Полезные руководства по Git

```
git help -g
```

Поиск по содержанию

```
git log -S'<a term in the source>'
```

Удаленная синхронизация и перезапись локальных изменений

```
git fetch origin && git reset --hard origin/master && git clean -f -d
```

Список всех файлов до коммита

```
git ls-tree --name-only -r <commit-ish>
```

Отмена коммита

```
git update-ref -d HEAD
```

- Про Git, Github и Gitflow простыми словами

Список всех конфликтующих файлов

```
git diff --name-only --diff-filter=U
```

Список всех файлов, измененных коммитом

```
git diff-tree --no-commit-id --name-only -r <commit-ish>
```

Изменения с момента последнего коммита

```
git diff
```

Изменения, выполненные для коммита

```
git diff --cached
```

Альтернатива:

```
git diff --staged
```

Показать подготовленные/неподготовленные файлы для коммита

```
git diff HEAD
```

Все ветки, которые уже соединены с веткой master

```
git branch --merged master
```

Быстрый переход к предыдущей ветке

```
git checkout -
```

Альтернатива:

```
git checkout @{-1}
```

Удалить ветки, которые уже объединены с master

```
git branch --merged master | grep -v '^\*' | xargs -n 1 git branch -d
```

Альтернатива:

```
git branch --merged master | grep -v '^\*\|  master' | xargs -n 1 git branch -d
```

Все ветки и выходящие из них, а также последние коммиты на ветке

```
git branch -vv
```

Отслеживание ветки

```
git branch -u origin/mybranch
```

Удаление локальной ветки

```
git branch -d <local_branchname>
```

Удаление ветки

```
git push origin --delete <remote_branchname>
```

Альтернатива:

```
git push origin :<remote_branchname>
```

Удаление локальной метки

```
git tag -d <tag-name>
```

Удаление метки

```
git push origin :refs/tags/<tag-name>
```

Отменена локальных изменений

```
git checkout -- <file_name>
```

Отмена коммита, создание нового коммита

```
git revert <commit-ish>
```

Отмена коммита, предпочительно для приватных веток

```
git reset <commit-ish>
```

Повтор предыдущего сообщение коммита

```
git commit -v --amend
```

Просмотр истории коммитов для текущей ветки

```
git cherry -v master
```

Изменить автора

```
git commit --amend --author='Author Name <email@address.com>'
```

- Модели ветвления в Git: какую выбрать?

Сброс автора, после того как автор был изменен в глобальной конфигурации

```
git commit --amend --reset-author --no-edit
```

Изменение удаленного URL

```
git remote set-url origin <URL>
```

Получение списка всех удаленных ссылок

```
git remote
```

Альтернатива:

```
git remote show
```

Получение списка всех локальных и удаленных веток

```
git branch -a
```

Получение списка только удаленных веток

```
git branch -r
```

Получение списка изменений файла, а не весь файл

```
git add -p
```

Получение git bash

```
curl http://git.io/vfhol > ~/.git-completion.bash && echo '[ -f
~/.git-completion.bash ] && . ~/.git-completion.bash' >> ~/.bashrc
```

- Git за полчаса: руководство для начинающих

Что изменилось за две недели?

```
git log --no-merges --raw --since='2 weeks ago'
```

Альтернатива:

```
git whatchanged --since='2 weeks ago'
```

Просмотреть все коммиты, сделанные с момента создания ветки мастера

```
git log --no-merges --stat --reverse master..
```

Выбор коммитов по веткам с помощью cherry-pick

```
git checkout <branch-name> && git cherry-pick <commit-ish>
```

Показать ветки, содержащие commit-hash

```
git branch -a --contains <commit-ish>
```

Альтернатива:

```
git branch --contains <commit-ish>
```

Git-алиасы

```
git config --global alias.<handle> <command>
git config --global alias.st status
```

Сохранение текущего состояния отслеживаемых файлов без коммитов

```
git stash
```

Альтернатива:

```
git stash save
```

Сохранение текущего состояния изменений в отслеживаемых файлах

```
git stash -k
```

Альтернатива:

```
git stash --keep-index
```

```
git stash save --keep-index
```

