**Lab 3: Working with Git**

The aim of this lab is to give you a chance to work with Git and Git Hub.

# Step 1: Git Client

In this first step we will first confirm that you have the git command available.

To do this open a Command Window on a Windows PC or a Terminal on a Mac or Linux box. Once you have done this at the prompt enter the command

> git --version

If you have the Git client installed then you should see the version number of the git client displayed, for example:

A screenshot of a cell phone screen with text

Description automatically generated

In this case the version of Git installed is version 2.2.5.0 for Windows.

If you do not have the Git client installed, then you will be told that the command cannot be found. In this case you need to install the Git client.

You can do this by going to the following web site and downloading the correct version for your operating system:

1. <https://git-scm.com/downloads>

That is, you need to select one of:

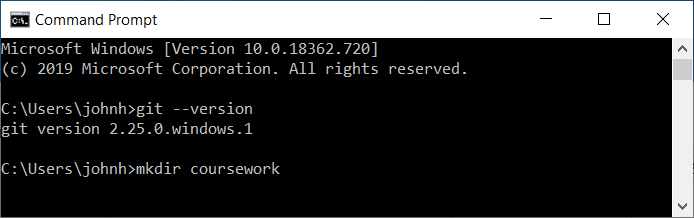
A picture containing shirt

Description automatically generated

# Step 2: Create a directory to work in

We will handle working with our labs using a specific directory. If you have not already done so, then create a directory for course work called something like coursework. On a Mac you might put this under /Users/<username> on Windows PC this might be in c:\Users\<username>.

For example:



Now change directory into this newly created directory, for example use

> cd coursework

# Step 3: Git Clone a Repository

We will now *clone* an existing Git project from a GitHub hosted repository. Remember to work with an existing repository in Git, you need to obtain a clone (copy) of that repository or project locally. This will copy the report project locally to your machine and allow you to make changes to that project.

A picture containing table

Description automatically generated

To do this you can issue the git clone command from within a Terminal or Command Window with an appropriate git project web URL.

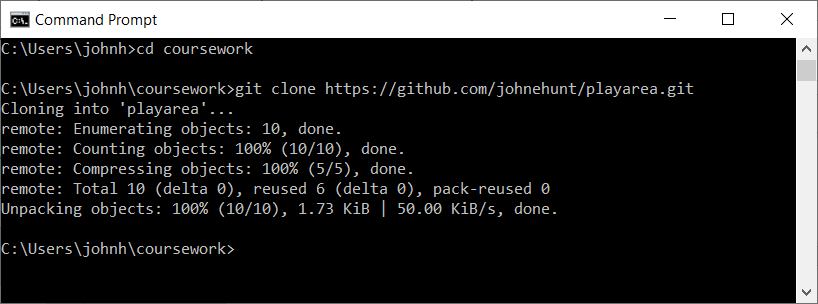
Return to your Terminal / Command Window and ensure that you have moved into the coursework directory (using cd coursework). For example:

> cd coursework

Next issue the git clone command as follows:

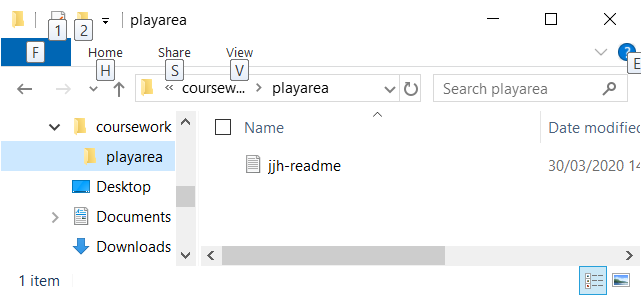
> git clone https://github.com/johnehunt/playarea.git

This will clone to your machine a copy of the playarea project from the johnehunt GitHub repository.



This repository contains a single file, called jjh-readme.txt, as shown below.

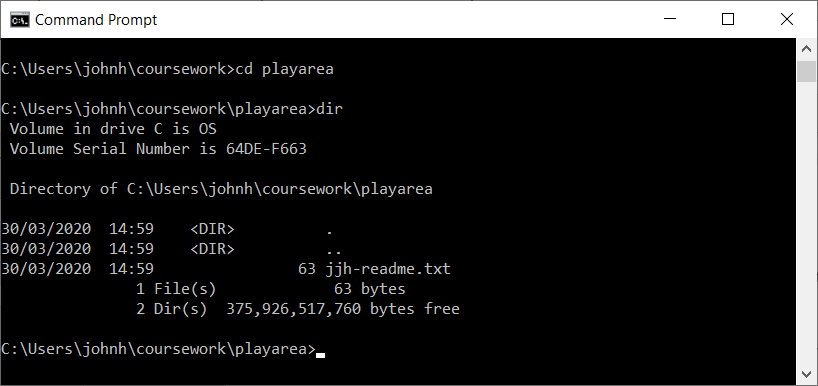
You should now change directory into the playarea directory:



# Step 4: Git History

We can now find out the commit history of the one file in the playarea. To do this return to your command window / terminal.

