Machine Learning 2018 – Git 101

Hong, ONG Xuan

November 24, 2018

Contents

What is Git?

Before Git

- Difficulties in manage 1 folders with multiple team members.
- Accidentally delete files.
- Something changes without notices such as sysconf.
- Backup by multiple *.zip files.

What is Git?

With Git

- Centralize and decentralize repository.
- Revert and get back files' content easily.
- Notice all changes in project.
- No need to backup, your project is backup every push command.

Contents

How to work alone?

- Offline
- Online/Remote

Git in 1 picture

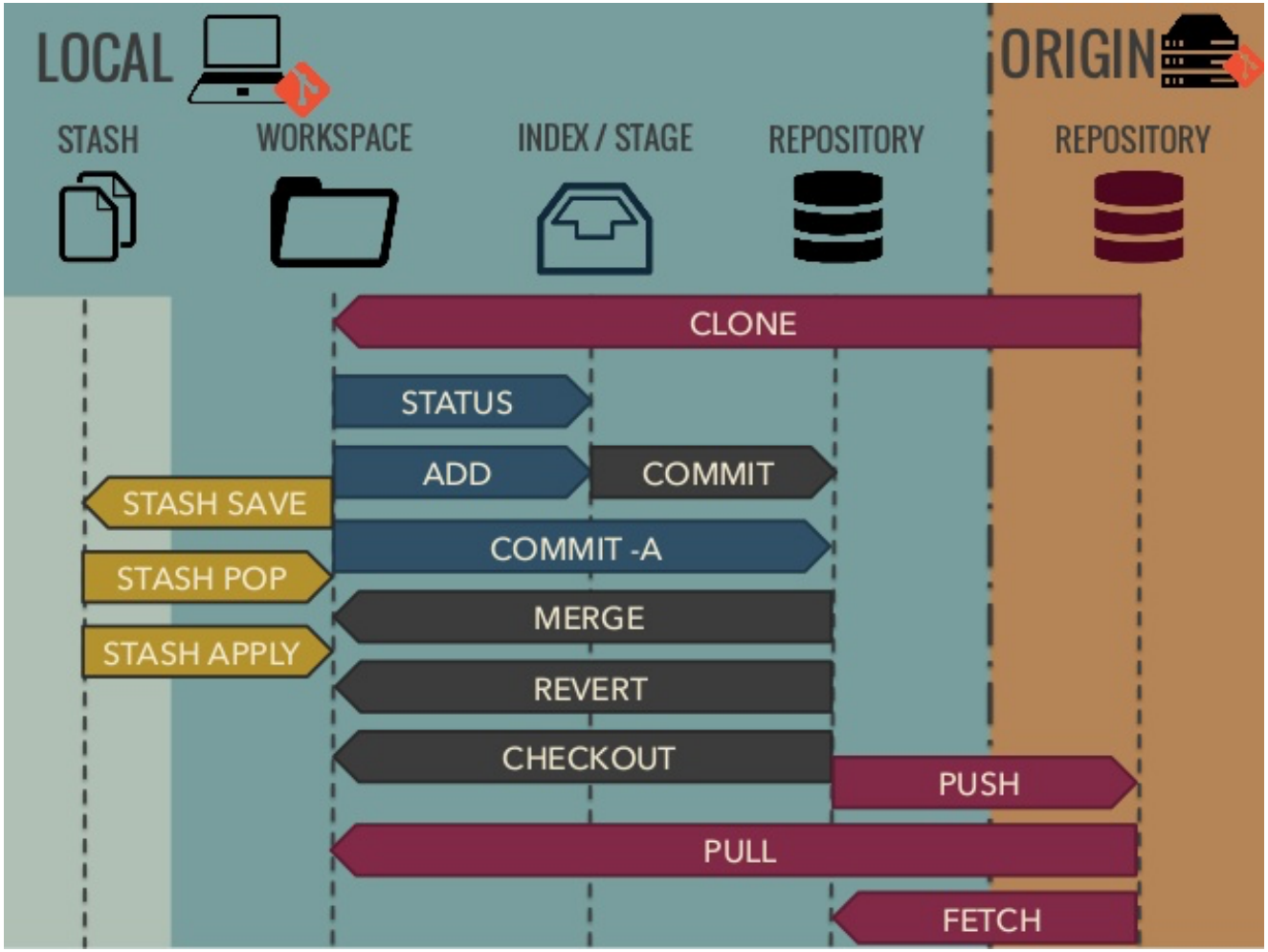

Figure: Git overview.

