



# **The Official GitHub Training Manual**

---

# Table of Contents

Introduction	1.1
	1.2
	1.3
GitHub	1.4
GitBranching	2.1
Git	2.2
	2.3
	2.4
GitHub	2.5
Pull Request	2.6
	2.7
	2.8
	3.1
<b>GitHub</b>	
	4.1
CODEOWNERS	4.2

---

Git Bisect	4.3
	4.4
Git	4.5
	4.6
	4.7
	4.8
	5.1
	5.2
Git Reset	5.3
	5.4
	5.5
	6.1
	6.2
	6.3
Jekyll	6.4
	6.5
	6.6
	6.7

---

# Welcome to GitHub

Today you will embark on an exciting new adventure: learning how to use Git and GitHub.

As we move through today's materials, please keep in mind: this class is for you! Be sure to follow along, try the activities, and ask lots of questions!

## License

The prose, course text, slide layouts, class outlines, diagrams, HTML, CSS, and Markdown code in the set of educational materials located in this repository are licensed as [CC BY 4.0](https://creativecommons.org/licenses/by/4.0/). The Octocat, GitHub logo and other already-copyrighted and already-reserved trademarks and images are not covered by this license.

For more information, visit: <http://creativecommons.org/licenses/by/4.0/>

---

## 1 GitHub.com

GitHub.com

- 
- 

github.com

1. GitHub.com Sign up
- 2.
- 3.
- 4.

## 2 Git

Git Git

Git Mac Windows PowerShell

```
$ git --version
```

```
$ git --version  
git version 2.11.0
```

2.0

## Git

Git [www.git-scm.com](http://www.git-scm.com) Git

## Git

- WindowsGit [Git Bash](#) [Posh-git](#) Powershell
- MacUnix

## 3

- [Atom](#)
- GitPad
- ViVim
- Sublime
- NotepadNotepad++

## Atom

```
$ atom .
```

MacAtomWindows

---

## Git

- [github.com/explore](https://github.com/explore) Explore GitHub Universe Star
- [services.github.com/on-demand](https://services.github.com/on-demand) GitHub contributionsIssues

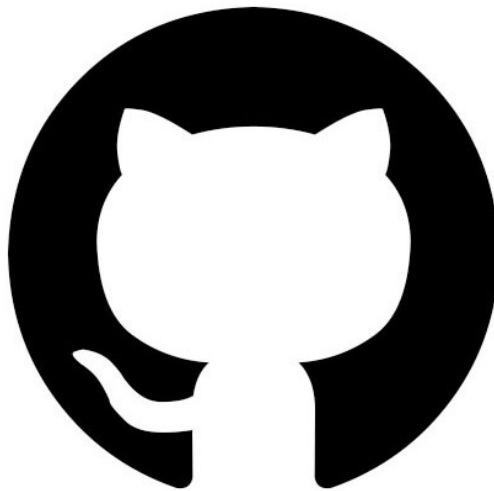
<https://github.com/github/training-kit>

