A TALE OF THREE TREES

a magical afternoon with Scott Chacon

About Me


github
SOCIAL CODING

Git Resources

git-scm.com

gitref.org

progit.org

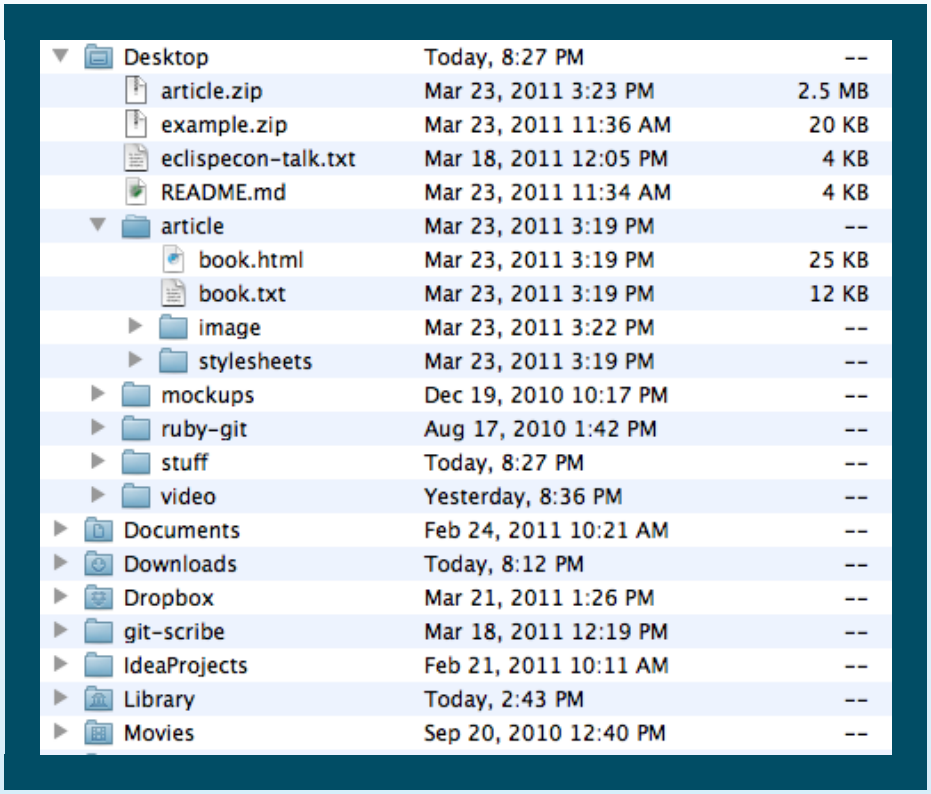
@chacon

</me>

PREFACE

How Git Works

It's all about the trees, baby


Desktop Today, 8:27 PM --
article.zip Mar 23, 2011 3:23 PM 2.5 MB
example.zip Mar 23, 2011 11:36 AM 20 KB
eclispecon-talk.txt Mar 18, 2011 12:05 PM 4 KB
README.md Mar 23, 2011 11:34 AM 4 KB
article Mar 23, 2011 3:19 PM --
book.html Mar 23, 2011 3:19 PM 25 KB
book.txt Mar 23, 2011 3:19 PM 12 KB
image Mar 23, 2011 3:22 PM --
stylesheets Mar 23, 2011 3:19 PM --
mockups Dec 19, 2010 10:17 PM --
ruby-git Aug 17, 2010 1:42 PM --
stuff Today, 8:27 PM --
video Yesterday, 8:36 PM --
Documents Feb 24, 2011 10:21 AM --
Downloads Today, 8:12 PM --
Dropbox Mar 21, 2011 1:26 PM --
git-scribe Mar 18, 2011 12:19 PM --
IdeaProjects Feb 21, 2011 10:11 AM --
Library Today, 2:43 PM --
Movies Sep 20, 2010 12:40 PM --

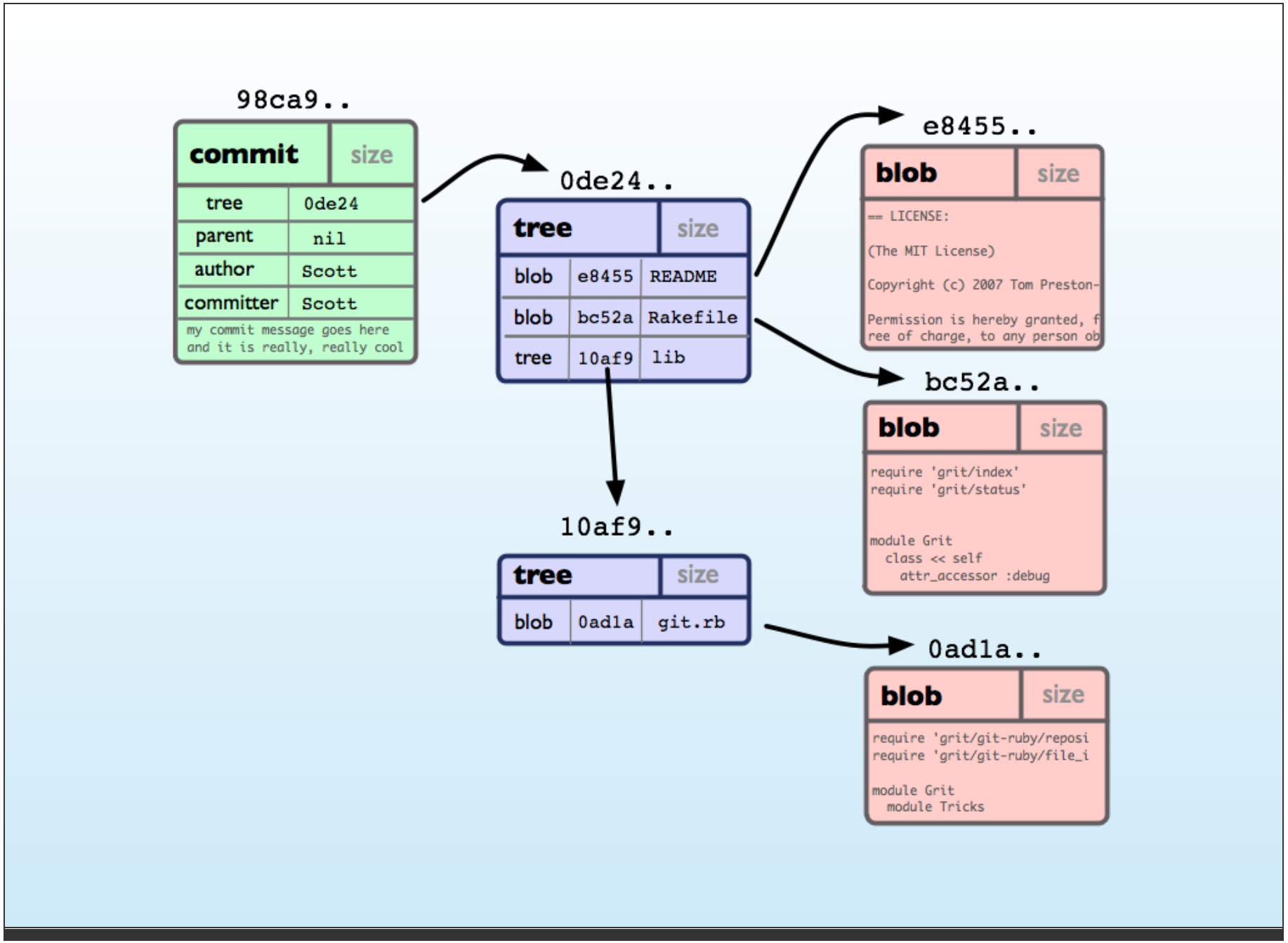
TREE IS

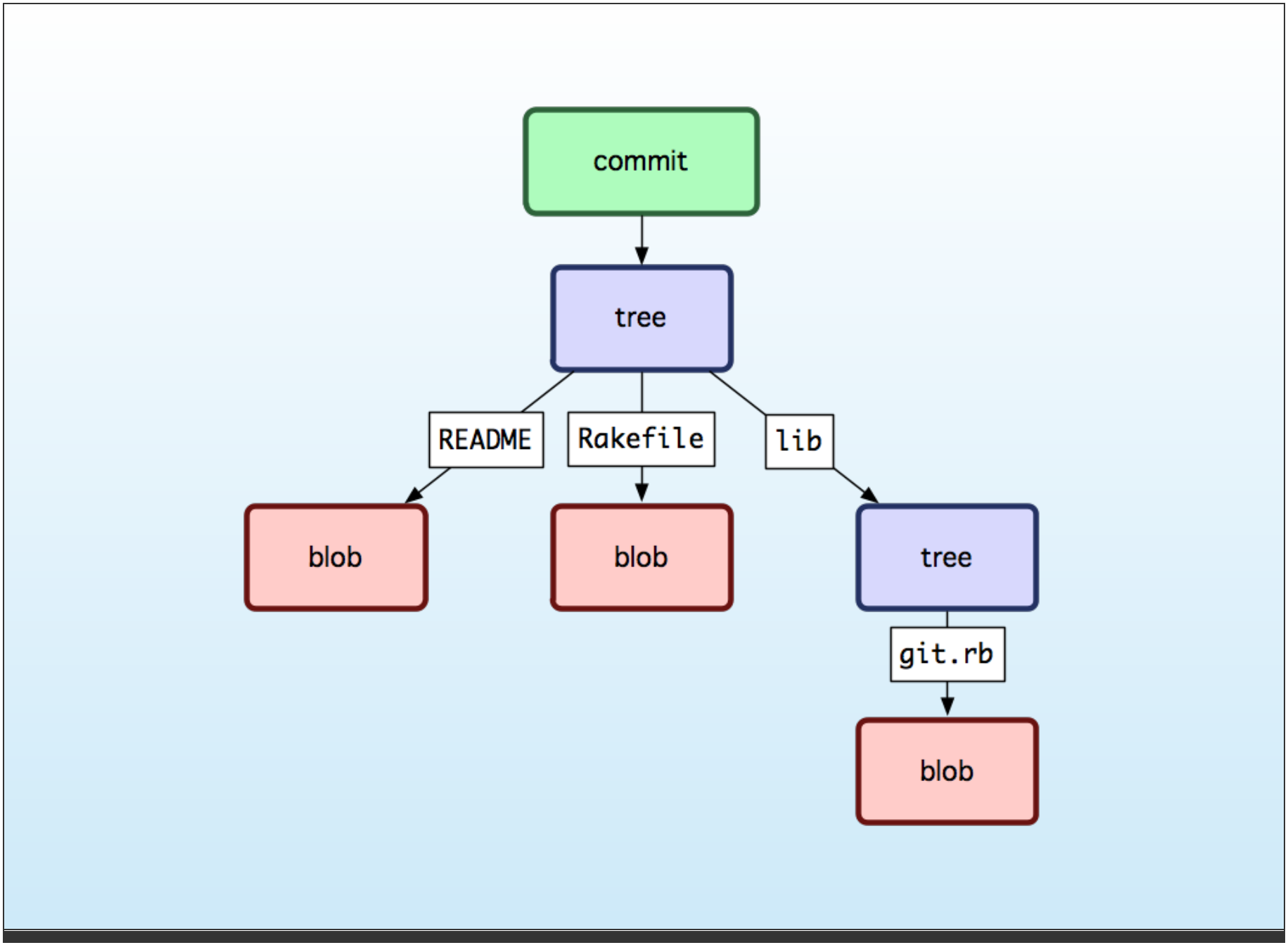
files and subtrees

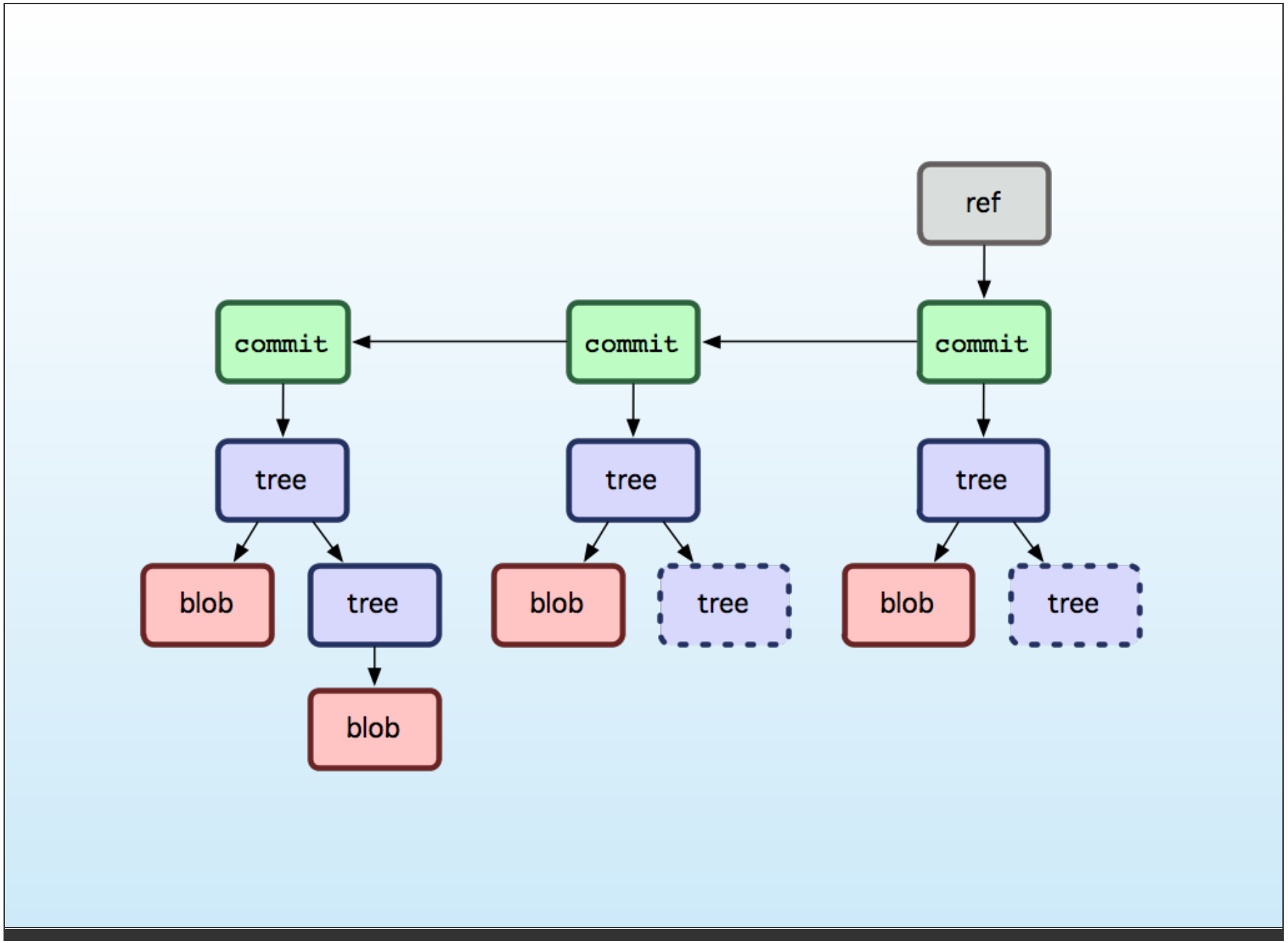
EXAMPLE

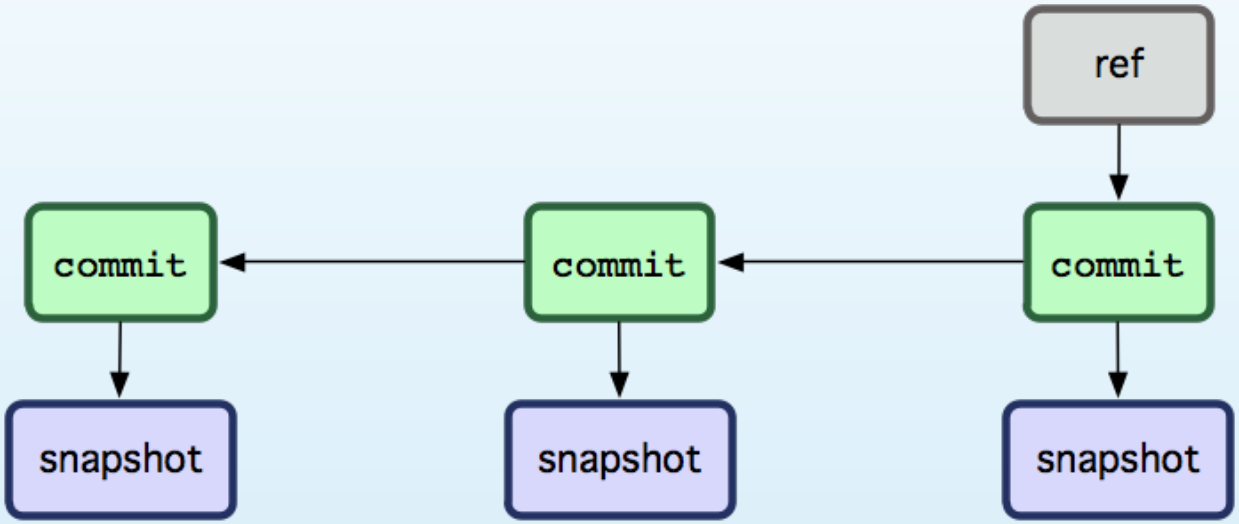
```
$ tree
.
├── README
├── Rakefile
└── lib
    └── git.rb

1 directory, 3 files
```


98ca9..
commit
size
tree
0de24
parent
nil
author
Scott
committer
Scott
my commit message goes here
and it is really, really cool
0de24..
tree
size
blob
e8455
README
blob
bc52a
Rakefile
tree
10af9
lib
e8455..
blob
size
== LICENSE:
(The MIT License)
Copyright (c) 2007 Tom Preston-
Permission is hereby granted, f
ree of charge, to any person ob
bc52a..
blob
size
require 'grit/index'
require 'grit/status'
module Grit
class << self
attr_accessor :debug
10af9..
tree
size
blob
0ad1a
git.rb
0ad1a..
blob
size
require 'grit/git-ruby/reposi
require 'grit/git-ruby/file_i
module Grit
module Tricks


commit
tree
README
Rakefile
lib
blob
blob
tree
git.rb
blob


ref
commit
commit
commit
tree
tree
tree
blob
tree
blob
tree
blob
tree
blob


ref
commit
commit
commit
snapshot
snapshot
snapshot

ACT ONE

The Three Trees

first tree

the HEAD

```
$ cat .git/HEAD
ref: refs/heads/master

$ cat .git/refs/heads/master
e9a570524b63d2a2b3a7c3325acf5b89bbeb131e

$ git cat-file -p e9a570524b63d2a2b3a7c3325acf5b8
tree cfda3bf379e4f8dba8717dee55aab78aef7f4daf
author Scott Chacon <schacon@gmail.com> 130151183
committer Scott Chacon <schacon@gmail.com> 130151

initial commit

$ git ls-tree -r cfda3bf379e4f8dba8717dee55aab78a
100644 blob a906cb2a4a904a152...   README
100644 blob 8f94139338f9404f2...   Rakefile
040000 tree 99f1a6d12cb4b6f19...   lib
```

second tree

the index

The Staging Area

```ruby
require 'rugged'

index = Rugged::Index.new("/opt/repo/.git/index")
index.refresh

index.each do |entry|
  puts "File Name: " + entry.path
  puts " Blob SHA: " + entry.sha
  puts "File Size: " + entry.file_size.to_s
  puts "File Mode: " + entry.mode.to_s
  puts "     mtime: " + entry.mtime.to_i.to_s
  puts "     ctime: " + entry.ctime.to_i.to_s
  puts "     Inode: " + entry.ino.to_s
  puts "       UID: " + entry.uid.to_s
  puts "       GID: " + entry.gid.to_s
  puts
end
```

