Webapp tips +
Distributed version control

CS-214 - 12 Nov 2025

Clément Pit-Claudel

Le retour du Git

Quick announcements

Webapp lecture
is up as a literate program

Unguided lab teams
are due this Friday

Exam grades
Soon™

Team-finding session
Friday 5PM at the front of INF3

RPS tests
Are integration tests!

How do you test code without overspecifying it?

Most other labs: Unit tests or simple integration

- One function at a time
- Single input, single output

```
test("play once"):
  assertEq(
    update(State(…), 0, 1),
    State(board = …,
          playerId = UID1)
  )
```

RPS lab: (Complex) integration tests

- Multiple functions
- Indirect tests (views+events)

```
test("play once"):
  val st = transition(
    init(UIDS), Move(0, 1))
  val vw = project(st)(UID0)
  assertEq(vw.playerId, UID1)
  (0 to 3).forall: r ⇒
    (0 to 3).forall: c ⇒
      assert(…vw.board(r, c)…)
)
```

How do you test code without overspecifying it?

Most other labs: Unit test

- One function at a time
- Single input, single output

```
test("encode"):
  assertEq(
    encode(…),
    Obj("abc" → …, "def" → …)
  )
```

RPS lab: (Complex) integration tests

- Multiple functions
- Indirect tests (views+events)

```
test("decode-encode"):
  assertEq(
    xyz,
    decode(encode(xyz))
  )
)
```

- Will this work if you use arrays or classes instead of Vectors and case classes?
- What kind of bugs will this fail to catch?

Tips to be successful in the unguided lab

- For `webapp-rps`
  - Do the exercises
  - Look around webapp-lib too! (`Wires.scala`)
  - Write your own tests
- For the unguided lab
  - Don't skip `webapp-rps`
  - Review Monday's lecture
  - Plan and communicate (plan first, code second)
  - Need proposal tips? **Ask in person**

Practice: Design an online card game

- **States**
- **Transition function**
- **Events**
- **Views**
- **Projection function**

Practice: Design an online card game

- **States**
  `Playing, Finished`
  + Game-specific (Jass: `Bidding`, Tarot: `Écart...`)
- **Transition function**
  `Playing → Playing, Playing → Finished`
  + Game-specific ones (`Bidding → Playing...`)
- **Events**
  `Bid, Pass, PlayCard`
- **Views**
  Game phase, individual hand
- **Projection function**
  Hide other players' cards

App suggestions for the unguided lab

The following would all make for reasonable webapps to build in a team of three or four. Be creative! Let's not have 50 clones of chess, please.

- **Turn-based board games**: Games with geometric boards, such as memory, reversi, go, etc. work best, because they have simple UIs. Card games work great too, especially if you don't overthink the UI (emojis work great: ♣♠♥♦ 🃏 ♠♥♦♣♤♡♢♧🂠🂡🂢🂣🂤🂥🂦🂧🂨🂩🂪🂫🂬🂭🂮🂱🂲🂳🂴🂵🂶🂷🂸🂹🂺🂻🂼🂽🂾🃁🃂🃃🃄🃅🃆🃇🃈🃉🃊🃋🃌🃍🃎🃑🃒🃓🃔🃕🃖🃗🃘🃙🃚🃛🃜🃝🃞🃟 🃏 🂠).
- **Adventure games**: Interactive fiction and roguelikes (e.g.) work best. Make sure that your game is meaningfully multiplayer, and don't spend all your time on the art.
- **Productivity and planning**: Task lists, shopping lists, apartment chores schedulers, restaurant bill splitters, real-time voting and quiz apps, personal calendars, party planners, trip planners, tournament planners…
- **Sharing and collaboration**: Private photo albums, book clubs, collaborative music/art/dance/theater/… apps, flashcards for collaborative study, neighborhood borrow/exchange/buy/sell/give away platforms…

Today:

Distributed version control

Learning objective:

Handle divergence between codebases and resolve conflicts

- Single-user git recap
- Distributed Git basics
  - Remotes
  - Fetching
  - Branches
- Managing short-term divergence
  - Patches
  - Cherry-picks
  - Rebases
- Managing long-term divergence
  - Branching
- Handling conflicts
  - 3-way diffs
  - Conflict resolution

Single-user Git recap

```
git help
git config
mkdir
sbt new scala/scala3.g8
git init
git status
git add
git restore
git commit -m
git log
git show
.gitignore
git commit –amend
git tag -m “…”
git clone
git format-patch
git shortlog --since
git blame
git log -L
```

The problems with backups

- No meaningful changesets
- Too-large or too-small granularity
- Limited history browsing capabilities
- **Limited support for asynchronous collaboration**

The solution? Use a VCS!