---

GitGitHub GitHub

## GitHub

GitHubGit



GitHubGitGitHub

- Issues
- Pull requests
- Projects
- OrganizationTeam



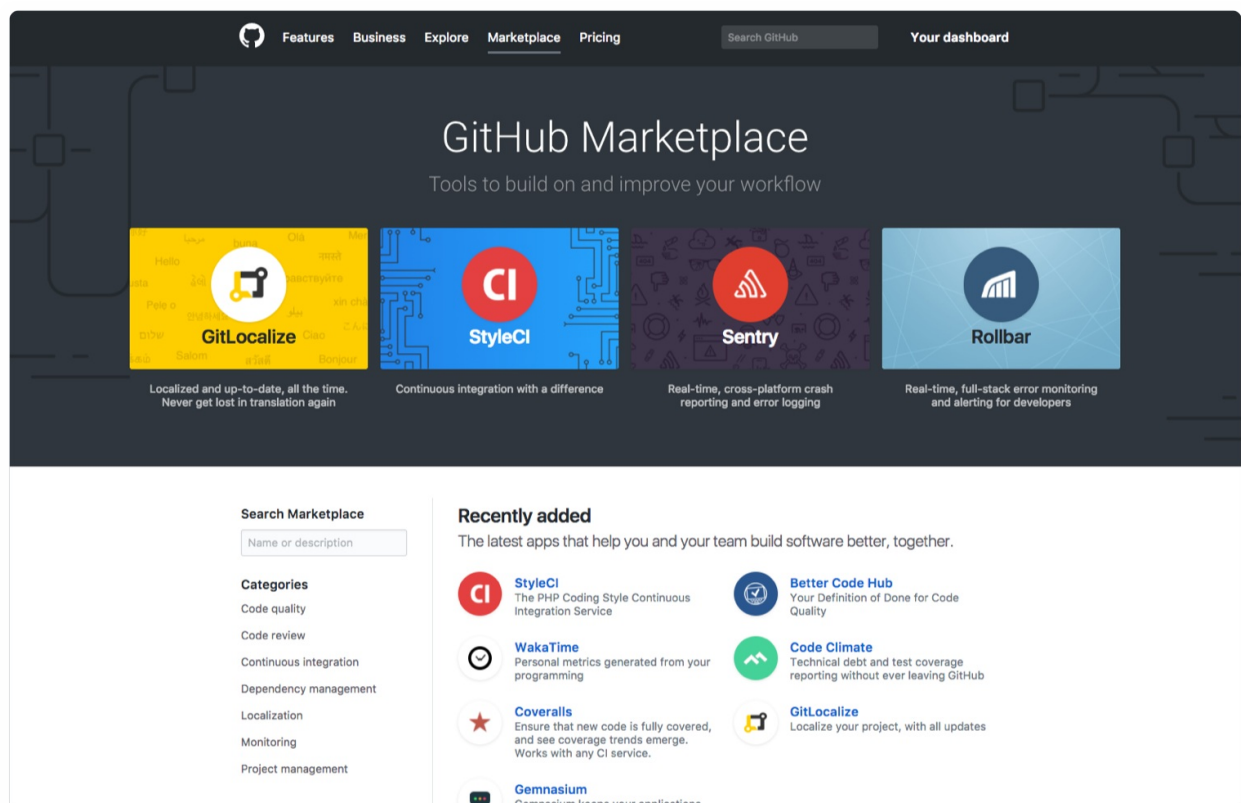
# GitHub



## GitHub Ecosystem

GitHubecosystem

<https://github.com/integrations>



Git

Git

- 
- DVCS
  - 
  - 
  -

Git

GitGitVCS GitVCS

GitGit

Git Git

Git

Git

Git Git

**Git**

Git Git

**GitHub**

GitHub GitHub



GitHubUser AccountOrganization Account

GitHub

## Code

code view Git

## Issues

---

Issues Issues

## **Pull requests**

Pull Requests Pull Request

## **Projects**

Projects

## **Wiki**

GitHubWiki WikiGitHub

## **Pulse**

Pulse

## **Graphs**

Graphs

## **README.md**

README.md GitHub README

## **CONTRIBUTING.md**

CONTRIBUTING.md CONTRIBUTING.mdIssuePull Request

## **ISSUE\_TEMPLATE.md**

ISSUE\_TEMPLATE.md (Pull Request)Issue Issue

## **GitHubIssues**

---

GithubIssue Issue

- 
- IssuePull Request
- 
- @

## GitHubIssue

Issue

1. *Issues*
2. *New Issue*
3. Subject line `YOUR-USERNAME Workflow`
4. Issue

YOUR-USERNAME

- [ ]
- [ ]
- [ ]
- [ ] Pull Request
- [ ]
- [ ]
- [ ]
- [ ] Pull Request

## Markdown

GitHub **Markdown**IssuePull Request `.md`

## Markdown

**# Header**

# # = Header 1## = Header 2

**\* List item**

---

\*

-

**\*\*Bold item\*\***

2

\*\*

- [ ] Checklist

- [ ] IssuePull Request

@mention

Issue@IssuessubscribeWatch

#975

# IssuePull Request

:smiley:

GitHub ID

:

## GitHub Pages

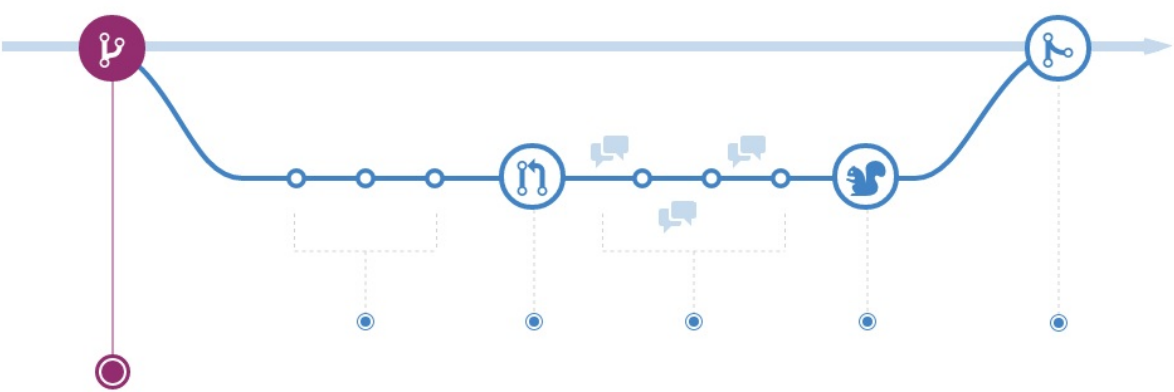
GitHub PagesGitHub GitHub Pages GitHub Pages

- User/OrganizationProject Project
- ProjectGitHub master gh-pages master /docs
- githubschool.github.io/repo-name

# GitHub

GitHub

## GitHub



GitHub

GitHub

master

master

Pull Request Pull Request

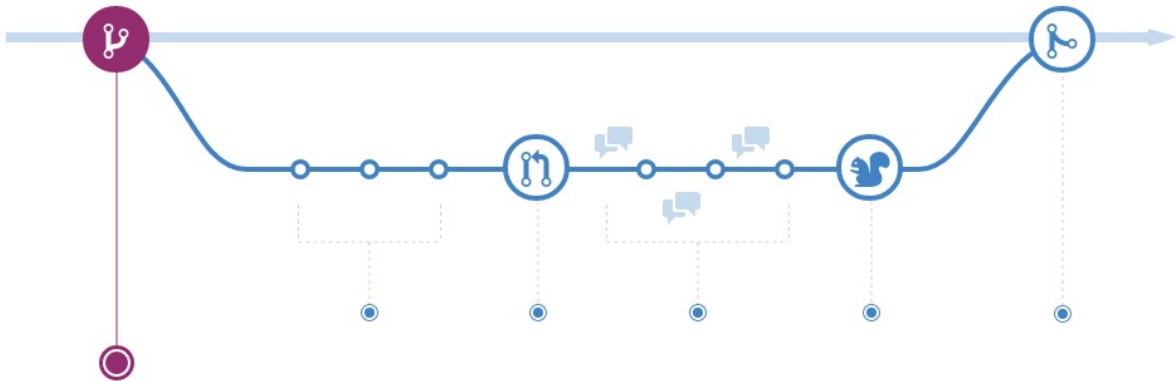
Pull Requestmaster GitHub

- [guides.github.com/introduction/flow/](https://guides.github.com/introduction/flow/) GitHub

# GitBranching

GitHub

## Branching



## GitHub

Issue

GitHub

1. *Code*
2. *branch dropdown*
3. *github-username-caption*
4.

GitHub

GitHub



- GitHub <https://youtu.be/H5GJfcp3p4Q>

# Git

Git

## Git

[Git Installation](#)

```
$ git --version  
$ git version 2.11.0
```

GitGit

Git [www.git-scm.com](http://www.git-scm.com)

## Git



--system



--global



--local

Git3

**--system**

**--global**

**--local**

```
git config --local
```

```
git config --list
```

```
$ git config --list
```

```
$ git config --global --list
```

Git

```
$ git config --global user.name "First Last"
$ git config --global user.email "you@email.com"
```

**Git**

```
user.name user.email --global --global
```

```
git config user.email "you@email.com"
```

Git GitHub

[Settings > Emails](#) **Keep my email address private**

```
ID +username@users.noreply.github.com
```

```
git config --global user.email 18249274+githubteacher@users.noreply.github.com
```

## Autocrlf

```
$ //for Windows users
$ git config --global core.autocrlf true
$ //for Mac or Linux users
$ git config --global core.autocrlf input
```

