K0TO Using Git

Nikola Whallon

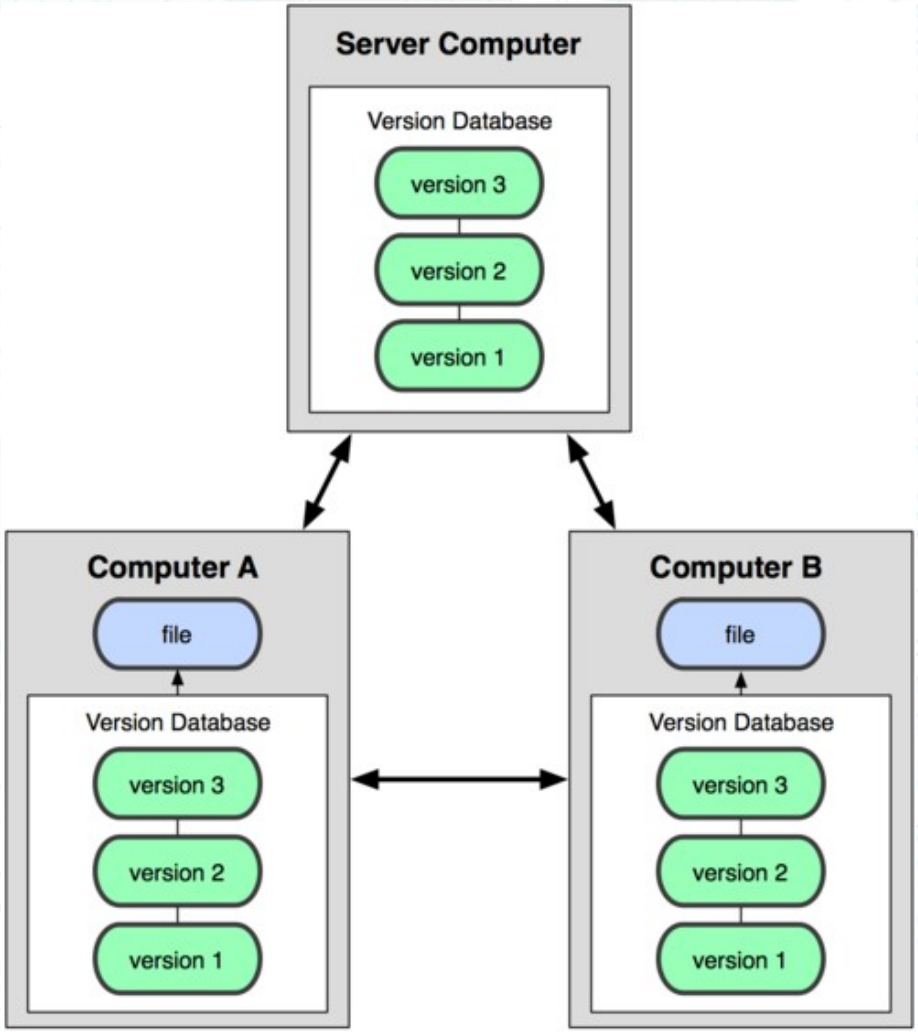
What is Git?

- Git is version control software.
- It runs on all unix-based systems, and there is also a Windows version.
- This presentation will show how to install and use git for acquiring and making changes to the Level 3 (L3) data acquisition (DAQ) code used by K0TO.
- For detailed information about git, check its documentation page at:

  http://git-scm.com/doc

Git's Method of Version Control

- Git uses a distributed version control system.
- Under this system, clients both check out the latest snapshot of files and fully mirror the repository.
- This makes using git easy to do on a local machine, simpler, and faster.



http://git-scm.com/figures/18333fig0103-tn.png

Installing Git

- The easiest way to install git is to use the following command (on Red Hat or Scientific Linux):

  yum install git-core

- You may need to install some dependencies in the following way:

  yum install curl-devel expat-devel gettext-devel openssl-devel zlib-devel

- To install git on a Mac, use the MacPorts software package manager and run:

  port install git-core

- To install git on Windows, download and run the installer found here:

  http://code.google.com/p/msysgit

Configuring Git

- Before using git, you must edit your git configuration file.
- The most basic change you must make to the configuration file is adding your user name. To do this, run the following command:

  git config --global user.name “my_username”

  (Replace “my_username” with the username you want to use.)
- You may also choose to add your email, set your default text editor, or set your default diff tool. To do these things, run the following commands (respectively):

  git config --global user.email my_email@something.com

  git config --global core.editor vim

  git config --global merge.tool vimdiff

  (Change the email, editor, and diff tool to your preference.)
- To check your settings, simply run the following command:

  git config --list

Codeblue and the K0TO L3 Repository

- The K0TO L3 code and documentation is kept in a codeblue repository.
- Codeblue is a University of Michigan software host, similar to hosts like SourceForge or GitHub.
- Anyone can register and create an account by going to the following page:

  http://codeblue.umich.edu/index/account/register
- Once registered, managers of the k0to daq repository can add you to the project.

Checking Out the Repository and Adding and Removing Files

- Once you are added to the project, you can check out the L3 repository using the following command:

  git clone https://your_user_name@codeblue.umich.edu/git/daq

- To add files or directories, run the following command:

  git add my_file_or_directory

- To remove files or directories, run the following command:

  git rm my_file_or_directory

Committing and Pushing

- Before committing your code (which will update your local repository) or pushing your code (which will update the remote repository), run the following command to check what files you will be adding, removing, or modifying:

  git status

- If you are ready to commit, run:

  git commit -m “my comment”

  (Replace “ “my comment” “ with a brief description of the changes you have made.)

- Then, if you are connected to the internet, push your updated repository to the remote master repository by running the command:

  git push origin master

Checking Version Differences

- One very useful way to check version differences is to use codeblue.
- First log in (at codeblue.umich.edu), then click on “Projects” → “daq” → “Repository”, then select the versions you would like to see the difference between, and finally click “View differences”.

Version Differences Example

/ - Diff - daq - codeblue.um

codeblue.umich.edu/index/projects/daq/repository/diff?utf8=✓&rev=980cd0ae427d74de29aa1810c47acb07845ae9ee&rev_to=9dcc2a94fe426233e32b3de4a9150b0538c5e345

13 #endif

Event.cpp

```
 7
 8  Event::Event()
 9  {
10          pktSum = 0;
11          timing = 0;
12
...
21
22  Event::~Event()
23  {
24  //      cout << "freeing Event" << endl;
25
26          delete[] data16;
27          delete[] data8;
28  }
29
30  uint32_t Event::GetPktSum()
31  {
32          return pktSum;
...
74          }
75          else
76          {
77                  cout << "error - data16 would have overflown,
    aborting AddData()" << endl;
78          }
79  }
```

```
 7
 8  Event::Event()
 9  {
10          isCut = 0;
11
12          pktSum = 0;
13          timing = 0;
14
...
23
24  Event::~Event()
25  {
26          delete[] data16;
27          delete[] data8;
28  }
29
30  uint8_t Event::GetIsCut()
31  {
32          return isCut;
33  }
34
35  uint32_t Event::GetPktSum()
36  {
37          return pktSum;
...
79          }
80          else
81          {
82                  printf("error - data16 would have overflown,
    aborting AddData()\n");
83          }
84  }
```

Now You are Using Git!

- Remember, there is a lot more to git than adding, changing, removing, committing, pulling, and pushing files – but these basic commands should be enough to get you started editing the K0TO L3 code!

Happy Coding,

Thanks!