Git во имя добра

Федотовский Павел

Почему Git?

- 90% разработчиков используют Git - .NET, Linux, Windows, Chrome
  - https://insights.stackoverflow.com/survey/2018#work-version-control
- TFS поддерживает Git, новые проекты по умолчанию с git
- Windows - самый большой репозиторий на земле
  - 300 GB
- http://z.github.io/whygitisbetter/

Немного истории

- Linux – использовали BitKeeper
- 2005 – Торвальдс в одиночку за пару месяцев написал git
- 16 июня 2005 – релиз linux 2.6.12 с помощью git

Git

Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

Git is easy to learn and has a tiny footprint with lightning fast performance.

Git

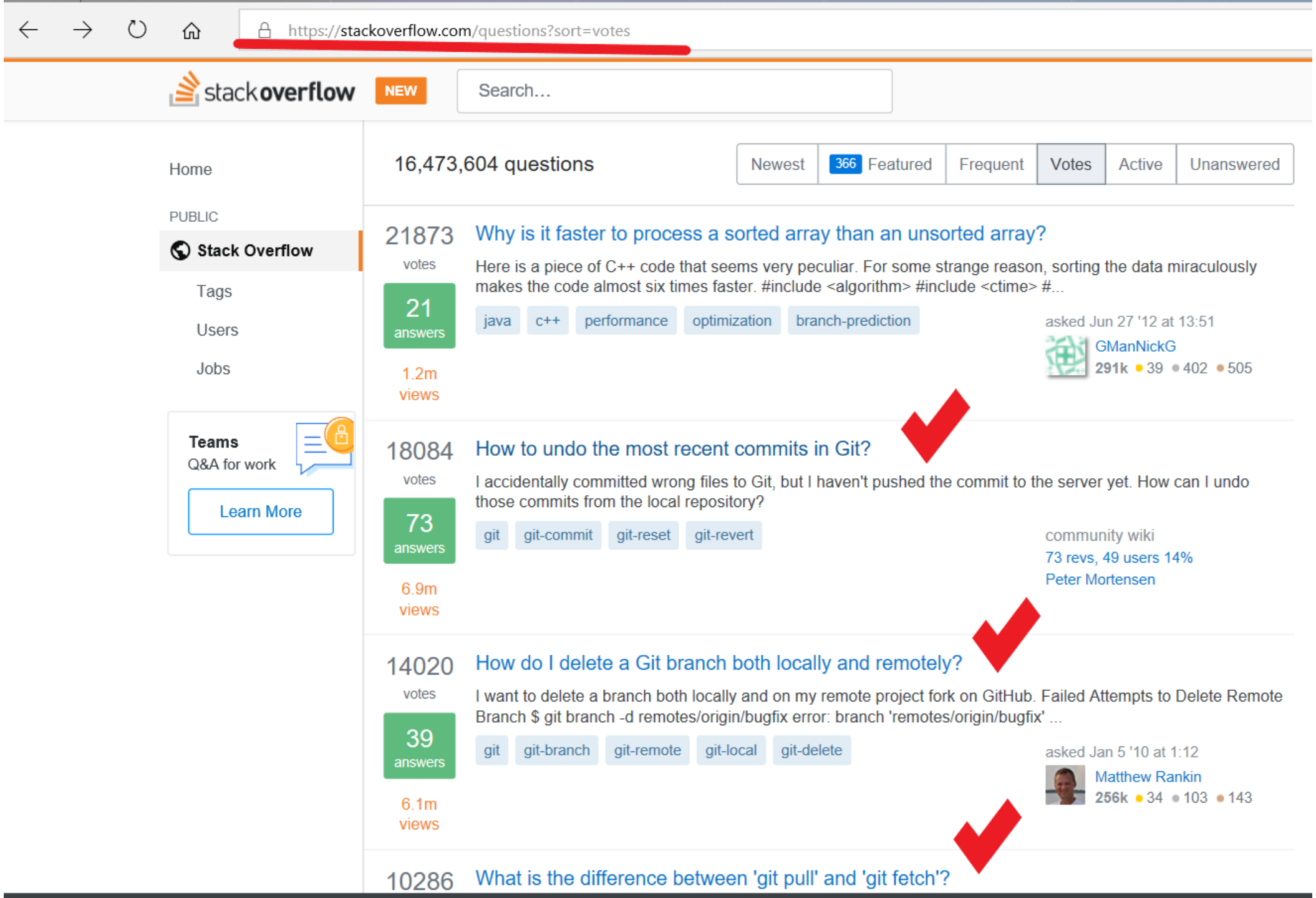
Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

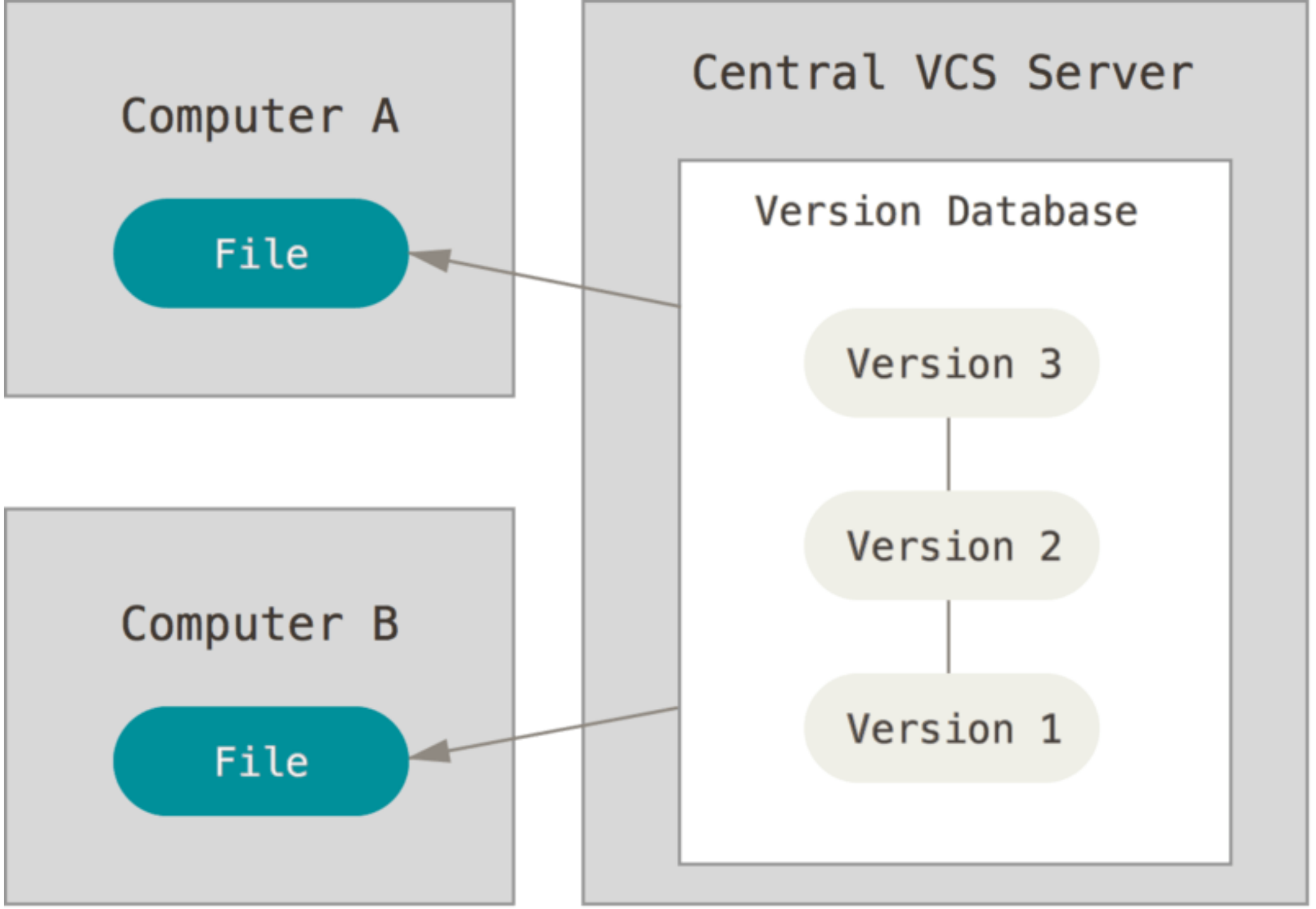
Git is easy to learn and has a tiny footprint with lightning fast performance.

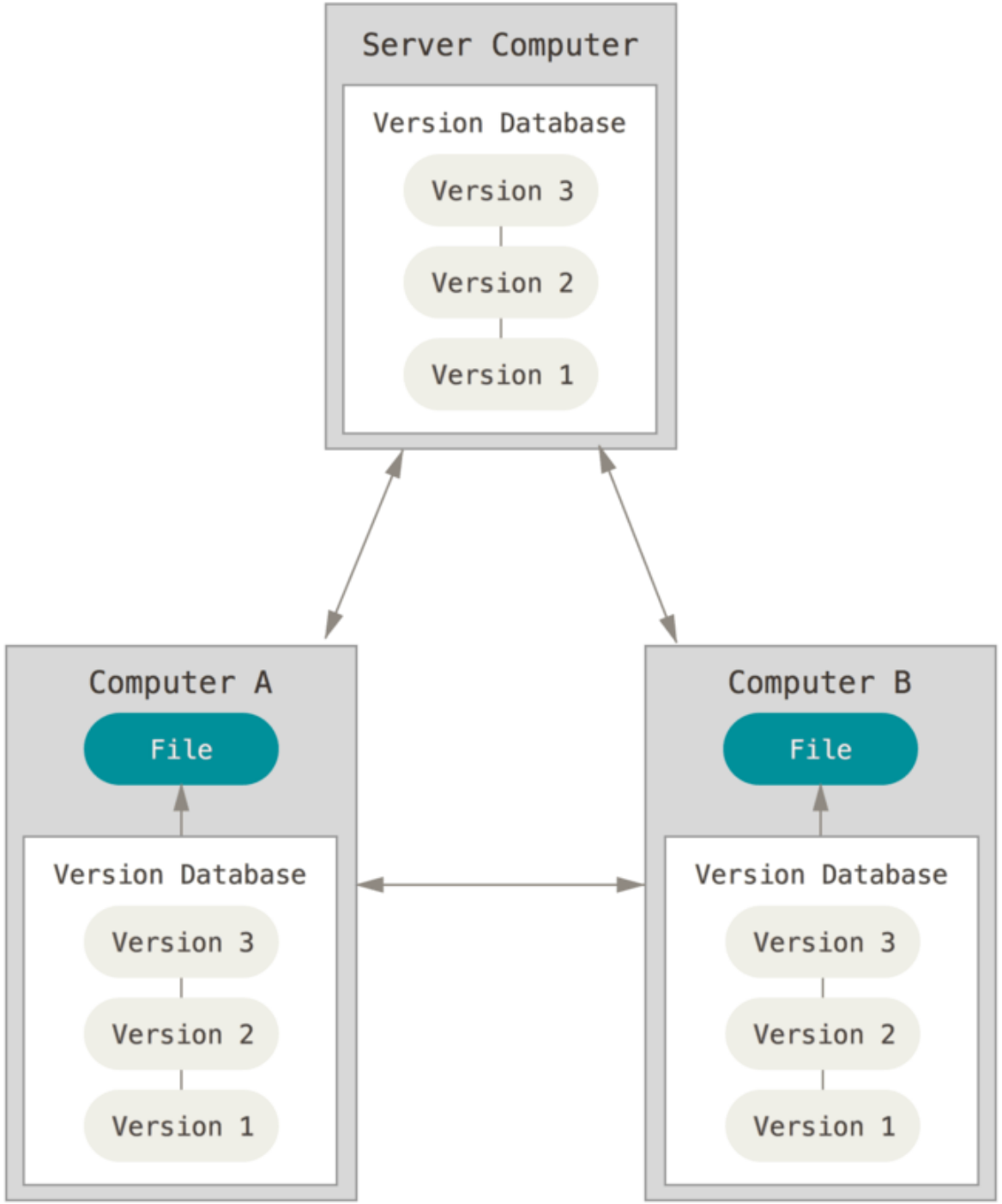
Уровни понимания Git

1. Ненависть – что за … ?
2. Хм.. прикольная штука
3. Да это круто!
4. Нужно использовать Git везде!

Доклад: переход на уровень 2 ☺


https://stackoverflow.com/questions?sort=votes
stackoverflow
NEW
Search...
Home
PUBLIC
Stack Overflow
Tags
Users
Jobs
Teams
Q&A for work
Learn More
16,473,604 questions
Newest
366 Featured
Frequent
Votes
Active
Unanswered
21873
votes
21
answers
1.2m
views
Why is it faster to process a sorted array than an unsorted array?
Here is a piece of C++ code that seems very peculiar. For some strange reason, sorting the data miraculously makes the code almost six times faster. #include <algorithm> #include <ctime> #...
java
c++
performance
optimization
branch-prediction
asked Jun 27 '12 at 13:51
GManNickG
291k • 39 • 402 • 505
18084
votes
73
answers
6.9m
views
How to undo the most recent commits in Git?
I accidentally committed wrong files to Git, but I haven't pushed the commit to the server yet. How can I undo those commits from the local repository?
git
git-commit
git-reset
git-revert
community wiki
73 revs, 49 users 14%
Peter Mortensen
14020
votes
39
answers
6.1m
views
How do I delete a Git branch both locally and remotely?
I want to delete a branch both locally and on my remote project fork on GitHub. Failed Attempts to Delete Remote Branch $ git branch -d remotes/origin/bugfix error: branch 'remotes/origin/bugfix' ...
git
git-branch
git-remote
git-local
git-delete
asked Jan 5 '10 at 1:12
Matthew Rankin
256k • 34 • 103 • 143
10286
What is the difference between 'git pull' and 'git fetch'?


Computer A
File
Computer B
File
Central VCS Server
Version Database
Version 3
Version 2
Version 1


Server Computer
Version Database
Version 3
Version 2
Version 1
Computer A
File
Version Database
Version 3
Version 2
Version 1
Computer B
File
Version Database
Version 3
Version 2
Version 1

