Scripting Guide
Tecplot 360 2025 Release 1

Tecplot, Inc.

Introduction

Tecplot 360 is a powerful tool for visualizing a wide range of technical data. It offers line plotting, 2D and 3D surface plots in a variety of formats, and 3D volumetric visualization. The user documentation for Tecplot 360 is divided into the following books:

Scripting Guide (this document)

Provides Tecplot macro command syntax and information on working with macro files and commands.

User's Manual

Provides a complete description of working with Tecplot 360 features.

Getting Started Manual

New users are encouraged to work through the tutorial provided in the Getting Started Manual to learn how to work with key product features.

Quick Reference Guide

Provides syntax for zone header files, macro variables, keyboard shortcuts, and more.

Data Format Guide

Provides information on outputting your simulator data into Tecplot 360 file format.

Installation Guide

Provides detailed instructions on how to install Tecplot 360 on your machine.

Release Notes

Provides information about new and/or updated Tecplot 360 features.

Macro programming capabilities are included in your Tecplot 360 distribution. Macro language syntax and usage are provided in this Scripting Guide. Macros can be accessed via the **Scripting** menu and from the **Quick Macro Panel** (also accessed via the **Scripting** menu).

Introduction to Macro Commands

A Tecplot 360 macro is a set of instructions, called macro commands, which perform actions in Tecplot 360. Macro commands can be used to accomplish virtually any task that can be done via the Tecplot 360 interface, offering an easy way to automate Tecplot 360 processes. The only things you can do interactively that cannot be done with macro commands are those actions that have no effect on a final, printed plot (such as resizing the Tecplot 360 process window). To augment this ability, there are

macro commands which have no corresponding interactive control, such as looping and conditional commands. These commands typically go hand in hand with the execution of a macro.

You can create macros by recording them from the Tecplot 360 interface using the **Macro Recorder** (accessed via the **Scripting**>**Record Macro** menu), or create them from scratch using any ASCII text editor. In most cases, the most effective approach to creating a macro is the following hybrid approach:

1. Run Tecplot 360 and choose to record a macro to a file. Perform tasks similar to those you are trying to capture in the final macro.
2. Close the recording session and examine the macro file. The commands generated by Tecplot 360 should be fairly readable and easy to understand.
3. Make minor modifications to the recorded macro. Typical modifications involve adding loops, adding variables, or adding commands that, for example, prompt the user to enter a file name.

One of the main reasons for using the approach above is the large number of commands and permutations of parameters. This manual provides an exhaustive listing of the available macro commands. However, it is often easier to have Tecplot 360 perform the action and record the relevant command than look up individual commands and their required parameters.

Managing Macros

Tecplot 360 macros are stored in files. These files are processed by loading them into Tecplot 360 and running them.

Macros vs. Macro Functions vs. Macro Commands

A Tecplot 360 macro is a file containing one or more macro commands. These files start with the following special comment line to notify Tecplot 360 that what follows is a Tecplot 360 macro:

```
#!MC  1410
```

Any number of macro commands or comments may follow.

Tecplot 360 macro functions are defined in Tecplot 360 macros by using the `$!MACROFUNCTION` and `$!ENDMACROFUNCTION` commands. Between the `$!MACROFUNCTION` and `$!ENDMACROFUNCTION` commands you may use any valid macro command (except `$!MACROFUNCTION`). When a Tecplot 360 macro is loaded, all macro functions are extracted and the attached commands are not executed until a `$!RUNMACROFUNCTION` command is encountered.

Macro functions may be "retained" if desired. A retained macro function remains defined in Tecplot 360 even if the macro in which it was defined is replaced by another macro. Retained macro functions may be called by other macros that are loaded at a later time.

Running Macros from the Command Line

See the Command Line chapter of the User's Manual for information on running macros on the command line. For issues relating to troubleshooting batch mode, see the Troubleshooting Appendix of the User's Manual.

Running Macros from the Tecplot 360 Interface

You can run a macro file by going to **Scripting>Play Macro/Script**. A dialog appears; choose the macro to play.

If you want to debug a macro file, go to the **Scripting** menu and select the "View/Debug Macro" option. The **Macro Debugger** dialog appears so you can load in a macro. When the macro is loaded, Tecplot 360 waits at the first macro command for you to step through the commands. See the User's Manual for complete details on how to use the Macro Debugger.

Running Macros from the Quick Macro Panel

Macros that you use frequently or want rapid access to may be defined as macro functions within a special file called `tecplot.mcr` in the current directory, your user home directory, or the Tecplot 360 home directory. When Tecplot 360 starts, it looks for this file in each of those directories in turn. If Tecplot 360 finds the file, it loads the macro definitions and creates an entry on the Quick Macro Panel (**Scripting>Quick Macros**) for each function in the file.

You can have Tecplot 360 load your own macro function file by using the `-qm` flag on the command line. The following command runs Tecplot 360 and installs the macro functions in the file `myteccmd.mcr` into the Quick Macro sidebar:

```
tec360 -qm myteccmd.mcr
```

By default, all macro functions defined in the `tecplot.mcr` file are listed in the Quick Macro Panel. See the `$!MACROFUNCTION···$!ENDMACROFUNCTION` command for more information on defining functions.

If `tecplot.mcr` does not contain any function definitions, nothing will appear the **Quick Macro Panel**.

If you want Tecplot 360 to display the Quick Macro Panel at startup, include the `-showpanel` flag on the command line.

To see an example of a macro function file, look at the file `tecplot.mcr` located in the Tecplot 360 home directory. This is where the default Quick Macro Panel entries are stored, for example 3D Rotation Animation and Reset Center of Rotation.

Writing Forward Compatible Macros

In order to ensure forward compatibility of your macro commands, please keep the following guidelines in mind. These guidelines will allow you to create macros that will work for years, on many machines and platforms.

1. Begin your macro by opening a layout.

   This will ensure that the final plot is consistent between versions of Tecplot 360 (even if the default style settings for Tecplot 360 have changed).

   

   An alternative to using a layout is to load data and then load a frame style file into each frame.

   If your macro will be used for more than one layout, you can ensure forward compatibility by:

   - Using the `$!PromptForFileName` command. This will allow the user to interactively specify the layout file.

     ```
     -or-
     ```

   - Launching Tecplot 360 from the command line, specifying the layout and the macro:

     ```
     tecplot mylayout.lay mydatafile mymacro.mcr
     ```

2. Store associated files and graphics in the same folder as the macro file.

   If your macro loads files or inserts images without allowing the user to choose them, it is a good practice to store them in the same folder as the macro file that uses them. After recording, edit the macro, and replace the path to the file with the intrinsic macro variable `|macrofilepath|`. For example:

   ```
   $!OpenLayout "|macrofilepath|\Density.lpk"
   ```

   This allows the macro to work without editing in any location as long as the entire folder of files was copied there.

3. Avoid using a `$!Pick` command in your macro.

   Changes to the aspect ratio can cause a recorded `$!Pick` command to fail when the macro is run on another machine or in another version of Tecplot 360.

   - In a plot with multiple frames, you cannot use `$!Pick` to change the active frame. Instead, give

each frame a meaningful name such as "Full View" and "Zoom Frame" in the layout. Then use the command:

```
$!FrameControl ActivateByName Name = "Full View"
```

to access the frame you want. This will also simplify later changes to the macro.

- If you must pick an item, make the pick as precise as possible. For example, clicking on the center, not the edge, of a zone or slice will increase the chances that the pick will be successful when the macro is replayed.

  When selecting text or geometries while recording a macro, click and drag in the widest possible area around the objects to select. The command will be recorded as

  ```
  $!PICK ADDALLINRECT
    SELECTTEXT = YES
    X1 = 1.56075949367
    X2 = 3.97088607595
    Y1 = 2.29556962025
    Y2 = 3.91582278481
  ```

  The x and y ranges can be expanded if needed.

4. Use plenty of comments in your macro so that when you need to modify it, you understand what it does.

Debugging Macros

In general, the best way to debug a macro is to use the **Macro Debugger**, and find which command is causing the problem. Here are some tips for specific problems:

Problem:

The macro was created with an earlier version of Tecplot 360, or Tecplot Focus to make the plot needed. With a newer version, the macro runs without error, but the plot looks different.

Solution:

Run the macro with the old version of the product, then save a frame style to a file. Begin your macro by loading the data, then pasting the frame style from a file. This will ensure that the final plot will be consistent from one version to the next, even if the default style settings have changed.

Problem:

The macro gives you errors such as "File does not exist" or "Cannot open file", but you can locate the

file.

Solution:

Copy the file to the same folder as the macro file that uses the file. Edit the macro, and replace the path to the file with the intrinsic macro variable `|macrofilepath|`. For Example :

```
$!Openlayout "|macrofilepath|\Density.lpk"
```

This allows the macro to work without editing in any location as long as the entire folder of files was copied there.

Problem:

Running the macro causes unusual error messages, such as: "No objects to cut or the objects selected not allowed to be cut" or "Not allowed to adjust zones or mappings when the mouse mode is set to SELECTOR". When you run the macro in the Macro Debugger, you see that the problem occurs with when a `$!Pick` command is run.

Solution:

Avoid using the `$!Pick` command in your macro. Changes to the aspect ratio can cause a recorded `$!Pick` command to fail when the macro is run on another machine or in another version of Tecplot 360.

To fix the problem in an existing macro, follow these steps to make the coordinates more precise:

1. Run the macro on the machine where the error message is generated.
2. Via the Macro Debugger or editor, identify the preceding `$!PICK ADDATPOSITION` or similar select type pick command. Note the X,Y coordinates of the command. A good way to do this is:
    1. Run the macro until you get the "No Objects to Shift" error message.
    2. Click Ok on the dialog.
    3. Bring up the Macro Debugger: **Scripting>View/Debug Macro**.
    4. Find the nearest `$!PICK ADDATPOSITION` command above the current command and put a break point on that command.
    5. Press "Reset" to reset the macro and then run the macro.

        

        If the problem only occurs when running in batch mode, try to determine the macro command by examining the `batch.log` file.

    6. Insert a `$!Pause` command in your macro just before the `$!Pick Add` command that precedes the offending command. Now run Tecplot 360 interactively from the Macro Debugger. You can then see the line number where you need to put the break.

3. Back in Tecplot 360, select the zoom tool.
4. Hold the shift key down and notice that the running coordinates in the lower right corner now show "PX = *xxxxx* PY = *yyyyyy*". *xxxxxx* and *yyyyyy* are the paper coordinates of the hot spot of the zoom tool. (If you see X and Y for grid coordinates, or FX and FY for frame coordinates, you need to hold down the Shift key. Pick commands always use paper coordinates.)
5. Move the zoom tool until *xxxxx* and *yyyyy* are close to the coordinates noted in step 2.
6. Note where the pick occurred. It is likely the pick occurred some distance away from the actual edge of the object to pick. Move the zoom tool to a "better" location for the pick and note the coordinates.

Edit the macro file and replace the old X,Y pick coordinates with those determined in step 6.

Macro Command Syntax

A macro file consists of one or more macro commands. Comments may be inserted anywhere in the file, except within a character string. Comments start with an "#" (octothorp) and extend to the end of the line. The first line of a macro file contains a special comment that identifies the version number of the macro file. For Tecplot 360, this line is: `#!MC 1410`.

A Tecplot 360 macro file has the form:

```
#!MC  1410
<macrocommand>
<macrocommand>
. . .
```

Each *macrocommand*, in turn, has the form:

```
$!commandname[commandspecificmodifiers]
  [mandatoryparameters]
  [optionalparameters]
```

where:

commandspecificmodifiers	These are optional command-specific modifiers. An example of a command that uses this is the `$!FIELDMAP` command. The `$!FIELDMAP` command can be followed by a "set." If it is not followed by a set, the $!FIELDMAP command applies to all enabled zones. A supplied set in this case is used to limit the zones to which the `$!FIELDMAP` command applies.
mandatoryparameters	commandparameter commandparameter...

optionalparameters	commandparameter commandparameter…
commandparameter	parameterassignment or parametersubcommand.
parameterassignment	parametername op value.
op	= or -= or += or *= or /=.
parametersubcommand	parametername {optionalparameters}.
commandname	The name of a major command, such as `REDRAW`.
parametername	The name of a valid parameter for the previously named major command. For example, the `$!REDRAW` major command has an optional parameter called `DOFULLDRAWING`.
value	*number*, *expression*, or *enumeratedvalue*.
number	Any valid integer or double value representation.
expression	Any valid infix notation expression. The entire expression must itself be enclosed in parenthesis. For example `(3+5)`.
enumeratedvalue	A key word that is unique to the variable being assigned a value. For example, if the variable being assigned a value is a basic color then the enumerated value can be one of the following: `BLACK, RED, GREEN, BLUE, CYAN, YELLOW, PURPLE, WHITE, CUSTOM1` through `CUSTOM56`.

Spacing and capitalization for macro commands are, for the most part, not important. The following examples show different ways to enter the same macro command to set the width and height for the custom1 paper:

Example 1:

```
$!PAGE
PAPERSIZEINFO
{
  CUSTOM1
  {
  WIDTH = 3
  }
}
```

Example 2:

```
$!PAGE PAPERSIZEINFO
  {CUSTOM1
```

```
   {WIDTH = 3}
  }
```

Example 3:

```
$!PAGE papersizeinfo {custom1 {width = 3}}
```

Macro Variables

Macro variables are identified by a sequence of characters surrounded by vertical bars ("|"). Some examples are:

```
|myvariable|
|loop|
|1|
|$HOME|
```

Macro variables can be placed anywhere within a macro command. Upper case and lower case characters are treated the same. For example `|ABC|` and `|aBc|` represent the same variable. `|loop|`, `|1|`, and `|$HOME|` are special uses of macro variables with `|loop|` being the value of the inner most loop counter, `|1|` being the value of the first parameter passed to a macro function and `|$HOME|` evaluating to the `HOME` environment variable. See subsections in this chapter for more information.

Macro variables will be expanded to their value at the time the macro statement is processed. The value of the variable is substituted into the command in place of the variable identifier. The result must be a syntactically valid Tecplot macro expression or command.

Examples:

The following macro commands will result in a rotation of the data about the X-axis by 10 degrees:

```
$!VARSET |a1| = 10
$!ROTATEDATA
   ANGLE = |a1|
   XVar = 1
   YVar = 2
   ZVar = 3
   NormalX = 1
   NormalY = 0
   NormalZ = 0
```

In the following macro command, the variable `|message|` contains a string (text). To produce a valid

Tecplot macro command when the variable's value is substituted in, the variable reference must be surrounded by quote marks.

```
$!VARSET |message| = "Hello, world!"
$!ATTACHTEXT
TEXT = "|message|"
```

The following, then, is an error:

```
$!ATTACHTEXT
TEXT = |message|
```

When the value of `|message|` is substituted into the command, it reads `TEXT = Hello World`, which is an error because it lacks the quote marks required for a string parameter.

Intrinsic Variables

The following table lists variables maintained by Tecplot 360 that may be referenced by macro commands. Since these variables are maintained by Tecplot 360, you cannot assign values to them.

For intrinsic variables that represent an attribute of a Tecplot 360 object of which Tecplot 360 supports multiple instances (such as a frame, a dataset, or a zone), the variable by default refers to the current or active instance: for example, to the active frame if the variable refers to an attribute of frames. If there is no obvious default instance, the variable name by itself usually refers to the first instance.

For such variables, you may specify the desired instance of the referenced type of object by enclosing an index in square brackets immediately following the variable name. For example, `|AUXZONE[3]:BC|` refers to the zone auxiliary data named `BC` in the third zone.

In most cases, the index may also be written as `ACTIVEOFFSET` = *n* to specify the *nth* active instance of an object type. For example, `|ENDSLICEPOS[ACTIVEOFFSET=2]|` refers to the end position of the second active slice group.

Variables	Notes
|AUXDATASET:*Auxname*|	Retrieves auxiliary data named *Auxname* from a dataset. For example, `|AUXDATASET:Reynolds|` retrieves auxiliary data "Reynolds".
|AUXFRAME:*Auxname*|	Retrieves auxiliary data named *Auxname* from a frame. For example, `|AUXFRAME:MyFrame|` retrieves auxiliary data "MyFrame" from the active frame.
|AUXZONE:*Auxname*|	Retrieves auxiliary data named *Auxname* from a specific zone. For example, `|AUXZONE[3]:BC|` retrieves auxiliary data `BC` from zone 3.
|AXISMAX*q*|	Maximum value of the *q*-axis range, where *n* is one of: A[1] R, X, Y, or Z.

Variables	Notes
|AXISMIN*q*|	Minimum value of the *q*-axis range, where *n* is one of A[1], R, X, Y, or Z.
|BYTEORDERING|	Returns the byte ordering (INTEL or MOTOROLA).
|DATASETFNAME*[nnn]*|	Contains full path to loaded data file. If multiple data files have been loaded, use e.g. `|DATASETFNAME[2]|` to specify the desired path (indexing by load order).
|DATASETTITLE|	Returns the title of the dataset, or "No Data Set" if a dataset does not exist.
|DATE|	Returns the date in the form of `dd Mmm yyyy`.
|ENDSLICEPOS|	Returns the position of the last slice in a group. (Not available for arbitrary slices.)
|EXPORTISRECORDING|	Returns YES/NO to help macros complete record commands in the proper order.
|FRAMENAME|	Returns the name of the active frame.
|PAGENAME|	Returns the name of the active page.
|INBATCHMODE|	Returns 1 if in batch mode, 0 if in interactive mode.
|ISDATASETAVAILABLE|	Returns 1 if a dataset exists, and 0 otherwise.
|ISOSURFACELEVEL|	Returns the current iso-surface's iso-value.
|LAYOUTFNAME|	Returns the current layout file name.
|LOOP|	Innermost loop counter.
|MACROFILEPATH|	Returns the path to the directory containing the most recently opened macro file.
|MAXB|	Maximum value of the blanking variable in the active zones.
|MAXC|	Maximum value of the contour variable in the active zones.
|MAXI|, |MAXJ|, |MAXK|	[I, J, K]-dimension of the first active zone. For finite-element zones, `MAXI` is the number of nodes, `MAXJ` is the number of elements, and `MAXK` is the number of nodes per element (cell-based) or total number of faces (face-based) of the first active finite-element zone.
|MAX*n*|	Maximum value of the variable assigned to the *n*-axis, where *n* is one of: A[1], R, X, Y, or Z.
|MAXS|	Maximum value of the scatter sizing variable in the active zones.
|MAXU|, |MAXV|, |MAXW|	Maximum value of the variable assigned to the [X, Y, Z]-vector component of the active zones.
|MAXVAR[*nnn*]|	Maximum value of the variable *nnn* in the active zones.

Variables	Notes
|MAXX|, |MAXY|, |MAXZ|	Maximum value of the variable assigned to the [X, Y, Z] axes of the active zones.
|MINB|	Minimum value of the blanking variable in the active zones.
|MINC|	Minimum value of the contour variable in the active zones.
|MINS|	Minimum value of the scatter sizing variable in the active zones.
|MINU|, |MINV|, |MINW|	Minimum value of the variable assigned to the [X, Y, Z]-vector component in the active zones.
|MINX|, |MINY|, |MINZ|	Minimum value of the variable assigned to the [X, Y, Z] axes of the active zones.
|MINVAR[*nnn*]|	Minimum value of the variable *nnn* in the active zones.
|MIN*n*|	Minimum value of the variable assigned to the *n*-axis, where *n* is one of: A[1], R, X, Y, or Z.
|NUMFRAMES|	Number of frames.
|NUMPAGES|	Number of pages.
|NUMFIELDMAPS|	Number of fieldmaps assigned to the active frame.
|NUMLINEMAPS|	Number of linemaps assigned to the active frame.
|NUMPROCESSORSUSED|	Number of processors that Tecplot 360 uses. This may differ from the total number in the machine if the `$!Limits MaxAvailableProcessors` configures usage differently. By default, Tecplot 360 uses all available processors in the machine.
|NUMVARS|	Number of variables in the current dataset.
|NUMZONES|	Number of zones in the current dataset.
|OPSYS|	Returns 1=Linux/Macintosh, 2=Windows.
|PAPERHEIGHT|	The height of the paper (in inches).
|PAPERSIZE|	The size of the paper (e.g. Letter or A4).
|PAPERWIDTH|	The width of the paper (in inches).
|PLATFORMNAME|	Returns the type of platform (e.g. SGI or Windows).
|PLOTTYPE|	Returns the plot type of a frame. 0 = Sketch, 1 = XY Line, 2 = 2D, 3 = 3D, 4 = Polar Line.
|PRIMARYSLICEPOS|	Returns the position of the primary slice. (Not available for arbitrary slices.)
|PRINTFNAME|	Returns the file name of the last file sent for printing.
|SLICEPLANETYPE|	Plane type to which slices are assigned.
|SOLUTIONTIME|	The current solution time for the specified zone, fieldmap, or linemap.

Variables	Notes
|STARTSLICEPOS|	Returns the position of the first slice in a group. (Not available for arbitrary slices.)
|STREAMSTARTPOS[*nnn*]|	Indicates the starting position in X, Y, Z coordinates of streamtrace number *nnn*. For example, to produce the starting position of the first streamtrace, use |STREAMSTARTPOS[1]|.
|STREAMTYPE|	Returns streamtrace type, such as "Surface Line" or "Surface Ribbon".
|TECHOME|	Path to the home directory.
|TECPLOTMINORREVISION|	Returns the minor revision number of the Tecplot product. This number is the last of the four numbers presented in the full version number. This number always increases and can be relied on to determine the exact revision.
|TECPLOTVERSION|	The version number of the Tecplot product.
|TIME|	The current time in the form of *hh:mm:ss*.
|VARNAME[*n*]|	The name of a variable specified by index.
|ZONEMESHCOLOR[*nnn*]|	Returns the color of the mesh for zone *nnn*.
|ZONENAME[*nnn*]|	Returns the name of zone *nnn*.

[1] where A represents the theta (or angle) axis variable in Polar line plots.

System Environment Variables

System environment variables can be accessed directly from within Tecplot 360 by preceding an environment variable name with a "$" and surrounding it with vertical bars ("|"). Using environment variables within Tecplot 360 adds another degree of flexibility to macros by taking advantage of each user's customized environment.

If an environment variable is missing, an error is generated and macro processing is terminated.

Example 1

To compare a macro variable with an environment variable:

```
$!IF |SESSION_COEFF| == |$DEFAULT_COEFF|
    # perform some default processing
$!ENDIF
```

Where the `DEFAULT_COEFF` environment variable was set to some specified value of type double before starting Tecplot 360.

Example 2

To create a string from an environment variable:

```
$!VARSET |AUTHOR| = "Author: |$LOGNAME|"
```

User-defined variables are written using the macro variable name surrounded by vertical bars ("|"). The variable name can be up to 32 characters in length. If a macro variable is defined (using the $!VARSET command) and it is named the same as an existing intrinsic macro variable, then the user-defined variable takes precedence and the intrinsic value is not effected. The intrinsic macro variable can be recovered if you remove the user-defined variable using $!REMOVEVAR.

Assigning Values to Macro Variables

The $!VARSET command is used to assign a value to a macro variable. The $!VARSET command has the following syntax:

$!VARSET <macrovar> <op> <double>

where <op> can be one of =, -=, +=, *=, or /=.

Examples:

Example 1:

Add 2 to the macro variable |ABC|:

```
$!VARSET |ABC| += 2
```

Example 2:

Set |ABC| to be equal to 37:

```
$!VARSET |ABC| = 37
```

Example 3:

Multiply |ABC| by 1.5:

```
$!VARSET |ABC| *= 1.5
```

Example 4:

Set |ABC| to the result of an expression involving other variables and a constant; the expression must be enclosed in parentheses:

```
$!VARSET |ABC| = (|A| + |B| * |C| / 2)
```

Assigning a String to a Macro Variable

Macro variables can be assigned to strings as well as to values. When using strings, only the "=" operator may be used.

Example:

Assign the string "myfile.plt" to the variable |FNAME|. Use |FNAME| in the $!READDATASET command:

```
$!VARSET |FNAME| = "myfile.plt"
$!READDATASET "|FNAME|"
```

Note that double quotes (") had to be used in the $!READDATASET command even though |FNAME| represents a string.

Replacement Text Use

You can assign replacement text to a macro variable. This is useful for handling cases where a macro variable may not be initialized. A macro variable with |AAAA:=XXXXX| will produce XXXXX if AAAA is not defined. This does not work with intrinsic variables.

Example:

Read in a data file assigned to the variable FNAME. If FNAME is unassigned, read in "t.dat":

```
$!READDATASET "|FNAME:=t.dat|"
```

Macro Function Variables

Macro function variables are written using a number *n*, surrounded by vertical bars (|). The number represents the *nth* parameter from the $!RUNMACROFUNCTION command.

Examples:

Example 1:

The following commands define a macro function that uses two parameters and a command to run the

macro function. The first parameter to the macro function is the amount to rotate about the X-axis and the second parameter is the amount to rotate about the Y-axis:

The command to run the macro function will cause a rotation of 10 degrees about the X-axis and 20 degrees about the Y-axis.

```
#!MC  1410
$!MACROFUNCTION NAME = "3D Rotation Animation"
  $!ROTATE3DVIEW X
    ANGLE = |1|
  $!ROTATE3DVIEW Y
    ANGLE = |2|
$!ENDMACROFUNCTION
#
#
$!RUNMACROFUNCTION "3D Rotate " (10, 20)
```

Example 2:

The following commands define a macro function that opens a layout file and then appends a second layout file.

```
$!MACROFUNCTION
  NAME = "OL2"
  $!OPENLAYOUT "|1|"
  $!OPENLAYOUT "|2|"
    APPEND = TRUE
$!ENDMACROFUNCTION
#
#
$!RUNMACROFUNCTION "OL2" ("g1.lay","g2.lay")
```

Using Formats in Macro Variables

When a macro variable is expanded and the macro variable is a numeric value, it is expanded using a "best float" format. It tries to make the number look as simple as possible while still retaining as much accuracy as possible. If you want the number to be formatted in a specific way then you can include C-style number formatting strings in the macro variable specification.

The syntax for including a format string is:

```
|macrovariable%formatstring|
```

The *formatstring* should be in the following format. Note the flags, width, and precision are all optional; only the specifier is required. The brackets shown are not part of the format string:

```
[flags][width][.precision]specifier
```

The following flags are available:

flag	Effect
-	Left-justify (default is right-justified).
+	Precede positive numbers with a + (default is to sign only negative numbers).
space	A blank space will be written in place of the sign if the number is positive.
0	Left-pads numbers with zeroes to fill the specified width (rather than spaces)

The width specifies the minimum number of characters to be printed. If the dynamic text string is shorter than this length, it is padded with spaces. The string is not truncated if it is longer than this length.

Specifier	Produces
s	string of characters
d	signed integer
e	scientific notation with a lowercase "e"
E	scientific notation with an uppercase "E"
f	floating point
g	use %e or %f, whichever is shorter
G	use %E or %f, whichever is shorter
u	unsigned integer, written out in decimal format
o	unsigned integer, written out in octal format
x	unsigned integer, written out in hexadecimal (where a - f are lowercase)
X	unsigned integer, written out in hexadecimal (where A - F are uppercase)

Example 1:

Suppose you want to pause a macro and display the message "Maximum contour value is: xxxxxx" where xxxxxxx only has two digits to the right of the decimal place. You would use:

```
$!Pause "Maximum contour value is: |MAXC%.2f|"
```

If `|MAXC|` currently has a value of 356.84206 then the dialog would show:

"Maximum contour value is: 356.84"

Example 2:

If, in the above example, you wanted to use exponential format you could use:

```
$!Pause "Maximum contour value is: |MAXC%12.6e|"
```

Here the result would be:

"Maximum contour value is: 3.568421e+02"

Macro Preprocessor Directives

There can be times when you need to maintain a macro that will run differently for different versions of Tecplot 360. One example is a custom tecplot.cfg file. You may want to add instructions that only work for the latest version of Tecplot 360. These newer commands will make your tecplot.cfg file fail when running an older version. To aleviate this problem, use the following preprocessing instructions:

```
#if TecplotVersion op yyyy.r
... macro instructions
#endif
```

Where *op* can be one of <, >, ==, >= or <=.

Example:

```
#if TecplotVersion >= 2019.1
  # Commands that only work with 2019.1 or newer ...
  $!SomeCommand
  $!SomeCommand
  $!SomeCommand
#endif
```

This capability was introduced in Tecplot 360 2019 R1 and thus is only useful when running versions >= to that version.

Macro Commands

This chapter lists Tecplot 360's macro commands alphabetically. Optional parameters are enclosed within square brackets ([]). Items within double angle brackets (<<>>) represent parameter subcommands listed and described in Parameter Subcommands.

A-D

$!ACTIVEFIELDMAPS

Syntax:

```
$!ACTIVEFIELDMAPS <op> <set>
[no optional parameters]
```

Description:

A SetValue command that changes the set of active field maps (thus changing the active zones) considered for plotting.

Examples

Example 1

Make only field maps 1, 3, 4 and 5 active for plotting:

```
$!ACTIVEFIELDMAPS = [1,3-5]
```

Example 2

Add zones 33, 34, 35, and 36 to the set of active field maps:

```
$!ACTIVEFIELDMAPS += [33-36]
```

Example 3

Remove zones 1, 2, 3, 9, 10 and 11 from the set of active field maps:

```
$!ACTIVEFIELDMAPS -= [1-3,9-11]
```

$!ACTIVELINEMAPS

Syntax:

```
$!ACTIVELINEMAPS <op> <set>
[no optional parameters]
```

Description:

A SetValue command that changes the set of line mappings considered for plotting.

Examples

Example 1:

Make only line-mappings 1, 3, 4 and 5 active for plotting:

```
 $!ACTIVELINEMAPS = [1,3-5]
```

Example 2:

Add line-maps 33, 34, 35 and 36 to the set of active line-mappings:

```
$!ACTIVELINEMAPS += [33-36]
```

Example 3:

Remove line-maps 1, 2, 3, 9, 10 and 11 from the set of active line-mappings:

```
$!ACTIVELINEMAPS -= [1-3,9-11]
```

$!ALTERDATA

Syntax:

```
$!ALTERDATA  [zonelist]
  EQUATION = <string>
  [optional parameters]
```

Description:

The ALTERDATA function operates on a data set within Tecplot 360 using FORTRAN-like equations. See

the User's Manual for more information on using equations in Tecplot 360. The *zonelist* parameter specifies the set of zones on which to operate, where zonelist is a list of zones or zone ranges separated by a comma (","). Zone ranges are separated by a hyphen ("-"). If *zonelist* is omitted, all zones are affected. NOTE: the values for the *zonelist* parameter must be enclosed in square brackets. (For example, use `$!ALTERDATA [1,3]` to apply `ALTERDATA` to zones 1 and 3).

Required Parameters

Parameter	Syntax	Default	Notes
EQUATION	= <string>		This assigns the equation to use to operate on the data.

Optional Parameters

Parameter	Syntax	Default	Notes
DATATYPE	= <datatype>	SINGLE	Assign the precision given to the destination variable (that is, the variable on the left hand side of the equation). This only applies if the equation creates a new variable. (See Example 2:)
IGNOREDIVIDEBYZERO	= <boolean>	NO	If YES, the equation will be processed even if it includes a division by zero. 0/0 will result in 0, while other divisions will return the largest or smallest possible result depending on the sign of the dividend. If NO, an equation with a division by zero will not be processed at all.
IRANGE {			See Range Parameters following, for information on specifying range index values.
MIN	= <integer>	1	
MAX	= <integer>	0	
SKIP }	= <integer>	1	
JRANGE {			See Range Parameters, following, for information on specifying range index values.
MIN	= <integer>	1	
MAX	= <integer>	0	
SKIP }	= <integer>	1	

Parameter	Syntax	Default	Notes
KRANGE {			See Range Parameters, following, for information on specifying range index values.
MIN	= <integer>	1	
MAX	= <integer>	0	
SKIP }	= <integer>	1	
VALUELOCATION	= <valuelocation>	AUTO	Assign the location to destination variable.
FEDERIVATIVEMETHOD	= <federivativemethod>	MOVINGLEASTSQUARES	Method for computing finite-element derivatives. If GREENGAUSS, the Green-Gauss method will be preferred for calculating derivatives of finite elements. The calculation will fall back to Moving Least Squares for polygonal or polyhedral elements or higher order elements. Moving Least Squares will also be used for second or higher order derivatives.

Range Parameters

The `IRANGE`, `JRANGE`, and `KRANGE` parameters limit the data altered by the equation. The specification of range indices follow these rules:

- All indices start with 1 and go to some maximum index *m*.
- The number 0 can be used to represent the maximum index *m*. If the maximum index *m* = 15, specifying 0 sets the range index to 15.
- Negative values represent the offset from the maximum index. If a value of -2 is specified, and the maximum index *m* is 14, the value used is 14-2, or 12.

Examples

Example 1:

The following example adds one to X for zones 1 and 3 for every data point:

```
$!ALTERDATA [1,3]
 EQUATION = "x = x+1"
```

Example 2:

The following example creates a new, double precision variable called `DIST`:

```
$!ALTERDATA
  EQUATION = "{DIST} = SQRT(X**2 + Y**2)"
  DATATYPE = DOUBLE
```

Example 3:

The following equations set a variable called P to zero along the boundary of an IJ-ordered zone:

```
$!ALTERDATA
 EQUATION = "{P} = 0"
 IRANGE {MAX = 1}
$!ALTERDATA
 EQUATION = "{P} = 0"
 IRANGE {MIN = 0}
$!ALTERDATA
 EQUATION = "{P} = 0"
 JRANGE {MAX = 1}
$!ALTERDATA
 EQUATION = "{P} = 0"
 JRANGE {MIN = 0}
```

Example 4:

By following a variable reference with brackets "[" and "]" you may designate a specific zone from which to get the variable value. For example:

```
V3 = V3 -V3[1]
X = (X[1] + X[2] + X[3]) / 3
{TempAdj} = {Temp}[7] - {Adj}
V7 = V1[19] - 2*C[21] + {R/T}[18]
```

The zone number must be a positive integer constant less than or equal to the number of zones. The zone designated must have the same structure (finite-element, I-, IJ-, or IJK-ordered) and dimensions (number of nodes and so forth)

$!ANIMATECONTOURLEVELS

Syntax:

```
$!ANIMATECONTOURLEVELS
  START = <integer>
  END   = <integer>
  [Optional Parameters]
```

Description

Produce an animation of a contour line plot by showing a single level at a time. The animation varies according to the currently defined contour levels and is limited by the values in the START, END, and SKIP parameters. To create a movie file, add $!EXPORTSETUP commands before this command.

Required Parameters

Parameter	Syntax	Default	Notes
START	= <integer>		Starting contour level number to animate.
END	= <integer>		Ending contour level number to animate.

Optional Parameters

Parameter	Syntax	Default	Notes
CREATEMOVIEFILE	= <boolean>	NO	If YES, must be preceded by $!EXPORTSETUP commands.
SKIP	= <integer>	1	Level skip.

Example:

The following command animates the first four contour levels to an AVI file:

```
$!EXPORTSETUP EXPORTFORMAT = AVI
$!EXPORTSETUP EXPORTFNAME = "contourlevels.avi"
$!ANIMATECONTOURLEVELS
  START = 1
  END   = 4
  CREATEMOVIEFILE = YES
```

$!ANIMATEIJKBLANKING

Syntax:

```
$!ANIMATEIJKBLANKING
NUMSTEPS = <integer>
  [Optional Parameters]
```

Description:

Produce an animation of different IJK-blankings in your plot. The animation starts at one IJK-blanking setting and marches through intermediate steps to a second setting. To create a movie file, add $!EXPORTSETUP commands before this command.

Required Parameter

Parameter	Syntax	Default	Notes
NUMSTEPS	= <integer>		Number of intermediate steps for the animation.

Optional Parameters

Parameter	Syntax	Default	Notes
IMINFRACT	= <dexp>	0.1	Minimum fraction for blanking at the start of animation for the I-index. Actual I-index is equal to IMINFRACT*IMAX.
JMINFRACT	= <dexp>	0.1	Minimum fraction for blanking at the start of animation for the J-index. Actual J-index is equal to JMINFRACT*JMAX.
KMINFRACT	= <dexp>	0.1	Minimum fraction for blanking at the start of animation for the K-index. Actual K-index is equal to KMINFRACT*KMAX.
IMAXFRACT	= <dexp>	1.0	Maximum fraction for blanking at the start of animation for the I-index. Actual I-index is equal to IMAXFRACT*IMAX.
JMAXFRACT	= <dexp>	1.0	Maximum fraction for blanking at the start of animation for the J-index. Actual J-index is equal to JMAXFRACT*JMAX.
KMAXFRACT	= <dexp>	1.0	Maximum fraction for blanking at the start of animation for the K-index. Actual K-index is equal to KMAXFRACT*KMAX.
IMINFRACT2	= <dexp>	0.8	Minimum fraction for blanking at the end of animation for the I-index. Actual I-index is equal to IMINFRACT*IMAX.
JMINFRACT2	= <dexp>	0.8	Minimum fraction for blanking at the end of animation for the J-index. Actual J-index is equal to JMINFRACT*JMAX.
KMINFRACT2	= <dexp>	0.8	Minimum fraction for blanking at the end of animation for the K-index. Actual K-index is equal to KMINFRACT*KMAX.
IMAXFRACT2	= <dexp>	1.0	Maximum fraction for blanking at the end of animation for the I-index. Actual I-index is equal to IMAXFRACT*IMAX.