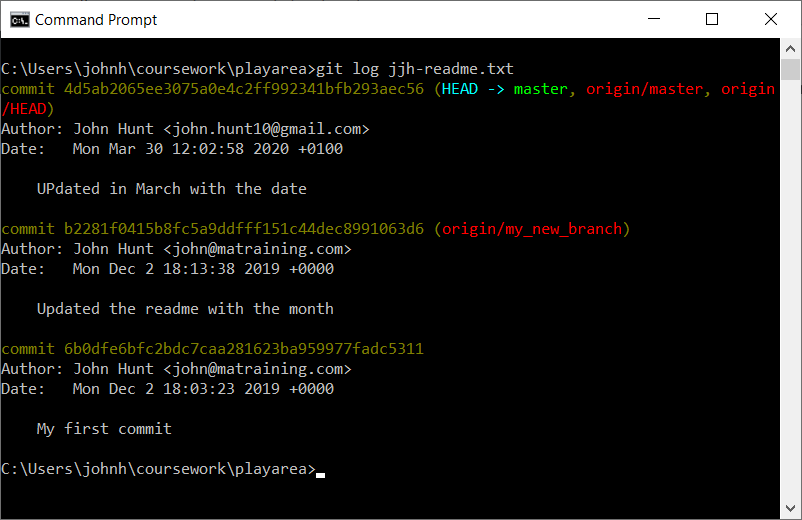
Make sure you are in the playarea directory for example by cd’ing into playarea:



Now issue the following git command:

> git log jjh-readme.txt

for example:



Now try some of these commands to see the various types of log information available:

> git show HEAD -- jjh-readme.txt

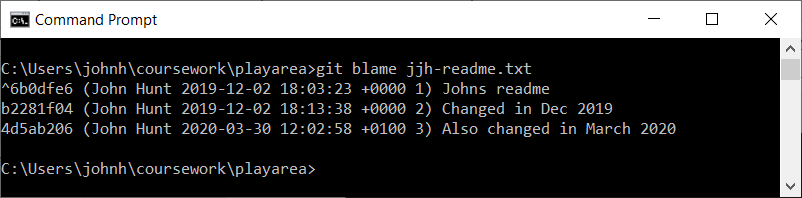
This will show the current HEAD version of the file; the HEAD represents the current latest version.

> git log -p jjh-readme.txt

And finally

> git blame jjh-readme.txt

In this last case the output is:



# Step 5. Create a branch

When working with Git it is usually to work in a development *branch*. That is all changes are made to a branch and then this branch can be merged back into the main *master* branch via a Pull Request (or PR). This process is illustrated below:

A screenshot of a cell phone

Description automatically generated

In the next few steps we will work through this process.

The first thing we will do is to create a new branch to work in. You can use the following commands to create a branch and change into that branch:

git branch <new-branch-name>

git checkout <new-branch-name>

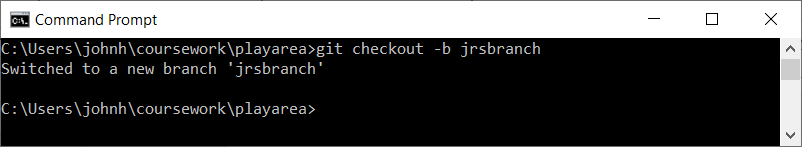
There is a shorthand from for this which is

> git checkout –b <new-branch-name>

This creates the new branch and switches to that branch.

Any changes you now make are applied to the branch you created.

We will now create a branch; - name the branch after yourself, for example, if you are John Reginald Smith then call your branch jrsbranch, for example:

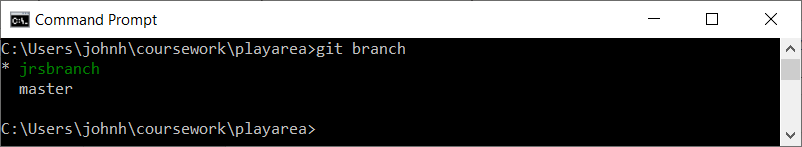


The current branch is now the development branch you created.

