Map scrpiting

- ArcPy mapping module: the ArcPy mapping module, arcpy.mapping, helps automating mapping tasks, including managing map documents, data frames, layer files, and the data within files.
  - To open map documents: MapDocument(mdx_path) or MapDocument(“CURRENT”). Script that use CURRENT keyword must be run from within ArcMap
  - Once the MapDocument object is created, you can modify the properties of map document. To save the changes, use save or saveACopy. saveACopy will save the mxd to a previous version

```
import arcpy
mapdoc=arcpy.mapping.MapDocument(“c:/data/study_area.mxd”)
<code that modify map document properties>
mapdoc.save()
del mapdoc
```

- Accessing map document properties and methods
  - The properties of a MapDocument object include most of the properties on the Map Document Properties dialog box on ArcMap.
  - Mthods: save and saveACopy; deleteThumbnail and makeThumbnail; findAnd ReplaceWorkspacePaths and replaceWorkspaces;

```
import arcpy
mapdoc=arcpy.mapping.MapDocument(“CURRENT”)
path=mapdoc.filepath
print path
mapdoc.title=“Final Map”
mapdoc.save()
del mapdoc
```

- Working with data frames
  - ListDataFrames (map_document, {wild_card}) returns a list of DataFrame objects in a map document.
  - Use index to access individual DataFrame object. The order of a list of data frames is the same as the order used in the ArcMap TOC.

```
import arcpy
mapdoc=arcpy.MapDocument("CURRENT")
listdef=arcpy.mapping.ListDataFrames(mapdoc)
for df in listdf
  print df.name
print listdef[0].name
del mapdoc
```

- Working with data frames
  - DataFrame object has many properties such as map extent, scale, rotation, and spatial reference, use map units,etc.
  - Scripting does not provide access to all the properties in the Data Frame Properties dialog box, and conversely, some DataFrame object properties are not on the Data Frame Properties dialog box. For example, the scale of a data frame can be set using scripting, but in ArcMap, it is accomplished by using a tool on the Standard toolbar.

```
import arcpy
dataset=“c:/map/boundary.shp”
spatialRef=arcpy.Describe (dataset).spatialReference
mapdoc=arcpy.mapping.MapDocument(“c:/map/final.mxd”)
for df in arcpy.mapping.ListDataFrames(mapdoc)
      df.spatialReference=spatialRef
      df.scale=24000
del mapdoc
```

Working with layers

- The Layer object provides access to many different layer properties and methods. There are two ways to reference Layer object:
  - Use the Layer function to reference a layer(.lyr) file on disk.

    ```
    lyr=arcpy.mapping.Layer(“c:/Mapping/study.lyr”)
    ```

  - Use the ListLayer function to reference the layers in an .mxd file, or just layers in a particular data frame in a map document, or the layers within a .lyr file

    ```
    import arcpy
    myDoc=arcpy.mapping.MapDocument(“CURRENT”)
    // dflist=arcpy.mapping.ListDataFrames(mapdoc)
    lyrlist=arcpy.mapping.ListLayers(mapdoc)
    //lyrlist=arcpy.mapping.ListLayers(mapdoc, “”,dflist[0])
    for lyr in lyrlist
        print lyr.name
    ```

- Working with layers
  - Three of the layer categories are commonly used: feature layers, raster layers, and group layers.
  - Layer objects have a number of properties, including the name of the layer, the name of the layer dataset, the ability to set a definition query, the ability to turn on the display of labels, and a number of display properties, such as brightness, contrast, and transparency.
  - Layer object has some methods: save and saveACopy; findAndReplaceWorkspacePath and replaceDataSource
  - Some functions of ArcPy mapping module can work with layers: AddLayer, AddLayerToGroup, InsertLayer, MoveLayer, RemoveLayer, UpdateLayer

- Working with page layout elements
  - ListLayoutElements (map_document, {element_type}, {wild_card}): returns a Python list of elements. element-type include DATAFRAME_ELEMETN, GRAPHIC_ELEMENT,LEGEND_ELEMENT,MAPSURROUND_ELEMENT,PICTURE_ELEMENT, TEXT_ELEMENT

```
import arcpy
mapdoc=arcpy.mapping.MapDocument(r “c:\mydata\project.mxd”)
elemlist=arcpy.mapping.ListLayoutElements(mapdoc)
for elm in elemlist:
  print elem.name & “   “& el.type
del mapdoc
```

- Working with page layout elements
  - A specific element can be selected by using
    - The index number of the element; the element_type parameter; the wild_card parameter

      title=arcpy.mapping.ListLayoutElements(mapdoc)[3]

      title=arcpy.mapping.ListLayoutElements(mapdoc, “TEXT_ELEMENT”)[0]

      title=arcpy.mapping.ListLayoutElements(mapdoc, “”, “Title”)[0]

Working with page layout elements

- Once a specific element is referenced, various properties can be accessed, such as element’s name, type, height, and width.

```
import arcpy
mapdoc=arcpy.mapping.MapDocument(“c:/mydata/project.mxd)
title=arcpy.mapping.ListLayoutElements(mapdoc,
“TEXT_ELEMENT”) [0]
title.text=“New Study Area”
mapdoc.save()
del mapdoc
```

- Working with page layout elements

```
import arcpy
mapdoc=arcpy.mapping.MapDocument(“CURRENT”)
df=arcpy.mapping.ListDataFrames(mapdoc)[0]
lyr1=arcpy.mapping.Layer(“c:/mydata/streets.lyr”)
lyr2=arcpy.mapping.Layer(“c:/mydata/ortho.lyr”)
legend=arcpy.mapping.ListLayoutElements(mapdoc,
“LEGEND_ELEMENT”)[0]
legend.autoAdd=True
arcpy.mapping.AddLayer(df, lyr1, “BOTTOM”)
legend.autoAdd=False
arcpy.mapping.AddLayer(df, lyr2, “BOTTOM”)
mapdoc.save()
del mapdoc
```