Parameter	Syntax	Default	Notes
JMAXFRACT2	= <dexp>	1.0	Maximum fraction for blanking at the end of animation for the J-index. Actual J-index is equal to JMAXFRACT*JMAX.
KMAXFRACT2	= <dexp>	1.0	Maximum fraction for blanking at the end of animation for the K-index. Actual K-index is equal to KMAXFRACT*KMAX.
CREATEMOVIEFILE	= <boolean>	NO	If YES, must be preceded by $!EXPORTSETUP commands.
LIMITSCREENSPEED	= <boolean>	NO	Whether to limit the maximum number of frames per second for animations displayed on the screen. See MAXSCREENSPEED.
MAXSCREENSPEED	= <double>	12.0	If LIMITSCREENSPEED is true, sets the maximum number of frames per second for animations displayed on the screen. If CREATEMOVIEFILE is true, this setting is ignored.

Example:

The following example produces an animation showing a band of I-planes traversing the entire data field:

```
$!ANIMATEIJKBLANKING
NUMSTEPS = 6
IMINFRACT = 0.1
JMINFRACT = 0.0
KMINFRACT = 0.0
IMAXFRACT = 1.0
JMAXFRACT = 1.0
KMAXFRACT = 1.0
IMINFRACT2 = 1.0
JMINFRACT2 = 0.0
KMINFRACT2 = 0.0
IMAXFRACT2 = 1.0
JMAXFRACT2 = 1.0
KMAXFRACT2 = 1.0
```

$!ANIMATEIJKPLANES

Syntax:

```
$!ANIMATEIJKPLANES
  START = <integer>
  END = <integer>
  [optional parameters]
```

Description:

Produce an animation that cycles through I-, J-, or K-planes in an IJK-ordered data set. To create a movie file, add `$!EXPORTSETUP` commands before this command.

Required Parameters

Parameter	Syntax	Default	Notes
START	= <integer>		Starting plane index
END	= <integer>		Ending plane index

Optional Parameters

Parameter	Syntax	Default	Notes
CREATEMOVIEFILE	= <boolean>	NO	If YES, must be preceded by $!EXPORTSETUP commands
LIMITSCREENSPEED	= <boolean>	NO	Whether to limit the maximum number of frames per second for animations displayed on the screen. See MAXSCREENSPEED.
MAXSCREENSPEED	= <double>	12.0	If LIMITSCREENSPEED is true, sets the maximum number of frames per second for animations displayed on the screen. If CREATEMOVIEFILE is true, this setting is ignored.
PLANES	= <ijkplane>	I	Specify I, J or K
SKIP	= <integer>	1	Index skip

Example

The following example generates an animation of the I-planes 1, 3, 5, 7, and 9:

```
$!ANIMATEIJKPLANES
  PLANES = I
  START = 1
END    = 9
SKIP   = 2
```

$!ANIMATEISOSURFACES

Syntax:

```
$!ANIMATEISOSURFACES
  [optional parameters]
```

Description:

The macro command $!ANIMATEISOSURFACES produces an animation of a series of iso-surfaces beginning with the iso-surface defined by STARTVALUE and ending with the iso-surface defined by ENDVALUE. To create a movie file, add $!EXPORTSETUP commands before this command.

Optional Parameters

Parameter	Syntax	Default	Notes
STARTVALUE	= <integer>	00	STARTVALUE is the value of the contour variable for the first iso-surface in the animation.
ENDVALUE	= <integer>	0.0	ENDVALUE is the value of the contour variable for the last iso-surface in the animation.
NUMSTEPS	= <integer>	20	Number of iso-surfaces to distribute between the start and end iso-surfaces values.
CREATEMOVIEFILE	= <boolean>	NO	If YES, must be preceded by $!EXPORTSETUP commands.
GROUP	= <integer>	1	values 1- 8
LIMITSCREENSPEED	= <boolean>	NO	Whether to limit the maximum number of frames per second for animations displayed on the screen. See MAXSCREENSPEED.
MAXSCREENSPEED	= <double>	12.0	If LIMITSCREENSPEED is true, sets the maximum number of frames per second for animations displayed on the screen. If CREATEMOVIEFILE is true, this setting is ignored.

Go To, Loop, Bounce, Forward, and Backward are only used by the interface. Forward and Backward can be simulated using appropriate values STARTVALUE and ENDVALUE. If ENDVALUE < STARTVALUE, the animation goes 'backward'. If ENDVALUE > STARTVALUE, the animation goes 'forward'. Goto can be simulated if ENDVALUE == STARTVALUE. That is, it can be simulated if the animation goes 'one step'. Loop and Bounce can be accomplished by animating the file multiple times.

When Recording, the macro recorded contains exactly the animation done in the interface. So if you bounce three times through the data, you will record three sets of forward and backward commands. Similarly, if you use the "one step" options a lot, you will record a lot of individual macro commands. If you interrupt part way through an animation, you will record a partial animation macro of those steps you did animate through.

Example

The following example creates an animation of iso-surfaces:

```
$!ANIMATEISOSURFACES
  STARTVALUE = 1
  ENDVALUE = 30
  NUMSTEPS = 30
```

$!ANIMATELINEMAPS

Syntax:

```
$!ANIMATELINEMAPS
  START = <integer>
  END = <integer>
  [Optional Parameters]
```

Description:

Produce an animation of one Line-mapping at a time. To create a movie file, add $!EXPORTSETUP commands before this command.

Required Parameters

Parameter	Syntax	Default	Notes
START	= <integer>		Starting Line-map number
END	= <integer>		Ending Line-map number

Optional Parameters

Parameter	Syntax	Default	Notes
CREATEMOVIEFILE	= <boolean>	NO	If YES, must be preceded by $!EXPORTSETUP commands

Parameter	Syntax	Default	Notes
LIMITSCREENSPEED	= <boolean>	NO	Whether to limit the maximum number of frames per second for animations displayed on the screen. See MAXSCREENSPEED.
MAXSCREENSPEED	= <integer>	12.0	If LIMITSCREENSPEED is true, sets the maximum number of frames per second for animations displayed on the screen. If CREATEMOVIEFILE is true, this setting is ignored.
SKIP	= <integer>	1	Line-map skip

Example

The following example creates an animation showing plots of Line-maps 2, 4, 6, 8 and 10:

```
$!ANIMATELINEMAPS
  START = 2
  END   = 10
  SKIP = 2
```

$!ANIMATESLICES

Syntax:

```
$!ANIMATESLICES [Group]
  START = <integer>
  END = <integer>
  [Optional Parameters]
```

Description:

The macro command `$!ANIMATESLICES` uses the currently defined start and end slice position. Use `$!SLICEATTRIBUTES` to set these positions; `$!ANIMATESLICES` then redefines how many intermediate slices are to be used, then animates a sub-set of those slices. To create a movie file, add `$!EXPORTSETUP` commands before this command.

Required Parameters

Parameter	Syntax	Notes
START	= <integer>	START and END are measured in steps based on NUMSLICES between the slice group's start slice value (at step=1) and end slice values (at step = NumSlices).
END	= <integer>	

Optional Parameters

Parameter	Syntax	Default	Notes
GROUP	= <integer>	1	values 1- 8
NUMSLICES	= <integer>	2	Number of slices to distribute between the start and end slice locations as defined by START and END in $!SLICEATTRIBUTES.
CREATEMOVIEFILE	= <boolean>	NO	If YES, must be preceded by $!EXPORTSETUP commands.
LIMITSCREENSPEED	= <boolean>		Whether to limit the maximum number of frames per second for animations displayed on the screen. See MAXSCREENSPEED.
MAXSCREENSPEED	= <double>	12.0	If LIMITSCREENSPEED is true, sets the maximum number of frames per second for animations displayed on the screen. If CREATEMOVIEFILE is true, this setting is ignored.

Go To, Loop, Bounce, Forward, and Backward are only used by the interface. Forward and Backward can be simulated using appropriate values STARTVALUE and ENDVALUE. If ENDVALUE < STARTVALUE, the animation goes 'backward'. If ENDVALUE > STARTVALUE, the animation goes 'forward'. Goto can be simulated if ENDVALUE == STARTVALUE, i.e. the animation goes 'one step'. Loop and Bounce can be accomplished by calling the file multiple times.

When Recording, the macro recorded contains exactly the animation done in the interface. So if you bounce three times through the data, you will record three sets of forward and backward commands. Similarly, if you use the "one step" options a lot, you will record a lot of individual macro commands. If you interrupt part way through an animation, you will record a partial animation macro of those steps you did animate through.

Example

The following example creates an animation of 3D slices:

```
$!ANIMATESLICES
  START = 1
  END = 30
  NUMSLICES = 30
```

$!ANIMATESTREAM

Syntax:

```
$!ANIMATESTREAM
  [optional parameters]
```

Description:

Produce an animation of stream markers or dashes, moving along the currently defined streamtrace paths. To create a movie file, add `$!EXPORTSETUP` commands before this command.

Optional Parameters

Parameter	Syntax	Default	Notes
CREATEMOVIEFILE	= <boolean>	NO	If YES, must be preceded by $!EXPORTSETUP commands
NUMCYCLES	= <integer>	3	Number of cycles in the animation. Each cycle shows stream markers or dashes, moving along a streamtrace path. If DT is the streamtrace delta time, then at the end of the cycle, the markers or dashes will have moved `(2*DT*(STEPSPERCYCLE-1))/(STEPSPERCYCLE)` in time.
STEPSPERCYCLE	= <integer>	20	Number of steps to use for each cycle of the animation. Increase this number to produce a smoother animation.

Example

The following example animates streamtraces for five cycles with each cycle using ten steps:

```
$!ANIMATESTREAM
  STEPSPERCYCLE = 10
  NUMCYCLES     = 5
```

$!ANIMATETIME

Syntax:

```
$!ANIMATETIME
  [optional parameters]
```

Description:

Produce an animation of transient data. To create a movie file, add `$!EXPORTSETUP` commands before this command.

Optional Parameters:

Parameter	Syntax	Default	Notes
CREATEMOVIEFILE	= <boolean>	NO	If YES, must be preceded by $!EXPORTSETUP commands.
ENDTIME	= <double>	The last timestep as defined by the currently active strands	If the SolutionTime entered does not exist, the nearest SolutionTime less than the entered time is used.
LIMITSCREENSPEED	= <boolean>	NO	Whether to limit the maximum number of frames per second for animations displayed on the screen. See MAXSCREENSPEED.
MAXSCREENSPEED	= <double>	12.0	If LIMITSCREENSPEED is true, sets the maximum number of frames per second for animations displayed on the screen. If CREATEMOVIEFILE is true, this setting is ignored.
SKIP	= <integer>	1	
STARTTIME	= <double>	first time step as defined by the currently active strands	If the SolutionTime entered does not exist, the nearest SolutionTime less than the entered time is used.

Go To, Loop, Bounce, Forward, and Backward are only used by the interface. Forward and Backward can be simulated using appropriate values `STARTVALUE` and `ENDVALUE`. If `ENDVALUE` < `STARTVALUE`, the animation goes 'backward'. If `ENDVALUE` > `STARTVALUE`, the animation goes 'forward'. Goto can be simulated if `ENDVALUE` == `STARTVALUE`, i.e. the animation goes 'one step'. Loop and Bounce can be accomplished by calling the file multiple times.

When Recording, the macro recorded contains exactly the animation done in the interface. So if you bounce three times through the data, you will record three sets of forward and backward commands. Similarly, if you use the "one step" options a lot, you will record a lot of individual macro commands. If you interrupt part way through an animation, you will record a partial animation macro of those steps you did animate through.

$!ANIMATEZONES

Syntax:

```
$!ANIMATEZONES
  START = <integer>
  END = <integer>
  [optional parameters]
```

Description:

Produce an animation showing one zone at a time. To create a movie file, add $!EXPORTSETUP commands before this command. This command will not work if the active frame contains a transient data set.

Required Parameters

Parameter	Syntax	Default	Notes
START	= <integer>		Starting zone number
END	= <integer>		Ending zone number

Optional Parameters

Parameter	Syntax	Default	Notes
CREATEMOVIEFILE	= <boolean>	NO	If YES, must be preceded by $!EXPORTSETUP commands.
LIMITSCREENSPEED	= <boolean>	NO	Whether to limit the maximum number of frames per second for animations displayed on the screen. See MAXSCREENSPEED.
MAXSCREENSPEED	= <integer>	12.0	If LIMITSCREENSPEED is true, sets the maximum number of frames per second for animations displayed on the screen. If CREATEMOVIEFILE is true, this setting is ignored.
SKIP	= <integer>	1	Zone skip.
ZONEANIMATIONMODE	= *[STEPBYNUMBER, GROUPSTEPBYNUMBER, STEPBYTIME]*	STEPBYNUMBER	

Example:

The following example animates just the first five zones:

```
$!ANIMATEZONES
  START = 1
  END = 5
```

$!ATTACHDATASET

Syntax:

```
$!ATTACHDATASET
  [optional parameters]
```

Description:

Attach the active frame to the data set of another frame. Use PAGENUM, if the other frame is on a difference page. This command is usually found only in layout files generated by Tecplot 360. Note that the $!PLOTTYPE command automatically executes an $!ATTACHDATASET command if a frame mode is requested in a frame that does not have an attached data set. Tecplot 360 attaches the data set from the closest frame (in drawing order) having an attached data set.

Optional Parameters

Parameter	Syntax	Default	Notes
FRAME	= <integer>	First frame with a data set	Within the page specified or implied, if FRAME is not supplied, Tecplot 360 searches for a data set in a frame below the topmost frame of the page to attach.
PAGENUM	= <integer>	current page	If PAGENUM is not supplied the current page is used.

Examples

Example 1:

The following example attaches to the active frame the data set from the second frame drawn when doing a Redraw All:

```
$!ATTACHDATASET
 FRAME = 2
```

Example 2:

The following example attaches to the active frame the data set from the frame drawn next-to-last when doing a Redraw All:

```
$!ATTACHDATASET
```

$!ATTACHGEOM

Syntax:

```
$!ATTACHGEOM
  [optional parameters]
  <geometryrawdata>
```

Description:

Attach a geometry to the active frame.

Required Parameter

Parameter	Syntax	Default	Notes
<geometryrawdata>			This is the data which defines the size and relative shape of the geometry. This must be at the end of the command after any other parameters. Not required if WorldFileName is used for Geo Referenced Images.

Optional Parameters

Parameter	Syntax	Default	Notes
ANCHORPOS	<<anchorpos>>	See Notes	This assigns the anchor position (X,Y and Z) of the geometry. Z values are only obeyed for geo referenced images and LineSeg with GRID3D positioncoordsys. For geo referenced images, omitting the Z value will auto-assign the Z positon to one of three values. If the spatial variable assigned to the Z axis has a range with positive and negative values then the Z AnchorPos will be set to 0. If the range of the Z axis variable is entirely positive, then it will be set to the minimum value of the Z range. If the range of the Z axis is entirely negative, then it will be set to the maximum value of the Z range.
ARROWHEADANGLE	= <dexp>	12	Set the angle for arrowheads (in degrees).
ARROWHEADATTACHMENT	= <arrowheadattachment>	NONE	
ARROWHEADSIZE	= <dexp>	5%	Set the arrowhead size in Y-frame units (0-100).
ARROWHEADSTYLE	= <arrowheadstyle>	PLAIN	
ATTACHTOZONE	= <boolean>	NO	If YES, must include ZONE.
CLIPPING	= <clipping>	CLIPTTOVIEWPORT	
COLOR	= <color>	BLACK	
DATATYPE	= <fielddatatype>	FLOAT	
DRAWORDER	= <draworder>	AFTERDATA	
FILLCOLOR	= <color>	WHITE	
GEOMTYPE	= <geomtype>	LINESEGS	
IMAGEFILENAME	= <string>		
ISFILLED	= <boolean>		

Parameter	Syntax	Default	Notes
LINEPATTERN	= <linepattern>	SOLID	
LINETHICKNESS	= <dexp>	0.1%	Set the line thickness in Y-frame units (0-100).
MACROFUNCTIONCOMMAND	= <string>	Null	Set the macro command to execute when you hover over the geometry and press Ctrl-right-click.
MAINTAINASPECTRATIO	= <boolean>	YES	
NUMELLIPSEPTS	= <integer>	72	Numbers of points to use when drawing ellipses and circles.
PATTERNLENGTH	= <dexp>	2%	Set the pattern length in Y-frame units (0-100).
POSITIONCOORDSYS	= <coordsys>	GRID	Not available when using WORLDFILENAME for geo referenced images.
RESIZEFILTER	= <resizefilter>		The Resize filter determines how the image is resized to fit the screen. The following filters are available: • TextureFilter (Fast) - default- Tecplot 360 uses OpenGL textures to resize the image. This is the fastest option (given sufficient graphics space). However, the accuracy of the image may suffer, especially when reducing an image to a size much smaller than it was before. • Pixelated - Choose this option when the image is much larger than its original size and you want to see the individual pixels. This option is slower than the Fast (textures) for increasing the size of images. • Smooth - There are seven smooth options, all producing slightly different effects. These options are slower than the Fast (textures), but produce better effects for highly reduced images. In general, use the Smooth (Lanczos2) option unless you have specific image processing needs.
SCOPE	= <scope>	LOCAL	Set the scope to GLOBAL to draw this geometry in all "like" frames.

Parameter	Syntax	Default	Notes
TEXTUREFILTER		CUBIC	
WORLDFILENAME	= <string>		Used for attaching a Geo Referenced Image. Needs to be used in conjunction with IMAGEFILENAME. No RAWDATA needed.
ZONE	= <integer>	1	This is only used if ATTACHTOZONE = YES. This geometry is disabled if the zone assigned here is inactive.

Examples

Example 1:

The following example creates a red circle, with a radius equal to 25 percent of the height of the frame, in the center of the frame:

```
$!ATTACHGEOM
  POSITIONCOORDSYS = FRAME
  ANCHORPOS
  {
    X = 50
    Y = 50
  }
  GEOMTYPE = CIRCLE
  COLOR = RED
  RAWDATA
  25
```

Example 2:

The following example creates an L-shaped polyline with an arrowhead at the end:

```
$!ATTACHGEOM
  POSITIONCOORDSYS = FRAME
  ANCHORPOS
  {
    X = 20
    Y = 80
  }
  GEOMTYPE = LINESEGS
  ARROWHEADATTACHMENT = ATEND
  RAWDATA
  1
  3
```

```
  0 0
  0 -60
  40 0
```

$!ATTACHTEXT

Syntax:

```
$!ATTACHTEXT
  TEXT = <string>
  [optional parameters]
```

Description:

Attach text to the active frame

Required Parameter

Parameter	Syntax	Default	Notes
TEXT	= <string> or <rawstring>		For text annotations, with few exceptions, all characters specified in the TEXT sub-command are passed along to Tecplot without modification. If the macro processor encounters a two character newline, "\n", or single quote escape sequence, "\'", that itself is not escaped, then the two character sequence is replaced with a single newline or single quote character respectively. The newline character produces multiple lines for REGULAR text in the plot and in the text editor of the Text Details dialog. LaTeX annotations make heavy use of backslashes for commands which have traditionally been used to escape characters in strings. For LaTeX annotations that contain the following characters, the text string must be defined using Tecplot's raw string formatting so that the text will be delivered to the LaTeX toolchain as-is for processing: • A two character newline sequence, \n, or a LaTeX command that looks like a two character newline sequence such as, \nonumber • A single newline character • A single quote character When recording LaTeX annotations Tecplot will automatically use the raw string formatting for any LaTeX expression that contains one or more of the above special characters or character sequences.

Optional Parameters

Parameter	Syntax	Default	Notes
ANCHOR	= <textanchor>	LEFT	Specifies what part of the text to anchor to the frame.
ANCHORPOS	<<anchorpos>>		This assigns the anchor position for the text. Units are dependent on POSITIONCOORDSYS.

Parameter	Syntax	Default	Notes
ANGLE	= <dexp>	0.0	Text angle (in degrees).
ATTACHTOZONE	= <boolean>	NO	If YES, must include ZONE.
BOX	<<textbox>>		style for the box around the text
CLIPPING	= <clipping>	CLIPTOVIEWPORT	
COLOR	= <color>	BLACK	
LINESPACING	= <dexp>	1.0	Line spacing to use if text contains multiple lines.
MACROFUNCTIONCOMMAND	= <string>	NULL	Set the macro command to execute when you hover over the geometry and press Ctrl-right-click.
POSITIONCOORDSYS	= <coordsys>	FRAME	values = FRAME, GRID or GRID3D
TEXTSHAPE	<<textshape>>		Font family etc.
TEXTTYPE	= <texttype>	REGULAR	Specifies the Text Type. Supported values are `LATEX` and `REGULAR`. `REGULAR` Produces normal text annotations. `LATEX` Instructs Tecplot to pass the text string through the LaTeX toolchain (see the User's Manual for more information) to produce images which are then placed in the plot at the specified location. Only Text, AnchorPos, Height, and SizeUnits are valid parameters for LaTeX type.
SCOPE	= <scope>	LOCAL	Set the scope to GLOBAL to include this text in all "like" frames.
ZONE	= <integer>	1	This is only used if ATTACHZONE == YES. This text is disabled if the zone assigned here is inactive.

Examples

Example 1:

The following example creates the text ABC and positions it in the lower left corner of the frame:

```
$!ATTACHTEXT
  TEXT = "ABC"
```

Example 2:

The following example creates text at an angle and places it in the center of the frame. The text is drawn at an angle of 45 degrees:

```
$!ATTACHTEXT
  TEXT =  "TEXT AT AN ANGLE"
  ANGLE = 45
  ANCHORPOS {X=50 Y=50}
```

Example 3:

The following example creates the text using the Times Roman font. This text includes a text box:

```
$!ATTACHTEXT
  TEXT =  "TIMES-ROMAN"
  TEXTSHAPE
  {
    FONTFAMILY = "Times"
    ISBOLD = NO
    ISITALIC = NO
  }
  BOX
  {
    BOXTYPE = PLAIN
    MARGIN  = 20
  }
  ANCHORPOS {X=20 Y=20}
```

$!AXIALDUPLICATE

Syntax:

```
$!AXIALDUPLICATE
  ANGLE = <dexp>
  NUMDUPLICATES = <integer>
  [optional parameters]
```

Description:

Using the right-hand rule, make the specified number of duplicates of the specified set of zones, rotating the specified axis variables and/or vector variables successively by the given angle. You may optionally specify the origin and axis of rotation. See also $!ROTATEDATA.

Required Parameters

Parameter	Syntax	Notes
ANGLE	= <dexp>	Angle to rotate between each new set of zones (in degrees).
NUMDUPLICATES	= <integer>	Number of duplicate sets of zones to create

Optional Parameters

Parameter	Syntax	Default	Notes
ADDZONESTOEXISTINGSTRANDS	= <boolean>	YES	If true, the new zones become a part of the time strands that the original zones belong to. If false, new zones are assigned new strand IDs if the soruce zones belonged to strands, otherwise they are made static.
OFFSETANGLE	= <dexp>	0.0	The rotation, in degrees, to be added to the first rotated duplicate. For example, if ANGLE is 20, and OFFSETANGLE is 40, the first duplicate is made at 60°.
NORMALX	= <dexp>	0.0	For 3D rotation, the X component of a point other than the origin on the axis of rotation. Invalid for 2D rotation. If NORMALs are not specified, the rotation will occur around the Z axis.
NORMALY	= <dexp>	0.0	For 3D rotation, the Y component of a point other than the origin on the axis of rotation. Invalid for 2D rotation. If NORMALs are not specified, the rotation will occur around the Z axis.
NORMALZ	= <dexp>	1.0	For 3D rotation, the Z component of a point other than the origin on the axis of rotation. Invalid for 2D rotation. If NORMALs are not specified, the rotation will occur around the Z axis.
ORIGINX	= <dexp>	0.0	X coordinate of the center of rotation.
ORIGINY	= <dexp>	0.0	Y coordinate of the center of rotation.
ORIGINZ	= <dexp>	0.0	Z coordinate of the center of rotation (for 3D rotation only).

Parameter	Syntax	Default	Notes
UVARLIST	= <varset>		Set containing vector variable U components to rotate. If omitted, XVAR, YVAR, and if performing 3D rotation ZVAR must be supplied.
VVARLIST	= <varset>		Set containing vector variable V components to rotate. If omitted, XVAR, YVAR, and if performing 3D rotation ZVAR must be supplied.
WVARLIST	= <varset>		Set containing vector variable W components to rotate if performing 3D rotation, otherwise it must be omitted. If performing 3D rotation and omitted, XVAR, YVAR, and if performing 3D rotation ZVAR must be supplied.
XVAR	= <varref>		X variable to rotate. XVAR may be omitted if only rotating vectory variables in which case YVAR and ZVAR must be omitted. If the spatial variables are omitted then UVARLIST, VVARLIST, and if performing 3D rotation, WVARLIST sets must be supplied.
YVAR	= <varref>		Y variable to rotate. YVAR may be omitted if only rotating vectory variables in which case XVAR and ZVAR must be omitted. If the spatial variables are omitted then UVARLIST, VVARLIST, and if performing 3D rotation, WVARLIST sets must be supplied.
ZVAR	= <varref>		Z variable to rotate if performing 3D rotation otherise it must be omitted. ZVAR may be omitted if only rotating vectory variables in which case XVAR and YVAR must be omitted. If the spatial variables are omitted then UVARLIST, VVARLIST, and if performing 3D rotation, WVARLIST sets must be supplied.
ZONELIST	= <set>	All zones	Set specifying the zones to be duplicated. If omitted, duplicate all zones.

$!BASICCOLOR

Syntax:

```
$!BASICCOLOR
```

```
  [Optional Parameters]
```

Description:

A SetValue command that sets the red, green and blue components for any of the basic colors in Tecplot 360.

Optional Parameters

Parameter	Syntax	Default	Notes
BLACK	<<rgb>>	See Notes	R=0, G=0, B=0
BLUE	<<rgb>>	See Notes	R=45, G=45, B=255
CUSTOM1… CUSTOM56	<<rgb>>	See Notes	
CYAN	<<rgb>>	See Notes	R=0, G=255, B=255
GREEN	<<rgb>>	See Notes	R=0, G=210, B=0
PURPLE	<<rgb>>	See Notes	R=255, G=0, B=255
RED	<<rgb>>	See Notes	R=210, G=0, B=0
WHITE	<<rgb>>	See Notes	R=255, G=255, B=255
YELLOW	<<rgb>>	See Notes	R=255, G=255, B=45

Example:

Set the `CUSTOM8` color to be brown:

```
$!BASICCOLOR
  CUSTOM8
  {
    R = 165
    G  = 42
    B  = 42
  }
```

$!BASICCOLORLEGEND

Syntax:

```
$!BASICCOLORLEGEND
  [Optional Parameters]
```

Description:

A SetValue command that allows you to create and set the style of a legend for the basic colors in Tecplot 360. The legend can be used to display any attribute of the plot represented by a basic color (for example, materials). Each frame maintains a mapping of basic colors to names. Each basic color actually used in selected layers of the plot appears in the legend unless it is excluded.

Optional Parameters

Parameter	Syntax	Default	Notes
ANCHORALIGNMENT	= <anchoralignment>	TOPRIGHT	
BASICCOLORCONTROL			Name may include dynamic text variables (for example, to incorporate auxiliary data)
{			
BLACK	<<basiccolorcontrol>>	See notes	NAME = ”, SHOW = YES
BLUE	<<basiccolorcontrol>>	See notes	NAME = ”, SHOW = YES
CUSTOM1.. CUSTOM56	<<basiccolorcontrol>>	See notes	NAME = ”, SHOW = YES
CYAN	<<basiccolorcontrol>>	See notes	NAME = ”, SHOW = YES
GREEN	<<basiccolorcontrol>>	See notes	NAME = ”, SHOW = YES
PURPLE	<<basiccolorcontrol>>	See notes	NAME = ”, SHOW = YES
RED	<<basiccolorcontrol>>	See notes	NAME = ”, SHOW = YES
WHITE	<<basiccolorcontrol>>	See notes	NAME = ”, SHOW = YES
YELLOW	<<basiccolorcontrol>>	See notes	NAME = ”, SHOW = YES
}			
BOX	<<textbox>>	See notes	BOXTYPE = HOLLOW, MARGIN = 10, LINETHICKNESS = 0.1, COLOR = BLACK, FILLCOLOR = WHITE

Parameter	Syntax	Default	Notes
FIELDLAYERCONTROL			Determines which field layers' colors are included in legend
{			
USEMESH	= <boolean>	NO	
USECONTOUR	= <boolean>	NO	
USEVECTOR	= <boolean>	NO	
USESCATTER	= <boolean>	NO	
USESHADE	= <boolean>	NO	
USEEDGE	= <boolean>	NO	
}			
HEADERTEXT	= <string>		May include dynamic text variables
HEADERTEXTCOLOR	= <color>	BLACK	
HEADERTEXTSHAPE	<<textshape>>	See notes	FONTFAMILY = 'Helvetica', ISBOLD = YES, ISITALIC = NO, SIZEUNITS = FRAME, HEIGHT = 2.5
LABELTEXTCOLOR	= <color>	BLACK	
LABELTEXTSHAPE	<<textshape>>	See notes	FONTFAMILY = 'Helvetica', ISBOLD = NO, ISITALIC = NO, SIZEUNITS = FRAME, HEIGHT =2.5
LINELAYERCONTROL			Determines which line layers' colors are included in legend
{			
USELINES	= <boolean>	NO	
USESYMBOLS	= <boolean>	NO	
USEBARS	= <boolean>	NO	
USEERRORBARS	= <boolean>	NO	
}			
ROWSPACING	<op> <dexp>	= 1.2	
SHOW	= <boolean>	NO	
SHOWSYMBOLOUTLINE	= <boolean>	YES	
SORTBYLABELTEXT	= <boolean>	YES	
SYMBOLHEIGHT	= <dexp>	2.5	

Parameter	Syntax	Default	Notes
SYMBOLLINETHICKNESS	= <dexp>	0.1	
SYMBOLWIDTH	= <dexp>	2.5	
XYPOS	= <<xy>>	X = 95, Y = 75	

Example:

Create a basic color legend for a fruity plot:

```
$!BASICCOLORLEGEND
 SHOW = YES
 LINELAYERCONTROL
   {
   USELINES = YES
   }
 XYPOS
   {
   X = 70
   Y = 80
   }
 HEADERTEXT = '&(ZoneName[1])'
 BOX
   {
   BOXTYPE = FILLED
   }
 BASICCOLORCONTROL
   {
   BLACK
     {
     SHOW = NO
     }
   RED
     {
      NAME = 'Apple'
     }
   GREEN
     {
      NAME = 'Bean'
     }
   BLUE
     {
      NAME = 'Blueberry'
     }
   CYAN
```

```
    {
     NAME = 'Kiwi'
    }
  YELLOW
    {
     NAME = 'Banana'
    }
  PURPLE
    {
     NAME = 'Grape'
    }
  }
```

$!BASICSIZE

Syntax:

```
$!BASICSIZE
  [optional parameters]
```

Description:

A SetValue command that sets sizes of various objects like line thicknesses, line pattern length, font height, and so forth. Sizes can be assigned when interacting with Tecplot 360 by either entering an exact value or by choosing from a preset list of values. The $!BASICSIZE command allows you to change the values in the preset lists.

Optional Parameters

Parameter	Syntax	Default	Notes
ARROWHEADSIZES	<<basicsizelist >>	See Notes	Tiny=1.0, Small=3.0, Medium=5.0, Large=8.0, Huge=12.0
FRAMETEXTSIZES	<<basicsizelist >>	See Notes	Tiny=1.5, Small=2.0, Medium=3.0, Large=6.0, Huge=10.0
LINEPATLENGTHS	<<basicsizelist >>	See Notes	Tiny=0.5, Small=0.8, Medium=2.0, Large=3.0, Huge=5.0
LINETHICKNESSES	<<basicsizelist >>	See Notes	Tiny=0.02, Small=0.1, Medium=0.4, Large=0.8, Huge=1.5
POINTTEXTSIZES	<<basicsizelist >>	See Notes	Tiny=8, Small=11, Medium=14, Large=28, Huge=50
SYMBOLSIZES	<<basicsizelist >>	See Notes	Tiny=0.5, Small=1.0, Medium=2.5, Large=4.0, Huge=8.0

Parameter	Syntax	Default	Notes
TICKLENGTHS	<<basicsizelist >>	See Notes	Tiny=0.5, Small=1.2, Medium=2.0, Large=3.0, Huge=5.0

Example:

Change the medium line pattern length to be 2.5 percent:

```
$!BASICSIZE
  LINEPATLENGTHS
  {
    MEDIUM = 2.5
  }
```

$!BLANKING

Syntax:

```
$!BLANKING
  [optional parameters]
```

Description:

A SetValue command that changes settings for IJK- or value-blanking.

Optional Parameters

Parameter	Syntax	Default	Notes
DEPTH			
{			
INCLUDE	= <boolean>	NO	If YES, draws only those portions at the plot with depth values within the FROMFRONT and FROMBACK limits.
FROMFRONT	= <double>	0	FROMFRONT and FROMBACK are expressed as percentages of the overall 3D depth.
FROMBACK	= <double>	0	FROMFRONT and FROMBACK are expressed as percentages of the overall 3D depth.
}			
IJK			
{			

Parameter	Syntax	Default	Notes
INCLUDE	= <boolean>	NO	
IJKBLANKMODE	= <ijkblankmode>		
IMINFRACT	<op> <dexp>	= 0	Minimum and maximum fractions are in terms of percentages (0-100). Zero represents an index of one and 100 the maximum index.
JMINFRACT	<op> <dexp>	= 0	
KMINFRACT	<op> <dexp>	= 0	
IMAXFRACT	<op> <dexp>	= 50	
JMAXFRACT	<op> <dexp>	= 50	
KMAXFRACT	<op> <dexp>	= 50	
ZONE	= <integer>	0	Only one zone can be assigned to use IJK-blanking.
}			
VALUE			
{			
BLANKENTIRECELL	= <boolean>	YES	Set to NO to get precision blanking.
CONSTRAINT nnn	nnn = <integer>	1	Use <integer> specify which constraint to modify.
{			
COLOR	= <color>	BLACK	
CONSTRAINTOP2MODE	= <constraintop2mode>	USECONSTANT	
INCLUDE	= <boolean>	NO	
LINEPATTERN	= <linepattern>	SOLID	
LINETHICKNESS	= <double>	0.4	
PATTERNLENGTH	= <dexp>	2	
RELOP	= <valueblankrelop>	LESSTHANOREQUAL	
SHOW	= <boolean>	NO	
VALUECUTOFF	= <double>	0	

Parameter	Syntax	Default	Notes
VARA	= <varref>	None	
VARB	= <varref>	None	
}			
INCLUDE	= <boolean>	NO	Set to NO to turn off all value-blanking.
VALUEBLANKCELLMODE	= <valueblankcellmode>	ANYCORNER	
}			

Examples

Example 1:

Set IJK-blanking to cut away the minimum index corner:

```
$!BLANKING
  IJK
  {
    INCLUDE   = YES
    IMINFRACT = 0
    JMINFRACT = 0
    KMINFRACT = 0
    IMAXFRACT = 50
    JMAXFRACT = 50
    KMAXFRACT = 50
  }
```

Example 2:

Use value blanking to cut away all cells that have at least one node where variable 3 is less than or equal to 7.5:

```
$!BLANKING
  VALUE
  {
    INCLUDE = YES
    CONSTRAINT 1
    {
       INCLUDE = YES
       VARA = 3
       RELOP = LESSTHANOREQUAL
       VALUECUTOFF = 7.5
```

```
    }
  }
```

$!BRANCHCONNECTIVITY

Syntax:

```
$!BRANCHCONNECTIVITY
  ZONE   = <integer>
  [no optional parameters]
```

Description:

For zones where connectivity is shared, this command allows for branching of connectivity information from the specified zone.

Required Parameters

Parameter	Syntax	Default	Notes
ZONE	= <integer>		

Example:

Suppose Zones 2, 3 and 4 share connectivity. This command branches the connectivity of the second zone. Zones 3 and 4 will still share connectivity.

```
$!BRANCHCONNECTIVITY
  ZONE = 2
```

$!BRANCHFIELDDATAVAR

Syntax:

```
$!BRANCHFIELDDATAVAR
  ZONE = <integer>
  VAR  = <varref>
  [no optional parameters]
```

Description:

Allows for branching of specified variable in the specified zone for zones that share variables.

Required Parameters

Parameter	Syntax	Default	Syntax
VAR	= <varref>		
ZONE	= <integer>		

Example:

Assume Zones 1, 2 and 4 share variables 3 and 5. This command branches the third variable from the second zone. Variable 3 will still be shared by zones 1 and 4, while variable 5 will still be shared by all three zones:

```
$!BRANCHFIELDDATAVAR
  ZONE = 2
  VAR   = 3
```

$!BREAK

Syntax:

```
$!BREAK
  [no parameters]
```

Description:

Jump out of the current $!LOOP-ENDLOOP or $!WHILE-$!ENDWHILE.

Example:

```
$!LOOP 5
$!BREAK
$!ENDLOOP
```

$!COLORMAPATTRIBUTES

Syntax:

```
$!COLORMAPATTRIBUTES <string>
  CONTROLPOINT <<colormapcontrolpoints>>
```

Description:

Sets the control point attributes of a custom color map. The named color map must exist and be a

custom color map.

Required Parameters

Parameter	Syntax	Default	Notes
CONTROLPOINT	<<colormapcontrolpoints>>		Defines a control point. Multiple control points for a given color map may set in a COLORMAPATTRIBUTES command.

Example

Move the third control point for the custom colormap "My Small Rainbow" to 44% of the way across the colormap, setting the leading and trailing red values of this point to 90. :

```
$!COLORMAPATTRIBUTES "My Small Rainbow"
  CONTROLPOINT 3
  { COLORMAPFRACTION = 0.44 LEADRGB {R=90} TRAILRGB {R=90} }
```

$!COMPATIBILITY

Syntax:

```
$!COMPATIBILITY
  [optional parameters]
```

Description

Allow datasharing access and setting, without warning.

