Git Commands

▪ Setup & Configuration:

git --version:

The command git version is used to check the version of git.

```
jcs@Jeevas-MacBook-Air ~ % git --version
git version 2.47.0
```

git config:

Configures Git settings. Commonly used to set up user information

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

git config --global user.email youremail@example.com

```
jcs@Jeevas-MacBook-Air ~ % git config --global user.email "jeeva.charan21@gmail.com"
jcs@Jeevas-MacBook-Air ~ % git config --global user.name "jcs-2004"
```

git config –list:

Displays all the Git configurations for the current user.

```
jcs@Jeevas-MacBook-Air ~ % git config --list
credential.helper=osxkeychain
user.name=jcs-2004
user.email=jeeva.charan21@gmail.com
core.repositoryformatversion=0
core.filemode=true
core.bare=false
core.logallrefupdates=true
core.ignorecase=true
core.precomposeunicode=true
```

Git Commands

▪ Repository Management

git init:

Initialises a new Git repository in the current directory

```
jcs@Jeevas-MacBook-Air ~ % cd desktop
jcs@Jeevas-MacBook-Air desktop % cd gittest
jcs@Jeevas-MacBook-Air gittest % git init
hint: Using 'master' as the name for the initial branch. This default branch name
hint: is subject to change. To configure the initial branch name to use in all
hint: of your new repositories, which will suppress this warning, call:
hint:
hint:   git config --global init.defaultBranch <name>
hint:
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and
hint: 'development'. The just-created branch can be renamed via this command:
hint:
hint:   git branch -m <name>
Initialized empty Git repository in /Users/jcs/Desktop/Gittest/.git/
```

git clone:

Creates a copy of an existing Git repository from a remote source (e.g., GitHub) to your local machine.

```
jcs@Jeevas-MacBook-Air gittest % git clone
fatal: You must specify a repository to clone.

usage: git clone [<options>] [--] <repo> [<dir>]

    -v, --[no-]verbose    be more verbose
    -q, --[no-]quiet      be more quiet
    --[no-]progress       force progress reporting
    --[no-]reject-shallow don't clone shallow repository
    -n, --no-checkout     don't create a checkout
    --checkout            opposite of --no-checkout
    --[no-]bare           create a bare repository
    --[no-]mirror         create a mirror repository (implies --bare)
    -l, --[no-]local      to clone from a local repository
    --no-hardlinks        don't use local hardlinks, always copy
    --hardlinks           opposite of --no-hardlinks
    -s, --[no-]shared     setup as shared repository
    --[no-]recurse-submodules[=<pathspec>]
                          initialize submodules in the clone
    --[no-]recursive ...  alias of --recurse-submodules
    -j, --[no-]jobs <n>   number of submodules cloned in parallel
    --[no-]template <template-directory>
                          directory from which templates will be used
    --[no-]reference <repo>
                          reference repository
    --[no-]reference-if-able <repo>
                          reference repository
    --[no-]dissociate     use --reference only while cloning
    -o, --[no-]origin <name>
                          use <name> instead of 'origin' to track upstream
    -b, --[no-]branch <branch>
                          checkout <branch> instead of the remote's HEAD
    -u, --[no-]upload-pack <path>
                          path to git-upload-pack on the remote
    --[no-]depth <depth>  create a shallow clone of that depth
    --[no-]shallow-since <time>
                          create a shallow clone since a specific time
    --[no-]shallow-exclude <revision>
                          deepen history of shallow clone, excluding rev
    --[no-]single-branch  clone only one branch, HEAD or --branch
    --no-tags             don't clone any tags, and make later fetches not to follow them
    --tags                opposite of --no-tags
    --[no-]shallow-submodules
                          any cloned submodules will be shallow
    --[no-]separate-git-dir <gitdir>
                          separate git dir from working tree
    --[no-]ref-format <format>
                          specify the reference format to use
    -c, --[no-]config <key=value>
                          set config inside the new repository
    --[no-]server-option <server-specific>
                          option to transmit
    -4, --ipv4            use IPv4 addresses only
    -6, --ipv6            use IPv6 addresses only
    --[no-]filter <args>  object filtering
    --[no-]also-filter-submodules
                          apply partial clone filters to submodules
    --[no-]remote-submodules
                          any cloned submodules will use their remote-tracking branch
    --[no-]sparse         initialize sparse-checkout file to include only files at root
    --[no-]bundle-uri <uri>
                          a URI for downloading bundles before fetching from origin remote
```

Git Commands

▪ Staging and Committing

git status:
Shows the status of changes in your working directoryand staging area. It tells you which iles are untracked, modi ied, or ready to be committed.

```
jcs@Jeevas-MacBook-Air gittest % git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        SE Unit 1 Tesseract.pdf

nothing added to commit but untracked files present (use "git add" to track)
```

git add:
Adds changes in the working directory to the staging area.

