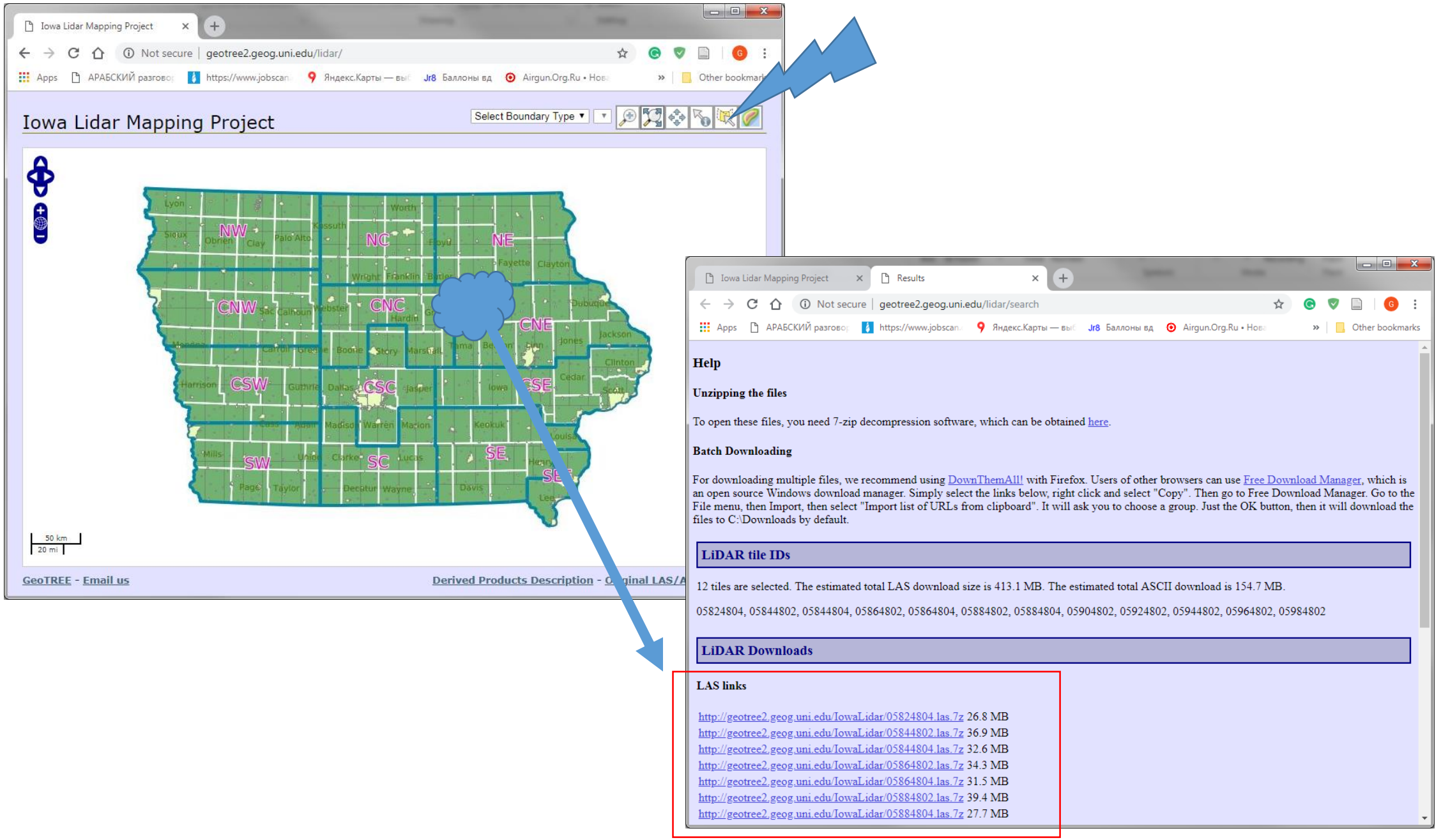
GeoTree LiDAR Downloader

Term project by Nikolay Golosov

Intro


Iowa Lidar Mapping Project
Not secure | geotree2.geog.uni.edu/lidar/
Iowa Lidar Mapping Project
Select Boundary Type
NW
NC
NE
CNW
CNC
CNE
CSW
CSC
CSE
SW
SC
SE
50 km
20 mi
GeoTREE - Email us
Derived Products Description
Iowa Lidar Mapping Project
Results
Not secure | geotree2.geog.uni.edu/lidar/search
Help
Unzipping the files
To open these files, you need 7-zip decompression software, which can be obtained here.
Batch Downloading
For downloading multiple files, we recommend using DownThemAll! with Firefox. Users of other browsers can use Free Download Manager, which is an open source Windows download manager. Simply select the links below, right click and select "Copy". Then go to Free Download Manager. Go to the File menu, then Import, then select "Import list of URLs from clipboard". It will ask you to choose a group. Just the OK button, then it will download the files to C:\Downloads by default.
LiDAR tile IDs
12 tiles are selected. The estimated total LAS download size is 413.1 MB. The estimated total ASCII download is 154.7 MB.
05824804, 05844802, 05844804, 05864802, 05864804, 05884802, 05884804, 05904802, 05924802, 05944802, 05964802, 05984802
LiDAR Downloads
LAS links
http://geotree2.geog.uni.edu/IowaLidar/05824804.las.7z 26.8 MB
http://geotree2.geog.uni.edu/IowaLidar/05844802.las.7z 36.9 MB
http://geotree2.geog.uni.edu/IowaLidar/05844804.las.7z 32.6 MB
http://geotree2.geog.uni.edu/IowaLidar/05864802.las.7z 34.3 MB
http://geotree2.geog.uni.edu/IowaLidar/05864804.las.7z 31.5 MB
http://geotree2.geog.uni.edu/IowaLidar/05884802.las.7z 39.4 MB
http://geotree2.geog.uni.edu/IowaLidar/05884804.las.7z 27.7 MB

Introduction:

GeoTree Iowa LiDAR Web-Site

- Convenient for single shot tasks, difficult for everyday use
- Need to know exact borders of the data
- Tedious and involving many steps