Version Control Systems (VCSs), or
Source Code Management systems (SCMs) are for:

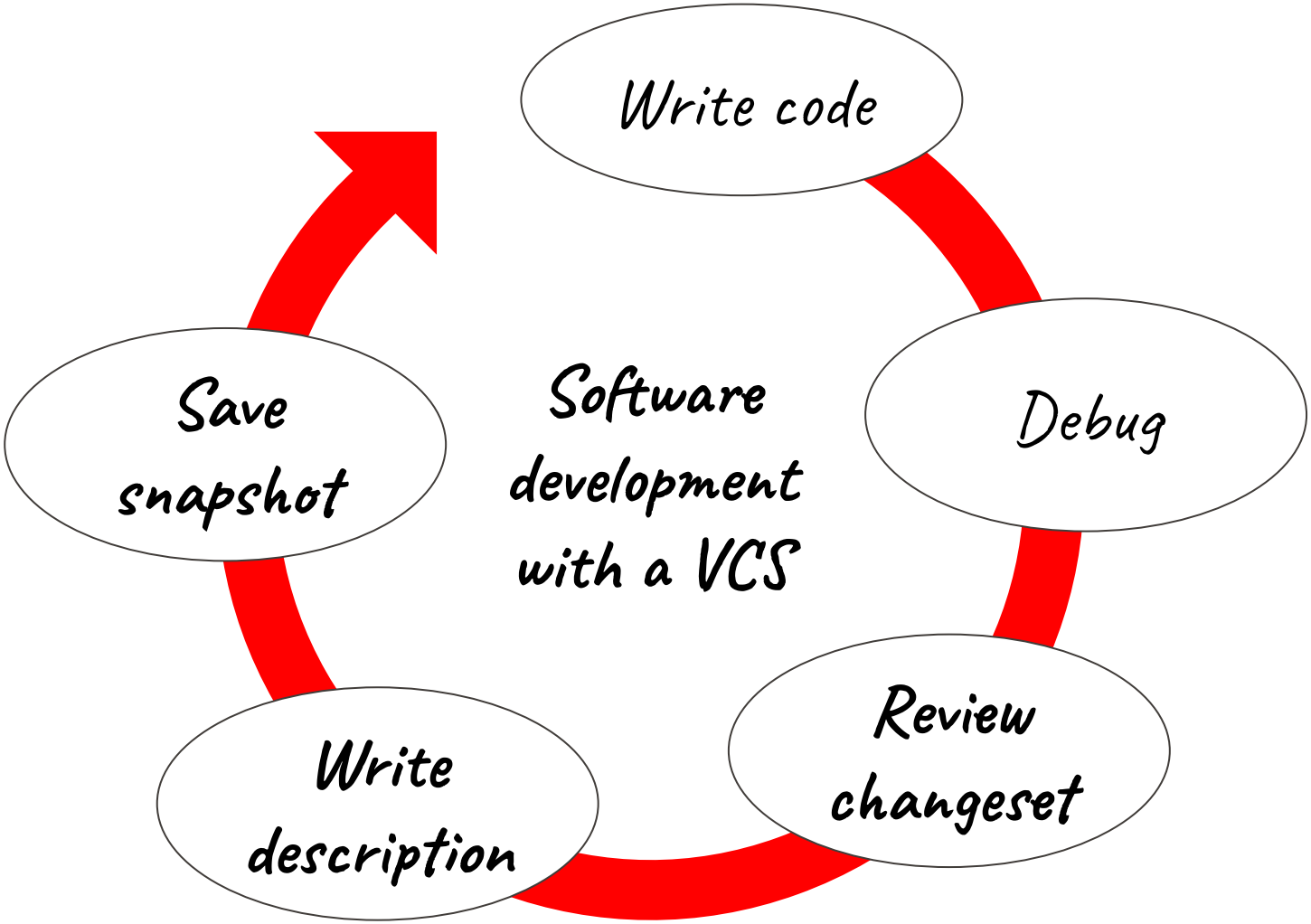
- Tracking the evolution of text or code ← *today*
- **Managing versions, distributed development ←*next time***

But not directly for:

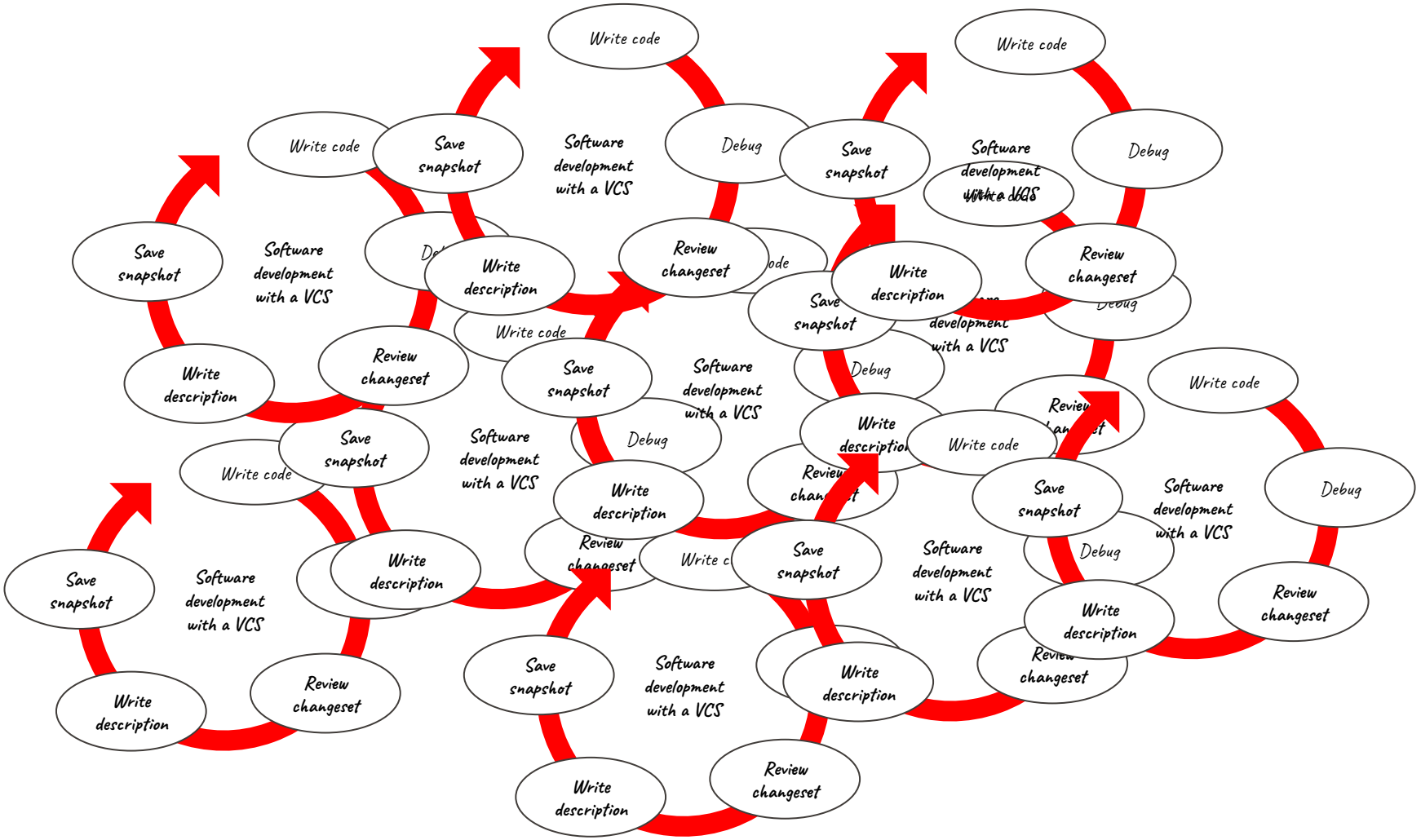
- Bug tracking, code review, ← *next year*
- Testing, CI, deployment ← *mostly next year*

