Python Software

GT Off-Road Racing | Data Acquisitions

[Benjamin Boeckman]

[06/18/2021]

Table of Contents

[1.0 Overview 2](#_Toc80963154)

[1.1 Introduction 2](#_Toc80963155)

[1.2 Point of Contact 2](#_Toc80963156)

[2.0 Software Needs and Installation Guide 3](#_Toc80963157)

[2.1 Python Installation and Version 3](#_Toc80963158)

[2.2 IDE Options 3](#_Toc80963159)

[2.3 Installing Software Dependencies 3](#_Toc80963160)

[3.0 Style and Structure Guide 5](#_Toc80963161)

[3.1 Python Style 5](#_Toc80963162)

[3.1.1 Variables and Functions 5](#_Toc80963163)

[3.1.2 Classes 5](#_Toc80963164)

[4.0 Python Development Projects 6](#_Toc80963165)

[4.1 DAATA (Data Acquisition and Telemetry Application) 6](#_Toc80963166)

[4.2 . . . 6](#_Toc80963167)

[5.0 References 7](#_Toc80963168)

[6.0 Revision History 8](#_Toc80963169)

# 1.0 Overview

## 1.1 Introduction

Data is central to what we do here in the DAQ (Data Acquisition) sub team of GT Off Road. Because of this, we need to take advantage of robust programming languages that are better for certain situations than others. On the lower-level side of collection, we use C++ because of its ease of memory handling and efficiency with weaker computational devices like the Teensy microcontrollers we connect to sensors on the car. Once the data is collected on those devices though, we then need to do something with it. This includes separating it, storing it, organizing it, displaying it, and anything else you can think of. This is where Python comes in.

Python is a great programming language because it is extremely versatile and has many ways of solving the same problem. We use it to parse packets, organize data, and display it through a GUI (graphical user interface) in our data **application**\*. This is critical as it gives us accurate feedback about the car so that we can assess its actual performance without having to speculate or just trust in the design.

Now that you know you need to use Python to get involved with development for dealing with collected data, you may be wondering where to get experience with it, what IDE you should use, what version of Python to install, what dependencies you may need to work on the software, and so on. I will do my best to tackle and assess each of these roadblocks that keep you from becoming the software engineering master we all know you can be for the DAQ Team and beyond.

\***For more information on how to use, develop in, and understand the structure of the application, see “DAATA Documentation” in the Software Documentation folder of the Documentation Repository.**

# 2.0 Python Setup and Getting Started

## 2.1 Python Installation and Version

Let’s first ensure that you have Python downloaded, the right version, and that it’s added to your PATH variables.

### 2.1.1 Python Version and Download

We use [Python 3.6.8](https://www.python.org/downloads/release/python-368/) here at DAQ for developing in our data GUI app.

After clicking the download link above, find the corresponding installer at the bottom of the page for your operating system and proceed to download it, and then follow all steps of the installer.

### 2.1.2 Adding Python to PATH

An important thing if you are going to be using Python is that it must be referenced to your PATH variables. If you don’t know what those are, don’t worry. These are just applications that your OS can recognize and run when they need to be executed for your code to be interpreted properly. While you’re installing Python, be sure to specify that you want it to automatically add it to your PATH if it asks. If it does not, or if you’re not sure, use [this walkthrough](https://www.tutorialspoint.com/python/python_environment.htm) and run through it to ensure that Python was added to the PATH successfully. It includes help for just about every operating system that you’ll need assistance for.

## 2.2 IDE Options

An IDE is an Integrated Development Environment. If you’ve done any programming before you’ve probably used one or are aware of what they are. They are very helpful programs as they help you organize your code, interface with git and version control, keep you from making mistakes, and include many other benefits when developing.

Two options are highly recommended: Visual Studio Code and PyCharm. This is because they are both widely used on the team and by many developers, so you will be able to find a lot of help and assistance from team members and on the internet. Some prefer one or the other, but both will work well for team-related purposes.

### 2.2.1 Visual Studio Code

First, [Visual Studio Code](https://code.visualstudio.com/download) is a very general use programming environment developed by Microsoft that is light and versatile so it can be used with most any language you develop in.

### 2.2.2 PyCharm

Second, [PyCharm](https://www.jetbrains.com/pycharm/download/#section=windows) is more tailormade for the Python language and is the Python equivalent of IntelliJ if you are already comfortable with that from other courses. Some prefer one or the other, but both will work well for team-related purposes.

### 2.2.3 Others

Of course, we can’t force you to use either of these IDEs, it’s just more likely that it will be simpler for you to get set up and start developing in our system with these two recommended options. If you’re more comfortable with something else, feel free to use it and we’ll work with you as best we can.

## 2.3 Installing Software Dependencies

Once you set yourself up in our GIT organization and team repositories, you’ll need to be in the [GTORDaata Repository](https://github.com/Georgia-Tech-Off-Road/GTORDaata) that you can find in the provided link or in your clone of the repository on your local machine to find the batch file needed to install dependencies for development.

Search the above link or in the repository on your machine until you locate the “**install\_dependencies.bat**” file on the root or highest directory of the repository. Download it, if it is not already on your machine, and then double click it to run it and let the command prompt do its magic. Be sure, though, to get the repositories you need cloned and on your local machine because this batch file could change from time to time, and you’ll need the most recent dependencies to continue to develop on projects.

## 2.4 Getting Python Experience

If you have prior knowledge and experience in Python and feel comfortable enough to start working on some projects, feel free to disregard this section.

Otherwise, below are some helpful resources for beginning to program in Python and being prepared for the variety of projects we are working on.

### 2.4.1 Sololearn Python Course

Sololearn’s [Python Course](https://www.sololearn.com/learning/1073) is a great starter for being introduced to the syntax of the language and getting experience coding in it from the beginning. They walk you through the basics and provide interactive examples and projects.

### 2.4.2 Other Python Courses

### 2.4.2 Crash Course on PyQt

# 3.0 Style and Structure Guide

## 3.1 Python Style

### 3.1.1 Variables and Functions

We use snake case for variables and functions as seen in the examples below:

![Text

Description automatically generated]()

### 3.1.2 Classes

As for class names, we use camel case as seen in the example below:



# 4.0 Python Development Projects

## 4.1 DAATA (Data Acquisition and Telemetry Application)

## 4.2 . . .

# 5.0 References

# 6.0 Revision History

06/18/2021 (Benjamin Boeckman) – First initial write-up of documentation

07/20/2021 (Benjamin Boeckman) – Adding style guide and clarifying sections to be elaborated on

08/27/2021 (Benjamin Boeckman) – Expanded and reformatted quite a bit