Git User Quick Reference Documentation

What is Git?

Git is a distributed revision control and source code management system with an emphasis on speed. If you have used SVN, CVS, or any other revision control system this shouldn’t feel out of place.

The Programs we use:

There are several different Programs available to use for Git. Of those we use one of two programs. Github.com’s Github, or TortoiseGit. Each software requires other software to also be installed. For instance, with Github, you need to have a difference and merger program, such as Kdiff3. With TortoiseGit you will need Git for Windows (aka msysgit). Please see links below:

Github: <https://github.com/>

Kdiff3: <http://kdiff3.sourceforge.net/>

Tortoisegit: <https://tortoisegit.org/>

Msysgit: <http://msysgit.github.io/>

Git Hub Procedures:

Branches:

The Master branch is the main branch, anything that is placed on QA shall be on the Master branch. Any additional branches shall be for:

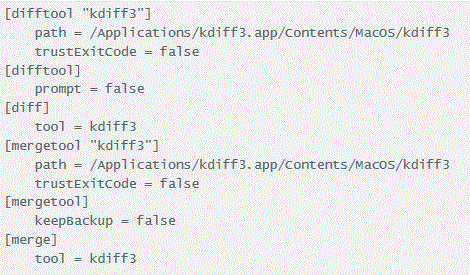
1. Personal development, items you want to test, but may not be used in the final build.   
2. Items that are not effecting the current sprint, but will eventually be merged into the Master Branch.  
3. Items that are temporary but are currently needed, such as a temporary change to a stored procedure.  
4. Resolving issues of overwriting. (ie. someone always overwrites everything, better to each have your own branch and merge into the master to minimize issues) You will have to work together to resolve any conflicts that happen when you merge together.

When to Commit:

Commit at the end of every day or when we have finished a story or made a correction or update to a story that is currently in QA.

Quick How to:

Installation is covered on the website links above. The only installation advice I have is if you are using Github and kdiff3 (or other merge tool with Github). You need to define the merge tool in your .getconfig. A hiddenfile located in C:\Users\username\ in windows 7, Vista or C:\Documents and Settings\username in Windows Xp. The information you need is :

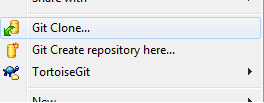


Path is where is installed your merge/diff program is installed, and the tool is the merge/diff tool you are using.

Tortoisegit quick start. Assuming the repository is created:

Getting a Repository:

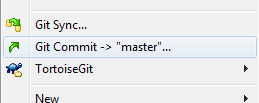
1. Right click in a folder where you want the repository to go. Select Get Clone.



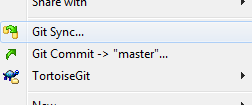
1. Enter in the URL, the putty key (if needed) and click ok. Your Repository will be created.

Committing and Pushing:

1. Click either on the folder, or inside the folder where your updates are.
2. Right Click and select Git Commit -> Master (ie appropriate Branch name)



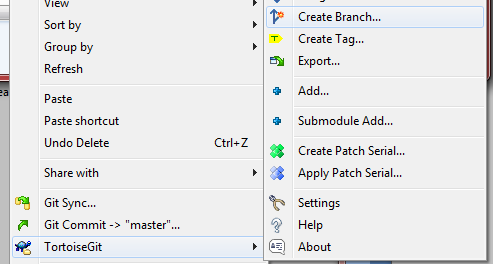
1. Enter in your message; make sure ALL files you want committed are checked. Note: Commit on the top most level otherwise not all files will be seen. (ie folders structure /floo/bob/gee and /floo/bob/go. If you commit at gee, then any files under /floo/bob/go will not commit, so always commit at /floo so all files under that directory and in sub directories will commit!)
2. Now you can either Push, or hit close. Let us assume you’re going to write some more code and commit a few more times, so hit close.
3. Now once you have gotten all the stuff committed, right click and select Sync



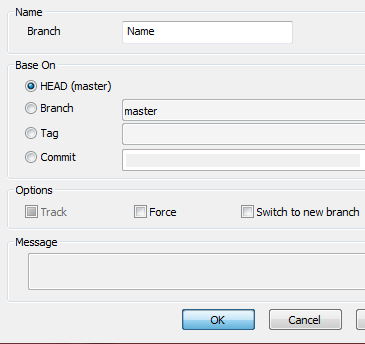
1. Please note: if you’re behind in pushes to your current branch you will want to pull, then push. For now, lets just hit Push. It will then ask you for your username, followed by your password. After you have entered those, you’re done.

