Distributed Version Control with  git

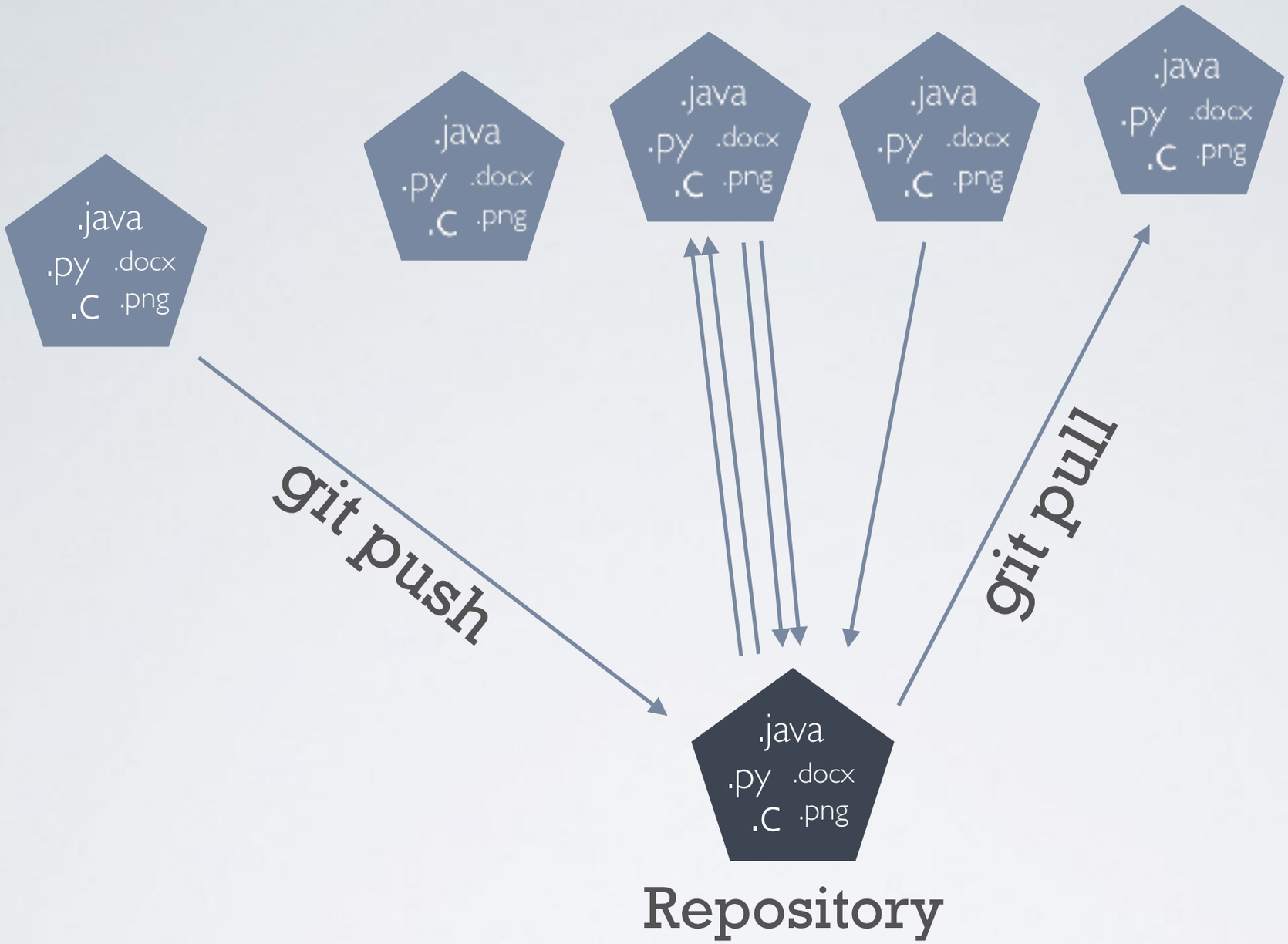
Gaute Solheim

Version control

- Keeps track of your files over time
- Aids teamwork and resolves inconsistency conflicts
- Lets you develop a project in several directions at once
- Is an important part of a project's documentation
- Dramatically reduces your FoSPFU (Frequency of Serious, Public Fuck-Ups)

git

- Distributed version control system
- Created by Linus Torvalds in 2005
- Open source (copyleft/non-permissive)
- Is awesome and <u>will</u> improve your workflow
- (Not the same as GitHub)



