Esri Developer Summit

March 8–11, 2016 | Palm Springs, CA

Creating Geoprocessing Services with Python Script Tools

Kevin Hibma

- Help topic http://esriurl.com/gpSrvPy

- Script tools as a GP Service usually:
  - Have some project data (data the server has access to)
  - User supplied inputs (small datasets, strings, longs, etc)
- Because of this the authoring pattern is usually straight forward

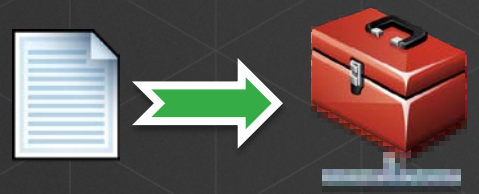
Topics

- Understanding and writing good input and output data paths:
  - ➕ (plus) is for math, not paths
  - *scratchFolder* and *scratchGDB*; variables you've been missing
- #1 Performance tip:
  - Do this (layers), not that (make feature layer)
- Validator code:
  - When, where and huh?
- 3rd party modules and packages
  - Folder variables can be used to do what?
- File output:
  - Let the gp framework do the work.
- Messages:
  - AddMessage, AddWarning, AddError: bend them to your will

What happens during publishing?

- Script is scanned
- Data identified

- If <u>in</u> datastore, update path (if necessary)



- **If <u>not</u> in datastore, consolidate data, update path**

Data Paths (inputs)

- Build up good paths – OS package is your best friend
  - Inputs / project data can be absolute or relative:
  - `pts = os.path.join(sys.path[0], “Tooldata”, “points.shp”)`
  - `pts = “c:\\projects\analysis\tooldata\points.shp”`

- Don’t build paths with “+” operator!
  - `pts = r“c:\projects\analysis” + “\tooldata\points.shp”`

- `“c:\\data”` or `r“c:\data”` or `“c:/data”`

*D1

Performance

- User layers!
  - Name match them in your script to the layer in the ToC

- Write output to in_memory
  - os.path.join("in_memory", "output") or "in_memory\\output"

- http://esriurl.com/gpPetTips
  - Local data (not network data)
  - Preprocess data (don't run tools needlessly)
  - Attribute indexes / Spatial indexes
  - Avoid different coordinate systems

- 0.x
- 0.5
- 0.x
- 0.x

x.x
x.x
1.x
1.x

= Tenths / Full seconds per execution

*D2

Validator Code

- Validator code is fired at service execution, not before on the client.
  - Where should you “validate”?

```python
def updateMessages(self):
  """Modify the messages created by internal validation for each tool
  parameter.  This method is called after internal validation."""

  #Value 1 must be between 1 and 10
  if self.params[0].value:
    if not ( int(self.params[0].value) > 0 and int(self.params[0].value) < 10):
      self.params[0].setErrorMessage("Value must be between 1 and 10")


  #Value 2 must be between 10 and 20
  if self.params[1].value:
    if not ( int(self.params[1].value) > 10 and int(self.params[0].value) < 21):
      self.params[1].setErrorMessage("Value must be between 11 and 20")

  return
```

*D3

3rd Party Modules (folder variables)

- 3rd party packages (SciPy) for example are not consolidated/moved to the server. You must ensure they have been installed on the Server

- Create a variable pointing to a folder
  - Folder will be copied or referenced depending on data store setup

*D4

File Output

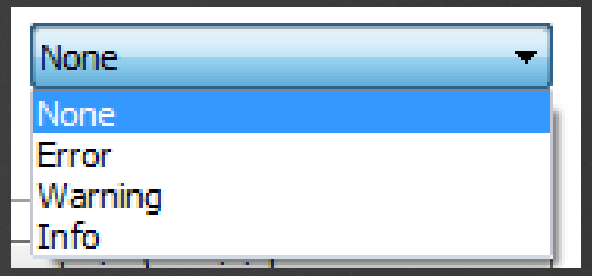
- It does not matter where the file will be created
- You don’t need to worry about virtual directories and file URLs

- Simply return the file.

*D5

Messages

- arcpy.AddError(“Oh-oh!”)
- arcpy.AddWarning(“Ahhhhh”)
- arcpy.AddMessage(“Hello!”)
- Remember: client needs to get and display these. ArcMap ‘just does it’. WebApps need code.



*D6

esri®