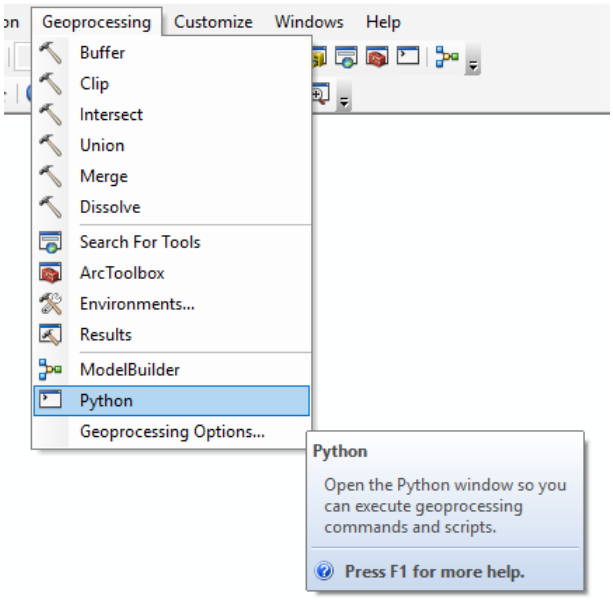
1 ArcPy module

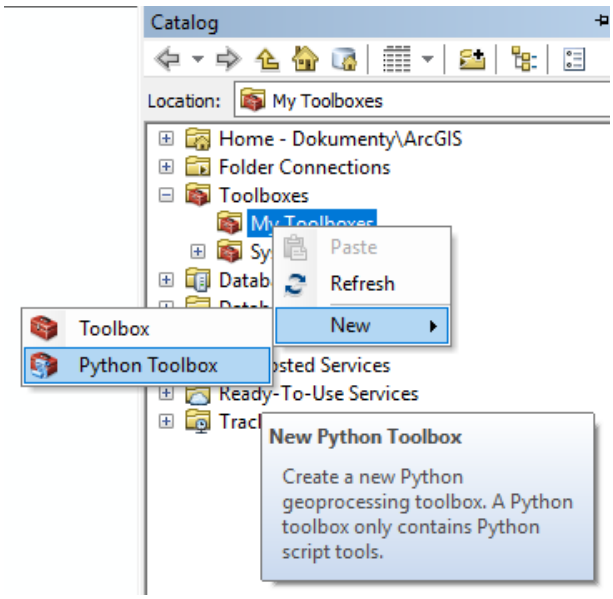
ArcGIS software comes with a Python module which provides its functionality for use in Python scripts. You can therefore use ArcPy with any other Python libraries and greatly enhance ArcGIS capabilities. You can use Python console directly in ArcMap (and test some functions for example):



Recommendation

ArcPy introduction: `http://desktop.arcgis.com/en/arcmap/latest/analyze/arcpy/a-quick-tour-of-arcpy.htm`

Or you can prepare a whole script with desired functionality (e.g. crop all shapefiles in a folder to a given `layer`, perform some geospatial operations on top of them and generate a text file with a report) and add it to ArcGIS as a tool you can use from the Toolbox:



Do this for tasks you might do frequently for different datasets or tasks you want to automate. You can then use this toolbox directly from ArcGIS with layers/files/directories you can specify before running the tool.

Another option is to use ArcPy outside of ArcGIS in a script. You could for example make a script for collecting data periodically and make it run at 3 A.M. every day, collect data from a remote location (some server), save it, analyse it and prepare a report. Scientists could use this to pinpoint some events that are of interest to them.

Note

In ArcGIS Python console, folders in paths have to be *escaped* – write `"\\"` or `"/"` instead of `"\"`, e.g. `"c:\\PGI\\loadAllShapefiles.py"` or `"c:/PGI/loadAllShapefiles.py"`

You use `ArcPy` like any other Python module, here is an example of a script you can use from ArcGIS Python console to load all shapefiles in a specified folder:

```
import arcpy
from arcpy import env

def loadAllShapefiles(path):
    # set workspace path to the one we will be searching SHPs in
    env.workspace = path
    # get a list of all SHPs in the folder
    fcList = arcpy.ListFeatureClasses()
    print fcList
    # sets mxd to the currently opened document
    mxd = arcpy.mapping.MapDocument("CURRENT")
    # sets df to the first data frame
    df = arcpy.mapping.ListDataFrames(mxd, '')[0]
    # iterates over every SHP name in the folder
    for fc in fcList:
        # get the whole path to the SHP
        pathToFC = env.workspace + "/" + fc
        # create a new layer from the SHP
        newLayer = arcpy.mapping.Layer(pathToFC)
        # add the new layer to the bottom of the data frame
        arcpy.mapping.AddLayer(df,newLayer,"BOTTOM")
    # refresh Table of contents and the active view (so the layers appear
    ↪ immediately when the script finished)
    arcpy.RefreshTOC()
    arcpy.RefreshActiveView()
```

It is then called like this: `>>> loadAllShapefiles("c:\\PGI\\")`

An example of a script you run from PyZo, other IDE or command line that clips all features in a specified directory:

```
import arcpy
import os
from arcpy import env
# set workspace (where input SHPs are located), output location (where
    ↪ clipped SHPs will be stored) and the clip feature class
env.workspace = "c:\\PGI\\in"
outputDir = "c:\\PGI\\out"
clipFC = "c:\\PGI\\jmk.shp"
# get all feature classes in a directory
fcs = arcpy.ListFeatureClasses()
for fc in fcs:
    # determine the full name of the output shapefile
    outFC = os.path.join(outputDir, fc)
    # execute the clip
    arcpy.Clip_analysis(fc, clipFC, outFC)
```