GIT CHEAT SHEET

Configuring

```
git config --global  user.name "NAME"      alias.ALIAS "COMMAND"
                     user.email "MAIL"
                     core.editor "EDITOR"
```

Cloning: git clone url [dir]

Gitignore: file named .gitignore in root dir. Supports shell globbing: *.a will ignore all .a-files. To make exceptions, use !. Ex:

```
*.a            -> will ignore all .a-files except for special.a
!special.a
```

Staging; status:

* git status -s:

```
 M .  -> file modified, not staged
MM -  -> file modified, staged, modified again
A  .  -> file newly added and staged
M  -  -> file modified and staged
?? -  -> file not yet added
ST WD
```

* git diff:

$$\begin{tikzcd} WD \arrow[r, leftrightarrow, "\text{git diff}"'] & SA \arrow[r, leftrightarrow, "\text{git diff --staged}"'] & CO \end{tikzcd}$$

* untrack file: git rm --cached file

rm; mv: prefix git stages changes.

* unstage file: git reset HEAD <file>

Git log

```
-p        : show changes between commits
-n        : show last n ∈ ℕ commits
--graph   : show branch graph
--oneline : describe each commit in one line
--decorate: show branch, HEAD, tags
--grep    : show commits with message matching regex
-S        : show commits adding/removing string
```

Remotes

* show all : git remote -v

* add : git remote add <name> <url>

* fetching : git fetch <name>

gets all data without merging

* pulling : git pull <name>

gets all data and merges

* pushing : git push <name> <branch>

* removing/renaming remotes : git remote rename <name> <newname>

rm <name>

Tags

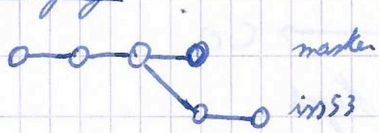
* show all : git tag

* show specific : git tag -l (shows tags matching pattern (sh. globbing))

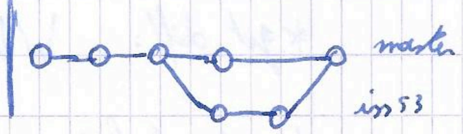
* add : git tag -a <tag> -m "msg" [commit_hash]

* inspect tag commit : git show <tag>

Merging



```
git checkout master
git merge iss53
```



resolving conflicts: - Open file with conflict

- Replace <<<< HEAD: file

...

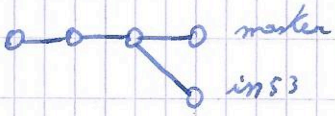
= = = = = = = = = =

...

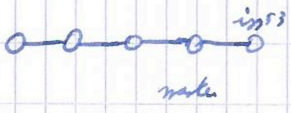
>>>> branch: file

- Git add file again
- Git commit

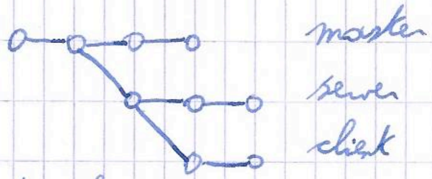
Rebasing



git checkout iss53
git rebase master



Rebase onto:



git rebase --onto master server client
="Check out client, get patches from common ancestor of server and client, rebase onto master"