



Do you really understand Git?

本当に分かる？

by Guille Polito, Pablo Tesone
@GuillePolito, @Tesonep

Inria



**Université
de Lille**

Why git ?

- Visibility
- Manage more than only code
- Lower entry barrier into the community
- Lots of existing tools
- Good branch support, good diff
- ...

Where would you place
yourself?



git

Where would you place yourself?



git

Where would you place yourself?



git

Where would you place yourself?



git

A Test

How do you ****clean**** your working copy?

\$



A Test

How do you ****clean**** your working copy?

```
$ git reset
```

This will reset your index



A Test

How do you ****clean**** your working copy?

```
$ git reset --hard
```

Well, yes, but not quite.
This will reset your working copy.
But not remove untracked files.



A Test

How do you ****clean**** your working copy?

```
$ git reset --hard
```

```
$ git clean -fd
```

FATALITY



git is not easy

- New terminology:
`index`, `HEAD`, `pull`, `push`, `fetch`
- False friends: `merge`, `commit`
- Command-line oriented
- Documentation is sometimes ambiguous, sometimes incomplete



git is not easy (2)

- Multiple workflows are a two-edged machete
There is no One, single ring to rule them all
- People mix workflows, philosophy and technical solutions (try reading some blogs...)
- Graphical tools are too general or too specific

An easy way to



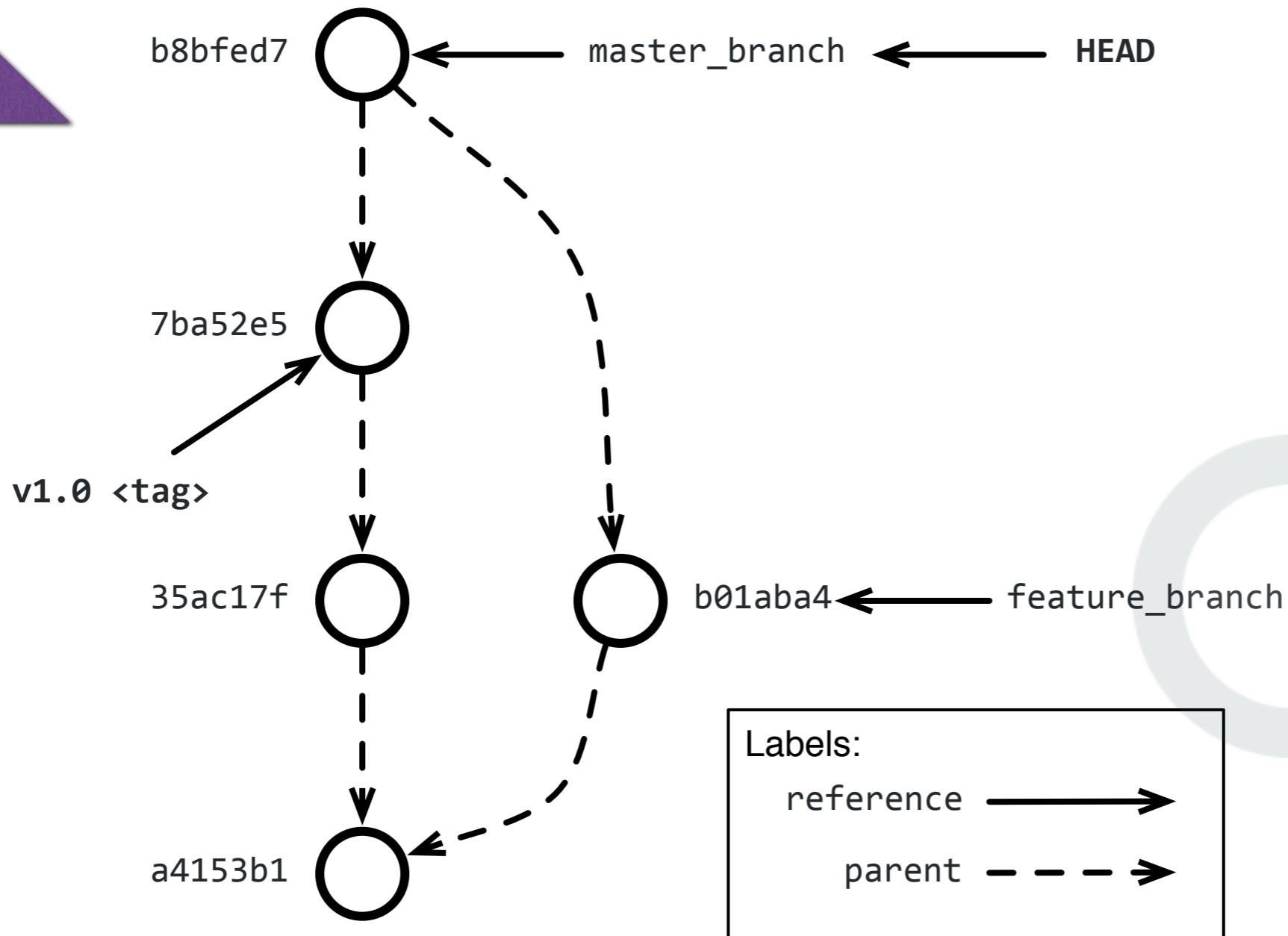
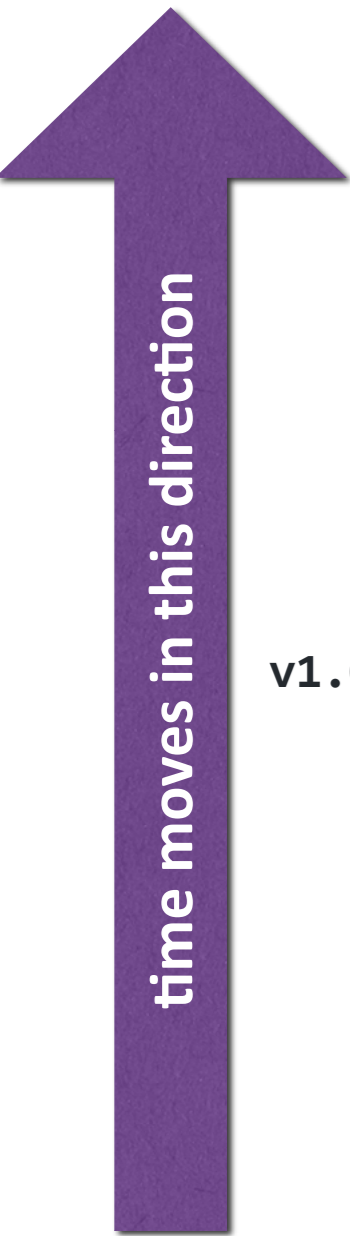
git

1. It's a graph!
2. It's a two-stage database!



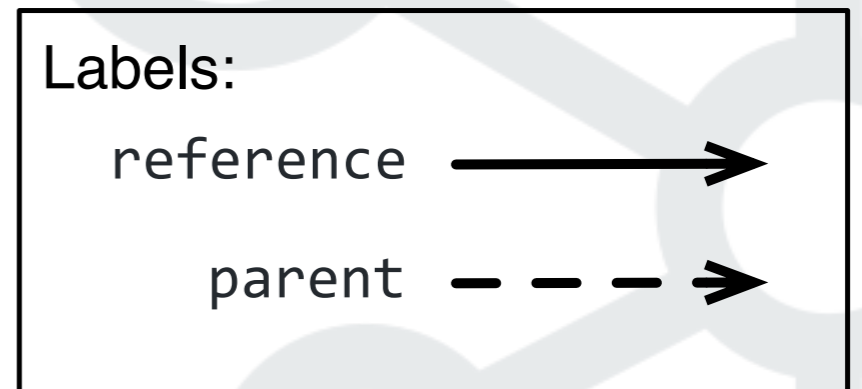
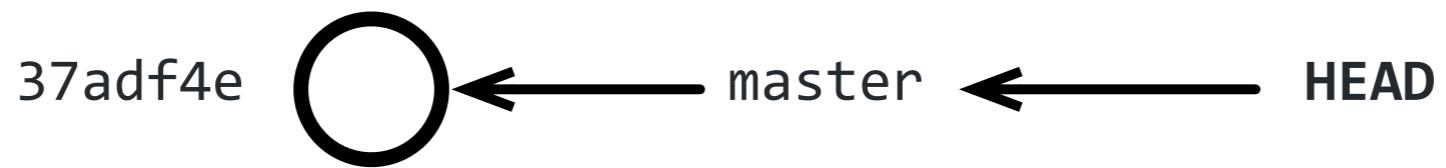
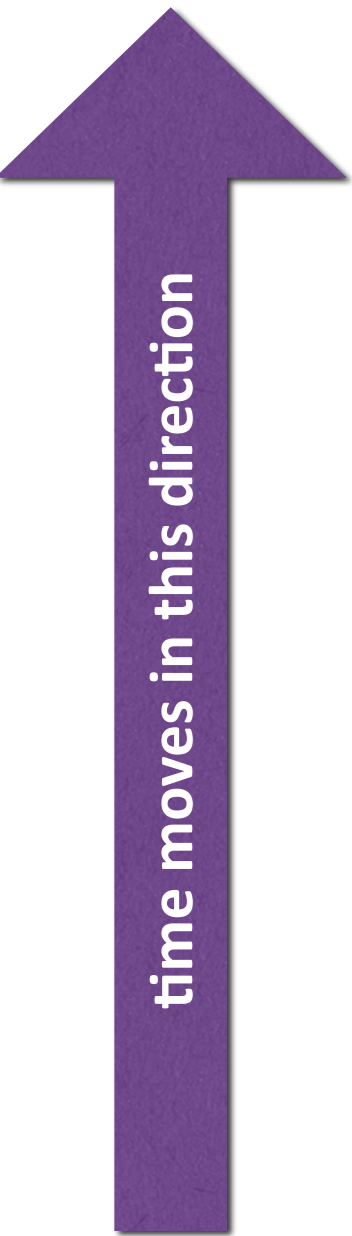
git

as a Graph



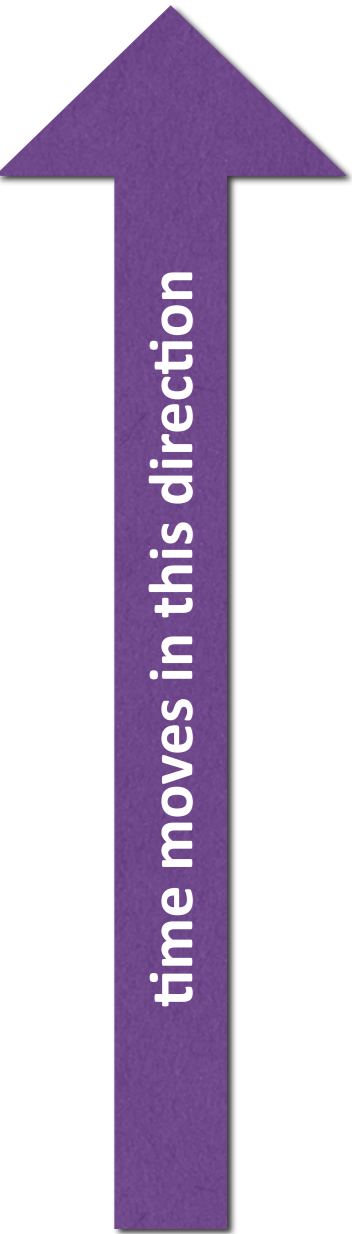


git commit

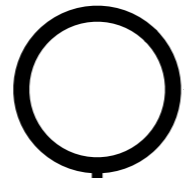




git commit



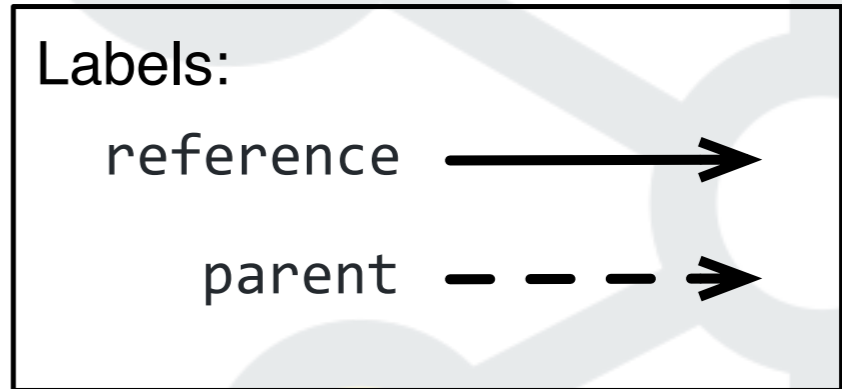
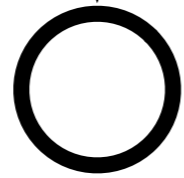
0c0e5ff