You can obtain a list of call branches using the command

> git branch

For example:



# Step 6: Adding a file

We will now create a file and add it to current branch. The steps we will follow are illustrated below:

A screenshot of a social media post

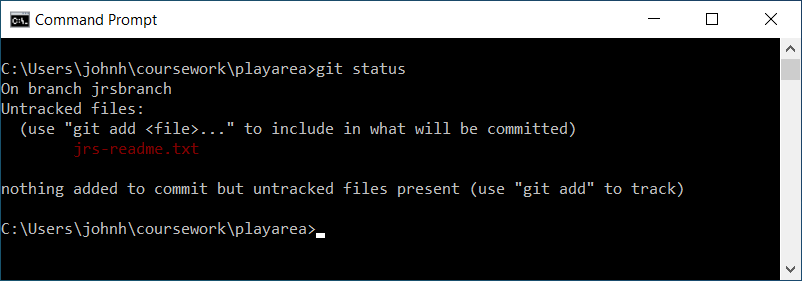
Description automatically generated

First of all, create a file in your branch again named after yourself, for example jrs-readme.txt within the playarea directory. You can do this in any way you wish.

Once you have created the file use the git command to see the status of your branch:

> git status

For example:

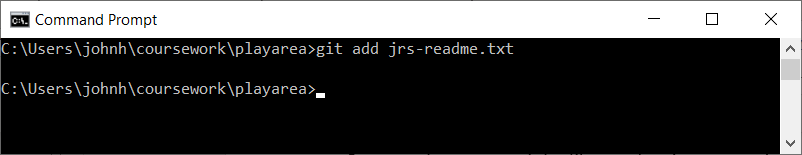


From this you can see that there is one new file but it is currently not being tracked by Git (i.e. Git knows nothing about it).

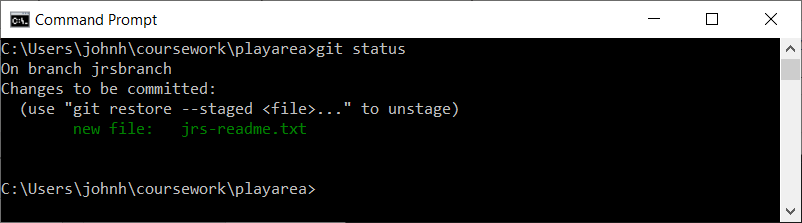
We will now add the file to the branch. To do this you use the git add command:

> git add <filename>

For example:



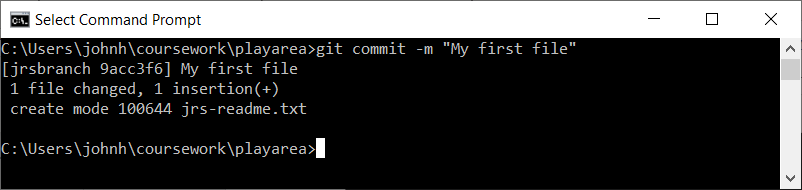
Once you have done this re-run the git status command, you should now see:



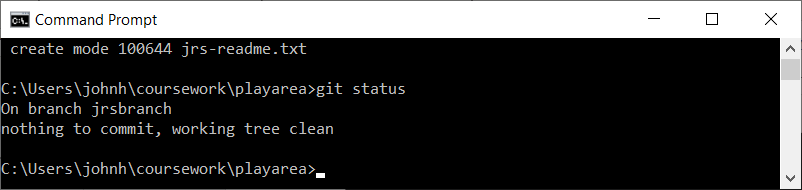
Notice that Git now knows about the new file; but the changes have not yet been committed (added to the branch).

# Step 8: Commit changes

Next we will commit the changes we have just made to the branch. To do this use the git commit command. This command needs a description for the commit you are making, we can provide this using the -m option, for example:



Now use git status to see what the status is:



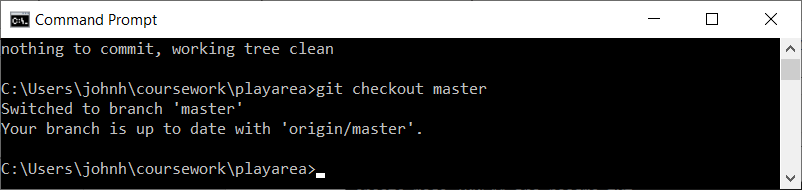
# Step 8: Switching branches

Next we will see the effect of switch between the master branch and the branch you have been working in.