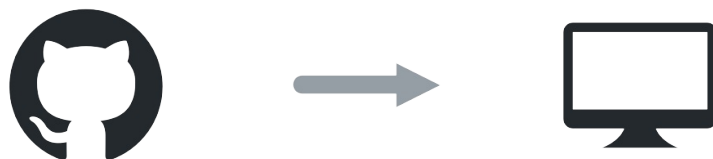
Git

```
autocrlf auto carriage return line feed
```

---

# Git

Git



git

1. *GitHub Code*
2. *Clone or download*
3. *clone URL*
- 4.
5. GitHub `git clone <CLONE-URL>`
6. `cd <REPOSITORY-NAME>`

Git `git status`

```
$ git status
On branch master
Your branch is up-to-date with 'origin/master'.
nothing to commit, working tree clean
```

---

```
git status gh-pagesorigin/gh-pages
```

```
$ git branch
```

```
git branch
```

```
$ git branch --all
$ git branch -a
```

```
--all
```

```
-a
```

```
--all -a Git
```

```
$ git checkout <BRANCH-NAME>
```

git checkout Gitorigin

```
remotes/origin
```

```
remotes/origin HEAD
```

1. `named 2010-02-##-USERNAME.md`
- 2.
- 3.
- 4.

```
...
CAPTION-HERE
{: .fragment}
...
```

---

1. *Save*

Git IDE VIM

IDE

atom . Atom

code . Visual Studio Code

## 2



working



staging



history

untrackedmodifiedstagedcommitted

Untracked

Git3 WorkingStaging History



working



staging



history

new



modified



working



staging



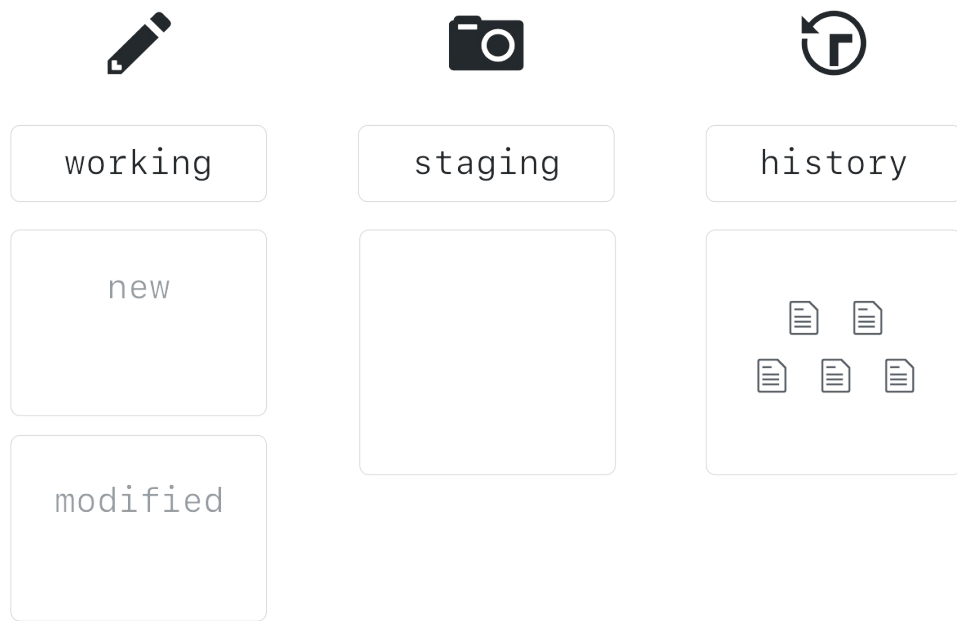
history

new



modified





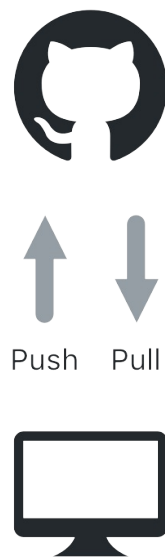
git addgit commit

1. `git status`
2. `git add my-file.md`
3. `git status`
4. `git commit`
5. Git
- 6.
7. `git status`

- 50
- 
-

---

## GitHub



GitHub.com GitHub Enterprise

GitHub

```
$ git push
```

GitHub Git

- Windows: `git config --global credential.helper wincred`
- Mac: `git config --global credential.helper osxkeychain`

## Pull Request

Pull Request Pull Request Issue Pull Request

---

Pull Request Pull Request

1. *Pull Request*
2. *New Pull Request*
3. *base* `gh-pages`
4. *compare*
5. Subject line
6. MarkdownPull Request
7. `closes fixes resolves` Pull RequestIssueIssue `This resolves`  
`#3`
8. Pull Request *Preview*
9. Pull Request
10. *Create pull request*

*Compare & pull request Pull Request*

## Pull Request

Pull RequestPull Request

## Conversation view

IssuePull Request ConversationPull Request

## Commits view

Commits view ID

## Files changed view

Files changed view `diff` `diffdiff`

## Pull Request

GitHub3

## General Conversation

---

ConversationPull Request

Files changed view + Conversation view

*Start a Review*

Pull Request

- 1. *Pull Request*
- 2. *AuthorPull Request*
- 3. *Files Changed*
- 4. 1+ +
- 5. *Start review*
- 6.
- 7. *Review*
- 8. *Approve Request changes*
- 9.
- 10. *Submit review*
- 11. *Conversation view*

# GitHub

Pull Request GitHub

## GitHub

Pull Request *Files Changed* view

1. diffGitHub
- 2.

