In [11]:
```
#Read in libraries/extensions
import arcpy
arcpy.CheckOutExtension("ImageAnalyst")
from arcpy.ia import *
from arcpy import env
from arcpy.sa import *
import os
import os.path

#Define input folders
in_fold = "E:/topo_proj/topo_check/"
in_toposF = "oh_needed_topos2/"
in_minesF = "oh_topo_mines2/"
in_quadsF = "oh_topo_quads/"
chips_outF = in_fold + "processing/oh_chips_background/"
to_8bitF = in_fold + "processing/oh_topo8bit_background/"

#Create new directories
os.mkdir(chips_outF)
os.mkdir(to_8bitF)

#List all topo maps
arcpy.env.workspace =  in_fold + in_toposF
chips = arcpy.ListRasters()
print(chips)
```

```
['OH_Addison_224687_1960_24000_geo.tif', 'OH_Addison_224689_1960_24000_geo.tif', 'OH_Addison_226205_1960_24000_geo.tif', 'OH_Albany_224714_1960_24000_geo.tif', 'OH_Albany_224716_1960_2
4000_geo.tif', 'OH_Dalzell_224610_1961_24000_geo.tif', 'OH_Dalzell_224612_1961_24000_geo.tif', 'OH_Lowell_225659_1961_24000_geo.tif', 'OH_Lowell_225661_1961_24000_geo.tif', 'OH_Macksbu
rg_225682_1961_24000_geo.tif', 'OH_Macksburg_225684_1961_24000_geo.tif', 'OH_Rutland_227264_1960_24000_geo.tif', 'OH_Rutland_227266_1960_24000_geo.tif', 'OH_Sarahsville_226922_1961_240
00_geo.tif', 'OH_Sarahsville_226923_1961_24000_geo.tif', 'OH_Shade_226959_1960_24000_geo.tif', 'OH_Shade_226961_1960_24000_geo.tif', 'OH_Stafford_227035_1961_24000_geo.tif', 'OH_Staffo
rd_227037_1961_24000_geo.tif', 'OH_Stafford_228181_2002_24000_geo.tif', 'OH_Summerfield_226730_1961_24000_geo.tif', 'OH_Summerfield_226731_1961_24000_geo.tif', 'OH_Summerfield_228206_2
002_24000_geo.tif']
```

In [12]:
```
#Make list and remove file extensions
chip_n = list()
for c in chips:
    c1 = os.path.basename(c)
    c2 = os.path.splitext(c1)[0]
    chip_n.append(c2)
print(chip_n)
```

```
['OH_Addison_224687_1960_24000_geo', 'OH_Addison_224689_1960_24000_geo', 'OH_Addison_226205_1960_24000_geo', 'OH_Albany_224714_1960_24000_geo', 'OH_Albany_224716_1960_24000_geo', 'OH_D
alzell_224610_1961_24000_geo', 'OH_Dalzell_224612_1961_24000_geo', 'OH_Lowell_225659_1961_24000_geo', 'OH_Lowell_225661_1961_24000_geo', 'OH_Macksburg_225682_1961_24000_geo', 'OH_Macks
burg_225684_1961_24000_geo', 'OH_Rutland_227264_1960_24000_geo', 'OH_Rutland_227266_1960_24000_geo', 'OH_Sarahsville_226922_1961_24000_geo', 'OH_Sarahsville_226923_1961_24000_geo', 'OH
_Shade_226959_1960_24000_geo', 'OH_Shade_226961_1960_24000_geo', 'OH_Stafford_227035_1961_24000_geo', 'OH_Stafford_227037_1961_24000_geo', 'OH_Stafford_228181_2002_24000_geo', 'OH_Summ
erfield_226730_1961_24000_geo', 'OH_Summerfield_226731_1961_24000_geo', 'OH_Summerfield_228206_2002_24000_geo']
```

