

Mastering git

Lesson 6 - last one 🎉🎉🎉

Irina Gulina

Tomas Tomecek

Course schedule

September						
S	M	T	W	T	F	S
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30

October						
S	M	T	W	T	F	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	×	26	27	28
29	30	31				

November						
S	M	T	W	T	F	S
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30		

Class N HW	2	3	4	5	6	Bonus Task
Deadline	11.10	18.10	25.10	8.11	13.11	13.11

Course grade: November 20

Class 4 HW feedback

- ▶ Some used rebase, others used rebase i.
 - What is the difference between those two?
 - Will both of them work in a case of HW class 4?
 - Which is easier / preferable in HW class 4 situation and why?
- ▶ Some used push after rebase, others used push --force.
 - What is the difference between those two?
 - Will both of them work?
 - Which is preferable in HW class 4 situation and why?
- ▶ ~~merge~~ rebase
- ▶ Commit messages

Class 5 HW feedback



Any questions or
suggestions?

Today's class

- ▶ Git Etiquette or mind your Git manners
- ▶ git blame, cherry-pick, submodule
- ▶ Efficient git
- ▶ Fetching pull/merge requests locally
- ▶ Activity in the end
- ▶ Let's do a group picture!

Git Etiquette or Mind your Git manners

Git Commit

Commit content

- Don't: **Two and more changes in one commit**

1e4faa0 Fix login timeout BZ, add logout step

- Do: **One commit = One logical change**

1e4faa0 Fix login timeout BZ

2r5asy8 Add logout step

Commit content

- Separate whitespace changes from code changes, especially unrelated.
 - Mixing those is a great way to introduce a bug and
 - Complicates code review

What is a commit message?

- Title/subject line
- Body

Commit message example

```
$ git log
```

```
commit <commit_id>
```

```
Author: <author_name> <author_email>
```

```
Date: Mon Apr 2 15:10:03 2020 -0400
```

```
Change how workers are represented
```

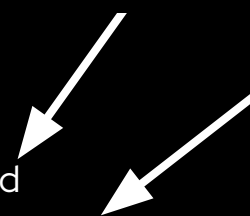
- * Don't serialize the 'gracefully_shutdown' field
- * Create a new 'missing' property and serialize it
- * In the status API, list both online and missing workers

```
Requires PR: https://github.com/ <project>/pull/921
```

```
closes #354498
```

```
https://bugzilla.redhat.com/show\_bug.cgi?id=354498
```

Commit Title or Subject line



Commit Body

What is a “bad” message?

```
$ git log --oneline -5 --author irina --before "Wed Apr 7 2021"
```

```
dcc2d35      address comments  
b7aac30      fix issue #123  
0b7a4e4      various docs fixes  
1e4faa0      ui bug fix  
fc3d081      readme      update  
d21660dc     ToDo  
0b7a4e4      Mix fixes and cleanups  
5h3d28g      refactoring
```

What is a “bad” message?

```
$ git log --oneline -8 --author irina --before "Wed Apr 7 2021"
```

dcc2d35	address comments	<- what comments?
b7aac30	fix issue #123	<- of what project?
0b7a4e4	various docs fixes	<- what docs? why?
1e4faa0	ui bug fix	<- what was the bug?
fc3d081	readme update	<- why?
d21660dc	ToDo	<- 🙄🙄🙄🙄
0b7a4e4	Mix fixes and cleanups	<- 🙄🙄🙄
5h3d28g	refactoring	<- 🤦

