Archelper User Guide

**The archelper module was designed to help users work with shapefiles and create maps from CSVs using graduated symbology. There are a few basic functions here that can automate the creation of any number of maps using the arcpy module and ArcMap templates (MXD documents). This tool can iterate through multiple CSVs, columns, and MXDs to create a large number of maps but requires the user to build MXD templates using the ArcMap windows application. This module is designed to work specifically with the ‘graduated symbology’ ArcGIS functionality.**

**Archelper relies heavily on the ESRI arcpy module. This module requires some knowledge of ArcGIS, but does not require knowledge of the arcpy module. If the user is comfortable in python, they should inspect the source code and learn to use the ArcPy module as it has a lot of great mapping functions. Users must have windows and ArcGIS 10.1+ installed on their machine and work with the ArcGIS install of python.**

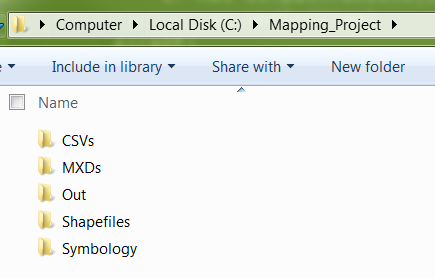
**For any questions or feature requests, please contact Shane Armour at shane.armour@rms.com or 510-713-3801.**

# ****Prerequisites****

* **ArcGIS 10.1 +**
* **The python version associated with the ArcGIS install. ArcGIS 10.3.1 uses Python 2.7.8 and Numerical Python 1.7.1**
* **This module imports arcpy by default, if there are any errors, please check the following link: http://resources.arcgis.com/en/help/main/10.2/index.html#//002z00000008000000**
* **A windows version compatible with ArcGIS 10.1+**
* **The archelper module**
* **MXD templates, shapefiles, and symbology specified in the set-up section below.**

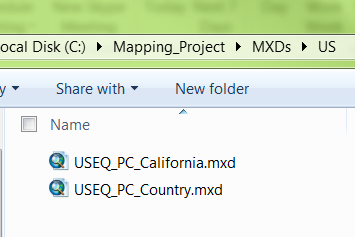
# MXD Template Set-up

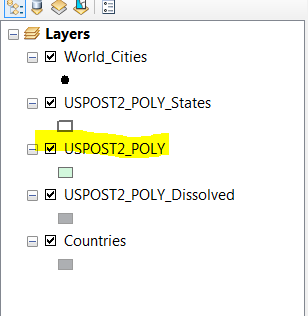
* Create a folder structure on the C drive called C:\Mapping\_Project with the following subfolders: MXDs, Out, Symbology, CSVs, Shapefiles. Alternatively, you can run the archelper.create\_dir() function to set up the folder structure.



* **Add the shapefiles to the shapefile folder and create MXD documents that point to the shapefiles in there. Make sure the shapefile you want to use for mapping has a unique name in the MXD template.**
* **Alternatively, you can use a file geodatabase as a replacement for shapefiles. They have less constraints than shapefiles. Here’s a list of shapefile limitations you NEED to be aware of when mapping. File geodatabases do not have these limitations.** 
  1. **Column headers must not start with numbers**
  2. **Column headers have a 10-character limit, thus make sure the first 10 characters of each column are unique otherwise archelper will not be able to handle multiple columns.**
  3. **All values that are null will be replaced with a 0 when using the field type double when adding columns and values to the shapefile, etc.**[[1]](#footnote-1) **When looking at value changes between numbers, this can be a huge issue as a null value will be shown as zero change.**

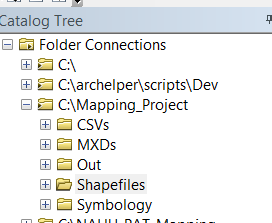
### **Multiple MXDs**

* To work with multiple MXDs in the same function, make sure each MXD has the same source layer and symbology type.
* MXD folder setup: Place multiple MXDs in the same folder and specify the paths of the MXDs in a python list to iterate through each MXD. Examples will be shown below. 
* If specifying multiple MXDs in a single create map command, make sure each one has the same mapping layer. In this example, USPOST2\_POLY will have the postcodes that should be mapped. Each MXD needs to have the same layer, but they do not need to be in the same order. **Do not have duplicate layer names**. Currently, the module is only programmed to work with one dataframe.