In [13]:
```
#Make chips for each topo map
for cr in chip_n:
    #Set workspace
    arcpy.env.workspace = in_fold + in_toposF
    #Set local variables and make folders
    quadNm = cr.split("_")[1].replace(" ", "_")
    os.mkdir(chips_outF + cr)
    subdir = chips_outF + cr + "/"
    out_folder=subdir
    #Read in topo
    inRaster = in_fold + in_toposF + cr   + ".tif"
    #Copy topo to 8-bit PNG
    inRaster2 = arcpy.CopyRaster_management(inRaster, to_8bitF + cr + ".png",
                                "", "",256,"NONE","NONE","8_BIT_UNSIGNED","NONE","NONE", "PNG",
                                  "NONE")
    #Read in Mines
    in_mines = in_fold + in_minesF + cr   + ".shp"
    in_training =  in_fold + in_minesF + cr   + ".shp"
    #Add and populate class value field if it doesn't already exist
    if 'classvalue' not in [f.name for f in arcpy.ListFields(in_training)]:
        arcpy.AddField_management(in_training, "classvalue", "SHORT")
        arcpy.CalculateField_management(in_training, "classvalue", 1, "PYTHON3")
    #Define image chip parameters
    image_chip_format = "PNG"
    tile_size_x = "128"
    tile_size_y = "128"
    stride_x= "128"
    stride_y= "128"
    output_nofeature_tiles= "ALL_TILES"
    metadata_format= "Classified_Tiles"
    start_index = 0
    classvalue_field = "classvalue"
    buffer_radius = 0
    in_mask_polygons = in_fold + in_quadsF + cr + ".shp"
    rotation_angle = 0
    reference_system = "MAP_SPACE"
    processing_mode = "PROCESS_AS_MOSAICKED_IMAGE"
    blacken_around_feature = "NO_BLACKEN"
    crop_mode = "FIXED_SIZE"

    # Create image chips
    ExportTrainingDataForDeepLearning(inRaster2, out_folder, in_training,
                 image_chip_format,tile_size_x, tile_size_y, stride_x,
                 stride_y,output_nofeature_tiles, metadata_format, start_index,
                 classvalue_field, buffer_radius, in_mask_polygons, rotation_angle,
                 reference_system, processing_mode, blacken_around_feature, crop_mode)
    #Make new labels and image directory
    os.mkdir(subdir + "labels2")
    os.mkdir(subdir + "images2")
    arcpy.env.workspace = subdir + "images"
    imgchips = arcpy.ListRasters()
    #Copy all only background chips and make 0 masks.
    for ic in imgchips:
        if os.path.isfile(subdir + "labels/" + ic):
            pass
        else:
            makeZeros = IsNull(subdir + "images/" + ic)
            arcpy.CopyRaster_management(subdir + "images/" + ic, subdir + "images2/" + ic,
                                    "", "",256,"NONE","NONE","8_BIT_UNSIGNED","NONE","NONE", "PNG",
                                    "NONE")
            arcpy.CopyRaster_management(makeZeros, subdir + "labels2/" + ic,
                                    "", "",256,"NONE","NONE","8_BIT_UNSIGNED","NONE","NONE", "PNG",
                                    "NONE")
```

In [2]:
```python
import arcpy
arcpy.CheckOutExtension("ImageAnalyst")
from arcpy.ia import *
from arcpy import env
from arcpy.sa import *
import os
import os.path

in_fold = "E:/topo_proj/topo_check/"
in_toposF = "va_needed_topos2/"
in_minesF = "va_topo_mines2/"
in_quadsF = "va_topo_quads/"
chips_outF = in_fold + "processing/va_chips_background/"
to_8bitF = in_fold + "processing/va_topo8bit_background/"
#os.mkdir(chips_outF)
#os.mkdir(to_8bitF)

arcpy.env.workspace =  in_fold + in_toposF
chips = arcpy.ListRasters()
print(chips)
```

```
['VA_Coeburn_184602_1957_24000_geo.tif', 'VA_Coeburn_184605_1957_24000_geo.tif', 'VA_Duty_184823_1958_24000_geo.tif', 'VA_Duty_184825_1958_24000_geo.tif', 'VA_Grundy_185240_1963_24000_
geo.tif', 'VA_Grundy_185241_1963_24000_geo.tif', 'VA_Haysi_185326_1963_24000_geo.tif', 'VA_Haysi_185327_1963_24000_geo.tif', 'VA_Honaker_185407_1968_24000_geo.tif', 'VA_Honaker_185409_
1968_24000_geo.tif', 'VA_Nora_186109_1958_24000_geo.tif', 'VA_Nora_8031190_1958_24000_geo.tif', 'VA_Norton_186163_1957_24000_geo.tif', 'VA_Norton_186165_1957_24000_geo.tif', 'VA_Pound_
186331_1957_24000_geo.tif', 'VA_Pound_186333_1957_24000_geo.tif', 'VA_Prater_186345_1963_24000_geo.tif', 'VA_Prater_186347_1963_24000_geo.tif', 'VA_Richlands_186494_1968_24000_geo.ti
f', 'VA_Richlands_186496_1968_24000_geo.tif', 'VA_Vansant_187042_1963_24000_geo.tif', 'VA_Vansant_187044_1963_24000_geo.tif', 'VA_Wise_187272_1957_24000_geo.tif', 'VA_Wise_187273_1957_
24000_geo.tif', 'VA_Wise_187274_1957_24000_geo.tif']
```

