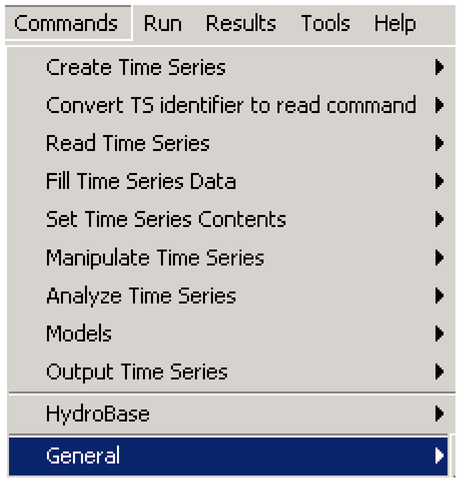
4 Commands

Version 06.10.08, 2005-09-22, Color, Acrobat Distiller

The ***Commands*** menu provides several submenus that allow commands related to time series to be inserted into the ***Commands*** list.



Menu_Commands

Commands Menu

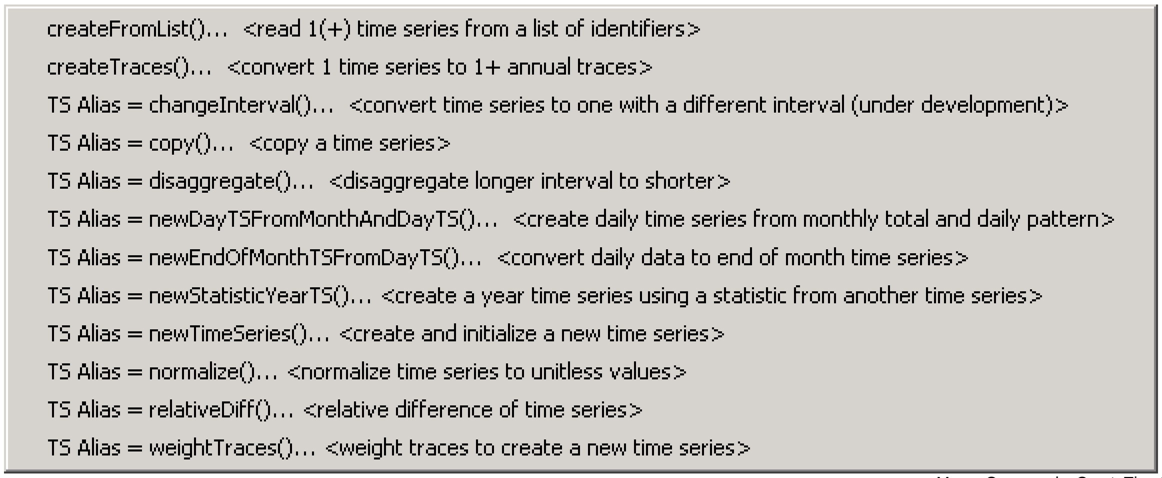
Time series commands are organized into the following categories:

1. ***Create Time Series* -** create one or more new time series in memory
2. ***Convert TS identifier to read command*** – convert a time series identifier in the ***Commands*** list area to a read command
3. ***Read Time Series*** – read time series from a file or database
4. ***Fill Time Series* -** fill missing data
5. ***Set Time Series*** – set time series data or properties
6. ***Manipulate Time Series* -** manipulate data by transforming the contents of the time series (e.g., scale a time series' data values)
7. ***Analyze Time Series*** – perform analysis on time series, without modifying the time series
8. ***Models*** – advanced or specific models that operate on time series data
9. ***Output Time Series* -** write time series results to a file or graph
10. Commands specific to various input types
11. ***General*** – general commands (e.g., to set output period)

Specific commands within each category are discussed in the following sections. The command syntax is described in detail in the **Command Reference** at the end of this documentation. Menu options are enabled/disabled depending on the state of the application.