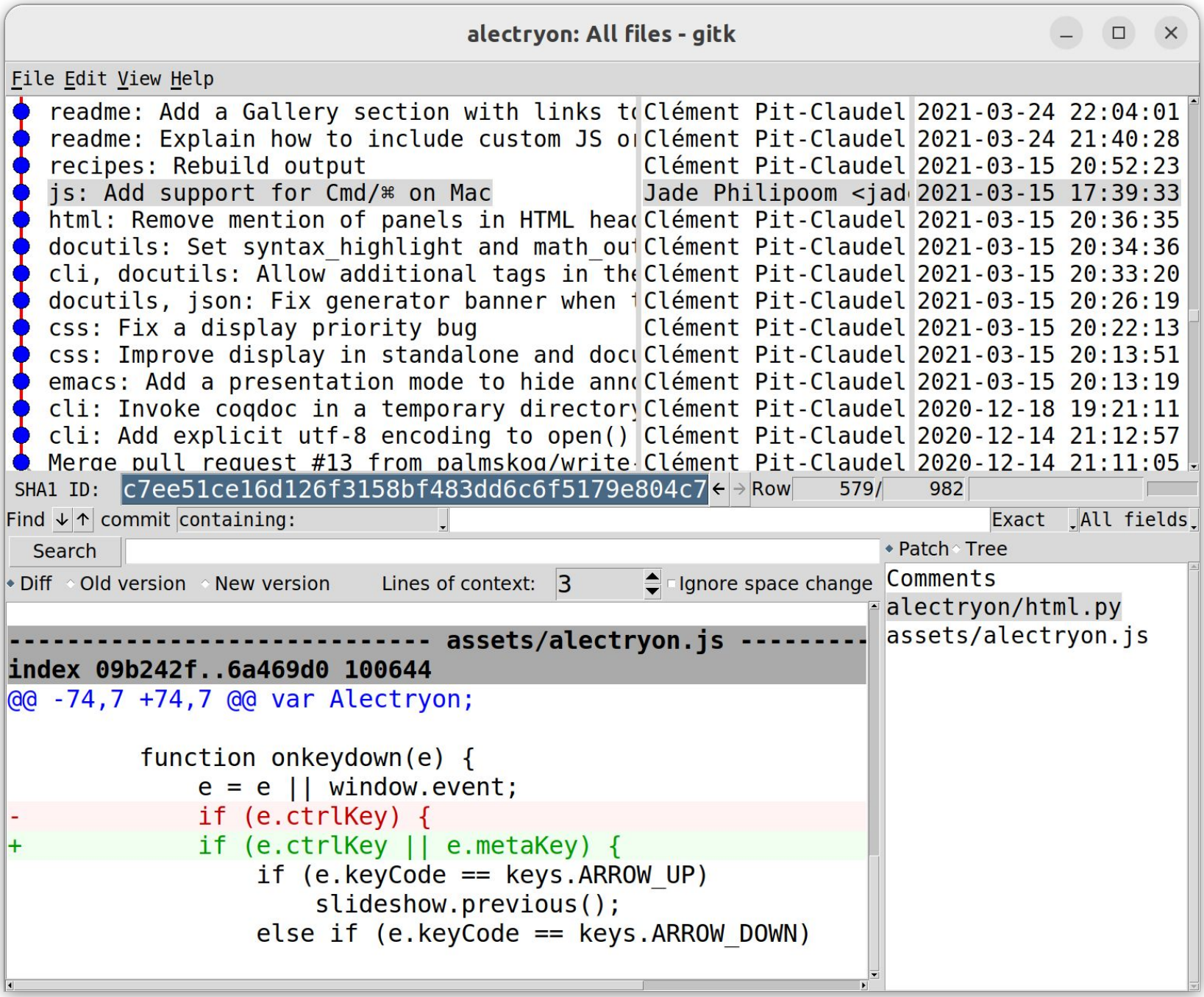
Git ≠ Github / Gitlab

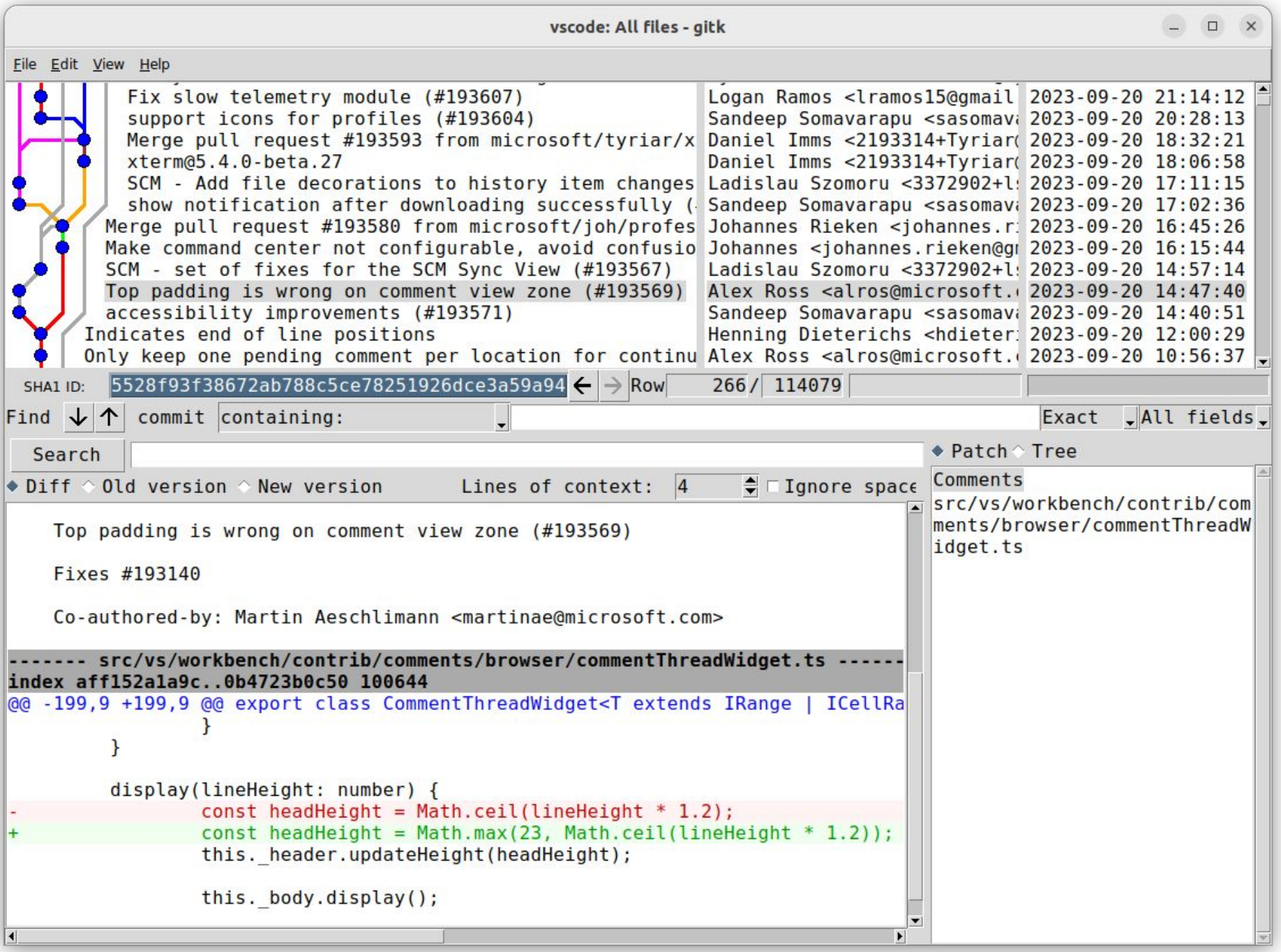
Version control is **labeled backups**



Version control is **distributed**, **labeled backups**




alectryon: All files - gitk
File Edit View Help
readme: Add a Gallery section with links t Clément Pit-Claudel 2021-03-24 22:04:01
readme: Explain how to include custom JS o Clément Pit-Claudel 2021-03-24 21:40:28
recipes: Rebuild output Clément Pit-Claudel 2021-03-15 20:52:23
js: Add support for Cmd/⌘ on Mac Jade Philipoom <jad 2021-03-15 17:39:33
html: Remove mention of panels in HTML hea Clément Pit-Claudel 2021-03-15 20:36:35
docutils: Set syntax_highlight and math_ou Clément Pit-Claudel 2021-03-15 20:34:36
cli, docutils: Allow additional tags in th Clément Pit-Claudel 2021-03-15 20:33:20
docutils, json: Fix generator banner when Clément Pit-Claudel 2021-03-15 20:26:19
css: Fix a display priority bug Clément Pit-Claudel 2021-03-15 20:22:13
css: Improve display in standalone and doc Clément Pit-Claudel 2021-03-15 20:13:51
emacs: Add a presentation mode to hide ann Clément Pit-Claudel 2021-03-15 20:13:19
cli: Invoke coqdoc in a temporary directory Clément Pit-Claudel 2020-12-18 19:21:11
cli: Add explicit utf-8 encoding to open() Clément Pit-Claudel 2020-12-14 21:12:57
Merge pull request #13 from palmskog/write Clément Pit-Claudel 2020-12-14 21:11:05
SHA1 ID: c7ee51ce16d126f3158bf483dd6c6f5179e804c7
Row 579/ 982
Find commit containing:
Exact
All fields
Search
Patch
Tree
Diff
Old version
New version
Lines of context: 3
Ignore space change
Comments
alectryon/html.py
assets/alectryon.js
---------- assets/alectryon.js ---------
index 09b242f..6a469d0 100644
