Operating Systems And Applications For Embedded Systems

The Git Version Control System

Plan I

- Git
  - Before You Start
  - Creating and Sharing a New Repository
  - GIT Hosting
  - Changing a brench
  - Available forms for repository
- Selected Commands
  - help
  - add
  - archive
  - branch
  - clone
  - commit
  - config
  - diff
  - fetch

Plan II

init
merge
pull
push
remote
remove
status

Before You Start I

1. git config –global user.name 'John Smith'
2. git config –global user.email 'jsmith@example.com'

Before You Start II

3. git clone git://git.kernel.org/pub/scm/linux/kernel/git/torvalds/ linux-2.6.git linux-2.6
   Initialized empty Git repository in /tmp/linux-2.6/.git/
   remote: Counting objects: 1177432, done.
   remote: Compressing objects: 100% (189064/189064), done.
   remote: Total 1177432 (delta 982454), reused 1176803 (delta 981879)
   Receiving objects: 100% (1177432/1177432), 288.00 MiB | 406 KiB/s, done.
   Resolving deltas: 100% (982454/982454), done.
   Checking out files: 100% (27842/27842), done.
4. cd linux-2.6

Before You Start III

5. git checkout v2.6.20
   Checking out files: 100% (33554/33554), done.
   Note: moving to 'v2.6.20' which isn't a local branch
   If you want to create a new branch from this checkout, you may do so
   (now or later) by using -b with the checkout command again. Example:
   git checkout -b <new_branch_name>
   HEAD is now at 62d0cfc... Linux 2.6.20

Creating and Sharing a New Repository

1. cd my-project
2. git init
   Initialized empty Git repository in .git/
3. git add .
4. git commit -m 'Initial commit'
5. git clone –bare /tmp/my-project /home/git/my-project.git
   Initialized empty Git repository in .git/
6. cd /home/git/my-project.git
7. git config core.sharedRepository group
8. chgrp -R mygroup .
9. chmod -R g+rwX .

GIT Hosting

1. git daemon
2. Gitosis
3. git shell
4. http://github.com
5. Gitorious
6. http://repo.or.cz

Changing a brench

1. git checkout new-feature
2. git checkout -b new-feature origin/master
   Branch new-feature set up to track remote branch refs/ remotes/origin/master.
   Switched to a new branch ńew-feature"

Available forms for repository

1. /path/repo.git
2. rsync://hostname/path/repo.git
3. http://hostname/path/repo.git
4. https://hostname/path/repo.git
5. git://host/path/repo.git
6. ssh://host/path/repo.git
7. ssh://user@host:port/path/repo.git
8. host:path/repo.git
9. user@host:path/repo.git

help

git help
The most commonly used git commands are:
add Add file contents to the index
apply Apply a patch on a git index file and a working tree
archive Create an archive of files from a named tree
...

add

git add filename ...
dd or update a file in the index so its changes will be committed upon the next call to git commit. This is called staging the change.

archive

git archive [–remote=repository-path] [–format=zip] revision – [path ...] >outputfile Create a tar or zip format archive from the current repository or from the repository at repository-path if –remote is specified. The default output file format is (uncompressed) tar, but git archive can also produce a zip file instead if you specify –format=zip

branch

git branch [-r] [-a] [–contains commit]
git branch [-f] [–track] branchname [commit]
git branch -m oldname newname
git branch -d|-D [-r] branchname
List, create, rename, and delete branches.

clone

git clone repository [local-directory]
Make a copy of repository (which can be a local or remote Git repository) in local-directory.

commit

git commit [– file ...]
Commit the changes that have already been staged in the index, updating HEAD.

config

config git config [–global] –list
git config [–global] name
git config [–global] name value
List, get, or set configuration values.

diff

git diff first-commit [second-commit] [– path ...]
git diff –cached [first-commit] [– path ...]
Show the differences between two revisions. If paths are specified, restricts the comparison to only the given files or directories.

fetch

git fetch [remote-name]
git fetch repository [remoteref[:localref]]
Fetch commits from a remote Git repository and adds it to the local one.

init

git init [-q] [–bare] [–shared=false|group|all]
Create a new Git repository in the current directory.

merge

git merge commit ...
Merge one or more other branches into the current HEAD.

pull

git pull
git pull repository branch
Fetch the given branch then merge it into HEAD. git pull is just a short form for git fetch followed by git merge.

push

git push [–all] [–tags] [repository [localref][:remoteref]]
Push changes from one or more branches or tags into the specified remote repository.

remote

git remote add [-f] name repository-url
git remote rm name
git remote show name
git remote prune [–dry-run] name
git remote update [name ...]
Manipulate remotes, which act as short forms for tracking repository URLs and branches.

remove

git rm [-f] [-r] – path ...
Remove files from the work tree and index. They will be permanently removed when you run git commit.

status

git status [path ...]
Check what would happen if you ran git commit.

References

C. Simmonds.
Mastering Embedded Linux Programming.
Packt Publishing, 2015.

The End