

# Git

Git is currently the most popular implementation of a distributed version control system

## INSTALL GIT

GitHub provides desktop clients that include a graphical user interface for the most common repository actions and an automatically updating command line edition of Git for advanced scenarios.

GitHub for Windows <https://windows.github.com>

Git for All Platforms <http://git-scm.com>

## CONFIGURE TOOLING

Configure user information for all local repositories

```
$ git config --global user.name "[name]"
```

Sets the name you want attached to your commit transactions

```
$ git config --global user.email "[email address]"
```

Sets the email you want attached to your commit transactions

```
$ git config --global color.ui auto
```

Enables helpful colorization of command line output

## CREATE REPOSITORIES

Start a new repository or obtain one from an existing URL

```
$ git init [project-name]
```

Creates a new local repository with the specified name

```
$ git clone [url]
```

Downloads a project and its entire version history

## MAKE CHANGES

Review edits and craft a commit transaction

```
$ git status
```

Lists all new or modified files to be committed

```
$ git add [file]
```

Snapshots the file in preparation for versioning

```
$ git reset [file]
```

Unstages the file, but preserve its contents

```
$ git diff
```

Shows file differences not yet staged

```
$ git diff --staged
```

Shows file differences between staging and the last file version

```
$ git commit -m "[descriptive message]"
```

Records file snapshots permanently in version history

## REVIEW HISTORY

Browse and inspect the evolution of project files

```
$ git log
```

Lists version history for the current branch

```
$ git log --follow [file]
```

Lists version history for a file, including renames

```
$ git diff [first-branch]...[second-branch]
```

Shows content differences between two branches

```
$ git show [commit]
```

Outputs metadata and content changes of the specific commit

## GROUP CHANGES

Name a series of commits and combine completed efforts

```
$ git branch
```

Lists all local branches in the current repository

```
$ git branch [branch-name]
```

Creates a new branch

```
$ git checkout [branch-name]
```

Switches to the specified branch and updates the working directory

```
$ git merge [branch]
```

Combines the specified branch's history into the current branch

```
$ git branch -d [branch-name]
```

Deletes the specified branch

## REFACTOR FILENAMES

Relocate and remove versioned files

```
$ git rm [file]
```

Deletes the file from the working directory and stages the deletion

```
$ git rm --cached [file]
```

Removes the file from version control but preserves the file local

## SYNCHRONIZE CHANGES

Register a repository bookmark and exchange version history

```
$ git push [alias] [branch]
```

Uploads all local branch commits to GitHub

```
$ git pull
```

Downloads bookmark history and incorporates changes

# Github

## Create a new repository

A repository contains all project files, including the revision history.

```
$mkdir test
```

```
echo "# test1" >> README.md
```

```
git init
```

```
git add README.md
```

```
git commit -m "first commit"
```

```
git remote add origin https://github.com/username/test1.git
```

```
git push -u origin master
```

## Clone an existing repository

```
$ git clone url
```