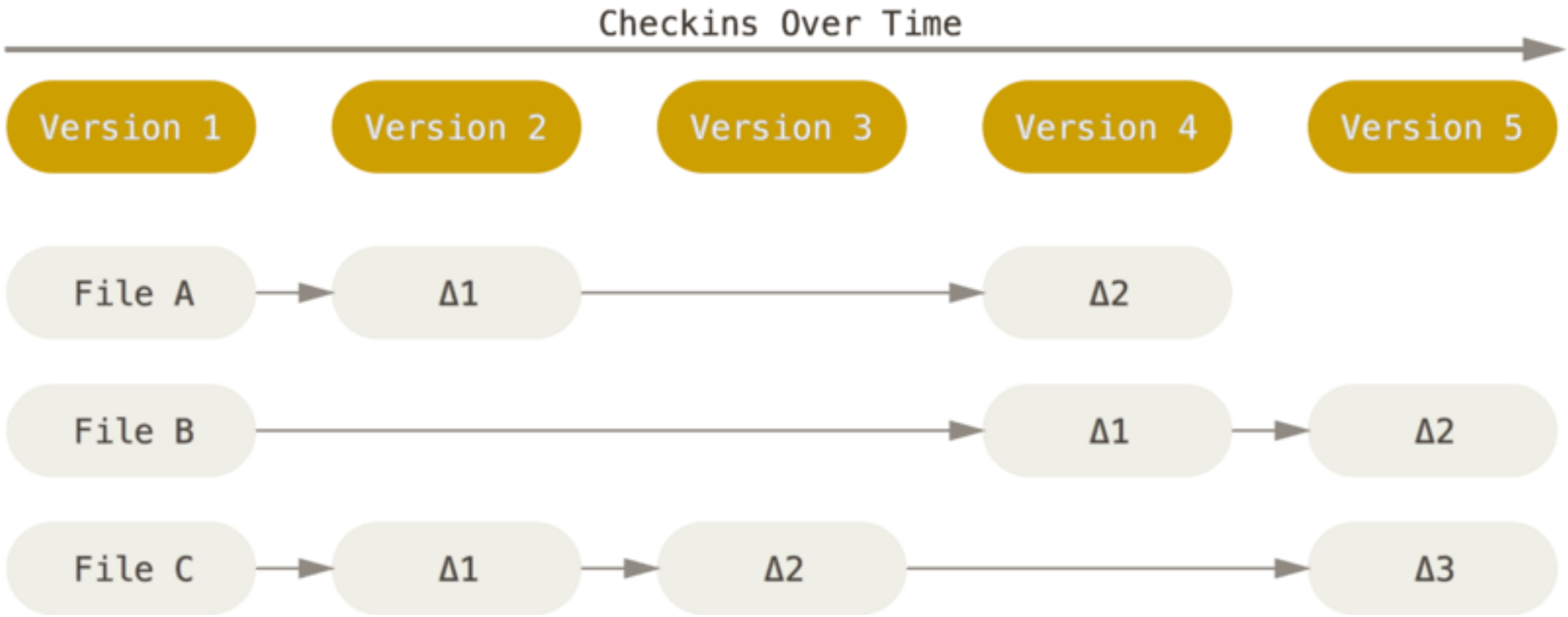
все локально

=> все очень быстро ☺

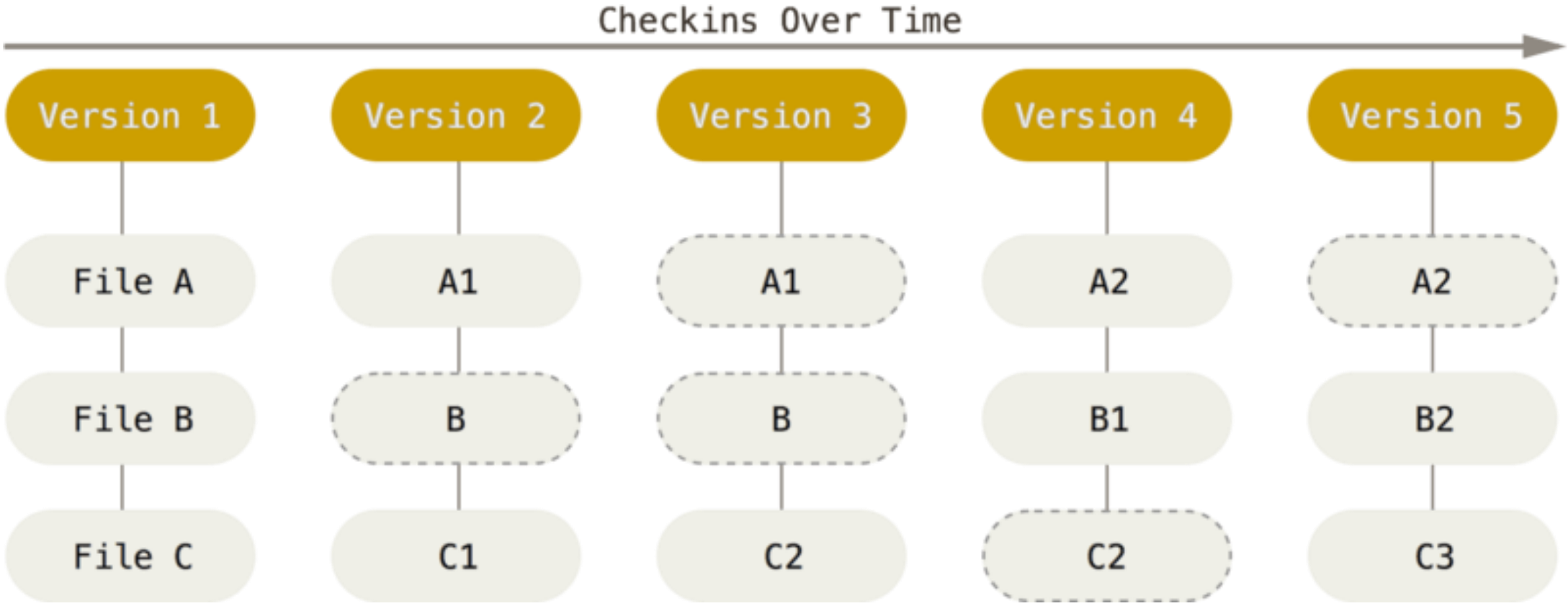
Все локально

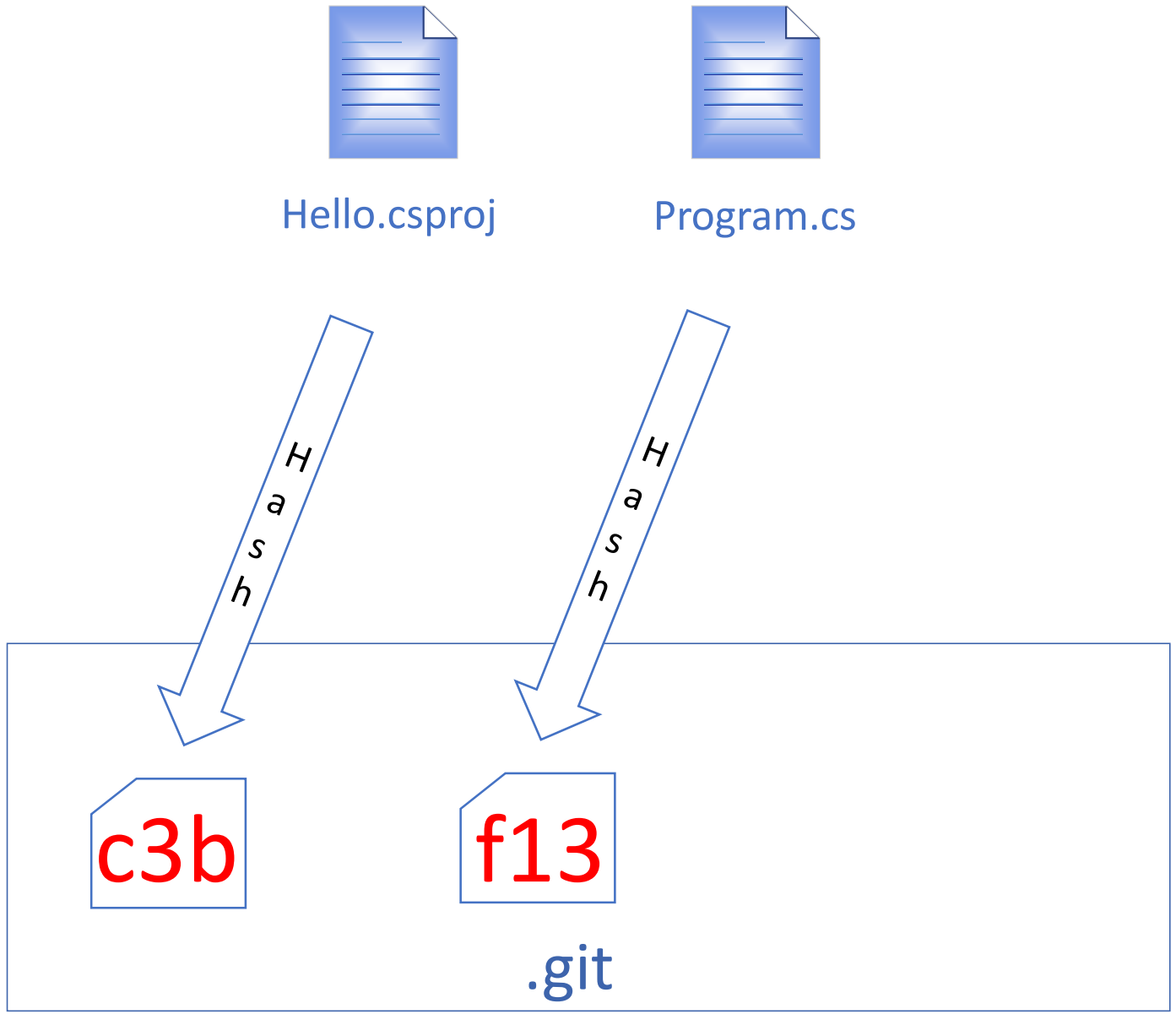
- Просмотр истории
- Коммиты
- Переключение на старые версии
- Переключение на другую ветку
- Сливание веток
- Просмотр изменений

Обычная система контроля версий

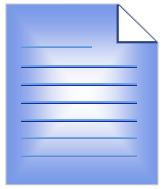


Git


Checkins Over Time
Version 1
Version 2
Version 3
Version 4
Version 5
File A
File B
File C
A1
B
C1
A1
B
C2
A2
B1
C2
A2
B2
C3

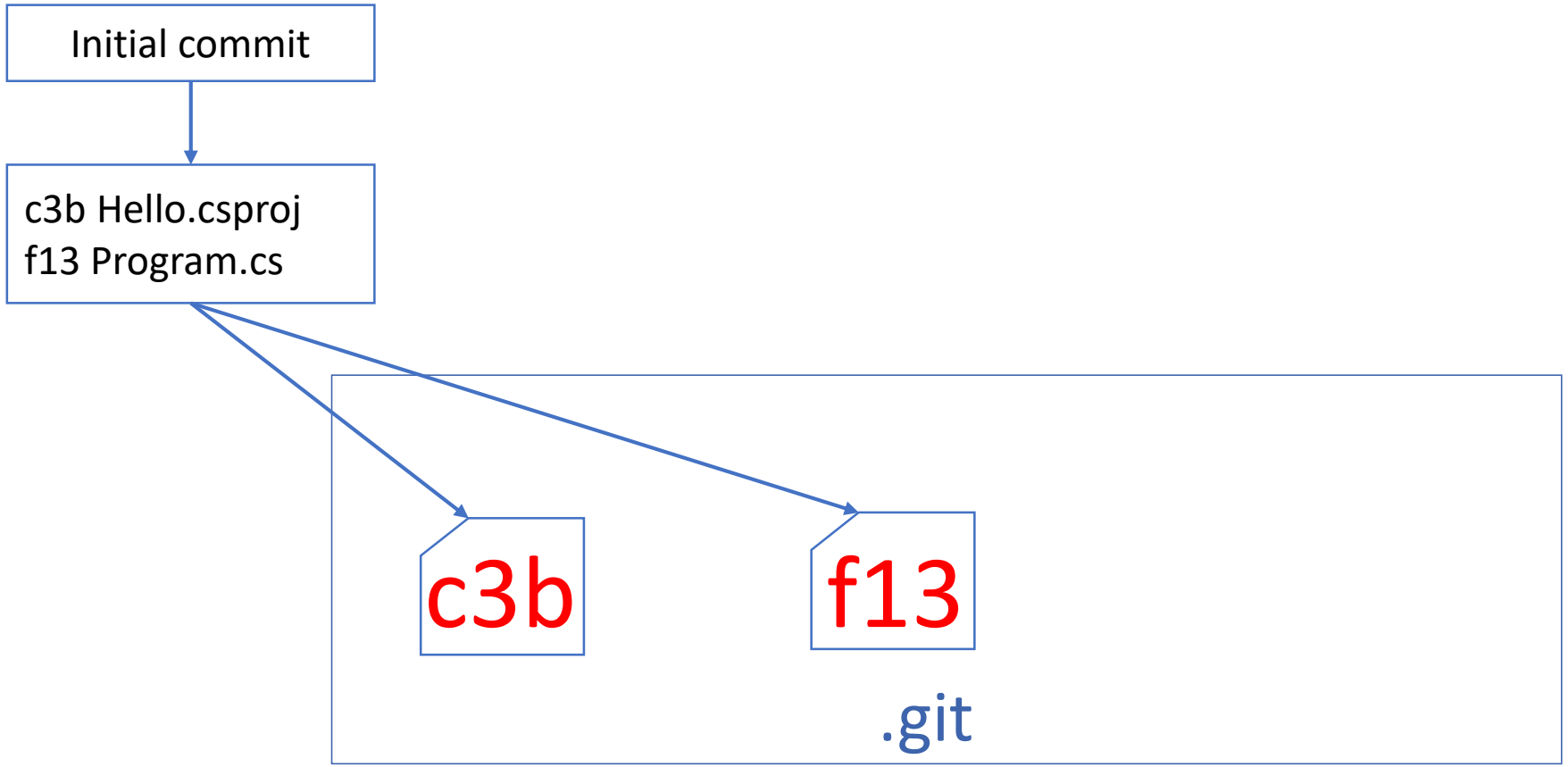

Hello.csproj
Program.cs
Hash
Hash
c3b
f13
.git

> git commit –m “Initial commit”

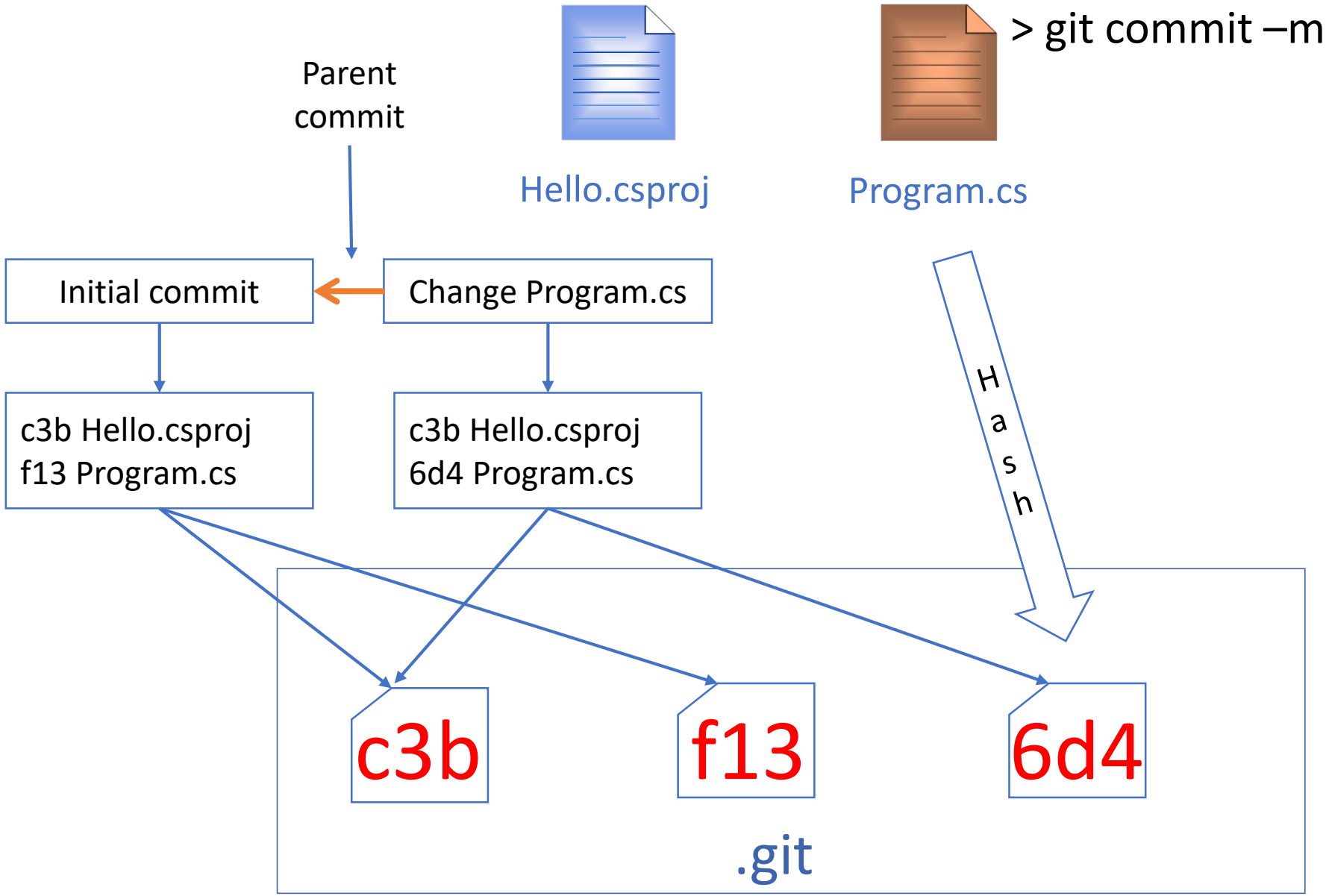


Hello.csproj

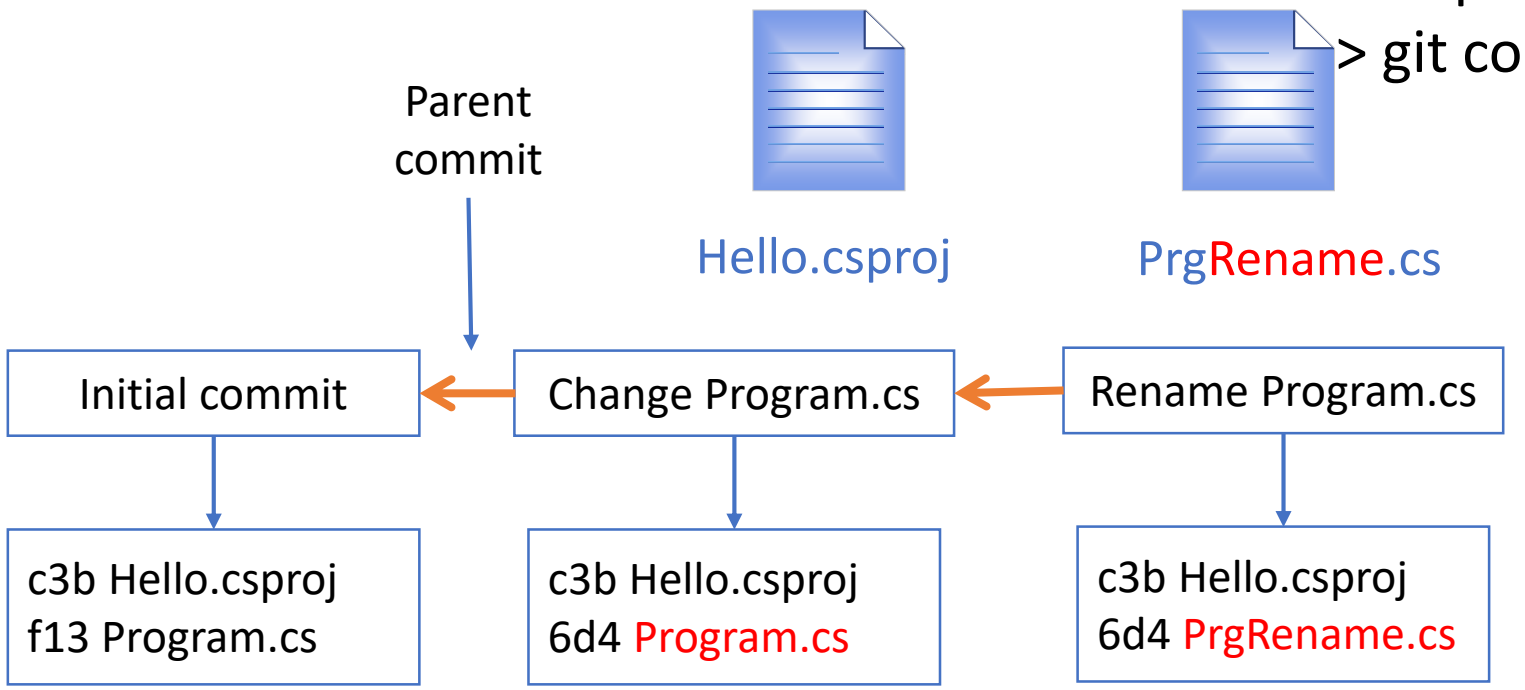
Program.cs

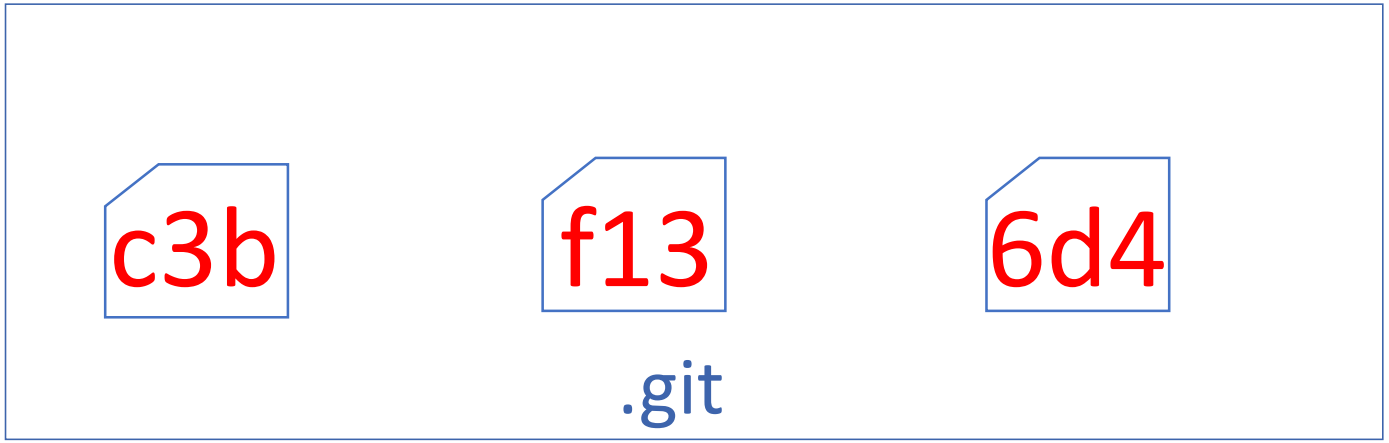


> Изменили Program.cs
> git commit –m “Change Program.cs”



> Переименовали Program.cs
> git commit –m “Rename Program.cs”

