

מודלים לפיתוח מערכות תוכנה

Software Systems Modeling

קורס 12003

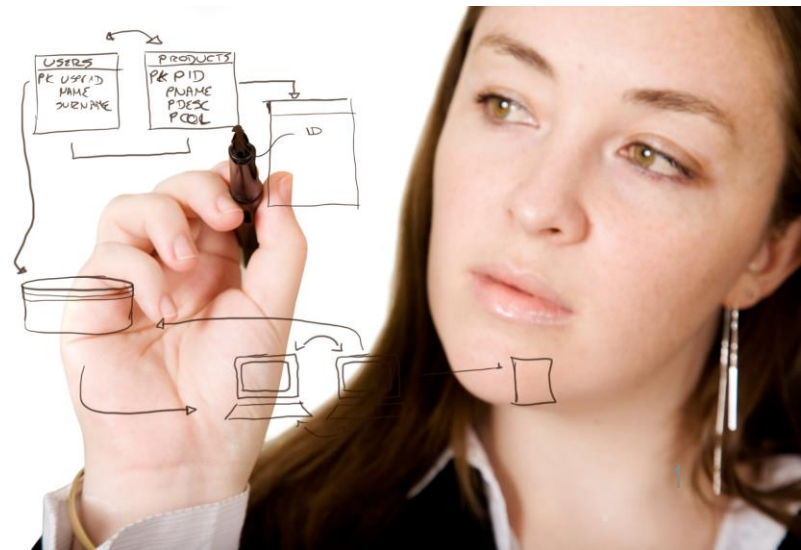
סמסטר ב' תשע"ו

2. מודל קונספטואלי לבקרת קוד מבוזרת

ד"ר ראובן יגל

robi@post.jce.ac.il

modeling17b-yagel



הפעם

- מבוא לבקרת תצורה
- משל גיט
- מידול גיט
- המשך תרגיל 1

מקורות

- Git usage
 - www...
- Git Modeling
 - The Architecture of Open Source Applications, Vol. II, Chap. 6. <http://www.aosabook.org/en/git.html>, 2012
 - “What’s Wrong with Git? A Conceptual Design Analysis” ([MIT paper](#), 2013) => [gitless](#)
 - www...

בקרת תצורה (SCM)

- לפרויקט תוכנה תוצרים שונים:
- מסמכי דרישות ותיכון, קוד, executables, מדריכי שימוש, בדיקות, ...
- פרויקט תוכנה משתמש בכלים שונים:
- מהדרים, עורכים, צד ג', שת"פ, (מ"ה), ...

בקרת תצורה

- מבחינת תהליך זה אלו נקראים
CI – Configuration Items
- לכל אחד יכולים להיות גרסאות ועותקים שונים
- אנו צריכים יכולת לזהות, לעקוב ולאחסן אותם
- נתמקד בנושא של גרסאות

בקרת גרסאות – Version Control

- איך (האם?) אתם שומרים את תוצרי העבודה שלכם?
- האם אפשר לשפר?
- האם יש הבדל בין מפתח בודד לחברה גדולה?
- שמות שונים:
 - בקרת תצורה
 - Revision Control
 - Software Configuration Management
 - Source-Code/**Version Control System**

בקרת גרסאות – בשביל מה? יעדים



- אחסון (גיבוי והצלה)
- איסוף כל הגרסאות ומעקב אחרי שינויים
 - חזרה לגרסה מסוימת, השוואה
 - מאפשר מחיקת קוד
- ניהול מספר גרסאות במקביל
- שיתוף מספר מפתחים (מרוחקים) בו זמנית
 - טיפול בסתירות
- מאגר מעודכן של תוצרי הפרויקט
 - במיוחד עם daily build

בפרויקט תדרשו להדגים את בקרת
התצורה שלכם

פעולות נדרשות

- בקרת שינויים
 - זיהוי ותיעוד (למשל מי משנה, הסיבה, זמן וכדו')
 - ניתוח והערכה (של שינוי)
 - אישור \ דחיה
 - אימות, מימש ושחרור
- בקרת גרסאות
 - מאגר
 - הכנסה והוצאה
 - ענפים ומיזוגים
 - תיוג



כלים: היסטוריה (השוואה)

Generation	Networking	Operations	Concurrency	Examples
1	None	One file at a time	Locks	RCS, SCCS
2	Centralized	Multi-file	Merge before commit	CVS, SourceSafe, Subversion, Team Foundation Server, IBM Rational ClearCase
3	Distributed	Changesets	Commit before merge	Bazaar, Git, Mercurial

40 Years of Version Control



SCCS & RCS (1970s)



CVS (1986)



Subversion (2001)



Git (2005)

Image © TheSun.au

Git Intro

- Distributed Version Control System
- Linux
 - Many contributors
 - Using “tarballs” / patches
 - BitKeeper & CVS
 - Linus develop alt. scripts, Goals:
 - Distributed workflows
 - Guard against content corruption
 - Performance

Distributed VCS

- Providing the ability for collaborators to work offline and commit incrementally
- Allowing a collaborator to determine when his/her work is ready to share
- Offering the collaborator access to the repository history when offline
- Allowing the managed work to be published to multiple repositories, potentially with different branches or granularity of changes visible
- Other: Bazaar, Darcs, Fossil, Mercurial, Veracity

Git

- Successful open source project
 - <https://git.wiki.kernel.org/index.php/GitProjects>
 - <https://github.com/google>, [microsoft](#), facebook, twitter...
 - <http://stackoverflow.com/research/developer-survey-2015#tech-sourcecontrol>
- Problems / Issues:
 - Usability!
 - Mainly a scripted / toolset (by now IDE integration and GUIs)
 - Binary/big file (also taken care)
 - Enterprise (e.g. locking)

משל גיט

- Tom Preston-Werner
<http://tom.preston-werner.com/2009/05/19/the-git-parable.html>
- Herland,
<http://www.infoq.com/presentations/git-details>, slides:
https://github.com/jherland/git_parable

The Git Parable

Johan Herland

johan@herland.net

The Git Parable

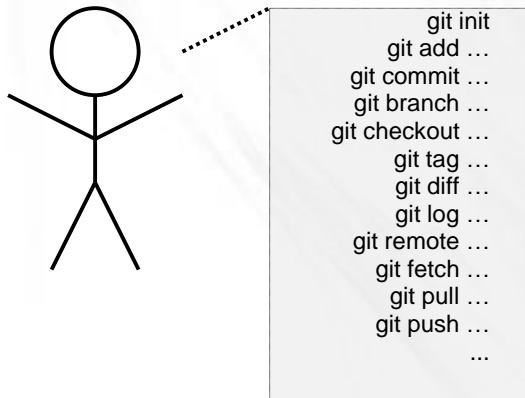
- Shamelessly stolen from Tom Preston-Werner
<http://tom.preston-werner.com/2009/05/19/the-git-parable.html>
- I'm lazy...
- Also: Best introduction to Git I've found so far

The Git Parable

- Git - simple & powerful

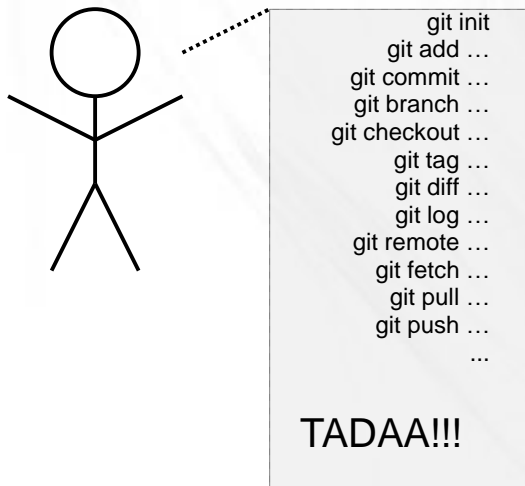
The Git Parable

- Git - simple & powerful



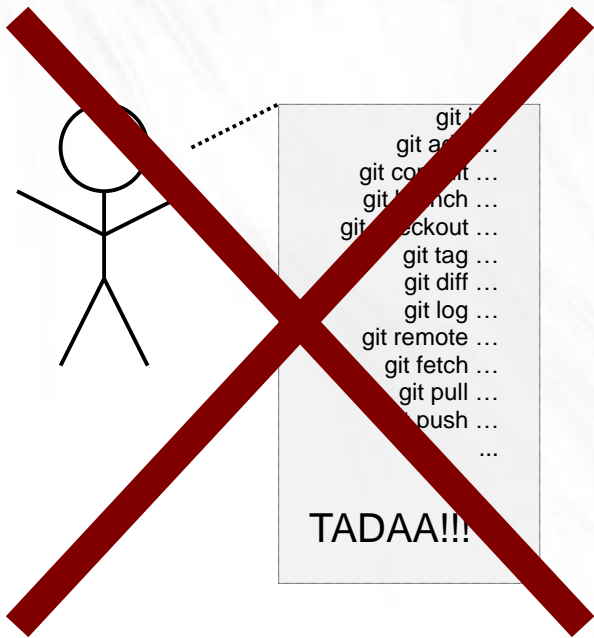
The Git Parable

- Git - simple & powerful



The Git Parable

- Git - simple & powerful



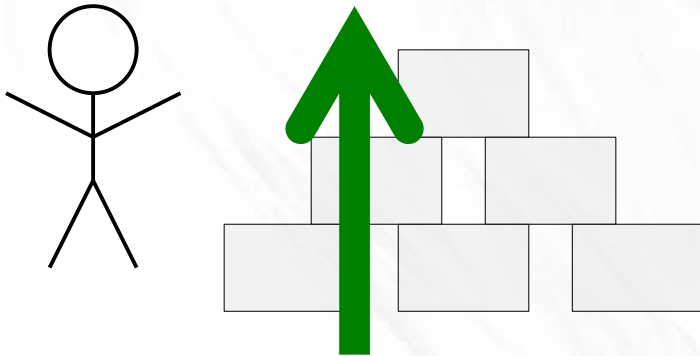
The Git Parable

- Git - simple & powerful



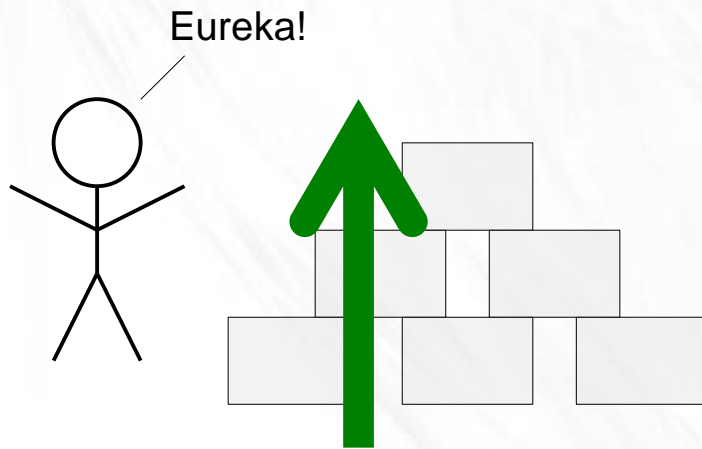
The Git Parable

- Git - simple & powerful



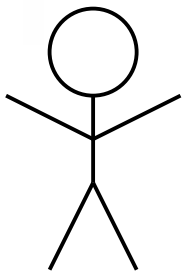
The Git Parable

- Git - simple & powerful



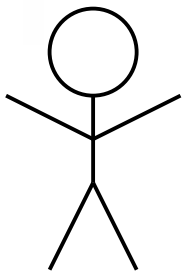
The Parable

- A simple computer
 - A text editor
 - A few filesystem commands



The Parable

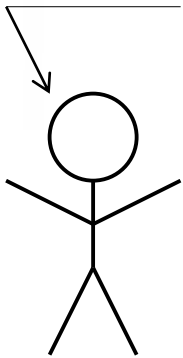
- Write a large software program



The Parable

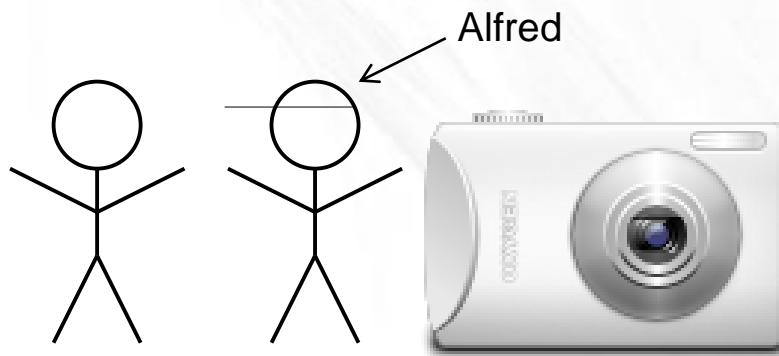
- Write a large software program
- Invent some method to keep track of versions
 - retrieve code that you changed/deleted

Responsible!



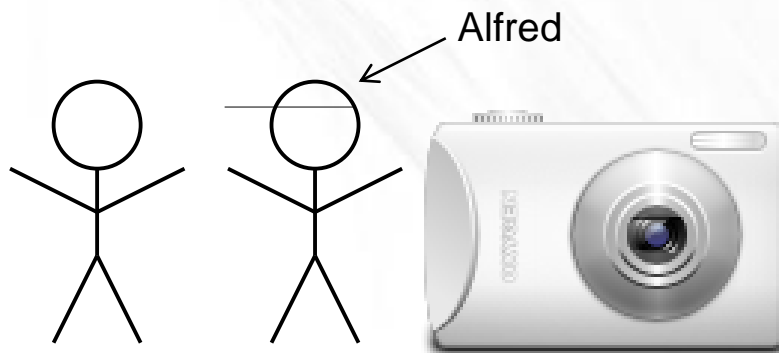
Snapshots

- Alfred, the photographer



Snapshots

- Alfred, the photographer



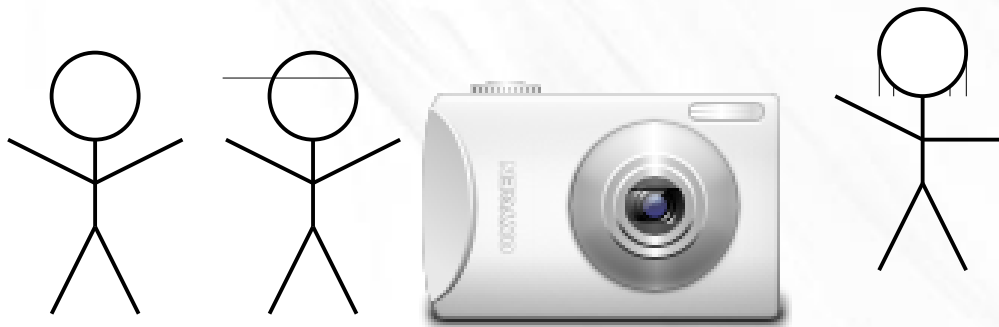
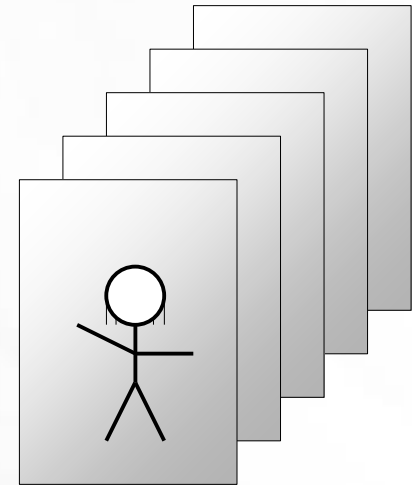
Snapshots

- Alfred, the photographer
- Hazel and her daughter



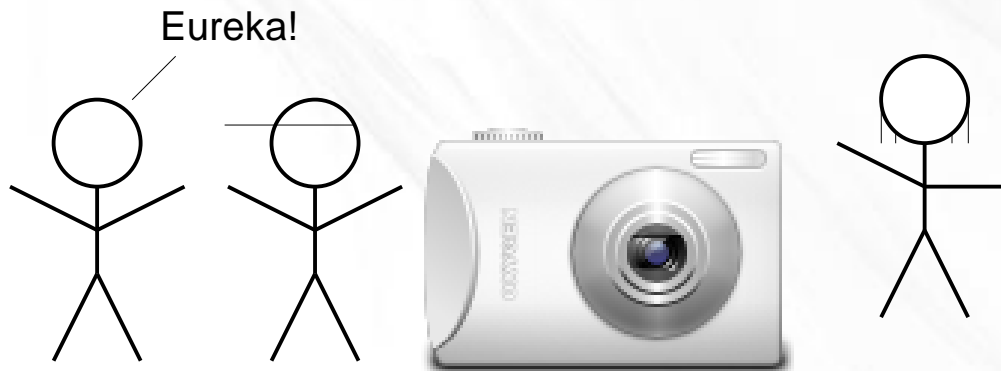
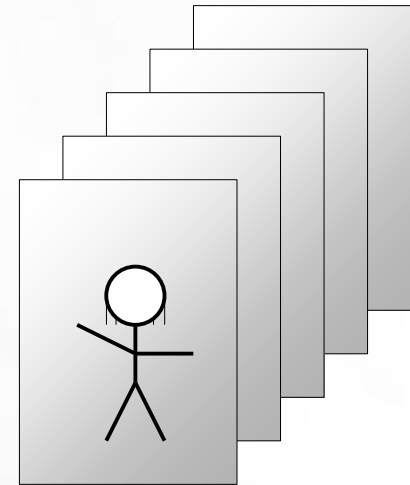
Snapshots

- Alfred, the photographer
- Hazel and her daughter
 - Remember what the daughter was like at each different stage



Snapshots

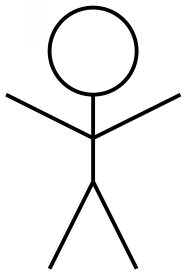
- Alfred, the photographer
- Hazel and her daughter
 - Remember what the daughter was like at each different stage



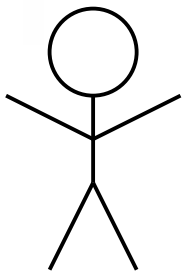
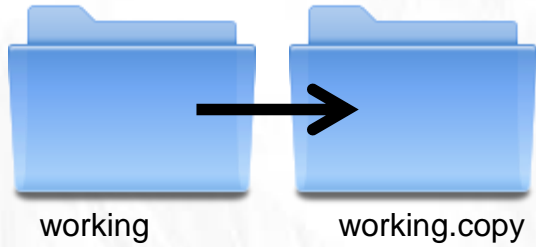
Snapshots



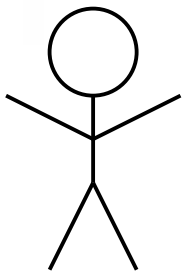
working



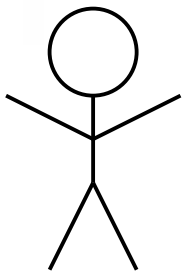
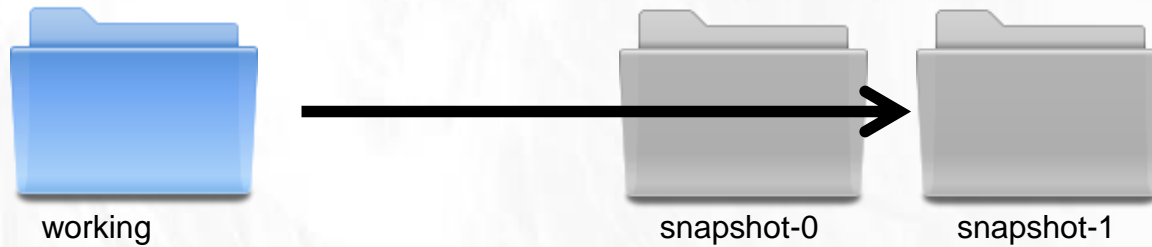
Snapshots



Snapshots



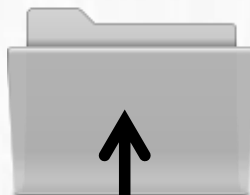
Snapshots



Snapshots



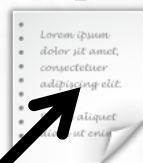
working



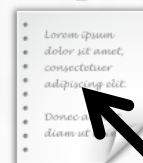
snapshot-0



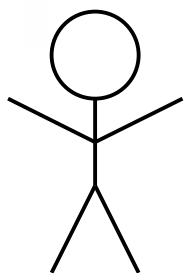
snapshot-1



message



message



2009-05-20 12:34:56

Initial version

2009-05-21 23:45:01

Introduced a new foo,
and reset the bar to
xyzy.

Branches



working



snapshot-0

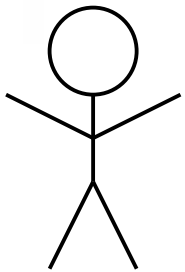


snapshot-1

...



snapshot-99



Branches



working



snapshot-0

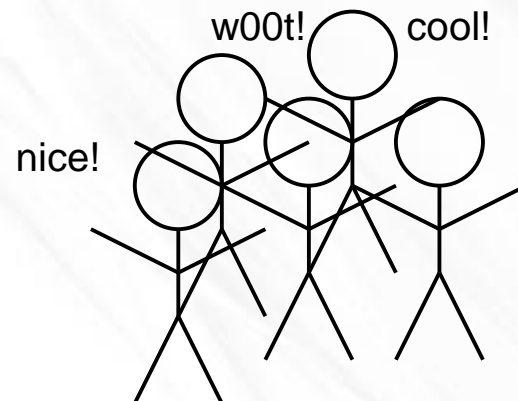
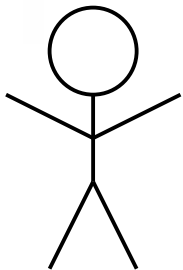


snapshot-1

...



snapshot-99



Branches



working



snapshot-0



snapshot-1

...

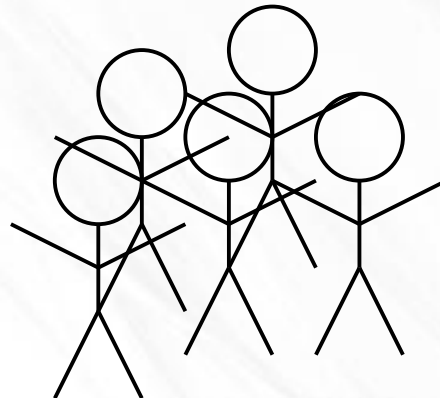
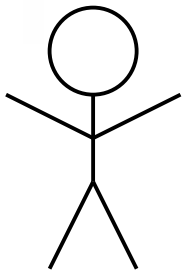


snapshot-99

...



snapshot-109



Branches



working



snapshot-0



snapshot-1

...



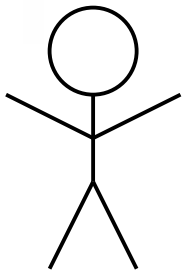
snapshot-99

...

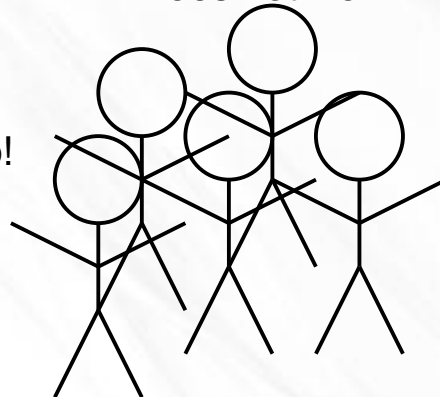


snapshot-109

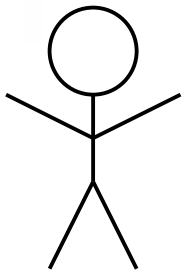
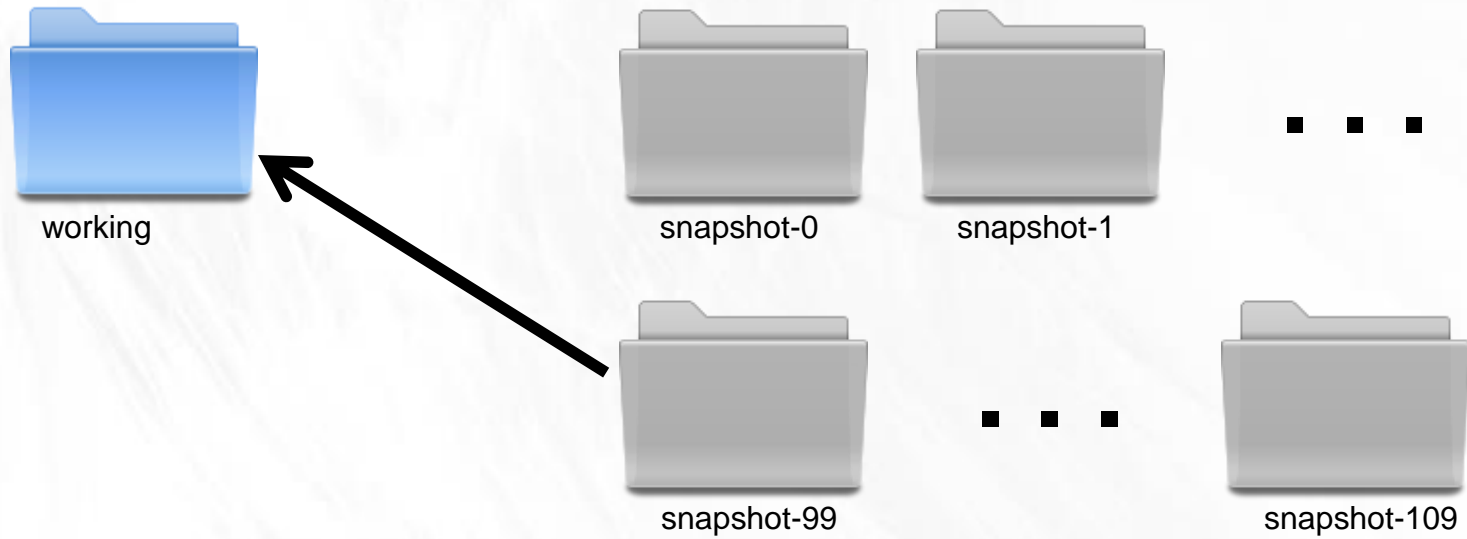
Does not work!



Boo!



Branches



Branches



working



snapshot-0



snapshot-1

...



snapshot-99

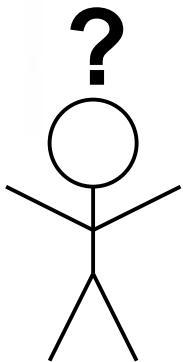
...



snapshot-109



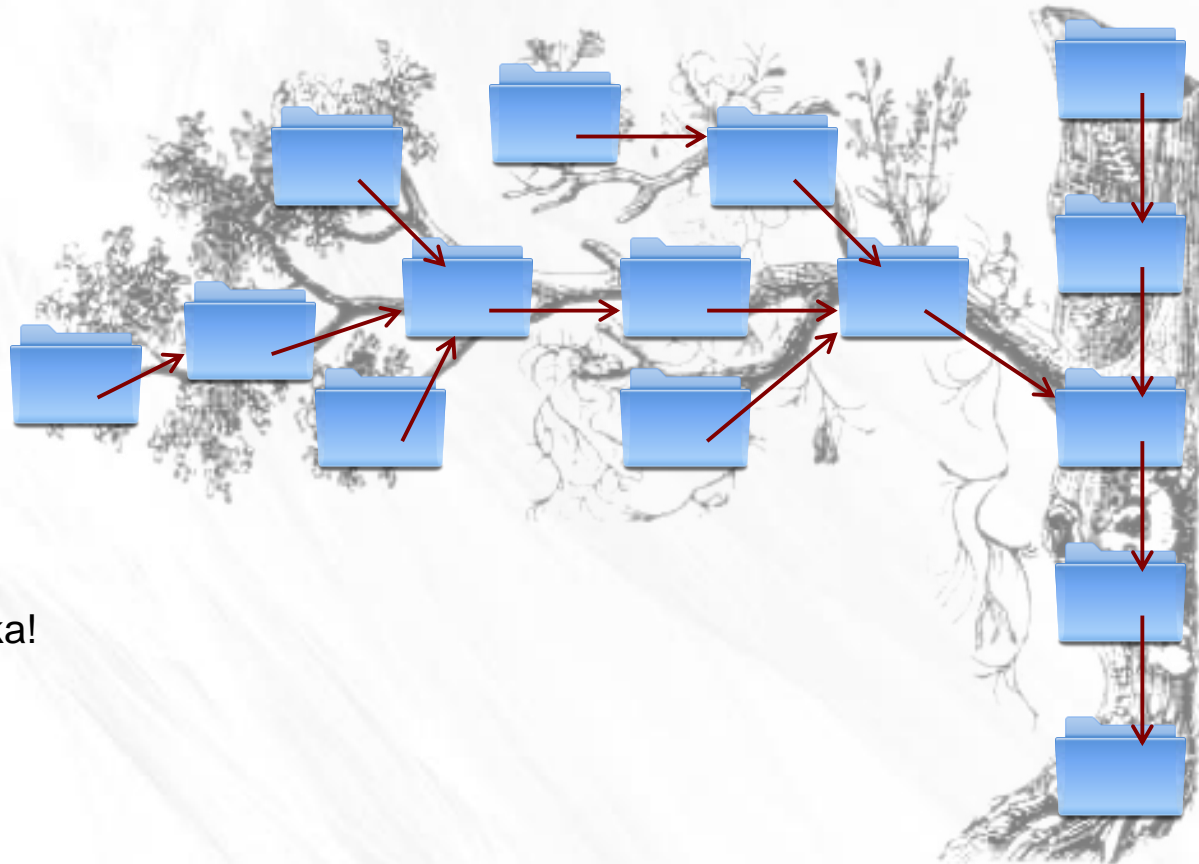
snapshot-110



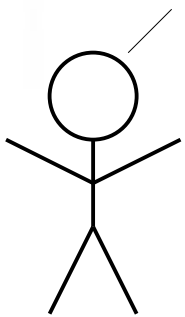
Branches



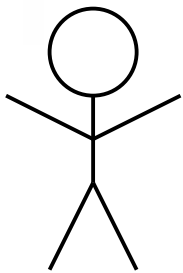
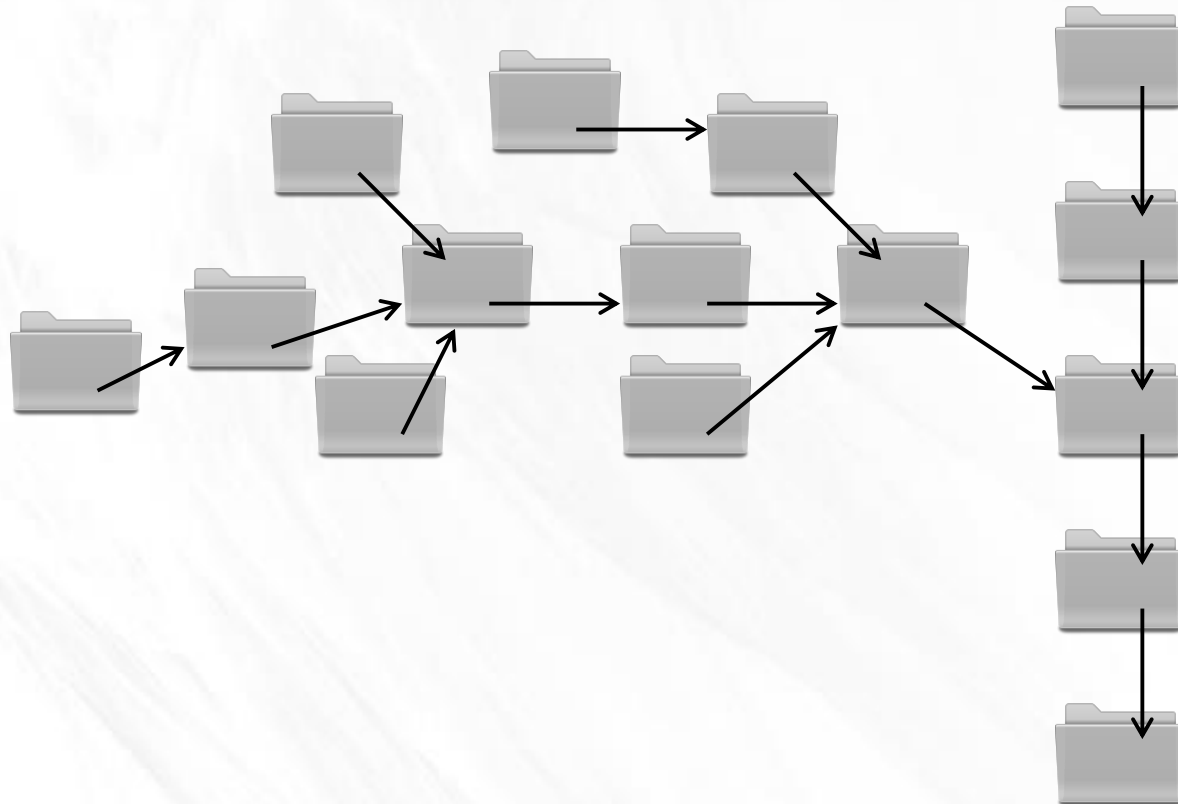
Branches



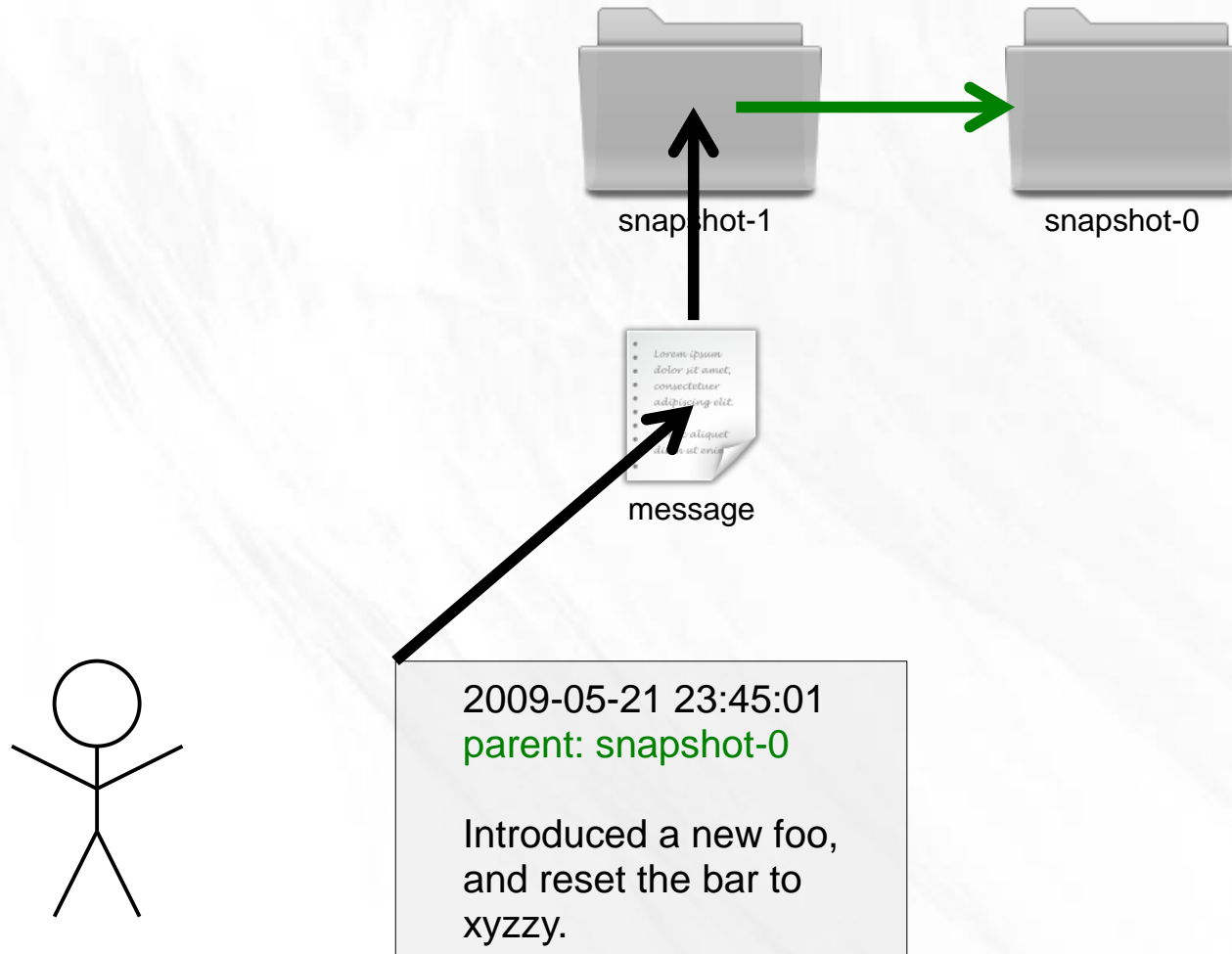
Eureka!



