Git

Kidane M. Tekle

March 2019

Disclarimer!

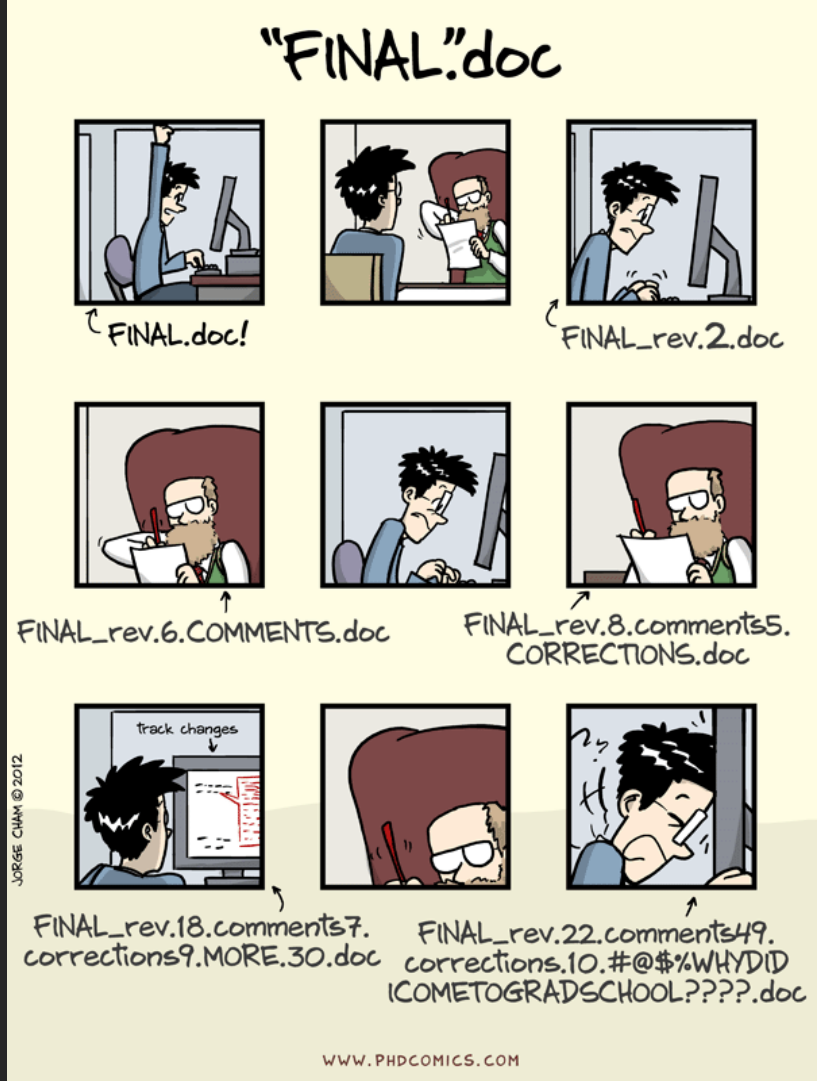
All pictures used are from random searches of the web and for educational purposes. They might be subject to specific licenses and should be checked before using further.

Introduction

Git hands on

Best practices

Summary



Introduction

https://boagworld.com/dev/to-version-control-or-not/

To Version Control or Not?

AUTHOR:
Paul Boag

DATE:
18 August 2008

CATEGORY:
Development, Digital Strategy

Version control can seem like a very daunting thing to incorporate into your work flow, but once it's there you can be left wondering how you ever lived without it. Paul Stanton gives his thoughts and experiences on the subject.

Introduction: Drivers of Version Control

- Individual drive
  - Proper backup
    (literally every [committed] change)
  - Make changes without fear
- Collaboration drive
  - Work on the same codebase
  - Ease for collaboration
  - Quality

Introduction: Three generations

Generation	Networking	Operations	Concurrency	Examples
First	None	One file at a time	Locks	RCS, SCCS
Second	Centralized	Multi-file	Merge before commit	CVS, SourceSafe, Subversion, Team Foundation Server
Third	Distributed	Changesets	Commit before merge	Bazaar, Git, Mercurial

https://ericsink.com/vcbe/html/history_of_version_control.html

Introduction

Git hands on

Best practices

Summary

Git hands on: installation

- Linux
  - $ apt-get install git
- Windows (https://gitforwindows.org)
  - – gui + git bash
- Mac

https://git-scm.com/downloads

Git hands on: Getting started

```
$ cd 'the-folder-to-version-control'
$ git init
$ git status
$ git add --all
$ git status
$ git commit –m 'first commit'
$ git status
$ git log
$ git HIT-TAB
```