## GitHub

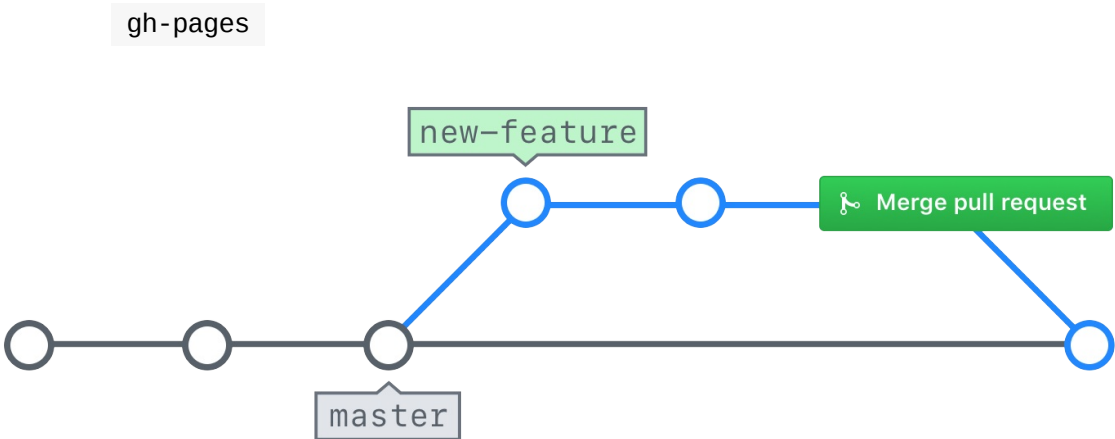
1. *Commit changes*
- 2.
3. *Commit directly to your branch*
4. *Commit changes*

## Pull Request

Pull Request

# Pull Request

Pull Request



Pull Request

- Pull Request
- 
- Pull Request

## Pull Request

Pull Request

1. Pull RequestPull RequestAuthorAssignee
2. Conversation
3. Pull Request Merge pull request
4. Confirm merge
5. Delete branch
6. IssuesIssue

GitHubPull Request

- **Create a merge commit:**
- **Squash and merge:**
- **Rebase and merge:** GitHubfast forward

Pull RequestGitHub

GitHub

1. `git checkout gh-pages`
2. GitHub `git pull`

`git pull GitHub` `git fetch` `git merge`

`git branch --all`

1. `git branch --all`
2. `git branch --merged`
3. `git branch -d <branch-name>`
4. `git branch --all`
5. `git pull --prune`

`--merged` `git branch` `gh-pages`

Pull `git config --global fetch.prune true`

---

## Git Log

log

```
$ git log
$ git log --oneline
$ git log --oneline --graph
$ git log --oneline --graph --decorate
$ git log --oneline --graph --decorate --all
$ git log --stat
$ git log --patch
```

↑↓

q



---

log Git

log

```
$ git log --oneline --graph --decorate --all
```

```
$ git config --global alias.lol "log --oneline --graph --decorate --all"
```

```
$ git lol
```

```
$ git config --global alias.co "checkout -b"  
$ git config --global alias.s "status -s"  
$ git config alias.dlb '!git checkout <DEFAULT-BRANCH> && git pull --prune &&  
git branch --merged | grep -v "\*" | xargs -n 1 git branch -d'
```

- [git-scm.com/book/en/v2/Git-Basics-Git-Aliases](https://git-scm.com/book/en/v2/Git-Basics-Git-Aliases)

---

1

- 
- 2
- 

12

1. `USERNAME-conflict`
- 2.
3. `base: master` `compare: USERNAME-conflict` Pull Request
4. 1 Pull Request
5. 2 Pull Request
  - i. `master`
  - ii. `Unmerged Paths` `git status`
  - iii. ( `<<<<<<` , `=====` , `>>>>>>` )
  - iv.
  - v.
  - vi. Pull Request
6. Pull Request

Git

---

## : GitHub Games

github-games

github-games githubschool organization <https://github.com/githubschool/github-games-username>

## : README.md

READMEGitHub

<https://githubschool.github.io/github-games-username>

URL

README.md

1. `git clone https://github.com/githubschool/github-games-USERNAME.git`
2. `readme-update` `git checkout -b readme-update`
3. README.mdURL
- 4.
5. GitHub `git push -u origin readme-update`
6. Pull Request (base: `master` , compare: `readme-update` )
7. Pull Request
8. GitHub
9. `git pull --prune`

`git checkout -b readme-update` `git branch readme-update` `git checkout`  
`readme-update` `-b Git`

`git push -u origin readme-update`

`-u` `--set-upstream` `Git`

`git push`

## CODEOWNERS

### Protected Branches   Code Owners

1. Settings
2.     **Branches**
3.     **Choose a branch...**     master
4. **Protect this branch**

## CODEOWNERS

### CODEOWNERS   CODEOWNERS

- Javascript     .js
- docs/
- package.json

### CODEOWNERS

1.     CODEOWNERS     .github/
2.     \* @YOUR\_USERNAME
  -
3.     \*.js @githubteacher
  -
4. CODEOWNERS     **Require pull request reviews before merging**     **Require review from Code Owners**     **Save changes**