Git commands (offline)

git <command>

- init: Create an empty Git repository or reinitialize an existing one.
- status: Show the working tree status.
- add: Add file contents to the index.
- commit: Record changes to the repository.
- commit -am: add and commit at same time.
- clone: Clone a repository into a new directory.

Git commands (offline)

git <command>

- stash list: show list stash.
- stash save "Do something": Stash the changes in a dirty working directory away.
- stash apply: Update changes from stash to directory.
- stash pop: apply and remove current stash.

Git branch

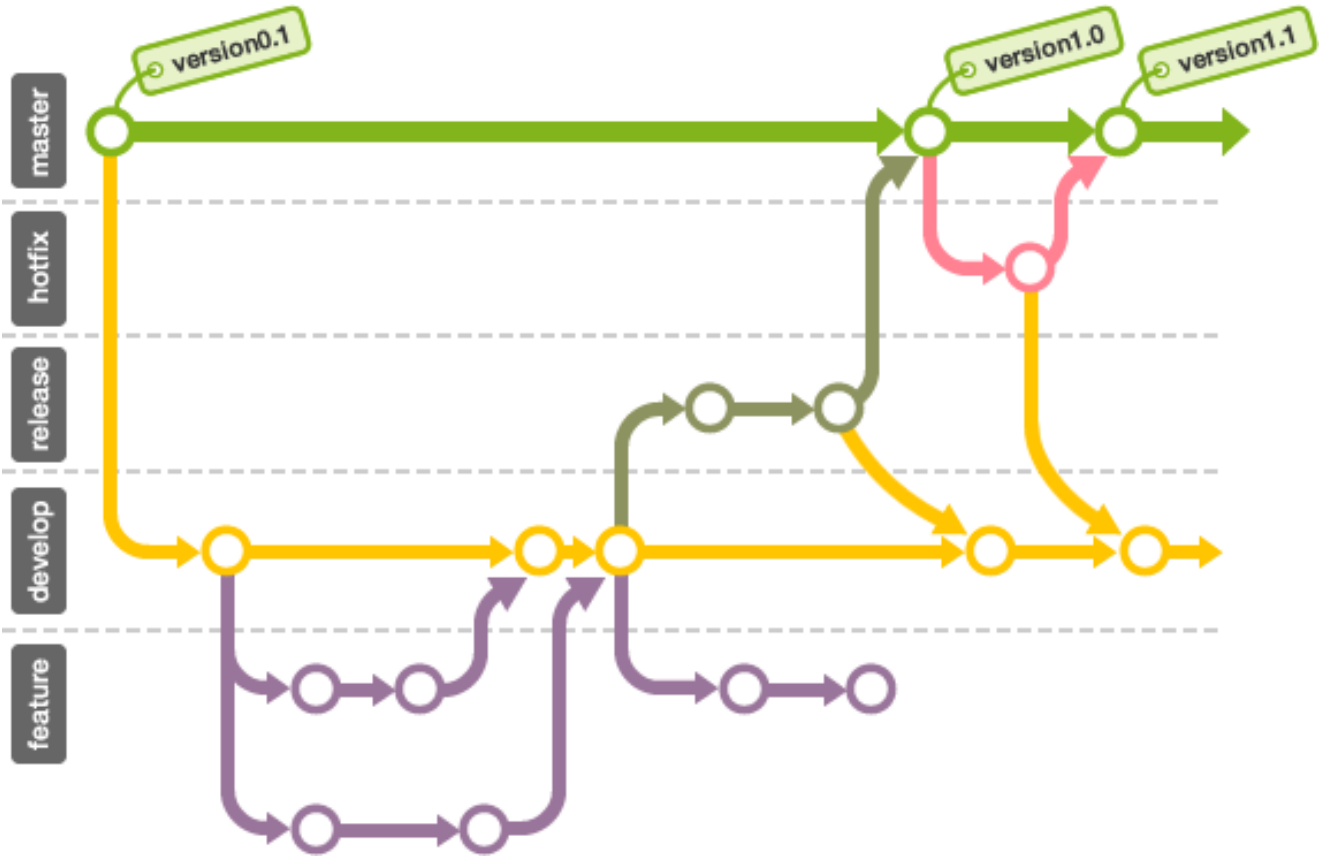

Figure: Git workflow

Git branch

- Try new ideas.
- Isolate new features and production features.
- Flexible for switching between features.
- With 1 working directory can store many phases at the same time.

Branching

```
# listing branch
git branch

# create new branch
git branch new_branch

# switch to new_branch
git checkout new_branch
```

Branching

```
# rename branch
git branch --move new_feature seo_title

# delete branch
git branch --delete branch_to_delete
```

Merge branch

```
# switch to master branch
git checkout master

# merge branch_to_merge to master branch
git merge branch_to_merge
```

