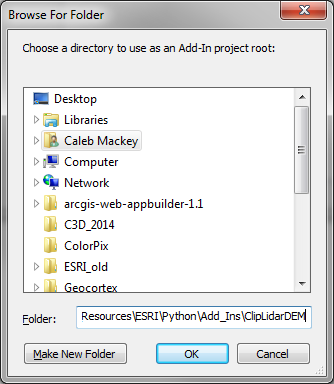
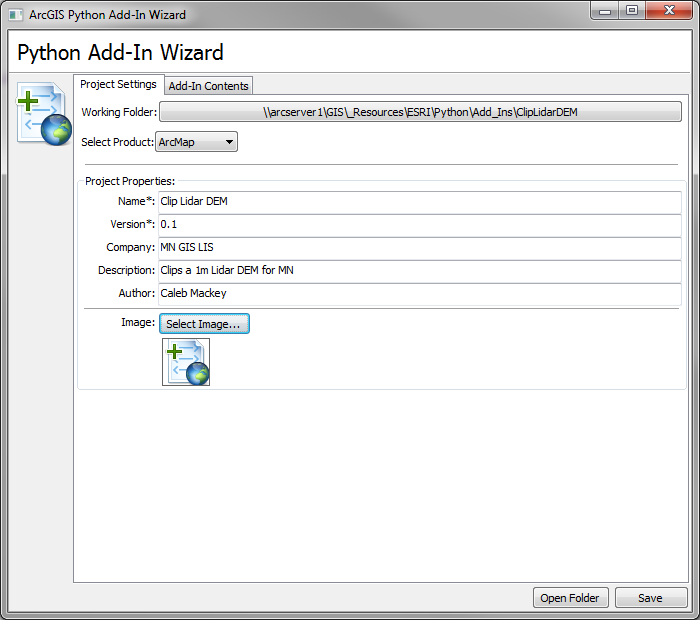
# Python Add-In Exercise: LiDar Clipper

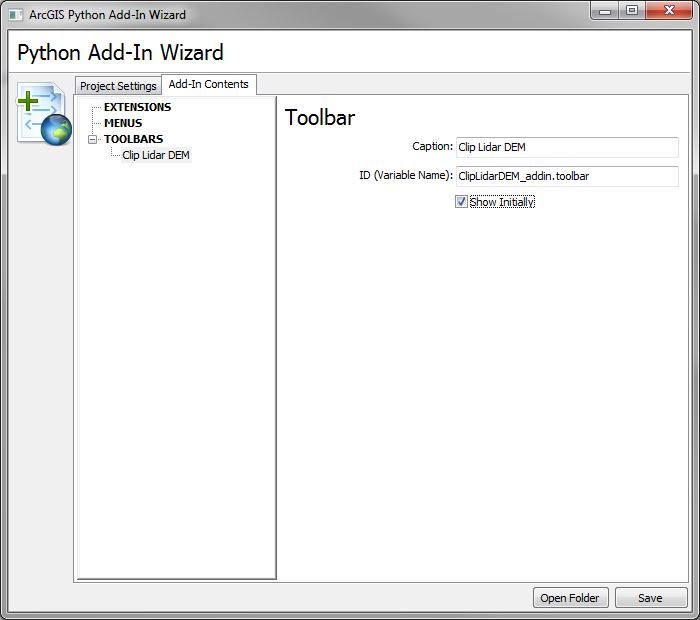
1. In Windows Explorer, create a folder and call it “ClipLidarDEM”. Copy the full path to the clipboard.
2. Find the “addin\_assistant” folder inside the class folder. Inside this folder, look inside the “bin” folder to find the “addin\_assistant.exe” and double click it to launch the Add-In Wizard. Paste the “ClipLidarDEM” folder path into “Folder” path. Hit OK.



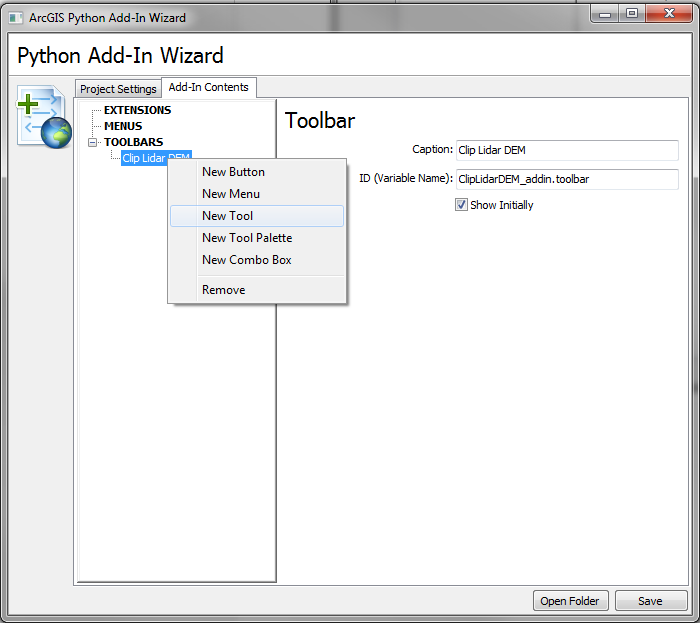
1. Fill out the information in the screenshot below appropriately.