Creating a new Branch:

1. Right Click in the top folder, select TortoiseGit, then Create Branch

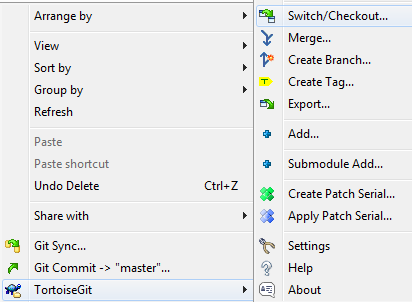


1. Enter in an appropriate name, and select from which branch/commit of current branch you want to use then click Ok.

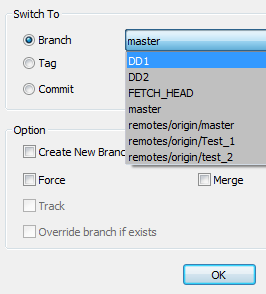


Switch Branches:

1. Right Click in the top folder, select TortoiseGit, then Switch/Checkout



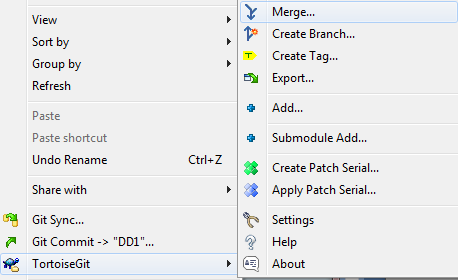
1. Select the Branch from the drop down menu. Click ok.



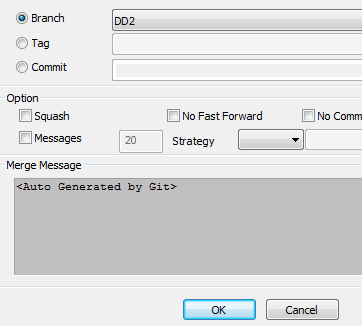
1. To ensure you always have the latest version after switching branches, click PULL.

Merging Branches/Conflict:

1. Switch to the Branch you want to merge to.
2. Right Click in the top folder, select TortoiseGit, then Merge



1. Select the Branch/commit you want Merge From. Click Ok.

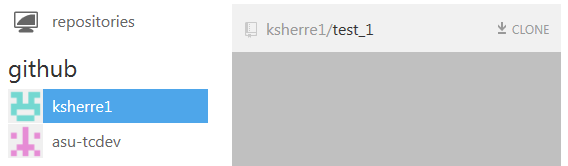


1. If all goes well, just push, if not click Resolve.
2. Double click on the file(s) in red to resolve. Once resolved, right click on the file and select Resolve. Click Yes. Then click ok. Now click Push to push the now merged branch to the repository.

Github quick start. Assuming the repository is created.

Getting a Repository:

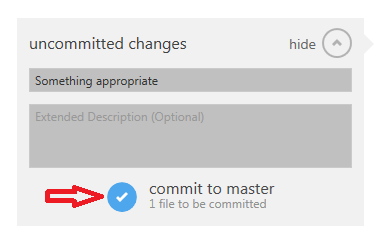
1. Under Github, select the repository User, then select the repository and click clone.



1. To select your newly created Repository, click on repositories, then click on the the 🡪 to go to your newly created repository.