4.1 Create Time Series

The ***Commands…Create Time Series*** menu inserts commands for creating new time series in memory.



createFromList()... <read 1(+) time series from a list of identifiers>
createTraces()... <convert 1 time series to 1+ annual traces>
TS Alias = changeInterval()... <convert time series to one with a different interval (under development)>
TS Alias = copy()... <copy a time series>
TS Alias = disaggregate()... <disaggregate longer interval to shorter>
TS Alias = newDayTSFromMonthAndDayTS()... <create daily time series from monthly total and daily pattern>
TS Alias = newEndOfMonthTSFromDayTS()... <convert daily data to end of month time series>
TS Alias = newStatisticYearTS()... <create a year time series using a statistic from another time series>
TS Alias = newTimeSeries()... <create and initialize a new time series>
TS Alias = normalize()... <normalize time series to unitless values>
TS Alias = relativeDiff()... <relative difference of time series>
TS Alias = weightTraces()... <weight traces to create a new time series>

Menu_Commands_CreateTimeSeries

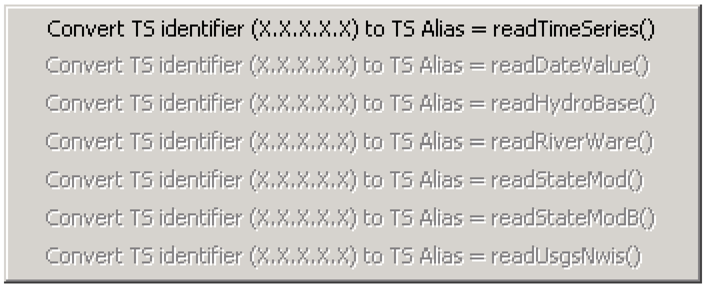
Commands...Create Time Series Menu

These commands create new time series from external data (see `createFromList()`), by using user-supplied data (see `newTimeSeries()`), or by operating on existing time series. Time series created from existing time series are fundamentally different from the original and cannot take its place with the same identifier. For example, the data interval or identifier is different in the new time series.

One important difference between the commands is whether one or multiple time series are created as the result of a command. For example, the `TS Alias =` commands create a single time series, which can be referenced using the time series alias (`Alias`). However, other commands may create more than one time series, and the identifiers for the time series may not be known until the commands are run and input data are read. Consequently, commands that operate on multiple time series are useful for bulk processing of data but may be more difficult to use for detailed analysis and manipulation.

4.2 Converting Time Series Identifier to Read Command

The ***Commands…Convert Time Series Identifier to Read Command*** menu converts a selected time series identifier in the ***Commands*** list to a read command.


Convert TS identifier (X.X.X.X.X) to TS Alias = readTimeSeries()
Convert TS identifier (X.X.X.X.X) to TS Alias = readDateValue()
Convert TS identifier (X.X.X.X.X) to TS Alias = readHydroBase()
Convert TS identifier (X.X.X.X.X) to TS Alias = readRiverWare()
Convert TS identifier (X.X.X.X.X) to TS Alias = readStateMod()
Convert TS identifier (X.X.X.X.X) to TS Alias = readStateModB()
Convert TS identifier (X.X.X.X.X) to TS Alias = readUsgsNwis()

Menu_Commands_ConvertTSID

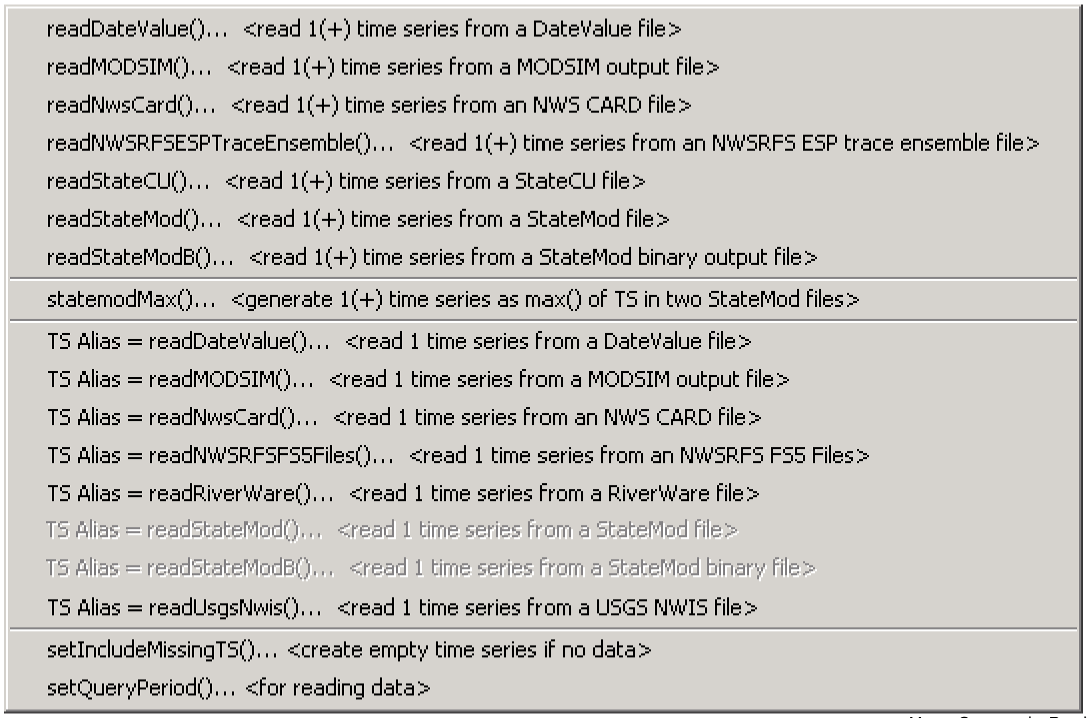
Commands...Convert Time Series Identifier to Read Command Menu

