Command Reference: WriteTableToExcel()

Write a table to a Microsoft Excel workbook file

Version 11.08.00, 2016-01-17

The `WriteTableToExcel()` command writes a table to a worksheet in a Microsoft Excel workbook file. A contiguous block of cells (rectangle) must be specified in one of the following ways to receive the table:

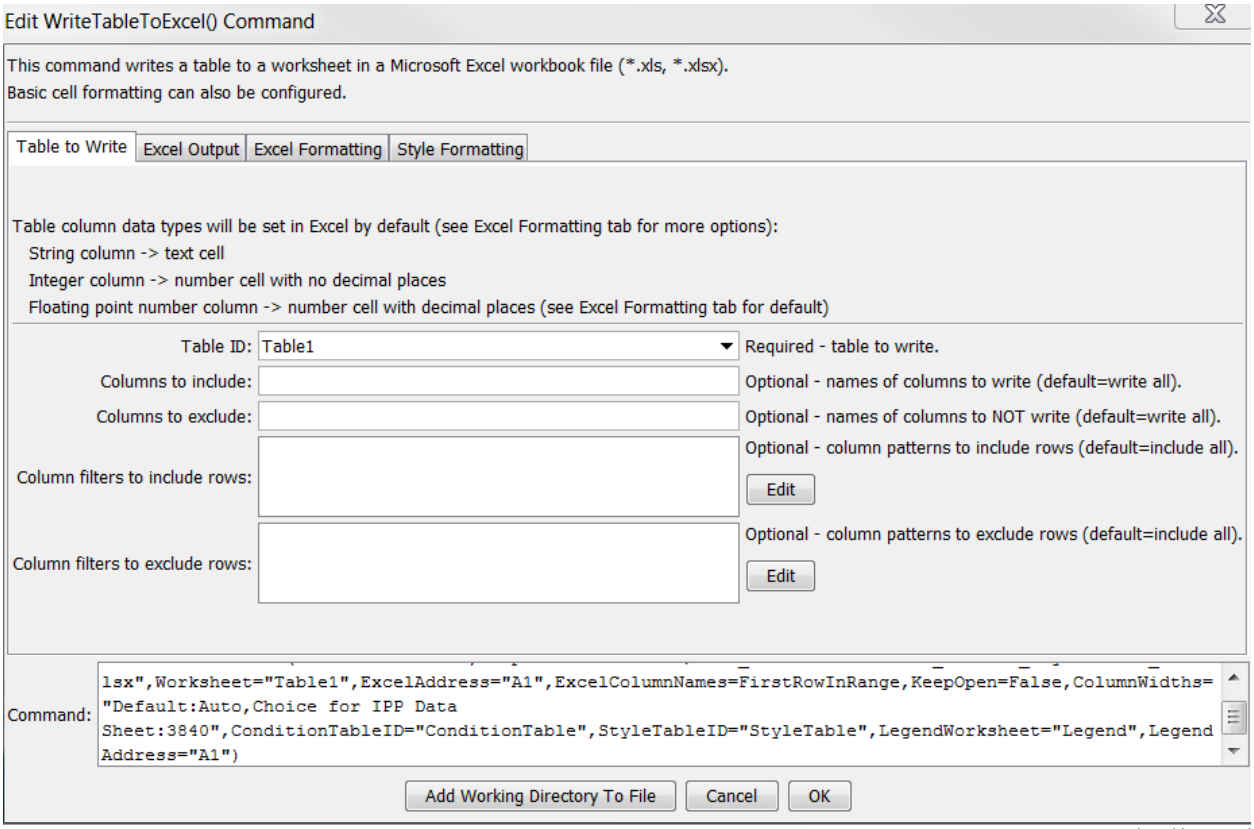
- Specify the upper-left cell in a range of cells using Excel address notation (e.g., `A1`)
- Specify a range of cells using Excel address notation (e.g., `A1:D10`)
- Specify the name of an Excel named range.
- Specify a table name (essentially a named range).

TSTool uses the Apache POI software (http://poi.apache.org) to read/write the Excel file and consequently functionality is constrained by the features of that software package.