← master

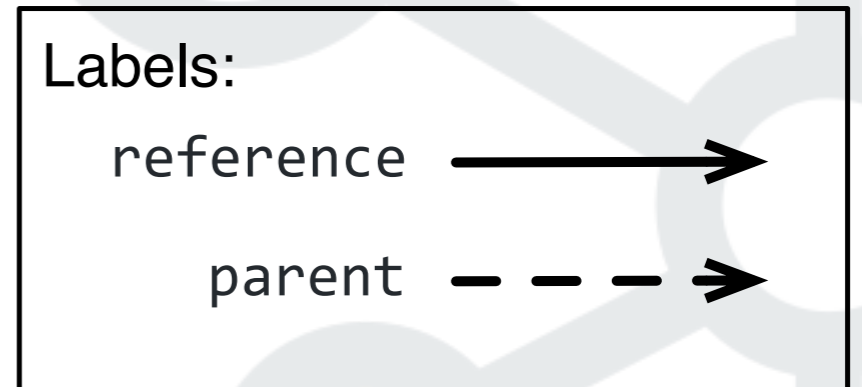
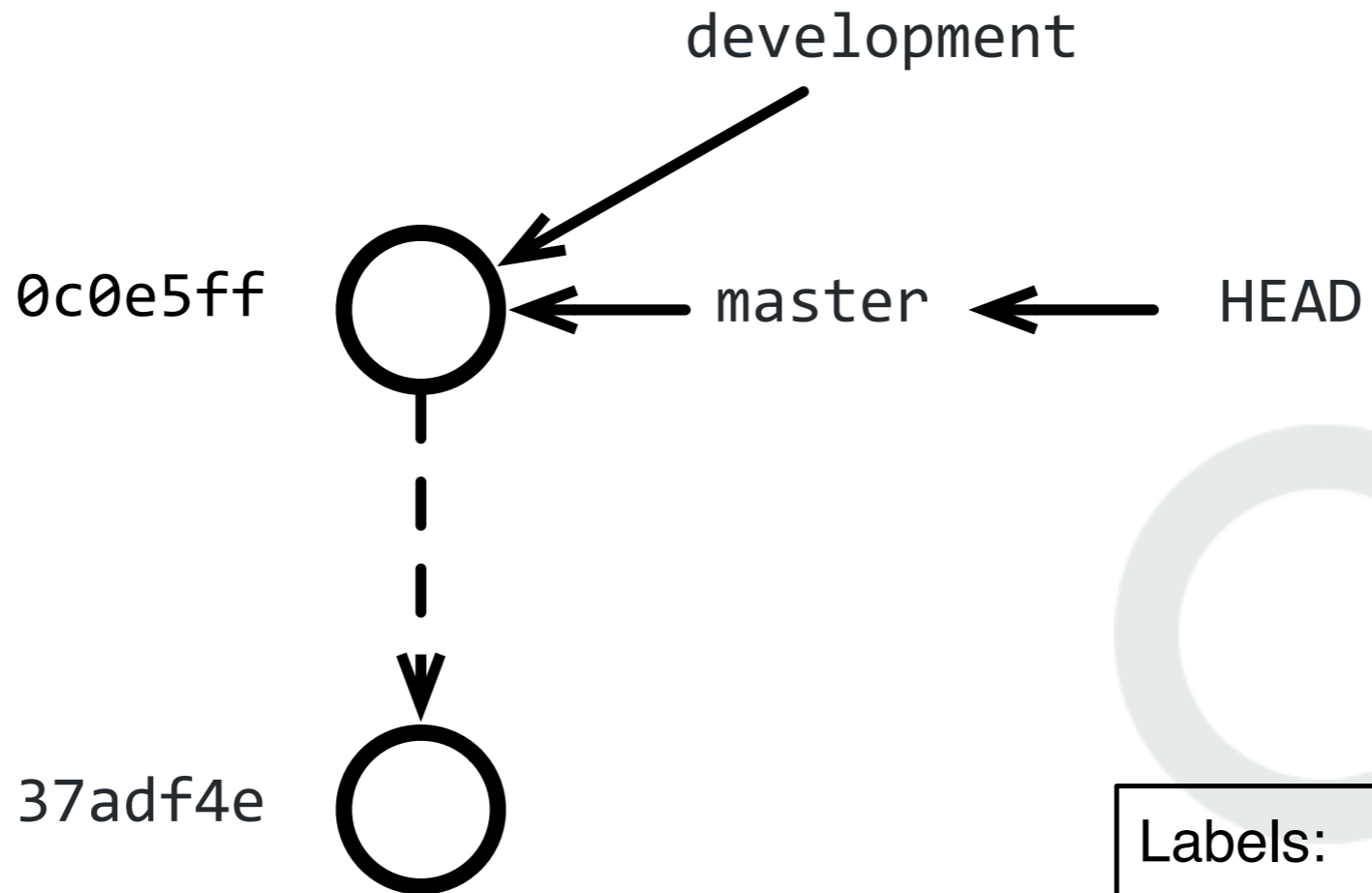
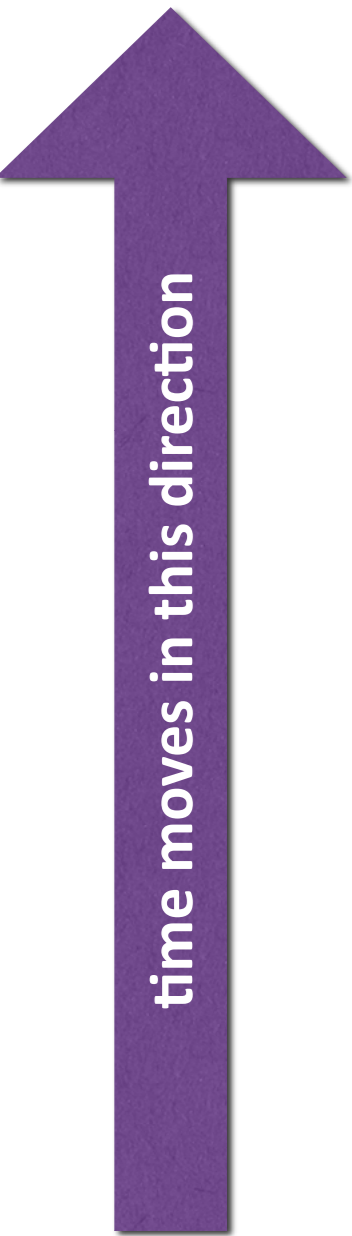
← HEAD

37adf4e



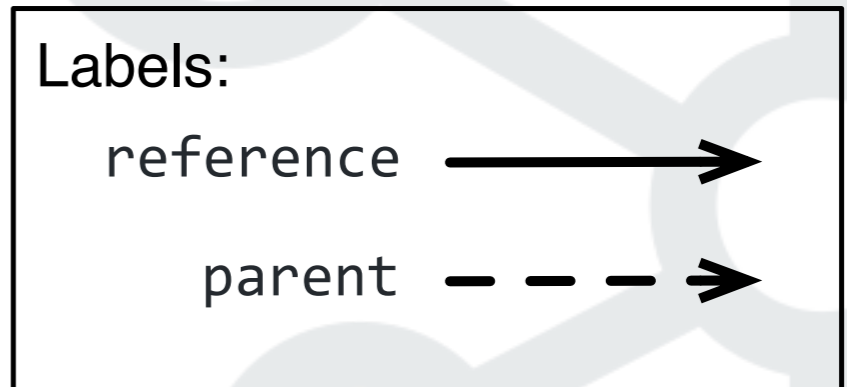
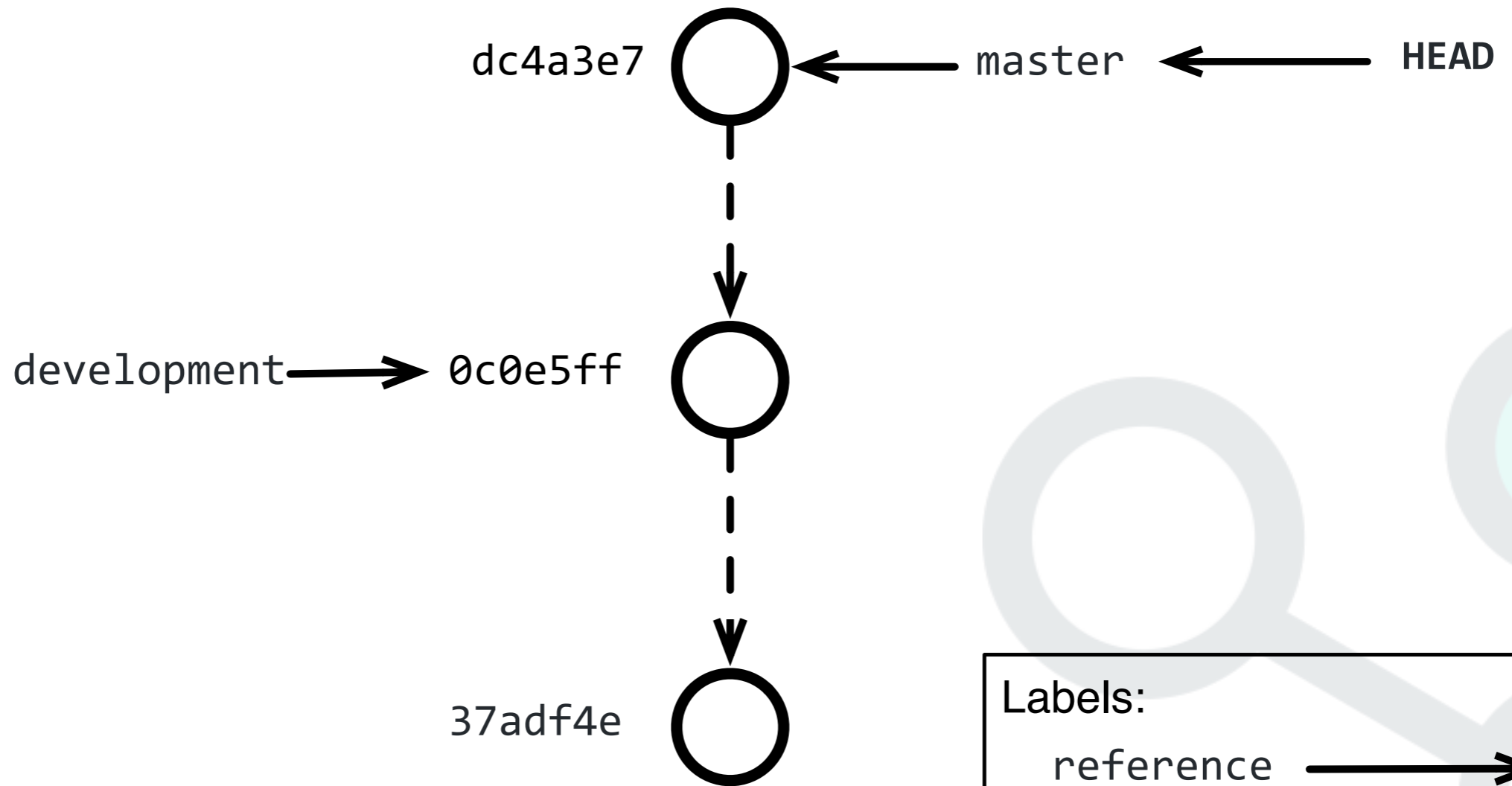
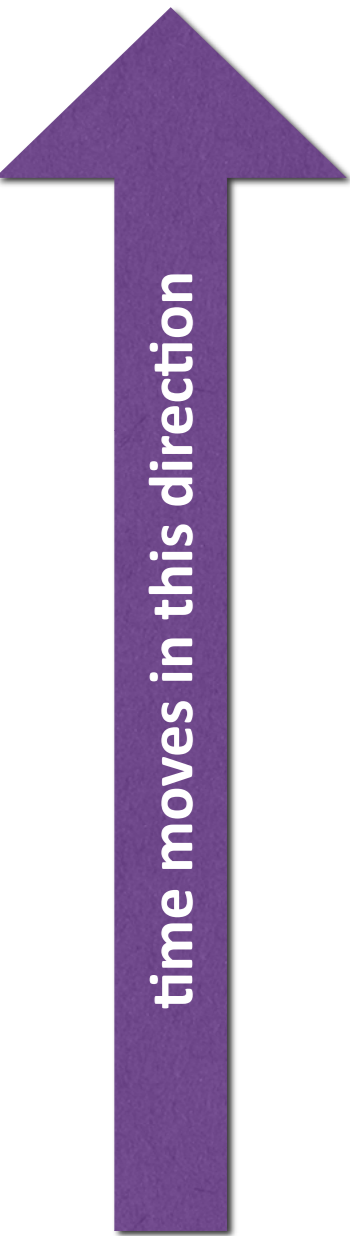