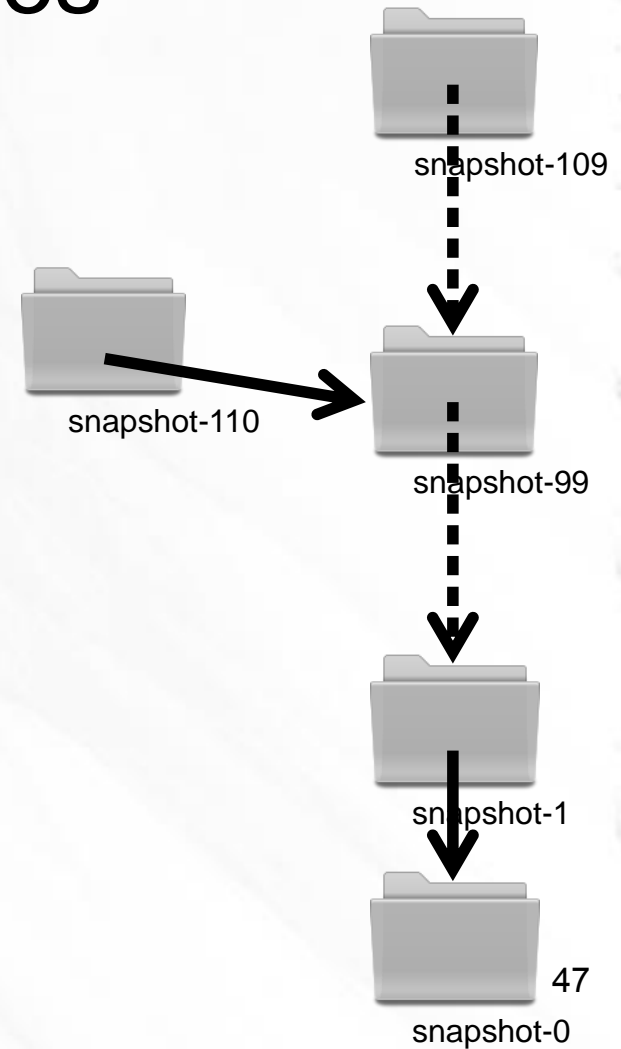
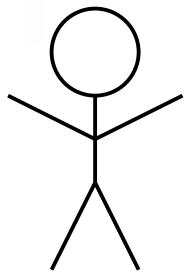
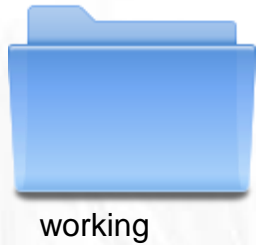
Branches



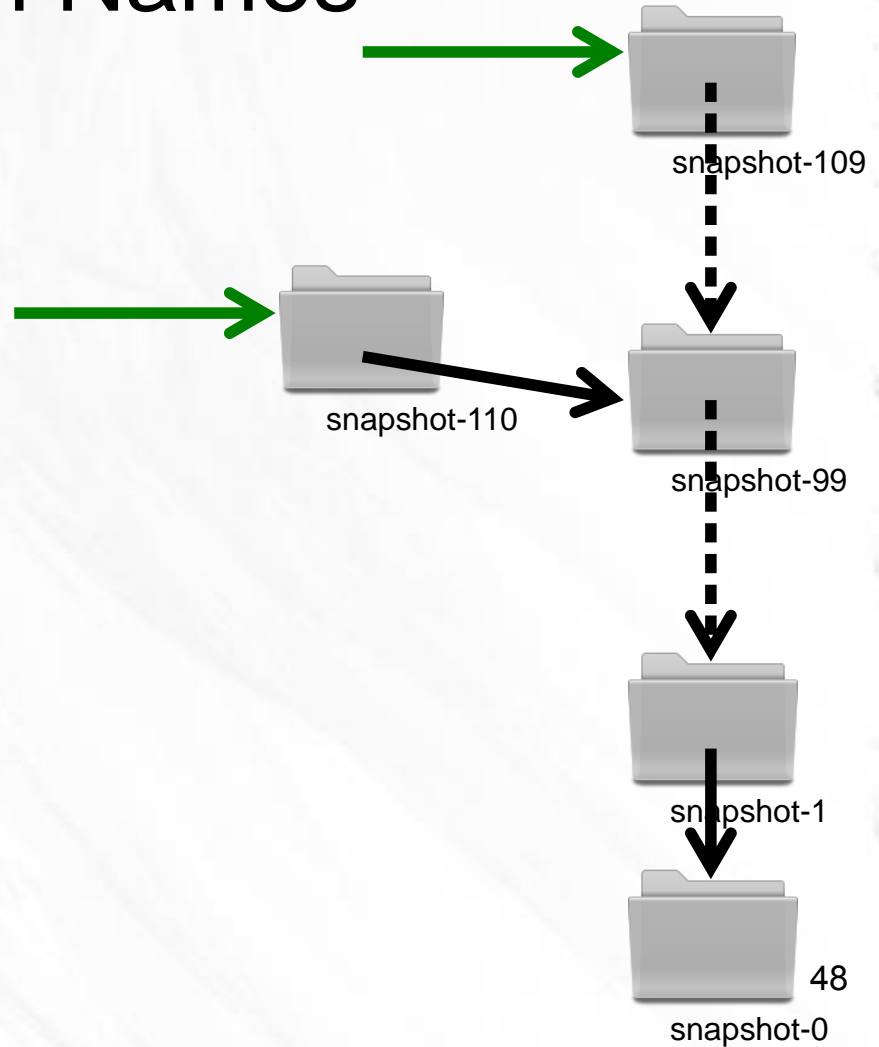
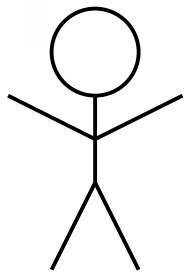
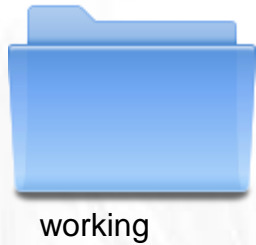
Branches



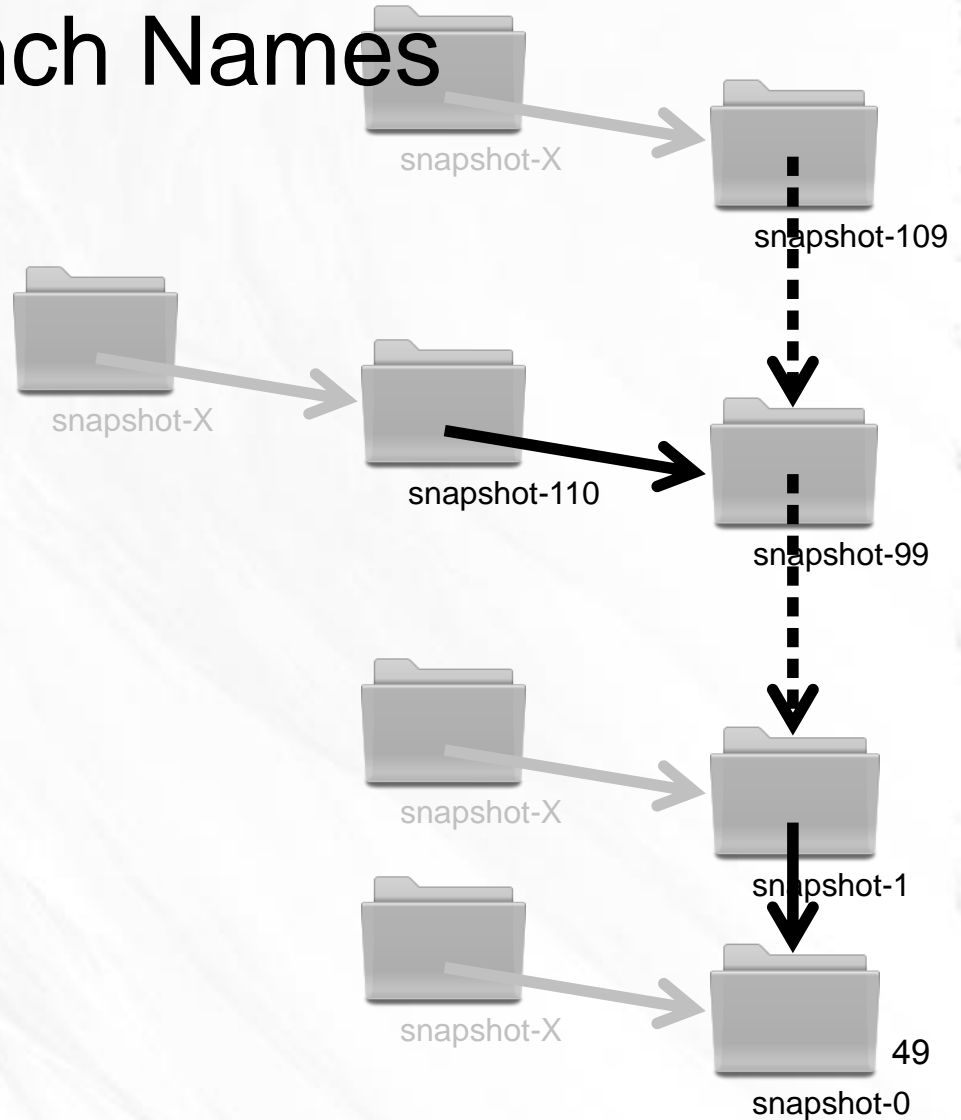
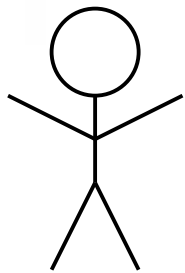
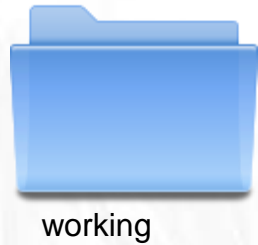
Branch Names



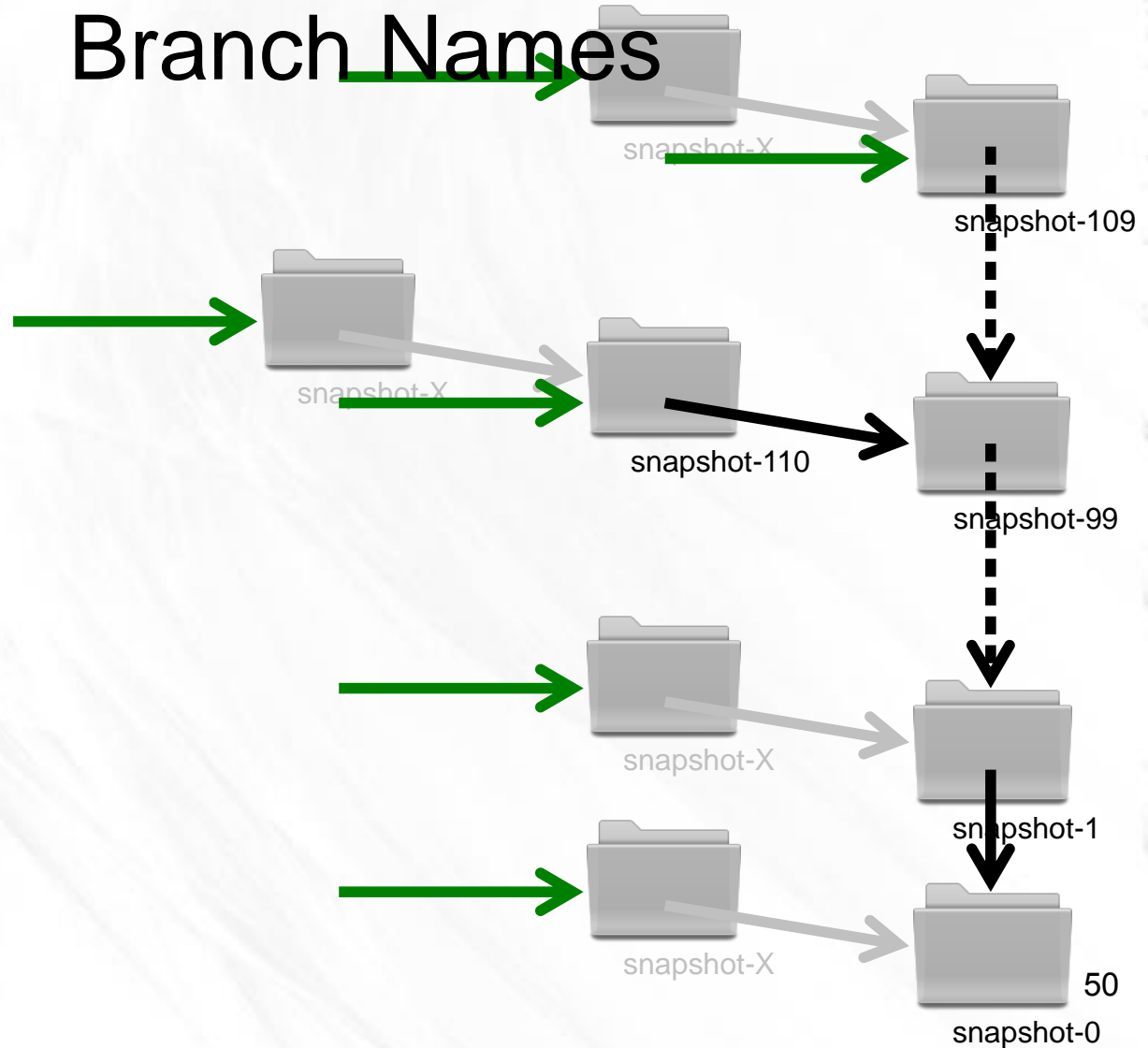
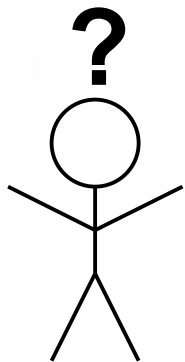
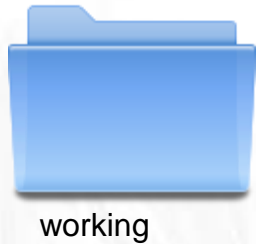
Branch Names



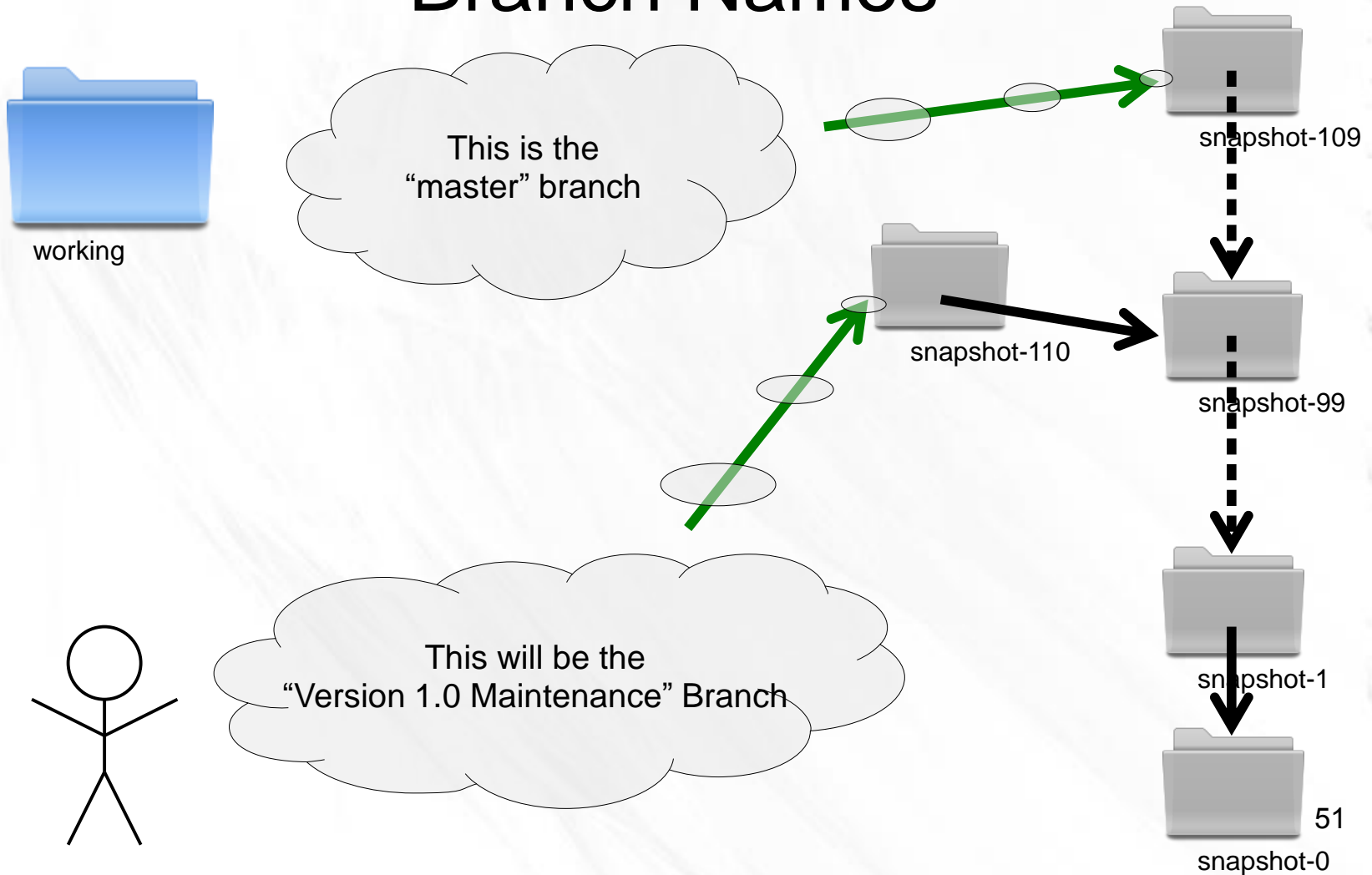
Branch Names



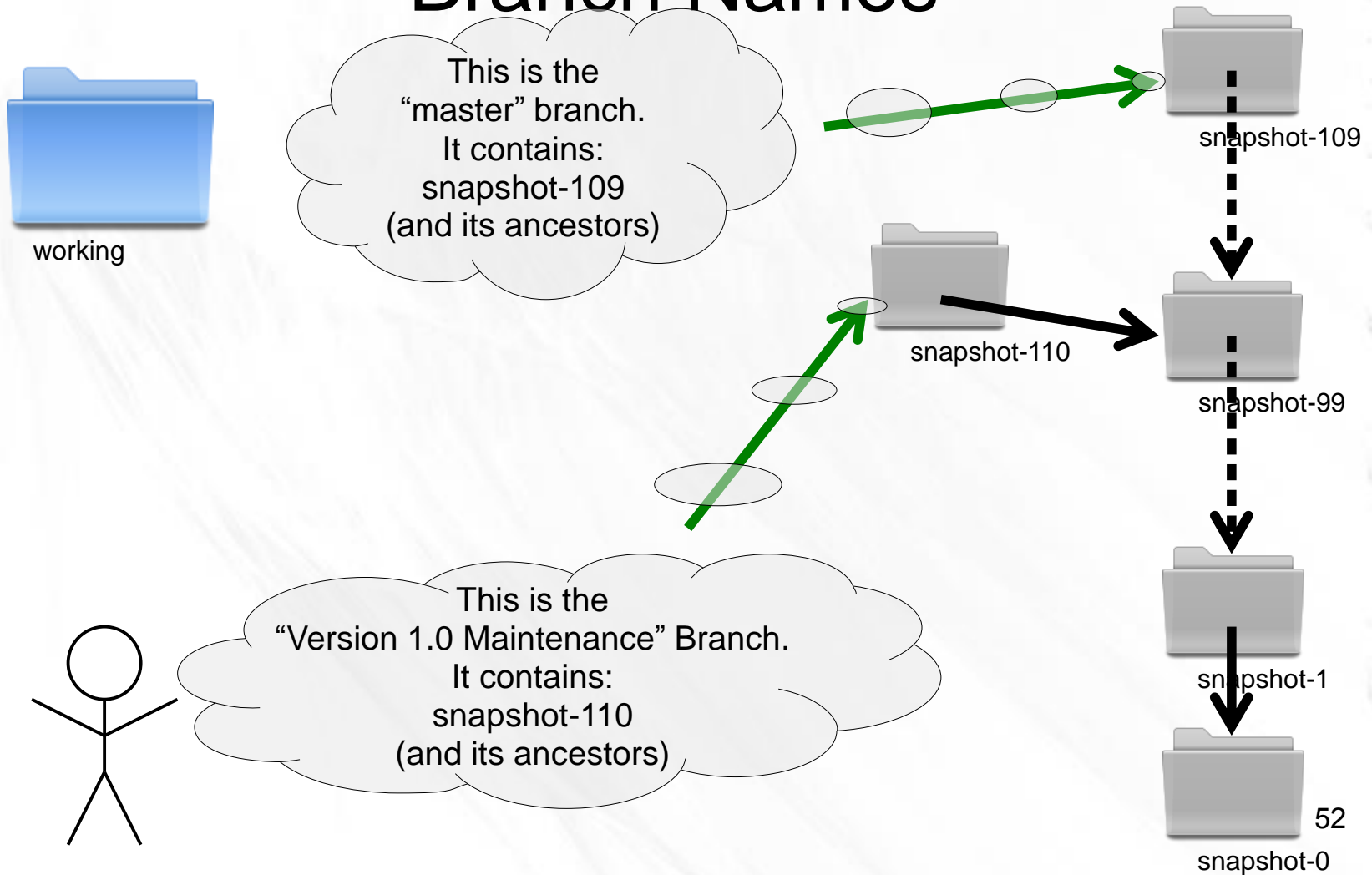
Branch Names



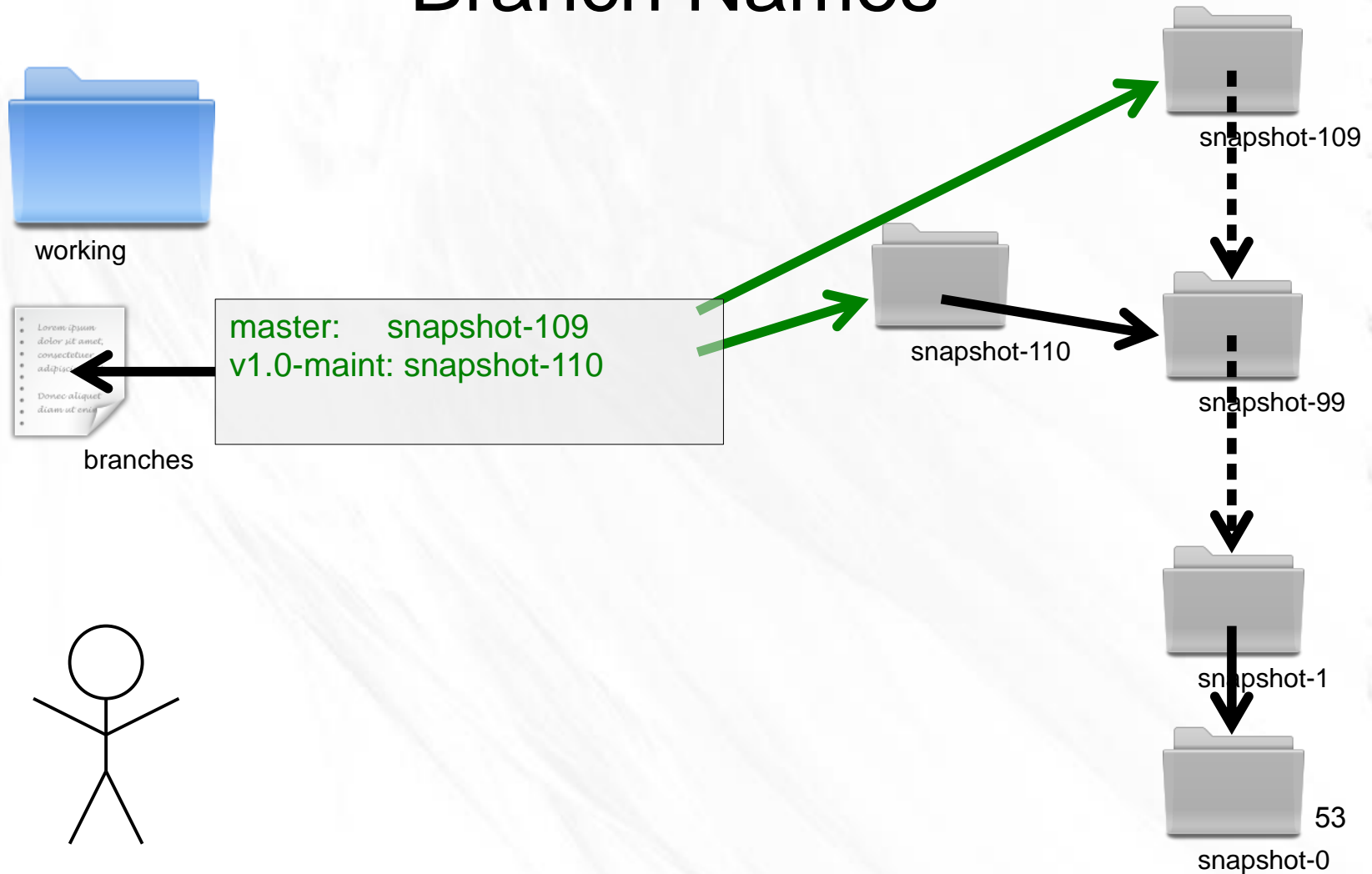
Branch Names



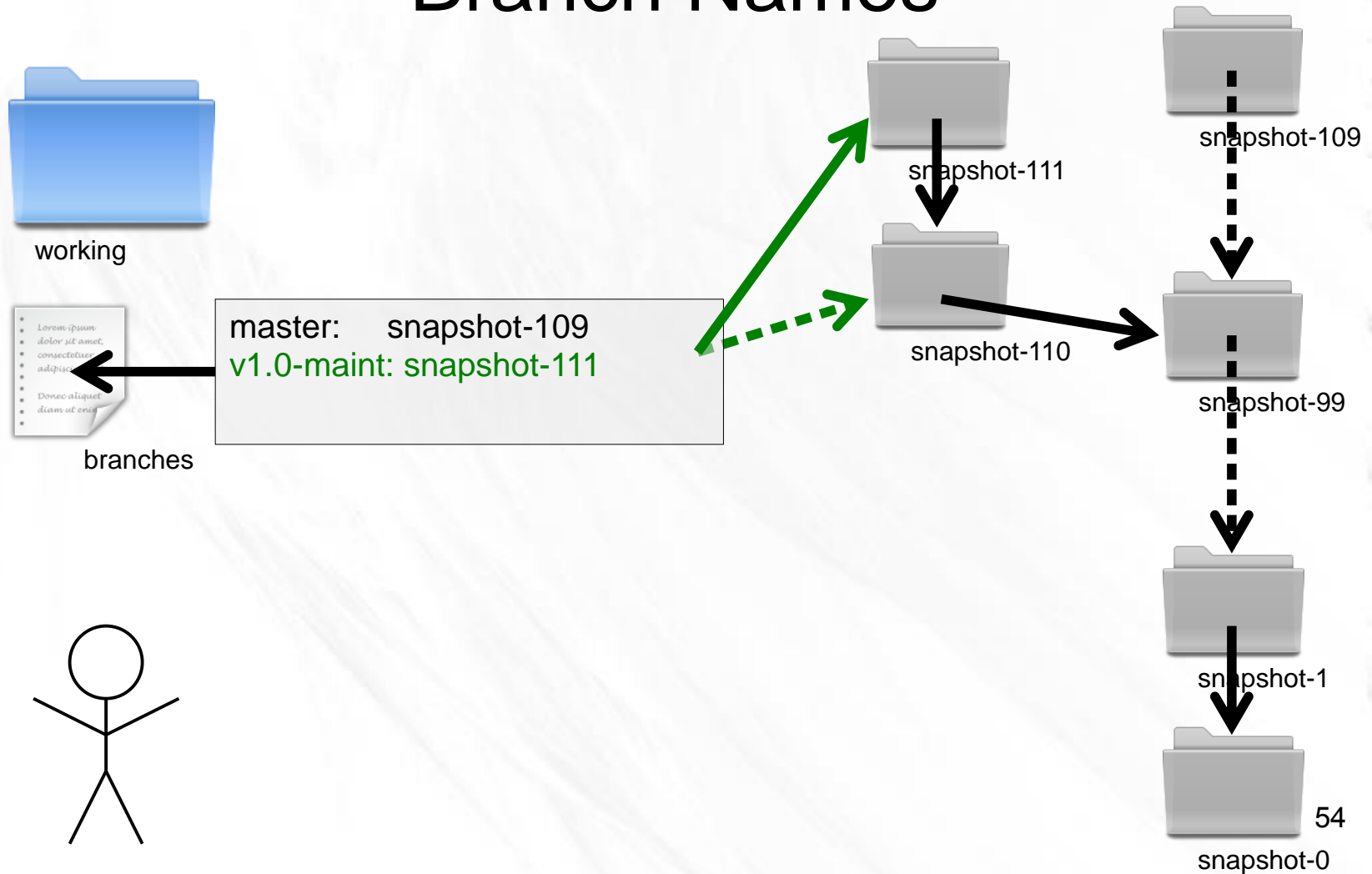
Branch Names



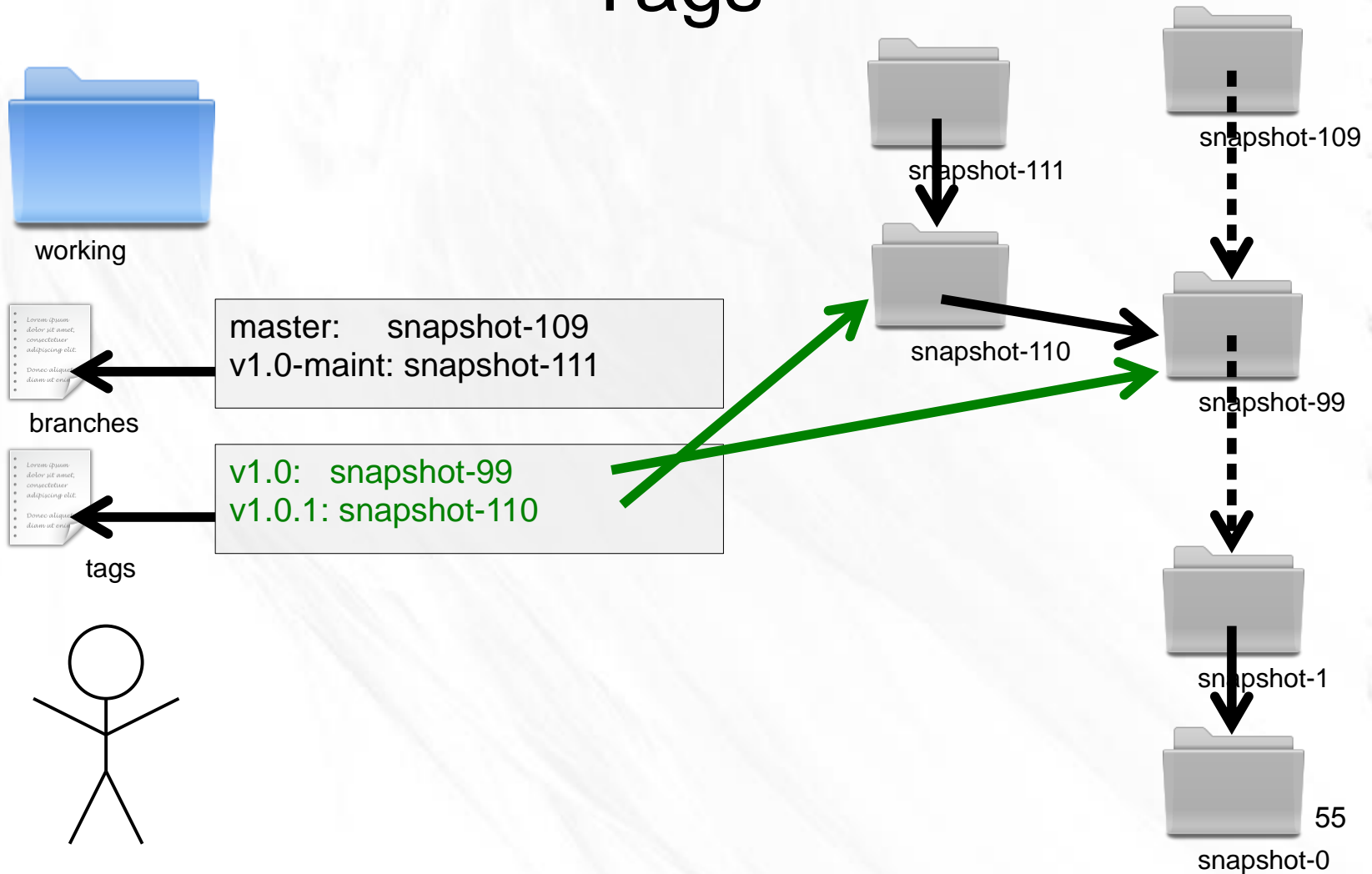
Branch Names



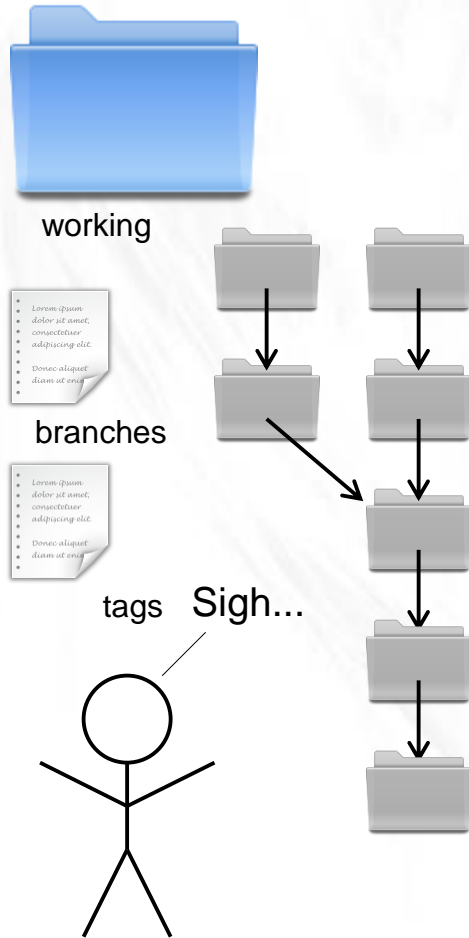
Branch Names



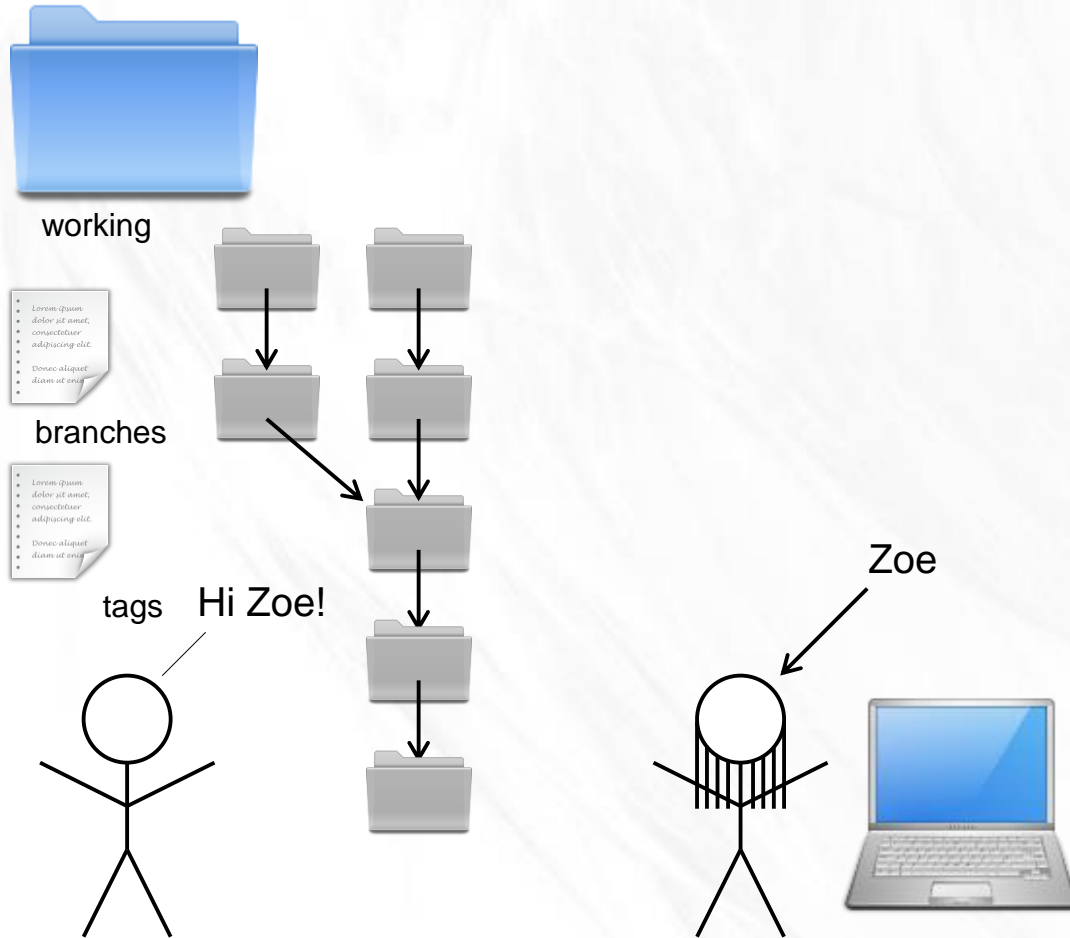
Tags



Distributed



Distributed



Distributed



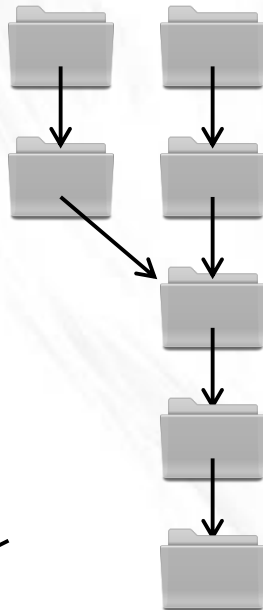
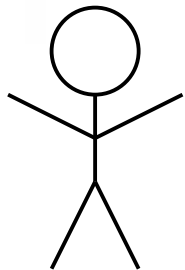
working



branches



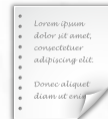
tags



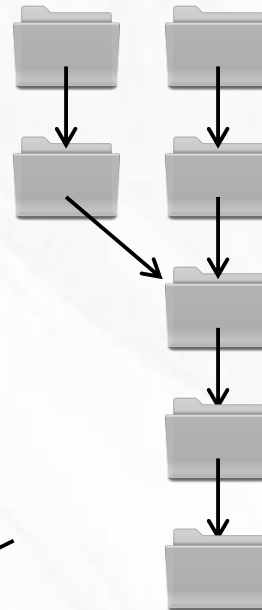
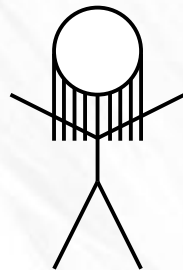
working



branches



tags



Distributed



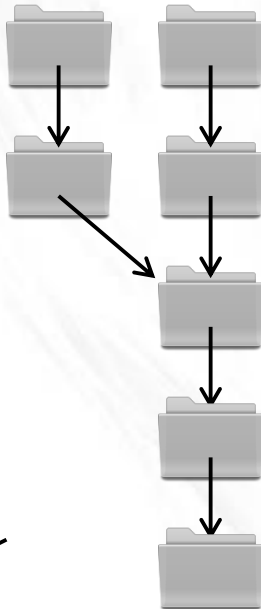
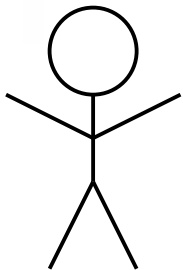
working



branches



tags



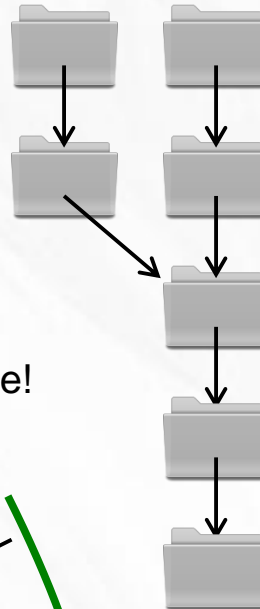
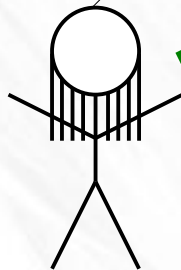
working



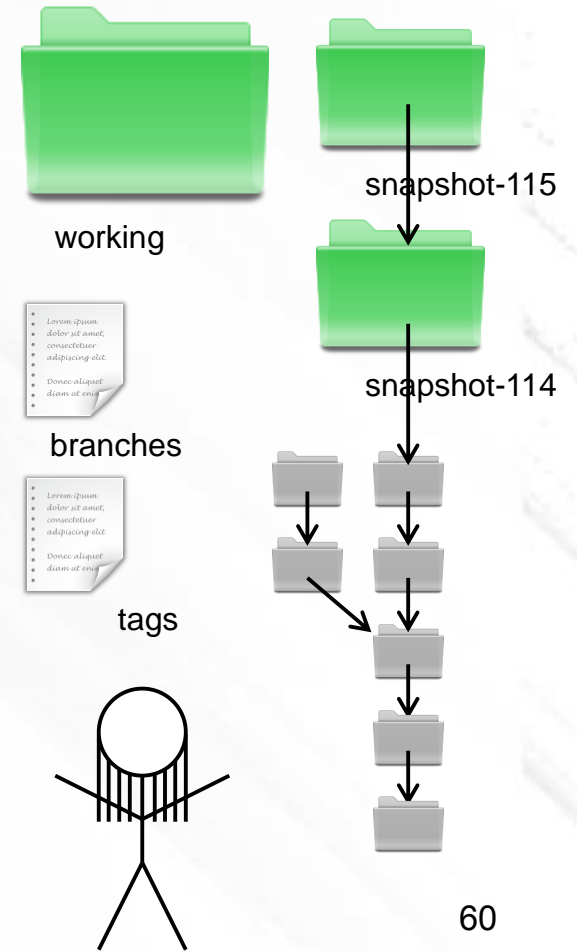
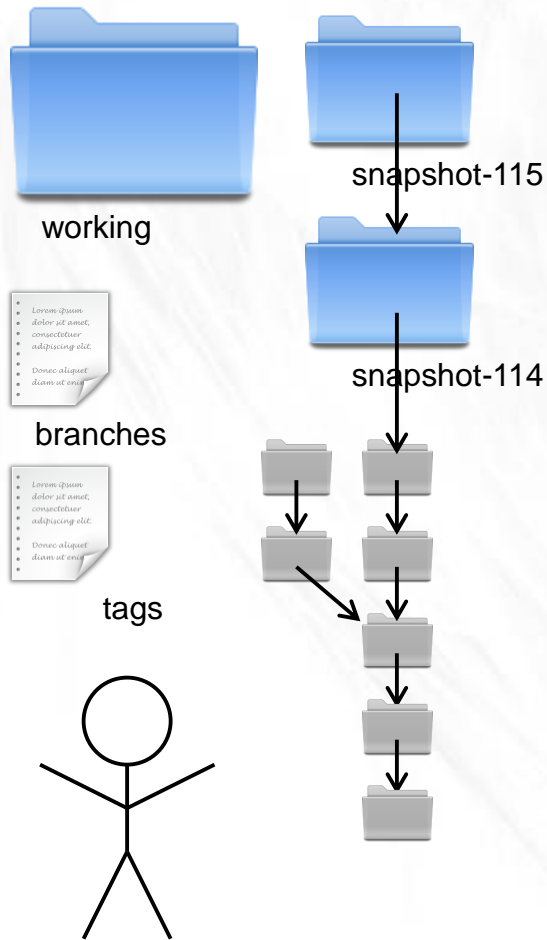
branches



tags Bye!