@@ -74,7 +74,7 @@ var Alectryon;
function onkeydown(e) {
e = e || window.event;
- if (e.ctrlKey) {
+ if (e.ctrlKey || e.metaKey) {
if (e.keyCode == keys.ARROW_UP)
slideshow.previous();
else if (e.keyCode == keys.ARROW_DOWN)


vscode: All files - gitk
File Edit View Help
Fix slow telemetry module (#193607)
support icons for profiles (#193604)
Merge pull request #193593 from microsoft/tyriar/x
xterm@5.4.0-beta.27
SCM - Add file decorations to history item changes
show notification after downloading successfully (
Merge pull request #193580 from microsoft/joh/profes
Make command center not configurable, avoid confusio
SCM - set of fixes for the SCM Sync View (#193567)
Top padding is wrong on comment view zone (#193569)
accessibility improvements (#193571)
Indicates end of line positions
Only keep one pending comment per location for continu
Logan Ramos <lramos15@gmail 2023-09-20 21:14:12
Sandeep Somavarapu <sasomava 2023-09-20 20:28:13
Daniel Imms <2193314+Tyriar(2023-09-20 18:32:21
Daniel Imms <2193314+Tyriar(2023-09-20 18:06:58
Ladislau Szomoru <3372902+l 2023-09-20 17:11:15
Sandeep Somavarapu <sasomava 2023-09-20 17:02:36
Johannes Rieken <johannes.r 2023-09-20 16:45:26
Johannes <johannes.rieken@gr 2023-09-20 16:15:44
Ladislau Szomoru <3372902+l 2023-09-20 14:57:14
Alex Ross <alros@microsoft. 2023-09-20 14:47:40
Sandeep Somavarapu <sasomava 2023-09-20 14:40:51
Henning Dieterichs <hdieter 2023-09-20 12:00:29
Alex Ross <alros@microsoft. 2023-09-20 10:56:37
SHA1 ID: 5528f93f38672ab788c5ce78251926dce3a59a94
Row 266 / 114079
Find commit containing:
Exact All fields
Search
Patch Tree
Diff Old version New version Lines of context: 4 Ignore space
Comments
