BlueSky General Notes/Tips

1. The qt version of BlueSky is recommended as it uses separate 'threads' for the simulation computations and the visuals on the screen (i.e., the graphical user interface). This means that the simulation calculations and the graphics are completely decoupled, reducing the load on your computers CPU. This in turn makes the simulation run faster.
   1. The graphics/visuals are in fact running on your graphics card using openGL. This also means that you need to have a graphics card that supports the required versions of pyqt (pyqt4 or pyqt5). For older computers with Intel CPUs and with 'integrated graphics' only (no dedicated GPU), these versions of pyqt are not supported.
   2. If this is the case, use the pygame version of BlueSky. It works basically the same as the qt version, but both graphics and simulation computations are performed on the CPU.
      1. In the code, the simulation module (sim) has different classes and functions for the qt and pygame versions of BlueSky. All other modules are the same.
   3. To find out which version works for you, run *check.py.* This script will check if the qt version will work for you. Else, the pygame version will be used, by accordingly changing the value of the *gui* variable in the *settings.cfg* file which houses the basic simulation parameters.
2. As the qt version is multi-threaded, the normal python debugger doesn't work as normal as it doesn't know what to do with the different threads. However, there is a simple workaround. Simply '*import pdb'*, the debugger module, and use the code '*pdb.set\_trace()'* at the point you want to enter the python debugger. After that the usual debugger commands all work!
3. The basic simulation parameters, such as the simulation dt (*simdt*), can be specified in the file 'settings.cfg', which is located in the home folder. These parameters can be changed to your desired values.
   1. The script settings.py (./bluesky) looks inside settings.cfg file and updates simulation parameters if they have been changed from default values. This happens when the first line of BlueSky.py is executed (*from bluesky.traf import Navdatabase*).
   2. The parameters in the settings.py are used to initialize member variables of the BlueSky objects, i.e., **the variables in settings are NOT global variables**.
   3. Therefore, if you want to use any of these parameters in your code, use '...import settings' (make sure you get the path to settings righ,t). Then, to use the simulation dt, for example, use 'settings.simdt' in your code.
4. The \_\_init\_\_.py script in each BlueSky Module converts all the scripts in that folder to a 'package'. This means that python can treat a BlueSky module folder as a python script, making it possible to import classes etc. in a module much more easily e.g. *from bluesky.traf import Navdatabase* instead of *from bluesky.traf.****navdb*** *import Navdatabase*