Merge conflict

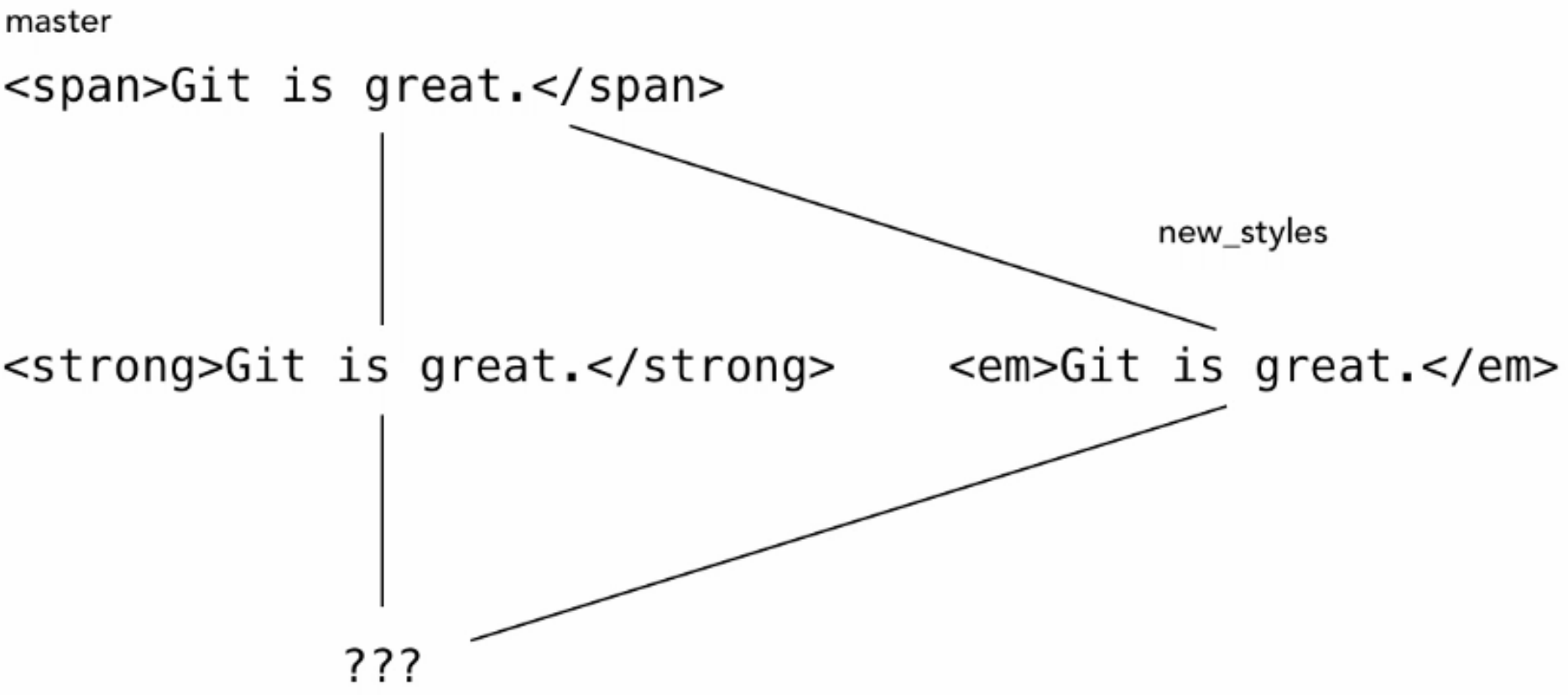

Figure: Git merge conflict.

Merge conflict

```
<<<<<<<<<< HEAD
<strong>Git is great.</strong>
======
<em>Git is great.</em>
>>>>>>>>>> text_edits
```

Merge conflict (resolve)

```
# 1) abort merge command
```

git merge –abort

```
# Or 2) change the content and recommit
```

git add .
git commit -m "Done merge with selected content"