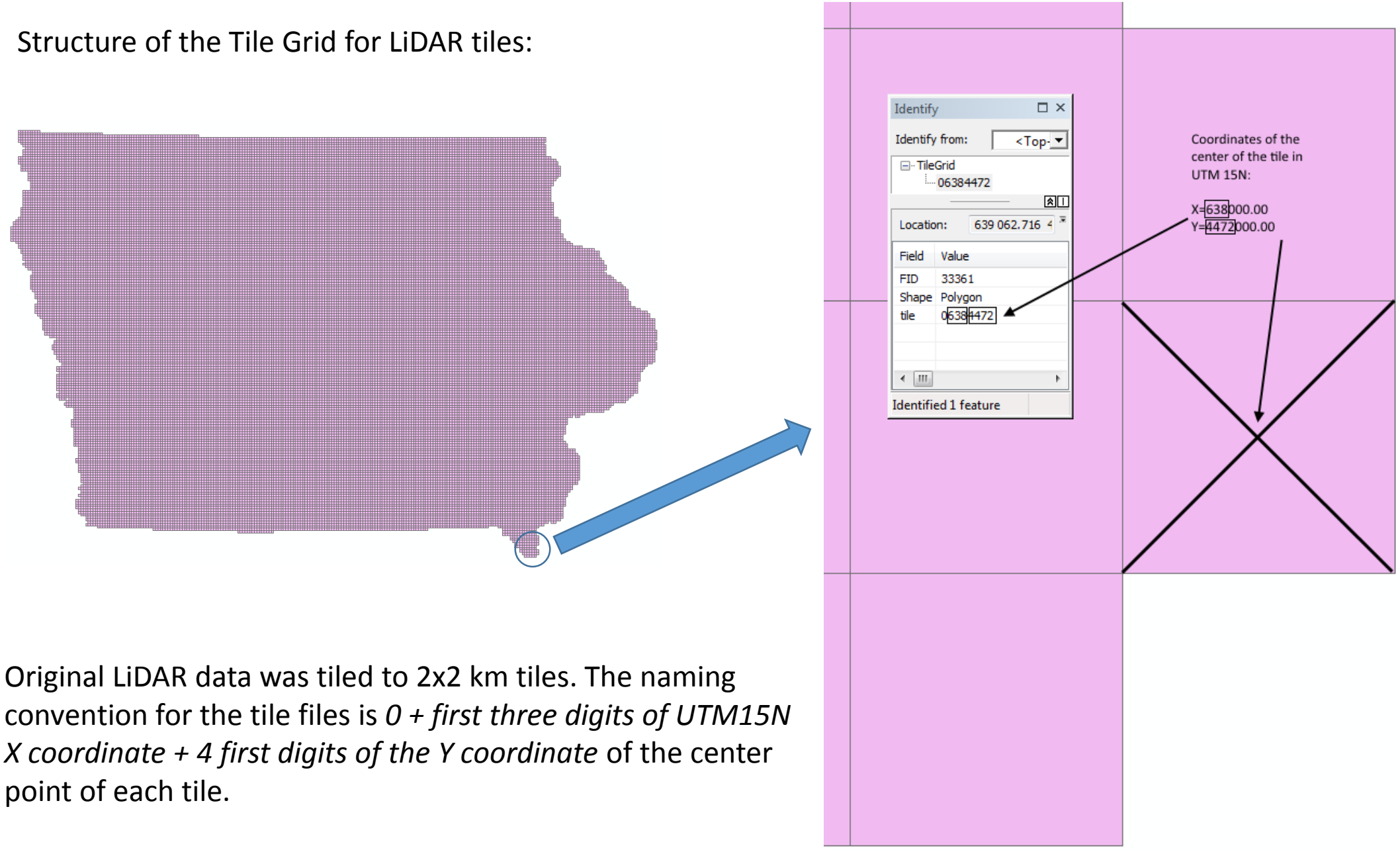
Why automation:

- Inspiration from GIS2 term project
- Spent a lot of time manually downloading the LAS and XYZI files
- inspiration from MNGeo tool

Goal:

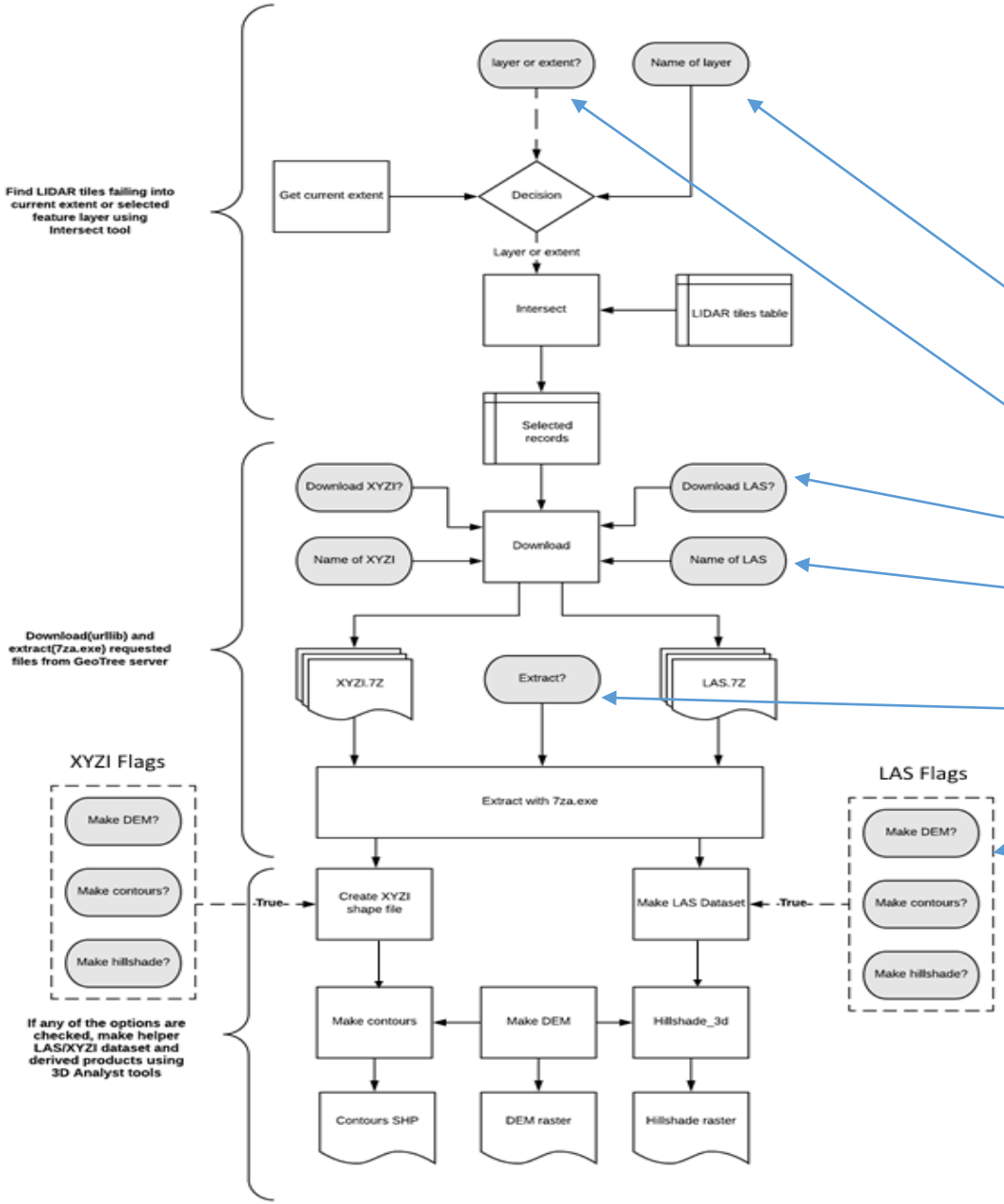
- Create ArcGIS Tool with functionality similar to the web-site

Structure of the Tile Grid for LiDAR tiles:

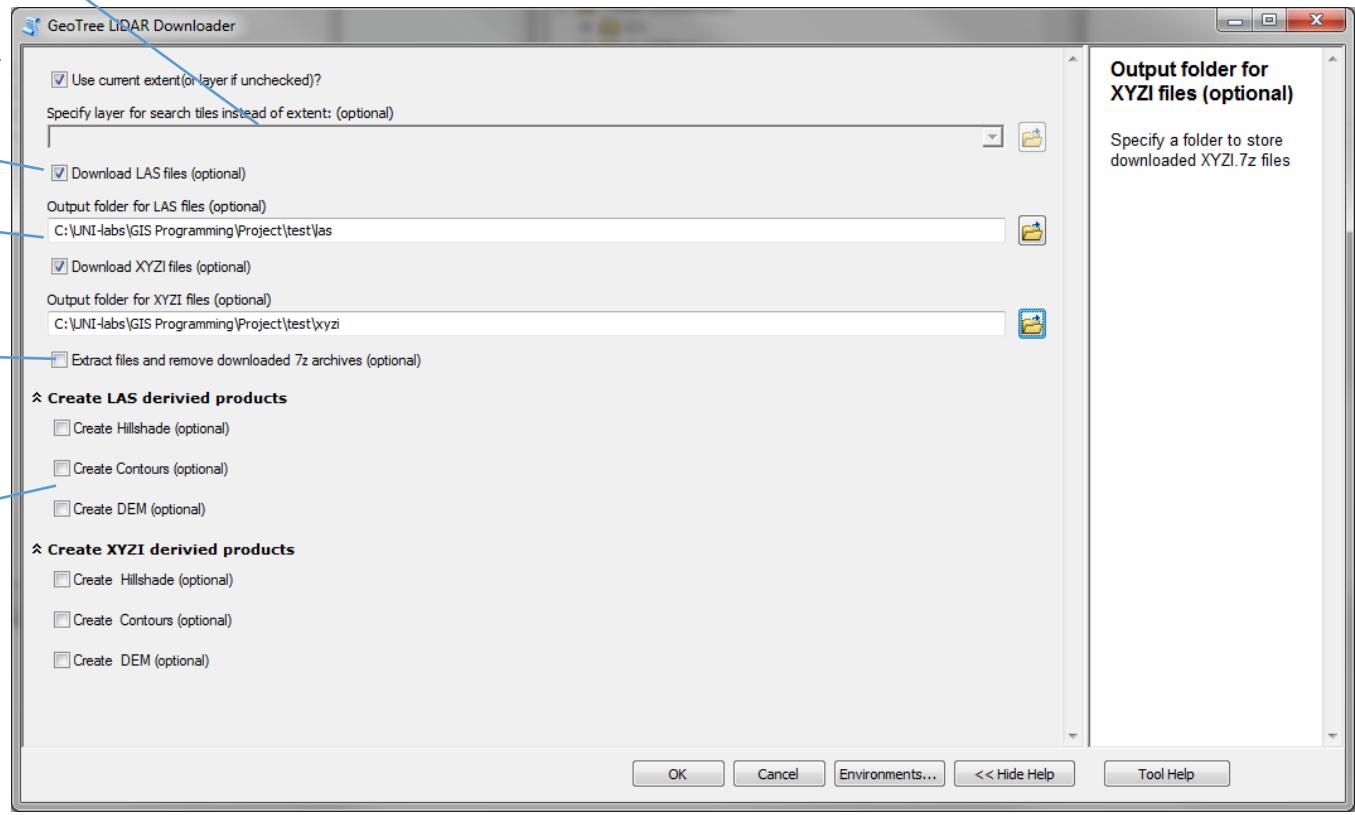


Original LiDAR data was tiled to 2x2 km tiles. The naming convention for the tile files is *0 + first three digits of UTM15N X coordinate + 4 first digits of the Y coordinate* of the center point of each tile.

Methods



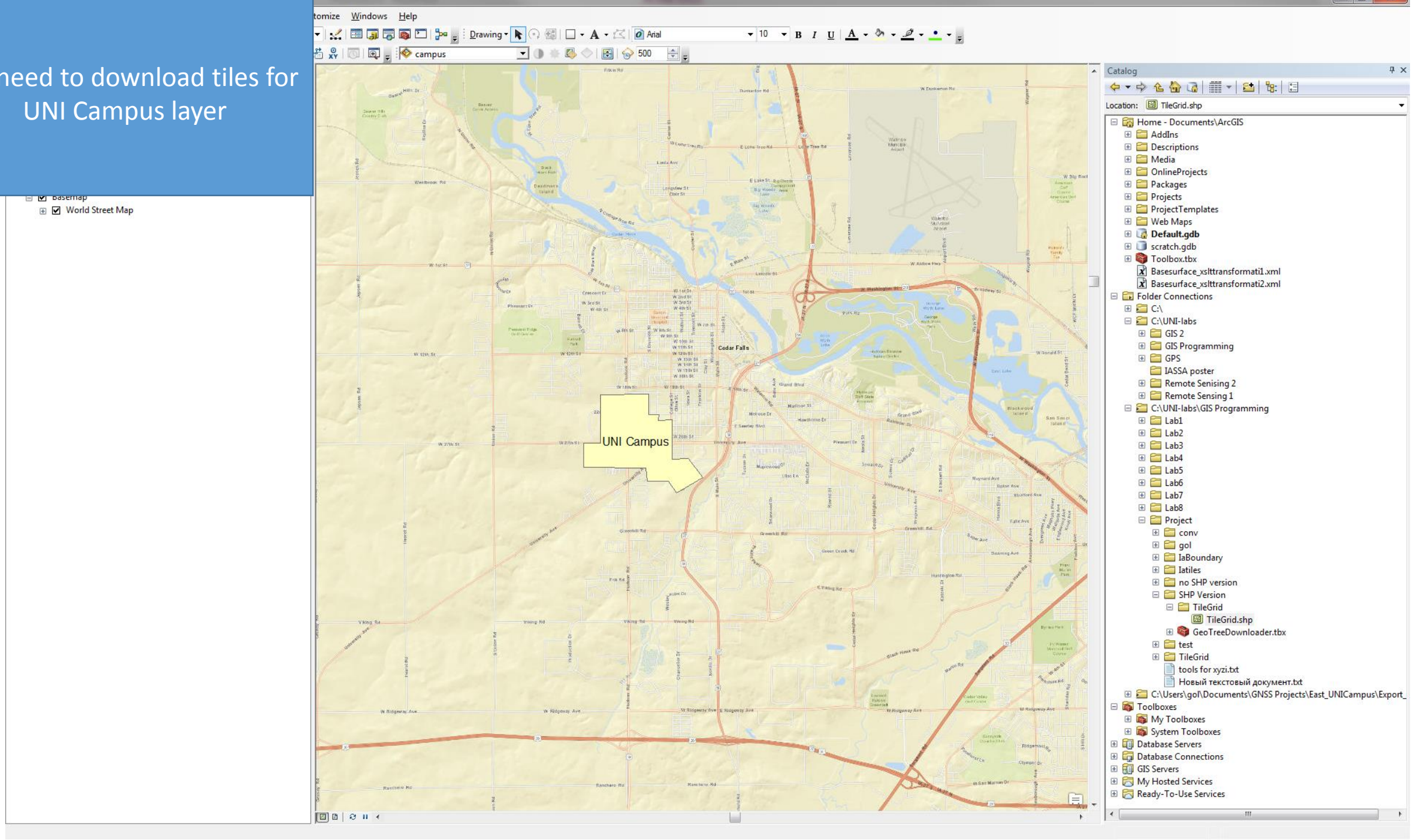

```
#reading parameters from the tool dialog:
#checkbox, if set, we processing the extent, if not set - layer
extentOrLayer=arcpy.GetParameter(0)
#layer name to check the intersections
studyLayer=arcpy.GetParameterAsText(1)
#Download LAS flag
downloadLAS=arcpy.GetParameter(2)
#output folder for LAS 7z files
```



