



Lesson 3: Geoprocessing Programming Model

Basic Programming in ArcGIS with Python
Workshop

RS/GIS Lab, Utah State University
With Material from ESRI

Learning Objectives:

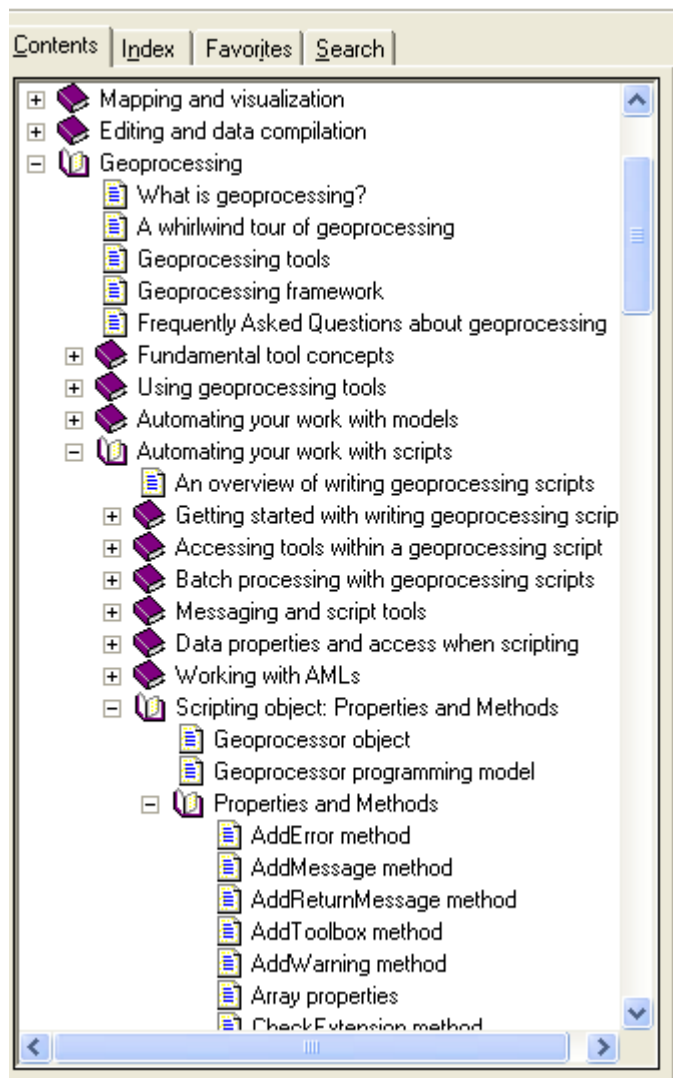
- Understand the role of the Geoprocessor (gp) Object
- Learn about Properties and Methods of the gp
- To be able to interpret the Geoprocessing Programming Model Diagram
- Learn that Properties and Methods return objects and standard data
- Understand why you would want to use a Describe object
- To be able to write a simple script using a Describe object

Lesson 3a: The Geoprocessor (gp) Object

- The Geoprocessor is an object (special variable) that has properties and methods
 - All tools are methods
 - All environmental settings are properties
- We access the methods and properties through the gp object

gp.Thin_sa	Tool	Method
gp.Buffer_analysis	Tool	Method
gp.extent	Env. Setting	Property
gp.workspace	Env. Setting	Property

- The Geoprocessor has properties and methods that are NOT environmental settings or tools:
 - gp.Toolbox
 - gp.Usage
 - gp.Exists
- There are other scripting objects that “sit on” the gp (i.e. are *returned* by using Methods or Properties of the gp.
 - Describe objects
 - Enumeration objects
 - Cursor objects
 - Create objects
- See the Geoprocessor Programming Model diagram



- The Geoprocessor Programming Model
- Diagram shows all objects, properties, methods
- Descriptions and Usage found in Desktop Help

arcgisscripting / GpDispatch

Properties
<ul style="list-style-type: none"> ■— MaxSeverity ■— MessageCount ■■ OverwriteOutput: Boolean ■— ParameterCount ■■ Toolbox
<ul style="list-style-type: none"> ← AddError (Message) ← AddMessage (Message) ← AddReturnMessage (Index) ← AddToolbox (Toolbox)

- Property

- Read only ■—

- Print gp.MessageCount

- Read/Write ■—■

- Print gp.Toolbox

- Methods ←

- Gp.AddError ("Error")

- Each box represents an object
- Objects grouped by functionality (colors)
- All objects “sit on” the geoprocessing object

Dynamic Methods and Properties

- Environment
- ← Tool (tool parameters)

Any tool from a referenced toolbox may be called as a method, while environment settings are properties.