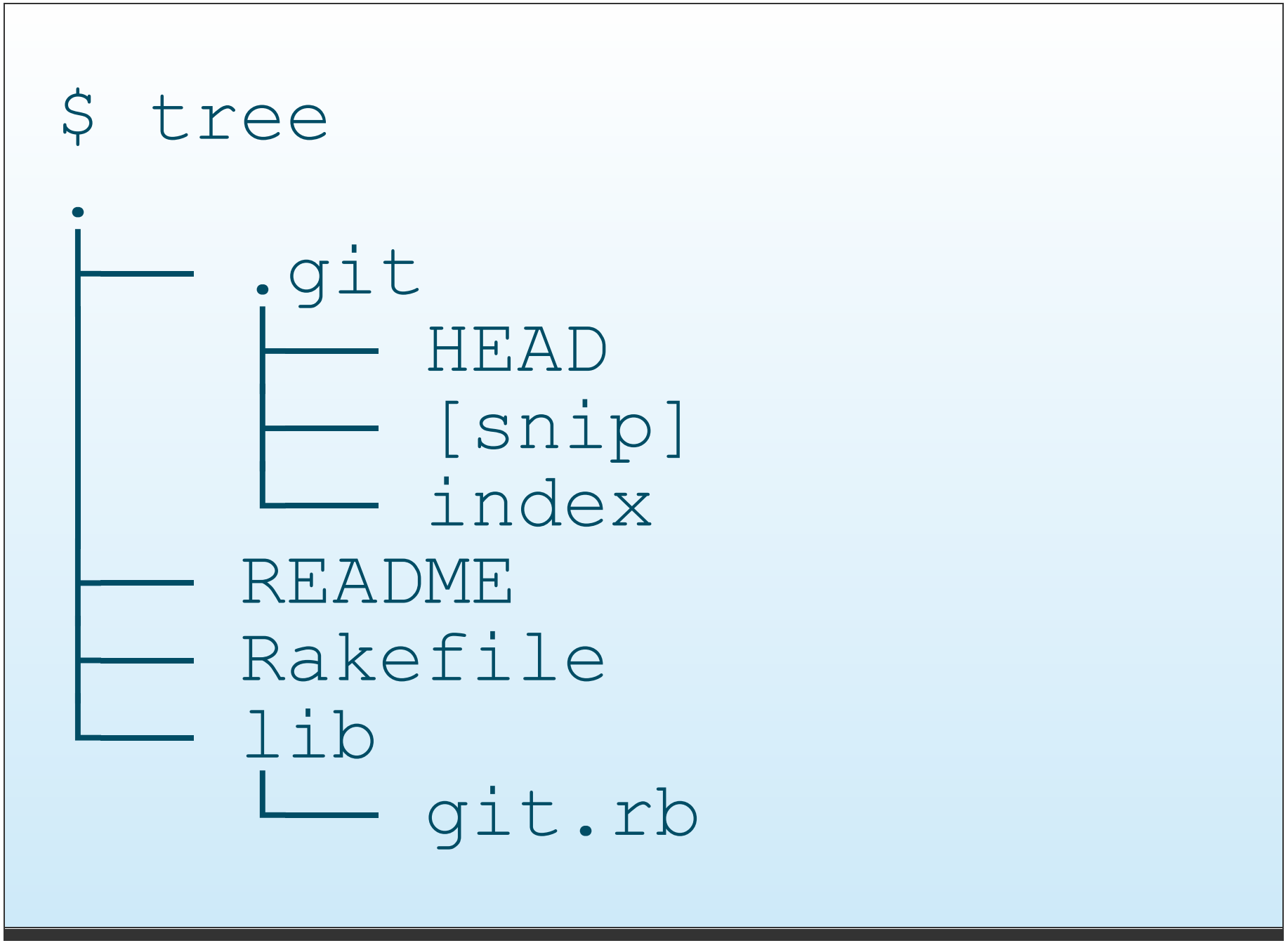
```
File Name: README
 Blob SHA: 45dc653de6860faeb30581cd7654f9a51fc2c443
File Size: 135
File Mode: 33188
    mtime: 1301512685
    ctime: 1301512685
    Inode: 15472643
      UID: 501
      GID: 0

File Name: Rakefile
 Blob SHA: ea3fe2ac46e92bf38dc824128e3eddd397f537e3
File Size: 604
File Mode: 33188
    mtime: 1301512703
    ctime: 1301512703
    Inode: 15472659
      UID: 501
      GID: 0

File Name: lib/simplegit.rb
 Blob SHA: 47c6340d6459e05787f644c2447d2595f5d3a54b
File Size: 355
File Mode: 33188
    mtime: 1301507599
    ctime: 1301507599
    Inode: 15445705
      UID: 501
      GID: 0
```

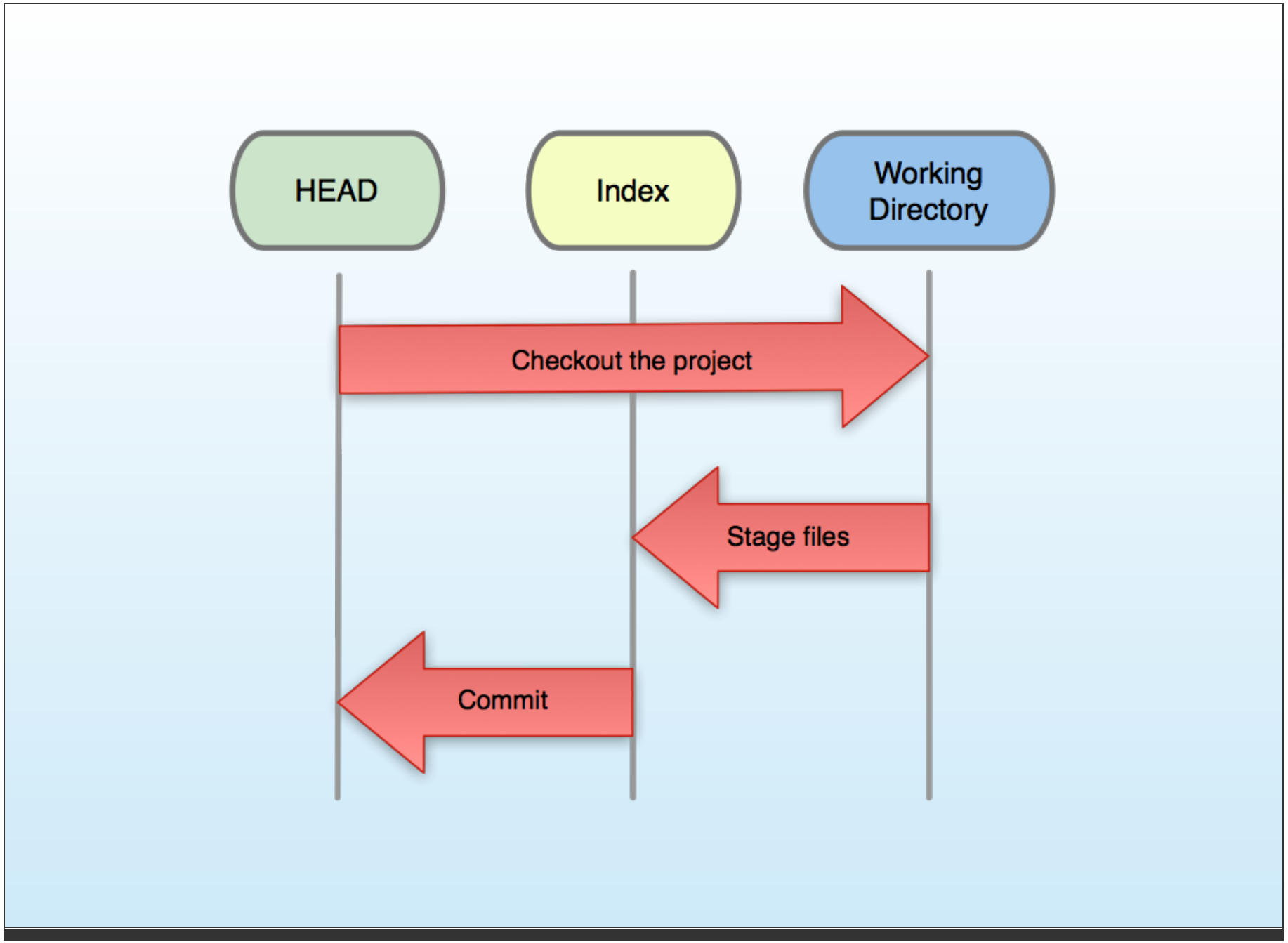
third tree

the working directory


$ tree
.
├── .git
│   ├── HEAD
│   ├── [snip]
│   └── index
├── README
├── Rakefile
└── lib
    └── git.rb

