GIT

Mark Yang

October 2010

COMMON VOCABULARY

- Terminology can vary from system to system, but some terms in common usage include:

- Repository
  - The repository is where files' current and historical data are stored, often on a server. Sometimes also called a depot (for example, by SVK, AccuRev and Perforce).
- Revision
  - Also version: A version is any change in form. In SVK, a Revision is the state at a point in time of the entire tree in the repository.

- Trunk
  - The unique line of development that is not a branch (sometimes also called Baseline or Mainline)
- Branch
  - A set of files under version control may be branched or forked at a point in time so that, from that time forward, two copies of those files may develop at different speeds or in different ways independently of each other.

- Export
  - An export resembles a check-out except that it creates a clean directory tree without the version-control metadata used in a working copy. Often used prior to publishing the contents.
- Import
  - An import involves copying a local directory tree (that is not currently a working copy) into the repository for the first time.

- Change
  - A change (or diff, or delta) represents a specific modification to a document under version control. The granularity of the modification considered a change varies between version control systems.
- Change list
  - On many version control systems with atomic multi-change commits, a changelist, change set, or patch identifies the set of changes made in a single commit. This can also represent a sequential view of the source code, allowing the examination of source "as of" any particular changelist ID.

- Checkout
  - A check-out (or co) creates a local working copy from the repository. A user may specify a specific revision or obtain the latest.
- Update
  - An update (or sync) merges changes made in the repository (by other people, for example) into the local working copy.
- Commit
  - A commit (checkin, ci or, more rarely, install, submit or record) occurs when writing or merging a copy of the changes made to the working copy into the repository.

- Conflict
  - A conflict occurs when different parties make changes to the same document, and the system is unable to reconcile the changes. A user must resolve the conflict by combining the changes, or by selecting one change in favour of the other.
- Resolve
  - The act of user intervention to address a conflict between different changes to the same document.

- Head
  - The most recent commit.
- Delta compression
  - Most revision control software uses delta compression, which retains only the differences between successive versions of files. This allows for more efficient storage of many different versions of files.

- Label, Tag
  - A tag or label refers to an important snapshot in time, consistent across many files. These files at that point may all be tagged with a user-friendly, meaningful name or revision number. See the discussion of baselines, labels, and tags (below).
- Baseline
  - An approved revision of a document or source file from which subsequent changes can be made. See the discussion of baselines, labels, and tags (below).

- Merge
  - A merge or integration is an operation in which two sets of changes are applied to a file or set of files. Some sample scenarios are as follows:
    - A user, working on a set of files, updates or syncs their working copy with changes made, and checked into the repository, by other users.
    - A user tries to check-in files that have been updated by others since the files were checked out, and the revision control software automatically merges the files (typically, after prompting the user if it should proceed with the automatic merge, and in some cases only doing so if the merge can be clearly and reasonably resolved).
    - A set of files is branched, a problem that existed before the branching is fixed in one branch, and the fix is then merged into the other branch.
    - A branch is created, the code in the files is independently edited, and the updated branch is later incorporated into a single, unified trunk.

WHAT IS GIT?

- Git is...
  - directory content management system
  - tree history storage system
  - stupid content tracker
  - a toolkit
  - not subversion!

WHAT IS GIT

- Git is NOT an evolution

delta storage
C1
C2
C3
C4
C5
file A
file B
file C
Δ1
Δ2
Δ1
Δ2
Δ1
Δ2
Δ3
DAG storage
C1
C2
C3
C4
C5
A
A1
A1
A2
A2
B
B
B
B1
B2
C
C1
C2
C2
C3

delta storage
C1
C2
C3
C4
C5
file A
Δ1
Δ2
file B
Δ1
Δ2
file C
Δ1
Δ2
Δ3
DAG storage
C1
C2
C3
C4
C5
A
A1
A1
A2
A2
B
B
B
B1
B2
C
C1
C2
C2
C3

delta storage
local
rcs
centralized
cvs
svn
perforce
distributed
darcs
mercurial
DAG storage
local
cp -r
time machine
centralized
bitkeeper
distributed
git
bazaar

OBJECT DATABASE (REPOSITORY)

- All git objects are stored this way:

blob

tree

commit

tag

- A **blob** object is the content of a file. Blob objects have no names, timestamps, or other metadata.
- A **tree** object is the equivalent of a (sub)directory: it contains a list of filenames, each with some type bits and the name of a blob or tree object that is that file, symbolic link, or directory's contents. This object describes a snapshot of the source tree.