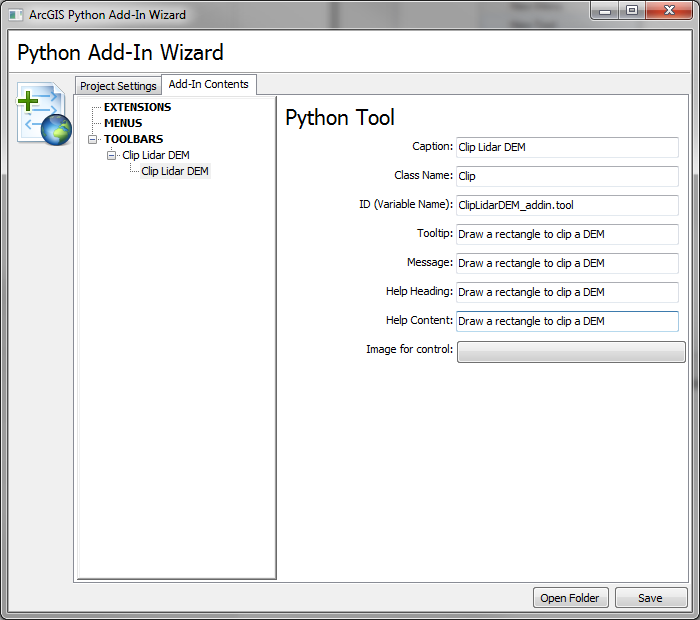
1. Switch to the “Add-In Contents” tab. Right click on the “TOOLBARS” button and choose “New Toolbar”. Change the caption to “Clip Lidar DEM”. Make sure the box for “Show Initially” is checked on.



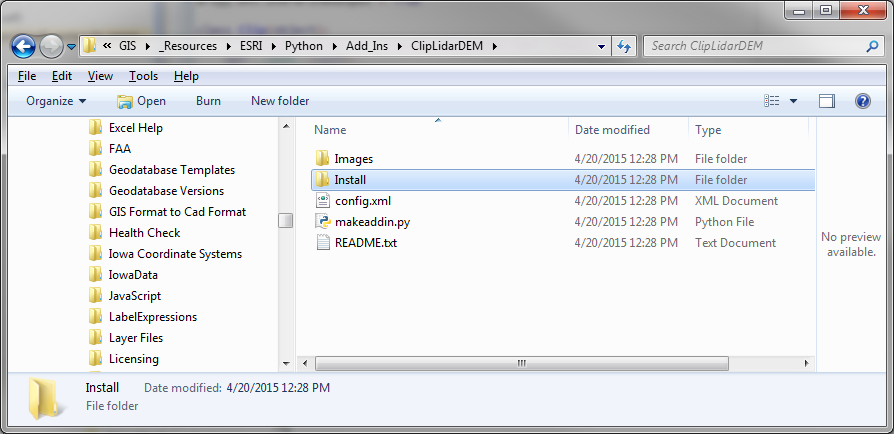
1. Right click on the toolbar you just created and select “New Tool”.



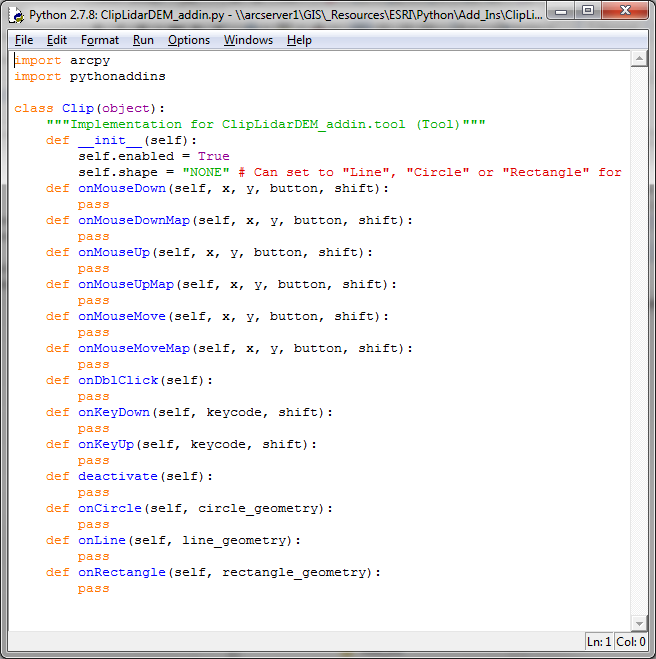
1. Set the caption to “Clip Lidar DEM” as you did with the toolbar name. Next, change the Class Name to “Clip” and leave the “ID (Variable Name)” as it is (this is used for the GUI in the config.xml file). Add appropriate messages and tool tips for this add in. I used “Draw a rectangle to clip a DEM” for each of the options. When you’re done, hit the “Save” button, and then choose “Open Folder”. This will launch the folder in Windows Explorer.



