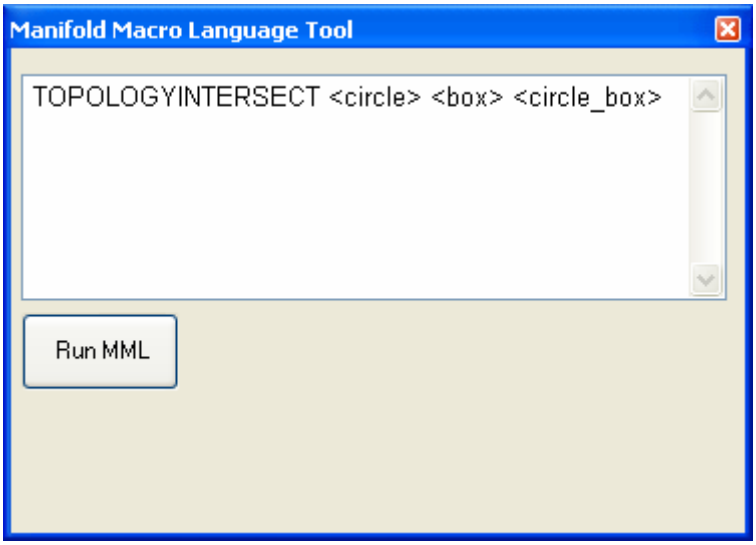
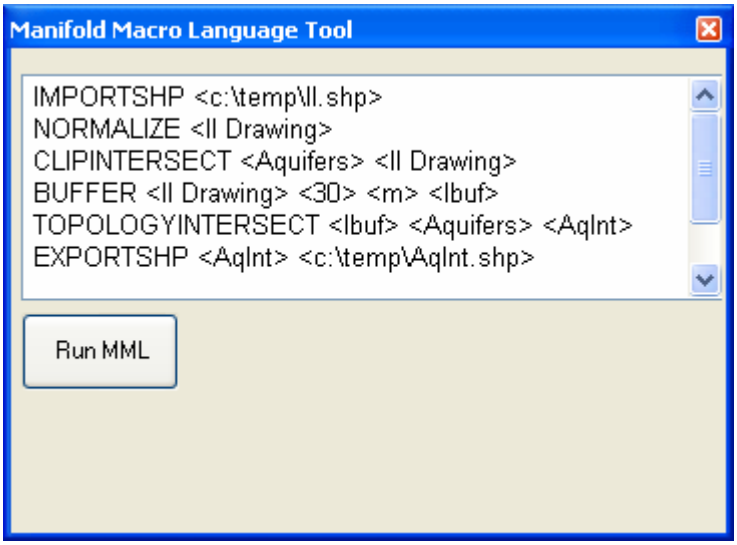
INSTALLATION OF THE MANIFOLD MACRO LANGUAGE

Installing the Manifold Macro Language (MML) .DLL is similar to the installation of any third party .dll in Manifold. In the **`C:\PROGRAM FILES\MANIFOLD SYSTEM\CONFIG`** directory you need to place the `MML.XML, MML.DLL, and MML.PNG` file in a directory called `MML`.

MML runs inside of Manifold, as a dockable pane, and executes when the user presses the Run MML button.



Multiple commands can be performed in sequence by entering the new commands on additional lines.




```
Imports a shapefile named ll.shp
Normalizes topology for the drawing
Clips ll with Aquifers
Buffers ll by 30 meters
Intersects lbuf and Aquifers
Exports resulting file to a .shp
```

SUPPORTED COMMANDS FOR THE MANIFOLD MACRO LANGUAGE

Currently, the Manifold Macro Language supports over unique 20 operations. However, with the incorporation of the `SQL` and `TRANSFORMWITH` macro commands, users have the ability to literally perform hundreds of Manifold based operations.

This section provides a brief overview of the commands used within the gisadvisor.com MML.