Distributed Version Control

Commit

- A point on the timeline that represents one change
- Can be rolled back to

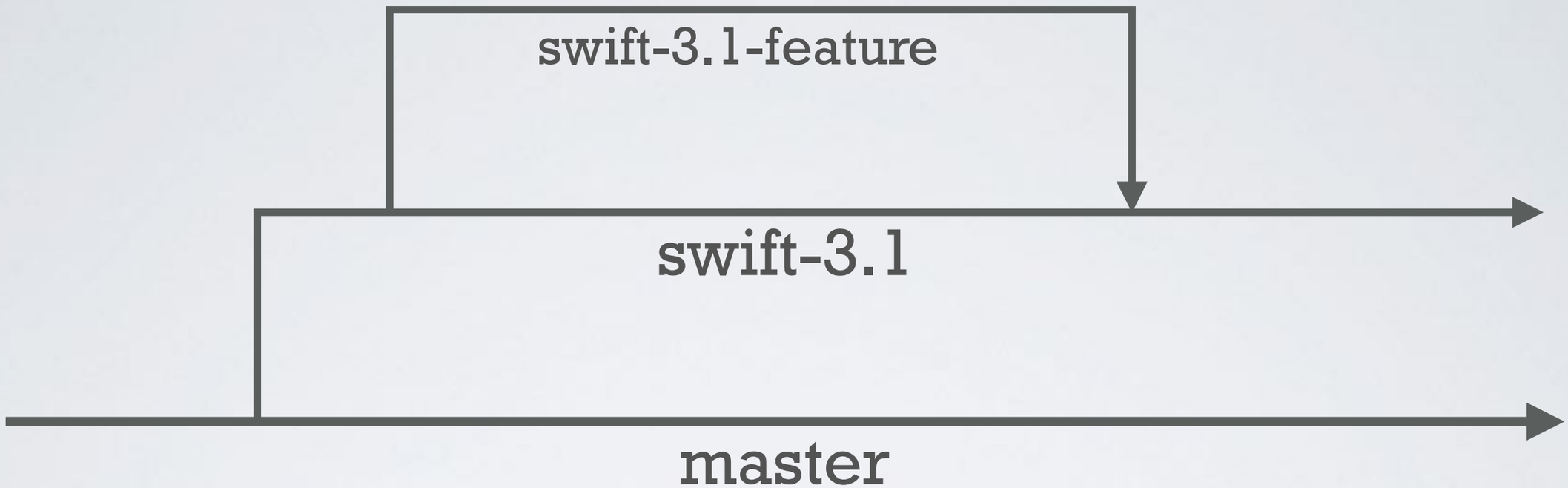
Action time!