Currently, the only enabled feature is to convert a time series identifier to a `TS Alias = readTimeSeries()` command. This is used to assign an alias to the time series, simplifying its use in other commands. The input type from the time series identifier is interpreted by TSTool to determine how to read the time series. Unlike other commands menus, this menu does not insert a new command. It converts a single selected time series identifier.

In the future, it is envisioned that this menu will be enhanced to enable conversion of a time series identifier to a specific read command – this will allow the features of each read command to be used.

4.3 Read Time Series

The ***Commands…Read Time Series*** menu inserts commands to read time series from databases or files.



readDateValue()... <read 1(+) time series from a DateValue file>
readMODSIM()... <read 1(+) time series from a MODSIM output file>
readNwsCard()... <read 1(+) time series from an NWS CARD file>
readNWSRFSESPTraceEnsemble()... <read 1(+) time series from an NWSRFS ESP trace ensemble file>
readStateCU()... <read 1(+) time series from a StateCU file>
readStateMod()... <read 1(+) time series from a StateMod file>
readStateModB()... <read 1(+) time series from a StateMod binary output file>
statemodMax()... <generate 1(+) time series as max() of TS in two StateMod files>
TS Alias = readDateValue()... <read 1 time series from a DateValue file>
TS Alias = readMODSIM()... <read 1 time series from a MODSIM output file>
TS Alias = readNwsCard()... <read 1 time series from an NWS CARD file>
TS Alias = readNWSRFSFS5Files()... <read 1 time series from an NWSRFS FS5 Files>
TS Alias = readRiverWare()... <read 1 time series from a RiverWare file>
TS Alias = readStateMod()... <read 1 time series from a StateMod file>
TS Alias = readStateModB()... <read 1 time series from a StateMod binary file>
TS Alias = readUsgsNwis()... <read 1 time series from a USGS NWIS file>
setIncludeMissingTS()... <create empty time series if no data>
setQueryPeriod()... <for reading data>