git branch



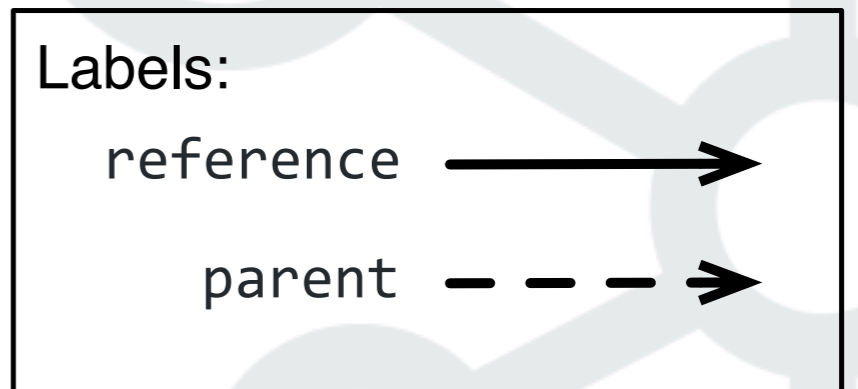
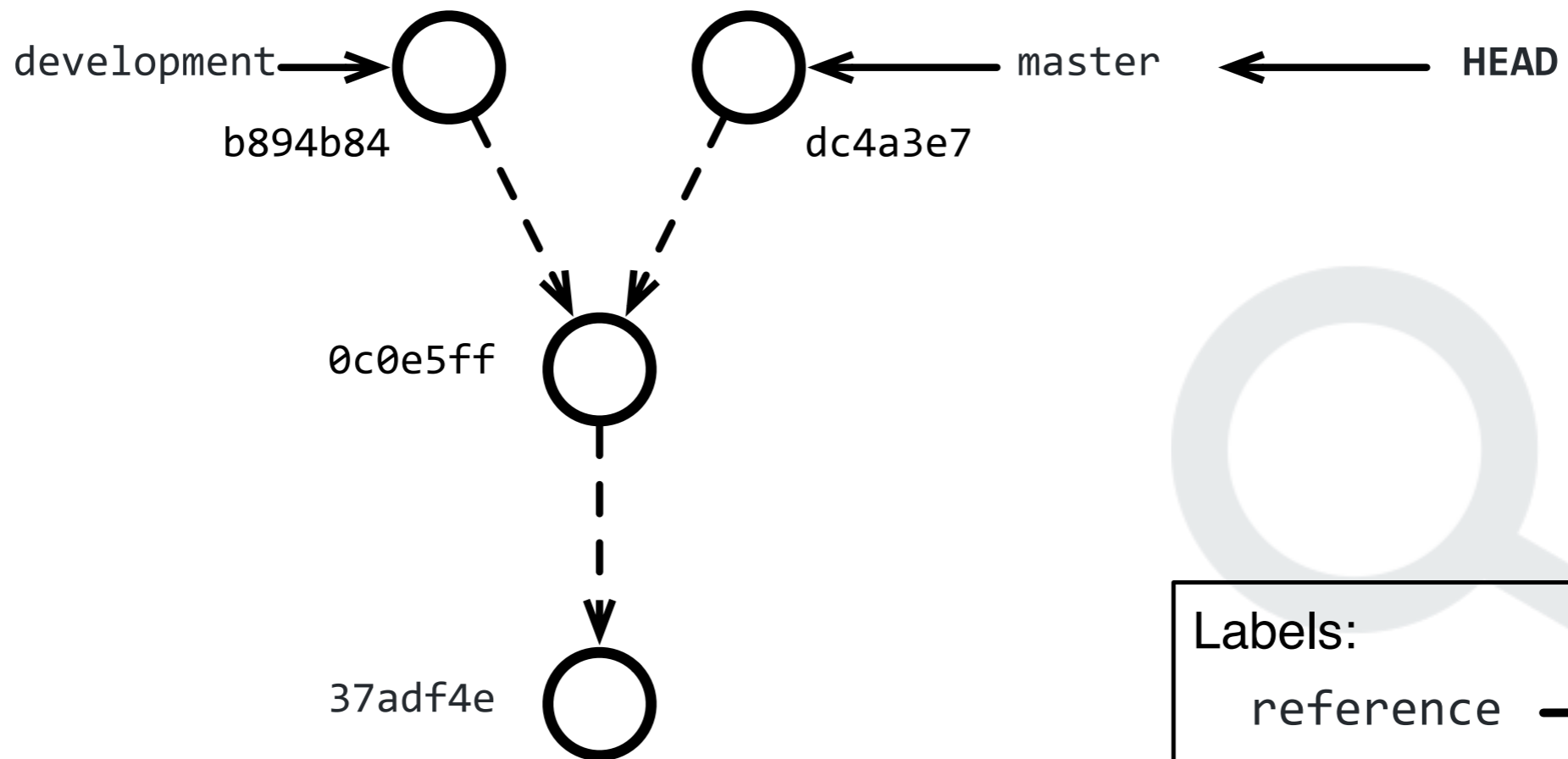
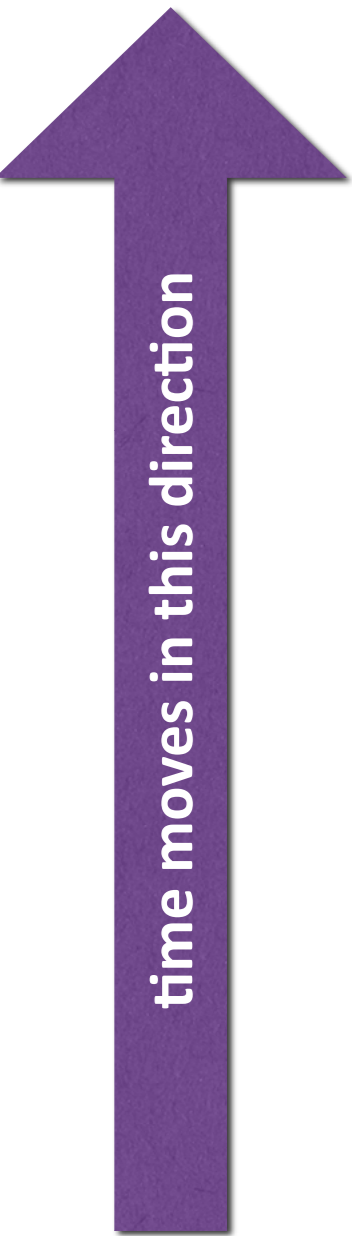


git commit (2)



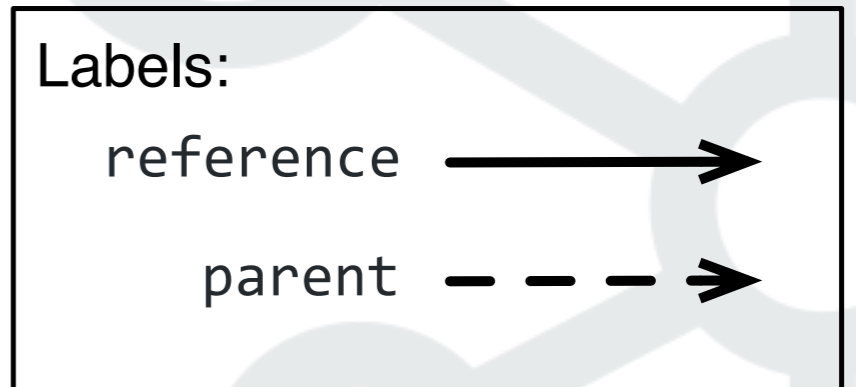
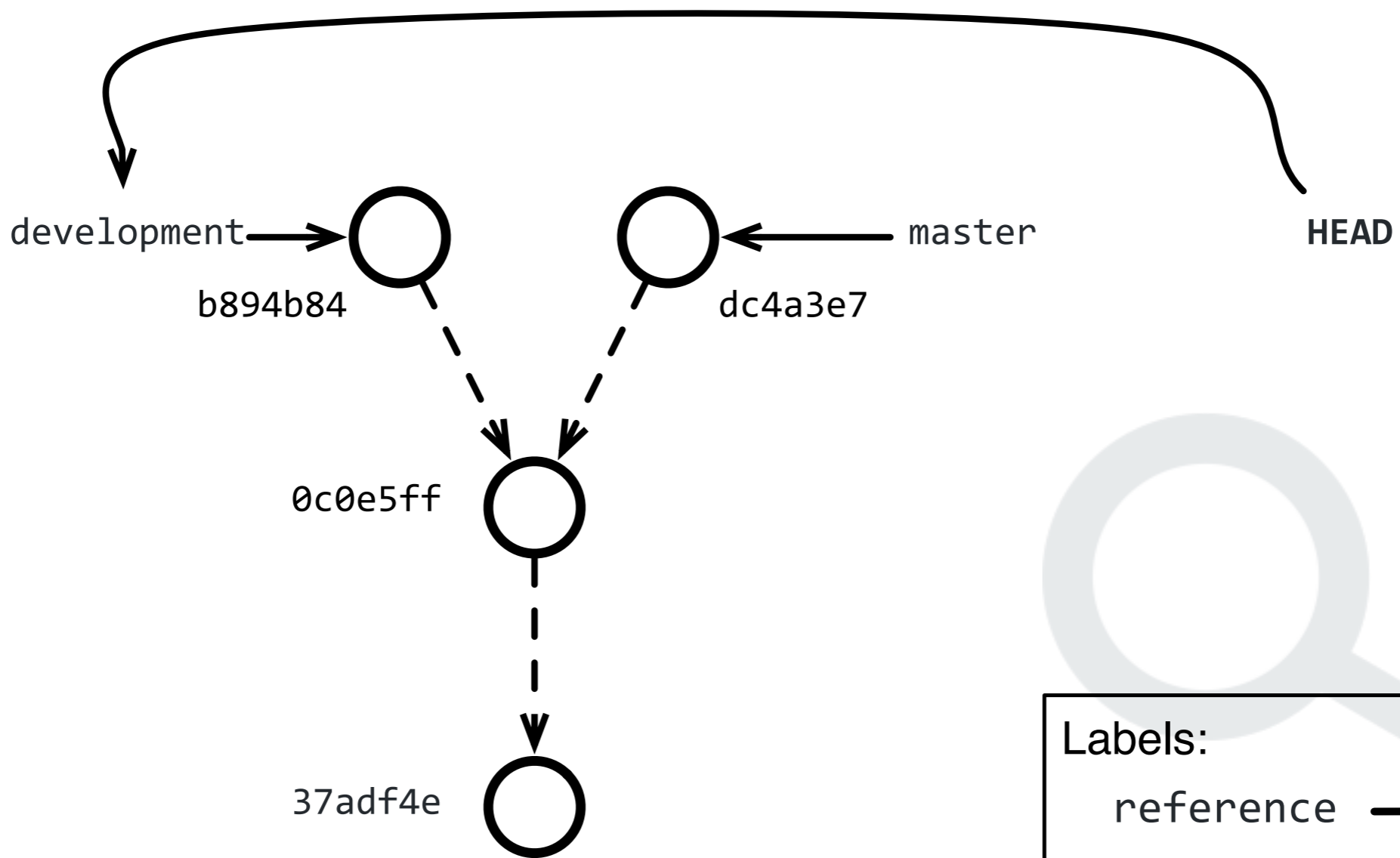
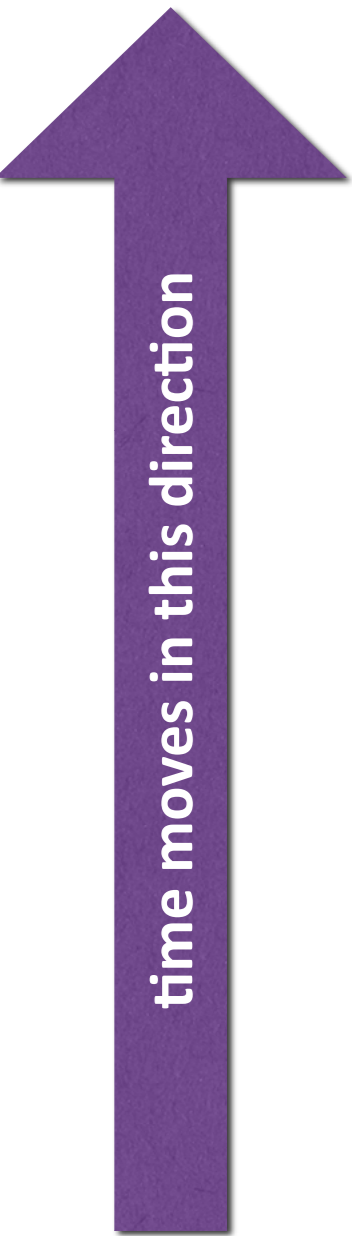


