Git and MATLAB

1. Create a repository on GitHub (When you want to initiate a project)
2. From top right of page, click + and choose new repository.
3. Choose owner and repository name and add description (optional).
4. Choose private for private repository.
5. For a new repository, check the README box.
6. Other options can be skipped for now.
7. Clone the repository on MATLAB (copy the master to your own computer)
8. Create folder where repository will live on your computer.
9. From within MATLAB, right-click on the empty space in the folder list.
10. Choose Source control > Manage files….
11. In the dialog box that opens, choose Git from the dropdown on the first line (Source control integration).
12. On the third line (Sandbox), enter the path of the repository folder.
13. From the repository page on GitHub, click the green button (Clone or download) and copy the URL.
14. Paste the URL into the second line of the MATLAB dialog box (Repository path) and click the Retrieve button.
15. Enter GitHub user and password when they’re requested.
16. Repository files (only README for a new repository) should now appear in the file list with a green circle on the right side.
17. Add files to repository
18. Create files in the repository path (local folder) or copy them from somewhere else. New files will now be listed under the folder with an open circle on the right side.
19. Add new files to source control
20. Mouse over the open circle next to a file to see that this means the file is not in source control.
21. Right click on file and click Source control > Add to Git. File will now have a plus on the right side.
22. Commit files
23. Right click on file and click Source control > View and commit changes….
24. Enter comment and click commit. File is now committed to **local** repository and has a green circle next to it.
25. Push files
26. Right click on file and click Source control > Push.
27. Enter username and password. If there is no moderation, file will now appear in the repository on GitHub.
28. Branches and pull requests
29. If the repository is moderated, push will not work. (Error: The remote update was rejected by the target.) Changes need to be pushed to a new branch.
30. Right click on file and click Source control > Branches.
31. At the bottom of the dialog box, click Branch and Tag Creation.
32. Click branch, give the branch a name and click create.
33. In the Branches dropdown, choose your new branch and click Switch.
34. Close the dialog box and push the file again. This time it should work.
35. Go to the repository on GitHub. Verify that the changes are there by choosing the new branch in the dropdown above the file list.
36. Click the New pull request button (next to the branch dropdown).
37. Choose new branch as compare (base is master). Enter comment and click Create pull request. Now owner can merge branch with master after review.
38. Updating user environment
39. Before starting new work, make sure repository is synced with the GitHub repository. This is accomplished by using the Fetch and Merge commands or the Pull command. In order to use both Merge and Pull from MATLAB, command line Git must be installed. Go to <https://git-scm.com/downloads> and download and install the Git that’s appropriate for your OS.
40. [Create .gitattributes file as detailed at <https://www.mathworks.com/help/matlab/matlab_prog/set-up-git-source-control.html#buhx2d6-1_3>. Nomi will put this file in each repository.]
41. Once Git is installed, open (or reopen) MATLAB.
42. From the Source Control menu (right click on folder space, choose Source Control), click Pull. New files and changes will be merged into your repository.
43. If there are changes that might create a conflict, it is better to Fetch and then Merge manually. In this case, choose Fetch instead of Pull on the Source Control menu.
44. Changes can then be inspected and Merged as appropriate (between master and origin/master): From the Source Control menu, choose Branches. In the Branches dropdown, choose origin/master and click Merge. If there are conflicts, an error message will pop up. Files with conflicts will now be visible in the file list with a red circle with an exclamation point. To resolve conflicts, right click on file and choose Source Control>View Conflicts. A window will pop up with the differences highlighted. Click the line number next to the preferred version to take it. Alternatively, open the file and edit it manually. Make sure to delete the “conflict markers” such as >>>>>>> and ========.
45. After saving the file, choose Source Control>Mark Conflict Resolved. Then commit changes, push to new branch, and submit pull request.

Useful links

Git command line initial setup:

<https://help.github.com/articles/setting-your-username-in-git/>

<https://help.github.com/articles/setting-your-commit-email-address-in-git/>

Create GitHub repository from existing project:

<https://help.github.com/articles/adding-an-existing-project-to-github-using-the-command-line/>

MathWorks documentation of MATLAB source control:

<https://www.mathworks.com/help/matlab/source-control.html>

Stack Overflow forum discussion on overwriting local repository with remote:

<https://stackoverflow.com/questions/4157189/git-pull-while-ignoring-local-changes>

Git commands:

<https://confluence.atlassian.com/bitbucketserver/basic-git-commands-776639767.html>

Useful Git commands

git status: shows which branch you’re on and files that need to be added/committed

git diff <filename>: shows file changes since last commit

git merge master: from branch, merges changes pulled to master into branch