Git in 1 picture

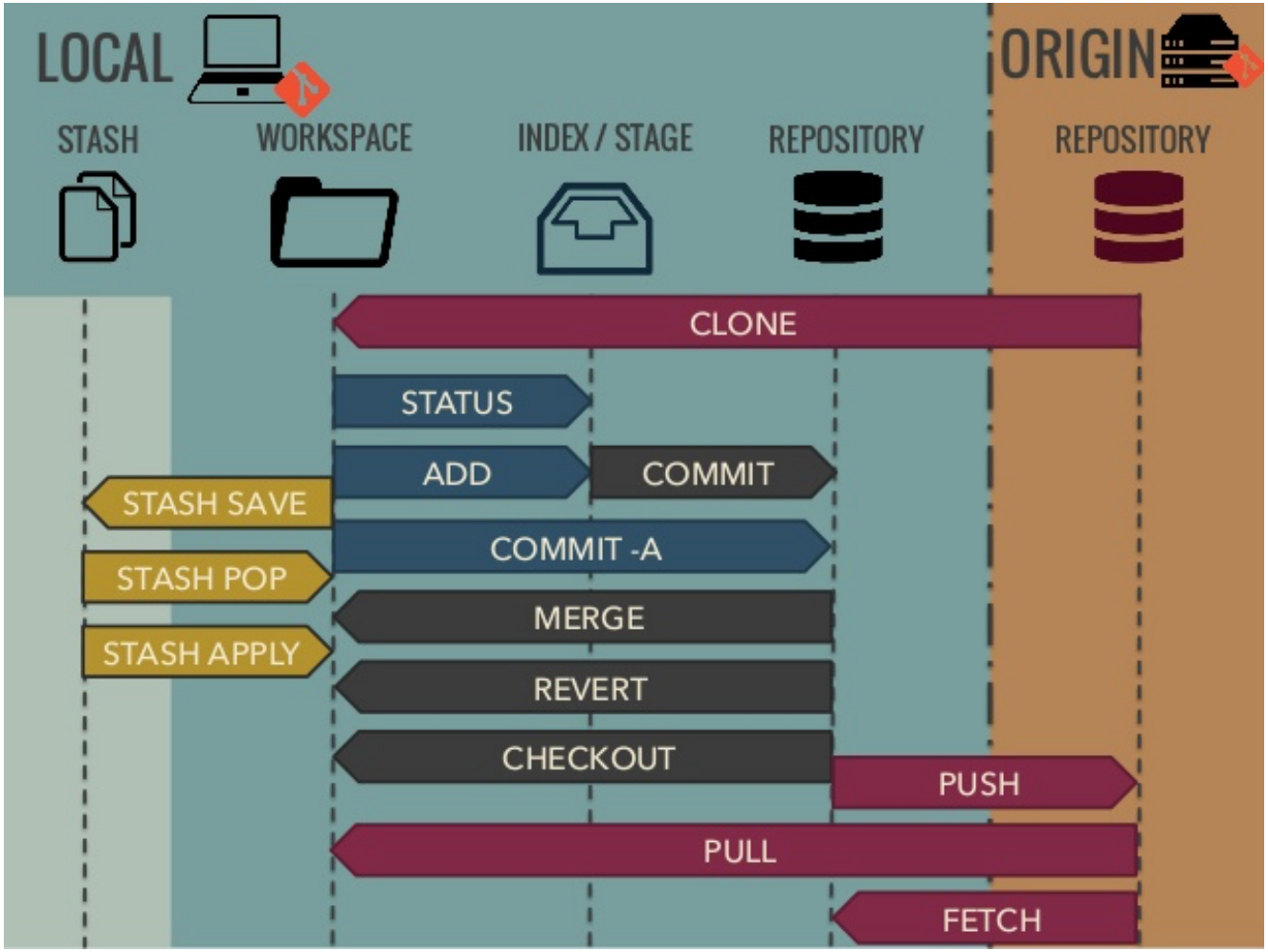

Figure: Git overview.

Git commands (online/remote)

git:

- fetch: Download objects and refs from another repository.
- merge: Join two or more development histories together.
- pull: fetch and merge at same time.
- push: Update remote refs along with associated objects.

Contents

Git workflow