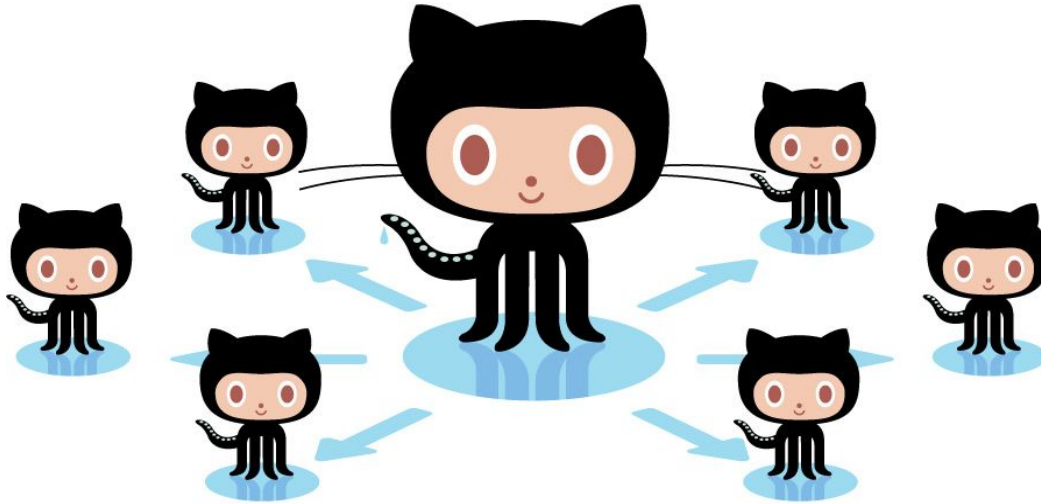
Uninformative, look-elsewhere commit messages (titles)

Usage of a commit title

- `git log --pretty=oneline`
- `git rebase --interactive`
- `merge.summary`
- `git shortlog`
- `git format-patch`, `git send-email`, ...
- `reflogs`
- GUI tools for committing and browsing
- GitHub, SourceForge, Bitbucket, GitLab, ... service

Poor quality code can be refactored.
A terrible commit message lasts
forever.

For whom do you write commit messages?



Why should I write 'good' commit messages?

- To help to understand the code change
 - What has been changed?
 - Why is that change necessary?
- To speed up the reviewing process
- To help to locate a bug
- To write a good release note or script it

What constitutes a good commit message?

- `git commit -m "Fix login timeout bug"`
- `git commit` or `git commit --verbose`

```
Redirect user to the requested page after login
```

```
https://link/to/issue/tracker
```

What constitutes a good commit message?

- Capital letter, 50/72, no punctuation in the end

```
$ git commit
```

```
A brief summary of the commit
```

```
A paragraph describing what changed and its impact.
```

What constitutes a good commit message?

- Present Tense and Imperative Mood










```
cf31d12 Adds login unit tests
7a9kj4f Fixed login unit tests
101q2wd Update login unit tests
1b7hn61 Removing login unit test
```

“If accepted, this commit will *<your commit message goes here>*.”

Conventional / Emoji / Semantic commits

Add “keywords and/or emoji to commit message and/or footer based on scope and set of rules. E.g.:

feat!: email customers on product changes
docs: correct spelling of CHANGELOG
feat(lang): add Polish language
fix: prevent racing of requests

 build(Electron): Bump version 7 to 9 
 Improve structure / format of the code
 Remove code or files
 Fix a bug
 Critical hotfix
 Introduce new features
 Add or update documentation
 Deploy stuff

In Footer: close/closes/closed/fix/fixes/fixed/resolve/resolves/resolved

E.g: Fixes: #1 or Fixes #1

CHECK CONTRIBUTING.md like guidance on the project

Ticket number in commit messages

- Ticketing system **!=** git log
 - "TICKET-123456 add missing params to class"
 - "Add missing meta fields to response"
- ❑ Takes space in 50 chars limit title
- ❑ Look-elsewhere for details message, I'm lazy
- ❑ May be not available for interested user or reviewer
(permissions, outage)

What constitutes a good commit message?

- Clear Title - What is commit about?
- Present Tense and Imperative Mood
- No punctuation in a title
- Clear Body - What and why is it needed/changed vs how?
- 50/72
- Reference to an issue in a body message
- Follow the commit convention defined by the team

Git Push

IF YOU DO FORCE PUSH...

May the force stay with you.

Git push --force trap

- It's ok to force push to your local branch
- It's ok to force push to your (unmerged/open) PR/MR
- **It's not ok to force push to a public branch**

Git push --force consequences

- Lost data
- Altered history
- Not happy colleagues
- Lost karma points

How to avoid unwanted force push

- Protect important branches
- Backup
- Use git branch/switch
- Use --force-with-lease, carefully
- Use PR revert

You have a great freedom...
to change your history **locally**.