Distributed



Distributed



snapshot-114



message

2009-05-22 12:12:12
parent: snapshot-113
author: Me <me@me.me>

Blarfle, a cool new
feature; extends the
existing blog.



snapshot-114



message

2009-05-21 23:45:01
parent: snapshot-113
author: Zoe <zoe@z.oe>

Introduced a new foo,
and reset the bar to 61
xyzyy.

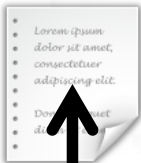
Distributed



8ba3441b6b89cad23387ee875f2ae55069291f4b



SHA1



message

2009-05-22 12:12:12
parent: snapshot-113
author: Me <me@me.me>

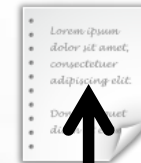
Blarfle, a cool new
feature; extends the
existing blog.



db9ecb5b5a6294a8733503ab57577db96ff2249e



SHA1

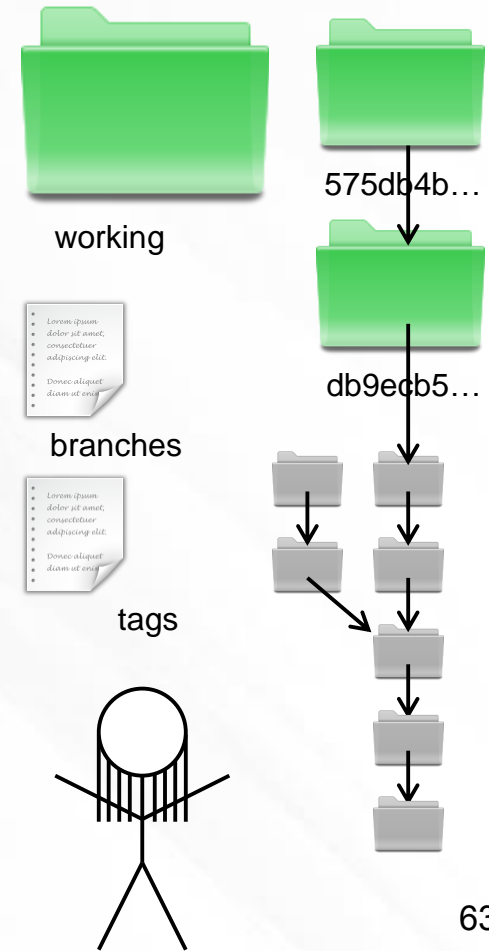
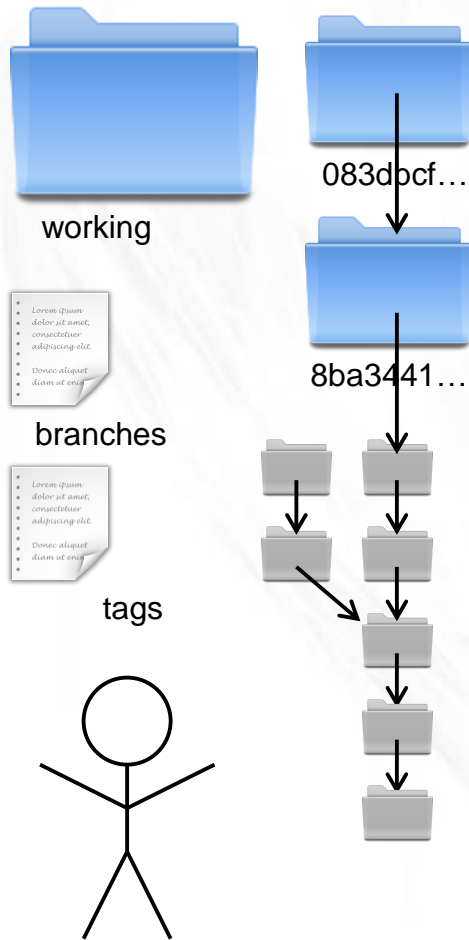


message

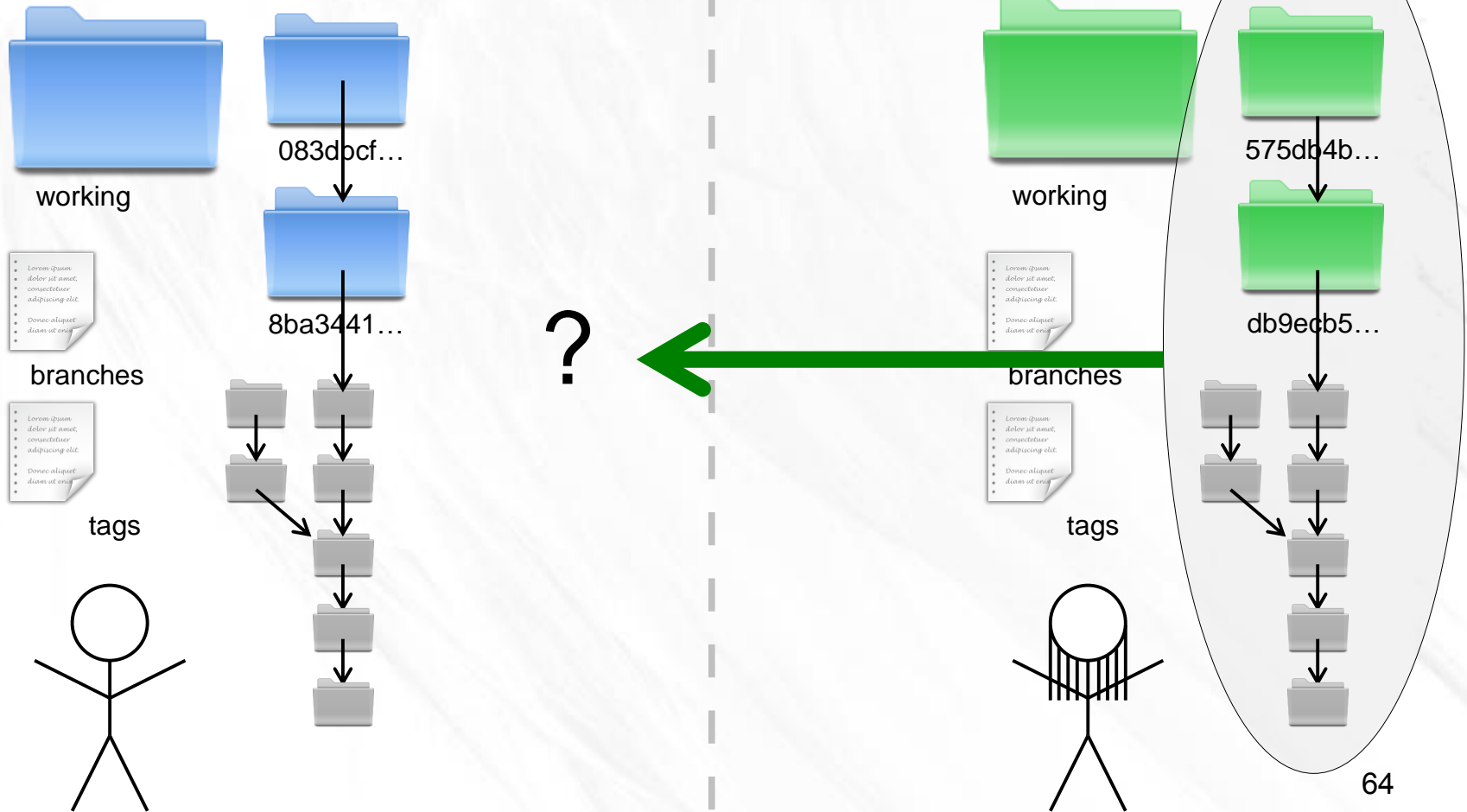
2009-05-21 23:45:01
parent: snapshot-113
author: Zoe <zoe@z.oe>

Introduced a new foo, 62
and reset the bar to
xyzyz.

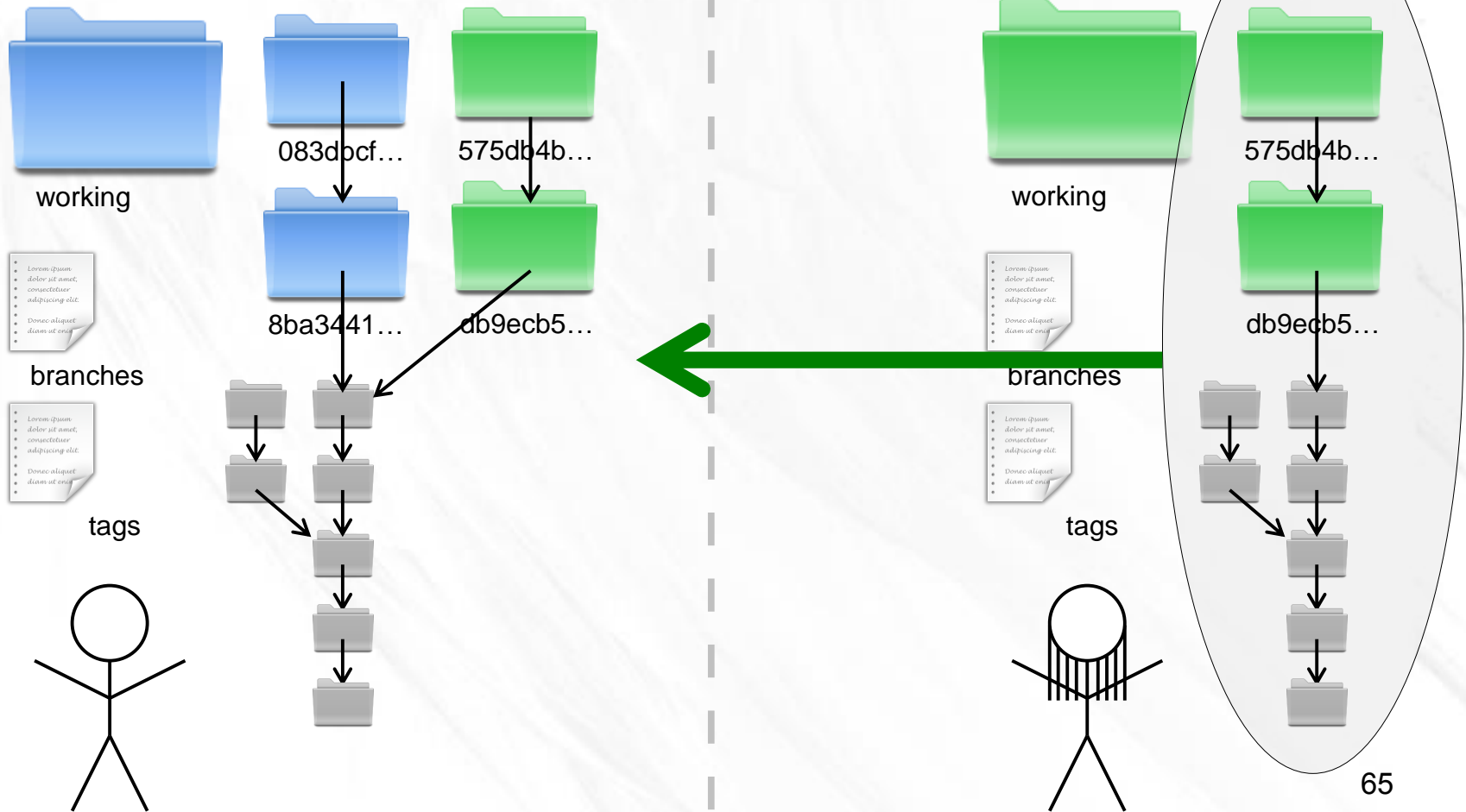
Distributed



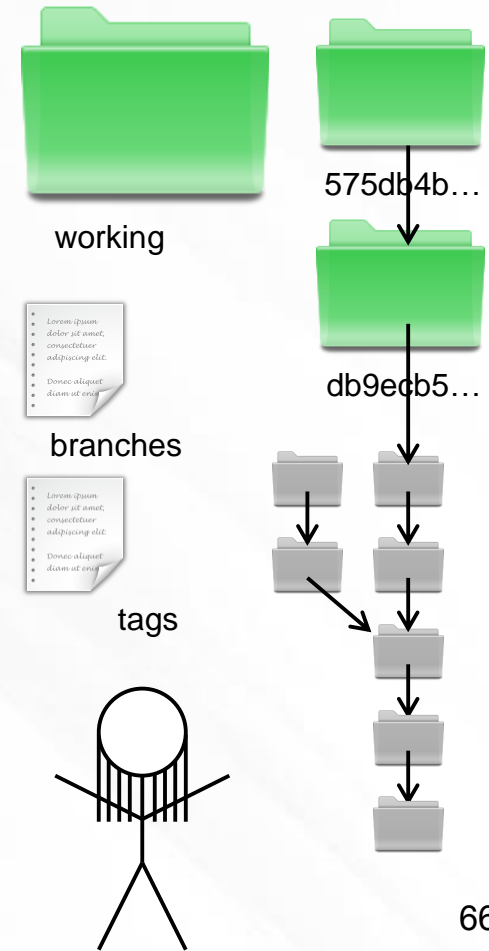
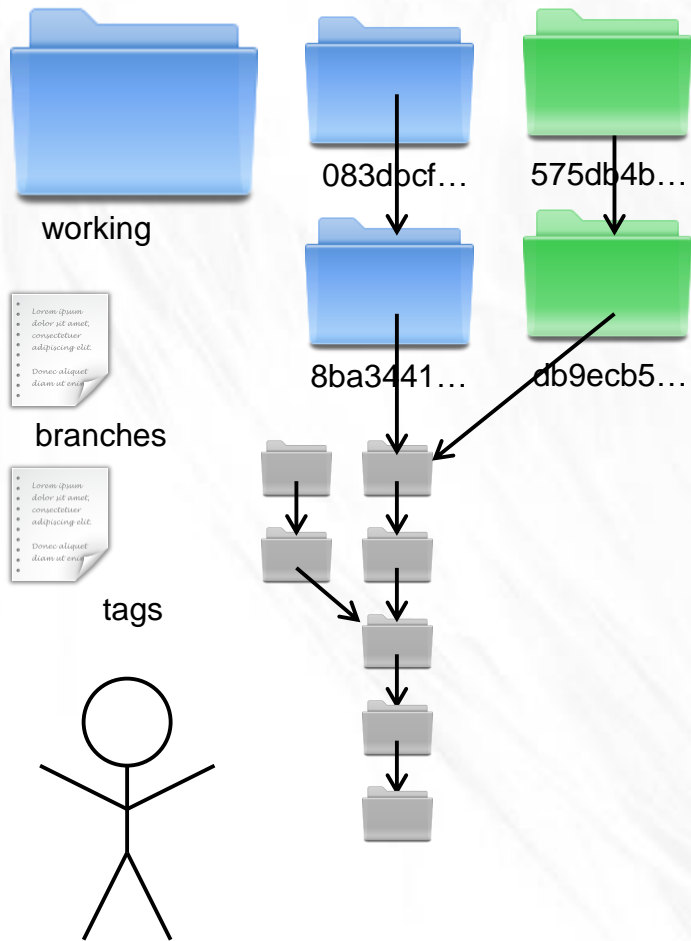
Distributed



Distributed



Distributed



Offline



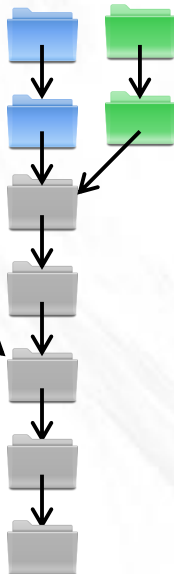
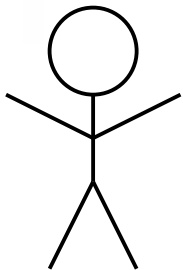
working



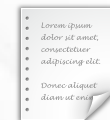
branches



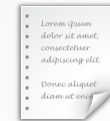
tags



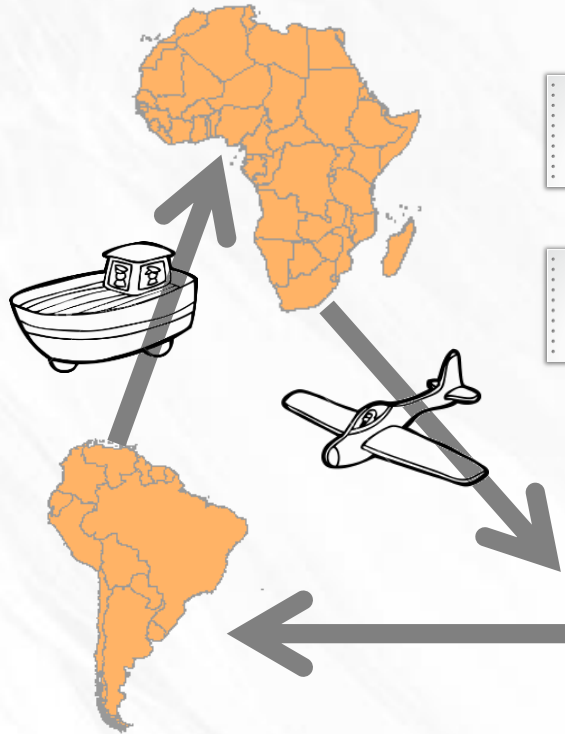
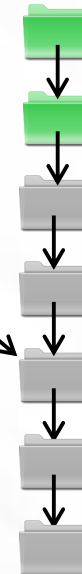
working



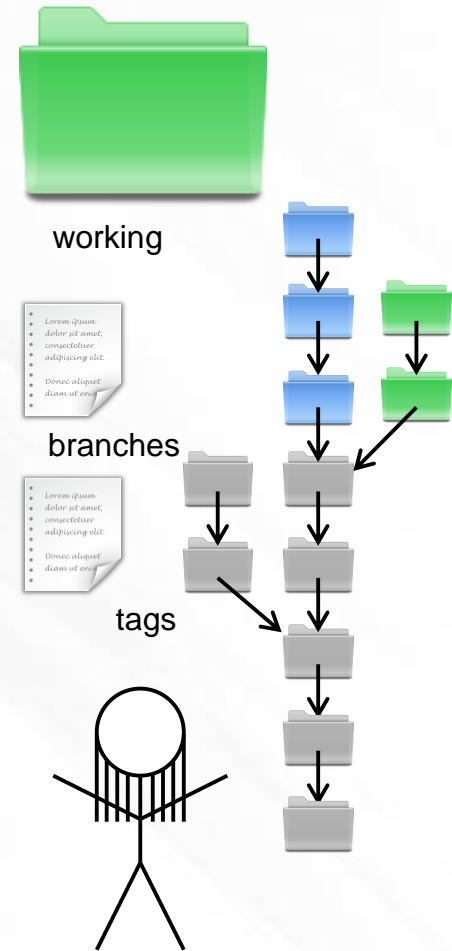
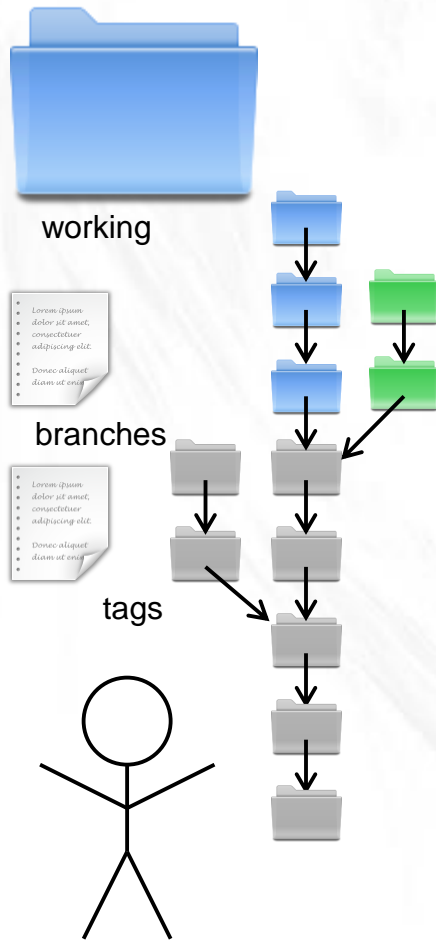
branches



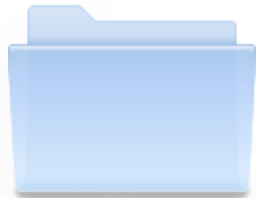
tags
w00t!



Offline



(simpler drawings)



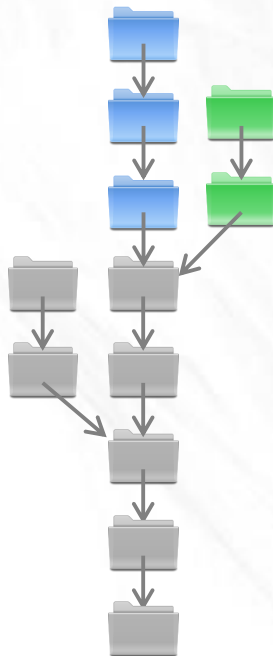
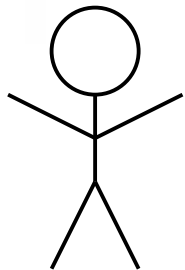
working



branches



tags



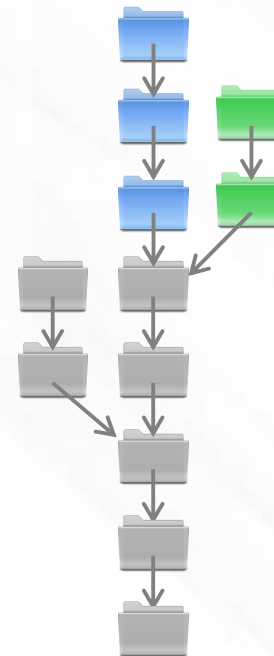
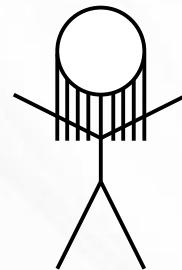
working



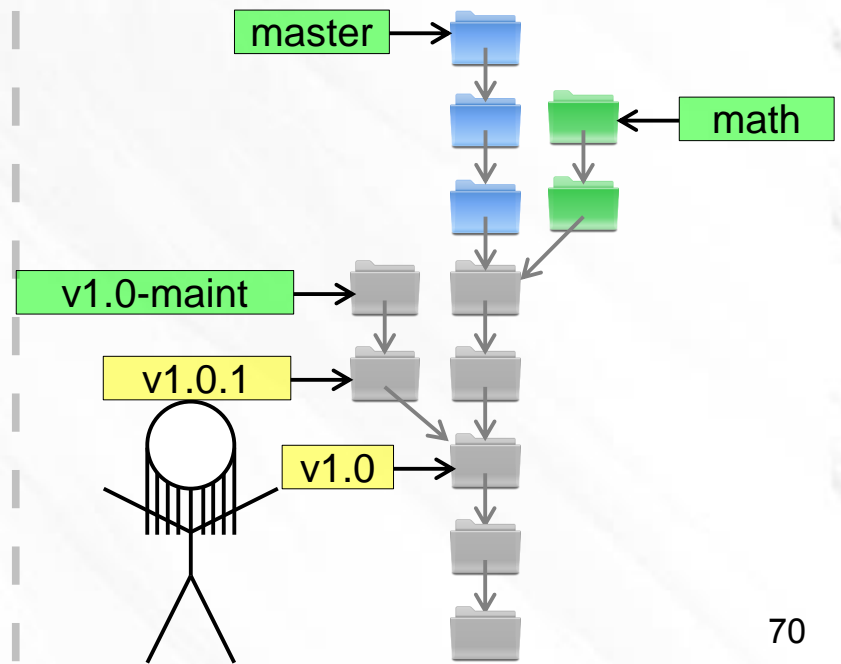
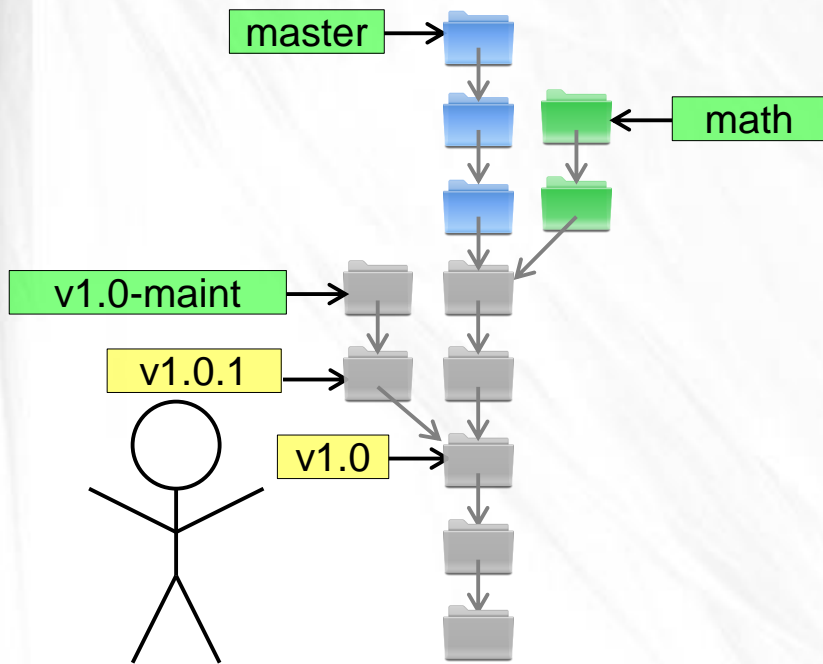
branches



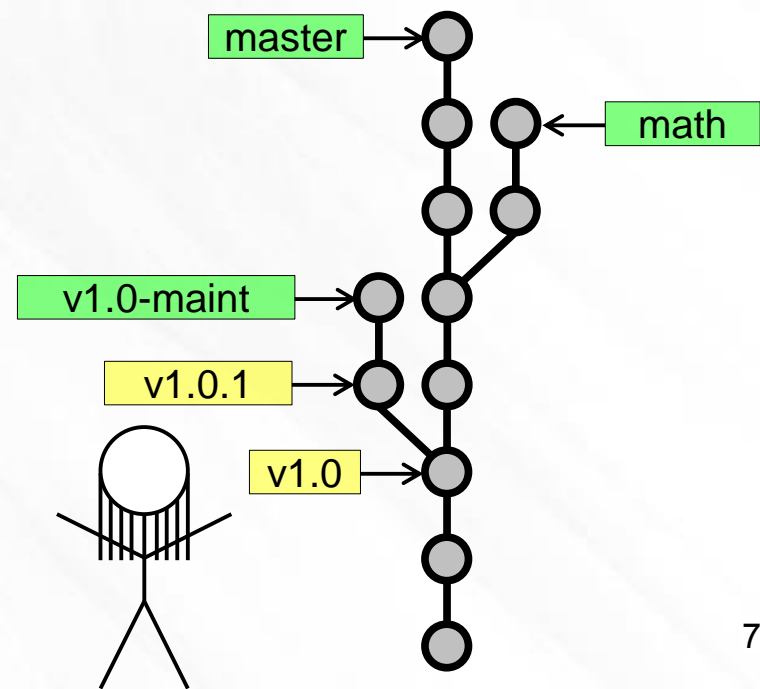
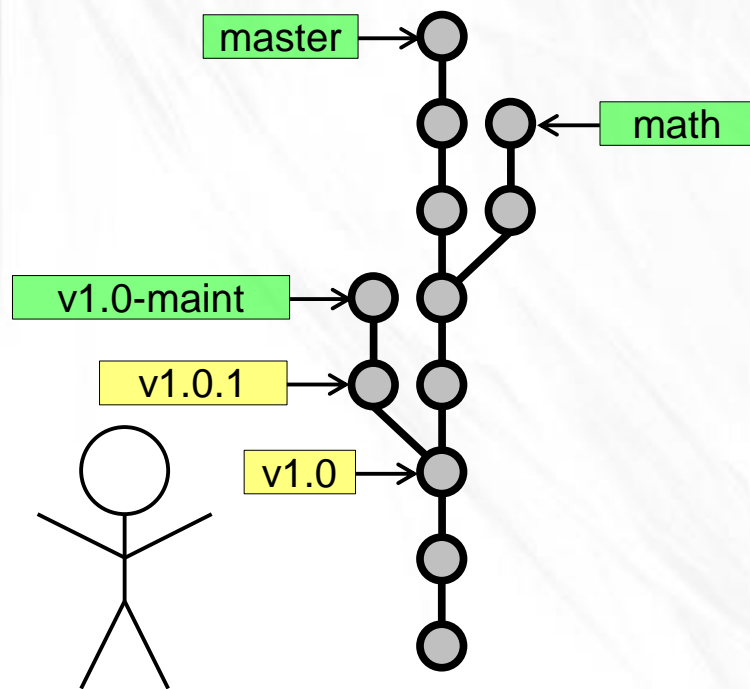
tags



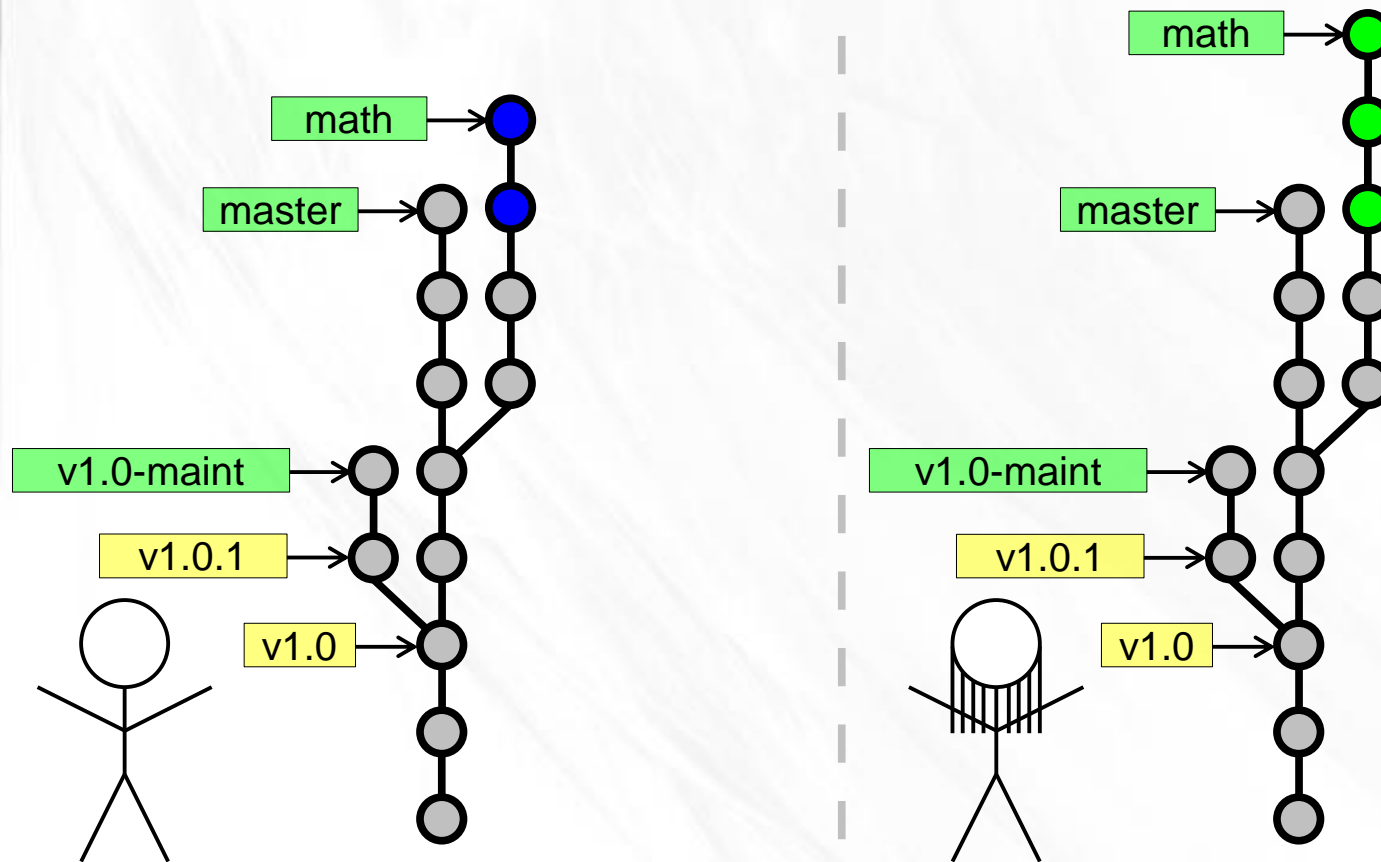
(simpler drawings)