Mission 1

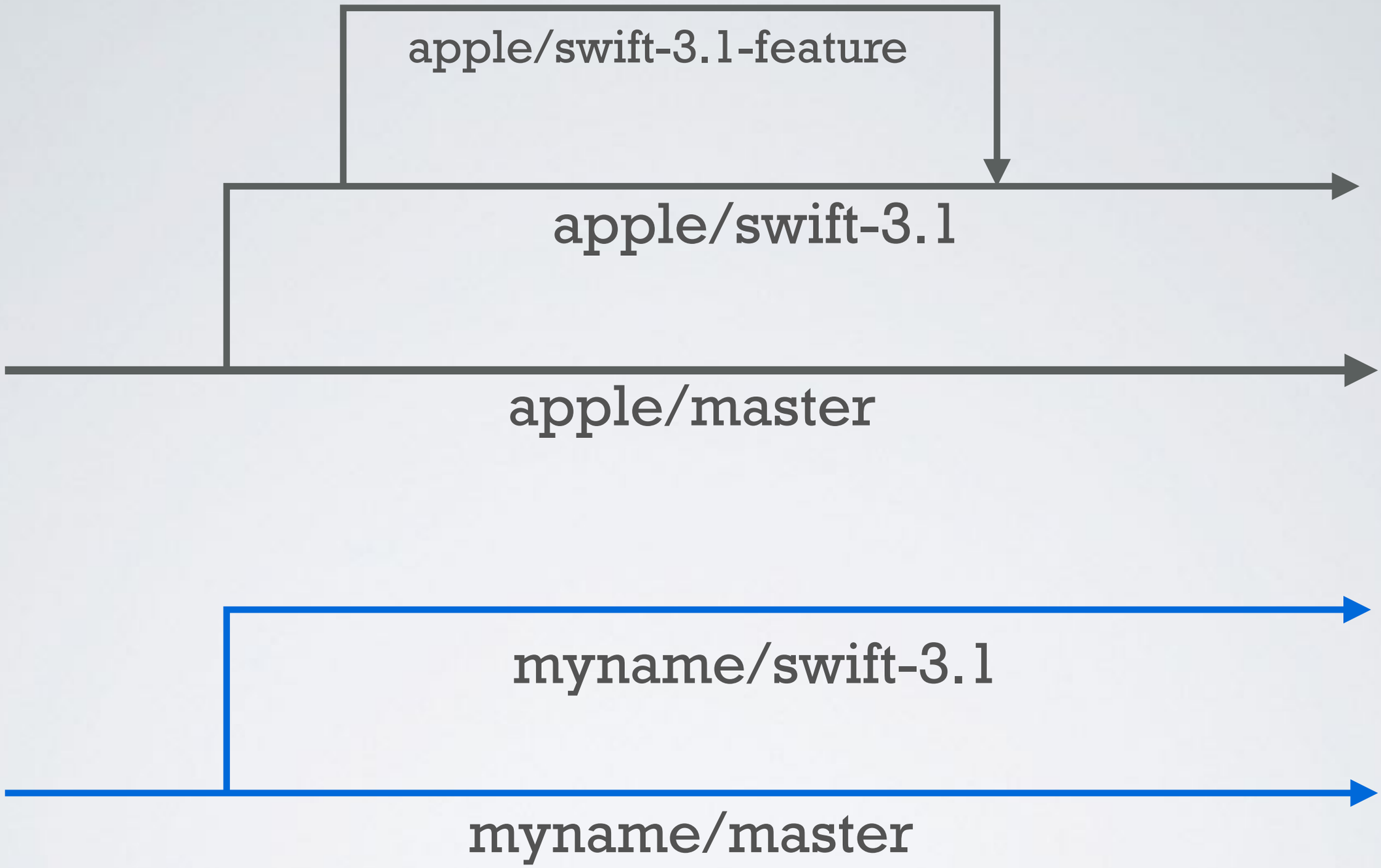
- Create your local clone of the repo:
  - `git clone` git@skiit.0b1.co`:git-init-demo`
  - `cd git-init-demo`
- View the commit history:
  - `git log`
- View the repository's status:
  - `git status`
- Add and commit a new file (.txt, .png, .docx, …)
- Push the changes to the remote repo:
  - `git push`
- Download other people's changes:
  - `git pull`

Repositories have…

- (Often) a .gitignore file
  - Patterns to ignore when adding files
- A .git folder
  - Everything git knows (file history, remotes, tags, branches)
- Remotes
  - Clones of and information about remote repositories (those you pull and push from)
- Branches…