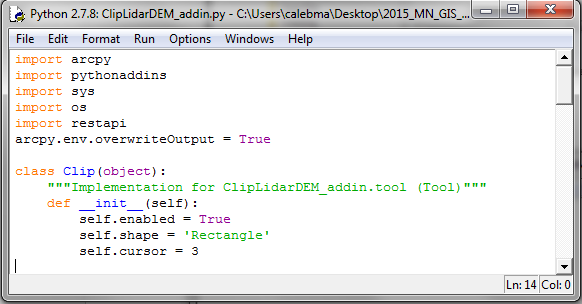
1. Examine the contents of the folder. The source code of the Add-In will be Install folder.



1. Inside the Install folder, find the ClipLidarDEM\_addin.py and open it with your favorite IDE, I recommend [PyScripter](https://code.google.com/p/pyscripter/downloads/list).

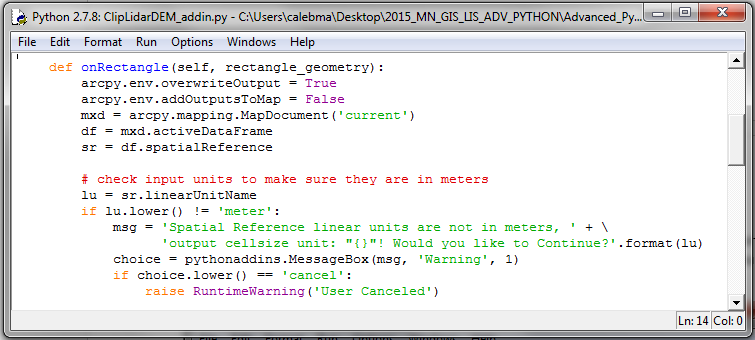


1. All the class methods can be deleted except for the “onRectangle” method. Also, import the os, sys, and restapi modules. Set the shape to “Rectangle” and the [cursor](http://resources.arcgis.com/EN/HELP/MAIN/10.1/index.html#/Tool/014p00000027000000/) can be set to your preference. A value of 3 represents the crosshair.



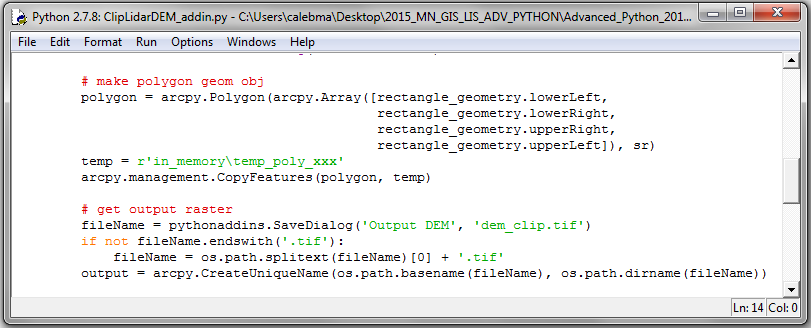
1. In the “OnRectangle” method, the first step is to set some environment variables and reference the current map document. The “addOutputsToMap” environment is set to false to prevent temporary outputs from being added to the map. The active data frame is then referenced and the spatial reference is derived from it.

Because the MPCA’s Lidar DEM image service has been published in the UTM Zone 15N coordinate system, a check is done to warn the user if the linear units are not meters for the spatial reference. This is done to maintain the original cell size of the raster (1 meter), otherwise, the output raster cell size will be resampled. If the units are not meters, a warning will be displayed to allow the user to cancel or continue the operation. This is done via the [MessageBox](http://resources.arcgis.com/EN/HELP/MAIN/10.1/index.html#/The_pythonaddins_module/014p00000021000000/). If the user chooses to cancel, the operation is aborted when the “RuntimeWarning” exception is raised.