- All tools and environment settings not listed in the diagram
- All can be called from the Geoprocessor
- Tools and env. settings depend on what's available (e.g. what toolbox has been added)

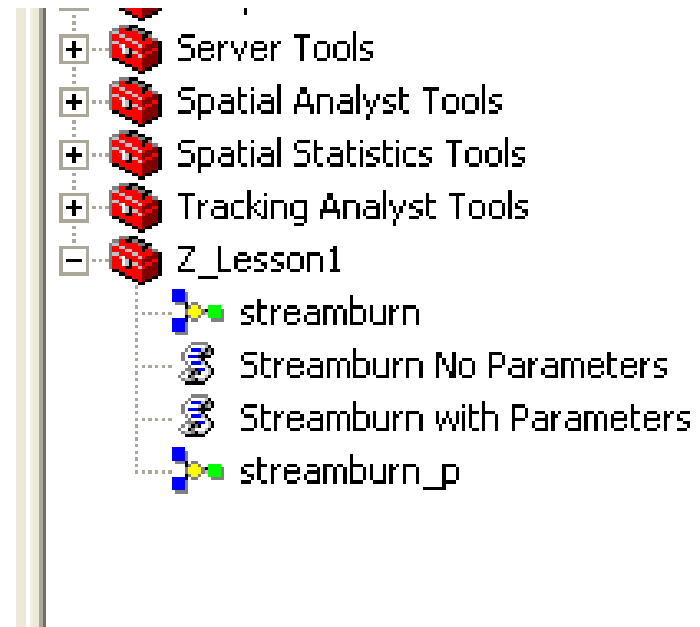
- System toolboxes available by default

```
Gp.buffer_analysis()
```

- If you call a tool from custom toolbox you must add it.

```
Gp.Addtoolbox  
(C:/john/Z_lesson.tbx)
```

- Addtoolbox method does not add toolbox from ArcToolbox, just makes tools available for the script



- Example of a Method:

- Check if data exists with the Exists method

```
If gp.Exists("flowline.shp"):
    Print "The flowline feature class exists"
```

```
If not gp.Exists("flowline.shp")
    Print "The flowline feature class does not exist"
```

- Example of a Property:

- Overwriteoutput property overwrites existing data

```
# will overwrite RBCreek.shp if it exists
gp.overwriteoutput = 1
gp.select_analysis ("flowline.shp", "RBCreek.shp", '
    "GNIS_Name" = \'Red Butte Creek \'')
```

Assignment 3a: Write a script using the Exists method and the Overwriteoutput property

- Write a short script that does the following:
 - Checks the current workspace (Lesson 3) for a dataset called landcover
 - If the dataset exists, then use the CopyRaster tool (Data Management) to make a backup copy called landcover_bak
 - If it doesn't exist, print a message that says “Dataset does not exist.”

Setting argument variables in a script

- Must import the system module first

```
Import arcpy, sys
```

- Set the variable using `argv` method

- The first variable is 1, second variable is 2, etc.

