


SENG 371  
SOFTWARE EVOLUTION



ADVANCED GIT

Prepared by  
Pratik Jain

Reference :-  
<http://git-scm.com/book/en>

LAB OUTLINE

- Introduction to VersionControlSystem
- Subversion and Git
- Github

GIT BRANCHING

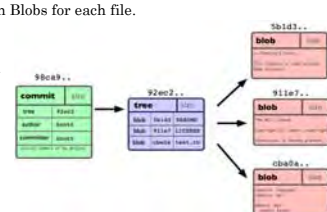
- Git stores data as snapshots not as differences.
- Every time you commit Git stores commit object which contains pointer to snapshot of content you staged, author and message metadata.
- It also stores zero or more pointer to commit. Zero parent for first commit, one parent for normal commit and multiple parent for commit resulting from merge.

Reference :-  
<http://git-scm.com/book/en>

GIT BLOBS

```
$ git add README test.rb LICENSE
$ git commit -m 'initial commit of my project'
```

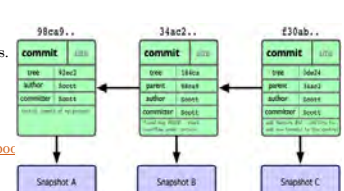
- Staging the files checksums each one, and stores content in Blobs for each file.
- Git repository containing 5 objects.
- Single commit repository will look like this :



Reference :-  
<http://git-scm.com/book/en>

MULTIPLE COMMITS

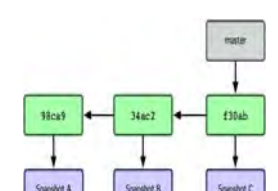
- Make some changes , commit again. Next commit points to commit before it.
- Check the checksum for parent commit.
- Git data for Multiple commits.



Reference :-  
<http://git-scm.com/book/en>

MASTER BRANCH

- Branch is lightweight movable pointer to commits.
- Default branch name is Master.
- Master branch points to last commit you made.

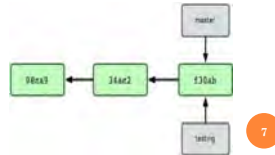


Reference :-  
<http://git-scm.com/book/en>

### CREATE BRANCH

- o Creating a new branch, will create a new pointer to the same commit you are currently in.

`$ git branch testing`

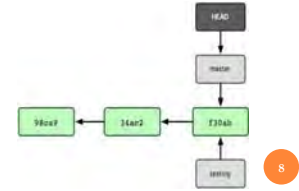


Reference :- <http://git-scm.com/book/en>

7

### GIT HEAD

- o Git has special pointer called Head to track in which branch you are in.
- o It's different from other VCS, In git it is a pointer to a local branch we are currently working with.
- o Local branches live in `.git/refs/heads`. Branches from other repositories are maintained in `.git/refs/remotes`.

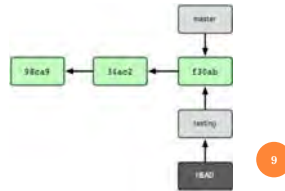


Reference :- <http://git-scm.com/book/en>

8

### GIT CHECKOUT

- o `git branch` command only creates branch.
- o To switch it to testing branch use :-
- o `git checkout testing`
- o Now heads point to testing.

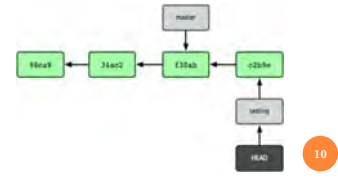


Reference :- <http://git-scm.com/book/en>

9

### GIT CHECKOUT

- o Change some file and commit.
- o Now check the checksum and Head.
- o Master is still on last checkout commit of master branch.

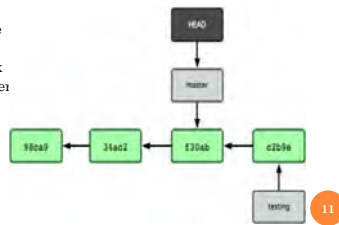


Reference :- <http://git-scm.com/book/en>

10

### REVERT TEMPORARY CHANGES

- o Switch again to master branch.
- o `git checkout master`
- o This will change the Head pointer to master and revert changes done in working directory. Now files will point back to snapshot where master points to .

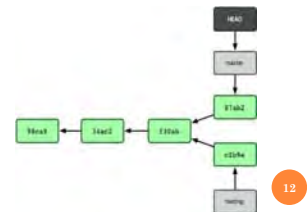


Reference :- <http://git-scm.com/book/en>

11

### PROJECT HISTORY

- o Lets make some more changes, then project flow will diverge from temporary testing branch.
- o In Git with Simple `branch` and `checkout` commands to and from moment is possible between branches.
- o Branch in git is simple checksum of 40 character. Its easy to create and destroy.



Reference :- <http://git-scm.com/book/en>

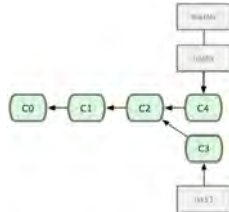
12

### GIT FAST FORWARD MERGE

- When commit pointed to by the branch in a merging is just upstream to commit you are on then it is called as fast forward merge.

`git merge hotfix`

- Git is just moving pointer forward, because there is no divergent work.



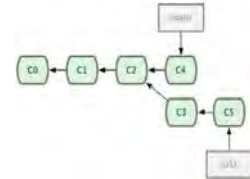
Reference :- <http://git-scm.com/book/en>

13

### DELETE HOTFIX

- Once we are done with our merge and hotfix is part of our master we can delete that branch.

`$ git branch -d hotfix`



Reference :- <http://git-scm.com/book/en>

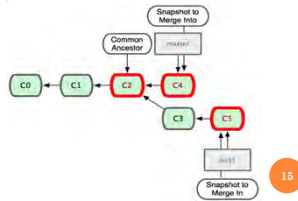
14

### BRANCH MERGE

- We will do merge in similar way we did it last time.

`$ git checkout master`  
`$ git merge iss53`

- Commit on the branch we are, is not a direct ancestor of branch. We are merging in.
- Git does a simple three-way merge, using the two snapshots pointed to by the branch tips and the common ancestor of the two.

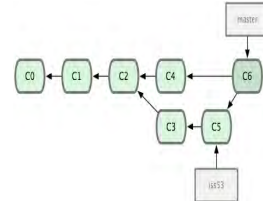


Reference :- <http://git-scm.com/book/en>

15

### MERGE COMMIT

- Git creates a new snapshot that results from this three-way merge and automatically creates a new commit that points to it. This is referred to as a merge commit and is special in that it has more than one parent.



Reference :- <http://git-scm.com/book/en>

16

### LAB EXERCISE

- Please follow lab manual for Git commands.

17

### REFERENCES

- [GIT- Book](#)
- [GIT - Quick Reference](#)

18