Three Trees

HEAD, Index and Working Directory


HEAD
Index
Working Directory
Checkout the project
Stage files
Commit

Tree Roles

HEAD last commit, next parent

Index proposed next commit

Work Dir sandbox

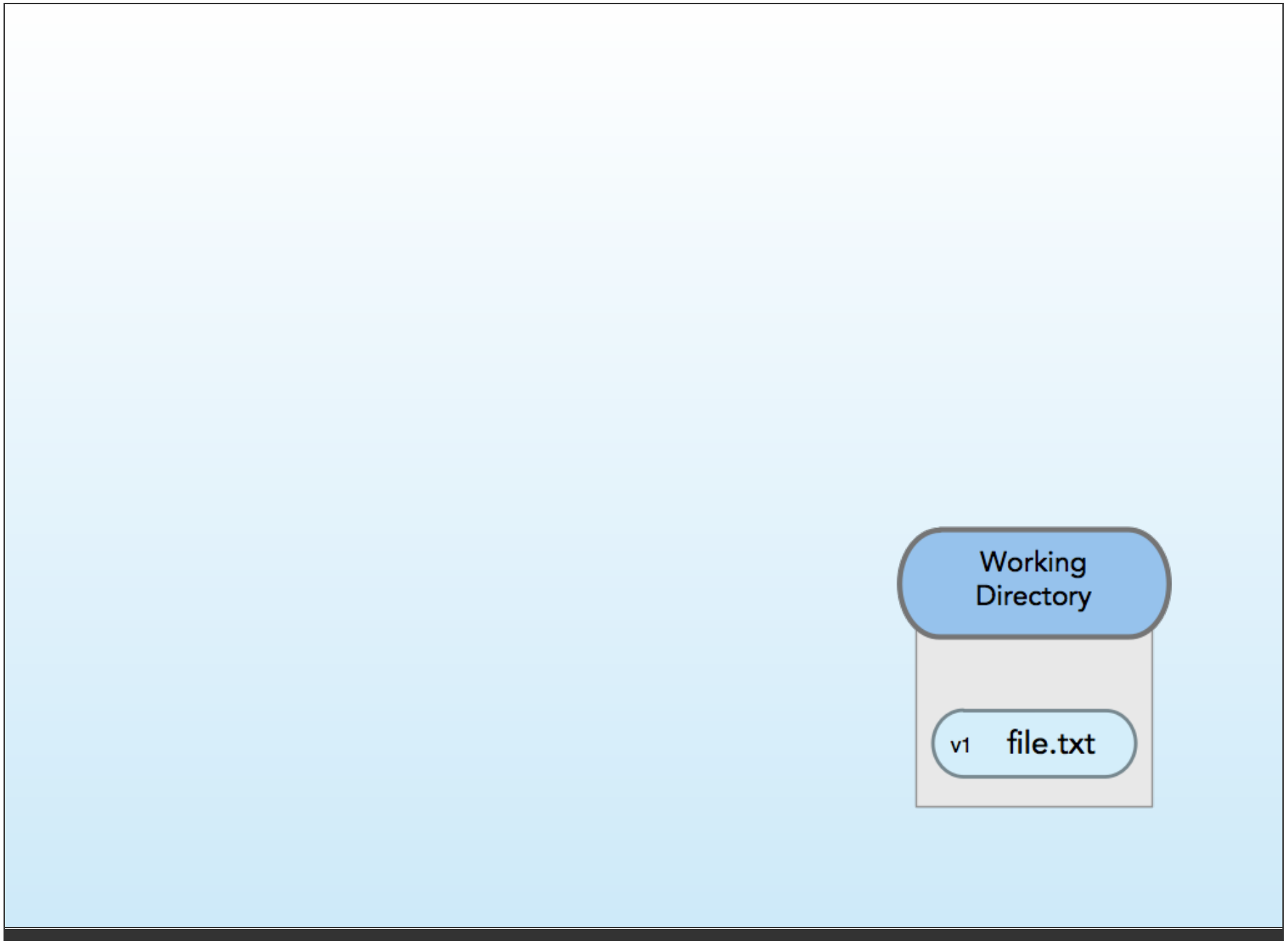
ACT TWO

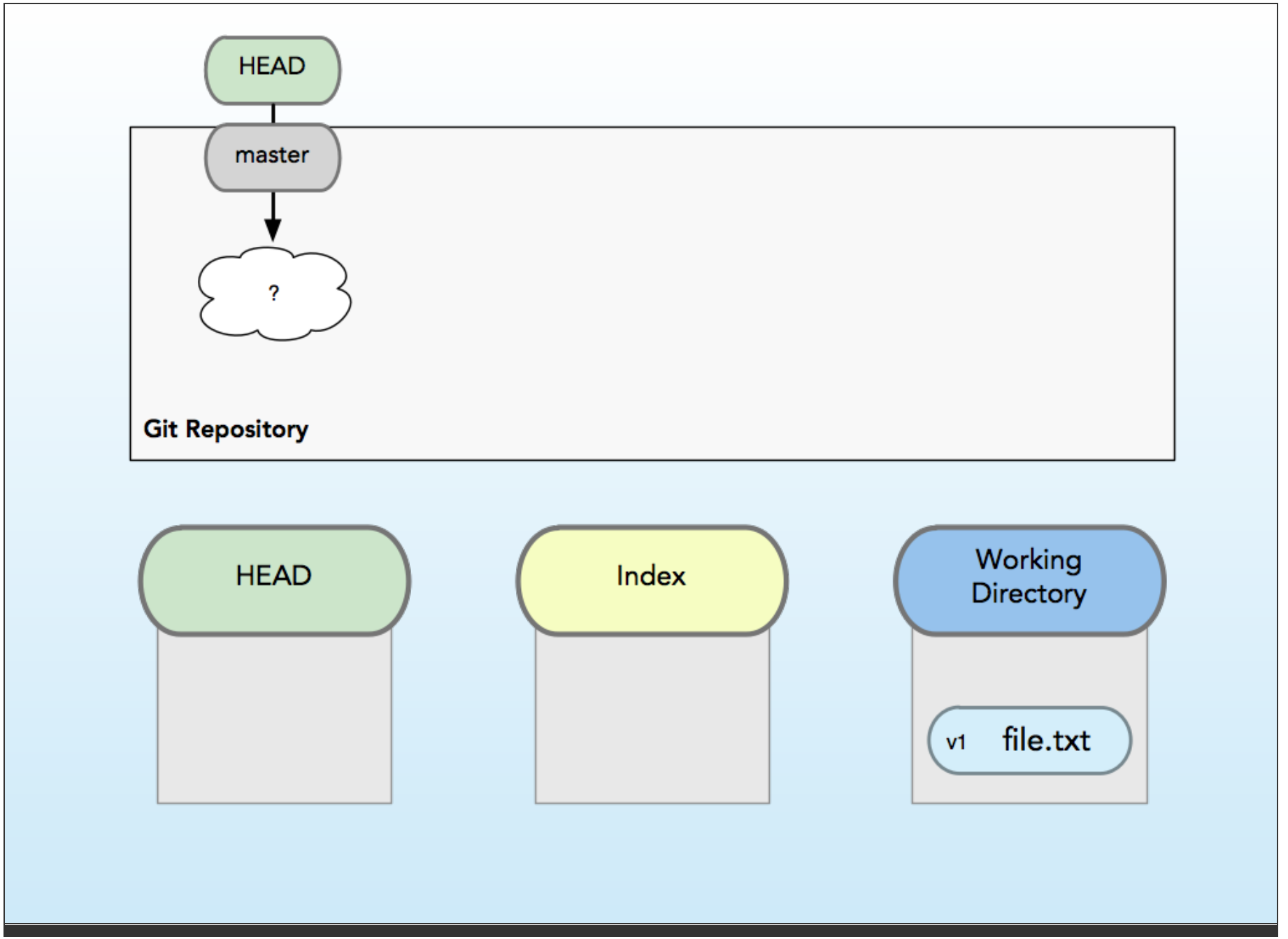
Working With Trees

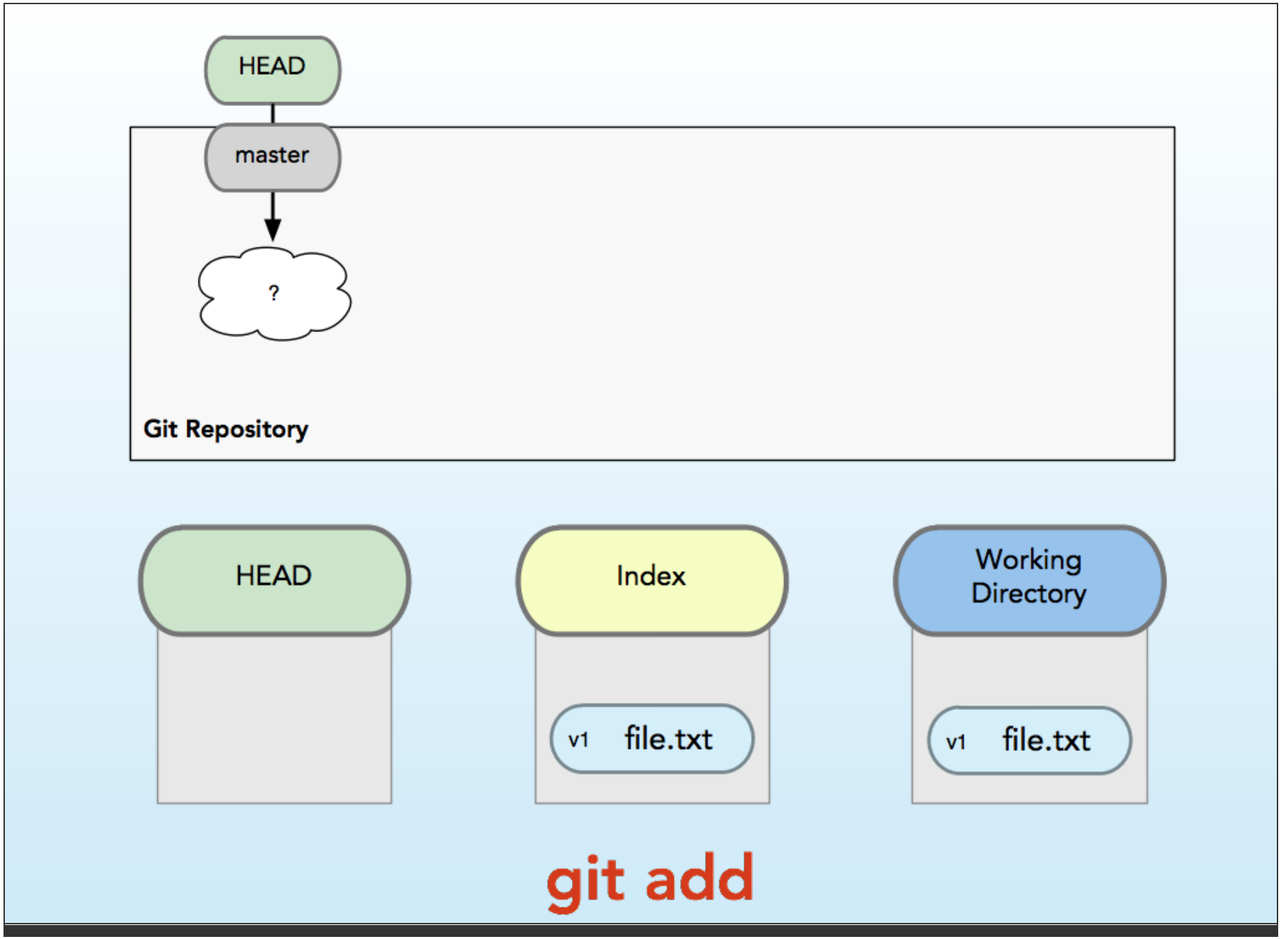
git status

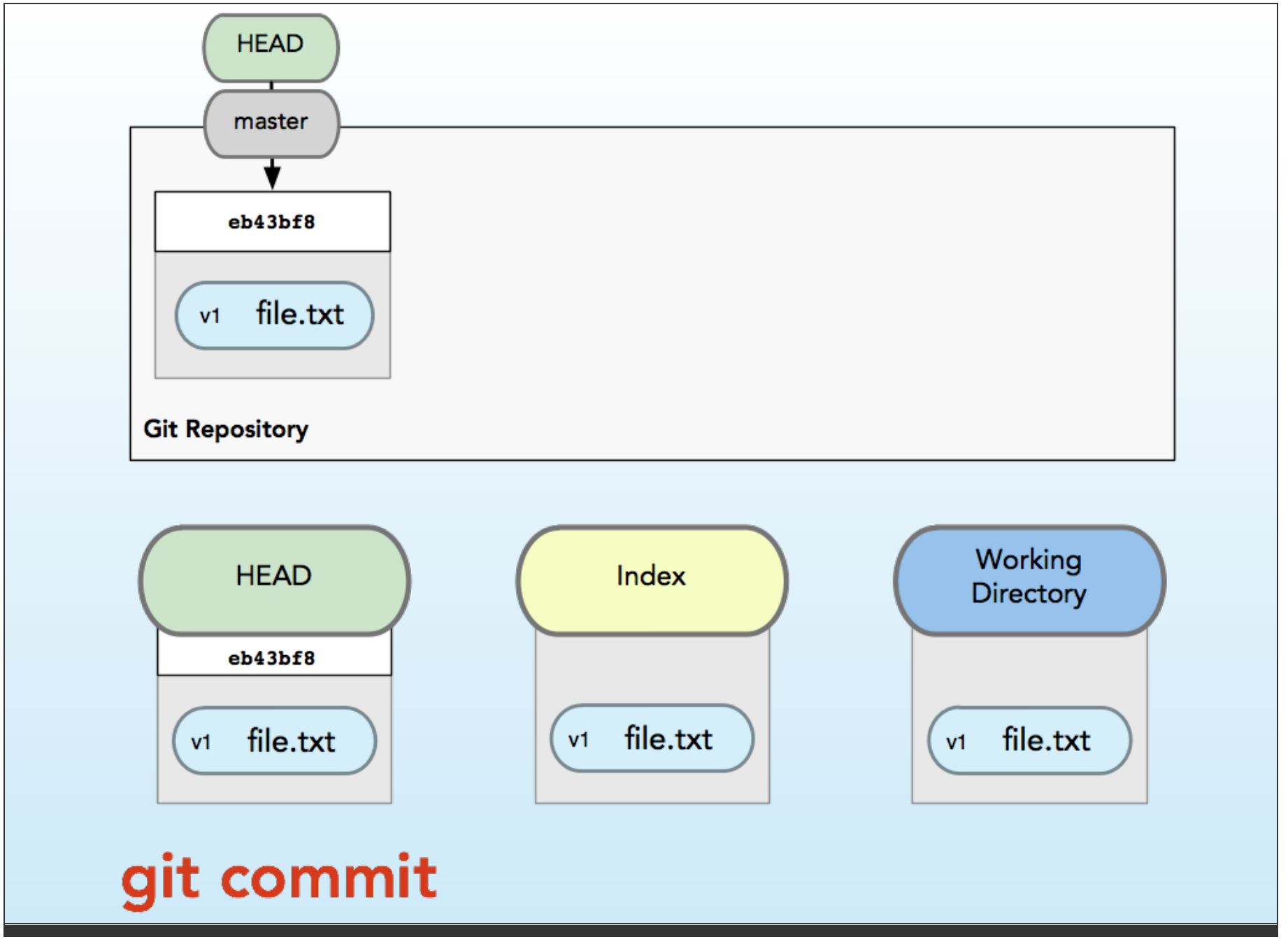
```
$ git status
# On branch master
# Your branch is behind 'origin/master' by 2 commits,
#  and can be fast-forwarded.
#
# Changes to be committed:
#   (use "git reset HEAD ..." to unstage)
#
#       modified:   jobs/email_reply.rb
#
# Changed but not updated:
#   (use "git add ..." to update what will be committed)
#   (use "git checkout -- ..." to discard changes
#     in working directory)
#
#       modified:   app/helpers/users_helper.rb
#       modified:   test/unit/email_reply_job_test.rb
#
```

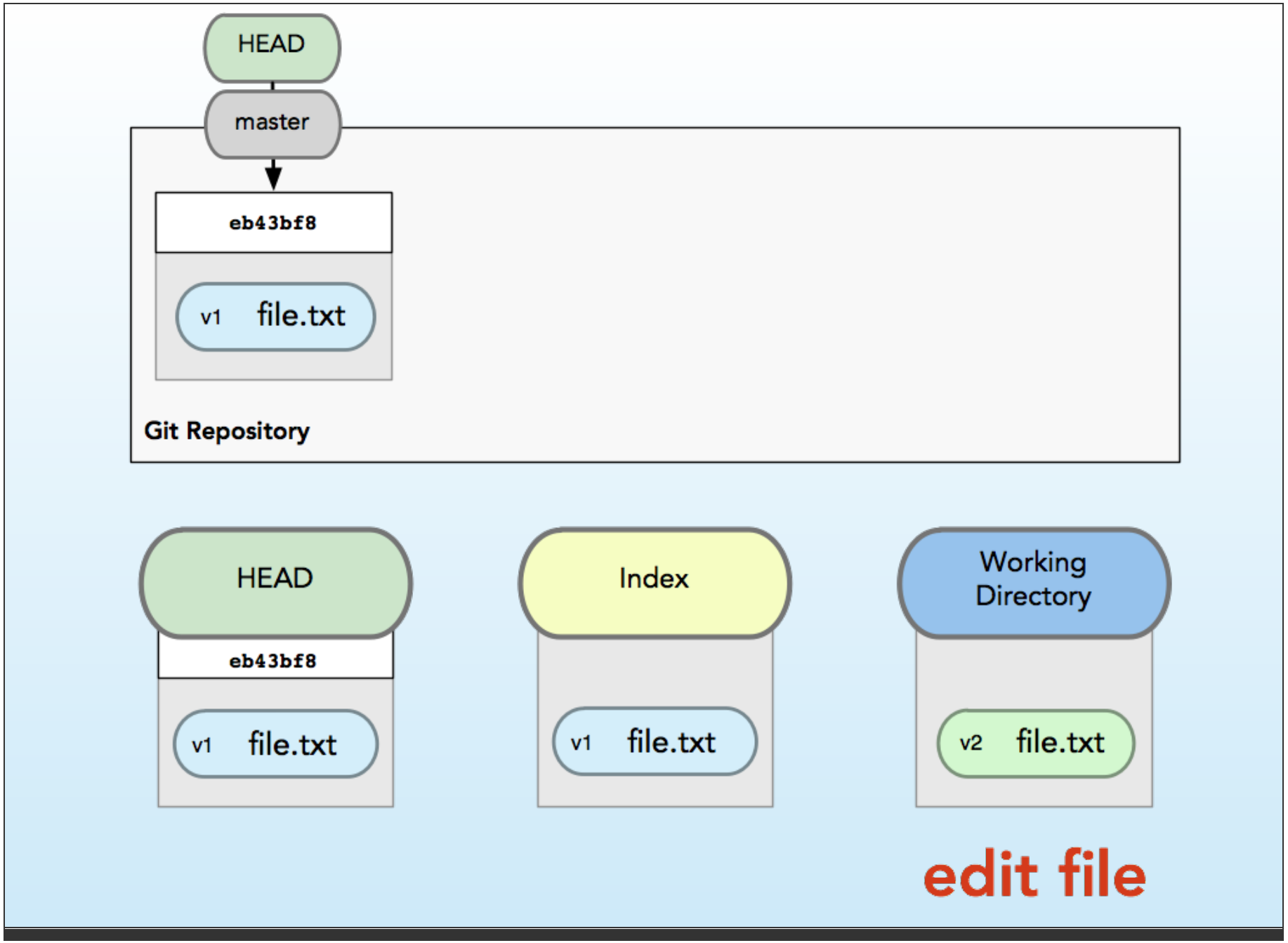
```
$ git status
# On branch master
# Your branch is behind 'origin/master' by 2 commits,
#  and can be fast-forwarded.
#
# Changes to be committed:
#   HEAD and index differ
#
#       modified:   jobs/email_reply.rb
#
# Changed but not updated:
#   index and working directory differ
#
#
#
#       modified:   app/helpers/users_helper.rb
#       modified:   test/unit/email_reply_job_test.rb
#
```


Working
Directory
v1
file.txt

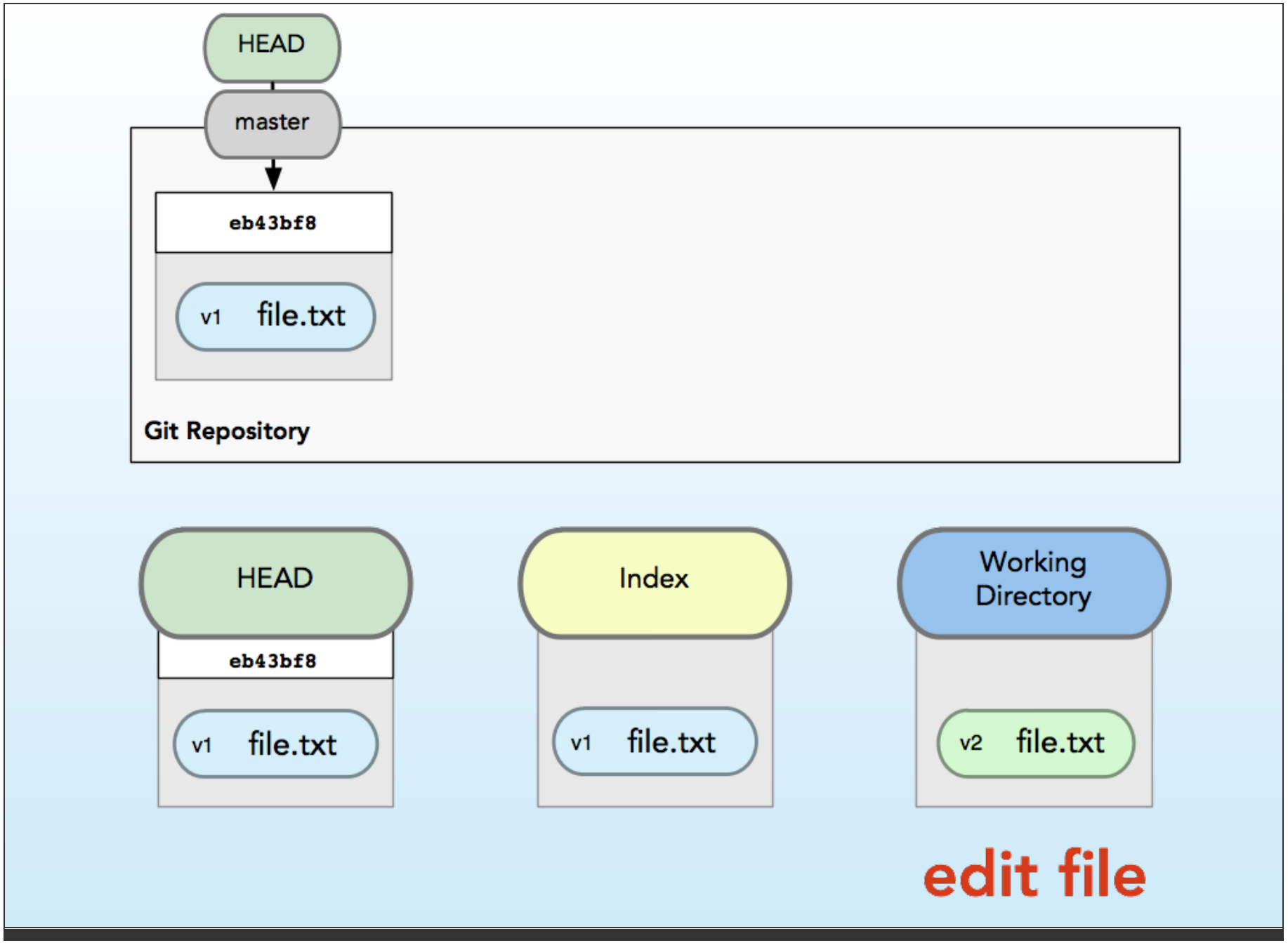

HEAD
master
?
Git Repository
HEAD
Index
Working Directory
v1
file.txt

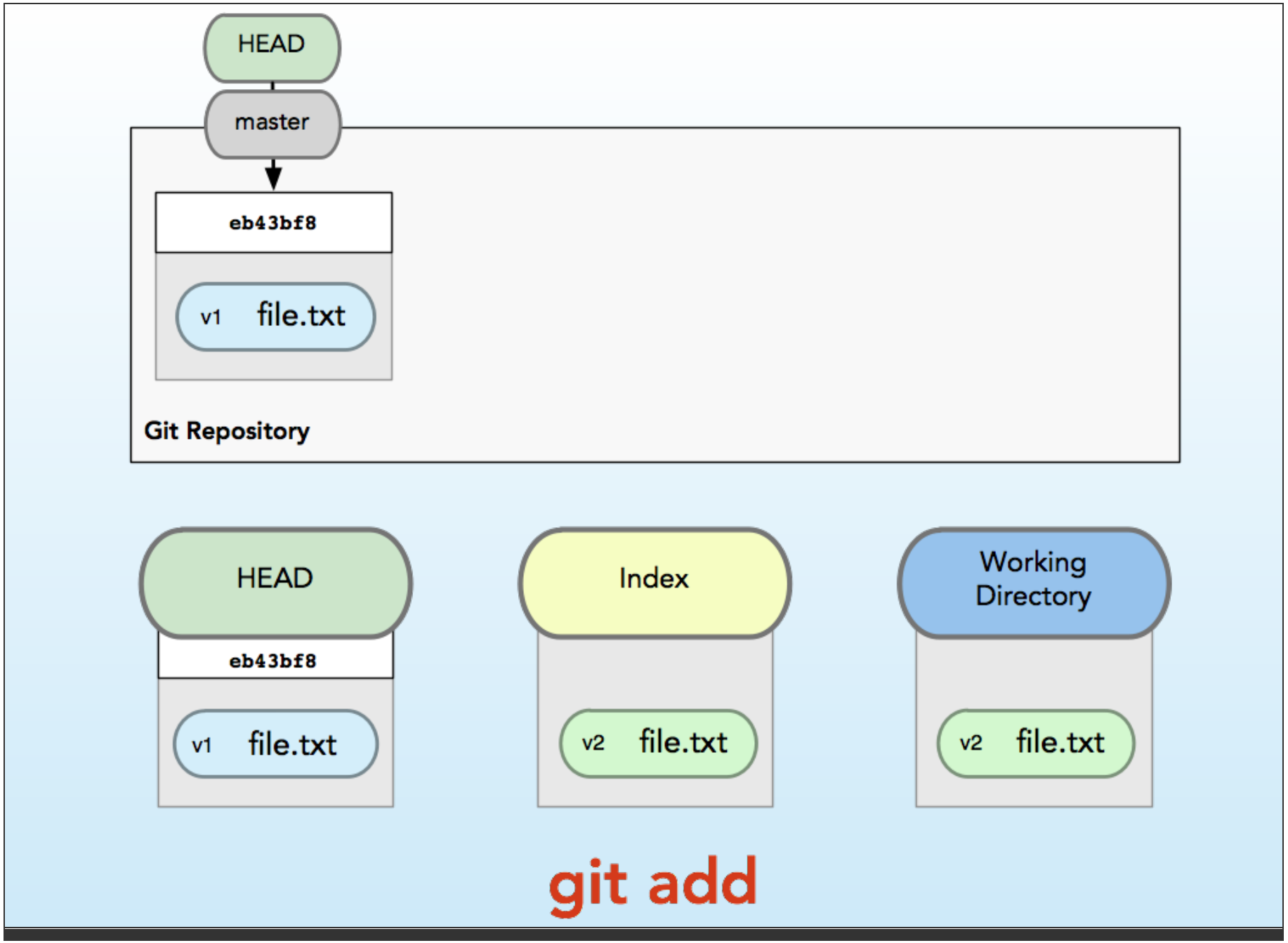

HEAD
master
?
Git Repository
HEAD
Index
v1 file.txt
Working Directory
v1 file.txt
git add


HEAD
master
eb43bf8
v1 file.txt
Git Repository
HEAD
eb43bf8
v1 file.txt
Index
v1 file.txt
Working Directory
v1 file.txt
git commit

