Saverio Proto (ZioPRoTo)

BattleMesh V6
Aalborg Denmark

uname

- uname -a
  - I am zioproto
  - I learnt git because of Halino @ Google Summer of Code 2012. Thanks to him ☺
- I work for GARR NOC
  - In the night I do Ninux NOC
  - NOC life 24h ☺
- Forget the past (svn, cvs)
- Add me to github
- http://github.com/zioproto

Git object store

- Forget about client and server
- Git init
  - The .git folder is your repository
  - Everything is local
- It is better to know the object store database
- Object type
  - Blobs: binary large object
  - Trees: represents one lever of directory information
  - Commits: Metadata pointing to a tree
  - Tags: human name connected to a object (usually a commit)

Hash Functions

- **each object in the object store has a unique name produced by applying SHA1 to the contents of the object**
  - SHA1 = 160bit digest

```
git init
ls .git/
ls .git/objects/
touch file
git add file
git write-tree
ls .git/objects/
git cat-file -p df2b8
```

Git Index

- The git Index
  - The index is a temporary and dynamic binary file that describes the directory structure of the entire repository. you execute Git commands to stage changes in the index. Changes usually add, delete, or edit some file or set of files. The index records and retains those changes, keeping them safe until you are ready to commit them

- Be aware of the index when using
  - Git add
  - Git rm
  - Git mv

Git index

- **Files can be:**
  - **Tracked**
    - **Any file staged in the index. Use git add to track files**
  - **Ignored**
    - **File that git ignores because specified in the .gitignore file**
  - **Untracked**
    - **Not tracked, not ignored**

commit

- Commit introduces a change in the repository
  - Freeze the index and add metadata
  - Commit by user
  - Commit by git itself (merge)
- Access a commit with explicit or relative reference
  - Git checkout ORIG_HEAD
  - Git checkout aefc6f
  - Git checkout HEAD~1
- Refs and symrefs
  - They are names that point to git objects
  - HEAD ORIG_HEAD FETCH_HEAD

commit

- The commit knows its parents
  - And it do not know its childs !

branches

- Git branch
  - Its just a ref to a commit
- Git checkout
  - Load the requested commit in the working directory
  - Git works on a single branch per time
- When you commit, the ref of the active branch moves to the new created commit
- Delete a branch
  - Just delete the ref
  - Git garbage collector will later delete unreferenced commits

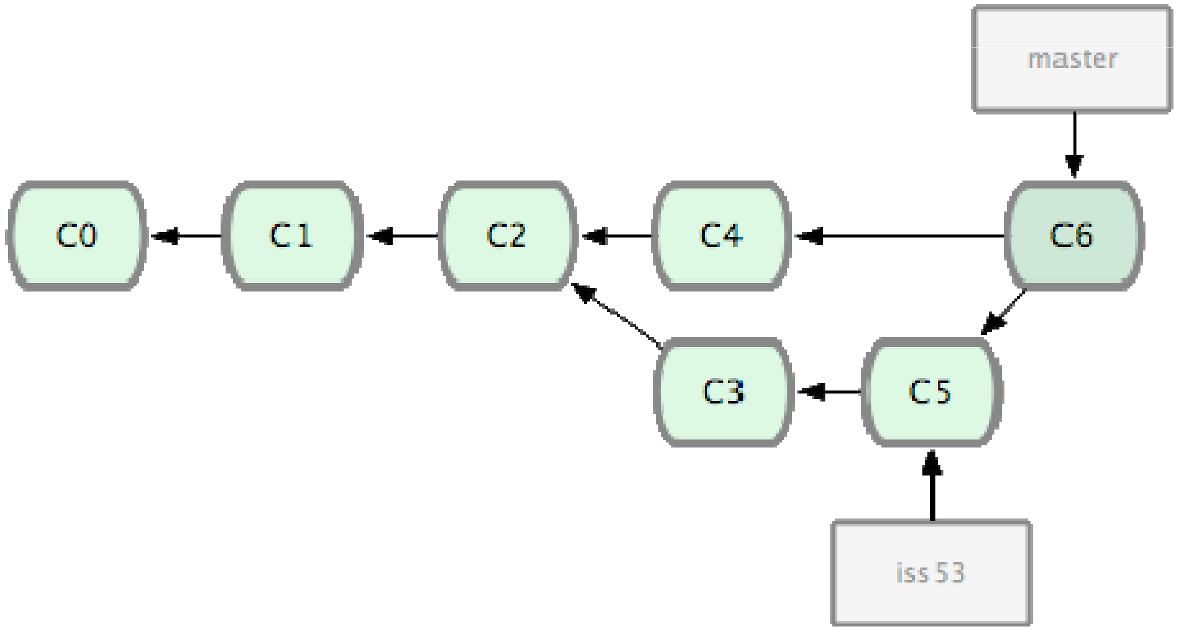
diff

- Its difficult because it is implicit
- Git diff
  - diff working directory & index
- Git diff <commit>
  - diff working directory & commit
  - use –cached for diff index & commit
- Git diff <commit1> <commit2>
  - Diff the two commits commit
- Git diff <branch1>...<branch2>
  - Checkout from book “git diff and Commit Ranges”

merge

- **Current branch is always the target branch**

- **You can merge with conflicts**
- **A merge produces a new commit with maybe two parents**



Git cherry-pick and git revert

- Git cherry-pick and git revert and easy and usefull tools !

Altering commits

- Altering commits
  - Do it only if you did not share yet

Altering commits – git rebase

- **Git rebase**
  - Moves a branch to another base
- **Git rebase interactive**
  - Git rebase –i
  - Squash multiple useless commits into less significant commits
  - Be a real git ninja !

Altering commits – git reset

- Git reset –soft <commit>
  - --soft changes the HEAD ref to point to the given commit
- Git reset –mixed <commit>
  - This is the default behaviour
  - --mixed changes HEAD to point to the given commit. Your index contents are also modified to align with the tree structure named by commit, but your working directory contents are left unchanged.
- Git reset –hard <commit>
  - This variant changes the HEAD ref to point to the given commit. The contents of your index are also modified to agree with the tree structure named by the named commit. Furthermore, your working directory contents are changed to reflect the state of the tree represented by the given commit.

Git remote repositories

- Let's rock on the Internet !

- Git remote
  - Manage remote git repositories
  - You will store the complete remote repository in local into a special readonly branch ref

- Git clone
  - Startup from a "origin"

- Git push
- Git pull (I like better "git fetch")

Random stuff

- Git blame
- Git bisect

```
saverio@nockid:~$ cat .gitconfig
[user]
        name = Saverio Proto
        email = zioproto@gmail.com
[core]
        editor = vim

[color]
        diff = auto
        branch = auto
        status = auto
[alias]
        superlog = log --graph --all --pretty=oneline --decorate
```

Github

At ninux we are very rich !

Check out our github

http://github.com/ninuxorg

Questions ?