Optional Parameters

Parameter	Syntax	Default	Notes
ALLOWDATASHARING	= <boolean>	YES	If NO, Tecplot 360 will not allow data sharing. This may be necessary to use older add-ons that cannot handle shared data.
ALLOWOLDTEXTFORMATTING	= <boolean>	NO	If NO, allows Tecplot 360 to display text subscripts and superscripts created with older Tecplot 360 versions without automatically converting the text to the new formatting.

Parameter	Syntax	Default	Notes
USENAMESFORVARIABLEASSIGNMENTS	= <boolean>	FALSE	If set to TRUE then all references to dataset variables in layouts, stylesheets, and recorded macros will use variable names instead of variable offsets. The only exception is the use of "letter codes" in equations (i.e. X,Y,Z,U,V,W etc. See "Equation Syntax" in the Tecplot Users Manual) when recording a macro, or when data is created (before var names exist).

Example:

The following commands turn on datasharing:

```
$!COMPATIBILITY ALLOWDATASHARING=YES
```

$!CONTINUE

Syntax:

```
$!CONTINUE
```

Description:

Transfer control back to nearest $!LOOP or $!WHILE.

Example:

```
$!LOOP 10
  ...
  $!If |tvar| > 1
    $!CONTINUE
  $!Endif
  ...
$!ENDLOOP
```

$!CONTOURLABELS [Required-Control Option]

Description:

The different commands in the CONTOURLABELS compound function family are described separately in the following sections.

The CONTOURLABELS compound functions are:

$!CONTOURLABELS ADD
$!CONTOURLABELS DELETEALL

$!CONTOURLABELS ADD

Syntax:

```
$!CONTOURLABELS ADD
  [optional parameters]
```

Description

Add contour labels to your plot.

Optional Parameters

Parameter	Syntax	Default	Notes
CONTOURGROUP	= <integer>	1	Defines which contour group is changed.
ISALIGNED	= <boolean>	YES	If YES then align the contour label along the contour line; if NO, draw the label horizontally.
XYZPOS			
{			
X	= <dexp>	0.0	X-position for contour label.
Y	= <dexp>	0.0	Y-position for contour label.
Z	= <dexp>	0.0	Z-position for contour label (use Z only for 3D plots).
}			

Example:

The following commands add labels at (0.5, 0.25) and (0.73, 0.17) in a 2-D field plot.

```
$!CONTOURLABELS ADD
  CONTOURGROUP = 2
  XYZPOS
    {
      X = 0.5
      Y = 0.25
    }
```

```
$!CONTOURLABELS ADD
  XYZPOS
  {
    X = 0.73
    Y = 0.17
  }
```

$!CONTOURLABELS DELETEALL

Syntax:

```
$!CONTOURLABELS
  DELETEALL
  [optional parameters]
```

Description:

Delete all currently defined contour labels.

Optional Parameters

Parameter	Syntax	Default	Notes
CONTOURGROUP	= <integer>	1	Defines which contour group is changed.

Example:

```
$!CONTOURLABELS DELETEALL
  CONTOURGROUP = 3
```

$!CONTOURLEVELS [Required-Control Option]

Description:

The different commands in the CONTOURLEVELS compound function family are described separately in the following sections.

The CONTOURLEVELS compound functions are:

$!CONTOURLEVELS ADD
$!CONTOURLEVELS NEW
$!CONTOURLEVELS DELETENEAREST
$!CONTOURLEVELS DELETERANGE
$!CONTOURLEVELS RESET

$!CONTOURLEVELS RESETTONICE

$!CONTOURLEVELS ADD

Syntax:

```
$!CONTOURLEVELS ADD
  <contourlevelrawdata>
  [optional parameters]
```

Description:

Add a new set of contour levels to the existing set of contour levels.

Required Parameter

Parameter	Syntax	Default	Notes
<contourlevelrawdata>			Supply a list of contour levels to add.

Optional Parameters

Parameter	Syntax	Default	Notes
CONTOURGROUP	= <integer>	1	Defines which contour group is changed.

Example

Add contour levels 1.7, 3.4 and 2.9 to the plot:

```
$!CONTOURLEVELS ADD
  RAWDATA
  3
  1.7
  3.4
  2.9
```

$!CONTOURLEVELS DELETENEAREST

Syntax:

```
$!CONTOURLEVELS DELETENEAREST
  RANGEMIN = <dexp>
  [optional parameters]
```

Description:

Delete the contour level whose value is nearest the value supplied in the `RANGEMIN` parameter.

Required Parameter

Parameter	Syntax	Default	Notes
RANGEMIN	= <dexp>		Delete the contour level whose value is nearest to this value.

Optional Parameters

Parameter	Syntax	Default	Notes
CONTOURGROUP	= <integer>	1	Defines which contour group is changed.

Example

Delete the contour level whose value is nearest to 3.4:

```
$!CONTOURLEVELS DELETENEAREST
  RANGEMIN = 3.4
```

$!CONTOURLEVELS DELETERANGE

Syntax:

```
$!CONTOURLEVELS DELETERANGE
  RANGEMIN = <dexp>
  RANGEMAX = <dexp>
  [optional parameters]
```

Description:

Delete all contour levels between a minimum and maximum contour value (inclusive).

Required Parameters

Parameter	Syntax	Default	Notes
RANGEMIN	= <dexp>		Minimum contour level to delete.
RANGEMAX	= <dexp>		Maximum contour level to delete.

Optional Parameters

Parameter	Syntax	Default	Notes
CONTOURGROUP	= <integer>	1	Defines which contour group is changed.

Example

Delete all contour levels between 0.1 and 0.7:

```
$!CONTOURLEVELS DELETERANGE
  RANGEMIN = 0.1
  RANGEMAX = 0.7
```

$!CONTOURLEVELS NEW

Syntax:

```
$!CONTOURLEVELS NEW
  <contourlevelrawdata>
  [optional parameters]
```

Description:

Replace the current set of contour levels with a new set.

Required Parameter

Parameter	Syntax	Default	Notes
<contourlevelrawdata>			Supply a list of contour levels to add.

Optional Parameters

Parameter	Syntax	Default	Notes
CONTOURGROUP	= <integer>	1	Defines which contour group is changed.

Example

Replace the current set of contour levels with the levels 0.5, 0.75 and 1.0:

```
$!CONTOURLEVELS NEW
  RAWDATA
  3
  0.5
  0.75
```

```
  1.0
```

$!CONTOURLEVELS RESET

Syntax:

```
$!CONTOURLEVELS RESET
  NUMVALUES = <integer>
  [optional parameters]
```

Description:

Reset the contour levels to a set of evenly distributed values spanning the entire range of the currently selected contouring variable.

Required Parameter

Parameter	Syntax	Default	Notes
NUMVALUES	= <integer>		New number of contour levels.

Optional Parameters

Parameter	Syntax	Default	Notes
CONTOURGROUP	= <integer>	1	Defines which contour group is changed.

Example

Reset the contour levels to use 150 levels:

```
$!CONTOURLEVELS RESET
  NUMVALUES = 150
```

$!CONTOURLEVELS RESETTONICE

Syntax:

```
$!CONTOURLEVELS RESETTONICE
  APPROXNUMVALUES = <integer>
  [optional parameters]
```

Description:

Reset the contour levels to a set of evenly distributed, nice values spanning the entire range of the currently selected contouring variable, with a specified number of entries.

Required Parameter

Parameter	Syntax	Default	Notes
APPROXNUMVALUES	= <integer>		Approximate number of contour levels desired. Actual value may be different.

Optional Parameters

Parameter	Syntax	Default	Notes
CONTOURGROUP	= <integer>	1	Defines which contour group is changed.

Example

Reset the contour levels to use 10 levels:

```
$!CONTOURLEVELS RESETTONICE
  APPROXNUMVALUES = 10
```

$!CREATECIRCULARZONE

Syntax:

```
$!CREATECIRCULARZONE
  IMAX = <integer>
  JMAX = <integer>
  [optional parameters]
```

Description:

Create a circular (or cylindrical) IJ- or IJK-ordered zone.

Required Parameters

Parameter	Syntax	Default	Notes
IMAX	= <integer>		Radial direction.
JMAX	= <integer>		Circumferential direction, must be greater than 3.

Optional Parameters

Parameter	Syntax	Default	Notes
DATATYPE	= <datatype>	SINGLE	
KMAX	= <integer>	1	Bottom to top direction
RADIUS	= <dexp>	1	
X	= <dexp>	0	X-coordinate for center.
XVAR	= <varref>	Auto	Only needed when processing journal instructions. Use of variable names in this command is only allowed when a dataset is already present.
Y	= <dexp>	0	Y-coordinate for center.
YVAR	= <varref>	Auto	Only needed when processing journal instructions. Use of variable names in this command is only allowed when a dataset is already present.
Z1	= <dexp>	0	Z-minimum if a cylinder is created.
Z2	= <dexp>	1	Z-maximum if a cylinder is created.
ZVAR	= <varref>	Auto	Only needed when processing journal instructions. Use of variable names in this command is only allowed when a dataset is already present.

Examples

Example 1:

Create a circular 10 by 20 IJ-ordered zone centered at (5, 5) with a radius of 2:

```
$!CREATECIRCULARZONE
  IMAX     = 10
  JMAX     = 20
  X        = 5
  Y        = 5
  RADIUS   = 2
```

Example 2:

Create a cylindrical 5 by 6 by 8 IJK-ordered zone with the bottom centered at (4, 4, 0) and the top centered at (4, 4, 7) and a radius of 3:

```
$!CREATECIRCULARZONE
  IMAX     = 5
  JMAX     = 6
  KMAX     = 8
  X        = 4
  Y        = 4
  Z1       = 0
  Z2       = 7
  RADIUS   = 3
```

$!CREATECOLORMAP

Syntax:

```
$!CREATECOLORMAP
  NAME = <string>
  [optional parameters]
```

Description:

Defines a color map. Only the name is required; it must be a valid non-zero-length string.

Color map names are case-insensitive, although the case used when creating the color map is retained for display. Leading and trailing spaces are stripped.

If the named color map does not exist, it is created and initialized to SOURCECOLORMAP if provided; otherwise to "Small Rainbow." If the named colormap exists, and is not a built-in colormap, it will be overwritten by SOURCECOLORMAP, or by "Small Rainbow" if SOURCECOLORMAP is not provided.

Required Parameters

Parameter	Syntax	Default	Notes
NAME	= <string>		Name of the new color map. May not be the name of any existing color map, including built-in color maps.

Optional Parameters

Parameter	Syntax	Default	Notes
SOURCECOLORMAP	= <string>	"Small Rainbow"	If specified, the color map is initialized to the color map with this name. The source color map must exist, and may be the name of either a built-in or custom color map. May not be used with NUMCONTROLPOINTS.
NUMCONTROLPOINTS	= <integer>		Number of control points in color map. May not be used if SOURCECOLORMAP is used.
CONTROLPOINT	<<colormapcontrolpoints>>		Defines a control point. Multiple control points for a new color map may set in a CREATECOLORMAP command.

Example

Create a custom color map initialized from the built-in "Small Rainbow" color map. :

```
$!CREATECOLORMAP
  NAME="My Small Rainbow Colormap"
  SOURCECOLORMAP="Small Rainbow
```

$!CREATECONTOURLINEZONES

Syntax:

```
$!CREATECONTOURLINEZONES [group]
  [optional parameters]
```

Description:

Create zones from the currently-defined contour lines. One zone can be created from each contour level in that plot, or one zone for every polyline can be generated.

Optional Parameters

Parameter	Syntax	Default	Notes
AUTOSTRANDTRANSIENTDATA	= <boolean>	YES	If set to YES, time strands are automatically created for transient data in the new zone.

Parameter	Syntax	Default	Notes
CONTLINECREATEMODE	= ONEZONEPERCONTOURLEVEL or ONEZONEPERINDEPENDENTPOLYLINE		Select whether one zone per contour lever will be created or whether there will be a zone for each polyline.

Example

Create a new zone for each contour line on an existing contour plot.

```
$!CREATECONTOURLINEZONES
  CONTLINECREATEMODE = ONEZONEPERCONTOURLEVEL
```

$!CREATEFEBOUNDARY

Syntax:

```
$!CREATEFEBOUNDARY
  SOURCEZONE = <integer>
  [optional parameters]
```

Description:

Zone edges for finite element data cannot be turned on or off using the edge plot layer in Tecplot 360. You can, however, create a separate zone which is the boundary of a finite element zone. This new zone can then be turned on or off.

Required Parameter

Parameter	Syntax	Default	Notes
SOURCEZONE	= <integer>		Zone to extract the boundary from.

Optional Parameters

Parameter	Syntax	Default	Notes
AUTOSTRANDTRANSIENTDATA	= <boolean>	YES	If set to YES, time strands are automatically created for transient data in the new zone.

Parameter	Syntax	Default	Notes
REMOVEBLANKEDSURFACES	= <boolean>	NO	Set to YES if you want the resulting zone to include only the boundary adjacent to non-blanked cells.

Example

Create an FE-boundary zone from zone 3:

```
$!CREATEFEBOUNDARY
  SOURCEZONE = 3
```

$!CREATEFESURFACEFROMIORDERED

Syntax:

```
$!CREATEFESURFACEFROMIORDERED
  SOURCEZONES = <set>
  [optional parameters]
```

Description:

A FE-Surface zone can be generated from two or more I-Ordered zones. To get the best possible output, it is recommended that the source zones should have their nodes arranged in a similar manner so that the connecting lines between points are as straightforward as possible. For this reason, indices from source zones should increase in the same direction.

Required Parameter

Parameter	Syntax	Default	Notes
SOURCEZONES	= <set>		Zones whose points will be used to create the new surface.

Optional Parameters

Parameter	Syntax	Default	Notes
AUTOSTRANDTRANSIENTDATA	= <boolean>	YES	If set to YES, time strands are automatically created for transient data in the new zone.
CONNECTSTARTTOEND	= <boolean>	NO	YES allows for closed surfaces.

Example

Create an FE-Surface zone from zones 3 and 4:

```
$!CREATEFESURFACEFROMIORDERED
  SOURCEZONES = [3-4]
```

$!CREATELINEMAP

Syntax:

```
$!CREATELINEMAP
  [no parameters]
```

Description:

Create a new Line-mapping.

Example

```
$!CREATELINEMAP
```

$!CREATEMIRRORZONES

Syntax:

```
$!CREATEMIRRORZONES
  SOURCEZONES = <set>
  [optional parameters]
```

Description:

Create new zones that are mirror images of the source zones

Required Parameter

Parameter	Syntax	Default	Notes
SOURCEZONES	= <set>		Zone(s) to create mirror zone(s) from.

Optional Parameters

Parameter	Syntax	Default	Notes
MIRRORVAR	= <mirrorvar>	'X'	This variable in the new zone is multiplied by -1 after the zone is copied. (Mutually exclusive with MIRRORVARS.)
MIRRORVARS	= <varset>		Set of variables in the new zone to be multiplied by -1 after the zone is copied. (Mutually exclusive with MIRRORVAR.)

Example

Create a mirror of zones 2-4 across the Y-axis (that is, mirror the X-variable) in 2D frame mode:

```
$!CREATEMIRRORZONES
  SOURCEZONES = [2-4]
  MIRRORVAR   = 'X'
```

$!CREATENEWFRAME

Syntax:

```
$!CREATENEWFRAME
  [optional parameters]
```

Description:

Creates a new frame.

Optional Parameters

Parameter	Syntax	Default	Notes
HEIGHT	= <dexp>	8	Units are in inches.
WIDTH	= <dexp>	9	Units are in inches.
XYPOS	<<xy>>	X = 1.0 Y = 0.25	Units are in inches; relative to the top left edge of the paper

The default position and size of the initial frame created when Tecplot 360 starts up can be changed in the Tecplot 360 configuration file.

Example

The following example creates a 5- by 5-inch frame with the upper left hand corner of the frame positioned 2 inches from the left edge of the paper and 1 inch from the top:

```
$!CREATENEWFRAME
  XYPOS
  {
    X = 2
    Y = 1
  }
  WIDTH = 5
  HEIGHT = 5
```

$!CREATERECTANGULARZONE

Syntax:

```
$!CREATERECTANGULARZONE
  [optional parameters]
```

Description:

Create a rectangular zone. If no data set exists when this command is executed, a data set is created with variables X, Y (and Z, if KMax > 1). If a data set exists prior to this command, the non-coordinate variables for the zone created are initialized to zero.

Optional Parameters

Parameter	Syntax	Default	Notes
IMAX	= <integer>	1	I-dimension.
JMAX	= <integer>	1	J-dimension.
KMAX	= <integer>	1	K-dimension.
X1	= <dexp>	0	X-minimum.
Y1	= <dexp>	0	Y-minimum.
Z1	= <dexp>	0	Z-minimum.
X2	= <dexp>	1	X-maximum.
Y2	= <dexp>	1	Y-maximum.
Z2	= <dexp>	1	Z-maximum.
XVAR	= <varref>	Auto	Only needed when processing journal instructions. Use of variable names in this command is only allowed when a dataset is already present.

Parameter	Syntax	Default	Notes
YVAR	= <varref>	Auto	Only needed when processing journal instructions. Use of variable names in this command is only allowed when a dataset is already present.
ZVAR	= <varref>	Auto	Only needed when processing journal instructions. Use of variable names in this command is only allowed when a dataset is already present.
DATATYPE	= <datatype>	SINGLE	

Example:

Create a rectangular IJ-ordered zone dimensioned 20 by 30 where X ranges from 0 to 3 and Y from 3 to 9:

```
$!CREATERECTANGULARZONE
  IMAX      = 20
  JMAX      = 30
  X1        = 0
  Y1        = 3
  X2        = 3
  Y2        = 9
```

$!CREATESIMPLEZONE

Syntax:

```
$!CREATESIMPLEZONE
  <xyrawdata>
  [optional parameters]
```

Description:

Create a new zone by specifying only a list of data. Rows represent the individual data points and columns the variables at each point.

You cannot specify more variables than the dataset already contains.

Required Parameter

Parameter	Syntax	Default	Notes
RAWDATA			May contain row and column counts or just a row count. If only a row count is specified, the number of columns is assumed to be 2, for XY data. See Raw Data for details.

Optional Parameter

Parameter	Syntax	Default	Notes
DATATYPE	= <datatype>	SINGLE	

Example 1:

Create a simple XY-zone that has the XY-pairs (1, 0), (2, 1), (3, 7) and (5 9):

```
$!CREATESIMPLEZONE
  RAWDATA
  4
  1 0
  2 1
  3 7
  5 9
```

Example 2:

Create a simple XYZ zone.

```
$!CREATESIMPLEZONE
  RAWDATA
  2 3               # Two rows of data with three columns each
  1 2 3
  4 5 6
```

Example 3:

The following is invalid, since the second `CREATESIMPLEZONE` is trying to set more variables than exist in the data set (assuming it was initially empty and created with the first `CREATESIMPLEZONE`).

```
$!CREATESIMPLEZONE
  RAWDATA
  2 3
  1 2 3
  4 5 6
```

```
$!CREATESIMPLEZONE
  RAWDATA
  2 6               # Doesn't work since there are only 3 variables
  1 2 3 4 5 6
  7 8 9 8 7 4
```

$!CREATESPHERICALZONE

Syntax:

```
$!CREATECIRCULARZONE
  IMAX = <integer>
  JMAX = <integer>
  [optional parameters]
```

Description:

Create a spherical IJK-ordered zone.

Required Parameters

Parameter	Syntax	Default	Notes
IMax	= <integer>		Psi direction.
JMax	= <integer>		Theta direction.

Optional Parameters

Parameter	Syntax	Default	Notes
DATATYPE	= <datatype>	SINGLE	
RADIUS	= <dexp>	1	
X	= <dexp>	0	X-coordinate for center.
XVAR	= <varref>	Auto	Only needed when processing journal instructions. Use of variable names in this command is only allowed when a dataset is already present.
Y	= <dexp>	0	Y-coordinate for center.
YVAR	= <varref>	Auto	Only needed when processing journal instructions. Use of variable names in this command is only allowed when a dataset is already present.
Z	= <dexp>	0	Z-coordinate for center.

Parameter	Syntax	Default	Notes
ZVAR	= <varref>	Auto	Only needed when processing journal instructions. Use of variable names in this command is only allowed when a dataset is already present.

Examples

Create a spherical 10 by 20 IJ-ordered zone centered at (5, 5) with a radius of 2:

```
$!CREATESPHERICALZONE
  IMax     = 10
  JMax     = 20
  X        = 5
  Y        = 5
  RADIUS   = 2
```

$!DATASETUP

Syntax:

```
$!DATASETUP
  [Optional Parameters]
```

Description:

A SetValue command that sets miscellaneous parameters related to data.

Optional Parameters

Parameter	Syntax	Default	Notes
COMMANDLINE			
{			This option allows you to auto-strand data files in Tecplot 360. This can be set to NO or commented-out of the configuration file (tecplot.cfg) to retain the Tecplot 10 compatibility
AUTOSTRANDDATAFILES	= <boolean>	YES	
}			

Parameter	Syntax	Default	Notes
SCRATCHDATAFIELDTYPE	= <datatype>		Set the data type for scratch arrays used for geometries line segments and other lines. The default is SINGLE for Windows and DOUBLE for other platforms. This parameter can only be used in the Tecplot 360 configuration file.
SUBZONEMINMAXCACHEMEMORYFRACTION	= <double>	0.90	working with non-szl data, Tecplot 360 will sub-divide large zones into smaller "subzones" to speed up various operations (like probing). Temporary storage can be employed to speed up the creation of the subzones. This temporary storage can be quite large thus we avoid using it if it causes swapping to occur. The engine tries to determine how much RAM is available before doing this operation and then multiplies that by the SUBZONEMINMAXCACHEMEMORYFRACTION. If the amount of RAM left over is larger than the amount of RAM needed for this temporary space it will be created. All machines and environments are different. Setting this smaller will make it less likely to employ the temporary storage but will avoid potential disk swapping.

Example:

Change the arguments used to Preplot ASCII files so only zones 1, 2, and 3 are processed:

```
$!DATASETUP
  PREPLOTARGS = "-zonelist 1:3"
```

$!DEFAULTGEOM

Syntax:

```
$!DEFAULTGEOM
  [optional parameters]
```

Description:

A SetValue command that sets the attributes for the default geometry. When a geometry is created

interactively, its color, line thickness, and so forth, are preset based on the default geometry. This command is usually used only in the Tecplot 360 configuration file.

Optional Parameters

Parameter	Syntax	Default	Notes
ANCHORPOS	<<xyz>>		
ARROWHEADANGLE	<op> <dexp>		
ARROWHEADATTACHMENT	<arrowheadattachment>		
ARROWHEADSIZE	<op> <dexp>		
ARROWHEADSTYLE	<arrowheadstyle>		
ATTACHTOZONE	= <boolean>		
COLOR	= <color>		
DATATYPE	= <fielddatatype>		
DRAWORDER	= <draworder>	AFTERDATA	
FILLCOLOR	= <color>		
ISFILLED	= <boolean>		
LINEPATTERN	= <linepattern>		
LINETHICKNESS	<op> <dexp>		
MACROFUNCTIONCOMMAND	= <string>		Set the macro command to execute when you hover over the geometry and press Ctrl-right-click.
MAINTAINASPECTRATIO	= <boolean>	YES	
NUMELLIPSEPTS	<op> <integer>		
PATTERNLENGTH	<op> <dexp>		
PIXELASPECTRATIO	= <double>	0	A value of 0 allows Tecplot 360 to select the aspect ratio. Use only if your circles or squares appear distorted due to the aspect ratio of your monitor.

Parameter	Syntax	Default	Notes
POSITIONCOORDSYS	= <coordsys>		
SCOPE	= <scope>		
ZONE	= <integer>		

Example:

Make the default geometry line thickness 0.2 percent:

```
$!DEFAULTGEOM
  LINETHICKNESS = 0.2
```

$!DEFAULTTEXT

Syntax:

```
$!DEFAULTTEXT
  [optional parameters]
```

Description:

A SetValue command that sets the attributes for the default text. When text is added to a plot interactively, its font, color, size, and so forth, are based on the default text. This command is used only in the Tecplot 360 configuration file.

Optional Parameters

Parameter	Syntax	Default	Notes
ANCHOR	= <textanchor>		
ANCHORPOS	<<xy>>		
ANGLE	<op> <dexp>		
ATTACHTOZONE	= <boolean>		
BOX	<<textbox>>		
CLIPPING	= <clipping>		
COLOR	= <color>		
LINESPACING	<dexp>		

Parameter	Syntax	Default	Notes
MACROFUNCTIONCOMMAND	= <string>		Set the macro command to execute when you hover over the geometry and press Ctrl-right-click.
POSITIONCOORDSYS	= <coordsys>		Currently not set up to add 3D Text
SCOPE	= <scope>		
TEXTSHAPE	<<textshape>>		
TEXTTYPE	= <texttype>		Specifies the Text Type. Supported values are LATEX and REGULAR. REGULAR: Produces normal text annotations. LATEX: Instructs Tecplot to pass the text string through the LaTeX toolchain (see the User's Manual for more information) to produce images which are then placed in the plot at the specified location. Only Text, AnchorPos, Height, and SizeUnits are valid parameters for LaTeX type.
ZONE	<op> < integer>		

Example:

Make the default text font Times bold with a character height of 14 points:

```
$!DEFAULTTEXT
  TEXTSHAPE
  {
    FONTFAMILY = "Times"
    ISBOLD = YES
    ISITALIC = NO
    SIZEUNITS = POINT
    HEIGHT = 14
  }
```

$!DELAY

Syntax:

```
$!DELAY <integer>
  [no parameters]
```

Description:

Delay Tecplot 360 Execution for the specified number of seconds.

Example

Delay Tecplot 360 for 3 seconds:

```
$!DELAY 3
```

$!DELETEAUXDATA

Syntax:

```
$!DELETEAUXDATA
  AUXDATALOCATION = [zone/var/dataset/frame/linemap|page]
  [Optional Parameters]
```

Description:

Delete Auxiliary Data in the form of name/value pairs from zones, frames or datasets.

Required Parameters

Parameter	Syntax	Default	Notes
AUXDATALOCATION	= [zone/var/dataset/frame/linemap/page]		

Optional Parameters

Parameter	Syntax	Default	Notes
NAME	= <string>		
NUM	= <integer>		
VAR	= <varref>		
ZONE	= <integer>		Only required if AUXDATALOCATION = zone

Example:

Delete the selected Auxiliary Data from Zone 2.:

```
$!DELETEAUXDATA
  AUXDATALOCATION = zone
  ZONE = 2
  NAME = VARIABLE DATA
```

$!DELETECOLORMAP

Syntax:

```
$!DELETECOLORMAP <string>
  [no parameters]
```

Description:

Deletes the specified custom color map, which must exist.

$!DELETELINEMAPS

Syntax:

```
$!DELETEMAPS <set>
  [no parameters]
```

Description:

Delete one or more line mappings. If <set> is omitted then all line mappings are deleted.

Example

Delete Line-mappings 2, 3, 4 and 8:

```
$!DELETELINEMAPS [2-4,8]
```

$!DELETEVARS

Syntax:

```
$!DELETEVARS <varset>
 [no parameters]
```

Description:

Delete one or more variables.

Example

Delete variables 4 and 10:

```
$!DELETEVARS [4,10]
```

$!DELETEZONES

Syntax:

```
$!DELETEZONES <set>
  [no parameters]
```

Description:

Delete one or more zones.

Example

Delete zones 3, 7, 8, 9 and 11:

```
$!DELETEZONES [3,7-9,11]
```

$!DOUBLEBUFFER [Required-Control Option]

Description:

The different commands in the DOUBLEBUFFER compound function family are described separately in the following sections.

The DOUBLEBUFFER compound functions are:

```
$!DOUBLEBUFFER OFF
$!DOUBLEBUFFER ON
$!DOUBLEBUFFER SWAP
```

$!DOUBLEBUFFER OFF

Syntax:

```
$!DOUBLEBUFFER OFF
  [no parameters]
```

Description:

Turn off double buffering; use this command once at the end of a sequence of using the double buffer.

Example:

See $!DOUBLEBUFFER SWAP.

$!DOUBLEBUFFER ON

Syntax:

```
$!DOUBLEBUFFER ON
  [no parameters]
```

Description:

Turn on double buffering; use this command once at the beginning of a sequence of using the double buffer. While double buffering is turned on all drawing is sent to the back buffer.

Example:

See $!DOUBLEBUFFER SWAP.

$!DOUBLEBUFFER SWAP

Syntax:

```
$!DOUBLEBUFFER SWAP
  [no parameters]
```

Description:

Swap the back buffer to the front. In other words, copy the image in the back buffer to the front.

Example:

The following example uses the double buffer to show the rotation of a 3-D object:

```
$!DOUBLEBUFFER ON
  $!LOOP 10
    $!ROTATE3DVIEW X
      ANGLE = 5
    $!REDRAW
    $!DOUBLEBUFFER SWAP
  $!ENDLOOP
$!DOUBLEBUFFER OFF
```

$!DRAWGRAPHICS

Syntax:

```
$!DRAWGRAPHICS <boolean>
  [no parameters]
```

Description:

Turn on or off all graphics drawing. Turning off all graphics during preliminary portions of a macro file can greatly increase the efficiency of the macro.

Example:

Turn off all graphics drawing:

```
$!DRAWGRAPHICS NO
```

$!DUPLICATELINEMAP

Syntax:

```
$!DUPLICATELINEMAP
  SOURCEMAP = <integer>
  DESTINATIONMAP = <integer>
  [no optional parameters]
```

Description:

Copy attributes from an existing line mapping to another.

Required Parameters

Parameter	Syntax	Default	Notes
DESTINATIONMAP	= <integer>		The destination can either be the number of an existing map or 1 greater than the current number of maps. If you choose the latter, a new line mapping will be created.
SOURCEMAP	= <integer>		Line mapping from which to copy.

Example:

Copy attributes of Line-mapping 3 to Line-mapping 7:

```
$!DUPLICATELINEMAP
  SOURCEMAP      = 3
  DESTINATIONMAP = 7
```

$!DUPLICATEZONES

Syntax:

```
$!DUPLICATEZONES
  SOURCEZONE = <integer>
  SOURCEZONES = <set>
  [optional parameters]
```

Description:

Make a copy of an existing zone or zones. You can use index ranges to create new zone(s) from a subset of the source zone(s). You may also specify a destination zone to overwrite existing zone(s).

Initially, the duplicate zone shares all variables with the source zone. To branch some or all variables in the destination zone(s), use `$!BRANCHFIELDDATAVAR`

Required Parameter

Either `SOURCEZONE` or `SOURCEZONES` must be specified, but not both.

Parameters	Syntax	Default	Notes
SOURCEZONE	= <integer>		Zone to duplicate (the source zone)
SOURCEZONES	= <set>		Zones to duplicate (the source zones)

Optional Parameters

Parameters	Syntax	Default	Notes
DESTINATIONZONE	= <integer>		If specified, the existing zone into which the copy is made, overwriting the zone. May not be a source zone. If not specified, a new zone is created. If multiple source zones are specified, the first source zone overwrites this zone. Source zones beyond the first are duplicated to subsequent zones, which may be existing or new.
IRANGE			See notes Range Parameters, for `$!ALTERDATA` action command.
{			
MIN	= <integer>	1	
MAX	= <integer>	0	
SKIP	= <integer>	1	
}			
JRANGE			See notes Range Parameters `$!ALTERDATA` action command.
{			
MIN	= <integer>	1	
MAX	= <integer>	0	
SKIP	= <integer>	1	
}			
KRANGE			See notes Range Parameters for `$!ALTERDATA` action command.
{			
MIN	= <integer>	1	
MAX	= <integer>	0	
SKIP	= <integer>	1	
}			

Examples

Example 1:

Make a complete copy of zone 2 to a new zone:

```
$!DUPLICATEZONES
  SOURCEZONE = 2
```

Example 2:

Copy zone 3 and 4 to zone 5 and 6, using only the I-index range from 2 to 7 from the source zones:

```
$!DUPLICATEZONES
  SOURCEZONES = [3-4]
  DESTINATIONZONE = 5
  IRANGE
  {
    MIN = 2
    MAX = 7
  }
```

E-I

$!ELSE

Syntax:

```
$!ELSE
  [no parameters]
```

Description:

Conditionally handle macro commands. Used when an `$!IF` statement is `FALSE`.

Example

```
$!VARSET |C| = 2
$!IF |C| == 5
  $!CREATENEWFRAME
    XYPOS
    {
      X = 2.5
      Y = 1.5
    }
```

```
    WIDTH = 4
    HEIGHT = 4
$!ELSE
 $!CREATENEWFRAME
   XYPOS
   {
     X = 3
     Y = 2
   }
   WIDTH = 3
   HEIGHT = 3
$!ENDIF
```

$!ELSEIF

Syntax:

```
$!ELSEIF <conditionalexp>
```

Description:

Conditionally handle macro commands. Used to create multiple options for statements should an `$!IF` statement be `FALSE`.

Example

```
$!VARSET |A| = 2
$!IF |A| < 5
 $!CREATENEWFRAME
   XYPOS
   {
     X = 1
     Y = 1
   }
   WIDTH = 3
   HEIGHT = 3
$!ELSEIF |A| > 5
 $!CREATENEWFRAME
   XYPOS
   {
     X = 2
     Y = 1
   }
   WIDTH = 5
   HEIGHT = 5
```

```
$!ELSE
 $!CREATENEWFRAME
   XYPOS
   {
     X = 3
     Y = 3
   }
   WIDTH = 9
   HEIGHT = 9
$!ENDIF
```

$!EXPORT

Syntax:

```
$!EXPORT
  [no parameters]
```

Description:

Export an image file from Tecplot 360. See the $!EXPORTSETUP command for details on setting up the exported image type. The $!EXPORT command is not valid for animation formats.

Optional Parameters

Parameter	Syntax	Default	Notes
EXPORTREGION	= <bitdumpregion>		If supplied this will override what is set in $!EXPORTSETUP

Example

```
$!EXPORTSETUP EXPORTFORMAT = PNG
$!EXPORT
```

$!EXPORTCANCEL

Syntax:

```
$!EXPORTCANCEL
  [no parameters]
```

Description:

Cancel out of the current export animation sequence. The animation file being generated is removed.

Example

```
$!EXPORTCANCEL
```

$!EXPORTFINISH

Syntax:

```
$!EXPORTFINISH
  [no parameters]
```

Description:

Signals the completion of an animation sequence and causes the animation file to be created. You must call `$!EXPORTSTART` prior to using `$!EXPORTFINISH`. This command is only valid for animation formats. You may use the `|EXPORTISRECORDING|` intrinsic variable to make sure that an animation sequence has been initiated.

Example

```
$!EXPORTSETUP
  EXPORTFNAME="rotate.avi"
  EXPORTFORMAT=AVI
$!EXPORTSTART
$!LOOP 5
  $!ROTATE3DVIEW X
    ANGLE=5
  $!EXPORTNEXTFRAME
$!ENDLOOP
$!IF "|EXPORTISRECORDING|" =="YES"
  $!EXPORTFINISH
$!ENDIF
```

$!EXPORTNEXTFRAME

Syntax:

```
$!EXPORTNEXTFRAME
  [no parameters]
```

Description:

Records the next frame of an animation. You must call $!EXPORTSTART prior to calling $!EXPORTNEXTFRAME. This command is only valid for animation formats. You may use the |EXPORTISRECORDING| intrinsic variable to make sure that an animation sequence has been initiated.)

Example

```
$!EXPORTSETUP
  EXPORTFNAME="rotate.avi"
  EXPORTFORMAT=AVI
  $!EXPORTSTART
  $!LOOP 5
    $!ROTATE3DVIEW X
      ANGLE=5
    $!EXPORTNEXTFRAME
  $!ENDLOOP
$!EXPORTFINISH
```

$!EXPORTSETUP

Syntax:

```
$!EXPORTSETUP
  [optional parameters]
```

Description:

A SetValue command that sets the attributes for exporting image files from Tecplot 360. Exporting is usually intended as a means to transfer images from Tecplot 360 to be imported by other applications. See $!PRINTSETUP and $!PRINT for generating output intended for printers and plotters.

Optional Parameters

Parameter	Syntax	Default	Notes
ANIMATIONSPEED	= <double>	10	Sets the animation speed in frames per second.
AVIFORMATOPTIONS	= <string>	-q:v 5	See online documentation for FFmpeg for explanation of options.
CONVERTTO256COLORS	= <boolean>	NO	Used for TIFF, BMP, and PNG formats.
EXPORTFNAME	= <string>		

Parameter	Syntax	Default	Notes
EXPORTFORMAT	= <exportformat>	WINDOWS METAFILE	
EXPORTREGION	= <bitdumpregion>	ALLFRAMES	
FFMPEGQSCALE	= <integer>		This option is no longer used. Use MPEGFORMATOPTIONS or WMVFORMATOPTIONS instead.
FLASHCOMPRESSIONTYPE	= <compressiontype>	SMALLESTSIZE	
FLASHIMAGETYPE	= <imagetype>	LOSSLESS	
IMAGEWIDTH	<op> <integer>	= 512	
JPEGENCODING	STANDARD or PROGRESSIVE	STANDARD	
MPEGFORMATOPTIONS	<string>	"-c:v libx264 -profile:v high -crf 20 -pix_fmt yuv420p"	See online documentation for FFmpeg for explanation of options.
PRINTRENDERTYPE	= <printrendertype>	VECTOR	
QUALITY	= <integer>	75	Range is from 1-100
SUNRASTERFORMAT	= <sunrasterformat>	STANDARD	Only applies if EXPORTFORMAT is SUNRASTER
SUPERSAMPLEFACTOR	= <integer>	3	The factor used in antialiasing while reducing the size of an exported image. A larger size can improve the quality of the image, but slows performance.
TIFFBYTEORDER	= <tiffbyteorder>	INTEL	
USEMULTIPLECOLORTABLES	= <boolean>	NO	Applies to Raster Metafile only.