&

git commit:
Commits the staged changes to the repository with a descriptive message. The -m option allows you to include a commit message.

```
jcs@Jeevas-MacBook-Air gittest % git add se.pdf
jcs@Jeevas-MacBook-Air gittest % git commit -m "se document"
[master (root-commit) daf41d0] se document
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 se.pdf
```

Git Commands

▪ Branching and Merging

git branch:

Lists all branches or creates a new branch

```
jcs@Jeevas-MacBook-Air gittest % git branch
* master
```

git checkout:

Switches to a different branch

git checkout branch-name

```
jcs@Jeevas-MacBook-Air gittest % git checkout master
Already on 'master'
```

▪ Undoing Changes

git revert:

Creates a new commit that undoes the changes from a Specified commit, leaving the history intact.

```
jcs@Jeevas-MacBook-Air gittest % git revert
usage: git revert [--[no-]edit] [-n] [-m <parent-number>] [-s] [-S[<keyid>]] <commit>...
   or: git revert (--continue | --skip | --abort | --quit)

    --quit                end revert or cherry-pick sequence
    --continue            resume revert or cherry-pick sequence
    --abort               cancel revert or cherry-pick sequence
    --skip                skip current commit and continue
    --[no-]cleanup <mode> how to strip spaces and #comments from message
    -n, --no-commit       don't automatically commit
    --commit              opposite of --no-commit
    -e, --[no-]edit       edit the commit message
    -s, --[no-]signoff    add a Signed-off-by trailer
    -m, --[no-]mainline <parent-number>
                          select mainline parent
    --[no-]rerere-autoupdate
                          update the index with reused conflict resolution if possible
    --[no-]strategy <strategy>
                          merge strategy
    -X, --[no-]strategy-option <option>
                          option for merge strategy
    -S, --[no-]gpg-sign[=<key-id>]
                          GPG sign commit
    --[no-]reference      use the 'reference' format to refer to commits
```

Git Commands

▪ Viewing History

git log:
Shows a history of commits in the repository, including commit hashes, messages, and timestamps. Use git log–one line for a more concise view.

git diff:
Displays differences between various commits, the working directory, and the staging area. git diff without arguments shows changes not yet staged.

```
jcs@Jeevas-MacBook-Air gittest % git log
commit daf41d00cfe87495e9820d13ce6c5fdef0bbe409 (HEAD -> master)
Author: jcs-2004 <jeeva.charan21@gmail.com>
Date:   Sat Oct 26 11:02:55 2024 +0530

    se document
jcs@Jeevas-MacBook-Air gittest % git show
commit daf41d00cfe87495e9820d13ce6c5fdef0bbe409 (HEAD -> master)
Author: jcs-2004 <jeeva.charan21@gmail.com>
Date:   Sat Oct 26 11:02:55 2024 +0530

    se document

diff --git a/se.pdf b/se.pdf
new file mode 100644
index 0000000..508e5b9
Binary files /dev/null and b/se.pdf differ
jcs@Jeevas-MacBook-Air gittest %
```

git show:
Shows the details of a specific commit, including the changes made and the commit message.

Git Commands

```
jcs@Jeevas-MacBook-Air gittest % git show
commit daf41d00cfe87495e9820d13ce6c5fdef0bbe409 (HEAD -> master)
Author: jcs-2004 <jeeva.charan21@gmail.com>
Date:   Sat Oct 26 11:02:55 2024 +0530

    se document

diff --git a/se.pdf b/se.pdf
new file mode 100644
index 0000000..508e5b9
Binary files /dev/null and b/se.pdf differ
jcs@Jeevas-MacBook-Air gittest %
```