The following are limitations of this command:

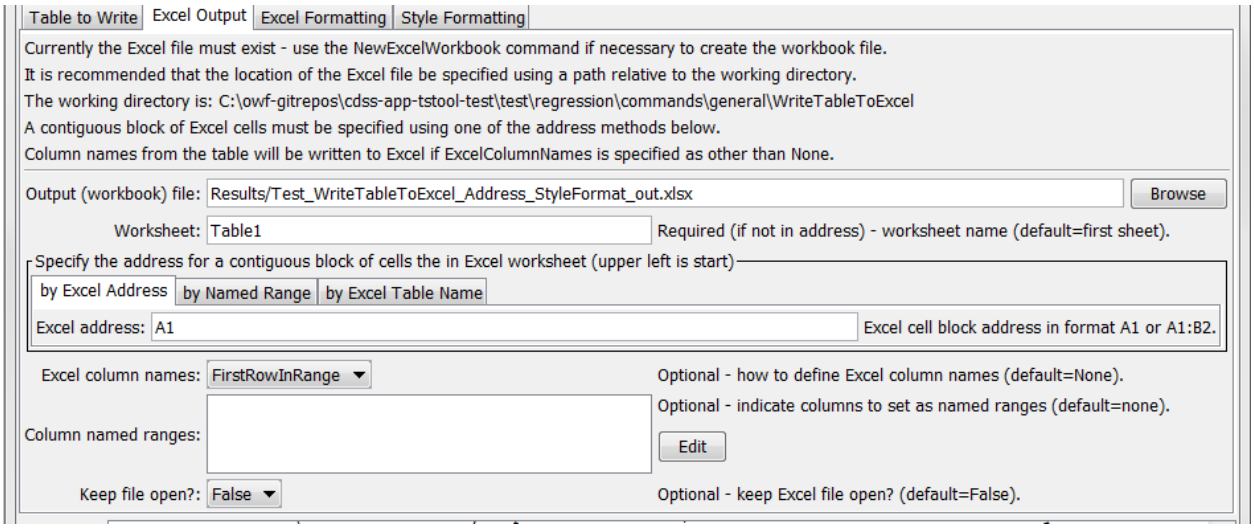
- Currently only the upper-left cell is utilized but in the future the range may be used to limit output.
- Cell data types are determined from the table columns being written. In the future a parameter may be provided to allow option of using original Excel formatting.

The following figures illustrate the dialog used to edit the command and the syntax for the command.