In [4]:
```python
chip_n = list()
for c in chips:
    c1 = os.path.basename(c)
    c2 = os.path.splitext(c1)[0]
    chip_n.append(c2)
print(chip_n)
print(len(chip_n))
```

```
['VA_Coeburn_184602_1957_24000_geo', 'VA_Coeburn_184605_1957_24000_geo', 'VA_Duty_184823_1958_24000_geo', 'VA_Duty_184825_1958_24000_geo', 'VA_Grundy_185240_1963_24000_geo', 'VA_Grundy
_185241_1963_24000_geo', 'VA_Haysi_185326_1963_24000_geo', 'VA_Haysi_185327_1963_24000_geo', 'VA_Honaker_185407_1968_24000_geo', 'VA_Honaker_185409_1968_24000_geo', 'VA_Nora_186109_195
8_24000_geo', 'VA_Nora_8031190_1958_24000_geo', 'VA_Norton_186163_1957_24000_geo', 'VA_Norton_186165_1957_24000_geo', 'VA_Pound_186331_1957_24000_geo', 'VA_Pound_186333_1957_24000_ge
o', 'VA_Prater_186345_1963_24000_geo', 'VA_Prater_186347_1963_24000_geo', 'VA_Richlands_186494_1968_24000_geo', 'VA_Richlands_186496_1968_24000_geo', 'VA_Vansant_187042_1963_24000_ge
o', 'VA_Vansant_187044_1963_24000_geo', 'VA_Wise_187272_1957_24000_geo', 'VA_Wise_187273_1957_24000_geo', 'VA_Wise_187274_1957_24000_geo']
25
```

In [5]:
```python
chip_n2 = chip_n[23:]
print(chip_n2)
```

```
['VA_Wise_187273_1957_24000_geo', 'VA_Wise_187274_1957_24000_geo']
```

In [6]:
```python
for cr in chip_n2:
    # Set local variables
    arcpy.env.workspace = in_fold + in_toposF
    quadNm = cr.split("_")[1].replace(" ", "_")
    os.mkdir(chips_outF + cr)
    subdir = chips_outF + cr + "/"
    out_folder=subdir
    inRaster = in_fold + in_toposF + cr  + ".tif"
    inRaster2 = arcpy.CopyRaster_management(inRaster, to_8bitF + cr + ".png",
                                "", "",256,"NONE","NONE","8_BIT_UNSIGNED","NONE","NONE", "PNG",
                                  "NONE")
    in_mines = in_fold + in_minesF + cr  + ".shp"
    in_training =  in_fold + in_minesF + cr  + ".shp"
    if 'classvalue' not in [f.name for f in arcpy.ListFields(in_training)]:
        arcpy.AddField_management(in_training, "classvalue", "SHORT")
        arcpy.CalculateField_management(in_training, "classvalue", 1, "PYTHON3")
    image_chip_format = "PNG"
    tile_size_x = "128"
    tile_size_y = "128"
    stride_x= "128"
    stride_y= "128"
    output_nofeature_tiles= "ALL_TILES"
    metadata_format= "Classified_Tiles"
    start_index = 0
    classvalue_field = "classvalue"
    buffer_radius = 0
    in_mask_polygons = in_fold + in_quadsF + cr + ".shp"
    rotation_angle = 0
    reference_system = "MAP_SPACE"
    processing_mode = "PROCESS_AS_MOSAICKED_IMAGE"
    blacken_around_feature = "NO_BLACKEN"
    crop_mode = "FIXED_SIZE"

    # Execute
    ExportTrainingDataForDeepLearning(inRaster2, out_folder, in_training,
                 image_chip_format,tile_size_x, tile_size_y, stride_x,
                 stride_y,output_nofeature_tiles, metadata_format, start_index,
                 classvalue_field, buffer_radius, in_mask_polygons, rotation_angle,
                 reference_system, processing_mode, blacken_around_feature, crop_mode)
    os.mkdir(subdir + "labels2")
    os.mkdir(subdir + "images2")
    arcpy.env.workspace = subdir + "images"
    imgchips = arcpy.ListRasters()
    for ic in imgchips:
        if os.path.isfile(subdir + "labels/" + ic):
            pass
        else:
            makeZeros = IsNull(subdir + "images/" + ic)
            arcpy.CopyRaster_management(subdir + "images/" + ic, subdir + "images2/" + ic,
                                "", "",256,"NONE","NONE","8_BIT_UNSIGNED","NONE","NONE", "PNG",
                                "NONE")
            arcpy.CopyRaster_management(makeZeros, subdir + "labels2/" + ic,
                                "", "",256,"NONE","NONE","8_BIT_UNSIGNED","NONE","NONE", "PNG",
                                "NONE")
```

In [ ]:
```python

```