Using Git in the ECASP Lab

By: Spenser Gilliland

3/4/2013
What is Git?

- Decentralized version control. You can safely ignore the decentralized part for our labs configuration.
- Used by many different projects
  - Linux
  - Xilinx
  - Buildroot
  - Etc
- Services available from
  - Github
  - Ephesus
  - etc
Version Control – Don't be scared!

- Simply means keeping multiple versions of a project.
  - Version 0: First time you used git. You thought your project worked well.
  - Version 1: Added an interesting feature.
  - Version 2: Added some awesome new feature, etc.
Branches in Version Control

• This is actually quite simple. Instead of linear versions. We can also have trees of versions.
  – Version 0: First time you used git. You thought your project worked.
    • Version 0.1: Sufeng is using this code but needs a small fix for his project.
    • Version 0.2: Sufeng and Thomas found another bug to have fixed.
  – Version 1: Added an interesting feature.
    • Version 1.1: Weidi needs a small bug fixed but is otherwise happy with this version.
      – Version 1.1.1: Spell Weidi’s name right.
  – Version 2: Added some awesome new feature, etc.

• You just learned the basics of Versioning and Releases!
Patch? Diff?

• A patch is a unified list of all differences (aka. diff) between two versions of a file. Example:

```diff
+++ Portfile 2011-07-25 18:53:35.000000000 -0700
@@ -2,7 +2,7 @@
 PortSystem 1.0
 name foo

-version 1.3.0
+version 1.4.0
 categories net
 maintainers nomaintainer
 description A network monitoring daemon.
@@ -13,9 +13,9 @@
 homepage http://rsug.itd.umich.edu/software/${name}

 master_sites ${homepage}/files/
-checksums rmd160 f0953b21c6b5eb327e40d4b215110b71
+checksums rmd160 01532e67a596bfff6a54aa36face26ae
 extract.suffix .tgz
 platforms darwin
```
Basic Git Usage
(All you really need to start)

- Copy project to your workstation
  - `git clone https://ephesus.ece.iit.edu/git/ <project>.git`
- Check changes in latest version
  - `git log`
- Make changes to source code (use vim, xps, ise, eclipse etc)
- See what you changed.
  - `git status`
- Save changes (runs diff; creates a patch)
  - `git add <changed files>`
  - `git commit -m "Summary of changes"`
- Get others changes and incorporate them into yours (patch yourself)
  - `git pull`
- Push your changes to others. (send patch to server)
  - `git push`
Branching

- Want to work with someone else but don't want to break the others tree.
  - `git branch fix_issue`
  - `git checkout fix_issue`
- Create new branch.
  - `git push origin/fix_issue`
- (on teammates comp) Checkout the branch
  - `git checkout origin/fix_issue`
Tag/Reset/Checkout

- You can even tag commits so others can find them.
  - `git tag v0.0.1`

- You can reset your branch to a tag or branch.
  - `git reset master` # reset to current master tag/branch

- You can checkout any tag/branch with
  - `git checkout tag/branch`
Merging/Rebase

- Git's provides excellent interface for merging.
  - `git mergetool` (starts up meld/kdiff any diff tool)
- Rebase: change patch messages and contents.
  - `git rebase -i`
  - NOTE: Do not rebase the public history. It's not nice to others.
More Reading

- http://byte.kde.org/~zrusin/git/git-cheat-sheet-large.png