How To Use Git At Scylla Efficiently

Goals of our work

Goals of our work

- Implement features

Goals of our work

- Implement features
- Make code reviews easier to perform

Goals of our work

- Implement features
- Make code reviews easier to perform
- Keep changes understandable in the future

Clone repository

```
$ git clone git@github.com:ewienik/zpp-git.git
$ git status
On branch master
Your branch is up to date with 'origin/master'.

nothing to commit, working tree clean
$ git log --oneline
8d7ec69 (HEAD -> master, origin/master, origin/feature, origin/HEAD) Prepare initial marp presentation
e4a1d03 Initial commit
```

Switch branch

```
$ git switch feature
branch 'feature' set up to track 'origin/feature'.
Switched to a new branch 'feature'
$ git status
On branch feature
Your branch is up to date with 'origin/feature'.

nothing to commit, working tree clean
```

Create branch

```
$ git switch -c change-docs
Switched to a new branch 'change-docs'
$ git status
On branch change-docs
nothing to commit, working tree clean
```

Create commits

```
# edit files
$ git status
$ git diff
$ git add .
$ git add -p
$ tig
$ git commit
# repeat as needed
```

Push changes

```
$ git log --oneline
$ git status
$ tig
$ git push # current branch
```

Rebase with master

```
# two developers working in parallel
$ git fetch origin
$ git fetch --all
$ git rebase origin/feature # not git merge
# resolve conflicts if any
```

Cleanup before review

Cleanup before review

- divide commits logically

Cleanup before review

- divide commits logically
- write good commit messages

Cleanup before review

- divide commits logically
- write good commit messages
- commits should be as a story

Rearrange, squash or fix commits

```
$ git log --oneline
$ tig
$ git rebase -i origin/feature --keep-base
# rearrange commits in editor
```

Split commits

```
$ git rebase -i origin/feature --keep-base
# break or edit at some commit to split
$ git log -1
$ git reset --soft HEAD~1
# split staged/not staged changes
$ git commit -m "part 1"
$ git add .
$ git commit -m "part 2"
```

Update commits

```
$ git rebase -i origin/feature --keep-base
# break or edit at some commit to update
$ git log -1
$ git status # nothing to commit, working tree clean
$ git commit --amend # update description
# add modification
$ git add .
$ git commit --amend # update description and content
$ git log -1
```

Rebase with stacked branches

```
$ git rebase -i origin/feature --keep-base --update-refs
# move refs to the correct commits
```

https://www.codetinkerer.com/2023/10/01/stacked-branches-with-vanilla-git.html

Rework after review

Tools

https://git-scm.com/

https://jonas.github.io/tig/

https://jj-vcs.github.io/