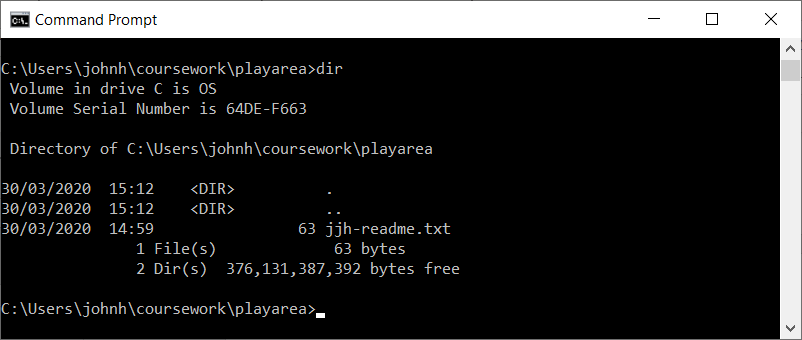
From the command window / Terminal enter the command

> git checkout master

This will switch the current branch back to the master branch.



Now look at the contents of your directory. Can you see the file you created?



It’s not there; why not?

Because it is not part of the master branch.

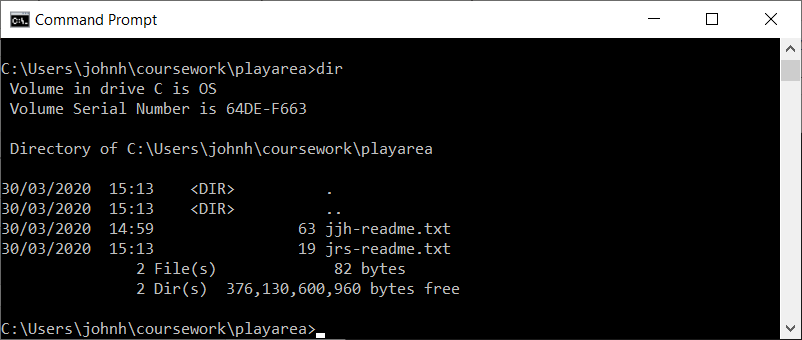
Next issue the git status command; what do you see?

Finally switch back to the branch you were working in, for example:

> git checkout jrsbranch

(Remember to replace jrsbranch with whatever you called your own branch).

Now look at the contents of your directory; what is the difference?



Your file is back again; because it is part of this branch.

# Step 9: Push changes

All the changes we have made so far have been limited to your local machine. If you go to the online GitHub repository which is centrally hosting this project you will see that none of your branches are available there.

For example:

* Go to the URL <https://github.com/johnehunt/playarea>

Next select the ‘Branch: master’ drop down button to see the list of branches available. Initially it looks something like:

# A screenshot of a cell phone Description automatically generated

Your own branch is not listed.

We will now push your branch to the GitHub repository. As this is the first time we have done this; and the remote Git repository does not know anything about our repository we need to use an extended form of the git push command:

> git push --set-upstream origin jrsbranch

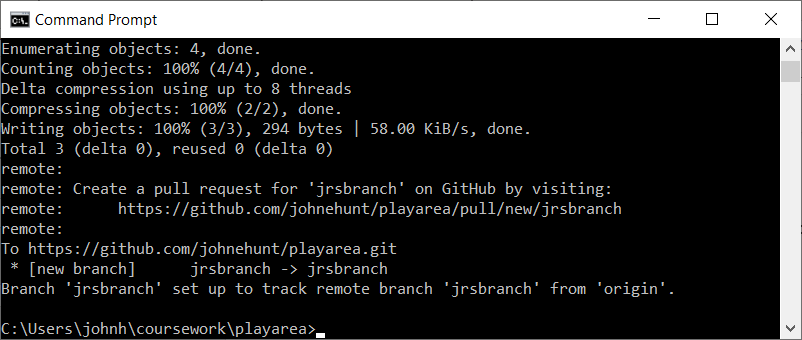
To do this you will need a GitHub account (if you do not have one then you will need to create one). You can create an account for free by going to:

<https://github.com/>

Once you have done this you can push a file to the playarea project within the johnehunt repository using git push command:

> git push --set-upstream origin jrsbranch

For example:



To see any other changes made to this project you can use the git pull command:

> git pull

If others have added their branches to the project when you do a git pull followed by a git branch you should see them listed.

If you now return to the GitHub web page you should now see your branch listed in the drop down ‘Branch: master’ menu.

A screenshot of a cell phone

Description automatically generated

Note you may need to refresh the web page to pick up the new branch. Select your branch now.

# Step 10: Pull Request

You can now create a new Pull Request (PR). A Pull Request is a request to pull the changes you have made in your development or feature branch into the main master branch.

You can start a Pull Request by selecting the ‘New pull request’ button which is next to the Branch drop down button.

You should then see a display similar to:

A screenshot of a social media post

Description automatically generated

Next click on the ‘Create pull request’ button.

Next provide a comment for this Pull Request and select one of the other people on the course to act as a reviewer and as an assignee.

Pull Requests are reviewed by one or more other team members. It is those team members who then accept the changes and merge them into the master branch.