src/vs/workbench/contrib/comments/browser/commentThreadWidget.ts
Top padding is wrong on comment view zone (#193569)
Fixes #193140
Co-authored-by: Martin Aeschlimann <martinae@microsoft.com>
------- src/vs/workbench/contrib/comments/browser/commentThreadWidget.ts ------
index aff152a1a9c..0b4723b0c50 100644
@@ -199,9 +199,9 @@ export class CommentThreadWidget<T extends IRange | ICellRa
}
}
display(lineHeight: number) {
- const headHeight = Math.ceil(lineHeight * 1.2);
+ const headHeight = Math.max(23, Math.ceil(lineHeight * 1.2));
this._header.updateHeight(headHeight);
this._body.display();

Source of divergence

- **Individual contributions happening at the same time**
- **Requirements of code review / quality processes**
  Most projects, even this class!
- **Long-time testing / evaluation of new features**
  Testing different approaches
- **Maintenance of older releases / LTS support**
  Python 2 / 3, most projects have a `next` branch
- **Long-lived forks / parallel development**
  Edge / Chromium

Risks of divergence

- Bitrot (code going stale)
- Complex merging
- Hard to use multiple unmerged features at once
- Uneven application of bugfixes

Today's lesson: Tools to deal with these issues!

Key question:

How do developers exchange and reconcile changes?

Distributed Git basics

- remotes
- fetching
- branches

4 important concepts

- “**remote**”
  `git remote -v`
  Another Git repository that you track
- “**fetching**”
  `git fetch`
  Retrieving all commits from a remote
- “**branches**”
  `git branch`, `git checkout`, `git switch`
  Labels on specific commits
- “**rebase/merge**”
  `git am`, `git cherry-pick`, `git rebase`, `git merge`
  Transferring code across branches