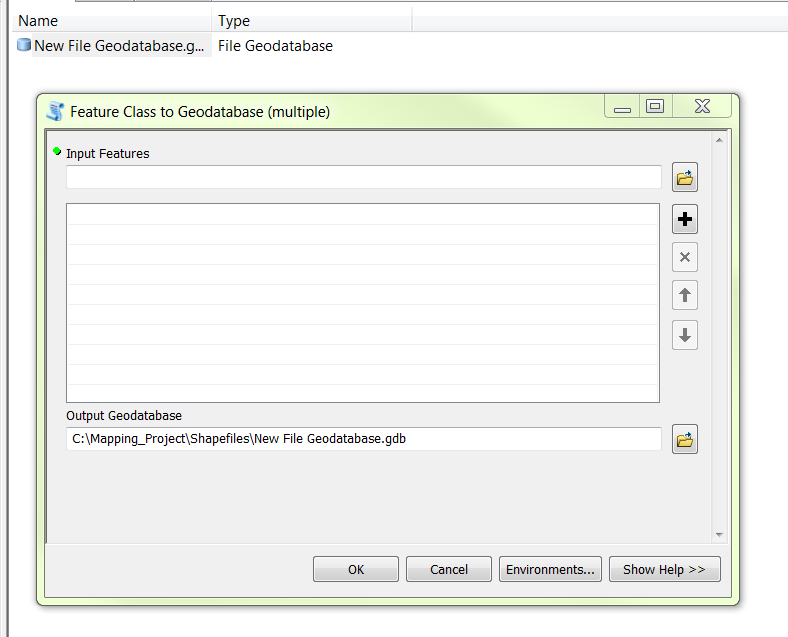


### **Working with a file geodatabase**

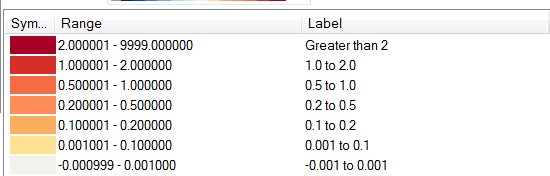
* To create a file geodatabase open Arc Catalog and navigate to the shapefiles folder. Right click on the folder and click new> file geodatabase.



* The file geodatabase can hold all of your shapefiles in one place. Right click the file geodatabase and select the import option. If you want to import multiple shapefiles simultaneously click the Feature class to geodatabase (multiple) option, otherwise select the single option. Navigate to the shapefiles you would like to use in the MXD template and import them. **Take note** of the path at the bottom. When referencing the shapefiles in the future, use that path plus the name of the new table (by default it will have the same name as the shapefile).



# Symbology Set-up

* To create the symbology, a separate layer needs to be created from a shapefile with values that are not being impacted from the script.
* The best option to keep they symbology layer active is to make a second copy of a shapefile, add data that reflects the symbology break values, and then right click the layer in a new MXD and save it as a layer file. The symbology source layer and the symbology file should be kept independent of the MXD template and mapping data that is supposed to be used.
* **Important:** When choosing data to add to the symbology, make sure the upper and lower limits of the data actively reflect all possible mapping options. If planning to make a map with an upper limit of 2, choose the max data value to be 9999 (or even greater 999999999) so that all values greater than 2 will be included in the map. If the max value in the data you choose for the symbology is 3, but you have values in the mapping data greater than 3, those values will not be show correctly in your maps.
* **Example:** 

# Archelper Setup

Copy the entire archelper folder to your local Python lib folder. For default ArcGIS installations, the folder will be here: C:\Python27\ArcGIS10.3\Lib. When using windows, make sure the environment variables PATH contains that folder path as well otherewise you will not be able to import modules.

