

git - Part III

Analysis Tools & Miscellaneous

Stefan Pranger

stefan.pranger@student.tugraz.at

November 4, 2020

Overview

- 1 Rebase
- 2 Analysis Tools
 - git grep
 - git blame
 - git log -L
 - git bisect
- 3 Hooks
- 4 .gitconfig
- 5 Workflows

Road Map

Rebase

Analysis Tools

Hooks

.gitconfig

Workflows

- 02. 11.: Local Repository and git basics
- 03. 11.: Branches, Remotes, non-linear workflow and working in a team
- *Today*: Miscellaneous: Rebase, Analysis Tools, .gitconfig, Best Practices and Workflows

Tags

Rebase

Analysis Tools

Hooks

.gitconfig

Workflows

- `$> git tag <tag_name> <commit>`
- very lightweight reference to a commit.
- `$> git show <object>` to inspect commits, tags or any other object!
- `$> git tag -a <tag_name> <commit>` to create an annotated tag.
- Useful for milestones such as new versions!
- Need to be pushed separately via
 - `$> git push origin <tag_name>` or
 - `$> git push origin --tags`.

Rebasing

Rebase

Analysis Tools

Hooks

.gitconfig

Workflows

- Reapplying commits an top of branches

```
$> git merge vs. $> git rebase
```

Rebasing

Rebase

Analysis Tools

Hooks

.gitconfig

Workflows

■ Reapplying commits on top of branches

`$> git merge` vs. `$> git rebase`

- 1 Linear history,
- 2 Some consider merge commits to be irritating in the history and
- 3 merge commits make `$> git bisect` harder to use.

Rebasing

Rebase

Analysis Tools

Hooks

.gitconfig

Workflows

■ Reapplying commits an top of branches

`$> git merge` vs. `$> git rebase`

- 1 Linear history,
- 2 Some consider merge commits to be irritating in the history and
- 3 merge commits make `$> git bisect` harder to use.

■ Possible team policy

Interactive Rebasing

Rebase

Analysis Tools

Hooks

.gitconfig

Workflows

Local Cleanup:

- Edit Messages
- Edit Commit
- Reorder Commits
- Remove Commits
- Squash Commits

Analysis Tools

Rebase

Analysis Tools

git grep
git blame
git log -L
git bisect

Hooks

.gitconfig

Workflows

- `$> git log` gives us a way to browse the projects history,
- sometimes we want to find specific commits,
- find specific parts of code or
- get all changes done to certain lines of code.

git grep

Rebase

Analysis Tools

[git grep](#)

[git blame](#)

[git log -L](#)

[git bisect](#)

Hooks

[.gitconfig](#)

Workflows

- `$> git grep <regex>`
 - `-A|B|C <n>`
 - `--heading`
 - `--show-function|-p`
 - `--function-context|-W`
- git version of `$> grep`
- Searches through the working tree
- Runs multithreaded, which means it is very fast.

git blame

Rebase

Analysis Tools

git grep

git blame

git log -L

git bisect

Hooks

.gitconfig

Workflows

- `$> git blame <filename>`
- Gives you information for each line of code

git log -L

Rebase

Analysis Tools

git grep

git blame

git log -L

git bisect

Hooks

.gitconfig

Workflows

- Traces the evolution of a range of lines.
- `$> git log -L <start>,<end>:<filename>`
- Results in a list of commits which have affected the given range.

git bisect

Rebase

Analysis Tools

- git grep
- git blame
- git log -L

- git bisect

Hooks

.gitconfig

Workflows

- What if you do now want to trace changes but find a change?
- `$> git bisect`
 - 1 `$> git bisect start`
 - 2 `$> git bisect good <good_commit>`
 - 3 `$> git bisect bad <bad_commit>`

git bisect

Rebase

Analysis Tools

[git grep](#)
[git blame](#)
[git log -L](#)[git bisect](#)

Hooks

[.gitconfig](#)

Workflows

- What if you do now want to trace changes but find a change?
- `$> git bisect`
 - 1 `$> git bisect start`
 - 2 `$> git bisect good <good_commit>`
 - 3 `$> git bisect bad <bad_commit>`
- git helps you perform a binary search through the commits.
- Your HEAD will be moved through the specified range.

Hooks

Rebase

Analysis Tools

Hooks

.gitconfig

Workflows

- Automate your workflow or
- prevent mistakes.
- Called before or after certain actions, like
`git commit`, `git push`, etc.

.gitconfig

Rebase

Analysis Tools

Hooks

.gitconfig

Workflows

- Your git configuration file: `$HOME/.gitconfig`, on Windows: `C:/Users/git-user/`, for global modifications.
- Every .git repository has its own configuration file. This stores information about branches, remotes, etc.
- May be modified directly or via
`$> git config --global <key>=<value>`

.gitconfig

Rebase

Analysis Tools

Hooks

.gitconfig

Workflows

- Lets you change:
 - Output of commands
 - Output colors,
 - Behaviour of git and
 - define aliases.

.gitconfig

Rebase

Analysis Tools

Hooks

[.gitconfig](#)

Workflows

- Lets you change:
 - Output of commands
 - Output colors,
 - Behaviour of git and
 - define aliases.
- `st = status`
- `unstage = reset HEAD --`
- `[merge] ff = only`

Workflows

Rebase

Analysis Tools

Hooks

.gitconfig

Workflows

- git offers many different ways to manage your code.
- Apart from commits there is nothing *carved in stone*.
- One of the great features, but also a source for conflicts.
- Several resources online!

Workflows

Rebase
Analysis Tools
Hooks
.gitconfig
Workflows

- Workflows should be discussed at the start of a project.
- Open source projects often use the concept of **Pull Requests**.
- Such projects often have two parallel running branches: `master` and `development`.
- A reviewer may then decide on whether a feature is added to `development`.