Parameter	Syntax	Default	Notes
USESUPERSAMPLEANTIALIASING	= <boolean>	NO	
WMVFORMATOPTIONS	= <string>	"-qscale 4"	See online documentation for FFmpeg for explanation of options.

Example:

Set up Tecplot 360 to export a Raster Metafile image to the file `movie.rm`:

```
$!EXPORTSETUP
  EXPORTFNAME = "movie.rm"
  EXPORTFORMAT = RASTERMETAFILE
```

$!EXPORTSTART

Syntax:

```
$!EXPORTSTART
 [optional parameter]
```

Description:

Signals the start of an animation sequence and records the first frame of the animation. This command is only valid for animation formats.

Optional Parameter

Parameter	Syntax	Default	Notes
EXPORTREGION	= <bitdumpregion>		

Example

```
$!EXPORTSETUP
  EXPORTFNAME="rotate.avi"
  EXPORTFORMAT=AVI
  EXPORTREGION = CURRENTFRAME
$!EXPORTSTART
$!LOOP 5
  $!ROTATE3DVIEW X
    ANGLE=5
  $!EXPORTNEXTFRAME
$!ENDLOOP
```

```
$!EXPORTFINISH
```

$!EXTENDEDCOMMAND

Syntax:

```
$!EXTENDEDCOMMAND
  COMMANDPROCESSORID = <string>
  COMMAND = <string>
  [optional parameters]
```

Description:

Send a command to an add-on. The add-on registers the name of a function that will be called when an $!EXTENDEDCOMMAND is processed. Tecplot 360 knows which registered function to call based on the COMMANDPROCESSORID string.

Required Parameters

Parameter	Syntax	Default	Notes
COMMANDPROCESSORID	= <string>		String that identifies the add-on. This must match the published ID string for the add-on.
COMMAND	= <string>		The command to be sent to the add-on.

Optional Parameters

Parameter	Syntax	Default	Notes
<extendedcommandrawdata>		NULL	If the RAWDATA section is supplied then each line of the RAWDATA section is appended to the COMMAND string. A leading new line character is appended first, and each line in the RAWDATA section will also be terminated with a new line (except for the last line).

Example:

Send the command GO to the add-on that has registered a command processor with an COMMANDPROCESSORID of XPROC:

```
$!EXTENDEDCOMMAND
  COMMANDPROCESSORID = "XPROC"
  COMMAND = "GO"
```

$!EXTRACTCONNECTEDREGIONS

Syntax:

```
$!EXTRACTCONNECTEDREGIONS
  SOURCEZONES = <set>
  [optional parameters]
```

Description:

Extract a separate zone from each isolated region of connected cells in one or more finite element zones.

Required Parameters

Parameter	Syntax	Notes
SOURCEZONES	<set>	Set of zones from which to extract connected regions.

Optional Parameters

Parameter	Syntax	Default	Notes
XVAR	= <varref>	Auto	If present then assign the x-axis variable. If omitted the current X-Axis variable assignment is used.
YVAR	= <varref>	Auto	If present then assign the y-axis variable. If omitted the current Y-Axis variable assignment is used.
ZVAR	= <varref>	Auto	If present then assign the z-axis variable. If omitted the current Z-Axis variable assignment is used.
AUTOSTRANDTRANSIENTDATA	= <boolean>	True	If set to FALSE then all newly created zones will be static. If set to TRUE then a single new strand id will be assigned for all newly created zones.

$!EXTRACTFROMGEOM

Syntax:

```
$!EXTRACTFROMGEOM
 [optional parameters]
```

Description:

Extract data from a 2- or 3D field plot. The locations at which to extract the data come from a polyline geometry that must be picked prior to issuing this command.

Optional Parameters

Parameters	Syntax	Default	Notes
EXTRACTLINEPOINTSONLY	= <boolean>	NO	If NO, must include NUMPTS.
EXTRACTTOFILE	= <boolean>	NO	If NO, a zone is created. If YES, must include FNAME.
FNAME	= <string>		File name for extracted file. Required if EXTRACTTOFILE is YES.
INCLUDEDISTANCEVAR	= <boolean>	NO	If YES, then Tecplot 360 includes an extra variable in the result which is the distance along the line of points extracted and EXTRACTTOFILE must also be YES.
NUMPTS	= <integer>		Required if EXTRACTLINEPOINTSONLY is NO.

Example

Extract 20 points from along the currently picked geometry. Send the result to a file called `extract.dat`:

```
$!EXTRACTFROMGEOM
  NUMPTS = 20
  EXTRACTTOFILE = YES
  FNAME = "extract.dat"
```

$!EXTRACTFROMPOLYLINE

Syntax:

```
$!EXTRACTFROMPOLYLINE
 [optional parameters]
 <xyrawdata>
```

Description:

Extract data from a 2- or 3D field plot. The locations of where to extract the data from come from a supplied polyline in the form of <xyrawdata> .

The coordinate system used is determined by the following rules:

- In 2D, the coordinates are the grid (same as used on the X and Y axis). Note that Z must still be supplied; use zero.
- In 3D with EXTRACTTHROUGHVOLUME set to TRUE, the coordinates are in the world coordinate system (same as used on the X, Y, and Z axes).
- In 3D with EXTRACTTHROUGHVOLUME set to FALSE, the coordinates are in the eye coordinate system (same as used for grid mode geometries). (As in 2D, Z must still be supplied; use zero.) Each point is projected down to the surface closest to the viewer to determine the final world coordinate location for the extraction.

Optional Parameters

Parameters	Syntax	Default	Notes
EXTRACTLINEPOINTSONLY	= <boolean>	NO	If NO, must include NUMPTS.
EXTRACTTHROUGHVOLUME	= <boolean>	NO	If YES, data is extracted from XYZ-coordinates in the polyline. If NO, data is extracted from the surface.
EXTRACTTOFILE	= <boolean>	NO	If NO, a zone is created. If YES, you must include FNAME.
FNAME	= <string>		File name for extracted file. Required if EXTRACTTOFILE is YES.
INCLUDEDISTANCEVAR	= <boolean>	NO	If YES, Tecplot 360 includes an extra variable in the result which is the distance along the line of points extracted and EXTRACTOFILE must also be YES.
NUMPTS	= <integer>		Required if EXTRACTLINEPOINTSONLY is NO.

Example

Extract 10 points from specific locations in a field plot. Create a zone with the extracted data:

```
$!EXTRACTFROMPOLYLINE
  EXTRACTLINEPOINTSONLY = YES
  RAWDATA
10
0 0 0
1 2 0
2 4 0
3 2 0
3 4 0
4 4 0
4 5 0
```

```
4 6 0
5 7 0
6 9 0
```

$!EXTRACTISOSURFACES

Syntax:

```
$!EXTRACTISOSURFACES
  [optional parameters]
```

Description:

Extracts the currently defined iso-surfaces or the iso-surfaces of the specified groups to zones. By default the resulting zones will be automatically assigned strands and are given the solution time of the current time step from which they are extracted.

Optional Parameters

Parameter	Syntax	Default	Notes
AUTOSTRANDTRANSIENTDATA	= <boolean>	TRUE	If set to TRUE, time strands are automatically created for transient data in the new zone.
GROUP	= <integer>		The Group and Groups commands are mutually exclusive and are used to specify which isosurface groups to extract to zones. If neither are specified, all active groups are extracted. Groups cannot be defined if Group is defined and vice versa.
GROUPS	= <set>	[1-8]	
EXTRACTMODE	= <extractmode>	SINGLEZONE	EXTRACTMODE has two valid options for this function: SingleZone and OneZonePerConnectedRegion. SingleZone: Extracts a single iso-surface to a single zone. OneZonePerConnectedRegion: Extracts the connected regions of a single isosurface to individual zones. Connected regions are those regions that are connected to one another. Note: OneZonePerSourceZone is a valid EXTRACTMODE value, however it is not supported by $!EXTRACTISOSURFACES

Example

```
$!EXTRACTISOSURFACES
  GROUPS = [4,6]
```

$!EXTRACTSLICES

Syntax:

```
$!EXTRACTSLICES
  [Optional Parameters]
```

Description:

Extracts the currently defined slices or the slices of the specified groups to zones. By default the resulting zones will be automatically assigned strands and are given the solution time of the current time step from which they are extracted. If the start and end position for the slice style is active then the zones are extracted in position order from the start position to the end position with the primary value slice, if active, in its position order.

Optional Parameter

Parameter	Syntax	Default	Notes
GROUP	= <integer>		The Group and Groups commands are mutually exclusive and are used to specify which isosurface groups to extract to zones. If neither are specified, all active groups are extracted. Groups cannot be defined if Group is defined and vice versa.
GROUPS	= <set>	[1-8]	

Parameter	Syntax	Default	Notes
EXTRACTMODE	= <extractmode>	SINGLEZONE	EXTRACTMODE has two valid options for this function: SINGLEZONE and ONEZONEPERCONNECTEDREGION. Extract Mode specifies how each slice is extracted into single or multiple zones. SINGLEZONE: Extracts each slice to a single zone. ONEZONEPERCONNECTEDREGION: Extracts each slice to its connected regions. Connected regions are defined by zone boundaries and areas of disconnected meshes. Note that setting EXTRACTMODE to ONEZONEPERCONNECTEDREGION and RESULTING1DZONETYPE to IORDEREDIFPOSSIBLE guarantees that any extracted linear zones will be I-ordered zones. Extracted I-ordered zones will be ordered according to the right-hand rule-the nodes will progress from 1 to IMax clockwise about the axis normal to the plane of extraction.

Parameter	Syntax	Default	Notes
EXTRACTION STRANDID ASSIGNMENT	= <extractionstrandidassignment>	AUTO	StrandID assignment on extracted slice. where: DONOTASSIGNSTRANDIDS: do not assign strand ID's ONESTRANDPERGROUP: Use the same strand id for all items produced by a given slice group. ONESTRANDPERSUBEXTRACTION: Use a different strand for every extracted item produced by a slice group. A given slice group can produce multiple items if the EXTRACTMODE is ONEZONEPERCONNECTEDREGION or when extracting using the start/end positions. AUTO: If extracting over all time (i.e. TRANSIENTOPERATIONMODE is set to ALLSOLUTIONTIMES) and the EXTRACTMODE is set to SINGLEZONE then use ONESTRANDPERSUBEXTRACTION otherwise use ONESTRANDPERGROUP
RESULTING 1DZONETYPE	= <resulting1dzonetype>	IORDERED IF POSSIBLE	RESULTING1DZONETYPE has two valid options for this function: IORDEREDIFPOSSIBLE and FELINESEGMENT. IORDEREDIFPOSSIBLE creates an I-ordered zone for each slice zone that is a single connected region. FELINESEGMENT creates all extracted linear zones as FELineSeg zones. If this is set to FELINESEGMENT, then all extracted zones will be FE line segment zones. If set to IORDEREDIFPOSSIBLE (the default), then any connected (contiguous) extracted zone will be an I-ordered zone, while zones that consist of multiple unconnected regions will be FE line segment zones.
TRANSIENT OPERATIONMODE	= <transientoperationmode>	ALL SOLUTION TIMES	Extracts for a single solution time when set to SINGLESOLUTIONTIME. Extracts for all solution times when set to ALLSOLUTIONTIMES.

Example:

```
$!GLOBALCONTOUR VAR = 4
$!SLICEATTRIBUTES 3 ENDPOSITION {X = 1}
$!SLICEATTRIBUTES 3 STARTPOSITION {X = 6}
$!SLICEATTRIBUTES 3 NUMINTERMEDIATESLICES = 6
$!SLICEATTRIBUTES 3 SHOWSTARTENDSLICE = YES
$!SLICEATTRIBUTES 3 SHOWINTERMEDIATESLICES = YES
$!SLICEATTRIBUTES 3 SHOWGROUP = YES
$!REDRAW
$!EXTRACTSLICES
  GROUP = 3
```

$!EXTRACTSLICETOZONES

Syntax:

```
$!EXTRACTSLICETOZONES
  ORIGIN <<xyz>>
  [optional parameters]
```

Description:

Create a new zone from a plane through existing data. This function allows you to extract slices through time and to define value blanking constraints to limit the region of the slice. This function is independent of the current plot style and similarly will not change the style of your plot. If you want to extract slices that are defined by $!SLICEATTRIBUTES, $!EXTRACTSLICES.

Required Parameters

Parameter	Syntax	Notes
ORIGIN	<<xyz>>	Origin of the slice.

Optional Parameters

Parameter	Syntax	Default	Notes
AUTOSTRAND TRANSIENTDATA	= <boolean>	YES	If set to YES, time strands are automatically created for transient data in the new zone.
COPYCELL CENTEREDVALUES	= <boolean>	YES	If YES, then the values of any variables that are cell-centered in all zones used to construct the slice are copied as cell-centered variables to the slice. Otherwise, all variables will be interpolated to the slice nodes.

Parameter	Syntax	Default	Notes
EXTRACTMODE	= <extractmode>	SINGLEZONE	EXTRACTMODE has three options: SingleZone, OneZonePerConnectedRegion, and OneZonePerSourceZone. SingleZone: Extracts a single slice to a single zone. OneZonePerConnectedRegion: Extracts the connected regions of a single slice to individual zones. Connected regions are those regions that are connected to one another. OneZonePerSourceZone: Extracts one zone for each source zone a single slice intersects.
NORMAL	<<xyz>>		Required if SLICESURFACE is ARBITRARY. Normal of the slice.
RESULTING1DZONETYPE	= <resulting1dzonetype>	FELINESEGMENT	RESULTING1DZONETYPE has two valid options for this function: IORDEREDIFPOSSIBLE and FELINESEGMENT. IORDEREDIFPOSSIBLE creates an I-ordered zone for each slice zone that is a single connected region. FELINESEGMENT creates all extracted linear zones as FELineSeg zones. For more information, see the Description of this parameter for macro `$!EXTRACTSLICES`.
SLICESOURCE	= <slicesource>	VOLUMEZONES	Select the source of the slice. Has four options: VOLUMEZONES - extracts the slice from a volume zone or zones. SURFACEZONES - extracts the slice from the surface of the surface zones. LINEARZONES - creates a slice from a linear zone. If in 2D, Tecplot switches to 3D, extracts, and then returns to 2D. SURFACEOFVOLUMEZONES - extracts a slice from the exposed surface of the volume zone.

Parameter	Syntax	Default	Notes
SLICESURFACE	= < slicesurface>	ARBITRARY	SLICESURFACE has four options: XPLANES, YPLANES, ZPLANES or ARBITRARY. If set to ARBITRARY then the NORMAL must be supplied. IPLANES, JPLANES or KPLANES are not allowed.
SOLUTIONTIME	= <double>	Current Solution Time	If TRANSIENTOPERATIONMODE is set to SINGLESOLUTIONTIME then this can be used to set the solution time for the extraction. If not present then the current solution time is used.
SOURCEFIELDMAPS	= <set>	See Notes	Set of fieldmaps to use for the slice. These fieldmaps will obey the blanking conditions assigned by VALUEBLANKING The default setting is existing active fieldmaps less fieldmaps that are marked to not show slices (See the "Show Slices" column in the "Volume" tab in the Zone Style dialog)
SOURCEFIELDMAPS IGNORINGBLANKING	= <set>	See Notes	Set of fieldmaps that will ignore value blanking during slice extraction. The default setting is to obey the "Use Value Blanking" setting for fieldmaps (See the "Use Value Blanking" column in the "Effects" tab in the Zone Style Dialog)"

Parameter	Syntax	Default	Notes
SURFACEGENERATION METHOD	= <surfacegenerationmethod>	AUTO	**Auto**: Selects one of the surface generation algorithms best suited for the zones participating in the slice generation. "All Polygons" is used if one or more of the participating zones is polytope, otherwise slices use "Allow Quads". **AllowQuads**: Produces quads or triangles, and the resulting surface more closely resembles the shape of the volume cells from the source zone. **AllTriangles**: An advanced algorithm that can handle complex saddle issues and guarantees that there will be no holes in the final surface. As the surface is composed entirely of triangles, it can be delivered more efficiently to the graphics hardware. **AllPolygons**: Similar to the "All triangles" method except that all interior faces generated as a result of triangulation that are not part of the original mesh are eliminated. This preserves the original mesh of the source zones on the resulting slice.
TRANSIENT OPERATIONMODE	= <transientoperationmode>	ALLSOLUTIONTIMES	Extracts for a single solution time when set to SINGLESOLUTIONTIME. Extracts for all solution times when set to ALLSOLUTIONTIMES.
VALUEBLANKING	See Notes	Current ValueBlanking options	If present then use provided value blanking. If omitted then use the current value blanking state. If you want to guarantee that blanking will not affect the slice then you must at least set the main value blanking include setting to FALSE. i.e.: `VALUEBLANKING {INCLUDE = FALSE}` See the VALUE section under `$!BLANKING` for full list of options.
XVAR	= <varref>	Auto	If present then assign the x-axis variable. If omitted the current X-Axis variable assignment is used.

Parameter	Syntax	Default	Notes
YVAR	= <varref>	Auto	If present then assign the y-axis variable. If omitted the current Y-Axis variable assignment is used.
ZVAR	= <varref>	Auto	If present then assign the z-axis variable. If omitted the current Z-Axis variable assignment is used.

Example

Create a slice zone at Y=1.0:

```
$!EXTRACTSLICETOZONES
  SLICESOURCE = VOLUMEZONES
  ORIGIN {X=0.0 Y=1.0 Z=0.0}
  NORMAL {X=0.5 Y=0.75 Z=0.25}
```

$!EXTRACTSTREAMTRACES

Syntax:

```
$!EXTRACTSTREAMTRACES
  [optional parameters]
```

Description:

Extract the currently defined streamtraces to zones. By default the resulting zones will be automatically assigned strands and are given the solution time the current time step from which they are extracted. If directed Tecplot will concatenate all streamtraces of the same type together.

Optional Parameters

Parameter	Syntax	Default	Notes
AUTOSTRANDTRANSIENTDATA	= <boolean>	TRUE	If set to TRUE, time strands are automatically created for transient data in the new zone.

Parameter	Syntax	Default	Notes
CONCATENATE	= <boolean>	FALSE	If CONCATONATE is set to TRUE then a single FE zone will be created (FE-Line Segment for streamlines or FE-Quad for rods or ribbons). If CONCATONATE is set to FALSE and timing is not turned on to draw dashes then a single I-Ordered zone (for streamlines) or IJ-Ordered zone (for ribbons and rods) will be created for each streamtrace.

Example

Create a single zone out of all common streamzones:

```
$!EXTRACTSTREAMTRACES
  CONCATENATE = TRUE
```

$!FIELDLAYERS

Syntax:

```
$!FIELDLAYERS
  [optional parameters]
```

Description:

A SetValue command that turns field plot layers on or off, or sets the 2D draw order.

Optional Parameters

Parameter	Syntax	Default	Notes
SHOWCONTOUR	= <boolean>	NO	
SHOWEDGE	= <boolean>	YES	
SHOWISOSURFACES	= <boolean>	NO	
SHOWMESH	= <boolean>	NO	
SHOWSCATTER	= <boolean>	NO	
SHOWSHADE	= <boolean>	YES	
SHOWSLICES	= <boolean>	NO	
SHOWVECTOR	= <boolean>	NO	Vector variables must be defined. See $!GLOBALTWODVECTOR or $!GLOBALTHREEDVECTOR.

Parameter	Syntax	Default	Notes
TWODDRAWORDER	= <twoddraworder>	BYLAYER	
USELIGHTINGEFFECT	= <boolean>	YES	
USETRANSLUCENCY	= <boolean>	YES	

Example:

Turn on the scatter layer:

```
$!FIELDLAYERS
  SHOWSCATTER = YES
```

$!FIELDMAP

Syntax:

```
$!FIELDMAP [<set>]
  [optional parameters]
```

Description:

A SetValue command that assigns zone attributes for field plots. The <set> parameter immediately following the $!FIELDMAP command is optional. If <set> is omitted then the assignment is applied to all zones. Otherwise the assignment is applied only to the zones specified in <set>.

Optional Parameters

Parameter	Syntax	Default	Notes
CONTOUR			
{			
COLOR	= <color>	BLACK	
CONTOURTYPE	= <contourtype>	FLOOD	
FLOODCOLORING	= <contourcoloring>	GROUP1	
LINECONTOURGROUP	= <integer>	1	

Parameter	Syntax	Default	Notes
LINEPATTERN	= <linepattern>	SOLID	
LINETHICKNESS	<op> <dexp>	= 0.1	
PATTERNLENGTH	<op> <dexp>	= 2	
SHOW	= <boolean>	YES	
USELIGHTINGEFFECT	= <boolean>	YES	
}			
EDGELAYER			
{			
COLOR	= <color>	BLACK	
EDGETYPE	= <edgetype>	BORDERSAND CREASES	
IBORDER	= <borderlocation>	BOTH	For IJ-, IK-, and IJK-ordered zones.
JBORDER	= <borderlocation>	BOTH	For IJ-, IK-, and IJK-ordered zones.
KBORDER	= <borderlocation>	BOTH	For IJ-, IK-, and IJK-ordered zones.
LINETHICKNESS	= <dexp>	0.1	
SHOW	= <boolean>	YES	
}			
EFFECTS			
{			
LIGHTINGEFFECT	= <lightingeffect>	GOURAUD	
SURFACETRANSLUCENCY	<translucency>	50	SURFACETRANSLUCENCY range is one to 99.
USETRANSLUCENCY	= <boolean>	YES	
USEVALUEBLANKING	= <boolean>	YES	Set to YES to include value blanking in the specified zones

Parameter	Syntax	Default	Notes
USECLIPPLANES	<set>	[1-6]	Use clipping planes specified to clip zones specified in set for FIELDMAP, or all zones if none specified. Possible values include [ ] (none), or any combination of the numbers 1 - 6, enclosed in brackets.
}			
MESH			
{			
COLOR	= <color>	BLACK	
LINEPATTERN	= < linepattern>	SOLID	
LINETHICKNESS	<op> <dexp>	= 0.1	
MESHTYPE	= <meshtype>	OVERLAY	
PATTERNLENGTH	<op> <dexp>	= 2	
SHOW	= <boolean>	YES	
}			
POINTS			
{			
IJKSKIP	<<ijk>>	1, J = 1, K = 1	Limits the number of vectors or scatter symbols drawn.
POINTSTOPLOT	<pointstoplot>	SURFACENODES	
}			
SCATTER			
{			
COLOR	= <color>	BLACK	
FILLCOLOR	= <color>	WHITE	
FILLMODE	= <fillmode>	NONE	
FRAMESIZE	<op> <dexp>	2.5	Size of symbols when SIZEBYVARIABLE is NO.
LINETHICKNESS	<op> <dexp>	0.1	
SHOW	= <boolean>	YES	

Parameter	Syntax	Default	Notes
SIZEBYVARIABLE	= <boolean>	NO	Scatter sizing variable must be defined before this can be set to YES. See the `$!GLOBALSCATTER` command.
SYMBOLSHAPE	<<symbolshape>>	See Notes	ISASCII = NO, GEOMSHAPE = SQUARE
}			
SHADE			
{			
COLOR	= <color>	WHITE	
SHOW	= <boolean>	YES	
USELIGHTINGEFFECT	= <boolean>	YES	
}			
SURFACES			
{			
IRANGE	<<indexrange>>	See Notes	MIN=1, Max=0, Skip=1
JRANGE	<<indexrange>>	See Notes	MIN=1, Max=1, Skip=1
KRANGE	<<indexrange>>	See Notes	MIN=1, Max=1, Skip=1
SURFACESTOPLOT	= <surfacestoplot>	NONE	
}			
VECTOR			
{			
ARROWHEADSTYLE	<arrowheadstyle>	PLAIN	
COLOR	= <color>	BLACK	
ISTANGENT	= <boolean>	NO	
LINEPATTERN	= <linepattern>	SOLID	
LINETHICKNESS	= <dexp>	0.1	
PATTERNLENGTH	= <dexp>	2	

Parameter	Syntax	Default	Notes
SHOW	= <boolean>	YES	
VECTORTYPE	= <vectortype>	TAILATPOINT	
}			
VOLUMEMODE			VOLUMEMODE applies to volume zones, with the exception that POINTSTOPLOT also applies to finite-element surface zones.
{			
VOLUMEOBJECTSTOPLOT	<<volumeobjectstoplot>>	SHOWISOSURFACES=YES, SHOWSLICES=YES, SHOWSTREAMTRACES=YES	
}			
GROUP	= <integer>	1	Assign a group number to the supplied set of zones.

Examples

Example 1:

Change the contour plot type to flood for zones 1-12:

```
$!FIELDMAP [1-12]
  CONTOUR
  {
    CONTOURTYPE = FLOOD
  }
```

Example 2:

Change the mesh color to red for all zones (default mesh color is black):

```
$!FIELDMAP
  MESH
  {
    COLOR = RED
  }
```

$!FILECONFIG

Syntax:

```
$!FILECONFIG
  [optional parameters]
```

Description:

A SetValue command that sets file path information in Tecplot 360.

Optional Parameters

Parameter	Syntax	Default	Notes
ADDZONESTOEXISTINGSTRANDS	= <boolean>	NO	If YES, Tecplot 360 will add the zones from the appended data to any existing strands in the dataset. If NO, Tecplot 360 will append the strands from the new data to any existing strands in the dataset.
ASSIGNSTRANDIDS	= <boolean>	NO	If YES, Tecplot 360 will assign strand ID's to zones (if time is supplied for the zones but not strand ID's). If NO, Tecplot 360 will not associate these zones with any strands.
DATAFILEVARLOADMODE	= <varloadmode>	BYNAME	Set the default loading mode for variables. BYNAME loads variables based on their name. If set to BYNAME, then VARNAMELIST must be supplied as well except when appending. BYPOSITION loads variables based on their position in the file. To get Tecplot Version 7.0 behavior, use BYPOSITION.
DOAUTOFNAMEEXTENSION	= <boolean>		
DOAUTOFNAMEEXTENSIONWARNING	= <boolean>		If YES a warning is displayed when attempting to save with an extension other than the default extension.
FNAMEFILTER			
{			
COLORMAPFILE	= <string>		Default extension for color map files.
EQUATIONFILE	= <string>		Default extension for equation files.
IMPORTIMAGEFILE	= <string>		Default extension for image files.

Parameter	Syntax	Default	Notes
INPUTDATAFILE	= <string>		Default extension for Tecplot 360 input data files.
INPUTLAYOUTFILE	= <string>		Default extension for loading layout files.
MACROFILE	= <string>		Default extension for macro files.
OUTPUTASCIIDATAFILE	= <string>		Default extension for ASCII output data files.
OUTPUTBINARYDATAFILE	= <string>		Default extension for binary output data files.
OUTPUTLAYOUTFILE	= <string>		Default extension for saving linked layout files.
OUTPUTLAYOUTPACKAGEFILE	= <string>		Default extension for saving layout package files.
STYLEFILE	= <string>		Default extension for style files.
}			
LAYOUTCONFIG			
{			
INCLUDEDATA	= <boolean>	NO	If YES, layout packages are the default format for layouts
INCLUDEPREVIEW	= <boolean>	NO	If YES, preview images are saved with layout packages.
USERELATIVEPATHS	= <boolean>	YES	If YES, files will be referenced using relative paths in layout files.
}			
LOADONDEMAND			
{			
ALLOW	= <boolean>	YES	If YES, Tecplot 360 will use its load-on-demand features for loading and unloading variables.

Parameter	Syntax	Default	Notes
DATASTORESTRATEGY	= <dataloadstrategy>	AUTO	Set the data store strategy for load-on-demand. If set to AUTO, Tecplot 360 will use store large allocations in the temporary directory and use memory mapped I/O to read and write to the regions when possible otherwise it will use the memory heap (usually this provides better performance for large data). If set to HEAP Tecplot 360 will not use the temporary directory for large allocations (this option is usually slower when working with large data).
UNLOADSTRATEGY	= <unloadstrategy>	AUTO	Set the unload strategy for load-on-demand. If set to AUTO Tecplot 360 will unload unused variables when the amount needed RAM begins to reach the maximum amount of RAM. If set to NEVERUNLOAD Tecplot 360 will load variables on demand but will never attempt to unload them even if it is running low on memory. If set to MINIMIZEMEMORYUSE Tecplot 360 will aggressively unload variables as soon as they are not needed regardless of the amount of memory available or in use.
}			
TEMPFILEPATH	= <string>		Set the directory where you want Tecplot 360 to store its temporary files.

File Name Filters

Valid characters are upper or lowercase A-Z, and 0-9. Each filter should be preceded by (*), or it will not filter properly. On Windows® operating systems, to allow more than one extension, separate them with a semicolon (;). On Linux and Mac platforms, multiple extensions will not filter correctly unless they follow the standard shell filter format.

Windows Example:

This example filters all four extensions when opening a layout file.

```
$!FILECONFIG FNAMEFILTER {INPUTLAYOUTFILE = "*.wsf;*.dwr;*.lay;*.lpk"}
```

Windows Example:

This example filters both extensions when writing a layout file. The default extension is .wsf because it

is the first extension presented in the list.

```
$!FILECONFIG FNAMEFILTER {OUPUTLAYOUTFILE = ".wsf;*.lay"}
```

Linux Example:

This example filters `.aek`, `.plt`, and more.

```
$!FILECONFIG FNAMEFILTER {INPUTDATAFILE = "*.[ap][el][kt]"}
```

Linux Example:

This example filters `.dat`, `.cam`, and more. The default extension is `.dat` because D and T are the first letters presented within the brackets.

```
$!FILECONFIG FNAMEFILTER {OUTPUTASCIIDATAFILE = "*.[dc]a[tm]"}
```

Example:

Set the directory where Tecplot 360 stores temporary files to be `/usr/tmp`:

```
$!FILECONFIG
DATAFILEVARLOADMODE = BYPOSITION
  TEMPFILEPATH = "/usr/tmp"
  LAYOUTCONFIG {USERELATIVEPATHS = YES}
  FNAMEFILTER
     {
     INPUTDATAFILE = "*.[pd][la]t"
     COLORMAPFILE = "*.clr"
     }
```

$!FONTADJUST

Syntax:

```
$!FONTADJUST
  [optional parameters]
```

Description:

A SetValue command that sets character spacing and sizing for fonts in Tecplot 360. These parameters

rarely change.

Optional Parameters

Parameter	Syntax	Default	Notes
BOLDFACTOR	<op> <double>		Thickness of bold characters relative to normal.
INTERCHARSPACING	<op> <integer>		Increase or decrease inter-character spacing. Units are in pixels on the screen.
STROKEFONTLINETHICKNESS	<op> <double>		Thickness (in frame units) of lines used to draw stroke fonts.
SUBSUPFRACTION	<op> <double>		Size of subscript and superscript characters relative to the font height.

Example:

Make superscript and subscript characters 1/3 the font height:

```
$!FONTADJUST
  SUBSUPFRACTION = 0.333
```

$!FOURIERTRANSFORM

Syntax:

```
$!FOURIERTRANSFORM
  INDEPENDENTVAR = <varref>
  WINDOWFUNCTION = <windowfunction>
  DEPENDENTVARS = <varset>
  SOURCEZONES = <set>
  INCLUDECONJUGATES = <boolean>
  OBEYSOURCEZONEBLANKING = <boolean>
  [no optional parameters]
```

Description:

Performs a Fourier transform for each dependent variable for each source zone. A new zone containing the resulting frequency, amplitude, and phase variables (three for each dependent variable) is created for each source zone. If the independent variable is non-uniform, the resulting frequency is a modification of the original data (see discussion below for the INDEPENDENTVAR and OBEYSOURCEZONEBLANKING parameters). Resulting zones are assigned new time strands using the same groupings as the source zones if they belong to time stands; otherwise, the resulting zones are

static.

Fourier transform result zones are named "Fourier Transform" followed by a mixture of text indicating the source zone, independent variable, and window function used. Similarly, the three variables created are given the names "Frequency", "Amplitude", and "Phase" followed by the dependent variable used. Newly-created zones are assigned passive variables for all variables that previously existed in the data set, and all previously-existing zones are assigned passive variables for all new variables created by the Fourier transform.

Required Parameters

Parameter	Syntax	Notes
INDEPENDENTVAR	= <varref>	Independent variable used for the frequency domain. If the spacing is non-uniform, this variable is used in conjunction with each dependent variable for interpolation to create a uniform frequency domain for the transform.
WINDOWFUNCTION	= <windowfunction>	Window function or tapering function applied to the dependent variables before performing the transform but after performing the non-uniform interpolation. For details, see en.wikipedia.org/wiki/Window_function
DEPENDENTVARS	= <varset>	Set of dependent variables on which to perform a Fourier transform. The variables must not be the same as the independent variable. This must not be NULL or an empty set and must be a subset of the variables in the source zones.
SOURCEZONES	= <set>	Set of source zones containing the dependent variables on which to perform the Fourier transform. This must not be NULL or an empty set and must be a subset of the data set's available zones.
INCLUDECONJUGATES	= <boolean>	For purely real numbers, the Fourier transform output satisfies the Hermitian redundancy where out[i] is the conjugate of out[n-i]. If this parameter is YES, the conjugates are included, otherwise they are not, and approximately half of the output values, n/2+1, are computed.

Parameter	Syntax	Notes
OBEYSOURCEZONEBLANKING	= <boolean>	If value blanking is active and this value is YES, value blanking is applied to the data values of both the independent and dependent variables before the data is interpolated for non-uniformity. If data values eliminated from the independent or dependent variables cause the data to be non-uniform, the values are interpolated appropriately. Additionally, all blanked data values up to the first non-blanked data value and all blanked data values after the last non-blanked data value are ignored, providing a mechanism to constrain the domain over which the Fourier transformation is performed.

Optional Parameters

Parameter	Syntax	Default	Notes
REPLACEMATCHINGRESULTZONES	= <boolean>	NO	If YES, any existing result zones for the specified combination of source zone, independent variable, and window function are re-used. If NO, new result zones are created.
REPLACEMATCHINGRESULTVARIABLES	= <boolean>	NO	If YES, any existing result variables for the specified dependent variables are re-used. If NO, new result variables are created.

Example

Perform a Fourier transform on variables 2 through 11 of zones 1 through 10 obeying source zone blanking, applying the Hann window function and excluding conjugates from the output.

```
$!FOURIERTRANSFORM
  INDEPENDENTVAR = 1
  WINDOWFUNCTION = HANN
  DEPENDENTVARS = [2-11]
  SOURCEZONES = [1-10]
  INCLUDECONJUGATES = NO
  OBEYSOURCEZONEBLANKING = YES
```

$!FRAMECONTROL [Required Control Option]

Description:

The different commands in the FRAMECONTROL compound function family are described separately in the following sections. When working with the FRAMECONTROL commands, it may help to realize that a command containing "Activate" changes the active frame; a command containing "MoveTo" changes the frame drawing order.

The FRAMECONTROL compound functions following are:

$!FRAMECONTROL ActivateTop
$!FRAMECONTROL ActivateNext
$!FRAMECONTROL ActivatePrevious
$!FRAMECONTROL ActivateAtPosition
$!FRAMECONTROL ActivateByName
$!FRAMECONTROL ActivateByNumber
$!FRAMECONTROL MoveToTopActive
$!FRAMECONTROL MoveToTopByName
$!FRAMECONTROL MoveToTopByNumber
$!FRAMECONTROL MoveToBottomActive
$!FRAMECONTROL MoveToBottomByName
$!FRAMECONTROL MoveToBottomByNumber
$!FRAMECONTROL DeleteActive
$!FRAMECONTROL FitAllToPaper

$!FRAMECONTROL ACTIVATETOP

Syntax:

```
$!FRAMECONTROL ACTIVATETOP
  [no parameters]
```

Description:

Changes the active frame to the frame that is topmost in the frame drawing order.

Example

```
$!FRAMECONTROL ACTIVATETOP
```

$!FRAMECONTROL ACTIVATENEXT

Syntax:

```
$!FRAMECONTROL ACTIVATENEXT
```

```
  [no parameters]
```

Description:

Changes the active frame to the next one up in the frame drawing order, or to the bottom frame if the active frame is at the top.

Example

```
$!FRAMECONTROL ACTIVATENEXT
```

$!FRAMECONTROL ACTIVATEPREVIOUS

Syntax:

```
$!FRAMECONTROL ACTIVATEPREVIOUS
  [no parameters]
```

Description:

Changes the active frame to the next one down in the frame drawing order, or to the top frame if the active frame is at the bottom.

Example

```
$!FRAMECONTROL ACTIVATEPREVIOUS
```

$!FRAMECONTROL ACTIVATEATPOSITION

Syntax:

```
$!FRAMECONTROL ACTIVATEATPOSITION
  X = <dexp>
  Y = <dexp>
```

Description:

Activates the topmost frame at the specified position. X and Y are in paper coordinates.

Required Parameters

Parameter	Syntax	Notes
X	= <dexp>	Specify X-coordinate of position.
Y	= <dexp>	Specify Y-coordinate of position.

Example

```
$!FRAMECONTROL ACTIVATEATPOSITION X=0 Y=0
```

$!FRAMECONTROL ACTIVATEBYNAME

Syntax:

```
$!FRAMECONTROL ACTIVATEBYNAME
  NAME = <string>
```

Description:

Changes the active frame to the specified frame. If no frame name is given, this will activate the bottom frame.

Required Parameter

Parameter	Syntax	Notes
NAME	= <string>	Specify name of the frame to activate.

Example

Activate a frame named Topography.

```
$!FRAMECONTROL ACTIVATEBYNAME
  NAME="Topography"
```

$!FRAMECONTROL ACTIVATEBYNUMBER

Syntax:

```
$!FRAMECONTROL ACTIVATEBYNUMBER
  FRAME = <integer>
```

Description:

Changes the active frame to the specified frame.

Required Parameter

Parameter	Syntax	Notes
FRAME	= <integer>	Specify number of the frame to activate.