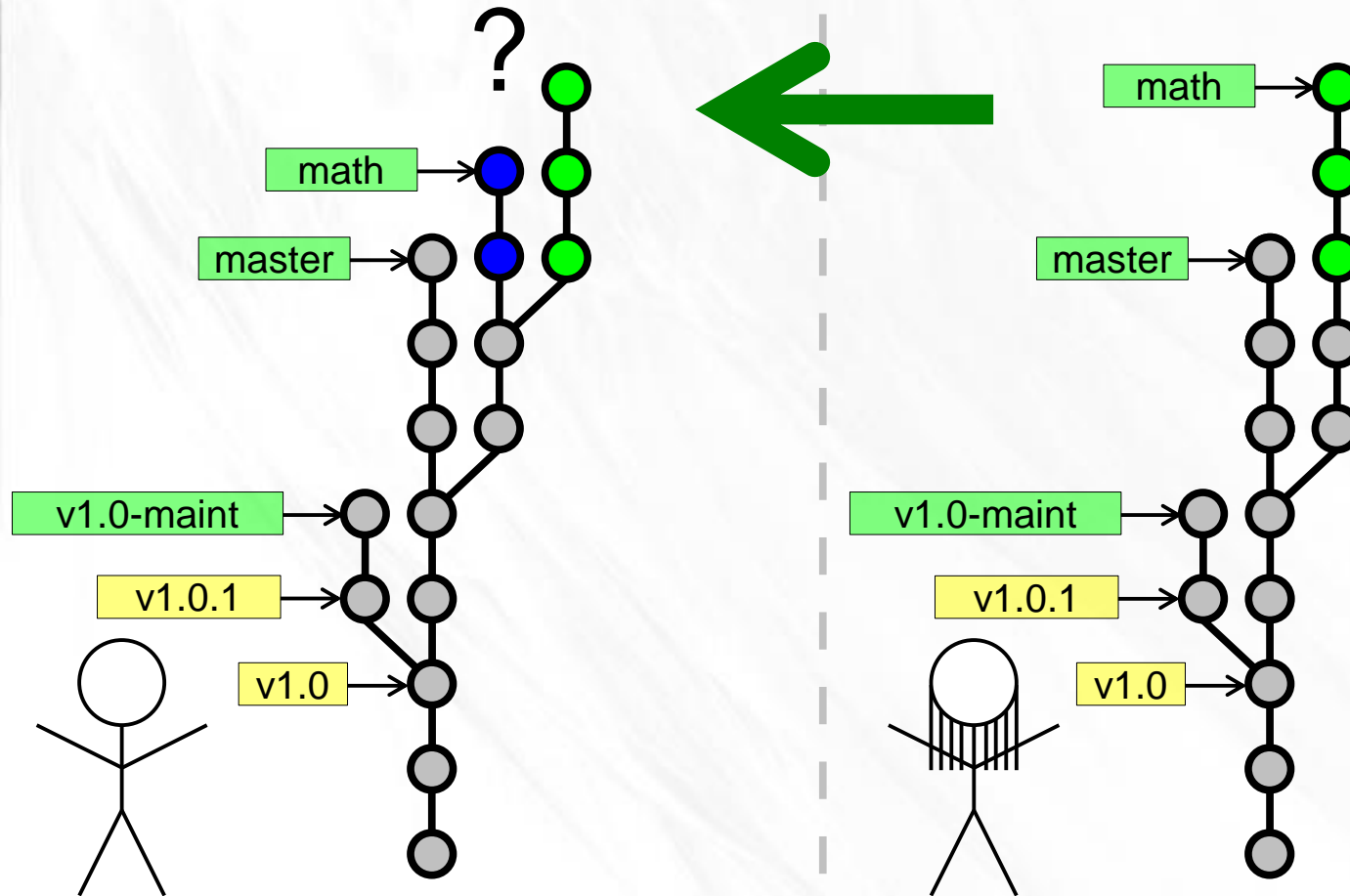
(simpler drawings)