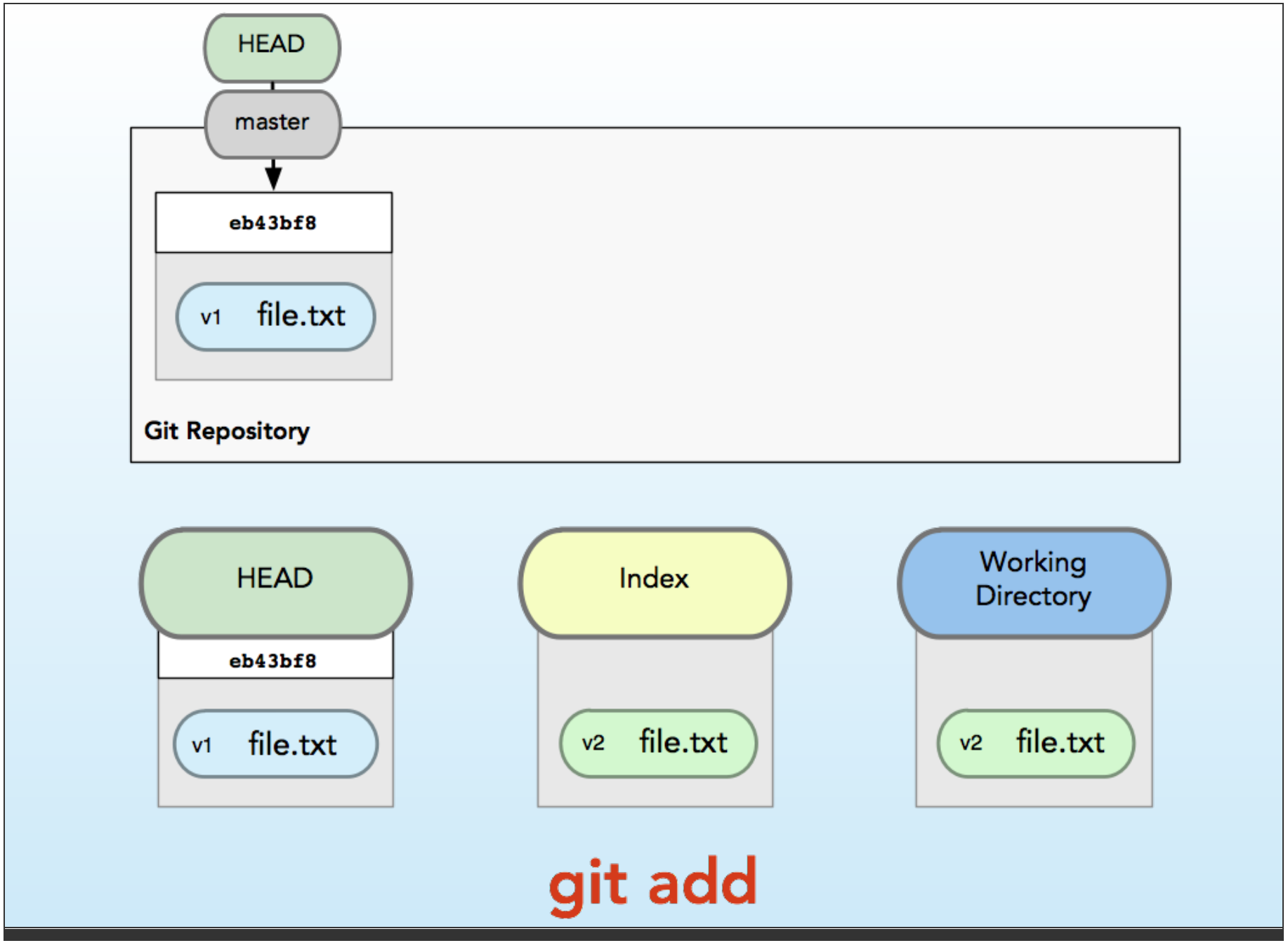

HEAD
master
eb43bf8
v1 file.txt
Git Repository
HEAD
eb43bf8
v1 file.txt
Index
v1 file.txt
Working Directory
v2 file.txt
edit file

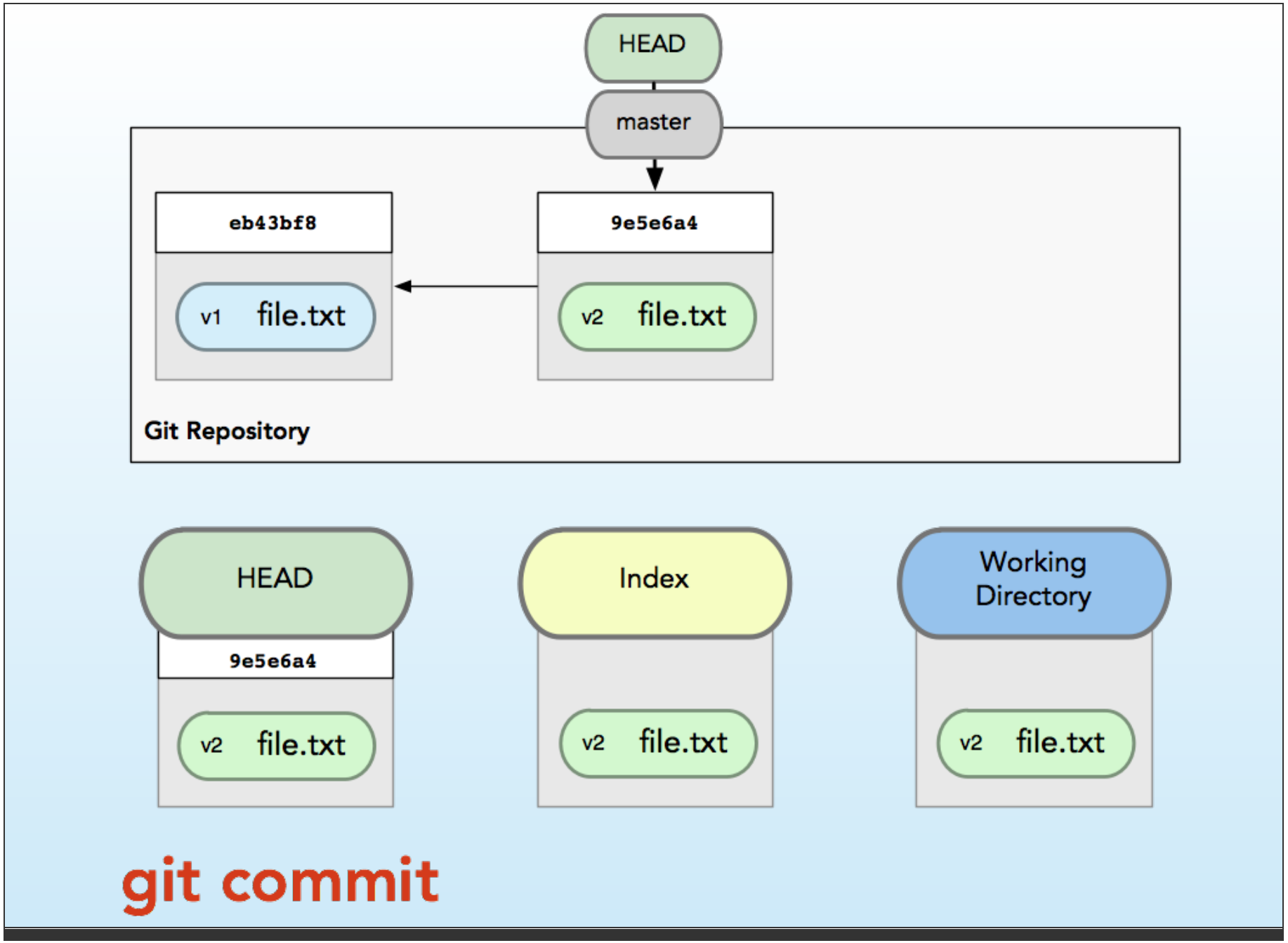
```
$ git status
# On branch master
# Your branch is behind 'origin/mas
#  and can be fast-forwarded.
#
# Changed but not updated:
#   (use "git add ..." to update wh
#   (use "git checkout -- ..." to d
#     in working directory)
#
#       modified:   file.txt
#
```


HEAD
master
eb43bf8
v1 file.txt
Git Repository
HEAD
eb43bf8
v1 file.txt
Index
v1 file.txt
Working Directory
v2 file.txt
edit file


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

```
$ git status
# On branch master
# Your branch is behind 'origin/mas
#  and can be fast-forwarded.
#
# Changes to be committed:
#   (use "git reset HEAD ..." to un
#
#       modified:   file.txt
#
```


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


HEAD
master
eb43bf8
v1 file.txt
9e5e6a4
v2 file.txt
Git Repository
HEAD
9e5e6a4
v2 file.txt
Index
v2 file.txt
Working Directory
v2 file.txt
git commit

