**Creating Windows executables from Python scripts using PyInstaller**

* This specifically addresses how to create .exe files from Python scripts when those scripts utilize an outside file (particularly .txt)

**Requirements inside .py file**

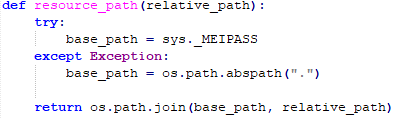
Generally, a .txt file is accessed and utilized inside a .py file using something like:

**with open(full\_file\_path, ‘r’) as file:**

Example of original approach used in Gene.py from PhageCommander:

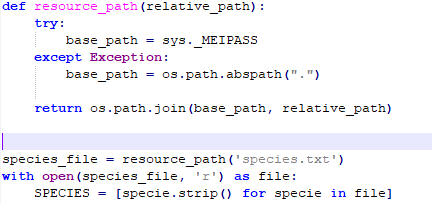


When creating an executable with PyInstaller and including a data file, PyInstaller puts your data into a temporary folder and stores the directory path in a \_MEIPASS temporary folder. In order to get the .exe to pull your data from that temporary location, files need to be accessed slightly differently:



This method will attempt to create a file path using the \_MEIPASS directory (which is appropriate when run from the .exe). This will point the program towards the correct location and the data file will be pulled appropriately. If running through Python, this path will not exist and the file will then be pulled from the usual location (which is generally the folder that phagecom is run from).

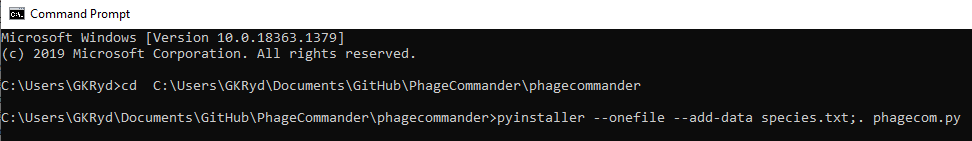
The full code in the updated version of Gene.py is



**Creating file using the command line**

The most convenient way to distribute an executable to people is likely through the --onefile option of PyInstaller. This creates a single .exe file (thought the size is about ~50 MB for PhageCommander as of right now) instead of a directory full of files and sub-directories. In order to create an executable and include the desired .txt file:

* In the command line, navigate to the directory containing the Python script you wish to convert into a Windows executable
* Then, use the --onefile and --add-data options with PyInstaller as seen below:



The text for a clearer example:

**pyinstaller --onefile --add-data species.txt;. phagecom.py**

Hitting enter will then create the .exe file inside a folder called “dist” that populates inside the folder where your Python script is.

Older versions of PyInstaller apparently employ a different method for adding data files to your executable. If there are significant changes with any future releases then a different approach might be needed. The older fixes are fairly similar so likely future “fixes” (if needed at all) wouldn’t be too different either.