Distributed Version Control (with Git)

Introduction and Tutorial

fhLUG

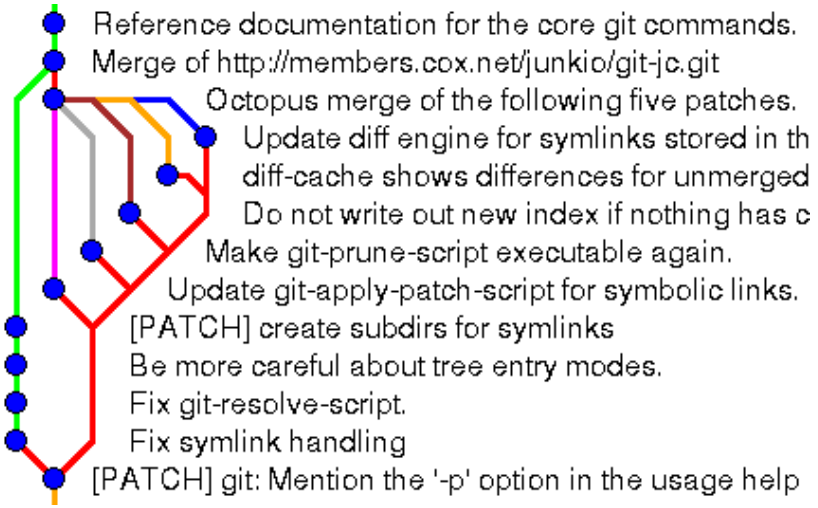
24. 03. 2011

Why Distributed?

- No single point of failure
- Automatic backups
- Fast local operations (log, diff, checkout, …)
- Authenticity of commits
- Easy branching and powerful merging
- Lower barrier for contributors

History Representation

- Commits form a “directed acyclic graph” (DAG)
- Allows precise selection of commits to be merged
- Changes cannot be accidentally merged again



Why Git?

- Because I like it ;)
- Used by the Linux kernel
- Combines speed and feature richness

Why not?

- “Complicated interface”
- Inability to track empty directories
- Necessity of repacking the object database

Projects managed with Git

- Git (obviously)
- Linux Kernel (the reason behind Git's existence)
- GNOME
- PhpMyAdmin
- X.Org
- VLC
- Wine
- Git mirrors: Rockbox, Redmine, Django

Tutorial

- Local:
  - Installation & setup
  - Creating a repository
  - Staging and committing changes
  - Ignoring files
  - Branching & merging
- Remote:
  - Clone
  - Push & pull

Install & Setup

- Install Git (v1.7.4.1)
- Set author and committer info
- Enable colored output

```
# apt-get install git gitk

$ git config --global user.name 'Daniel Knittl-Frank'

$ git config --global user.email 'S10104540016@students.fh-hagenberg.at'

$ git config --global color.ui true
```

Creating a Repository

- You start working on a project
- After a while you need version control

```
$ ls
  docs/ lib/ src/

$ git init

$ git add .

$ git commit
  -m'initial commit'
  [master (root-commit)
  3fd0f1b] initial
```

Staging & Committing Changes

- Git works with changes (at the interface level)
- Changes in files have to be staged
- `'git add -p'` allows to stage separate changes
- `'git reset'` to 'unstage' changes

```
$ vim src/Makefile
$ git add src/Makefile
$ git commit
  -m'makefile magic'
  [master 9bb1e56] makefile
magic
  1 files changed, 100
  insertions(+), 0
  deletions(-)
  create mode 100644
  src/Makefile
```

Ignoring files

- You don’t want all files to be versioned
  - Temporary files
  - Build products
  - Configuration
- Does not work for already tracked files

```
$ >.gitignore cat
*~
*.swp
./build
config.xml
^D
$ git add .gitignore
$ git commit
-m'adding
.gitignore'
```