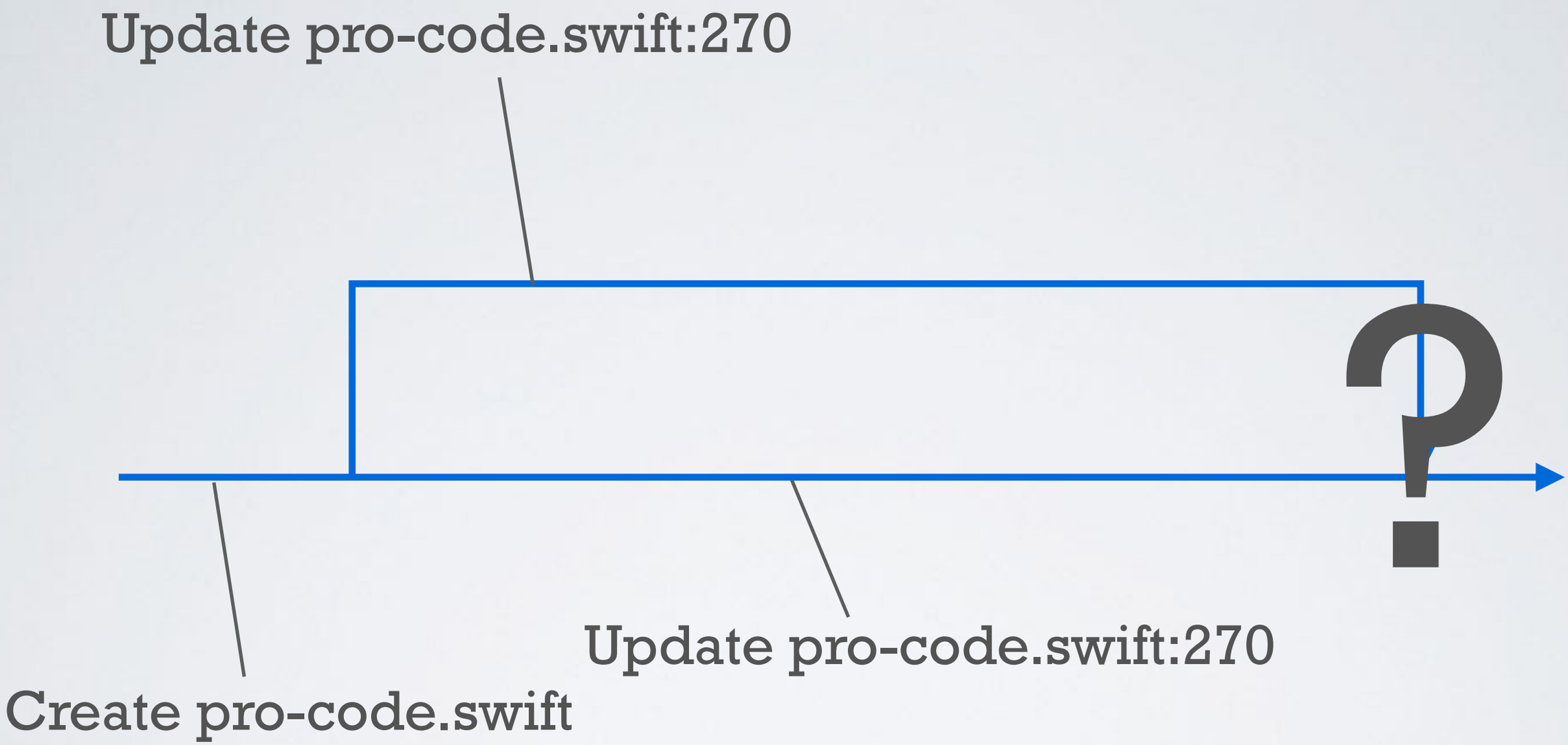
Branches



Branches and forks

Mission 2

- View all branches
  - `git branch`
- Check out one of them
  - `git checkout <branch-name>`
- Commit a change (pro tip: git commit -m «My commit message»)
- Check out master again
- Make sure you're back on master
  - `git branch/git status`
- Merge your changes into master
  - `git merge <branch-that's-not-master>`
- Bonus: create and check out a new branch from master
  - `git checkout -b <new-branch-name> master`
- Bonus: try to push the new branch



Merge conflicts

Demo

Diffs

- Shows detailed changes between two points on the repository timeline
- `git diff/git diff --staged`
- Pro tip: GitHub Desktop…

What does it do???

gautesolheim/SnapOnboarding-iOS

master | No Uncommitted Changes | History | Pull Request

Compare | Sync

applitude-workshop
applitude.github.io
CleanStore
CommonSnapFramework
dagens-backend
dagens-frontend-ios
gautesolheim.github.io
hashcode2015
minesweeper-console
one-word-pun-app
paletti-app
RazzleDazzle
skiit-backend
skiit-frontend
SnapFBSnapshotBase
SnapFonts-iOS
SnapImagePicker
SnapOnboarding-iOS
SnapOperationQueue
SnapPods
snapsale-design
snapsale-ios
SnapTagsView

Enterprise

applitude-assets
oblig-9

Revert changes from Swift 3 conversion
1 month ago by gautesolheim — 2

Update podspec version
1 month ago by gautesolheim

Merge branch 'swift3'
1 month ago by gautesolheim — 183

Fix rapid frame change on animation start
1 month ago by gautesolheim — 2

Update snapshot reference images
1 month ago by gautesolheim — 105

Integration adjustments for welcome back
1 month ago by gautesolheim — 100

Update tests for Swift 2.3 and update pods
2 months ago by gautesolheim — 45

Merge branch 'swift2.3'
2 months ago by gautesolheim

Add protocol methods for starting and stopping animations
2 months ago by gautesolheim — 8

