

# CITS4407 Open Source Tools and Scripting

## Semester 1, 2021

### Week 8 workshop – Assignment 1

*Before starting this workshop, make sure you’ve reviewed the recommended reading for weeks 1–7, and completed the lab sheets for weeks 2–5.*

#### 1. Assignment work

We will use the lab/workshop time this week to continue work on Assignment 1.

If you have a question about the assignment, we ask that you post it on the MS Teams “Assignment 1” channel, so that (a) we can reduce duplicated effort if multiple people have the same question, and (b) teaching staff have some idea of how to allocate their time across the different questions.

#### 2. Git tips

##### Git remotes

When you run `git clone` to make a copy of some repository (e.g. a repository on GitHub), Git makes a copy of that repository on your local machine. In addition, it creates a link back to the original repository.

In Git terminology, a repository you’re linked to in this way is called a *remote*.

Try cloning the example repository from the week 3 workshop, by executing

```
$ git clone https://github.com/cits4407/example.git week8-example-repo
```

and then `cd` into `week8-example-repo`. You can see the link Git has created by running the command `git remote -v show`, which will produce output something like this:

```
$ git remote -v show
origin https://github.com/cits4407/example.git (fetch)
origin https://github.com/cits4407/example.git (push)
```

It is possible to remove a remote. Type:

```
$ git remote remove origin
$ git remote -v show
```

You should see no output at all – your repository now has *no* remote repositories at all. But you can always add a remote repository back again. Type:

```
$ git remote add origin https://github.com/cits4407/example.git
$ git remote -v show
```

and you should see that the original remote repository has been restored.

It's also possible to have *more than one* remote repository. Try creating a repository on GitHub using your GitHub account. Once you're logged into GitHub, click the plus (“+”) sign in the top right-hand corner of the webpage, and select “new repository”.

You should see a form come up in your browser entitled “Create a new repository”. We'll give your repository a name – for instance, “cits4407-week8” – and you should fill it in under “Repository name”.

You can also fill in the “Description”, if you like, but it's not necessary. Feel free to add whatever comments you want here.

Below the description, you'll see two options – “Public” (indicated with a book icon) and “Private” (indicated with a padlock icon). By default, repositories you create on GitHub are *public*. Anyone can view the contents, but only the repository owner (you) can make changes to the repository.

For this lab, we'll leave the selection as “Public”; but if you're creating a repository for University assignment code, you should normally select “Private”. (Otherwise, your work will be visible to the public at large, and that will be a breach of the University's Academic Conduct policy.)

Finally, click the button labelled “Create repository”.

The URL in your browser should change to something like “[https://github.com/YOUR\\_GITHUB\\_USERNAME/cits4407-week8](https://github.com/YOUR_GITHUB_USERNAME/cits4407-week8)”, and a page with “Quick setup” should be shown.

We won't use those, for the moment; instead just type

```
$ git remote add secondary https://github.com/YOUR_GITHUB_USERNAME/
  ↪ cits4407-week8
```

(replacing “YOUR\_GITHUB\_USERNAME” with your GitHub user name).

Now, if you type `git remote -v show`, you should see something like:

```
origin  https://github.com/cits4407/example.git (fetch)
```

```
origin  https://github.com/cits4407/example.git (push)
secondary  https://github.com/YOUR_GITHUB_USERNAME/cits4407-week8 (
    ↪ fetch)
secondary  https://github.com/YOUR_GITHUB_USERNAME/cits4407-week8 (
    ↪ push)
```

Your local repository now has *two* remote repositories, and you can push changes to either one.

Try typing:

```
$ git push -u secondary master
```

Git will prompt you for your GitHub username and password, so enter those. Assuming you got them correct, you should see something like:

```
Counting objects: 5, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (4/4), done.
Writing objects: 100% (5/5), 627 bytes | 0 bytes/s, done.
Total 5 (delta 0), reused 0 (delta 0)
To https://github.com/YOUR_GITHUB_USERNAME/cits4407-week8
 * [new branch]      master -> master
Branch master set up to track remote branch master from secondary.
```

This is telling you that all the content in your local repository has been *pushed* to the remote repository. The portion of the output that says `[new branch] master -> master` means that a new *branch* called “master” has been created in the remote repository, and that a *local* branch called master has been linked to it.

After you’ve typed `git push -u secondary master`, Git knows about this link. In future, you can just type `git push secondary`.

Try that now. Create a new file to your local repository by typing `echo new file > new-file.txt`, and add and commit that to your repository using `git add` and `git commit`. (Refer back to the week 3 lab/workshop if you don’t recall the syntax.)

Once a new change has been committed, you should be able to push it to your GitHub repository using `git push secondary`. Visit the GitHub page for your repository (or reload it if you already have it open), and you should see the changes you have made are now stored on GitHub.

## Submitting Assignment 1

The submission process for Assignment 1 works by having you add the CITS4407 Git marking server to your Assignment 1 code repository using `git remote add`; the process is outlined here: <https://github.com/cits4407/assignment1#submission>.

If you run into issues submitting your code, though, we have a backup submission procedure; you can submit using the `cssubmit` system at <https://secure.csse.uwa.edu.au/run/cssubmit>. Make sure you read the tips on using `cssubmit` on the CITS4407 website, here, <https://cits4407.github.io/assessment/#cssubmit-tips>, and always *check* that your submission has been properly validated, and *print out* a copy of the validation page.