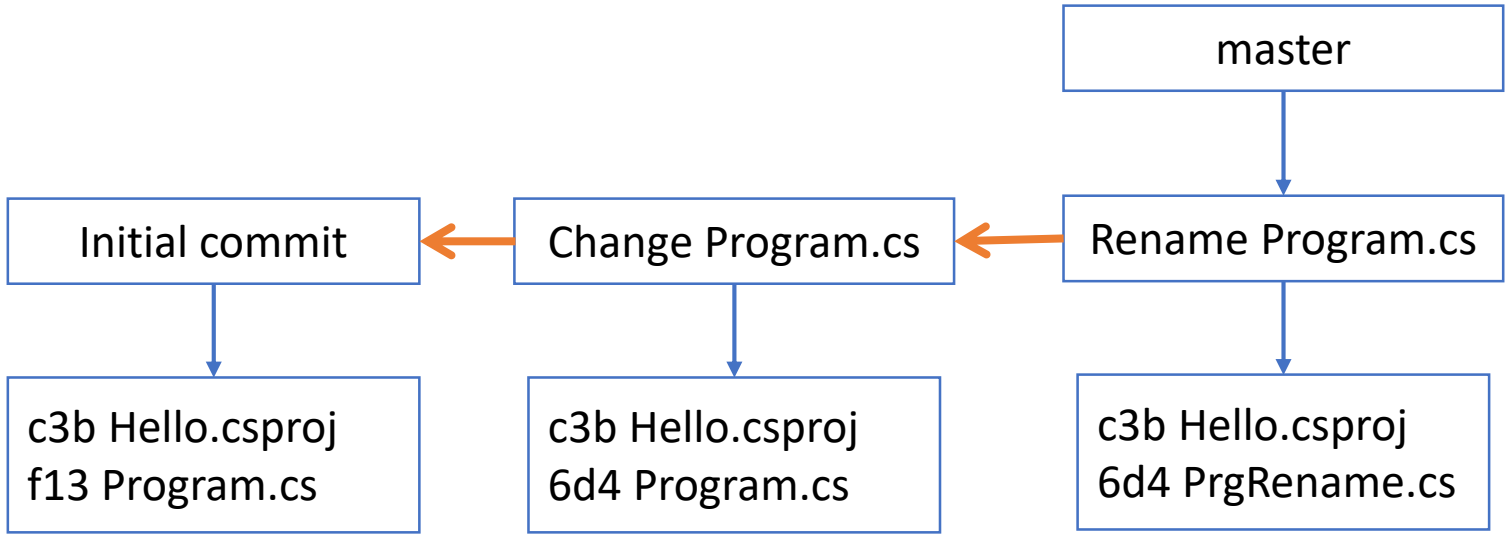


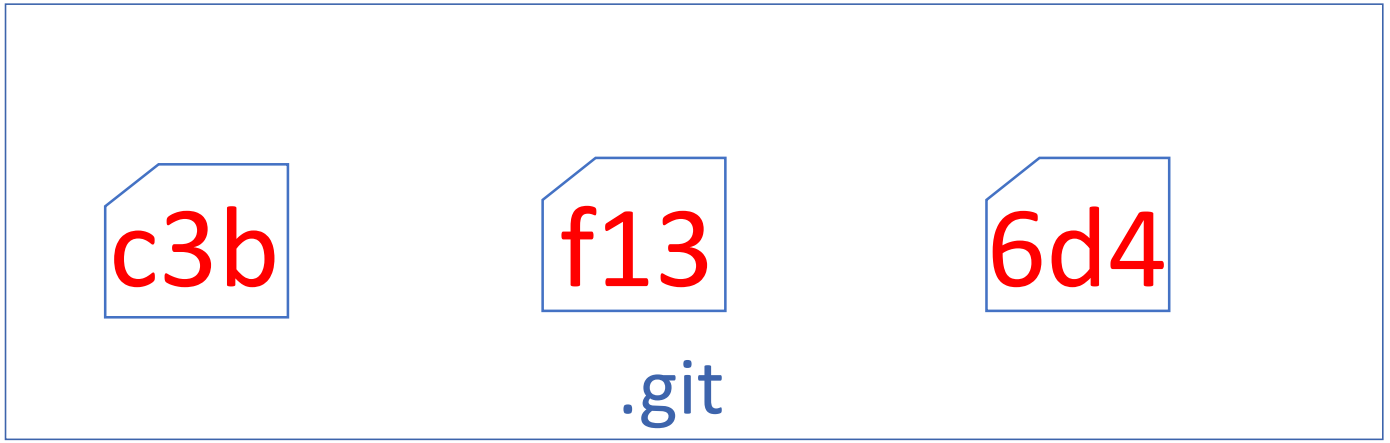


Переименования в Git

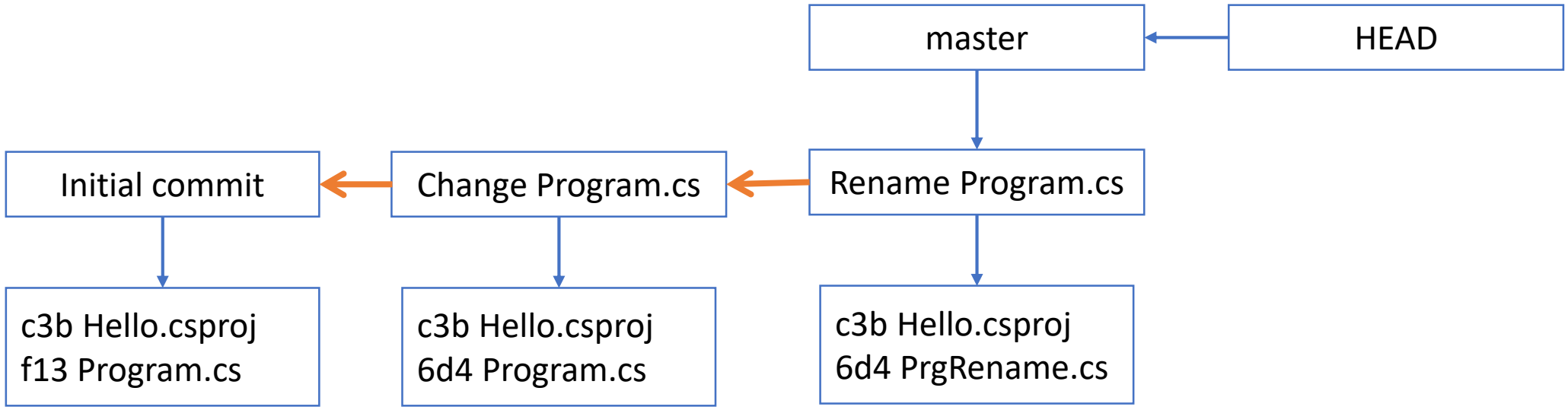
- Для Git их не существует ☺
- Rename определяется неявно, можно указать степень схожести (e.g. 80%)
- Program.cs -> PrgRename.cs и поменялось 10% файла
  - Diff может быть показан как
    - Удаление/создание
    - Переименование с изменением

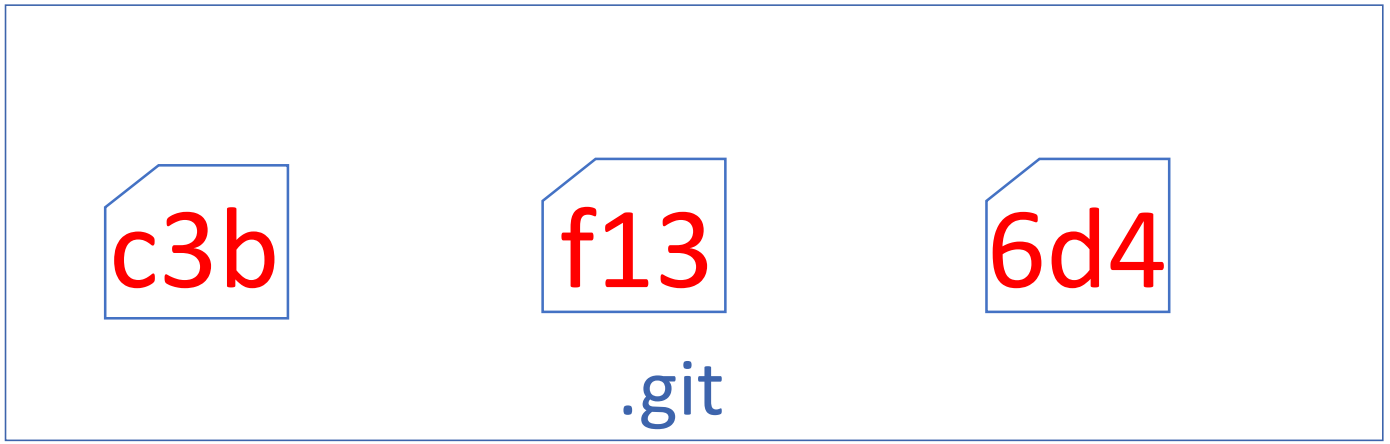
Бранч – указатель на коммит



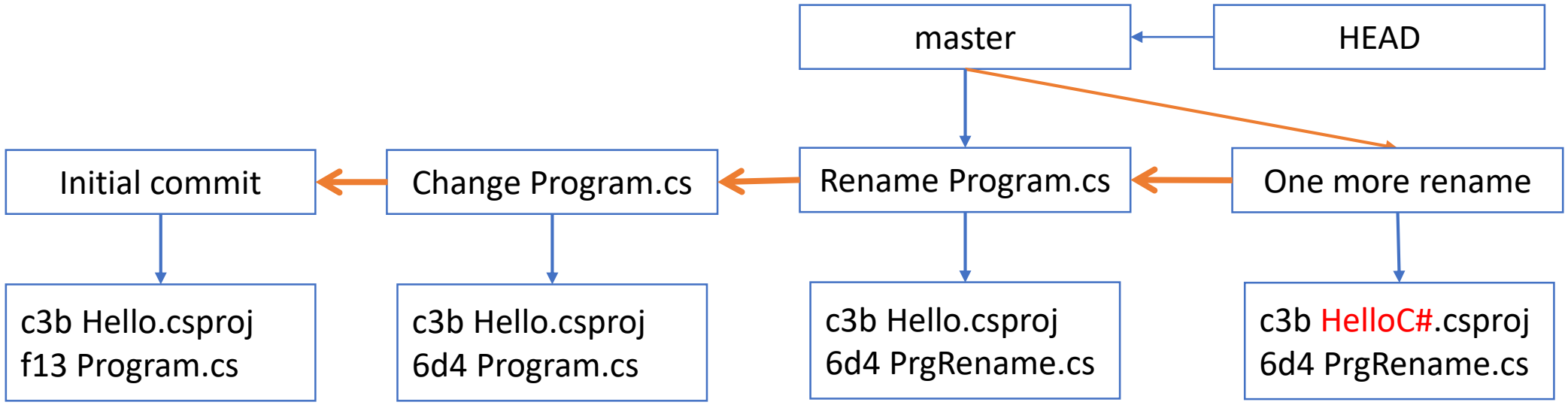


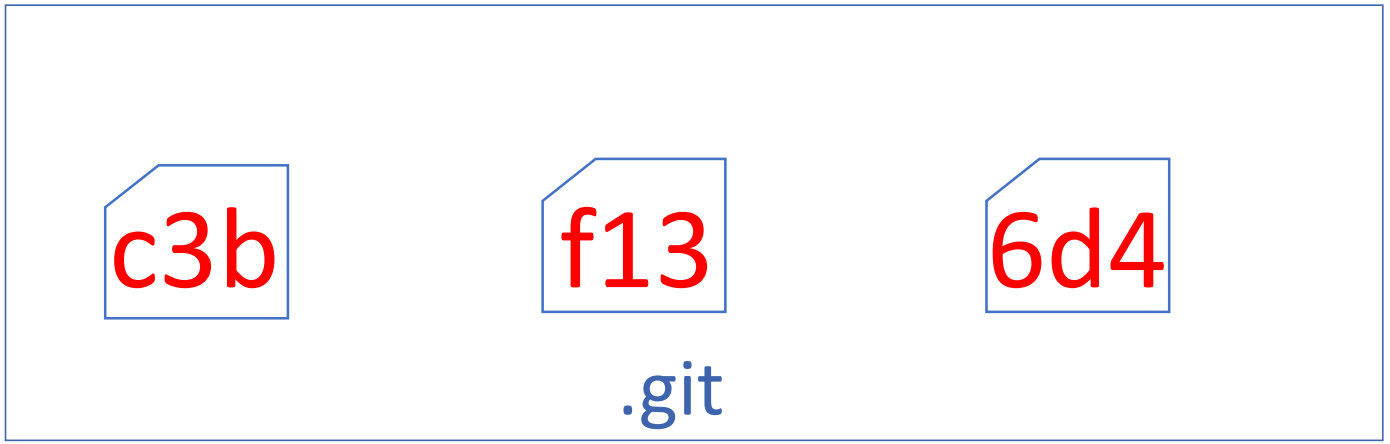
HEAD – указатель на текущий бранч





git commit


master
HEAD
Initial commit
Change Program.cs
Rename Program.cs
One more rename
c3b Hello.csproj
f13 Program.cs
c3b Hello.csproj
6d4 Program.cs
c3b Hello.csproj
6d4 PrgRename.cs
c3b HelloC#.csproj
6d4 PrgRename.cs


c3b
f13
6d4
.git

Index или Staging Area

- Поменяли что-то в рабочей директории
- Хотим коммитить по частям

Что делать?

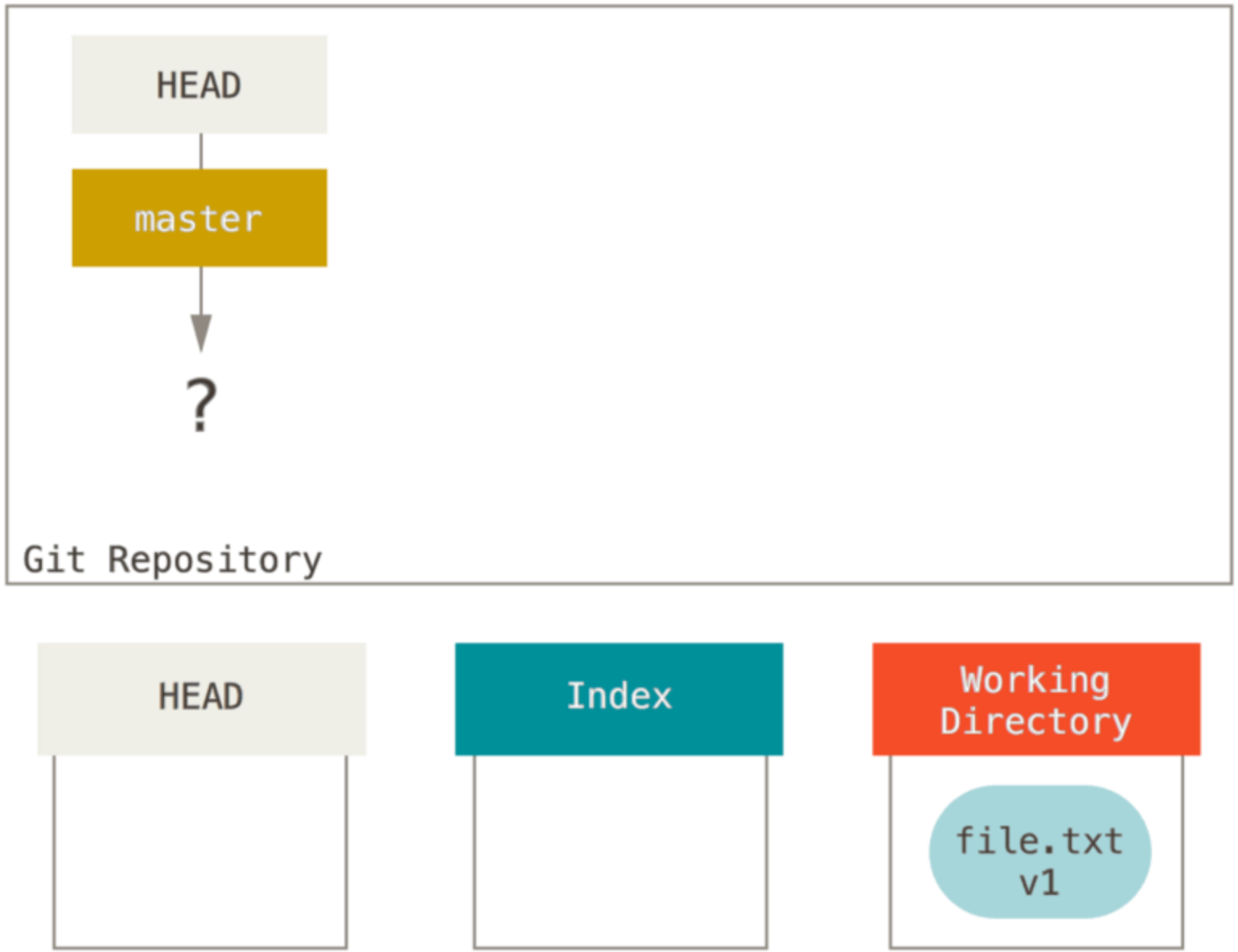
- git add – говорит git *“я хочу видеть текущее состояние этого файла в следующем коммите”*

Index или Staging Area

- Всего лишь файл (*.git/index*)
- По сути, снэпшот вашей директории, который будет следующим коммитом

Хочу просто все закоммитить!

- git commit --all --message “Skip index”
  - == git commit -am “Skipp index”


HEAD
master
?
Git Repository
HEAD
Index
Working Directory
file.txt
v1


HEAD
master
?
Git Repository


HEAD
Index
Working Directory
file.txt v1
file.txt v1
git add


HEAD
master
eb43bf8
file.txt
v1
Git Repository


HEAD
eb43bf8
file.txt
v1
Index
file.txt
v1
Working
Directory
file.txt
v1

git commit

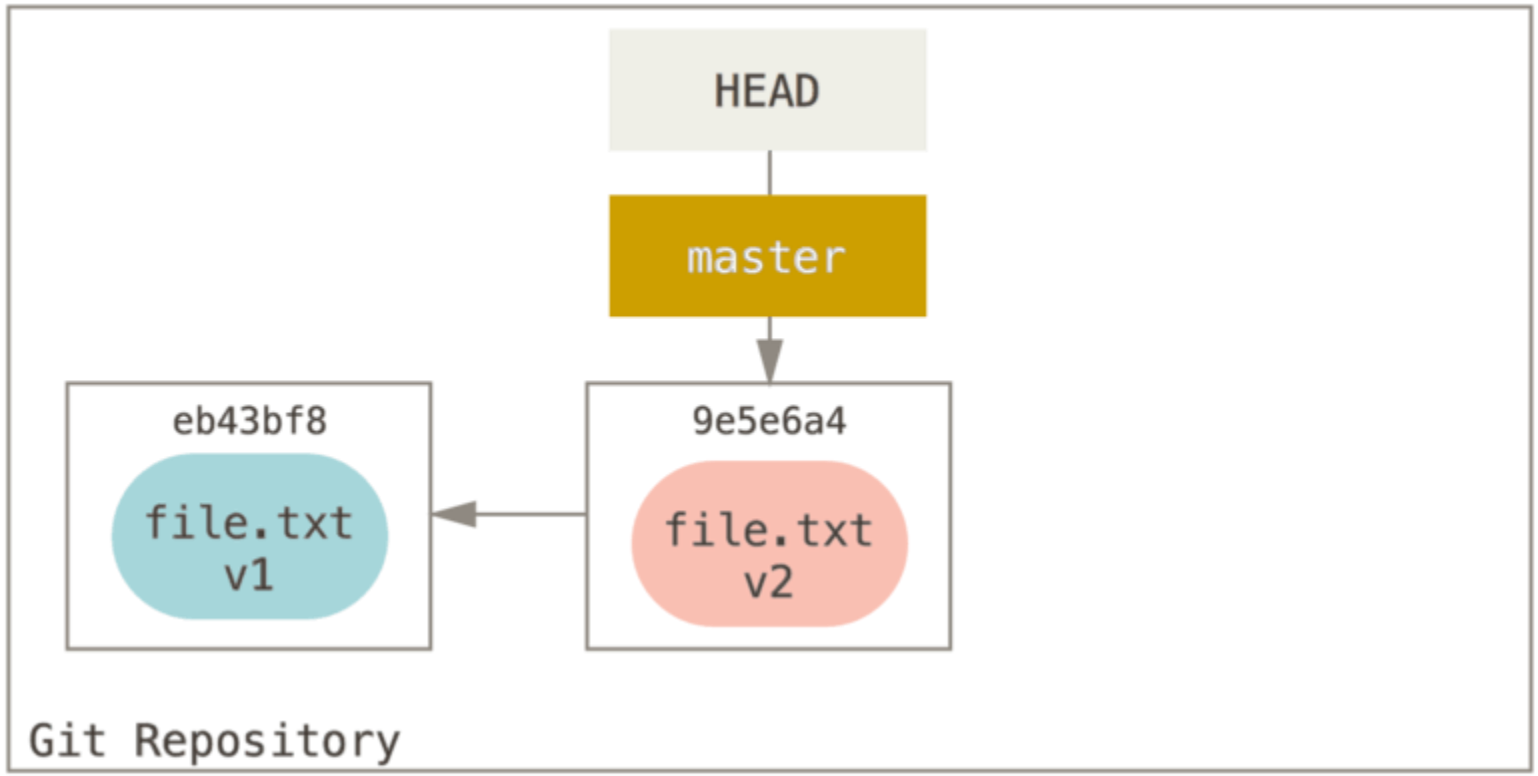

HEAD
master
eb43bf8
file.txt
v1
Git Repository


HEAD
eb43bf8
file.txt
v1
Index
file.txt
v1
Working
Directory
file.txt
v2

edit file


HEAD
master
eb43bf8
file.txt
v1
Git Repository


HEAD
eb43bf8
file.txt
v1
Index
file.txt
v2
Working
Directory
file.txt
v2
git add


HEAD
master
eb43bf8
file.txt
v1
9e5e6a4
file.txt
v2
Git Repository


HEAD
9e5e6a4
file.txt
v2
Index
file.txt
v2
Working
Directory
file.txt
v2

git commit

Demo: .git, index, add, commit

Утилиты

- GUI: SourceTree, GitKraken, Tower, Git Extensions, …
  - https://git-scm.com/downloads/guis/
- Visual Studio
- PowerShell

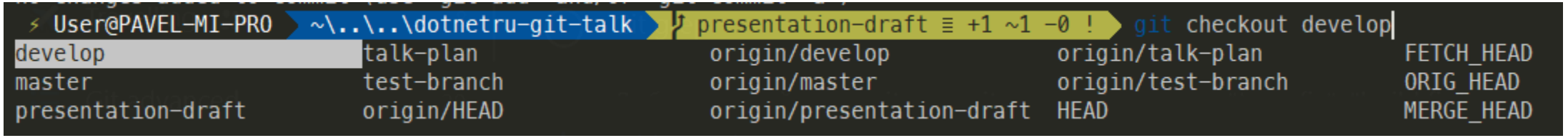
Powershell во имя добра

Настройка (лайт): PowerShell + Posh-Git

Настройка (хардкор): https://gist.github.com/agnoster/3712874

Фичи:

- Текущий статус
- Автодополнение бранчей/команд



Игнорирование файлов

.gitignore

https://www.gitignore.io

Unstage изменений

- Изменили файлы в рабочей директории
- Добавили их в индекс (git add), stage в Visual Studio
- Поняли, что добавили не то ☹

Что делать?