# Archelper Functions

**create\_dir**()

Creates an empty folder directory on the C drive called Mapping\_Project. Use this folder for mapping projects. under C:\Mapping\_Project a subdirectory will be created with the following folders:  
MXDs: Put the mxd templates here.  
Out: All maps will go to this folder by default.  
Symbology: Create symbology layer files in this folder. This is not necessary, but is recommended.  
CSVs: You can loop through all CSVs in a folder. Like symbology, CSVs can go anywhere, but placing them in this folder is convenient.  
Shapefiles: You can use either a folder of shapefiles or a file geodatabase with shapefiles. The MXD templates must point to the corresponding shapefiles used in mapping. You can use this folder for convenience if there are many shipefile files.

**create\_workspace**(name)

Checks if a .gbp workspace exists in the mapping\_project folder. This script returns the path of the workspace. The file geodatabase workspace replaces shapefiles in ArcGIS10.1 and later. You can create a file geodatabase using ArcCatalog as well. Once you create a filegeodatabase, import shapefiles into it using the Table to Geodatabase tool in the "ArcToolbox>Conversion Tools>To Geodatabase"  section.

**csv\_checkmissingshpvals**(inputfile, joincol, shapefile, shapefileheader, headers=True, filedelimiter=',')

Checks to see if any join column values in the inputfile are missing in the  shapefile or geodatabase table. The shapefile/geodatabase table can also be a geodatabase table. Returns a list of missing inputfile values in a shapefile that wil not be mapped. Headers is True by default.  
   
Default file delimiter is comma delimited but tab delimited can be used as well by specifiying '        '.

**file\_getcols**(inputfile, filedelimiter=',')

Returns a list of the inputfile headers.  
Default file delimiter is comma delimited but tab delimited can be used as well by specifiying '    '.

**file\_jointable**(inputfile, workspace, delimiter)

Imports the file to an arcgis geodatabase workspace and returns a string with the workspace and table name. The first row of data in a csv/txt file will be used for column headers.  
   
File format options are csv, txt, dbf, xls, xlsx, OLE, INFO, VPF, personal/file/SDE geodtabase.

**file\_sort**(inputfile, colindex=0, reverse=False, headers=True, filedelimiter=',')

Sorts an inputfile based on the colindex. If reverse is True, the values will be sorted in reverse index. Common file delimiters are comma ',' and tab '       '.  
   
Default file format is comma delimited

**map\_create1**(mxds, shapefile, mapcols, symbology, labels=False, prefix=None, perchange\_labels=False, LC\_labels=False, mapresolution=600, nodatavalue=-9999)

This function will create maps for all mxds specified and all fields in the mapcols list. Shapefile can be a shapefile or geodtabase table. The prefixed symbology options are 'Percent\_Change' and 'Diff\_LC' or importy symbology from  your own layer file. Labels can be set to True or False. If Diff\_LC or Percent\_change is specified, labels will be formatted accordingly. Use the prefix variable to add a prefix to the output file name. Using a prefix is strongly recommended when mapping multiple CSVs.

**shp\_addcols**(shapefile, cols, datatype)

Adds each column in the list of cols to the shapefile or geodatabase table. Columns can only have 10 characters in a shapefile. All columns added will be given the same datatype. If you try to add a duplicate column that is already in the shapefile, the existing duplicate column will be deleted.  
   
    Possible fields types:  
   
    TEXT Any string of characters.  
    FLOAT  Fractional numbers between -3.4E38 and 1.2E38.  
    DOUBLE  Fractional numbers between -2.2E308 and 1.8E308.  
    SHORT  Whole numbers between -32,768 and 32,767.  
    LONG  Whole numbers between -2,147,483,648 and 2,147,483,647.  
    DATE Date and/or time.  
    BLOB Long sequence of binary numbers. You need a custom loader or viewer or a third-party application to load items into a BLOB field or view the contents of a BLOB field.  
    RASTER Raster images. All ArcGIS software-supported raster dataset formats can be stored, but it is highly recommended that only small images be used.  
    GUID Globally unique identifier.  
   
    NOTE: Geodatabase fields will be nullable.  For shapefiles, null values will be converted to the following value based upon  
   
Important: When adding values to a shapefile, Null values will be substituted.  
   