Example

Activate Frame 4.

```
$!FRAMECONTROL ACTIVATEBYNUMBER
  FRAME=4
```

$!FRAMECONTROL MOVETOTOPACTIVE

Syntax:

```
$!FRAMECONTROL MOVETOTOPACTIVE
  [no parameters]
```

Description:

Moves the active frame to the top of the drawing order.

Example

```
$!FRAMECONTROL MOVETOTOPACTIVE
```

$!FRAMECONTROL MOVETOTOPBYNAME

Syntax:

```
$!FRAMECONTROL MOVETOTOPBYNAME
  NAME = <string>
```

Description:

Moves the frame specified by name to the top of the frame drawing order.

Required Parameter

Parameter	Syntax	Notes
NAME	= <string>	Specify name of the frame to move to the top of the drawing order.

Example

Moves the frame named Topography to the top of the drawing order.

```
$!FRAMECONTROL MOVETOTOPBYNAME
  NAME="TOPOGRAPHY"
```

$!FRAMECONTROL MOVETOTOPBYNUMBER

Syntax:

```
$!FRAMECONTROL MOVETOTOPBYNUMBER
  FRAME = <integer>
```

Description:

Moves the frame specified by number to the top of the frame drawing order. If no frame number is specified, this command will move the bottom frame to the top of the frame drawing order.

Required Parameter

Parameter	Syntax	Notes
FRAME	= <integer>	Specify number of the frame to move to the top of the drawing order.

Example

Moves frame 4 to the top of the drawing order.

```
$!FRAMECONTROL MOVETOTOPBYNUMBER
  FRAME=4
```

$!FRAMECONTROL MOVETOBOTTOMACTIVE

Syntax:

```
$!FRAMECONTROL MOVETOBOTTOMACTIVE
```

```
  [no parameters]
```

Description:

Moves the active frame to the bottom of the frame drawing order.

Example

```
$!FRAMECONTROL MOVETOBOTTOMACTIVE
```

$!FRAMECONTROL MOVETOBOTTOMBYNAME

Syntax:

```
$!FRAMECONTROL MOVETOBOTTOMBYNAME
  NAME = <string>
```

Description:

Moves the frame specified by name to the bottom of the frame drawing order.

Required Parameter

Parameter	Syntax	Notes
NAME	= <string>	Specify name of the frame to move to the bottom.

Example

```
$!FRAMECONTROL MOVETOBOTTOMBYNAME
  NAME = "my frame name"
```

$!FRAMECONTROL MOVETOBOTTOMBYNUMBER

Syntax:

```
$!FRAMECONTROL MOVETOBOTTOMBYNUMBER
  FRAME = <integer>
```

Description:

Moves the frame specified by number to the bottom of the frame drawing order.

Required Parameter

Parameter	Syntax	Notes
FRAME	= <integer>	Specify number of the frame to move to the bottom.

Example

Move Frame 003 to the bottom.

```
$!FRAMECONTROL OVETOBOTTOMBYNUMBER
  FRAME=003
```

$!FRAMECONTROL DELETEACTIVE

Syntax:

```
$!FRAMECONTROL DELETEACTIVE
  [no parameters]
```

Description:

Delete the active frame.

Example

```
$!FRAMECONTROL DELETEACTIVE
```

$!FRAMECONTROL FITALLTOPAPER

Syntax:

```
$!FRAMECONTROL FITALLTOPAPER
  [no parameters]
```

Description:

Resize all frames so that they fit inside the hardclip limits of the paper.

Example

```
$!FRAMECONTROL FITALLTOPAPER
```

$!FRAMELAYOUT

Syntax:

```
$!FRAMELAYOUT
  [optional parameters]
```

Description:

A SetValue command that sets the position, border, and background attributes for the active frame. Use the $!FRAMECONTROL action command to push and pop frames if you want to change the settings for a frame other than the active frame.

Optional Parameters

Parameter	Syntax	Default	Notes
BACKGROUNDCOLOR	= <color>	WHITE	Only applies if ISTRANSPARENT = NO.
BORDERTHICKNESS	<op> <dexp>	= 0.1	Value is in Y-frame units.
FRAMESIZEPOSUNITS	= <framecoordmode>		Whether frame sizes and positions are in absolute units (paper) or relative screen units from 0 to 1 (workspace)
HEADERCOLOR	= <color>	RED	Only applies if SHOWHEADER = YES.
HEADERFONTAMILY	= <string>	"Helvetica"	
HEIGHT	<op> <dexp>	= 8	Value is in inches.
HEADERFONTISBOLD	= <boolean>	YES	
HEADERFONTISITALIC	= <boolean>	NO	
ISTRANSPARENT	= <boolean>	NO	
SHOWBORDER	= <boolean>	YES	
SHOWHEADER	= <boolean>	NO	
WIDTH	<op> <dexp>	= 9	Value is in inches.
XYPOS	<<xy>>	X=1, Y=0.25	Position of upper left corner of the frame in inches from left and top edge of the paper.

Example:

Place the active frame in the upper left corner of the paper (offset 0.5 inches from the top and left edges), make the frame dimensions 3 by 4 inches, and turn off the frame border:

```
$!FRAMELAYOUT
  SHOWBORDER = NO
  XYPOS
  {
    X = 0.5
    Y = 0.5
  }
  WIDTH = 3
  HEIGHT = 4
```

$!FRAMENAME

Syntax:

```
$!FRAMENAME = <string>
 [no parameters]
```

Description:

Set the name for the active frame.

Example:

```
$!FRAMENAME = "Pressure Contours for well 33"
```

$!FRAMESETUP

Syntax:

```
$!FRAMESETUP
  [optional parameters]
```

Description:

A SetValue command that sets parameters used to preset dynamic frame attributes when a frame is initialized.

Optional Parameters

Parameter	Syntax	Default	Notes
ALIGNINGCONTOURLABELS	= <boolean>	YES	If YES, the next interactively placed contour label is aligned to the contour line.
ASSIGNSEQUENCEDZONECOLORS	= <boolean>	NO	If set to YES, the values set via $!FIELDMAP [nnn] MESHLAYER {COLOR = <color>} are used for default zone mesh colors (similarly for Edge colors).
FITINITIALFRAMETOWORKSPACE	= <boolean>	YES	If set to NO, the new layout is shown as in V10 with the entire paper fit to the work area. If set to YES, the new layout is shown with the active frame fit to the work area. This command changes the behavior of Tecplot 360 as it first appears during a session and as it appears after a new layout command. It has no effect on the current plot, but it can be used in a macro to set the value for future new plots. It is typically found in the tecplot.cfg file.
INITIAL3DFITTOSURFACES	= <boolean>	YES	If YES, initial 3D plot shows a view equivalent to using "$!View FitSurfaces" or the View→Fit Surface menu item, and ignores the "$!FrameSetup Initial3DScale" parameter. If NO, initial 3D plot is equivalent to using the value of "$!FrameSetup Initial3DScale" in the View→Translate/Magnify dialog or using the "$!View Scale=<double>" command. This value defaults to YES for 360 and NO for Focus.
INITIAL3DSCALE	<op> <dexp>	= 1.1	Initial scale for 3D plots.
NUMSTREAMRAKEPOINTS	<op> <integer>	= 10	Number of points to place along streamtrace rakes.
NUMSTREAMSURFACEPOINTS	<op> <integer>	= 1	Number of points to place when seeding streamtraces on surfaces.
RODRIBBONDEFLEN	<op> <dexp>	= 0.06	Default width (in frame units) of a streamtrace or ribbon
SHOWAPPENDEDZONES	= <boolean>	YES	If set to NO, zones added via an append data operation will be turned off initially.

Parameter	Syntax	Default	Notes
SHOWNONWALLBOUNDARYZONES	= <boolean>	NO	If set to YES, any new zone that has a non-wall BOUNDARYCONDITION defined will be turned off initially.
VECTDEFLEN	<op> <dexp>	= 0.06	When a vector plot is drawn for the first time the vector magnitude is adjusted so the longest vector is VECTDEFLEN units long. VECDEFLEN is in frame units.
VECTMINLEN	<op> <dexp>	= 0.0005	Minimum length in centimeters. Vectors shorter than this length are not drawn.
VECTORDEFAULTSPACINGCOUNT	= <integer>	40	evenly spaced vectors are first drawn or reset, the vector spacing is set so that about VectorDefaultSpacingCount vectors are drawn in each direction across the frame. (See also $!RESETVECTORSPACING.)
USECOMMONSORTSTACK	= <boolean>	YES	

Example:

Make the default length for the longest vector five percent:

```
$!FRAMESETUP
  VECTDEFLEN = 5
```

$!GETAUXDATA

Syntax:

```
$!GETAUXDATA  <macrovar>
  AUXDATALOCATION = <auxlocation>
  NAME = <string>
  [optional parameters]
```

Description:

Retrieve Auxiliary Data in the form of name/value pairs from the given location and store it in the macro variable.

Required Parameters

Parameter	Syntax	Notes
AUXDATALOCATION	= <auxlocation>	
NAME	= <string>	Name of existing auxiliary data

Optional Parameters

Parameter	Syntax	Default	Notes
MAP	= <integer>		Only required if AUXDATALOCATION = linemap
VAR	= <varref>		Only required if AUXDATALOCATION = var
ZONE	= <integer>		Only required if AUXDATALOCATION = zone

Example:

Get the Auxiliary Data from Zone 2, and store it in the macro variable `|ABC|`:

```
$!GETAUXDATA |ABC|
  AUXDATALOCATION = zone
  NAME = 'ABC.Aux.Data'
  ZONE = 2
```

$!GETCONNECTIVITYREFCOUNT

Syntax:

```
$!GETCONNECTIVITYREFCOUNT <macrovar>
  ZONE = <integer>
  [no optional parameters]
```

Description:

Fetch the count of how many zones share connectivity with the specified zone. Count includes specified zone.

Required Parameters

Parameter	Syntax	Default	Notes
ZONE	= <integer>		

Example:

Fetch the connectivity count from Zone 2, and store it in the macro variable |ABC|. If zones 2, 5 and 6 share connectivity, |ABC| = 3:

```
$!GETCONNECTIVITYREFCOUNT |ABC|
  ZONE = 2
```

$!GETCURFRAMENAME

Syntax:

```
$!GETCURFRAMENAME <macrovar>
  [no parameters]
```

Description:

Query Tecplot 360 for the name of the active frame. The <macrovar> represents the macro variable to receive the results.

Example:

Put the name of the active frame into the macro variable |CFRAME|.

```
$!GETCURFRAMENAME |CFRAME|
```

$!GETFIELDVALUE

Syntax:

```
$!GETFIELDVALUE <macrovar>
  ZONE   = <integer>
  VAR    = <varref>
  INDEX  = <integer>
```

Description:

Fetch the field value (data set value) at the specified point index and assign the value to <macrovar>. If the zone referenced is IJ- or IJK-ordered, then the point index is calculated by treating the 2- or 3Dimensional array as a 1-D array.

Required Parameters

Parameter	Syntax	Notes
INDEX	= <integer>	
VAR	= <varref>	
ZONE	= <integer>	

Example:

A data set contains 2 zones and 3 variables. Zone 2 is dimensioned 5 by 3. Fetch the value from variable 3 at IJ-location (2, 2), and store it in the macro variable `|ABC|`:

```
$!GETFIELDVALUE |ABC|
  ZONE = 2
  VAR   = 3
  INDEX = 7
```

Note: `INDEX` was calculated using:

```
INDEX = I + (J-1)*|MAXI| + (K-1) * |MAXI| * |MAXJ|
      = 5*(2-1)+2
      = 7
```

$!GETFIELDVALUEREFCOUNT

Syntax:

```
$!GETFIELDVALUEREFCOUNT <macrovar>
  ZONE = <integer>
  VAR  = <varref>
  [no optional parameters]
```

Description:

Get the count of how many zones share the indicated variable with the specified zone. Count includes the specified zone.

Required Parameters

Parameter	Syntax	Notes
VAR	= <varref>	
ZONE	= <integer>	

Example:

A data set contains 5 zones and 3 variables. Zones 1, 2 and 4 share variable 3, and zones 3 and 5 share variable 3.

```
$!GETFIELDVALUEREFCOUNT |ABC|
  ZONE = 2
  VAR   = 3
```

This returns `|ABC|` = 3

```
$!GETFIELDVALUEREFCOUNT |DEF|
ZONE = 5
VAR   = 3
```

This returns `|DEF|` = 2 because the variable is not shared across all five zones.

$!GETNODEINDEX

Syntax:

```
$!GETNODEINDEX <macrovar>
  ZONE = <integer>
  ELEMENT = <integer>
  CORNER = <integer>
  [no optional parameters]
```

Description:

This function only works for finite-element zones. Query for the node index in the specified location as described by the `ZONE`, `ELEMENT`, and `CORNER` parameters.

Required Parameter

Parameter	Syntax	Default	Notes
ZONE	= <integer>		Zone must be greater than or equal to one.
CORNER	= <integer>		Possible values are 1-3, 1-4, or 1-8, depending upon the element type.
ELEMENT	= <integer>		Must be greater than or equal to one and less than or equal to |MAXJ|.

Example

Get the index for the node at corner 3 of the last element in zone number 1.

```
$!GETZONETYPE |ZONETYPE|
  ZONE = 1
$!IF "|ZONETYPE|" = "FE BRICK"
$!GETNODEINDEX |INDEX|
  ZONE = 1
  ELEMENT = |MAXJ|
CORNER = 3
... Do something with |INDEX|...
$!ENDIF
```

$!GETVARLOCATION

Syntax:

```
$!GETVARLOCATION <macrovar>
  ZONE = <integer>
  VAR = <varref>
```

Description:

Returns the location of the variable in the zone as either CELLCENTERED or NODAL and saves in the macro variable.

Required Parameter

Parameter	Syntax	Notes
VAR	= <varref>	
ZONE	= <integer>	

Example:

Get the variable location for the third variable in zone 1.

```
$!GETVARLOCATION |ABC|
  ZONE = 1
  VAR = 3
```

$!GETVARNUMBYNAME

Syntax:

```
$!GETVARNUMBYNAME <macrovar>
  NAME = <string>
```

Description:

Given a variable name, get the number for that variable. This variable number can then be used to assign attributes, such as what variable to use for contouring.

Required Parameter

Parameter	Syntax	Notes
NAME	= <string>	Name of the variable. If a variable has aliases, the name must correspond to one of the aliases.

Example:

Get the variable number for the variable named PRESSURE and make it the contouring variable.

```
$!GETVARNUMBYNAME |PVARNUM|
  NAME = "PRESSURE"
$!GLOBALCONTOUR
  VAR = |PVARNUM|
```

$!GETZONETYPE

Syntax:

```
$!GETZONETYPE <macrovar>
  ZONE = <integer>
  [no optional parameters]
```

Description:

Query for the zone type of the specified zone. The zone type will be assigned to <macrovar>.

The possible return values are ORDERED, FELINESEG, FETRIANGLE, FEQUAD, FETETRA, FEBRICK, FEPOLYGON, and FEPOLYHEDRON.

Required Parameter

Parameter	Syntax	Notes
ZONE	= <integer>	Zone must be greater than or equal to one.

Example

```
$!GETZONETYPE |ZONETYPE|
  ZONE = 1
$!IF "|ZONETYPE|" == "FETRIANGLE"
   $!PAUSE "The zone is FE-Triangle."
$!ENDIF
```

$!GLOBALCONTOUR

Syntax:

```
$!GLOBALCONTOUR <contourgroup>
  [optional parameters]
```

Description:

A SetValue command that changes global attributes associated with contour plots or contour levels. The optional parameter <contourgroup> refers to the defined contour groups, 1-8, allowed in Tecplot 360, and takes an integer value of one through eight. The <contourgroup> parameter is optional, and if omitted, Tecplot 360 will use contour group 1. If you would like the settings in these commands to persist, add them to your `tecplot.cfg` file, located in your installation directory. The `NUMBERFORMAT` setting for `LABELS` also controls the number format in the legend.

Optional Parameters

If you would like the settings in these commands to persist, add them to your `tecplot.cfg` file (located in your installation directory).

Parameter	Syntax	Default	Notes
COLORMAPNAME	= <string>		Assigns the named colormap to the indicated global contour group. Names are case-insensitive.
CONTOURLINESTYLE			This is used to assign a special line pattern scheme for contour line plots.
{			

Parameter	Syntax	Default	Notes
CONTOURLINEMODE	= <contourlinemode>	USEZONELINETYPE	
LINESKIP	= <integer>	4	
PATTERNLENGTH	= <dexp>	2	
}			
COLORCUTOFF			
{			
RANGEMIN	= <double>	-1 x 10^{150}	
RANGEMAX	= <double>	-1 x 10^{150}	
INCLUDEMIN	= <boolean>	NO	
INCLUDEMAX	= <boolean>	NO	
INVERTCUTOFF	= <boolean>	NO	
}			
COLORMAPFILTER			
{			
COLORMAPCYCLES	<op> <integer>		
COLORMAPDISTRIBUTION	<colormapdistribution>	BANDED	
COLORMAPOVERRIDE	<integer> <<colormapoverride>>		Use <integer> choose which override to operate on.
COLORMAPOVERRIDEACTIVE	= <boolean>	NO	
CONTINUOUSCOLOR	<<continuouscolor>>	CMIN=0, CMAX=1	
REVERSECOLORMAP	= <boolean>		
USEFASTAPPROXCONTINUOUSFLOOD	= <boolean>		
ZEBRA	<<zebrashade>>		
}			

Parameter	Syntax	Default	Notes
DEFNUMLEVELS	= <integer>	15	Sets the target number of contour levels for situations where contour levels are automatically reset. Tecplot 360 will attempt to create levels where the start, end and increment values are all clipped floating point values.
LABELS			
{			
ALIGNAUTOLABELS	= <boolean>	YES	If YES, automatic labels are aligned with the contour lines, otherwise they are horizontal.
AUTOLABELSPACING	<op> <dexp>	= 30	
AUTOLEVELSKIP	<op> < integer>	= 1	Value is in Y-frame units.
COLOR	= <color>	BLACK	
FILLCOLOR	= <color>	WHITE	
GENERATEAUTOLABELS	= <boolean>	YES	If YES, automatic labels are repositioned on each redraw.
ISFILLED	= <boolean>	YES	
LABELWITHVALUE	= <boolean>	YES	If YES, automatic labels show the contour value otherwise they show the contour level number.
MARGIN	<op> <dexp>	= 5	
NUMFORMAT	<<numberformat>>	See Notes	FORMATTING=BESTFLOAT, CUSTOMLABEL=1, PRECISION=4, SHOWDECIMALSONWHOLENUMBERS=NO, REMOVELEADINGZEROS=NO, SHOWNEGATIVESIGN=YES, TIMEDATEFORMAT='yyyy-mm-dd hh:mm:ss.00'
SHOW	= <boolean>	NO	
TEXTSHAPE	<<textshape>>	FONTFAMILY ="Helevetica", ISBOLD=NO, ISITALIC=NO, SIZEUNITS=FRAME, HEIGHT=1.5	Not allowed to change size units parameter.

Parameter	Syntax	Default	Notes
}			
LEGEND			
{			
ANCHORALIGNMENT	<anchoralignment>	TOPRIGHT	
AUTORESIZE	= <boolean>	NO	
AUTOSIZEMAXLIMIT	= <double>	0.5	
BOX	<<textbox>>	See Notes	BOXTYPE=HOLLOW, MARGIN=10, LINETHICKNESS=0.1, COLOR=BLACK, FILLCOLOR=WHITE
HEADER	<<header>>		
INCLUDECUTOFFLEVELS	= <boolean>	NO	
ISVERTICAL	= <boolean>	YES	
LABELINCREMENT	= <double>	1	
LABELLOCATION	= <contourlabellocation>	CONTOURLEVELS	
NUMBERTEXTSHAPE	<<textshape>>	FONTFAMILY="Helvetica", ISBOLD=NO, ISITALIC=NO, SIZEUNITS=FRAME, HEIGHT=2.5	
OVERLAYBARGRID	= <boolean>	YES	Thin line around each band in the color bar.
ROWSPACING	<op> <dexp>	= 1.2	
SHOW	= <boolean>	NO	
TEXTCOLOR	= <color>	BLACK	
XYPOS	<<xy>>	X=95, Y = 80	
}			
VAR	= <varref>	3	Var used for contour levels.

Example:

This example does the following: Turns on the contour legend; Sets the flood cutoff to go from 3 to 5;

Reverses the color map; Inserts a color map override of yellow between contour level number 7 and level number 9.

```
$!GLOBALCONTOUR 1
  LEGEND
  {
    SHOW = YES
  }
  COLORCUTOFF
  {
    RANGEMIN = 3
    RANGEMAX = 5
    INCLUDEMIN   = YES
    INCLUDEMAX = YES
  }
  COLORMAPFILTER
  {
    REVERSECOLORMAP = YES
    COLORMAPOVERRIDEACTIVE = YES
    COLORMAPOVERRIDE 1
    {
      INCLUDE    = YES
      COLOR      = YELLOW
      STARTLEVEL = 7
      ENDLEVEL   = 9
    }
  }
```

$!GLOBALEDGE

Syntax:

```
$!GLOBALEDGE
```

Description:

A SetValue command that sets attributes which sets the minimum crease angle for edges.

Optional Parameters

If you would like the settings in this command to persist, add them to your `tecplot.cfg` file (located in your installation directory).

Parameter	Syntax	Notes
MINCREASEANGLE	= <double>	

$!GLOBALFRAME

Syntax:

```
$!GLOBALFRAME
  [Optional Parameters]
```

Description:

A SetValue command that sets attributes which apply to all frames. If you would like the settings in this command to persist, add it to your tecplot.cfg file, located in your installation directory.

Optional Parameters

If you would like the settings in this command to persist, add them to your `tecplot.cfg` file (located in your installation directory).

Parameter	Syntax	Default	Notes
FRAMEHEADERFORMAT	= <string>		The <string> the text that appears in each of Tecplot 360's frame headers. This string typically contains dynamic text. See also the User's Manual. The default string is: &(FRAMENAME)|&(DATE)|&(DATASETTITLE).
FRAMEHEADERHEIGHT	<op> <dexp>	= 0.2	Value is in inches.
SNAPTOGRID	= <boolean>	NO	Even if set to YES, Tecplot 360 may not allow snapping in some situations.
SNAPTOPAPER	= <boolean>	NO	Even if set to YES, Tecplot 360 may not allow snapping in some situations.
USETHICKERACTIVEFRAMEBORDERONSCREEN	= <boolean>	YES	When YES, the active frame displays in the work area with a border one pixel wider than specified to make the active frame more visible. Set to NO to keep the active frame the same size as specified (but still black while inactive frames are gray). This will not affect printed or exported material unless the image is exported directly from the workspace.

Example:

Customize the frame header text, and set the frame header height to be 0.25 inches:

```
$!GLOBALFRAME
  FRAMEHEADERFORMAT = "My frame, the current date is &(Date), &(Time)"
  FRAMEHEADERHEIGHT = 0.25
```

$!GLOBALLINEPLOT

Syntax:

```
$!GLOBALLINEPLOT
  [Optional Parameters]
```

Description:

A SetValue command that changes global attributes associated with Line-plots. If you would like the settings in these commands to persist, add it to your `tecplot.cfg` file, located in your installation directory.

Optional Parameters

If you would like the settings in these commands to persist, add them to your `tecplot.cfg` file (located in your installation directory).

Parameter	Syntax	Default	Notes
DATALABELS			Text values that can be added to a plot to show the indices or values for the data points.
{			
COLOR	= <color>	BLACK	
COLORBYZONEMAP	= <boolean>	NO	
DISTANCESKIP	<op> <dexp>	= 5	
INCLUDEBOX	= <boolean>	YES	
INDEXSKIP	<op> < integer>	= 1	
NODELABELTYPE	= <labeltype>	INDEX	
NUMFORMAT	<<numberformat>>		
SHOWNODELABELS	= <boolean>	NO	

Parameter	Syntax	Default	Notes
SKIPMODE	= <skipmode>	BYINDEX	
TEXTSHAPE	<<textshape>>	See Notes	FONTFAMILY="Helvetica", ISBOLD=NO, ISITALIC=NO, SIZEUNITS=FRAME, HEIGHT=3
CELLLABELTYPE	= <labeltype>	INDEX	
}			
LEGEND			Attributes for an optional legend added to a Line-plot. Entries in the legend are determined dynamically by Tecplot 360 depending on which mappings are turned on.
{			
ANCHORALIGNMENT	= <anchoralignment>	TOPRIGHT	
BOX	<<textbox>>	See Notes	BOXTYPE=HOLLOW, MARGIN=10, LINETHICKNESS=0.1, COLOR=BLACK, FILLCOLOR=WHITE
ROWSPACING	<op> <dexp>	= 1.2	
SHOW	= <boolean>	NO	
SHOWTEXT	= <boolean>	YES	
TEXTCOLOR	= <color>	BLACK	
TEXTSHAPE	<<textshape>>	See Notes	FONTFAMILY="Helvetica", ISBOLD=YES, ISITALIC=NO, SIZEUNITS=FRAME, HEIGHT=3
XZYPOS	<<xy>>	X=95, Y=80	
}			

Example:

Turn on the data labels and show the line legend. Use boldface Times font in the legend:

```
$!GLOBALLINEPLOT
  DATALABELS
  {
    SHOWNODELABELS = YES
  }
  LEGEND
  {
    SHOW = YES
    TEXTSHAPE
```

```
    {
      FONTFAMILY = "Times"
      ISBOLD = YES
      ISITALIC = NO
    }
  }
```

$!GLOBALPAPER

Syntax:

```
$!GLOBALPAPER
  [optional parameters]
```

Description:

A SetValue command that sets the paper size characteristics. If you would like the settings in this command to persist, add it to your tecplot.cfg file, located in your installation directory.

Optional Parameters

If you would like the settings in these commands to persist, add them to your `tecplot.cfg` file (located in your installation directory).

Parameter	Syntax	Default	Notes
PAPERSIZEINFO			
{			
LETTER	<<papersize>>		WIDTH=8.5, HEIGHT=11, LEFTHARDCLIPOFFSET=0.125, RIGHTHARDCLIPOFFSET=0.125, TOPHARDCLIPOFFSET=0.125, BOTTOMHARDCLIPOFFSET=0.125
DOUBLE	<<papersize>>		
A3	<<papersize>>		WIDTH=11.693, HEIGHT=16.535, LEFTHARDCLIPOFFSET=0.125, RIGHTHARDCLIPOFFSET=0.125, TOPHARDCLIPOFFSET=0.125, BOTTOMHARDCLIPOFFSET=0.125

Parameter	Syntax	Default	Notes
A4	<<papersize>>		WIDTH=8.2677, HEIGHT=11.693, LEFTHARDCLIPOFFSET=0.125, RIGHTHARDCLIPOFFSET=0.125, TOPHARDCLIPOFFSET=0.125, BOTTOMHARDCLIPOFFSET=0.125
CUSTOM1	<<papersize>>		
CUSTOM2	<<papersize>>		
}			

See also: `$!PAGE`.

$!GLOBALPOLAR

Syntax:

```
$!GLOBALPOLAR
  [optional parameters]
```

Description:

Allows polar plots to have curved lines that are interpolated along the R-Axis between data points.

Optional Parameters

If you would like the settings in this command to persist, add them to your `tecplot.cfg` file (located in your installation directory).

Parameter	Syntax	Default	Notes
ANGLE	= <double>	5	Determines the angle for which lines will be approximated as curves.
DRAWSTRAIGHTLINES	= <boolean>	YES	Alternates between straight and curved interpolated lines for polar plots.

Example:

This example turns on curved lines and defines the maximum angle to be approximated as a curved line to be 2.0 degrees.

```
$!GLOBALPOLAR
  DRAWSTRAIGHTLINES = NO
```

```
  ANGLE = 2.0
```

$!GLOBALRGB

Syntax:

```
$!GLOBALRGB
  RGBMode = <rgbmode>
  [Optional Parameters]
```

Description:

Allows RGB coloring for plots which have RGB values specified at each vertex. This coloring option is valuable for plots with entities such as Gas, Oil and Water. RGB Coloring can be assigned to field plot objects such as zones, iso-surfaces and slices

Required Parameter

Parameter	Syntax	Default	Notes
RGBMODE	= <rgbmode>	SPECIFYRGB	Sets whether the user specifies all three color variables for RGB Coloring, or if Tecplot 360 calculates one variable while the user specifies two.

Optional Parameters

If you would like the settings in these commands to persist, add them to your `tecplot.cfg` file (located in your installation directory).

Parameter	Syntax	Default	Notes
REDCHANNELVAR	= <varref>		Sets variable for the red channel.
GREENCHANNELVAR	= <varref>		Sets variable for the green channel.
BLUECHANNELVAR	= <varref>		Sets variable for the blue channel.
RANGEMIN	= <double>	0.0	
RANGEMAX	= <double>	1.0	
LEGEND			
{			
ANCHORALIGNMENT	= <anchoralignment>	TOPCENTER	

Parameter	Syntax	Default	Notes
BLUECHANNELLABEL	= <string>		
BOX	<<textbox>>	See Notes	BOXTYPE=NONE, MARGIN=10, LINETHICKNESS=0.1, COLOR=BLACK, FILLCOLOR=WHITE
GREENCHANNELLABEL	= <string>		
HEIGHT	= <double>	10	
REDCHANNELLABEL	= <string>		
RGBLEGENDORIENTATION	= <rgblegendorientation>	RGB	
SHOW	= <boolean>	NO	
SHOWLABELS	= <boolean>	YES	
TEXTCOLOR	= <color>	BLACK	
TEXTSHAPE	<<textshape>>	See Notes	FONTFAMILY="Helvetica", ISBOLD=NO, ISITALIC=NO, SIZEUNITS=FRAME, HEIGHT=2.5
USEBLUEVARNAME	= <boolean>	YES	
USEGREENVARNAME	= <boolean>	YES	
USEREDVARNAME	= <boolean>	YES	
XYPOS	<<xy>>	X = 80, Y = 80	
}			

Example:

This example turns on RGB Coloring and defines variables for the Red and Green Channel, leaving Tecplot 360 to calculate the Blue Channel values.

```
$!GLOBALRGB
  RGBMODE = SPECIFYRG
  REDCHANNELVAR = 1
  GREENCHANNELVAR = 4
```

$!GLOBALSCATTER

Syntax:

```
$!GLOBALSCATTER
  [optional parameters]
```

Description:

A SetValue command that changes global attributes associated with scatter plots.

Optional Parameters

If you would like the settings in these commands to persist, add them to your `tecplot.cfg` file (located in your installation directory).

Parameter	Syntax	Default	Notes
BASEFONTFAMILY	= <string>	"Helvetica"	
DATALABELS			These are text labels that can be added to a plot to show node or cell values.
{			
CELLLABELTYPE	= <labeltype>	INDEX	
CELLLABELVAR	= <varref>		
COLOR	= <color>	BLACK	
COLORBYZONEMAP	= <boolean>	NO	
DISTANCESKIP	<op> <dexp>		
INCLUDEBOX	= <boolean>	YES	
INDEXSKIP	<op> < integer>	1	
NODELABELTYPE	= <labeltype>	INDEX	
NODELABELVAR	<op> <varref>		
NUMFORMAT	<<numberformat>>	See Notes	FORMATTING=BESTFLOAT, CUSTOMLABEL=1, PRECISION=4, SHOWDECIMALSONWHOLENUMBERS=NO, REMOVELEADINGZEROS=NO, SHOWNEGATIVESIGN=YES, TIMEDATEFORMAT='yyyy-mm-dd hh:mm:ss.00'
SHOWCELLLABELS	= <boolean>	NO	
SHOWNODELABELS	= <boolean>	NO	

Parameter	Syntax	Default	Notes
SKIPMODE	= <skipmode>		
TEXTSHAPE	<<textshape>>	See Notes	FONTFAMILY="Helvetica", ISBOLD=NO, ISITALIC=NO, SIZEUNITS=FRAME, HEIGHT=3
}			
ISBOLD	= <boolean>	YES	
ISITALIC	= <boolean>	NO	
LEGEND			
{			
ANCHORALIGNMENT	<<anchorpos>>	TOPRIGHT	
BOX	<<textbox>>	See Notes	BOXTYPE=HOLLOW,MARGIN=10, LINETHICKNESS=0.1, COLOR=BLACK, FILLCOLOR=WHITE
ROWSPACING	<op> <dexp>	= 1.2	
SHOW	= <boolean>	NO	
SHOWTEXT	= <boolean>	YES	
TEXTCOLOR	= <color>	BLACK	
TEXTSHAPE	<<textshape>>	See Notes	FONTFAMILY="Helvetica", ISBOLD=YES, ISITALIC=NO, SIZEUNITS=FRAME, HEIGHT=3
XYPOS	<<xy>>	X=95, Y=80	
}			
NUMCIRCLEPOINTS	= <integer>	12	Number of points used to draw circle scatter symbols. Range 12-360.
REFSCATSYMBOL			
{			
COLOR	= <color>	RED	
FILLCOLOR	= <color>	RED	
ISFILLED	= <boolean>	NO	
LINETHICKNESS	<op> <dexp>	= 0.1	
MAGNITUDE	<op> <dexp>	= 1	
SHOW	= <boolean>	NO	
SYMBOLSHAPE	<<symbolshape>>	See Notes	ISASCII=NO, GEOMSHAPE=SQUARE

Parameter	Syntax	Default	Notes
XYPOS	<<xy>>	X=80, Y=80	
}			
RELATIVESIZE	<op> <dexp>	= 0	Scaling factor for scatter symbols sized "By Variable."
RELATIVESIZEINGRIDUNITS	= <boolean>	YES	If YES, scatter sizing "By Variable" is in grid units /magnitude otherwise centimeters/magnitude.
SPHERESCATTERRENDERQUALITY	= <spherescatterrenderquality>	HIGH	Config file and stylesheet only option.
VAR	= <varref>		Scatter sizing variable.

Example:

This example does the following:

- Increases the relative size of scatter symbols that are sized by variable by ten percent.
- Turns on the scatter sizing legend.
- Turns on the reference scatter symbol and makes it red.
- Turns on data labels for nodes.

```
$!GLOBALSCATTER
  RELATIVESIZE * = 1.1
  LEGEND
  {
    SHOW = YES
  }
  REFSCATSYMBOL
  {
    SHOW = YES
    COLOR = RED
  }
  DATALABELS
  {
    SHOWNODELABELS = YES
  }
```

$!GLOBALTHREED

Syntax:

```
$!GLOBALTHREED
  [optional parameters]
```

Description:

A SetValue command that changes global attributes associated with 3D plots.

Optional Parameters

If you would like the settings in these commands to persist, add them to your `tecplot.cfg` file (located in your installation directory).

Parameter	Syntax	Default	Notes
AXISBOXPADDING	<op> <dexp>	= 5	
AXISSCALEFACT	<<xyz>>	X=1, Y=1, Z=1	The 3D axis must be INDEPENDENT for this option to work properly. See `$!THREEDAXIS`.
LIGHTSOURCE			
{			
BACKGROUNDLIGHT	= <double>	25	
INCLUDESPECULAR	= <boolean>	YES	
INTENSITY	= <double>	75	
SPECULARINTENSITY	= <integer>	40	Range = 1-100
SPECULARSHININESS	= <integer>	60	Range = 1-100
SURFACECOLORCONTRAST	= <double>	100	
XYZDIRECTION	<<xyz>>	X = -0.2, Y = -0.2, Z = 0.959...	Always specify all three components here. Tecplot 360 normalizes X, Y and Z after processing the Z-component. X, Y and Z represent a vector in the eye coordinate system.
FORCEGOURADFOR3DCONTFLOOD	= <boolean>	YES	
FORCEPANELEDFOR3DCELLFLOOD	= <boolean>	YES	
}			

Parameter	Syntax	Default	Notes
LINELIFTFRACTION	<op> <dexp>	= 0.2	
NEARPLANEFRACTION	<dexp>	0.1	Specify the position of the "near plane". In a 3D plot, the "near plane" acts as a windshield: anything in front of this plane does not display.
PERFORMEXTRA3DSORTING	<boolean>	NO	
PLACEMENTPLANEPOSITION			Specify the coordinate of the placement plane.
{			
X	= <double>		
Y	= <double>		
Z	= <double>		
}			
ROTATEORIGIN	<<xyz>>	X=0.5, Y=0.5, Z=0.5	
SLICE			
{			
ORIGIN	<<xyz>>	X=0, Y=0, Z=0	
NORMAL	<<xyz>>	X=0, Y=0, Z=1	
}			
SYMBOLLIFTFRACTION	<op> <dexp>	= 0.6	
VECTORLIFTFRACTION	<op> <dexp>	= 0.7	

Example

```
$!GLOBALTHREED ROTATEORIGIN{X = 4.36052333891}
$!GLOBALTHREED
  LIGHTSOURCE
  {
    XYZDIRECTION
    {
      X = 0.398226616447
      Y = 0.435028248588
```

```
      Z = 0.807567944438
    }
  }
$!GLOBALTHREED LIGHTSOURCE{INTENSITY = 80}
$!GLOBALTHREED LIGHTSOURCE{BACKGROUNDLIGHT = 25}
$!GLOBALTHREED LIGHTSOURCE{SURFACECOLORCONTRAST = 85}
$!GLOBALTHREED LINELIFTFRACTION = 7
$!GLOBALTHREED SYMBOLLIFTFRACTION = 0.5
$!GLOBALTHREED VECTORLIFTFRACTION = 6
$!GLOBALTHREED PERFORMEXTRA3DSORTING = YES
```

$!GLOBALTHREEDVECTOR

Syntax:

```
$!GLOBALTHREEDVECTOR
  [optional parameters]
```

Description:

A SetValue command that changes global attributes associated with 3D vector plots.

Optional Parameters

If you would like the settings in these commands to persist, add them to your `tecplot.cfg` file (located in your in installation directory).

