Module 6 - Git Collaboration

GT-WU-A-06 - Collaboration

This exercise assumes completion of previous lab exercises. If you have not completed the previous lab exercises ensure you are working in an environment with the following :

- At least one git repository - populated with content
- A central storage location for all repositories

By the end of this lab you should be able to:

- Clone a repository
- Create or Identify a remote
- Pull changes
- Push changes
- Create a bare repository

Step 1 - Clone a Repository and Identify the Remote

Putting aside your work in previous modules we are now going to simulate sharing that work with other users.

- Begin by cloning your existing repository "myrepo" into a new location - name this "myclone"
    - *cd c:/git_repos*
    - *git clone myrepo myclone*
- From inside your new clone, use the *git remote* command
    - *cd myclone*
    - *git remote*
- As this is a clone we can see a remote named origin. Can you find out the location this remote is pointing to?
    - *git remote -v*

Step 2 - Make Some Changes and Pull Them

- navigate back into the original repository
    - *cd ../myrepo*
- Use what you know to make some changes on the master branch either by adding, modifying or deleting files.
- Commit those changes - use a meaningful commit message, and take a note of the hash-id.
- Navigate back to "myclone" and use the *git pull* command to retrieve your changes from the origin remote
    - *git pull origin master*
    - You could also use just *git pull* here but it is good idea to get used to specifying the branch.
    - Remember that by pulling rather than fetching we are automatically performing a merge.
- Verify your working directory contains the changes you made in the original repository.
    - check the hash id with *git log -1*

Step 3 - Make Some Changes and Push Them

- Make some changes in your "myclone" either by adding, modifying or deleting files.
    - make sure you are on the master branch when you do this.
- Commit those changes - use a meaningful commit message.
- Send those changes to your new repository with the *git push* command
    - *git push origin master*
    - You could also use just *git push* here but it is good idea to get used to specifying the branch.
- You'll notice this command fails.
    - The error you see is because you cannot push changes a branch that is checked out in a remote repository - in case you force someone to overwrite local changes in their working directory
    - To solve this, you'll need to switch branches in your original "myrepo"
        - *cd c:/git_repos/myrepo*
        - *git branch newbranch*

        - *git checkout newbranch*
- Back in "myclone", try to push the changes again with *git push origin master*
    - *cd c:/git_repos/myclone*
    - *git push origin master*
- Check the changes now exist in your original "myrepo"
    - Don't forget to checkout the master branch!
        - *cd c:/git_repos/myrepo*
        - *git checkout master*

Going Further - Optional Steps

Step one - Make some changes in "myrepo"

Use what you have learned to complete the following steps:

- Make changes in "myrepo" which include at least three commits.
- These changes should be performed on the master branch
- This must include the creation of three new files (and any other changes you like)
    - file1.txt - with the contents "This file was created on the server"
    - file2.txt - with the contents "This file was created on the server"
    - file3.txt - with the contents "This file was created on the server"

Step 2 - Make some conflicting changes in "myclone"

- Make changes in "myclone" which include at least three commits.
- These changes should be performed on the master branch
- This must include the creation of three new files (and any other changes you like)
    - file1.txt - with the contents "This file was created in the clone"
    - file2.txt - with the contents "This file was created in the clone"
    - file3.txt - with the contents "This file was created in the clone"

Step 3 - Push Changes

- Push the changes from the master branch in "myclone" to the master branch in "myrepo"
    - This push will be rejected - there are changes on the server which you do not have in your clone yet.

Step 4 - Merge between Repositories

- Before you can push, you must always pull the latest changes from the server.
    - You can either pull these changes, or fetch and merge them.
- Either way, the merge will result in a conflict.
- Use git status to to identify the conflicted files and resolve them
- There may be three of more conflicts, depending ont he changes you made in each repository.
- Once resolved, complete the merge with *git commit*

Step 5 - Push changes

- Now you have merged the changes, you are free to share your work with others
- Push your work, including the merge commit back to "myrepo"
- remember that this will fail if the master branch is still checked out.

You have reached the end of this module and should now move onto module 7