SIT217 Software Engineering 1: Robotics Project

Pass Task 2.2 - Introduction to GitHub

Overview of Task

In this task you will learn the basics of using GitHub, a Git repository provider used widely by Open Source development communities. You will also create your first GitHub project and discover some of the basic commands for using Git.

This task will cover the following:

1. Installing Git in your VM
2. Your First Git Project

Submission Details

You will be required to submit metadata regarding your Git project to OnTrack, to evidence completion of this Task. Details are provided below.

Task Instructions

Dependencies

To complete this task you must have completed tasks 1.1PT, in order to have a basic working VM instance running Ubuntu 18.04 LTS, that has a working internet connection.

Activity - Installing Git

Within your VM, execute the following commands to install Git.

```
sudo apt-get update
sudo apt-get install git
```

Verify that the installation was successful

```
git --version
```

You should obtain an output indicating the current git version installed. Mine was 2.17.1.

Now configure Git to set your name and email address, which will be used to identify you whenever you make a commit to a project (either local or remote):

```
git config --global user.name "Your Name"
git config --global user.email "name@domain"
```

Replace the quoted strings with your name and email address (keeping the quote characters)

Your First Git Project

Create a directory called *projects* under your home directory in the VM and a subdirectory called *GitIt*. This can be achieved with a single "make directory" command:

```
mkdir ~/projects/GitIt
```

We will now create a git repository for our GitIt project. It doesn't matter that the project is empty, git can still create the meta data needed for this to be a repository.

```
cd ~/projects/GitIt
git init
```

If you look at a file listing of your GitIt directory (e.g., by using the `ls` command) you will still see an empty directory. However, `ls -a` will show all contents of the directory, and you should now note a subdirectory `.git` within the `GitIt` directory. This is where git stores information about this local repository, and can use this information to compare the local repository to remote repositories for this same project. Let's look at the status of our local repository

```
git status
```

You'll note the output indicates we're in the **master** branch of the project, that there have been no **commits** yet, and there is nothing awaiting a commit. Now we're going to add some content to the project. Create a subdirectory `doc` within the `GitIt` directory and copy the Task 2.2PT PDF document into this directory. Use the following command to update the metadata for this project to include this document (and directory) in our project

```
git add doc/2.2_github_intro.pdf
```

Run the status command again - you should receive the following output:

```
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

        new file:    doc/2.2_github_intro.pdf
```

While you have successfully added content to the project, it is not yet actually a committed change to the project. It is only a *staged change*. We can, as the status message indicates, unstage a change. For now we will commit this change to our repo. To do so, we must abide by a fundamental rule of version control - **ALWAYS SAY WHY**!!! The **WHAT** is the change we are making, the **WHY** is a commit message to say why we made the change. You can adapt the message below as you like, but when making commits to repos, **ALWAYS** add a message.

```
git commit -m "Added tutorial document for new git users"
```

You will receive output indicating how the project repo was changed as a result of the changes you committed. If you again run a `git status` command you will see that we are still working on the **master** branch, and that we have a "clean tree". We'll learn more about this in coming weeks. Take a screenshot of your VM window that shows the terminal window with the output of your `git commit` command and submit this to OnTrack, as evidence of completing this task.

You have completed Task 2.2PT and the Week 2 Pass Tasks. You should consider attempting the Week 2 Credit Task, or commencing on the Week 3 Pass Tasks.