Committing and Pushing:

1. In Github on the bottom put an appropriate message then click Commit to “Branch”

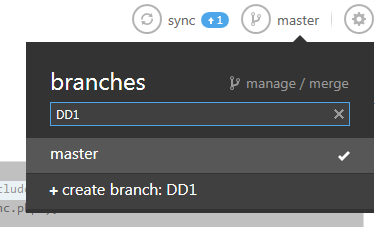


1. Now you can Sync if you want to push the changes to the repository. Lets assume you’ve written some more code and committed a few more times. Simply click Sync and Github will Pull, then push from/to the repository.

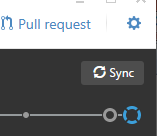


Creating a new Branch:

1. Make sure you are on the branch in the place you want to create a new branch from. Now click on the name of the current Branch.
2. In the Box, fill in an appropriate name of a new Branch. Hit Enter and your new branch will be created. You will also automatically be switch over to the branch

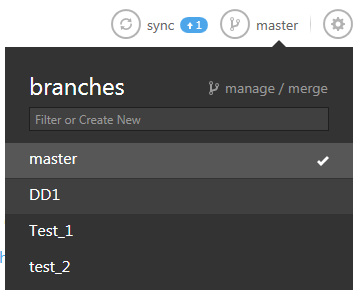


1. Note: You can commit to this branch right now, BUT until you click sync the branch is **NOT** pushed out to the repository. Click sync to push the branch to the repository.



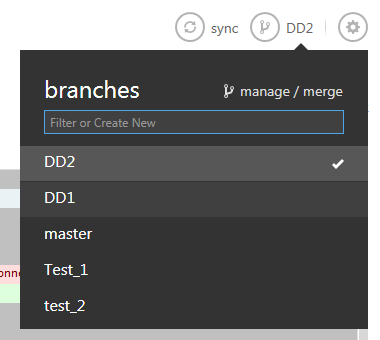
Switch Branches:

1. Click on the current Branch name, then click on the Branch Name you want to go to.

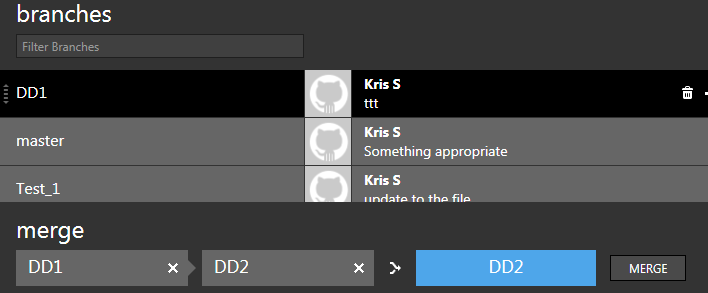


Merging Branches/Conflict:

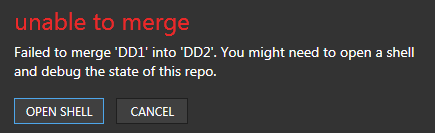
1. Click on the current Branch name. Then click on manage/merge



1. Drag the First Branch(the From Branch) to the left most square, then move the second branch (the to branch) to the square just to the right, then click Merge.



1. If all is successful, your done, if not you will get a pop up telling you it was unable to merge.



1. There could be several issues but most likely there is a conflict, no matter the possible issue, click OPEN SHELL.
2. Your command prompt should open up. Type git mergetool And solve your conflict.



1. Once you have solved your conflict type git commit –m “your message” Then type git push. Your done.



Github Command line Commands:

All commands start with git

mergetool - > Does a merge on conflicted documents

push -> Pushes commits into a repository

pull -> Pulls commits from a repository

status -> Show the working tree status

rebase -> Forward-local commits to the updated upstream head

log -> show commit logs

diff -> show changes between commits, commit and working tree, etc.

help -> show a list of commands

There are many, many more commands, if you want to view more of them these are two great sites to look at:

<http://www.siteground.com/tutorials/git/commands.htm>

<https://www.kernel.org/pub/software/scm/git/docs/>