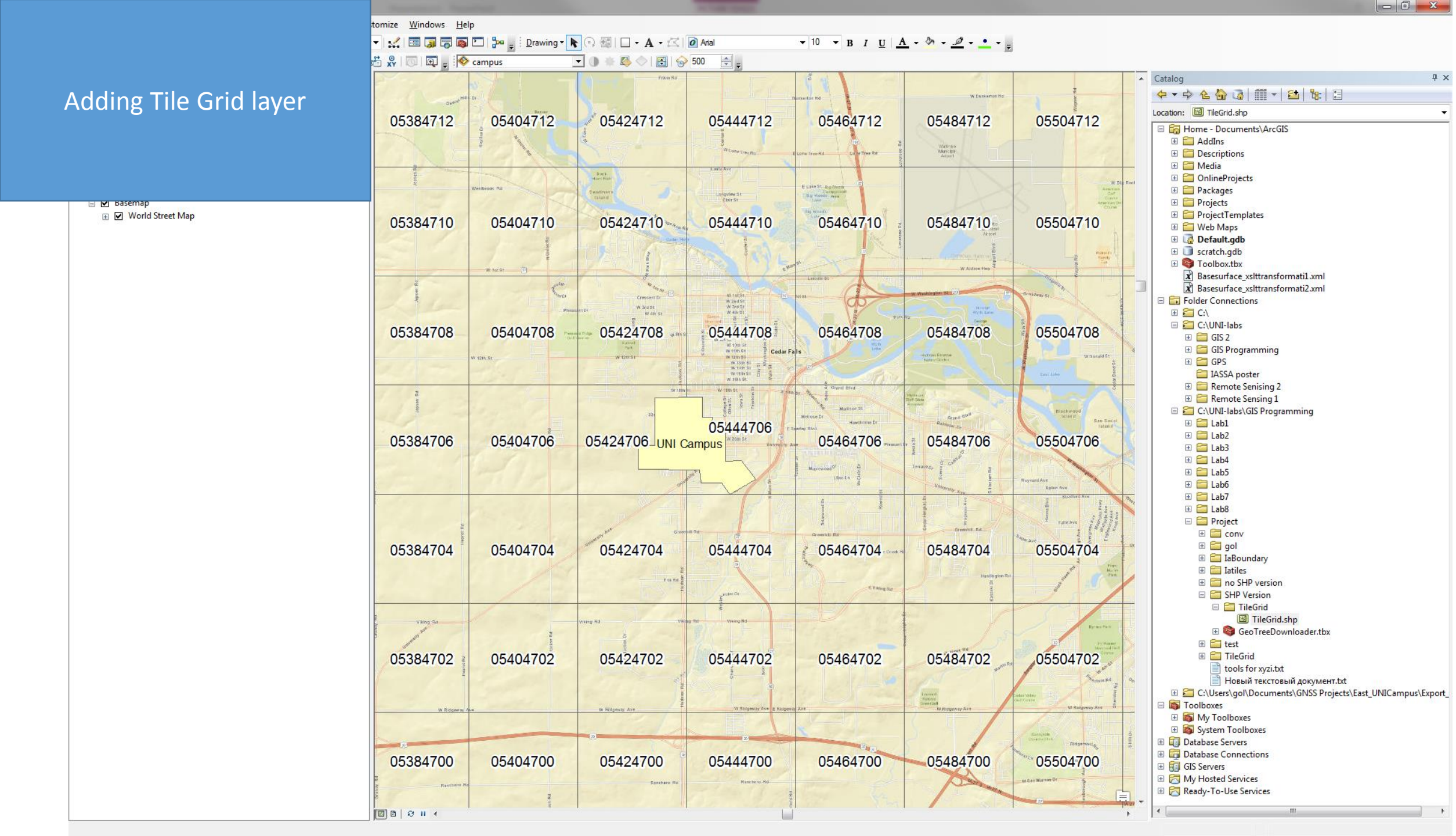
Programmatically create tile grid feature class and its geometry:

```
#setting a tuple with WKT feature representation and attribute value
row_values = [('POLYGON ((202000 4732000,202000 4734000,204000 4734000,204000 4732000,202000 4732000))', '02024732'),
('POLYGON ((202000 4734000,202000 4736000,204000 4736000,204000 4734000,202000 4734000))', '02024734'),
('POLYGON ((202000 4736000,202000 4738000,204000 4738000,204000 4736000,202000 4736000))', '02024736')]

#setting workspace to use in-memory workspace
arcpy.env.workspace='in_memory'

#creating an empty feature class in the in-memory workspace
arcpy.CreateFeatureclass_management('in_memory', 'tilegrid', 'POLYGON', '', '', '', arcpy.SpatialReference(26915))

#adding a field to store attribute values
arcpy.AddField_management ('tilegrid', 'TILE', 'TEXT')

# Open an InsertCursor
cursor = arcpy.da.InsertCursor('tilegrid',['SHAPE@WKT', 'TILE'])

# Insert new rows that include the geometry, converted from WKT and attribute values
for row in row_values:
    cursor.insertRow(row)

# Delete cursor object
del cursor
```

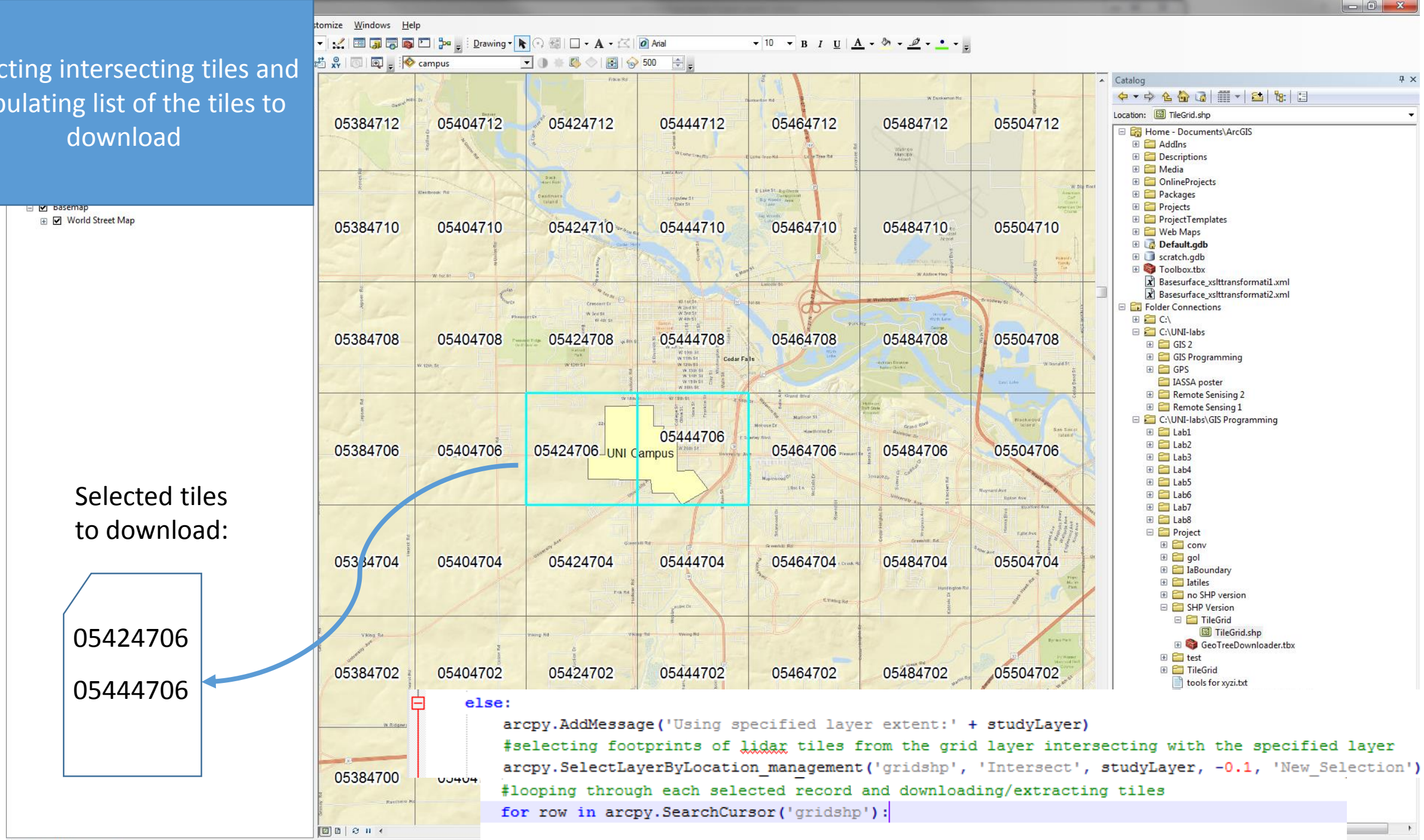
We need to download tiles for UNI Campus layer



