To: Brady Augedahl, Elliot DuCharme

From: Larry Basegio

Subject: Use of Git/GitHub to historically log program changes

Date: 18 December 2018

Summary

During the Java training day of 16 December, the emphasis was not necessarily on the Java language but more on program revision control. This is something quite powerful and of great utility for our robotics efforts this year. I would strongly suggest that we place a high priority on becoming familiar with these concepts.

It allows us to solve two significant issues that we all recognized last time:

- Keeping track of program changes.
- Being able to reverse course when we have made a programming error or the system breaks (allows us to return to where things were still working).
- The ability for each of us to work independently on the program. GitHub will store the various changes and allow individuals to "check out" and modify the code, keeping track of the changes. The code is stored in a "cloud" and would be retrievable by any of us.

Also discussed and demonstrated was Microsoft VS. The instructor Tom Harron was very good at Java and with software revision control but knows less about control of hardware. The other programming mentor present was Bennet Norris (LaCrescent). Bennet recommended that we hold off on the switch to VS until the first of the year in that other teams have reported problems with the present beta version. I have copied both of you on the materials presented at the training center – take a look at them when you have time. This is "real world" stuff.

Git Install Procedure Example

I have an example that we can work with tonight to illustrate Git functionality. We should also discuss what to name our GitHub site. We can put our current robot test program under revision control.

Here are the steps to install Git on Windows:

1. Go to http://cmder.net/
2. Download Cmder.zip
3. Extract (I did it in the downloads directory). The result will be an unpacked directory cmder.
4. Copy the directory to c:\Program Files (x86).
5. Within the cmder directory, start the application Cmder.exe. A command screen will appear.
6. The directory that appears at the command prompt may need to be changed. Change the directory (if necessary) to the location c:\Program Files (x86)\cmder

7. At the command prompt type: git –version (that's a double hyphen). The response I received was "git version 2.19.0.windows.1"
8. Perform the following setup tasks (my example is shown, yours will be specific to your liking:
    - git config –global user.name larrybasegio
    - git config –global user.email larrybasegio@gmail.com
9. Change to the Eclipse project directory. Note that typing ".." at the command line will back you up one directory level, e.g., from "c:\Program Files (x86)\cmder to "c:\Program Files (x86)\". Back yourself out to "c:\" and then use "cd c:*******\ whatever to get to the directory holding the project. In my case it went like this (red is system response, black is my typing):
    - C:\Program Files (x86)\cmder
    - ..
    - C:\Program Files (x86)
    - ..
    - C:\
    - cd user\3Zman\eclipse-workspace\Complex3
    - C:\Users\3Zman\eclipse-workspace\Complex3
    - git init
    - Initialized empty Git repository in C:\Users\3Zman\eclipse-workspace\Complex3\.git
    - cd src
    - C:\Users\3Zman\eclipse-workspace\Complex3\src (master -> origin)
    - git status
    - On branch master
    - No commits yet
    - Untracked files: … (it lists them and tells you to use git add <file> to track.
    - add . (tracks all files within the src directory, and "stages" them for "commit"

OK, this is the basic install procedure. I will demo tonight to show how the commit and "back up" – not backup (literally back up to any previous version) works.

If I have time today I will attempt to generate a similar document on the use of GitHub – another application we will really want to learn about and adopt.