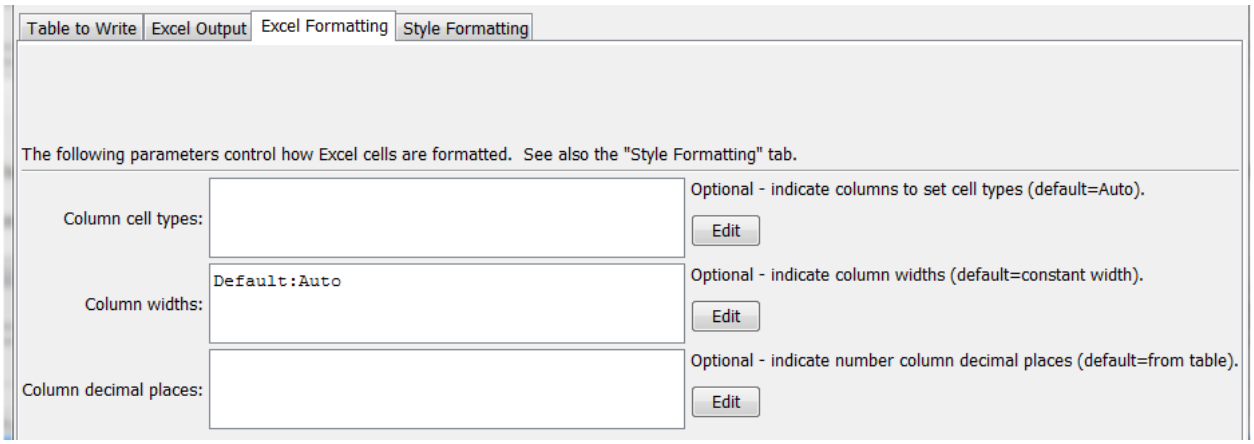
WriteTableToExcel

WriteTableToExcel() Command Editor



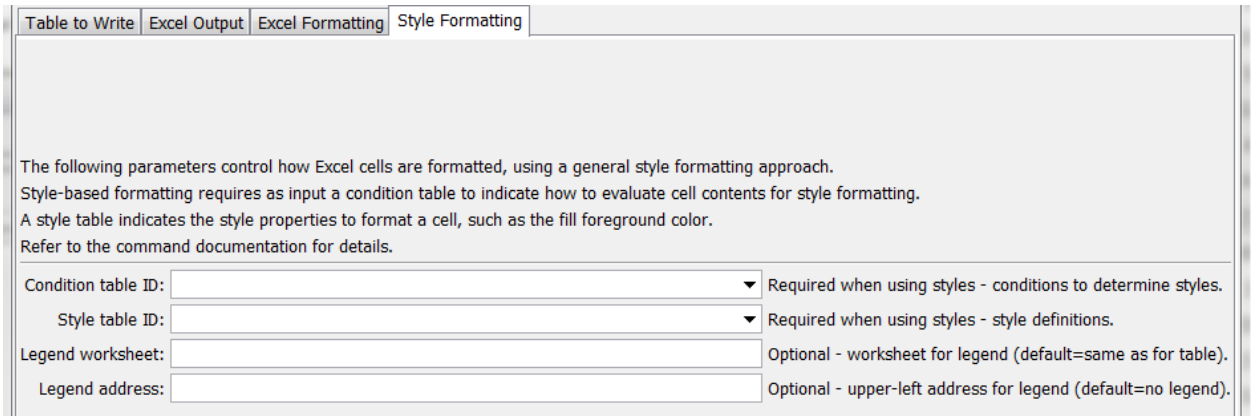
WriteTableToExcel_Excel

WriteTableToExcel() Command Editor for Excel Output Parameters



WriteTableToExcel_ExcelFormat

WriteTableToExcel() Command Editor for Excel Formatting Parameters



WriteTableToExcel_Style

WriteTableToExcel() Command Editor for Style Formatting Parameters

The command syntax is as follows:

```
WriteTableToExcel(Parameter=Value,…)
```

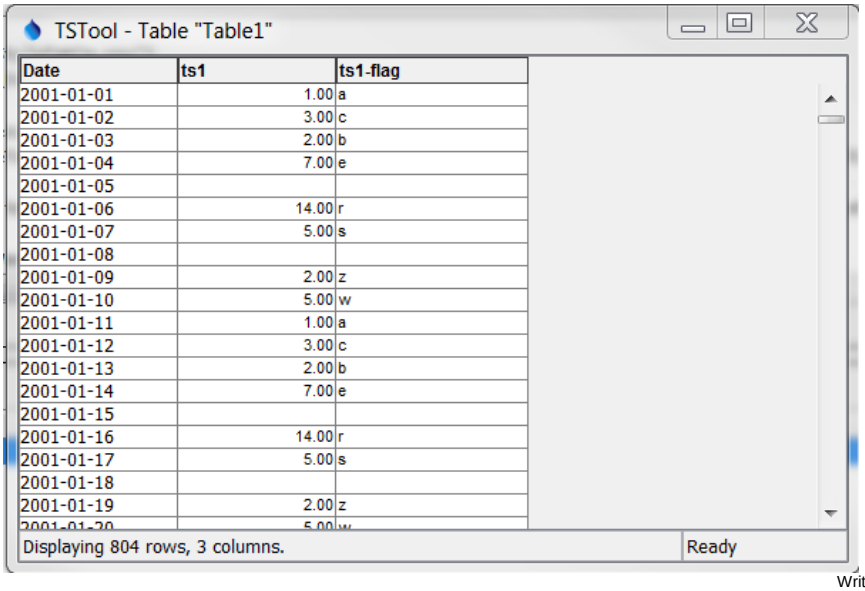
Command Parameters

Parameter	Description	Default
`TableID`	Identifier for table to write. Can specify using processor `${Property}`.	None – must be specified.
`IncludeColumns`	Names of columns in table to write.	Write all columns.
`ExcludeColumns`	Names of columns in table to exclude from writing.	Write all columns.
`ColumnInclude Filters`	Indicate table column names and pattern to use to include rows. For example, include rows with blanks in columns. The format of the parameter is: `ColumnName1:Pattern1, ColumnName2:Pattern2,…` where patterns can contain `*` to match a substring.	Include all rows.
`ColumnExclude Filters`	Indicate table column names and pattern to use to exclude rows. For example, exclude rows with blanks in columns. The format of the parameter is:	Include all rows.