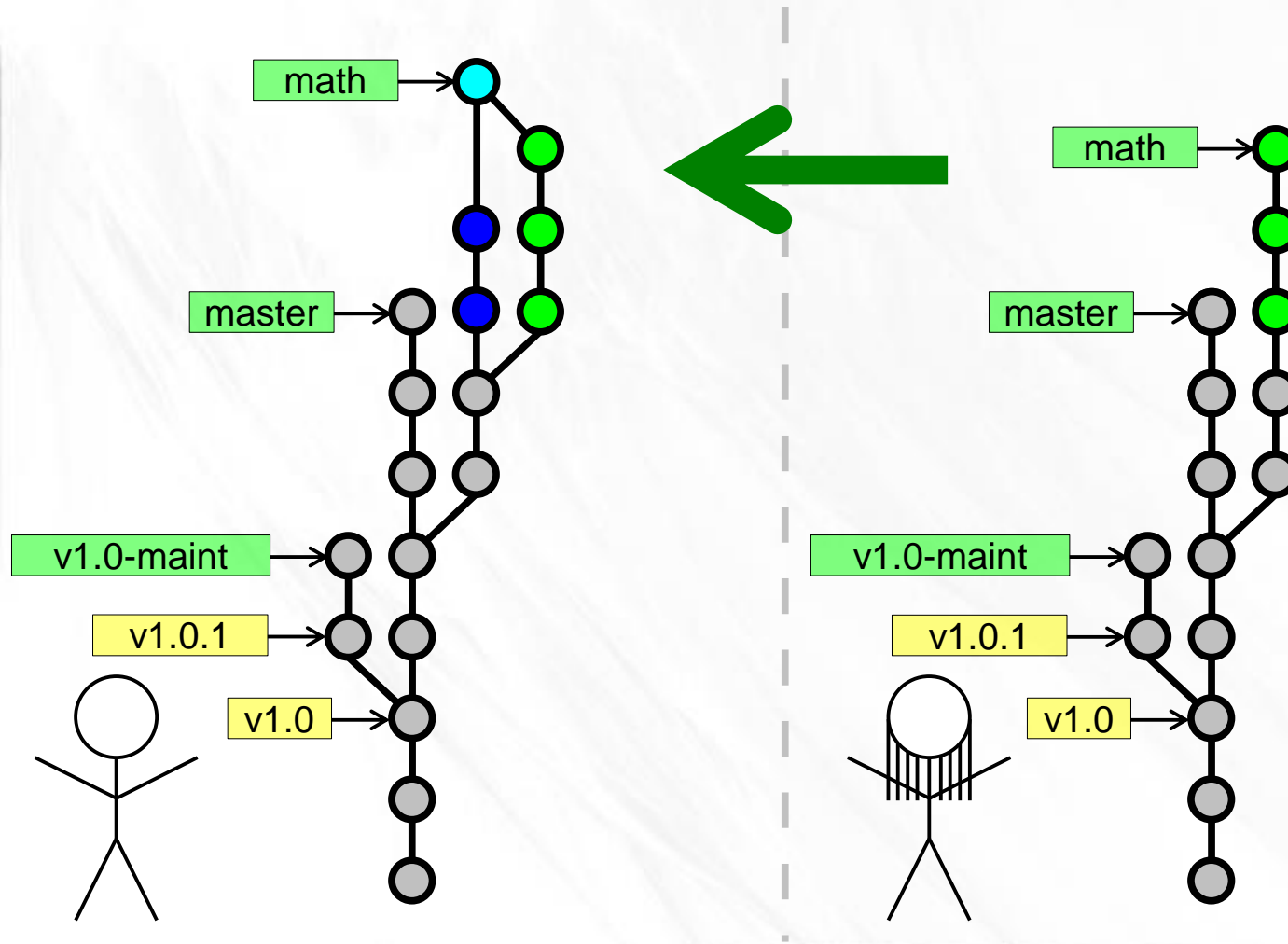
Merges



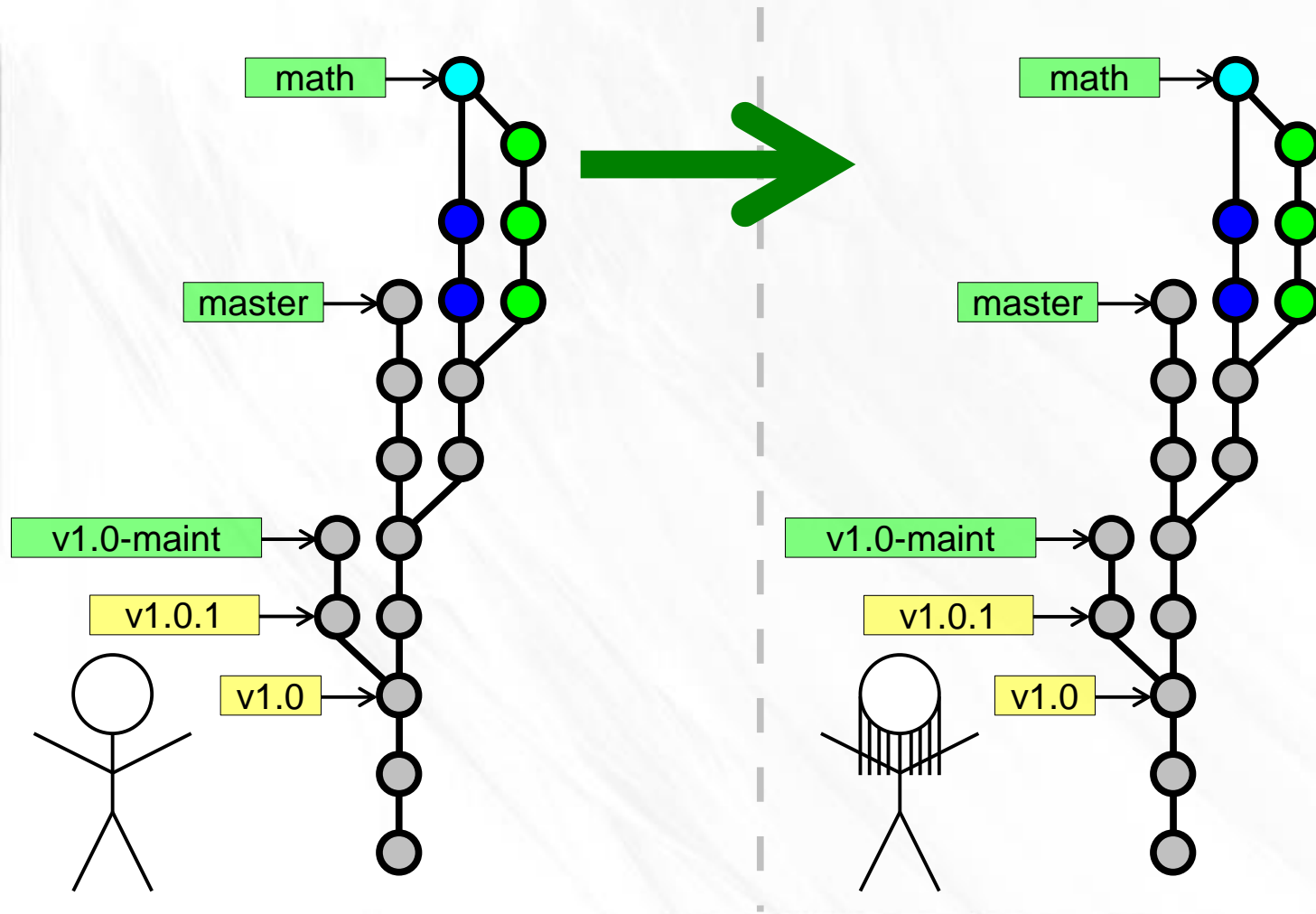
Merges



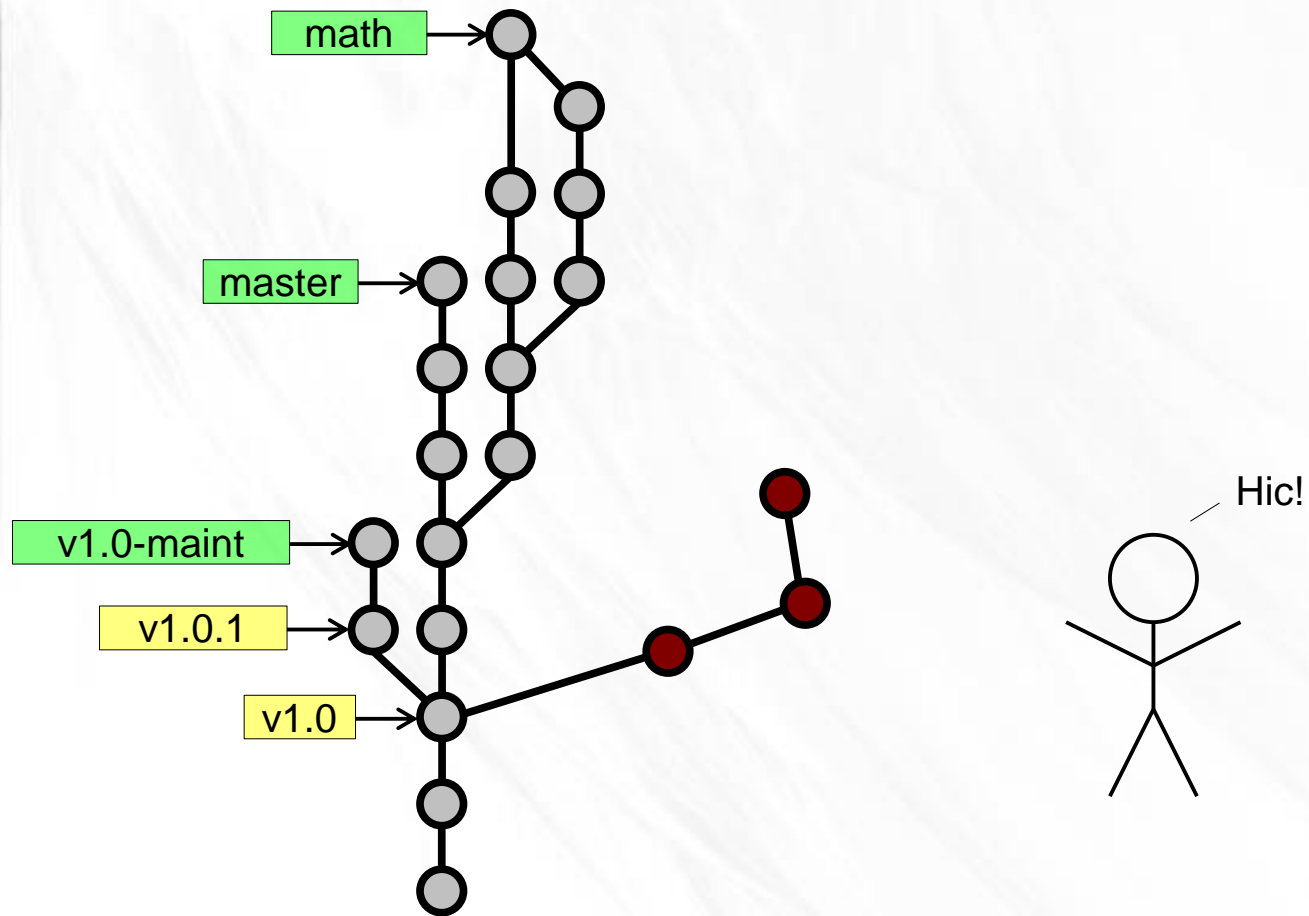
Merges



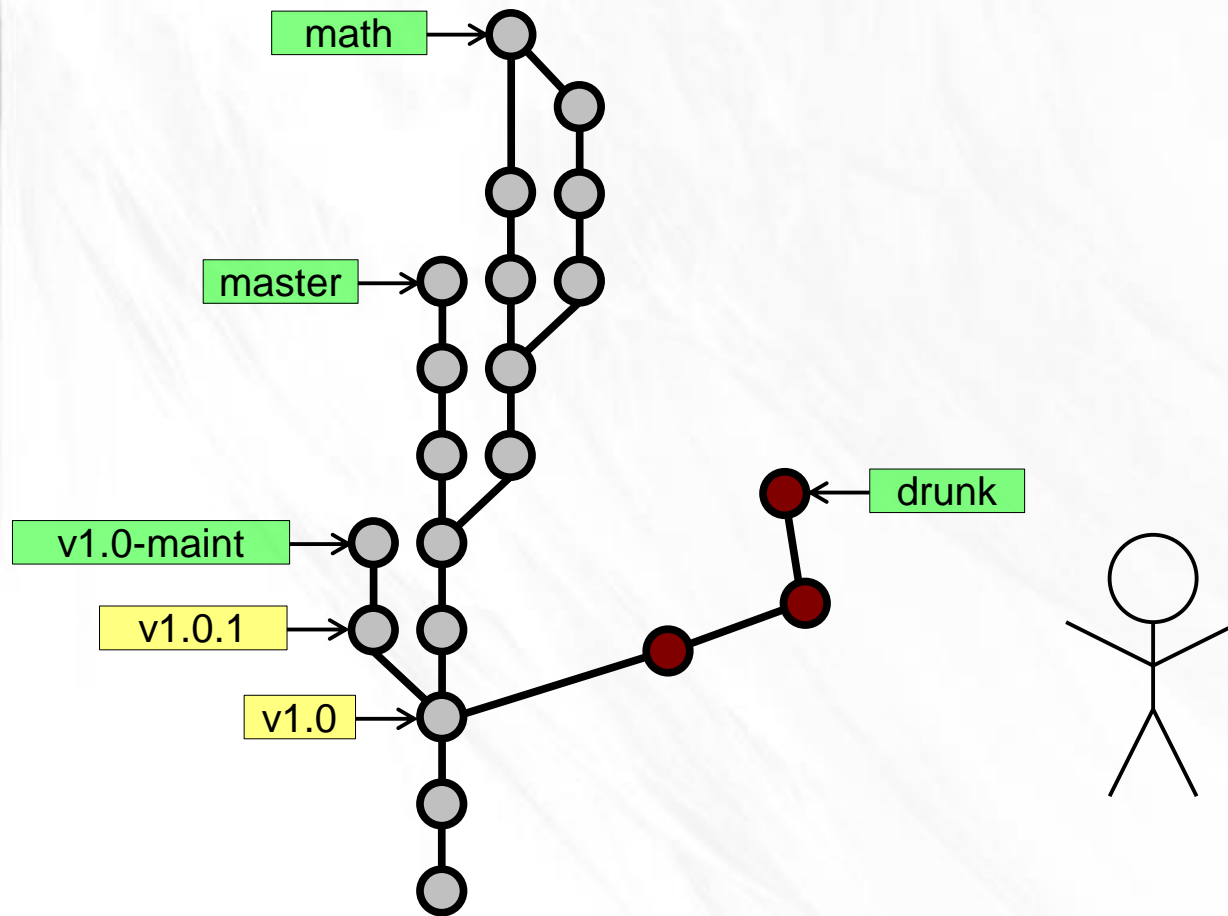
Merges



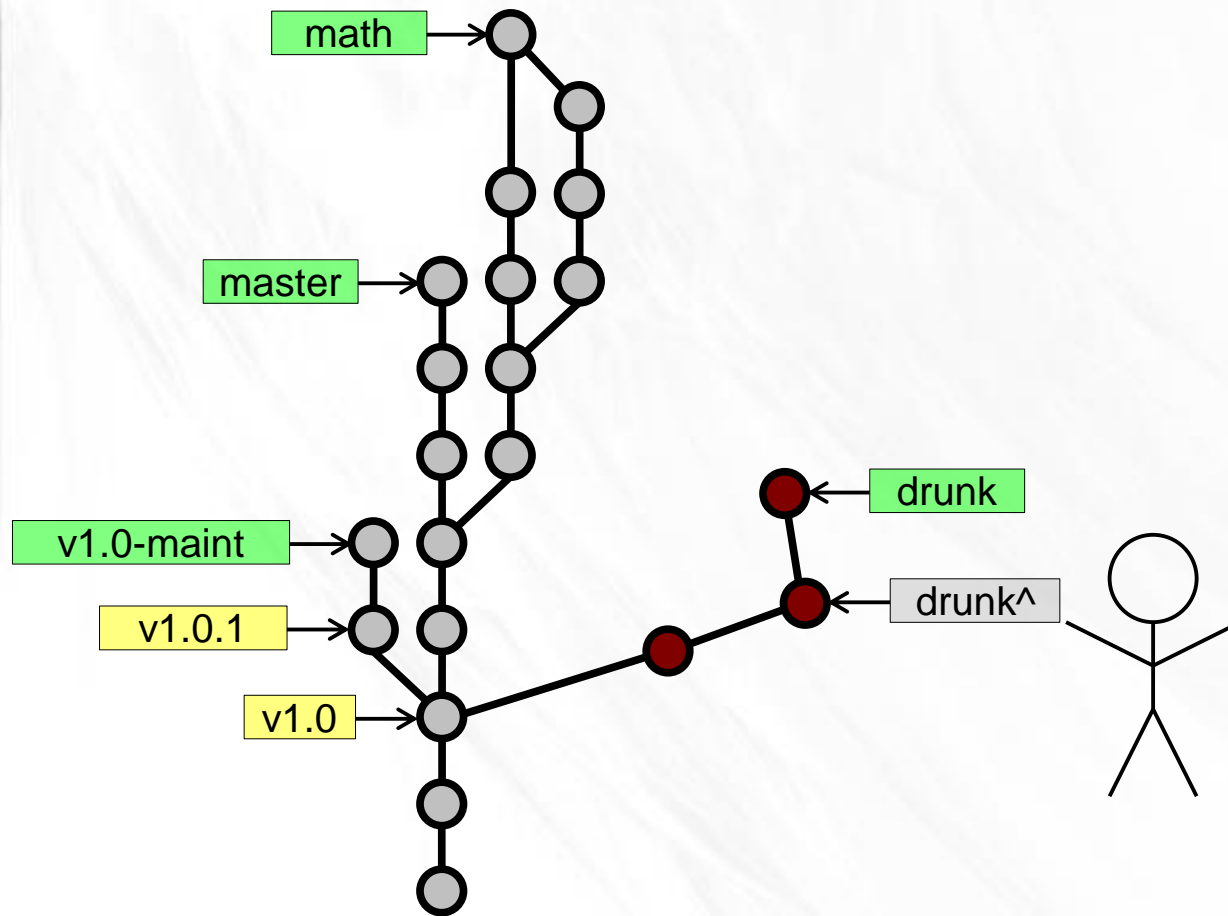
Rewriting History



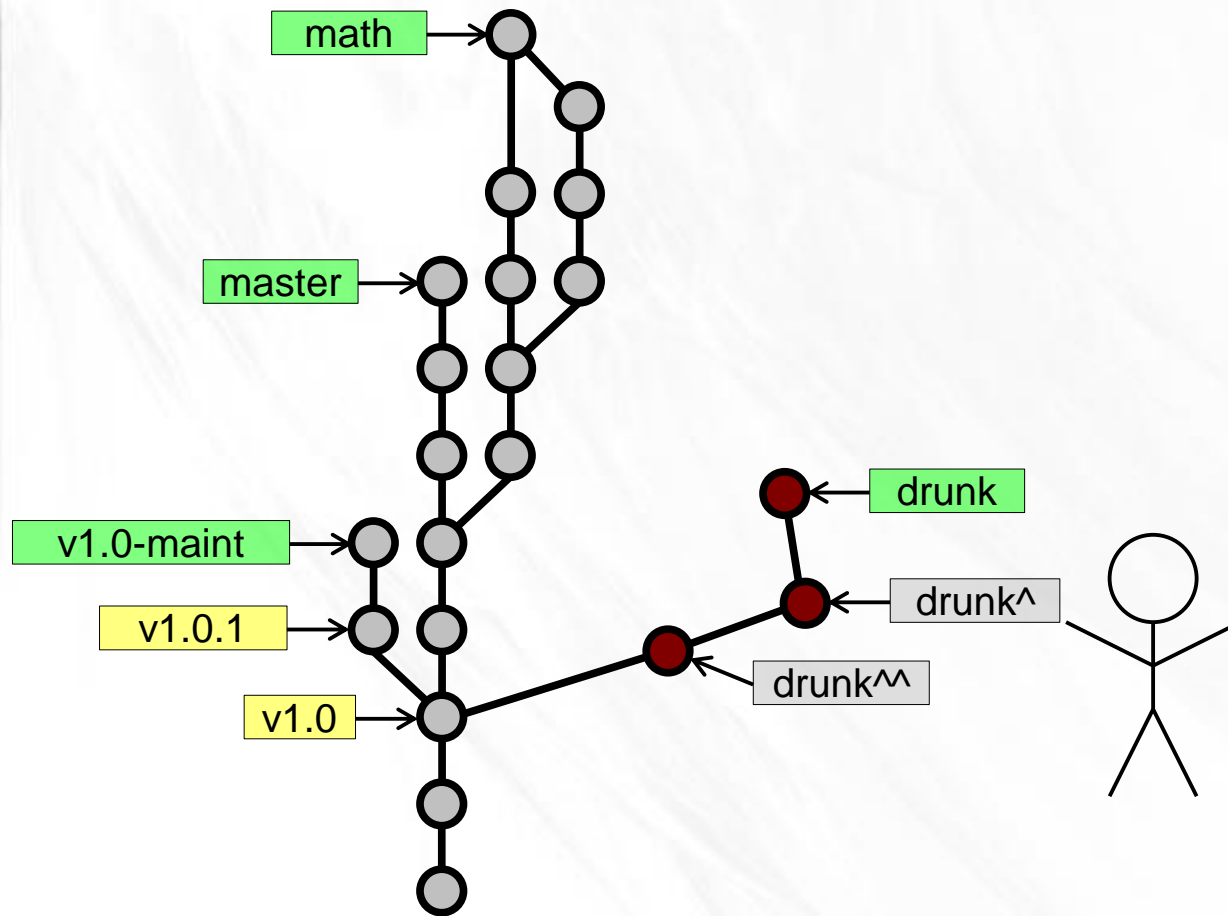
Rewriting History



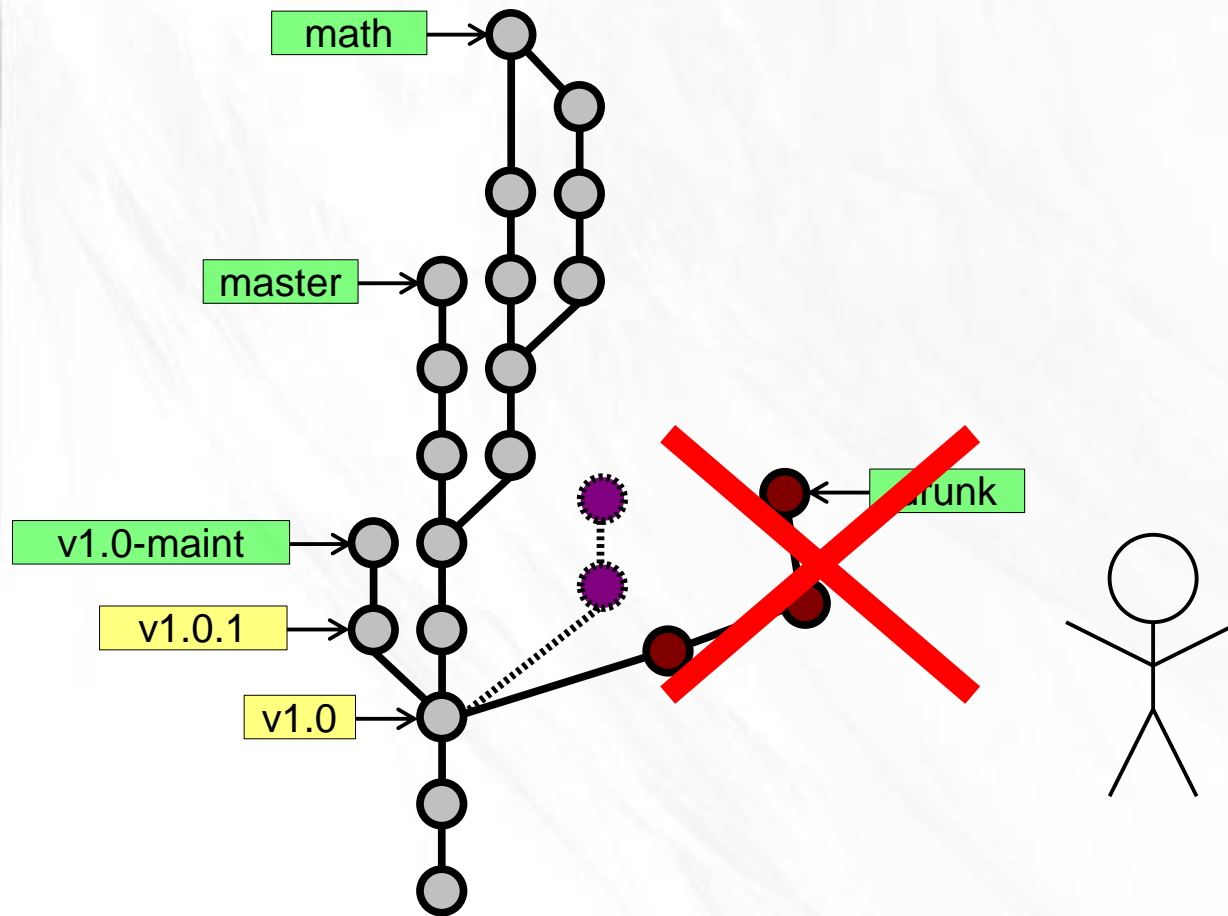
Rewriting History



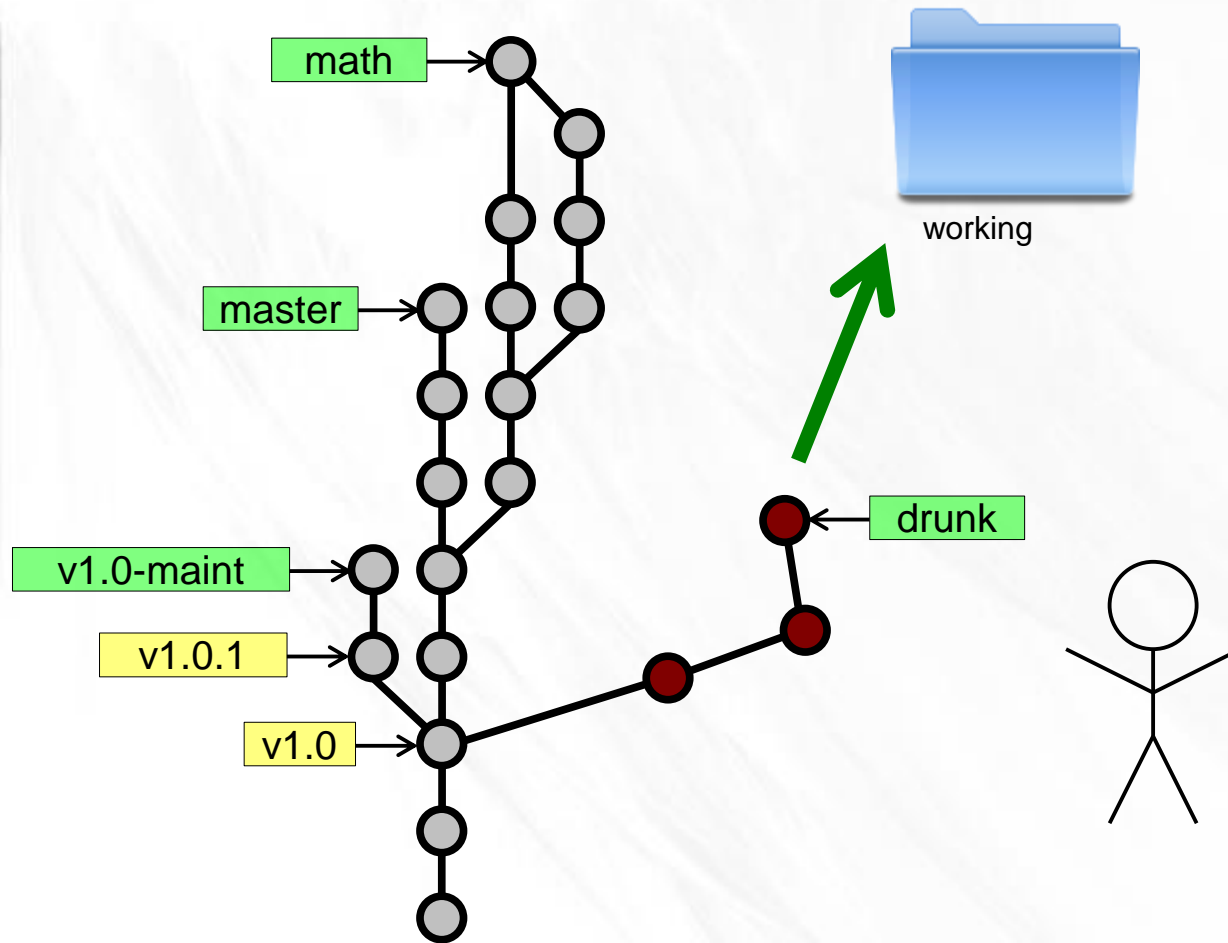
Rewriting History



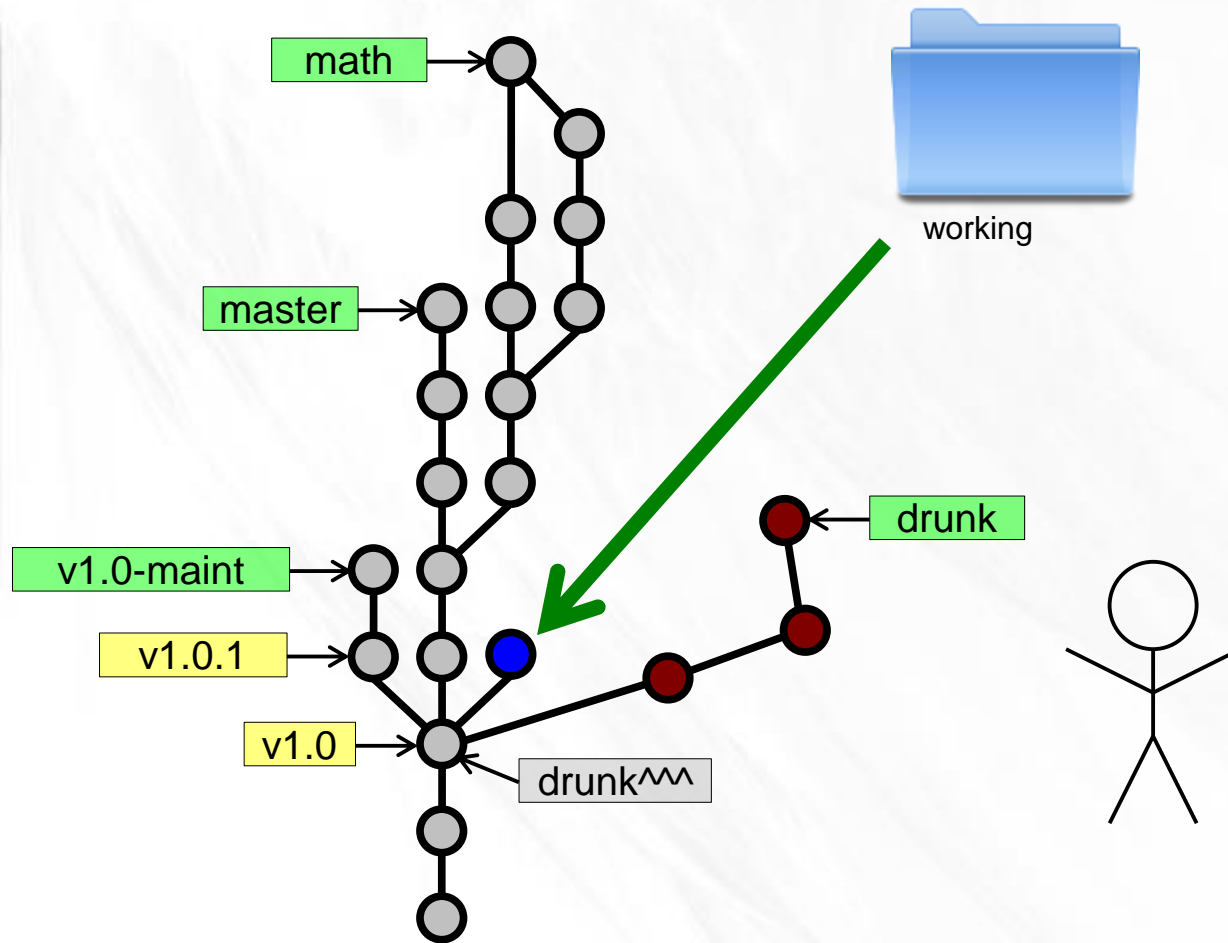
Rewriting History



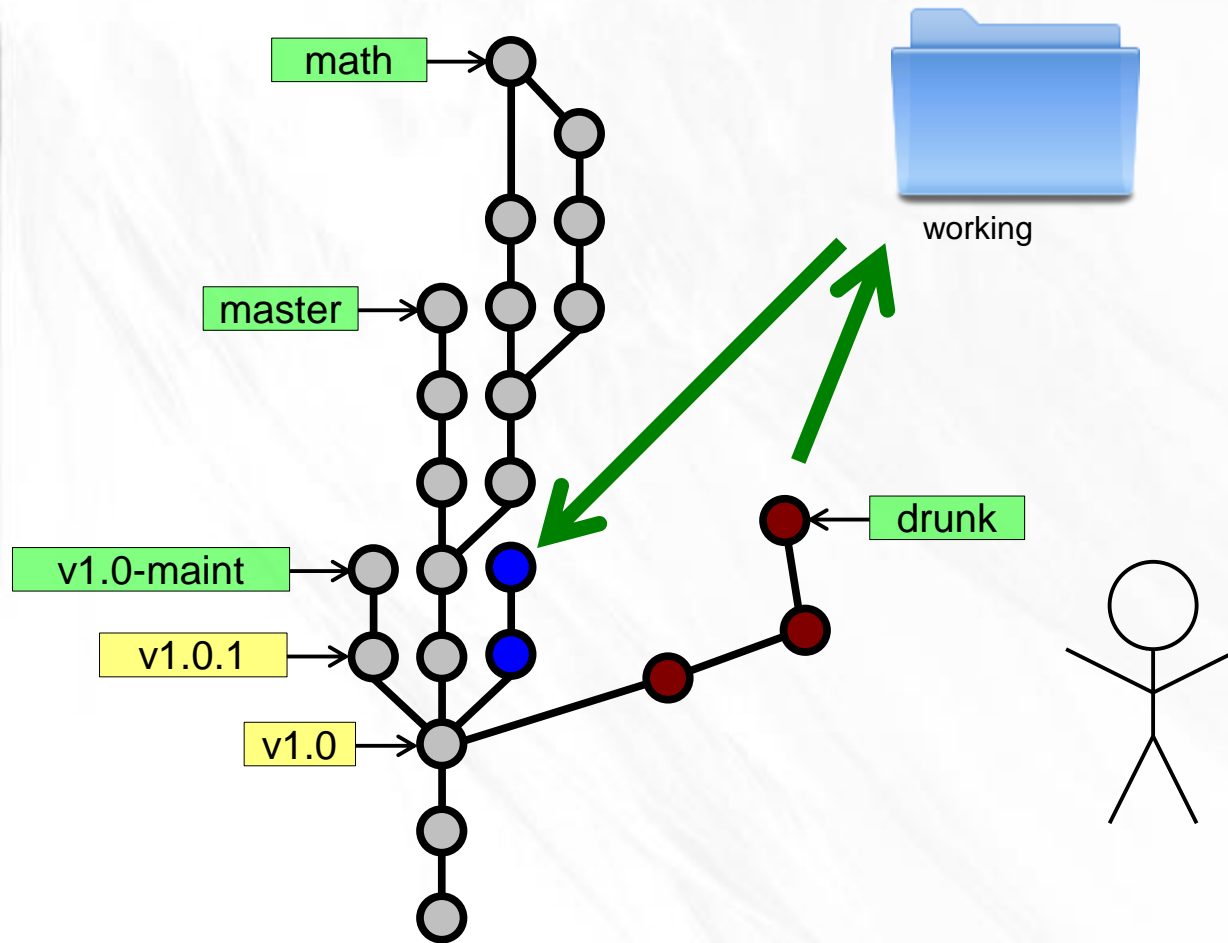
Rewriting History



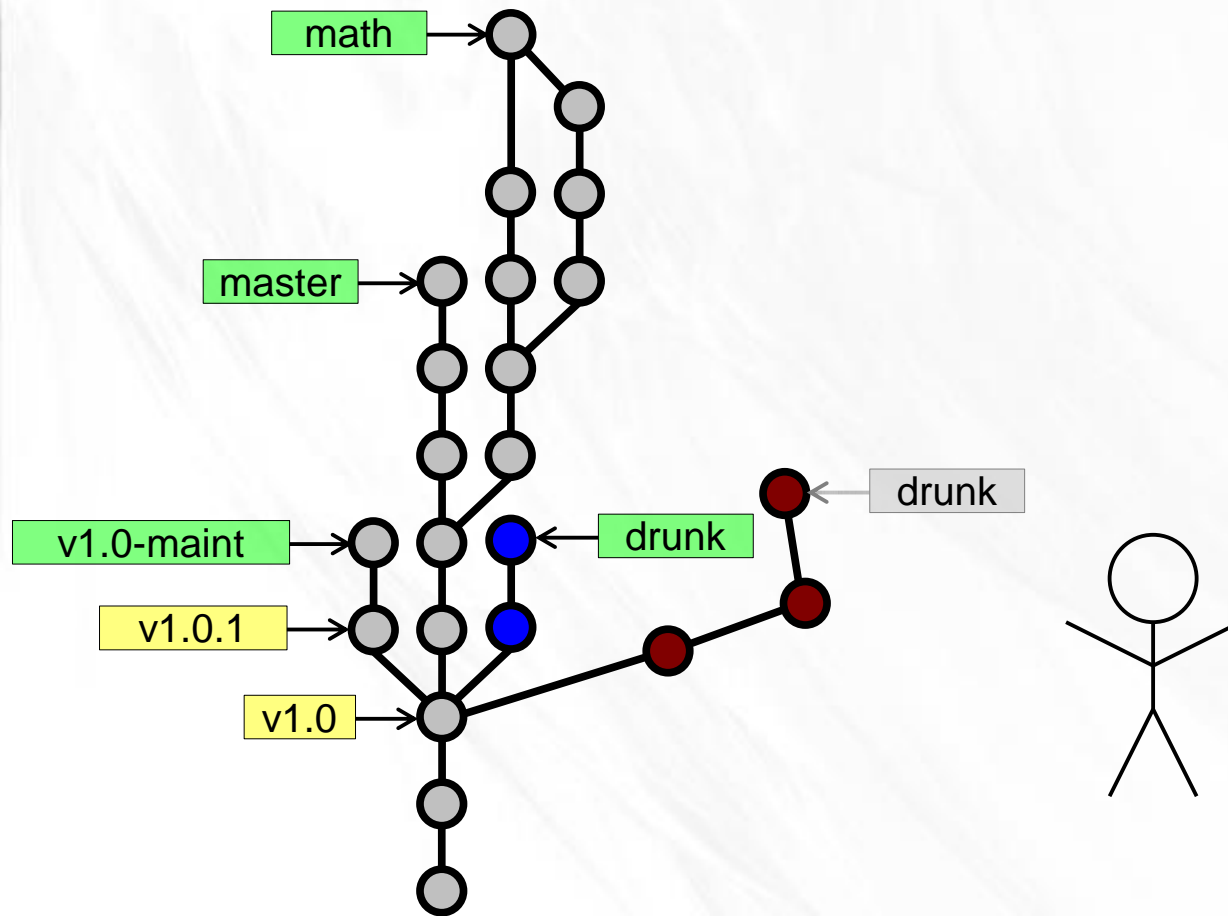
Rewriting History



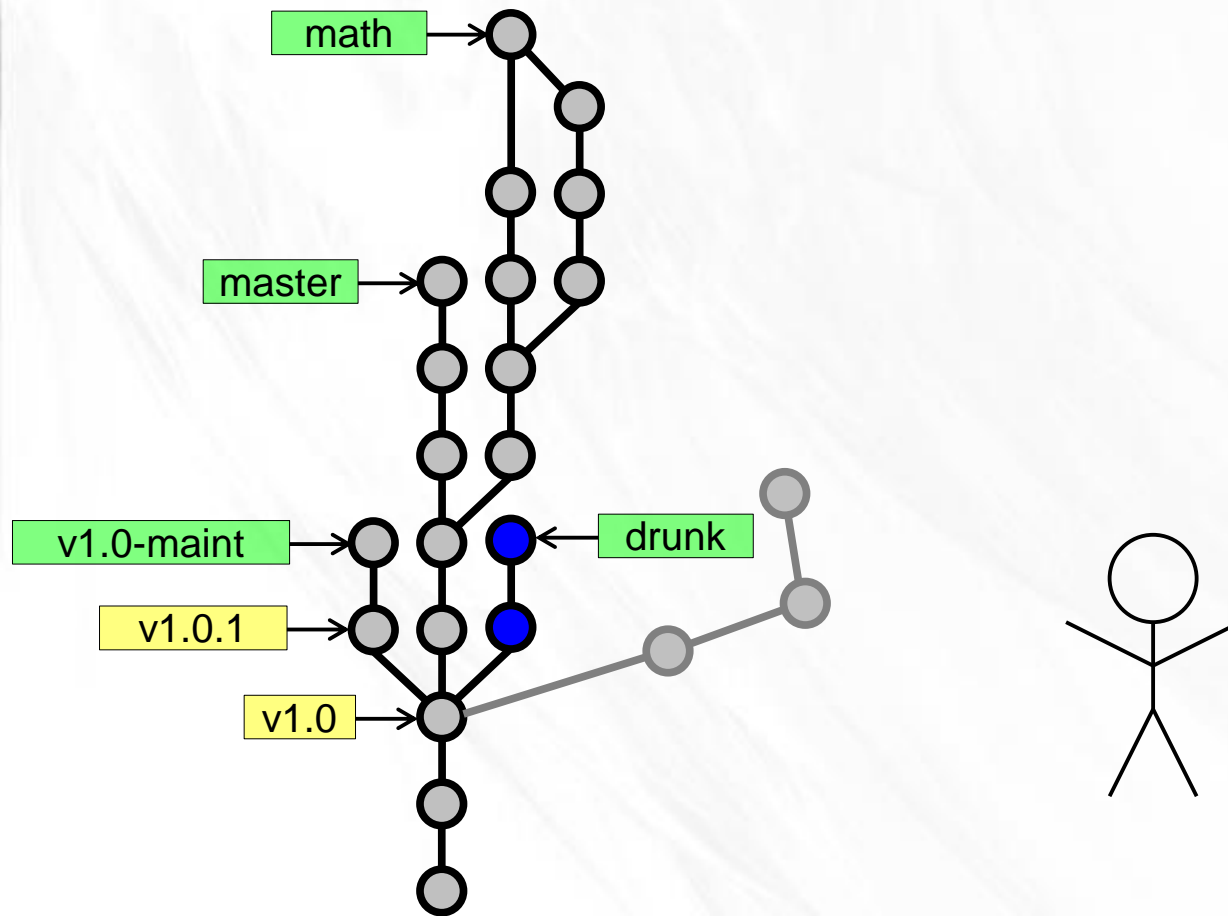
Rewriting History



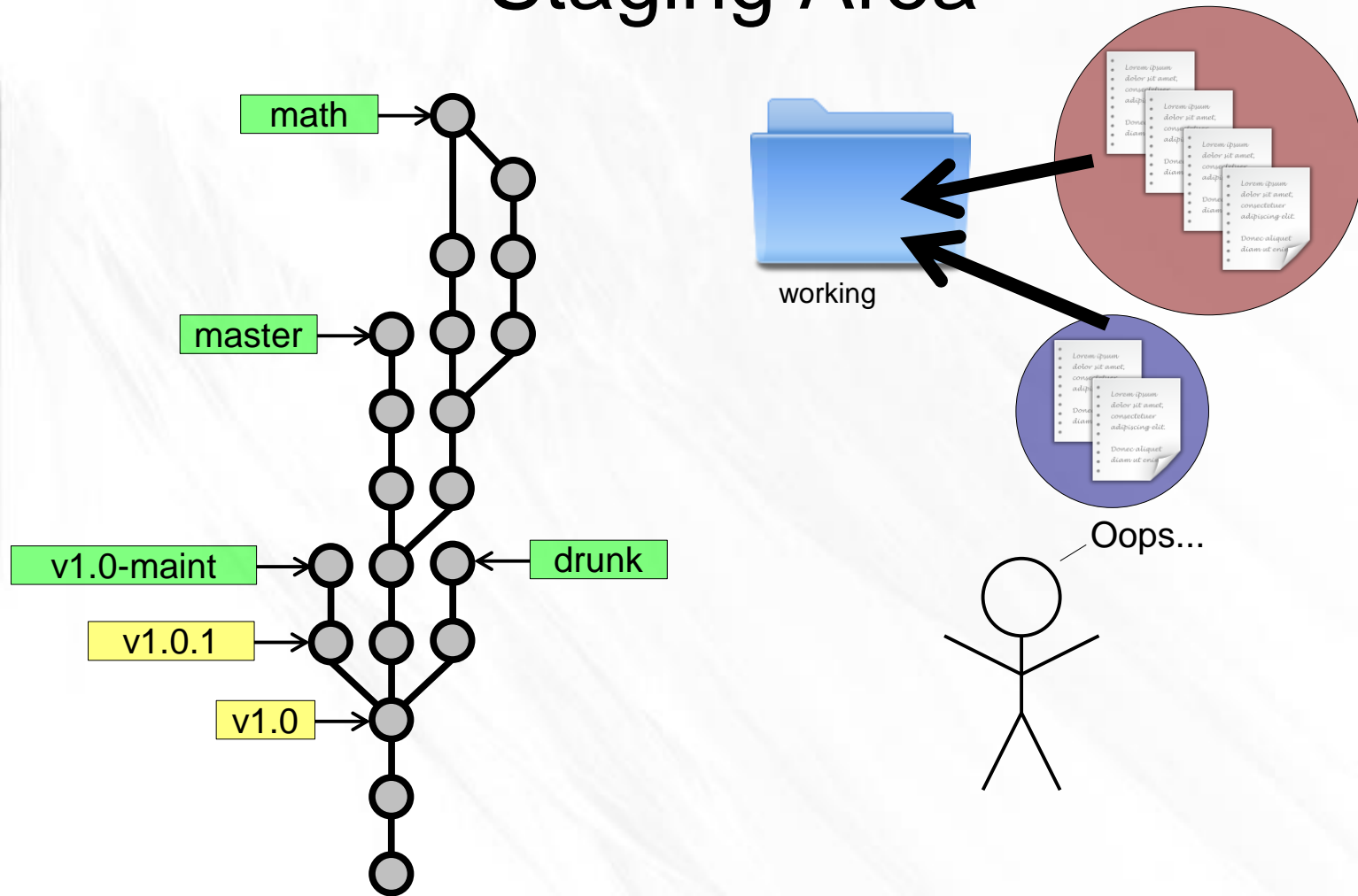
Rewriting History



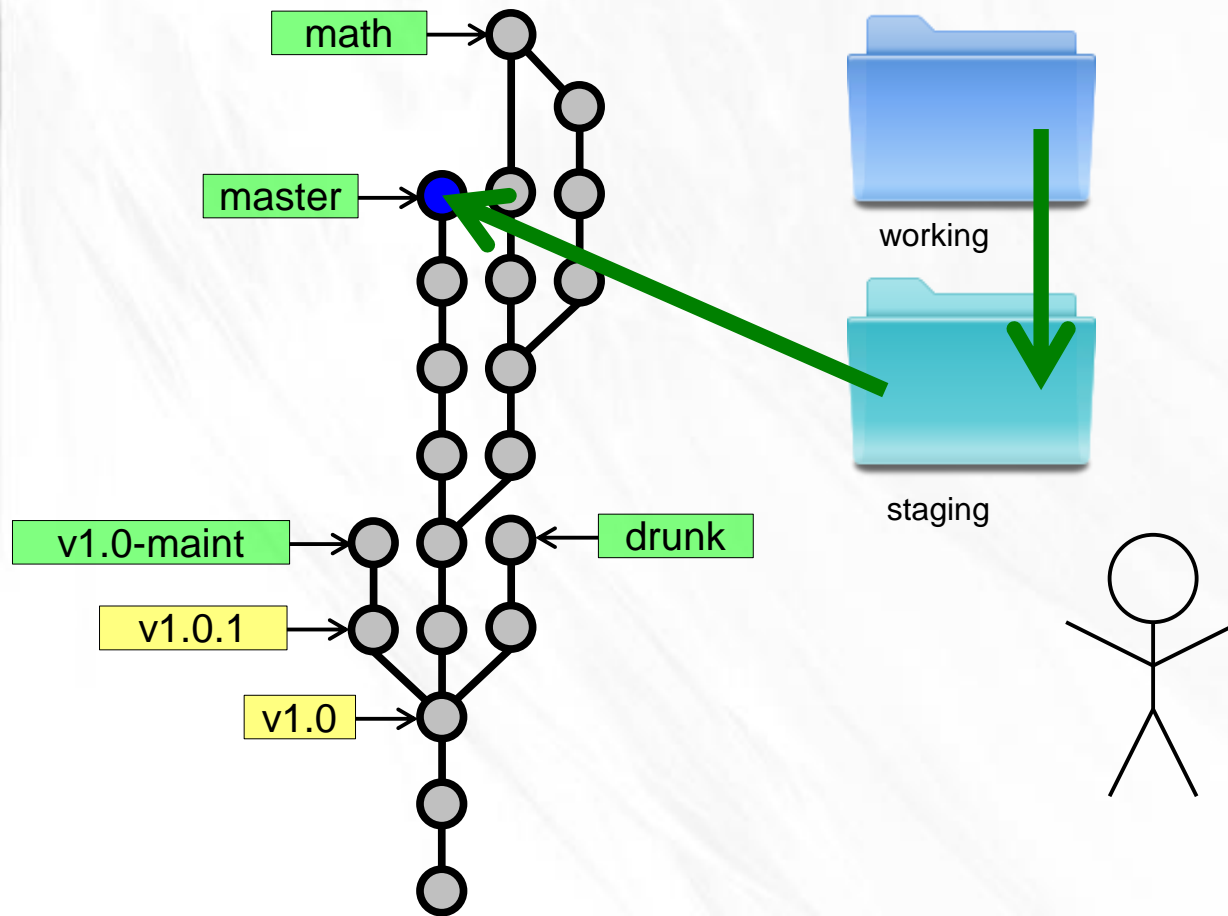
Rewriting History



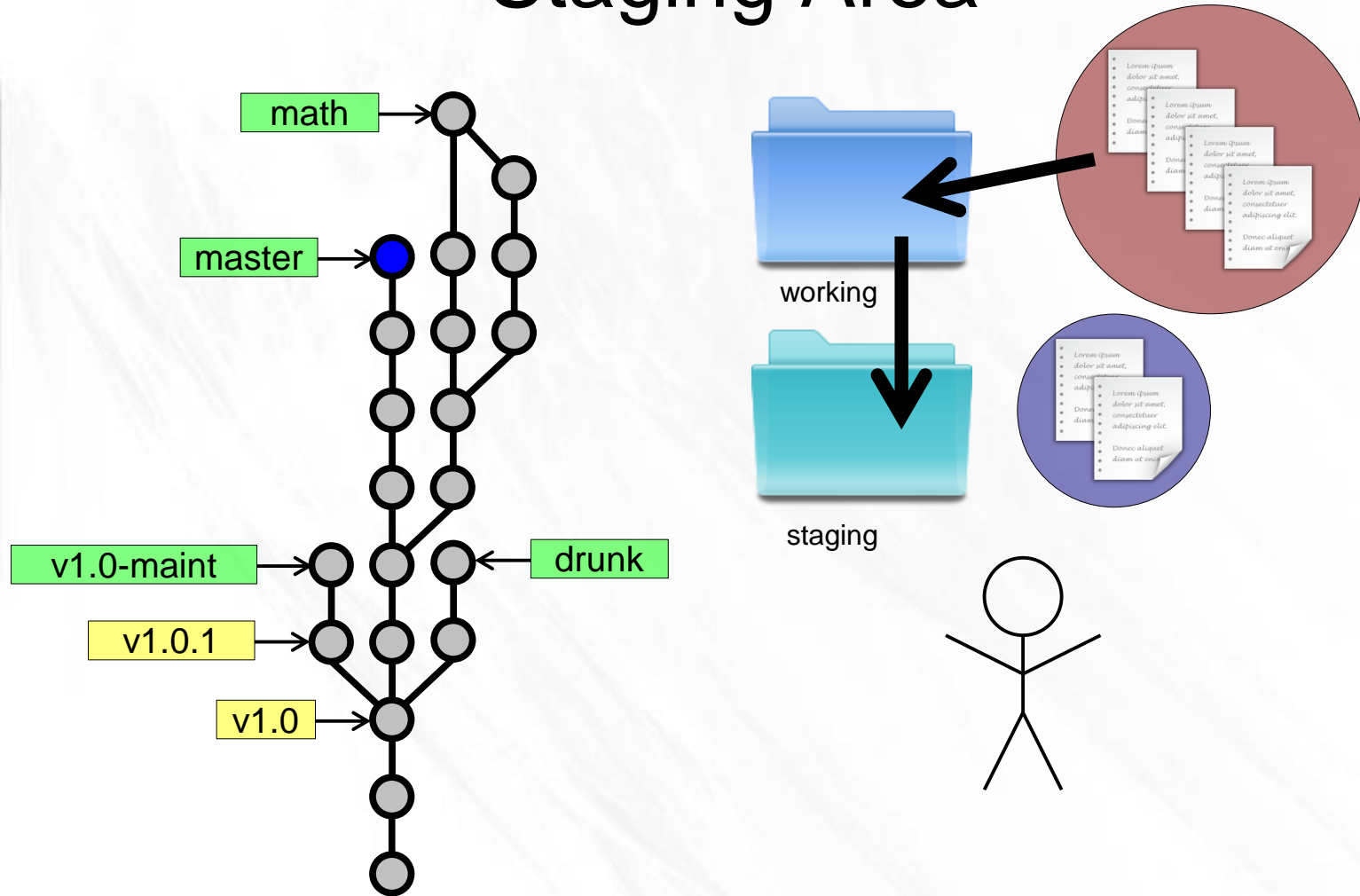
Staging Area



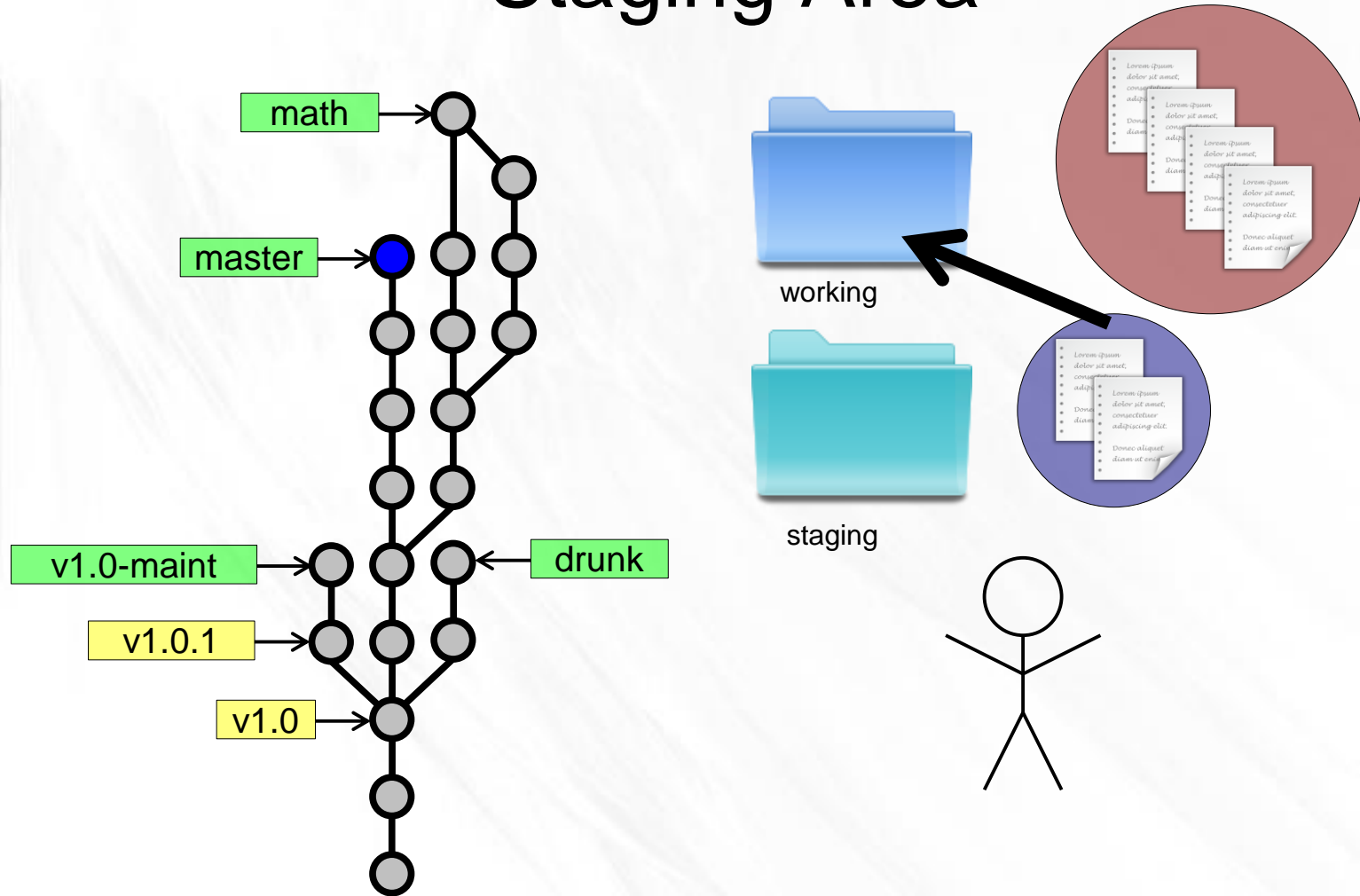
Staging Area



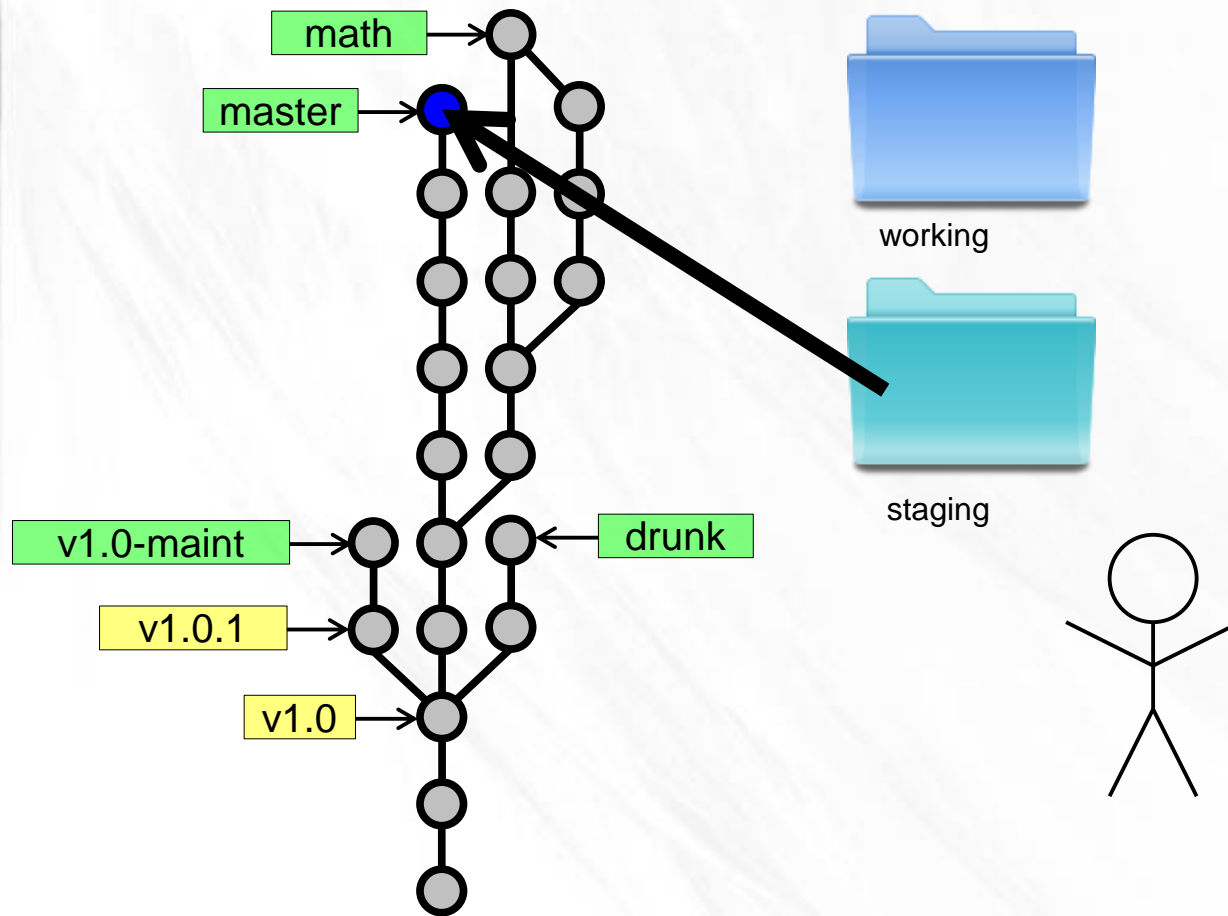
Staging Area



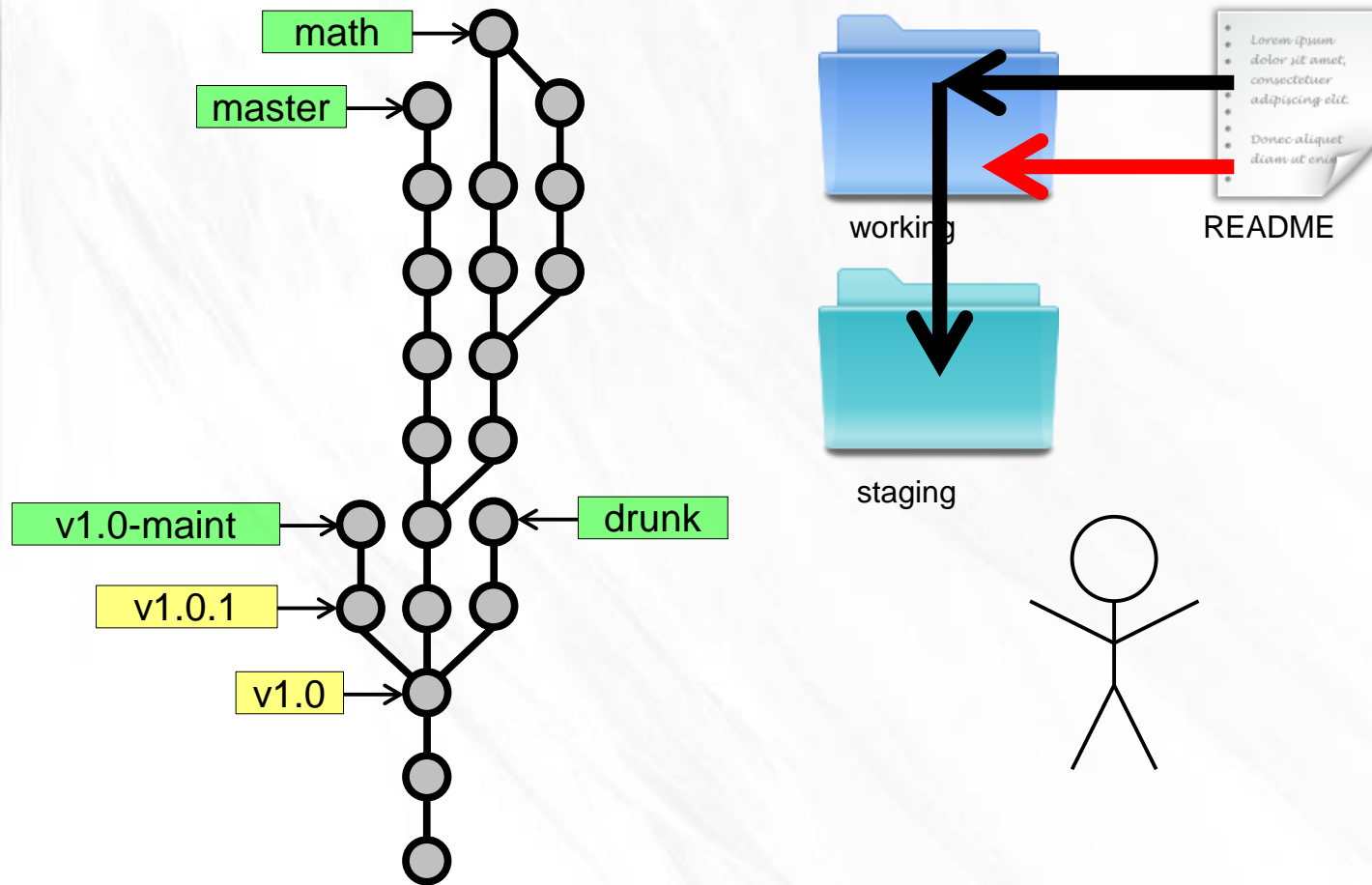
Staging Area



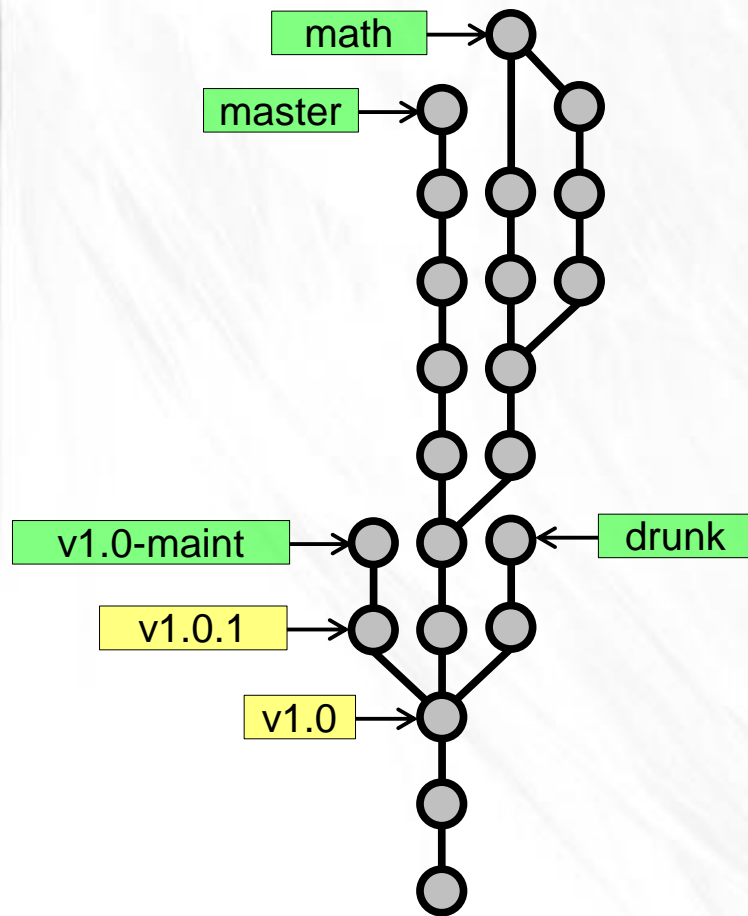
Staging Area



Staging Area



Diffs



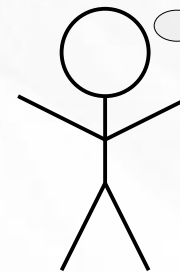
working



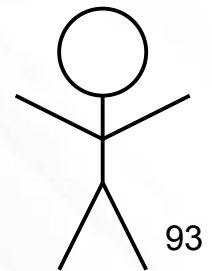
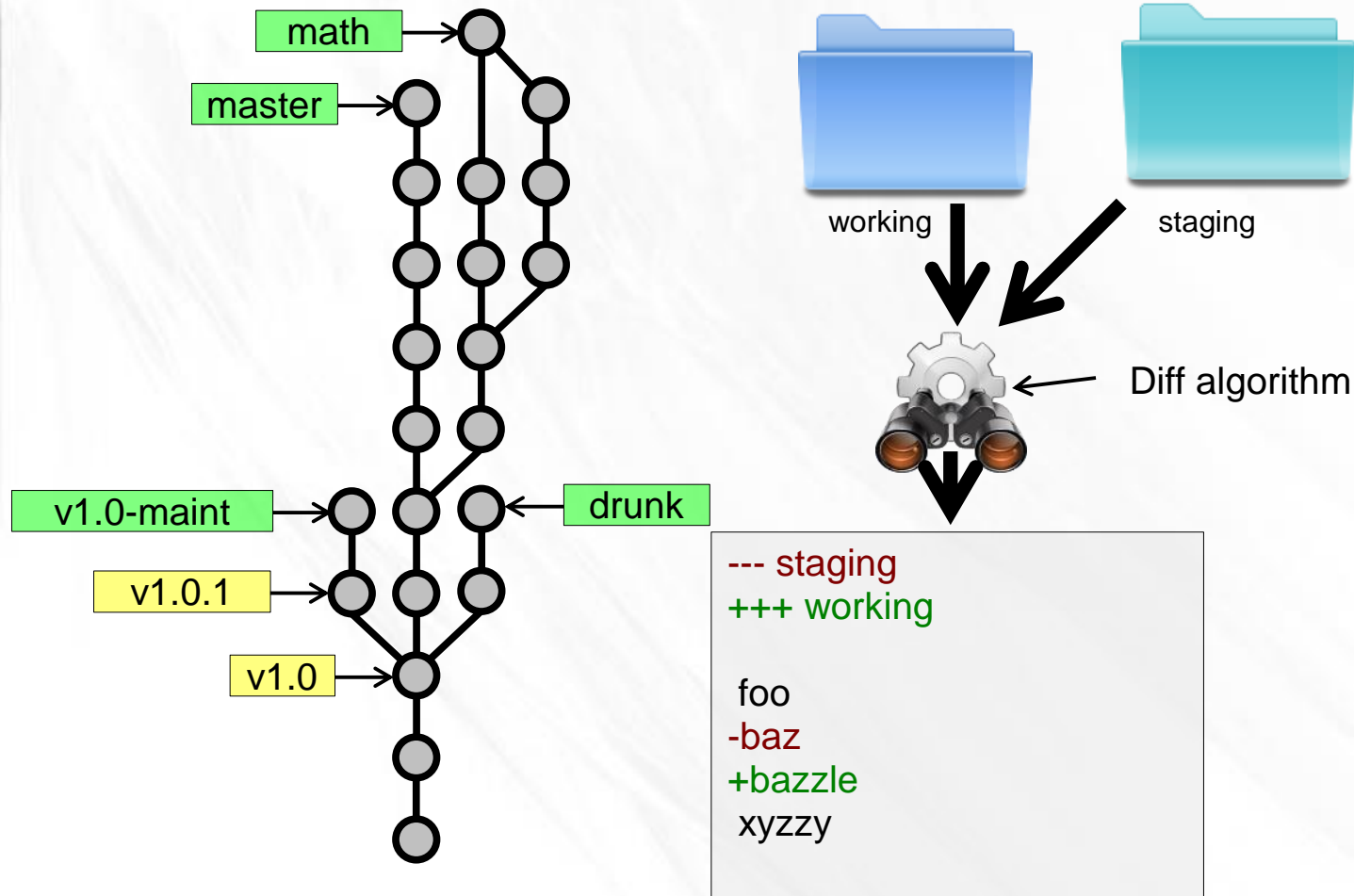
staging

What are the changes?

