We have compiled some helpful exercises to get you started.

Since “being able to effectively make use of developer documentation” is an important part of developer life, the exercises are in the form of “achievements” rather than full instructions.

For the “easy” ones (that you want to have done by end of today), there are some explicit instructions; and links to relevant documentation are provided for some. Whereas, for the more difficult ones, making use of earlier links, or use of on-line documentation that you search for may be necessary.

“medium” and “hard” are for the quick and/or those with substantial prior experience – don’t worry if you don’t get to the end.

Ask one of the tutors present if you get stuck!

Achievements

Basic (you should already have done that)

Hello VS Code

Ensure MS VS code and a python environment manager (conda/miniconda, venv/pip, etc) are installed on your computer. Ensure the python Extension pack is installed and recognizes your python installation.

Hello Git

Ensure Git and the MS VS code Git Extension pack are installed. Ensure VS code recognizes your Git installation.

Easy difficulty (not trivial, but self-explanatory. You should have these done by end of day!)

Hello Environment

Create a new python environment called “MyNewProject”.
Ensure that the packages scikit-learn, pandas, jupyter (standard data science packages), plus all dependencies are installed.

You have succeeded if your package list printout shows these three packages (and maybe others).

https://docs.conda.io/projects/conda/en/latest/user-guide/tasks/manage-environments.html

Hello Project

Create a new folder to be the home of your new project on your local hard drive. You should have full write access to this folder, and it should not be in your OneDrive or other auto-sync location.
Suggestion: “C:/Workspace/MyNewProject/”
(or “C:/Apps/Username/Workspace/MyNewProject” if your laptop is locked down)
Open this folder with VS code.

You have succeeded if you see the folder’s name in your sidebar under “open editors”.

https://code.visualstudio.com/docs/getstarted/introvideos

Hello Python

In your project folder, create a python file “helloworld.py” in a sub-folder “src”.

In the file “helloworld.py”, write the two lines
print(‘Hello World!’)

print(‘Hello Python!’)

and save the file.

Ensure a(ny) python environment is selected as an interpreter.

Run the file in the python terminal, and run the file line by line in the python interactive window.
Hint: right click, and shift-enter.

You have succeeded if you can see the file in the file explorer, and the expected printouts (“Hello …”) appear.

https://code.visualstudio.com/docs/python/python-tutorial

Hello Package

Ensure the active python environment is now MyNewProject (created above).

Re-start the kernel in the interactive python window, or re-open the window.

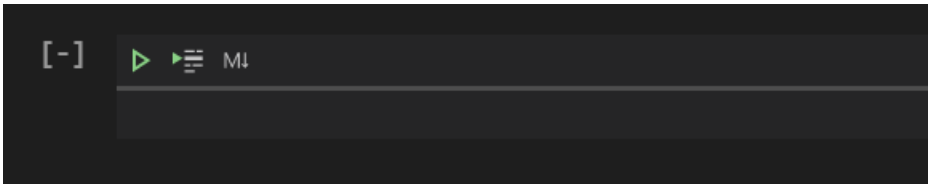
Import the pandas package in the python interactive window by typing “import pandas”.

You have succeeded if you do not get an error message when trying this.

Hello Jupyter

Create a new juptyer notebook called “helloworld.ipynb” in a sub-folder “notebooks” of your project repository. View the file in VS code.

You have succeeded if you can see this on your screen.



https://code.visualstudio.com/docs/python/jupyter-support

Hello Jupyter 2

Ensure your python kernel is using the MyNewProject environment (hint: click on “python sthsth”, top right)

Type into the jupyter cell:

print(‘Hello World!’)

and execute the cell (hint: play button, or shift-enter)

Type into the next jupyter cell:

import sklearn

and execute the cell.

You have succeeded if the “Hello World” printout appears, but no error message.

https://code.visualstudio.com/docs/python/jupyter-support

Hello Github

On Github, create a new repository, with the name liebenzell-[yournamehere], where your name replaces [yournamehere].

Hint: www.github.com/[yourusername] & 

For the time being, it's not necessary to choose any of the options.

You have succeeded if the URL www.github.com/[yourgithubname]/[yourrepositoryname] does not give you a 404, but shows a file/code overview.

Medium difficulty (may require straightforward consultation of course materials, docs or google)

Git-Github handshake

Create a new, local git repository for your project (which has the src and notebook folders with helloworld files), using the VS code Git Extensions (hint: bottom left).

Add both helloworld files and their containing folders to the repository, and commit (to the main branch – typically, you shouldn't, but it's fine for the first time).

https://code.visualstudio.com/docs/editor/github

#therealhelloworld

In VS code, configure your Github repository as the remote for your local Git repository.

Hint: bottom left, and 

Then, push your local change to the remote.

