Help Topic: Submitting 2.680 Lab Assignments

Spring 2021

Michael Benjamin, mikerb@mit.edu
Department of Mechanical Engineering
MIT, Cambridge MA 02139

Submitting 2.680 Lab Assignments

Lab submissions for 2.680 will be done through a version control system such as SVN. Although there are other ways to do this, e.g., via the MIT Stellar uploads, or Dropbox, we prefer SVN (or Git) for a couple reasons:

- When our labs begin to involve lab partner collaboration, you will need to access each other's code.
- You will be required in this course to use a version control tool such as SVN that allows merges between two people's work and allows one to revert to prior revisions of code.
- You won't have access to DropBox on the robots. You will however be able to check out your code onto the robots with SVN or Git.

Submission Format

In this course, your lab assignments may be viewed as successive additions to the `moos-ivp-extend` tree. Here you can insert new MOOS applications, Helm behaviors, autonomy missions, and of course regular lab report write-ups or presentations.

The overall submission process will look like this:

- One time: Set up an SVN service account and upload your `moos-ivp-johndoe` tree.
- For each lab, develop and add to your `moos-ivp-johndoe` tree, and check in the changes.
- Declare to the 2.680 staff that you have finished your lab.
- 2.680 staff will grade the lab after checking out or updating your lab tree.

Step 1 (one time): Set up your Version Control Service

We recommend using an inexpensive service such as CloudForge detailed below for your version control service. You can use another service if you like, or use a *git* service such as GitHub. Mainly we care only that we can grab a latest version of your tree from the web. CloudForge and SVN just seem the easiest and it's free.

Alternatively if you would like us to set up an account for you on the MIT "oceanai" server, we can do that too. The only drawback is that you likely won't have access to this server after the end of 2.680. We also view the use of a version control service to be a good practice in general for graduate studies, not just 2.680 code, but all your valuable class projects, documents, and thesis.

(CloudForge Option) Set up your CloudForge account and set up a new project

- Create a new account on CloudForge or equivalent. See the help page on setting this up: http://oceanai.mit.edu/ivpman/help/svn_setup_your_own_repo.
- In CloudForge, once you have set up a new account, create a new project with SVN service as described in the above help link.
- Make sure you follow the above steps to the end. The end result should be that you can check out a newly created project from CloudForge, initially with nothing in it.

```
$ svn co https://YourCloudForgeAcct.svn.cloudforge.com/YourProjectName
Checked out revision 0.
$ cd YourProjectName
$ ls
(nothing)
```

(Oceanai Option) Set up an account on the oceanai server

To use oceanai, you will need to (a) have an account set up on oceanai, and (b) create a tar file of your tree and send it to be set up. The tarfile is discussed below. To initiate an account on oceanai, let us know this is your choice, and provide us with a username. This username should ideally be the same as used on your laptop and on MIT Athena.

Prepare to upload your material

- Check out the `moos-ivp-extend` tree as a starting template. This is checked out from the class oceanai.mit.edu server and you likely have done this aleady as the last step in Lab 3. Re-name it to something indicative of who you are, e.g., your Athena/MIT account name, `moos-ivp-johndoe`.
- You will need to prepare this tree for uploading to a different SVN server, the one you set up with CloudForge or similar. All SVN trees contain "meta-information" indicating where the tree was originally checked out from. You will need to remove this SVN meta information from the tree you check out from the class server. To do this, remove the `.svn/` directory in the top level of this tree. `"rm -rf ./.svn"`
- IMPORTANT: Before you upload your `moos-ivp-johndoe` tree, remove any generated files. Don't check in things that have been built as part of the build process. For example your `build/` and `lib/` directories should be *empty*. Your `bin/` directory should only have what was in it originally. Your `missions/` directory should *not* have log files. A virgin `moos-ivp-extend` tree is less then a megabyte. Your checked in version shouldn't be much bigger. You can use the `clean.sh` script in the `moos-ivp-extend` directory as a shortcut for cleaning. But as your directory evolves, this clean script may need to evolve too.
- CloudForge: Upload your `moos-ivp-johndoe` directory to your newly created project on CloudForge. If the CloudForge project name is `YourProjectName` and your local moos-ivp-extend tree is `moos-ivp-johndoe`, then just do the following:

```
$ mv moos-ivp-johndoe YourProjectName
$ cd YourProjectName
$ svn add moos-ivp-johndoe
$ svn commit -m "comment, e.g., initial setup"
```

- Oceanai: If you're using the oceanai server, the send a tar file of your directory to me (mikerb). I will initiate a new tree by this name on the oceanai server and give you access. If you haven't used `tar` before here's the quick start (assuming you are in the directory containing your moos-ivp-johndoe tree:

```
$ tar cvf moos-ivp-johndoe.tar moos-ivp-johndo
$ ls
moos-ivp-johndoe/ moos-ivp-johndoe.tar
```

To untar the file back into directory form:

```
$ tar xvf moos-ivp-johndoe.tar      (note the 'x' replaces the 'c')
```

Confirm it Works and Provide Access to 2.680 staff

- Confirm in a temporary directory that you can checkout this tree from CloudForge. After this, only use this tree.
- Take note of the SVN URL and email it to the 2.680 staff. This should only need to be done once. Subsequent TA access will come by the TA entering his/her local copy of your tree and just updating the tree from the server.

In CloudForge, you have the option of allowing access to your tree with a password, or without. Allowing access without a password is called "anonymous access". This is ok, but it seems that CloudForge does not support anonymous read-only access. So if you allow anonymous access, others can check in changes to your tree. This may be ok for the class. It's up to. If your tree *does* need a password, you will need to let the TAs know what that password is.

0.1 Step 2 (each lab): Submit your 2.680 Lab Assignment

For each lab assignment, here's what to do:

- Develop and test your code.
- Follow the requested naming conventions: If a particular MOOS module is part of the assignment, and we ask that you call it `pFooBar`, then please name if `pFooBar`.
- IMPORTANT: Make sure you have not checked in build-generated files, in `bin/`, `lib/`, or `build/` directories. Or any log files in your `missions/` directory.
- Check in your changes to the server. (Actually don't wait until your done with the lab to check in changes. This is something you should be doing regularly as a way of backing up your work.)
- Inform the 2.680 staff that you are done with you lab. Include the SVN revision number of your final check-in.