### CODEOWNERS

git bisect

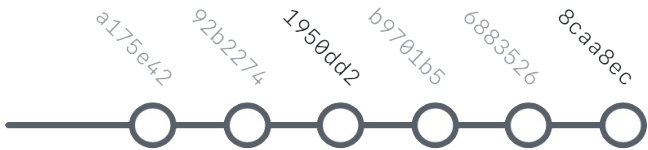
git bisect

git bisect bisect

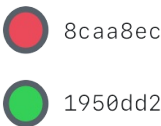
- 
- 
- 

Bisect

git bisect Git Git



git bisect



BisectHEAD (detached HEAD) Git git bisect reset bisect {:  
.warning}

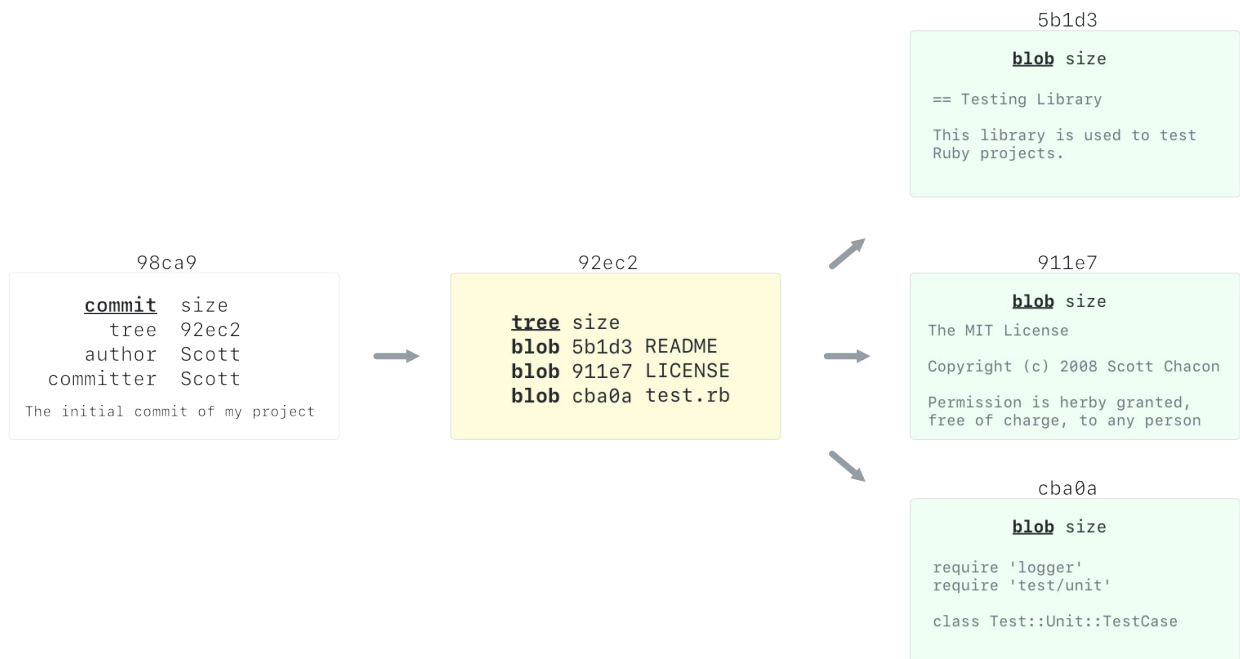
1. `git bisect start`
2. `git bisect bad <SHA>`
3. `git bisect good <SHA>`
4. Bisect
5. `ls index.html`
6. `index.html git bisect bad`
7. `index.html git bisect good`
8. Gitbisect Git
9. Git `SHA is the first bad commit.`
10. Bisect `git bisect reset`

### Bisect

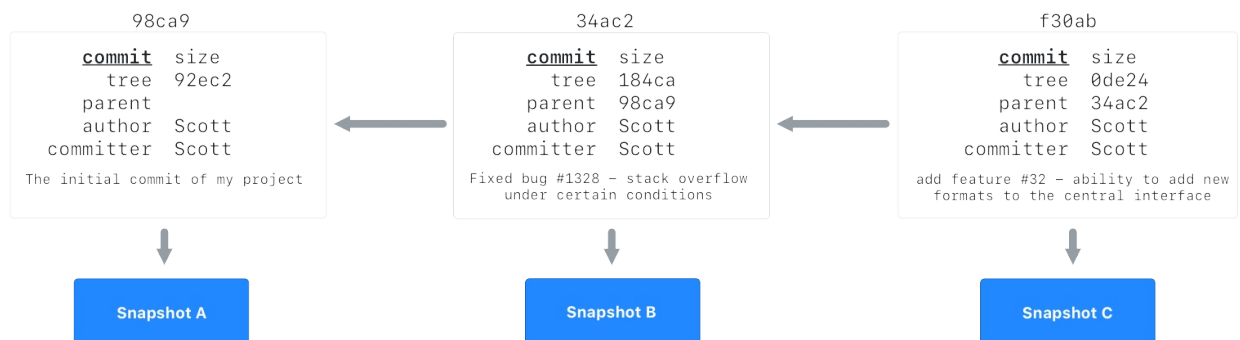
1. `git bisect start <bad-SHA> <good-SHA>`
2. `git bisect run ls index.html`

## Git

- 
- 
- 
- 



## ID



Git

ID

revert	
commit --amend	
reset	
cherry-pick	
rebase	

- index.html
- git show SHA
- git revert <SHA>
  - 
  - GitHub



# Git

Git

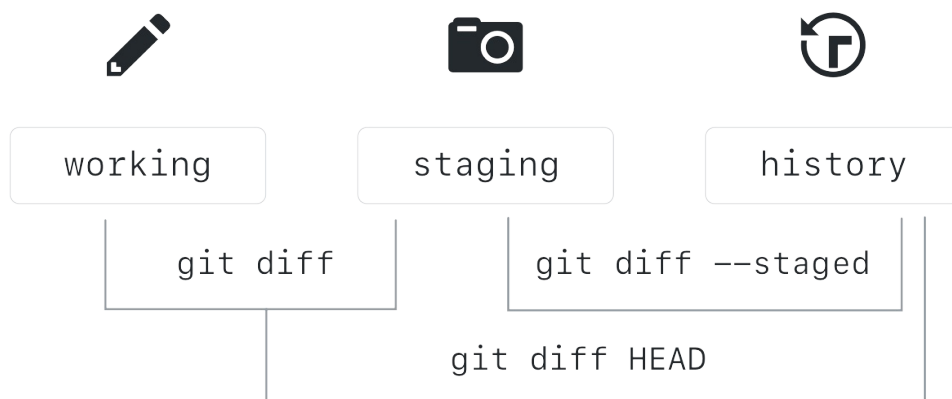
## Git

1. `slow-down`
2. `index.html` 9 URL (*images/texture.jpg*)
3. 78
- 4.
5. Git `git status`
6. `mkdir images`
7. Git `git mv texture.jpg images/texture.jpg`

1. Git `git status`
  2. `--patch` `git add -p`
  3. `y`
  4. `n`
- | `?`

---

```
git diff 2ref HEAD
```



```
$ git diff
$ git diff --staged
$ git diff HEAD
$ git diff --color-words
```

```
git diff
```

```
$ git diff <REF-1> <REF-2>
$ git diff gh-pages slow-down
$ git diff origin/gh-pages gh-pages
$ git diff 2710 b745
```

```
diff diff
```

---

Git                      **tag** GitHub    **release**

GitGitHub                      -a            v1.0   SHA

- `git tag -a v1.0 <SHA>`

`git tag --list`

`git push --tags`

config   `git config push.followTags true`                      .

