**Python Mosaic Script**

**Jennifer Reinke**

**5.14.2015**

**Objective:**

To create a mosaic dataset through Python scripting from four georeferenced historical aerials

**Introduction:**

I created a trial mosaic for the Center of Urban and Regional Affairs (CURA) department at the University of Minnesota-Twin Cities. CURA intends on building historical mosaics for the entire Twin Cities Metropolitan Area to be hosted online. Four photographs from 1956 showcasing Minneapolis and St. Paul completed the mosaic set. The aerial rasters must be georeferenced prior to using the script. The aerials provided were in jpeg format and I kept that format throughout the code and filtered for only jpgs in the script. Therefore, when using this code other than for this exercise, make sure rasters are in .jpg format or change the extension format in the code for individual use.

**Open Python’s Integrated Development and Learning Environment (IDLE):**

IDLE is the user interface to write and run the Python code.

**Import System Modules:**

A module is a file composed of Python code in forms of definitions, statements, functions, classes, and variables that is stored. To begin the scripting process, it is important to import all the modules that you will need to reference or call later in the code. I typically import the same 3-4 modules while working with ArcGIS software and any other modules required. They are listed below:

**\*import arcpy**: It is required in order to execute python scripts in junction with ArcGIS because the package was based on the arcgis scripting module.

\***from arcpy import env:** The below example imports the env class consisting of the geoprocessing environments.

**\*import sys:** This module provides a number of functions and variables that can be used to manipulate different parts of the Python runtime environment.

**\*import os:** The os module provides a portable way of using operating system dependent functionality, which allows one to interface with the underlying OS that Python is running on.

**\*import numpy:** Numpy is often used for scripts employing scientific computation (Not necessary for this script, therefore omitted).

**Step 1: Import these Python modules**

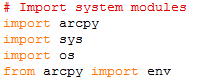


Figure 1.

**Workspace:**

To set-up a workspace, choose a location or workspace where the files are to be held, maintained, and created in. I chose to put my workspace in the *Temp folder* because most computers have the same generic Temp folder on the C: drive and I wanted accessibility for the end-users. Usually, there is a folder titled, “Workspace” within the Temp folder. If the Temp or Workspace folder is missing, right-click > new > folder and rename it to Temp and/or Workspace. The workspace path should look like the Figure 2. Place the georeferenced rasters, their supplementary files, and the inputFiles folder from the folder ***reinj05\_files*** *(ancillary files)* into the “Workspace” folder (i.e. Copy and paste everything within the folder, reinj05\_files, into the Workspace folder). Confirm that the jpeg files are there since the code selects jpeg raster files to create the mosaic. Python is case sensitive; confirm that the “T” in Temp and the “W” in Workspace are capitalized.

**Step 2: Define Workspace**



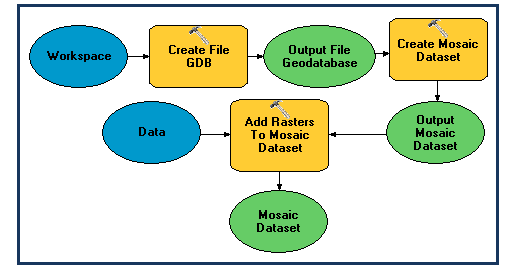
Figure 2.

Set paths to the input folder (which might later contain additional rasters) and an output folder.

**Step 3: Set Local Variables**



Figure 3.



**Workflow:**

My workflow is similar to figure 4. Create a file geodatabase base > empty mosaic dataset > list raster files in workspace >

add rasters to mosaic dataset.

Figure 4.

**Procedure:**

The first step of the script, create a file geodatabase because a mosaic dataset cannot be created without being contained within a file geodatabase (FGDB). For each section, the variables are set prior use in the function if it was not already defined.

**a.)** I created the geodatabase-name variable (gdbName) and had it equal to CURA.gdb in order to create a FGDB that denotes what it is for (CURA) and where I can place the mosaic dataset.

**b.)** Write a conditional statement using **if/else** to check **if** a file geodatabase of the same name exists in the workspace and if so, verify the FGDB name. **Else,** when the workspace did not have the stated FGDB, the script created a FGDB with the given variable name as seen in figure 5. I also informed the user with a print statement to indicate whether the FGDB existed and/or the code executed.