Parameter	Syntax	Default	Notes
ARROWHEADANGLE	<op> <dexp>	= 18	Angle is in degrees.
EVENSPACING	<<xyz>>		Approximate spacing in the coordinate directions between evenly spaced vectors.
HEADSIZEASFRACTION	<op> <dexp>	= 0.2	Head is sized as a fraction of the stem length.
HEADSIZEINFRAMEUNITS	<op> <dexp>	= 2	Value is in Y-frame units.
REFVECTOR			
{			
SHOW	= <boolean>	NO	
COLOR	= <color>	BLACK	
MAGNITUDE	<op> <dexp>	= 1	

Parameter	Syntax	Default	Notes
LINETHICKNESS	<op> <dexp>	= 0.1	
ANGLE	<op> <dexp>	= 0	
XYPOS	<<xy>>	X=80, Y=80	
MAGNITUDELABEL			
{			
SHOW	= <boolean>	NO	
TEXTCOLOR	= <color>	BLACK	
TEXTSHAPE	<<textshape>>		
NUMFORMAT	<<numberformat>>	See Notes	FONTFAMILY="Helvetica", ISBOLD=YES, ISITALIC=NO, SIZEUNITS=FRAME, HEIGHT=3
OFFSET	= <double>	2	
}			
}			
RELATIVELENGTH	<op> <dexp>	= 0	
RELATIVELENGTHIN GRIDUNITS	= <boolean>	YES	If YES and USERELATIVE is YES then vectors are sized in Grid Units/Magnitude. If NO and USERELATIVE is YES then vectors are sized in cm/magnitude.
SIZEHEADBY FRACTION	= <boolean>	YES	If YES, HEADSIZEASFRACTION is used to size arrowheads otherwise HEADSIZEINFRAMEUNITS is used.
USEEVENSPACING	= <boolean>	NO	Disperses vectors more evenly throughout the plot.
UNIFORMLENGTH	<op> <dexp>	= 2	Value is in Y-frame units.
USERELATIVE	= <boolean>	YES	If NO, vectors are all the same size (UNIFORMLENGTH).
UVAR	= <varref>		Variable reference for the X-vector component.
VVAR	= <varref>		Variable reference for the Y-vector component.
WVAR	= <varref>		Variable reference for the Z-vector component.

Example:

This example does the following:

- Makes all vectors be uniform in size; 5 percent in Y-frame units.
- Makes the arrowheads 0.2 times the size of the stems.
- Turns off the reference vector.

```
$!GLOBALTHREEDVECTOR
  USERELATIVE = NO
  UNIFORMLENGTH = 5
  HEADSIZEASFRACTION = .2
  REFVECTOR
  {
    SHOW = NO
  }
```

$!GLOBALTIME

Syntax:

```
$!GLOBALTIME
  [optional parameters]
```

Description:

A SetValue command for frames (2D and 3D ONLY). Different frames can have different values of `$!GLOBALTIME`. If you would like the settings in this command to persist, add them to your `tecplot.cfg` file (located in your installation directory).

Parameter	Syntax	Default	Notes
SOLUTIONTIME	= <double>	0	If SolutionTime is not set to a solution time in set of solution times from all zones of the active strand field-maps, SolutionTime is adjusted to the closest value in that set.

Parameter	Syntax	Default	Notes
TRANSIENTZONEVISIBILITY	= <transientzonevisibility>	ZonesAtOrBeforeSolutionTime	ZonesAtOrBeforeSolutionTime: For each strand at a given solution time, the zones shown are those that are at the solution time, within a tolerance. If no zones exist at the solution time but the solution time falls within the range of times spanned by the strand, zones are shown from the strand's first prior solution time. If the solution time falls outside the range spanned by the strand, no zones from the strand are shown. ZonesAtSolutionTime: For each strand at a given solution time, the zones shown are those that are at the solution time, within a tolerance.

$!GLOBALTWODVECTOR

Syntax:

```
$!GLOBALTWODVECTOR
  [optional parameters]
```

Description:

A SetValue command that changes global attributes associated with 2D vector plots.

Optional Parameters

If you would like the settings in these commands to persist, add them to your `tecplot.cfg` file (located in your installation directory).

Parameter	Syntax	Default	Notes
ARROWHEADANGLE	<op> <dexp>	= 18	Angle is in degrees.
EVENSPACING	<<xy>>		Approximate spacing in the coordinate directions between evenly spaced vectors.
HEADSIZEASFRACTION	<op> <dexp>	= 0.2	Head is sized as a fraction of the stem length.
HEADSIZEINFRAMEUNITS	<op> <dexp>	= 2	Value is in Y-frame units.
REFVECTOR			

Parameter	Syntax	Default	Notes
{			
SHOW	= <boolean>	NO	
COLOR	= <color>	BLACK	
MAGNITUDE	<op> <dexp>	= 1	
LINETHICKNESS	<op> <dexp>	= 0.1	
ANGLE	<op> <dexp>	= 0	
XYPOS	<<xy>>	X=80, Y=80	
MAGNITUDELABEL			
{			
SHOW	= <boolean>	NO	
TEXTCOLOR	= <color>	BLACK	
TEXTSHAPE	<<textshape>>	See Notes	FONTFAMILY="Helvetica", ISBOLD=YES, ISITALIC=NO, SIZEUNITS=FRAME, HEIGHT=3
NUMFORMAT	<<numberformat>>		
OFFSET	= <double>	2	
}			
}			
RELATIVELENGTH	<op> <dexp>	= 0	
RELATIVELENGTHINGRIDUNITS	= <boolean>	YES	If YES and USERELATIVE is YES then vectors are sized in Grid Units/Magnitude. If NO and USERELATIVE is YES then vectors are sized in cm/magnitude.
SIZEHEADBYFRACTION	= <boolean>	YES	If YES, HEADSIZEASFRACTION is used to size arrowheads otherwise HEADSIZEINFRAMEUNITS is used.
USEEVENSPACING	= <boolean>	NO	Disperses vectors more evenly throughout the plot.
UNIFORMLENGTH	<op> <dexp>	= 2	Value is in Y-frame units.
USERELATIVE	= <boolean>	YES	If NO, vectors are all the same size (UNIFORMLENGTH).
UVAR	= <varref>		Variable reference for the X-vector component.

Parameter	Syntax	Default	Notes
VVAR	= <varref>		Variable reference for the Y-vector component.

Example:

This example does the following:

- Doubles the vector length (assume vectors currently drawn using relative length).
- Make the vector heads uniform in size; 2 percent in frame units.
- Make the head angle 15 degrees.

```
$!GLOBALTWODVECTOR
  RELATIVELENGTH *= 2
  SIZEHEADBYFRACTION = NO
  HEADSIZEINFRAMEUNITS = 2
  HEADANGLE = 15
```

$!IF···$!ENDIF

Syntax:

```
$!IF <conditionalexp>
$!ENDIF
```

Description:

Conditionally process macro commands.

Examples

Example 1:

Process macro commands if the macro variable `|myvar|` is less than 73.2:

```
$!IF |myvar| < 73.2
  ... macro commands ...
$!ENDIF
```

Example 2:

Process macro commands if the macro variable `|response|` is `YES`:

```
$!IF "|response|" == "YES"
   .
   .
   .
$!ENDIF
```

$!INCLUDEMACRO

Syntax:

```
$!INCLUDEMACRO <string>
```

Description:

Insert commands from another macro file. Note that any macro variables used within the string must either pre-exist loading of this macro or be macro variables that reference environment variables. Macro variables that are defined within the loaded macro are not assigned until execution of the macro and thus will not expand.

Example

Include the macro file `m2.mcr`:

```
$!INCLUDEMACRO "m2.mcr"
```

$!INTERFACE

Syntax:

```
$!INTERFACE
  [optional parameters]
```

Description:

A SetValue command that sets attributes related to the Tecplot 360 interface.

Optional Parameters

If you would like the settings in these commands to persist, add them to your `tecplot.cfg` file (located in your installation directory). Some can be used only in `tecplot.cfg`.

Parameter	Syntax	Default	Notes
ALLOWDATAPOINTSELECT	= <boolean>	YES	If YES, Tecplot 360 allows you to use the Adjustor tool to select and move data points.
ALLOWDATAPOINTADJUST	= <boolean>	YES	
ALWAYSPOPACTIVEFRAME	= <boolean>	NO	If YES, whenever the active frame changes in the interface, Tecplot 360 moves the active frame to the top of the draw order. This makes Tecplot 360 run like v.11.3 and earlier versions, and may help older macros to run correctly. Default is NO.
AUTOREDRAWISACTIVE	= <boolean>	YES	Set to NO to turn Auto Redraw off.
BACKINGSTOREMODE	= <backingstoremode>	REALTIMEUPDATE	
BEEPONFRAMEINTERRUPT	= <boolean>	NO	
CACHELIGHTDISPLAYLISTSONLY	= <boolean>	NO	When caching graphics in display lists, only cache those objects which use little memory. When this is on, only approximated plots are saved. Full plots are not saved. Only has an effect if USEDISPLAYLISTS is set to YES and USEAPPROXIMATEPLOTS is YES.
CONSERVEDERIVEDVARIABLESPACE	= <boolean>		
COLLECTTIMESTATISTICS	= <boolean>	NO	
DATA			
{			
SMOOTHBNDRYCOND	= <boundarycondition>	FIXED	
NUMSMOOTHPASSES	<op> <integer>	= 1	
SMOOTHWEIGHT	<op> <dexp>	= 0.5	
INVDISTEXPONENT	<op> <dexp>	= 3.5	
INVDISTMINRADIUS	<op> <dexp>	= 0	

Parameter	Syntax	Default	Notes
LINEARINTERPCONST	<op> <dexp>	= 0	
LINEARINTERPMODE	= <linearinterpmode>	SETTOCONST	
INTERPPTSELECTION	= <pointselection>	OCTANTNPOINTS	
INTERPNPOINTS	<op> <integer>	= 8	
KRIGRANGE	<op> <dexp>	= 0.3	
KRIGZEROVALUE	<op> <dexp>	= 0	
KRIGDRIFT	= <krigdrift>	LINEAR	
DERIVATIVEBOUNDARY	= <derivpos>	SIMPLE	
TRIANGLEKEEPFACTOR	<op> <dexp>	= 0.25	
VARIABLEDERIVATIONMETHOD	= [ACCURATE or FAST]	FAST	
VOLUMECELL INTERPOLATIONMODE	= [TRILINEAR or PIECEWISELINEAR]	PIECEWISELINEAR	By default, volume interpolation is piece-wise linear and calculates for first-order accuracy. This tri-linear interpolation option calculates for second-order accuracy.
EXTRACTSLICEFROMPLANECOPYCELLCENTERS	= <boolean>	NO	
CONTLINECREATEMODE	= [ONEZONEPER CONTOURLEVEL, ONEZONEPER INDEPENDENT POLYLINE]	ONEZONEPER CONTOURLEVEL	

Parameter	Syntax	Default	Notes
POLYCELL INTERPOLATIONMODE	= [USECCVALUE, AVERAGENODES]	AVERAGENODES	This parameter only appears in the tecplot config file. When using contour flooding, contour lines, or precise value blanking for polyhedral and polygonal data, it chooses between using the cell-centered value or using the average of the nodes. This parameter also applies to the creation of iso-surfaces for cell-centered values in a polyhedral zone. It does not apply to Primary Value contour flooding.
USESTABLESORTFORCONNECTIVITY	= <boolean>	NO	Ensures consistent node map ordering of extracted slices and iso-surfaces. Comes with a performance penalty.
}			
ENABLEDELAYS	= <boolean>	YES	Enable or disable delays in macro commands.
ENABLEERRORS	= <boolean>	YES	If NO, no error message dialogs are displayed.
ENABLEINTERRUPTS	= <boolean>	YES	Enable or disable user interrupts.
ENABLEPAUSES	= <boolean>	YES	Enable or disable pause.
ENABLEWARNINGS	= <boolean>	YES	Enable or disable warning dialogs.
GLOBALPLACEMENTPLANE			Use a placement plane.
{			
SHOW			Turn the placement plane on/off.
PLACEMENTPLANEORIENTATION			Specify the axis orientation of the placement plane.
FONTPATH	<string>	The OS font folder(s)	One or more directories where TrueType fonts can be found, e.g. '"PATH1" "PATH2" "PATH3"'. Note use of double quotes around each path and single quotes around the entire string.
IDOTSPERINCH	= <double>		This along with JDOTSPERINCH set the size and aspect of the screen. If left unspecified, Tecplot 360 will determine the value for you.
INITIALPLOTFIRSTZONEONLY	= <boolean>	NO	If YES, only the first enabled zone is activated. Default shows all zones (except from within a layout).
INITIALPLOTTYPE	= <plottype>	AUTOMATIC	

Parameter	Syntax	Default	Notes
INTERRUPTCHECKINGFREQUENCY	= <integer>		Set the number of milliseconds between checks for a key- or button-press by the user to interrupt processing in Tecplot 360.
JDOTSPERINCH	= <double>		This along with IDOTSPERINCH set the size and aspect of the screen. If left unspecified, Tecplot 360 will determine the value for you.
LISTCOMMANDSINMACROVIEWER	= <boolean>		If NO, macro commands are displayed in full one at a time.
LOADADDONSUSINGLAZYRELOCATE	= <boolean>	YES	If set to NO, all add-on symbols are loaded immediately.
MAXNUMUNDOLEVELS	= <integer>	50	
MAXRENDERTIMEBEFOREAPPROX	= <double>	0.8	When not using the "All Frames Always Approximated" mode use this value to determine when, in seconds, to automatically turn on plot approximation for all frames. All frames will be approximated when the time to render the plot (all frames) when doing interactive view changes (rotation, translation, scaling) is greater than the supplied value
MINPIXELSFORDRAG	= <integer>	1	Number of pixels to move the pointer before it is considered a drag.
MOUSEACTIONS			
{			
MIDDLEBUTTON			
{			
BUTTONCLICK	= <mousebuttonclick>	REDRAW	
SIMPLEDRAG	= <mousebuttondrag>	ZOOMDATA	
CONTROLLEDDRAG	= <mousebuttondrag>	ZOOMDATA	

Parameter	Syntax	Default	Notes
ALTEDDRAG	= <mousebutton drag>	ZOOMVIEWER	
SHIFTEDDRAG	= <mousebutton drag>	ZOOMPAPER	
CONTROLALTEDDRAG	= <mousebutton drag>	ZOOMVIEWER	
CONTROLSHIFTEDDRAG	= <mousebutton drag>	ZOOMPAPER	
ALTSHIFTEDDRAG	= <mousebutton drag>	ZOOMVIEWER	
CONTROLALTSHIFTEDDRAG	= <mousebutton drag>	ZOOMPAPER	
}			
RIGHTBUTTON			
{			
BUTTONCLICK	= <mousebutton click>	REVERTTOSELECT	
SIMPLEDRAG	= <mousebutton drag>	TRANSLATEDATA	
CONTROLLEDDRAG	= <mousebutton drag>	ROLLERBALL ROTATEDATA	
ALTEDDRAG	= <mousebutton drag>	TRANSLATE VIEWER	
SHIFTEDDRAG	= <mousebutton drag>	TRANSLATEPAPER	

Parameter	Syntax	Default	Notes
CONTROLALTEDDRAG	= <mousebutton drag>	ZOOMVIEWER	
CONTROLSHIFTEDDRAG	= <mousebutton drag>	ROLLERBALL ROTATEDATA	
ALTSHIFTEDDRAG	= <mousebutton drag>	TRANSLATEPAPER	
CONTROLALTSHIFTEDDRAG	= <mousebutton drag>	ZOOMVIEWER	
}			
}			
NUMCOLORCELLSTORETURN	= <integer>	10	
NUMMOUSEBUTTONS	= <integer>		Only for Linux users who are using MIDDLEMOUSEBUTTONMODE or RIGHTMOUSEBUTTONMODE.
NUMPTSALLOWEDBEFOREAPPROX	= <integer>	500,000	Ignored. Use MAXRENDERTIMEBEFOREAPPROX instead
OKTOEXECUTESYSTEMCOMMAND	= <boolean>	YES	Allow use of $!SYSTEM commands in macros. This is a security issue. If set to NO and the macro is run interactively, you will be asked for permission to execute the $!SYSTEM command. If Tecplot 360 is run in batch mode and this is NO an error will be generated and the macro will terminate.
OPENGLCONFIG			
{			
RUNDISPLAYLISTSAFTERBUILDING	= <boolean>	YES	Tecplot 360 defaults to building and running display lists simultaneously. Turn RunDisplayListsAfterBuilding on if you want to run the display lists after they are built. This may increase display list performance on some machines. The difference is often times negligible.

Parameter	Syntax	Default	Notes
ALLOWHWACCELERATION	= <boolean>	YES	Windows only. Disables hardware acceleration in Tecplot 360 without having to change the Windows Display properties. Setting ALLOWHWACCELERATION to NO may fix errors caused by hardware acceleration on buggy graphics card drivers.
SCREENRENDERING	<<renderconfig>>		Sets options for screen rendering.
IMAGERENDERING	<<renderconfig>>		Sets options for offscreen rendering (for image exports or raster printing).
MAXFILTERMAGNIFICATION	= <integer>	2	Sets the maximum magnification by non-texture resize filter before textures are used. This keeps Tecplot 360 from creating textures which are too large. Setting this above three is not recommended. Setting below 1.0 will result in the use of a faster texture algorithm.
INCLUDEBACKBUFFER DURINGFRONTBUFFERDRAWS	= <boolean>	NO	
PRETRANSLATEDATA	= <pretranslatemode>	AUTO	Pre-translation may be used to solve jitter issues with data that is far from zero but has small deltas. On data surrounding zero it is better to turn it off. AUTO turns the setting on when necessary based on your data.
USEADVANCEDVIUSUALCHOOSER	= <boolean>	YES	If YES, activate Tecplot's advanced OpenGL visual chooser, which will attempt to choose a reasonable visual based on your system capabilities. If NO, the Qt visual chooser is used; this may choose a different visual. Also allows a fixed visual mode to be specified using VISUALID (next). Available in `tecplot.cfg` only and has effect only on Linux.

Parameter	Syntax	Default	Notes
VISUALID	= <integer>	0	Specify the OpenGL visual. If 0, the system automatically chooses a visual based on your system capabilities. Other values may be obtained using the `glxinfo` tool. (The visual IDs displayed in `glxinfo` are in hexadecimal and must be converted to decimal for use here.) USEADVANCEDVISUALCHOOSER must be YES. Available in `tecplot.cfg` only and has effect only on Linux.
}			
PERCENTAGEOFPOINTSTOKEEP	= <integer>	10	Sets the percentage of points to keep in a frame when a frame is approximated. See also Best Practices for Screen Performance in the User's Manual.
PICKHANDLEWIDTH	<op> <dexp>	= 0.1	Value is in inches on the screen.
PLOTAPPROXIMATIONMODE	= <plotapproximationmode>	AUTOMATIC	Specifies the mode in which you want the plots to be approximated. See Plot Approximations in the User's Manual for a complete description of each mode.
PRINTDEBUG	= <boolean>	NO	If YES, debugging information is sent to the standard output.
ROTATION			Settings for interactive rotations in 3D.
{			
ARBSLICESTEPANGLE	= <double>	0.5	Step value for orienting arbitrary slice using +/- buttons (degrees).
ROTATIONMODE	= <rotationmode>	XYAXIS	
CURRENTANGLE	<op> <dexp>	= 5	
SMALLANGLE	<op> <dexp>	= 1	
MEDIUMANGLE	<op> <dexp>	= 5	
LARGEANGLE	<op> <dexp>	= 15	
ROTATEDEGPERFRAMEUNIT	= <integer>		
SHOWGEOMS	= <boolean>	YES	
}			

Parameter	Syntax	Default	Notes
ROTATEDEGPERFRAMEUNIT	= <integer>		
RULERPADDING	<op> <dexp>	= 0.05	Distance between workarea ruler and clipping edge for the paper and frames. Units are inches.
RULERTHICKNESS	<op> <dexp>	= 0.15	Value is in inches on the screen.
SCALE			Settings for interactive scaling.
{			
STEPSIZE	<op> <dexp>	= 10	
SMALLSTEP	<op> <dexp>	= 1	
MEDIUMSTEP	<op> <dexp>	= 10	
LARGESTEP	<op> <dexp>	= 20	
ZOOMSCALEPERFRAMEUNIT	<op> <double>	= 4	
}			
SCRBACKGROUNDCOLOR	= <color>		Set the workspace background color.
SECURESPOOLCOMMANDS	= <boolean>	YES	Set to NO to allow $!SPOOLER commands outside the configuration file.
SHOWCONTINUOUSSTATUS	= <boolean>	YES	
SHOWCOORDINATES	= <boolean>	YES	
SHOWCOORDINATESWITHSELECTORADJUSTOR	= <boolean>	NO	When YES, running coordinates for both X&Y are displayed in grid coordinates in the status line for all plot types but 3D. If $!INTERFACE SHOWCOORDINATES is off, value of SHOWCOORDINATESWITH SELECTORADJUSTOR is ignored. This command is available via the config file (tecplot.cfg) only.
SHOWFRAMEBORDERSWHENOFF	= <boolean>	NO	If YES, frame borders are drawn using a dashed line when they are turned off. This applies only to the screen and does not effect the hardcopy.
SHOWSTATUSLINE	= <boolean>	YES	

Parameter	Syntax	Default	Notes
SHOWTEXTGEOMSINAPPROXVIEWS	= <boolean>	YES	Set to YES if you want text and geometries to show up in frames using approximated plots
SHOWTOOLTIPS	= <boolean>	YES	
SHOWWAITDIALOGS	= <boolean>	YES	If NO, all "Please Wait" and "Percent Done" dialogs will be disabled.
SIDEBARSIZING	= <sidebarsizing>	MAXOFALL	
TRANSLATION			Settings for interactive translation.
{			
STEPSIZE	<op> <dexp>	= 10	
SMALLSTEP	<op> <dexp>	= 5	
MEDIUMSTEP	<op> <dexp>	= 10	
LARGESTEP	<op> <dexp>	= 20	
ZOOMSCALEPERFRAMEUNIT	= <double>		
}			
TRUETYPEMINOUTLINEPOINTSIZE	= <integer>	19 on Windows 16 on other platforms	Tecplot 360 will use a bitmap font for text smaller than this size.
USEMOD2MASKFORALTDETECTION	= <boolean>	NO	Certain platforms have a problem with the ALT key. Set to YES to bypass the problem.
USECLASSICADJUSTORTOOL	= <boolean>	NO	If set to YES, uses the old-style adjustor tool that selects the closest control point, rather than the whole object, when clicking.
TRYTOUSEDOUBLEBUFFER	= <boolean>	YES	
USEAPPROXIMATEPLOTS	= <boolean>	NO	Set to YES to use approximate plots. See Plot Approximation in the User's Manual for further details.
USEDISPLAYLISTS	= <boolean>	YES	
USEDOUBLEBUFFERING	= <boolean>		

Parameter	Syntax	Default	Notes
USEDOUBLEFOR DISPLAYLISTS	= <boolean>	YES	
USEFASTAPPROX CONTINUOUSFLOOD	= <boolean>		
USEOPENGL	= <boolean>	YES	
USESTROKEFONTSFO R3DTEXT	= <boolean>	YES	Set to YES to use Tecplot 360's internal stroke fonts, set to NO to use True Type fonts. This option is only available on Windows platforms.
USESTROKEFONTS FORSMALLSCREENTE XT	= <boolean>		See above
USESTROKEFONTSON SCREEN	= <boolean>	NO	See above
USETECPLOTPRINTDR IVERS	= <boolean>	NO	Set to YES to use Tecplot 360's printer drivers. Set to NO to use your default printer driver.
XORCOLOR	<op> < integer>	= 0	Color index to use for XORed lines. Set to 0 to make Tecplot 360 calculate.
XPGONFILLOFFSET	= <integer>	0	
YPGONFILLOFFSET	= <integer>	0	
ZONEBOUNDINGBOX MODE	= <boundingbox mode>	AUTO	Corresponds to Show Bounding Boxes for Enabled Volume Zones with No Style on Option menu.

Example:

This example does the following:

- Makes the frame borders show on the screen when they are turned off.
- Makes the middle mouse button be Redraw.
- Makes the right mouse button revert to Selector.
- Makes the default number of passes for smoothing 20.
- Turns off the status line.

```
$!INTERFACE
  SHOWFRAMEBORDERSWHENOFF = YES
  MOUSEACTIONS
  {
```

```
    MIDDLEBUTTON
    {
      BUTTONCLICK = REDRAW
    }
    RIGHTBUTTON
    {
      BUTTONCLICK = REVERTTOSELECT
    }
  }
  DATA
  {
    NUMSMOOTHPASSES = 20
  }
  SHOWSTATUSLINE = NO
```

$!INVERSEDISTINTERPOLATE

Syntax:

```
$!INVERSEDISTINTERPOLATE
  DESTINATIONZONE = <integer>
  [optional parameters]
```

Description:

Interpolate selected variables from one or more zones onto a destination zone using the inverse distance method.

Required Parameter

Parameters	Syntax	Notes
DESTINATIONZONE	= <integer>	Zone to which to interpolate.

Optional Parameters

Parameters	Syntax	Default	Notes
INTERPNPOINTS	= <integer>	8	
INTERPPTSELECTION	= <interpptselection>	OCTANTNPOINTS	
INVDISTEXPONENT	= <dexp>	3.5	
INVDISTMINRADIUS	= <dexp>	0.0	

Parameters	Syntax	Default	Notes
SOURCEZONES	= <set>	All zones except destination zone.	
VARLIST	= <varset>	All variables except spatial variables.	Choose the variables to interpolate. The spatial variables (X, Y and Z if 3D) are not allowed.

Example

Interpolate variables 7-10 from zone 4 to zone 2:

```
$!INVERSEDISTINTERPOLATE
  SOURCEZONES     = [4]
  DESTINATIONZONE = 2
  VARLIST       = [7-10]
```

$!ISOSURFACEATTRIBUTES

Syntax:

```
$!ISOSURFACEATTRIBUTES [<group>]
  [optional parameters]
```

Description:

A SetValue command which changes attributes associated with iso-surfaces. The optional group parameter can range from 1-8 and defaults to 1 when absent.

Optional Parameters

Parameter	Syntax	Default	Notes
SHOWGROUP	= <boolean>	YES	
ISOSURFACESELECTION	= <isosurfaceselection>	ONESPECIFICVALUE	
ISOVALUE1	= <double>	1×10^{150}	
ISOVALUE2	= <double>	1×10^{150}	
ISOVALUE3	= <double>	1×10^{150}	

Parameter	Syntax	Default	Notes
MESH			
{			
SHOW	= <boolean>	NO	
COLOR	= <color>		
MESHTYPE	= <meshtype>	OVERLAY	
}			
CONTOUR			
{			
SHOW	= <boolean>	YES	
CONTOURTYPE	= <contourtype>	FLOOD	PRIMARYVALUE and AVERAGECELL not allowed.
FLOODCOLORING	= <contourcoloring>	GROUP1	
LINECONTOURGROUP	= <integer>	1	
COLOR	= <color>		
LINETHICKNESS	= <double>	0.1	
USELIGHTINGEFFECT	= <boolean>	NO	
}			
EFFECTS			
{			
USEVALUEBLANKING	= <boolean>	YES	
LIGHTINGEFFECT	= <lightingeffect>	GOURAUD	
SURFACETRANSLUCENCY	= <translucency>	50	
USETRANSLUCENCY	= <boolean>	NO	
USECLIPPLANES	<set>	[1-6]	
}			
DEFINITIONCONTOURGROUP	= <integer>	1	Contour group from which iso-surfaces are based.

Parameter	Syntax	Default	Notes
OBEYSOURCEZONEBLANKING	= <boolean>	NO	
OBEYCLIPPLANES	= <boolean>	YES	Clip the iso-surface by any clipping planes that intersect the iso-surface.
SHADE			
{			
COLOR	= <color>		
SHOW	= <boolean>	NO	
USELIGHTINGEFFECT	= <boolean>	YES	
}			
SURFACEGENERATIONMETHOD	= <surfacegenerationmethod>	AUTO	**Auto**: Selects one of the surface generation algorithms best suited for the zones participating in the iso-surface generation. "All Polygons" is used if one or more of the participating zones is polytope, otherwise isosurfaces use "All Triangles" unless the iso-surface is defined by a coordinate variable in which case "Allow Quads" is used. **AllowQuads**: Produces quads or triangles, and the resulting surface more closely resembles the shape of the volume cells from the source zone. **AllTriangles**: An advanced algorithm that can handle complex saddle issues and guarantees that there will be no holes in the final surface. As the surface is composed entirely of triangles, it can be delivered more efficiently to the graphics hardware. **AllPolygons**: Similar to the "All triangles" method except that all interior faces generated as a result of triangulation that are not part of the original mesh are eliminated. This preserves the original mesh of the source zones on the resulting iso-surface.
VECTOR			
{			

Parameter	Syntax	Default	Notes
SHOW	= <boolean>	NO	
VECTORTYPE	= <vectortype>	TAILATPOINT	
ARROWHEADSTYLE	= <arrowheadstyle>	PLAIN	
COLOR	= <color>	BLACK	
ISTANGENT	= <boolean>	NO	
LINETHICKNESS	= <double>	0.1	
}			

Example:

```
$!ISOSURFACEATTRIBUTES
 ISOSURFACESELECTION = ONESPECIFICVALUE
 ISOVALUE1 = 113.626812744
 MESH{SHOW = YES}
 MESH{COLOR = BLUE}
 MESH{LINETHICKNESS = 0.4}
 CONTOUR{SHOW = YES}
 SURFACEEFFECTS{LIGHTINGEFFECT = PANELED}
 SURFACEEFFECTS{SURFACETRANSLUCENCY = 60}
```

$!ISOSURFACELAYERS

Syntax:

```
$!ISOSURFACELAYERS
  SHOW = <boolean>
```

Description:

Turn iso-surfaces on or off.

Required Parameters

Parameter	Syntax	Default	Notes
SHOW	= <boolean>	NO	

When iso-surfaces are activated, they are colored using the contour variable by

default. When you activate iso-surfaces via the macro language you must perform one of the following steps in order for the iso-surfaces to be visible in your final plot:

- Set the contour variable by adding the following command to your macro file (prior to calling `$!ISOSURFACELAYERS`):
  `$!GLOBALCONTOUR` *n* VAR=*m*
  where *n* is the contour group number and *m* is the number of the variable to use for contouring.

or

- Set `CONTOURSHOW = NO` via the `$!ISOSURFACEATTRIBUTES` command. If you choose this option, you may want to turn on shading to improve the visibility of your slice.

K-N

$!KRIG

Syntax:

```
$!KRIG
  DESTINATIONZONE = <integer>
  [optional parameters]
```

Description:

Interpolate selected variables from a set of source zones to a destination zone using the kriging method.

Required Parameter

Parameters	Syntax	Default	Notes
DESTINATIONZONE	= <integer>		Zone to interpolate to.

Optional Parameters

Parameters	Syntax	Default	Notes
INTERPNPOINTS	= <integer>	8	
INTERPPTSELECTION	= <interpptselection>	OCTANTNPOINTS	
KRIGDRIFT	= <krigdrift>	LINEAR	

Parameters	Syntax	Default	Notes
KRIGRANGE	= <dexp>	0.3	
KRIGZEROVALUE	= <dexp>	0.0	
SOURCEZONES	= <set>	All zones except the destination zone.	
VARLIST	= <varset>	All variables except spatial variables.	Choose the variables to interpolate. The spatial variables (X, Y and Z if 3D) are not allowed.

Example

Krig from zones 3 and 4 onto zone 2. Only interpolate variable 7:

```
$!KRIG
  SOURCEZONES = [3, 4]
  DESTINATIONZONE = 2
  VARLIST      = [7]
```

$!LATEX

Syntax:

```
$!LATEX <string>
  [optional parameters]
```

Description

Set configuration attributes for using LaTeX expressions in Tecplot plots. By default LaTeX configuration attributes are read from a tecplot_latex.mcr file placed in standard search locations such as the home folder.

Optional Parameters

Parameter	Syntax	Default	Notes
DVIPNGCMD	<string>	N/A	A path to an utility that converts DVI files to PNG.

Parameter	Syntax	Default	Notes
Preamble	<string> OR <rawstring>	N/A	A LaTeX preamble that will be prepended to every LaTeX expression. For multi-line support use the raw string format. For more information about raw strings see the TEXT parameter of `$!ATTACHTEXT`.
LATEXCMD	<string>	N/A	A path to a LaTeX engine that generates DVI files.

$!LIMITS

Syntax:

```
$!LIMITS
  [optional parameters]
```

Description:

A SetValue command that sets some of the internal limits in Tecplot 360. See the User's Manual for additional information. The `$!LIMITS` command can only be used in the Tecplot 360 configuration file.

Optional Parameters

Parameter	Syntax	Default	Notes
LODTHRESHOLDMINFRACT	<op> <double>	= 0.3	When Load on demand is set to auto-unload, set the minimum and maximum memory thresholds. The values may be from 0 to 1, where .5 corresponds to a threshold of 50%. If Tecplot 360's memory usage exceeds the maximum threshold, it continues to unload variables until it either runs out of variables or reaches the minimum threshold. These values may also be set interactively via the Performance dialog. Refer to Miscellaneous Settings in the User's Manual for additional information.
LODTHRESHOLDMAXFRACT	<op> <double>	= 0.7	

Parameter	Syntax	Default	Notes
MAXAVAILABLEPROCESSORS	<op> <integer>	= 0	Restrict the number of processors (processor cores) employed by Tecplot 360 to the *numprocs* specified. Some tasks can be performed in parallel, so using all available processors greatly increases performance of those tasks. By default, Tecplot 360 uses all processors available on the machine to provide the best performance in most cases. Assign a value less than the total number of available processors to limit the number of processors used by Tecplot 360 to the assigned number.
MAXPTSINALINE	<op> <integer>		Maximum number of points for geometry polylines.
MAXCHRSINTEXTLABELS	<op> <integer>		Maximum number of characters in text labels.
MAXNUMCONTOURLEVELS	<op> <integer>		Maximum number of contour levels.
MAXNUMPICKOBJECTS	<op> <integer>		Maximum number of objects to pick.
MAXUSABLEMEMORY	<op> <integer>		Limit the amount of memory used by Tecplot 360 (units of MB).
PRELOADDATATIMETHRESHOLDINMS	<integer>		Sets the maximum time in milliseconds to spend pre-loading data.

Example:

Increase the maximum number of contour levels allowed to 1,000:

```
$!LIMITS
  MAXNUMCONTOURLEVELS = 1000
```

$!LINEARINTERPOLATE

Syntax:

```
$!LINEARINTERPOLATE
  DESTINATIONZONE = <integer>
  [optional parameters]
```

Description:

Interpolate selected variables from a set of source zones to a destination zone using linear interpolation. The source zones cannot be I-ordered. Values assigned to the destination zone are equivalent to the results of using the probe tool in Tecplot 360.

Required Parameter

Parameters	Syntax	Default	Notes
DESTINATIONZONE	= <integer>		Zone to interpolate to.

Optional Parameters

Parameters	Syntax	Default	Notes
SOURCEZONES	= <set>	All zones except the destination zone.	
VARLIST	= <varset>	All variables except spatial variables.	Choose the variables to interpolate. The spatial variables (X, Y and Z if 3D) are not allowed.
LINEARINTERPCONST	= <double>		Constant value if LINEARINTERPMODE is set to SETTOCONST. Otherwise not used.
LINEARINTERPMODE	= <linearinterpmode>	= DONTCHANGE	Choose how to treat points that lie outside the sourcezone data field. DONOTCHANGE - Preserves the values of points outside the data field. CONSTANT - Sets all points outside the data field to a constant value specified by LINEARINTERPCONST.

Example

Do linear interpolation from zones 2, 3 and 4 onto zone 7. Interpolate only variables 3-7:

```
$!LINEARINTERPOLATE
  SOURCEZONES     = [2-4]
  DESTINATIONZONE = 7
  VARLIST         = [3-7]
```

$!LINEMAP

Syntax:

```
$!LINEMAP  [<set>]
  [Optional Parameters]
```

Description:

A SetValue command that assigns attributes for individual line mappings. The <set> parameter immediately following the $!LINEMAP command is optional. If <set> is omitted then the assignment is applied to all line mappings, otherwise the assignment is applied only to the line mappings specified in <set>.