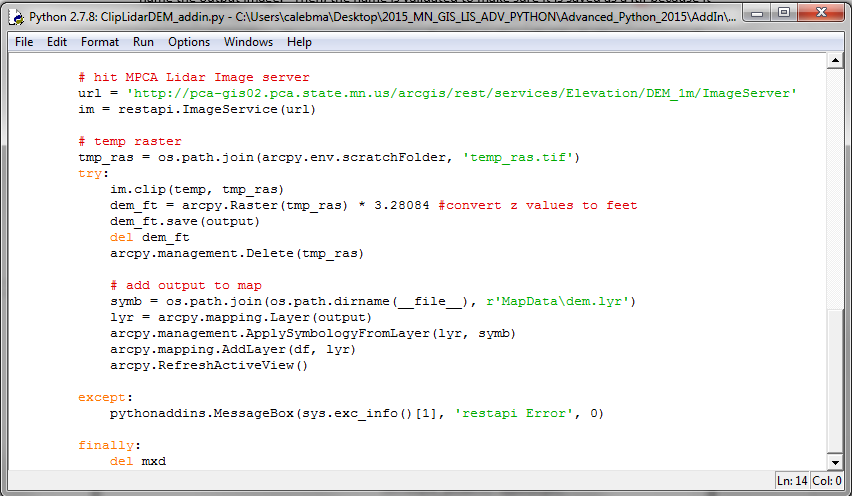
1. When the user draws a rectangle, the “onRectangle” method is fired. The “rectangle\_geometry” that is passed into the method is an [Extent](http://resources.arcgis.com/en/help/main/10.2/index.html#//018z00000072000000) object. This will be converted to a [Polygon](http://resources.arcgis.com/en/help/main/10.2/index.html#/Polygon/018z00000061000000/) so that it can be used to clip the ImageService. The polygon is then saved as a temporary feature class in the “in\_memory” work.

Next, the “saveDialog” function is called from the pythonaddins module to allow the user to name the output image. Then, the name is validated to make sure it is saved as a .tif because it is the only format the ArcGIS REST API will allow to maintain the pixel values and spatial reference information. The [CreateUniqueName](http://resources.arcgis.com/en/help/main/10.2/index.html#//018v0000004v000000) function is also called for further validation of the name.



1. Now that code has been entered to handle the user defined polygon, the restapi module can be used to create an “ImageService” object. The image service URL can be copied [here](http://pca-gis02.pca.state.mn.us/arcgis/rest/services/Elevation/DEM_1m/ImageServer). A variable is then created to store a temporary raster file in the “scratchFolder” environment. The image is then clipped by calling the “clip” method of the **restapi.ImageService** and the temporary raster is multiplied by 3.28084 to convert the Z values to feet as the output raster.

Finally, the output is added to the map. Because, the DEM will not always come in as the standard black and white stretched format, a layer file is referenced to set the proper symbology. Inside the “AddIn” folder, find the “AddIn\_Data” folder and copy the contents into the install folder. The layer is added to the map and [ApplySymbologyFromLayer](http://resources.arcgis.com/en/help/main/10.2/index.html#//00170000006n000000) is called to set the symbology. All of this is wrapped in a try/except block. If it fails, a MessageBox is displayed to show the error reported from the restapi module. Finally, the reference to the mxd is deleted to prevent any issues.



1. Save and check the module to make sure there are no syntax errors (Alt + x or Run > Check Module). Make sure the restapi module is accessible to the script, **otherwise the Add-In will appear broken in ArcMap**. This can be done by placing the restapi in one of the valid PYTHONPATHS (can view these by using print sys.path) or by saving the restapi folder inside the Install folder where this script resides (PYTHONHOME).

If everything looks good close the script and find the “makeaddin.py” script that is provided by Esri (up one directory). Double click it to execute it. This will create a \*.esriaddin file. Double click on this to install the Add-In. This can now be tested in ArcMap and it \*should\* show up automatically when you open ArcMap. If not, right click in the grey area in the standard toolbar and add the “Clip Lidar Data” toolbar by checking it. It is also good practice when testing Python Add-Ins to **open the** **python window to see if any error messages pop up when running the Add-In**. This is the easiest way to debug an Add-In. The important thing to note is if you are debugging your script, you will have to close down ArcMap (so the Add-In can be refreshed) and make the necessary changes to the script. **Once the errors have been fixed, you will have to run the “makeaddin.py” script again and double click on the .esriaddin file again to install it. At this point ArcMap can be opened again and the Add-In should be refreshed.**

Open the “Counties\_v10\_1.mxd” and test the Add-In to clip both the raster data.