Adding Tile Grid layer



Selecting intersecting tiles and populating list of the tiles to download



Selected tiles to download:

05424706

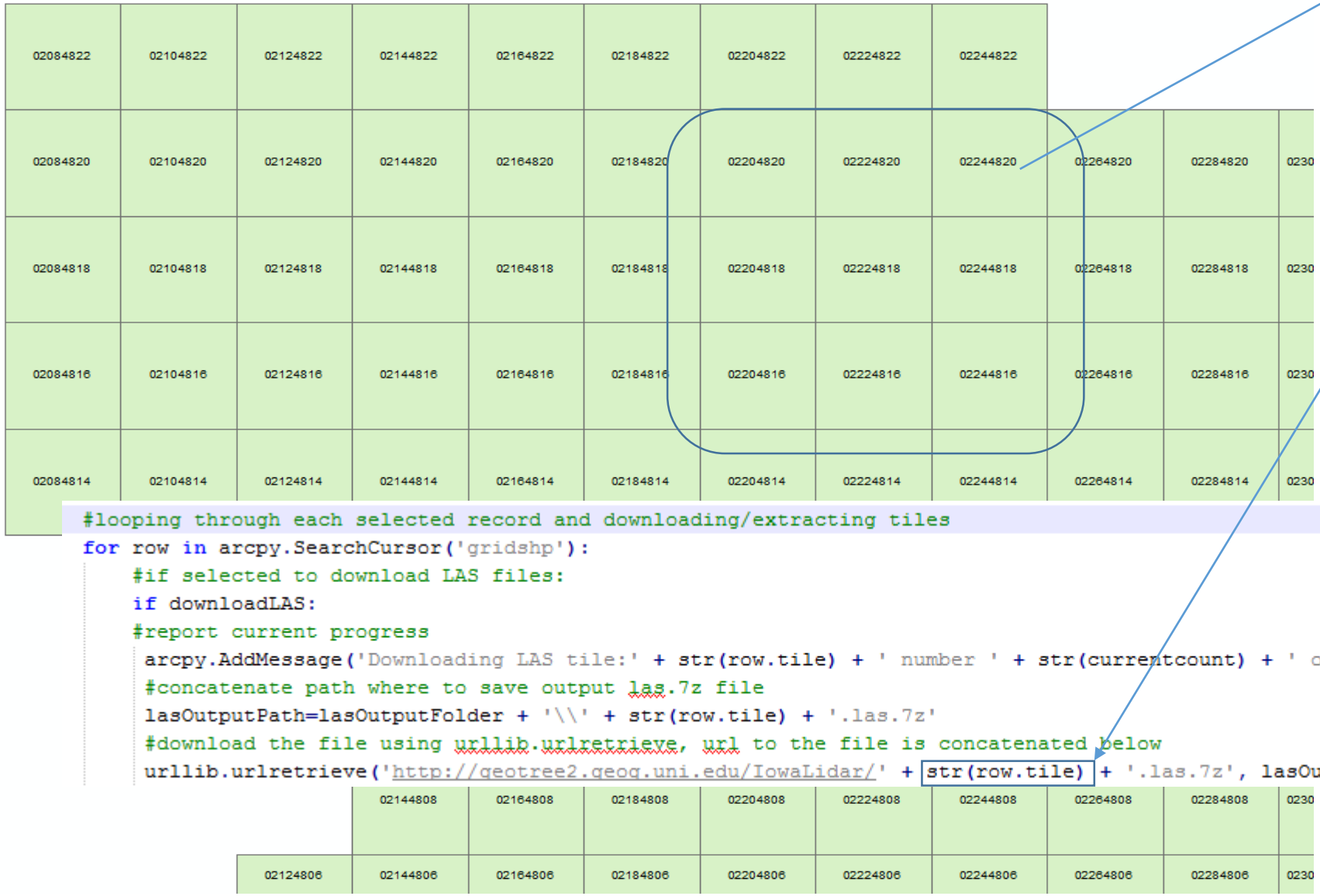
05444706

```
    else:
        arcpy.AddMessage('Using specified layer extent:' + studyLayer)
        #selecting footprints of lidar tiles from the grid layer intersecting with the specified layer
        arcpy.SelectLayerByLocation_management('gridshp', 'Intersect', studyLayer, -0.1, 'New_Selection')
    #looping through each selected record and downloading/extracting tiles
    for row in arcpy.SearchCursor('gridshp'):
```

Alternatively, we may select for download all the tiles in the current map extent


```
if extentOrLayer:
    arcpy.AddMessage('Using current extent...')
    mxd = arcpy.mapping.MapDocument("CURRENT") # get current map
    df = arcpy.mapping.ListDataFrames(mxd)[0] # get 1st data frame
    sr = df.spatialReference # get spatial reference
    #checking if spatial reference is not empty
    if sr.type=='':
        arcpy.AddError('Spatial reference should not be Unknown, add some layers to the current map
        sys.exit(0)
    ext = df.extent # extent object
    BL = arcpy.Point(ext.XMin,ext.YMin) # bottom left
    BR = arcpy.Point(ext.XMax,ext.YMin) # bottom right
    TR = arcpy.Point(ext.XMax,ext.YMax) # top right
    TL = arcpy.Point(ext.XMin,ext.YMax) # top left
    df_poly = arcpy.Polygon(arcpy.Array([[BL,BR,TR,TL,BL]]),sr) # create polygon
    #selecting footprints of lidar tiles from the grid layer intersecting with the current extent
    arcpy.SelectLayerByLocation_management('gridshp', 'Intersect', df_poly, 0, 'New_Selection')
    #otherwise using specified layer extent instead of current extent to find tiles to download
#looping through each selected record and downloading/extracting tiles
for row in arcpy.SearchCursor('gridshp'):
```

Having selected the tiles, we're setting search cursor loop to download each tile with **urlretreive** function from the standard Python **urllib** module.



