Automate Your GIS Using Python

Ryan Moore

Manager of Geographic Information Services

Rochester Public Utilities

MID - WEST EUUG ESRI Utility Users Group

What is Python and ArcPy?

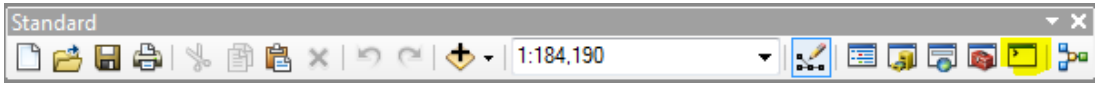
- Python is a free, cross-platform, open-source programming language that is both powerful and easy to learn. It is widely used and supported. To learn more about Python, visit python.org.
- Python was introduced to the ArcGIS community with ArcGIS 9.0.
- ArcPy is a Python site package that provides a useful and productive way to perform geographic data analysis, data conversion, data management, and map automation with Python.
- ArcPy includes modules covering other areas of ArcGIS. ArcPy is supported by a series of modules, including a data access module (arcpy.da), a mapping module (**arcpy.mapping**), an ArcGIS Spatial Analyst extension module (**arcpy.sa**), and an ArcGIS Network Analyst extension module (arcpy.na).

Source
http://desktop.arcgis.com/en/arcmap/latest/analyze/python/what-is-python-.htm

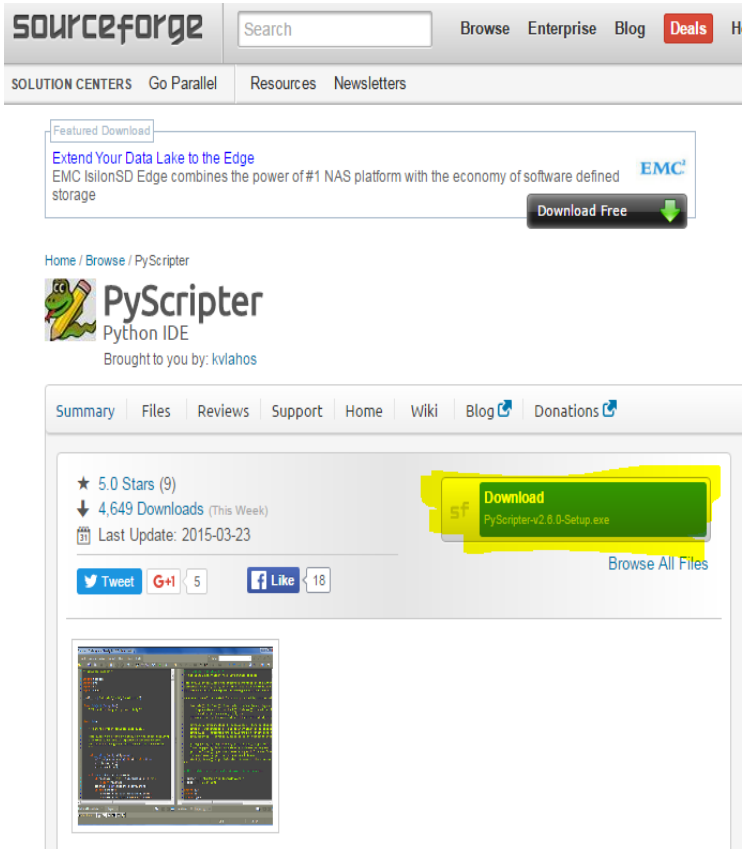
http://pro.arcgis.com/en/pro-app/arcpy/get-started/a-quick-tour-of-arcpy.htm

How to work with ArcPy?