git reset

2 forms

```
git reset [commit] [path]

git reset [commit]
```

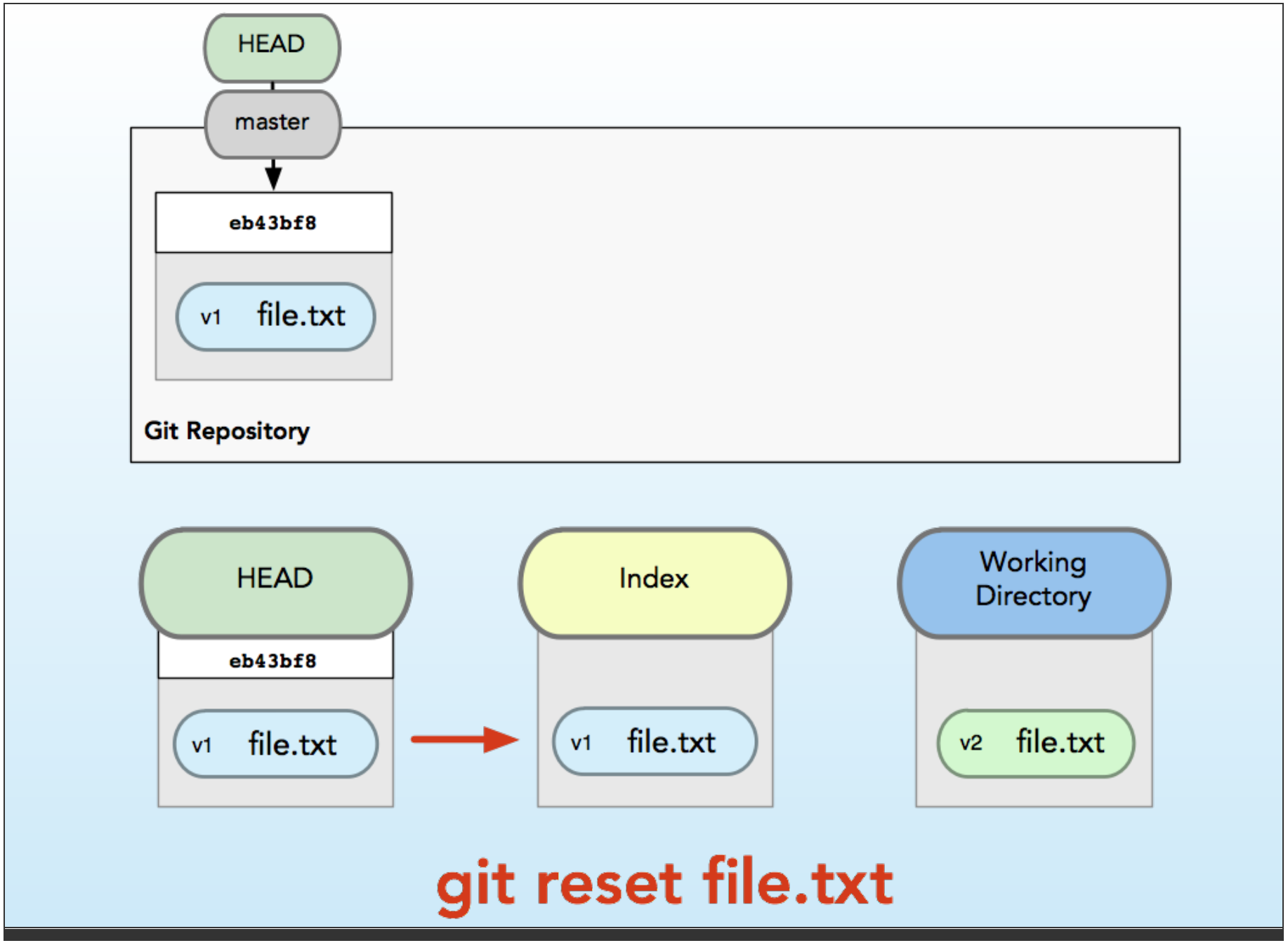
1. Path Form

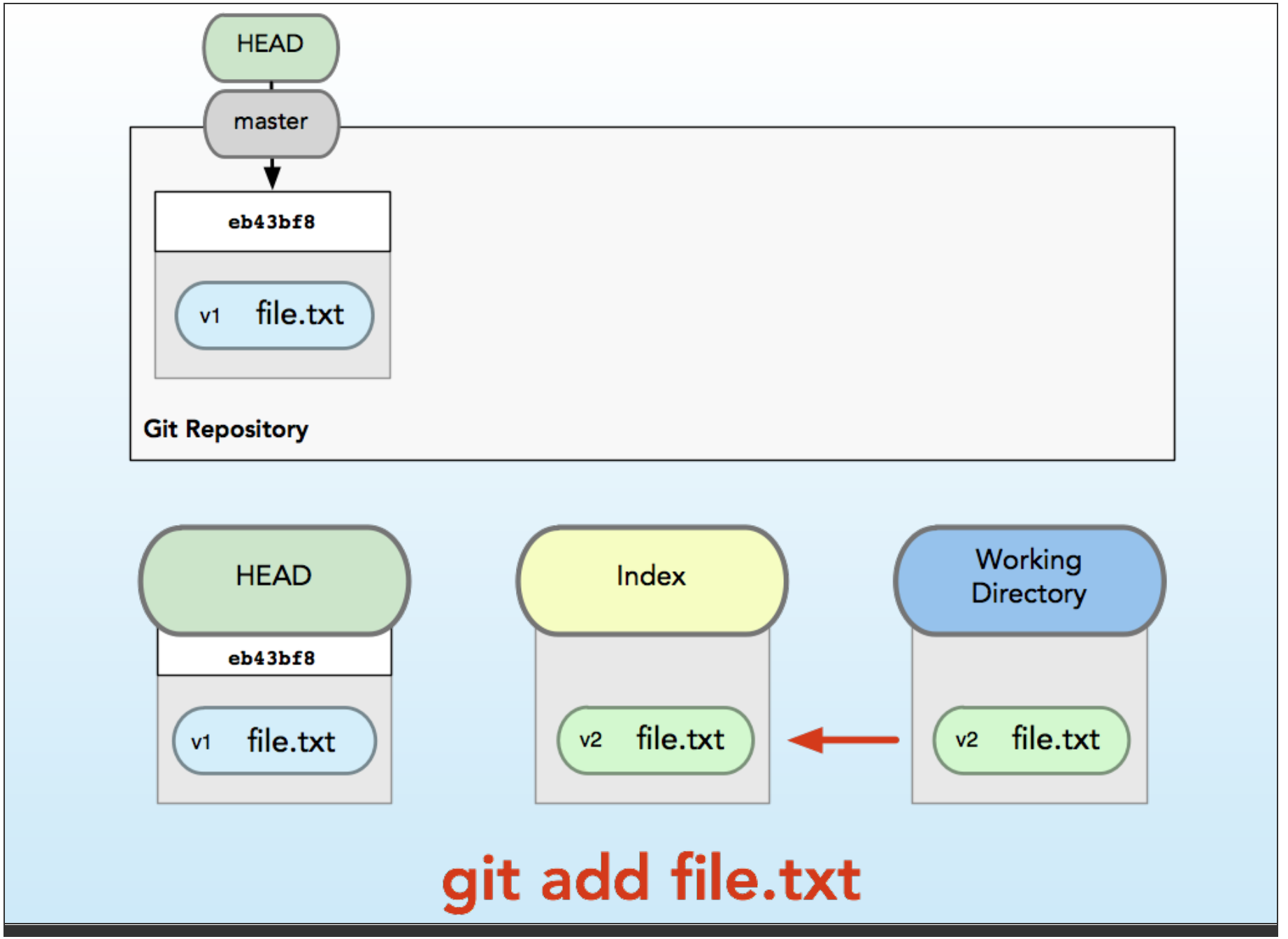
```
git reset [commit] [path]
```

`git reset [file]`

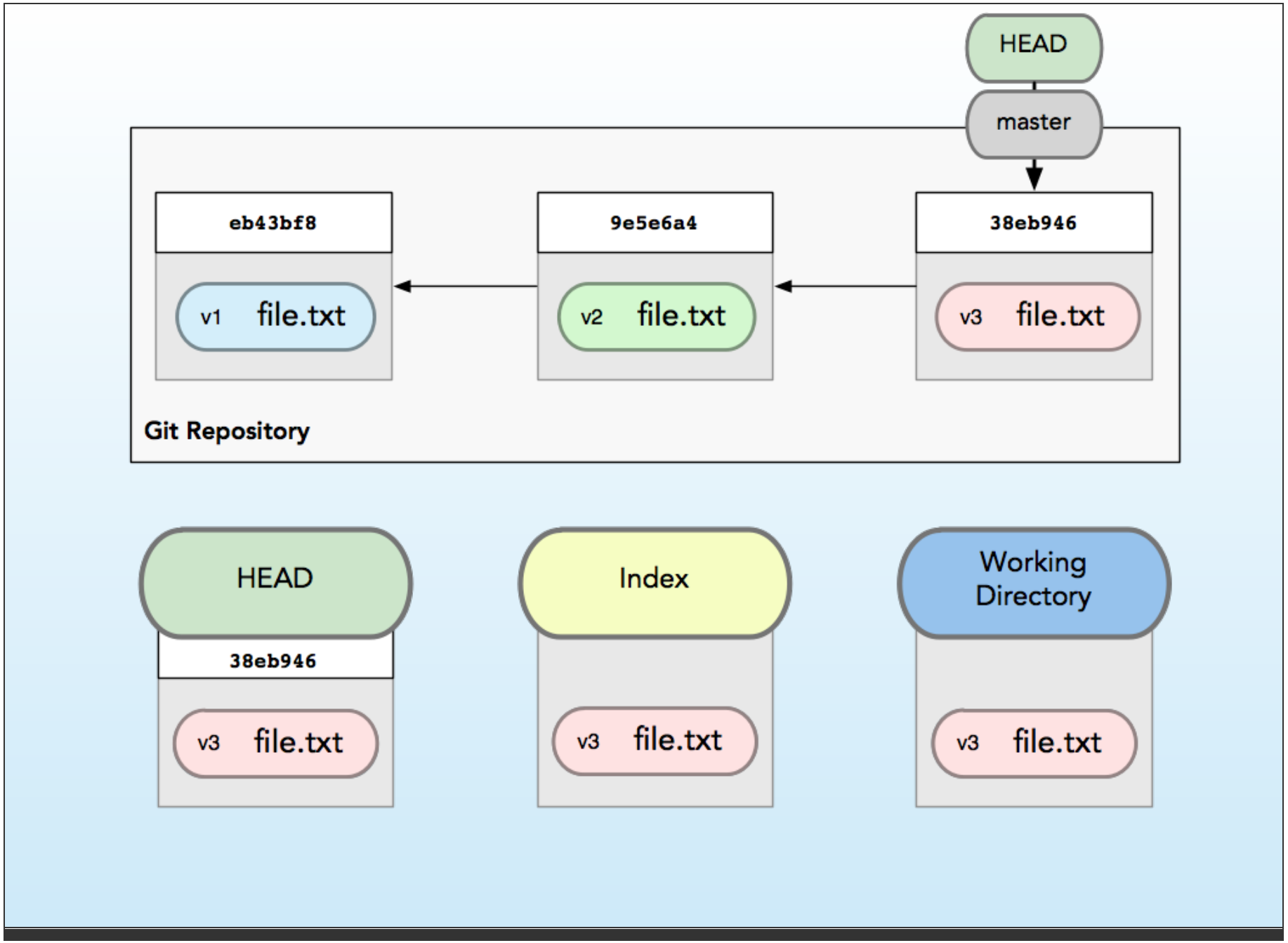
is the opposite of

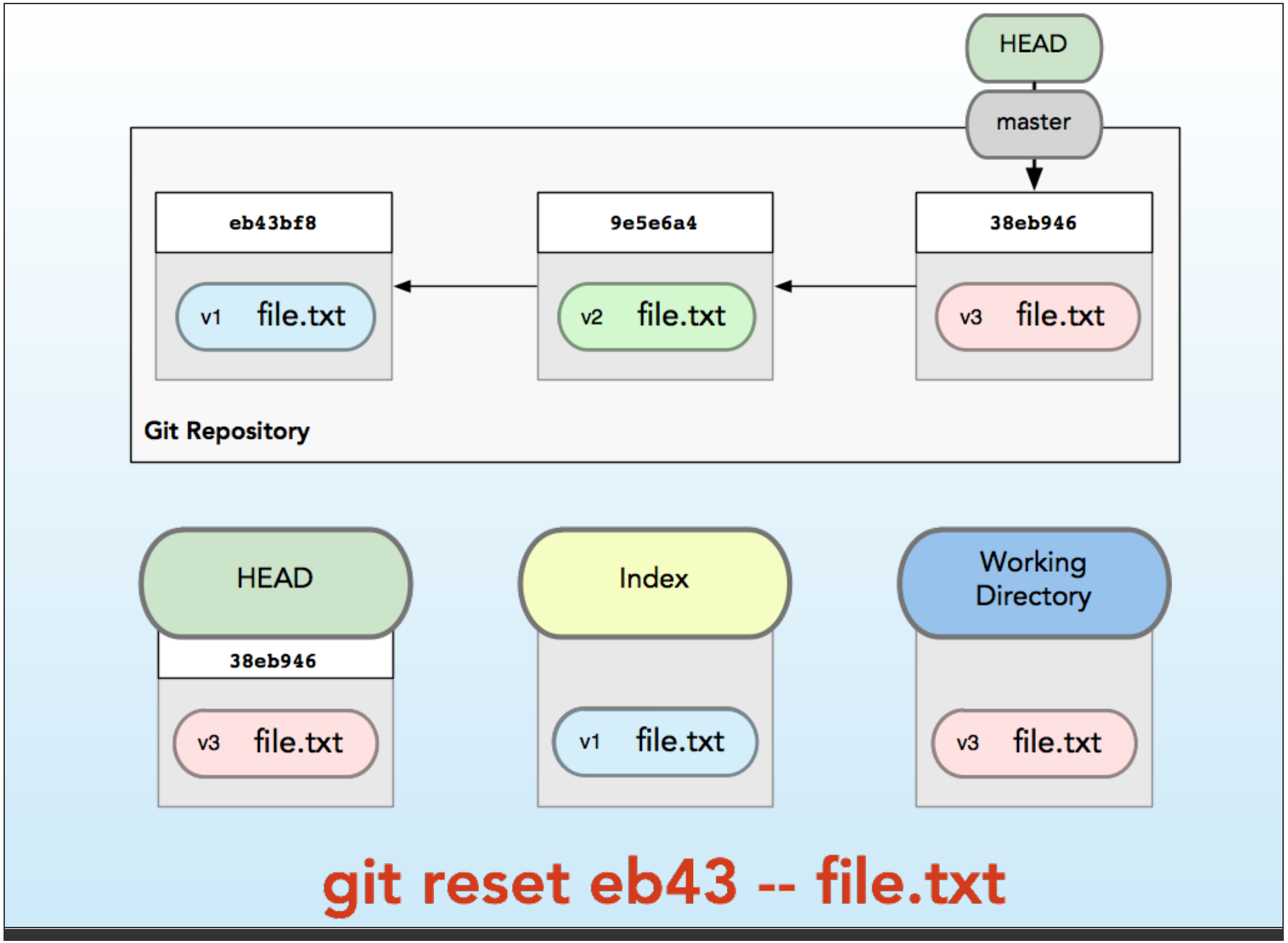
`git add [file]`


HEAD
master
eb43bf8
v1 file.txt
Git Repository
HEAD
eb43bf8
v1 file.txt
Index
v1 file.txt
Working Directory
v2 file.txt
git reset file.txt


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

Reset to an older file


HEAD
master
eb43bf8
9e5e6a4
38eb946
v1 file.txt
v2 file.txt
v3 file.txt
Git Repository
HEAD
38eb946
v3 file.txt
Index
v3 file.txt
Working Directory
v3 file.txt


HEAD
master
eb43bf8
9e5e6a4
38eb946
v1 file.txt
v2 file.txt
v3 file.txt
Git Repository
HEAD
38eb946
v3 file.txt
Index
v1 file.txt
Working Directory
v3 file.txt
git reset eb43 -- file.txt

```
$ git status
# On branch master
# Changes to be committed:
#   (use "git reset HEAD ..." to un
#
#       modified:   file.txt
#
# Changed but not updated:
#   (use "git add ..." to update wh
#   (use "git checkout -- ..." to d
#
#       modified:   file.txt
#
```

2. Commit Form

git reset [commit]

Reset Options

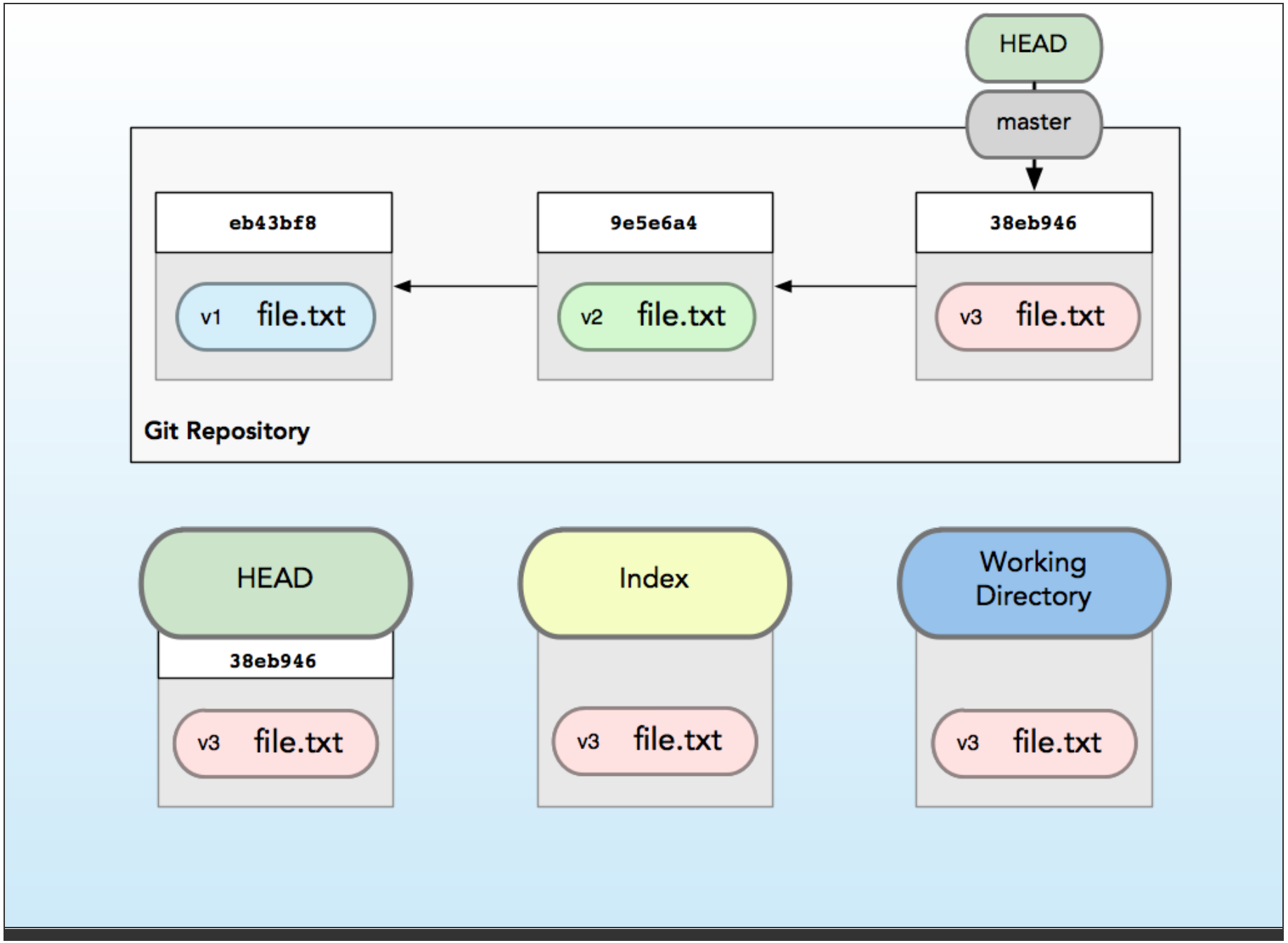
--soft move HEAD to target

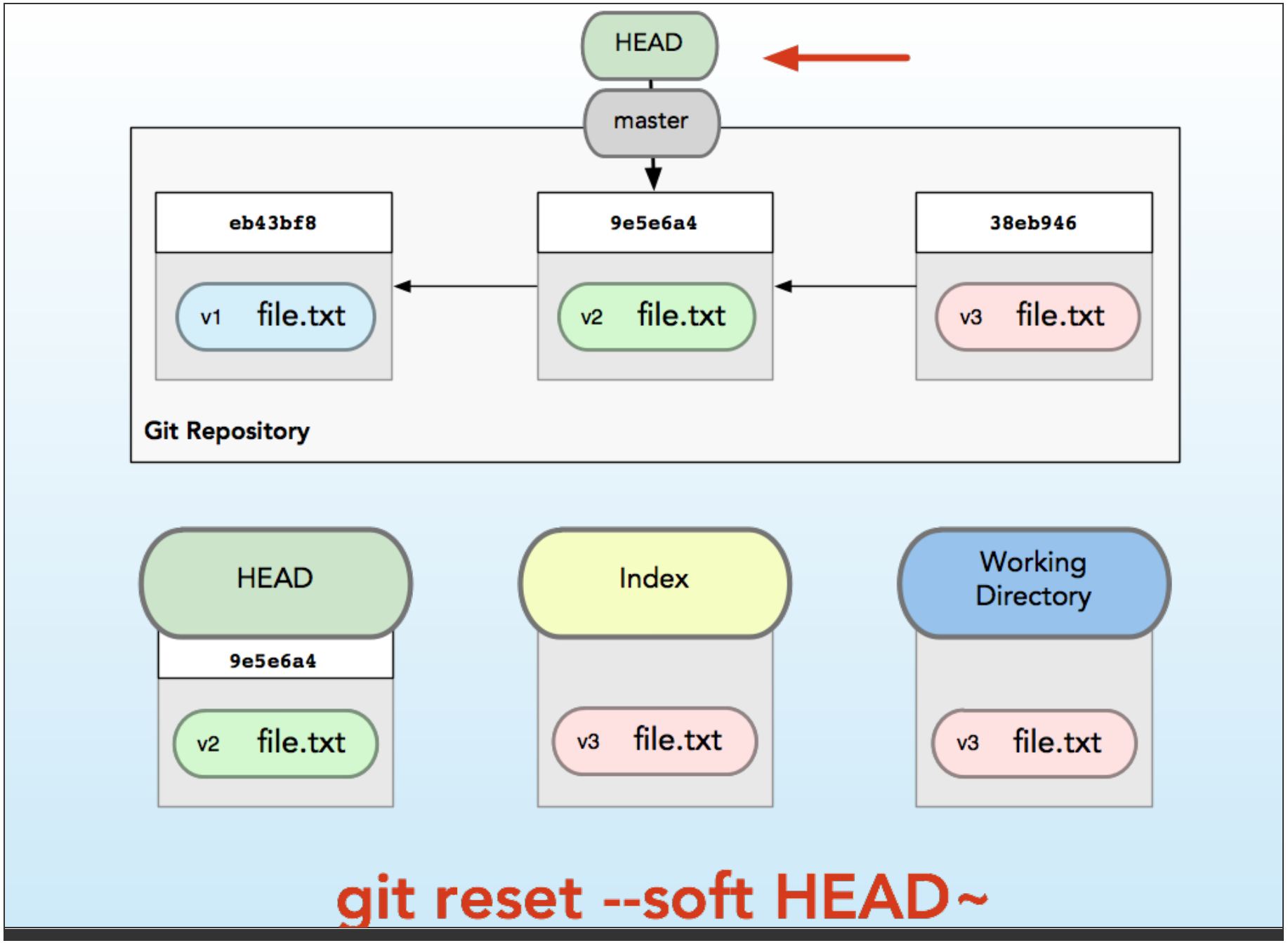
[--mixed] then copy to index

--hard then copy to work dir

--soft

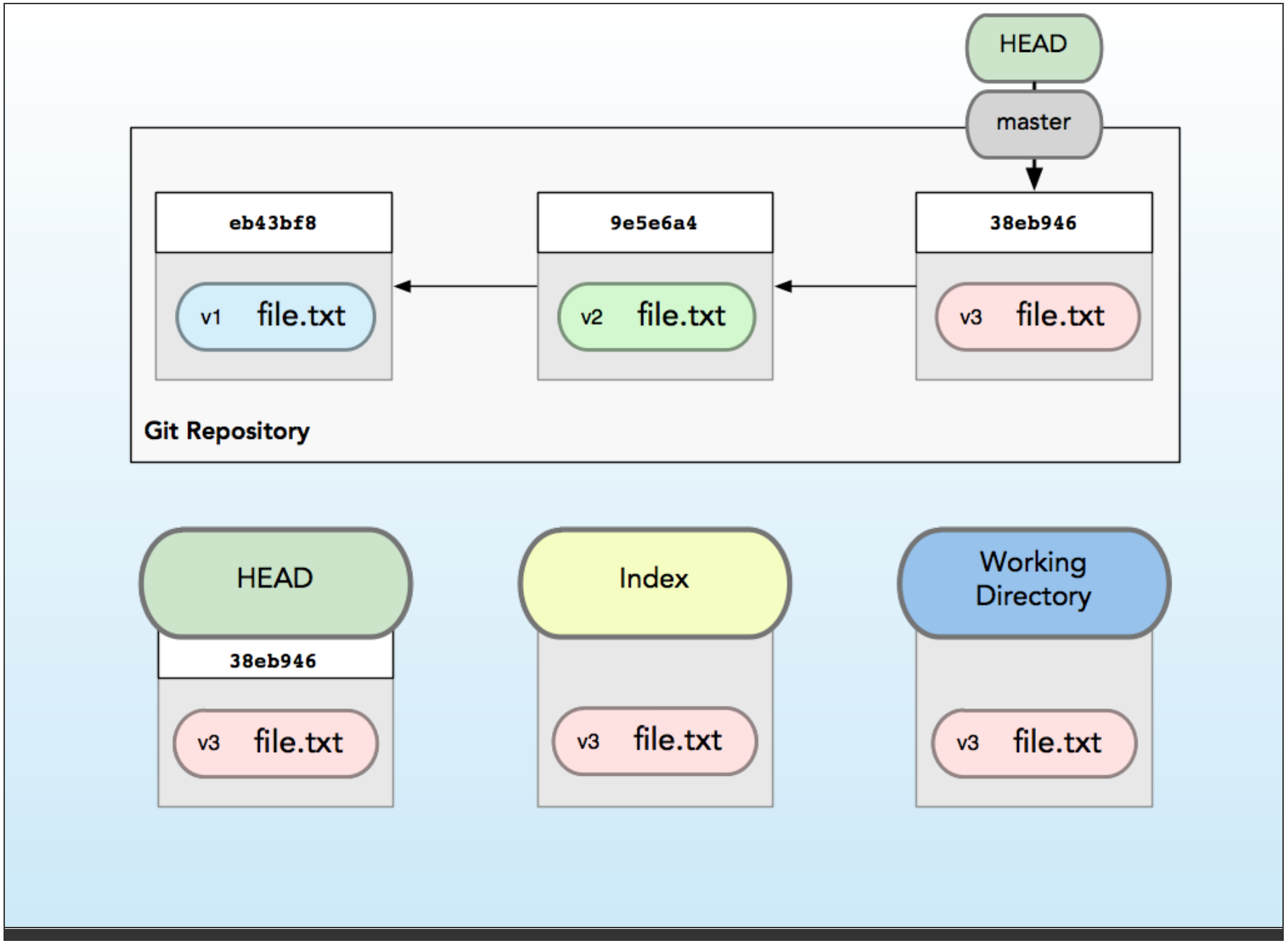
move HEAD to another commit

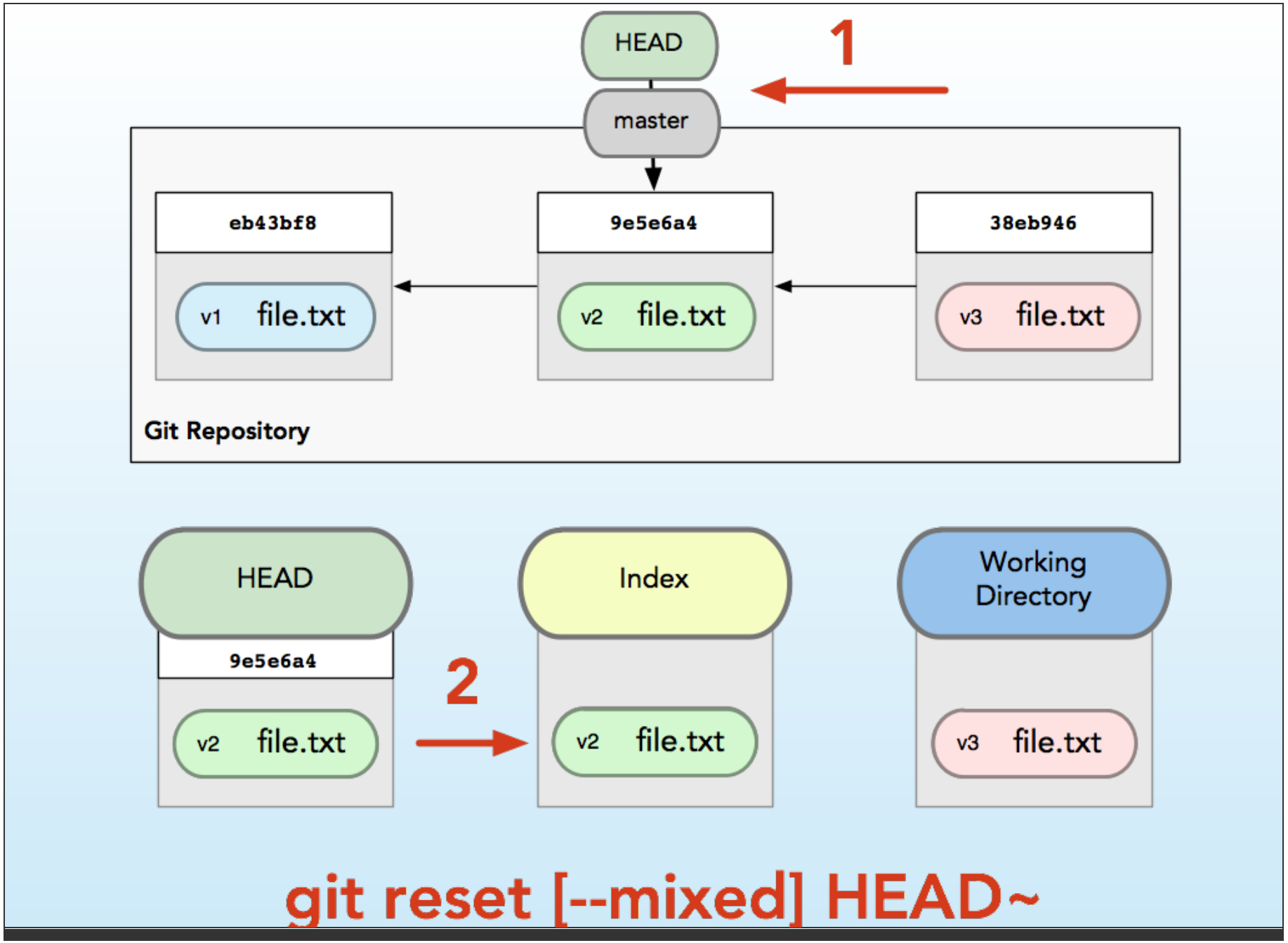

HEAD
master
eb43bf8
9e5e6a4
38eb946
v1 file.txt
v2 file.txt
v3 file.txt
Git Repository
HEAD
38eb946
v3 file.txt
Index
v3 file.txt
Working Directory
v3 file.txt


