

# git - Part III Analysis Tools & Miscellaneous

Stefan Pranger

stefan.pranger@student.tugraz.at

October 29, 2019



### Road Map

- 22. 10.: Local Repository and git basics
- 24. 10.: Branches, Remotes, non-linear workflow and working in a team
- Today: Miscellaneous: Rebase, Analysis Tools, .gitconfig, Best Practices and Workflows



### Overview

Rebase

Analysis Tools

.gitconfig

- Rebase
- 2 Analysis Tools
  - git grep
  - git blame
  - git log
  - git bisect
- .gitconfig
- Workflows



# Tags

#### Rebase

Analysis Tools

.gitconfig

- \$> git tag <tag\_name> <commit>
- very lightweight reference to a commit.



# Tags

#### Rebase

Analysis Tools .gitconfig

- \$> git tag <tag\_name> <commit>
- very lightweight reference to a commit.
- \$> git show <object> to inspect commits, tags or any other object!



### Tags

#### Rebase

Analysis Tools

- \$> 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 seperately via
  \$> git push origin <tag\_name> Or
  \$> git push origin --tags .



# Cherry Pick

#### Rebase

Analysis Tools .gitconfig

- \$> git cherry-pick <commit>...
- Picks a range of commits and applies them onto your HEAD.
- Requires you to be in a clean working tree.
- May cause merge-conflicts!



# Rebasing

Rebase

Analysis Tools

.gitconfig

Workflows

Reapplying commits onto tip of branches\$> git merge vs.\$> git rebase



# Rebasing

#### Rebase

Analysis Tools

.gitconfig

- Reapplying commits onto tip of branches
  - \$> git merge VS. \$> git rebase
  - 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

.gitconfig

- Reapplying commits onto tip of branches
  - \$> git merge VS. \$> git rebase
  - 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 or useful for local cleanup.



## Interactive Rebasing

#### Rebase

Analysis Tools

.gitconfig

Workflows

### Local Cleanup:

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



## Interactive Rebasing

#### Rebase

Analysis Tools

.gitconfig

Workflows

### Local Cleanup:

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

Never apply any of the above on already shared commits!



# Analysis Tools

Rebase

#### Analysis Tools

git grep git blame git log git bisect

#### .gitconfig

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



## git grep

Rebase

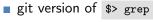
Analysis Tools

git grep git blame

git log git bisect

.gitconfig

Bircoiiii





### git grep

Rebase

Analysis Tools

git grep

git log

git bisec

.gitconfig

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



### git blame

Rebase

Analysis Tools

git grep

git blame

git bisect

.gitconfig

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



### git blame

Rebase

Analysis Tools

git grep

git blame

git bised

.gitconfig

- \$> git blame <filename>
- Gives you information for each line of code
- But! This changes the author when the file was renamed!
- Better: \$> git log -p -M --follow -- <filename>
- This gives you the history of the file.



### git log -L

Rebase

Analysis Tools

git gre

git log

git bisec

.gitconfig

- 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 blam

git log git bisect

.gitconfig

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



### git bisect

Rebase

Analysis Tools

git blame

git bisect

.gitconfig

- 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>
  - \$ git bisect bad <bad\_commit>
- git helps you perform a binary search through the commits.
- Your HEAD will be moved through the specified range.



# .gitconfig

Rebase

Analysis Tools

.gitconfig

- 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

.gitconfig

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



# .gitconfig

Rebase

Analysis Tools

.gitconfig

- 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

.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

.gitconfig

- 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.



## Summary

Rebase

Analysis Tools

.gitconfig

- What is a tag?
- How you can you pick certain commits?
- How can you get commit-specific informations?
- How does a git rebase work?
- How do you apply changes with an interactive rebase?
- How can you search for specific code within your project?
- How can you trace the history of a file or range of lines?
- How can you manipulate your git configuration?