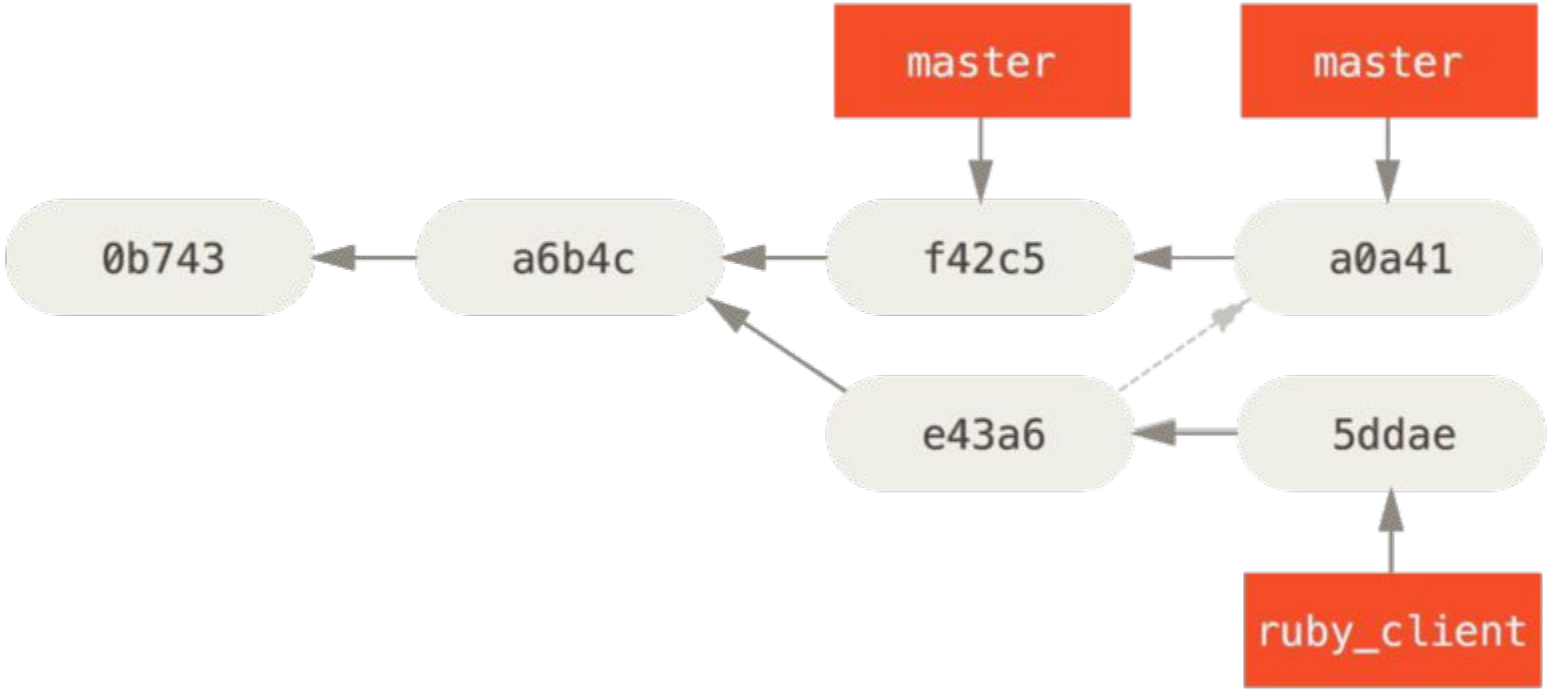
`pull` == `fetch` + `merge`

Short-term divergences

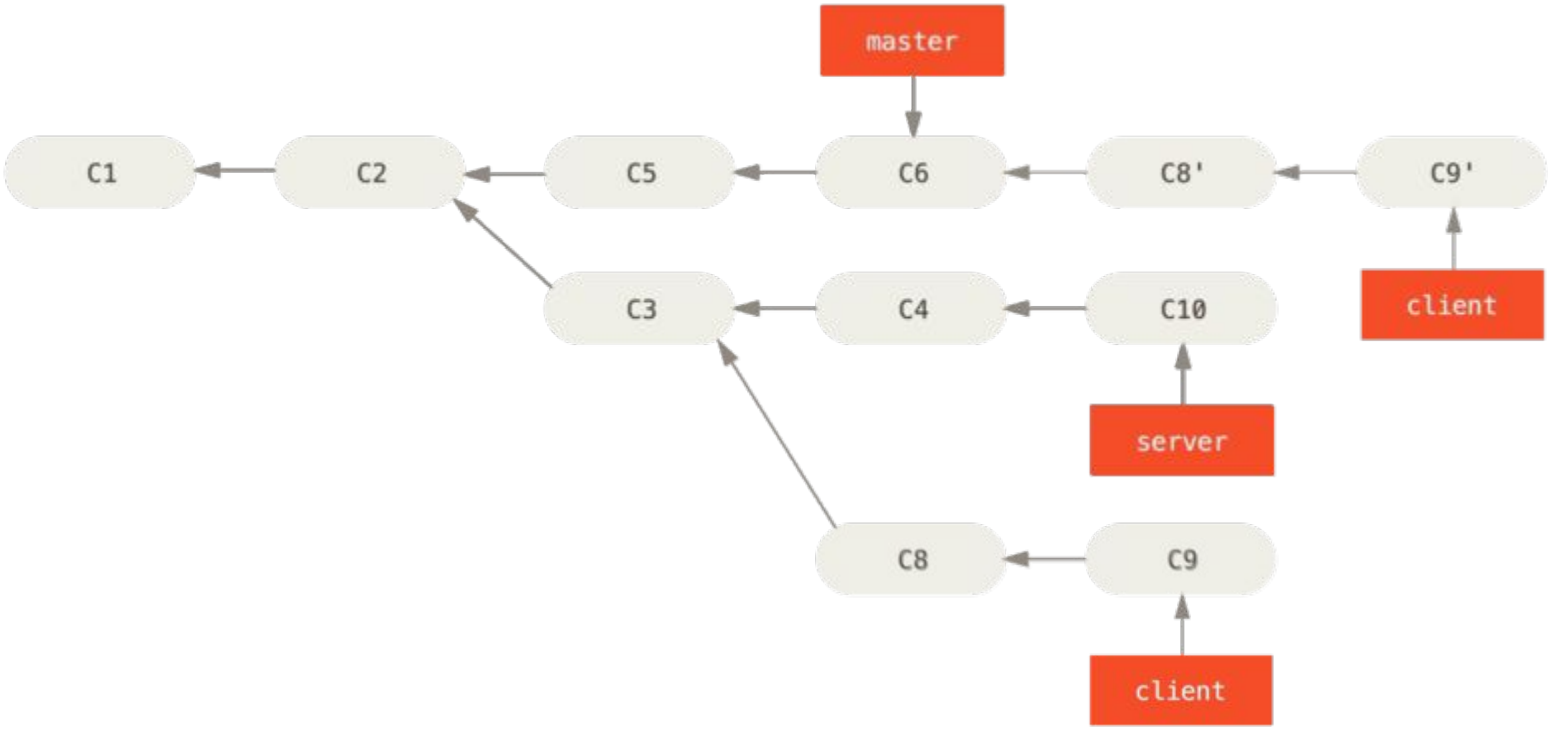
- Patches
- Cherry-picks
- Rebases

All illustrations taken from the Git book, which you should read!