GitHub

## GitHub-Games

1. GitHub    **Code**
2. Releases
3. Draft a new release
- 4.
5. master commits
- 6.
7. Publish release Save draft

This is a pre-release

---

## Issues

- 
1. `cd ..`
  2. `git init practice-repo - CD` `cd practice-repo - README.md`  
`touch README.md - README.md`

## Bash:

```
for d in {1..6}; do touch "file${d}.md"; git add "file${d}.md"; git commit -m  
"adding file ${d}"; done
```

## PowerShell:

```
for ($d=1; $d -le 6; $d++) { touch file$d.md; git add file$d.md; git commit -  
m "adding file$d.md"; }
```

---

## GitGitHub

```
git commit --amend HEAD 2
```

- 
- 

1. `touch file7.txt`
2. `git add file7.txt`
3. `git commit --amend`
- 4.

|

# Git Reset

```
git reset
```

## Reset



Before

---

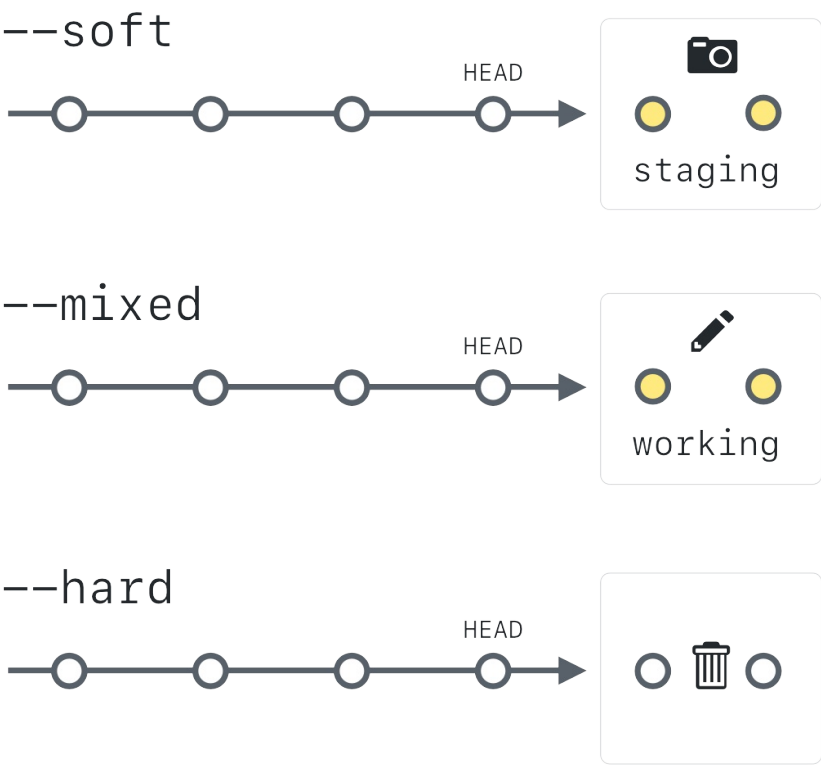
After



3 1HEAD 2 3

```
git reset 3
```

## Reset



git reset3    --soft ,   --mixed ,   --hard

**--soft**

git reset --soft <SHA>    <SHA>    1

**--mixed**

git reset --mixed <SHA>    <SHA>    Git    --mixed

**--hard**

git reset --hard <SHA>    Git    3    <other-commit>  
git reset --hard    --hard

**Reset Soft**

reset --soft



1. `git log --oneline --decorate`
2. `HEAD`
3. 2 `git reset --soft HEAD~2`
4. `HEAD 2` `git log --oneline --decorate`
5. 2 `git status`
6. `git commit -m "re-add file 5 and 6"`

`HEAD 2git ID`

## Reset Mixed

reset `reset --mixed`

1. `git log --oneline`
2. 1 `git reset HEAD~`
3. `git log --oneline --decorate`
4. `git status`
5. `git add file5.md file6.md`
6. `git commit -m "re-add file 5 and 6"`

`56HEADID reset`

## Reset Hard

hard reset

1. `git log --oneline`
2. `README.md` `git reset --hard <SHA>`
3. `git log --oneline`
4. `git status`
5. `README.md` `ls`

`git reset --hard`

```
$ git reflog
```

reflogHEAD reflog reflog

- **reflog** reflogreflog
- **reflog90**reflog30

---

# git cherry-pick

reflog

1

reflog

```
$ ls
README.md
$ git log --oneline
84nqdkq initializing repo with README
```

4

1. file4.mdID `git reflog`
2. `git cherry-pick <SHA>`

```
$ ls
file4.md
README.md
$ git log --oneline
eanu482 adding file 4
84nqdkq initializing repo with README
```

ID

GitHub ID {:.warning}

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

1. HEAD `git reflog`
2. `file6.md` `git reset --hard <SHA>`

---

3. `git log --oneline`

`git log --oneline ID`    `git reflog`

4                    `git reset --hard 2`

---

## Git

Git3

## Fast forward

fast forward fast forward

git 2

3

## Git

```
git rebase 1squash
```

```
rebase
```

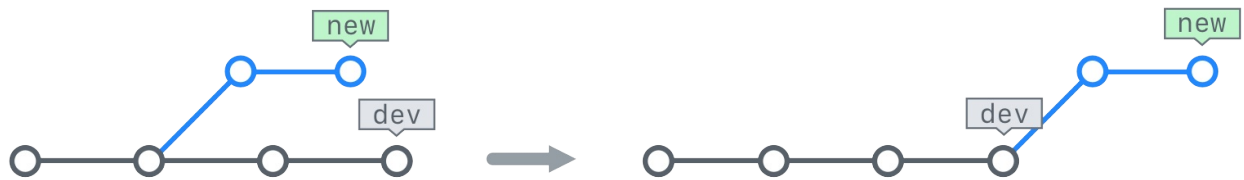
```
git rebase --interactive
```

```
git rebase -i
```

```
git rebase -i
```

- 
- 
- 1
- 

1



git rebase dev

1. SHA `git log --oneline`
2. SHA `git reset --hard SHA`
3. `git checkout -b rebase-me`
4. rebase-me `rebase-me`
5. master `git checkout master`
6. rebase-me `rebase-me`
7. `git log --oneline --graph --decorate --all`
- 8.