Submitting a PR

Why do we use PR/MR workflow?

- Share changes
- Get review and feedback
- Encourage quality

What constitutes a good PR/MR?

- Complete piece of work
- Adds value in some way
- Solid title and body
- Clear commit history
- Small
- Meets project's contribution guidelines

Contributors (before submitting a PR/MR)

- Follow the repo's conventions
- Double check your code (and Todos)
- Add docs
- Keep changes small
- Separate branch
- Be clear and specific
- Check your ego and be polite,

Open Source Contribution (GitHub)

Providerless tag for client go auth plugins #100606

Merged k8s-ci-robot merged 1 commit into `kubernetes:master` from `dims:providerless-tag-for-client-go-auth-plugins` 18 hours ago

Conversation 9 Commits 1 Checks 0 Files changed 3 +36 -4

dims commented yesterday • edited

client-go auth plugins are another vector to pull in cloud specific code when they are not needed at all. So ensure that the existing `providerless` tag is honored. This gets rid of the `gcp/openstack/azure` built-in auth plugins.

/kind bug
/release-note-none

What type of PR is this?

What this PR does / why we need it:

Which issue(s) this PR fixes:

Fixes #

Special notes for your reviewer:

Does this PR introduce a user-facing change?

Additional documentation e.g., KEPs (Kubernetes Enhancement Proposals), usage docs, etc.:

Reviewers
wojtek-t
xichengludui

Assignees
ligitt

Labels
approved area/code-organization
cncf-cla: yes kind/bug lgtm
needs-triage priority/important-soon
release-blocker release-note-none
sig/api-machinery sig/auth size/M

Projects
None yet


Milestone
v1.21

Linked issues
Successfully merging this pull request may close these issues.
None yet

Added labels
do-not-merge/work-in-progress size/M do-not-merge/release-note-label-needed do-not-merge/needs-kind
do-not-merge/needs-sig needs-triage labels yesterday

Open Source Contribution (GitLab)

Inkscape > inkscape > Merge Requests > !2946

Merged Created 4 days ago by  **Jyoti Balodhi** Contributor



Typo Fixed parent ->parentItem





Overview 6 Commits 1 Pipelines 3 Changes 1 ✔ All threads resolved



Hello, I have fixed a typo in <https://gitlab.com/inkscape/inkscape/-/blob/master/src/selection-chemistry.cpp#L3007-3009> as:-






```
SPIItem *parentItem = dynamic_cast<SPIItem *>(parent);
    if (parentItem) { // parent was being checked here before rather than parentItem
        parent_transform = parentItem->i2doc_affine();
```



Edited 4 days ago by Jyoti Balodhi


 Request to merge Jyoti_Balodhi:fix into master 

 Merged result pipeline #277765637 passed for 48600ee5   

 Merge request approved. Approved by 

>      [View eligible approvers](#)

 Merged by  **Thomas Holder** 1 day ago

The changes were merged into `master` with `cbfa6879` 

The source branch has been deleted

Contributors (after submitting a PR/MR)

- Check your ego and be polite
 - **@username ping!**
 - **@username review please**
- Ensure your branch merge and tests pass
- Use --amend, --fixup or rebase -i
- Don't merge your own PR

WIP PR/MR

- Don't overuse WIP label
- Remove WIP label when ready
- "This is ready for review, please."

Reviewing a PR

PR Reviewers

- Be kind and polite
 - **@username ping, error here!**
 - **@username s/foo/bar**
- Check commit history
- Don't fix issues
- Ensure the branch can be merged
- CI Tests pass
- Don't merge WIPs
- Squash
- Delete branch

We do have questions!

1. What's the difference between **plain**, **--soft** and **--hard reset**?

We do have questions!

2. What's wrong with this commit title?

“aDDED new best functionality in this commmmmit: implementing option --foobar.”

We do have questions!

2. What's wrong with this commit title?

“aDDED new best functionality in this commmmmit: implementing option --foobar.”