- ArcGIS's Python Command Window (Available in ArcMap and ArcGIS Pro)



- Develop Python scripts using an integrated development environment (IDE) such as PyScripter



MID - WEST
EUUG
ESRI Utility Users Group

Running a Simple ArcToolbox Tool from the ArcPY Command Window

- To figure out the syntax search for the tool
- Right click on the tool and select Help
- Scroll Down to bottom for sample code for the Python Window and stand alone script
- **Tip:** You can drag datasets and or workspaces from the ArcCatalog to the Python window to get the file paths

Code Sample

Buffer example (Python window)

The following Python window script demonstrates how to use the Buffer tool.

```
import arcpy
arcpy.env.workspace = "C:/data"
arcpy.Buffer_analysis("roads", "C:/output/majorrdsBuffered", "100 Feet", "F
```

Buffer example (stand-alone script)

Find areas of suitable vegetation that exclude areas heavily impacted by major roads:

```
# Name: Buffer.py
# Description: Find areas of suitable vegetation which exclude areas heavil

# import system modules
import arcpy
from arcpy import env

# Set environment settings
env.workspace = "C:/data/Habitat_Analysis.gdb"

# Select suitable vegetation patches from all vegetation
veg = "vegtype"
suitableVeg = "C:/output/Output.gdb/suitable_vegetation"
whereClause = "HABITAT = 1"
arcpy.Select_analysis(veg, suitableVeg, whereClause)

# Buffer areas of impact around major roads
roads = "majorrds"
roadsBuffer = "C:/output/Output.gdb/buffer_output"
distanceField = "Distance"
sideType = "FULL"
endType = "ROUND"
dissolveType = "LIST"
dissolveField = "Distance"
arcpy.Buffer_analysis(roads, roadsBuffer, distanceField, sideType, endType,

# Erase areas of impact around major roads from the suitable vegetation pat
eraseOutput = "C:/output/Output.gdb/suitable_vegetation_minus_roads"
xyTol = "1 Meters"
arcpy.Erase_analysis(suitableVeg, roadsBuffer, eraseOutput, xyTol)
```

MID - WEST EUUG
ESRI Utility Users Group

ArcPY Syntax and Parameter Tips

Paths

Programming languages, such as Python, treat a backslash (\) as an escape character. For instance, `\n` represents a line feed, and `\t` represents a tab. When specifying a path, a forward slash (/) can be used in place of a backslash. Two backslashes can be used instead of one to avoid a syntax error. A string literal can also be used by placing the letter `r` before a string containing a backslash so it is interpreted correctly.

```
import arcpy

arcpy.GetCount_management("c:/temp/streams.shp")
arcpy.GetCount_management("c:\\temp\\streams.shp")
arcpy.GetCount_management(r"c:\temp\streams.shp")
```

Required versus optional parameters

Tool parameters can be either required or optional. Optional parameters are surrounded by braces `{ }`; required parameters are not.

Parameter Type	Symbol	Meaning
Required		Required parameter. These parameters are always the first parameters in the command. You must provide a value for required parameters.
Optional	`{ }`	Optional parameter. These parameters always follow the required parameters. If you don't enter a value for an optional parameter, the default value is calculated and used. A parameter's default value can be found in the tool's help.

Parameter types

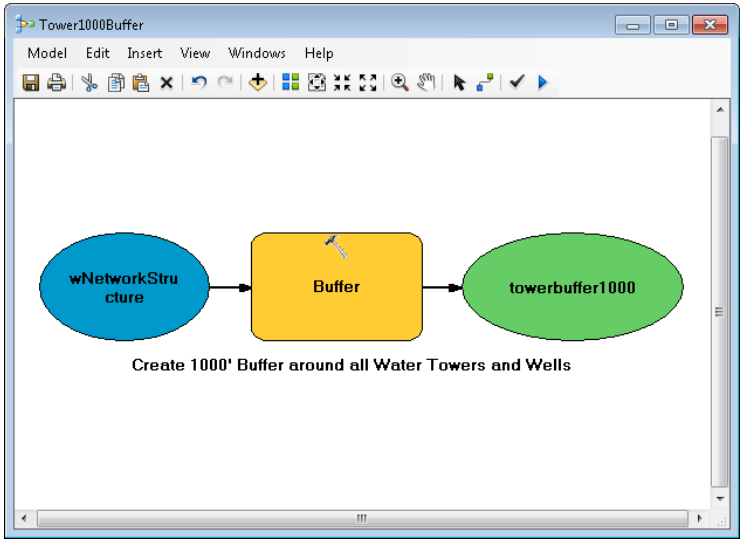
MID - WEST
EUUG
ESRI Utility Users Group