1. rebase-me `git checkout rebase-me`
2. `git rebase -i master`
- 3.
4. rebase-todo
- 5.
6. `git log --oneline --graph --decorate --all`
7. fast-forward

1. master `git checkout master`
2. master `git merge rebase-me`

---

# 1

## Pull Request

1. `git checkout -b NEWBRANCHNAME`
2. `_posts YYYY-MM-DD-username-description.md ---- (`  
`)`
3. "[Activity: Edit Your File](#)"
4. ```` --- layout: slide ---`

```
IMAGEURL
{: .center}

CAPTION-HERE
{: .fragment}
```
```

- 1.
2. `git push -u origin NEWBRANCHNAME .`
3. Pull RequestPull Request@
4. Pull RequestPull Request

## Pull Request

1. Pull Request
2. files changedview
3.
  - 
  - `git pull`
  - Pull Request `git checkout BRANCHNAME`
- 4.

- `git push`
5. Pull Request @ Pull Request

## README.md

README.md

1. `git checkout -b NEWBRANCHNAME`
2. README.md
3. `git push -u origin NEWBRANCHNAME` .
4. Pull RequestPull Request `base: master` `compare: NEWBRANCHNAME`
5. @ mention
6. Pull RequestPull Request

1. `git checkout -b NEWBRANCHNAME`
2. `_sass / solarized / solarized.scss`
3.
  - 1219
  - 3335
  - 5255
- 4.
5. `git push -u origin NEWBRANCHNAME`
6. `base: master` `compare: NEWBRANCHNAME` Pull Request
7. @ mention
8. Pull RequestPull Request



- 
- `github.com/githubschool/conflict-practice-username`
  - 🤔

1. `github.com/githubschool/USERNAME` USERNAME
2. **Pull Requests**
3. Pull Request
  - README
  - 
  - CSS
4. Pull Request Pull Request

- 
- 1.
  2. master
  - 3.
  4. labelsassignees
  - 5.
  6. IssuePull Request
  7. Pull Request
  8. Pull Request

# Jekyll

GitHub Pages

OSmacOS

Ruby Ruby [ruby-lang.org](https://ruby-lang.org)

- Windows <https://rubyinstaller.org> 2.3.3 Add Ruby executables to your PATH
- MacRuby [homebrew](https://brew.sh) `brew install ruby`

1. Bash `cd`
2. Ruby
  - `ruby -v` `gem -v`
  - Ruby `2.3.x :+1:` 🙌
3. `Gemfile.lock`
4. Bundler `gem install bundler`
5. gems `script/setup`
6. `script/server`
7. 🎉

---

# Git

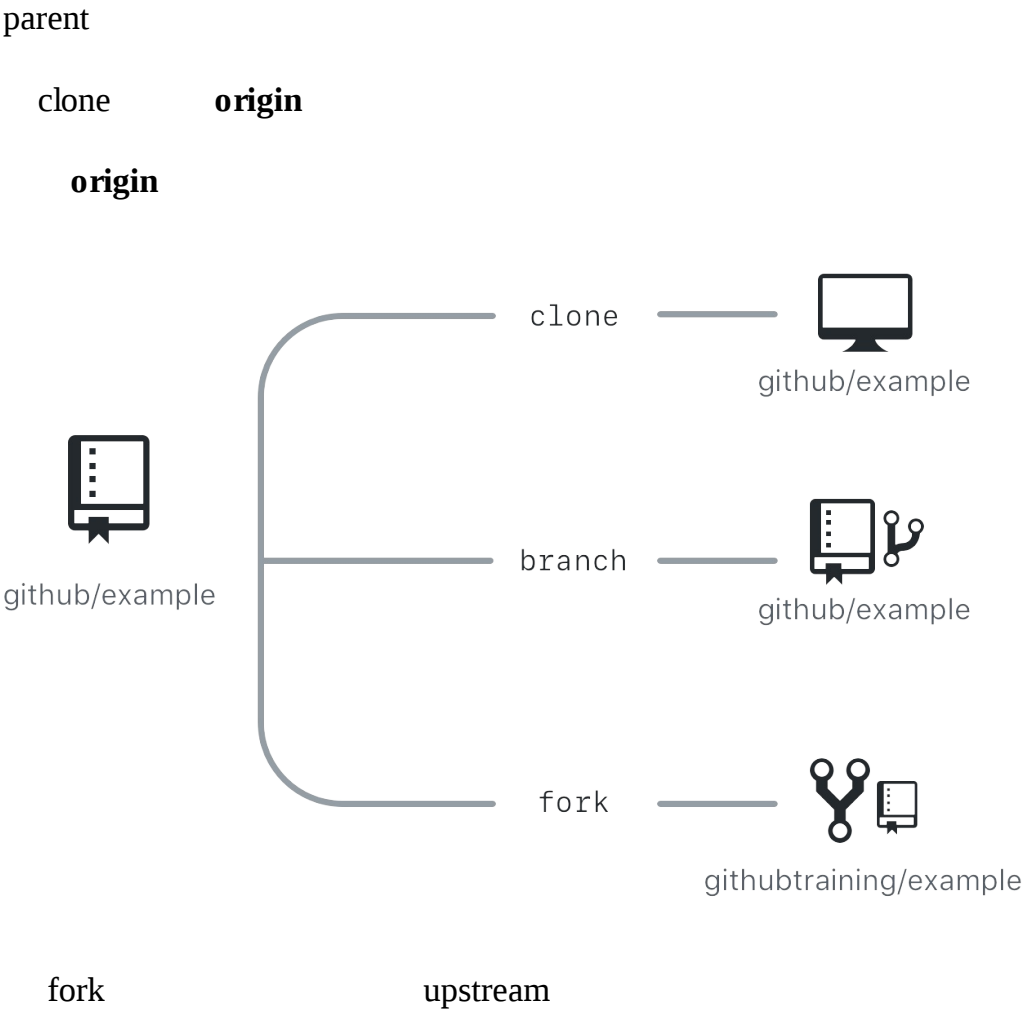
## Git

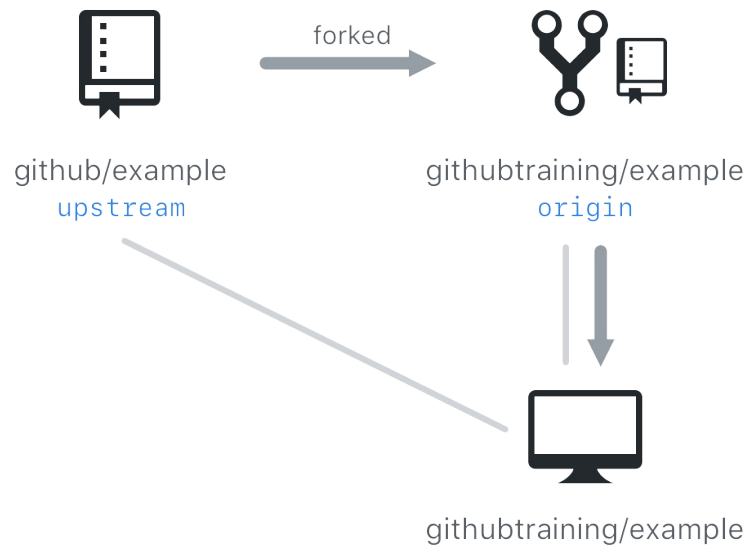
- **Mac** `.bash_profile`
- **Linux** `.bashrc`
- **Windows**

