Instructions For Running the Package Data Gathering Code in ROS and RTK

1. On the GitHub (https://github.com/parkerm1417/ROS_Data_Collection), the user should download the four python files: packages.py, messages.py, services.py, and TextOutput.py. **These files must all be saved in the same file location, as they rely on each other for information**.
2. Follow the procedures on the ROS website to install and build the ROS environment
    a. http://wiki.ros.org/kinetic/Installation/Ubuntu
3. Ensure that the RTK files are correctly sourced by the ros commands by typing the following command into the terminal:
    a. `echo “source /opt/rtk/kinetic/setup.bash” >> ~/.bashrc`
    b. `source ~/.bashrc`

    Now if “rospack list” is run, it should list all ROS and RTK packages

4. Python 3.8 must be installed if it has not already been. To do so, run the following commands in the terminal:
    a. `sudo add-apt-repository ppa:deadsnakes/ppa`
    b. `sudo apt update`
    c. `sudo apt install python3.8`
    d. `python3.8 --version`
        i. Output should be a version of 3.8 probably either Python 3.8.4 or Python 3.8.5
    e. `sudo update-alternatives --install /usr/bin/python3 python3 /usr/bin/python3.5 1`
    f. `sudo update-alternatives --install /usr/bin/python3 python3 /usr/bin/python3.8 2`
    g. `sudo rm /usr/bin/python3`
    h. `sudo ln -s python3.5 /usr/bin/python3`
5. If the user has not already installed idle3.8 onto their machine, the user should install it by typing in the terminal:
    a. `sudo apt install idle-python3.8`
    b. Idle 3.8 is a runtime environment that has a very easy to use GUI, and clearly shows the progression of the code. Idle 3.8 is also necessary to run the code on python 3.8.
6. In the Ubuntu terminal first ensure that the user is accessing the correct directory that the python scripts are housed in, then type the command:
    a. `idle-python3.8 packages.py`
    b. This accesses the code in the idle GUI. Ensure that at the top of the GUI after the file path that there is a (3.8.5) or another version of 3.8.
7. Within the idle interface, to run the code, select the “Run” menu at the top of the window and then select “Run Module”. A small window may pop up that reads “Source Must Be

Saved OK to Save?", and the user should click "OK" to save the code and continue to run the script.

8. After the code begins running, a roscore will be started in the terminal. If a previous run of the program stopped prematurely, then the program will ask if the user wants to continue the program where it left off. If the user responds with "n" or a previous run is not found, then the user will then be prompted to enter the packages that they desire to get information from. A list of all the possible packages to choose from will be put in to the packageslist.txt file on the Desktop. The user is to choose between one of three input options for the packages they desire:
    a. The packages should be listed using the exact package names and separated by only a comma (NO SPACES BETWEEN).
    b. The user enters "ALL" in the prompt if they wish for all the packages in their current build to be processed.
    c. The user enters the exact file path location (ex. /home/mie/Desktop/packages.txt) of a .txt file that has all the desired packages in it, with each package listed on a separate line. (Make sure there are no extra spaces)

    If any package entered is not a valid package the program will display a message stating that a package is invalid. The user will be prompted to enter the packages again until all entered packages are valid.

9. **If all packages from both ROS and RTK are being run, expect the code to take near 2 hours to complete.**
10. The user will then be prompted to enter what they wish to name the files that will be created once the program is completed. The file names will be in the format: 'YOUR INPUT'_msgs_Date.json or 'YOUR INPUT'_PackageDictionary_Date.json Everything after "YOUR INPUT" is generated automatically.
11. The code will then begin a process of starting and killing the nodes within the distribution of ROS or RTK that is installed. Within the command terminal, there will be lots of output that gets displayed, and there will be many warn messages that appear as such:
    a. [WARN] [1595946514.361387185]: Shutdown request received.
    b. [WARN] [1595946514.361858065]: Reason given for shutdown: [user request]
    c. These warn messages are normal and should happen because they indicate that the nodes are being killed correctly.
    d. When running the code in idle, the output window will not display the terminal output, rather increasing values will be printed for each package that is completed. The output will look something like: "Package 1 of 230 completed". The numbers will reset back to 1 when the code moves to gathering message info, and then will reset to 1 again when gathering service info. This indicates that the code is moving along smoothly.
    e. If the increasing numbers seem to stop or get stuck, the code may have crashed and will need to be restarted, however the previous info that was gathered prior to the crash will be saved and when the code is restarted, it will pick back up where it left off.