- The script is considered 0

```
# Set the argument variables
```

```
wks = sys.argv[1]
```

```
data = sys.argv[2]
```

```
# Specify the workspace
```

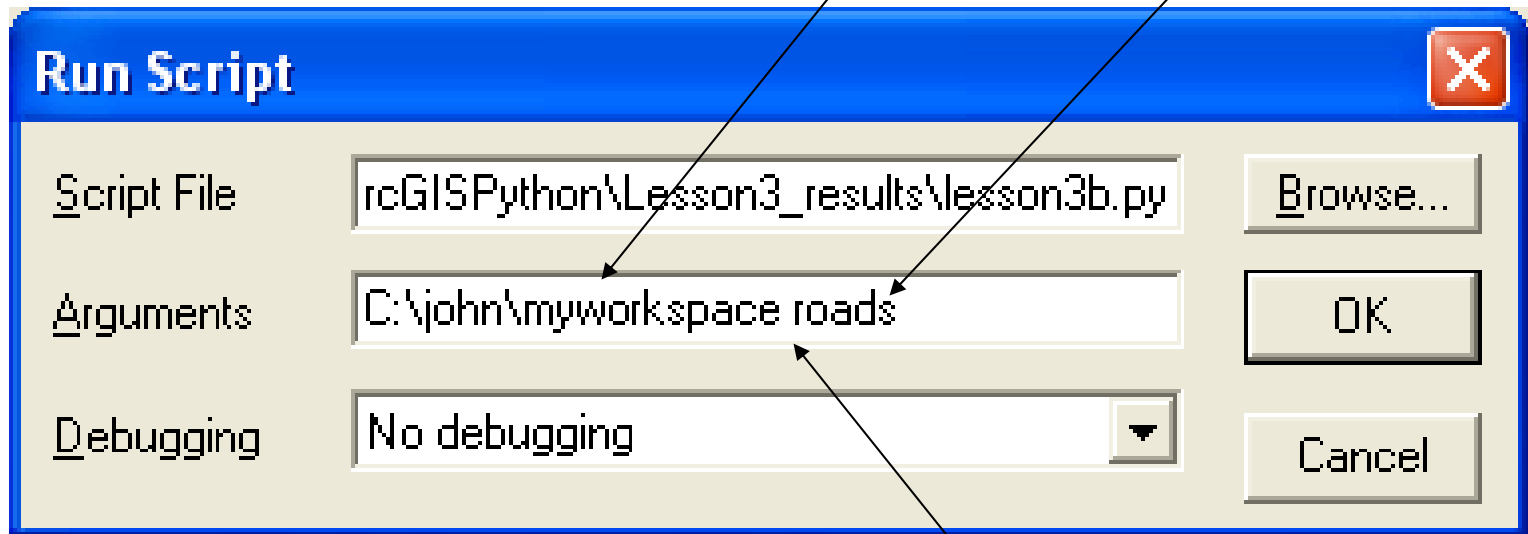
```
gp.workspace = wks
```

```
# buffer the dataset
```

```
gp.buffer_analysis = (str(data)+".shp", str(data)+"_buf.shp",  
    "100 feet")
```

Using Arguments in PythonWin

- Arguments must be:
 - Entered in order
 - Separated by a space



Space separating arguments

Assignment 3b: Write a script with Arguments

- Write a script that does the following:
 - Requires an argument for the name of a grid dataset
 - Checks the current workspace (Lesson 3) for the dataset.
 - If the dataset exists, then use the CopyRaster tool (Data Management) to make a backup copy called `<dataset_name>_bak`
 - If it doesn't exist, print a message that says "Dataset `<dataset_name>` does not exist."