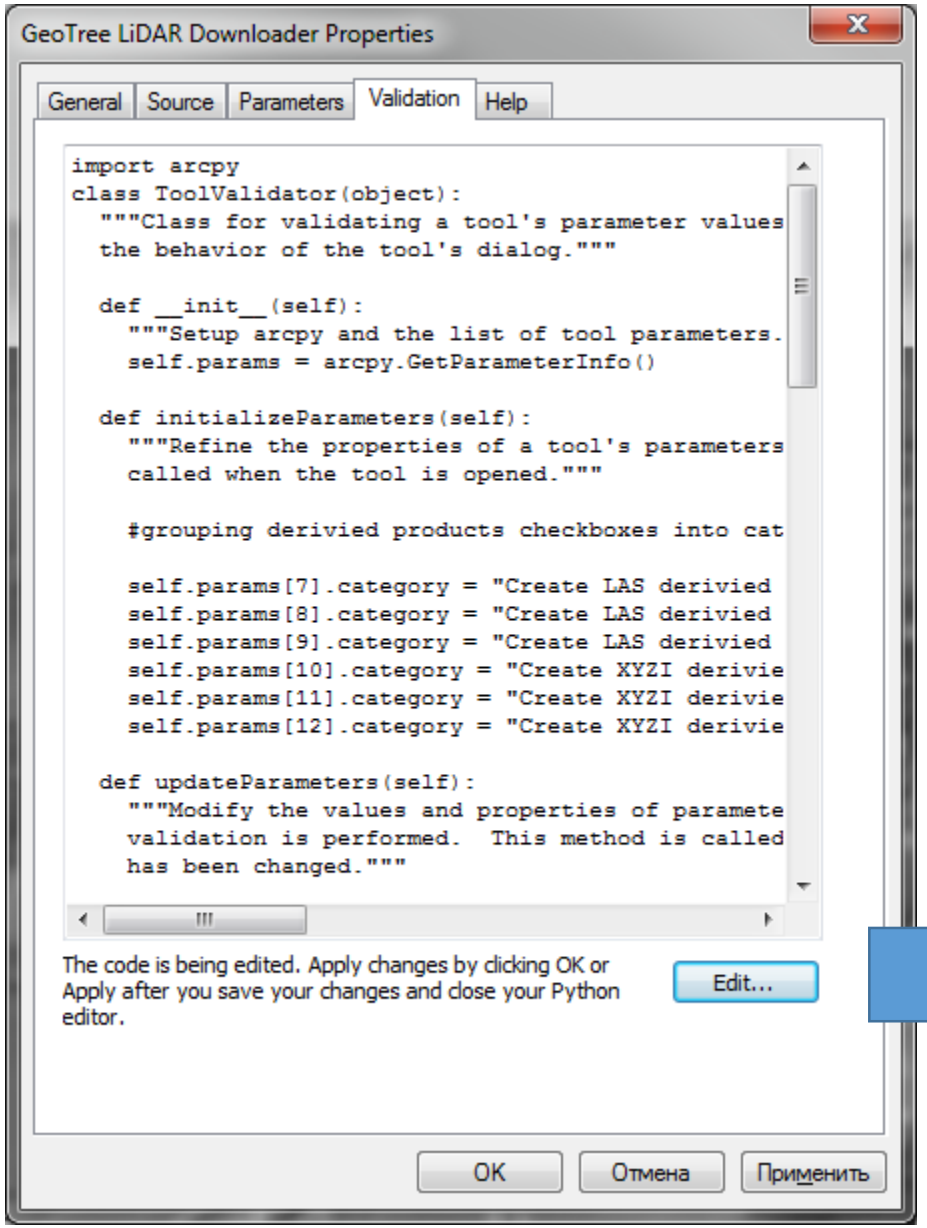
02204820
02204811
01104823
02224832
02224841

```
#looping through each selected record and downloading/extracting tiles
for row in arcpy.SearchCursor('gridshp'):
    #if selected to download LAS files:
    if downloadLAS:
    #report current progress
     arcpy.AddMessage('Downloading LAS tile:' + str(row.tile) + ' number ' + str(currentcount) + ' out of ' + str(totalcount))
     #concatenate path where to save output las.7z file
     lasOutputPath=lasOutputFolder + '\\' + str(row.tile) + '.las.7z'
     #download the file using urllib.urlretrieve, url to the file is concatenated below
     urllib.urlretrieve('http://geotree2.geog.uni.edu/IowaLidar/' + str(row.tile) + '.las.7z', lasOutputPath)
```

Finally creating the derived products from the downloaded files using 3D Analyst functions:

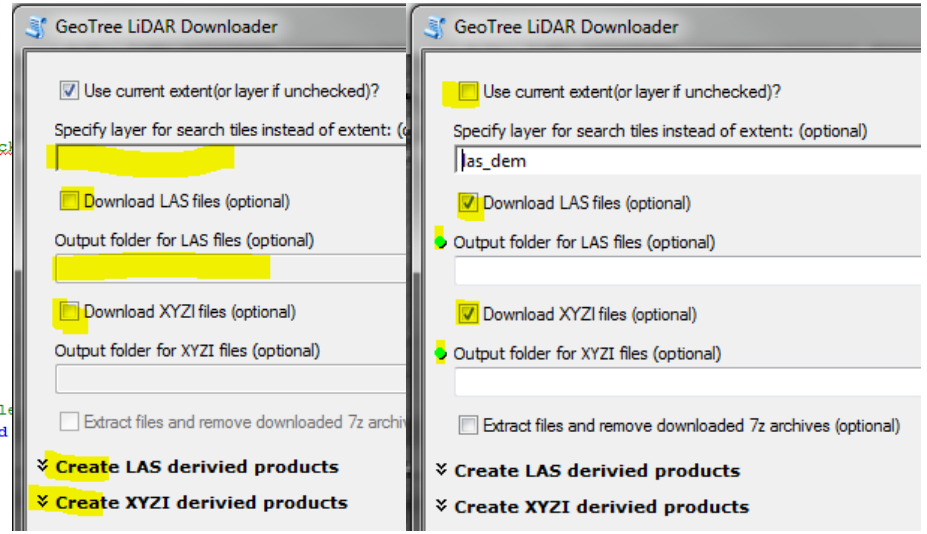
```
#if create contours flag is selected, run SurfaceContour_3d tool to create contours
    if createContoursLAS:
        arcpy.AddMessage('Creating LAS contours...')
        arcpy.SurfaceContour_3d(in_surface=lasOutputFolder+'\\output\\las_dataset.lasd', out_feature_class=lasOutputFolder+'\\output\\las_contours.shp', interval="2", base_contour="0", contour_field="Contour",
        contour_field_precision="0", index_interval="", index_interval_field="Index_Cont", z_factor="1", pyramid_level_resolution="0")

#if create DEM is selected, run LasDatasetToRaster_conversion tool to create DEM raster from las dataset
    if createDemLAS:
        arcpy.AddMessage('Creating LAS DEM...')
        arcpy.LasDatasetToRaster_conversion(in_las_dataset=lasOutputFolder+'\\output\\las_dataset.lasd', out_raster=lasOutputFolder+'\\output\\las_dem', value_field="ELEVATION", interpolation_type="BINNING AVERAGE LINEAR",
        data_type="FLOAT", sampling_type="CELLSIZE", sampling_value="10", z_factor="1")

#if create Hillshade is selected, run HillShade_3d tool to create hillshade from DEM raster from las dataset
    if createHillshadeLAS:
        arcpy.AddMessage('Creating LAS hillshade...')
        #checking if DEM rasted was created above, otherwise creating it here
        if not arcpy.Exists(lasOutputFolder+'\\output\\las_dem'):
            arcpy.LasDatasetToRaster_conversion(in_las_dataset=lasOutputFolder+'\\output\\las_dataset.lasd', out_raster=lasOutputFolder+'\\output\\las_dem', value_field="ELEVATION", interpolation_type="BINNING AVERAGE LINEAR
            data_type="FLOAT", sampling_type="CELLSIZE", sampling_value="10", z_factor="1")
        arcpy.HillShade_3d(in_raster=lasOutputFolder+'\\output\\las_dem', out_raster=lasOutputFolder+'\\output\\las_hillshade', azimuth="315", altitude="45", model_shadows="NO_SHADOWS", z_factor="1")

#if create contours flag is selected, run Contour_3d tool to create contours
    if createContoursXYZI:
        createXyziDem()
        arcpy.AddMessage('Creating XYZI contours...')
        arcpy.Contour_3d(xyziOutputFolder+'\\output\\xyzi_dem', xyziOutputFolder+'\\output\\xyzi_contours.shp', "2", "0", "1", "CONTOUR", "")

#if create xyzi DEM is selected, run createXyziDem() custom function which will create DEM if DEM is not exists
    if createDemXYZI:
        createXyziDem()

#if create Hillshade from XYZI is selected, run HillShade_3d tool to create hillshade from XYZI DEM
    if createHillshadeXYZI:
        createXyziDem()
        arcpy.AddMessage('Creating XYZI hillshade...')
        arcpy.HillShade_3d(in_raster=xyziOutputFolder+'\\output\\xyzi_dem', out_raster=xyziOutputFolder+'\\output\\xyz_hillshade', azimuth="315", altitude="45", model_shadows="NO_SHADOWS", z_factor="1")
```

