01 Git Configuration

\$ git config --global user.name "Your Name"

Set the name that will be attached to your commits and tags.

\$ git config --global user.email "you@example.com"

Set the e-mail address that will be attached to your commits and tags.

\$ git config --global color.ui auto

Enable some colorization of Git output.

02 Starting a Project

\$ git init [project name]

Create a new local repository. If [project name] is provided, Git will create a new directory name [project name] and will initialize a repository inside it. If [project name] is not provided, then a new repository is initialized in the current directory.

\$ git clone [project url]

Downloads a project with the entire history from the remote repository

03 Day-to-Day Work

\$ git status

Displays the status of your working directory. Options include new, staged, and modified files. It will retrieve branch name, current commit identifier, and changes pending commit.

\$ git add [file]

Add a file to the staging area. Use in place of the full file path to add all changed files from the current directory down into the directory tree.

\$ git diff [file]

Show changes between working directory and staging area.

\$ git diff --staged [file]

Shows any changes between the staging area and the repository.

\$ git checkout -- [file]

Discard changes in working directory. This operation is unrecoverable.

\$ git reset [file]

Revert your repository to a previous known working state.

\$ git commit

Create a new commit from changes added to the staging area. The commit must have a message!

\$ git rm [file]

Remove file from working directory and staging area.

\$ git stash

Put current changes in your working directory into stash for later use.

\$ git stash pop

Apply stored stash content into working directory, and clear stash.

\$ git stash drop

Delete a specific stash from all your previous stashes

04 Git Branching Model

\$ git branch [-a]

List all local branches in repository. With -a: show all branches (with remote).

\$ git branch [branch_name]

Create new branch, referencing the current HEAD.

\$ git checkout [-b][branch_name]

Switch working directory to the specified branch. With -b: Git will create the specified branch if it does not exist.

\$ git merge [from name]

Join specified [from name] branch into your current branch (the one you are on currently).

\$ git branch -d [name]

Remove selected branch, if it is already merged into any other. -D instead of -d forces deletion.

05 Review your Work

\$ git log [-n count]

List commit history of current branch. -n count limits list to last n commits.

\$ git log --oneline --graph --decorate

An overview with reference labels and history graph. One commit per line.

\$ git log ref..

List commits that are present on the current branch and not merged into ref. A ref can be a branch name or a tag name.

\$ git log ..ref

List commit that are present on ref and not merged into current branch.

\$ git reflog

List operations (e.g. checkouts or commits) made on local repository

06 Tagging Known Commits

\$ git log [-n count]

List commit history of current branch. -n count limits list to last n commits.

\$ git log --oneline --graph --decorate

An overview with reference labels and history graph. One commit per line.

\$ git log ref..

List commits that are present on the current branch and not merged into ref. A ref can be a branch name or a tag name.

\$ git log ..ref

List commit that are present on ref and not merged into current branch.

\$ git reflog

List operations (e.g. checkouts or commits) made on local repository

07 Reverting Changes

\$ git reset [--hard] [target reference]

Switches the current branch to the target reference, leaving a difference as an uncommitted change. When -- hard is used, all changes are discarded.

\$ git revert [commit sha]

Create a new commit, reverting changes from the specified commit. It generates an inversion of changes.

08 Synchronizing Repositories

\$ git fetch [remote]

Fetch changes from the remote, but not update tracking branches.

\$ git fetch --prune [remote]

Delete remote Refs that were removed from the remote repository.

\$ git pull [remote]

Fetch changes from the remote and merge current branch with its upstream.

\$ git push [--tags] [remote]

Push local changes to the remote. Use --tags to push tags.

\$ git push -u [remote] [branch]

Push local branch to remote repository. Set its copy as an upstream.