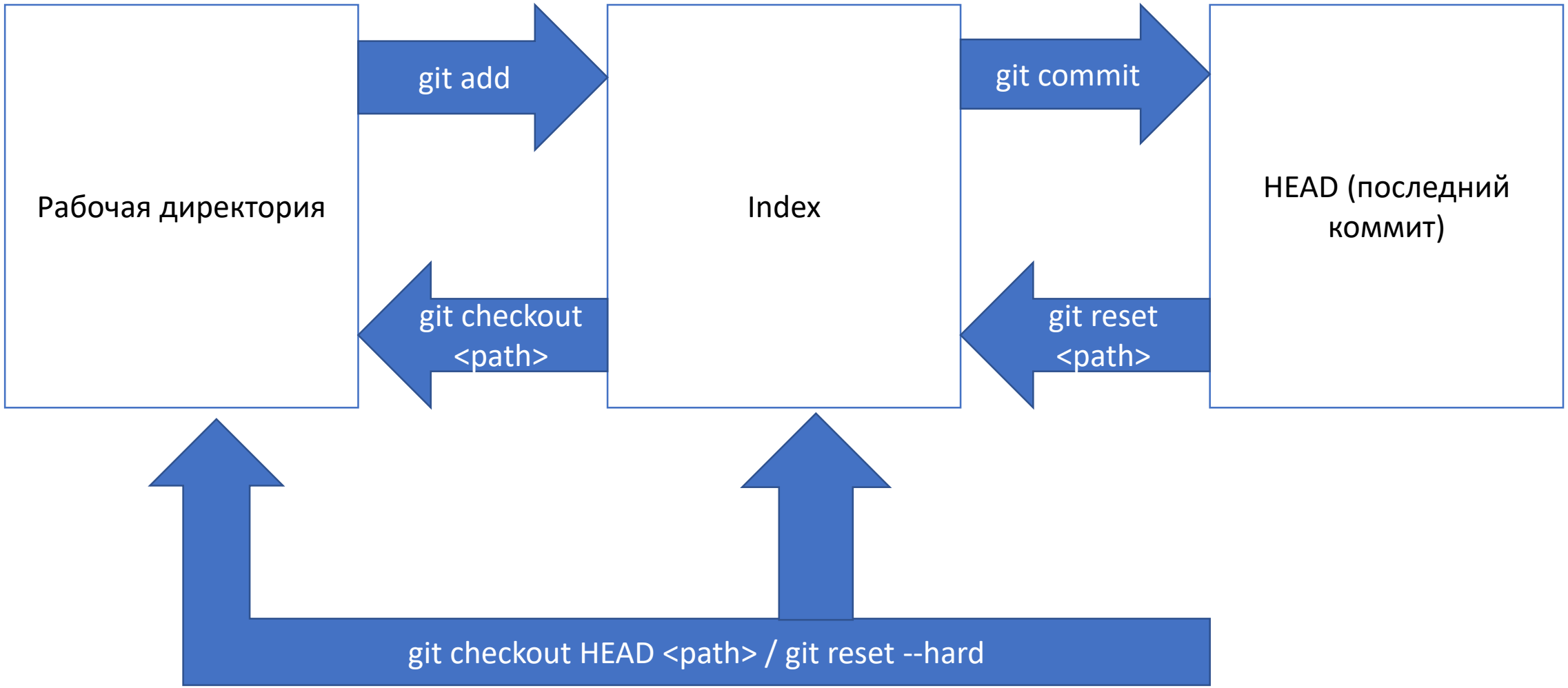
- Unstage в Visual Studio, под капотом git reset <path>

Откат изменений

- Изменили файлы в рабочей директории
- Поняли, что изменили не то ☹

Что делать?

- Undo changes в Visual Studio, под капотом git checkout <path>


Рабочая директория
git add
Index
git commit
HEAD (последний коммит)
git checkout <path>
git reset <path>
git checkout HEAD <path> / git reset --hard

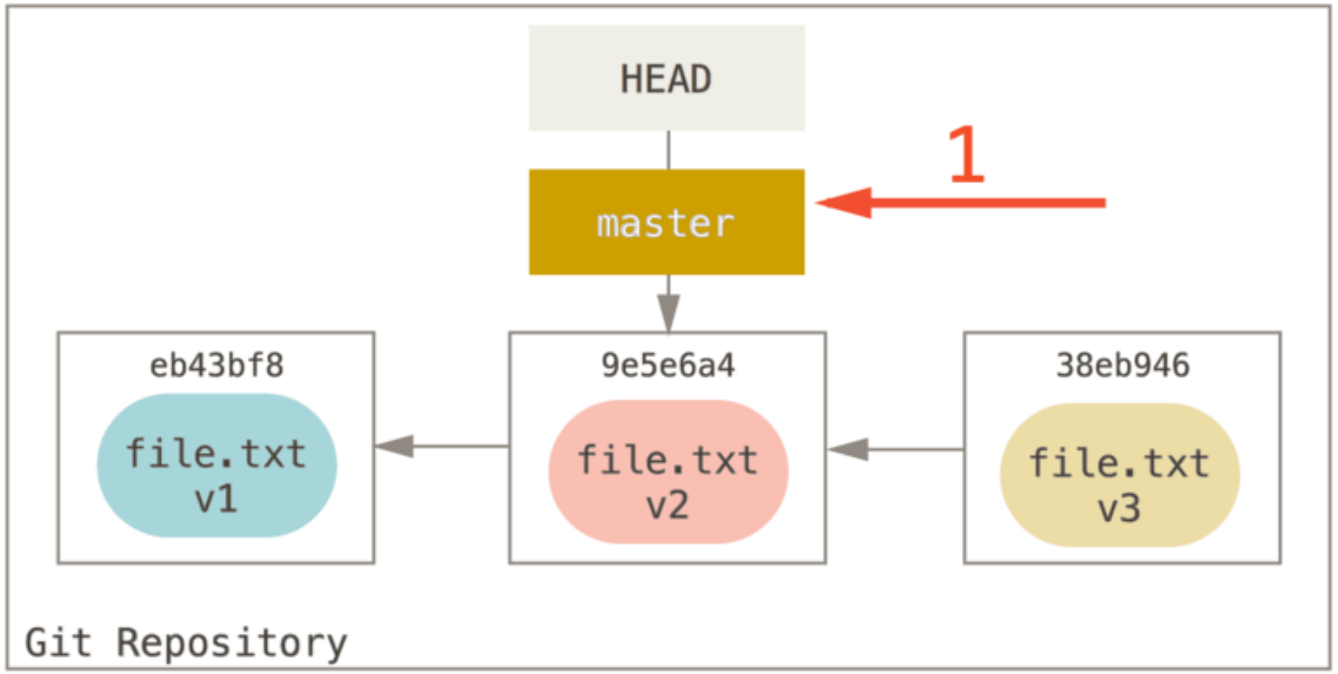
Изменение последнего коммита

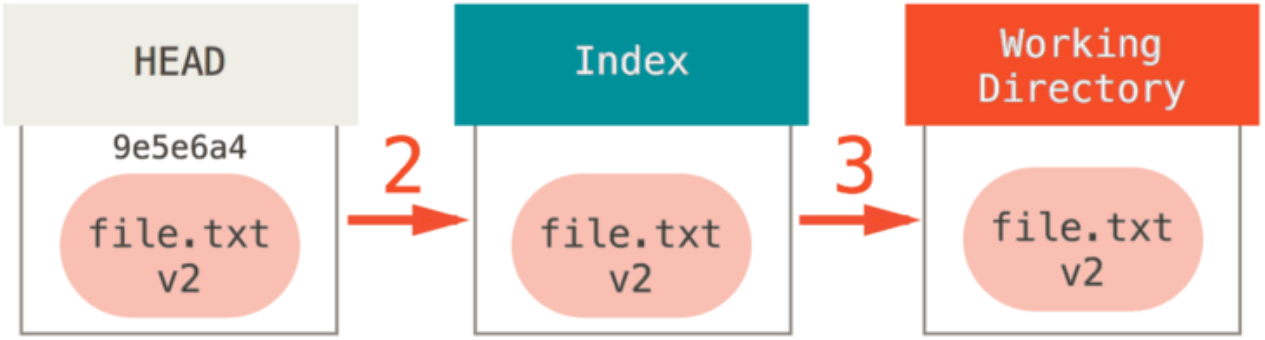
- Изменили файлы в рабочей директории
- Закоммитили
- Поняли, что закоммитили не то ☹

Что делать?

- SourceTree: reset current branch to commit
- git reset HEAD^ - изменит указатель (бранч) на пред. коммит, оставит изменения в рабочей директории
- git reset –hard HEAD^ - изменит указатель, откатит изменения

Удаление последнего коммита






```
git reset --hard HEAD~
```

Удаление untracked файлов

- Добавили новые файлы
- Поняли, что добавили не то ☹

Что делать?

- git clean
  - -x (удаление игнорируемых файлов)
  - -f (force), или проставить *clean.requireForce true*
  - -d (включать папки)

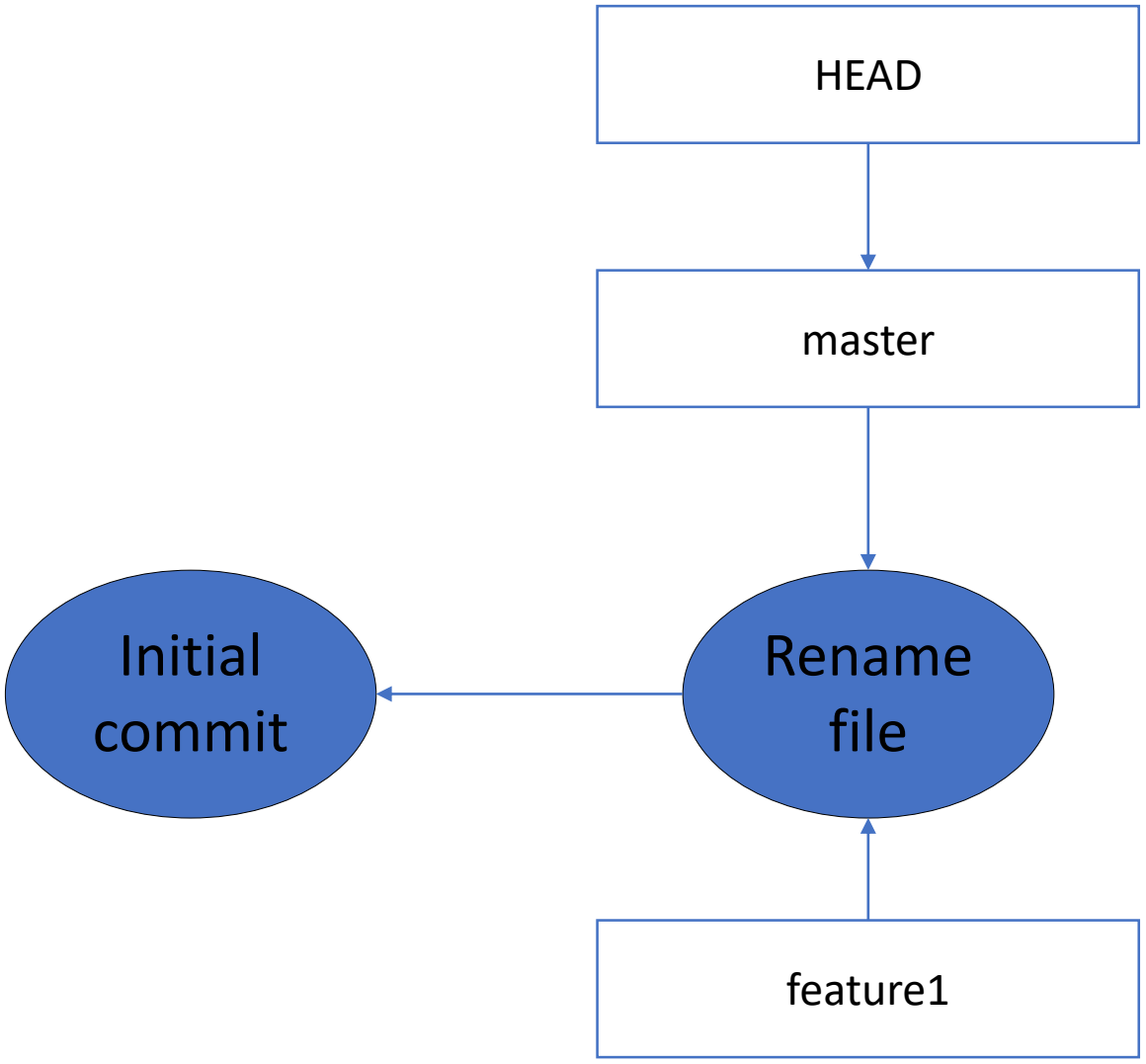
- “git clean –xfd” – удалить все, чего нету в source control
  - По сути, сделать clean build
  - Самая частая команда при разработке на Xamarin ☺

Сохранение изменений без коммита

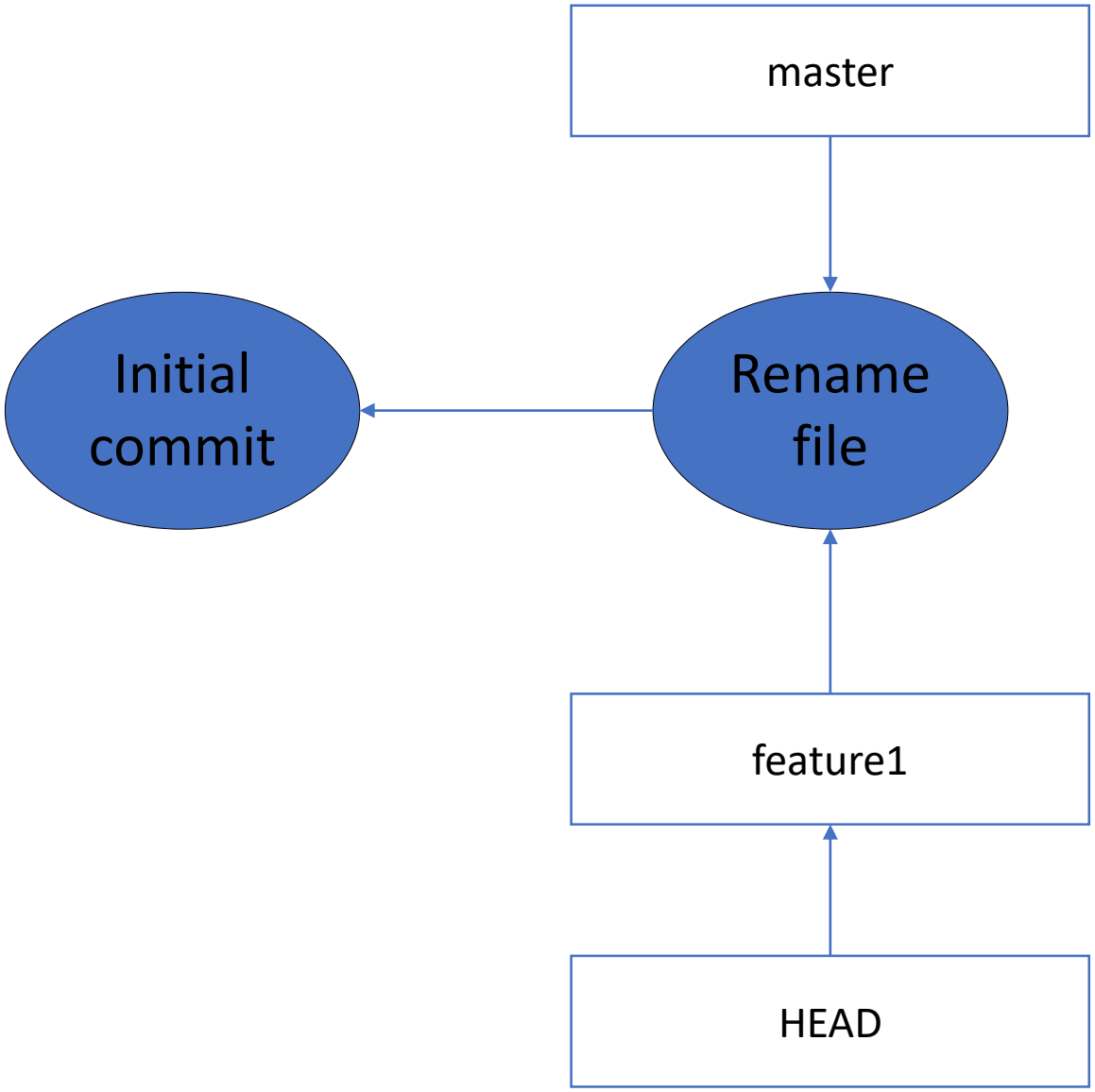
- Сделали изменения, но еще не коммитили
- Нужно срочно переключиться на другую ветку (e.g. бага в PROD)

- Что делать?
  - git stash push –m “Add new awesome feature” --include-untracked (**!!!**)
- Применить изменения
  - GUI удобнее для просмотра списка stash и контента
  - git stash list
  - git stash pop|apply

> git branch feature1



> git checkout feature1



git checkout

1. Перемещает HEAD на указанную ветку
2. Обновляет index и рабочую директорию
3. Безопасно для рабочей директории, т.к. будет сделан merge

- => Рабочая директория одна, просто достаем нужную версию из локальной базы (.git)

- *git checkout –b <newbranch>* - создание новой ветки и сразу переключение на нее

Одновременная работа с несколькими бранчами

- Нужно одновременно работать с несколькими ветками
- E.g. долго пересобирать и запускать
- E.g. хочется сравнить поведение в разных ветках

Что делать?