Branch:

- **master** for production.
- **develop** for weekly development.
- **features** for team member create new features/scripts and make pull requests.

Notes:

- Protected branches: master, develop.
- Free to generate: feature branches.

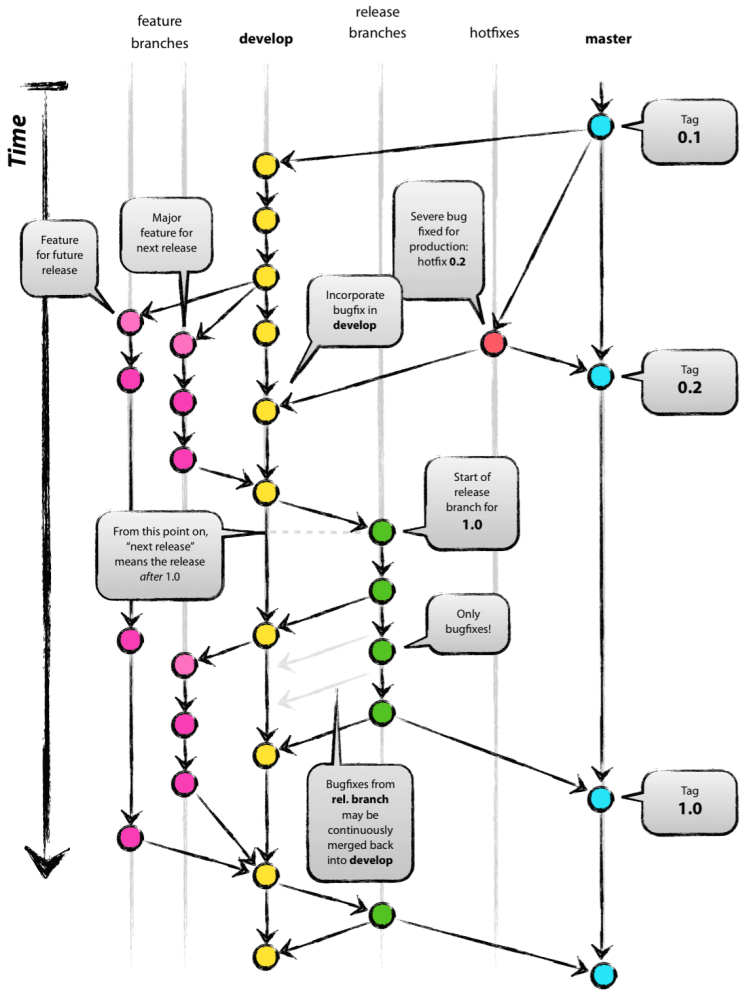

Figure: Git workflow

Contents

User A: create new project

User A do:

- Login to Github/Bitbucket.
- Create new repository "TestRepo" (initialize README.md).
- Clone repository to local machine.
- mv TestRepo TestRepoOrigin
- git checkout -b develop master
- git –set-upstream origin develop

User B: clone new project

User B do:

- Login to Github.
- Clone forked repository to local machine.
- mv TestRepo TestRepoFork

User A: generate project structure

User A do:

- git checkout master
- mkdir lib scripts exploreFeat libNotebookExamples
- cp README.md lib && cp README.md exploreFeat && cp README.md libNotebookExamples && cp README.md scripts
- git add .
- git commit -m "Generate project structure"
- git push
- Check remote repository.

User B: synchronize origin repository

User B do:

- git checkout master
- git pull

User B: developing new feature

User B do:

- git checkout -b feat/hong develop
- Do something.
- git commit -am "hong: adding new feature"
- git push –set-upstream origin feat/hong
- Create pull requests (Please help merge to develop branch).

User A: checking new pull requests

User A do:

- Check pull requests.
- Review code.
- Add comment if needed.
- Confirm merge (if everything ok).

User B: synchronize with origin

User B do:

- git checkout develop
- git fetch upstream origin/develop
- git branch -D feat/hong (delete local branch)
- git push origin :feat/hong (delete remote branch)
- User B may create any files/folders without pull requests (for personal purposes) so that origin repository stays clean.

Tools GUI for Git

- Bitbucket commits
- SourceTree
- PyCharm

Bitbucket commits

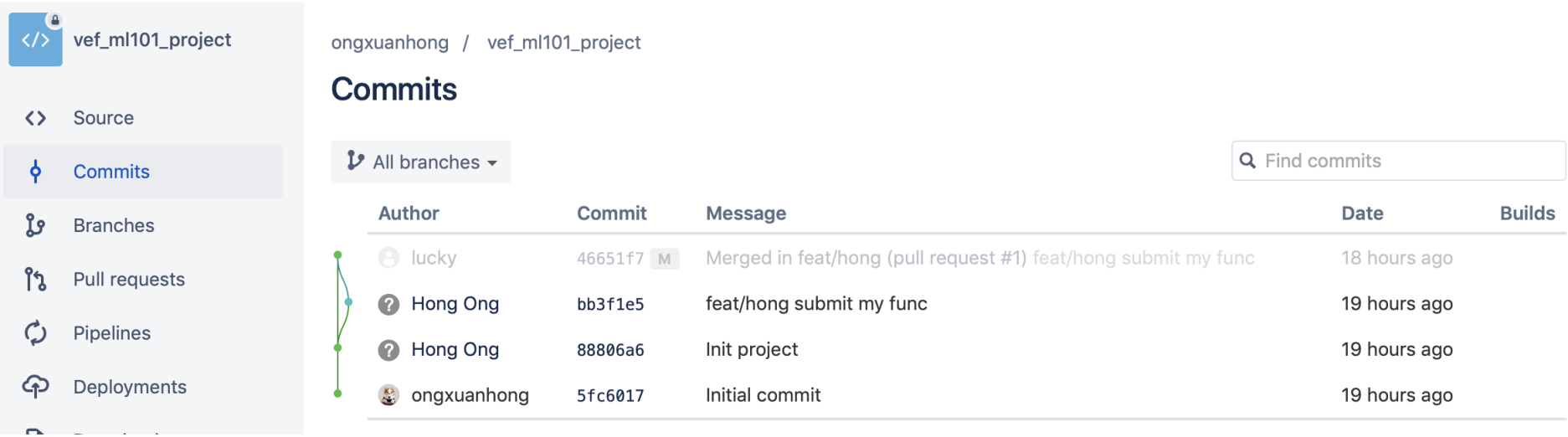

Figure: Bitbucket commits.