Optional Parameters

Parameter	Syntax	Default	Notes
ASSIGN			
{			
ZONE	= <integer>	1	
XAXISVAR	<op> <varref>	= 1	
YAXISVAR	<op> <varref>	= 2	
THETAAXISVAR	<op> <varref>		
RAXISVAR	<op> <varref>		
XAXIS	<op> < integer>	= 1	
YAXIS	<op> < integer>	= 1	
FUNCTIONDEPENDENCY	= <functiondependency>	XINDEPENDENT	
SHOWINLEGEND	= [ALWAYS,NEVER, AUTO]	AUTO	
SORT	<sortby>	NONE	
SORTVAR	= <varref>		
}			
BARCHARTS			

Parameter	Syntax	Default	Notes
{			
SHOW	= <boolean>	YES	
COLOR	= <color>	RED	
FILLMODE	= <fillmode>	USESPECIFICCOLOR	
FILLCOLOR	= <color>	RED	
SIZE	<op> <dexp>	= 2.5	
LINETHICKNESS	<op> <dexp>	= 0.4	
}			
CURVES			
{			
CURVETYPE	= <curvetype>	LINESEG	
EXTENDEDNAME	= <string>		Only used by the Extended Curve-fit Add-on.
EXTENDEDSETTINGS	= <string>		Only used by the Extended Curve-fit Add-on.
USEWEIGHTVAR	= <boolean>	NO	
NUMPTS	<op> <integer>	= 200	
POLYORDER	<op> <integer>	= 3	
WEIGHTVAR	= <varref>	0	
INDVARMIN	<op> <dexp>	-1 x 10^{150}	
INDVARMAX	<op> <dexp>	1 x 10^{150}	
USEINDVARRANGE	= <boolean>	NO	
CLAMPSPLINE	= <boolean>	NO	
SPLINEDERIVATIVEATSTART	<op> <dexp>	= 0	
SPLINEDERIVATIVEATEND	<op> <dexp>	= 0	
}			
ERRORBARS			
{			
SHOW	= <boolean>	NO	

Parameter	Syntax	Default	Notes
VAR	= <varref>		
BARTYPE	= <errorbartype>	VERT	
COLOR	= <color>	RED	
LINETHICKNESS	<op> <dexp>	= 0.1	
SKIPPING	<op> <dexp>	= 1	Skip can be by index or distance depending on SKIPMODE.
SKIPMODE	= <skipmode>	BYINDEX	
SIZE	<op> <dexp>	= 2.5	
}			
INDICES			The indices parameter is used to restrict the range of data plotted (and which lines are plotted if the data is IJ- or IJK-ordered).
{			
IJKLINES	= <ijklines>	I	
IRANGE	<<indexrange>>	MIN=1, MAX=0,SKIP=1	
JRANGE	<<indexrange>>	MIN=1, MAX=0, SKIP=1	
KRANGE	<<indexrange>>	MIN=1, MAX=0, SKIP=1	
}			
LINES			
{			
SHOW	= <boolean>	YES	
COLOR	= <color>	RED	
LINEPATTERN	= <linepattern>	SOLID	
PATTERNLENGTH	<op> <dexp>	= 2	
LINETHICKNESS	<op> <dexp>	= 0.1	
}			

Parameter	Syntax	Default	Notes
NAME	= <string>	'&DV&'	
SYMBOLS			
{			
SHOW	= <boolean>	YES	
COLOR	= <color>	RED	
FILLMODE	= <fillmode>	NONE	
FILLCOLOR	= <color>	RED	
SIZE	<op> <dexp>	= 2.5	
LINETHICKNESS	<op> <dexp>	= 0.1	
SKIPPING	<op> <dexp>	= 1	Skip can be by index or distance depending on SKIPMODE.
SKIPMODE	= <skipmode>	BYINDEX	
SYMBOLSHAPE	<<symbolshape>>	ISASCII = NO, GEOMSHAPE = SQUARE	
}			

Examples

Example 1:

Assign variable 1 to be on the X-axis and variable 4 to be on the Y-axis for Line-mapping number 7:

```
$!LINEMAP [7]
  ASSIGN
  {
    XAXISVAR = 1
    YAXISVAR = 4
  }
```

Example 2:

Make Error Bars red for all Line-mappings:

```
$!LINEMAP
  ERRORBARS
  {
    COLOR = RED
```

```
  }
```

Example 3:

Set Line-mappings 3-5 to draw a polynomial curve fit of order 5:

```
$!LINEMAP [3-5]
 CURVES
 {
  POLYORDER = 5
  CURVETYPE = CURVFIT
 }
 LINES
 {
  SHOW = YES
 }
```

$!LINEPLOTLAYERS

Syntax:

```
$!LINEPLOTLAYERS
  [optional parameters]
```

Description:

A SetValue command that turns on or off Line-plot layers.

Optional Parameters:

Parameter	Syntax	Default	Notes
SHOWLINES	= <boolean>	YES	
SHOWSYMBOLS	= <boolean>	NO	
SHOWBARCHARTS	= <boolean>	NO	
SHOWERRORBARS	= <boolean>	NO	Line-mapping must have an error bar variable assigned for this to have an effect.

Example:

Turn on the symbols layer for line plots:

```
$!LINEPLOTLAYERS
```

```
  SHOWSYMBOLS = YES
```

$!LINKING

Syntax:

```
$!LINKING
  [optional parameters]
```

Description:

Link attributes in two or more frames so that changes to attributes of one frame effect all linked frames.

Optional Parameters

Parameter	Syntax	Default	Notes
BETWEENFRAMES			
{			
LINKCONTOURLEVELS	= <boolean>	NO	
LINKFRAMESIZEANDPOSITION	= <boolean>	NO	
LINKXAXISRANGE	= <boolean>	NO	
LINKYAXISRANGE	= <boolean>	NO	
LINKPOLARVIEW	= <boolean>	NO	
LINK3DVIEW	= <boolean>	NO	
LINKGROUP	= <integer>	1	
LINKAXISPOSITION	= <boolean>	NO	
LINKVALUEBLANKING	= <boolean>	NO	
LINKSLICEPOSITIONS	= <boolean>	NO	
LINKISOSURFACEVALUES	= <boolean>	NO	
LINKSOLUTIONTIME	= <boolean>	NO	
}			
WITHINFRAME			

Parameter	Syntax	Default	Notes
{			
LINKAXISSTYLE	= <boolean>	NO	
LINKGRIDLINESTYLE	= <boolean>	NO	
LINKLAYERLINECOLOR	= <boolean>	NO	
LINKLAYERLINEPATTERN	= <boolean>	NO	
}			

Example:

The following example will set the link attribute for all frames in the layout to `LINK3DVIEW`.

```
$!LOOP |NUMFRAMES|
$!LINKING BETWEENFRAMES LINK3DVIEW = YES
$!FRAMECONTROL MOVETOBOTTOMBYNUMBER
$!FRAMECONTROL ATIVATETOP
$!ENDLOOP
```

$!LOADADDON

Syntax:

```
$!LOADADDON <string>
  [Optional Parameters]
```

Description:

Load an add-on into Tecplot 360. The <string> is the library name of the add-on to load. See the User's Manual for instructions on how to specify the add-on's library name.

Optional Parameters

Parameters	Syntax	Default	Notes
AddonStyle	= <string>	V7Standard	Style of the add-on to load. This can be either V7STANDARD or V7ACTIVEX. Usually not needed; applies to Windows only

Parameters	Syntax	Default	Notes
IsOptional	= <boolean>	False	False: If the add-on fails to load for any reason, Tecplot will post an error dialog indicating the reason for the failure and stop processing macro commands that follow the failing $!LoadAddOn command. True: If the add-on fails to load for any reason, Tecplot will print the reason for the failure to the Linux or Mac command line (Windows command line output is swallowed by the operating system) and Tecplot will continue to process macro commands that follow the failed $!LoadAddOn command as if the error did not occur.

Example:

Load the Key Frame Animation add-on, stored in a library named `tecutiltools_keyframe`.

```
$!LOADADDON "tecutiltools_keyframe"
```

$!LOADCOLORMAP

Syntax:

```
$!LOADCOLORMAP <string>
  [no parameters]
```

Description:

Load a color map file. The <string> is the name of the file to load.

Example

```
$!LOADCOLORMAP "mycolors.map"
```

$!LOOP⋯$!ENDLOOP

Syntax:

```
$!LOOP <integer>
```

```
$!ENDLOOP
```

Description:

Process macro commands in a loop. Within the loop you may access the current loop counter using the intrinsic macro variable |Loop|. Loops may be nested up to 10 levels deep.

Example:

Process macro commands 3 times over:

```
$!LOOP 3
.... macro commands ...
$!ENDLOOP
```

$!MACROFUNCTION···$!ENDMACROFUNCTION

Syntax:

```
$!MACROFUNCTION
  NAME = <string>
  [optional parameters]
$!ENDMACROFUNCTION
```

Description:

Define a macro function. All commands between a $!MACROFUNCTION and the $!ENDMACROFUNCTION are associated with the macro function NAME. These commands are not executed when they are defined but are executed when a $!RUNMACROFUNCTION command is processed. Parameters can be passed to a macro function. Use |n| to reference the *nth* parameter. (See $!RUNMACROFUNCTION. In the user-interface, [Ctrl]-M must be pressed, before using the keyboard shortcut specified by the KEYSTROKE option. For example, if the KEYSTROKE option is set to "P", the user must press [Ctrl]-M-P.

Required Parameter

Parameter	Syntax	Notes
NAME	= <string>	Name of the macro function.

Optional Parameters

Parameter	Syntax	Default	Notes
KEYSTROKE	= <string>		Allows keyboard shortcuts

Parameter	Syntax	Default	Notes
RETAIN	= <boolean>	NO	Set this to YES if you want Tecplot 360 to retain this macro function when the macro in which this macro function was defined terminates. If the macro function is retained then it can be called when another macro is loaded at a later time.
SHOWINMACROPANEL	= <boolean>	YES	Used only for macro functions within the tecplot.mcr file. Set this to NO if you do not want Tecplot 360 to include the macro function in Tecplot 360's Quick Macro Panel.

Example:

Define a macro function that redraws the active frame *n* times when <Ctrl>+M is hit and then the 'R' key is pressed, where *n* is passed to the macro function:

```
$!MACROFUNCTION
  NAME = "ABC"
  KEYSTROKE = "R"
  $!LOOP |n|
    $!REDRAW
  $!ENDLOOP
$!ENDMACROFUNCTION
```

$!NEWLAYOUT

Syntax:

```
$!NEWLAYOUT
  [no parameters]
```

Description:

Clear the current layout and start again.

When a new layout is created, the following steps occur:

1. All pages are deleted.
2. A new page is created.
3. A new frame is added to the page.

O-R

$!OPENLAYOUT

Syntax:

```
$!OPENLAYOUT <string>
  [optional parameters]
```

Description:

Open and read in a new layout file. The <string> is the name of the file to open.

Optional Parameters

Parameter	Syntax	Default	Notes
ALTDATALOADINSTRUCTIONS	= <string>	Null	Specify alternate data load instructions. Tecplot 360 data files: This is a list of filenames to use as replacements for data files referenced in the layout file. Use " to enclose file names that contain spaces or the + symbol. By default, separate file names listed in the ALTDATALOADINSTRUCTIONS are assigned to successive data sets that are referenced within a layout file. If you have a data set that references multiple data files, use the plus symbol, +, to group file names. Non-Tecplot 360 formats (including data being input via a data loader add-on): This is a list of instructions that are passed on to the loader.
APPEND	= <boolean>	NO	Set to NO if you want Tecplot 360 to delete the current layout prior to reading in the new one.

Examples

Example 1:

Open a new layout file called `abc.lay` and replace the data file referenced in the layout file with `t.plt`:

```
$!OPENLAYOUT "abc.lay"
```

```
  ALTDATALOADINSTRUCTIONS = "t.plt"
```

Example 2:

Open a new layout file called `multiframe.lay` and replace the first data set with `t.plt` and the second data set with the two files, `a.plt` and `b.plt`.

```
$!OPENLAYOUT "multiframe.lay"
  ALTDATALOADINSTRUCTIONS = '"t.plt" "a.plt"+"b.plt"'
```

$!PAGE

Syntax:

```
$!PAGE
  [optional parameters]
```

Description:

A SetValue command that sets the paper characteristics.

Optional Parameters

Parameter	Syntax	Default	Notes
PAPERATTRIBUTES			
{			
BACKGROUNDCOLOR	= <color>	WHITE	
ISTRANSPARENT	= <boolean>	YES	
ORIENTPORTRAIT	= <boolean>	NO	
PAPERGRIDSPACING	= <papergridspacing>	HALFINCH	Set the spacing for the tick marks on the paper.
PAPERSIZE	= <papersize>	LETTER	
REGIONINWORKAREA	<<rect>>	X1=1, Y1=0.25, X2=10, Y2=8.25	Specify rectangle that must fit within the workarea. Units are in inches (that is, in the paper coordinate system).
RULERSPACING	= <paperrulerspacing>	ONEINCH	

Parameter	Syntax	Default	Notes
SHOWGRID	= <boolean>	YES	
SHOWPAPER	= <boolean>	NO	
SHOWRULER	= <boolean>	NO	
}			
USESOFTWARERENDERING	= <boolean>		Use the painter's algorithm (meaning, display graphics cell-by-cell from front-to-back), instead of using OpenGL.
NAME	= <string>		Pages are unnamed by default.

Example:

This example does the following:

1. Turns off the paper grid.
2. Makes the paper size CUSTOM1.
3. Makes the dimensions for CUSTOM1 to be 4 by 5 inches.

```
$!PAGE
 PAPERATTRIBUTES
 {
   SHOWGRID = NO
   PAPERSIZE = CUSTOM1
   PAPERSIZEINFO
   {
     CUSTOM1
     {
      WIDTH = 4
      HEIGHT = 5
     }
   }
 }
```

See also: `$!GLOBALPAPER`

$!PAGECONTROL [Required-Control Option]

Description:

The different commands in the `PAGECONTROL` compound function family are described separately in the following sections.

The PAGECONTROL compound functions are:

$!PAGECONTROL CREATE
$!PAGECONTROL SETCURRENTTONEXT
$!PAGECONTROL SETCURRENTTOPREV
$!PAGECONTROL SETCURRENTBYNAME
$!PAGECONTROL DELETE
$!PAGECONTROL CLEAR

$!PAGECONTROL CREATE

Syntax:

```
$!PAGECONTROL CREATE
  [no parameters]
```

Description:

Create a new page. The created page will include an initial frame.

$!PAGECONTROL SETCURRENTTONEXT

Syntax:

```
$!PAGECONTROL SETCURRENTTONEXT
  [no parameters]
```

Description:

Set the next page to be the current page.

$!PAGECONTROL SETCURRENTTOPREV

Syntax:

```
$!PAGECONTROL SETCURRENTTOPREV
  [no parameters]
```

Description:

Set the previous page to be the current page.

$!PAGECONTROL SETCURRENTBYNAME

Syntax:

```
$!PAGECONTROL SETCURRENTBYNAME
  NAME = <string>
```

Description:

Set the current page to the page specified.

Example:

```
$!PAGECONTROL SETCURRENTBYNAME
  NAME = "BANANA"
```

$!PAGECONTROL DELETE

Syntax:

```
$!PAGECONTROL DELETE
```

Description:

Delete the current page. If the command is operated on the only page, then an initial page is created with an initial frame.

$!PAGECONTROL CLEAR

Syntax:

```
$!PAGECONTROL CLEAR
```

Description:

Clears all frames in the current page and creates a default initial frame.

$!PAGEGETNAME

Syntax:

```
$!PAGEGETNAME |MACROVAR|
```

```
  [no parameters]
```

Description:

Action command to get the name of the current page.

$!PAGENAME

Syntax:

```
$!PAGENAME <string>
  [no parameters]
```

Description:

Set the name of the page.

$!PAUSE

Syntax:

```
$!PAUSE <string>
 [no parameters]
```

Description:

Stop execution of a macro and optionally display a dialog with a message. If <string> is set to "" then no dialog is displayed and the user must click in the work area to continue.

Example

Pause and display the message: "This is the first example plot":

```
$!PAUSE "This is the first example plot."
```

$!PICK [Required-Control Option]

Description:

The different commands in the PICK compound function family are described separately in the following sections.

The PICK compound functions are:

$!PICK ADDATPOSITION
$!PICK ADDALL
$!PICK ADDALLINRECT
$!PICK CLEAR
$!PICK COPY
$!PICK CUT
$!PICK EDIT
$!PICK MAGNIFY
$!PICK PASTE
$!PICK POP
$!PICK PUSH
$!PICK SETMOUSEMODE
$!PICK SHIFT

$!PICK ADDATPOSITION

Syntax:

```
$!PICK ADDATPOSITION
  X = <dexp>
  Y = <dexp>
  [optional parameters]
```

Description:

Attempt to pick an object at a specific location on the paper. Does not pop or activate frames.

Required Parameters

Parameters	Syntax	Default	Notes
X	= <dexp>		X-location (in inches) relative to the left edge of the paper.
Y	= <dexp>		Y-location (in inches) relative to the top edge of the paper.

Optional Parameters

Parameters	Syntax	Default	Notes
COLLECTINGOBJECTS	= <boolean>	NO	If NO, the list of picked objects is cleared before the attempt is made to add a new object.

Parameters	Syntax	Default	Notes
COLLECTINGOBJECTSMODE	= <collectingobjectsmode>		**InvertingAdd**: Picking an object that is not selected adds the object to the collection of selected objects. Picking an object that is selected removes the object from the collection of selected objects. Picking in a region of the work area where there is nothing to select leaves the collection of selected objects unchanged. **AlwaysAdd**: Picking an object that is not selected clears the collection of selected objects and adds the newly selected item to the collection of selected objects. Picking an object that is selected leaves the selected object and all other objects in the collection of selected objects unchanged. Picking in a region of the work area where there is nothing to select clears the collection of selected objects. **HomogeneousAdd**: Picking an object that is not selected clears the collection of selected objects and adds the newly selected item to the collection of selected objects. Picking an object that is selected leaves the selected object and all other objects of the same type in the collection of selected objects unchanged. Objects of differing types are removed from the collection of selected objects. Picking in a region of the work area where there is nothing to select clears the collection of selected objects.
CONSIDERSTYLE	= <boolean>	NO	If YES, objects that are not being drawn, or which are styled such that they are not visible, are ignored.
DIGGINGFOROBJECTS	= <boolean>	NO	If YES, attempt to pick objects below any currently picked objects at this location.
IGNOREZONEOBJECTS	= <boolean>	NO	If YES, pick operations will ignore zones and pick objects such as slices, iso-surfaces and streamtraces.

Example

Attempt to add to the list of picked objects by picking at paper location (1.0, 7.0). Do not clear the list of picked objects before picking:

```
$!PICK ADDATPOSITION
  X = 1.0
  Y = 7.0
  COLLECTINGOBJECTS = YES
```

$!PICK ADDALL

Syntax:

```
$!PICK ADDALL
  [optional parameters]
```

Description:

Add all objects of a certain type to the list of picked objects.

Optional Parameters

Parameters	Syntax	Default	Notes
CONSIDERSTYLE	= <boolean>	NO	If YES, objects that are not being drawn, or which are styled such that they are not visible, are ignored.
SELECTTEXT	= <boolean>	NO	Select all text objects in the active frame.
SELECTGEOMS	= <boolean>	NO	Select all geometry objects in the active frame.
SELECTFRAMES	= <boolean>	NO	Select all frames.
SELECTSTREAMTRACES	= <boolean>	NO	Select all streamtrace objects in the active frame.
SELECTMAPS	= <boolean>	NO	Select all line map objects in the active frame.
SELECTZONES	= <boolean>	NO	Select all zone objects in the active frame.

Example:

Add all text and geometries in the active frame to the list of picked objects:

```
$!PICK ADDALL
  SELECTTEXT = YES
```

```
  SELECTGEOMS = YES
```

$!PICK ADDALLINRECT

Syntax:

```
$!PICK ADDALLINRECT
  X1 = <dexp>
  Y1 = <dexp>
  X2 = <dexp>
  Y2 = <dexp>
  [optional parameters]
```

Description:

Add objects defined within a specified region to the list of picked objects. The region is defined in terms of the paper coordinate system. Optional filters can be used to restrict the objects selected. The region is defined by the two corner points (X1, Y1) and (X2, Y2).

Required Parameters

Parameters	Syntax	Default	Notes
X1	= <dexp>		X-location (in inches) relative to the left edge of the paper.
Y1	= <dexp>		Y-location (in inches) relative to the top edge of the paper.
X2	= <dexp>		X-location (in inches) relative to the left edge of the paper.
Y2	= <dexp>		Y-location (in inches) relative to the top edge of the paper.

Optional Parameters

Parameters	Syntax	Default	Notes
COLORFILTER	= <color>	Not used[1].	Only objects of this color will be selected.
CONSIDERSTYLE	= <boolean>	NO	If YES, objects that are not being drawn, or which are styled such that they are not visible, are ignored.
FONTFAMILYFILTER	= <string>	Not used[1].	Only text objects with this font will be selected.
ISBOLD	= <boolean>	NO	Only boldface text objects will be selected.

Parameters	Syntax	Default	Notes
ISITALIC	= <boolean>	NO	Only italicized text objects will be selected.
GEOMFILTER	= <geomtype>	Not used[1].	Only geometry objects of this type will be selected.
LINEPATTERNFILTER	= <linepattern>	Not used[1].	Only geometry objects with this line pattern will be selected.
SELECTCONTOURLABELS	= <boolean>	NO	Select all contour labels in specified region
SELECTFRAMES	= <boolean>	NO	Select all frame objects in the specified region.
SELECTGEOMS	= <boolean>	NO	Select all geometry objects in the specified region.
SELECTGRIDAREA	= <boolean>	NO	Select the grid area in specified region
SELECTMAPS	= <boolean>	NO	Select all line map objects in the specified region.
SELECT STREAMTRACES	= <boolean>	NO	Select all streamtrace objects in the specified region.
SELECTTEXT	= <boolean>	NO	Select all text objects in the specified region.
SELECTZONES	= <boolean>	NO	Select all zone objects in the specified region.

[1] There is no default for this parameter. If this parameter is omitted then the orresponding filter is not used.

Example:

Pick all circles using a dashed line pattern within the rectangle bounded by the points (0, 0) and (3, 5):

```
$!PICK ADDALLINRECT
  SELECTGEOMS = YES
  LINEPATTERNFILTER = DASHED
  GEOMFILTER = CIRCLE
  X1 = 0
  Y1 = 0
  X2 = 3
  Y2 = 5
```

$!PICK CLEAR

Syntax:

```
$!PICK CLEAR
  [no parameters]
```

Description:

Delete all objects that are currently picked. (These objects cannot be retrieved.)

Example

```
$!PICK CLEAR
```

$!PICK COPY

Syntax:

```
$!PICK COPY
  [no parameters]
```

Description:

Copy all objects that are currently picked to the paste buffer.

Example

```
$!PICK COPY
```

$!PICK CUT

Syntax:

```
$!PICK CUT
  [no parameters]
```

Description:

Copy all objects that are currently picked to the paste buffer and then delete them.

Example:

```
$!PICK CUT
```

$!PICK EDIT

Syntax:

```
$!PICK EDIT
  [parameters]
```

Description:

Perform a global edit operation on the currently picked objects. Only one edit operation is allowed per `$!PICK EDIT` command. Objects are edited only if the supplied parameter is relevant. Actions taken using the **Quick Edit** dialog in Tecplot 360 generate these commands.

Parameters

Must select one from this table.

Parameters	Syntax	Default	Notes
ANGLE	= <dexp>		Angle in degrees.
ARROWHEADANGLE	= <dexp>		Angle is in degrees.
ARROWHEADATTACHMENT	= <arrowheadattachment>		
ARROWHEADSIZE	= <dexp>		Value is in Y-frame units (0-100).
ARROWHEADSTYLE	<arrowheadstyle>		
ASCIICHAR	= <string>		
BARCHARTS			Only operates on XY line mapping objects.
{			
SHOW	= <boolean>		
ISFILLED	= <boolean>		
}			
COLOR	= <color>		
CONTOUR			Only operates on 2D or 3D zone objects.
{			

Parameters	Syntax	Default	Notes
SHOW	= <boolean>		
CONTOURTYPE	= <contourtype>		
}			
CURVES			Only operates on XY line mapping objects.
{			
CURVETYPE	= <curvetype>		
}			
EDGELAYER			Only operates on 2D or 3D zone objects.
{			
SHOW	= <boolean>		
SUBBOUNDARY	= <subboundary>		
}			
ERRORBARS			Only operates on XY line mapping objects.
{			
SHOW	= <boolean>		
BARTYPE	= <errorbartype>		
}			
FILLCOLOR	= <color>		
FONTFAMILY	= <string>		
GEOMSHAPE	= <geomshape>		Applies only to scatter symbols or XY-plot symbols.
ISBOLD	= <boolean>		
ISFILLED	= <boolean>		Applies only to geometries.
ISITALIC	= <boolean>		
LINEPATTERN	= <linepattern>		
LINES			Only operates on XY line mapping objects.
{			

Parameters	Syntax	Default	Notes
SHOW	= <boolean>		
}			
LINETHICKNESS	= <dexp>		Value is in Y-frame units (0-100).
MESH			Only operates on 2D or 3D zone objects.
{			
SHOW	= <boolean>		
MESHTYPE	= <meshtype>		Only operates on 2D or 3D zone objects.
}			
OBJECTALIGN	= <objectalign>		Only allowed if selected objects are all text and/or geometries.
PATTERNLENGTH	= <dexp>		Value is in Y-frame units (0-100).
SCATTER			Only operates on 2D or 3D zone objects.
{			
SHOW	= <boolean>		
FILLMODE	= <fillmode>		
}			
SHADE			Only operates on 2D or 3D zone objects.
{			
SHOW	= <boolean>		
SHADETYPE	= <shadetype>		
}			
SHOWBORDER	= <boolean>		Only operates on frame objects.
SIZE	= <dexp>		Value is in Y-frame units. This applies to things like symbols.
SYMBOLS			Only operates on line mapping objects.
{			
SHOW	= <boolean>		
ISFILLED	= <boolean>		
}			
TEXTCOLOR	= <color>		
TEXTHEIGHTBYPERCENT	= <dexp>		Value is in Y-frame units (0-100).

Parameters	Syntax	Default	Notes
TEXTHEIGHTBYPOINTS	= <dexp>		Value is in points.
VECTOR			Only operates on 2D or 3D zone objects.
{			
SHOW	= <boolean>		
VECTORTYPE	= <vectortype>		
}			

Examples

Example 1:

Set all picked objects to use the color yellow:

```
$!PICK EDIT
  COLOR = YELLOW
```

Example 2:

Set all picked objects to use the dashed line pattern:

```
$!PICK EDIT
  LINEPATTERN = DASHED
```

Example 3:

Set all picked objects (which are zones) to use the contour plot type of flooding:

```
$!PICK EDIT
  CONTOUR {CONTOURTYPE = FLOOD}
```

$!PICK MAGNIFY

Syntax:

```
$!PICK MAGNIFY
  MAG = <dexp>
```

Description:

Magnify all picked objects. The objects will also be translated proportional to the distance between their anchor position and the anchor position of the first object picked.

Example:

Magnify all objects by 1.5:

```
$!PICK MAGNIFY
  MAG = 1.5
```

$!PICK PASTE

Syntax:

```
$!PICK PASTE
  [no parameters]
```

Description:

Paste the currently picked objects from the paste buffer to the work area.

$!PICK POP

Syntax:

```
$!PICK POP
  [no parameters]
```

Description:

Change the order in which objects are drawn by popping the currently picked objects to the front. Only frames, text, geometries, and the grid area for 2D plots are allowed.

$!PICK PUSH

Syntax:

```
$!PICK PUSH
  [no parameters]
```

Description:

Change the order in which objects are drawn by pushing the currently picked objects back. Only frames, text, geometries, and the grid area for 2D plots are allowed.

$!PICK SETMOUSEMODE

Syntax:

```
$!PICK SETMOUSEMODE
  MOUSEMODE = <mousemode>
```

Description:

Prepare to pick objects by setting the mouse mode to SELECT, ADJUST, or ADVANCEADJUST. ADVANCEADJUST is a combination of SELECT and ADJUST that allows an object's control points to be adjusted by dragging, or the entire object to be selected by clicking away from the control points. This command also clears the list of picked objects (that is, unpicks all picked objects).

Required Parameter

Parameter	Syntax	Default	Notes
MOUSEMODE	= <mousemode>		Set to SELECT or ADJUST.

Example

Set the mouse mode so picked objects are adjusted:

```
$!PICK SETMOUSEMODE
  MOUSEMODE = ADJUST
```

$!PICK SHIFT

Syntax:

```
$!PICK SHIFT
  X = <dexp>
  Y = <dexp>
  [optional parameters]
```

Description:

Shift the currently picked objects that are allowed to shift. Picked objects such as zones or linemaps cannot be moved when using the selector mouse mode (these can however be moved when using the adjuster mouse mode).

Objects are shifted relative to their starting position. X and Y shift amounts are in paper units (inches). If snapping is in effect then it is applied after shifting in X and Y. (See the SetValue commands for SNAPTOGRID and SNAPTOPAPER in `$!GLOBALFRAME`.

Required Parameters

Parameters	Syntax	Default	Notes
X	= <dexp>		Shift amount in the X-direction. Units are inches.
Y	= <dexp>		Shift amount in the Y-direction. Units are inches.

Optional Parameter

Parameters	Syntax	Default	Notes
POINTERSTYLE	= <pointerstyle>	ALLDIRECTIONS	Only frames and non-3D grid area objects can use a pointer style that is not ALLDIRECTIONS.

Example

Shift the currently picked objects 1 inch to the right and 2 inches down:

```
$!PICK SHIFT
  X = 1
  Y = 2
```

$!PLOTTYPE

Syntax:

```
$!PLOTTYPE = <plottype>
  [no parameters]
```

Description:

Changes plot types between valid Tecplot 360 modes such as XYLine and Cartesian2D. Valid options shown below.

Required Parameters

Parameter	Syntax	Default	Notes
PLOTTYPE	= <plottype>	CARTESIAN3D	

Example

Change the plot style to show a polar plot:

```
$!PLOTTYPE = POLARLINE
```

$!POLARAXIS

Syntax:

```
$!POLARAXIS
  [optional parameters]
```

Description:

A SetValue command that assigns attributes for axes in a polar frame.

Optional Parameters

Parameter	Syntax	Default	Notes
GRIDAREA	<<areastyle>>	See Notes	DRAWBORDER=YES, ISFILLED=NO, FILLCOLOR=WHITE, DRAWGRIDLAST=NO
PRECISEGRID	<<precisegrid>>		INCLUDE=NO, SIZE=0.0045, COLOR=BLACK
PRESERVEAXISSCALE	<boolean>	YES	
RDETAIL	<<axisdetail>>		
THETADETAIL	<<axisdetail>>		
THETAMODE	= <thetamode>	DEGREES	
THETAPERIOD	= <double>	360	
VIEWPORTPOSITION	<<rect>>	See Notes	X1=0, Y1=0, X2=100, Y2=100
VIEWPORTSTYLE	<<areastyle>>	See Notes	DRAWBORDER=NO, COLOR=BLACK, LINETHICKNESS=0.4, ISFILLED=NO, FILLCOLOR=WHITE

Example:

Set the Theta range, in Radians, from Pi to -Pi.

```
$!POLARAXIS THETAMODE = RADIANS
$!POLARAXIS THETAPERIOD = 6.28318530718
$!POLARAXIS THETADETAIL{VALUEATORIGIN = 0}
$!POLARAXIS THETADETAIL{RANGEMIN = -3.14159265359}
```

$!POLARTORECTANGULAR

Syntax:

```
$!POLARTORECTANGULAR <set>
  [no parameters]
```

Description:

Treat the variables currently assigned to X and Y as referring to R and q and convert them to X and Y. In 3D, X, Y and Z refer to R, q, and y. Tecplot 360 has additional capabilities for transforming coordinates, please see $!TRANSFORMCOORDINATES.

Example:

Convert zones 1, 2 and 3 from polar to rectangular:

```
$!POLARTORECTANGULAR [1-3]
```

$!POLARVIEW

Syntax:

```
$!POLARVIEW
  EXTENTS <<rect>>
```

Description:

Sets the viewing style for polar plots in a layout.

Required Parameters

Parameter	Syntax	Default	Notes
EXTENTS	<<rect>>	X1=-1.29771, Y1=-1.15352, X2=1.29771, Y2=1.15352	View extents of transformed X & Y in polar plots. Numbers listed are in the form of grid units.

Example

Set the view of the polar plot to view the full extents of the plot area.

```
$!POLARVIEW
  EXTENTS
    {
      X1=10
      Y1=10
      X2=90
      Y2=90
    }
```

$!POLARWATERMARK

Syntax:

```
$!POLARWATERMARK
  [Optional Parameters]
```

Description

Set the size and placement of watermarks for Polar plots for a specific frame.

Optional Parameters

Parameter	Syntax	Default	Notes
ANCHORALIGNMENT	= <anchoralignment>	BOTTOMRIGHT	Sets the origin location on the watermark itself. Similar to legends.
XYPOS	<<xy>>	X = 99, Y = 1	Location of watermark origin.
WIDTH	= <double>	15	Width of the watermark in X Frame units (0-100).

$!PRINT

Syntax:

```
$!PRINT
  [no parameters]
```

Description

Print the current layout to a printer or send the print instructions to a file. Use the $!PRINTSETUP SetValue command to configure printing.

Example:

```
$!PRINT
```

$!PRINTSETUP

Syntax:

```
$!PRINTSETUP
  [Optional Parameters]
```

Description:

A SetValue command that sets the attributes for printing. Use $!PRINT to do the actual printing. See $!EXPORTSETUP and $!EXPORT if you intend to create image files destined for desktop publishing programs.

Optional Parameters

Parameter	Syntax	Default	Notes
DRIVER	= <printerdriver>		Only applies if using the Tecplot 360 printer drivers. See USETECPLOTPRINTDRIVERS in $!INTERFACE.
FORCEEXTRA3DSORTING	= <boolean>		

Parameter	Syntax	Default	Notes
JOBCONTROL			These strings contain characters to be sent at the beginning and ending of a print file. These strings most often contain escape sequences used to switch modes on the printer. Non-printable characters can be inserted. Use ^nnn to insert a character with ordinal value nnn. Use \ to force the character after the \ to be inserted. Use $B for a Backspace, $E for Esc, $C for a carriage return, and $X for the Delete key.
{			
POSTMOPUPSTR	= <string>		
LGMOPUPSTR	= <string>		
POSTSETUPSTR	= <string>		
LGSETUPSTR	= <string>		
}			
IMAGERESOLUTION	= <integer>		
NUMHARDCOPYCOPIES	<op> <integer>		Applies only when DRIVER = PS.
NUMLIGHTSOURCESHADES	= <integer>		
PALETTE	= <palette>		Must choose options valid for current DRIVER setting.
PRECISION	<op> <integer>		Applies only if EXPORTFORMAT is PS, EPS, or RASTERMETAFILE.
PRINTFNAME	= <string>		Name of the file to write to if SENDPRINTTOFILE is YES.
PRINTRENDERTYPE	= <printrendertype>		
RGBLEGENDOUTPUTRESOLUTION	= <integer>	50	Determines the number of triangles which compose the bottom layer of the RGB Legend. This option is only available through macro language (for example, the config file)
SENDPRINTTOFILE	= <boolean>		If YES then PRINTFNAME is name of file to write to.
SPOOLER			These strings contain the system command needed to send a file to the print spooler on your computer. Use the @ symbol as a place holder for where you normally insert the name of the file to be printed. For security reasons these commands can only be used in the Tecplot 360 configuration file.
{			
PSMONOSPOOLCMD	= <string>		
PSCOLORSPOOLCMD	= <string>		
LGSPOOLCMD	= <string>		
}			

Parameter	Syntax	Default	Notes
USEISOLATIN1FONTSINPS	= <boolean>		Use extended ISO-Latin1 fonts when generating PostScript® output using Tecplot 360's internal PostScript driver.

Example:

This example does the following:

1. Instruct Tecplot 360 to send print output to the print spooler.
2. Sets the spooler command for monochrome PostScript to be `lpr @`.
3. Sets the print driver to be monochrome PostScript.

```
$!PRINTSETUP
  SENDPRINTTOFILE = NO
  DRIVER = PS
  PALETTE = MONOCHROME
  SPOOLER
  {
    PSMONOSPOOLCMD = "lpr @"
  }
```

$!PROMPTFORFILENAME

Syntax:

```
$!PROMPTFORFILENAME <macrovar>
  DIALOGTITLE = <string>
  DEFAULTFNAME = <string>
  FILEFILTER = <string>
```

Description:

Instruct Tecplot 360 to launch a file selection dialog. The resulting file name will be placed in <macrovar>. If the user cancels out of the dialog then <macrovar> will be empty (see the example below).

Optional Parameters

Parameter	Syntax	Default	Notes
DIALOGTITLE	= <string>	""	Include a title at the top of the dialog.

Parameter	Syntax	Default	Notes
DEFAULTFNAME	= <string>	""	Make the dialog come up with a default file name.
FILEFILTER	= <string>	""	Set the filter for the file selection dialog.
FILEMUSTEXIST	= <string>	YES	

Example:

Prompt the user for the name of a file to delete:

```
$!PROMPTFORFILENAME|filetodelete|
  DIALOGTITLE = "Delete File"
  FILEFILTER = "*.*"
  $!IF "|filetodelete|" != ""
    $!IF |OPSys| = 1 # Linux/Mac
      $!System "rm |filetodelete|"
    $!Endif
    $!IF |OPSys| = 2 # Windows
      $!System "cmd /c del |filetodelete|"
    $!Endif
  $!Endif
```

$!PROMPTFORTEXTSTRING

Syntax:

```
$!PROMPTFORTEXTSTRING <macrovar>
  INSTRUCTIONS = <string>
```

Description:

Instruct Tecplot 360 to launch a dialog containing a single line text field and optional instructions. The user enters text into the text field and the resulting string is assigned to <macrovar>.

Optional Parameter

Parameter	Syntax	Default	Notes
INSTRUCTIONS	= <string>	""	Include text at the top of the dialog to instruct the user regarding the value to enter. On Windows platforms, this is limited to three lines of text.

Example:

```
$!PROMPTFORTEXTSTRING |timestring|
  INSTRUCTIONS = "Enter the time of the experiment"
```

$!PROMPTFORYESNO

Syntax:

```
$!PROMPTFORYESNO <macrovar>
  INSTRUCTIONS = <string>
```

Description:

Instruct Tecplot 360 to launch a dialog containing two buttons, one labeled Yes and the other No. The <macrovar> is assigned the string Yes or No depending on the selection.

Optional Parameter

Parameter	Syntax	Default	Notes
INSTRUCTIONS	= <string>	""	Include text at the top of the dialog with instructions.

Example:

```
$!PROMPTFORYESNO |goforit|
  INSTRUCTIONS = "Do you want to go for it?"
  $!IF "|goforit|" == "YES"
  ... code that goes for it....
  $!ENDIF
```

$!PROPAGATELINKING

Syntax:

```
$!PROPAGATELINKING
  [Optional Parameters]
```

Description:

Link multiple frames, either within frame or between frames.

