# Git Cheat Sheet

**I’m about to start working on ticket sd-123, which is part of sprint-101, how do I create a fresh branch to work from?**

$ git checkout sprint-101 ( switch your working directory to be the sprint-101 branch )

$ git pull ( make sure your local copy is up to date )

$ git status ( make sure your copy is clean )

$ git checkout –b sd-123 ( -b means “branch” )

NOTE: sprint-101 is now a clone of sd-123. If sprint-101 doesn’t yet exist for this sprint, it needs to be created (see the next scenario)

**I need to create a branch for sprint-101**

$ git checkout master ( switch your working directory to match the current dev branch )

$ git pull ( make sure your local copy is up to date )

$ git status ( make sure your copy is clean )

$ git checkout –b sprint-101 ( -b means “branch” )

$ git push origin sprint-101 (put it up on the server, so other people will be able to use it too)

**I’ve finished ticket sd-123, and I want to push my changes to the server**

$ git status ( make sure the list of changes doesn’t include unintentional stuff )

$ git diff ( look over your code, do a quick PR, make sure you are happy with it all )

$ git add . ( you only need to do this if there are new files to add )

$ git commit –am ‘sd-123 some notes about what that ticket was about’

$ git push origin sd-123 ( watch to make sure it doesn’t fail )

If it failed … you need to get the latest changes out on the server (someone else added stuff)

$ git pull origin sd-123 ( that merges the changes on the server with your local copy )

If there were conflicts, you need to resolve them.

If there were no conflicts (or once you resolve them), you can push your copy up again.

$ git push origin sd-123

Now, you can go into Jira, open up your ticket, and add a comment saying “GIT Branch: sd-123” … this is especially helpful if the branch name is not a perfect match to the ticket name for some reason. It would also be nice to have that comment tell them what branch you branched from (especially if it wasn’t the sprint’s main branch).

**Merge**

$ git status (Make sure it’s clean)

$ git checkout master

$ git merge sd-123

$ git push

Make sure to go back to a new branch instead of staying on master.

**I got a Merge Conflict!!! How do I handle that?**

$ git mergetool ( this opens up a file diff, and you use it to do the merge )

Once you like it, commit the result

$ git commit –am ‘sd-123 fixed merge conflict’

$ git push origin sd-123 ( might as well update the server with your merge fixes )

**I’ve been asked to PR ticket sd-123 for someone**

Go into Jira, and get the branch number for the ticket out of the comments (it’s probably ‘sd-123’ but if it isn’t, there should be a note … it should also tell you what they branched off of, which is the branch you are going to compare against)

Familiarize yourself with the Jira ticket. Make sure you understand what the goal of the ticket was BEFORE you PR the code that theoretically fixes the problem.

Before you can switch to checkout their code, you need to make sure your working directory is nice and clean. So, commit (or stash) whatever you are currently working on. You don’t have to push it all the way to the server, but you do need to at least commit it locally so you don’t mix things up.

$ git pull ( make sure you have the latest branch info from the server )

$ git checkout sprint-101 ( this is the branch they branched FROM )

$ git pull origin sprint-101 ( you want the most up to date copy of the branch )

$ git checkout sd-123 ( this is the branch for the ticket you want to PR, if you get an error, tell them they forgot to commit their branch )

$ git merge sprint-101 ( update their branch to have any new changes from the main branch, if there are merge issues, feel free to fix them if they are simple, or tell the other guy to fix them if they are involved)

$ git diff sprint-101 ( you want to see all of the differences between sprint-101 and sd-123 )

Once you feel like the code itself makes sense, and should address the original problem, and that there are unit tests that verify it really does work, then you need to run it and visually verify that the changes for the ticket seem to be doing what we say they do.

If you find problems, or have questions about what they did, or recommendations about how it should be tweaked before you are comfortable with the changes … put notes in comments on Jira, and send the ticket back to the person working on it.

Once you have PR’d and then QA’d the ticket, and you are comfortable with the changes, make a comment on the ticket saying you did the PR and you are happy with everything that was done. And merge their ticket into the sprint branch:

$ git checkout sprint-101 ( switch over to the sprint branch )

$ git merge sd-123 ( add their changes to the sprint )

$ git push origin sprint-101 ( update the server copy of the branch to have their changes )

You are done, Feel free to switch back to wherever else you need to work on.

**I did Git Status, and there are unexpected changes**

$ git diff ( look through the diffs, do you want to save them, or discard them? )

If you want to save them:

$ git reset --hard ( that’s two dashes, and it rolls you back to the last local commit )

$ git status (make sure it’s good and clean now)

If you want to keep them:

$ git checkout <the branch you want to save them under>

$ git commit –am ‘SD-<ticketnumber> some comments about what you are doing’

**I want to rollback one file to the last local commit**

$ git checkout – file/name ( that’s two dashes )

**I want to rollback one file to the way it was on branch XYZ**

$ git checkout XYZ – file/name ( this gets the file from the other branch, so you can restore it even if you already committed your changes on your current branch )

# Branch names:

**master** – this is the branch that contains code that is considered good enough to move to production next time we do a deployment. As sprint branches are completed, they get tested out on a dev server, and then merged into master.

**deploy-yyyy-mm-dd** – when we go to do a deployment, we branch off of master creating a deploy branch with the current date as part of the branch name. We deploy that to a staging server. And make sure we love it. When we are done, we check out that branch on the prod server and deploy it.

**sprint-xyz** – in jira, each sprint is represented by a ticket number such as SD-101. That ticket number should be used as the XYZ part of the sprint branch name … so … sprint-101.

**sd-xyz** – in jira, sprint tickets have SUB-tickets for each of the individual tasks that need to be done. The ticket number of that sub ticket (for example jira ticket SD-3112) is used to make a branch number like sd-3112.