- git worktree
- Позволяет сделать checkout любого коммита в отдельную директорию ☺

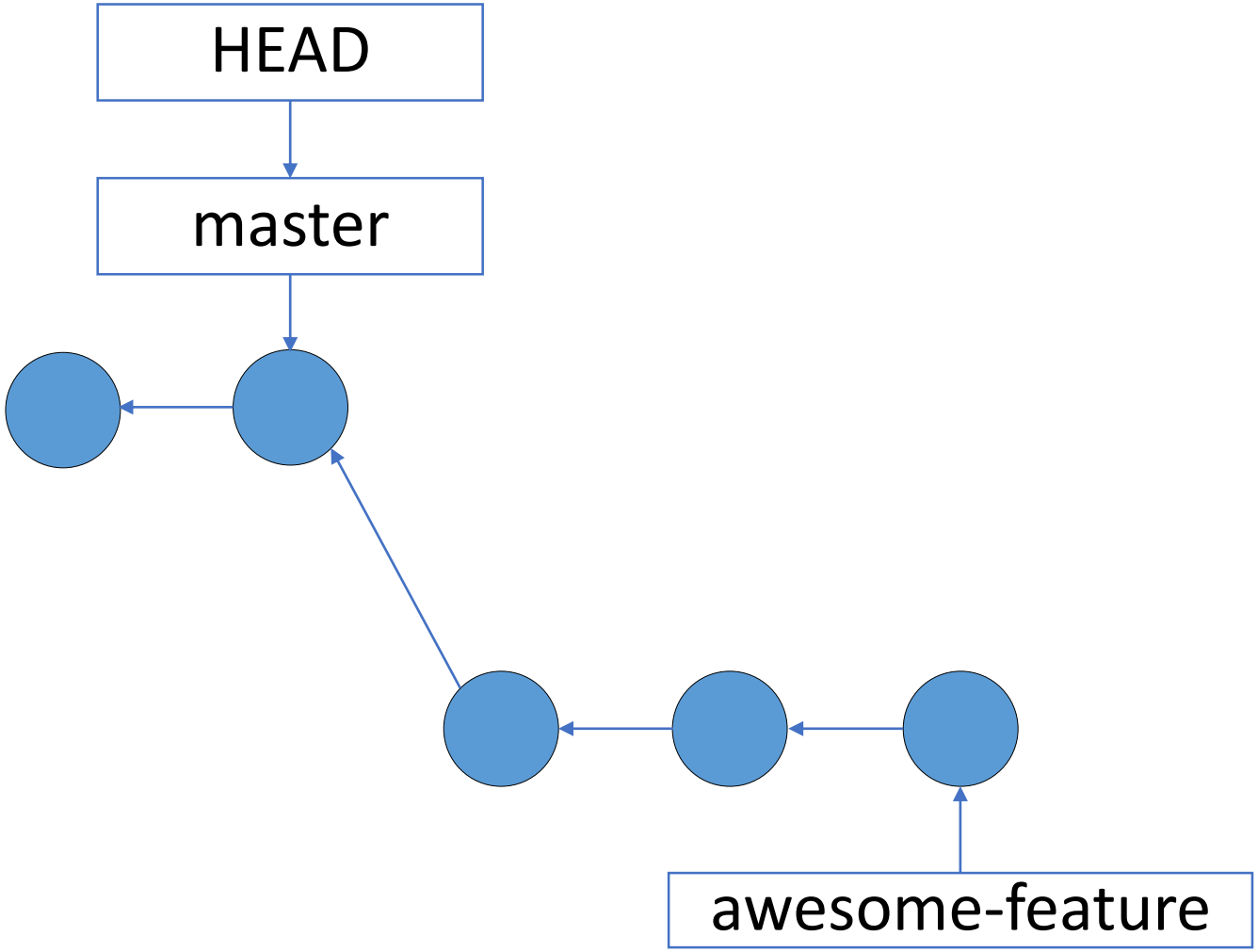
Интеграция бранчей

- Создали feature ветку, что-то там сделали
- Хотим смержить ее в master

Что делать?