Merge branch 'returning-users'
2 months ago by gautesolheim — 66

Update example with new delegate method
3 months ago by gautesolheim

Merge branch 'swift3'
gautesolheim 71aa53e 1 month ago

SnapOnboarding-iOS.podspec.json

```
@@ -1,5 +1,6 @@
{
  "name": "SnapOnboarding-iOS",
-  "version": "1.4.4",
+  "version": "1.5.1",
  "summary": "The first-launch onboarding used in Snapsale's iOS app.",
  "homepage": "https://github.com/skylib",
  "license": {
@@ -13,10 +13,10 @@
  "social_media_url": "http://twitter.com/snapsale_com",
  "source": {
    "git": "https://github.com/skylib/SnapOnboarding-iOS.git",
-    "tag": "1.4.4"
+    "tag": "1.5.1"
  },
  "platforms": {
-    "ios": 8.0
+    "ios": 9.0
  },
  "source_files": "SnapOnboarding-iOS/SnapOnboarding-iOS/**/*.{swift}",
  "resources": [
```

SnapOnboarding-iOS/.swift-version

Stash

- Lets you save your changes without committing them
- Works across branches

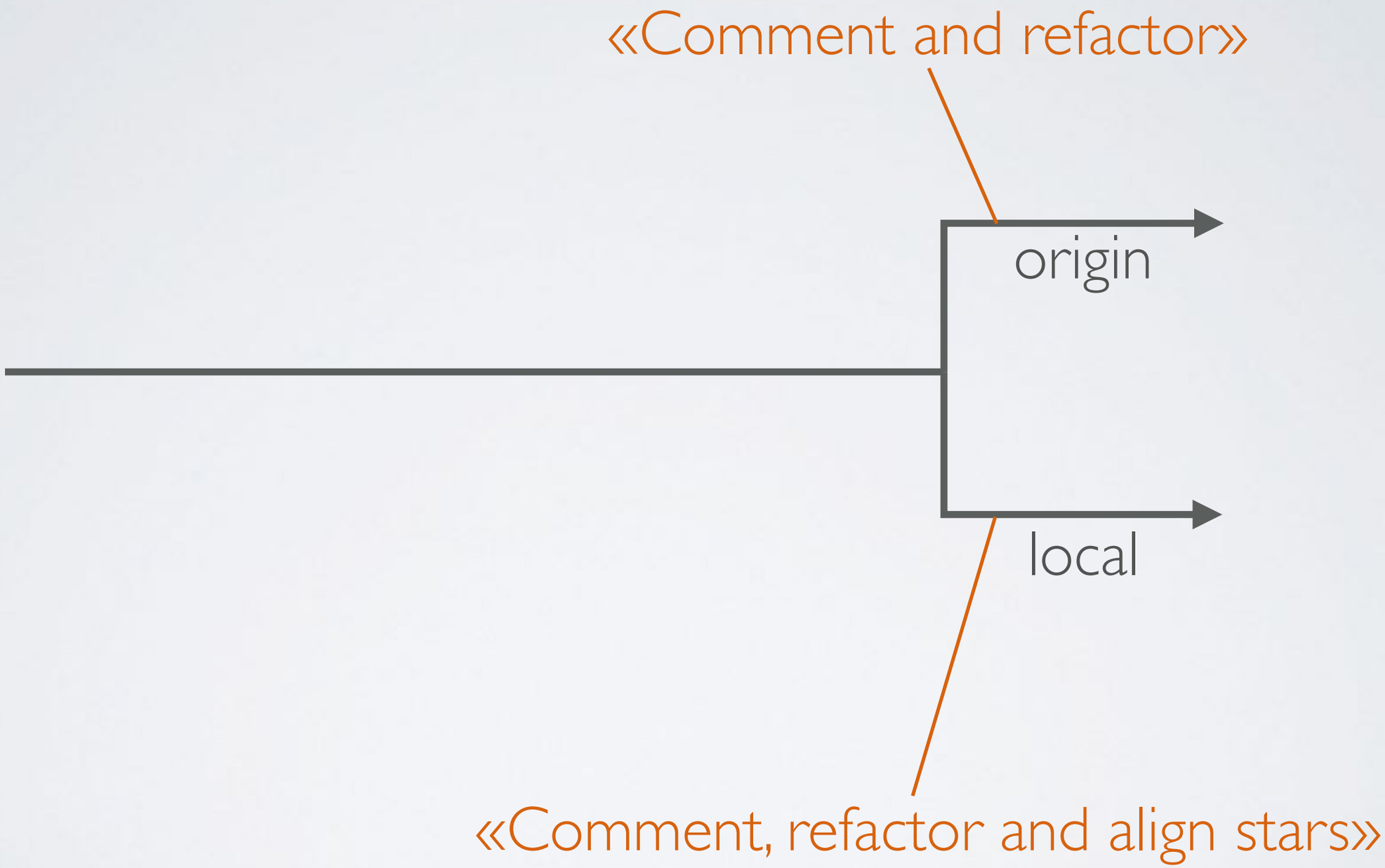
Mission 3

- View the current stash
  - `git stash list`
- Make a change and stash it
  - `git stash save <change-description>`
- Make sure the changes were stashed
  - `git stash list`
- Switch to another branch and apply the changes
  - `git stash apply stash@{0}`
- Delete the stash entry (hint: git help stash)

