Max's Git CheatSheet

Wednesday, November 7, 2018 9:35 AM

Changes with a commit:
Start a story
Pull latest changes from release branch (git pull)
Create a branch from the release branch (git checkout –b release/feature/feature-name-dashed) – the –b means create a new branch
Make changes to code
Check files changed (git status) – red means changes made but not added
When done, add your changes (git add –p [to review each change before adding] git add . [to add all changes])
Check files changed again (git status) – green means changes made and files are added
Commit the changes, this will save a point in time for the code on your *local* machine (git commit –m 'add commit message here')
Check status again (git status) – files should be missing, but it should say that you are now one commit ahead of the branch
Push the changes to the remote repository (git push origin head)

Ready for merge to release branch:
When ready to merge your changes into the release branch:
Checkout the release branch (git checkout release-branch-name)
Pull the latest changes (git pull)
Checkout the branch you want to merge (git checkout release/feature/feature-name-dashed)
If the branch is complete and **will not** be used again, rebase the release branch into the feature branch (git rebase release-branch-name)
    Else if the branch is not complete or will be used again, merge the release branch into the feature branch (git merge release-branch-name)
        Else if you don't know, merge the changes and do not rebase them. It is the safer option
Fix any merge/rebase conflicts (go into VS and click on Team tab→Click on Changes→Click on Conflicts→Click on the file with conflict and fix like with TFS)
After all conflicts are resolved, continue the rebase or merge (git rebase --continue || git merge --continue)
When complete, push the changes
    If rebasing (git push -f origin head)
        Else if merging (git push origin head)
Switch to the release branch (git checkout release-branch-name)
Now you can merge/Pull Request your code to the release branch (git merge release/feature/feature-name-dashed **OR** create a pull request in TFS)
Notes:
NEVER force push (git push -f) if the branch is being used by or has been pulled by other teammates. It will corrupt history and create conflicts
If you ever really screw up reflog and reset an old point in time (git reflog to see where to go back to and git reset --hard <SHA number>)
IF you need to get rid of all local changes (git reset --hard HEAD) or if you just want to match whatever remote has (git reset --hard remote/name-of-branch)

Commands Sheet:
git status - check the status of the working directory
git add - add files to working shelf
git add -p - Check it change before adding. You can add some changes in a file and not others, etc.

Typicall what we use
git commit - commit the files, creating a reference point and allowing you to push changes to remote
git push - push to specified branch (typically it is the head of your current branch, ie. git push origin head)
git log - check out history of commits
git log --oneline ~5 - check shorthand version of last 5 commits
git reset - reset your files to specified pointer
git reset --hard HEAD - undo local commit changes (this is if you want to remove local commits)
git checkout . - undo all local changes (use this to get back to where you where before making changes)
git checkout filename - undo changes made to this specific file(s)
git checkout branch-name - Change to another branch
git rebase name-of-branch-to-rebase-from - Merge and maintain history of the branch you are rebasing from (this will change history and make you force push, so do not use this on branches shared with others or which will not be deleted)
git rebase -i HEAD~5 - Interactive rebase getting the last 5 commits from branch. (This allows you to squash mulitlple commits into a single commit. This is good for getting rid of Works In Progress (WIPS).)
git merge name-of-branch-to-merge-from - Merge one branch into the other
git rebase/merge --continue - continue the rebase/merge
git rebase/merge --abort - Abort the rebase/merge
git reflog - See all previous commands and so you can see which you need to reset to if something goes wrong
git diff - See the diff in files
git c