12. While the nodes are being run, GUI windows will pop up. These windows can be exited out of and will not affect the success of the code. Exiting these windows will help the code complete faster and will help keep your computer from crashing. A list of windows that have had trouble exiting automatically are listed below (the package numbers refer to when all the packages are being run):
    a. There may be multiple windows that come up throughout that say "Sorry the application has stopped unexpectedly". This is nothing to worry about, but will need to be manually exited
    b. Behavior Switch Test GUI (Package 13 or 14)
    c. Gazebo (Package ~80)
    d. Untitled Window (Package ~144)
    e. Figure Path Curvature (Package ~158)
    f. safety_monitor_speed_inflator (Package ~170)
    g. About 25 various rqt windows (Packages ~275-300)
    h. Myviz (Package 316)
    i. Toy Car (Package ~370)
13. Once all the packages are completed, the code will then start running through all the messages from the selected packages. If Idle is used, a clear counter indicating the message that is being run will be shown like such: Message 1 of 500 Complete. These messages are stored in a separate dictionary.
14. Once the messages are completed, the same will be done for the services.
15. Once the code is completed, a message reading "PROGRAM IS DONE RUNNING PLEASE KILL ANY PROGRAMS STILL RUNNING" will be printed out until the program is killed or exited. Four .json files will be completed in the directory that the code was run from. The files will be named with the title the user inputted before the code began.
    a. These dictionaries hold the information that was gathered during the execution of the code.
    b. The dictionaries are saved as the code progresses, and if the code is exited or crashes, the information will be saved, and the code will pick back up from where it previously ended when run again.
16. Even after the program has finished running and it has been killed, there may continue to be output appearing in the terminal window. If this is the case, close the terminal window to kill any node or other programs that could still be running on the machine.

Instructions for Running the TextOutput.py

1) Ensure that TextOutput.py has been downloaded from the ROS_Data_Collection GitHub.
2) In the terminal type:
    a. `idle-python3.8 TextOutput.py`
    b. This will open the code in the idle GUI
3) Click Run at the top of the GUI and then click Run Module. The user may be prompted to save the code before proceeding to run, click OK.
4) The idle shell will start up and the user will be prompted to enter the file path of the .json package dictionary that they want to get output from. This should be the full pathway such as /home/mie/Desktop/my_PackagesDictionary.json. The user should then click enter.
5) The user will be prompted to enter the .json messages dictionary that they want to access information from. This should be the messages dictionary that was created during the same run as the packages dictionary that is being used. The full pathway should be entered, the same way as the package dictionary pathway was entered. The user should then click enter.
6) The code will begin running and shouldn't take very long to complete, as only text files are being created. There will be an output in the idle shell indicating the package that the code is processing through.
7) When the code completes, a directory titled "Architecture_Documents" will be created on the desktop. Within that directory there will be two folders created, "RTK" and "ROS", and those folders will have package folders within them, and each package folder stores the documentation for each node within that package. This documentation is formatted to be copied over into a word document and given a multilevel list format. The indentation of the output will correctly align the text when given the multilevel list format.

Toubleshooting

- When switching the script between ROS and RTK, ensure that the directories that are being accessed with the code matches the directories you are desiring to pull information from
- When entering the desired packages to grab information from, make sure that the package names you type are exact and that commas (with no space) separate the packages listed. If this is not done exactly, there will be an error.
- If all else fails, try restarting the Ubuntu virtual machine because for some reason that's all it takes sometimes.