**Step 4: Create File Geodatabase**

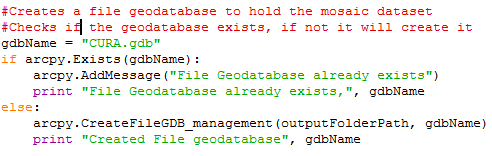


Figure 5.

**c.)** Create an empty mosaic dataset within the file geodatabase. Once again, in order to create a mosaic dataset, it needs to be within the file geodatabase or the parameters will not be met. One also needs an empty mosaic dataset as to facilitate adding batch rasters or single rasters to the mosaic dataset.

**d.)** Similarly to step 4, use an **if/else statement** to check if the mosaic is already present in the workspace and to create a mosaic if it does not exist. Continue to inform that user about the status. Remember to set variable parameters that were going to be used in the conditional.

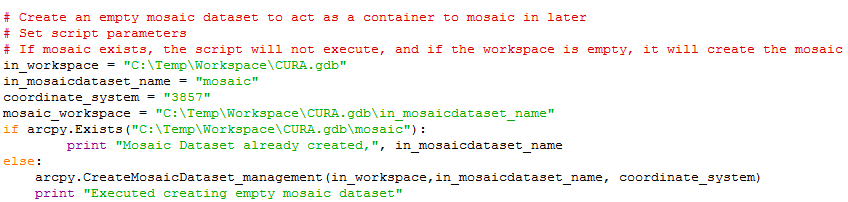
**e.)** Set the workspace to the file geodatabase (CURA.gdb) within the set workspace environment ("C:\Temp\Workspace\CURA.gdb").

**f.)** Assign a variable in\_mosaicdataset\_name for the mosaic dataset (named mosaic).

**g.)** Set the coordinate system (the coordinate system should match the spatial reference defined by the raster during georeferencing). I used Web Mercator by its spatial reference number, 3857, because the mosaic was destined for the web and I previously georeferenced the images as Web Mercator in ArcMap.

**h.)** Use the syntax (can be viewed in ArcHelp Resources) for the “Create a Mosaic Dataset”.

**CreateMosaicDataset\_management (in\_workspace, in\_mosaicdataset\_name, coordinate\_system, {num\_bands}, {pixel\_type}, {product\_definition}, {product\_band\_definitions})**

Figure 6.

**Step 5: Create an Empty Mosaic Dataset**

**i.)** In order to see what jpg raster images are in the Workspace, print a list of all the files within the folder with a .jpg extension. This helps to see what rasters will/could be added to the mosaic dataset. I used the ListRasters script with a wild card (\*) and (JPG) to select for jpegs. I also used a print statement so that the files would be seen once the script was running. It will also save the list of rasters to the variable “rasters”, which will be used as a parameter in the last step for adding rasters to a mosaic dataset.

**Step 6: List Rasters**

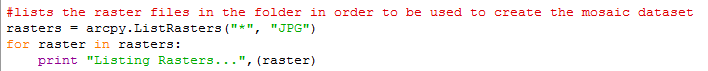


Figure 7.

The final section of the script adds the georeferenced rasters from the workspace environment to the mosaic dataset. I used the sample parameters from ArcHelp Resources, which requires three parameters and offers optional parameters. I used a combination of the attributes. The first three parameters in figure 8 shows the required attributes, which are the **target mosaic, the type of raster format, and the rasters** themselves. The rasters are stored in the variable “rasters” where I listed all the rasters in step 6. Then I used the “#” to act as null/ a placeholder so I could make sure to keep the correct number and order for the input parameters. I also added the aforementioned 3857 spatial reference to force the mosaics into this projection and made sure to select jpgs for the file format. The second half of the function as seen in figure 9, shows the remaining attributes. Most of them are null, but I wanted to construct pyramids for the larger mosaics in mind. Also, I wanted to exclude duplicates, by adding “EXCLUDE\_DUPLICATES” to the parameters. This provides a check to see if a raster with the same name in the same workspace will be added, and will not permit the raster to be added.

**Step 7: Add Rasters to Mosaic Dataset**

Figure 8.