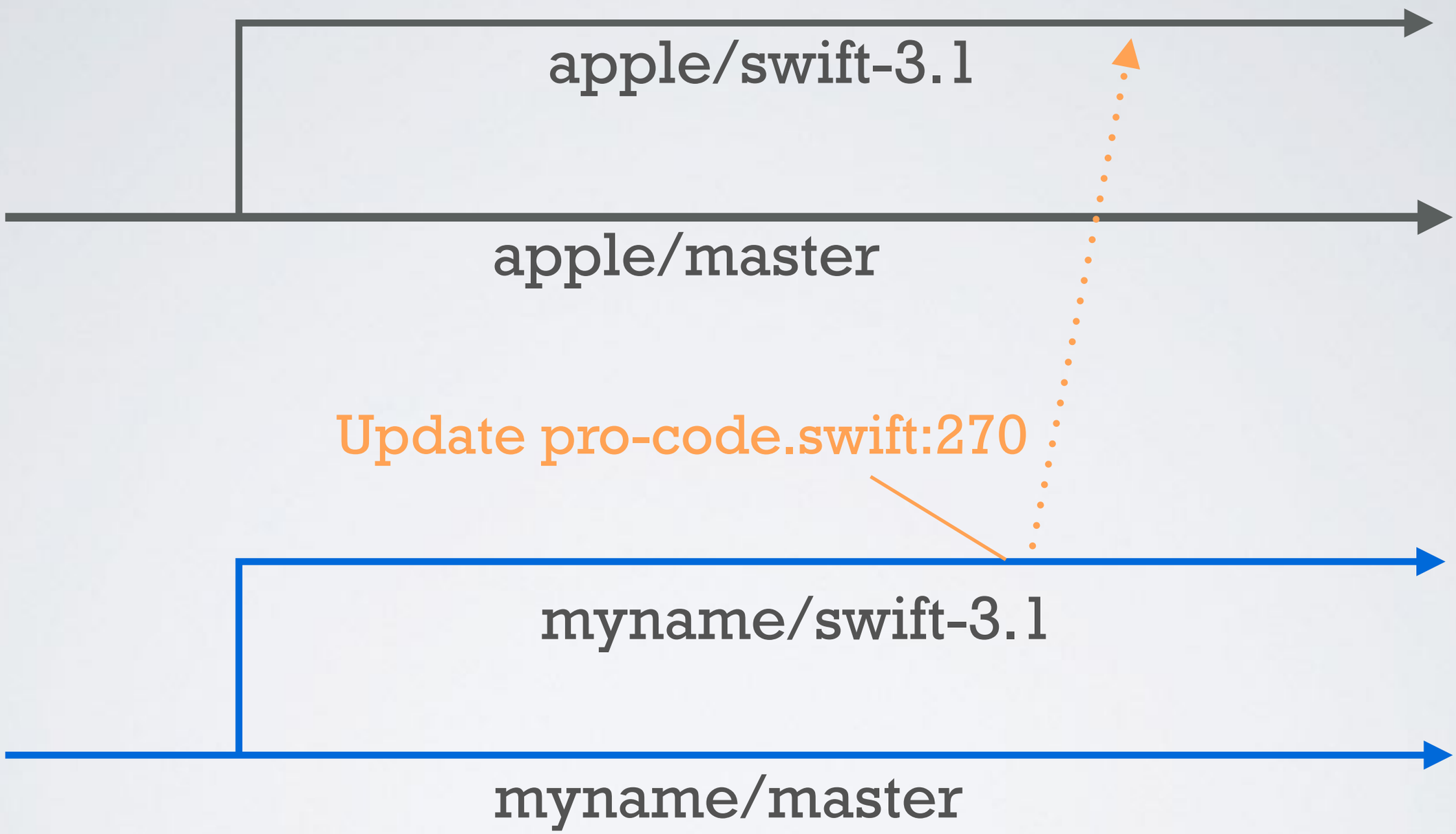
Rebasing

- Lets you rewrite the commit history
- Can only be done safely on your local (non-pushed) changes