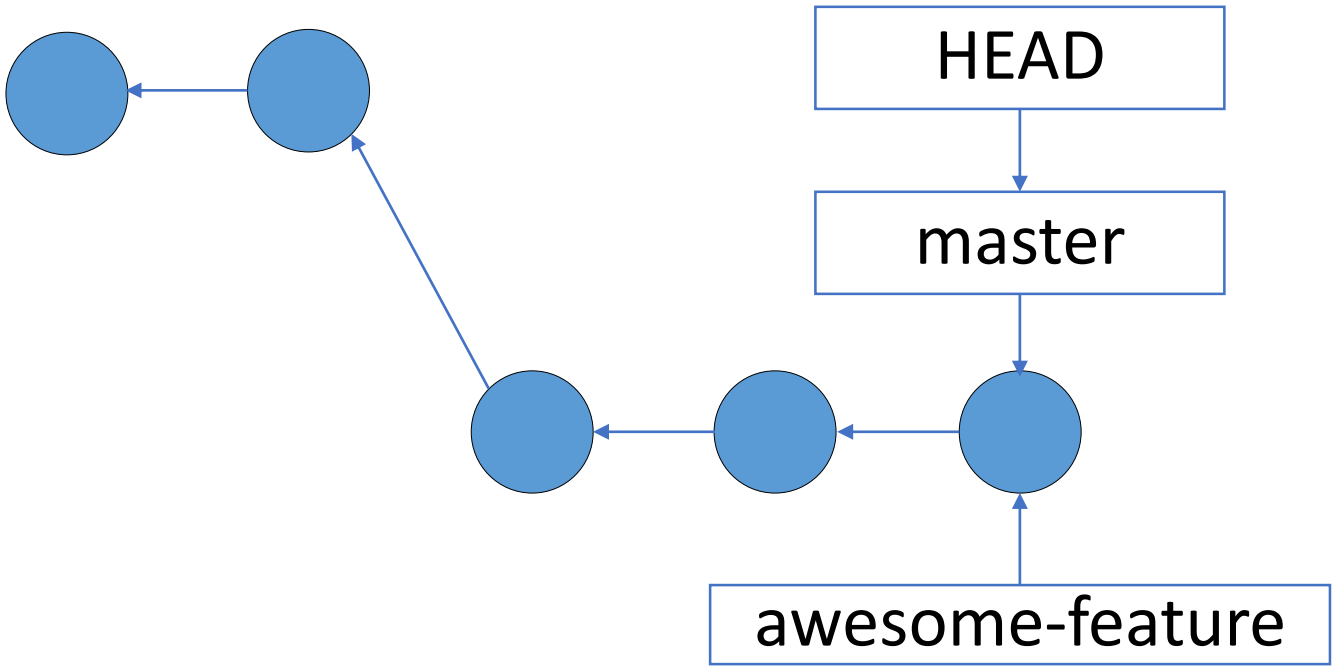
- Git merge
- Git rebase

git merge awesome-feature

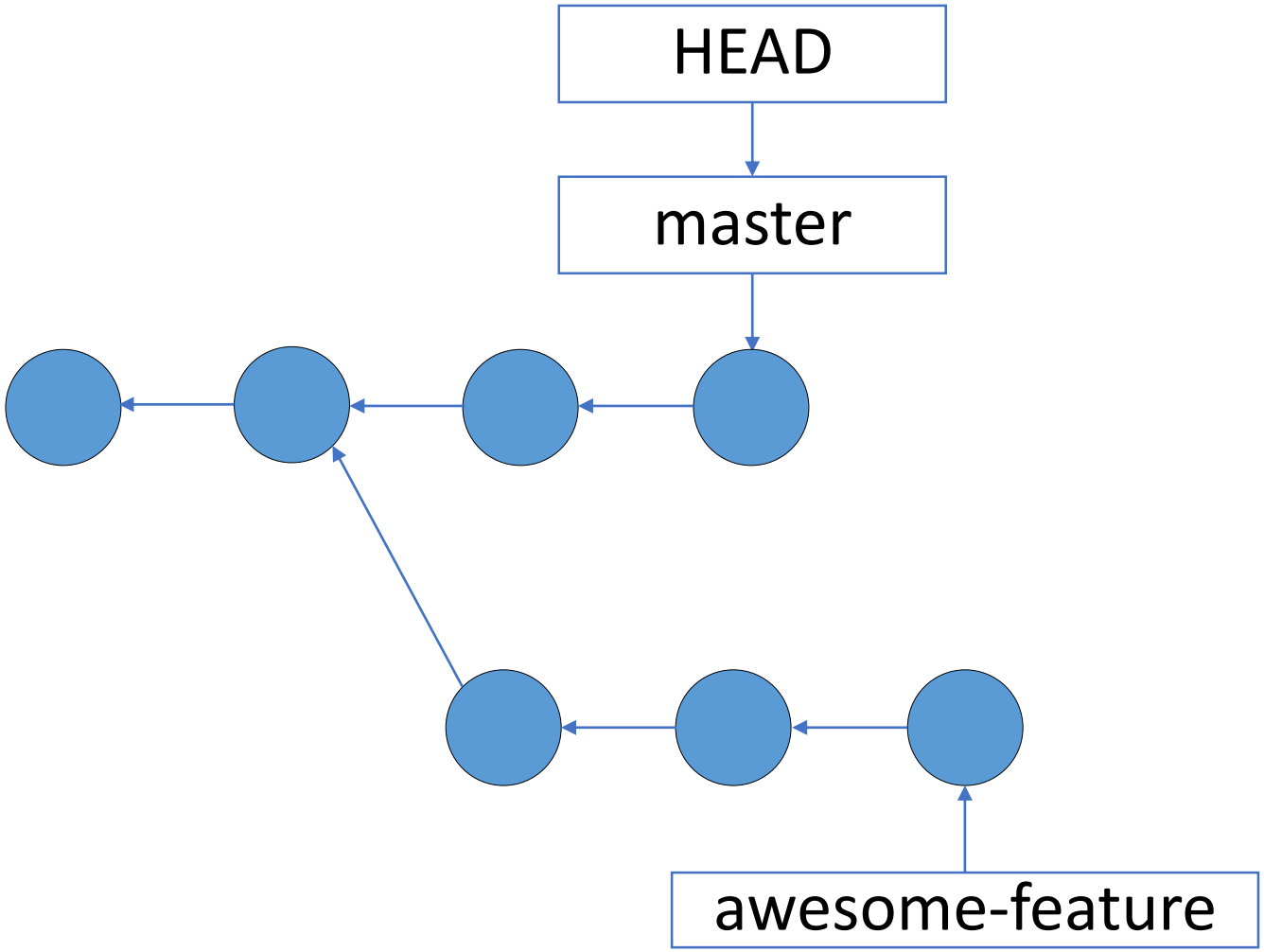


git merge awesome-feature

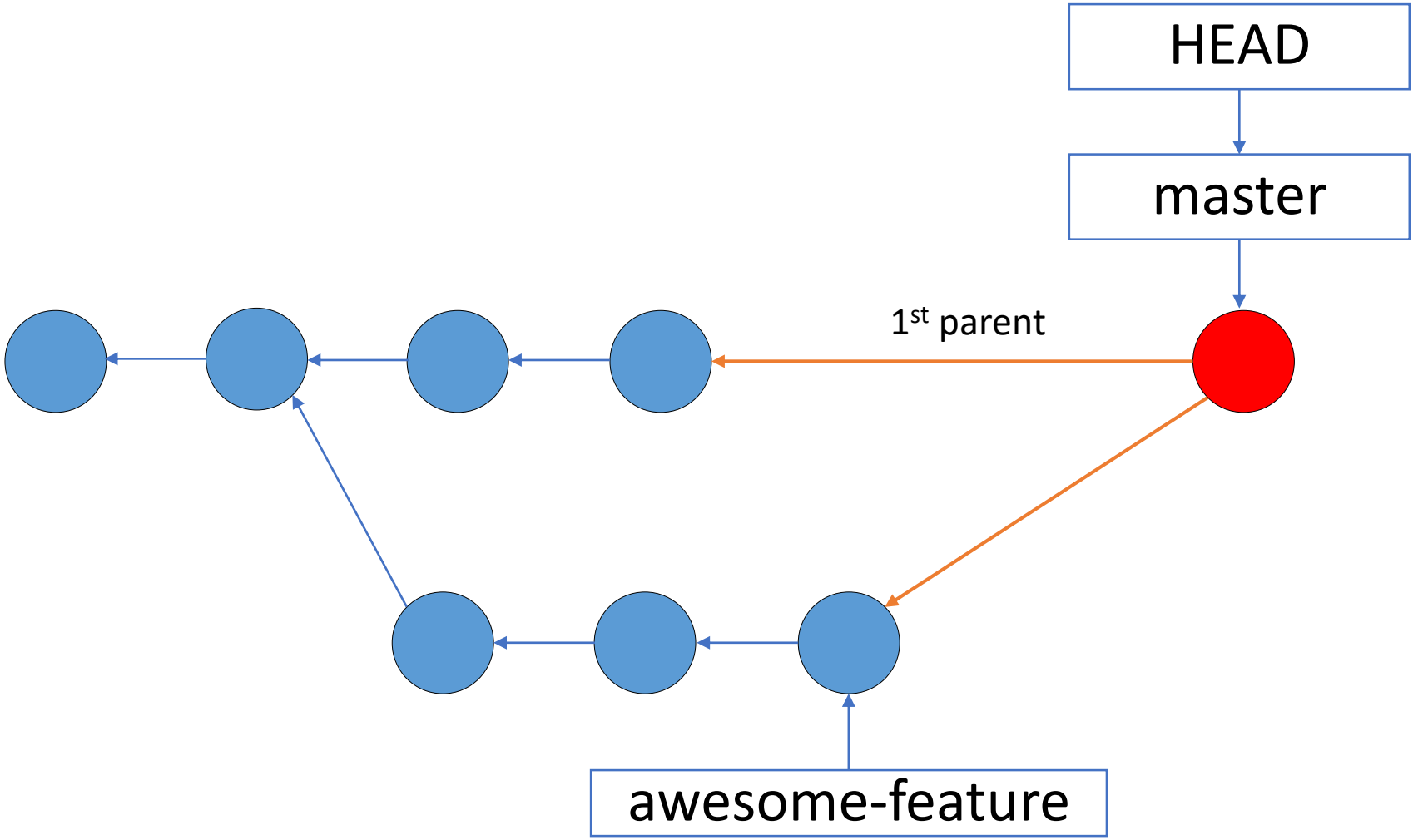
Fast Forward Merge



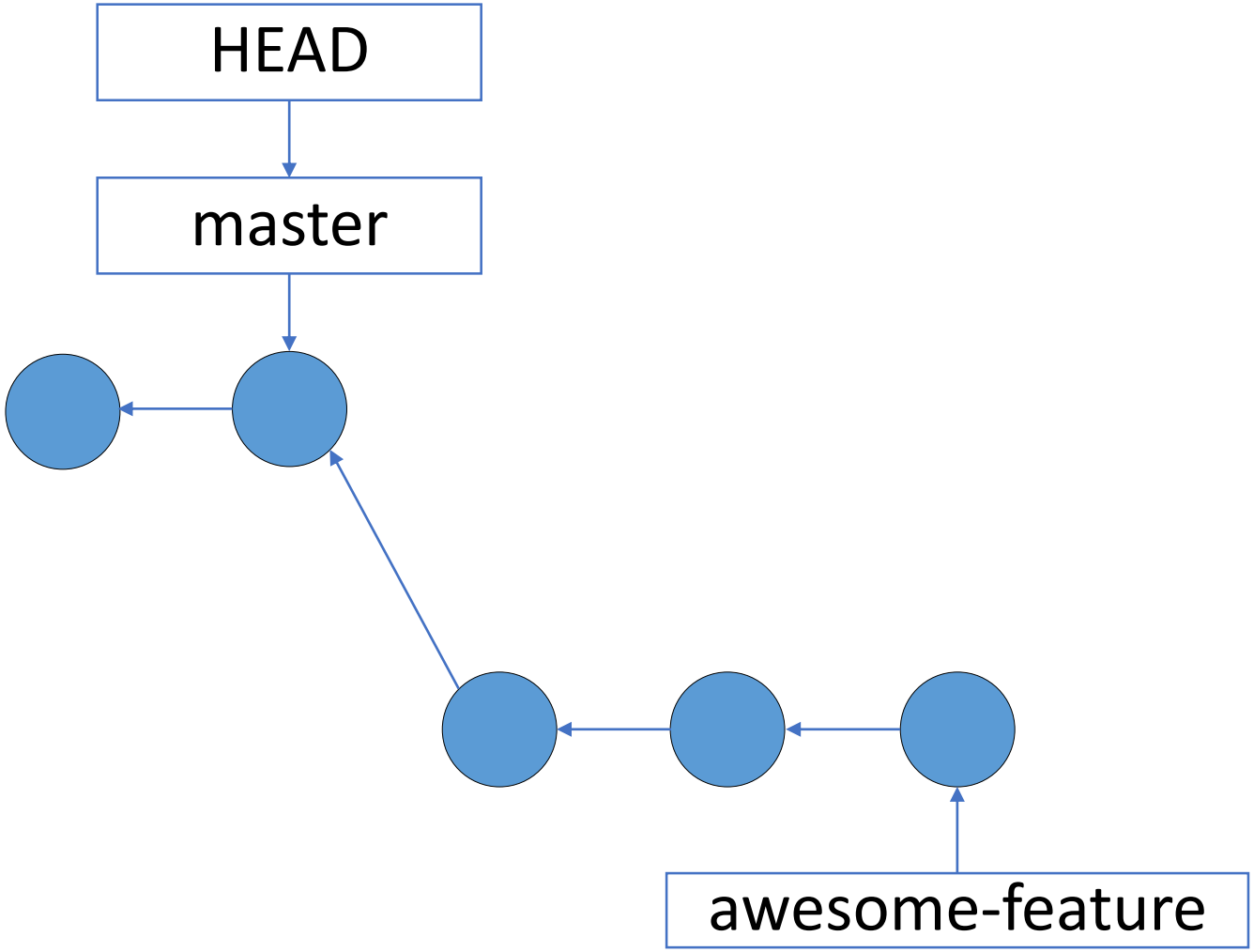
git merge awesome-feature



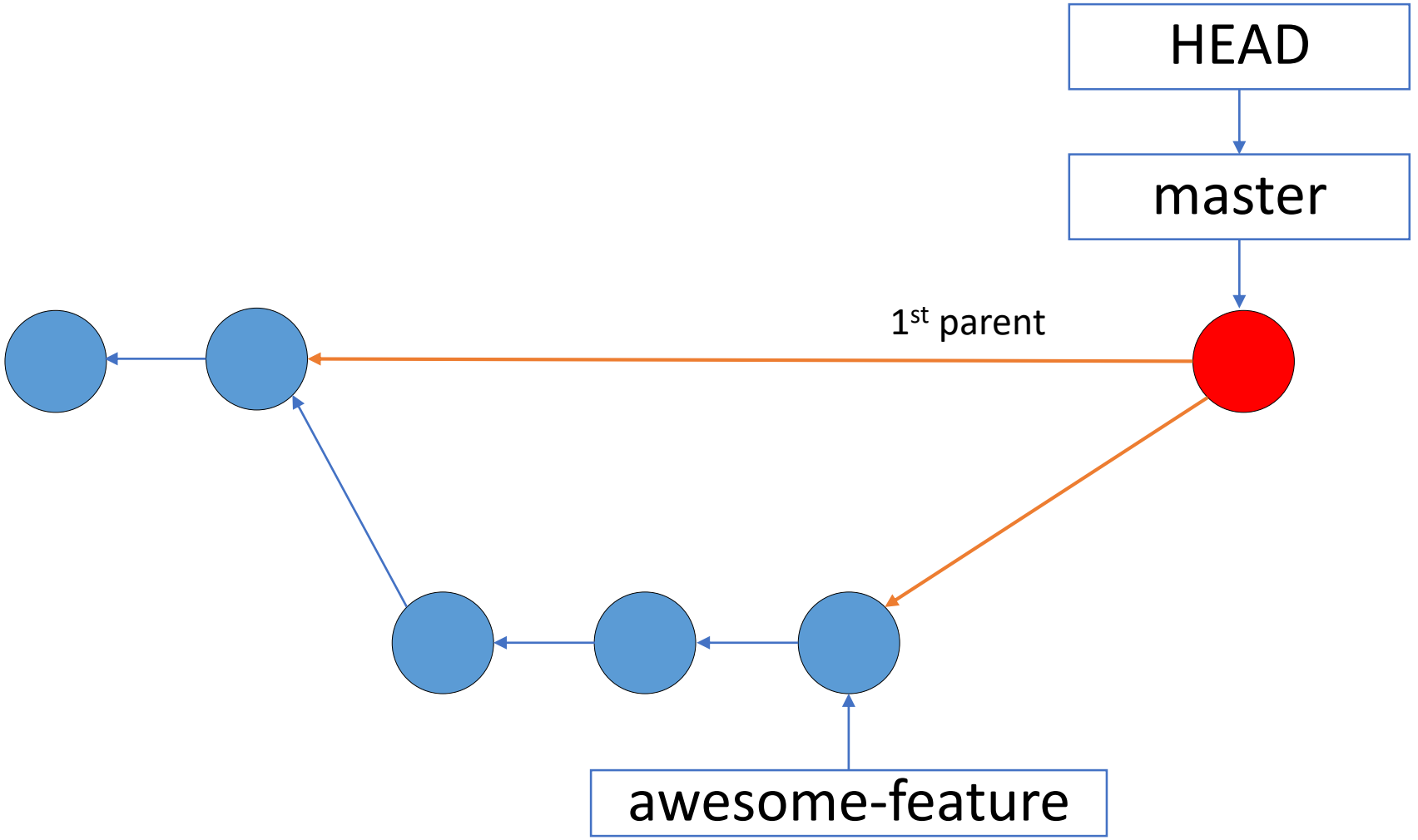
git merge awesome-feature



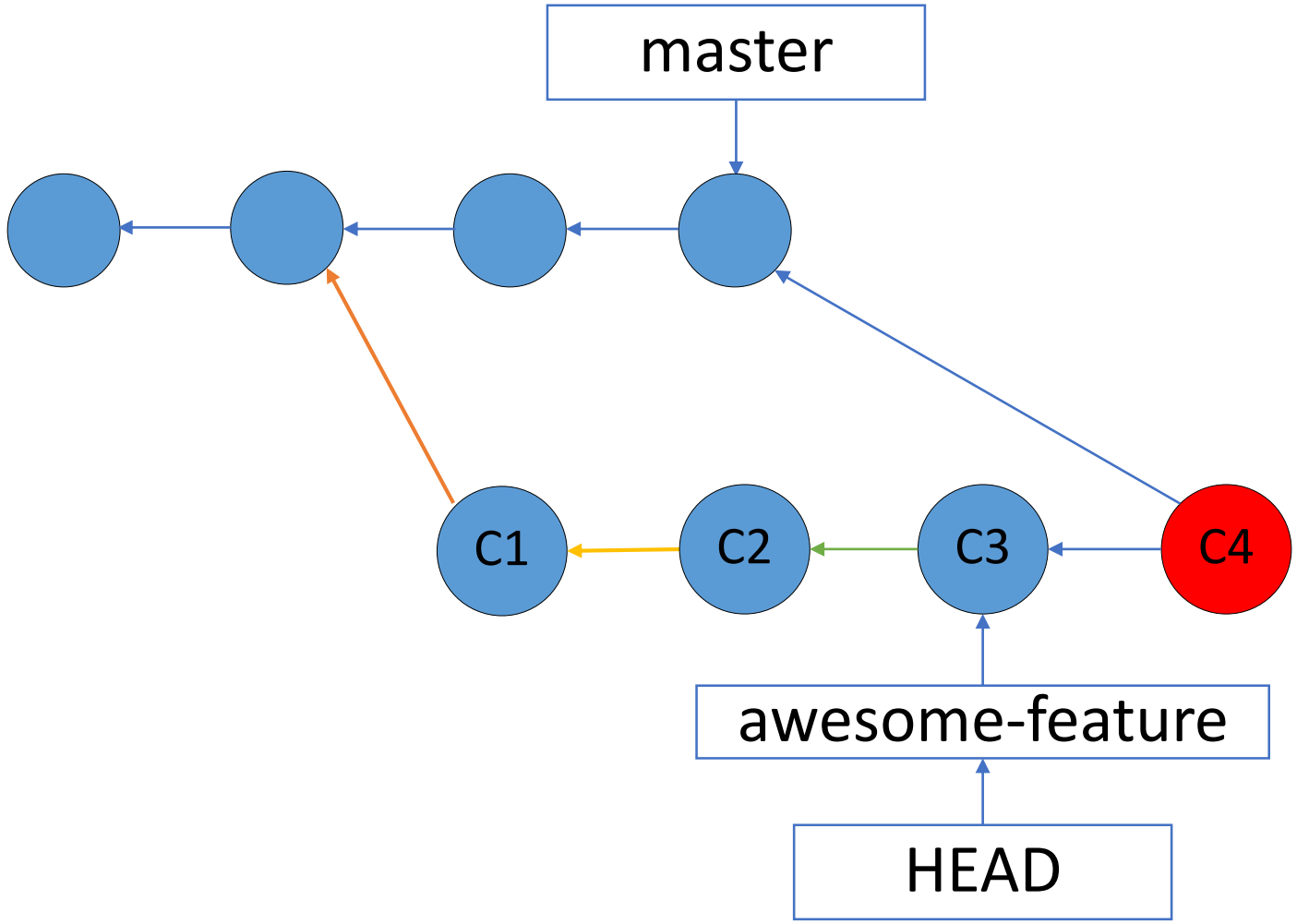
git merge awesome-feature --no-ff

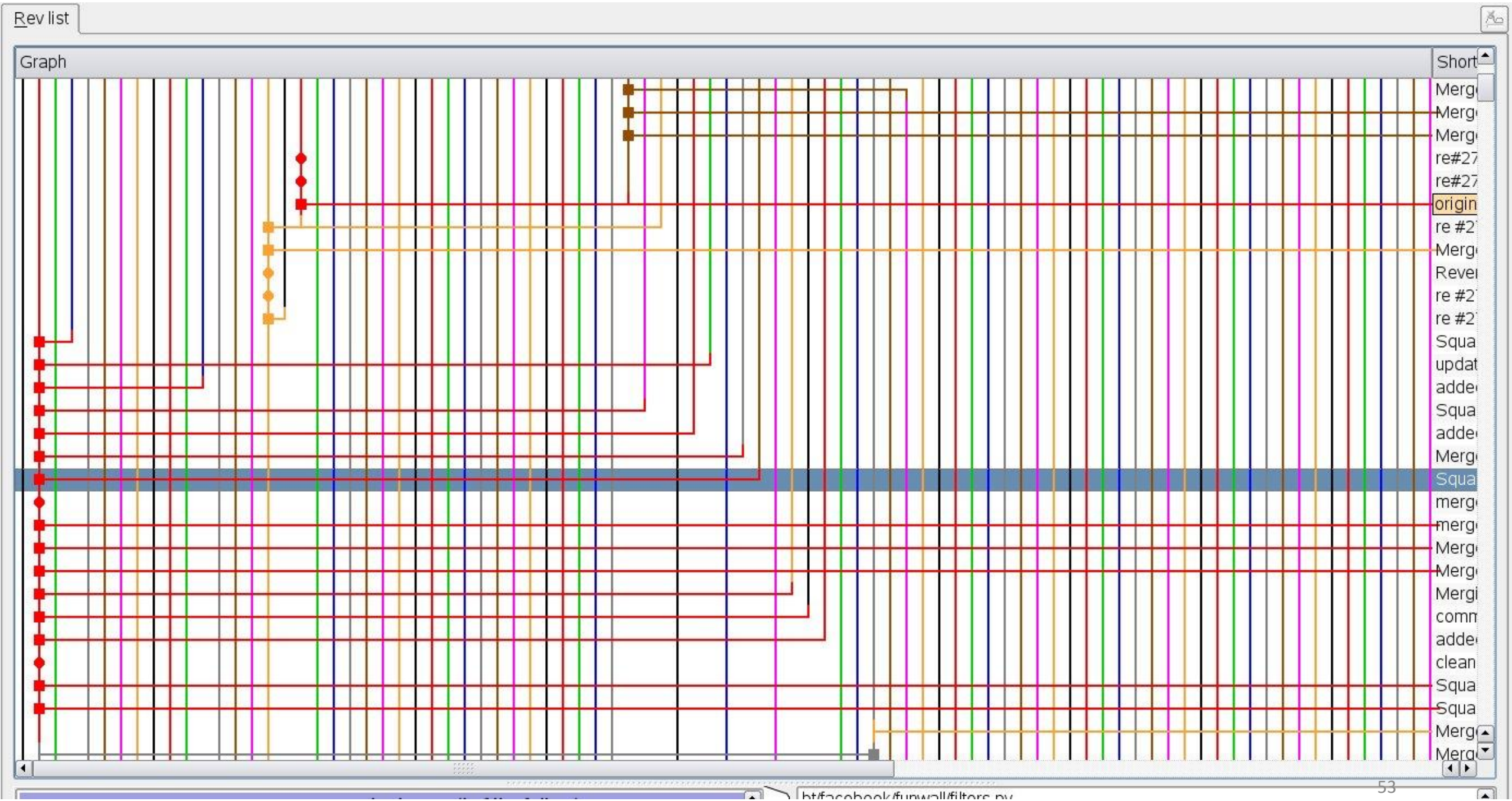


git merge awesome-feature --no-ff



git merge master

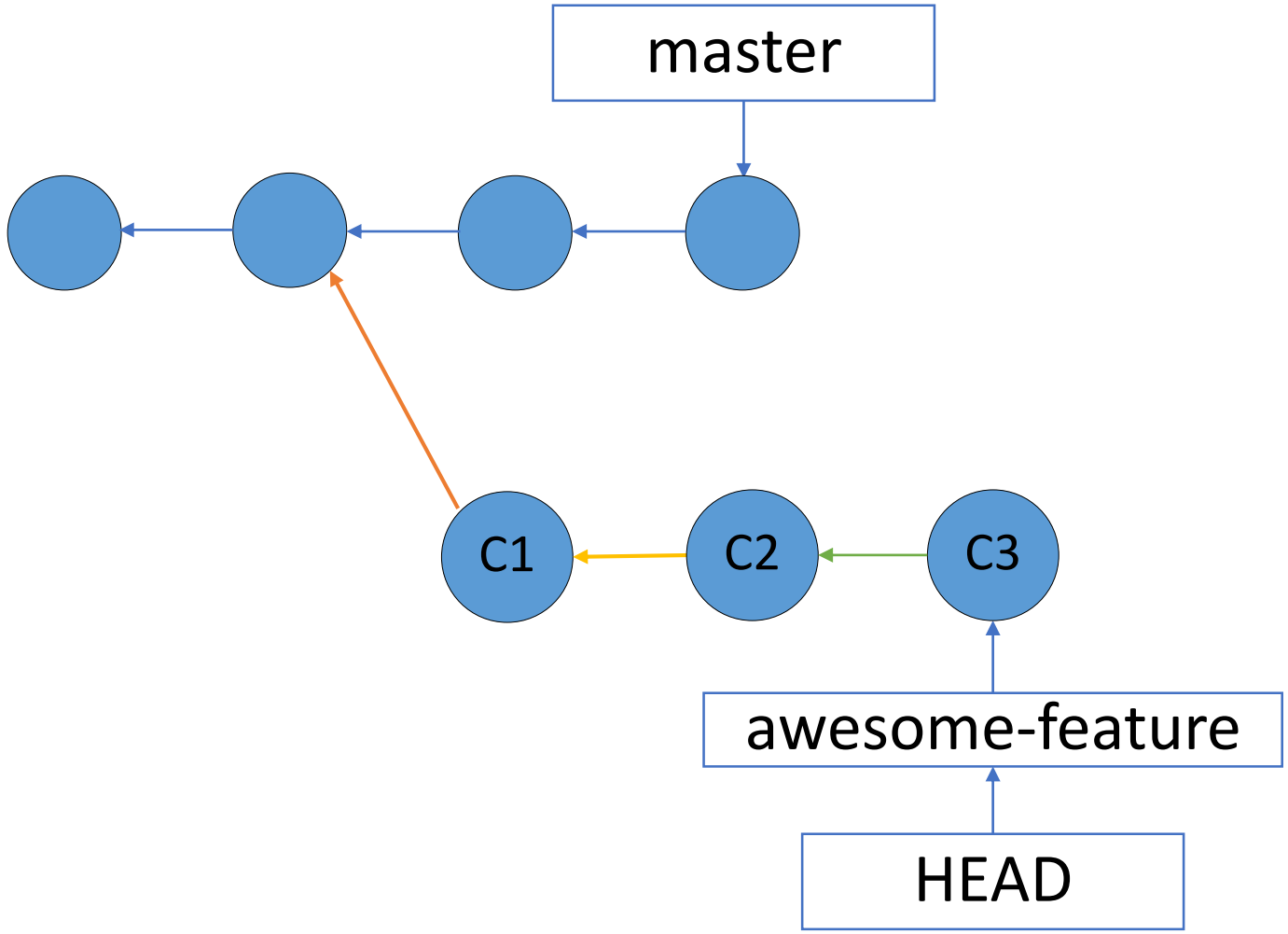



Rev list
Graph
Short
Merg
Merg
Merg
re#27
re#27
origin
re #2
Merg
Revei
re #2
re #2
Squa
updat
adde
Squa
adde
Merg
Squa
merg
merg
Merg
Merg
Mergi
comm
adde
clean
Squa
Squa
Merg
Merg

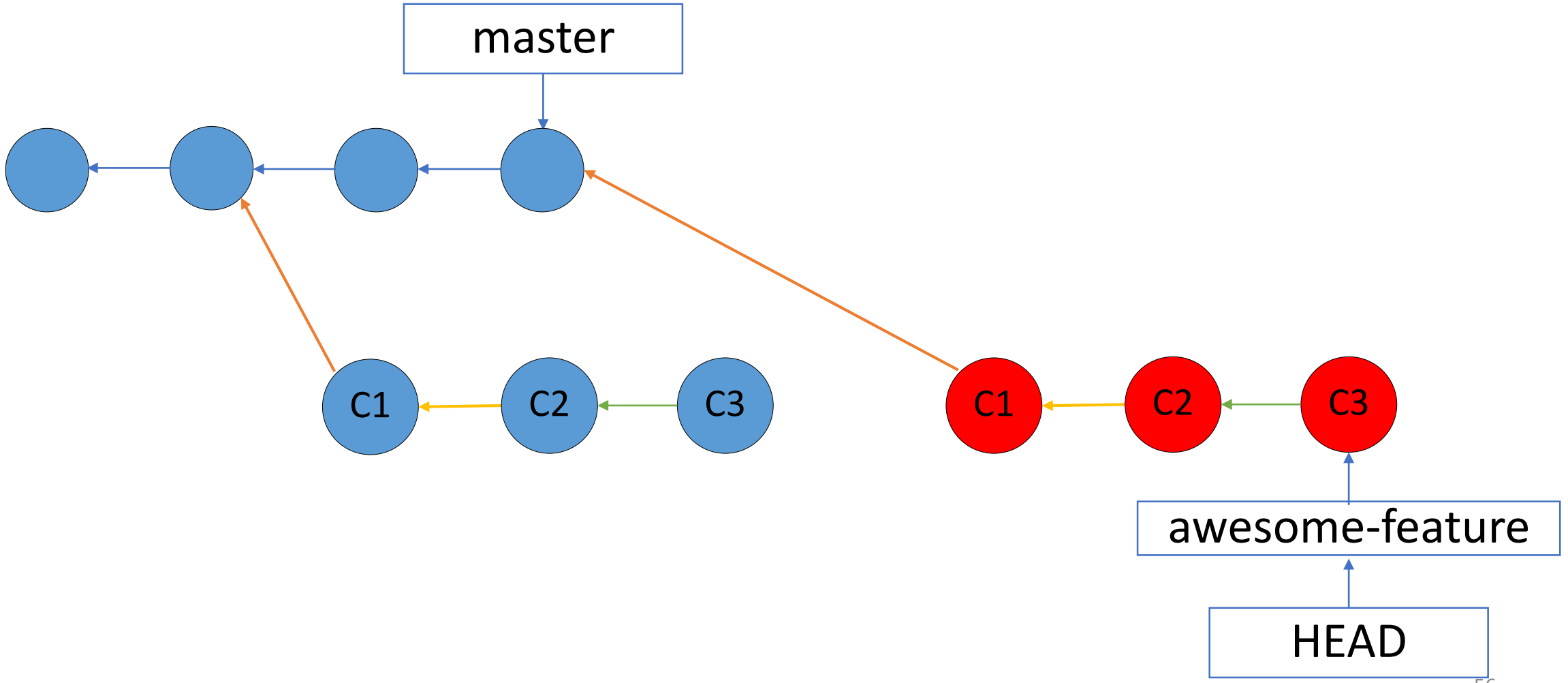
Минусы merge **из** master

- История получается нелинейная, в ней сложно ориентироваться
- Merge коммит содержит много изменений
  - Что если бага?
  - merge конфликты затрудняют понимание истории
- Что делать?
  - Git rebase

git rebase master



git rebase master



Плюсы rebase

- Легче понять историю, т.к. все линейно
- Легче найти багу, если после rebase оказалось, что что-то не работает
- Merge конфликты решаются по частям в каждом коммите

never use rebase on *public* branches

golden rule of rebase

Подводные камни rebase

- Не так просто откатить rebase (**решение:** *git reflog*)
- Приходится делать git push --force с риском перезаписать чью-то историю (**решение:** *git push --force-with-lease*)
- Необходимо понимать, как взять актуальную версию после force push от коллеги (**решение:** *git pull --rebase*)

- Даты коммитов меняются
- Ваши коллеги фанаты merge ☹

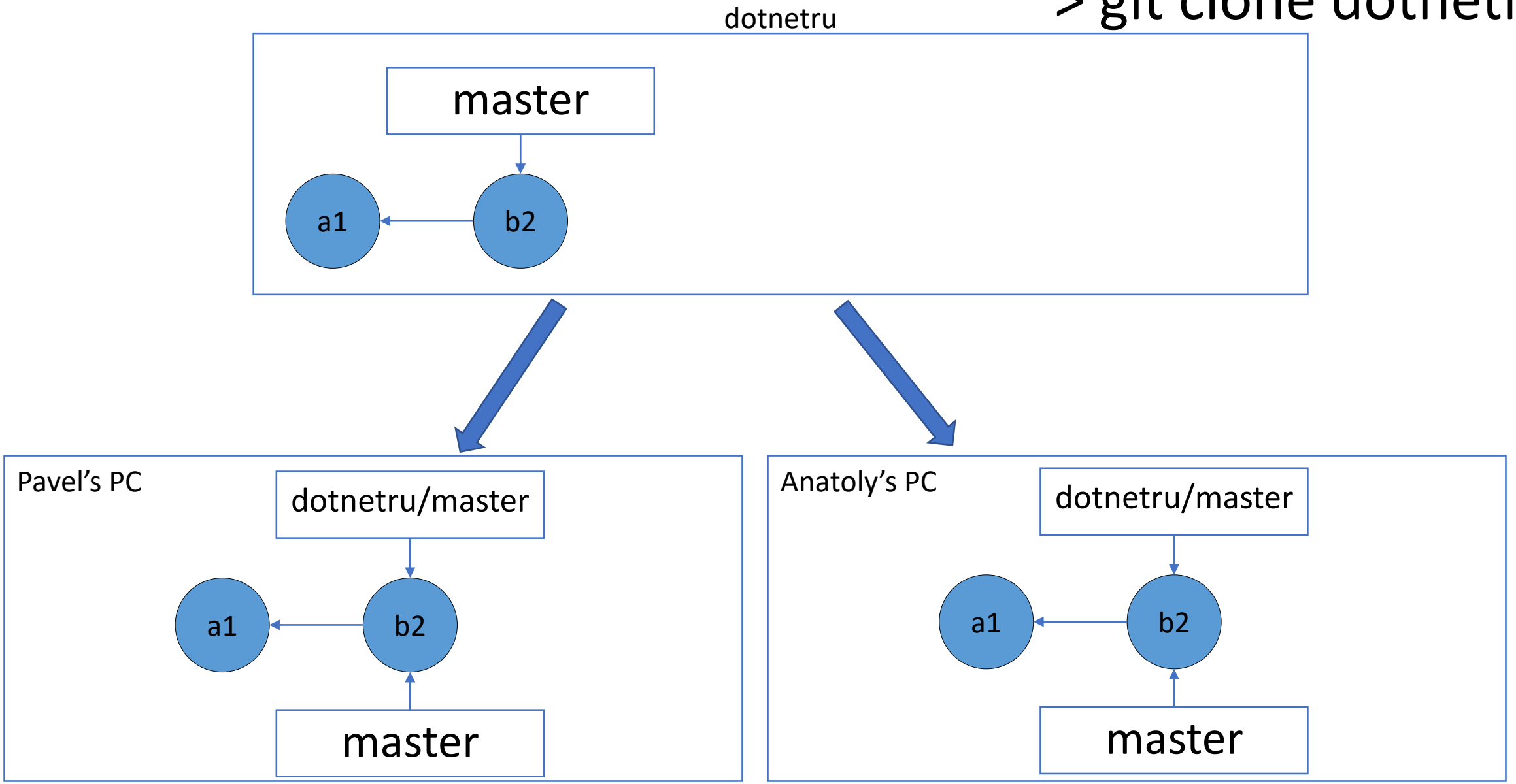
- **IMHO:** always rebase, если понимаете все подводные камни выше