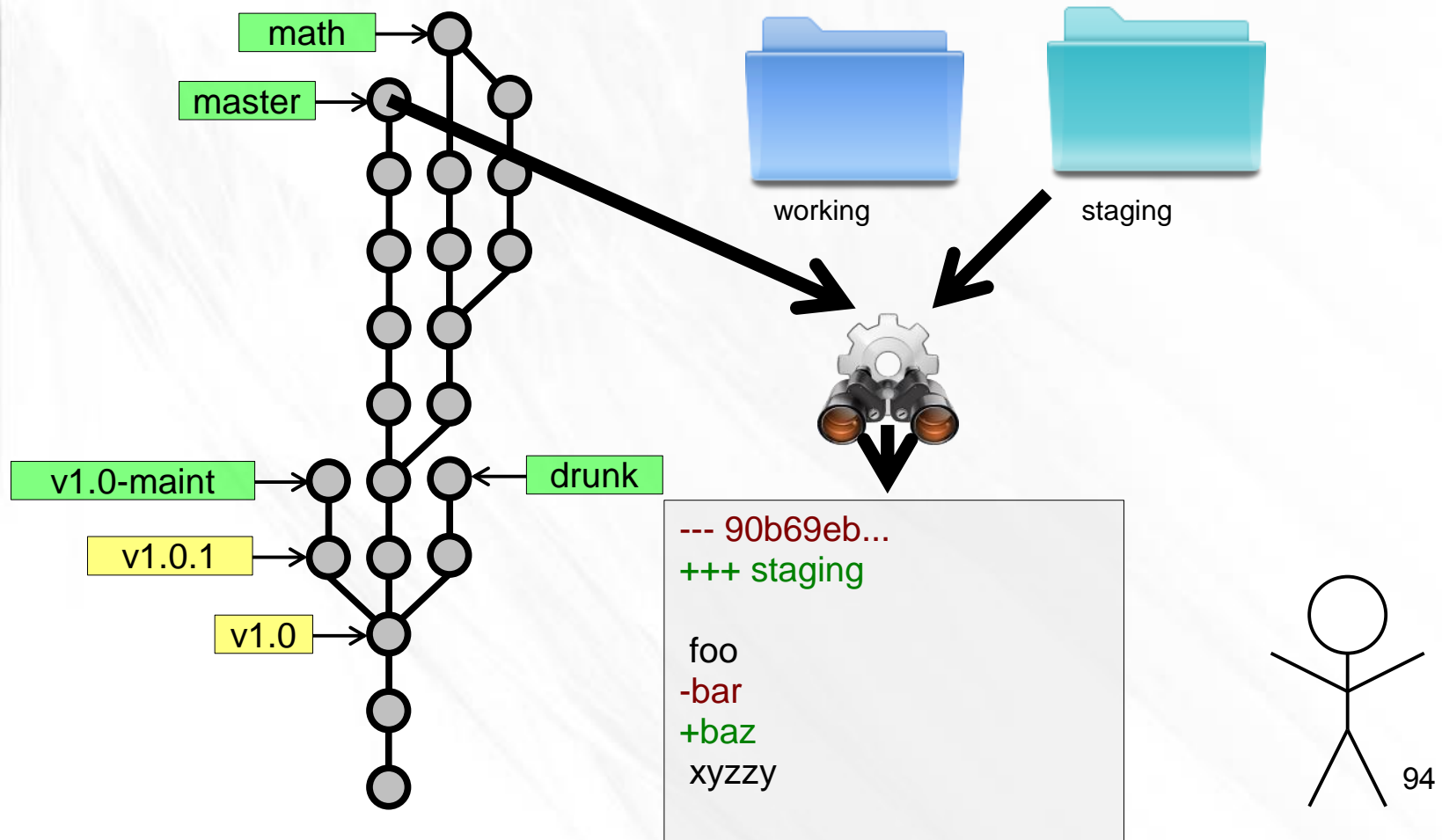
- working vs. staging
- working vs. snapshot X
- staging vs. snapshot X
- snapshot X vs. snapshot Y



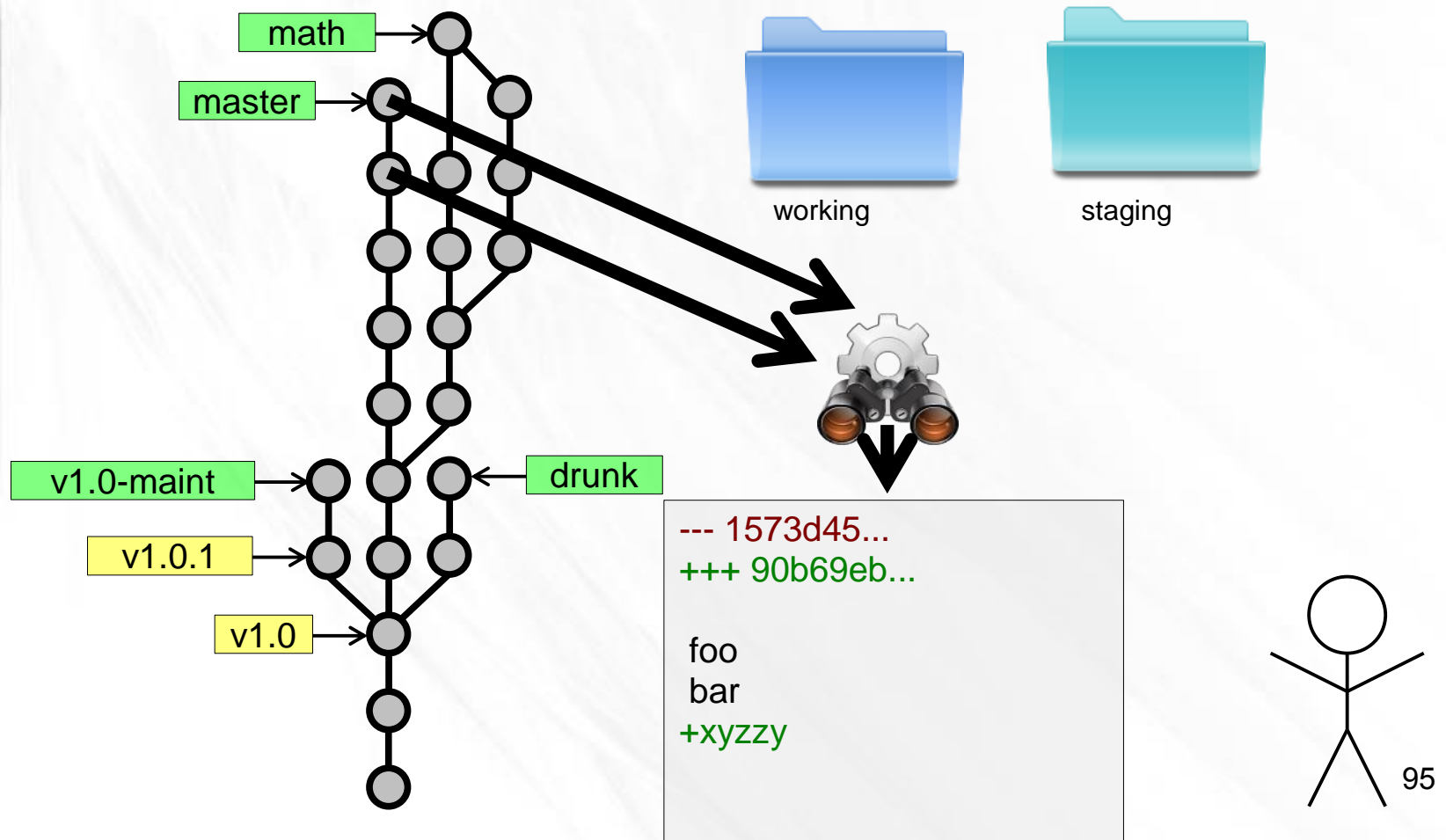
Diffs



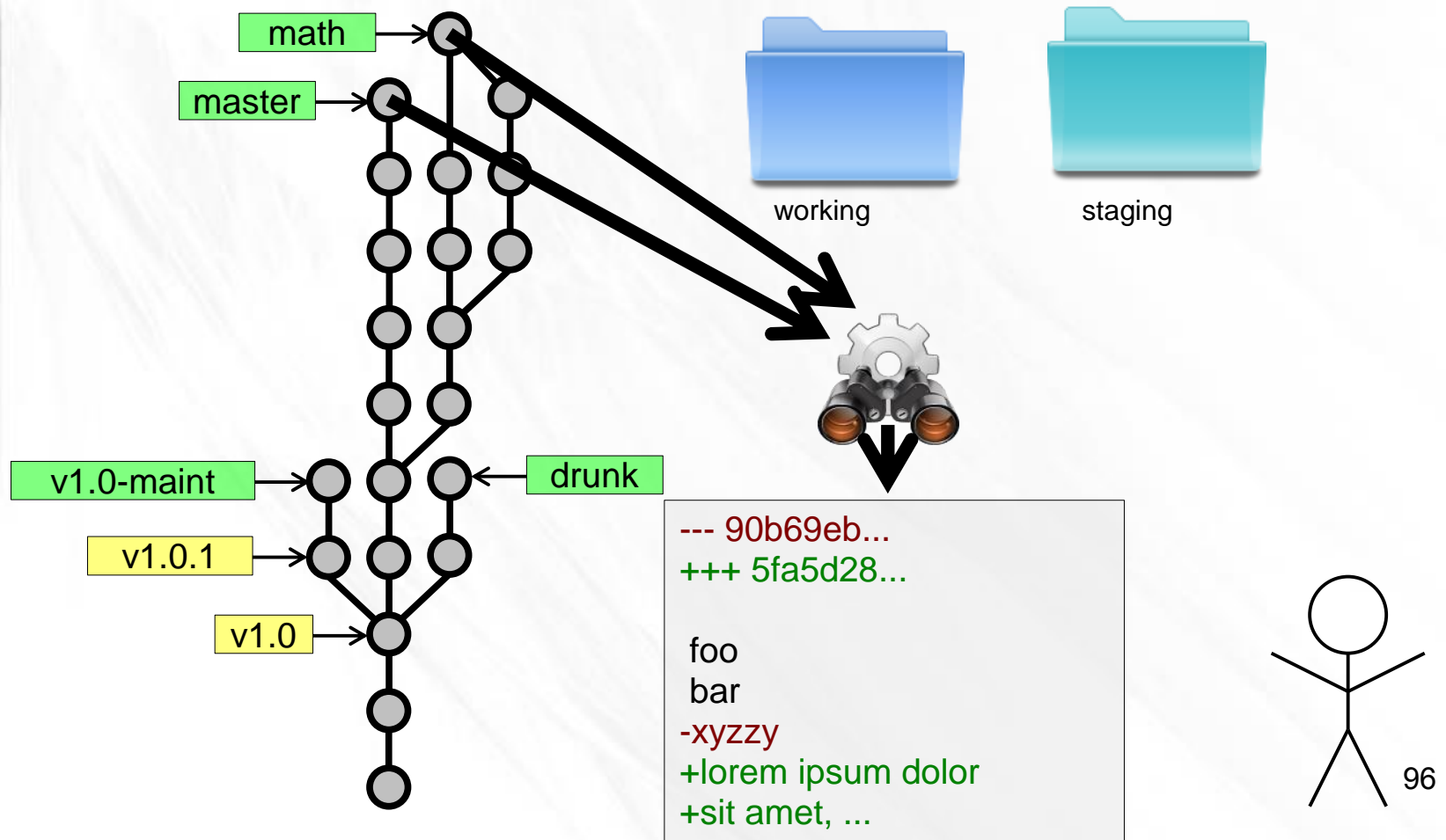
Diffs



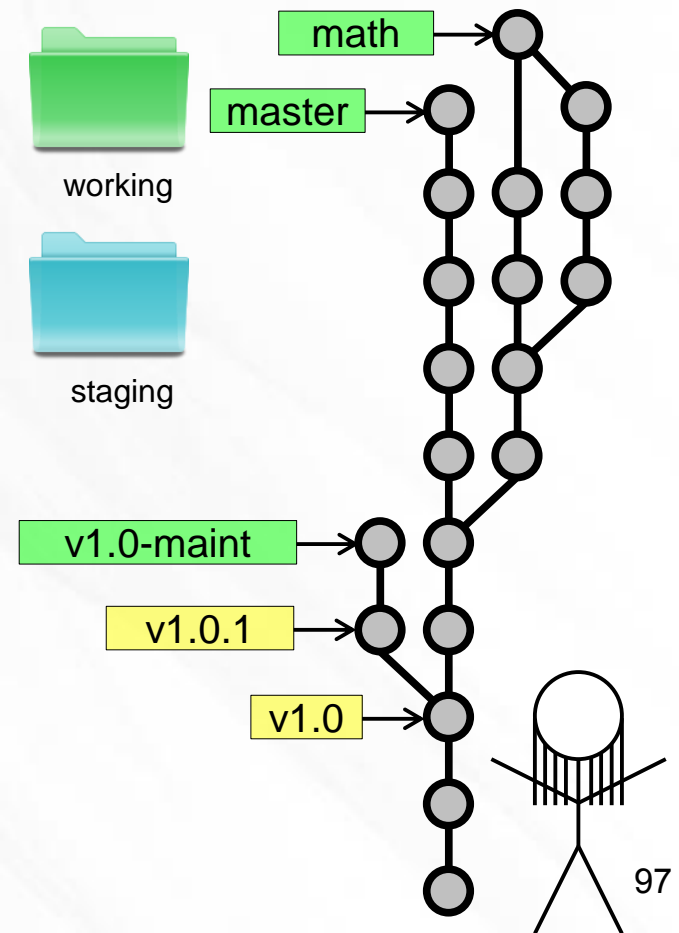
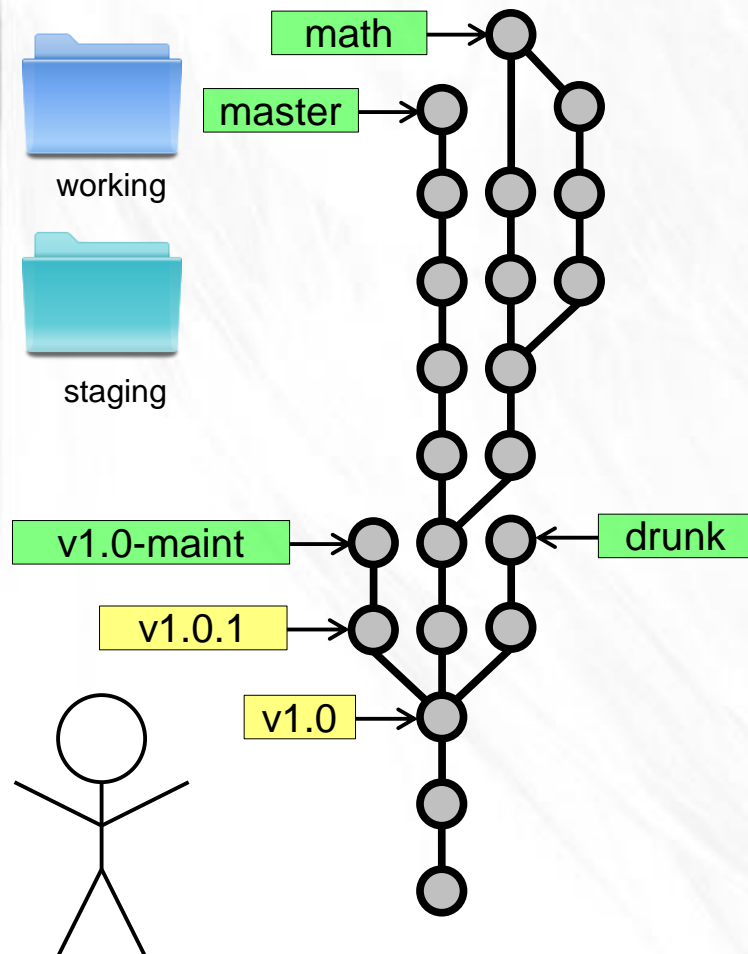
Diffs



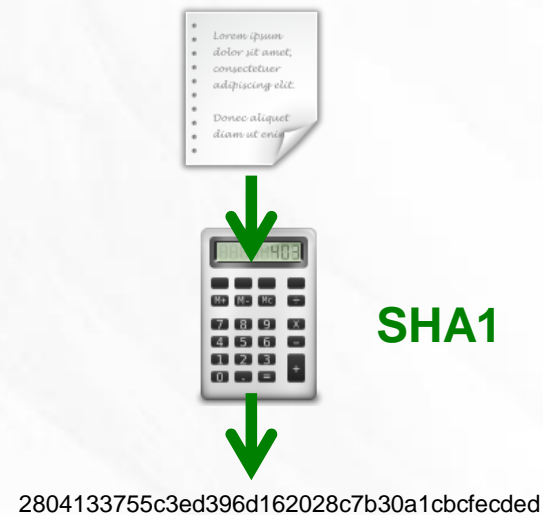
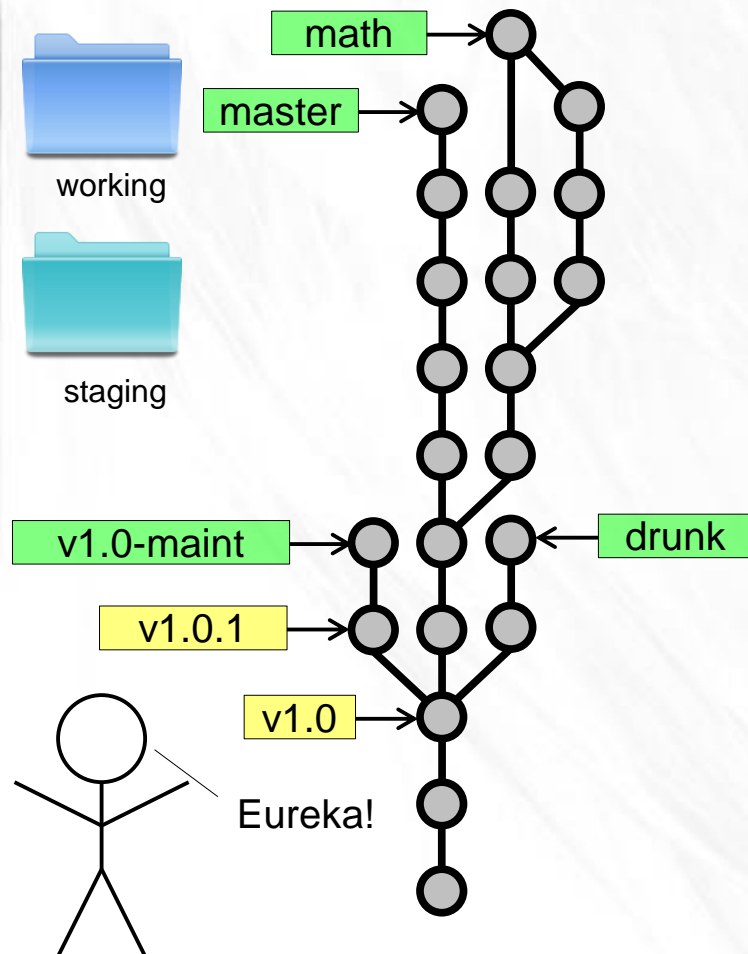
Diffs