- A **commit** object links tree objects together into a history. It contains the name of a tree object (of the top-level source directory), a timestamp, a log message, and the names of zero or more parent commit objects.

- A **tag** object is a container that contains reference to another object and can hold additional meta-data related to another object. Most commonly it is used to store a digital signature of a commit object corresponding to a particular release of the data being tracked by Git.

BLOB & TREE

Working Directory
Git Directory
./
tree : 1a738d
Rakefile
blob : a874b7
README
blob : a906cb
lib/
tree : fe8971
simplegit.rb
blob : a0a60a

COMMIT

commit
tree
blob

tag
commit
tree
blob

GETTING GIT

- Web site: http://git-scm.com
- Download: http://www.kernel.org/pub/software/scm/git/
- Different platform:
  - Redhat : yum install git-core
  - Debian : apt-get install git-core
  - Mac : port install git-core +svn

SETUP

- git config --global user.name “Mark Yang”
- git config --global user.email “markyang@skyviia.com.tw”
- Config file : ~/.gitconfig

GETTING A REPO

- git clone git@172.26.40.16:test/testproject.git
- Git protocol :
  - ssh://markyang@gitserver:29418/test/testproject.git
  - http[s]://gitserver/test/testproject.git
  - git://gitserver:test/testproject.git

INITIALIZING A REPO

- Initializing:
  - $ cd myproject
  - $ git init
  - $ git add .
  - $ git commit -a -m “my comment bla bla bla.”
- Push to remote:
  - $ git remote add origin git@gitserver:test/myproject.git
    (Only in the first time!!)
  - $ git push origin master

GIT & REPO

- Repo
  - is a tool that we built on top of Git.
  - Repo helps us manage the many Git repositories, does the uploads to our revision control system , and automates parts of the Android development workflow.
  - Repo is not meant to replace Git, only to make it easier to work with Git in the context of Android.
  - The repo command is an executable Python script that you can put anywhere in your path.

ROLES AND RELATION SHIP

ROLES AND RELATION SHIP

ENVIRONMENT SETTING

- NIS Server : 172.26.201.31 r3lnx01
- SVN Server : 172.26.201.31 r3lnx01
- Build Machine : 172.26.201.32 r3lnx02
- Git Server : 172.26.40.16 gitserver
- Git Web : http://172.26.40.16
- Gerrit Server : https://172.26.40.16

GITWEB

http://172.26.40.16

To clone one of these trees, install git, and run:

```
$ git clone git@172.26.40.16: + project path.

Example:
$ git clone git@172.26.40.16:test/testproject.git
```

To clone the entire platform, install skyviia repo:

```
$ curl http://172.26.40.16/repo > ~/bin/repo
$ chmod a+x ~/bin/repo
```

and run:

```
$ mkdir skydroid
$ cd skydroid
$ repo init -u git://172.26.40.16/platform/manifest.git
$ repo sync
```

For more information about git, see an overview, the tutorial or the man pages.

Skydroid / git

Search:

Project	Description	Owner	Last Change	
device/common.git	none	android-team	8 days ago	summary | shortlog | log | tree
device/htc/common.git	none	android-team	2 weeks ago	summary | shortlog | log | tree
device/htc/dream-sapphire.git	Dream / Sapphire device config...	android-team	2 weeks ago	summary | shortlog | log | tree
device/htc/dream.git	Dream (G1) device configuratio...	android-team	2 weeks ago	summary | shortlog | log | tree
device/htc/passion-common.git	Nexus one device common config...	android-team	2 weeks ago	summary | shortlog | log | tree
device/htc/passion.git	Nexus one device configuration...	android-team	2 weeks ago	summary | shortlog | log | tree
device/htc/sapphire.git	Sapphire device configuration...	android-team	2 weeks ago	summary | shortlog | log | tree
device/sample.git	Device configuration sample...	android-team	2 weeks ago	summary | shortlog | log | tree
device/skyviia/Library.git	SV886x specified libraries	android-team	2 days ago	summary | shortlog | log | tree
device/skyviia/SkyLibrary.git	SkyLibrary JNIs	android-team	2 days ago	summary | shortlog | log | tree
device/skyviia/Tools.git	Tools for SV886x	android-team	2 weeks ago	summary | shortlog | log | tree

GERRIT

https://172.26.40.16

All | My | Admin | Documentation

Mark Yang (1000

Open Merged Abandoned

status:merged

Change I1601111f: Modified for A3-2.2

Change-Id:	I1601111f56111e095f90cf24ec8299baf1fb4d84
Owner	Anonymous Coward (1000005)
Project	platform/packages/apps/SkyLauncher
Branch	master
Topic	
Uploaded	Oct 7, 2010 6:03 PM
Updated	Oct 7, 2010 6:09 PM
Status	**Merged**