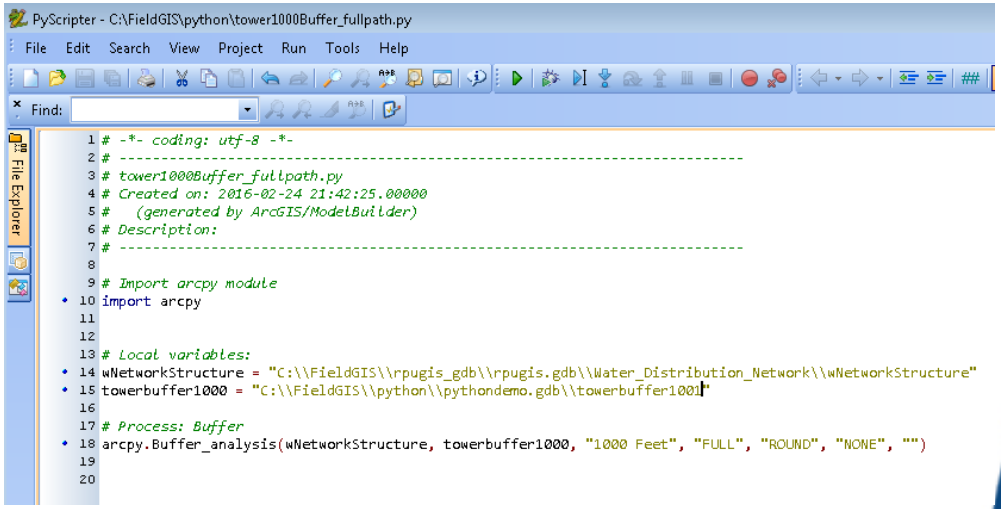
ArcPy Command Window Demo (Buffer Command)

MID - WEST
EUUG
ESRI Utility Users Group

Building an ArcPy script from a model

- Create the model in Model Builder
- Save the Model
- Click the Model – Export – To Python Scrip
- **Tip:** The script won't run "as-is" if you have any variables in the model that were created by dragging layers from the Table of Contents.





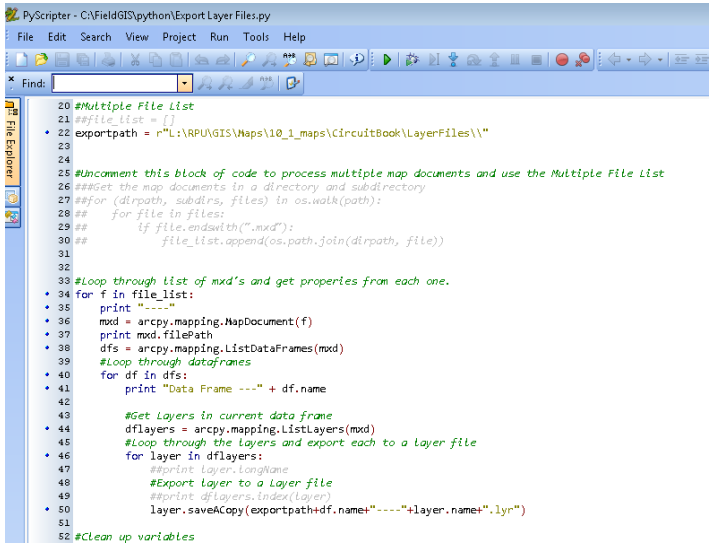
MID - WEST EUUG ESRI Utility Users Group

Model Builder to Python Demo

MID - WEST
EUUG
ESRI Utility Users Group

Building a Script in PyScripter

- Import the ArcPy and other necessary modules
- Keep in mind that Python is case sensitive
- Comments can be made starting a line with a # sign
- Indentation is used to organize blocks of code (Ex. If and For statements)
- Make use of the Debug tools and set breakpoints
- Utilize the IDE windows under the View menu Pulldown
- Be aware of Python Escape Sequences.



