BlueSky\_qtgl.py

* This script contains the *qtgl* version of the MainLoop function (see below)
* The MainLoop function can be called from either BlueSky.py, or directly by running this script (BlueSky\_qtgl.py)

# Function MainLoop()

* This function is in BlueSky\_qtgl.py

Goal: The MainLoop function first instantiates the basic BlueSky objects:

1. *navdb*: an instance of the *Navdatabase* class (./bluesky/traf/navdb.py)
2. *gui*: an instance of the *Gui* class (./ui/gtgl/gui.py)
3. *sim*: an instance of the Simulation class (..\bluesky/sim/qtgl/simulation.py)
   1. This object also contains the other important BlueSky objects such as *traf* (Traffic.py), *stack*, *screenio* etc.

After the above objects are created, the MainLoop function starts the simulation and gui threads. Although this function is called ‘MainLoop’, there is no 'loop' in this function in the strictest sense. However, when the simulation thread is started, the doWork method of the Simulation class, which contains the ‘main’ BlueSky loop, is activated. Thus MainLoop function is responsible for triggering the main loop, and thus deriving its name (the name is also due to historical reasons from the pygame version of BlueSky).

Note that the simulation thread object (*simthread*) is created and started first. Then the gui thread is started second, causing the splash screen to be displayed. When the gui and simulation threads have really finished starting (it takes a few seconds), the splash screen disappears, and the main BluSky Gui is ready for user inputs. The simulation thread finishes initializing before the gui thread as it was started first. It then starts the doWork function of the Simulation class (before the Gui pops-up). Once both threads are started, the MainLoop function does **not** proceed to the next lines of cod, until the sim and gui threads are stopped.

Inputs: -

Outputs:

1. gui(object, instance of the Gui class) -> this is only returned so that the gui can be deleted when BlueSky exits.