HEAD
master
eb43bf8
v1 file.txt
9e5e6a4
v2 file.txt
38eb946
v3 file.txt
Git Repository
HEAD
9e5e6a4
v2 file.txt
Index
v3 file.txt
Working Directory
v3 file.txt
git reset --soft HEAD~

--mixed

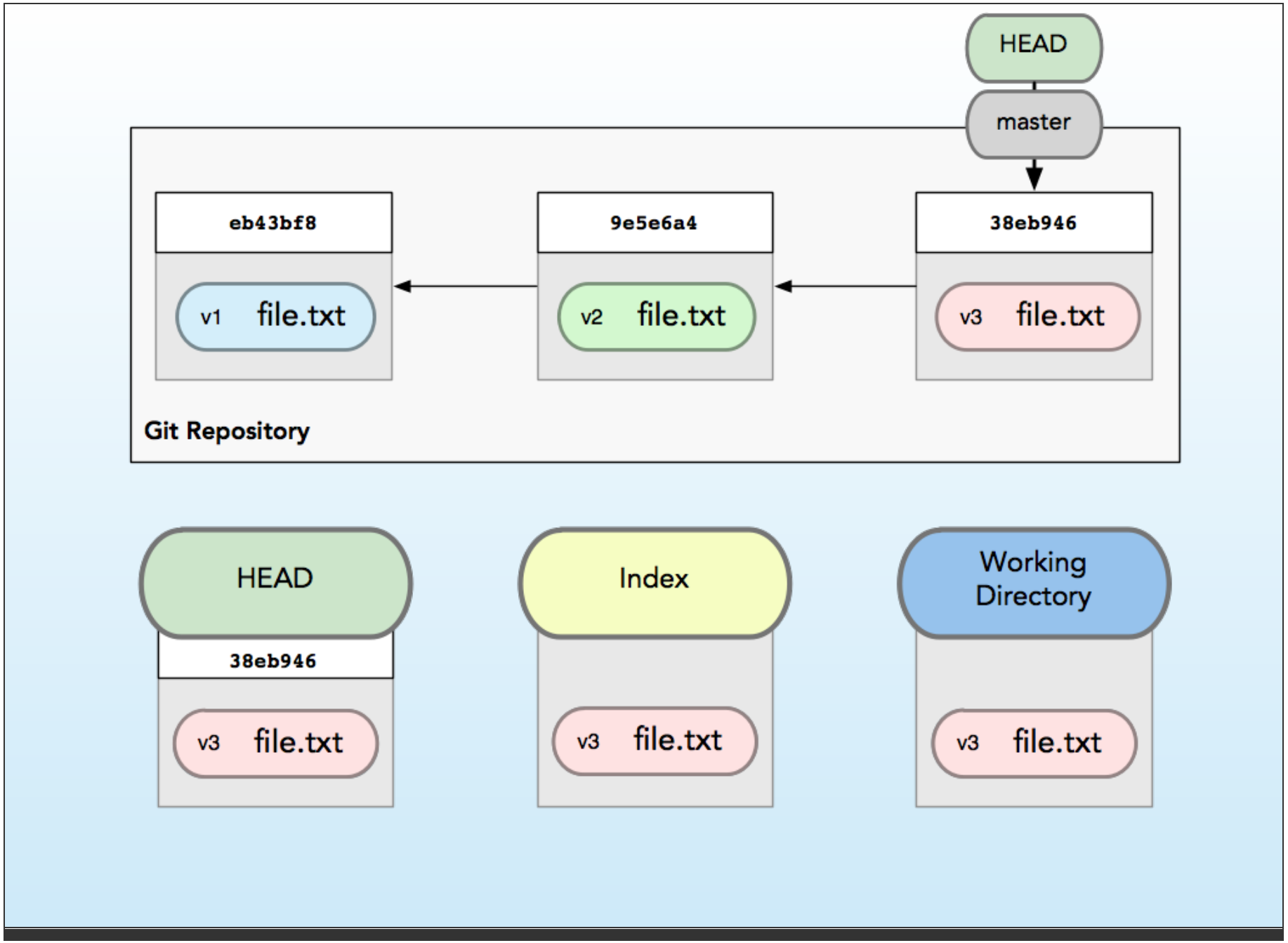
move HEAD to another commit, then copy into index

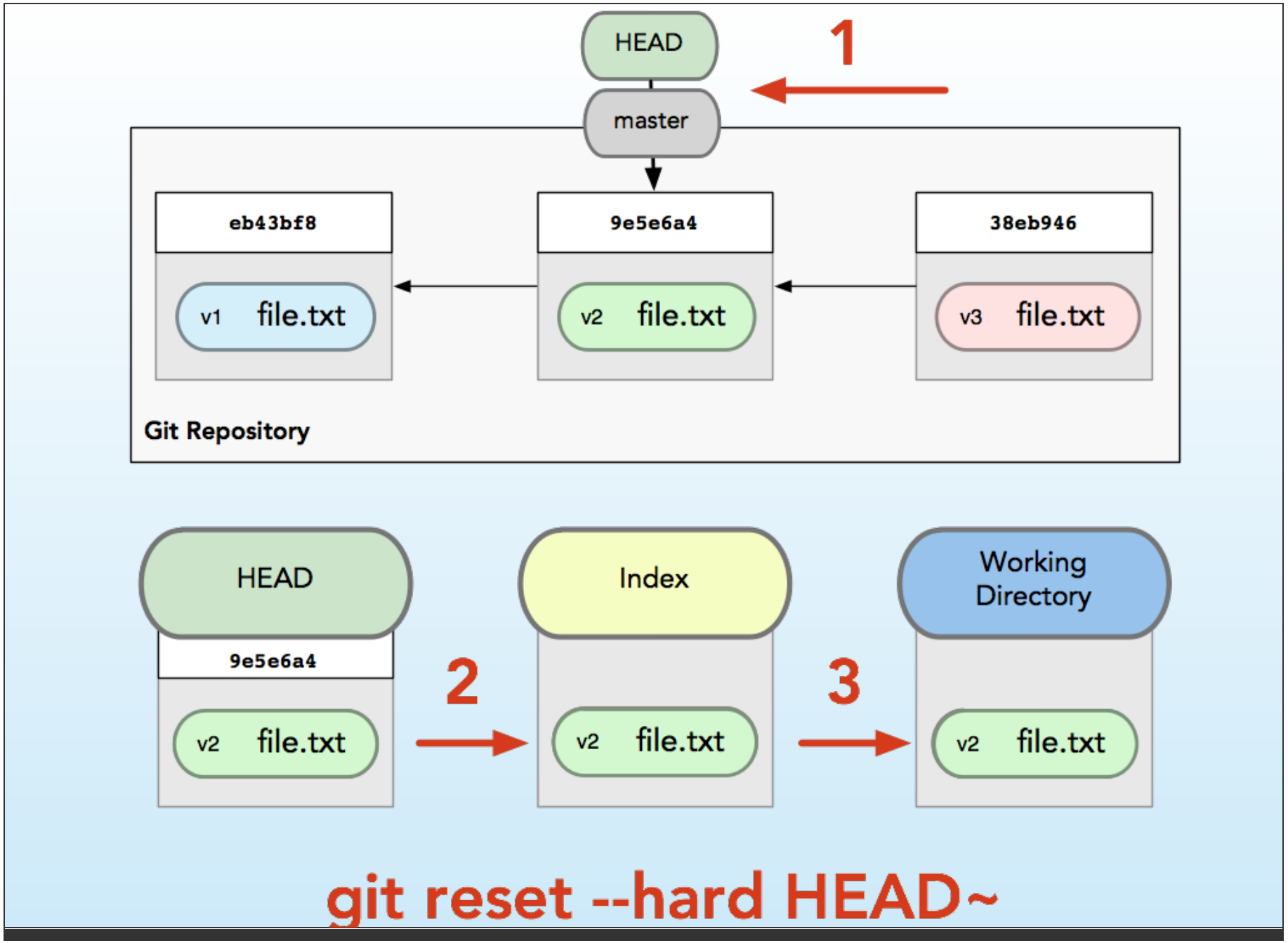

HEAD
master
eb43bf8
9e5e6a4
38eb946
v1 file.txt
v2 file.txt
v3 file.txt
Git Repository
HEAD
38eb946
v3 file.txt
Index
v3 file.txt
Working Directory
v3 file.txt


HEAD
1
master
eb43bf8
9e5e6a4
38eb946
v1 file.txt
v2 file.txt
v3 file.txt
Git Repository
HEAD
9e5e6a4
v2 file.txt
2
Index
v2 file.txt
Working Directory
v3 file.txt
git reset [--mixed] HEAD~

--hard

move HEAD, copy to index, copy to working directory


HEAD
master
eb43bf8
9e5e6a4
38eb946
v1 file.txt
v2 file.txt
v3 file.txt
Git Repository
HEAD
38eb946
v3 file.txt
Index
v3 file.txt
Working Directory
v3 file.txt


HEAD
1
master
eb43bf8
9e5e6a4
38eb946
v1 file.txt
v2 file.txt
v3 file.txt
Git Repository
HEAD
9e5e6a4
v2 file.txt
2
Index
v2 file.txt
3
Working Directory
v2 file.txt
git reset --hard HEAD~

WTFWIEWTUT

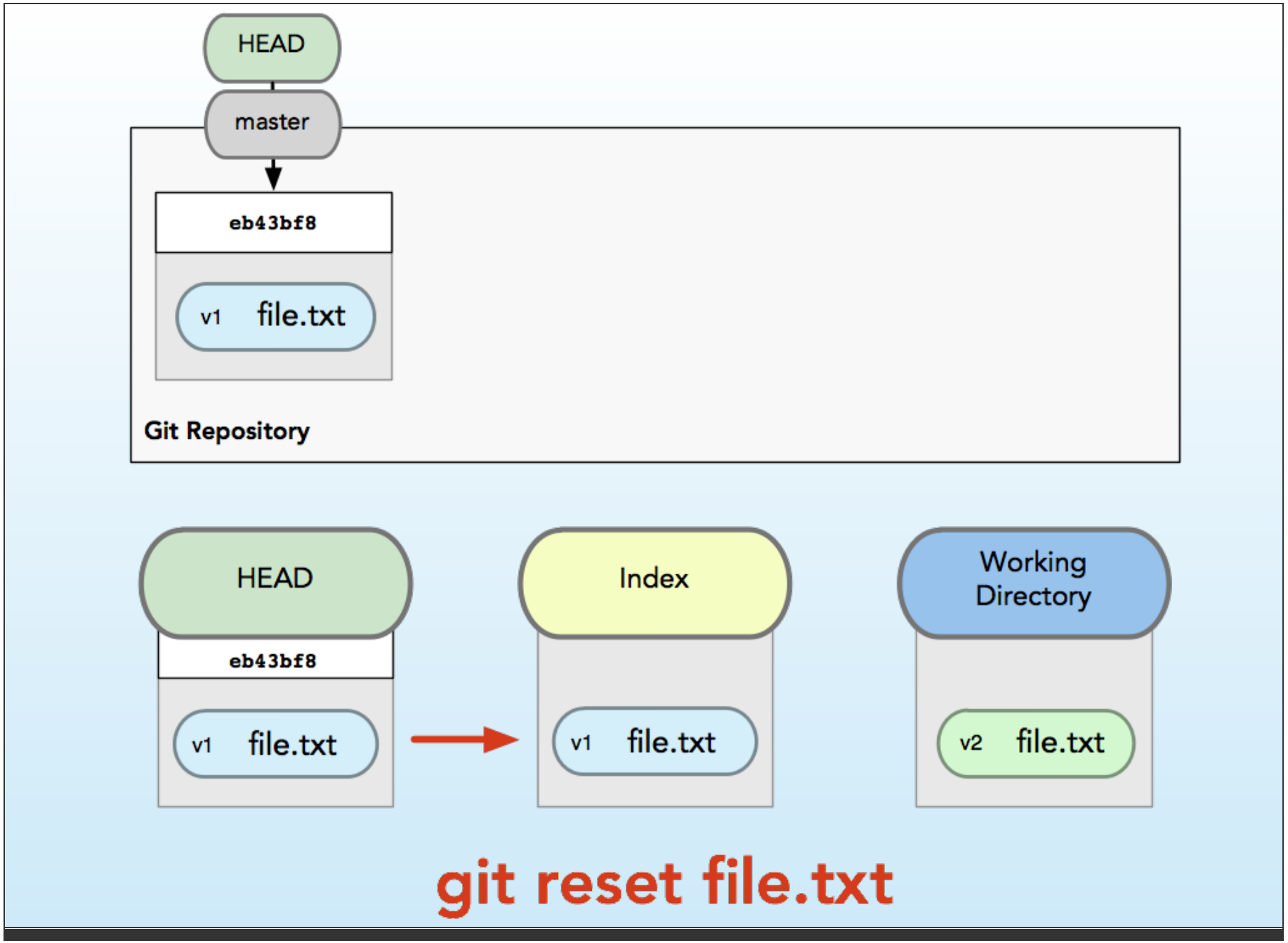
whythefuckwouldieverwanttousethis

unstaging changes

git reset

OR

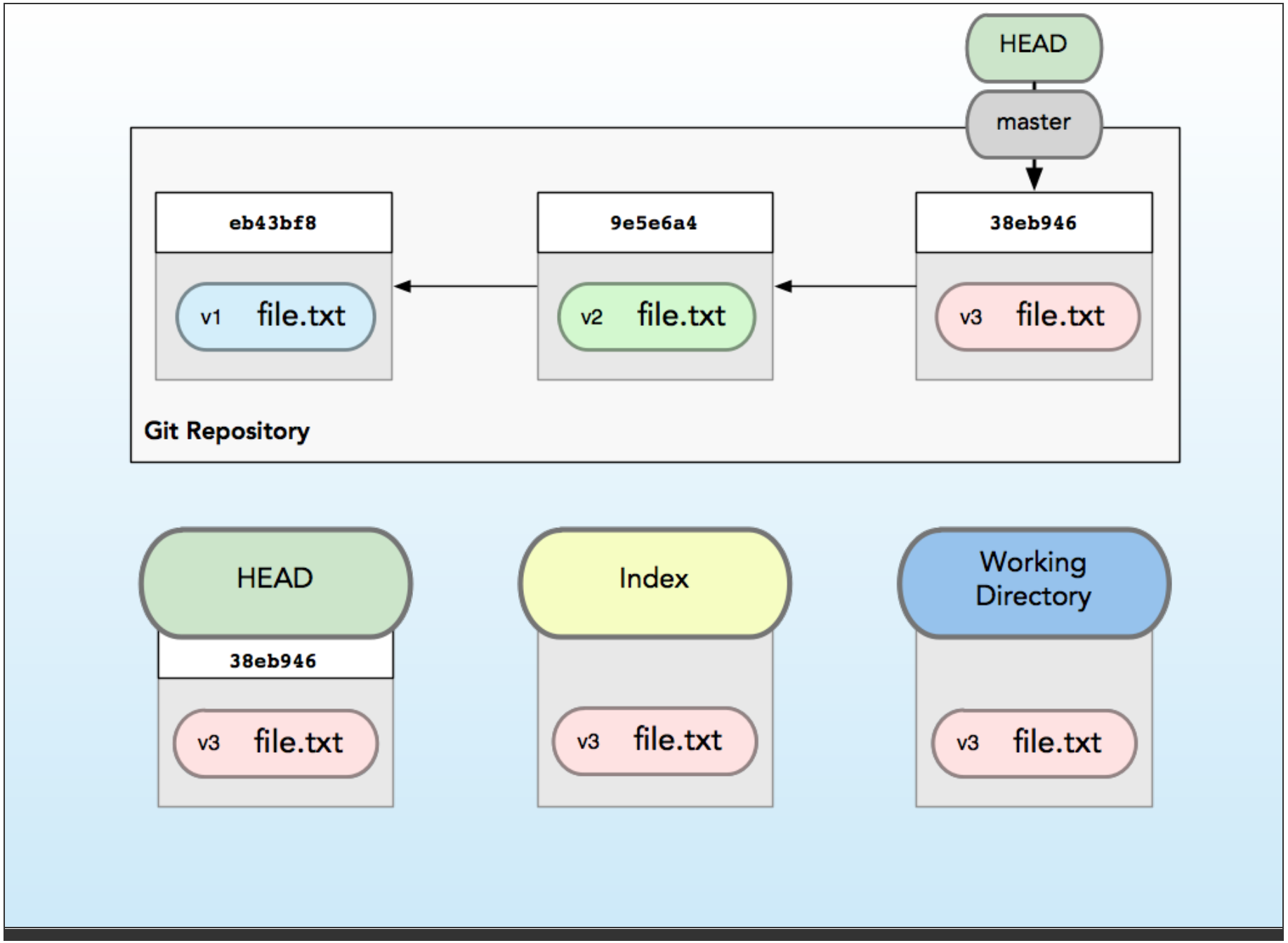
```
git reset [file]
```

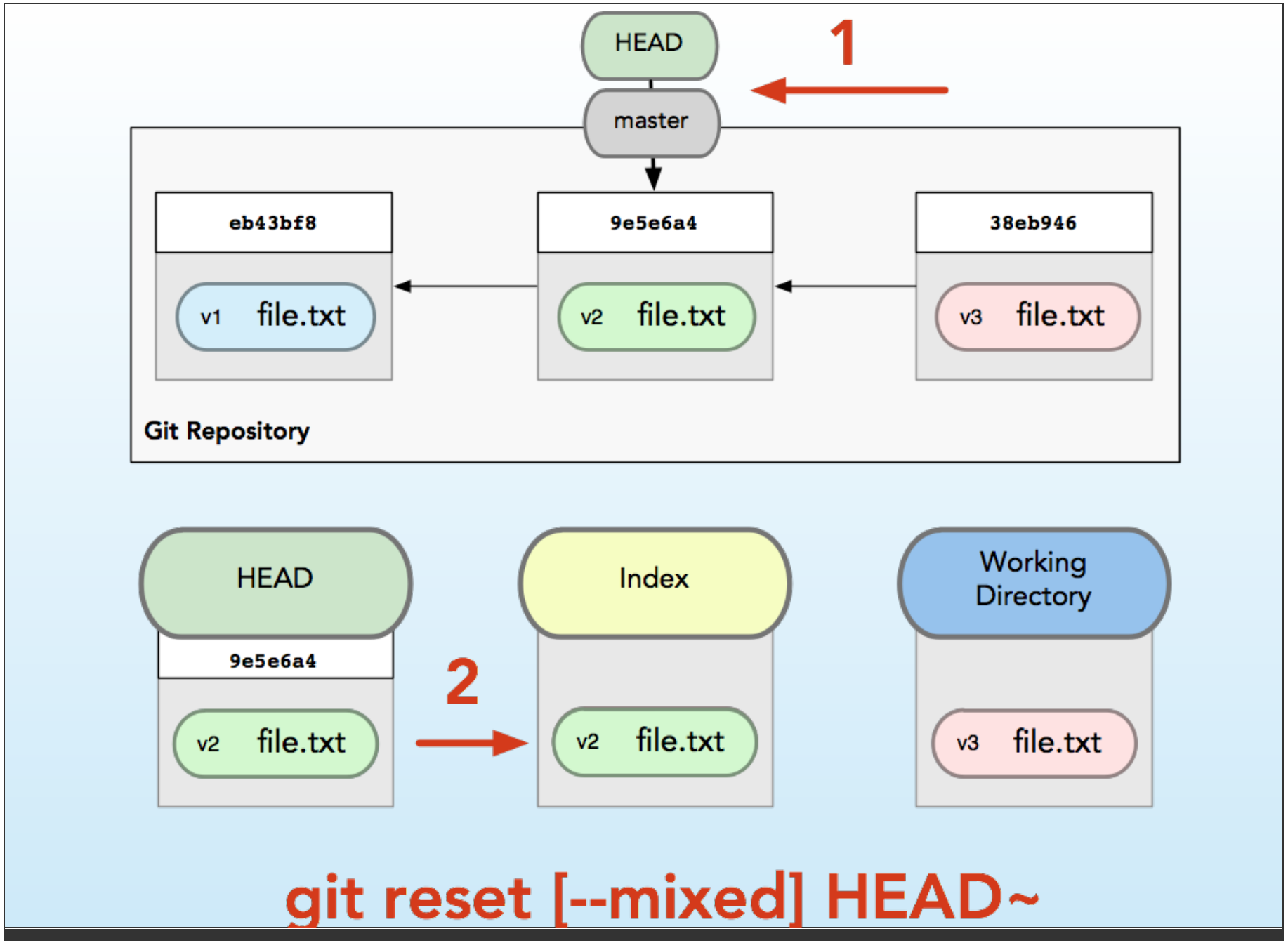

HEAD
master
eb43bf8
v1 file.txt
Git Repository
HEAD
eb43bf8
v1 file.txt
Index
v1 file.txt
Working Directory
v2 file.txt
git reset file.txt