Remotes

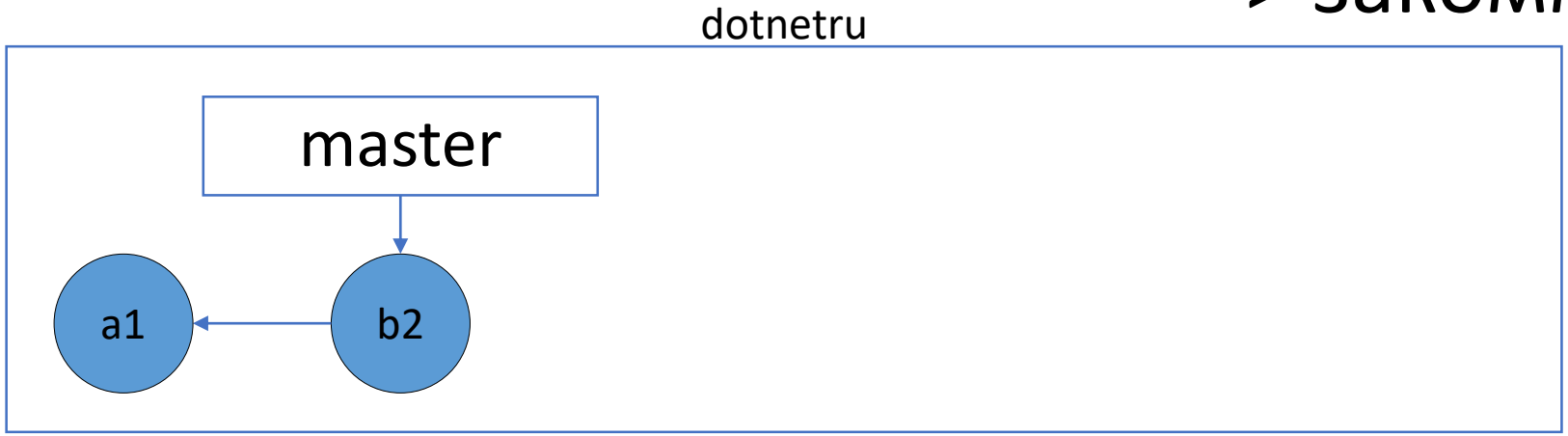
- До этого все было локально ☺
- Хотим поделиться нашим кодом с коллегами

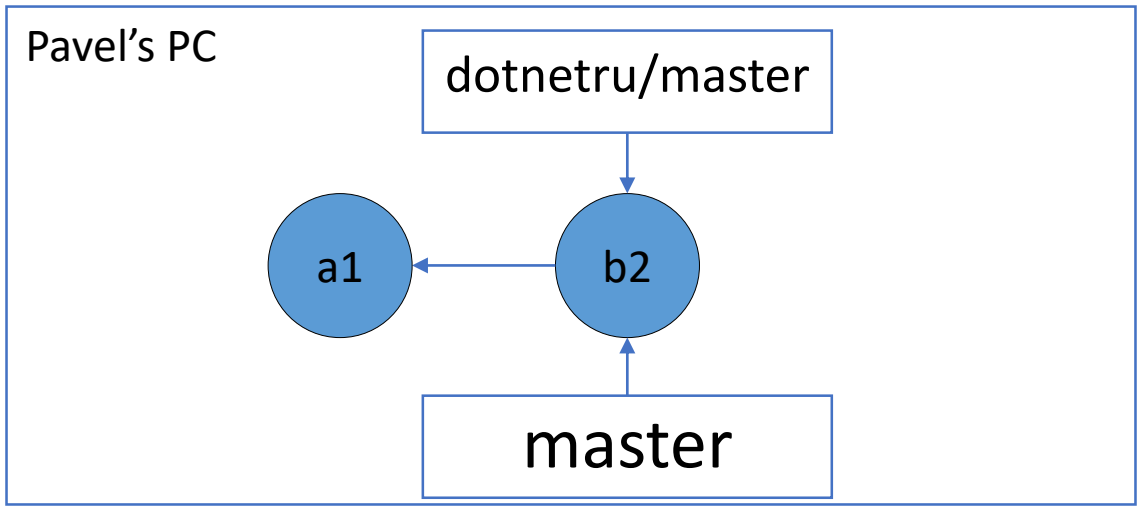
- Что делать?
  - git remotes
  - clone/fetch/pull/push для взаимодействия с сервером

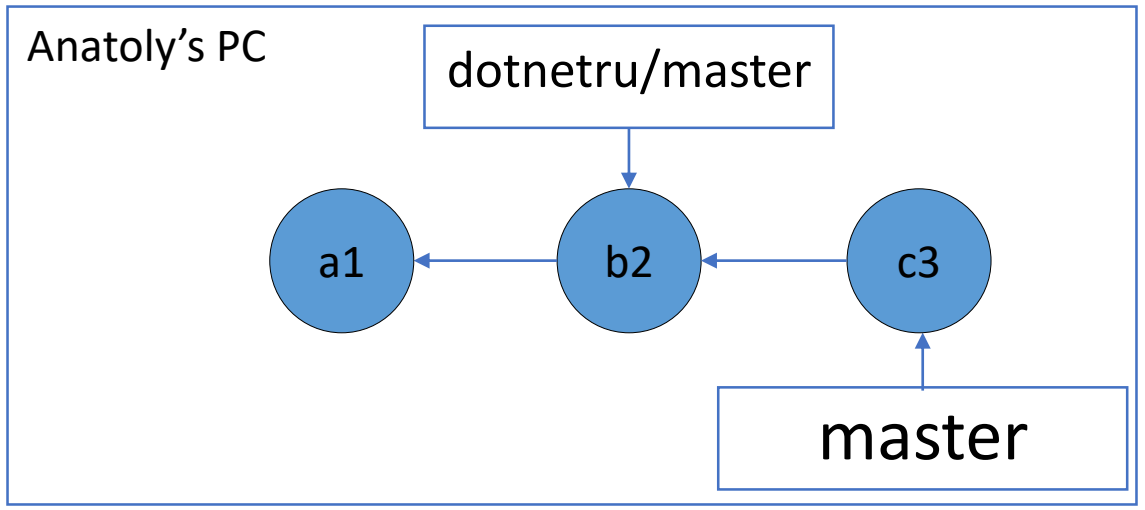
> git clone dotnetru



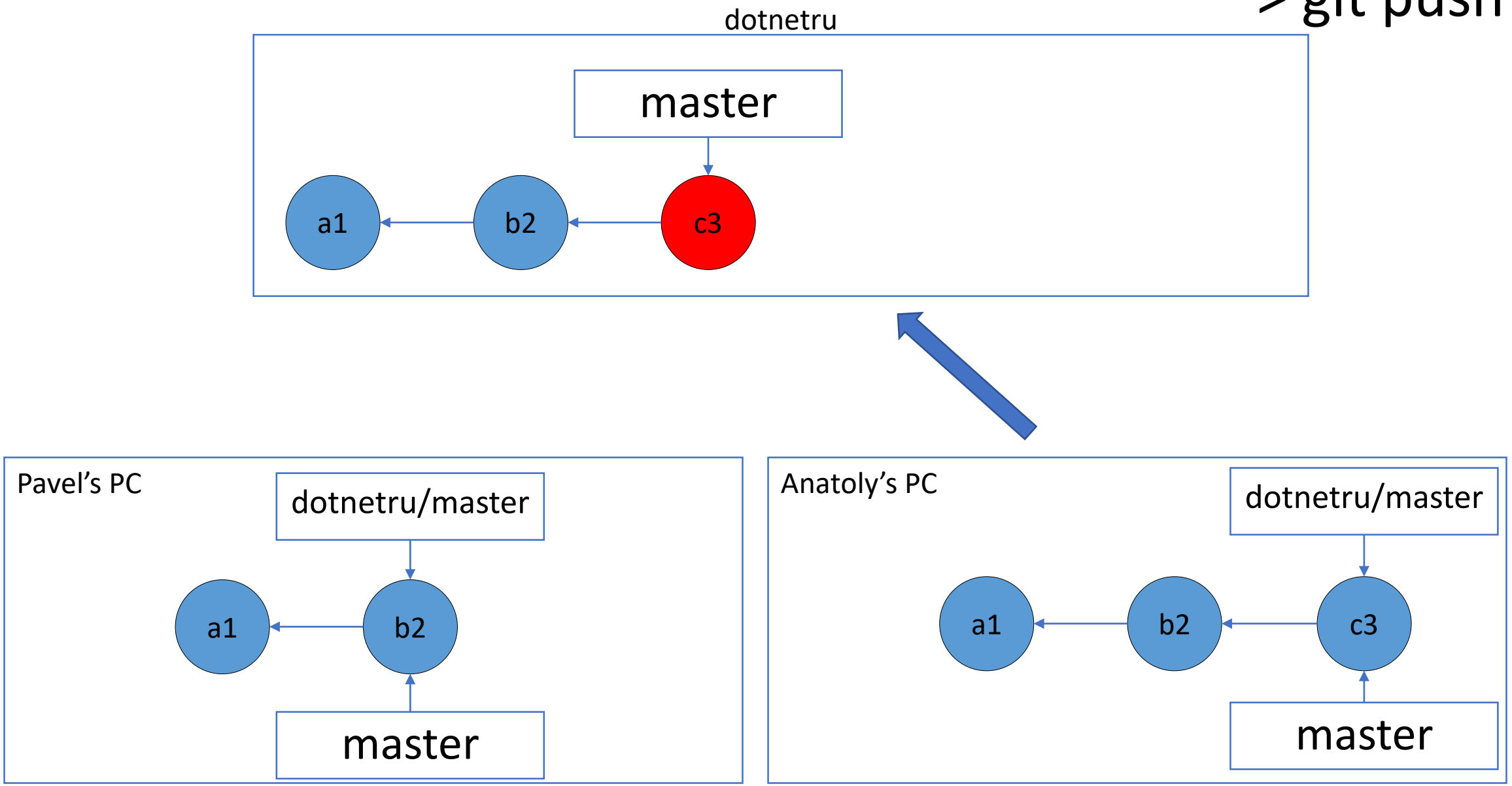
> закоммитили c3



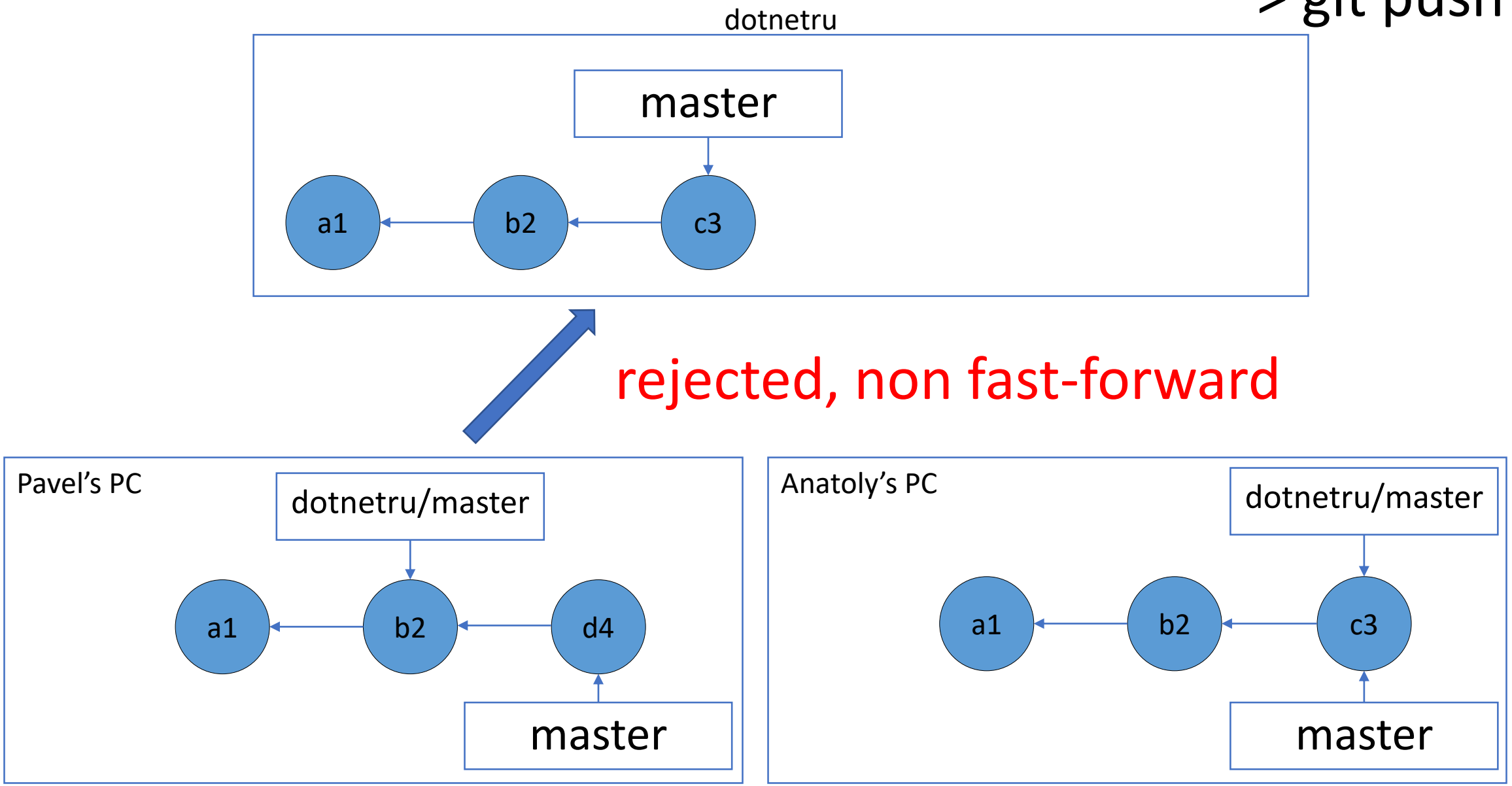




> git push


dotnetru
master
a1
b2
c3
Pavel's PC
dotnetru/master
a1
b2
master
Anatoly's PC
dotnetru/master
a1
b2
c3
master

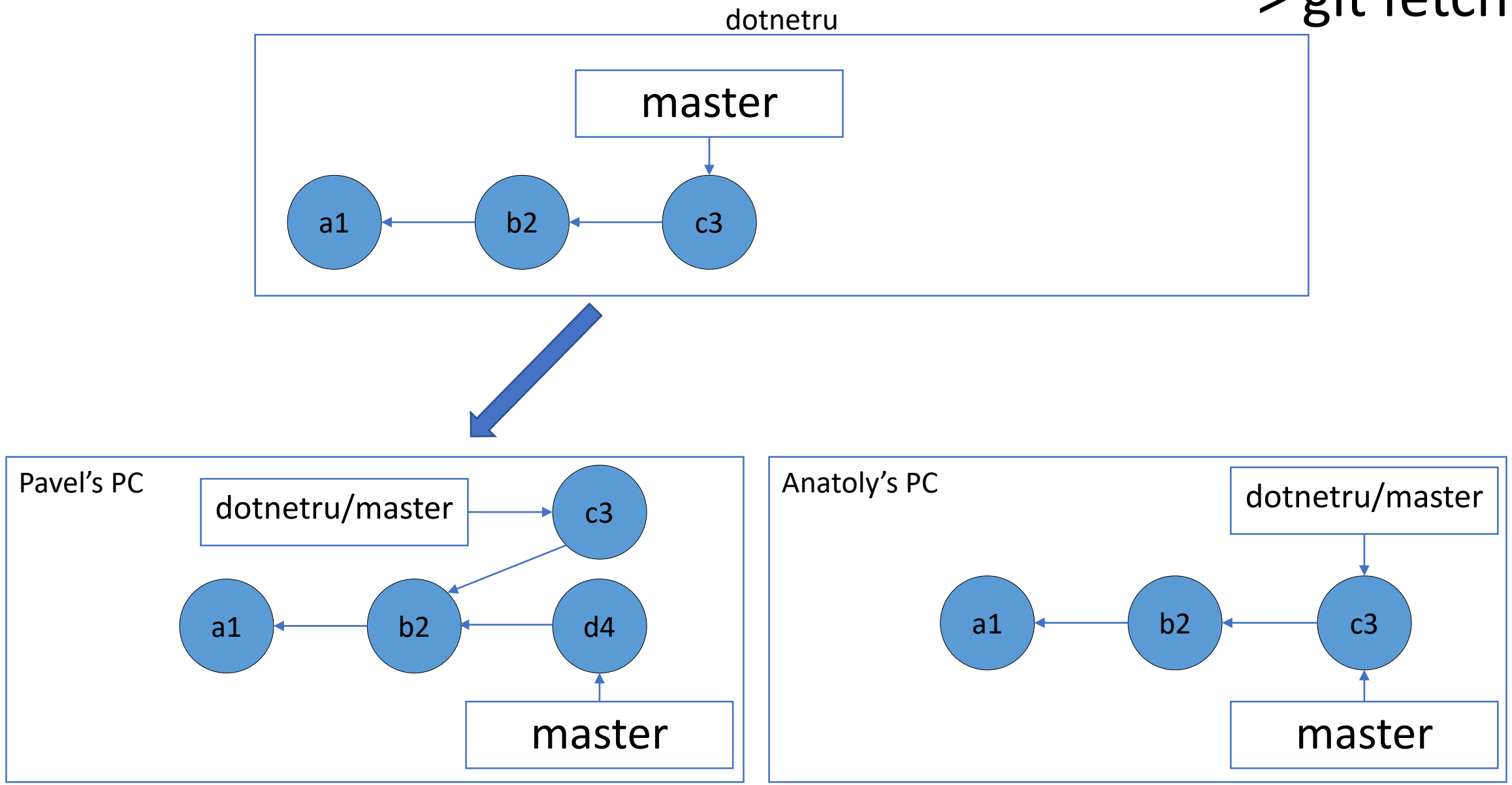
> git push


dotnetru
master
a1
b2
c3
rejected, non fast-forward
Pavel's PC
dotnetru/master
a1
b2
d4
master
Anatoly's PC
dotnetru/master
a1
b2
c3
master