git checkout





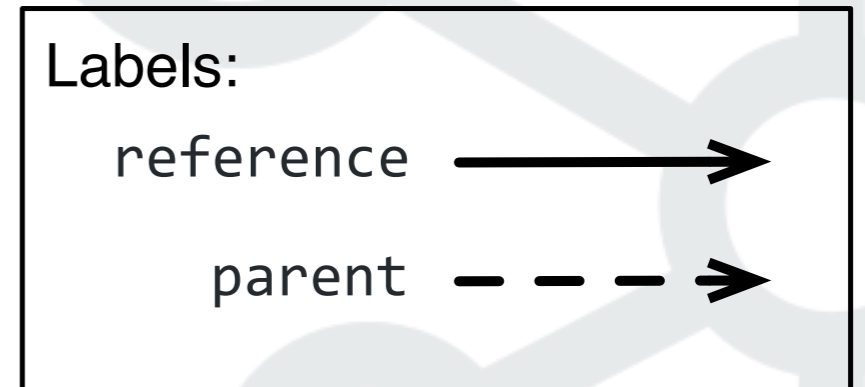
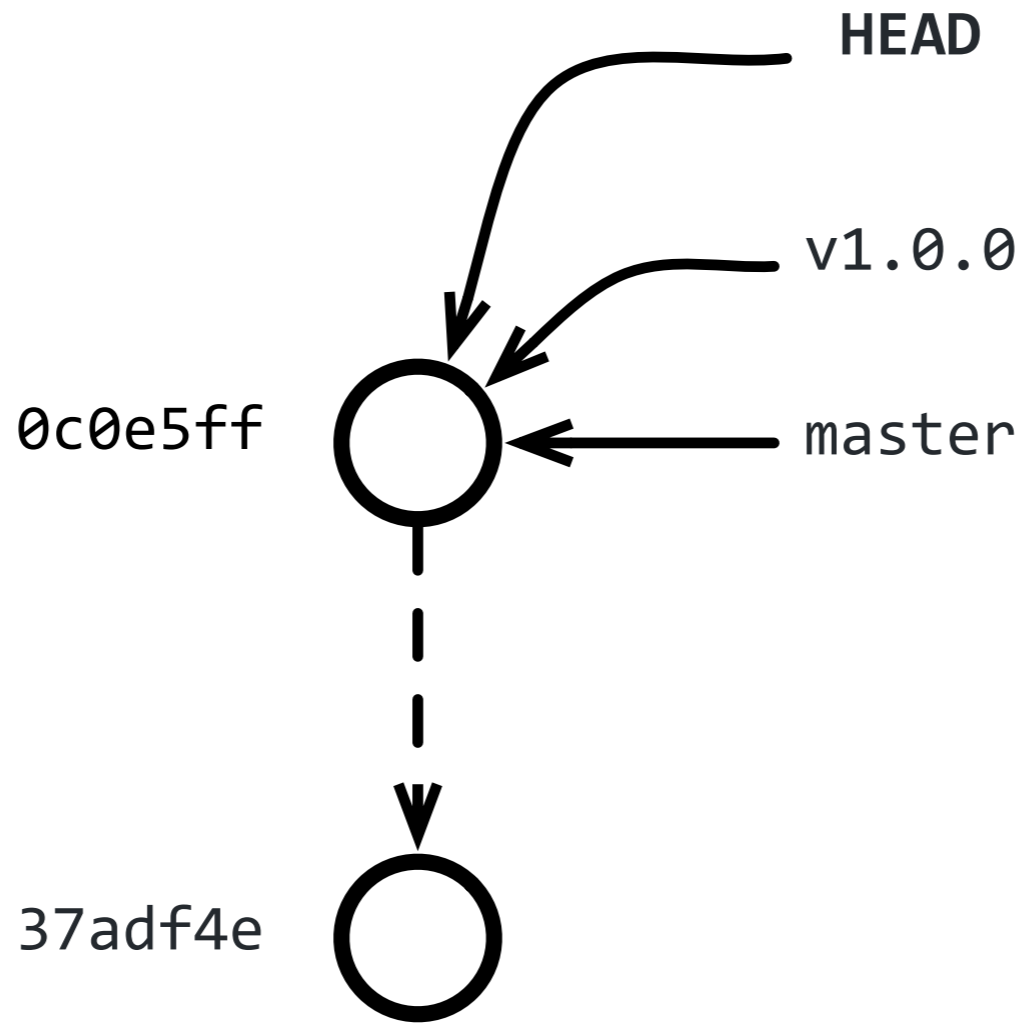
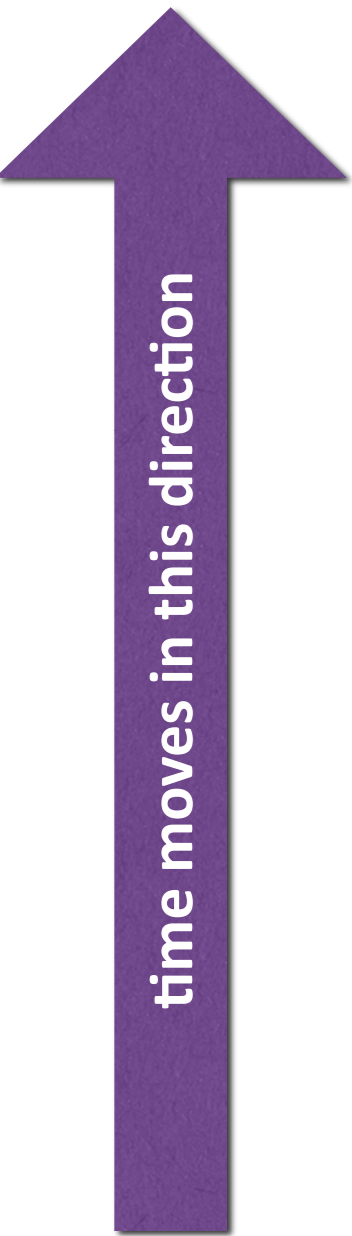
git checkout





git

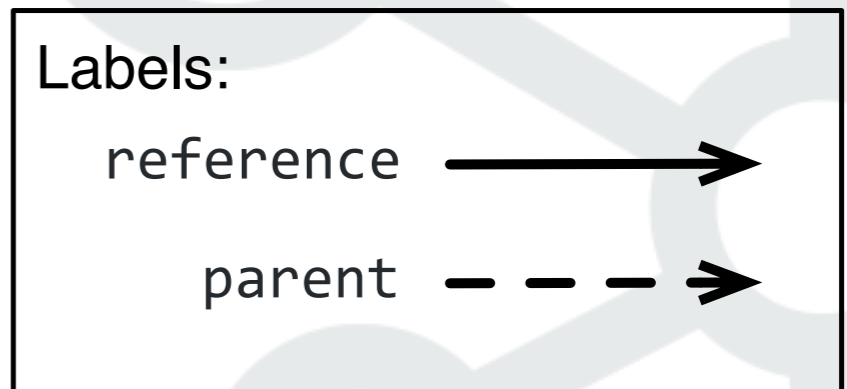
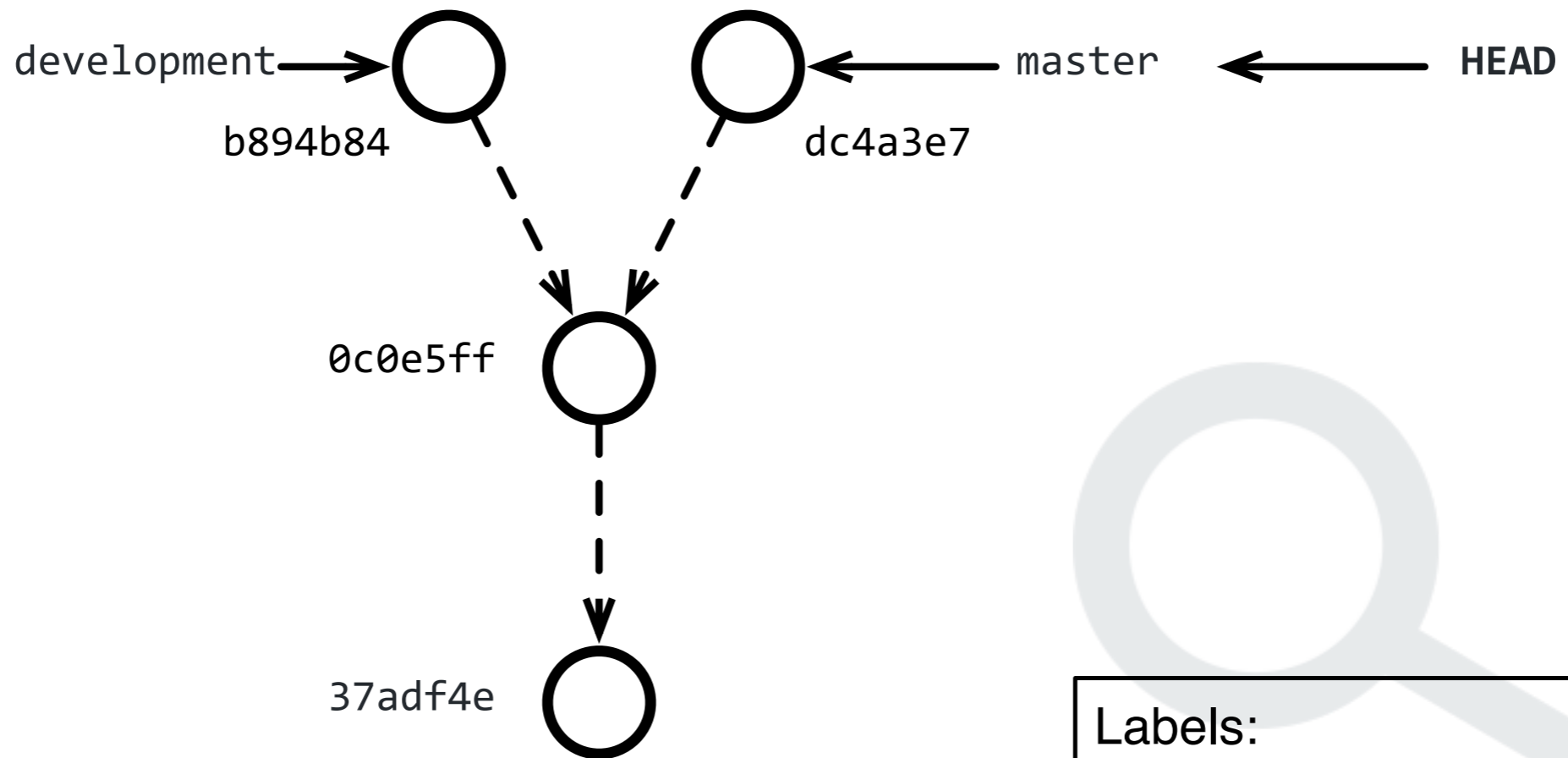
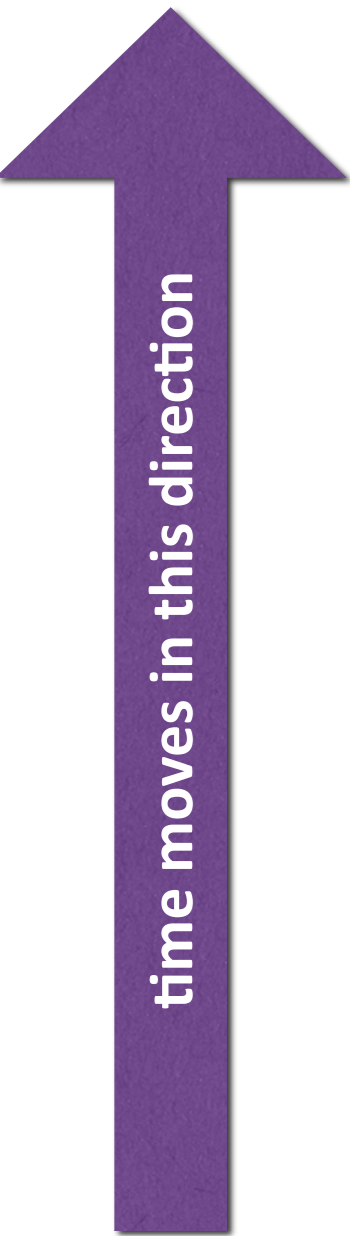
DETACHED HEAD?