Tool validation code:

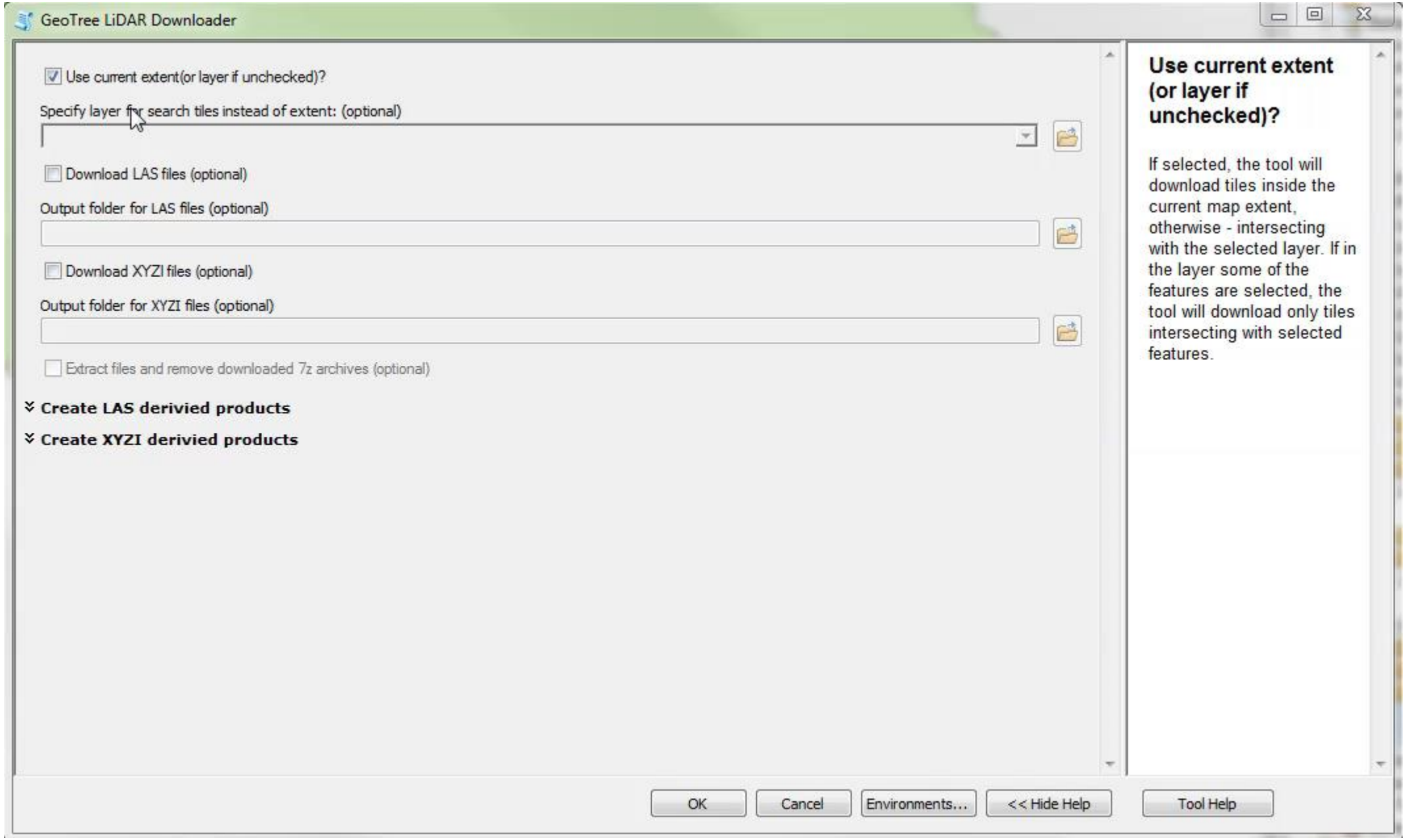



```
def initializeParameters(self):
  """Refine the properties of a tool's parameters.  This method is
  called when the tool is opened."""

  #grouping derivied products checkboxes into categories

  self.params[7].category = "Create LAS derivied products"
  self.params[8].category = "Create LAS derivied products"
  self.params[9].category = "Create LAS derivied products"
  self.params[10].category = "Create XYZI derivied products"
  self.params[11].category = "Create XYZI derivied products"
  self.params[12].category = "Create XYZI derivied products"

def updateParameters(self):
  """Modify the values and properties of parameters before internal
  validation is performed.  This method is called whenever a parameter
  has been changed."""
  pass
  #Checking "Use current extent..." checkbox and enabling/disabling "Specify layer for search tiles"
  if self.params[0].value == True:
      self.params[1].enabled = False
  else:
      self.params[1].enabled = True

  #Checking "Download LAS files" checkbox and enabling/disabling "Output folder for LAS tiles" and flags for derivied products
  if self.params[2].value == False:
      self.params[3].enabled = False
      #self.params[6].enabled = False
      self.params[7].enabled = False
      self.params[8].enabled = False
      self.params[9].enabled = False
  else:
      self.params[3].enabled = True
      #self.params[6].enabled = True
      self.params[7].enabled = True
      self.params[8].enabled = True
      self.params[9].enabled = True
  #Checking "Download XYZI files" chec
  if self.params[4].value == False:
      self.params[5].enabled = False
      self.params[10].enabled = False
      self.params[11].enabled = False
      self.params[12].enabled = False
  else:
      self.params[5].enabled = True
      self.params[10].enabled = True
      self.params[11].enabled = True
      self.params[12].enabled = True

#disabling Extract and remove source file
  if self.params[2].value == False and
      self.params[6].enabled = False
  else:
      self.params[6].enabled = True
```