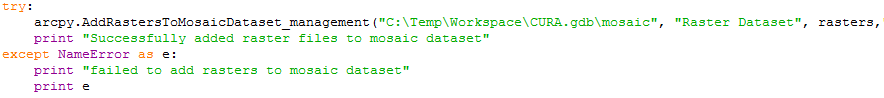
Figure 9.

Figure 10. (Code to add rasters to mosaic dataset)

**Step 8. Verify Mosaic Dataset**

1. Open ArcMap
2. Connect to Workspace Folder and navigate to the CURA FGDB mosaic data set (Figure 11)
3. Open Mosaic and see if all rasters are present

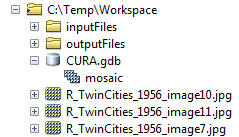
****

Figure 11. ArcCatalog folder connection to CURA mosaic dataset

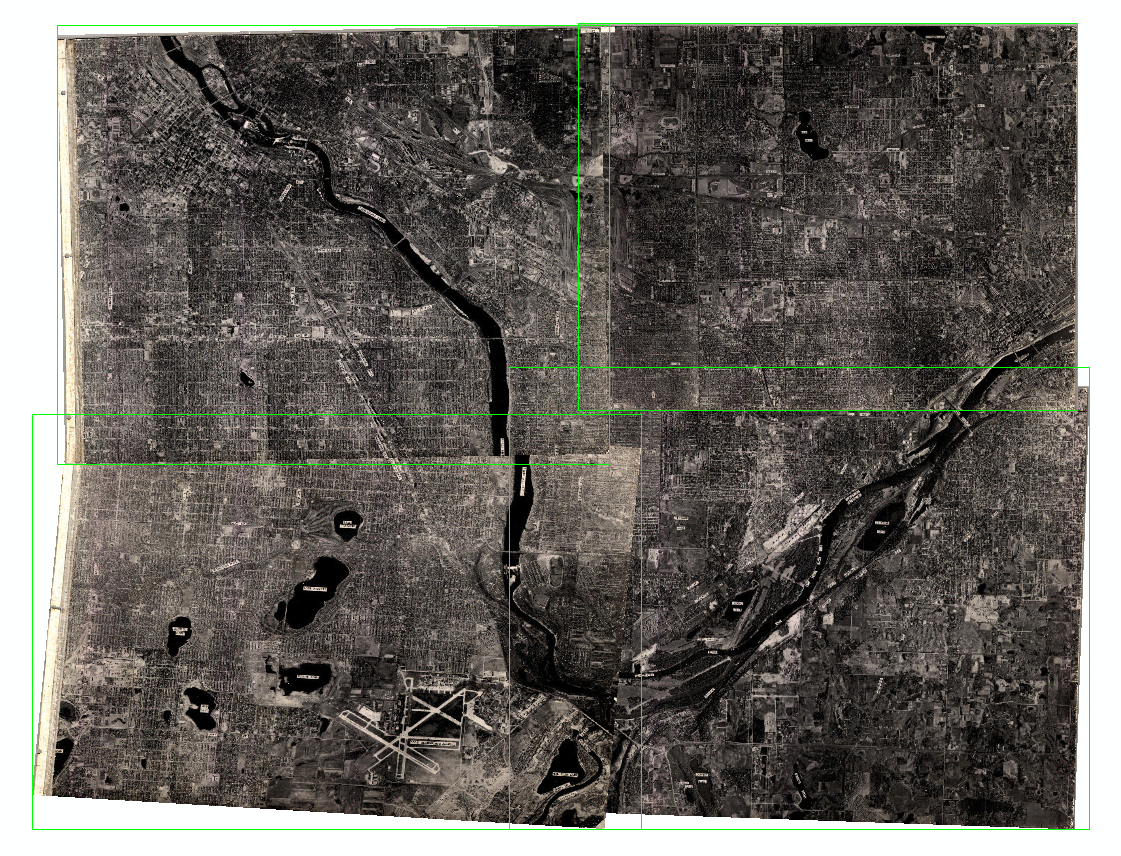


Figure 12. Image of mosaic created after running script

**References:**

http://resources.arcgis.com/en/help/main/10.1/index.html#//00170000008n000000

http://resources.arcgis.com/en/help/main/10.1/index.html#//001700000085000000

<http://www.tutorialspoint.com/python/python_modules.htm>

<https://docs.python.org/2/tutorial/modules.html>