Создать новый репозиторий:

git init project-name

Если вы планируете клонировать его по ssh с удаленной машины, также скажите:

git config --bool core.bare true

… иначе при git push вы будете получать странные ошибки вроде:

Refusing to update checked out branch: refs/heads/master  
By default, updating the current branch in a non-bare repository  
is denied, because it will make the index and work tree inconsistent  
with what you pushed, and will require 'git reset --hard' to match  
the work tree to HEAD.

Клонировать репозиторий с удаленной машины:

git clone git@bitbucket.org:afiskon/hs-textgen.git

Если хотим пушить один код в несколько репозиториев:

git remote add remotename git@gitlab.example.ru:repo.git

Добавить файл в репозиторий:

git add text.txt

Удалить файл:

git rm text.txt

Текущее состояние репозитория (изменения, неразрешенные конфликты и тп):

git status

Сделать коммит:

git commit -a -m "Commit description"

Сделать коммит, введя его описание с помощью $EDITOR:

git commit -a

Замерджить *все ветки* локального репозитория на удаленный репозиторий (аналогично вместо origin можно указать и remotename, см выше):

git push origin

Аналогично предыдущему, но делается пуш *только ветки master*:

git push origin master

Запушить *текущую ветку*, не вводя целиком ее название:

git push origin HEAD

Замерджить все ветки с удаленного репозитория:

git pull origin

Аналогично предыдущему, но накатывается только ветка master:

git pull origin master

Накатить текущую ветку, не вводя ее длинное имя:

git pull origin HEAD

Скачать все ветки с origin, но не мерджить их в локальный репозиторий:

git fetch origin

Аналогично предыдущему, но только для одной заданной ветки:

git fetch origin master

Начать работать с веткой some\_branch (уже существующей):

git checkout -b some\_branch origin/some\_branch

Создать новый бранч (ответвится от текущего):

git branch some\_branch

Переключиться на другую ветку (из тех, с которыми уже работаем):

git checkout some\_branch

Получаем список веток, с которыми работаем:

git branch *# звездочкой отмечена текущая ветвь*

Просмотреть все существующие ветви:

git branch -a *# | grep something*

Замерджить some\_branch в текущую ветку:

git merge some\_branch

Удалить бранч (после мерджа):

git branch -d some\_branch

Просто удалить бранч (тупиковая ветвь):

git branch -D some\_branch

История изменений:

git log

История изменений в обратном порядке:

git log --reverse

История конкретного файла:

git log file.txt

Аналогично предыдущему, но с просмотром сделанных изменений:

git log -p file.txt

История с именами файлов и псевдографическим изображением бранчей:

git log --stat --graph

Изменения, сделанные в заданном коммите:

git show d8578edf8458ce06fbc5bb76a58c5ca4a58c5ca4

Посмотреть, кем в последний раз правилась каждая строка файла:

git blame file.txt

Удалить бранч из репозитория на сервере:

git push origin :branch-name

Откатиться к конкретному коммиту (хэш смотрим в «git log»):

git reset --hard d8578edf8458ce06fbc5bb76a58c5ca4a58c5ca4

Аналогично предыдущему, но файлы на диске остаются без изменений:

git reset --soft d8578edf8458ce06fbc5bb76a58c5ca4a58c5ca4

Попытаться обратить заданный commit (но чаще используется branch/reset + merge):

git revert d8578edf8458ce06fbc5bb76a58c5ca4a58c5ca4

Просмотр изменений (суммарных, а не всех по очереди, как в «git log»):

git diff *# подробности см в "git diff --help"*

Используем vimdiff в качестве программы для разрешения конфликтов (mergetool) по умолчанию:

git config --global merge.tool vimdiff

Отключаем диалог «какой mergetool вы хотели бы использовать»:

git config --global mergetool.prompt false

Отображаем табы как 4 пробела, например, в «git diff»:

git config --global core.pager 'less -x4'

Создание *глобального* [файла .gitignore](https://git-scm.com/docs/gitignore):

git config --global core.excludesfile ~/.gitignore\_global

Разрешение конфликтов (когда оные возникают в результате мерджа):

git mergetool

Создание тэга:

git tag some\_tag *# за тэгом можно указать хэш коммита*

Удаление untracked files:

git clean -f

«Упаковка» репозитория для увеличения скорости работы с ним:

git gc

Иногда требуется создать копию репозитория или перенести его с одной машины на другую. Это делается примерно так:

mkdir -p /tmp/git-copy  
cd /tmp/git-copy  
git clone --bare git@example.com:afiskon/cpp-opengl-tutorial1.git  
cd cpp-opengl-tutorial1.git  
git push --mirror git@example.com:afiskon/cpp-opengl-tutorial2.git