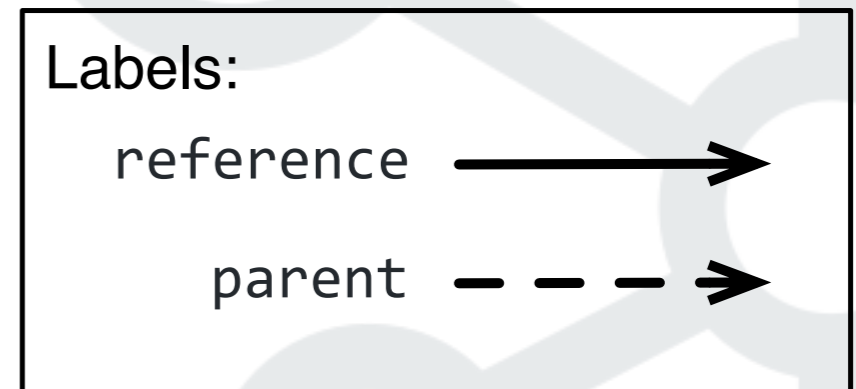
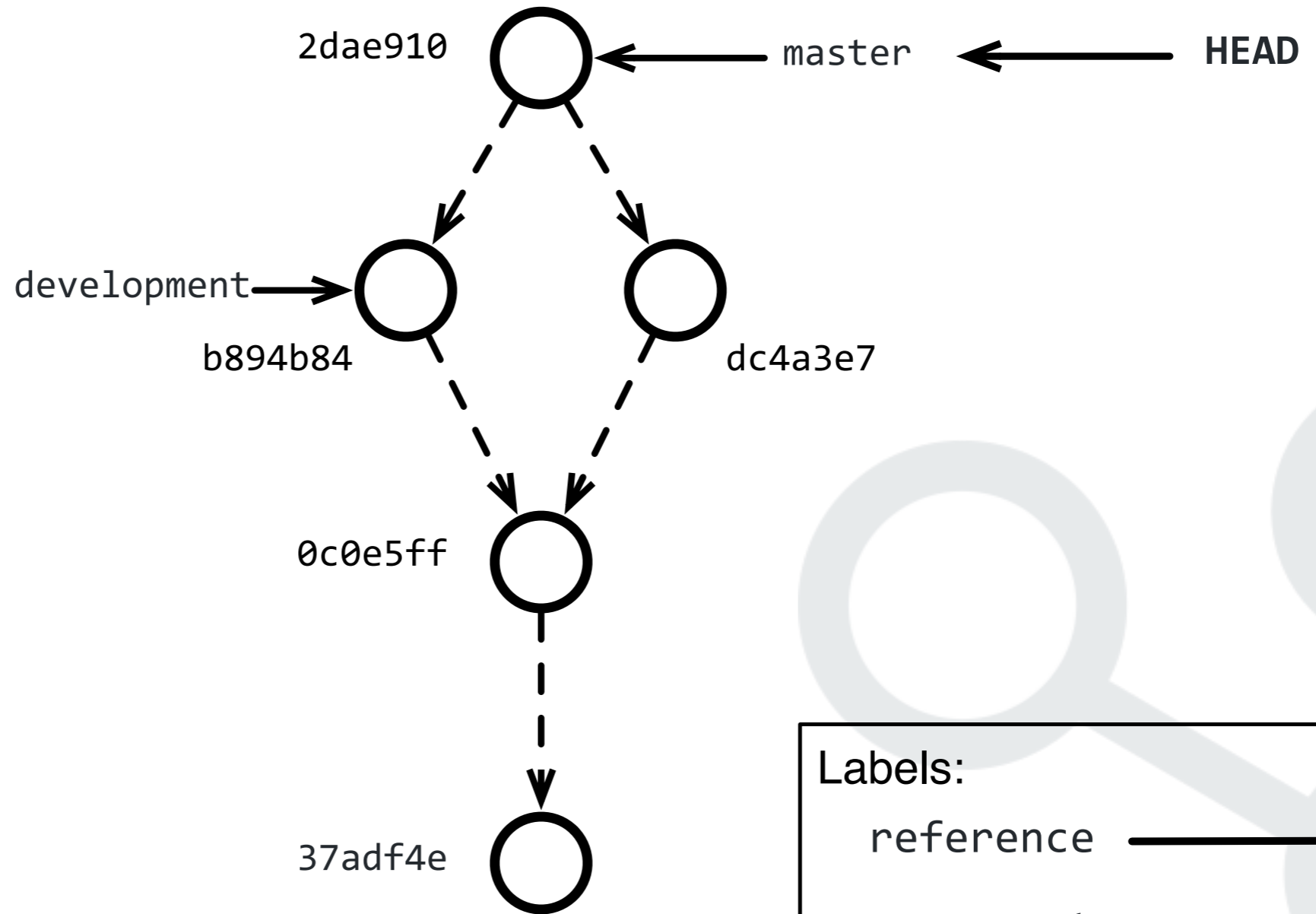
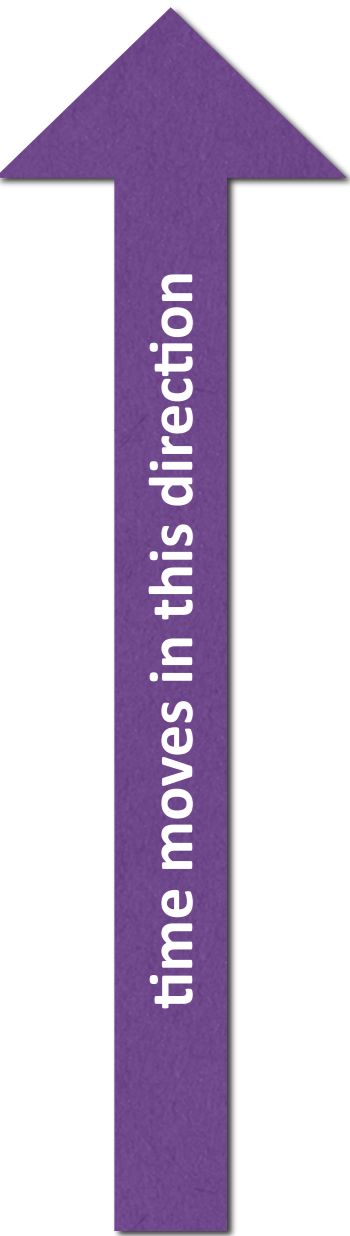
git

merge





git merge

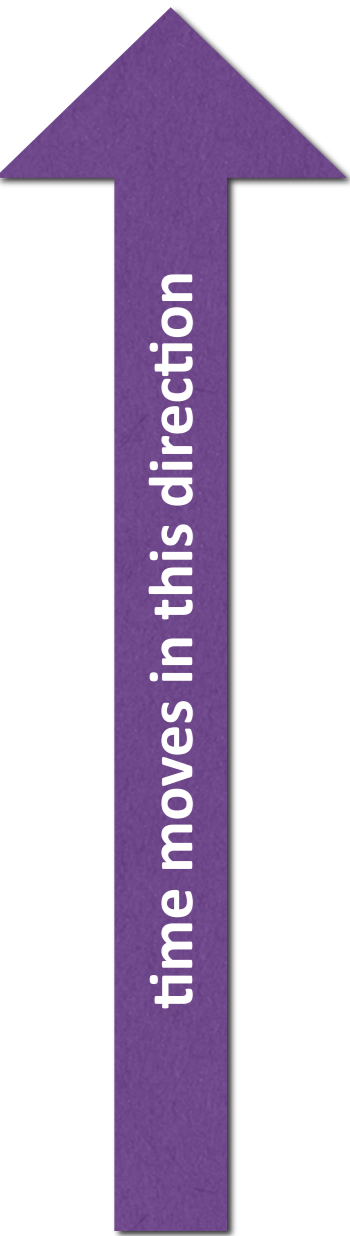




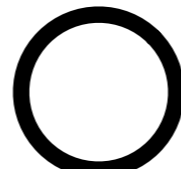
git

merge

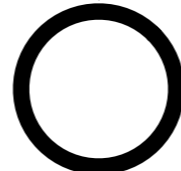
fast-forward



development → dc4a3e7



0c0e5ff



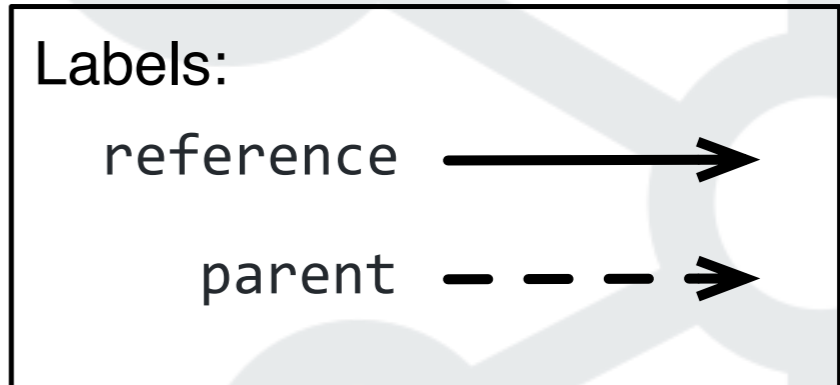
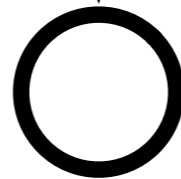
master



HEAD



37adf4e

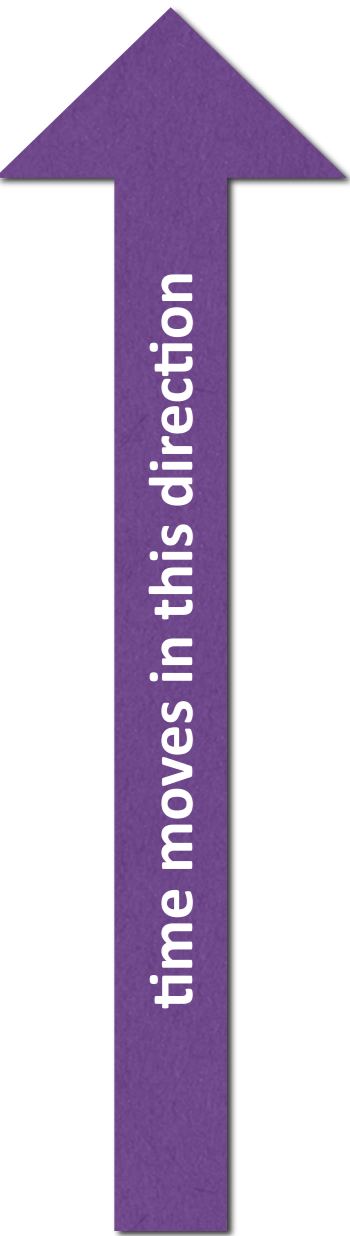




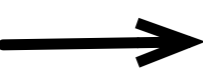
git

merge

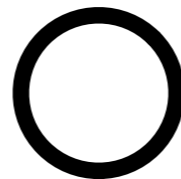
fast-forward



development



dc4a3e7

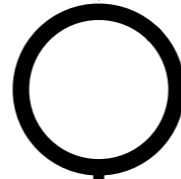


master

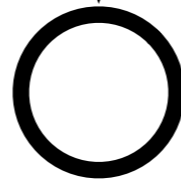


HEAD

0c0e5ff



37adf4e



Labels:

reference



parent





git

as a 2-stage Database

Remote
repositories



Working Copy

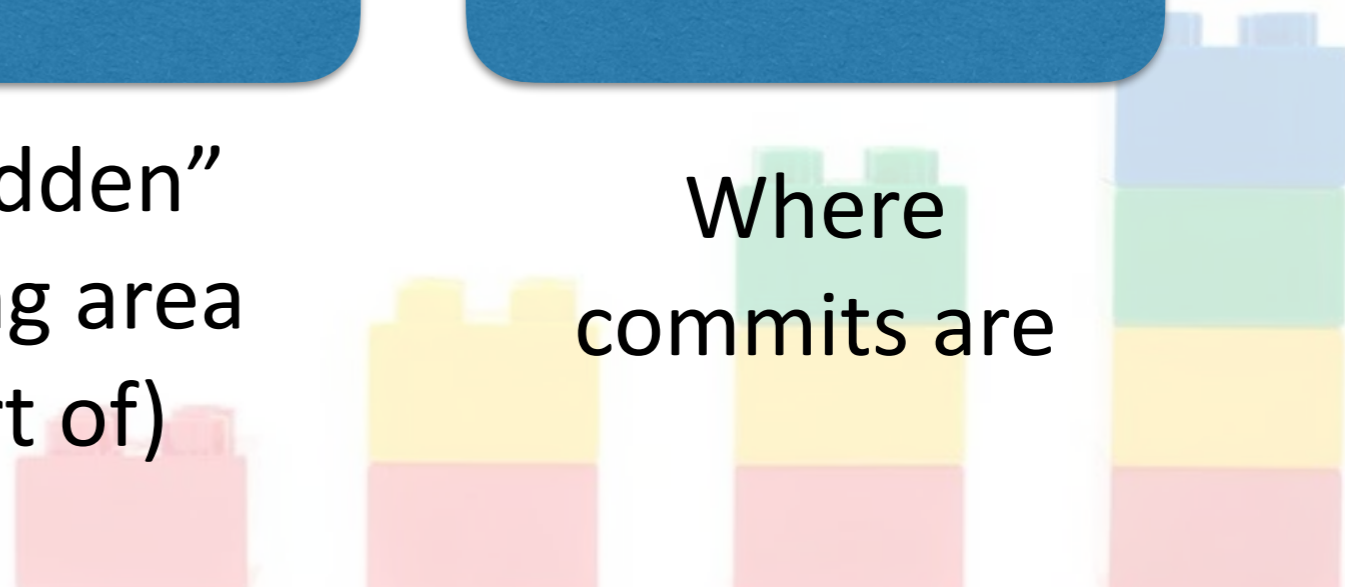
Index

Repository

What you see

A “hidden”
staging area
(sort of)