Vector Operations

Normalize Topology

Syntax: `NORMALIZE <Drawing>`
Example: NORMALIZE <Land Cover Drawing>

Bounded Areas

Syntax: `BOUNDEDAREAS <Drawing>`
Example: BOUNDEDAREAS <Land Cover Drawing>

Decompose a Drawing

Syntax: `DECOMPOSE <Drawing>`
Example: DECOMPOSE <Land Cover Drawing>

Convert Table to a Drawing

This command allows a user to turn a table with a geometry field into a new drawing. This is useful when users create new tables with SQL commands.

In this example, we have a table named drwtable with a geometry field called geomfld, and another field called LC.

Syntax: `TABLEDRW <Table Name> <Drawing Name> <Columns>` **`note: columns must start with the geometry column first>`**
Example: TABLEDRW <drwtable> <newdrw> <geomfld,LC>

Finding the nearest objects in one drawing to the objects in another drawing

In this example, for each customer we find the nearest ATM location and the distance between them

***Syntax**:* `NEAR <Drawing1> <Drawing2>`
***Example**: NEAR <Customer Locations> <ATM Locations>*

Clipping one drawing with another drawing

In this example, we clip the land cover drawing with the features in the study boundary drawing.

Syntax: `CLIPINTERSECT <Drawing1> <Drawing2>`
Example: CLIPINTERSECT <Land Cover Drawing> <Study Boundary>

Topological Intersection of two drawings

In this example, we perform a topological intersection operation between two drawings. Note that the result is a table with a geometry field called geomfld. This table can then be converted to a drawing using TABLEDRW.

Syntax: `TOPOLOGYINTERSECT <Drawing1> <Drawing2> <new drawing>`
Example: TOPOLOGYINTERSECT <Land Cover Drawing> <Soils> <Land_Soil_Intersect>

Buffer features in a drawing

In this example, we buffer the land cover drawing by 500 meters, and place the results into a new drawing named LandBuf

Syntax: `BUFFER <Drawing1> <Distance> <units> <NewDrawing>`
Example: BUFFER <Land Cover Drawing> <500> <m> <LandBuf>

Copy-Paste-Delete Operations

Deleting a component

This command works similar to the ESRI KILL command. It allows the user to delete any component in the .map project.
Syntax: `KILL <Component Name>`
Example: KILL <Land Cover Drawing>

Copy a drawing to another drawing

In addition to copying one drawing to another drawing, this command also appends the data from `<Drawing1>` into `<Drawing2>`. Therefore, users can issue the COPYDRW command multiple times, pasting numerous drawings into an existing drawing. In other words, this is a great way to merge drawings together.

Syntax: `COPYDRW <Drawing1> <Drawing2>`
Example: COPYDRW <Land Cover Drawing> <Land Cover Drawing 2>

Copy a surface to another surface

In addition to copying one surface to another surface, this command also appends the data from `<Surface 1>` into `<Surface 2>`. Therefore, users can

issue the COPYSURF command multiple times, pasting numerous surfaces into an existing surface. In other words, this is a great way to merge surfaces together.

Syntax: `COPYSURF <Surface 1> <Surface 2>`
Example: COPYSURF <Land Cover Surface> <Land Cover Surface 2>

Copy an image to another image

In addition to copying one image to another image, this command also appends the data from `<Image 1>` into `<Image 2>`. Therefore, users can issue the COPYIMG command multiple times, pasting numerous images into an existing image. In other words, this is a great way to merge images together.

Syntax: `COPYIMG <Image 1> <Image 2>`
Example: COPYIMG <Satellite Image> <Satellite Image 2>

Raster Operations

This command exposes the entire Surface Transform Dialog box to the user. In the two following examples, we are converting a digital elevation model into a slope map, and also performing mathematical operations on two different surfaces. The `TRANSFORMWITH` command performs all operations on the input surface. If you don't want to overwrite your original surface, use the `COPYSURF` command first, to create a new surface.

Syntax: `TRANSFORMWITH <Surface> <Transform Dialog text>`
Examples: TRANSFORMWITH <DEM> <slope[DEM]>
TRANSFORMWITH <Land Cover Surface> <(([Land Cover Surface]) * 2) / slope([DEM])

Query Operations

Using freeform SQL

This command exposes the entire Manifold SQL language into a macro command. Therefore, users can select, update, transform, and even create new database tables.

