed25519

# Добавьте публичный ключ на GitHub/GitLab

**10. Случайно отправил в паблик приватный файл (секреты/ключи)**

Если такой файл попал в историю git,

- Удалите файл и очистите историю:

git rm --cached путь/к/файлу

git commit -m "Удалён приватный файл"

git filter-branch --force --index-filter

git rm --cached --ignore-unmatch путь/к/файлу"

--prune-empty --tag-name-filter cat -- --all

git push --force --all

**11. Ошибка: “fatal: not a git repository”**

Если в директории нет репозитория,

- Проверьте, что находитесь в папке проекта.

- Если git не инициализирован:

git init

**12. Вдруг надо удалить удалённую ветку**

Если хотите удалить ветку из origin,

- Выполните:

git push origin --delete <branch>

**13. Иногда pull/push требует логин через токен**

Если просят логин/токен,

- Сгенерируйте Personal Access Token в настройках GitHub/GitLab и используйте его вместо пароля.

**14. Не удаётся завершить rebase — rebase застрял**

Если не можете завершить rebase,

- Разрешите конфликты,

git add .

git rebase --continue

- Чтобы отменить rebase:

git rebase --abort

**15. Часть файлов игнорируется при коммите**

Если git не видит файлы при коммите,

- Проверьте `.gitignore` — возможно, нужные файлы указаны для игнорирования.

16. Хотите полностью стереть все свои незакоммиченные локальные изменения\*\*

\*\*Если\*\* изменения не важны,

- Выполните:

git reset --hard HEAD

**17. В удалённом репозитории появилась новая ветка, а локально её не видно**

Если\*не видите новую ветку,

- Получите все ветки:

git fetch

git checkout <branch>

**18. Ошибка “refusing to merge unrelated histories”**

\*\*Если\*\* при merge/pull видите эту ошибку,

- Используйте:

git pull --allow-unrelated-histories

**19. Случайно удалили не тот файл и ещё не закоммитили**

Если удалили файл, но ещё не коммитили,

- Восстановите:

git checkout <имя\_файла>

**20. «Закоммитил мусор», надо исправить сообщение последнего коммита**

\*\*Если\*\* хотите изменить сообщение последнего коммита,

- Выполните:

git commit –amend Введите новое сообщение