SourceTree

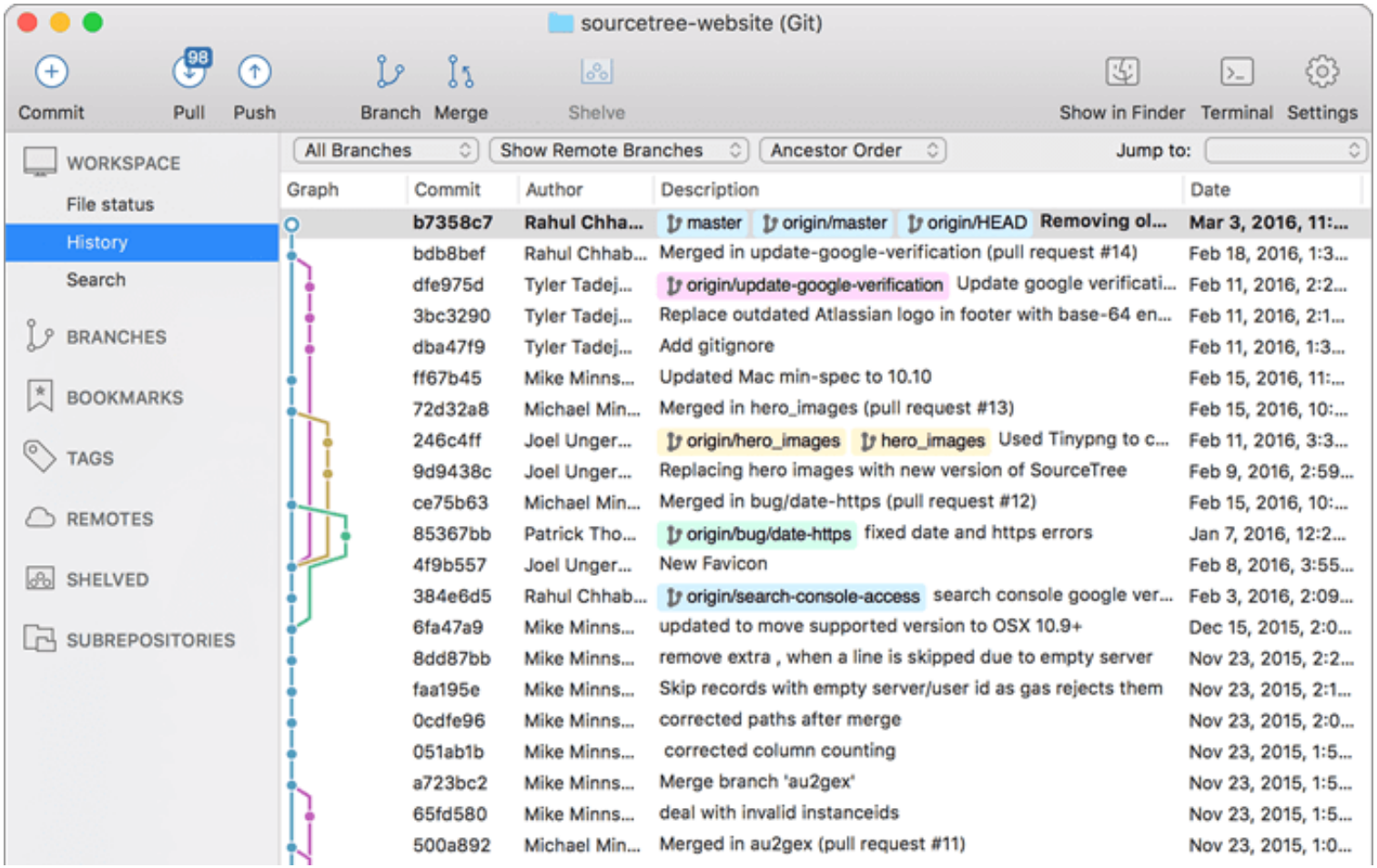

Figure: SourceTree

Contents

References

[1] Vincent Driessen, A successful Git branching model.