Merging

Merge a branch into the current branch

```
git merge <OTHERBRANCHNAME>
```

Merge multiple branches into the current branch

```
git merge <BRANCHONE> <BRANCHTWO> <BRANCHTHREE> <BRANCHFOUR>
```

Or squashing all the commits into one, even if a fast forward:

```
git merge <BRANCHONE> --squash
```

Merge Conflicts

Merge conflict. How do we fix it? How do we continue?

```
git add
git merge --continue
```

Or aborting

```
git merge --abort
```

Merge with rebase is a better option for maintaining linear history

Merge without commit

```
git merge --no-commit
```

Using the `ours` strategy

```
git merge -s ours <sombranch>
```

Using the `recursive` strategy with `ours` or `theirs` suboptions. Details here.

```
git merge -Xtheirs <sombranch>
```

Visualizing Merges

Show the current branch's merge status

```
git show-branch
```

Show every branch's merge status

```
git show-branch --all
```

Show the contents further back the parent tree

```
git show-branch --more=5
```

Show a max of 10 entries from no more than 1 hour ago for the master branch

```
git show-branch --reflog="10,1 hour ago" --list master
```

Show three branches and their merge status

```
git show-branch master feature1 feature2
```

Understanding the branch visualizations

- The branch head that is pointed at by HEAD is prefixed with an asterisk * character
- Other heads are prefixed with a ! character.
- Following these N lines, one-line log for each commit is displayed, indented N places.
  - If a commit is on the I-th branch, the I-th indentation character shows a + sign
  - Otherwise it shows a space (does not exist here). * Merge commits are denoted by a - sign.

Rebase

Linearize the branch commits. Rebranches at the latest and replays committed branch work on top of that.

```
git rebase <source branch name>
```

Or perform the rebase interactively, where you can change the order of the commits

```
git rebase -i <branchname>
```

If conflicts occur:

```
git add
git rebase --continue
```

Or if you want to bail out, a similar command to aborting a merge:

```
git rebase --abort
```

You can squash multiple commits into one commit with the interactive rebase. Simple mark a commit as squashed.

Commands:

```
p, pick = use commit
e, edit = use commit, but stop for amending
s, squash = use commit, but meld into previous commit
Or if you remove a line, that commit will be excluded from the rebase.
```

Rerere

If you merge a branch often, you don't want to keep telling Git how to merge it every time you merge. With rerere enabled (just an option switch), it will remember your resolutions and use them next time to minimize your effort.

```
git config --global rerere.enabled 1
```

External Merge Tools

Setup diff in `~/.gitconfig` global configuration file:

```
#[diff]
#   tool = adifftool
#   external = git-difftool--helper
```

or

```
[difftool "adifftool"]
    cmd = araxis-difftool.sh \"$LOCAL\" \"$REMOTE\" \"$MERGED\"
[difftool]
    prompt = false

#Automatically do all merges with this mergetool
[merge]
    tool = amergetool
[mergetool "amergetool"]
    cmd = araxis-mergetool.sh \"$PWD/$LOCAL\" \"$PWD/$BASE\" \"$PWD/$REMOTE\" \"$PWD/$MERGEI
[mergetool]
    prompt = false
```

Using Araxis for diffing

```
git difftool HEAD HEAD^ -- test2.txt
```