Stash

- Temporarily ‘hide’ changes
- Handy for urgent bugfixes
- You will love this feature

```
$ git stash save
Saved working directory
and index state WIP on
master: 6c528aa Merge
branch 'B'
HEAD is now at 6c528aa
Merge branch 'B'

$ # fix a bug

$ git stash pop
Dropped refs/stash@{0}
(f8559222cfd8ec100554f9061
244b5d011c6e824)
```

Tag

- Permanently assign a name to a commit and its history
- Tags can be signed with PGP/GPG

```
$ git tag v0.1
$ git tag -s -m'this is a signed tag' v0.2
$ git tag -v v0.2
object 6c528aa115267c2edf…
type commit
tag v0.2
tagger Daniel …

this is a signed tag
gpg: Signature made Sun 20 Mar 2011 …
gpg: Good signature from …
```

Branch

- Different lines of development
- Feature branches
- Custom configuration

```
$ git branch A
$ git checkout A
  Switched to branch 'A'
$ git checkout -b B
  Switched to a new branch
  'B'
$ git branch
    A
  * B
    master
```

Merge

- Changes in branches need to be synced
- ‘Real’ merges are recorded in history
- When one branch is fully contained in the other, a ‘fast-forward merge’ is performed

```
$ # new commits on both
  ‘master’ and ‘B’
$ git checkout master
$ git merge B
  Merge made by recursive.
```

Clone

- First thing to do when working on a foreign project
- Retrieves complete history of a project

```
$ git clone
  git://git.kernel.org/
  pub/scm/git/
  git.git my-git
  Cloning into my-git
  Remote: Counting objects:
  110550, done.

$ cd my-git

$ git branch -a
  * master
    origin/HEAD -> origin/master
    …
```

Pull

- Fetch all new commits and then merges upstream branch of current branch
- `'git fetch'` does the same thing without merging

```
$ git pull
…
$ git pull origin
  master
$ git fetch origin
```

Creating a Mirror

- Special ‘bare’ repositories are necessary for push operations
- It’s basically a .git directory without its working tree
- Local repositories need to be told about the mirror or cloned from it

```
$ ssh://knittl@…
$ git init --bare
  mirror.git
$ logout
$ git remote add mirror
  ssh://knittl@…/mirror
  .git
$ # or create a new clone
$ git clone
  ssh://knittl@…/mirror
  .git mirror
```

Push

- Pushes local changes to the specified remote repository

```
$ git push
$ git push origin
  master
$ git push
  ssh://knittl@…/mirror
  .git branchA branchB
```

Specifying Commits

- Git provides a special syntax to name commits and commit ranges

```
$ git log
  branchA...branchB~5

$ git format-patch
  origin/master..master

$ git diff
  origin/next^^2^

$ git show ':/fix nasty
  bug'
```

Browsing History

- Several ways to browse/visualize history
- Commit ranges may be specified

```
$ gitk --all
```

```
$ tig
```

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

Rebase

- Take a line of history and move it on top of another commit
- Potentially dangerous operation
- Rewrites (as in creates new) history
- Cherry-pick: similar concept, but for single commits

```
$ # 10 last commits to master
$ git rebase --onto master featureA~10 featureA
$ # current branch to master
$ git rebase master
```

Finding Errors: Bisect

- Often we want to know when a certain breakage (bug, regression) was introduced
- Bisect allows to search commits in logarithmic time (10 steps for 1000 commits)

```
$ git bisect HEAD v1.5
  Bisecting: 675
  revisions left to
  test after this

$ # run your tests and
  mark as good or bad

$ git bisect (bad|good)

$ # 0 good, <127 bad:

$ git bisect run
  unit_test.sh
```

Other Distributed Systems (FOSS)

- Mercurial – Direct competitor of Git
- Bazaar – Developed by Canonical (Ubuntu)
- Fossil – Integrated wiki and bug-tracker
- Monotone – PGP-Keys for authenticity
- Darcs – Algebra of patches
- ...

More Interesting Stuff

- Git website: http://git-scm.com/
- Google tech talk by Linus Torvalds:
  http://youtube.com/watch?v=4XpnKHJAok8
- Pro-Git book: http://progit.org/
- Many many more ...

The End.

Questions?
Comments?
Pub!