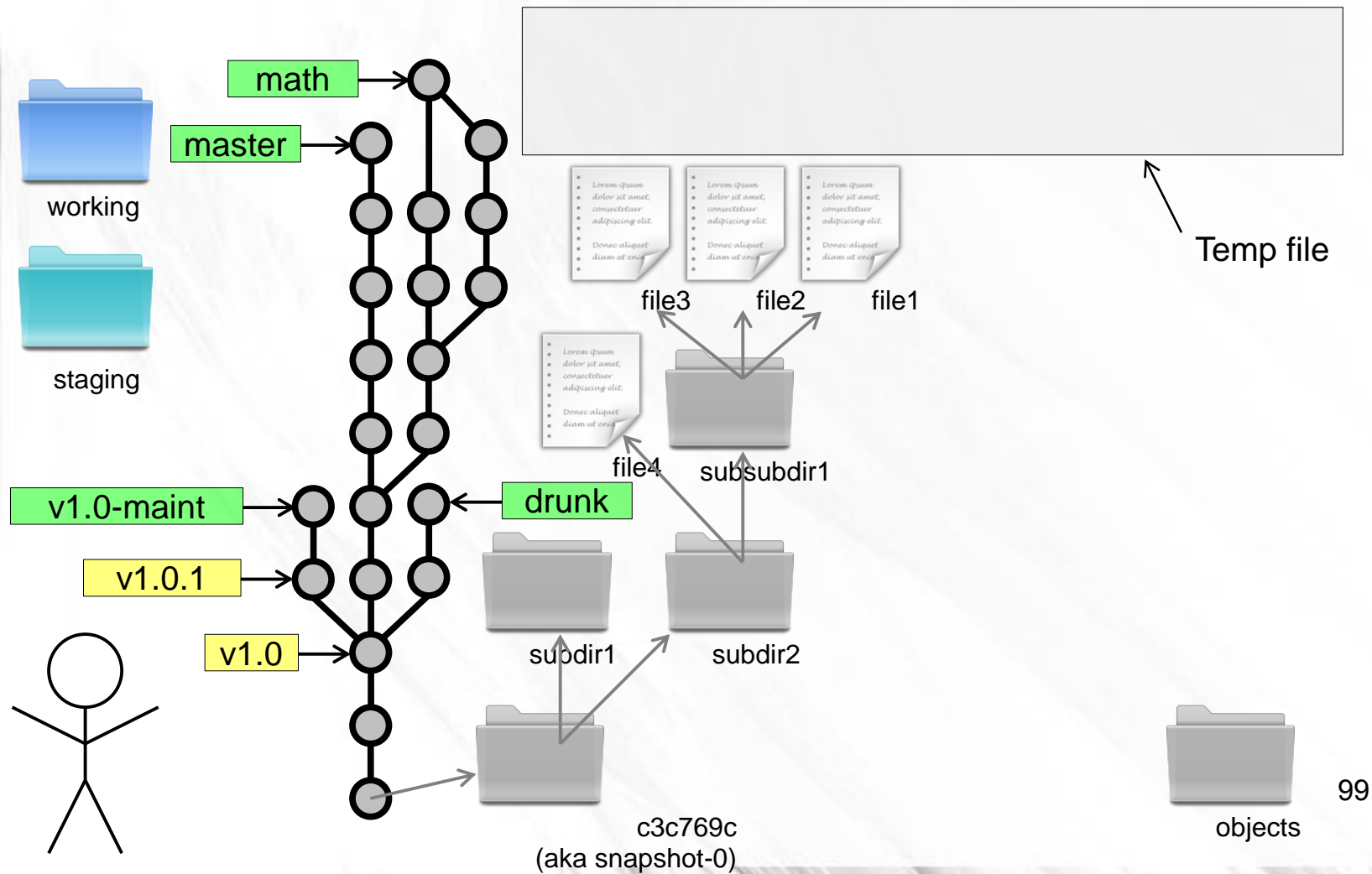
Eliminating Duplication



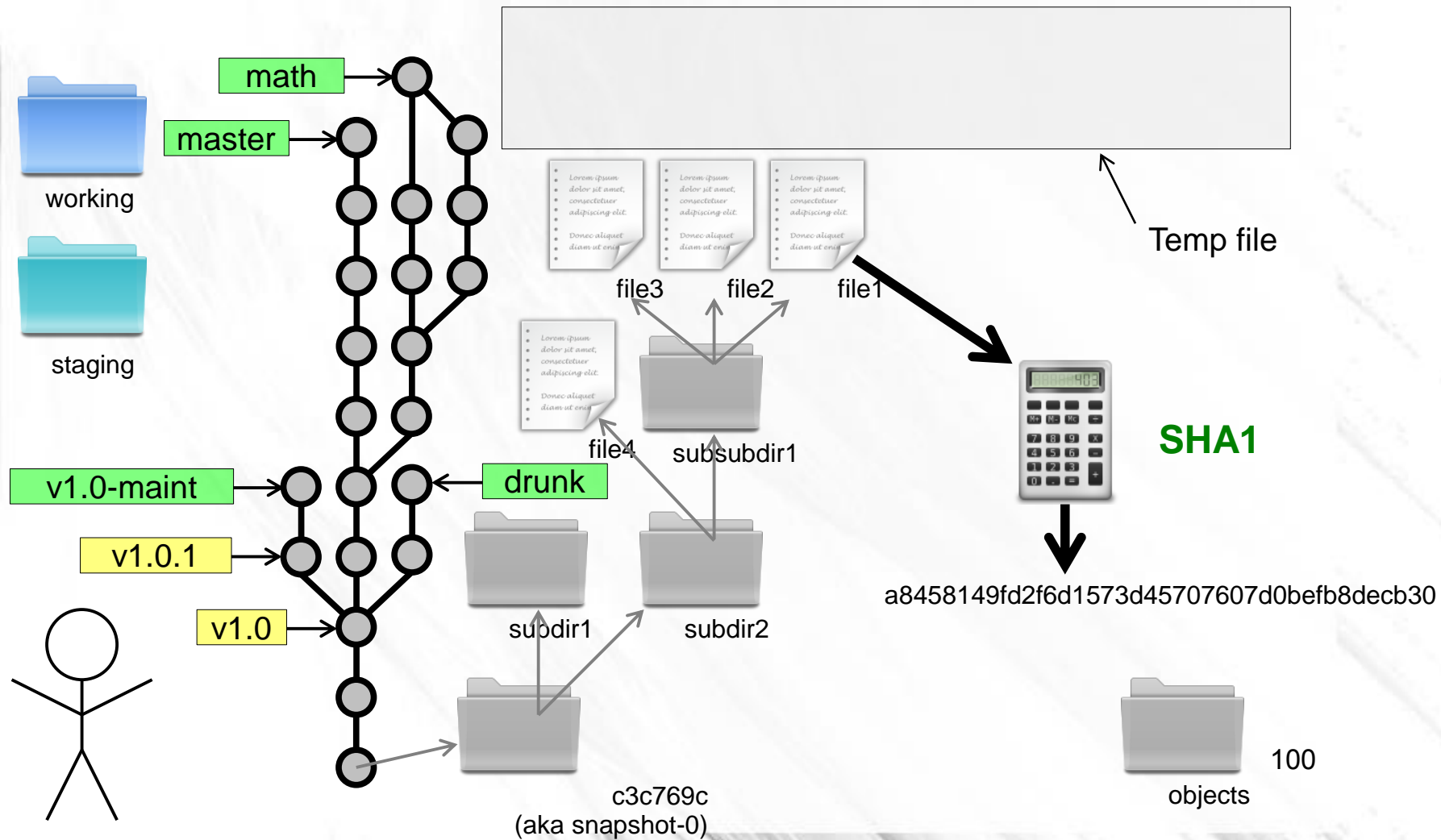
Eliminating Duplication



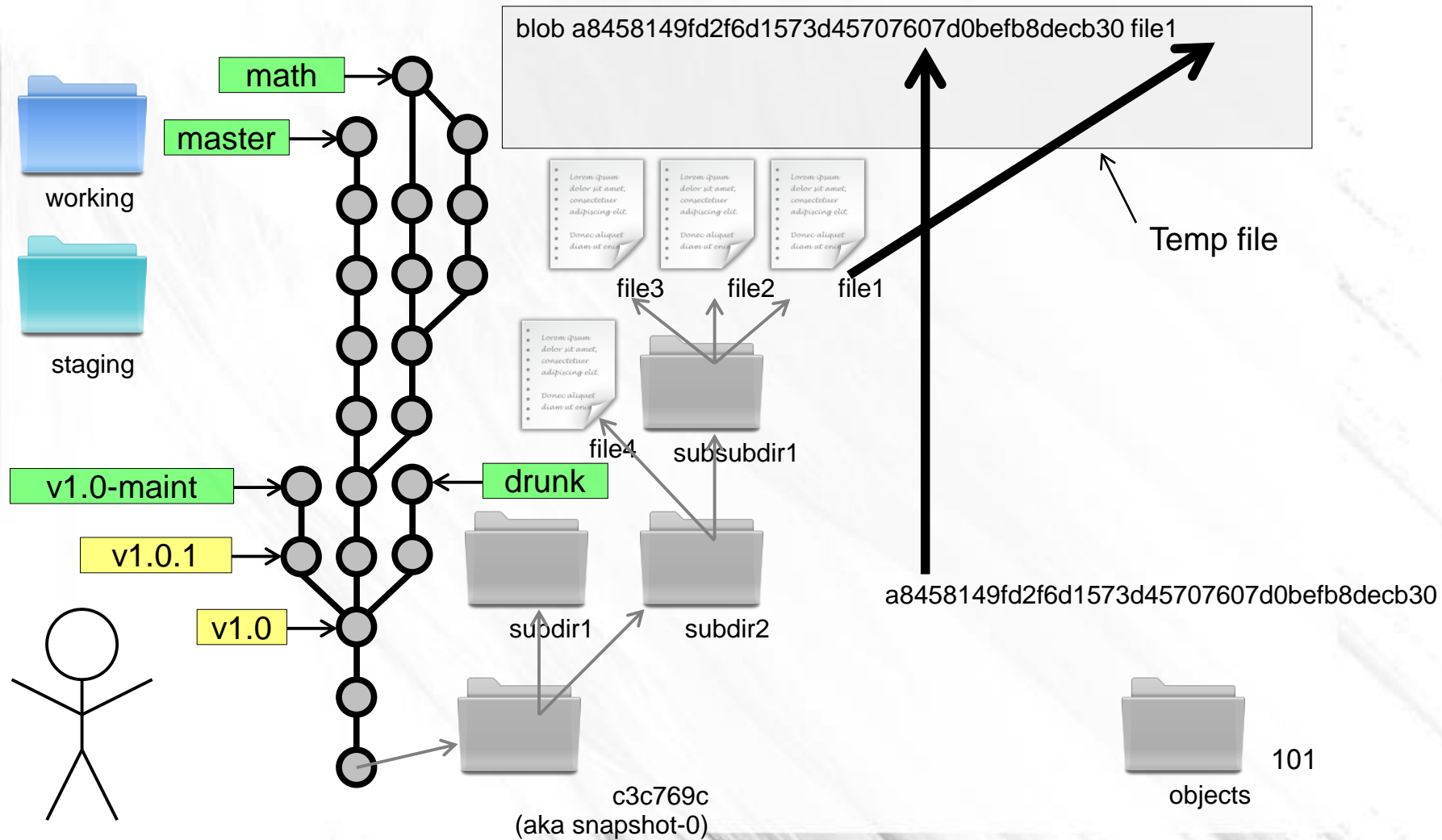
Eliminating Duplication



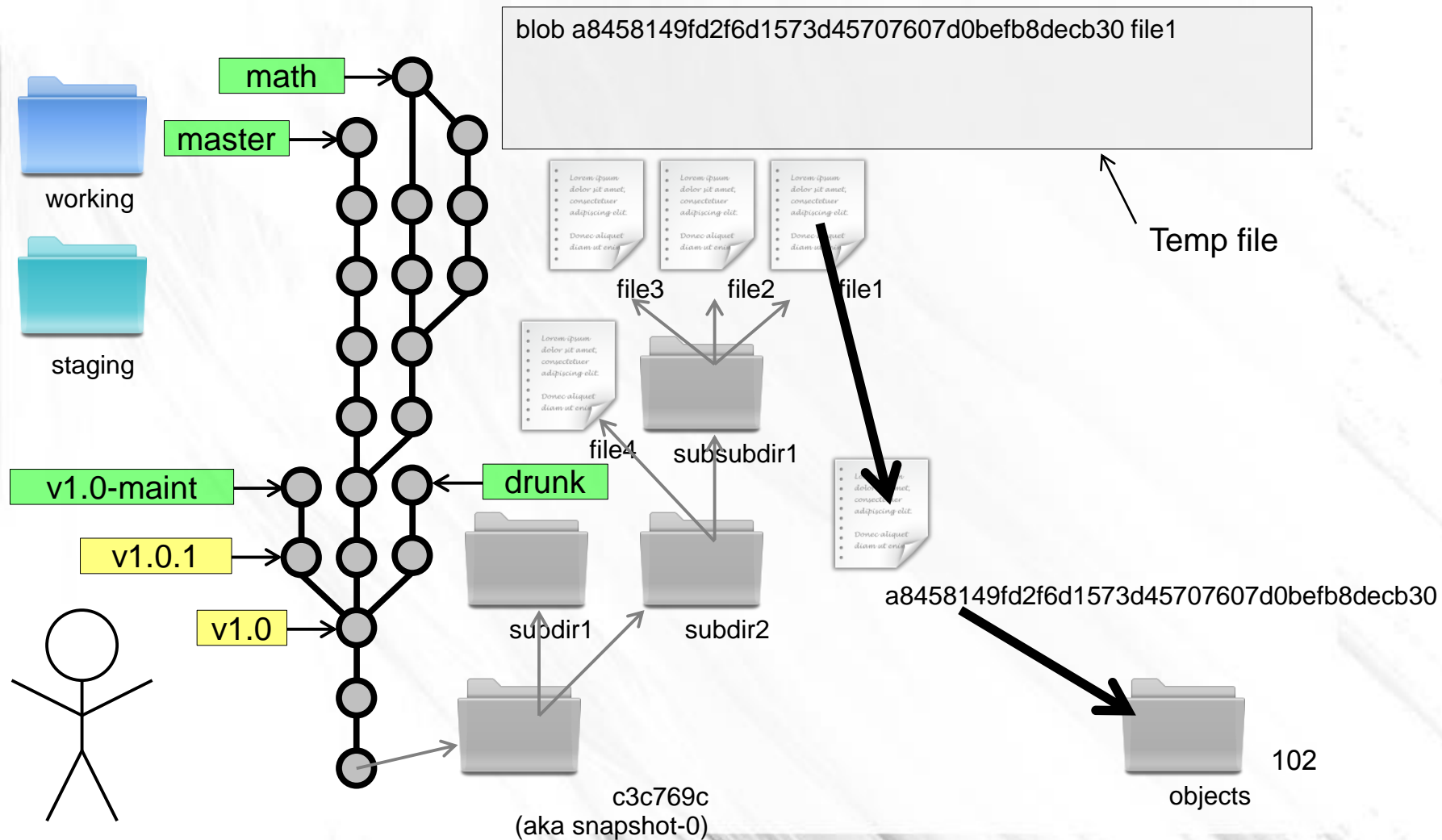
Eliminating Duplication



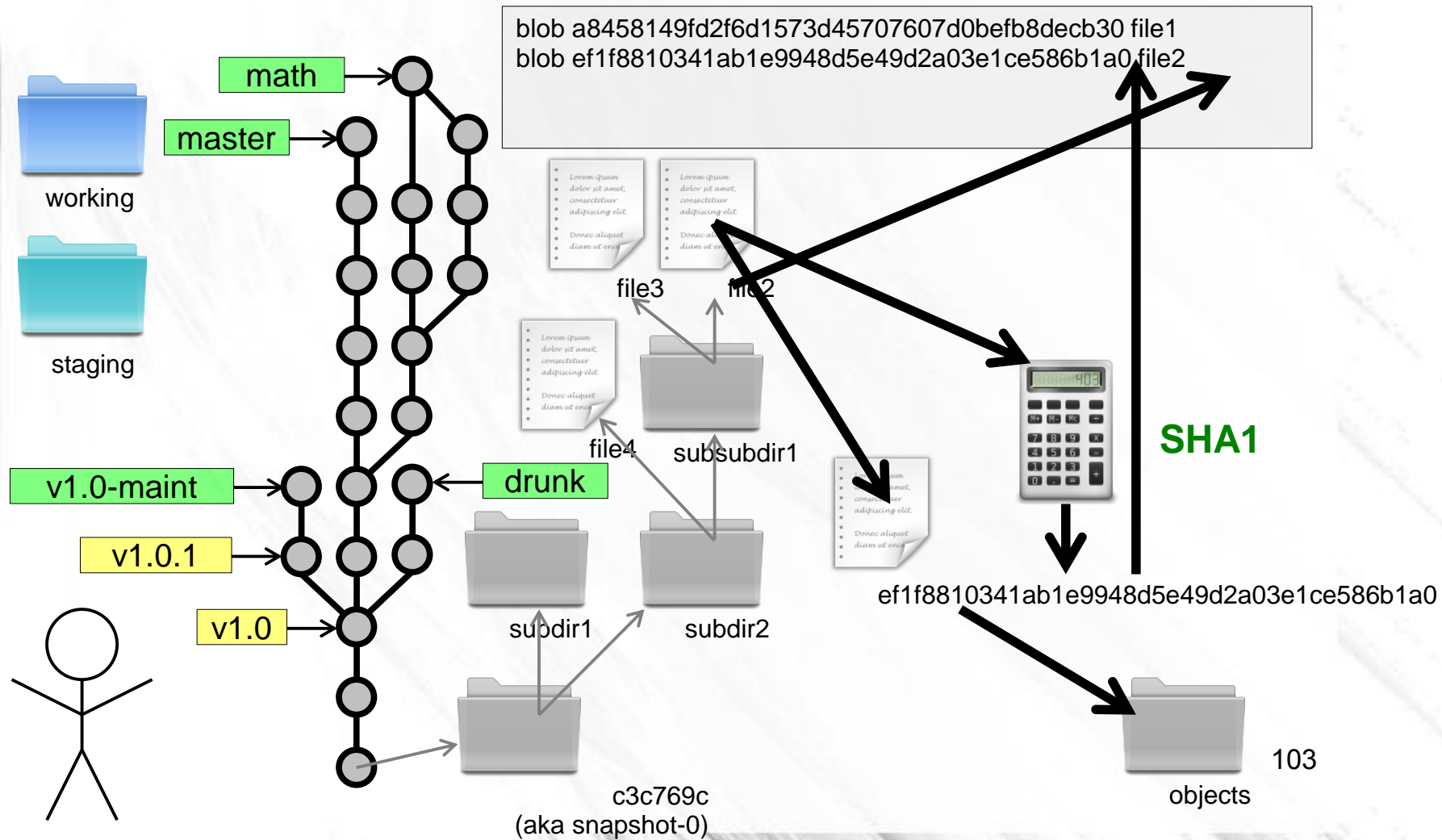
Eliminating Duplication



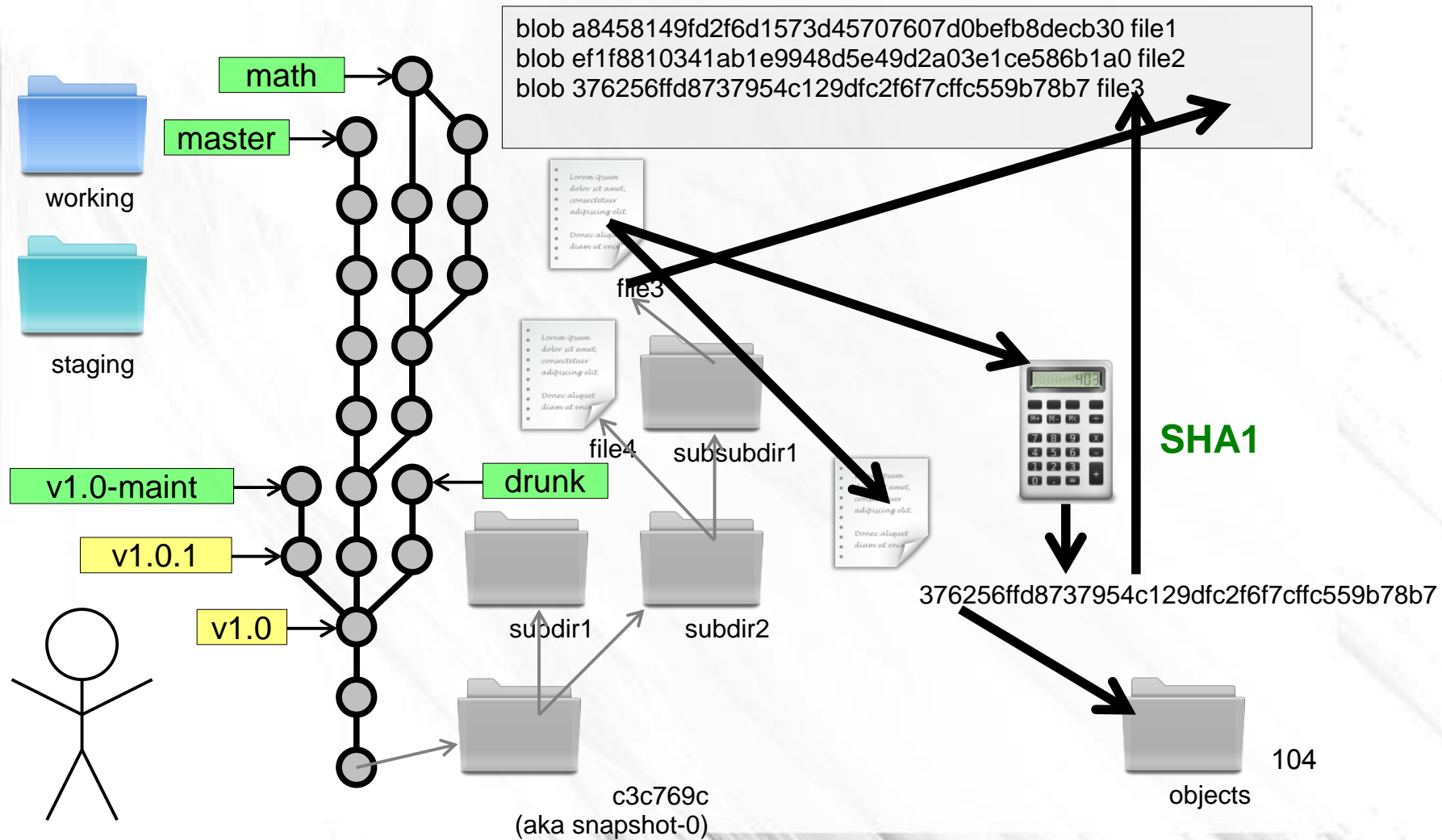
Eliminating Duplication



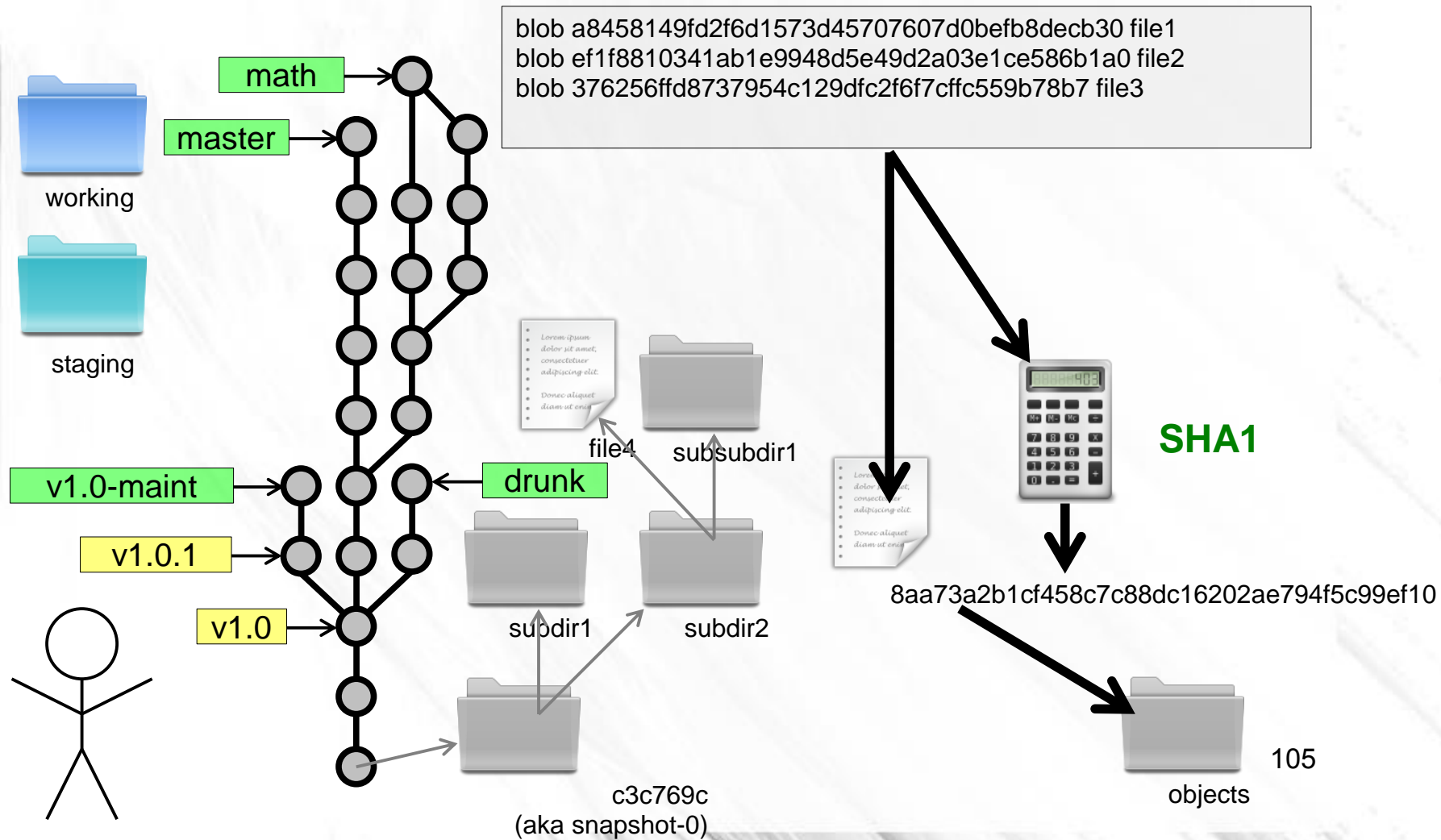
Eliminating Duplication



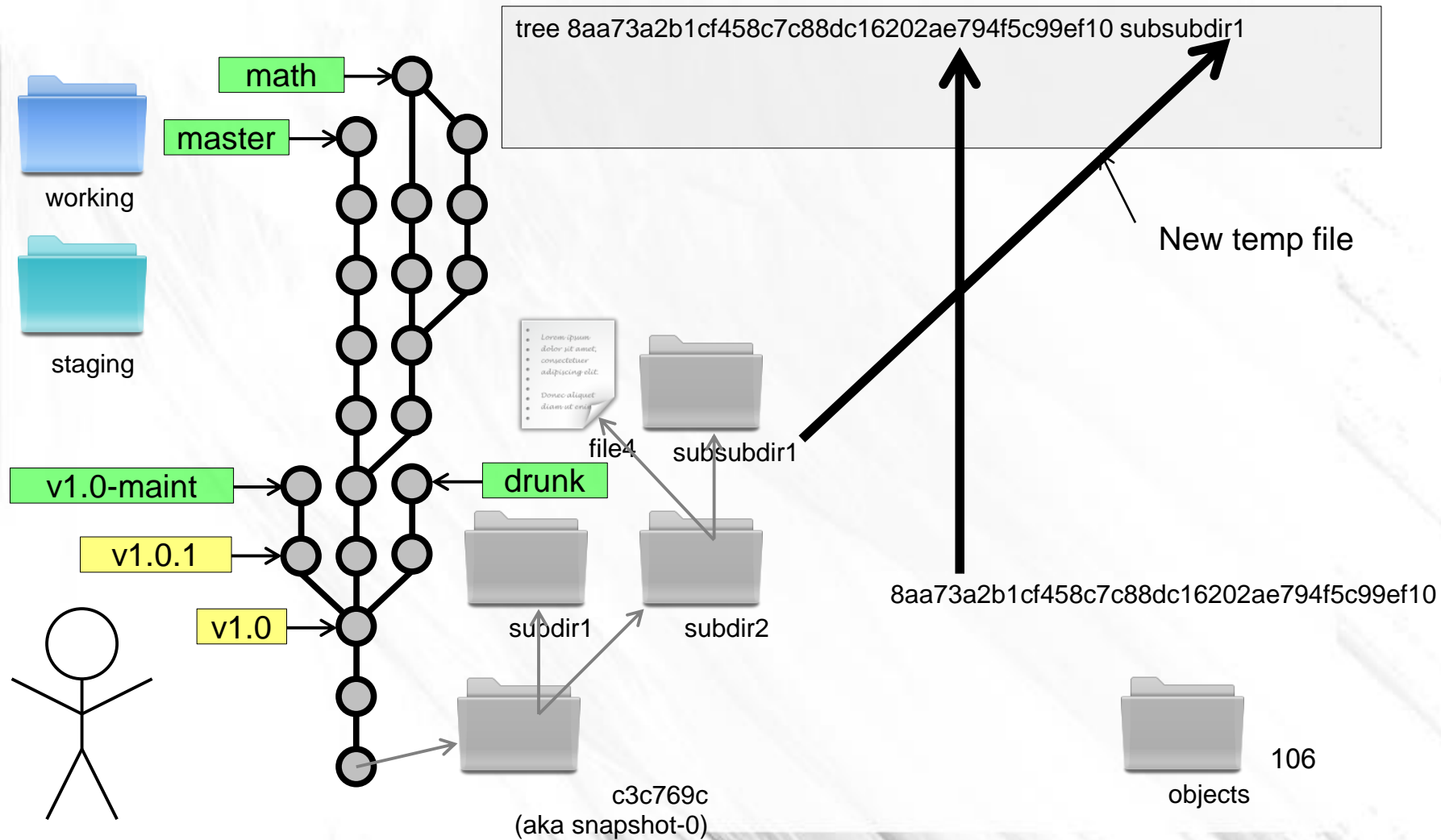
Eliminating Duplication



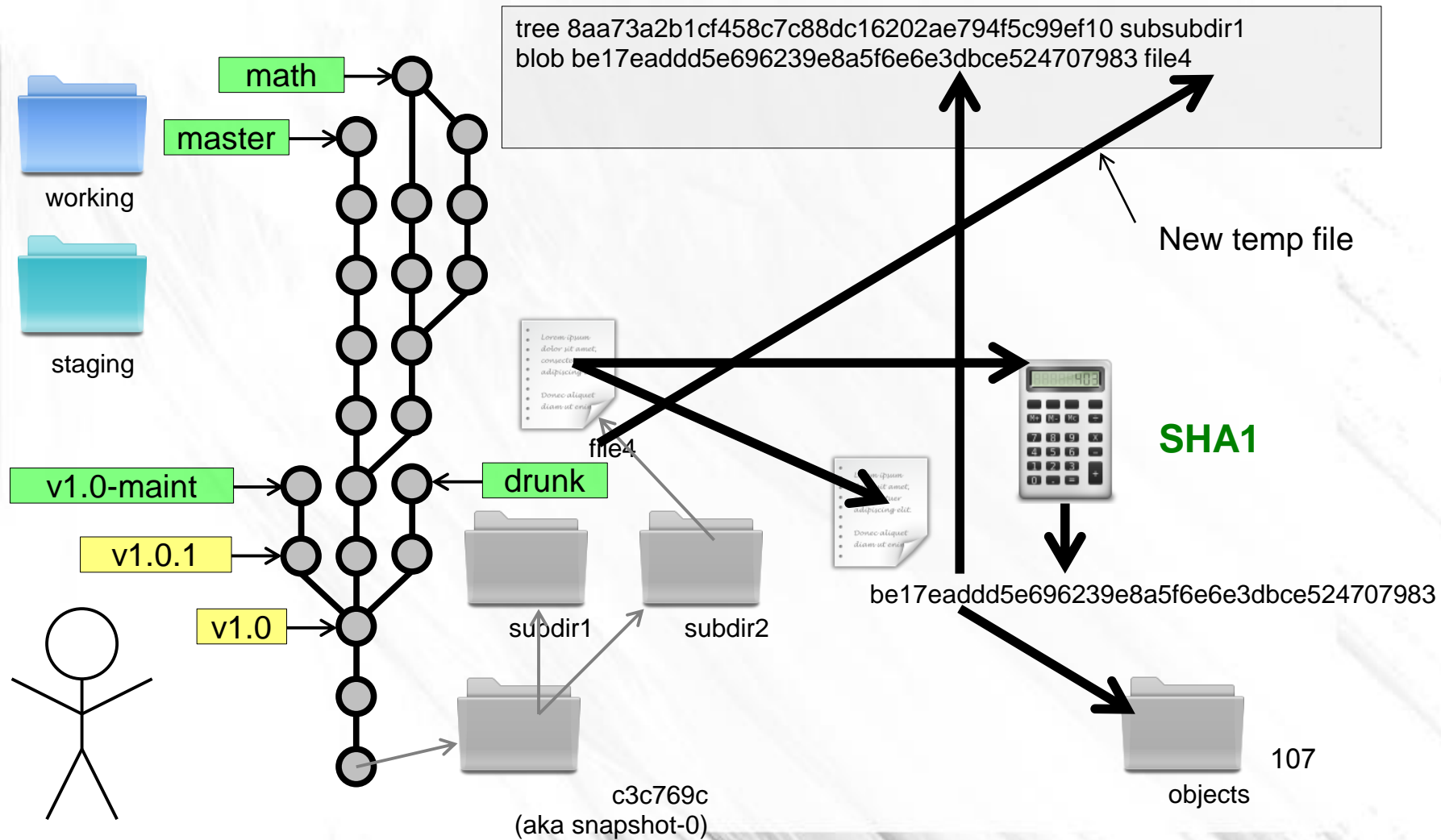
Eliminating Duplication



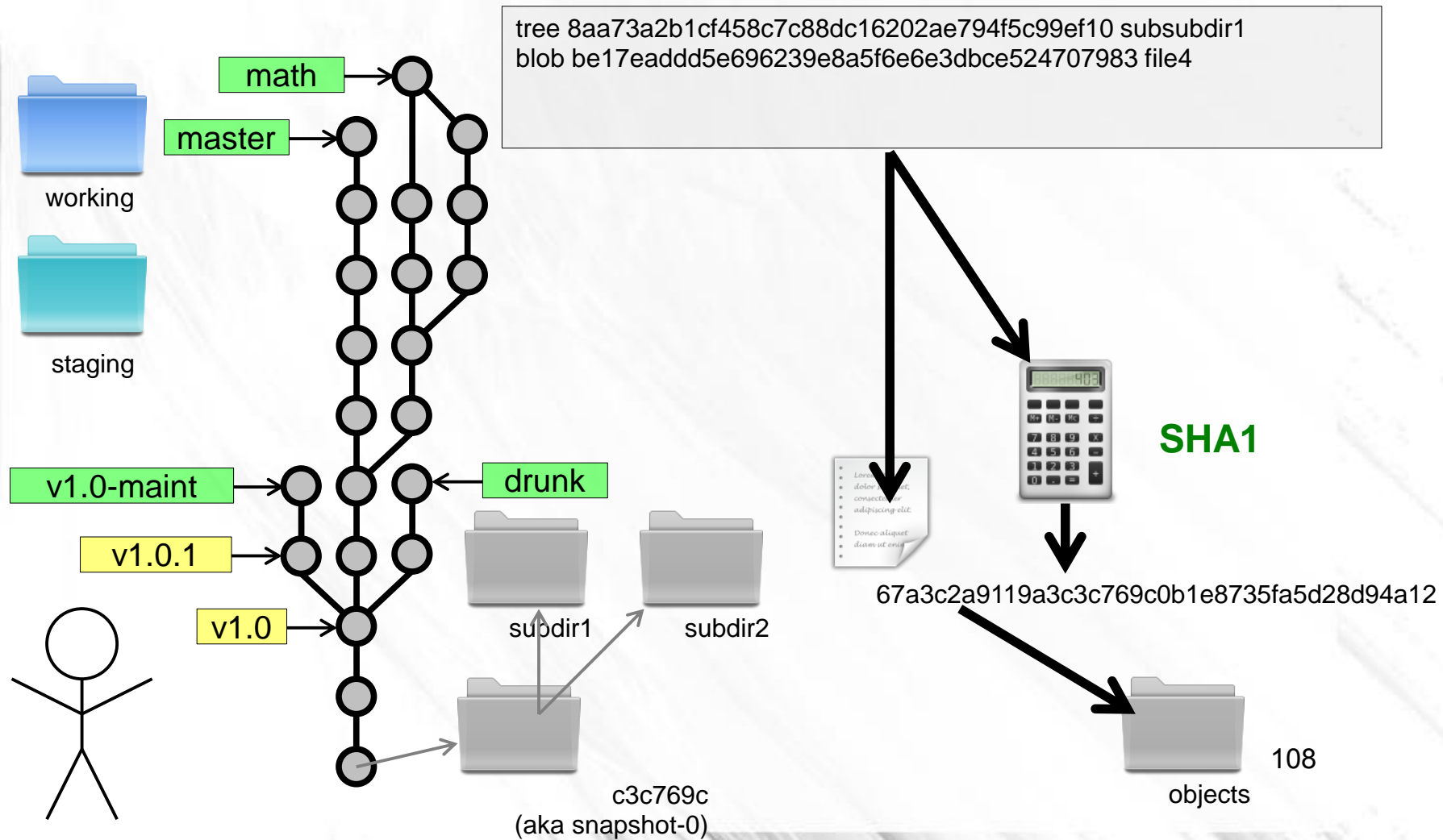
Eliminating Duplication



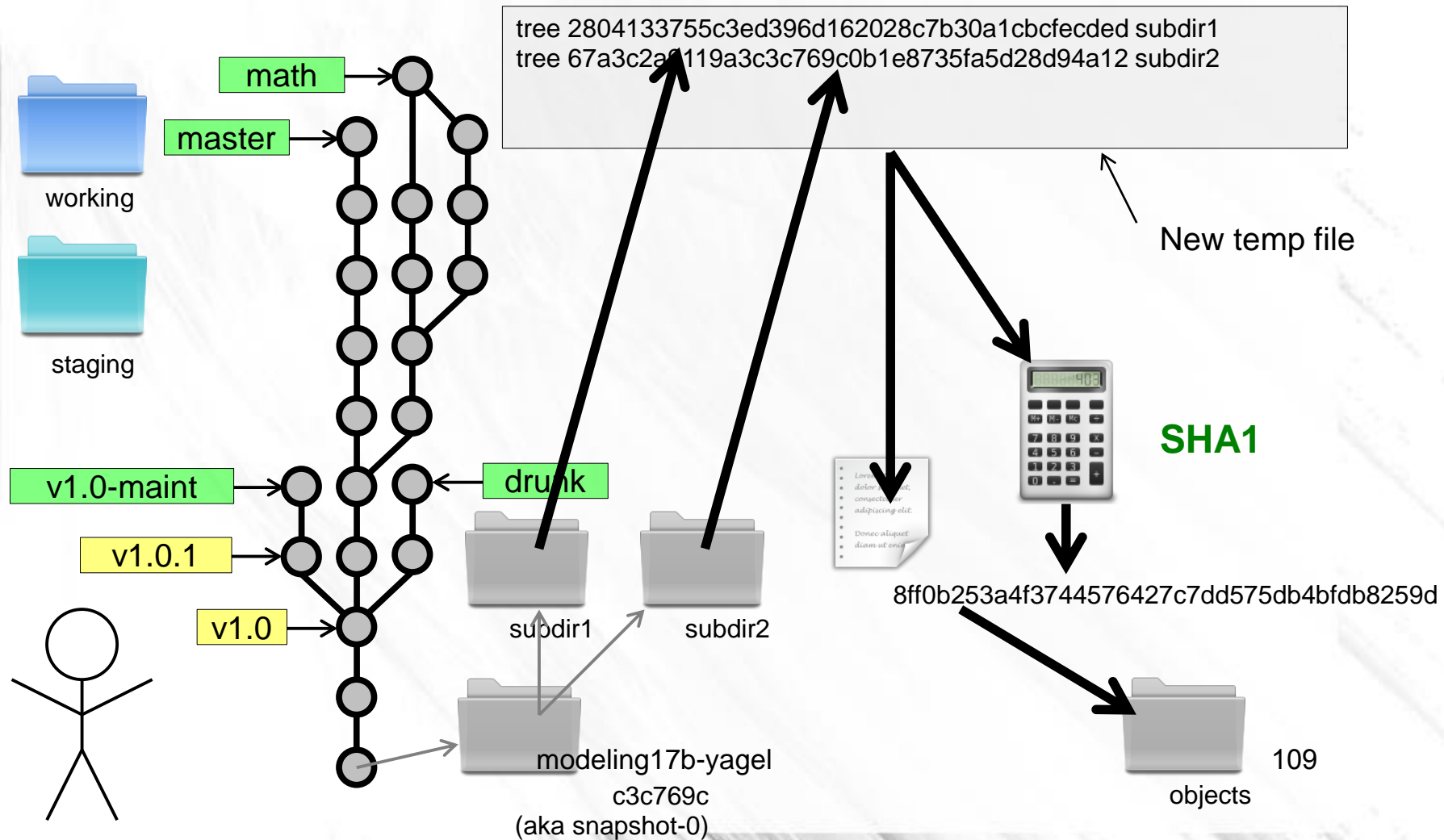
Eliminating Duplication



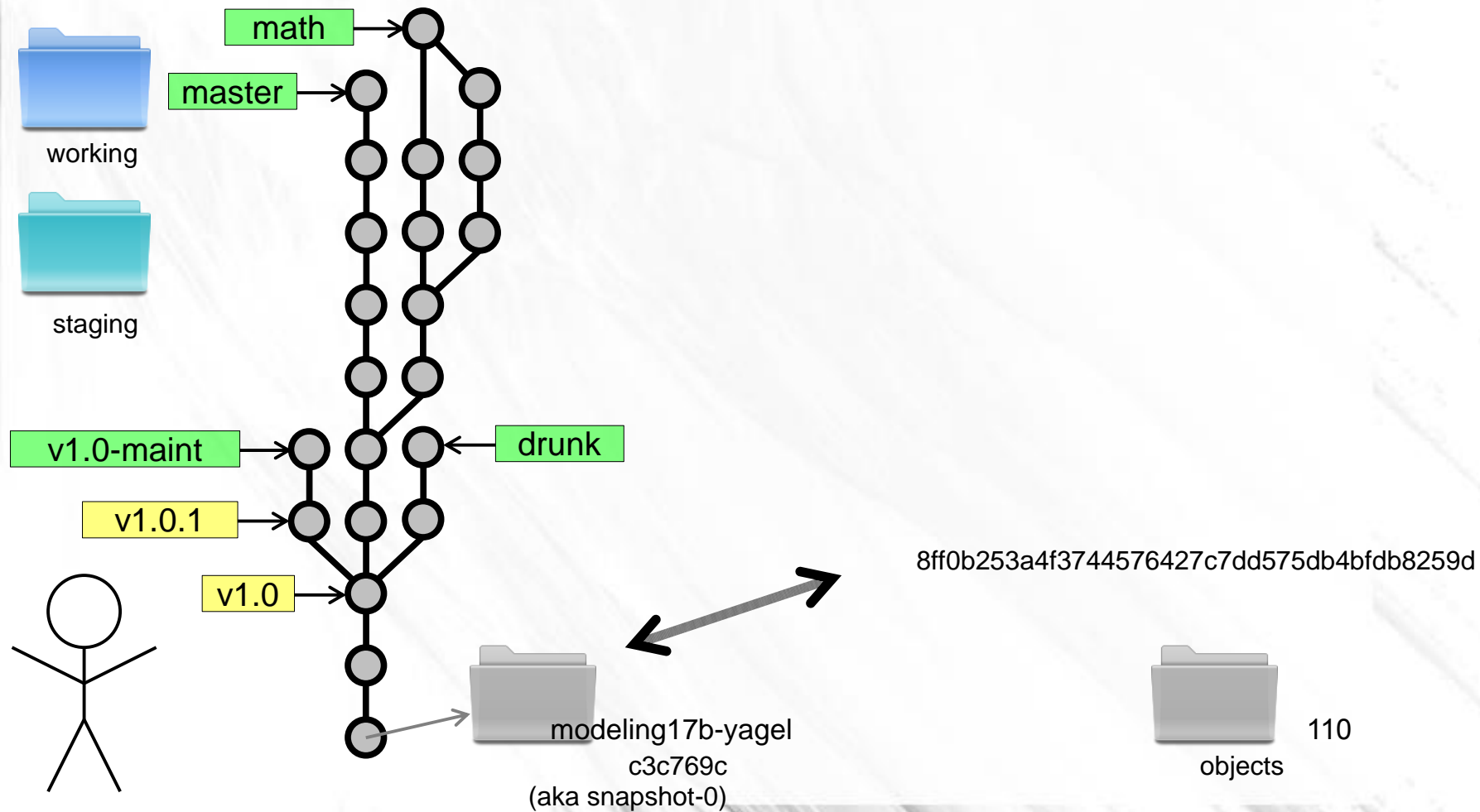
Eliminating Duplication



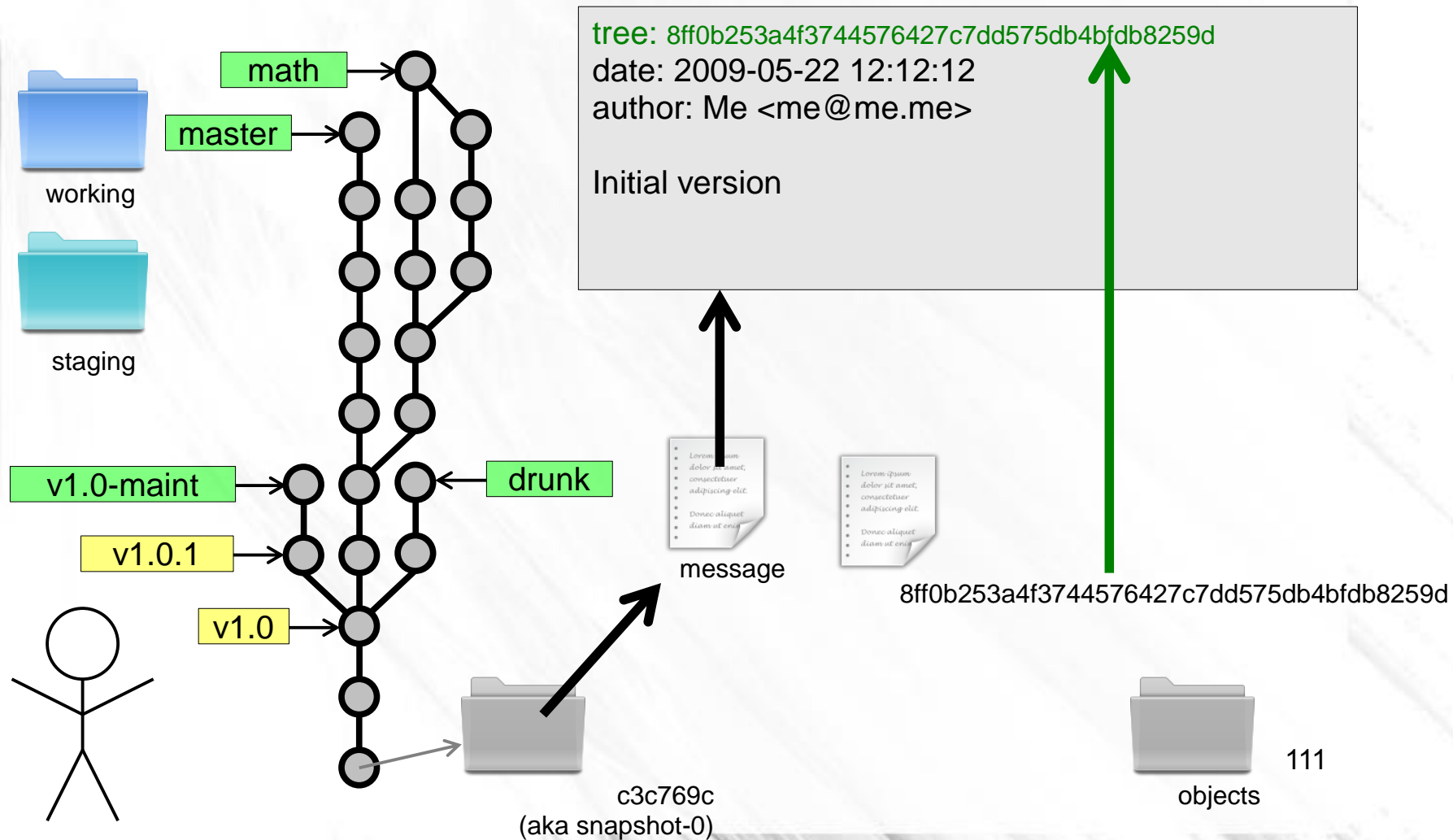
Eliminating Duplication



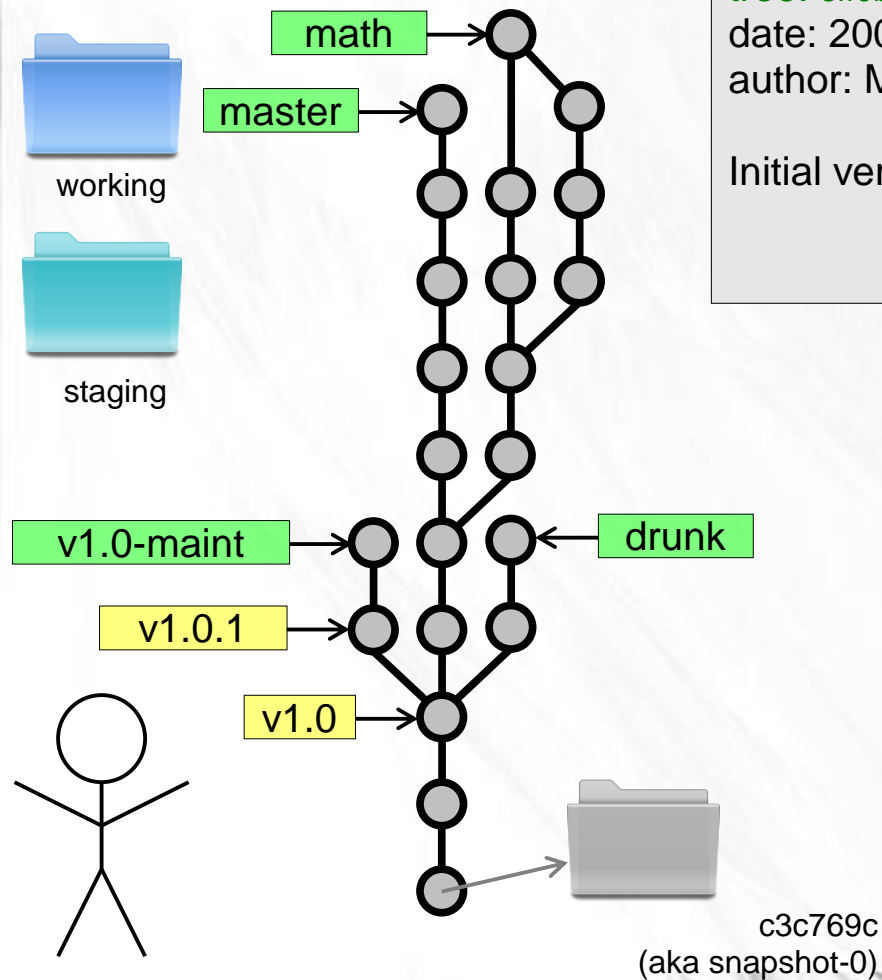
Eliminating Duplication



Eliminating Duplication



Eliminating Duplication

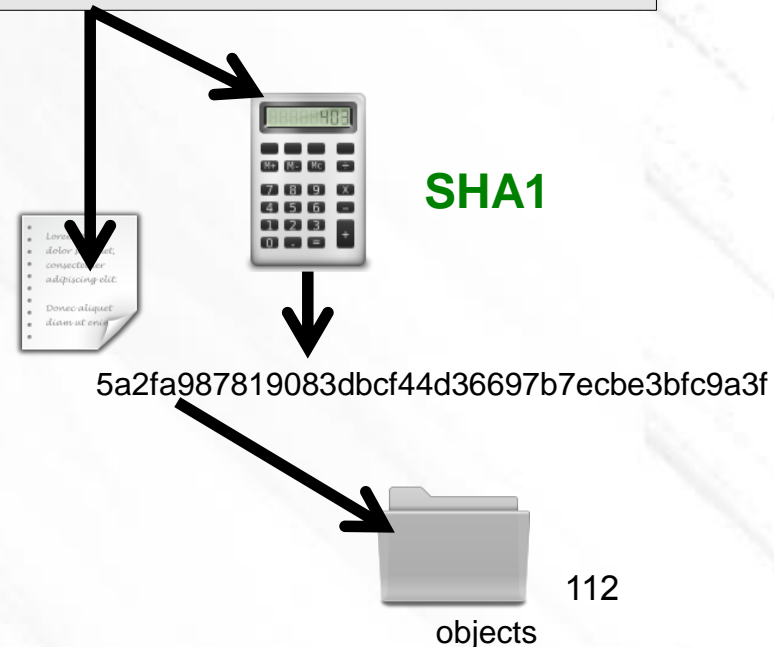


tree: 8ff0b253a4f3744576427c7dd575db4bfdb8259d

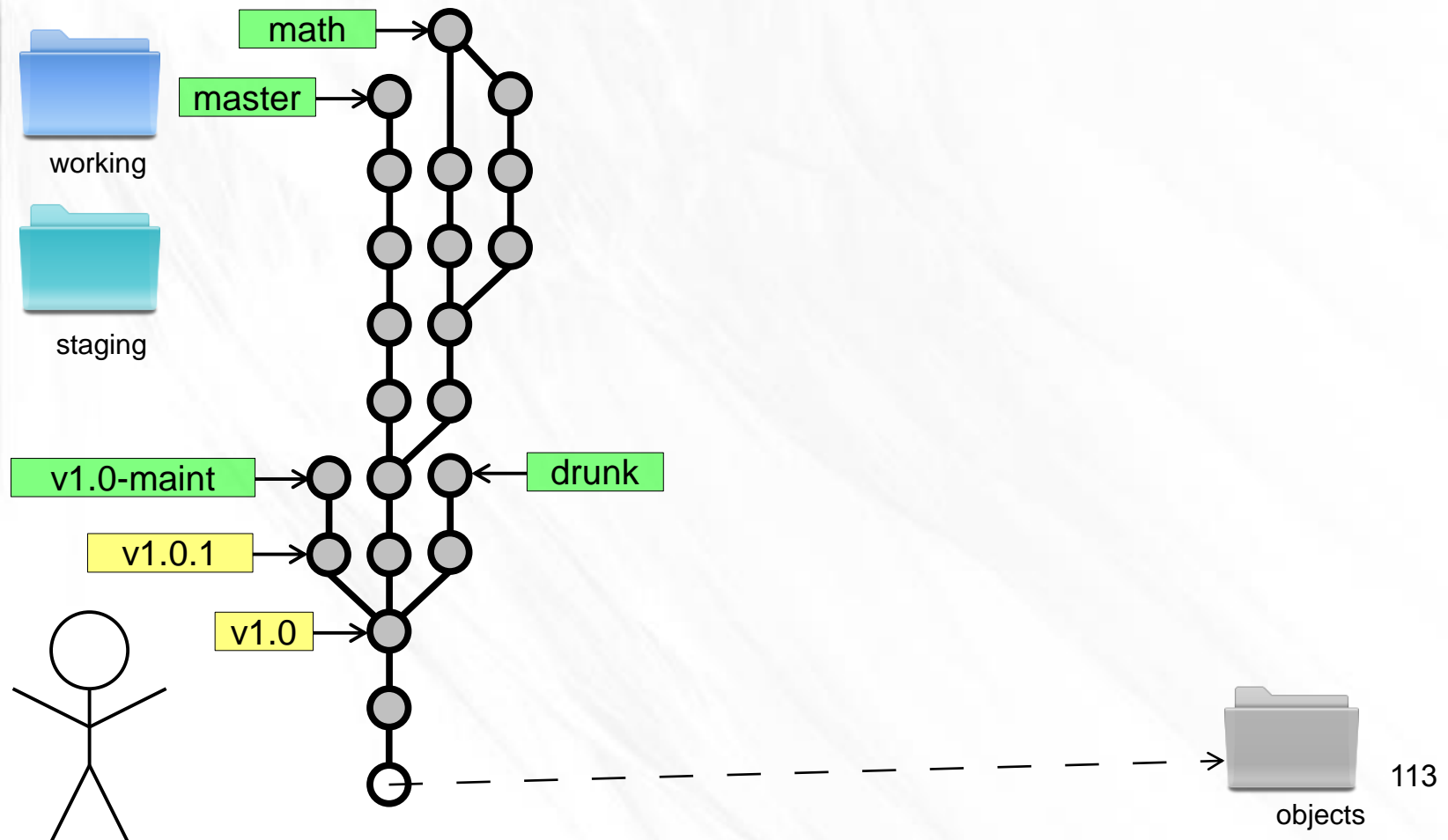
date: 2009-05-22 12:12:12

author: Me <me@me.me>

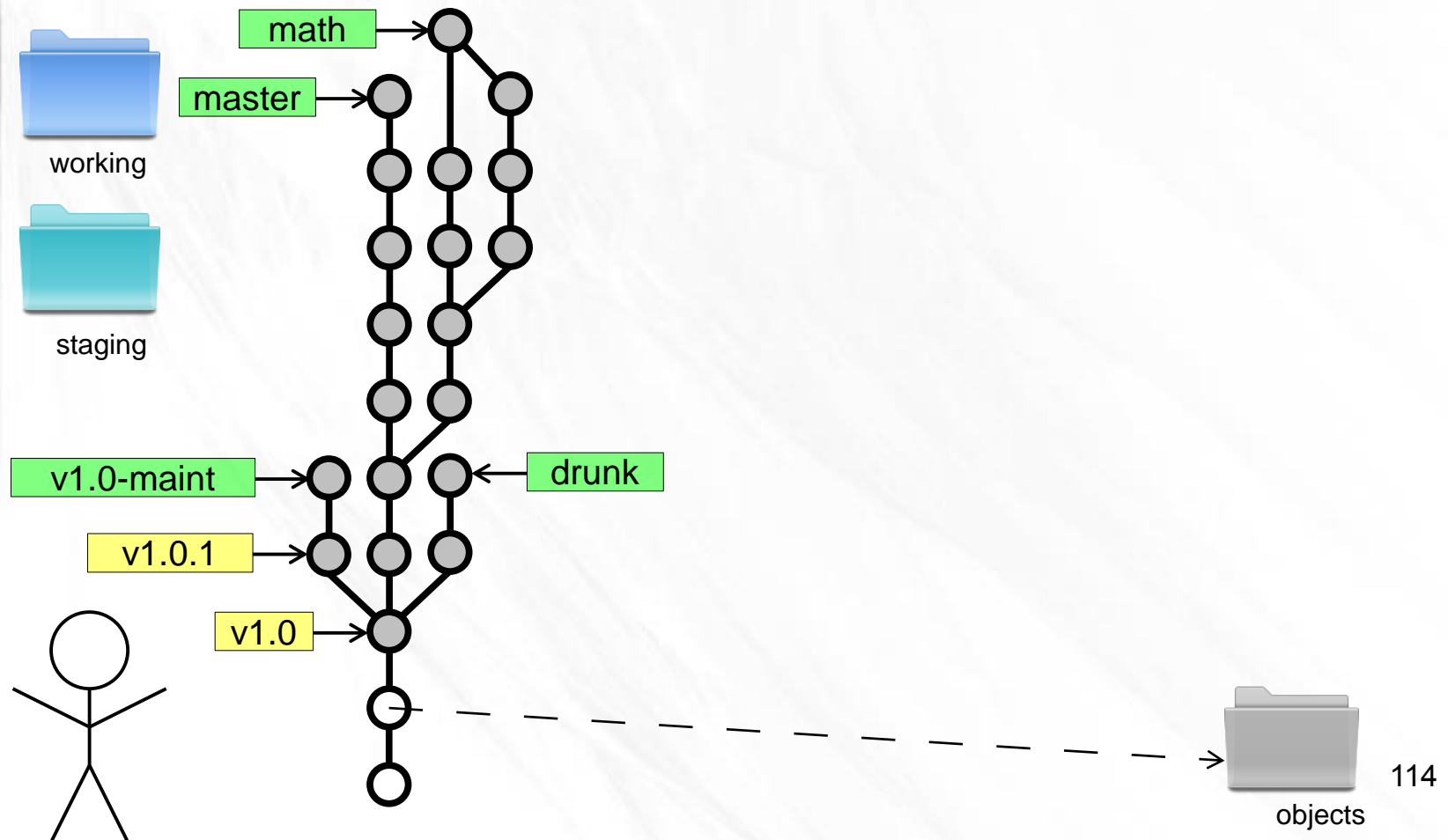
Initial version



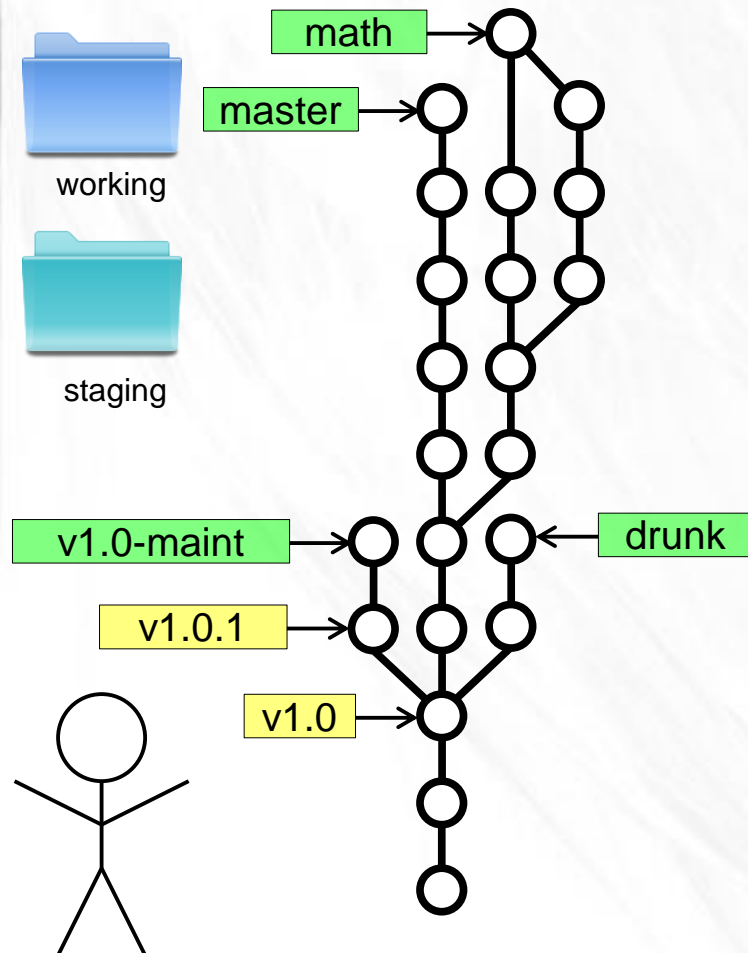
Eliminating Duplication



Eliminating Duplication



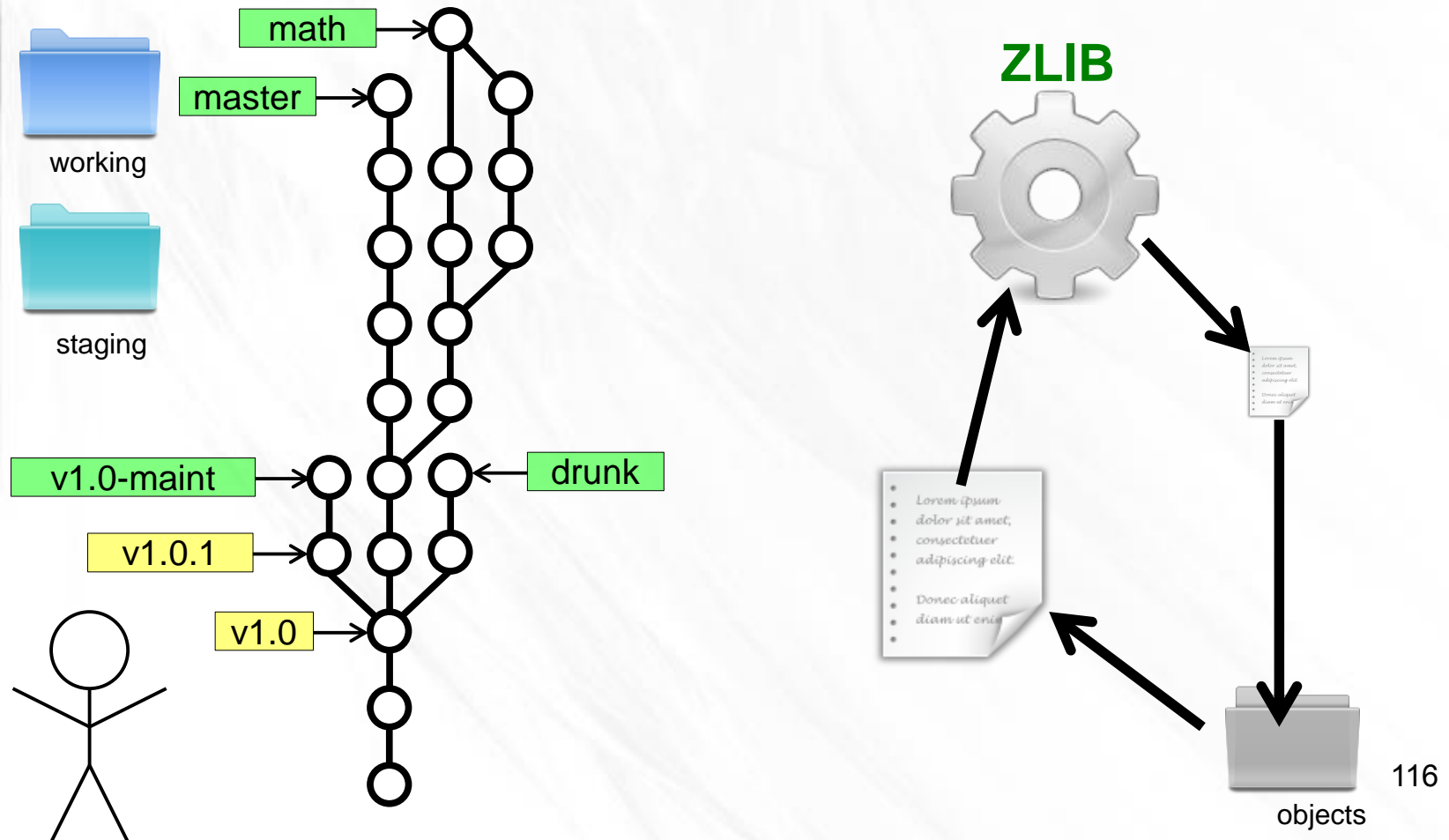
Eliminating Duplication



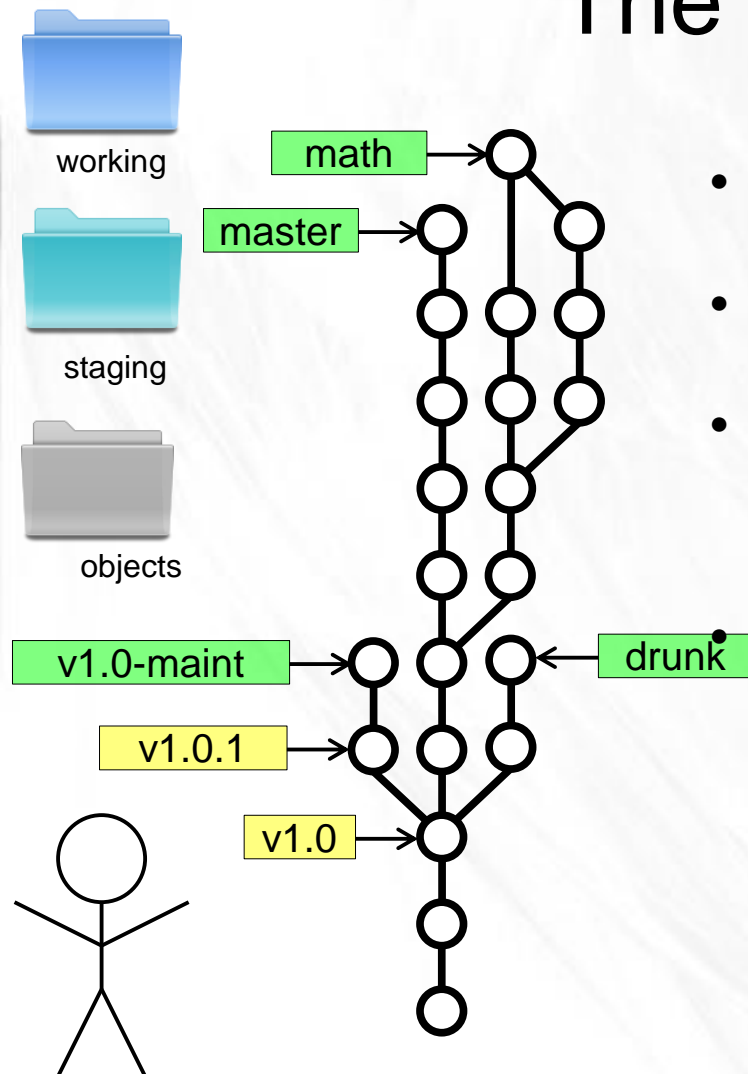
115

objects

Compressing Blobs



The True Git

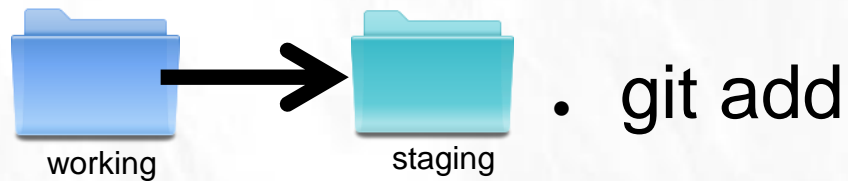


- TADAA!
- This is pretty much Git
- Nicer command line tools for all these operations
- Many, many other tools

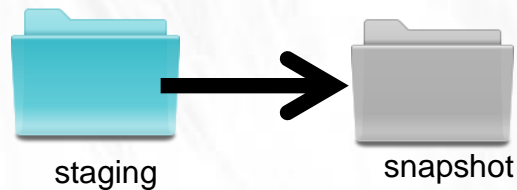
Commands: Getting Started

- First, tell Git who you are:
 - `git config --global user.name "My Name"`
 - `git config --global user.email "my@email.address"`
- Get help:
 - `git <command> -h`
 - `git help <command>`
- Start a new Git repository:
 - `git init`

Commands: Making snapshots



• git add



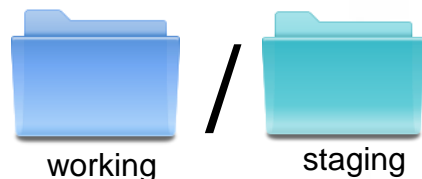
• git commit



git commit -a



• git log



• git status

gitk

```

Add the simple scripts I used to do a me
Merge the new object model thing from
[PATCH] Switch implementations of merg
[PATCH] Port fsck-cache to use parsing fi
[PATCH] Port rev-tree to parsing functi
[PATCH] Implementations of parsing func
[PATCH] Header files for object parsing
[PATCH] fix bug in read-cache.c which lo
[PATCH] Fix confusing behaviour of upda
Make "commit-tree" check the input obje
Make "parse_commit" return the "str119
Do a very simple "merge-base" that finds
Make "rev-tree.c" use the new-and-impro

```

Commands: Diffing



vs.



• `git diff`



vs.



• `git diff --staged`



vs.



• `git diff HEAD`



vs.



• `git diff <from> <to>`

