GEOG 432/832: Programming, Scripting, and Automation for GIS

Unit 05.01: Interrogating spatial data structures

Dr. Bitterman

Today's schedule

- Open discussion

- Readings

- Discussion and exercises

- For next class

- Maybe some work time?

Open discussion

- Lab 2 progress?

Does the data exist?

Why do (or should) we check if the data exist?

The syntax of the Exists() function is:

```
arcpy.Exists(<dataset>)
```

For example:

```
import arcpy
print(arcpy.Exists("C:/Data/streams.shp"))
```

What does it return?

- Remember, two types of paths
    - i. **System paths:** these are the paths recognized by the operating system
    - ii. **Catalog paths:** these are the paths that only ArcGIS Pro recognizes

4

But what if we want to know more about our data?

Describing data

- Certain tools only work certain types of data

- For example...
    - **Union:** requires 2 polygon feature classes

- ArcGIS Pro GUI tool dialog boxes have built-in type validation

- In Python, **you** have to determine the feature type of a dataset before using it in a tool

But it's YOUR script, why bother with type validation?

Functions to describe data

Two primary ways:

1. `arcpy.Describe()`

2. `arcpy.da.Describe()`

Accomplish same tasks, but structured a bit differently

- `da.Describe()` is newer, and ESRI encourages its use
- but `arcpy.Describe()` is functionally fine, and NOT deprecated

`Describe()`

- `Describe()` returns an object of type Describe
- Describe object properties are *dynamic* - differ by the object being described!
  - Flexible
  - Sometimes confusing...

which is why checking for types is useful!

`da.Describe()`

- Returns the same information as `Describe()`, but the result is a **dictionary**
- Wait, what's a dictionary?

The dictionary data structure

Main property is `key:value` **pairs**

Example (from W3schools)

```
thisdict = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
print(thisdict) # prints dictionary contents
print(thisdict["year"]) # prints 1964
```

Let's try it

using `arcpy.Describe()`

- Open an ArcGIS Pro project, create a new notebook

- Using `Streams_303_d_` feature class from week03inclass dataset

General Syntax: `arcpy.Describe(<input dataset>, {datatype})`

Our example (note: you can setup a workspace or point directly to the shapefile)

First, what do we expect the code to do?

```python
import arcpy
desc = arcpy.Describe("C:/YOURLOCATION/Streams_303_d_.shp")
print(desc.shapeType)
```

What happened?

Try with `da.Describe()`

General syntax is `arcpy.da.Describe(<input dataset>, {datatype})`

First, what do we expect the code to do?

```python
import arcpy
desc = arcpy.da.Describe("C:/YOURLOCATION/Streams_303_d_.shp")
print(desc["shapeType"])
```

What happened?

So how is this useful?

Textbook example:

```python
import arcpy
arcpy.env.workspace = "C:/Data"
infc = "streams.shp"
clipfc = "study.shp"
outfc = "streams_clip.shp"
desc = arcpy.da.Describe(clipfc)
type = desc["shapeType"]
if type == "Polygon":
    arcpy.Clip_analysis(infc, clipfc, outfc)
else:
    print("The clip features are not polygons.")
```

Let's break down the code -- what does it do?

Another example

```python
import arcpy
fc = "C:/Data/streams.shp"
desc = arcpy.da.Describe(fc)
sr = desc["spatialReference"]
print("Dataset type: " + desc["shapeType"])
print("Spatial reference: " + sr.name)
```

Let's break down the code -- what does it do?

Even more utility

```python
import arcpy
arcpy.env.workspace = "C:/Data/study.gdb"
element = "roads"
desc = arcpy.da.Describe(element)
print("Data type: " + desc["dataType"])
print("File path: " + desc["path"])
print("Catalog path: " + desc["catalogPath"])
print("File name: " + desc["file"])
print("Base name: " + desc["baseName"])
print("Name: " + desc["name"])
```

Let's break down the code -- what does it do?

Let's use it with our data (back to our earlier example)

```python
import arcpy
desc = arcpy.da.Describe("C:/YOURLOCATION/Streams_303_d_.shp")
print("Shape type:" + desc["shapeType"])
print("Data type: " + desc["dataType"])
print("File path: " + desc["path"])
print("Catalog path: " + desc["catalogPath"])
print("File name: " + desc["file"])
print("Base name: " + desc["baseName"])
print("Name: " + desc["name"])
```

What happened?

Can you envision utility in your labs? In your project? In research/jobs?

For next class:

- Read Chapters 6 & 7 for this week

- Project proposal is due Friday!