Permalink

```
Modified for A3-2.2

Change-Id: I1601111f56111e095f90cf24ec8299baf1fb4d84
```

Reviewer	Verified	Code Review	
Michael Tsai	✓	✓	Verified; Looks good to me, approved

Included in

Dependencies

Patch Set 1 577c133de9d4458ac331b5f8df78115a33da695e (gitweb)

Author	Ethan Chen <ethanchen@skyviia.com.tw> Oct 7, 2010 6:05 PM
Committer	Ethan Chen <ethanchen@skyviia.com.tw> Oct 7, 2010 6:05 PM
Download	checkout | pull | cherry-pick | patch | SSH | HTTP

```
git fetch ssh://markyang@gitserver:29418/platform/packages/apps/SkyLauncher refs/changes/59/59/1 && git checkout FETCH_HEAD
```

Review | Diff All Side-by-Side | Diff All Unified

	File Path	Comments	Size	Diff		Reviewed
	Commit Message			Side-by-Side	Unified	
M	src/com/skyviia/skylauncher/BookMarksAdapter.java	**1 comment**	+66, -90	Side-by-Side	Unified	
M	src/com/skyviia/skylauncher/BookMarksGridView.java		+1, -1	Side-by-Side	Unified	
			+67, -91			

Comments

Expand Recent | Expand All | Collapse All

Michael Tsai Patch Set 1: (1 inline comment) Oct 7

Michael Tsai Patch Set 1: Verified; Looks good to me, approved Oct 7

Michael Tsai Oct 7

Change has been successfully merged into the git repository.

Expand Recent | Expand All | Collapse All

INSTALL REPO

- Installing Repo
  - $ curl http://172.26.40.16/repo > ~/bin/repo
  - $ chmod a+x ~/bin/repo
  - Add ~/bin into your $PATH
  - $ mkdir working-directory-name
  - $ cd working-directory-name
  - $ repo init-u git://172.26.40.16/platform/manifest.git

SYNCHRONIZE YOUR CLIENT

- To synchronize the files for all available projects:
  - $ repo sync
- To synchronize the files for selected projects:
  - $ repo sync project1 project2 ...
- Web site : http://source.android.com/source/git-repo.html

GERRIT CODE REVIEW

- Gerrit : Web based code review and project management for Git based projects.
- Gerrit is a web based code review system, facilitating online code reviews for projects using the Git version control system.
- Gerrit makes reviews easier by showing changes in a side-by-side display, and allowing inline comments to be added by any reviewer.

REPO/GERRIT/GIT WORKFLOW

- Android Git Workflow : http://groups.google.com/group/repo-discuss/web/repo-gerrit-git-workflow
- This is overview of the standard workflow we suggest for working on a project using repo.
- It is designed as a simple outline reference, with links to more details on each step.
- Your first time through you should read through the details as you follow the steps.
- For more information about repo, see : http://source.android.com/source/git-repo.html

SIMPLE WORKFLOW

- Setup repo : **$ repo init -u git://172.26.40.16/platform/manifest.git**
- Retrieve current source : **$ repo sync**
- Select work project : **$ cd project/path**
- Create work branch : **$ repo start branch_name .**
- Modify your file...
- Add files if needed : **$ git add path/file**
- Commit files to branch : **$ git commit -a -m “bla bla bla...”**
- Check status : **$ repo status .**
- Send out review : **$ repo upload .**
- Find a approver
- Submit change in gerrit (the code review web site)
- Finish branch(es) if desired : **$ repo prune** or **$ repo abandon**

Git
Repo
Gerrit
2. repo sync
Developer
1. repo init -u http://...
3. cd working-dir
4. repo start branch .

Git
Repo
Gerrit
7. repo upload .
Developer
5. start to coding
6. git add / commit

Git
Repo
9. submit
Gerrit
8. review
9. publish comment
Developer
10. repo prune

HOW TO CREATE NEW PROJECT

- Find administrator to add a project!
  - git repository path name, ex: platform/framework/base.git
- Push repository of new project from remote client!
  - git init; git add .; git commit -a; git add remote....; git push;
- Find administrator to restart gerrit server to import new project.
- Add your repository into /platform/manifest.git/default.xml if necessary.
- Assign properly permissions in Gerrit.
- Announce to team member.

Q&A