“Add option --foobar”

We do have questions!

3. What's the difference between **fetch** and **pull**

We do have questions!

4. What does **push** do?

We do have questions!

5. What does **force push** do?

More handy commands

git blame

display author metadata of lines in a committed file

\$ git blame README.md

```
6997a160 (Tomas Tomecek 2023-03-10 11:24:09 +0100 25) ## Lectors
6997a160 (Tomas Tomecek 2023-03-10 11:24:09 +0100 26) * Irina...
cb4e8b8b (Tomas Tomecek 2023-05-26 18:56:07 +0200 27) * [Tomáš...
cb4e8b8b (Tomas Tomecek 2023-05-26 18:56:07 +0200 28)
41c6d3a1 (Tomas Tomecek 2023-05-29 13:43:00 +0200 29) You can...
41c6d3a1 (Tomas Tomecek 2023-05-29 13:43:00 +
```


git blame

display of author metadata attached to specific committed lines in a file

git blame NAME_OF_THE_FILE

If we want to know more about the commit, what command we can use?

git blame

display of author metadata attached to specific committed lines in a file

git blame NAME_OF_THE_FILE

If we want to know more about the commit, what command we can use?

git log -p SHA-1

git show SHA-1

git blame

display of author metadata attached to specific committed lines in a file

git blame -e NAME_OF_THE_FILE

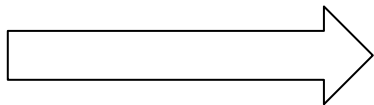
git blame -L startLineNumber,endLineNumber filePath

git blame -w NAME_OF_THE_FILE

git cherry-pick

Pick a commit by reference and append to the current working HEAD

a - b - c - d Main
|
e - **f** - g Feature



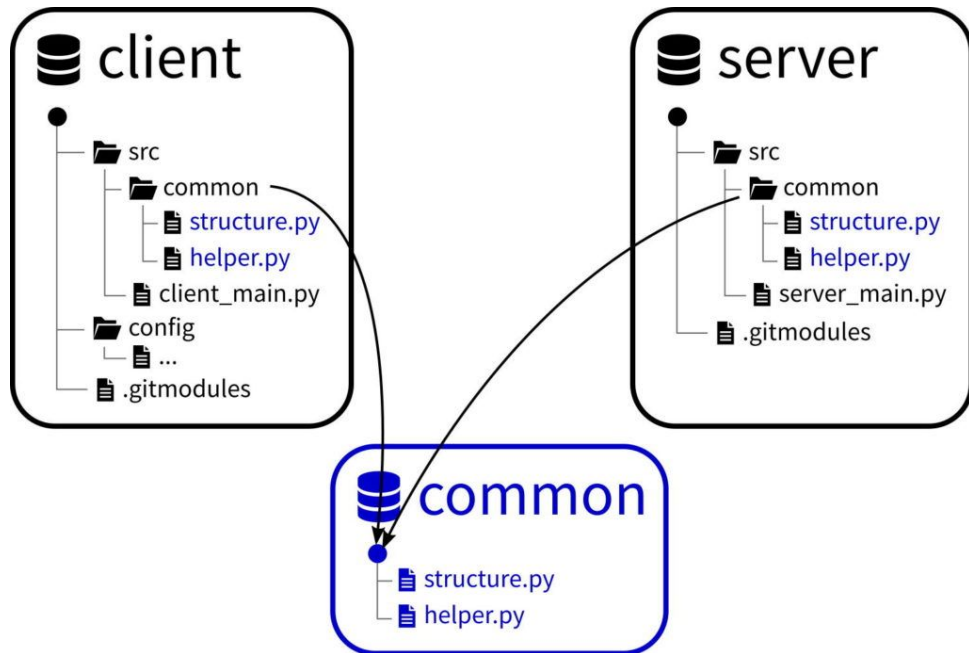
a - b - c - d - **f** Main
|
e - f - g Feature

When may it be useful?

