Source Control: Fundamentals and Git

Purdue Linux Users Group
CSWN

Speaker: Thor Smith
Outline

- Introduction to Source Control
- Source Control Software
- Source Control Philosophies?
- Create a Project in Git
- List of Commands
- Adding Changes to the Stage
- Commiting Changes to the Project
- Tracking Your Changes
- Branching Out
- Merging Branches
Source control (also known as version control) developed to allow multiple versions of a project to exist.

Separates the working version from developing versions.

Allows groups of people to work on a single project.

Steps

- Make a local copy of files.
- Change them and confirm your changes.
- Write your new changes to the main project.
Source Control Software

- CVS
- SVN
- Git
- Hg
- Bazaar
Two major philosophies

- Centralized Version Control
  - There is one master version of the project which everyone checks into
  - Each person checks out a view of the project.

- Decentralized Version Control
  - There may be many master versions of a project.
  - A single project may diverge in many different directions.
  - Each person creates a clone of the project.
Create a Project in Git

- Download the packages for Git
  - `Sudo apt-get install git-core gitosis gitk`
    - Git-core //git itself
    - Gitosis //for hosting git repositories
    - Gitk //graphical visualization of git repository
- Create a project folder and initialize Git
  - `Mkdir myproject`
  - `Cd myproject`
  - `Git init`
List of Commands

- Help Commands
  - Git version
  - Git help <command>

- Other commands
  - `git init; git add; git status; git log; git commit -m; git diff; git show; git branch; git checkout <branch>; git mergetool; git config user.name "My Name"; git merge`
Adding Changes to the Stage

- In Git, the stage is a buffer between the changes you make in your directory and what has been committed to your project.

- If you have an existing Project:
  - `cp /path/to/my/project .`
  - `git add -A`

- If you are starting from scratch:
  - `echo ”// This is a c comment” > program.c`
  - `git add program.c`
Committing Changes to the Project

- Changes do not automatically get officially added to your project when you edit, create, or delete files. This helps protect the integrity of your current project.

- To commit changes:
  - `git commit -m "Describe what I committed"`

- To remove a file:
  - `git rm <file>`
Some tracking commands will help you to remember what you were working on and what exactly you have changed about the project.

- Find out what has yet to be added to the stage:
  - `git diff`
- Find out what will be committed from the stage:
  - `git diff --cached`
- Find out both (above)
  - `git status`
Some tracking commands will allow you to see the entire history of changes made to the project.

- List the commits that have been made:
  - `git log`

- List commits and see differences between commits:
  - `git log -p`
Branching allows you to make experimental changes or to divide up changes that need to be made to a project.

The default branch for a new project is called "master".

To create, delete, and list branches

- git branch <branchname>
- git branch -d <branchname>
- git branch
Branching Out

- Branches are completely independent of each other, but they all contain the whole project. (The project hasn't been copied. Your differences have been logged.)

- To switch between branches:
  - Git checkout <branch>

- To switch to a previous version of the project
  - Git checkout HEAD^(version #) -b <branchname>
Merging Branches

- When you make enough changes on a branch that you think should be included in your overall project, you can merge your changes into the project.

- Merge the contents of branch A into branch B
  - Git checkout A
  - Git merge B

- This can also serve to update branches to the latest changes in the main project.
Occasionally you will encounter conflicts when you merge from one branch to another. Git informs you of conflicts and puts both versions in the file that has the conflict.

Conflicts may be resolved by:

- Picking your favorite editor and opening the files
- Using "git mergetool" to see and edit all files involved.

After it has been resolved type:

- `git add <stuff>`
- `git commit`
Questions???
Linux 201 session 2 is planned for two weeks from today!! (April 11th)

Open Source Gaming LAN will be hosted on April 2nd in STEW 312 from 11:00am – 8:00pm.

”CS Events” on Facebook.

Resume Clinic on Thursday at 7:30 in B151 hosted by CSWN.