undo last commit

`git reset [--mixed] HEAD~`

moves HEAD back and moves index back


HEAD
master
eb43bf8
9e5e6a4
38eb946
v1 file.txt
v2 file.txt
v3 file.txt
Git Repository
HEAD
38eb946
v3 file.txt
Index
v3 file.txt
Working Directory
v3 file.txt

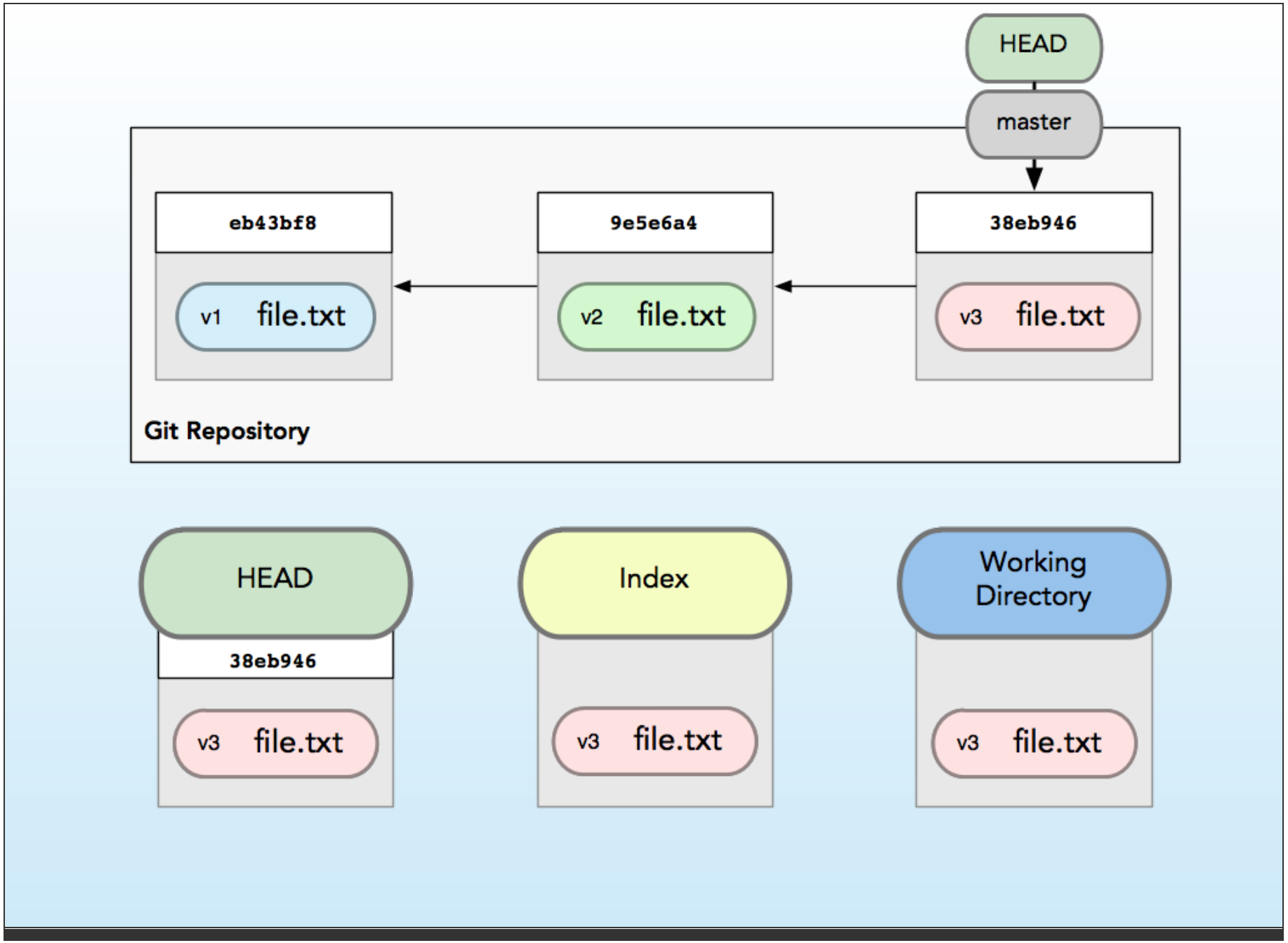

HEAD
1
master
eb43bf8
9e5e6a4
38eb946
v1 file.txt
v2 file.txt
v3 file.txt
Git Repository
HEAD
9e5e6a4
v2 file.txt
2
Index
v2 file.txt
Working Directory
v3 file.txt
git reset [--mixed] HEAD~

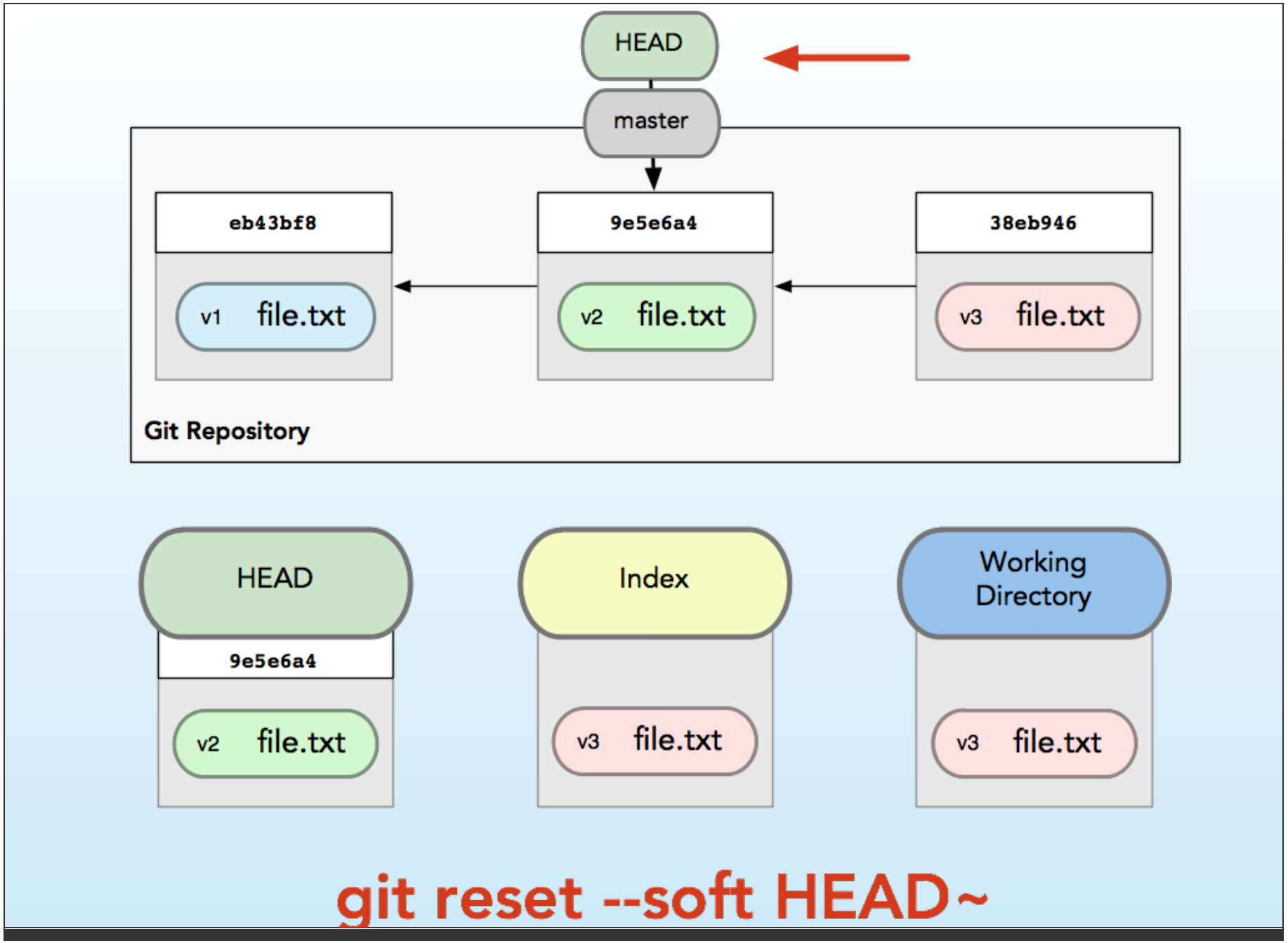
undo last commit

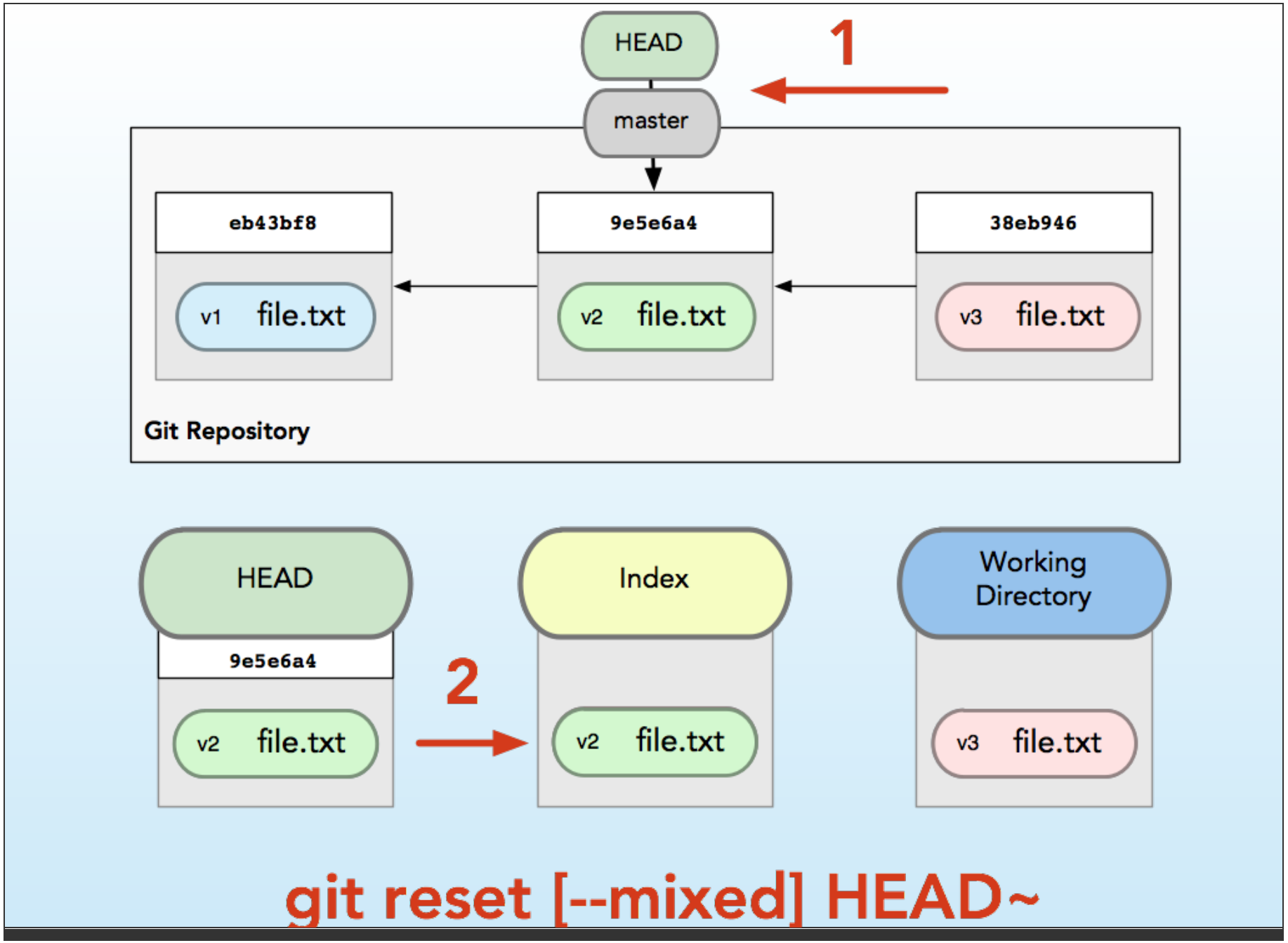
but keep the stage

`git reset --soft HEAD~`

moves HEAD back but keeps index


HEAD
master
eb43bf8
9e5e6a4
38eb946
v1 file.txt
v2 file.txt
v3 file.txt
Git Repository
HEAD
38eb946
v3 file.txt
Index
v3 file.txt
Working Directory
v3 file.txt

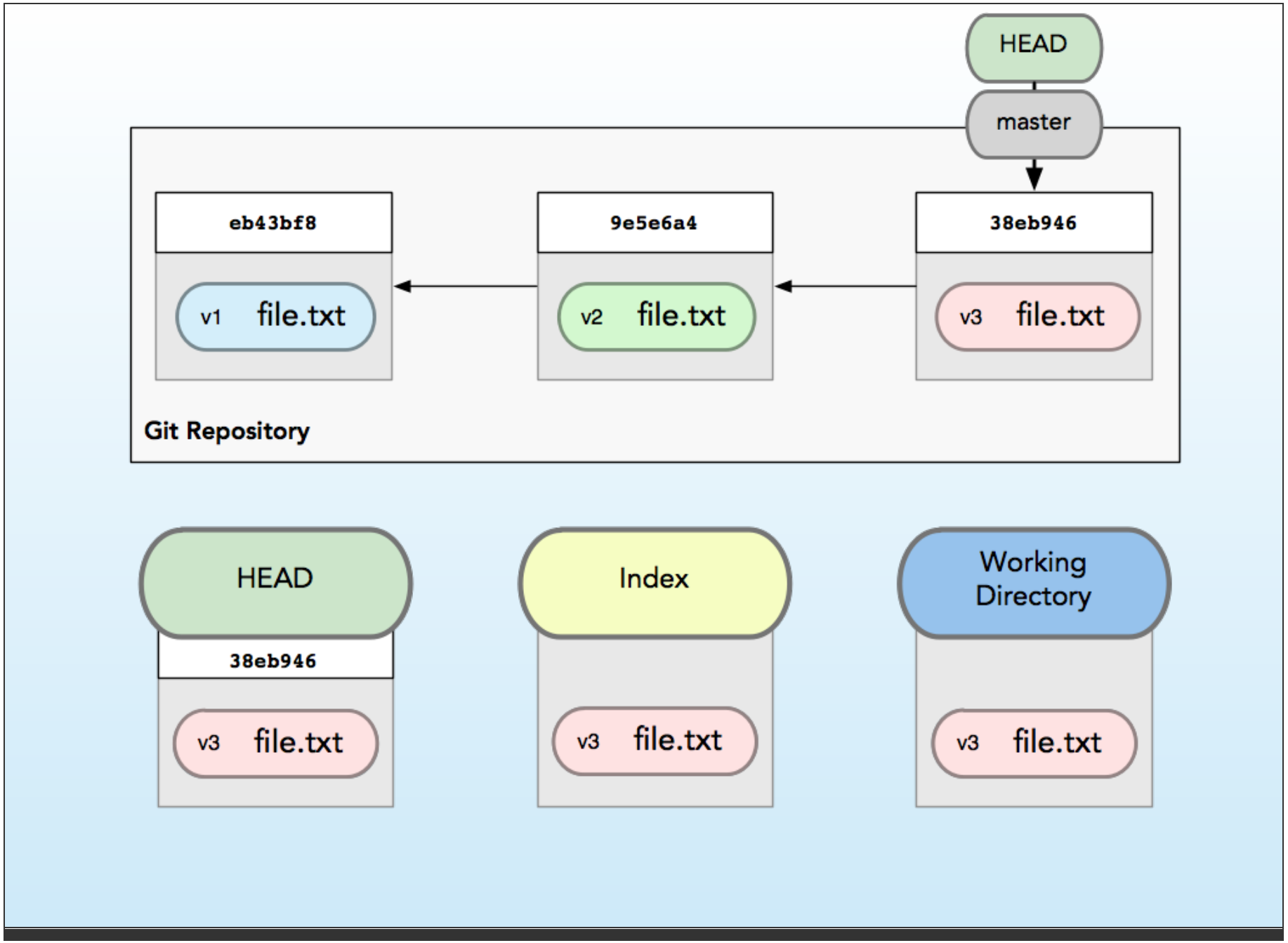

HEAD
master
eb43bf8
v1 file.txt
9e5e6a4
v2 file.txt
38eb946
v3 file.txt
Git Repository
HEAD
9e5e6a4
v2 file.txt
Index
v3 file.txt
Working Directory
v3 file.txt
git reset --soft HEAD~