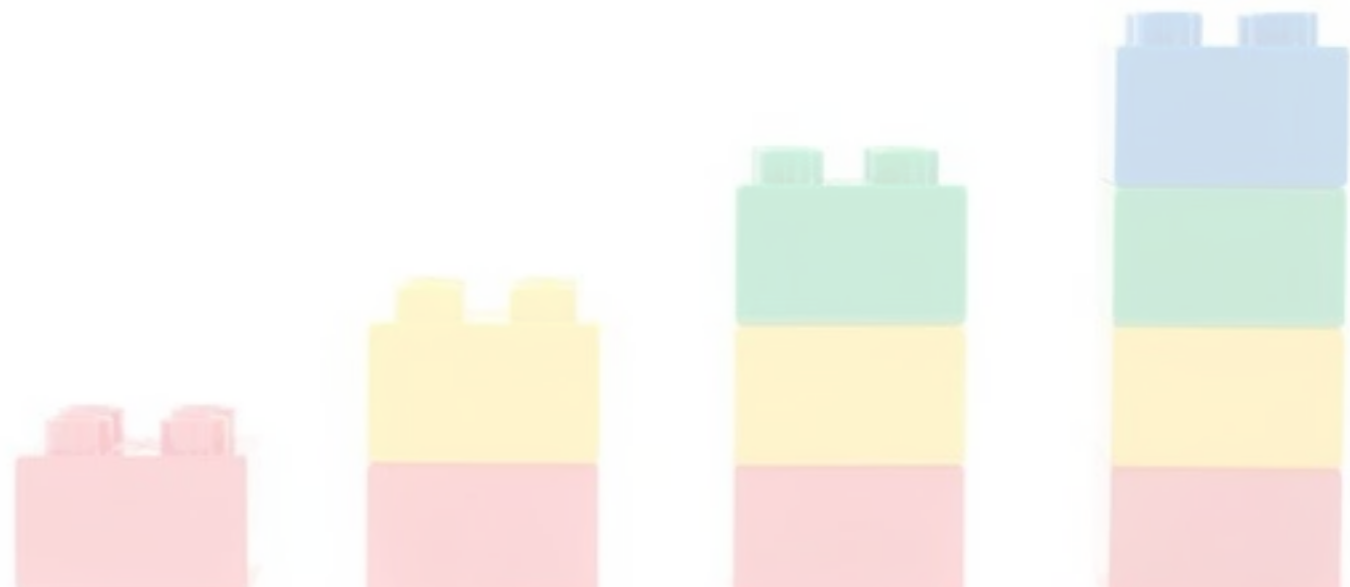
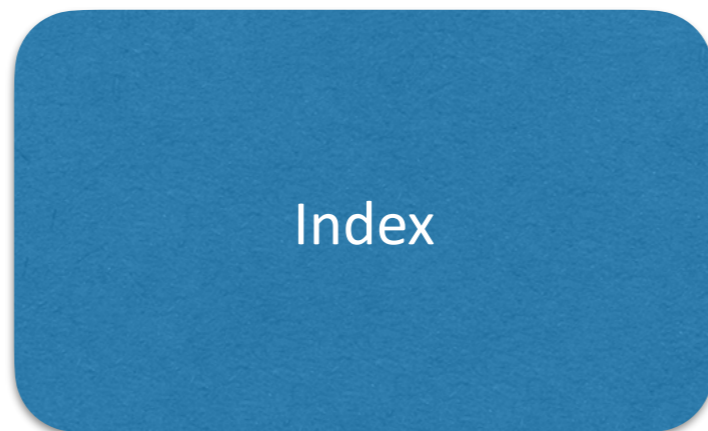
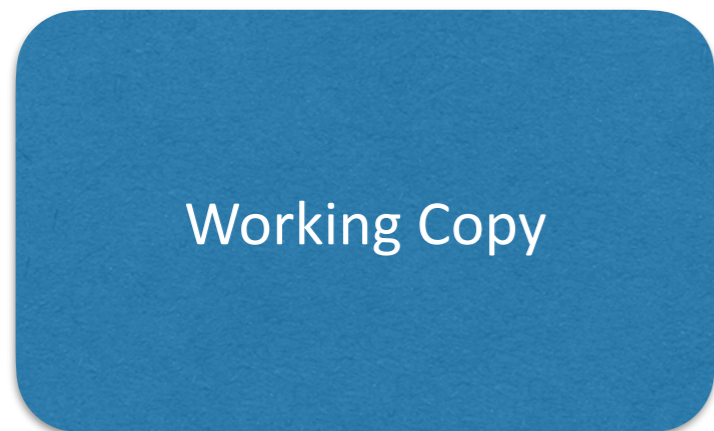
Where
commits are