What is the danger of using it?

git submodule

allows to keep a git repository as a subdirectory of another git repository



git submodule

allows to keep a git repository as a subdirectory of another git repository

- ▶ When an external component is changing too fast.
- ▶ When an external component isn't updated very often
- ▶ When a piece of the project is delegated to a third party and you want to integrate their work at a specific time or release.

Sometimes, it's the best way to create and manage internal packages or manage changing micro services

git submodule

allows to keep a git repository as a subdirectory of another git repository

- ▶ Points to a specific commit, not to a ref or branch
- ▶ Static, not updated automatically when a host repo is updated

git submodule

allows to keep a git repository as a subdirectory of another git repository

git submodule add URL NAME

```
git submodule add https://gitlab.com/user-name/repo my-submodule
```

.git submodule

```
[submodule "my-submodule"]
  path = my-submodule
  url = https://gitlab.com/user-name/repo
```

my-submodule

Those are new files on the project and needed to be committed.

git submodule

allows to keep a git repository as a subdirectory of another git repository

git submodule init

git submodule update

git clone --recurse-submodules URL

git push --recurse-submodules=check

Continuous Integration (CI)

Why?

- ▶ Run tests once a pull request is created or updated
- ▶ Run tests once a pull request is merged
- ▶ GitHub example →



vinzenz testing: Start using GH Actions for unit tests (#719) ... ✓

1 contributor

38 lines (36 sloc) | 1.28 KB

```
1 name: Unit Tests
2 on:
3   push:
4     branches:
5       - main
6   pull_request:
7     branches:
8       - main
9
10 jobs:
11   test:
12     name: Run unit tests in containers
13     runs-on: ubuntu-latest
14     strategy:
15       fail-fast: false
16     matrix:
17       scenarios:
18         - name: Run unit tests with python3.9 on el8
19           python: python3.9
20           container: ubi8
```

How?

- ▶ GitHub Checks interface

Release 0.15.0 #795

Merged

MichalHe merged 1 commit into `oamg:main` from `pirat89:new-release` on Aug 23

Conversation 2

Commits 1

Checks 12

Files changed 6



Release 0.15.0 558d299

Unit Tests

on: pull_request

✓ Run unit tests in containers (Ru...

✓ Run unit tests in containers (Ru...

✓ Run unit tests in containers (Ru...

Packit-as-a-Service

✓ rpm-build:epel-7-ppc64le

✓ rpm-build:epel-7-x86_64

✓ rpm-build:epel-8-x86_64

Run unit tests in containers (Run unit tests with python3.9 on el8, python3.9, ubi8)

succeeded on Aug 23 in 1m 43s

> ✓ Set up job

> ✓ Checkout code

> ✓ Set main to origin/main

✓ Run unit tests with python3.9 on el8

```
1 ▶ Run script -e -c /bin/bash -c 'TERM
opt=seccomp=unconfined -t leapp-test
tests/Containerfile.ubi8 res/contain
```

git hooks

- ▶ “Git has a way to fire off custom scripts when certain important actions occur.”
- ▶ `.git/hooks/`
- ▶ Client side, Server side hooks

pre-commit

- ▶ <https://pre-commit.com/>
- ▶ Run checks & linters locally
- ▶ Fix code before committing
- ▶ Run them during the CI process as well
- ▶ `ls .git/hooks/pre-commit`

queued at: 2022-10-19 11:19:09

total time to completion: 29.6s

- queue (93ms)
- mergeable check (910ms)
- clone (735ms)
- ci config (1ms)
- pull image (196µs)
- build (2.9s)
- download (3.2s)
- run (21.8s)

```

pyupgrade..... Passed
black..... Passed
prettier..... Passed
check for added large files..... Passed
check python ast..... Passed
check builtin type constructor use..... Passed
check docstring is first..... Passed
check that executables have shebangs..... Passed
check for merge conflicts..... Passed
check for broken symlinks..... Passed
check yaml..... Passed
detect private key..... Passed
fix end of files..... Passed
mixed line ending..... Passed
trim trailing whitespace..... Passed
flake8..... Passed
mypy..... Passed