Git hands on: Ignore things

$ echo ”*.sh” >> .gitignore

Get recipes from: https://github.com/github/gitignore

Git hands on: working with remote repo

Start from remote:

$ git clone 'project-repo-url'

$ git remote –v (see remote info)

$ 'do stuff'

$ git add –all

$ git commit –m 'some comment'

$ git push

Start from local:

$ cd 'the-folder-to-version-control'

$ git init

$ 'do stuff'

$ git add –all

$ git commit –m 'some comment'

$ git remote add origin git-repo-uri

$ git remote –v (see remote info)

$ git push -u origin master

Git hands on: branching &merging

$ git checkout –b <branch-name>

$ git add –all

$ git commit –m 'modifications in a new branch'

$ git push –u origin <branch-name>

Submit a merge request !

Git hands on: more . . .

- Forking
  - Is fature of github & gitlab(not part of basic git)
  - fork process: Fork => make changes => submit a pull request
  - It creates a local copy of the base repo and gives more freedom to play with it
- Advanced stuff
  - git blame some-file
  - git stash . . .
  - git diff branch-1 branch-2
  - . . .

https://medium.freecodecamp.org/useful-tricks-you-might-not-know-about-git-stash-e8a9490f0a1a

Introduction

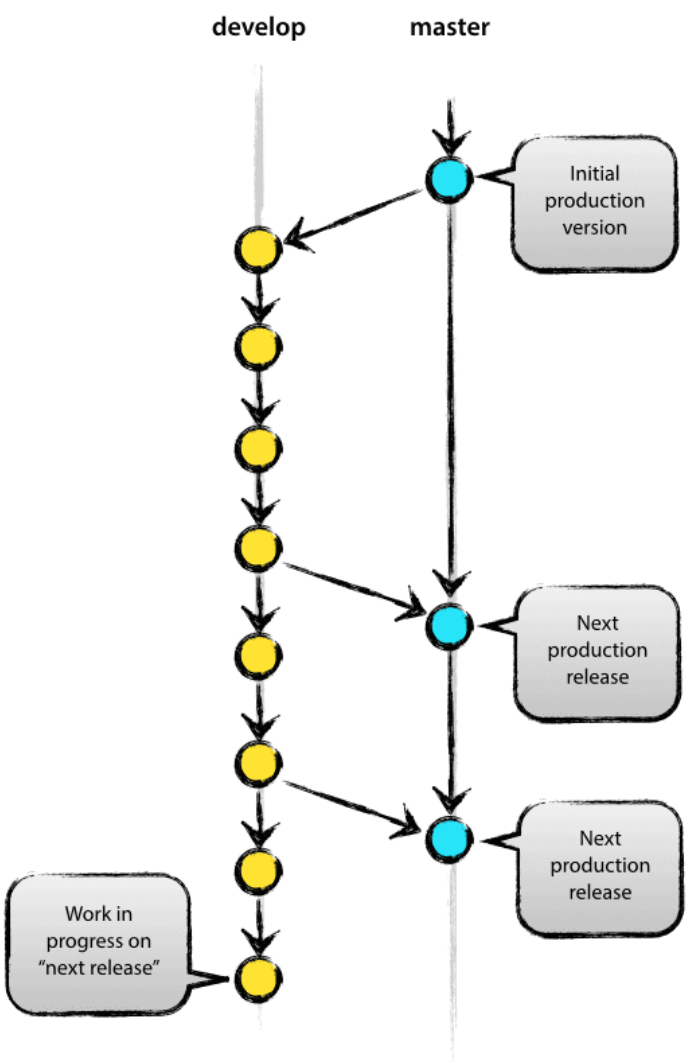
Git hands on

Best practices

Summary

Best practices

- Branch, don't fork (if possible)
- Commit your changes as often as you can
- Limit the number of "maintainer" role members
- Follow recommended branching strategy
  - master, feature-branch
  - master, develop, feature-branch
- Feature-branch should be
  - single purpose
  - short lived
- Tag releases properly
- Use CI / CD (Continuous Integration, Continuous Delivery)
  - trigger on specified branches
  - automate build, test, deploy actions



Introduction

Git hands on

Best practices

Summary