LEEDS
BECKETT
UNIVERSITY

Introduction to Programming

Week 2

Tony Jenkins
a.m.jenkins@leedsbeckett.ac.uk

OBJECTIVE

Welcome to the second week!

This week we will practice using our tools, by working on some sample Python programs.

Intro to Programming

GitHub

You have created an account on GitHub.

You should be able to view your GitHub repos on the web, with a URL something like:

https://github.com/TonyJenkins

One day you may want to send this to a potential employer!

Take some time to upload an avatar, and also to complete the profile.

Note: If your GitHub account name is not a version of your real name, this will be very useful to the Module Team!

GitHub: Repos

Basics

Last week, you created a repo and did some simple tasks.

Start this week by deleting that repo.

A repo usually contains all the code relevant to one project. It can be owned by an individual or by an org(anisation).

Many repos will have collaborators, with several programmers working on one codebase.

At the simplest level, repos can be *public* or *private.*

GitHub: Repos

A New Repo

Make a new repo.

Assuming you have an account that allows you to, make it *private*. Include a README.md file and a LICENCE.

Once it is created, verify that the person sitting next to you cannot see it online.

Clone the repo into PyCharm.

GitHub: Repos

Text Files

Remember that Git stores *plain text files*.

It stores versions of these files to allow, among other things, changes to be undone.

This implies the need for a way to store documents (with headings, paragraphs, emphasis and the like), in plain text.

Happily, that is easy.

Documenting
Markdown

The README file is plain text, using a "markup" language called Markdown. You will meet Markdown *a lot*.

Check:

https://www.markdownguide.org/

for all the details.

Change the file to include your name, and anything else you like. Commit and push your changes. (You can make simple changes like this directly on the GitHub website.)

Documenting
Markdown

The README file is plain text, using a "markup" language called Markdown. You will meet Markdown *a lot*.

Markup languages like this as important. They are a *much better* way to store text documents than, say, Microsoft Word.

Consider that we will be able to view a Markdown file easily in 50 years. We would struggle to open the contents of a Word file written in the 1980s.

(Your module book is written using reStructuredText, which is a similar language popular in the Python world.)

Coding
Start with a Repo

Today we will enter three programs, and store each one in the new repo.

But first, get the IDE customised as you like it.

Using the licence file and the README, experiment with tweaking settings until you arrive at a visual appearance you like.

Note: If you prefer VS Code, exit PyCharm, and open VS Code on the same folder.

Note: If you create a JetBrains account you can sync your IDE settings in PyCharm so that, say, the same settings will apply at home. Some versions of VS Code have this feature too.

Coding
Sample Programs

There are three sample programs in the book, at the start of Chapter 4. The direct link is:

https://www.tony-jenkins.org.uk/pybook/030_getting_started/index.html#getting-started

Enter these programs, one at a time, and run them.

Once each one works, add it to your repo:

```
$ git add prog_name.py
$ git commit -m "Add some new program"
$ git push
```

Repos
Local and Remote

You *could* enter all three programs, "add" and "commit" each, and then "push" once.

This works because you have a local copy of the repo.

Typically, a programmer will make a bunch of commits, save them locally, and then push the whole lot once some work is complete.

If you have the same repo on a different machine, then you can:

```
$ git pull
```

to get any changes in the remote that you do not have locally.

Repos
Local and Remote

Finally, move to another PC in the lab.

Check that you can access your code from there.

(How you do this will depend on where your files were stored.)

Add your tutor as a **Collaborator** to your repo. Ask them to verify that they can see your programs!

Hint: Look under the Settings for your repo.

NEXT

Continue working through the practicals on MyBeckett.

If you create more programs, add them to GitHub.

Consider keeping any notes in Markdown, so they can be added to GitHub too!

Thank you

LEEDS
BECKETT
UNIVERSITY