Сохранение текущего состояния, включая неиспользуемые файлы

```
git stash -u
```

Альтернатива:

```
git stash save -u
```

```
git stash save --include-untracked
```

Сохранение текущего состояния с сообщением

```
git stash save <message>
```

Сохранение текущего состояния всех файлов (игнорируются, не отслеживаются и отслеживаются)

```
git stash -a
```

Альтернатива:

```
git stash --all
```

```
git stash save --all
```

Показать список всех данных, сохраненных в тайнике

```
git stash list
```

Прятанье без удаления из списка

```
git stash apply <stash@{n}>
```

Взять файл из тайника

```
git checkout <stash@{n}> -- <file_path>
```

Альтернатива:

```
git checkout stash@{0} -- <file_path>
```

Показать все отслеживаемые файлы

```
git ls-files -t
```

Показать все неотслеживаемые файлы

```
git ls-files --others
```

Показать все проигнорированные файлы

```
git ls-files --others -i --exclude-standard
```

Создание нового рабочего дерева из репозитория (git 2.5)

```
git worktree add -b <branch-name> <path> <start-point>
```

Отслеживание файлов без удаления

```
git rm --cached <file_path>
```

Альтернатива:

```
git rm --cached -r <directory_path>
```

Перед удалением ненужных файлов / каталогов, получить список этих файлов / каталогов

```
git clean -n
```

Удаление неотслеживаемых файлов

```
git clean -f
```

Удаление неотслеживаемого каталога

```
git clean -f -d
```

Обновление всех подмодулей

```
git submodule foreach git pull
```

Альтернатива:

```
git submodule update --init --recursive
```

```
git submodule update --remote
```

Показать все коммиты в текущей ветке

```
git cherry -v master
```

Альтернатива:

```
git cherry -v master <branch-to-be-merged>
```

Переименование ветки

```
git branch -m <new-branch-name>
```

Альтернатива:

```
git branch -m [<old-branch-name>] <new-branch-name>
```

Архивация ветки master

```
git archive master --format=zip --output=master.zip
```

Изменить предыдущий коммит без сообщения об редактировании

```
git add --all && git commit --amend --no-edit
```

Удаление ссылок, ссылающихся на удаленные ветки

```
git fetch -p
```

Альтернатива:

```
git remote prune origin
```

Визуализация дерева версий

```
git log --pretty=oneline --graph --decorate --all
```

Альтернатива:

```
gitk --all
```

Развертывание вложенной папки git в gh-pages

```
git subtree push --prefix subfolder_name origin gh-pages
```

Добавление проекта к репозиторию с использованием поддерева

```
git subtree add --prefix=<directory_name>/<project_name> --squash
git@github.com:<username>/<project_name>.git master
```

Получение последних изменений в репозитории проекта с использованием поддерева

```
git subtree pull --prefix=<directory_name>/<project_name> --squash
git@github.com:<username>/<project_name>.git master
```

Экспорт истории ветки в файл

```
git bundle create <file> <branch-name>
```

Импорт из пакета

```
git clone repo.bundle <repo-dir> -b <branch-name>
```

Получение имени текущей ветки

```
git rev-parse --abbrev-ref HEAD
```

Игнорировать один файл при коммите

```
git update-index --assume-unchanged Changelog; git commit -a; git update-index
--no-assume-unchanged Changelog
```

Запрос по идентификатору в локальную ветку

```
git fetch origin pull/<id>/head:<branch-name>
```

Альтернатива:

```
git pull origin pull/<id>/head:<branch-name>
```

Показать самую используемую метку в текущей ветке

```
git describe --tags --abbrev=0
```

Показать встроенное слово diff

```
git diff --word-diff
```

Не учитывать изменения отслеживаемого файла

```
git update-index --assume-unchanged <file_name>
```

Отмена без изменений

```
git update-index --no-assume-unchanged <file_name>
```

Удаление файлов из `.gitignore`

```
git clean -X -f
```

Восстановление удаленного файла

```
git checkout <deleting_commit>^ -- <file_path>
```

Восстановление файлов к определенному коммиту

```
git checkout <commit-ish> -- <file_path>
```

Список всех алиасов и конфигов

```
git config --list
```

Добавление пользовательских редакторов

```
git config --global core.editor '$EDITOR'
```