```
parse_git_branch() {  
    git branch 2> /dev/null | sed -e '/^[^*]/d' -e 's/* \(.*)/ (\1)/'  
}  
export PS1="\w\[\033[36m\]\$(parse_git_branch) \[\033[00m\] > "
```

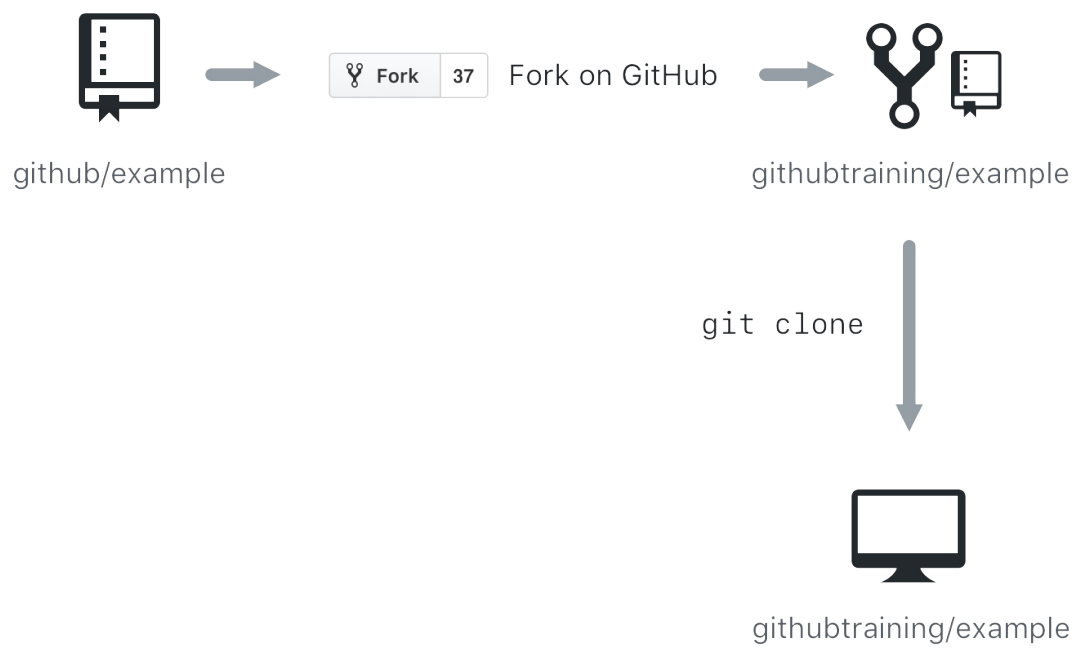
```
function parse_git_branch () {  
    git branch 2> /dev/null | sed -e '/^[^*]/d' -e 's/* \(.*)/ (\1)/'  
}  
  
RED="\[\033[0;31m\  
YELLOW="\[\033[0;33m\  
GREEN="\[\033[0;32m\  
NO_COLOR="\[\033[0m\  
  
PS1="$GREEN\u@h$NO_COLOR:\w$YELLOW\$(parse_git_branch)$NO_COLOR$ "
```

Git

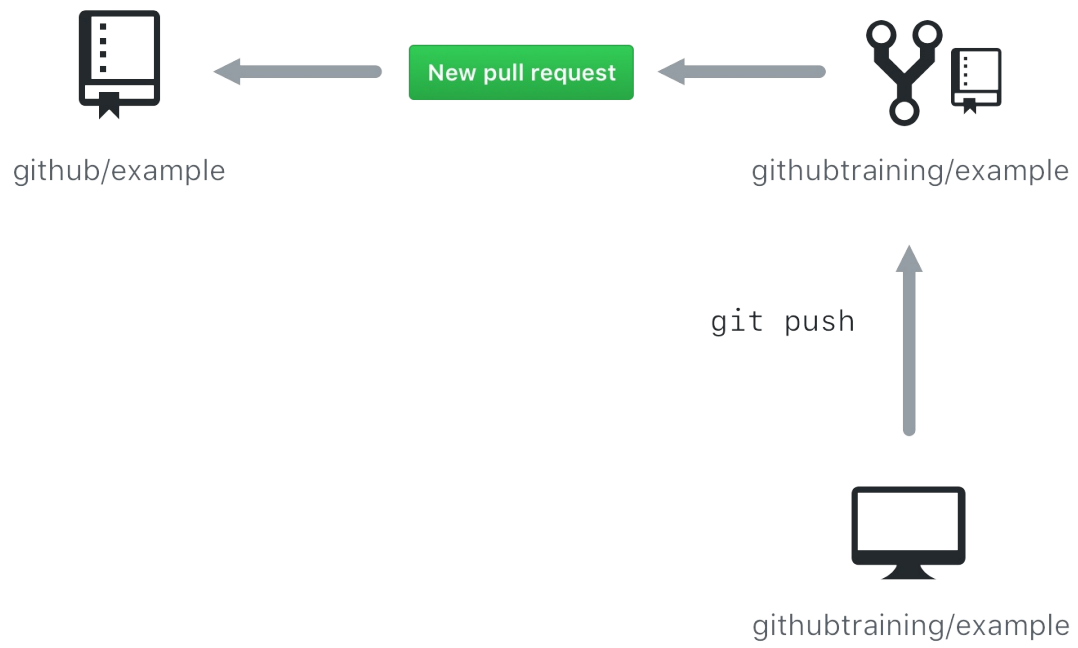




### origin



**origin** **origin** Pull Request



---

## 1config

1. `git config --unset --global user.name`
2. `git config --unset --global user.email`

## Step 2:

- **Bash:** `history -c && history -w && exit`
- **Windows:** `Alt+F7`

## 3

### Windows

- 1.
- 2.
- 3.
- 4.
5. WindowsGitGitHub

### Mac

- 1.
2. GitHub