Syntax: `SQL <SQL Statement>`
Examples: SQL <SELECT * FROM [Land Cover Drawing] WHERE LC = "A">
SQL <UPDATE [Land Cover Drawing] SET LC = "Ag" WHERE LC = "A">

Import-Export Functions

Syntax: `EXPORTSHP <Drawing> <filename>`
Examples: EXPORTSHP <Land Cover Drawing> <c:\temp\lc.shp>

Other Import-Export Syntax:

Syntax: `EXPORTKML <Drawing> <filename>`

Syntax: `EXPORTE00 <Drawing> <filename>`

Syntax: `EXPORTCSV <Table> <filename>`

Syntax: `EXPORTDXF <Drawing> <filename> <Z column name>`
If no Z column, then leave the value blank: <>

Syntax: `EXPORTXLS <Table> <filename>`

Syntax: `EXPORTECW <Drawing> <filename>`

Syntax: `IMPORTSHP <filename>`

Syntax: `IMPORTSID <filename>`

Syntax: `IMPORTE00 <filename>`

Syntax: `IMPORTTIF <filename>`

RUNNING MML IN SEQUENCE

In this section we will describe a number of ways to make use of the Manifold Macro Language to run a sequence of commands. The examples will include:

- Sequence of SQL commands
- Merging TIF images into a large image file and exporting to ECW
- Merging shapefiles together and export to a larger shapefile
- Building a raster-to-vector converter

Merge TIF together and export to a larger ECW

This example brings in four images, merges them into a single image, and then exports them out as an ECW image. If you want to save space, you can issue a KILL command to get rid of the images that are brought in. Please note that a file named east.tif gets renamed East Image.

```
IMPORTTIF <c:\farm\east.tif>
COPYIMG <East Image> <WholeFam>
IMPORTTIF <c:\farm\west.tif>
COPYIMG <West Image> <WholeFarm>
IMPORTTIF <c:\farm\north.tif>
COPYIMG <North Image> <WholeFarm>
IMPORTTIF <c:\farm\south.tif>
COPYIMG <South Image> <WholeFarm>

EXPORTECW <WholeFarm> <c:\temp\WholeFarm.ecw>
```

Merge Shapefiles together and export to a larger shapefile
This example brings in four shapefiles, merges them into a single drawing, and then exports them out as a shapefile. If you want to save space, you can issue a KILL command to get rid of the shapefiles that are brought in. Please note that a file named east.shp gets renamed East Drawing.

```
IMPORTSHP <c:\farm\east.shp>
COPYDRW <East Drawing> <WholeFam>
IMPORTSHP <c:\farm\west.shp>
COPYDRW <West Drawing> <WholeFarm>
IMPORTSHP <c:\farm\north.shp>
COPYDRW <North Drawing> <WholeFarm>
IMPORTSHP <c:\farm\south.shp>
COPYDRW <South Drawing> <WholeFarm>

EXPORTSHP <WholeFarm> <c:\temp\WholeFarm>
```

Raster-to-Vector

This is a really cool command sequence! Please note that you must not have any carriage returns within the SQL statement. But, consider what it does:

- the first SQL command creates a new drawing
- the second SQL command takes every pixel coordinate in the surface, and the associated Height value and:
    - converts it to a point,
    - Buffers the point by a distance equal to the pixel height
    - Creates a bounding box of the buffer
    - Then groups the geometric objects by their Height value
- The next command Normalizes the topology of all those boxes we just created
- The final command decomposes the boxes into individual grouped areas

```
SQL <CREATE DRAWING [newdrawing] (val real)>
SQL <insert into [newdrawing] ([Geom (I)], [val]) select geomfld, val
from (select * from (select unionall(geomfld) geomfld,val FROM (SELECT
boundingbox(Buffer(newpoint([Easting (I)],[Northing (I)]),50))
geomfld,[Height (I)] as val FROM [Land 2]) GROUP BY val))
NORMALIZE <newdrawing>
DECOMPOSE <newdrawing>
```

So here you can see the power of stringing together these simple macro commands.

© www.gisadvisor.com, LLC