Lesson 3b: Using Describe Objects

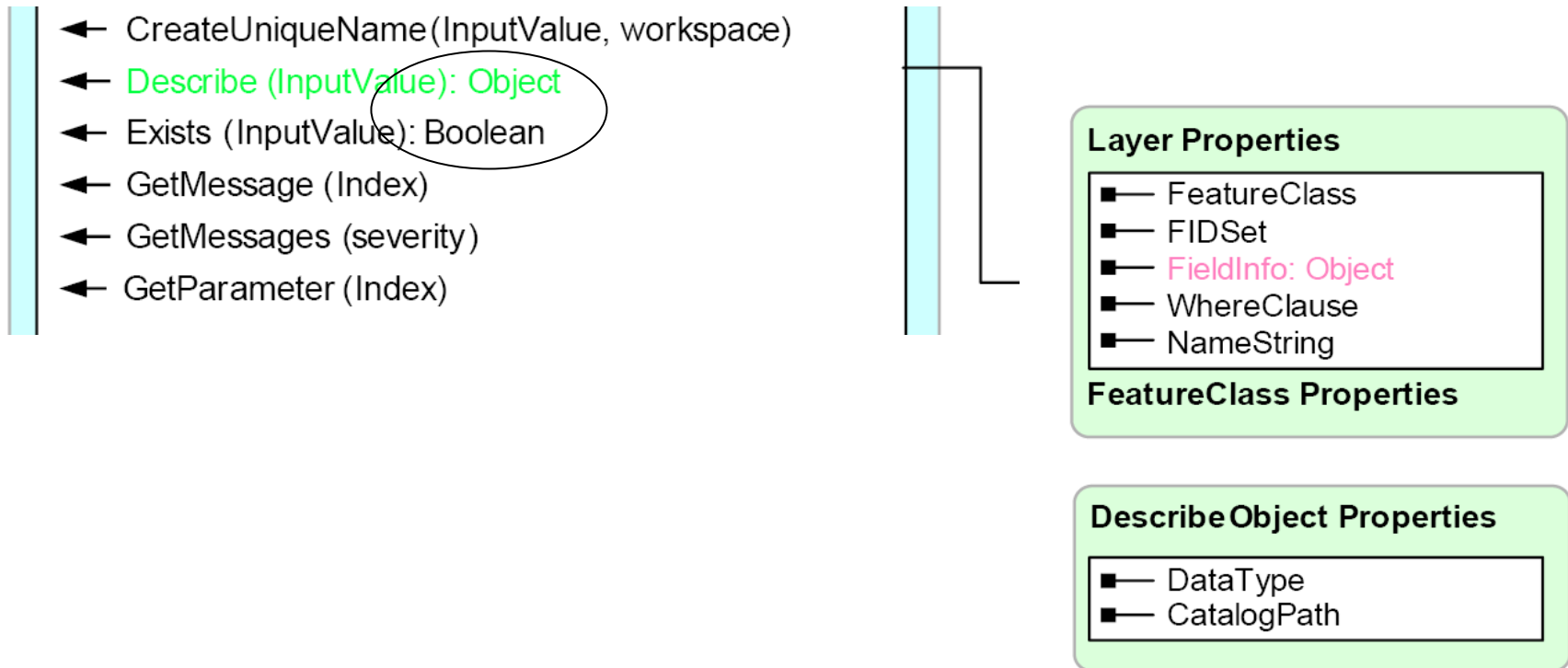
- Properties and methods return either standard data or objects
 - Most return standard data (a string or number (sometimes a boolean))
 - **Exists(InputChange): Boolean**, returns 1 or 0 and can be used in an If statement:

```
If gp.Exists("C:/john/flowline.shp"):  
    print "The feature class exists"
```

- Some return an object
- **Describe(IntputValue): Object**, returns an new object

```
dscDS = gp.Describe ("C:/john/flowline.shp")  
    print dscDS.Extent
```

- In the model diagram:
 - Type of data returned by property or method on right of colon
 - If not indicated, a string or numeric data is returned.



- The Describe method on the geoprocessor object returns an object (a *describe* object)
- The type of describe object returned, depends on the type data described (feature class, table, raster dataset, etc.)
- All describe objects provide descriptive information about the data being described
 - Extent
 - Spatial reference
 - Coverage tolerances
 - Cell size
 - Band count
 - Etc.