```

Efficient git

git aliases

- ▶ `git config --global --add alias.co commit`
- ▶ `$ cat ~/.gitconfig`
[alias]
 co = commit
- ▶ `git co -m 'this is a commit'`
- ▶ less typing \o/

shell aliases

- ▶ `$ cat ~/.bashrc # or ~/.your-shell-rc`
- ▶ `alias g=git`
`alias gc=git commit --verbose`
- ▶ `g co -m 'this is a commit'`
- ▶ `gc -m 'this is also a commit'`
- ▶ even less typing
`\ /`
`\o/`

git blame

display of author metadata attached to specific committed lines in a file

git blame NAME_OF_THE_FILE

git who, git history, git praise → how to add an alias?

git blame

display of author metadata attached to specific committed lines in a file

git blame NAME_OF_THE_FILE

git who, git history, git praise → how to add an alias?

git config --global alias.history blame

git config --global alias.who blame

git config --global alias.praise blame

git status in your shell prompt (Fedora)

- ▶ [tt@cashew]\$ cat ~/.bashrc
export GIT_PS1_SHOWDIRTYSTATE=y
source /usr/share/git-core/contrib/completion/git-prompt.sh
export PS1='[\u@\h \W\$(__git_ps1 " (%s)")]\\$ '
- ▶ [tt@cashew git-repo (oct-copr-storage-move *)]\$ vim README.md

- ▶ no typing

```
\      /  
 \    /  
  \o/
```

Text-mode interface for git

- ▶ `$ tig`

- ▶ <https://github.com/jonas/tig>

- ▶ `git log`, `git add`, `git branch -a`, `git blame` on steroids

Check out a pull request locally (why)

- ▶ Why?
 - Review the code locally
 - Try the code
 - Cherry-pick commits

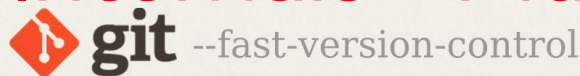
Check out a pull request locally (how)

- ▶ `$ cat ~/.gitconfig`
`[remote "upstream"]`
`fetch = +refs/pull/*/head:refs/remotes/upstream/pr/*`
`fetch = +refs/merge-requests/*/head:refs/remotes/upstream/mr/*`
- ▶ `git fetch origin pull/ID/head:BRANCH_NAME`
`git fetch origin pull/978/head:my_branch`
`git switch my_branch`

fetch all by default

fetch only one

Git Internals – Plumbing and Porcelain



About

Documentation

Reference

Book

Videos

External Links

Downloads

Community

This book is available in [English](#).

Full translation available in

[azərbaycan dili](#),

[български език](#),

[Deutsch](#),

Chapters ▾ 2nd Edition

10.1 Git Internals - Plumbing and Porcelain

You may have skipped to this chapter from a much earlier chapter, or you may have gotten here after sequentially reading the entire book up to this point — in either case, this is where we'll go over the inner workings and implementation of Git. We found that understanding this information was fundamentally important to appreciating how useful and powerful Git is, but others have argued to us that it can be confusing and unnecessarily complex for beginners. Thus, we've made this discussion the last chapter in the book so you could read it early or later in your learning process. We leave it up to you to decide.

Now that you're here, let's get started. First, if it isn't yet clear, Git is fundamentally a content-addressable filesystem with a VCS user interface written on top of it. You'll learn more about what this means in a bit.

git rebase my-branch // internals

- ▶ All changes made by commits in the current branch but that are not in `<upstream>` are saved to a temporary area. This is the same set of commits that would be shown by `git log <upstream>..HEAD`;
- ▶ The current branch is reset to `<upstream>`. This has the exact same effect as `git reset --hard <upstream>`. `ORIG_HEAD` is set to point at the tip of the branch before the reset.
- ▶ The commits that were previously saved into the temporary area are then reapplied to the current branch, one by one, in order. Note that any commits in `HEAD` which introduce the same textual changes as a commit in `HEAD..<upstream>` are omitted (i.e., a patch already accepted upstream with a different commit message or timestamp will be skipped).
- ▶ It is possible that a merge failure will prevent this process from being completely automatic. You will have to resolve any such merge failure and run `git rebase --continue`. Another option is to bypass the commit that caused the merge failure with `git rebase --skip`. To check out the original `<branch>` and remove the `.git/rebase-apply` working files, use the command `git rebase --abort` instead.
- ▶ MUCH MORE INFO IN THE git-rebase MANPAGE