You have succeeded if your "hello world" files show up in the file view on www.github.com/[yourusername]/etc

https://code.visualstudio.com/docs/editor/github

Good documentation practice

Using VS code, in your local repository, add a markdown cell to the top of your notebook "helloworld.ipynb" that describes the purpose of the notebook, and commit your changes.

Also add a "readme.md" markdown file to your repository's root which has descriptive section headers and describes content and purpose of the repository, and how other people can contribute to the project.

Export a "MyNewProject.yml" file with the list of your environment packages and versions, add it in a folder "environments" in your repository (including commit/push).

Add/commit these changes to your local git repo, and push them to the remote Github repo.

You have succeeded if you can see the changes on the landing URL of your Github repo, in the jupyter preview of the helloworld.ipynb file on Github, and in the folder/file overview.

https://docs.conda.io/projects/conda/en/latest/user-guide/tasks/manage-environments.html

https://www.markdownguide.org/getting-started

https://guides.github.com/features/wikis/

Fruitful collaboration

Find a different participant's Github repository on public Github (you can ask them in real life, or search using the naming convention). Give the repository a star, and open a new issue that either (a) reports a critical bug in the library helloworld.py because it does not print out your favourite sentence, or (b) offers your help to enhance the software interface of helloworld.py so the printed sentence can be partly specified by the user. Ensure you make use of the "label" function to properly categorize your issue.

You have succeeded once the repository owner closes your issue due to (a) not a bug but a feature, or (b) out of scope for the proof-of-concept phase.

https://code.visualstudio.com/docs/editor/github

My own wiki

Create a wiki for your Github repository, with a homepage (table of contents) and at least two sub-pages and one sub-sub-page (e.g., terms of reference, project goals, meeting minutes, results of machine learning experiments, or software interface design). The homepage should link to the sub-pages, and the sub-pages should link to the homepage and sub-sub-page, and the sub-sub-pages should link to the sub-page. The table of content should have descriptive headers and sub-categories. Your wiki pages should contain at least one markdown table and one emoji.

You have succeeded if, from the Github page, you can click on a sequence of links starting at the wiki homepage (and excluding the side bar), so you can reach each page and get back to the homepage. In the process, you should be able to see a (non-broken) table and an emoji.

https://www.markdownguide.org/getting-started

https://guides.github.com/features/wikis/

Forever gitignored

By now, you might have accidentally committed files in the __pycache__ or .vscode folders. Ensure your repository's .gitignore file is set up in a way that those files are not version tracked. In addition, ensure

that no file with the ending ".foo" is version tracked. Remove any content of __pycache__ or .vscode directories from the latest commit in your repository, if necessary.

You have succeeded if you have at least one file ending in .foo in your respository which is not added or tracked, and no __pycache__ or .vscode filels in your repository.

https://docs.github.com/en/free-pro-team@latest/rest/reference/gitignore

Hard difficulty (may require some documentation or internet research, or day 2)

We're a team now

From the participant cohort, add a collaborator to your repository, with write access.

You have succeeded, if either:

(a) your new collaborator succeeded in accidentally deleting helloworld.py in their newest pushed commit to the master branch, and you've succeeded in restoring it to its earlier state, e.g., by reverting their commit
(b) your new collaborator failed in their attempts of pushing a commit to master because you've already enabled branch protection (see below)

In the GitFlow

Since directly pushing to main from a remote is bad practice, locally create a feature branch named "my_great_feature". Optionally, enable suitable branch protection for the main branche.

Then, make a useful change to one of your files, and commit to the feature branch. Then push, and make a pull request to main. Request a review from a competent person (e.g., your new collaborator, see above).

You have succeeded once the pull request has been competently reviewed and accepted.

Hint: you can watch the GitHub learning lab, lesson 1, first.

High TeX

To the markdown documentation of your repository, add a page with very important formulae, e.g., the Maxwell equations or the reflection formula for the general n-th order polygamma function. One option for VS code integration is the Markdown+Math extension with display support for the LaTeX typesetting language. Ensure to commit and push your changes back to the Github repository, and take note that (currently) you can see the formulae only via VS code.

You have succeeded once you can see nicely printed math formulae in VS code.

Pylint groups

Enable a linter for your project in VS code, e.g., pylint or flake8.

You've succeeded once you've linted at least one character.

Cloning around

Clone an arbitrary python library repository from GitHub (e.g., scikit-learn or sktime) to a local folder. Set up a new python environment, for that repository, and install the package from that repository.

Create – or find – a jupyter notebook with tutorial code for the package.

Create a second environment for the same repository, but this time ensure the installation is editable.

Now, intentionally break something in the cloned package so the tutorial code above would break, by committing ot your local repository. Observe how the jupyter notebook behaves in the first vs the second environment.

You have succeeded once you've succeeded in breaking the tutorial (and explaining your observations).

Later today: fork and contribute