Parameter	Description	Default
	`ColumnName1:Pattern1, ColumnName2:Pattern2,…` where patterns can contain `*` to match a substring.	
`OutputFile`	The name of the Excel workbook file (*.*xls* or *.*xlsx*) to write, as an absolute path or relative to the command file location. If the Excel file does not exist it will be created. Can specify using processor `${Property}`.	None – must be specified.
`Worksheet`	The name of the worksheet in the workbook to write. If the worksheet does not exist it will be created. Can specify using processor `${Property}`.	Write to the first worksheet.
`ExcelAddress`	Indicates the block of cells to write, using Excel address notation (e.g., `A1:D10`).	Must specify address using one of available address parameters.
`Excel NamedRange`	Indicates the block of cells to write, using an Excel named range.	Must specify address using one of available address parameters.
`ExcelTableName`	Indicates the block of cells to write, using an Excel named range.	Must specify address using one of available address parameters.
`ExcelColumn Names`	Indicate how to determine the column names for the Excel table (in order to not overwrite with data rows), one of: • `FirstRowInRange` – column names are written to the first row in the Excel address range • `None` – column names are not written • `RowBeforeRange` – column names are written to the row before the Excel address range	`None`
`ColumnNamed Ranges`	The map of column names to named ranges, useful when the column of values is used as choices in Excel data validation.	No named ranges will be defined.
`KeepOpen`	Indicate whether to keep the Excel file open (`True`) or close after creating (`False`). Keeping the file open will increase performance because later commands will not need to reread the workbook. Make sure to close the file in the last Excel command.	`False`
`Column CellTypes`	Column names and corresponding cell types using notation: `ColumnName1:CellType1,ColumnName2:CellType2.` Column name can be `Default` to set the default for all output columns. Supported cell types are: • `Auto` – determine cell type from table column • `Text` – Excel text cell	`Auto`
`ColumnWidths`	Column names and corresponding widths using notation: `ColumnName1:Width1,ColumnName2:Width2.`	Default column with determined by Excel.

Parameter	Description	Default
	Column name can be `Default` to set the default for all output columns and `EmptyColumns` to set the width for columns with no data values. Supported width values are: • `Auto` – determine width from table contents • `N` – number of 1/256 of character widths (maximum is 256*256)	
`ColumnDecimal Places`	Column names and corresponding number of decimal places, for floating point numbers, using notation: `ColumnName1:Num1,ColumnName2:Num2`.	Determine from table column precision, or 6 if unable to determine from table.
`Condition TableID`	Identifier for condition table (see below). Can be specified using processor `${Property}`.	Style formatting is not used.
`StyleTableID`	Identifier for style table (see below). Can be specified using processor `${Property}`.	Style formatting is not used.
`Legend Worksheet`	Name of worksheet where the legend should be created. The legend displays conditions and styles.	Data table worksheet.
`LegendAddress`	Address A1, etc. for upper-left of legend.	No legend will be created.

Excel cell formatting consists of number formatting, cell colors, cell width, etc. The ***Excel Formatting*** tab allows several formatting parameters to be specified. However, a more general formatting capability based on styles is being phased in and is configured using the ***Style Formatting*** tab. Consider the following data table, where the goal is to write the TSTool table to Excel and format cells to indicate specific conditions of interest. This approach is implemented similarly in the `WriteTimeSeriesToExcel()` command.