Commands: Branches & Tags

- git branch
 - git branch <branch>
 - git checkout <branch>
 - git tag -l
 - git tag <tag>
- } git checkout -b ...

Commands: Fetching & Merging

- `git remote add <name> <URL>`

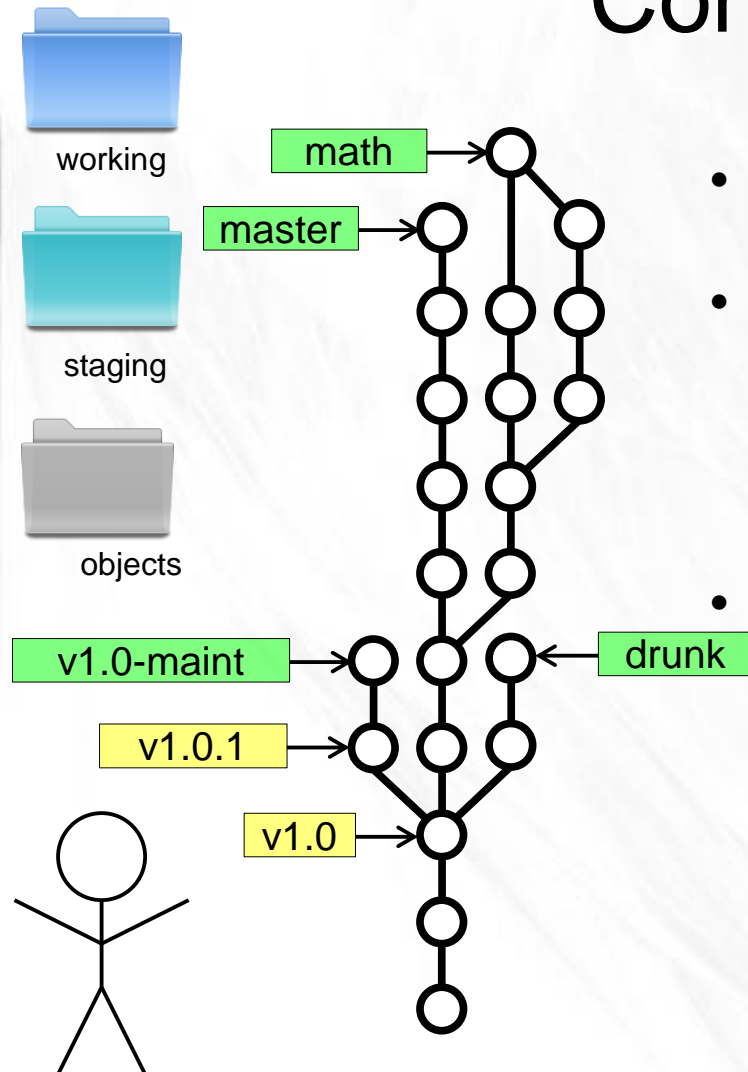
- `git fetch <name>`

}

`git pull`

- `git merge <name>/<branch>`

Conclusion



- Keep this parable in mind
- Git is simple and powerful

- One more thing:

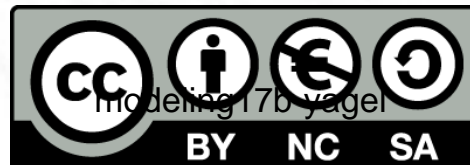
git reflog

Where to go next?

- Git homepage: <http://git-scm.com>
- Pro Git: <http://git-scm.com/book>
- Git Reference: <http://gitref.org>
- GitHub: <http://github.com>
- Gitorious: <http://gitorious.org>

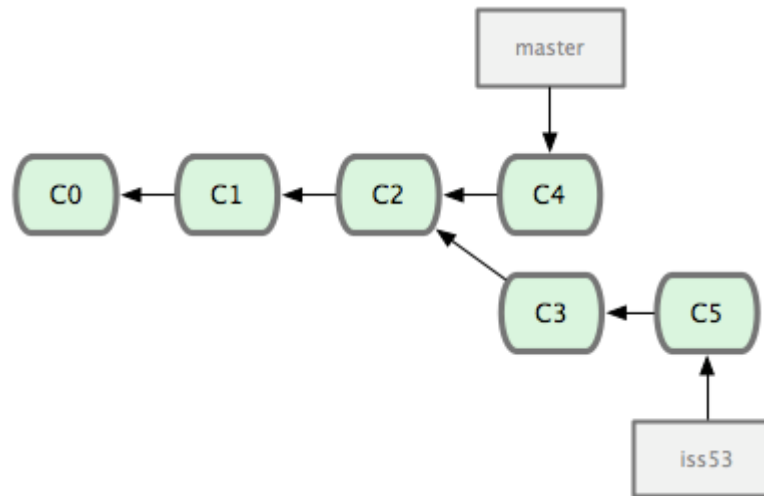
Questions?

- Thanks for your attention!
- These slides are available at:
https://github.com/jherland/git_parable
- Reach me at <johan@herland.net>

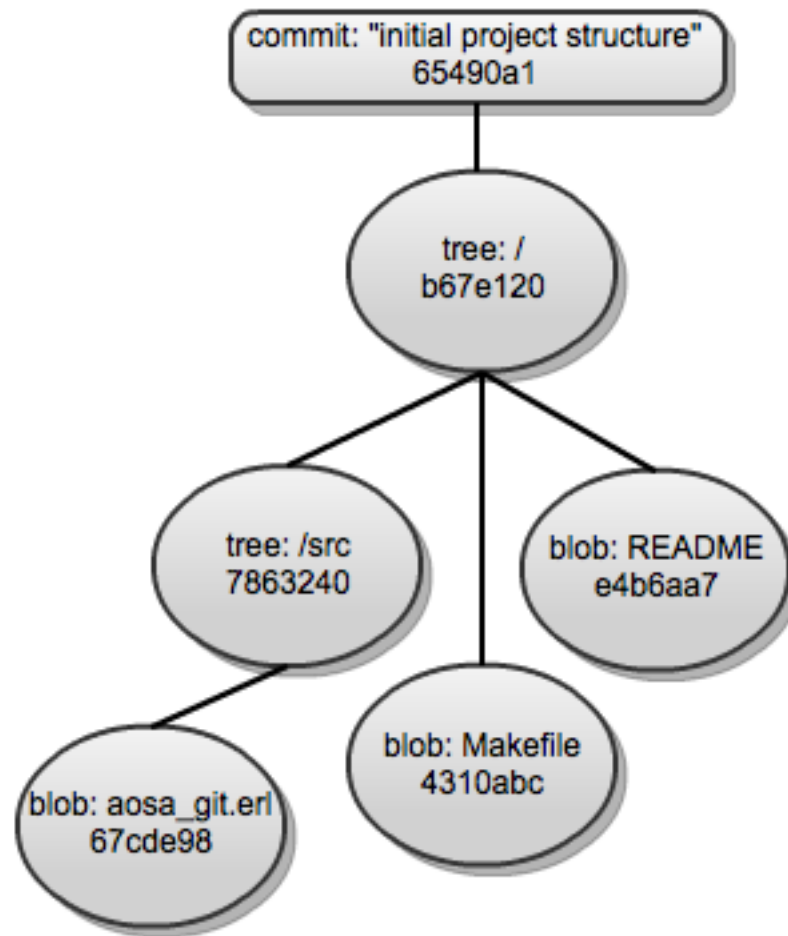


Content

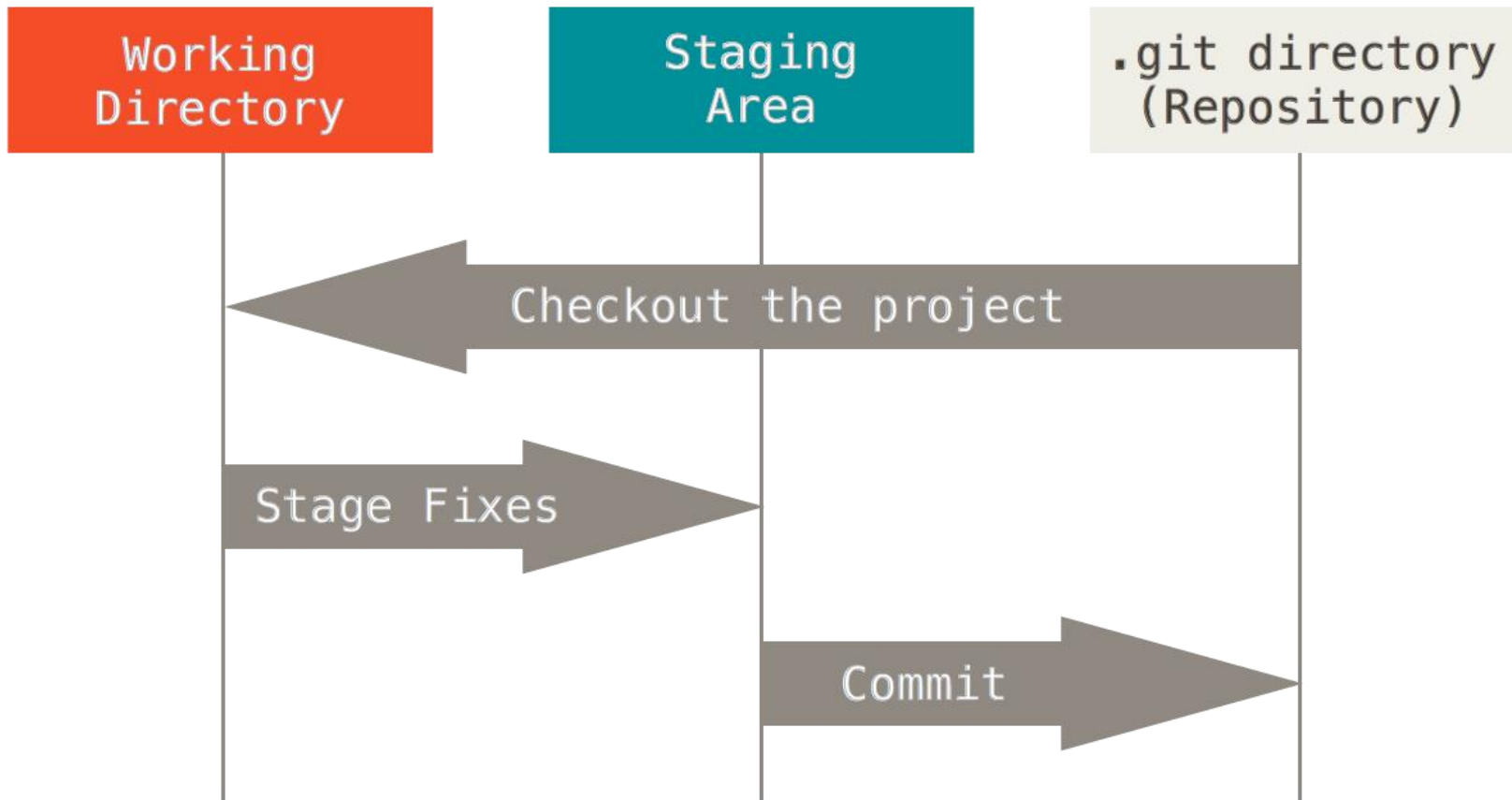
- DAG



Commit and Merge Histories



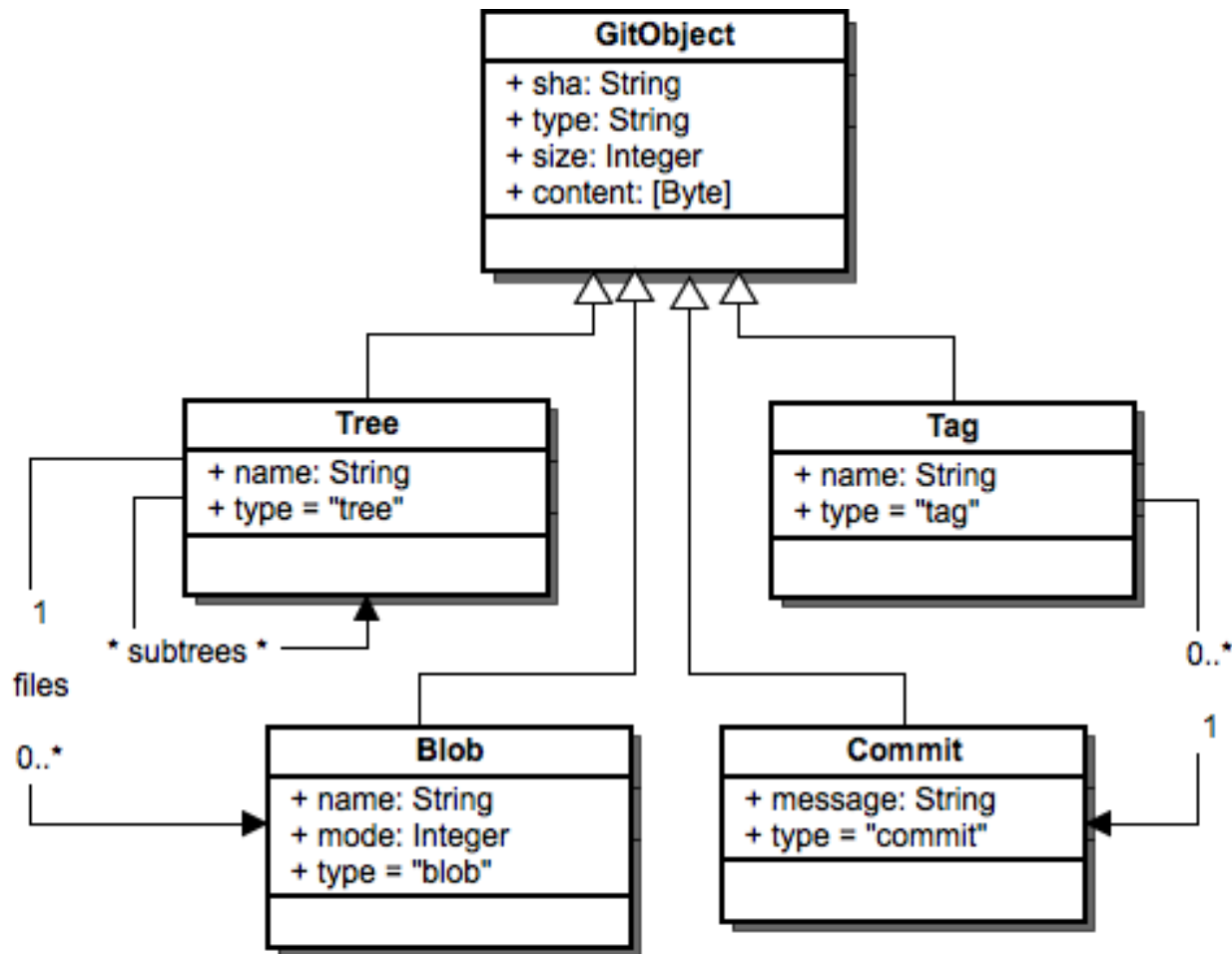
The 3 states



Git Init

```
tree .git/  
.git/  
|-- HEAD  
|-- config  
|-- description  
|-- hooks  
|   |-- applypatch-msg.sample  
|   |-- commit-msg.sample  
|   |-- post-commit.sample  
|   |-- post-receive.sample  
|   |-- post-update.sample  
|   |-- pre-applypatch.sample  
|   |-- pre-commit.sample  
|   |-- pre-rebase.sample  
|   |-- prepare-commit-msg.sample  
|   |-- update.sample  
|-- info  
|   |-- exclude  
|-- objects  
|   |-- info  
|   |-- pack  
|-- refs  
    |-- heads  
    |-- tags  
modeling17b-yagel
```

Object Database



Cons

- IDE Integration (not anymore really)
- Script based (“), Toolkit
- => complexity / confusion for users

Before

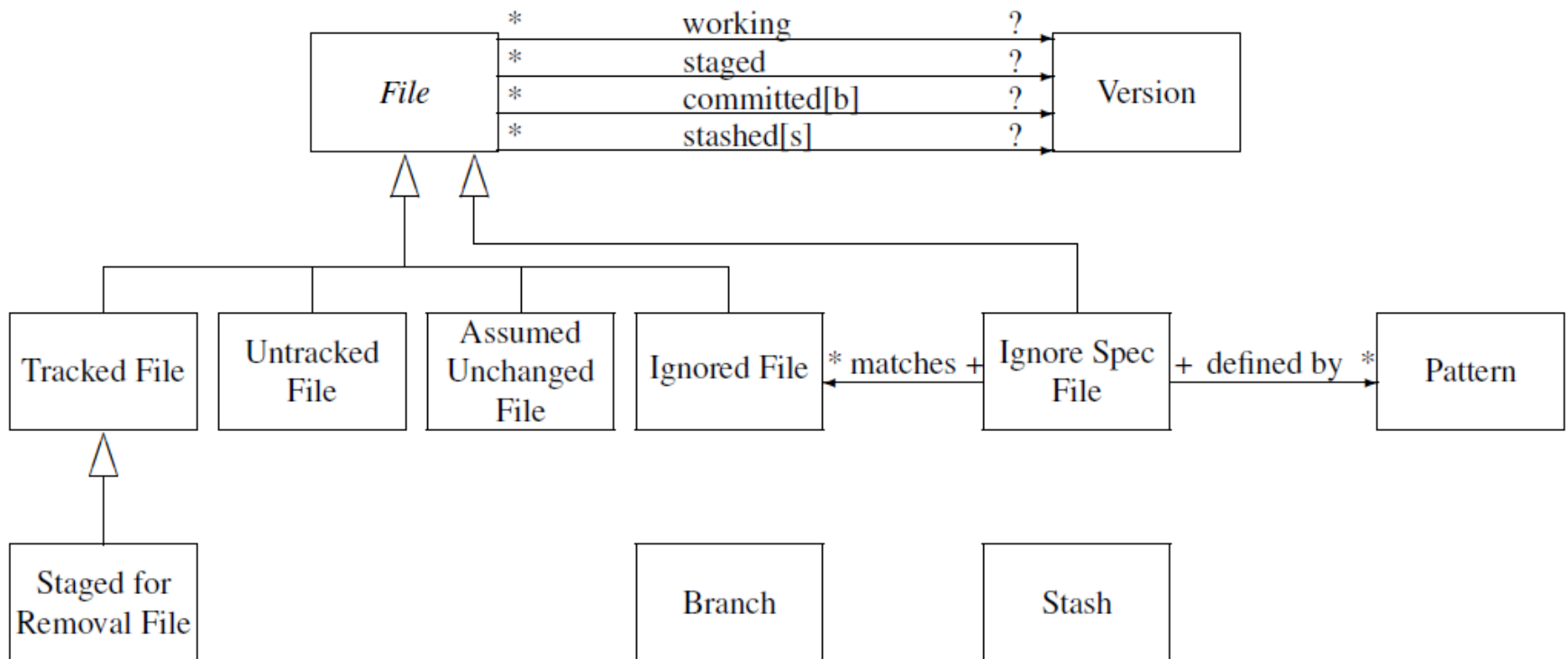


Figure 1. Graphical representation of Git's conceptual model.

Usability Issues Examples

- Let me just “checkin”
`git commit -a`
- Add -> edit -> commit -> reset.
What is the version now? (depends on reset params)
- Switching branches when there are edited new files existing in the other branch
(git checkout help: “Local modifications to the files in the working tree are kept, so that they can be committed to the <branch>.”)

Code Review

After (Gitless)

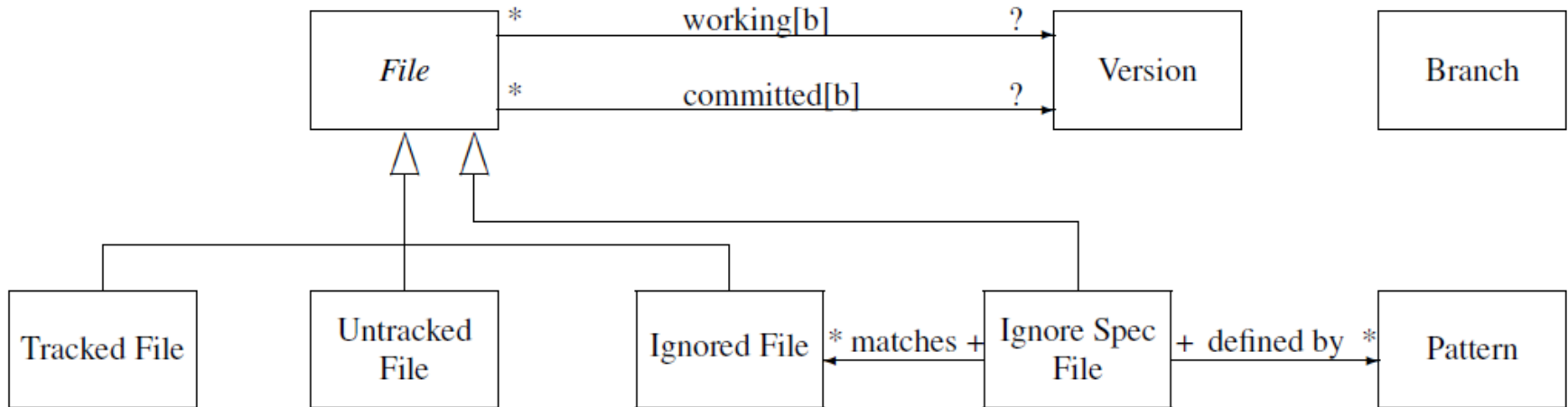
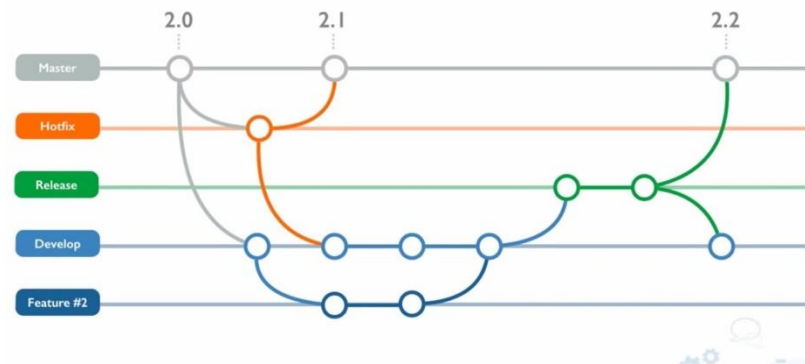


Figure 2. Graphical representation of Gitless's conceptual model.

Git Workflows - שיטות

- Centralized Workflow
- Feature Branch Workflow
- Gitflow Workflow
- Forking Workflow (OSS)



- [Github Flow](#)
- [Pull request](#) – בפרויקט?
- דוגמא: תהליך העבודה בפרויקט Nuget:
[Contributing a Bug Fix or Feature](#)
- קישורים נוספים בויקי

לסיכום

- תהליך: בקרת תצורה וגרסאות

- כלים: git / github

- שיטות: למשל git flow

- עוד: [Git For Ages 4 And Up](#)

- פעם הבאה: מידול עם UML