Автоматическое исправление опечаток

```
git config --global help.autocorrect 1
```

Проверьте, является ли изменение частью релиза

```
git name-rev --name-only <SHA-1>
```

Помечает ваш коммит, как исправление предыдущего коммита

```
git commit --fixup <SHA-1>
```

Пропустить область во время коммита

```
git commit --only <file_path>
```

Список игнорируемых файлов

```
git check-ignore *
```

Статус игнорируемых файлов

```
git status --ignored
```

Список n последних коммитов

```
git log -<n>
```

Альтернатива:

```
git log -n <n>
```

Открыть все конфликтующие файлы в редакторе

```
git diff --name-only | uniq | xargs $EDITOR
```

Мгновенный просмотр рабочего репозитория в gitweb

```
git instaweb [--local] [--httpd=<httpd>] [--port=<port>] [--browser=<browser>]
```

Просмотр GPG подписи в журнале коммитов

```
git log --show-signature
```

Удаление записи в глобальной конфигурации

```
git config --global --unset <entry-name>
```

Создание новой ветки без истории

```
git checkout --orphan <branch_name>
```

Извлечь файл из другой ветки

```
git show <branch_name>:<file_name>
```

Изменить предыдущие два коммита с интерактивной перестановкой

```
git rebase --interactive HEAD~2
```

Список всех веток — WIP

```
git checkout master && git branch --no-merged
```

Бинарный поиск ошибок

```
git bisect start
git bisect bad
git bisect good v2.6.13-rc2
git bisect bad
git bisect good
git bisect reset
```

Создание списка коммитов и изменений в конкретном файле

```
git log --follow -p -- <file_path>
```

Клонировать ветку

```
git clone -b <branch-name> --single-branch https://github.com/user/repo.git
```

Создание и редактирование новой ветки

```
git checkout -b <branch-name>
```

Альтернатива:

```
git branch <branch-name> && git checkout <branch-name>
```

Отключить цветной вывод git в терминал

```
git config --global color.ui false
```

Настройки цвета

```
git config --global <specific command e.g branch, diff> <true, false or always>
```

Клонировать репозиторий

```
git clone https://github.com/user/repo.git --depth 1
```

Поиск коммитов по всем веткам

```
git log --all --grep='<given-text>'
```

Получение первого коммита в ветке

```
git log master..<branch-name> --oneline | tail -1
```

Показать коммиты по авторам и названию

```
git shortlog
```

Количество коммитов в ветке

```
git rev-list --count <branch-name>
```

Алиас: git undo

```
git config --global alias.undo '!f() { git reset --hard $(git rev-parse --abbrev-ref HEAD)@{${1-1}}; }; f'
```

Добавление примечаний к объекту

```
git notes add -m 'Note on the previous commit....'
```

Показать все git-notes

```
git log --show-notes='*'
```

Применение коммита из другого репозитория

```
git --git-dir=<source-dir>/.git format-patch -k -1 --stdout <SHA1> | git am -3 -k
```

Список запрещенных git коммитов

```
git log --branches --not --remotes
```

Альтернатива:

```
git log @{u}..
```

```
git cherry -v
```

Изменить git конфигурацию

```
git config [--global] --edit
```

Показать логическую переменную Git

```
git var -l | <variable>
```

Узнать название репозитория

```
git rev-parse --show-toplevel
```

Показать логи между диапазоном дат

```
git log --since='FEB 1 2017' --until='FEB 14 2017'
```

Исключить автора из логов

```
git log --perl-regexp --author='^((?!excluded-author-regex).*)
```

Создание сводки ожидающих изменений

```
git request-pull v1.0 https://git.ko.xz/project master:for-linus
```

Получение списка ссылок в удаленном репозитории

```
git ls-remote git://git.kernel.org/pub/scm/git/git.git
```

Список всех git алиасов

```
git config -l | grep alias | sed 's/^alias\.//g'
```

Альтернатива:

```
git config -l | grep alias | cut -d '.' -f 2
```

- Исходная статья

Перенос локальной ветки в удаленный репозиторий

```Shell
git push -u origin <branch_name>
```