Menu_Commands_ReadTimeSeries

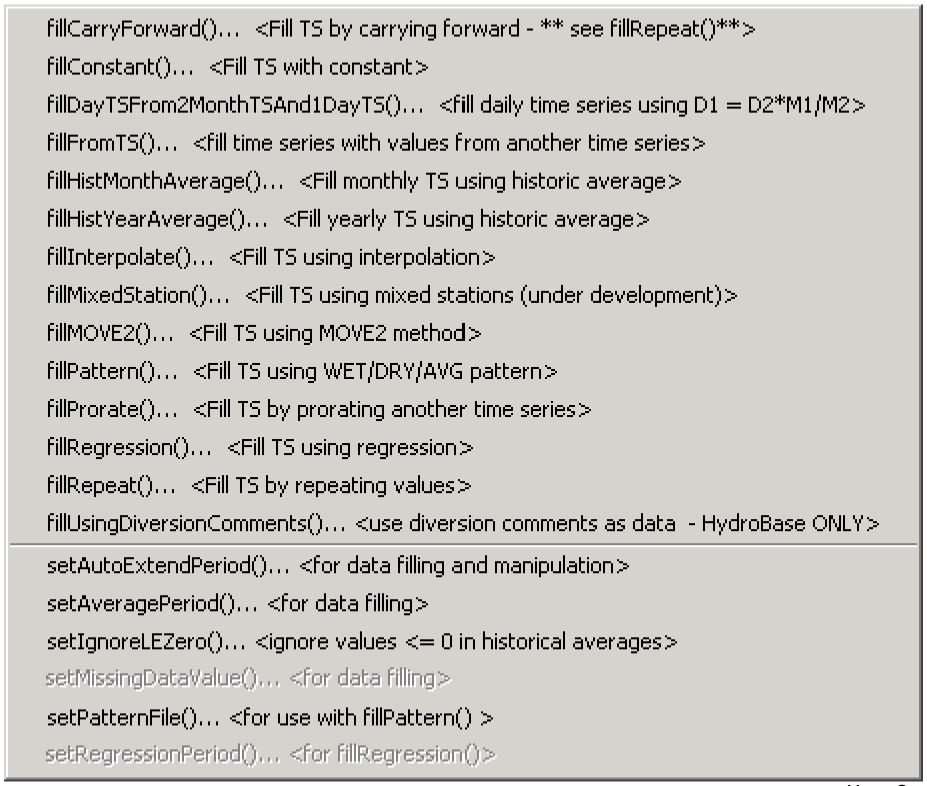
Commands...Read Time Series Menu

Read commands are grouped into commands that process one or more time series in a file (top) and commands that read a specific time series (`TS Alias =` commands). The `TS Alias =` commands assign an alias to the time series, which can then be referenced by other commands.

Commands that read multiple time series do not usually have access to the time series while commands are edited; consequently, they are useful for bulk processing of data and may be more difficult to use for detailed analysis and manipulation. It may be useful to read many time series and then use the `selectTimeSeries()` and `deselectTimeSeries()` commands to select a subset of the results (see output commands).

4.4 Fill Time Series Data

The ***Commands…Fill Time Series Data*** menu inserts commands for filling missing data in time series.



fillCarryForward()... <Fill TS by carrying forward - ** see fillRepeat()**>
fillConstant()... <Fill TS with constant>
fillDayTSFrom2MonthTSAnd1DayTS()... <fill daily time series using D1 = D2*M1/M2>
fillFromTS()... <fill time series with values from another time series>
fillHistMonthAverage()... <Fill monthly TS using historic average>
fillHistYearAverage()... <Fill yearly TS using historic average>
fillInterpolate()... <Fill TS using interpolation>
fillMixedStation()... <Fill TS using mixed stations (under development)>
fillMOVE2()... <Fill TS using MOVE2 method>
fillPattern()... <Fill TS using WET/DRY/AVG pattern>
fillProrate()... <Fill TS by prorating another time series>
fillRegression()... <Fill TS using regression>
fillRepeat()... <Fill TS by repeating values>
fillUsingDiversionComments()... <use diversion comments as data - HydroBase ONLY>
setAutoExtendPeriod()... <for data filling and manipulation>
setAveragePeriod()... <for data filling>
setIgnoreLEZero()... <ignore values <= 0 in historical averages>
setMissingDataValue()... <for data filling>
setPatternFile()... <for use with fillPattern() >
setRegressionPeriod()... <for fillRegression()>

Menu_Commands_FillTimeSeries

Commands...Fill Time Series Data Menu

Fill commands will only change values that are missing, whereas set commands will change values regardless of whether they are missing or non-missing. These commands can be executed in sequence to apply multiple fill techniques to time series. Many commands accept the `*` wildcard for the time series identifier, allowing all time series to be filled.

Time series may contain missing data due to the following reasons:

1. The data collection system is unavailable because of a failure, maintenance cycle, or hardware that is turned off because of seasonal use
2. In a real-time system the most current data have not yet been received
3. Data collection hardware was not in place during a period (e.g., an early period)
4. Measured values are suspected of being in error and are changed to missing
5. Values in a computed time series cannot be computed (e.g., input data are missing)
6. A data source stores only observed values and non-recorded values are assumed to be missing rather than a specific value (e.g., zero)

Observations that are available are typically either measured values or values that have been estimated by the organization that collects and/or maintains the data. Data flags indicating missing data may or may not be available in the original source data (e.g., an ‘`m`’ or ‘`e`’ character flag is often used to indicate missing and estimated data).

TSTool handles missing data by internally assigning a special numeric value where data are missing. Different input types may have different missing data values but typically `-999` or a similar extreme value is used. If the output period is specified using `setOutputPeriod()`, then extensions to the available time series period are filled with the missing data value. Data flags are supported for some input types. TSTool displays and output products indicate missing data by blanks, showing the missing data value, or a string (e.g., `NC`).

Filled time series are often required for use in computer models. TSTool provides a number of features to fill time series data. The data filling process consists of analyzing available data and using the results to estimate missing data values. The estimation process can be simple or complex, resulting in varying degrees of error and statistical characteristics of the final time series. The data analysis uses data available at the time that the fill command is encountered. Consequently if values have been changed since the initial read (e.g., because of layered fill commands), the changed values may impact the analysis. Basic statistical properties of the original data are saved after the initial read to allow use in later fill commands. For example, for monthly time series, the historical monthly averages are computed after the initial read to allow use with a `fillHistMonthAverage()` command.

The overall period that is being filled is controlled by the time series period or analysis period that is specified with fill commands. TSTool will not automatically extend the period of a filled time series after the time series is initially read. Use the `setQueryPeriod()` and `setOutputPeriod()` commands to control the time series period.

The following table lists the fill techniques that are supported by TSTool.

TSTool Fill Techniques and Associated Commands

Technique	Command	Typical Use
Carry Forward	`fillCarryForward()`	See `fillRepeat()`. This command is being phased out.
Constant	`fillConstant()`	Use when missing data can be estimated as a constant. For example, if only the early period of a "regulated" (e.g., reservoir) time series is missing, it may be appropriate to set the values to zero.
Monthly total, daily pattern	`fillDayTSFrom 2MonthTSAnd1DayTS()`	Use to estimate a daily time series by applying the pattern of a related daily time series to monthly totals from the related and current time series. For example, use to estimate daily streamflow from monthly total values.
Fill from time series	`fillFromTS()`	Use non-missing values from a time series to fill missing values in another time series.
Historic Monthly Average	`fillHistMonthAverage()`	Use with monthly time series to estimate missing monthly values as the average of historic monthly values. For example, if applied to monthly precipitation data, a missing July value would be set to the average of observed July precipitation values (zero is an observation).

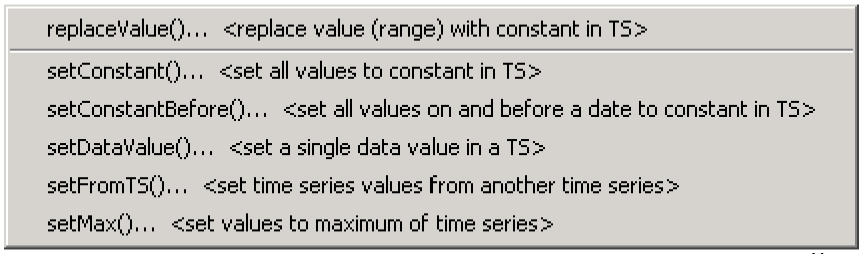
TSTool Fill Techniques and Associated Commands (continued)

Technique	Command	Typical Use
Historic Year Average	`fillHistYearAverage()`	Use with yearly time series to estimate missing data as the average of annual values.
Interpolation	`fillInterpolate()`	Use to estimate missing data by interpolating between non-missing values. For example, use to estimate reservoir level changes.
Mixed Station	`fillMixedStation()`	This command tries various combinations of `fillRegression()` and `fillMove2()` parameters with time series at different locations, to use the best combination.
Maintenance of Variance	`fillMOVE1()`	Use to estimate missing data using the Maintenance of Variance Extension (MOVE.1). For example, use to estimate unregulated streamflow from a related gage. **This command is currently not enabled.**
Maintenance of Variance	`fillMOVE2()`	Use to estimate missing data using the Maintenance of Variance Extension (MOVE.2). For example, use to estimate unregulated streamflow from a related gage. This approach has been shown to be slightly better than the MOVE.1 approach.
Historic Pattern Averages	`fillPattern()`	Similar to filling with historic averages with additional complexity of classifying historic months into categories. For example, historic averages for wet, dry, and average periods are computed and used as the historic averages. This command requires that the `setPatternFile()` control command also be used.
Prorate	`fillProrate()`	Fill a time series by prorating known values with another time series.
Regression	`fillRegression()`	Use to estimate missing data by using ordinary least squares regression. For example, use to estimate streamflow from a related gage.
Repeat	`fillRepeat()`	Use when it can be assumed that the last observed value before a missing period is a good estimate for missing data. For example, use with "forecasted" data where no future value is available for interpolation.
Using diversion comments	`fillUsingDiversionComments()`	This command is only available with the HydroBase input type and uses diversion comments to set additional diversion amounts to zero.

Fill commands can be layered (e.g., use `fillRegression()`, then `fillInterpolate()`, then `fillConstant()`. However, the analysis that occurs for each command may be impacted by earlier fill commands. If necessary, use the `setFromTS()` command to piece together the results of independent fill commands into a final time series. The ***Results...Graph - XY-Scatter*** output provides options for selecting different fill techniques and viewing analysis details.

4.5 Set Time Series Data

The ***Commands…Set Time Series Contents*** menu inserts commands that set time series. Unlike fill commands, set commands reset values regardless of whether the values were missing in the time series.


replaceValue()... <replace value (range) with constant in TS>
setConstant()... <set all values to constant in TS>
setConstantBefore()... <set all values on and before a date to constant in TS>
setDataValue()... <set a single data value in a TS>
setFromTS()... <set time series values from another time series>
setMax()... <set values to maximum of time series>

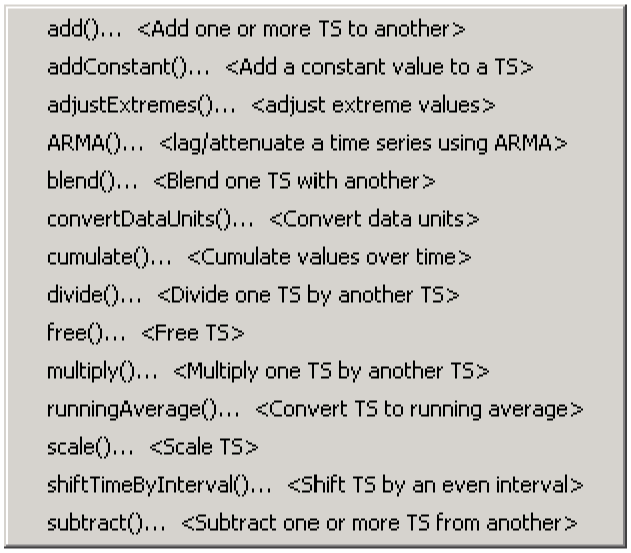
Menu_Commands_SetTimeSeries

Commands...Set Time Series Contents Menu

Additional commands may be added in the future to set time series properties, including description/name, comments, etc.

4.6 Manipulate Time Series

The ***Commands…Manipulate Time Series*** menus insert commands for manipulating time series.


add()... <Add one or more TS to another>
addConstant()... <Add a constant value to a TS>
adjustExtremes()... <adjust extreme values>
ARMA()... <lag/attenuate a time series using ARMA>
blend()... <Blend one TS with another>
convertDataUnits()... <Convert data units>
cumulate()... <Cumulate values over time>
divide()... <Divide one TS by another TS>
free()... <Free TS>
multiply()... <Multiply one TS by another TS>
runningAverage()... <Convert TS to running average>
scale()... <Scale TS>
shiftTimeByInterval()... <Shift TS by an even interval>
subtract()... <Subtract one or more TS from another>

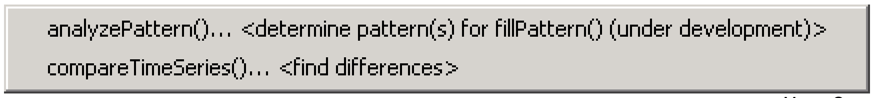
Menu_Commands_Manipulate

Commands...Manipulate Time Series Menu

Because the fundamental nature of the time series (e.g., data type, interval) is not changed, these commands do not result in the creation of a new time series. Manipulation commands typically add comments to the time series history, which is displayed in summary output.

4.7 Analyze Time Series

The ***Commands…Analyze Time Series*** menu inserts commands for analyzing time series, which typically produce a report or other output product:

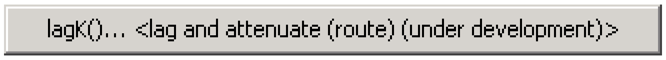

analyzePattern()… <determine pattern(s) for fillPattern() (under development)>
compareTimeSeries()… <find differences>

Menu_Commands_AnalyzeTimeSeries

Commands…Analyze Time Series Menu

4.8 Models

The ***Commands…Models*** menu inserts commands that perform tasks that are more complex than simple data processes.



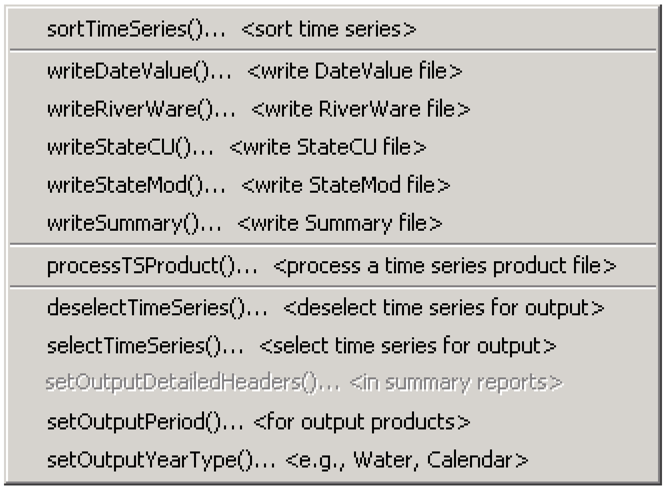
Menu_Commands_Models

Commands…Models Menu

Minimal model features are currently available. However, it is envisioned that additional capabilities will be added in the future.

4.9 Output Time Series

The ***Commands…Output Time Series*** menu inserts commands for outputting time series.


sortTimeSeries()... <sort time series>
writeDateValue()... <write DateValue file>
writeRiverWare()... <write RiverWare file>
writeStateCU()... <write StateCU file>
writeStateMod()... <write StateMod file>
writeSummary()... <write Summary file>
processTSProduct()... <process a time series product file>
deselectTimeSeries()... <deselect time series for output>
selectTimeSeries()... <select time series for output>
setOutputDetailedHeaders()... <in summary reports>
setOutputPeriod()... <for output products>
setOutputYearType()... <e.g., Water, Calendar>

Menu_Commands_OutputTimeSeries

Commands...Output Time Series Menu

Commands that operate on time series are listed first and commands that set global configuration values (e.g., output period) are listed at the end of the menu.

Using the output commands allows the results of processing to be saved but increases processing time. If commands are processed repeatedly during analysis or debugging, the following steps may be taken to increase overall efficiency:

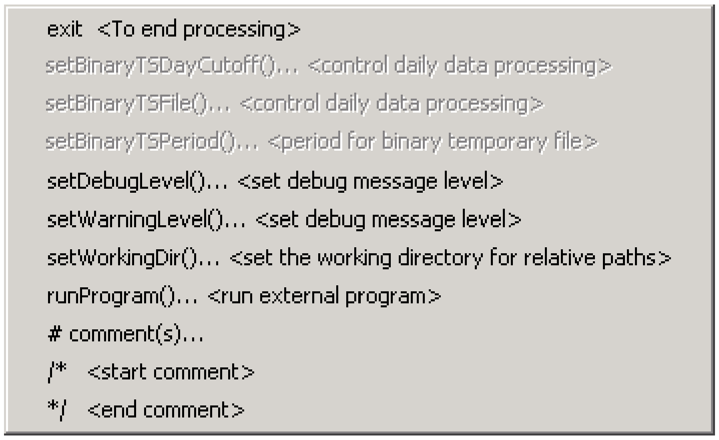
1. Output commands that produce output files are not executed if the ***Commands*** list is processed with ***Run… All Commands (ignore output commands)*** or ***Run…Selected Commands (ignore output commands)***. Therefore, use this menu choice to ignore the output commands.
2. Only selected commands are processed. Therefore select all but the output commands.
3. Use an `exit` control command before output commands to skip the output commands. This command can then be deleted or commented out when not needed.
4. Commands can be converted to comments using the ***Commands*** menu or the popup menu that is displayed when right-clicking on the ***Commands*** list. Therefore, output commands can be temporarily converted to comments until output needs to be created.