- git rebase -i <earliest-commit-not-to-rebase>

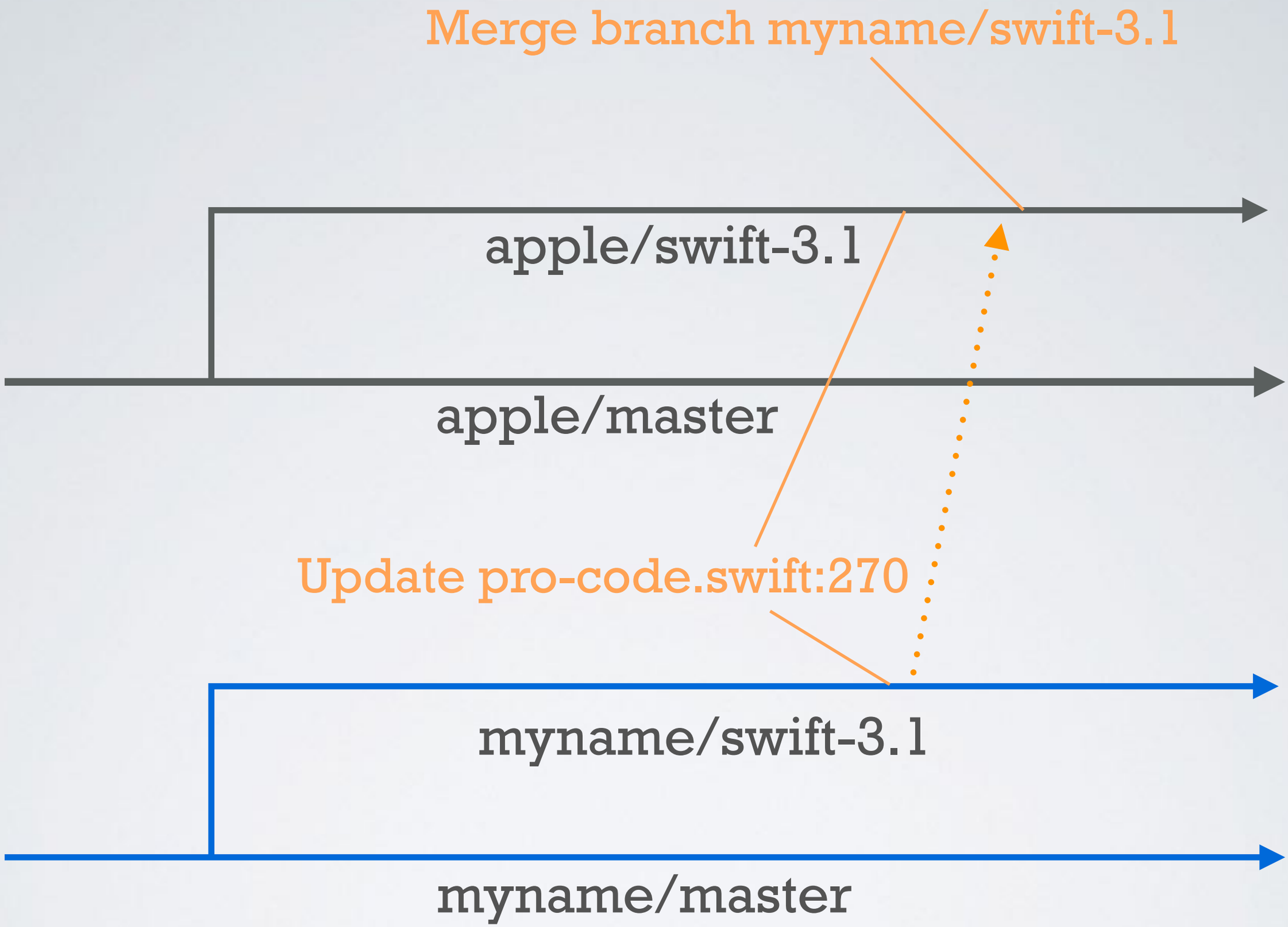
Demo


«Comment and refactor»
origin
local
«Comment, refactor and align stars»

