A Software Introduction to the Concordia Aircraft Systems Research Lab

Contents

1 Introduction

Welcome to the Concordia Aircraft Systems Research Lab! This document will serve as an introduction to some of the software tools we use, as well as some programs and links to get you started if you're new to working with Python and software development.

2 Python

Python is one of the most popular programming languages today, due to it having simple syntax and being easy to learn, having a large community which develops useful libraries and makes it easy to find help, and it being free.

2.1 Where to find help

Given Python's popularity, there is a lot of help available online. The Python Foundation lists many tutorials on this page. They also have created their own tutorial here, and if you have the time for it, MIT OpenCourseWare has provided a full free introductory course on Python here.

For more specific questions you might have about Python, Googling for it will usually lead to finding an answer through Stack Overflow posts or programming websites.

2.2 Creating and Running Python Files

Python files are text files with the extension *.py*. When files have this extension, programs (such as the text editor you are using to edit the Python file) will recognize the file as being a Python file and might enable some Python-specific formatting features. It will also allow you to run these files through the Python interpreter.

To run Python files, you may want to use Conda, which is covered in Section 3.3

2.3 Python Style Guide

Part of what makes Python so appealing is that it is pretty and readeable. While some of that is due to Python's inherent structure (for example, you must tab to be inside of a loop of function), it is mostly due to the style conventions outlined in PEP-8 (PEP = Python Enhancement Proposal). This document describes style conventions which are meant to make Python be as readeable as possible, as code is read more often than it is written. It is highly recommended to give it a read, as it is not too long, and will help you write beautiful code.

Given the importance of readeability and following the style guide (also known as code linting), there are many tools to help enforce it. Most modern text editors will recognize that you are writing a Python file (thanks to the file extension) and set the tab width to be equal to 4 spaces, as defined in PEP-8. They might also auto-tab for you once you define a function or create a loop. However to make sure that you are respecting the style guide, you can use the flake8 package. By running this package on your Python file, your code will be checked against PEP-8 and have every style infraction output to the console. You can fix those errors and rerun flake8 until you do not get any more errors.

2.4 Documentation

Documenting your code and projects are important, as people are not likely to read through your actual code to figure out what it does and if they should use it. It also helps you understand your code, as over long-term projects you might not remember how the code you wrote at the beginning of the project works.

PEP-257 describes how you should document your functions and classes using Document Strings, or docstrings for short. Using this format can also let you automatically generate documentation pages using the pydoc module. You should also use comments within the code, PEP-8 describes the style you should use, and this tutorial gives examples for how to comment.

When creating a repository in GitHub (which will be covered in more detail in Section 3.1), you will want to create a README.md file to describe the project in that repository.

Github has created this page describing what typically goes in a README.md file. These files use the Markdown formatting language, and this page describes how to use it.

3 Useful Programs

This section describes some tools that aid in developing tools and writing scripts.

3.1 GitHub

GitHub is a version control system that uses the Git program to save your projects in the cloud and allow for collaboration with others. GitHub (and other Git-based version control hosting platforms) are ubiquitous in software development as it makes it easy to both track changes made to programs and collaborate on the same code without hassle.

To get started, create a GitHub account. Make sure to choose the Free plan. Once your account is made, you can transform your account into an educational account, which gives you all the perks of the Pro plan for free. To do so, follow these instructions.

A great Git (and GitHub) tutorial is this Hello World tutorial, which will introduce you to the Git workflow. If you'd like to learn more about some of the more advanced GitHub commands not covered in the Hello World tutorial, this page contains some more definitions and has links to other tutorials, projects, and videos.

As a lab, all shared tools are created in the ConcordiaAircraftSystemsLab GitHub Organization. Organizations are spaces where multiple people can collaborate and access repositories. The advantage is that all the Organization's tools are in the same spot, rather than having different tools existing in the repositories of various members. To gain access to the Lab's GitHub Organization, you will need an admin (as of time of writing, this is Prof. Liscouët-Hanke) to add you to it using your GitHub username.

To use GitHub without needing to go through the browser interface, the GitHub Desktop application is a GUI tool that makes it easy to manage your repositories and interface between your computer and GitHub. It is recommended to download and use this tool.

For command-line usage of GitHub, Windows users can install the Git for Windows console. Linux and MacOS users can install Git through their package manager and use their native Terminal program.

As a summary, GitHub is useful for us as it allows for collaboration on tools, keeping track of changes to code, and for creating a single space that we all have access to for working on those tools.

3.2 Text Editors

Python files can be written using any text editor, though some will have nice code highlighting and some automatic formatting which make it much easier to read and write code. One popular editor is VSCode, which has these features, and also supports GitHub so you can see which lines in a given file have been changed since the last commit. Documentation about using VSCode with Python (and other programming languages) is found here, and you can install a VSCode extension for Python here.

For an excellent text editor without the bells and whistles, download Notepad++. While VSCode is a much more pleasant environment to program in, Notepad++ opens files super quickly and lets you view and edit files without having to open up VSCode, which can take a few seconds to open. It is also useful for opening and reading/writing to any text file, including CSV files.

3.3 Conda

Conda is a package manager for programming languages, and has gained a lot of popularity for usage with Python. A package manager is a program that lets you easily download packages (such as installing numPy) from one or several package repositories. What makes a package manager special is that it will ensure that if you install a package, all its dependencies (other packages that it depends on to run properly) are also installed. It will also keep track of any updates to packages, so with one command (`conda update`) you can update all your packages to the most recent versions.

However, what makes Conda special is that it is also an environment manager. An environment manager lets you create environments, and install packages within that environment that do not affect other environments. For example, you might have one environment where you have Python 3.6 installed, as well as a whole host of packages that are compatible with that version of Python. If you also want to have Python 2.7 installed for running some legacy application, you can create a new environment with that version of Python and the packages that work with it. In this way, you now have two completely separate installations of Python and assorted packages that all work well together within their environment, but cannot impact each other outside of it. For more information about environments, view this document

To use Conda, you must install Anaconda, which is a collection of tools that Conda is a part of. It is recommended to go through the Conda tutorial, which will explain how to install and update packages, create and use environments, and switch between environments.

Conda in general is useful for us as, by default, it comes with most of the Python packages we would typically use for development, and makes it very easy to install new packages that we need. Environments are useful as we can develop tools using specific versions of packages, and as long as other users of those tools have the same environment, it is certain that the tools will work for them. It also means that we can install programs and use environenements that the program developers define to ensure that those programs will work for us.

3.4 Jupyter Lab

Jupyter Lab is a browser-based program that lets you create and edit Jupyter Notebooks, which are documents which contain live code, text, graphs, equations, and more! Notebooks are made up of "cells", each of which can contain code and have their output expressed below. This format is conducive to exploring or documenting a workflow. For some relevant examples, look at the notebooks available at this link (Notebooks have the *.ipynb* extension), which come from this program. This GitHub link documents scores of example Jupyter Notebooks and tutorials for a variety of different programming applications.

Since Jupyter Lab comes with Anaconda, your installation of Anaconda should have also installed Jupyter Lab on your computer. If for whatever reason it has not, you can follow the installation instructions for Jupyter Lab here. For usage, this link gives a good overview of Jupyter Lab and this link gives a user guide. This tutorial is concise and explains how to use Jupyter Lab and notebooks, as well as covering the differences between Jupyter Lab and the Jupyter Notebooks App.