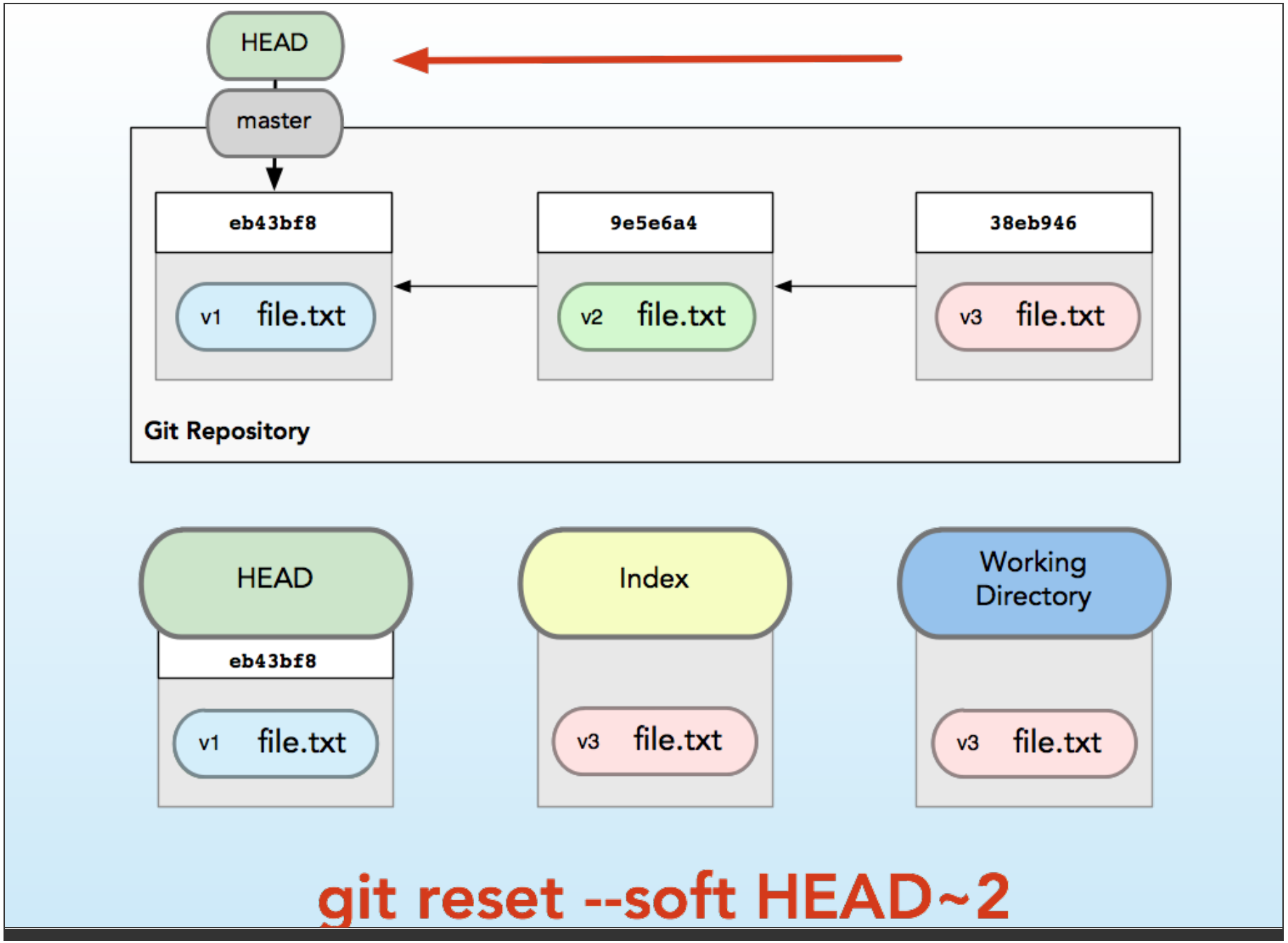

HEAD
1
master
eb43bf8
9e5e6a4
38eb946
v1 file.txt
v2 file.txt
v3 file.txt
Git Repository
HEAD
9e5e6a4
v2 file.txt
2
Index
v2 file.txt
Working Directory
v3 file.txt
git reset [--mixed] HEAD~

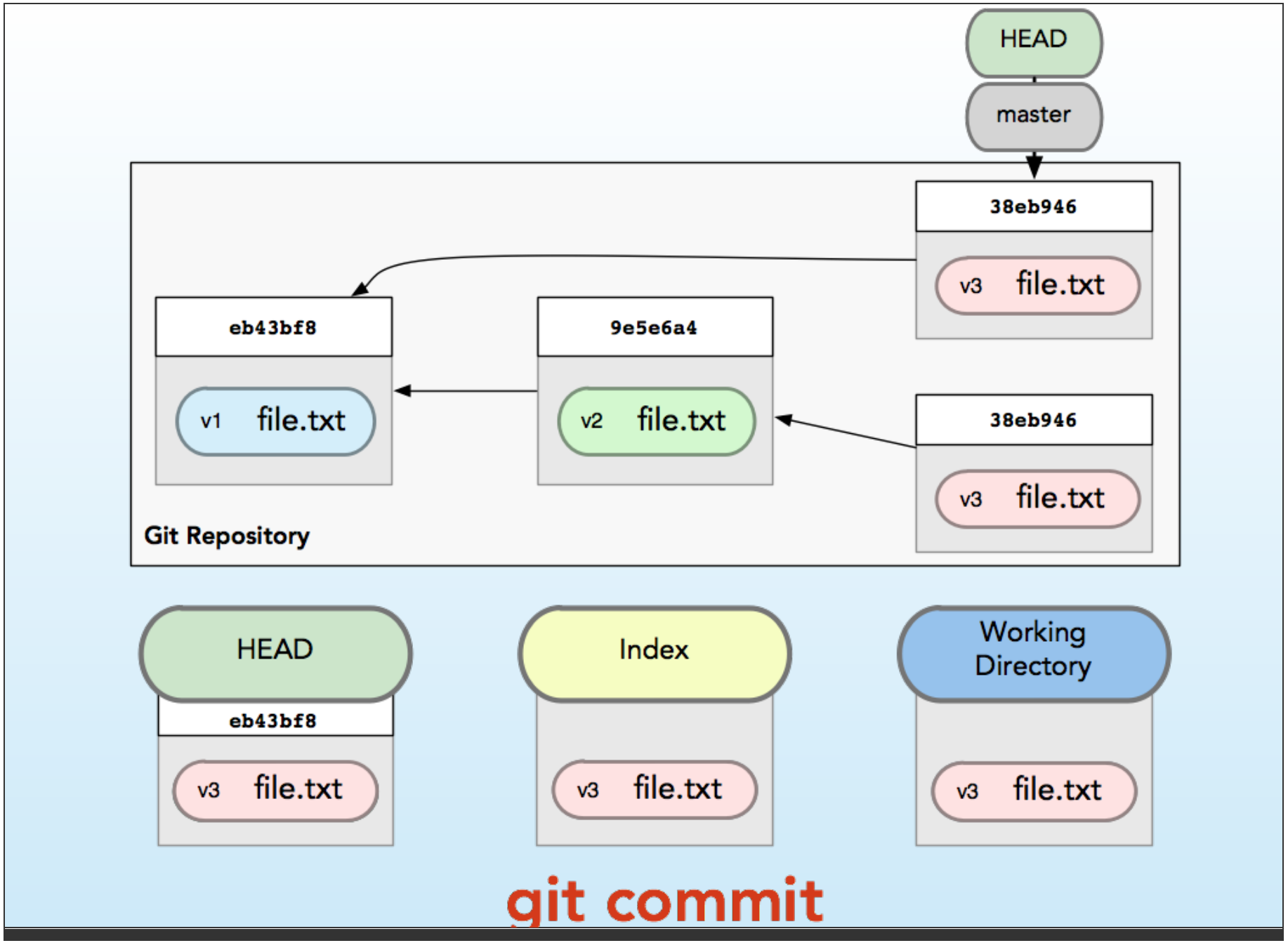
squash the last 2 commits into one

git reset --soft HEAD~2
git commit

moves HEAD back, keeps index


HEAD
master
eb43bf8
9e5e6a4
38eb946
v1 file.txt
v2 file.txt
v3 file.txt
Git Repository
HEAD
38eb946
v3 file.txt
Index
v3 file.txt
Working Directory
v3 file.txt


HEAD
master
eb43bf8
9e5e6a4
38eb946
v1 file.txt
v2 file.txt
v3 file.txt
Git Repository
HEAD
eb43bf8
v1 file.txt
Index
v3 file.txt
Working Directory
v3 file.txt
git reset --soft HEAD~2


HEAD
master
38eb946
v3 file.txt
eb43bf8
v1 file.txt
9e5e6a4
v2 file.txt
38eb946
v3 file.txt
Git Repository
HEAD
eb43bf8
v3 file.txt
Index
v3 file.txt
Working Directory
v3 file.txt
git commit

</reset>

Certificate of Achievement

This to certify that

you people

has successfully completed

Git Reset Training

Almost DC — Location

Apr 1, 11 — Date

Scott "Dragon" Chacon

Course Manager

git checkout

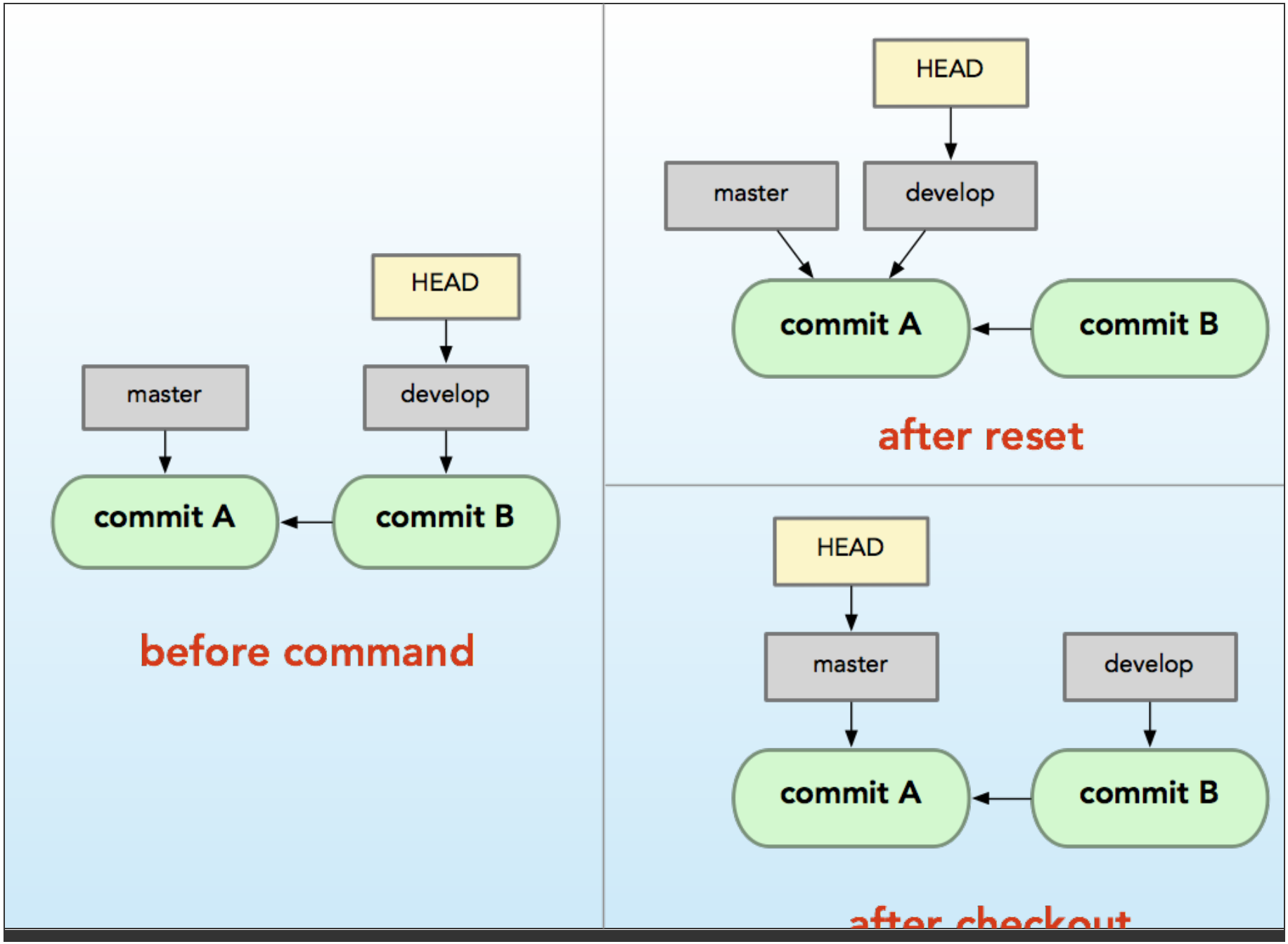
just a bit outside

tried the corner and missed

2 forms

```
git checkout [commit] [path]

git checkout [commit]
```


HEAD
master
develop
commit A
commit B
before command
HEAD
master
develop
commit A
commit B
after reset
HEAD
master
develop
commit A
commit B
after checkout

Reset v. Checkout

schacon.github.com/resetvcheckout.html

	HEAD	Index	Work Dir	WD Safe
Commit Level				
`reset --soft [commit]`	REF	NO	NO	YES
`reset [commit]`	REF	YES	NO	YES
`reset --hard [commit]`	REF	YES	YES	NO
`checkout [commit]`	HEAD	YES	YES	YES
File Level				
`reset (commit) [file]`	NO	YES	NO	YES
`checkout (commit) [file]`	NO	YES	YES	NO

ACT THREE

Fun With Your Trees

Patchy Work

```
git add --patch [file]

git reset --patch (commit) [file]

git checkout --patch (commit) [file]
```

git read-tree

`git write-tree`

```
$ ls
README              Rakefile            lib

$ git init
Initialized empty Git repository in /private/tmp/
nothing added to commit but untracked files prese

$ git log
fatal: bad default revision 'HEAD'

$ git add --all

$ git write-tree
4b8ad0172510761cb0e07d2c4220932bf41bbd07

$ git ls-tree 4b8ad0172510761cb0e07d2c4220932bf41
100644 blob 45dc653de6860...    README
100644 blob ea3fe2ac46e92...    Rakefile
040000 tree 99f1a6d12cb4b...    lib
```

git commit-tree

Environment Variables

moving around your trees

GIT_DIR

```
$ mv .git /opt/repo.git

$ git --git-dir=/opt/repo.git log

$ export GIT_DIR=/opt/repo.git
$ git log
```

GIT_INDEX_FILE

```
$ git status -s
 M README
 M kidgloves.rb

$ git add kidgloves.rb

$ git status -s
 M README
S  kidgloves.rb
```

```
$ export GIT_INDEX_FILE=/tmp/index
$ git read-tree HEAD
$ git add README

$ git status -s
S  README
 M kidgloves.rb
```

```
$ unset GIT_INDEX_FILE

$ git status -s
 M README
S  kidgloves.rb

$ export GIT_INDEX_FILE=/tmp/index

$ git status -s
S  README
 M kidgloves.rb
```

GIT_WORK_TREE

```
$ git status -s
 M README
S  kidgloves.rb

$ export GIT_DIR=$(pwd)/.git
$ export GIT_WORK_TREE=$(pwd)

$ cd /tmp
$ git status -s
 M README
S  kidgloves.rb
```

WTFWIEWTUT

whythefuckwouldieverwanttousethis

Publishing Docs to Another Branch

```
task :publish_docs do
  `rocco libgit.rb` # creates libgit.html
  ENV['GIT_INDEX_FILE'] = '/tmp/i'
  `git add -f libgit.html`
  tsha = `git write-tree`
  csha = `echo 'boom' | git commit-tree #{tsha}`
  `git update-ref refs/heads/gh-pages #{csha}`
  `git push -f origin gh-pages`
end
```

Making Tarballs of Project Subsets

```
`rm /tmp/in`
ENV['GIT_INDEX_FILE'] = '/tmp/in'
`git read-tree --prefix lib   master:lib`
`git read-tree --prefix ext-m extras:ext`
tsha = `git write-tree`
`git archive --format=zip -o out.zip #{tsha}`
```

```
$ unzip out.zip | head
Archive:  out.zip
   creating: ext-m/
   creating: ext-m/java/
   creating: ext-m/java/nokogiri/
  inflating: ext-m/java/nokogiri/EncodingHandler.
  inflating: ext-m/java/nokogiri/HtmlDocument.jav
  inflating: ext-m/java/nokogiri/HtmlElementDescr
  inflating: ext-m/java/nokogiri/HtmlEntityLookup
  inflating: ext-m/java/nokogiri/HtmlSaxParserCon
  inflating: ext-m/java/nokogiri/NokogiriService.
```

Auto-Backup Script

```
back_branch = 'refs/heads/backup'

`rm /tmp/backup_index`
ENV['GIT_INDEX_FILE'] = '/tmp/backup_index'

last_commit = `git rev-parse #{back_branch}`.strip
last_tree   = `git rev-parse #{back_branch}^{tree}`.strip

`git add --all`
next_tree = `git write-tree`.strip

if last_tree != next_tree
  extra = last_commit.size == 40 ? "-p #{last_commit}" : ''
  csha = `echo 'back' | git commit-tree #{next_tree} #{extra}`
  `git update-ref #{back_branch} #{csha}`
end
```

```
$ git log backup
commit e3219f9d18ac485f563995a39c139736abd75420
Author: Scott Chacon <schacon@gmail.com>
Date:   Thu Mar 31 13:51:05 2011 -0700

    back

commit cff888a65f56572358bdd233fe6af46c48f1d36d
Author: Scott Chacon <schacon@gmail.com>
Date:   Thu Mar 31 13:50:54 2011 -0700

    back

commit 15f3b561b351187bf712037f267036b90438c987
Author: Scott Chacon <schacon@gmail.com>
Date:   Thu Mar 31 13:45:56 2011 -0700

    back
```

Freeze Submodules Before Push

```ruby
current_commit = `git rev-parse HEAD`
current_tree = `git rev-parse HEAD^{tree}`

# get a list of submodules
status = `git submodule status`.chomp
subdata = status.split("\n")
subdata.each do |subline|
  sharaw, path = subline.split(" ")
  sha = sharaw[1, sharaw.size - 1]
  remote = path.gsub('/', '-')
  `git remote add #{remote} #{path} 2>/dev/null` # fetch each subm
  `git fetch #{remote}`
  `git read-tree --prefix=#{path} #{sha}` # for each submodule/sha
end

# find heroku parent
prev_commit = `git rev-parse heroku 2>/dev/null`.chomp
pcommit = (prev_commit != "heroku") ? "-p #{prev_commit}" : ''

# write-tree/commit-tree with message of what commit sha it's base
tree_sha = `git write-tree`.chomp
commit_sha = `echo "deploy at #{current_commit}" | git commit-tree

# update-ref
`git update-ref refs/heads/heroku #{commit_sha}`

# reset
`git reset HEAD`
```

CODA

Let's Review

Tree Roles

HEAD last commit, next parent

Index proposed next commit

Work Dir sandbox

Hi, you've reached Jimmy

If you can dream it, you can do it!

thanks!

questions?

questions?

threetrees.heroku.com

schacon.github.com/resetvcheckout.html

github.com/schacon/tale_of_three_trees

gist.github.com/582888