git

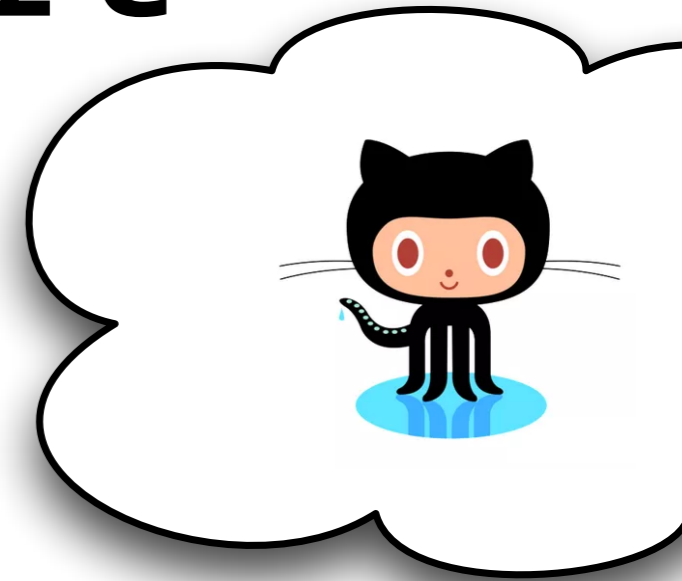
add





git

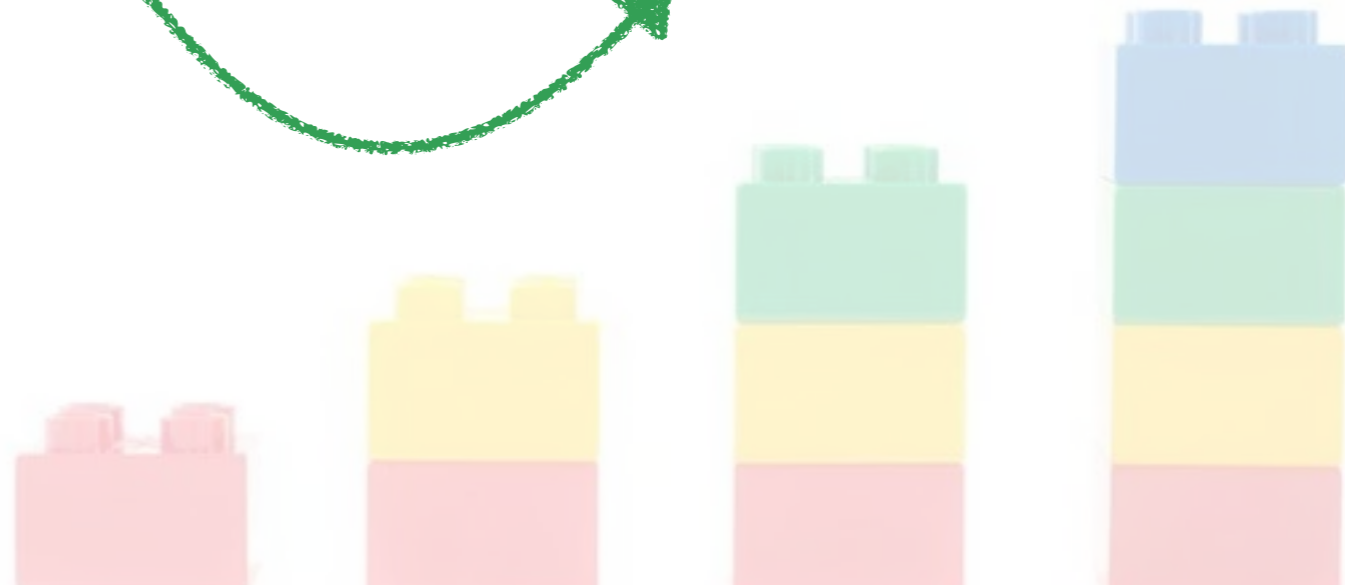
commit



Working Copy

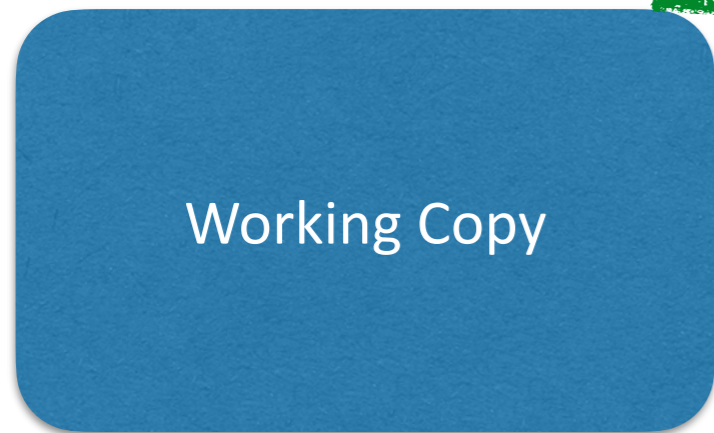
Index

Repository





git checkout





git

fetch



Working Copy

Index

Repository





git

push



Working Copy

Index

Repository





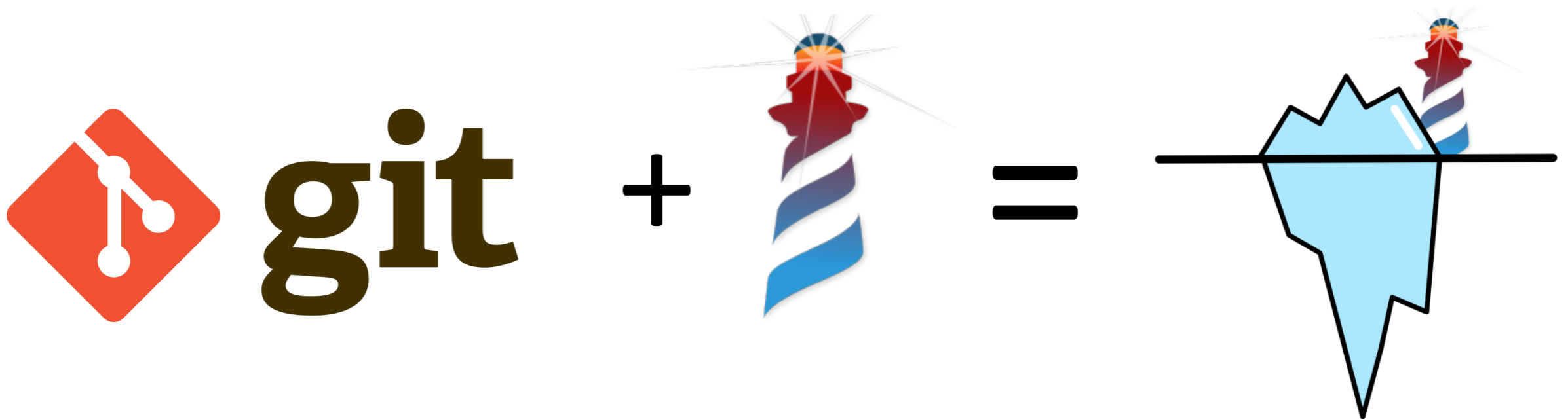
git

pull?

```
$ git fetch  
$ git merge
```


Iceberg

<https://github.com/pharo-vcs/iceberg>





git

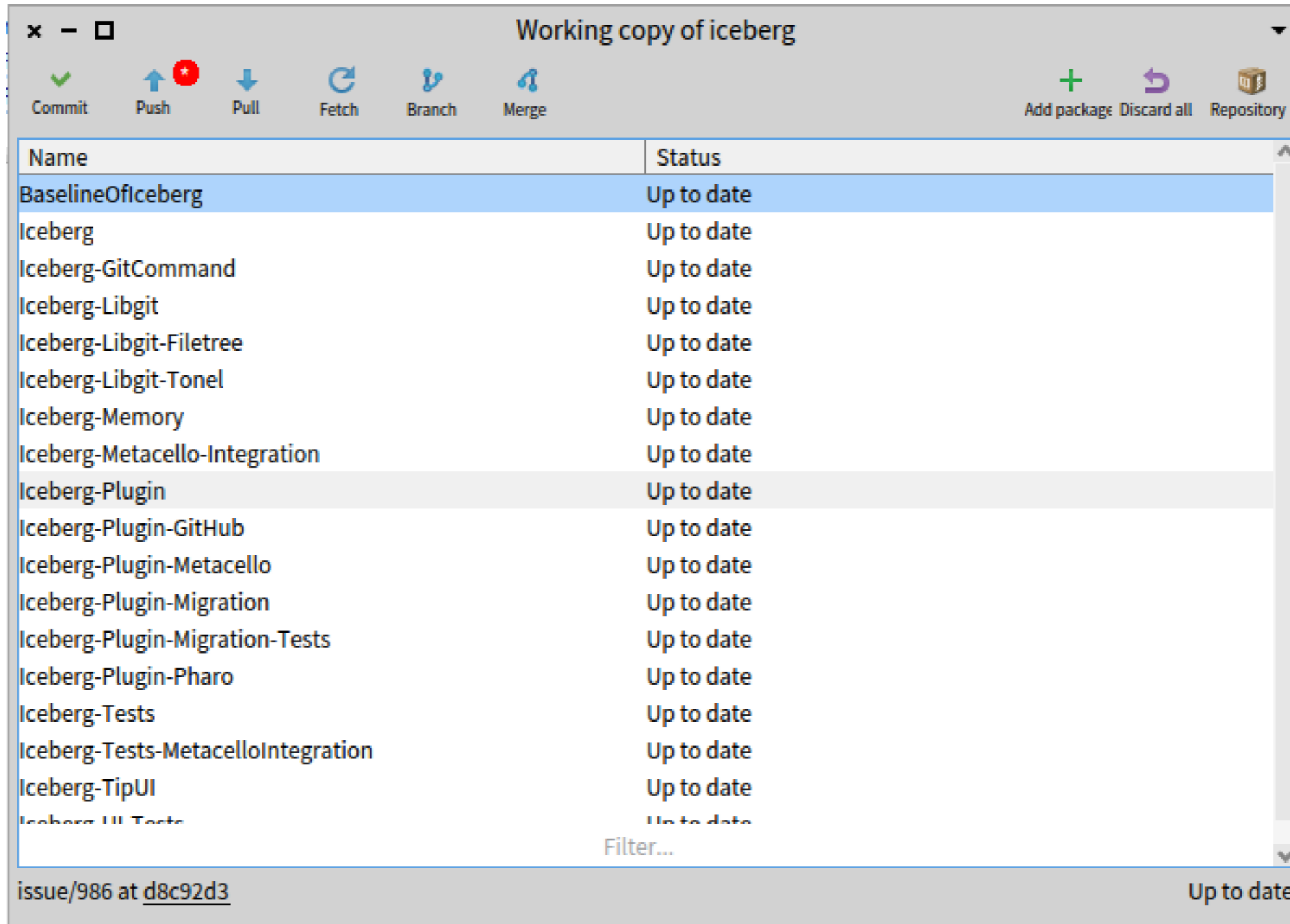
+



Just makes it harder

- Image-based persistence
- Files vs packages
- Monticello adds more false friends!

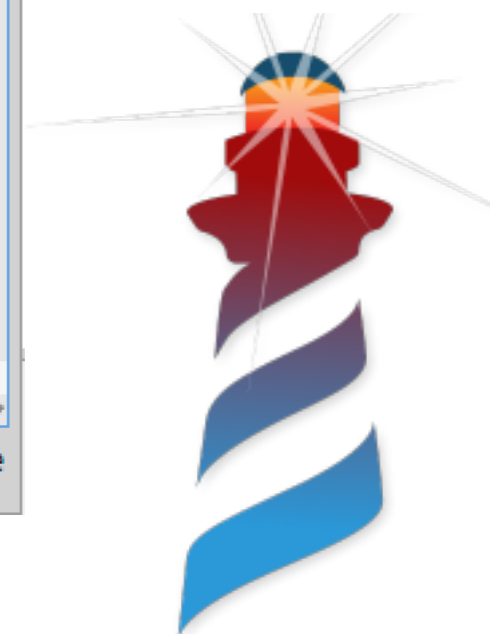
The 4 tips of the Iceberg



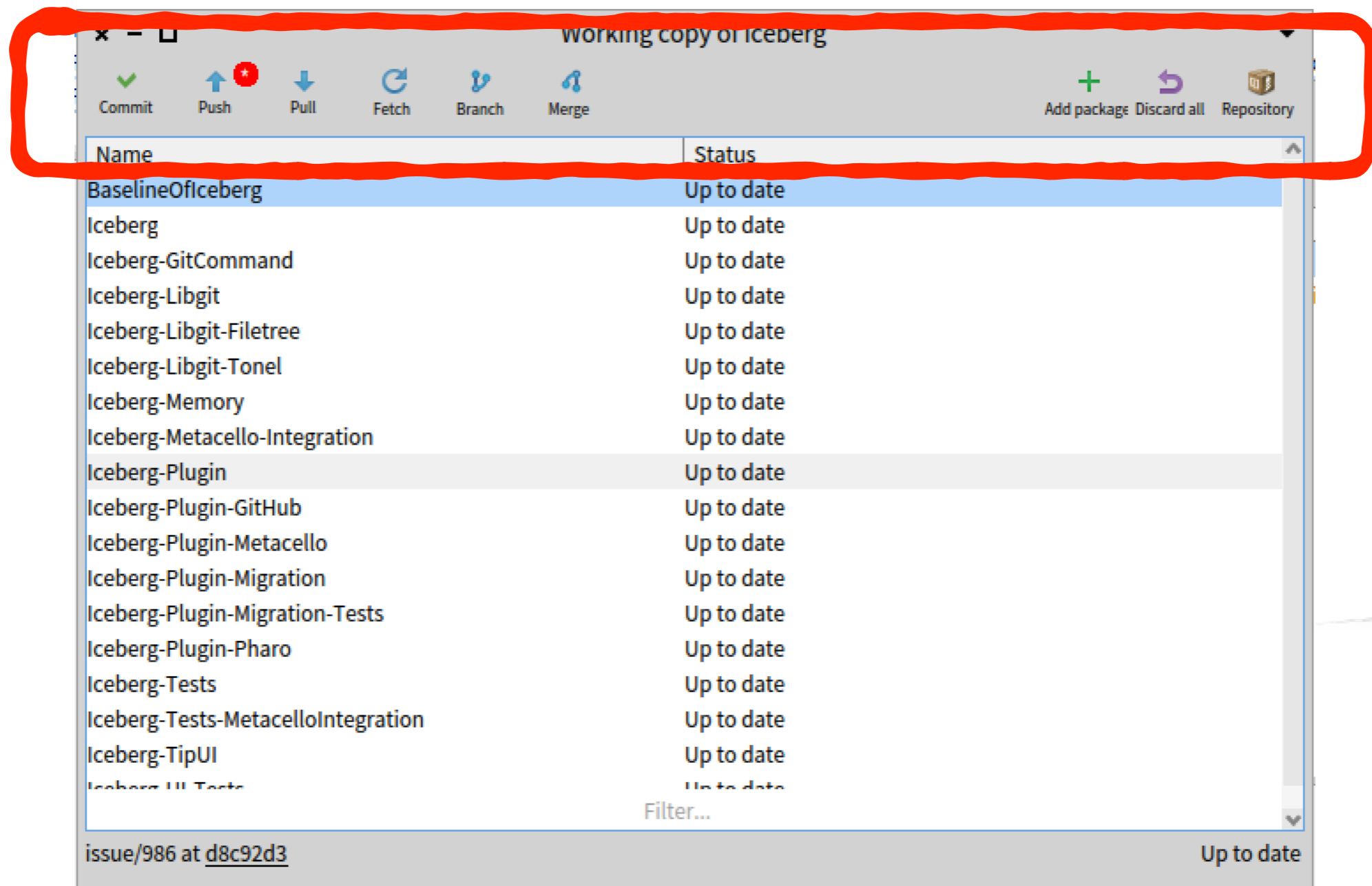
The screenshot shows a window titled "Working copy of iceberg" with a toolbar containing icons for Commit, Push, Pull, Fetch, Branch, Merge, Add package, Discard all, and Repository. Below the toolbar is a table with two columns: "Name" and "Status". The table lists various packages, all of which are marked as "Up to date".

Name	Status
BaselineOfIceberg	Up to date
Iceberg	Up to date
Iceberg-GitCommand	Up to date
Iceberg-Libgit	Up to date
Iceberg-Libgit-Filetree	Up to date
Iceberg-Libgit-Tonel	Up to date
Iceberg-Memory	Up to date
Iceberg-Metacello-Integration	Up to date
Iceberg-Plugin	Up to date
Iceberg-Plugin-GitHub	Up to date
Iceberg-Plugin-Metacello	Up to date
Iceberg-Plugin-Migration	Up to date
Iceberg-Plugin-Migration-Tests	Up to date
Iceberg-Plugin-Pharo	Up to date
Iceberg-Tests	Up to date
Iceberg-Tests-MetacelloIntegration	Up to date
Iceberg-TipUI	Up to date
Iceberg-UI-Tests	Up to date

At the bottom of the window, there is a status bar with the text "issue/986 at [d8c92d3](#)" on the left and "Up to date" on the right.



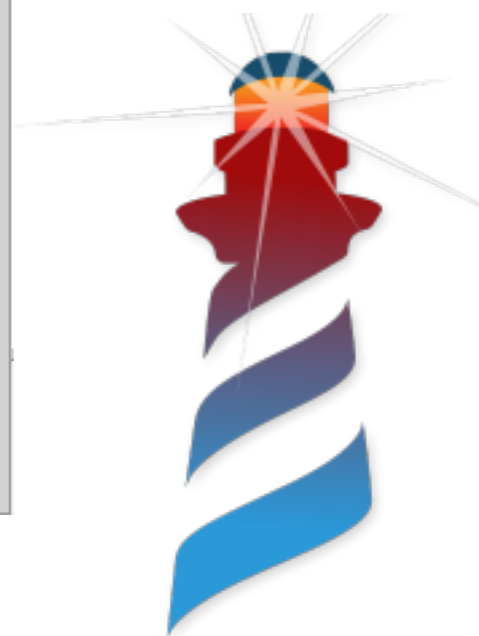
Show the “Good” Path



The screenshot shows the IntelliJ IDEA interface for a 'working copy of iceberg'. The toolbar at the top is highlighted with a red circle and contains the following icons: Commit (green checkmark), Push (blue up arrow with a red asterisk), Pull (blue down arrow), Fetch (blue circular arrow), Branch (blue fork icon), Merge (blue merge icon), Add package (green plus), Discard all (purple undo), and Repository (gold cube icon).