    Shapefile null value substitution-  
        Number  replaced by 0  
        Text replaced by " "  
        Date replaced by 0 but displays <null>  
   
    For more info visit: <http://desktop.arcgis.com/en/desktop/latest/manage-data/shapefiles/geoprocessing-considerations-for-shapefile-output.htm#GUID-A10ADA3B-0988-4AB1-9EBA-AD704F77B4A2>

**shp\_getcols**(shapefile)

Returns a list of shapefile or geodtabase table columns.

**shp\_joincsv**(csvfile, shapefile, shapefilejoincol, csvjoinindex, csvstartfield, csvfieldtype='double', filedelimiter=',', csvendfield=None, usecustomlabel=False, customnodatalabel=-9999)

This function manually joins the CSV to the shapefile and does not use geodatabase tables. This method should be easier and faster in most cases. In the CSV, the join column must be before the columns with mapping values. This code will map all fields from the mapping column onward (to the right). Headers of the csv will be used as field names and have a 10 character limit. Field names must not start with numbers either.  
CSV field type can be 'double' or 'text'. If the fieldtype added is a double and there is no value, a a custom value of -9999 by default will be added. Arcpy automatically converts nulls to the values shown below, which is misleading. Use the customnodatalabel value and the symbology settings to represent no data. The default is -9999.  
   
Shapefile null value substitution- Use a feature class to avoid this  
    Number 0  
    Text " "  
    Date 0 but displays <null>

**shp\_jointable**(injointable, injoinfield, combinedtable, combinedtablefield, fields)

Joins a workspace table to another workspace table or shapefile. The workspace table is generated by csv\_jointable(). The jointable and shapefile variables should include the full file path ie. 'C:/path/to/shapefile.shp' or 'c:path/to/workspace.gbp/tablename'

**shp\_removecols**(shapefile, cols)

Removes fields from shapefile or geodatabase table specified in the cols list. Columns can only have 10 characters if a shapefile is being edited.

# Example Scripts

### Set Up Environment

### Check For Any Issues With Mapping Data

### Create Maps

1. """This script will create postcode maps with the specified MXDs and symbology paths."""
2. **import** glob
3. **import** os
5. #The layer in the mxd you want to map. Point to the full table path
6. shpfiletable = r"C:\mapping\_project\JPTY\_PAT\_Data.gdb\Japan\_CityWard\_2011"
7. csvjoinindex = 0 #The index (starting at 0) of the join column in the CSV file
8. csvstartfield = 1 #The index (starting at 0) of the mapping columns in the CSV file. All columns starting at the index (and right) will be added to the shapefile
9. shpfilejoincol = "CTYCODE" #The column header in the map shapefile that should be joined
11. #glob.glob returns a list with full paths of the CSVs used for mapping
12. csvfiles = glob.glob(r"C:\Mapping\_Project\CSVs\\*.csv")
14. **for** f **in** csvfiles:
15. #UPDATE THIS VALUE TO GET CORRECT MXDs. \*.mxd will return all MXDs in the folder.
16. mxds = glob.glob(r'C:\Mapping\_Project\MXDs\\*.mxd')
18. #this section joins the client CSV file to the shapefile
19. mapnameprefix = os.path.basename(f).rstrip(".csv")
20. mapping\_files.shp\_joincsv(f, shpfiletable, shpfilejoincol, csvjoinindex, csvstartfield)
22. mappingcols = ["TIV"] #update to correct column header of the csvfile
23. symbfile = r"C:\Mapping\_Project\MXDs\Symbology\TIV\_symb.lyr" #point to symbology you want to use
24. mapping\_files.map\_create1(mxds, shpfiletable, mappingcols, symbfile, prefix = mapnameprefix)
26. mappingcols = ["RL\_GU\_AAL","HD\_GU\_AAL"]
27. symbfile = r"C:\Mapping\_Project\MXDs\Symbology\AAL\_25mto1b.lyr"
28. mapping\_files.map\_create1(mxds, shpfiletable, mappingcols, symbfile, prefix = mapnameprefix)

### Add Or Remove Shapefile Columns

1. http://desktop.arcgis.com/en/desktop/latest/manage-data/shapefiles/geoprocessing-considerations-for-shapefile-output.htm#GUID-A10ADA3B-0988-4AB1-9EBA-AD704F77B4A2 [↑](#footnote-ref-1)