**IMPORTANT:**

**DO NOT use GUI-based git applications to source-control your work - they will break stuff and it will be very painful to clean the mess.**

**Command line git is your friend!**

**INITIAL SET UP**

1. Install git from http://git-scm.com/book/en/Getting-Started-Installing-Git

2. Go to the overview page of your repo at bitbucket.com and copy the repo HTTPS location. Clone the repo:

$ git clone https://your-repo-address

This will create a local version of the master branch on your machine.

3. Configure your personal information so that your name is displayed correctly in the repo history:

$ git config --global user.name "Rick Born"

$ git config --global user.email richard\_born@hms.harvard.edu

**TYPICAL GIT WORKFLOW**

1. Checkout your master branch and make sure it is up to date

$ git checkout master

$ git pull

2. Create a new branch that tracks master:

$ git checkout -b your\_branch\_name --track master

3. If you are reusing an existing branch, make sure to rebase it before starting a new big change (see KEEPING YOUR BRANCH UP TO DATE below)

—> do changes

4. See what files are modified

$ git status

5. If you want to discard certain changes do:

$ git checkout -- file\_to\_undo

6. Stage files to be committed:

$ git add file\_name1

$ git add file\_name2

7. Commit files, provide commit message:

$ git commit -m “Commit message”

—> if needed, do more changes and create another commit(s)

8. Make sure the branch is up to date with the master before pushing your branch out (see KEEPING YOUR BRANCH UP TO DATE below)

9. Push your local branch into the remote repo (either to review and merge into master, or just as a means of backup)

$ git push origin your\_branch\_name

**KEEPING YOUR BRANCH UP TO DATE (MERGING)**

Make sure to do this regularly to minimize conflict resolution efforts.

1. Update your local copy of master branch

$ git checkout master

$ git pull

2. Merge master with your working branch (you will need to make sure there are no un-staged, un-tracked and un-committed files in your branch, otherwise you won’t be allowed)

$ git checkout my\_dev\_branch

$ git merge master

3. If conflicts occur, resolve them, save affected files then do:

$ git add name\_of\_file\_where\_conflict\_was

$ git commit –m “Merging: message here.”

**OTHER USEFUL COMMANDS**

0. Checkout out a remote branch from the repository

$ git fetch

$ git checkout remote\_branch\_name

NB: on orchestra it wasn’t necessary to use $ git fetch. The checkout alone works.

1. Switch between different branches (great when you need to work on multiple things at once):

$ git checkout my\_other\_branch\_name

2. See what commits you have in the checked out branch:

$ git log

3. See all branches you have:

$ git branch

4. Delete a local branch you no longer need

$ git branch -D not\_needed\_branch

5. A handy visual tool to see the diff of your changes and to see a branch/commit tree

$ gitk

6. Compare files, commits or branches

$ git diff # compare local and remote files

$ git diff dev\_branch master # compare dev\_branch with master

**PULL REQUESTS (PR)**

All changes need to be reviewed by your peers before merging into master (or a feature branch).

SUBMITTER:

1. After the development branch is pushed to the remote repo as described above, go to the repo in bitbucket.com

2. Click Pull Requests icon in the navigation menu on the left-hand side

3. You will see all currently open, merged and declines PRs listed there

4. To create a new PR, click “Create pull request”

5. Select what branch you want to merge into master (the branch that you just pushed with the latest changes)

6. Edit description of your change so that reviewers know what you did

7. Specify reviewers if you want to, they will get notified

8. Click “Create pull request”

9. If reviewers request changes, fix your files then repeat steps 6-9 from the TYPICAL GIT WORKFLOW to create another commit and push an updated branch out for another round of review

REVIEWER:

1. Open the list of current PRs

2. Click the one to review

3. On the next page you will see the change set that the submitter wants to merge into the master

4. Review the work and either Approve or leave comments for the submitter to address

5. To check out somebody else’s remote branch on your machine:

a. Update your local master:

$ git checkout master

$ git pull

b. Checkout the remote branch

$ git checkout origin/remote\_branch\_name

6. Once the patch is approved, merge the work

7. If the auto-merge is impossible due to conflicts request the submitter to rebase and push their branch again

MISC:

1. To disregard an erroneous/unwanted PR, click Decline and it will be removed from the Open PR collection

2. Once you’ve merged a branch into master and you have no intention of going back and working with the branch further, feel free to delete it from the repo to keep your branches manageable.

**OTHER USEFUL RESOURCES**

1. For more in-depth understanding of git read:

http://git-scm.com/book

2. To figure out how to do something, you can rely on stackoverflow.com. It is very possible that your question was already asked and answered there.