GitHub commit using Git Bash

The software tool Git Bash is an emulation window which allows the user to work on GitHub from a command line. A user may commit to their GitHub repository so to edit, update, files, code or documentation to the Git repository.



Once a user has created an account, then signed up and created a repository on GitHub, they must be able to access the contents on the repository from Git Bash. They must track the directory of the repository.

On Bash: $ git init

This command creates and initializes a new subdirectory with the source code .git. The directory will contain all your stored files.

To store a file to the repository, so to track a file within it, a user must *commit* that file to the repository.

With an empty repository, one should add some files, code or documentation so to start version-controlling them (changes are recorded so to recall versions of the files stored, tracked in the GitHub repository. An initial commit to the repository is displayed below

$ git add \*.c

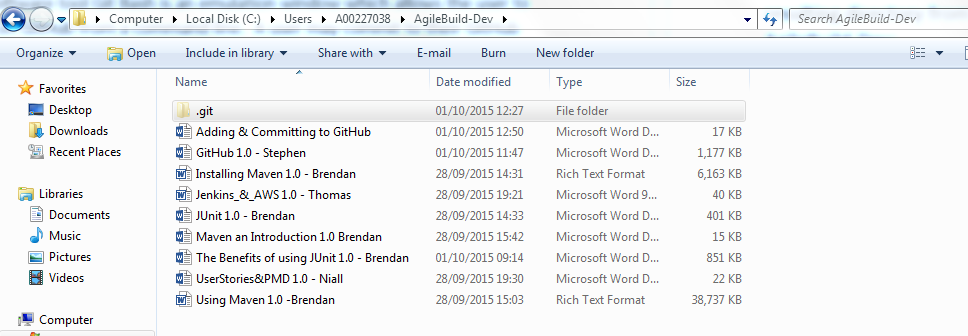
$ git commit -m

If one wishes to create a copy of the subdirectory of the existing github repository of the PC they are operating from, the command one will type in is git clone- followed by the URL of the GitHub repository. For this command to execute the user has to be typing on the command line from their working directory. For example:

A00227036@X206-A02 /c/Users/a00227038

$ git clone <https://github.com/stephenajacob/AgileBuild-Dev>

This will create a working directory from where a user can store, edit their documentation that they want to store in the GitHub repository. For example, the following screenshot shows the resulting directory from cloning the initialized GitHub repository AgileBuild-Dev:



All the following word documents are now stored in the directory. However they have yet to be tracked, included in the actual GitHub repository.

Each time a user enters the *clone* command a copy of the working directory checks out. This pulls down all the current data contained in the repository.

After one has cloned the repository directory, one must then check the status of the GitHub repository. This is to ensure there are no recently tracked or modified files to avoid separate unsaved versions of the repository.

The required command is:

S git status

If the result is:

On branch master

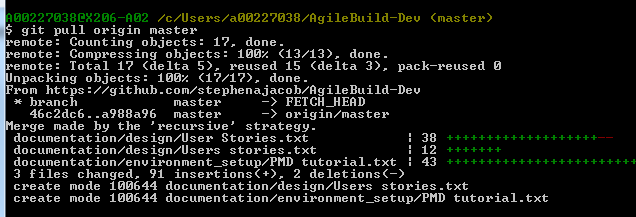
nothing to commit, working directory clean

This means there is no modified files for the user to commit.

Secondly it is necessary to pull down changes and commits from the GitHub repository by other invitees to the local checked out working directories. The following command to pull down the repository changes is this:

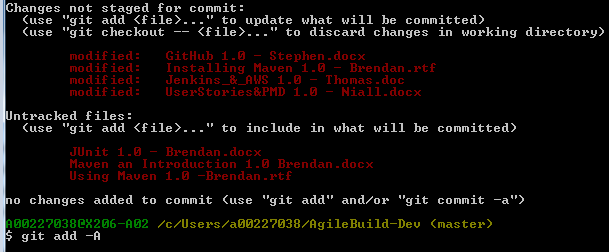
$ git pull origin master

This programming is exemplified below:



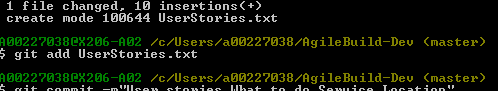
If there were files that had to be committed or were not being tracked, the solution to this issue is the following command and screenshot.

$ git add –A



If one wanted to avoid adding unnecessary files, one could simply add a specific file:

$ git add UserStories.txt



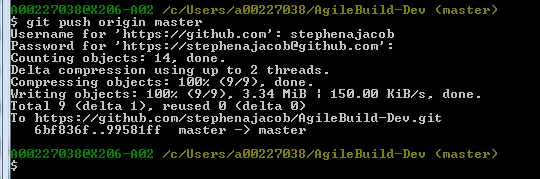
If one wanted to remove any files from the GitHub repository the command to do so, the command needed is this:

$ git remove UserStories.txt

To commit their new working directories, the command needed is the following:

$ git push origin master

This command will commit and push any edits you have made to any documentation in your working cloned directory. This is exemplified below:



A user may also mark their commits by adding a message to the commit by use of following command and example:

$ git commit –m”Moved documentation to folder and added .gitignore file

This will result in the repository being displayed as follows:

