**How to Create a Python Package**

A python package is a collection of modules that are grouped together that work with one another for a specific purpose. An example of a Package is PyGame which involves a series of modules that deal with graphics and games design, or mys-math 0.7.0 which deals with mathematical equations.

Packages help structure the different modules use through the use of ‘dotted module names’. I.e. *NameA****.****NameB* where NameA is the name of the package and NameB is the name of the sub-module. For example, in a package named PyCars we have a module named Emissions.py. The full path of the module is now *PyCars.Emissions.py.* A major advantage of using the approach to naming modules and packages is that such a method enable multiple authors to work on a package without them needing to worry about the specific module names they each use.  
(<https://docs.python.org/3/tutorial/modules.html#packages>, 23/10/2020)

Packages also allow you to go to deeper levels. For example, we could have a package named Vehicles, with modules dedicated for Cars, and another set of modules dedicated for Boats. In effect, Cars and Boats are sub-packages of Vehicles. To use the Emissions as an example: Vehicles.Cars.Emissions is the full name for the Emissions module.

The following steps are designed to assist you with creating a Python Package:

1. You should have a top level directory for your package. To use the above example: Vehicles
2. Inside this, on older versions of Python, a \_\_init\_\_.py module was required. However, with newer versions of Python (3.0 and up), this is no longer needed.
3. The next directory level should be groupings of modules with a common link. Using our example: Cars and Boats
4. Inside each of these sub-directories should be the modules required for each. i.e. Emissions.py or PropSpeed.py