Esri Developer Summit

March 26–29, 2012 | Palm Springs, California

esri.com/events/devsummit

Introducing the ArcPy Data Access module

Dave Wynne

Jason Pardy

Abstract

In this workshop, see highlights and demonstrations of the new data access module, arcpy.da, a Python module in ArcGIS 10.1 for working with data.

The module allows control of edit sessions and operations and improved cursor support and performance, provides functions for converting data to and from NumPy arrays, and supports versioning and replica workflows.

What is arcpy.da?

- A new arcpy module for working with data

1. Faster cursors
2. Edit session support
3. NumPy array conversion
4. Support for replicas, versions, subtypes, and domains

Cursors

- **Cursors provide record-by-record access**
  - **Are a workhorse for many workflows**

SearchCursor	Read-only access
UpdateCursor	Update or delete rows
InsertCursor	Insert rows

- **The ‘classic’ cursor model works, but…**
  - **Not fast enough**
  - **Bottleneck for some Python workflows**

Cursor Performance

- SearchCursor
  - Up to 30 times faster
- InsertCursor
  - Up to 12 times faster

Getting better performance

- Maximizing performance

1. Use only those fields you need

2. Use geometry tokens
   - “shape@xy”, “shape@centroid”, etc.
   - Asking for full geometry is expensive

Cursors

- arcpy.da cursors use lists and tuples
  - Row values are accessed by index

```
cursor = arcpy.da.InsertCursor(table, ["field1", "field2"])
cursor.insertRow([1, 10])
```

- ‘Classic’ cursors work with row objects
  - Row values are accessed with setValue/getValue properties

```
cursor = arcpy.InsertCursor(table)
row = cursor.newRow()
row.setValue("field1", 1)
row.setValue("field2", 10)
cursor.insertRow(row)
```

with statements

- Historically, needed to be careful with database locking and cursors
- arcpy.da Cursors support with statements
  - Guarantees close and release of database locks
  - Regardless of success or failure

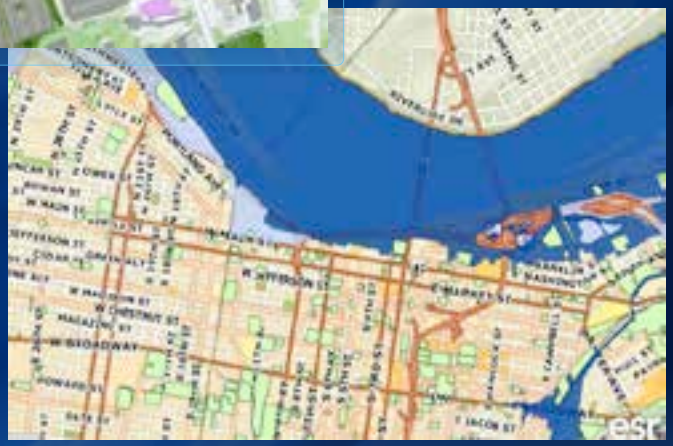
with statements

- for loop

```
cursor = None
try:
    cursor = arcpy.da.SearchCursor(table, field)
    for row in cursor:
        print row[0]
except Exception as err:
    raise err
finally:
    if cursor:
        del cursor
```

- with statement

```
with arcpy.da.SearchCursor(table, field) as cursor:
    for row in cursor:
        print row[0]
```



Demo: Cursors

Editor class

- Supports edit sessions and edit operations
- Edits are temporary until saved and permanently applied
- Can quit an edit session without saving changes

Editor class

Editor methods	
startEditing ({with_undo}, {multiuser_mode})	Starts an edit session.
stopEditing(save_changes)	Stops an edit session.
startOperation()	Starts an edit operation.
stopOperation()	Stops an edit operation.
abortOperation()	Aborts an edit operation.
undoOperation()	Undo an edit operation (roll back modifications).
redoOperation()	Redoes an edit operation.

- **If you've worked with edit sessions in ArcObjects, this will look familiar**

Sometimes you just need an edit session

- To edit feature classes that participate in a…
  - Topology
  - Geometric network
- Versioned datasets in ArcSDE geodatabases
- Some objects and feature classes with class extensions

With statements (part deux)

- Editor also supports with statements
  - Handle appropriate start, stop and abort calls for you

```
with arcpy.da.Editor(workspace) as edit:
    #<your edits>
```

Open an edit session and start an edit operation

Exception—operation is aborted, and edit session is closed without saving

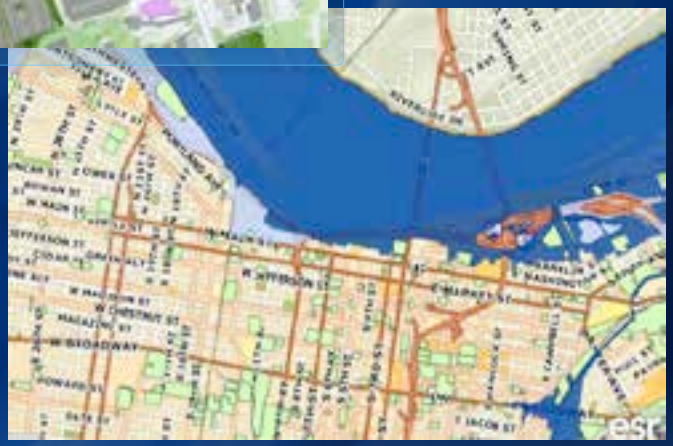
No exceptions—stop the operation and save and close the edit session

Demo: Editor class

New list functions

- **More list functions (and objects) to support workflows**

Functions	
ListDomains	Lists the attribute domains belonging to a geodatabase.
ListReplicas	Lists the replicas in a workspace.
ListSubtypes	Return a dictionary of the subtypes for a table or feature class.
ListVersions	List the versions in a workspace.



Demo: arcpy.da list functions

NumPy support

- NumPy is a 3rd party Python library for scientific computing
  - A powerful array object
  - Sophisticated analysis capabilities
- ArcGIS 10.0 added support for converting rasters to and from numpy arrays
  - RasterToNumPyArray
  - NumPyArrayToRaster

NumPy functions

- **At 10.1, arcpy.da provides additional support for tables and feature classes**

Function	
FeatureClassToNumPyArray	Convert a feature class to an array
TableToNumPyArray	Convert a table to an array
NumPyArrayToFeatureClass	Convert an array to a Feature Class
NumPyArrayToTable	Convert an array to a Table
ExtendTable	Join an array to a Table

Export to NumPy

- Can convert tables and feature classes into numpy arrays for further analysis

```
import arcpy
import numpy

in_fc = "c:/data/usa.gdb/USA/counties"
field1 = "INCOME"
field2 = "EDUCATION"

array = arcpy.da.FeatureClassToNumPyArray(in_fc, [field1, field2])

# Print correlation coefficients for comparison of 2 fields
print numpy.corrcoef((array[field1], array[field2]))
```

Import from NumPy

- Take the product of your work in numpy and export it back to ArcGIS

```
array = numpy.array([(1, (471316.3, 5000448.7)),
                     (2, (470402.4, 5000049.2))],
                    numpy.dtype([('idfield', numpy.int32),
                                 ('XY', '<f8', 2)]))

SR = arcpy.Describe(template).spatialReference

# Export the numpy array to a feature class using the XY
#  field to represent the output point feature
arcpy.da.NumPyArrayToFeatureClass(array, outFC, ['XY'], SR)
```

Demo: Exporting to NumPy

Thank you!

- **ArcGIS Python resource center**
  http://resourcesbeta.arcgis.com/en/communities/python/

- **Twitter**

  http://twitter.com/#arcpy

esri