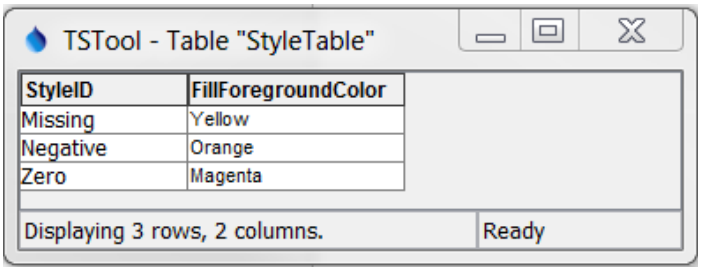


Date	ts1	ts1-flag
2001-01-01	1.00	a
2001-01-02	3.00	c
2001-01-03	2.00	b
2001-01-04	7.00	e
2001-01-05		
2001-01-06	14.00	r
2001-01-07	5.00	s
2001-01-08		
2001-01-09	2.00	z
2001-01-10	5.00	w
2001-01-11	1.00	a
2001-01-12	3.00	c
2001-01-13	2.00	b
2001-01-14	7.00	e
2001-01-15		
2001-01-16	14.00	r
2001-01-17	5.00	s
2001-01-18		
2001-01-19	2.00	z

WriteTableToExcel_DataTable

Data Table used with WriteTableToExcel() Command

To configure style-based formatting, a style table is defined listing properties for formatting a cell. This table can be defined as a CSV file, Excel worksheet or other format and read into TSTool using a suitable command. The following figure illustrates a basic style table, which can be shared among commands.



WriteTableToExcel_StyleTable

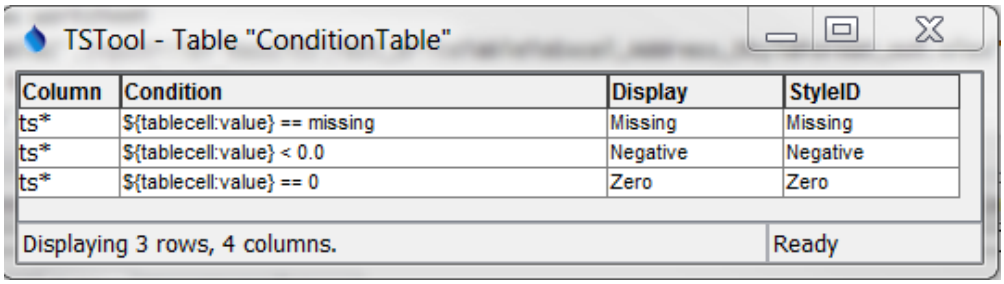
Style Table used with WriteTableToExcel() Command Style Formatting

The following style table column names are recognized. The default values for cell style properties not listed in the table are those provided by Excel.

Recognized Style Table Column Names

Column Name	Description	Default
`StyleID`	An identifier for the style, which is used in the format table below.	None – must be specified.
`FillForegroundColor`	The foreground fill color as a named color (e.g., "Red"), RGB triplet (`255,255,255`), or hex color `0xFFFFFF`. The following named colors are recognized: `black`, `blue`, `cyan`, `darkgray`, `gray`, `green`, `lightgray`, `magenta`, `none`, `orange`, `pink`, `red`, `white`, `yellow`.	No fill color.
`FillPattern`	Fill pattern for cells using `FillForegroundColor` and `FillBackgroundColor`.	**Currently always defaults to solid.**

The condition table indicates how table values should be evaluated to determine styles. The following example indicates that any columns with names starting with "ts" should be processed to evaluate for missing, negative, and zero values.



WriteTableToExcel_ConditionTable

Condition Table used with WriteTableToExcel() Command for Specific Checks and Formatting

For the above style and format tables, the column names must be specified as shown. The ***Condition*** column can contain the following specifiers:

- `${tablecell:value}` – the cell value (raw value before any formatting)
- `${tablecell:comment}` – the cell comment (currently not implemented in TSTool tables but is reserved for future implementation and Excel cell comments)

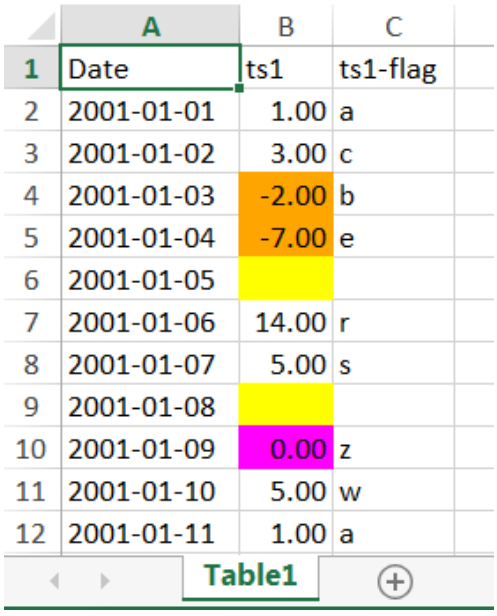
Values on the left and right of the operator must be separated with a space to facilitate parsing the condition. The ***Condition*** column can contain the following operators. Currently conditions can be specified for table column types double, float, integer, long, and string.

Condition Table Operators

Operator	Description
<	Less than.
<=	Less than or equal to.
==	Equal to. Specify the right-side value as `missing` to check for missing.
!=	Not equal to. Specify the right-side value as `missing` to check for missing.
>	Greater than.
>=	Greater than or equal to.
`contains`	Specify for string values to check for substring (case-dependent).
`startswith`	Specify for string values to check for substring at start (case-dependent).
`endswith`	Specify for string values to check for substring at end (case-dependent).

Multiple conditions can be specified by using `AND` (surrounded by a single space) between conditions. The ***Display*** column in the condition table is optional and provides test to use in the legend. If the ***Display*** column is not provided, the ***Condition*** column contents will be used for the legend.

The following figure illustrates the output from the above example.



	A	B	C
1	Date	ts1	ts1-flag
2	2001-01-01	1.00	a
3	2001-01-02	3.00	c
4	2001-01-03	-2.00	b
5	2001-01-04	-7.00	e
6	2001-01-05		
7	2001-01-06	14.00	r
8	2001-01-07	5.00	s
9	2001-01-08		
10	2001-01-09	0.00	z
11	2001-01-10	5.00	w
12	2001-01-11	1.00	a

Table1

WriteTableToExcel_Output

WriteTableToExcel() Command Example Output for Specific Checks and Formatting

The following example illustrates using multiple conditions to implement a color scale.

TSTool - Table "Style...

StyleID	FillForegroundColor
Color1	Red
Color2	Orange
Color3	Yellow
Color4	Green
Color5	Cyan
Color6	Blue

Displaying 6 rows, 2 columns.

WriteTableToExcel_StyleTable2

Style Table used with WriteTableToExcel() Command for a Color Scale

TSTool - Table "ConditionTable"

Column	Condition	Display	StyleID
ts*	${tablecell:value} < 0	Value < 0	Color1
ts*	${tablecell:value} >= 0.0 AND ${tablecell:value} < 2	0 >= Value < 2	Color2
ts*	${tablecell:value} >= 2.0 AND ${tablecell:value} < 4	2 >= Value < 4	Color3
ts*	${tablecell:value} >= 4.0 AND ${tablecell:value} < 6	4 >= Value < 6	Color4
ts*	${tablecell:value} >= 6.0 AND ${tablecell:value} < 8	6 >= Value < 8	Color5
ts*	${tablecell:value} > 8.0	Value > 8	Color6

Displaying 6 rows, 4 columns. Ready

WriteTableToExcel_ConditionTable2

Condition Table used with WriteTableToExcel() Command for a Color Scale

	A	B	C
1	Date	ts1	ts1-flag
2	2001-01-01	-1.00	a
3	2001-01-02	1.00	c
4	2001-01-03	3.00	b
5	2001-01-04	5.00	e
6	2001-01-05		
7	2001-01-06	7.00	r
8	2001-01-07	9.00	s
9	2001-01-08		
10	2001-01-09	11.00	z
11	2001-01-10	13.00	w
12	2001-01-11	-1.00	a
13	2001-01-12	1.00	c
14	2001-01-13	3.00	b
15	2001-01-14	5.00	e

WriteTableToExcel_Output2

WriteTableToExcel() Command Example Output for Color Scale