Dataset Properties

- DatasetType
- Extent
- SpatialReference: Object

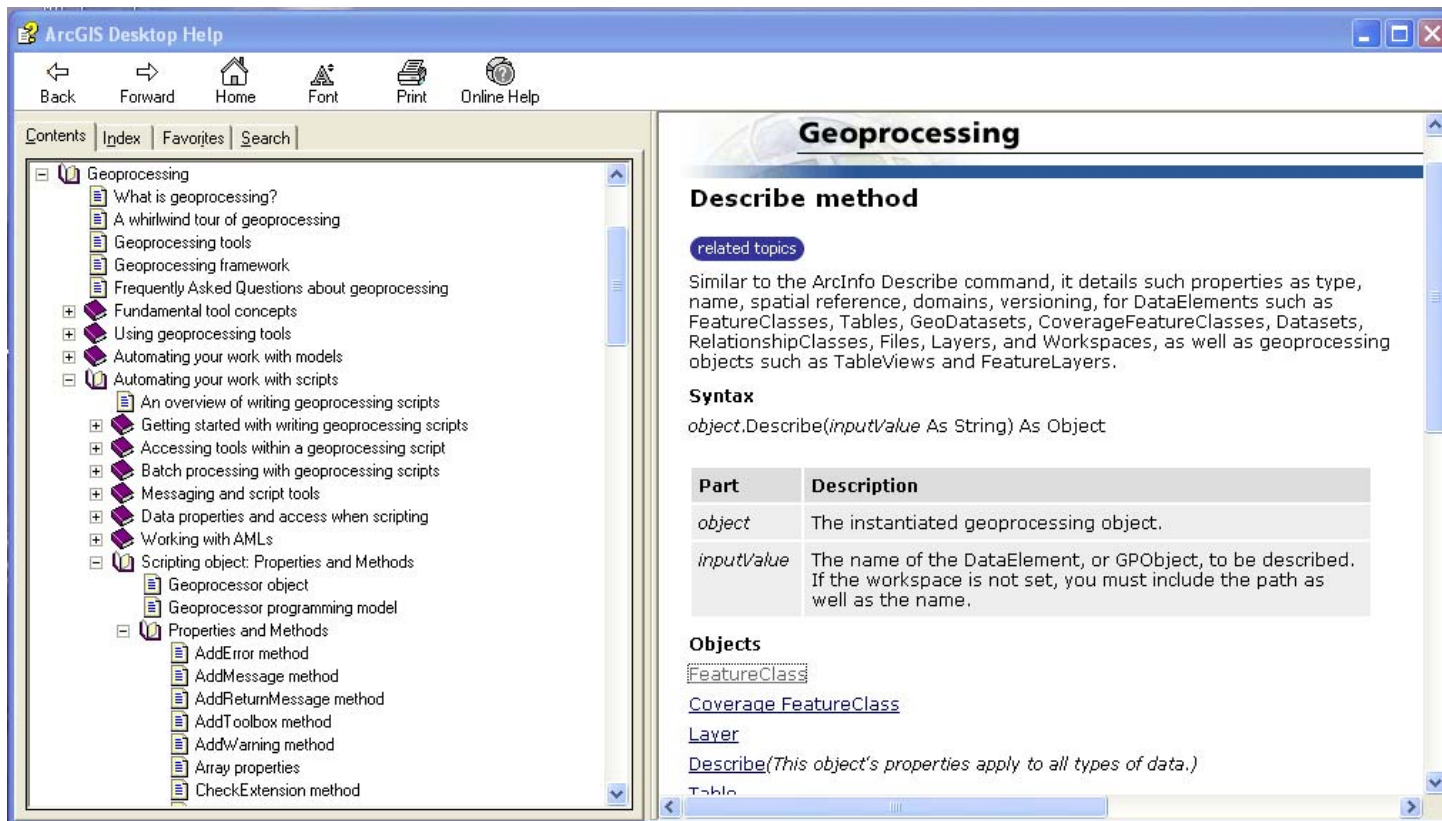
```
dscDS = gp.describe
print dscDS.Extent
```

Raster Band Properties

- Height
- IsInteger: Boolean
- MeanCellHeight

```
dscRB = gp.describe
print dscRB.Height
```

- Help for the Describe Method from ArcGIS Desktop Help
- Contents Tab > Geoprocessing > Automating your work with scripts > Scripting object: Properties and Methods> Describe method



Assignment 3c: Create Describe objects using the Describe method

- As a class: Use the model diagram to answer the following questions:
 - After the Describe method on the gp object, what method would be used to find out the Type of dataset we're describing?
 - What about the Spatial Reference name?
 - How could we determine the cell size of a raster dataset?
- Write a script that describes a dataset and prints this information to Python's interactive window.