Escape Sequences

This all of the escape sequences Python supports. You may not use many of these, but memorize their format and what they do anyway. Try them out in some strings to see if you can make them work.

ESCAPE	WHAT IT DOES.
\\	Backslash (\)
\'	Single-quote (')
\"	Double-quote (")
\a	ASCII bell (BEL)
\b	ASCII backspace (BS)
\f	ASCII formfeed (FF)
\n	ASCII linefeed (LF)
\N{name}	Character named name in the Unicode database (Unicode only)
\r	Carriage Return (CR)
\t	Horizontal Tab (TAB)
\uxxxx	Character with 16-bit hex value xxxx (Unicode only)
\Uxxxxxxxx	Character with 32-bit hex value xxxxxxxx (Unicode only)
\v	ASCII vertical tab (VT)
\ooo	Character with octal value ooo
\xhh	Character with hex value hh

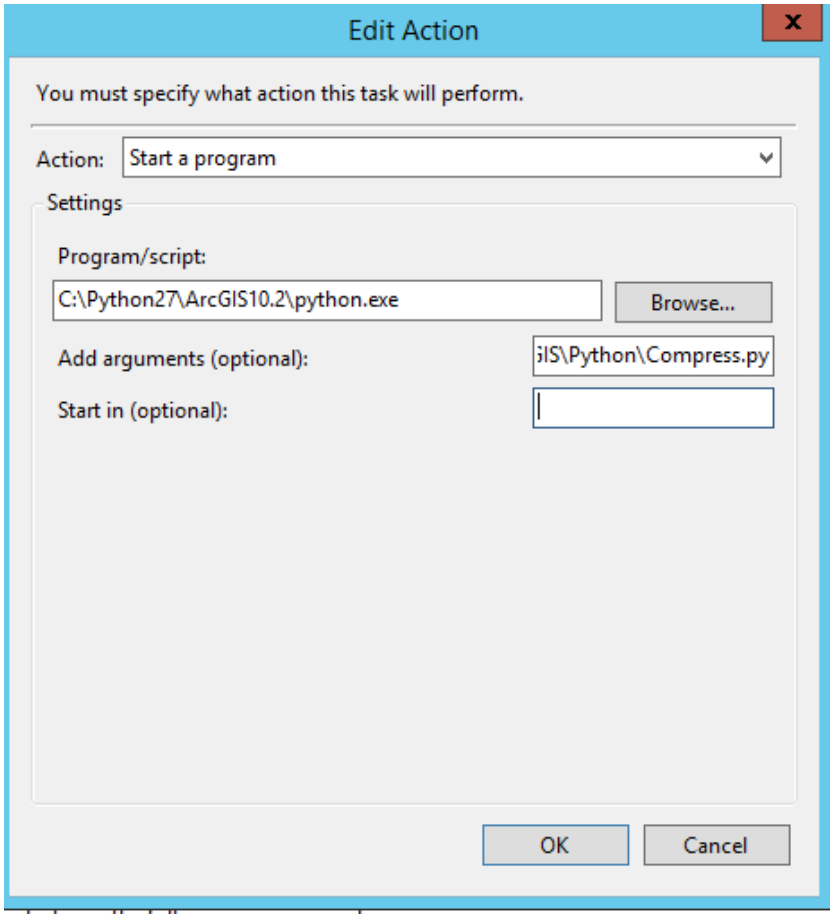
Scripts used at RPU

- Geodatabase Maintenance Scripts
  - Reconcile Versions.py
  - Compress.py (Runs as a scheduled Task)

- Map document changes/exports
  - Export Layers.py
  - Update Definition Queries.py
  - Export map documents to PDF.py

Running a Python Script using Task Scheduler

- Open Task Scheduler and create a new task
- Under the Action Tab create a new action “Start a program”
- Browse for your python.exe
- In the Add arguments text box put the path and filename of the python script you want to run



Where to find ArcPy documentation and examples?

- ArcToolbox Desktop Help – Geoprocessing Tools
- ArcGIS for Desktop online help (Click the Analyze Tab)
- http://desktop.arcgis.com/en/arcmap/latest/analyze/main/what-is-geoprocessing.htm