Conditionally Enable/Disable controls



Results

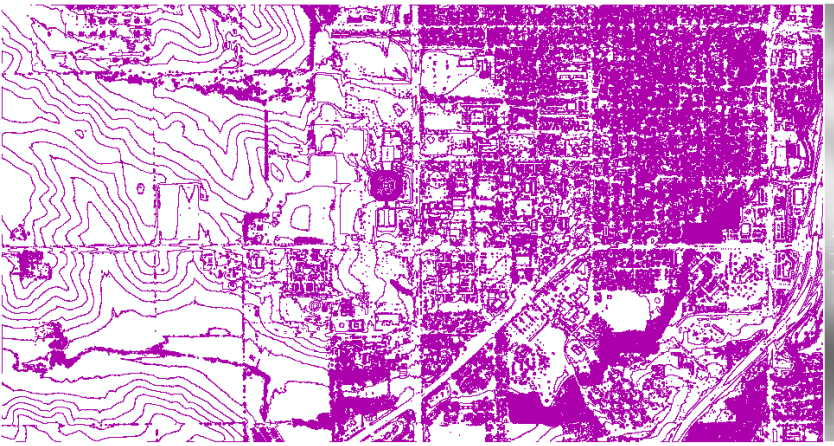

GeoTree LiDAR Downloader
Use current extent(or layer if unchecked)?
Specify layer for search tiles instead of extent: (optional)
Download LAS files (optional)
Output folder for LAS files (optional)
Download XYZI files (optional)
Output folder for XYZI files (optional)
Extract files and remove downloaded 7z archives (optional)
Create LAS derivied products
Create XYZI derivied products
Use current extent (or layer if unchecked)?
If selected, the tool will download tiles inside the current map extent, otherwise - intersecting with the selected layer. If in the layer some of the features are selected, the tool will download only tiles intersecting with selected features.
OK
Cancel
Environments...
<< Hide Help
Tool Help


Contours from LAS files

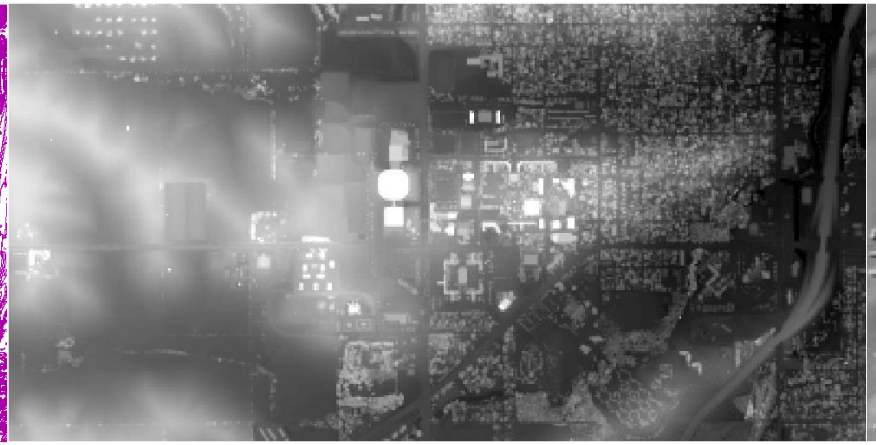
Final derived products:

From LAS files:

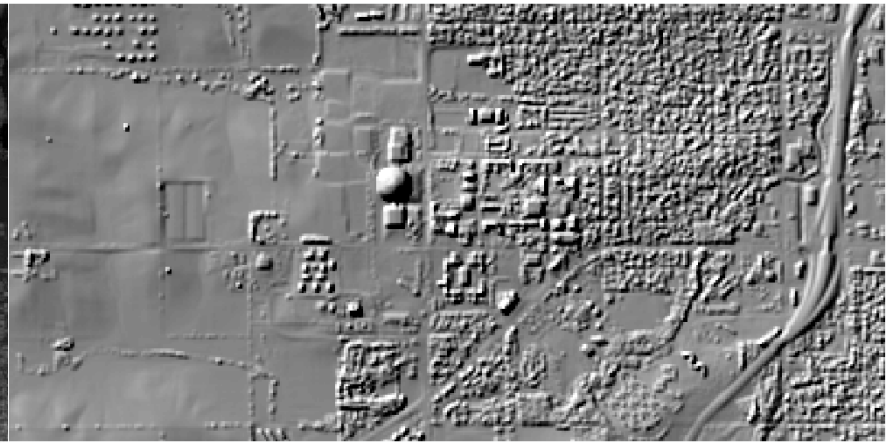
Contours:



DEM:

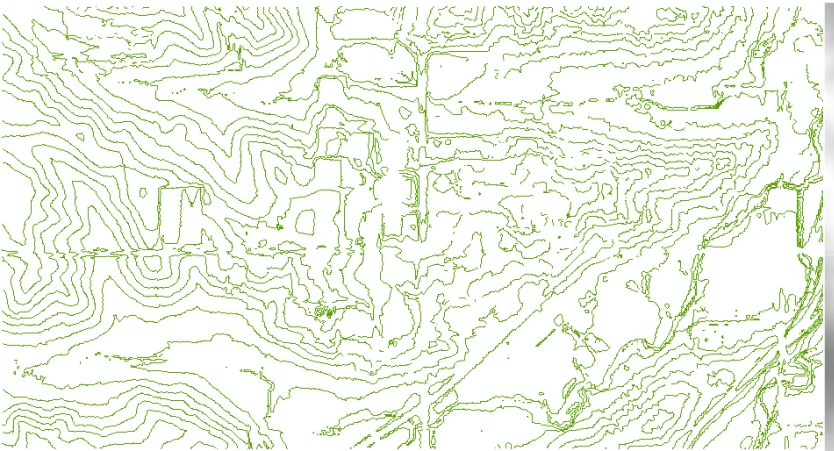


Hillshade:



From bare-earth XYZI files:

Contours:

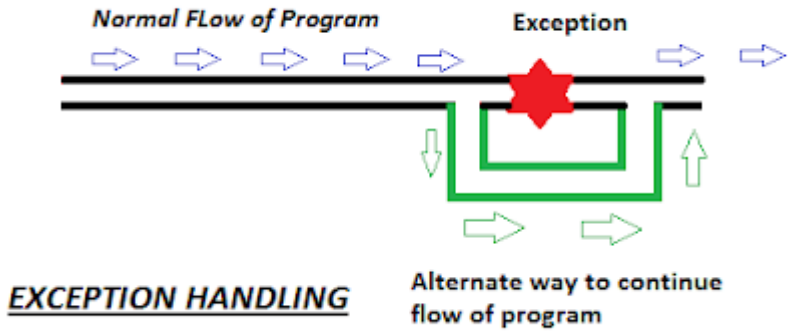


DEM:

Hillshade:

Lessons learned

- Use GetParameter() instead of GetParameterAsText() function to get Boolean values
- Handle exception effectively



- Need to use advanced multi-treaded libraries for faster and more reliable file' downloading instead of urrlib.urlretreive

Conclusion

- A tool for convenient downloading of the LiDAR data, bare earth files, and producing derived data products within a few mouse clicks was developed
- The tool is portable and could be easily re-used by others. I hope that it will be published on GeoTree website to further assist users in their work