4.10 Commands for Specific Input Types

Depending on the input types that are enabled, additional command menus may be enabled. For example, if the HydroBase input type is enabled, a HydroBase menu will be enabled, and will list commands specific to HydroBase. In particular, commands like `openHydroBase()` are provided to make database connections while processing commands.

4.11 General Commands – Global Settings

The ***Commands…General*** menu provides choices to insert commands to set parameters that affect the behavior of other commands.


exit <To end processing>
setBinaryTSDayCutoff()… <control daily data processing>
setBinaryTSFile()… <control daily data processing>
setBinaryTSPeriod()… <period for binary temporary file>
setDebugLevel()… <set debug message level>
setWarningLevel()… <set debug message level>
setWorkingDir()… <set the working directory for relative paths>
runProgram()… <run external program>
# comment(s)…
/* <start comment>
*/ <end comment>

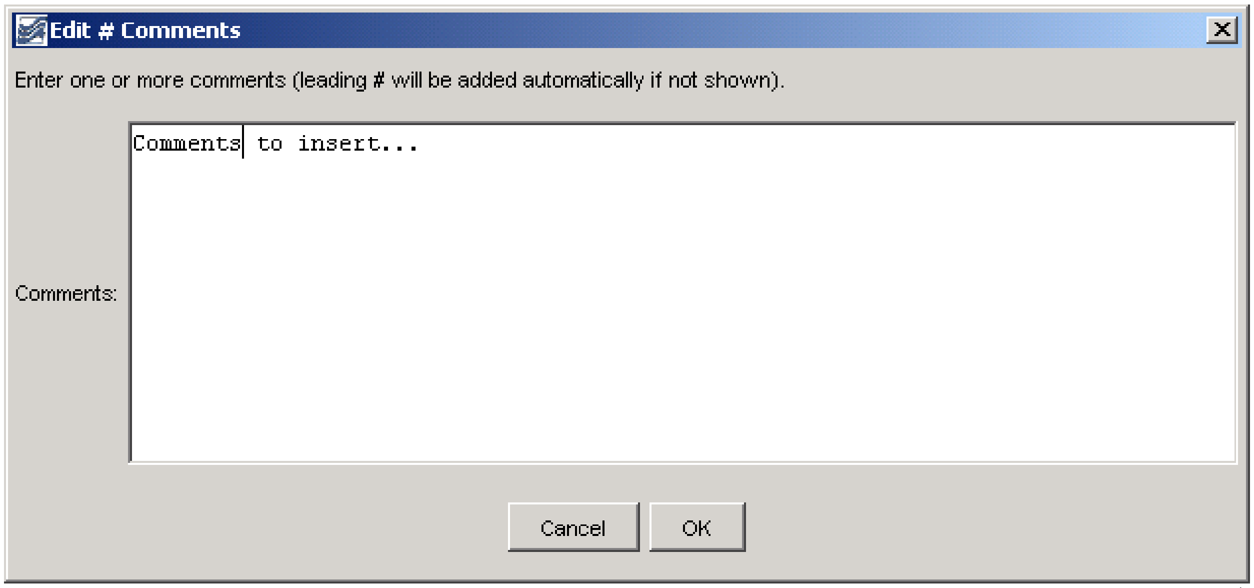
Menu_Commands_General

Commands…General Menu

These commands can be repeated and the effects of a command are in effect after it is encountered until another command changes the setting.

4.11.1 Inserting # Comments

Comments are inserted by selecting a line in the ***Commands*** list and then selecting ***Commands…General…# Comment(s)***. The comments will be inserted before the selected commands. Unlike most other command editors, multiple command lines can be selected. The interface will automatically insert the # character. The following dialog is used to edit comments.


Edit # Comments

Enter one or more comments (leading # will be added automatically if not shown).

Comments:

```
Comments to insert...
```

comments

Comment Editor

4.11.2 Start /* */ Comments

Multiple commands can be commented using the following notation:

```
/*
commented lines
commented lines
*/
```

This syntax is consistent with a number of programming languages, including C, C++, and Java, and can be used to quickly disable multiple commands. Use the ***Commands…General…/* <start comment>*** menu to start a comment above the selected command. Matching start and end comments should be inserted. See also the `exit` control command.

4.11.3 End /* */ Comments

Use the ***Commands…General…*/ <end comment>*** menu to end a multi-line comment in commands.