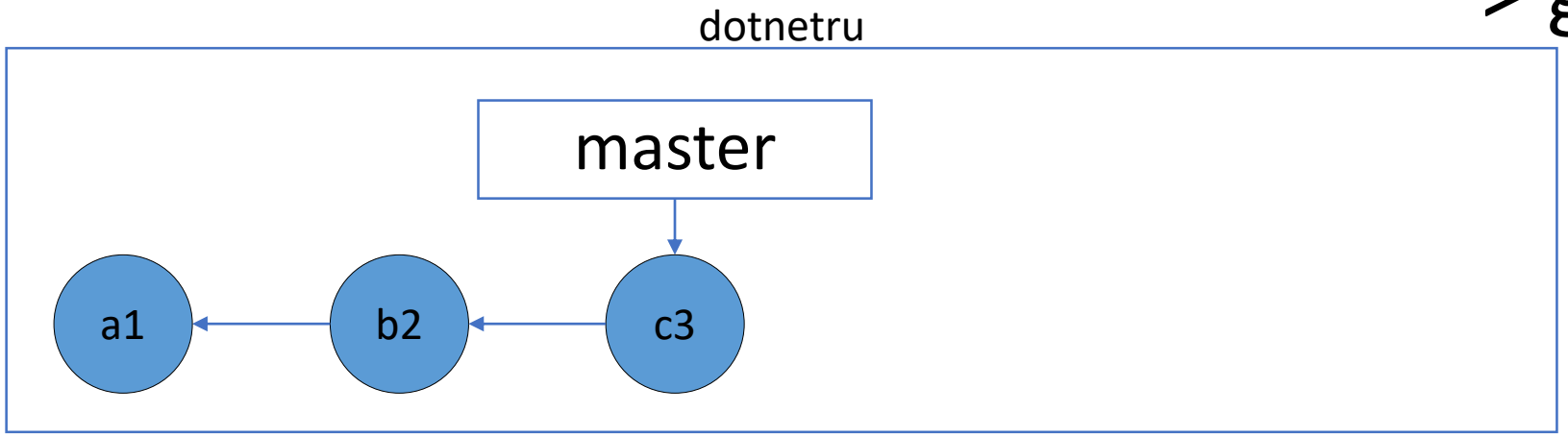
git fetch

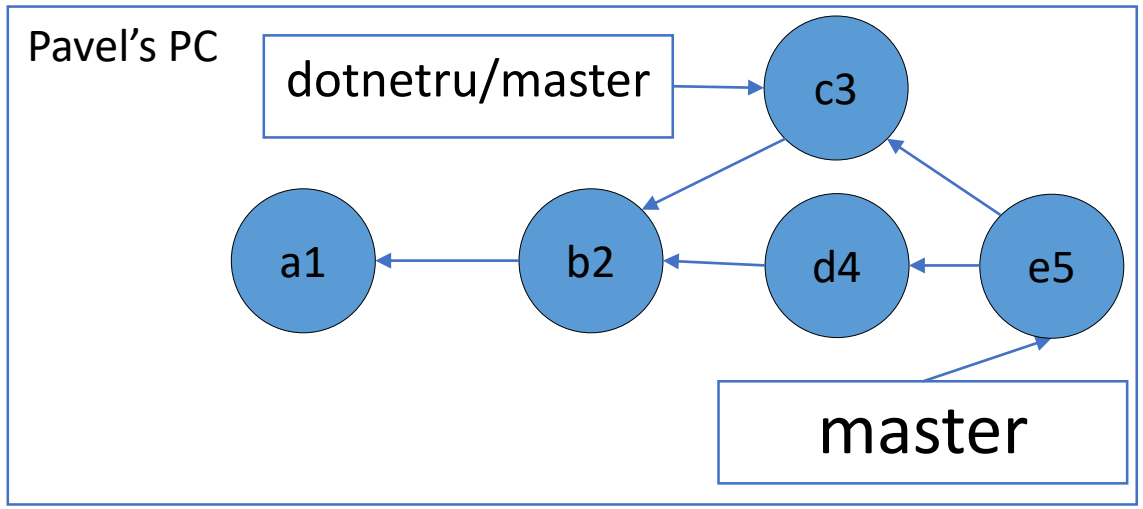
- Скачивает новые коммиты (!) с сервера
  - Не изменения, а именно коммиты
- Обновляет ветки в соответствие с сервером
  - Технически, проставляет указатели <remote>/<branchname>

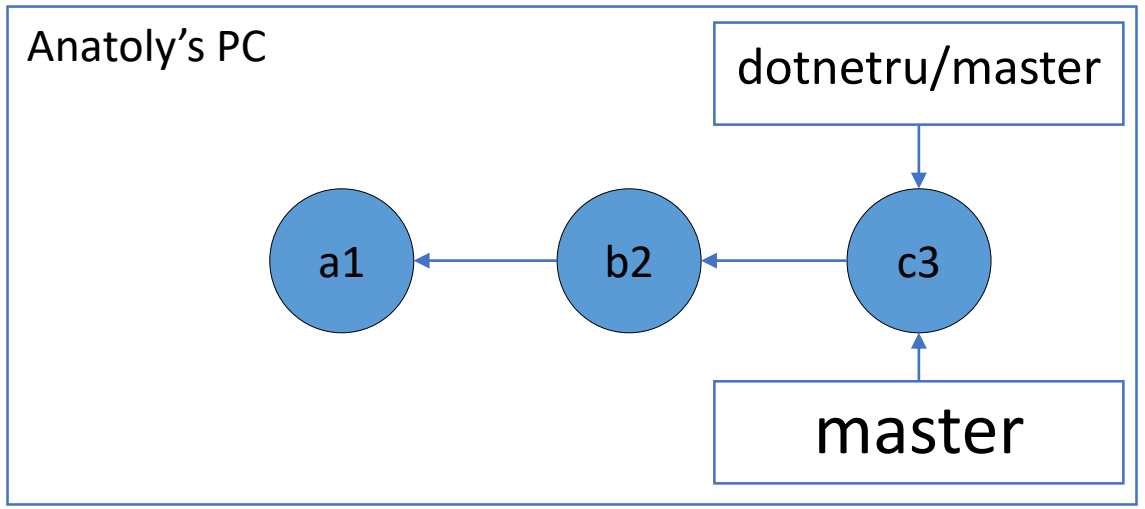
> git fetch


dotnetru
master
a1
b2
c3
Pavel's PC
dotnetru/master
c3
a1
b2
d4
master
Anatoly's PC
dotnetru/master
a1
b2
c3
master

> git merge


dotnetru
master
a1
b2
c3


Pavel's PC
dotnetru/master
c3
a1
b2
d4
e5
master

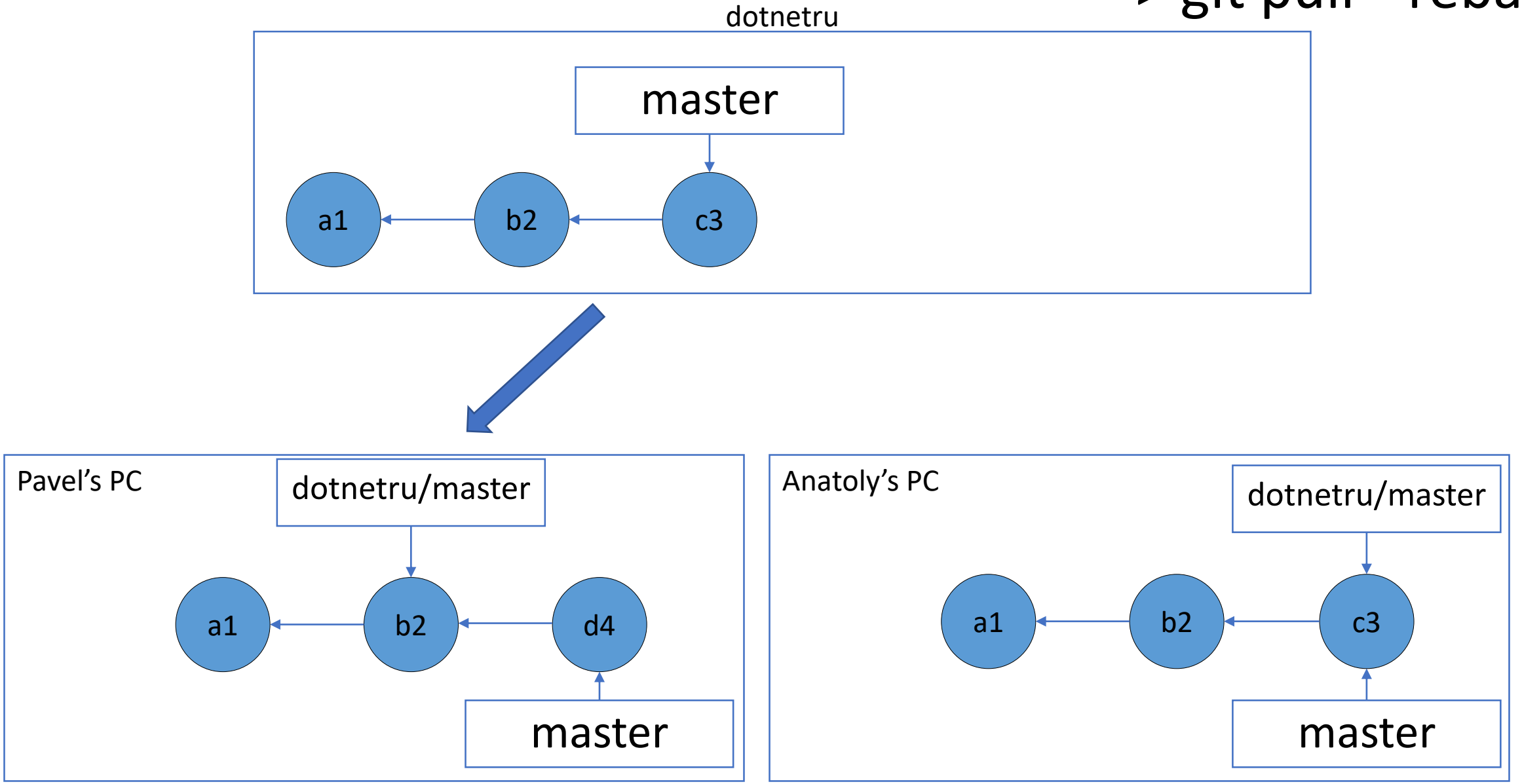

Anatoly's PC
dotnetru/master
a1
b2
c3
master

git pull == fetch + merge

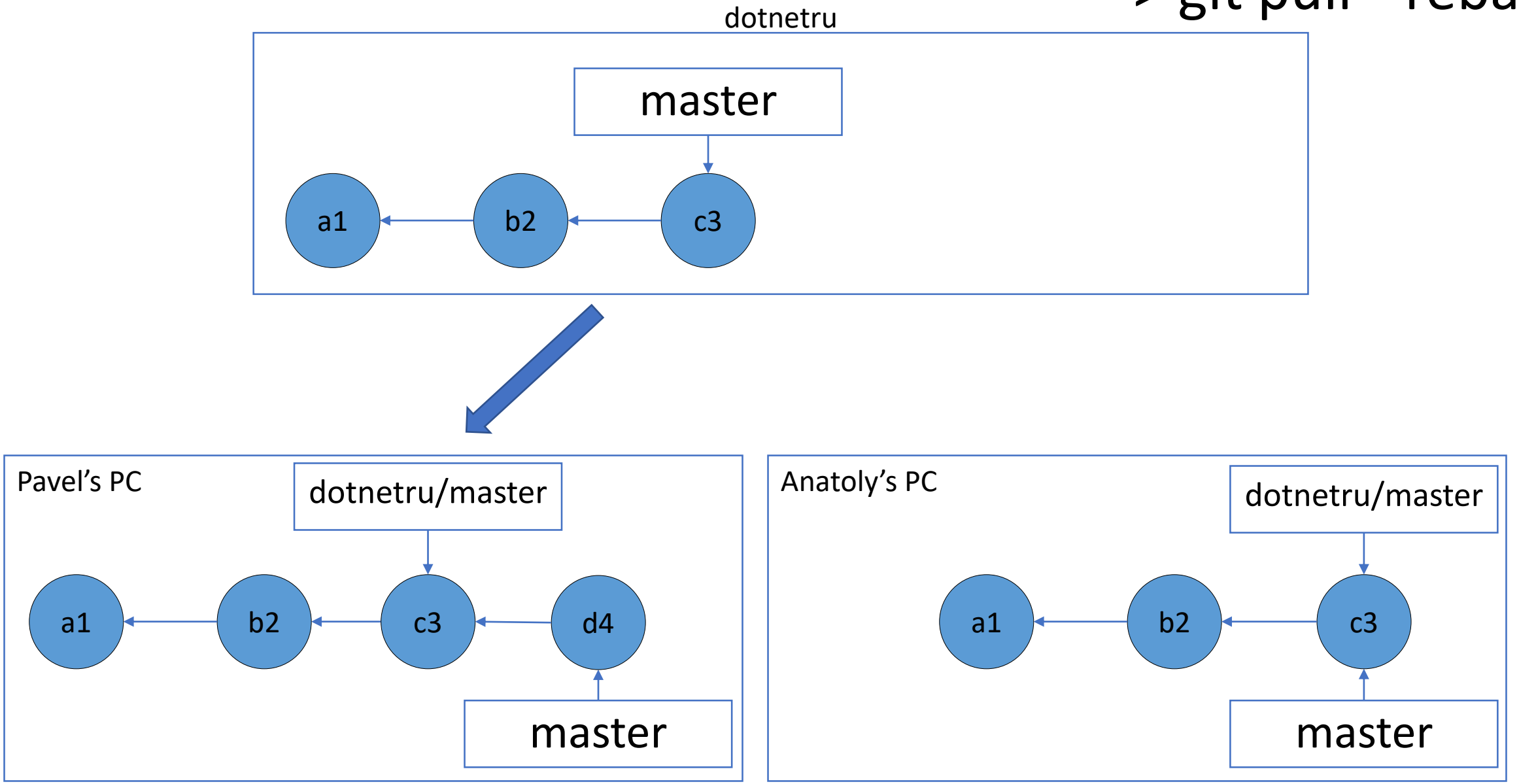
git config pull.rebase true

git pull == fetch + rebase

> git pull --rebase



> git pull --rebase



Tracking branches

- Git хранит соответствие между локальной веткой и серверной
  - Они могут называться совершенно по-разному
- Зачем?
  - Git push понимает, какую ветку на сервере вы хотите обновить
  - Git pull понимает, какую локальную ветку вы хотите обновить
- Новый бранч?
  - git push --set-upstream origin <featurename>
  - git config push.default current -> **git push -u**

Основной сценарий

- Создали новую ветку (git checkout –b <featurename>)
- Изменили файлы в рабочей директории
- Добавили их в индекс (git add)
- Коммит (git commit)
- Выложили (git push)

Shallow fetch

- git clone скачивает ВСЁ – всю историю, медленно ☹
  - страдают билды

Что делать?

- git clone|fetch --depth N

Case sensitivity

- Git учитывает регистр => можно иметь файлы *Hello.cs & hello.cs*
- В Windows так нельзя ☺

- Случается так, что в репозитории файлы с одинаковым именем

- Что делать?
  - Не делать так
  - TFS: Case Enforcement https://docs.microsoft.com/en-us/azure/devops/repos/git/repository-settings?view=vsts#case-enforcement

git config

- --local (по умолчанию, текущий репозиторий)
- --global (все репозиторий, текущий пользователь)
- --system (все репозитории, все пользователи)

git config –{local|system|global} –edit

https://gist.github.com/pfedotovsky/f77698a9be2ad4ae65fa774d89d78f1c

Как перейти на Git?

- Все используют Git: Windows, .NET, Linux, etc.
- http://z.github.io/whygitisbetter/
- Готовые утилиты для миграции
  - Microsoft рекомендует не переносить историю ☺
  - https://docs.microsoft.com/en-us/azure/devops/repos/git/import-from-tfvc?view=vsts

Git at Enterprise scale

- Git LFS: https://git-lfs.github.com
  - Для работы с бинарными файлами
- GIT GVFS: https://gvfs.io
  - Для частичного clone репозитория
  - Скачивает файлы по мере необходимости
  - Работает на уровне драйвера файловой системы

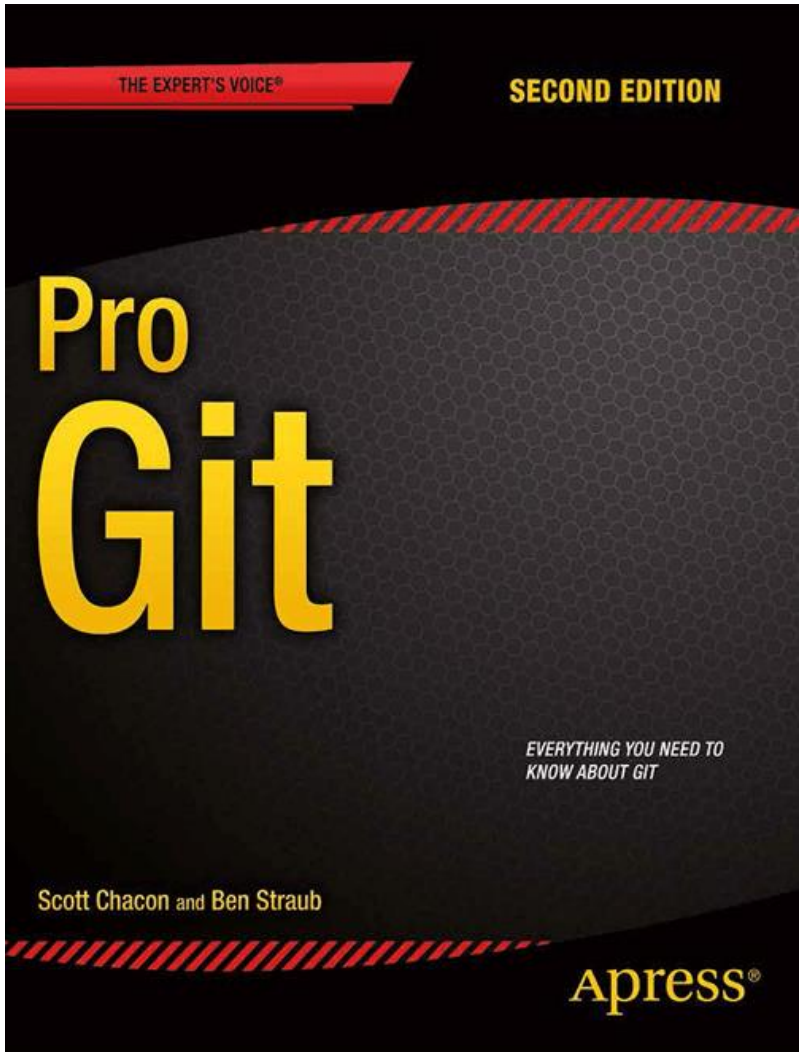
Misc

- Git Flow
- Дифф для docx/pptx и др. бинарных файлов
- Git shortlog – группировка по авторам, полезно для релизов
- Git grep
- Git replace
- Git revert
- Git bisect
- Git rebase –i
- Git log – различные опции
- Git rerere
- Git commit –amend
- Git hooks
- Git attributes

Выводы

- Git – де-факто стандарт для контроля версий
- Git содержит огромное количество утилит на все случаи жизни
- В Git вы можете настроить свой процесс разработки, как хотите (e.g. Git Flow)
- Перейти на Git просто

Домашнее чтение



Полезные ссылки

- Документация
  - http://git.github.io/git-reference/
  - https://git-scm.com
- https://sethrobertson.github.io/GitFixUm/fixup.html
- Enterprise Git
  - Git LFS: https://git-lfs.github.com
  - Git GVFS: https://gvfs.io
- Rebase
  - https://www.atlassian.com/git/articles/git-team-workflows-merge-or-rebase
  - https://developer.atlassian.com/blog/2015/04/force-with-lease/
  - Письмо Линуса: https://www.mail-archive.com/dri-devel@lists.sourceforge.net/msg39091.html

Задачка

Как смержить два бранча не используя git merge?