Scripting

Automated Geospatial Processing with TNTscript

TNTscript is designed for automated production processing of geospatial data using scripts written in the TNTgis geospatial processing language (SML). TNTscript enables SML scripts to be executed on computers that do not have any other professional TNTgis product installed. If you develop SML processing scripts using TNTmips Pro or TNTedit Pro, TNTscript allows you to execute these scripts on additional computers, at remote sites, or using cloud-computing resources. TNTscript can also be used separately from the other TNTgis professional products to write, edit, and execute SML programs.

TNTscript executes SML scripts without user interaction during processing. All data-specific variables are passed to the SML script by TNTscript at the start of execution. The script to be used and the required variable values can be specified on the command line or in an XML-formatted text file (.smlx).

A TNTscript license includes the runtime TNTscript executable (with its own installer) and a professional license for TNTview (optional installation using the TNTgis installer). TNTview used with the TNTscript software license key provides access to the SML Script Editor for writing, editing, and testing SML scripts for use with TNTscript. The built-in Script Reference documentation indicates which functions and classes (those related to interactive components such as pop-up dialogs, View windows, etc.) should not be used for scripts to be executed in TNTscript. Fully interactive scripts can be written and executed from TNTview.

Runtime TNTscript is a Windows or Mac executable program that provides all of the non-interactive SML processing functionality found in TNTmips Pro. It does not have a user interface, but instead is launched from the command line manually or by another program. Status messages from the TNTscript program by default are printed to the console but can be saved instead to a designated log file. The output of print statements and error messages in the SML script are printed to the console or can be redirected to a designated text file.

Command Line Syntax

TNTscript can be run manually or programmatically from the command line in any of the following ways:

```
1) tntscript scriptfile parm1:“value” parm2:“value” ... parmX:“value”
2) tntscript scriptfile parm1:“value” parm2:“value” > messagefile
3) tntscript scriptfile -log logfile parm1:“value” parm2:“value”
4) tntscript smlxfile
5) tntscript smlxfile > messagefile
```

where: `scriptfile` = the path to the SML script to be executed

`logfile` = the path to a log file

`parmX` = the name of a process variable

`“value”` = the value of the process variable

`messagefile` = the path to a text file in which to save the output of SML script messages

`smlxfile` = the path to an SMLX file specifying the SML script and the name and values of variables

In examples 1 through 3 the path to the SML script to be executed as well as the name and value of each processing parameter are passed to TNTscript on the command line. In example 1 the output of any script print statements and error messages is printed to the console, while in example 2 the script console output is redirected to a specified text file. In example 3 a log file is specified to save status messages from the TNTscript program (while script print output is printed to the console).

In examples 4 and 5 the name of an SMLX file is passed to TNTscript; this file contains the name of the script, the name of a log file (optional), and the names and values of script variables. Example 5 also redirects the script console output to a text file.

Examples for a simple script that prints a message string to a file are shown below.

```
1) tntscript  C:/test/messagescript.sml message:“Hello World 1234.”
      file:“C:/test/message.txt”

2) tntscript  C:/test/messagescript.sml message:“Hello World 1234.”
      file:“C:/test/message.txt” > c:/test/scriptmessages.txt

3) tntscript  C:/test/tntscript.sml -log E:/test/tntscript.log message:“Hello World 1234.”
      file:”C:/test/message.txt”

4) tntscript C:\test\message.smlx

5) tntscript C:\test\message.smlx > c:/test/scriptmessages.txt
```

(continued)

SMLX File Format

The SMLX file has a simple XML format that specifies the SML script to be called, (optionally) the location of a log file, and the values of the required parameters. The following is an SMLX file for the message script used in the previous syntax examples.

```
<smlx>
     <script>E:/test/messagescript.sml</script>
     <log>E:/test/messagescript.log</log>
     <parm name=“message”>Hello World 1234.</parm>
     <parm name=“file”>E:/test/messagescript.txt</parm>
</smlx>
```

An SMLX file has a root element <smlx>. The <script> element can contain either the path to the SML script file to use for processing (as in the example above) or the script text itself (shown in the example below). An optional <log> element specifies the path to a log file that records status messages from TNTscript. Each parameter to be passed to the SML script is specified in a <parm> element. The *name* attribute of the <parm> element specifies the variable’s name in the script, and the value of the element specifies the value for the variable.

```
<smlx>
     <script>
     # fprintf
     string file;
     string message;
     # Create a file, then write to the file, then close it
     class FILE f;
     f=fopen(file, “w”);
     fprint(f, message);
     fclose(f);
     print(“console test”);
     </script>
     <parm name=“message”>Hello World 1234.</parm>
     <parm name=“file”>E:/test/tntscript.txt</parm>
</smlx>
```

An example SMLX file for a script that converts TIFF images to JP2 files is shown below. In this example the inputPath$ and outputDir$ variables specify the name of the input TIFF file and the directory in which to write the output JP2 file (respectively). The compType$ variable specifies the type of JPEG2000 compression use for the JP2 file; the assigned value “user” specifies user-defined lossy compression. The compRatio variable sets the compression ratio for lossy compression.

```
<smlx>
     <script>C:\test\tntscript\TiffToJP2tntscript.sml</script>
     <log>C:/test/tntscript/TiffToJP2.log</log>
     <parm name=“inputPath$”>G:/Arizona/AzTIFF/3110901_se.tif</parm>
     <parm name=“outputDir$”>G:/Arizona/scriptout</parm>
     <parm name=“compType$”>user</parm>
     <parm name=“compRatio”>15.0</parm>
</smlx>
```

SML Script Structure for TNTscript

A script designed to run under TNTscript has a form similar to that used for a script designed to run under the TNTgis Job Processing System (see the Technical Guide entitled *Custom Job Processing with Geospatial Scripts*). All basic data- and location-specific variables are simply declared in the script. The names and values for these variables are then passed to the SML script by TNTscript as specified on the command line or in an SMLX file.

Implementation

TNTscript allows powerful and efficient TNTgis geospatial processes to be integrated into complex automated workflows that utilize multiple software products and custom programs. A custom program monitoring the workflow can call TNTscript as needed and specify the name of an SML script and the required script parameters on the command line or by creating and specifying an SMLX file. The data produced by this processing script can then be passed along to the next step in the overall workflow. Network licenses for TNTscript are available to allow TNTscript processing to be distributed over multiple networked computers or use cloud-computing resources.