Collaboration



Pull requests



Pull requests

apple / swift

Watch 2,411 | Unstar 35,595 | Fork 5,173

Code | Pull requests 131 | Projects 0 | Pulse | Graphs

Remove the `mutating` fixit for getter subscript #6245

Open **KingOfBrian** wants to merge 6 commits into `apple:master` from `KingOfBrian:bugfix/SR-3401`

Conversation 19 | Commits 6 | Files changed 2 | +24 −0

KingOfBrian commented a day ago

This PR removes the mutating fixit for subscript blocks that only contain a getter. When the fixit is performed on a subscript that contains an implicit get block, there's no valid source change to be made because the mutating keyword is not supported on implicit get blocks.

This does remove the fixit for explicit get blocks, which do work correctly. However the value of this fixit is marginal since mutating getters are not very common. I would like to be able to distinguish between these two cases if anyone has any thoughts.

This resolves SR-3401.

KingOfBrian added some commits 2 days ago

- `Add a unit test for mutating subscript in default getter block` 3da0a45
- `Do not suggest the mutating keyword for getter subscripts` 1cad6ca

slavapestov reviewed a day ago — View changes

`test/decl/subscript/addressors.swift` Show outdated

`lib/AST/Decl.cpp` Show outdated

Reviewers

slavapestov

CodaFi

jrose-apple

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

Notifications

Subscribe

Licenses

apple / swift

Watch 2,411 | Unstar 35,595 | Fork 5,173

Code | Pull requests 131 | Projects 0 | Pulse | Graphs

The Swift Programming Language https://swift.org/

45,321 commits | 32 branches | 112 releases | 427 contributors | Apache-2.0

Branch: master | New pull request | Create new file | Upload files | Find file | Clone or download

slavapestov committed on GitHub Merge pull request #6133 from slavapestov/argument-tuple-crap ... Latest commit a1b113f 33 minutes ago

.github	Reduce boilerplate in the GitHub PR template	4 months ago
apinotes	[APINotes] Source compatibility for nullability on array parameters. (#...	28 days ago
benchmark	Revert "SILOptimizer: Replace Array.append(contentsOf: with Array.app...	8 days ago
bindings/xml	Nesting parameter/returns/throws doc comments for closure parameters	8 months ago
cmake	Merge pull request #6219 from compnerd/icu-include-flags	a day ago

Licenses

- Default: You own your content and no one can use it publicly
- If you want people to be able to reuse your code, you need to add a license
- Two types
  - Permissive: Redistribute (almost) freely
  - Copyleft: Redistribute under the same terms

Choose an open source license

Which of the following best describes your situation?

I want it simple and permissive.

The **MIT License** is a permissive license that is short and to the point. It lets people do anything they want with your code as long as they provide attribution back to you and don't hold you liable.

jQuery, **.NET Core**, and **Rails** use the MIT License.

I'm concerned about patents.

The **Apache License 2.0** is a permissive license similar to the MIT License, but also provides an express grant of patent rights from contributors to users.

Android, **Apache**, and **Swift** use the Apache License 2.0.

I care about sharing improvements.

The **GNU GPLv3** is a copyleft license that requires anyone who distributes your code or a derivative work to make the source available under the same terms, and also provides an express grant of patent rights from contributors to users.

Bash, **GIMP**, and **Privacy Badger** use the GNU GPLv3.

choosealicense.com

More pro tips

- Working alone? Store the repo in Dropbox
  - Free continuous backups
  - No need to push/pull
- Running macOS?
  - .DS_Store's everywhere!
  - Solution: global gitignore
  - `echo .DS_Store >> ~/.gitignore_global`
  - `git config --global core.excludesfile ~/.gitignore_global`

Questions/ demo requests