git Issues

git issues

a lightweight issue-tracking system that is available in all repositories

ansible / ansible Public Watch 2k

<> Code **Issues 663** Pull requests 332 Actions Projects 14 Security Insights

Filters Labels 602 Milestones 3 New issue

663 Open ✓ 30,212 Closed Author Label Projects Milestones Assignee Sort

- apt module with cache_valid_time does not update when cache was never updated before** affects_2.13 bug
module needs_verified P3
#79206 opened 17 hours ago by tumbl3w33d 3
- Update endoflife.date** affects_2.15 docs has_pr on_hold
#79190 opened 4 days ago by mariolenz 1 task done 2
- Link to meetup page on community page on docsite is 404** affects_2.15 docs docsite has_pr
#79188 opened 4 days ago by felixfontein 1 task done 2
- Unable to install collection with type: git** affects_2.13 bug has_pr P3
#79168 opened 6 days ago by kennedyjosh 1 task done 1
- Documentation about required installed tools on the remote host?** affects_2.15 docs has_pr 1 9
- Unexpected exception while installing collections from requirements.yml file** affects_2.13 bug has_pr traceback 1 6

#79109 opened 14 days ago by kennedyjosh 1 task done

Red Hat

git issues






a lightweight issue-tracking system that is available in all repositories

- ▶ Small? Go defaults
- ▶ More traffic? Introduce templates, labels
 - Avoid duplicates
 - Add structure, be specific
- ▶ Vulnerability reports
- ▶ Triage and close issues
 - Labels
 - By responsibility
 - Non triages
 - NeedInfo
 - HelpWanted
 - Mention people
 - Assign

git issues

ansible / ansible Public Watch 2k

<> Code **Issues 663** Pull requests 332 Actions Projects 14 Security Insights

-  **Bug report**
Create a report to help us improve [Get started](#)
-  **Documentation Report**
Ask us about docs [Get started](#)
-  **Feature request**
Suggest an idea for this project [Get started](#)
- Report a security vulnerability**
Please review our security policy for more details [View policy](#)
-  **Security bug report** 🙏
Please learn how to report security vulnerabilities here. For all security related bugs, email security@ansible.com instead of using this issue tracker and you will receive a prompt response. For more information, see https://docs.ansible.com/ansible/latest/community/reporting_bugs_and_features.html [Open](#)
-  **Ansible Code of Conduct**
♥ Be nice to other members of the community. © Behave. [Open](#)

Lab

- ▶ <https://gitlab.com/redhat/research/mastering-git/-/blob/main/labs/lab6.md>

Class 6 homework

<https://gitlab.com/redhat/research/mastering-git#class-6-homework>

Questions to answer

1. How many commits were merged into main between September 1st and October 30th 2023?
2. Who authored commit "Merge branch 'hw03' into 'main'" and what is the commit body?
3. Find one commit that you think should have better title or body and tell us how and why?
 - If you are the author, that's amazing :)
4. Show us a commit that DID NOT change README.md in the month of September?
5. How many unique contributors the repository has at October 31st?
6. How many commits differ between main and main-oct-9?
7. Which commit added this line?

****Grading****: successfully complete 5 mandatory homeworks, or 4/5 homeworks plus a bonus task

Class 6 homework

- ▶ Push a single commit with answers in your fork of mastering-git in a text file <UCO>.txt in class6_homework/ directory in a branch named class6-homework
- ▶ No MR
- ▶ For every question, send us all commands you ran to find out the answer.
- ▶ The file syntax:

```
# Question 1
command 1
command 2
Answer/Text comment
---
# Question 2
command 2
Answer
---
Question 3
```


Beer/Lemonade with the teachers

- ▶ Make a group Picture
- ▶ Today
- ▶ 18:00
- ▶ U drevaka

THE END



THANK YOU!