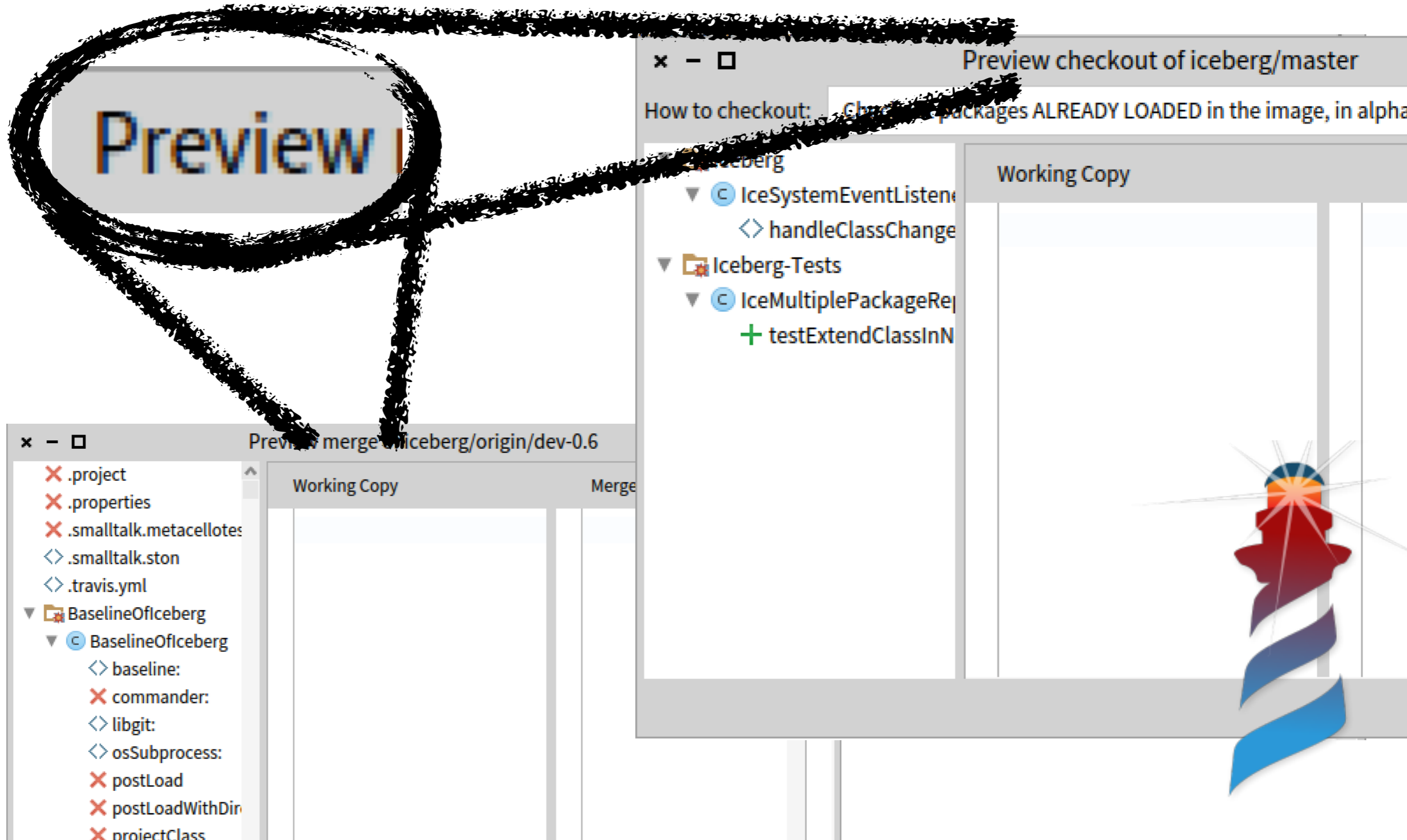
Name	Status
BaselineOfIceberg	Up to date
Iceberg	Up to date
Iceberg-GitCommand	Up to date
Iceberg-Libgit	Up to date
Iceberg-Libgit-Filetree	Up to date
Iceberg-Libgit-Tonel	Up to date
Iceberg-Memory	Up to date
Iceberg-Metacello-Integration	Up to date
Iceberg-Plugin	Up to date
Iceberg-Plugin-GitHub	Up to date
Iceberg-Plugin-Metacello	Up to date
Iceberg-Plugin-Migration	Up to date
Iceberg-Plugin-Migration-Tests	Up to date
Iceberg-Plugin-Pharo	Up to date
Iceberg-Tests	Up to date
Iceberg-Tests-MetacelloIntegration	Up to date
Iceberg-TipUI	Up to date
Iceberg-UI-Tests	Up to date

At the bottom left, the text 'issue/986 at d8c92d3' is visible. At the bottom right, the text 'Up to date' is visible. A 'Filter...' input field is located at the bottom center of the list.

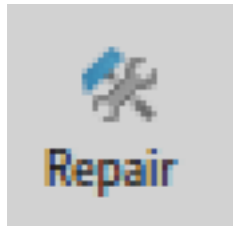


Warn about Destruction

Preview






Guide out of the Darkness



Repair repository

This repository was created from commit f564539 but the commit is not in your disk repository. You may fetch the correct commit from a remote repository or discard the code in your image and load what is in the repository.

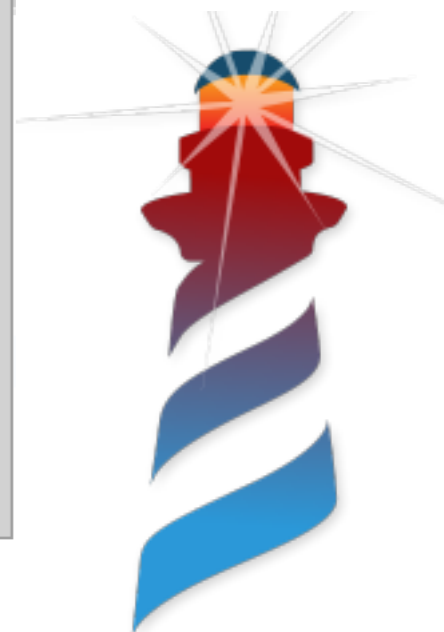
-  Fetch remote commits to find the reference commit
-  Discard local changes and checkout an existing branch
-  Discard image changes and load repository version

Discard all changes in your image and checkout an existing branch.
This action will checkout the branch in the repository and align your image with your repository.

This situation happens because the code loaded in your image does not correspond with the status of your repository.

Opens a preview window before doing any change.

This operation will modify the state of your working copy in disk. All non-committed changes in your disk working copy that are not in the image will be lost. If you want to keep them, perform a commit from outside before.



Use a Clear Language

`workingCopy commitWithMessage: 'message'.`

`aRepository head.`

`aBranch pull.`



Iceberg's Working Copy

Pharo Image

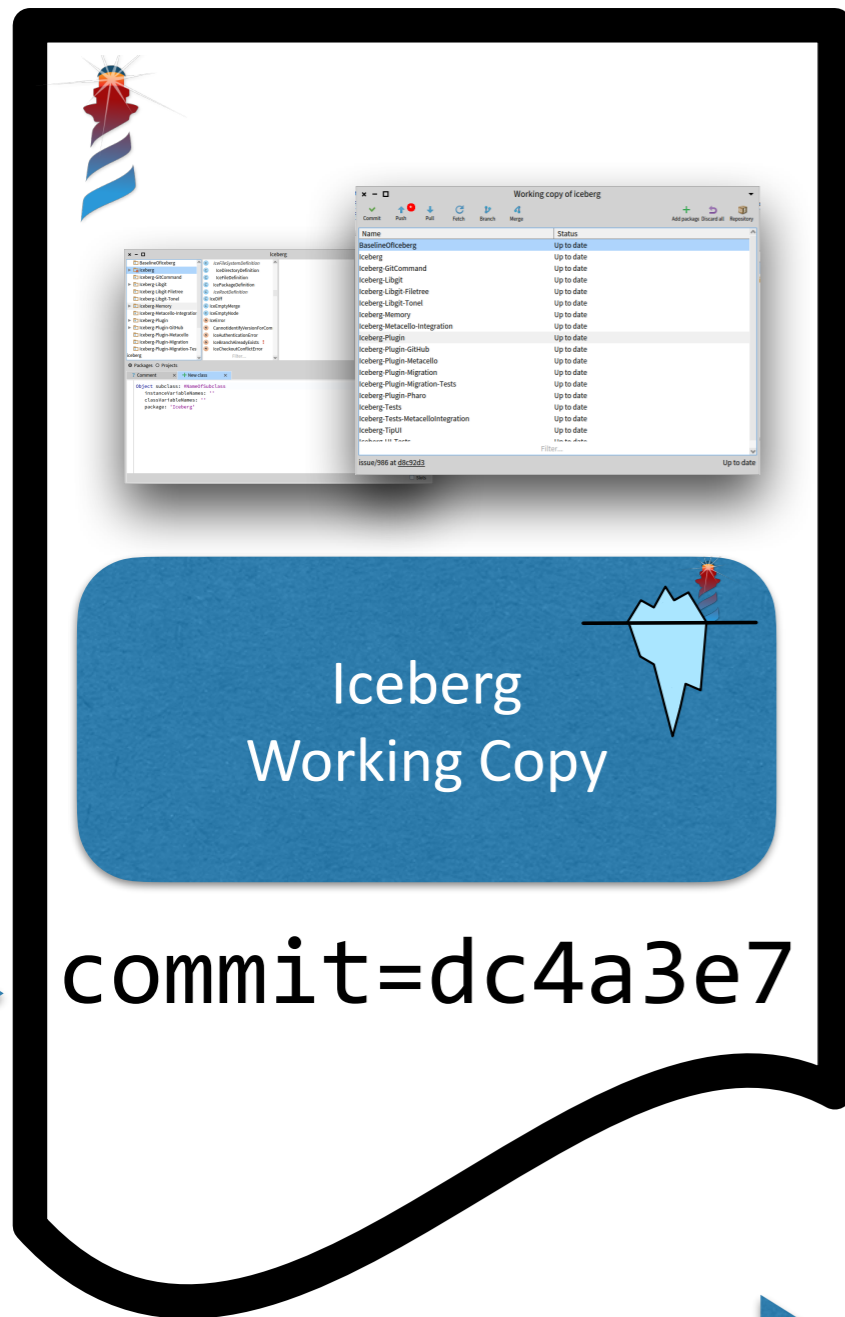
Iceberg Working Copy

commit=dc4a3e7

- A repository has an **in-image working copy**
- A working copy remembers its **current commit**
- Required for pulling, pushing, diffing...

Dual Working Copy

Pharo Image



To be able to work, the working copy should be **in-sync** with the working copy in disk

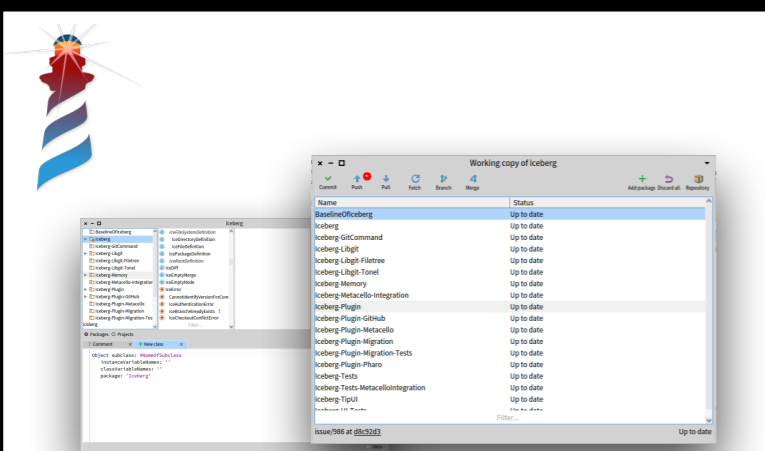


...



Detached Working Copy

Pharo Image



The screenshot shows a Pharo IDE window titled "Working copy of iceberg". The interface includes a package browser on the left, a central workspace, and a console at the bottom. A table in the workspace lists various packages and their status:

Name	Status
BaselineOfIceberg	Up to date
Iceberg	Up to date
Iceberg-GitCommand	Up to date
Iceberg-Git	Up to date
Iceberg-Light	Up to date
Iceberg-Light-Filetree	Up to date
Iceberg-Light-Tonnel	Up to date
Iceberg-Memory	Up to date
Iceberg-MetacelloIntegration	Up to date
Iceberg-Plugin	Up to date
Iceberg-Plugin-GitHub	Up to date
Iceberg-Plugin-Metacello	Up to date
Iceberg-Plugin-Migration	Up to date
Iceberg-Plugin-Migration-Tests	Up to date
Iceberg-Plugin-Pharo	Up to date
Iceberg-Tests	Up to date
Iceberg-Tests-MetacelloIntegration	Up to date
Iceberg-TipUI	Up to date

Below the screenshot is a blue rounded rectangle with the text "Iceberg Working Copy" and an iceberg icon. A red arrow points from the left towards this box.

commit=**dc4a3e7**

The working copy in the image can get **desync** from the working copy in disk



git
Working Copy

...



git
Repository



commit=**0c0c5ff**



Why do I get Detached Working Copy?

Although quite rare:

- Possibility 1) You touched your git repository from outside the image
- Possibility 2) Your image crashed in between commits
- Possibility 3) You forgot to save your image after a commit

How do I get out of Detached Working Copy?

- Option 1) Move your image to your repository state (load what is in the disk)
- Option 2) Move your repository state to what is in the image (checkout image's commit)
- Option 3) Try merging both

Takeovers

- Git is the assembly of version control
- But you need to know the internals
 - commit graph and references
 - working copy, staging area
- Iceberg deals with image-based persistency