Patching / cherry-picking



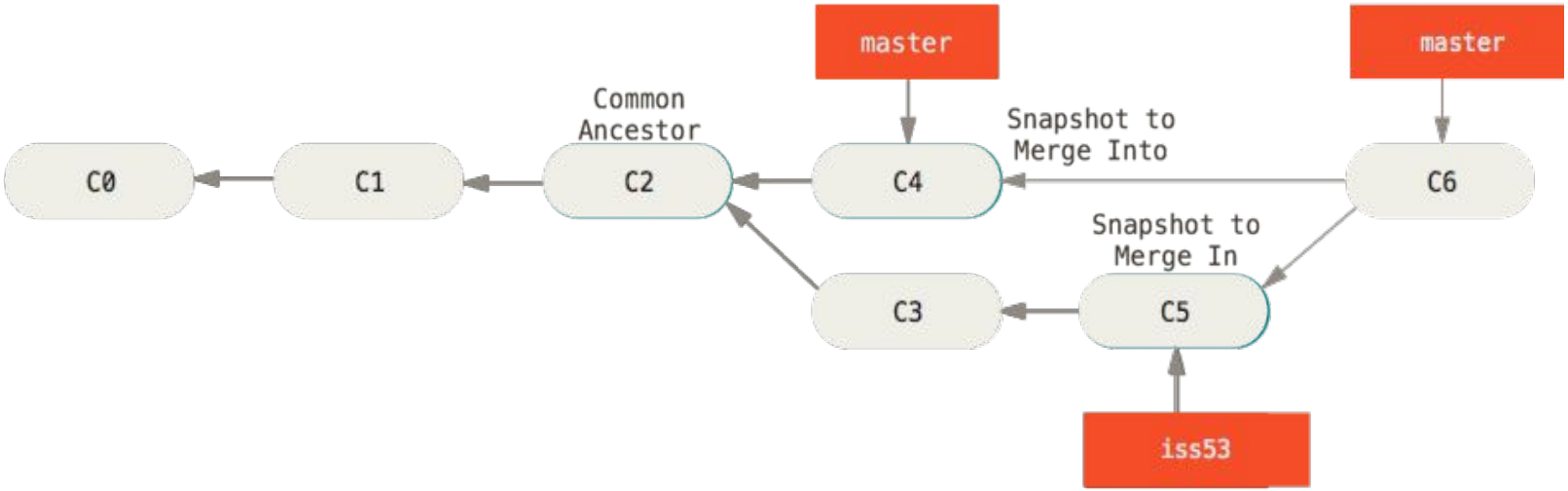
Rebasing



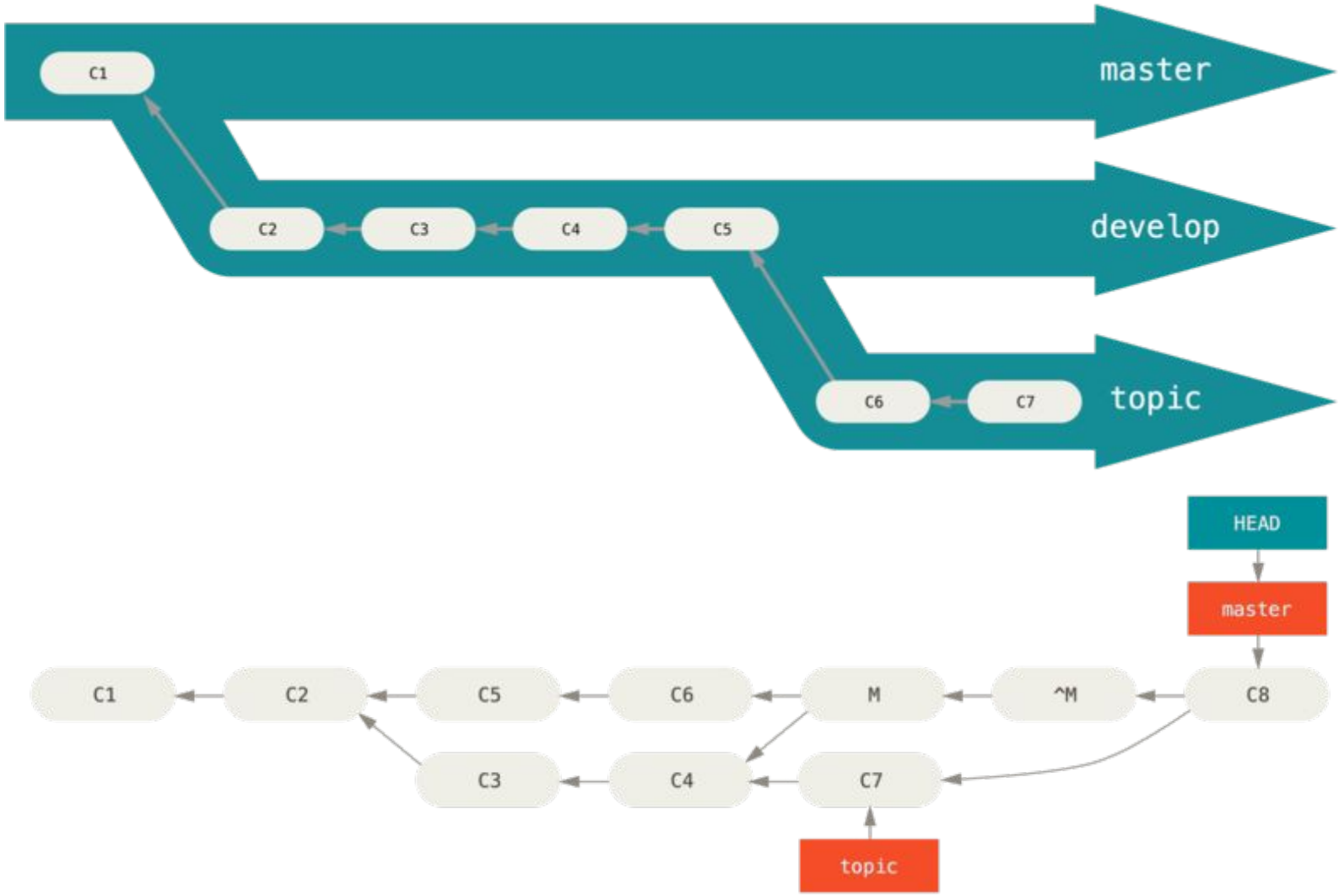
Long-term divergences

- Branch workflows
- Merging

Simple merges



Long lived branches and stability workflows



Dealing with conflicts

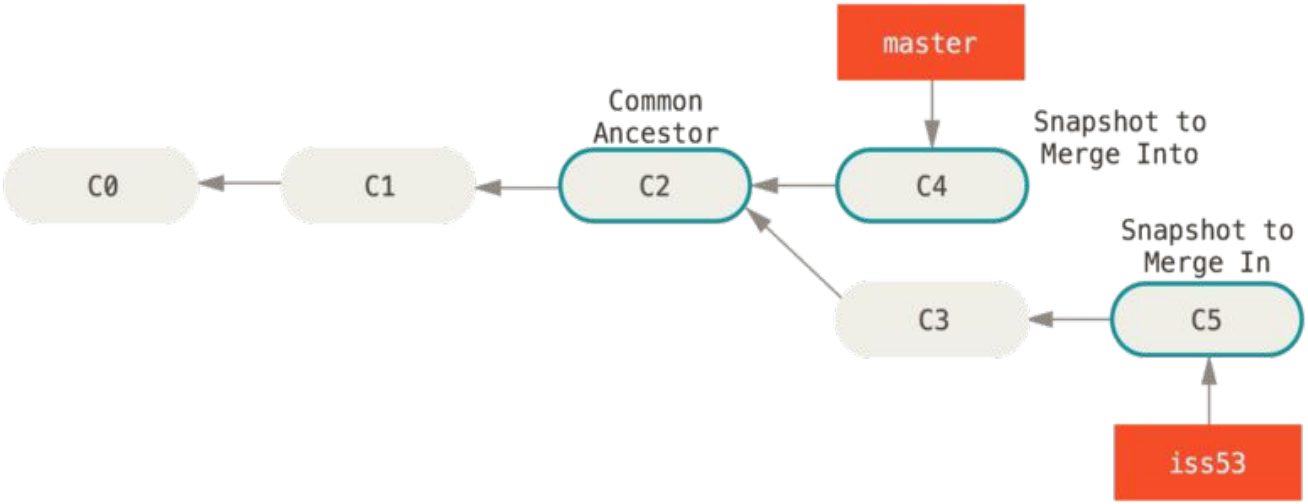
- 3-way diffs
- Resolving conflicts

Key configuration setting

```
git config --global merge.conflictstyle  diff3
```

(or)

```
git config --global merge.conflictstyle zdiff3
```



At-home topics

(on your own)

- Exercises
  - rebase, merge
  - conflicts handling
- Advanced topics:
  - Octopus merges
  - `git range-diff`
  - `git send-email`
  - `git request-pull`
  - `git rerere`