Optional Parameters

Parameter	Syntax	Default	Notes
FRAMECOLLECTION	= [ALL, PICKED]		
LINKTYPE	= [WITHINFRAME, BETWEENFRAMES]		

Example:

```
$!PROPAGATELINKING
  LINKTYPE = BETWEENFRAMES
  FRAMECOLLECTION = ALL
```

$!PUBLISH

Syntax:

```
$!PUBLISH <string>
```

Description:

Create an HTML file displaying one or more images. A linked layout with packaged data may be included. You must provide the file name.

Parameter	Syntax	Default	Notes
IMAGESELECTION	= <imagestyle>	ONEPERFRAME	Choosing ONEPERFRAME will create one image per frame, selecting WORKSPACEONLY creates one image which includes all your frames.
INCLUDELAYOUTPACKAGE	= <boolean>	No	Choose YES to create a linked layout file.

Optional Parameters

Example:

```
$!PUBLISH "C:\TEC360\separate.html"
```

```
  INCLUDELAYOUTPACKAGE = NO
  IMAGESELECTION = ONEPERFRAME
```

$!QUIT

Syntax:

```
$!QUIT
  [no parameters]
```

Description:

Terminate the execution of the Tecplot 360 program.

Example:

```
$!QUIT
```

$!RAWCOLORMAP

Syntax:

```
$!RAWCOLORMAP  <colormaprawdata>
```

Description:

Assign the RGB values that define the Raw user-defined color map. This does not choose the Raw user-defined color map for use in a contour group. Use `$!GLOBALCONTOUR COLORMAPNAME` to set the color map used by a contour group.

Required Parameter

Parameter	Syntax	Default	Notes
<colormaprawdata>			This is a list of RGB values.

Example

Assign the Raw user-defined color map to a gray scale using 11 colors:

```
$!RAWCOLORMAP
RAWDATA
11
```

```
0    0    0
25    25    25
50    50    50
75    75    75
100    100    100
125    125    125
150    150    150
175    175    175
200    200    200
225    225    225
255    255    255
```

$!READDATASET

Syntax:

```
$!READDATASET <string>
  [Optional Parameters]
```

Description:

The $!READDATASET macro command has two separate uses. The parameters available for the command are dependent upon the intended use. It may either be used to load data in Tecplot 360's file format (*.plt or *.dat) or in a foreign data file format. To load data in Tecplot 360's file format, use the parameters listed in Table 2. To load data in a foreign file format, use the parameters listed in Table 1 along with a set of name/value pairs. The name/value pairs are specific to the data loader and described in the User's Manual.

Examples

FLUENT® Loader Example:

The following example loads one case file and one data file with the FLUENT file loader. Note that the DATASETREADER parameter is at the end of the command call.

```
$!READDATASET '"STANDARDSYNTAX" "1.0" "LoadOption" "MultipleCaseAndData" "FILELIST_Files"
"2" "triangular.cas" "triangular.dat" "UnsteadyOption" "ReadTimeFromDataFiles"
"AssignStrandIDs" "Yes" "GridZones" "CellsAndBoundaries" "IncludeParticleData" "Yes"
"AverageToNodes" "Yes" "AveragingMethod" "Arithmetic"'
  DATASETREADER = 'Fluent Data Loader'
```

Ensight Loader Example

The following example loads one Ensight case file. Note that the DATASETREADER parameter is at the

end of the command call.

```
$!READDATASET '"STANDARDSYNTAX" "1.0" "FILENAME_CASEFILE" "wing.case" "ISkip" "1" "JSkip"
"1" "KSkip" "1"'
  DATASETREADER = 'EnSight Loader'
```

Table 1. Parameters for loading data in a foreign file format

Parameters	Syntax	Default	Notes
DATASETREADER	= <string>		Used to specify an alternate data reader for Tecplot 360.

Table 2. Parameters for loading data in Tecplot format

Parameters	Syntax	Default	Notes
ADDZONESTOEXISTINGSTRANDS	= <boolean>	NO	If YES, Tecplot 360 will add the zones from the appended data to any existing strands in the dataset. If NO, Tecplot 360 will append the strands from the appended data to any existing strands in the dataset.
ASSIGNSTRANDIDS	= <boolean>	NO	If YES, Tecplot 360 will assign strand ID's to zones if time is supplied for the zones but strand ID's are not. If NO, Tecplot 360 will not associate these zones with any strands.
IJKSKIP			Use values greater than 1 to skip data points.
{			
I	= <integer>	1	
J	= <integer>	1	
K	= <integer>	1	
}			
COLLAPSEZONESANDVARS	= <boolean>	NO	Renumber zones and variables if zones or variables are disabled.
INCLUDECUSTOMLABELS	= <boolean>	YES	Set to YES to load in any custom labels in the data files.
INCLUDEDATA	= <boolean>	YES	Set to YES to load in any field data in the data files.
INCLUDEGEOM	= <boolean>	YES	Set to YES to load in any geometries in the data files.
INCLUDETEXT	= <boolean>	YES	Set to YES to load in any text in the data files.

Parameters	Syntax	Default	Notes
INITIALPLOTFIRSTZONEONLY	= <boolean>		Allows faster performance for files with multiple zones.
INITIALPLOTTYPE	= <plottype>		
READDATAOPTION	= <readdataoption>	NEW	Set to APPEND to append the new zones to the zones in the data set that existed prior to using this command. Set to NEW to remove the data set from the active frame prior to reading in the new data set. If other frames use the same data set they will continue to use the old one. Set to REPLACE to replace the data set attached to the active frame and to all other frames that use the same data set, with the new data set.
RESETSTYLE	= <boolean>	YES	Set to NO if you want Tecplot 360 to keep the current style. This only applies if READDATAOPTION is set to REPLACE. When RESETSTYLE is set to NO, the prior style is matched to the incoming data as best as possible and in some cases some style settings may need to be turned off or reassigned. The option $!COMPATIBILITY USENAMESFORVARIABLEASSIGNMENTS, when set to TRUE will make style settings match variable assignments based on the variable names, regardless of their position in the old and new datasets. If $!COMPATIBILITY USENAMESFORVARIABLEASSIGNMENTS is set to FALSE then the variable offsets from the old style assignments will be used with the new data regardless of the old and new variable names.
VARLOADMODE	= <varloadmode>	BYPOSITION	Set to BYPOSITION to load variables based on their position in the file. Set to BYNAME to load variables based on their name. If set to BYNAME, then VARNAMELIST must be supplied as well except when appending.

Parameters	Syntax	Default	Notes
VARNAMELIST	= <string>		Use this to list the names of the variables to load into Tecplot 360. Names separated by a ; or a + are joined together to form a set of aliases for a given variable. If you are appending data then this can be omitted and in that case all variables from the incoming dataset will be loaded and associated with existing variables and all variables not in the existing dataset will be added to the end of the list.
VARPOSITIONLIST	= <varset>	All vars.	Use this to reduce the number of variables loaded.
ZONELIST	= <set>	All zones.	Use this to reduce the number of zones loaded.

Example 1:

Read in the data files `t1.plt` and `t2.plt` to form a single data set in Tecplot 360:

```
$!READDATASET "t1.plt t2.plt"
```

Example 2:

Read in the datafile `t1.plt`. Only read in zones 1 and 4. Skip over every other I-index:

```
$!READDATASET "t1.plt"
  ZONELIST = [1,4]
  IJKSKIP
  {
    I = 2
  }
```

Example 3:

Read in the data files `t1.plt`, `t2.plt`, and `t3.plt`. Append the new data set to the current one:

```
$!READDATASET "t1.plt t2.plt t3.plt"
READDATAOPTION = APPEND
```

Example 4:

Read in the data files `t1.plt` and `t2.plt` from directory, `/users/john/testrun7/runb`:

```
$!VARSET |BASEDIR| = "/users/john/testrun7/runb"
$!READDATASET "|basedir|/t1.plt |basedir|/t2.plt"
```

$!READSTYLESHEET

Syntax:

```
$!READSTYLESHEET <string>
  [Optional Parameters]
```

Description:

Read in a stylesheet file. The <string> is the name of the file to read.

Optional Parameters

Parameters	Syntax	Default	Notes
INCLUDEAUXDATA	= <boolean>	YES	Set to YES to read auxiliary data#.
INCLUDECONTOURLEVELS	= <boolean>	YES	Set to YES to read in all contour levels.
INCLUDEFRAMESIZEANDPOSITION	= <boolean>	NO	Set to YES if you want the active frame to be sized and positioned exactly like the frame used to create the stylesheet.
INCLUDEGEOM	= <boolean>	YES	Set to YES to load in any geometries in the stylesheet file.
INCLUDEPLOTSTYLE	= <boolean>	YES	Set to YES to process commands related to plot style (mesh color, vector type, and so on).
INCLUDESTREAMPOSITIONS	= <boolean>	YES	Set to YES to read in streamtrace starting positions.
INCLUDETEXT	= <boolean>	YES	Set to YES to load in any text in the stylesheet file.
MERGE	= <boolean>	NO	Set to NO to reset all frame attributes back to their factory defaults prior to reading in the stylesheet.

Example

Read the stylesheet file `t.sty`. Do not read in any text or geometries:

```
$!READSTYLESHEET "t.sty"
  INCLUDETEXT     = NO
  INCLUDEGEOM     = NO
```

$!REDISTRIBUTECOLORMAPCONTROLPOINTS

Syntax:

```
$!REDISTRIBUTECOLORMAPCONTROLPOINTS <string>
```

Description:

Redistributes control points for the named color map, which must exist and must be a custom color map (not a built-in color map such as "Small Rainbow").

$!REDRAW

Syntax:

```
$!REDRAW
  [Optional Parameters]
```

Description:

Redraw the active frame.

Optional Parameter

Parameter	Syntax	Default	Notes
DOFULLDRAWING	= <boolean>	YES	Set to NO to draw only a "trace" of the data in the frame.

Example

```
$!REDRAW
```

$!REDRAWALL

Syntax:

```
$!REDRAWALL
  [Optional Parameters]
```

Description:

Redraw all frames.

Optional Parameter

Parameter	Syntax	Default	Notes
DOFULLDRAWING	= <boolean>	YES	Set to NO to draw only a "trace" of the data in each frame.

Example

```
$!REDRAWALL
```

$!REMOVEVAR

Syntax:

```
$!REMOVEVAR <macrouserdefvar>
```

Description:

Remove a user-defined macro variable. This frees up space so another user-defined macro variable can be defined.

Example:

Remove the macro variable `|ABC|`:

```
$!REMOVEVAR |ABC|
```

$!RENAMECOLORMAP

Syntax:

```
$!RENAMECOLORMAP
  OLDNAME  = <string>
  NAME     = <string>
  [no optional parameters]
```

Description:

Renames a custom color map.

Required Parameters

Parameter	Syntax	Default	Notes
OLDNAME	= <string>		The name of the color map to be renamed. The named color map must exist.
NAME	= <string>		The new name for the color map. May not be the same as an existing color map name (including built-in color maps).

Example

Rename My Small Rainbow to My New Small Rainbow.

```
$!RENAMECOLORMAP
  OLDNAME = "My Small Rainbow"
  NAME = "My New Small Rainbow"
```

$!RENAMEDATASETVAR

Syntax:

```
$!RENAMEDATASETVAR
  VAR  = <varref>
  NAME = <string>
  [no optional parameters]
```

Description:

Rename a data set variable in Tecplot 360.

Required Parameters

Parameter	Syntax	Default	Notes
VAR	= <varref>		Specify the variable number.
NAME	= <string>		Specify the new variable name.

Example

Rename variable 1 to be Banana:

```
$!RENAMEDATASETVAR
  VAR = 1
  NAME = "Banana"
```

$!RENAMEDATASETZONE

Syntax:

```
$!RENAMEDATASETZONE
  ZONE = <integer>
  NAME = <string>
  [no optional parameters]
```

Description:

Rename a data set zone in Tecplot 360.

Required Parameters

Parameter	Syntax	Default	Notes
ZONE	= <integer>		Specify the zone number.
NAME	= <string>		Specify the new zone name.

Example

Rename zone 1 to be Banana:

```
$!RENAMEDATASETZONE
  ZONE = 1
  NAME = "Banana"
```

$!RESET3DAXES

Syntax:

```
$!RESET3DAXES
  [no parameters]
```

Description:

Reset the ranges on the 3D axes.

Example

```
$!RESET3DAXES
```

$!RESET3DORIGIN

Syntax:

```
$!RESET3DORIGIN
  [Optional Parameters]
```

Description:

Reposition the rotation origin in 3D to be at the specified location.

Optional Parameter

Parameter	Syntax	Default	Notes
ORIGINRESETLOCATION	= <originresetlocation>	DATACENTER	

Example

```
$!RESET3DORIGIN
  ORIGINRESETLOCATION = DATACENTER
```

$!RESET3DSCALEFACTORS

Syntax:

```
$!RESET3DSCALEFACTORS
```

```
  [no parameters]
```

Description:

Recalculate the scale factors for the 3D axes. Aspect ratio limits are taken into account.

Example

```
$!RESET3DSCALEFACTORS
```

$!RESETRAWUSERDEFINEDCOLORMAP

Syntax:

```
$!RESETRAWUSERDEFINEDCOLORMAP
  SOURCECOLORMAP = <string>
```

Description:

Resets the raw user-defined color map to the color map named by SOURCECOLORMAP.

Required Parameter

Parameter	Syntax	Default	Notes
SOURCECOLORMAP	= <string>		The color map named must exist and may be either a built-in or custom color map.

$!RESETVECTORLENGTH

Syntax:

```
$!RESETVECTORLENGTH
  [no parameters]
```

Description:

Reset the length of the vectors. Tecplot 360 will find the vector with the largest magnitude and set the scaling factor so it will appear on the screen using the length specified by `$!FRAMESETUP VECTDEFLEN`

Example

```
$!RESETVECTORLENGTH
```

$!RESETVECTORSPACING

Syntax:

```
$!RESETVECTORSPACING
  [no parameters]
```

Description:

Reset the spacing between evenly spaced vectors to an appropriate value for the current plot.

Tecplot 360 uses $!FRAMESETUP VECTORDEFAULTSPACINGCOUNT along with the current view to reset the spacing in each of the axial directions.

Example

```
$!RESETVECTORSPACING
```

$!ROTATE2DDATA

Syntax:

```
$!ROTATE
  ANGLE = <dexp>
  [Optional Parameters]
```

Description:

Rotate field data in 2D about any point. See also $!ROTATEDATA.

Required Parameter

Parameter	Syntax	Default	Notes
ANGLE	= <dexp>		Specify angle of rotation in degrees.

Optional Parameters

Parameter	Syntax	Default	Notes
ZONELIST	= <set>	All zones.	Zones to rotate.

Parameter	Syntax	Default	Notes
X	= <dexp>	0	X-origin to rotate about.
Y	= <dexp>	0	Y-origin to rotate about.

Example

Rotate zone 3 30 degrees about the point (7, 2):

```
$!ROTATE2DDATA
  ANGLE    = 30
  ZONELIST = [3]
  X        = 7
  Y        = 2
```

$!ROTATE3DVIEW

Syntax:

```
$!ROTATE <rotateaxis>
  ANGLE = <dexp>
  [Optional Parameters]
```

Description:

Do a 3D rotation about a given axis. The <rotateaxis> must be supplied.

Required Parameter

Parameter	Syntax	Default	Notes
ANGLE	= <dexp>		Angle to rotate (in degrees).

Optional Parameters

Parameter	Syntax	Default	Notes
ROTATEORIGINLOCATION	= <rotateoriginlocation>		
VECTORX	= <dexp>		Required when rotate axis is ABOUTVECTOR.
VECTORY	= <dexp>		Required when rotate axis is ABOUTVECTOR.
VECTORZ	= <dexp>		Required when rotate axis is ABOUTVECTOR.

Example

```
$!ROTATE3DVIEW PSI
  ANGLE = 10
```

$!ROTATEDATA

Syntax:

```
$!ROTATEDATA
  ANGLE = <dexp>
  XVAR = <varref>
  YVAR = <varref>
  UVARLIST = <varset>
  VVARLIST = <varset>
  [Optional Parameters]
```

Description:

Using the right-hand rule, rotate the axis variables and/or vector variables in the specified set of zones. You may optionally specify the origin and axis of rotation. See also $!ROTATE2DDATA and $!AXIALDUPLICATE.

Required Parameters

Parameter	Syntax	Notes
ANGLE	= <dexp>	Angle to rotate (in degrees).

Optional Parameters

Parameter	Syntax	Default	Notes
NORMALX	= <dexp>	0.0	For 3D rotation, the X component of the axis of rotation. If NORMALs are not specified, the rotation will occur around the Z axis.
NORMALY	= <dexp>	0.0	For 3D rotation, the Y component of the axis of rotation. If NORMALs are not specified, the rotation will occur around the Z axis.
NORMALZ	= <dexp>	1.0	For 3D rotation, the Z component of the axis of rotation. If NORMALs are not specified, the rotation will occur around the Z axis.
ORIGINX	= <dexp>	0.0	X coordinate of the center of rotation.

Parameter	Syntax	Default	Notes
ORIGINY	= <dexp>	0.0	Y coordinate of the center of rotation.
ORIGINZ	= <dexp>	0.0	Z coordinate of the center of rotation (for 3D rotation only).
UVARLIST	= <varset>		Set containing vector variable U components to rotate. If omitted, XVAR, YVAR, and if performing 3D rotation ZVAR must be supplied.
VVARLIST	= <varset>		Set containing vector variable V components to rotate. If omitted, XVAR, YVAR, and if performing 3D rotation ZVAR must be supplied.
WVARLIST	= <varset>		Set containing vector variable W components to rotate if performing 3D rotation, otherwise it must be omitted. If performing 3D rotation and omitted, XVAR, YVAR, and if performing 3D rotation ZVAR must be supplied.
XVAR	= <varref>		X variable to rotate. XVAR may be omitted if only rotating vectory variables in which case YVAR and ZVAR must be omitted. If the spatial variables are omitted then UVARLIST, VVARLIST, and if performing 3D rotation, WVARLIST sets must be supplied.
YVAR	= <varref>		Y variable to rotate. YVAR may be omitted if only rotating vectory variables in which case XVAR and ZVAR must be omitted. If the spatial variables are omitted then UVARLIST, VVARLIST, and if performing 3D rotation, WVARLIST sets must be supplied.
ZVAR	= <varref>		Z variable to rotate if performing 3D rotation otherise it must be omitted. ZVAR may be omitted if only rotating vectory variables in which case XVAR and YVAR must be omitted. If the spatial variables are omitted then UVARLIST, VVARLIST, and if performing 3D rotation, WVARLIST sets must be supplied.
ZONELIST	= <set>	All zones	Set specifying the zones to be rotated. If omitted, rotate all zones.

$!RUNMACROFUNCTION

Syntax:

```
$!RUNMACROFUNCTION <string> [<macroparameterlist>]
```

Description:

Execute commands defined in a macro function. The <string> references the name of the macro function to run. If the macro command requires parameters, then include them (within parentheses) after the macro command name.

Example:

Run macro function XYZ and pass the value 7 as the first parameter and the value 3.5 as the second parameter:

```
$!RUNMACROFUNCTION "XYZ" (7,3.5)
```

S-X

$!SAVELAYOUT

Syntax:

```
$!SAVELAYOUT <string>
  [Optional Parameters]
```

Description:

Save the current layout to a file. You must supply the file name. Note that variable assignments will use either integers or variable names depending on the current value of $!COMPATIBILITY USENAMESFORVARIABLEASSIGNMENTS (if set to TRUE then variable names will be used otherwise integers).

Optional Parameters

Parameters	Syntax	Default	Notes
INCLUDEDATA	= <boolean>	NO	If YES, a layout package file will be created. The extension .lpk is recommended.
INCLUDEPREVIEW	= <boolean>	YES	Applies only if INCLUDEDATA is YES.

Parameters	Syntax	Default	Notes
USERELATIVEPATHS	= <boolean>	NO	If YES, all files referenced in the layout file will use relative paths.

Example:

Save the current layout to a file called `ex1.lay`:

```
$!SAVELAYOUT "ex1.lay"
```

$!SET3DEYEDISTANCE

Syntax:

```
$!SET3DEYEDISTANCE
  EYEDISTANCE = <dexp>
```

Description:

Sets the distance from the viewer to the plane of the current center of rotation.

Example:

```
$!SET3DEYEDISTANCE
  EYEDISTANCE = 13.5
```

$!SETARBITRARYSLICEUSINGTHREEPOINTS

Syntax:

```
$!SETARBITRARYSLICEUSINGTHREEPOINTS
  GROUP = <integer>
  X1 = <dexp>
  Y1 = <dexp>
  Z1 = <dexp>
  X2 = <dexp>
  Y2 = <dexp>
  Z2 = <dexp>
  X3 = <dexp>
  Y3 = <dexp>
  Z3 = <dexp>
```

Description:

Set the orientation of an arbitrarily-oriented slice by specifying the X, Y, and Z coordinates of three points on a plane. The three points must not be coincident or collinear. The slice's origin is set to the third point and its normal is recalculated such that the cutting plane passes through all three points.

Required Parameters

Parameter	Syntax	Default	Notes
GROUP	= <integer>		The slice group.
X1	= <dexp>		Coordinates of the first point on the cutting plane.
Y1	= <dexp>		
Z1	= <dexp>		
X2	= <dexp>		Coordinates of the second point on the cutting plane.
Y2	= <dexp>		
Z2	= <dexp>		
X3	= <dexp>		Coordinates of the third point on the cutting plane. This point is used as the origin of the slice.
Y3	= <dexp>		
Z3	= <dexp>		

$!SETAUXDATA

Syntax:

```
$!SETAUXDATA
  AUXDATALOCATION = <auxlocation>
  NAME = <string>
  VALUESTRING = <string>
  [Optional Parameters]
```

Description:

Add Auxiliary Data in the form of name/value pairs to layouts, zones, frames or datasets. The name must begin with an underscore or letter, and may be followed by one or more underscore, period, letter, or digit characters.

Required Parameters

Parameter	Syntax	Default	Notes
AUXDATALOCATION	= <auxlocation>		

Parameter	Syntax	Default	Notes
NAME	= <string>		
RETAIN	= <boolean>	NO	Set this to YES to have this auxiliary data item saved when exporting a data file or layout.
VALUESTRING	= <string>		

Optional Parameters

Parameter	Syntax	Default	Notes
MAP	= <integer>		Only required if AUXDATALOCATION = linemap
VAR	= <varref>		Only required if AUXDATALOCATION = var
ZONE	= <integer>		Only required if AUXDATALOCATION = zone

Example:

Set the selected Auxiliary Data to Zone 2:

```
$!SETAUXDATA
  AUXDATALOCATION = zone
  ZONE = 2
  NAME = "VARIABLE.DATA"
  VALUESTRING = "WEST SECTOR"
```

$!SETDATASETTITLE

Syntax:

```
$!SETDATASETTITLE <string>
[no optional parameters]
```

Description:

Set the title for the current data set.

Example:

```
$!SETDATASETTITLE "My data set"
```

$!SETFIELDVALUE

Syntax:

```
$!SETFIELDVALUE
  ZONE = <integer>
  VAR = <varref>
  INDEX = <integer>
  FIELDVALUE = <dexp>
  [optional parameters]
```

Description:

Specify a field value (data set value) at a specified point index. If the zone referenced is IJ- or IJK-ordered then the point index is calculated by treating the 2- or 3D array as a 1-D array.

Required Parameters

Parameters	Syntax	Default	Notes
FIELDVALUE	= <dexp>		
INDEX	= <integer>		
VAR	= <varref>		
ZONE	= <integer>		

Optional Parameters

Parameter	Syntax	Default	Notes
AUTOBRANCH	= <boolean>	YES	Affects shared variables only. If YES, the specified zone will no longer share that variable with the other zones. If NO, the variable will still be shared, and the change to the variable will be shown for all zones where it is shared.

Example:

A data set contains 2 zones and 3 variables. Zone 2 is dimensioned 5 by 3. Set the value for variable 3 at IJ-location (2, 2) to be 37.5:

```
$!SETFIELDVALUE
  ZONE        = 2
  VAR         = 3
  INDEX       = 7
```

```
  FIELDVALUE = 37.5
  AUTOBRANCH = YES
```

Note that the INDEX value was calculated using:

```
INDEX = I + (J-1)*|MAXI| + (K-1) * |MAXI| * |MAXJ|
      = 5*(2-1)+2
      = 7
```

$!SETFRAMEBACKGROUNDCOLOR

Syntax:

```
$!SETFRAMEBACKGROUNDCOLOR <color>
```

Description:

Sets the frame background to the specified color and surveys all basic color assignments in Tecplot 360, converting the all basic colors using the following rules to achieve the best contrast:

1. For all line type basic colors that match the new basic frame color, set the basic line color to the best show color of the basic frame color.
2. For all fill type basic colors that match the best show color of the new basic frame color, set the fill color to the new frame color.

Exceptions:

1. For geometries and text boxes if the line and fill colors are the same and filling is active then both lines and fill follow the fill rules above.
2. For zone, slice, iso-surface, and streamtrace object types the basic color shading (i.e. fill) only follows the fill rules above if lighting effects are not being used.

$!SETSOLUTIONTIMECLUSTERING

Syntax:

```
$!SETSOLUTIONTIMECLUSTERING
  [one or more optional parameters]
```

Description:

Sets the solution time clustering options for the dataset that specify how Tecplot gathers together

solution times when forming time steps from the dataset's zones.

Optional Parameters

Parameters	Syntax	Default	Notes
TIMESCALING	= < timescaling>	LINEAR	Identifies if the dataset's zone solution times are distributed linearly or logarithmically. Linear: Used to cluster transient zones by solution time that is varying linearly. Logarithmic: Used to cluster transient zones by solution time that is varying logarithmically.
ABSOLUTETOLERANCE	= <double>	0.0	Absolute tolerance applied to each solution time when clustering with nearby times. It must be greater than or equal to zero.
TOLERANCEFACTOR	= <double>	1e-6	Tolerance factor, which is multiplied by the overall solution time delta and applied to each solution time when clustering nearby times. It must be a value between zero and one, inclusive.

$!SETSTYLEBASE

Syntax:

```
$!SETSTYLEBASE <stylebase>
  [no parameters]
```

Description:

Instruct Tecplot 360 on how to initialize frame style values when a new frame is created. During normal operation, Tecplot 360 bases the style of a new frame on the factory defaults plus any changes assigned in the Tecplot 360 configuration file. Layout files and stylesheet files, however, rely on Tecplot 360 basing new frames only on the factory defaults. This command is typically not used by the casual user.

Example:

Set the style base for frames to use the factory defaults:

```
$!SETSTYLEBASE FACTORY
```

$!SHARECONNECTIVITY

Syntax:

```
$!SHARECONNECTIVITY
  SOURCEZONE = <integer>
  DESTINATIONZONE = <integer>
  [no optional parameters]
```

Description:

Share the nodemap between the source and destination zones, presuming that the zones are FE and have the same element type and number of nodes.

Required Parameters

Parameter	Syntax	Default	Notes
DESTINATIONZONE	= <integer>		
SOURCEZONE	= <integer>		

Example:

Shares the connectivity of the second zone with the sixth zone:

```
$!SHARECONNECTIVITY
  SOURCEZONE = 2
  DESTINATIONZONE = 6
```

$!SHAREFIELDDATAVAR

Syntax:

```
$!SHAREFIELDDATAVAR
  SOURCEZONE = <integer>
  VAR = <varref>
  DESTINATIONZONE = <integer>
  [no optional parameters]
```

Description:

Allows sharing of the specified variable from the source zone to the destination zone. Zone must be of the same type (ordered or FE) and dimensions. Cell centered variables in FE must have the same number of cells. Sharing is not allowed if either zone has global face neighbors.

Required Parameters

Parameter	Syntax	Default	Notes
DESTINATIONZONE	= <integer>		
SOURCEZONE	= <integer>		
VAR	= <varref>		

Example:

Shares the third variable from the second zone, with the fifth zone:

```
$!SHAREFIELDDATAVAR
  SOURCEZONE = 2
  VAR   = 3
  DESTINATIONZONE = 5
```

$!SHIFTLINEMAPSTOBOTTOM

Syntax:

```
$!SHIFTLINEMAPSTOBOTTOM <set>
  [no parameters]
```

Description:

Shift a list of Line-mappings to the bottom of the Line-mapping list. This in effect causes the selected Line-mappings to be drawn last.

Example:

Shift Line-mappings 2 and 4 to the bottom:

```
$!SHIFTLINEMAPSTOBOTTOM [2,4]
```

$!SHIFTLINEMAPSTOTOP

Syntax:

```
$!SHIFTLINEMAPSTOTOP <set>
  [no parameters]
```

Description:

Shift a list of Line-maps to the top of the Line-map list. This in effect causes the selected Line-maps to be drawn first.

Example:

Shift Line-maps 2 and 4 to the top:

```
$!SHIFTLINEMAPSTOTOP [2,4]
```

$!SHOWMOUSEPOINTER

Syntax:

```
$!SHOWMOUSEPOINTER <boolean>
  [No Optional Parameters]
```

Description:

The mouse icon may be deactivated within a macro to enhance the on-screen animation. It must be reactivated before exiting the macro.

Example:

```
$!SHOWMOUSEPOINTER NO
$!LOOP 36
  $!ROTATE3DVIEW X
    ANGLE = 5
  $!REDRAW
$!ENDLOOP
$!SHOWMOUSEPOINTER YES
```

$!SKETCHAXIS

Syntax:

```
$!SKETCHAXIS
  [Optional Parameters]
```

Description:

A SetValue command that assigns attributes for axes in a sketch mode frame. Axes are rarely used in

sketch frames.

Optional Parameters

Parameter	Syntax	Default	Notes
AUTOADJUSTRANGESTONICEVALEUS	= <boolean>	NO	
AXISMODE	= <axismode>	XYDEPENDENT	Set to INDEPENDENT or XYDEPENDENT.
DEPXTOYRATIO	<op> <dexp>	= 1	AXISMODE must be XYDEPENDENT to use this.
GRIDAREA	<<gridarea>>	DRAWBORDER=NO, COLOR=BLACK, LINETHICKNESS=0.4	
PRECISEGRID	<<precisegrid>>	INCLUDE=NO, SIZE=0.0045, COLOR=BLACK, ISFILLED=NO, FILLCOLOR=WHITE, DRAWGRIDLAST=NO	
PRESERVEAXISSCALE	= <boolean>	NO	
VIEWPORTNICEFIT BUFFER	= <double>		
VIEWPORTPOSITION	<<rect>>	X1=0, Y1=0, X2=100, Y2=100	
VIEWPORTTOPSNAPTARGET	= <double>	100	
VIEWPORTTOPSNAPTOLERANCE	= <double>	10	
XDETAIL	<<axisdetail>>		
YDETAIL	<<axisdetail>>		

Example:

Change the axis mode to be `INDEPENDENT` for sketch mode in the active frame:

```
$!SKETCHAXIS
  AXISMODE = INDEPENDENT
```

$!SKETCHWATERMARK

Syntax:

```
$!SKETCHWATERMARK
  [Optional Parameters]]
```

Description

Set the size and placement of watermarks for Sketch plots for a specific frame.

Optional Parameters

Parameter	Syntax	Default	Notes
ANCHORALIGNMENT	= <anchoralignment>	BOTTOMRIGHT	Sets the origin location on the watermark itself. Similar to legends.
XYPOS	<<xy>>	X = 99, Y = 1	Location of watermark origin.
WIDTH	= <double>	15	Width of the watermark in X Frame units (0-100).

$!SLICEATTRIBUTES

Syntax:

```
$!SLICEATTRIBUTES [<slicegroup>]
  [Optional Parameters]
```

Description:

A SetValue command that changes global attributes associated with slices.

Optional Parameters

Parameter	Syntax	Default	Notes
CONTOUR			
{			
SHOW	= <boolean>	YES	
CONTOURTYPE	= <contourtype>	FLOOD	CORNERCELL and AVERAGECELL options not allowed for CONTOURTYPE.
COLOR	= <color>	BLACK	
LINETHICKNESS	= <double>	0.1	
USELIGHTINGEFFECT	= <boolean>	NO	
FLOODCOLORING	= <contourcoloring>	GROUP1	
LINECONTOURGROUP	= <integer>	1	
}			
EDGELAYER			
{			
EDGETYPE		BORDERS	
SHOW	= <boolean>	NO	
COLOR	= <color>	BLACK	
LINETHICKNESS	<op> <double>	= 0.4	
}			
EFFECTS			
{			
LIGHTINGEFFECT		GOURAUD	
SURFACETRANSLUCENCY	= <integer>	10	
USETRANSLUCENCY	= <boolean>	YES	
USEVALUEBLANKING	= <boolean>	YES	
USECLIPPLANES	<set>	[1-6]	
}			
ENDPOSITION			
{			
X	= <double>	0.95	

Parameter	Syntax	Default	Notes
Y	= <double>	0.95	
Z	= <double>	0.95	
I	= <integer>	10	
J	= <integer>	10	
K	= <integer>	10	
}			
MESH			
{			
SHOW	= <boolean>	NO	
COLOR	= <color>		
LINETHICKNESS	= <double>	0.1	
}			
NUMINTERMEDIATESLICES	= <integer>	1	
NORMAL			Used for defining arbitrary slice orientation. These settings are only used if the SliceSurface is set to ARBITRARY
{			
X	= <double>	1	
Y	= <double>	0	
Z	= <double>	0	
}			
OBEYSOURCEZONEBLANKING	= <boolean>	NO	
CLIPPLANE	= <clipplane>	NONE	Use slice as a clipping plane with one of the clip plane types.
OBEYCLIPPLANES	= <boolean>	YES	Clip slice by any clipping planes that intersect it.
PRIMARYPOSITION			XYZ are used as the origin when SLICESURFACE = ARBITRARY
{			
X	= <double>	0.5	
Y	= <double>	0.5	

Parameter	Syntax	Default	Notes
Z	= <double>	0.5	
I	= <integer>	5	
J	= <integer>	5	
K	= <integer>	5	
}			
SHADE			
{			
SHOW	= <boolean>	NO	
COLOR	= <color>	BLACK	
USELIGHTINGEFFECT	= <boolean>	YES	
}			
SHOWGROUP	= <boolean>	YES	
SHOWINTERMEDIATE SLICES	= <boolean>	NO	
SHOWPRIMARYSLICE	= <boolean>	YES	
SHOWSTARTENDSLICE	= <boolean>	NO	
SLICECONSTRAINT			
{			
INCLUDE	= <boolean>		
BOXDIMENSION	<<xyz>>		
ORIGIN	<<xyz>>		
}			
SLICESOURCE	= <slicesource>	VOLUMEZONES	
SLICESURFACE	= <slicesurface>	XPLANES	
STARTPOSITION			
{			
X	= <double>	0.05	
Y	= <double>	0.05	
Z	= <double>	0.05	

Parameter	Syntax	Default	Notes
I	= <integer>	0	
J	= <integer>	0	
K	= <integer>	0	
}			
VECTOR			
{			
SHOW	= <boolean>	NO	
COLOR	= <color>	BLACK	
ISTANGENT	= <boolean>	NO	
LINETHICKNESS	= <double>	0.1	
VECTORTYPE	= <vectortype>	TAILATPOINT	
ARROWHEADSTYLE	= <arrowheadstyle>	PLAIN	
}			

Parameter	Syntax	Default	Notes
SURFACEGENERATIONMETHOD	= <surfacegenerationmethod>	AUTO	**Auto**: Selects one of the surface generation algorithms best suited for the zones participating in the slice generation. "All Polygons" is used if one or more of the participating zones is polytope, otherwise slices use "Allow Quads". **AllowQuads**: Produces quads or triangles, and the resulting surface more closely resembles the shape of the volume cells from the source zone. **AllTriangles**: An advanced algorithm that can handle complex saddle issues and guarantees that there will be no holes in the final surface. As the surface is composed entirely of triangles, it can be delivered more efficiently to the graphics hardware. **AllPolygons**: Similar to the "All triangles" method except that all interior faces generated as a result of triangulation that are not part of the original mesh are eliminated. This preserves the original mesh of the source zones on the resulting slice.

Example:

```
$!GLOBALCONTOUR VAR = 4
$!SLICEATTRIBUTES ENDPOSITION {X = 1}
$!SLICEATTRIBUTES STARTPOSITION {X = 6}
$!SLICEATTRIBUTES NUMINTERMEDIATESLICES = 6
$!SLICEATTRIBUTES SHOWSTARTENDSLICE = YES
$!SLICEATTRIBUTES SHOWINTERMEDIATESLICES = YES
$!REDRAW
$!EXTRACTSLICES
```

$!SLICELAYERS

Syntax:

```
$!SLICELAYERS SHOW  = <boolean>
```

Description:

Turn slicing on or off.

Required Parameters

Parameter	Syntax	Default	Notes
SHOW	= <boolean>	NO	

$!SMOOTH

Syntax:

```
$!SMOOTH
  ZONE = <integer>
  VAR = <varref>
  [Optional Parameters]
```

Description:

Smooth data (reduce the spikes) for selected variables in selected zones.

Required Parameters

Parameter	Syntax	Default	Notes
ZONE	= <integer>		Zone to smooth.
VAR	= <varref>		Variable to smooth. This cannot be X or Y if in 2D or Z if in 3D and they must be a dependent variable in XY-plots.

Optional Parameters

Parameter	Syntax	Default	Notes
NUMSMOOTHPASSES	= <integer>	1	
SMOOTHWEIGHT	= <dexp>	0.8	
SMOOTHBNDRYCOND	<boundarycondition>	FIXED	

Example:

Smooth variable 3 in zone 2:

```
$!SMOOTH
```

```
  ZONE = 2
  VAR = 3
  NUMSMOOTHPASSES = 5
  SMOOTHBNDRYCOND = ZERO2ND
```

$!STREAMATTRIBUTES

Syntax:

```
$!STREAMATTRIBUTES
  [Optional Parameters]
```

Description:

A SetValue command that changes global attributes associated with streamtraces.

Optional Parameters

Parameter	Syntax	Default	Notes
ADDARROWS	= <boolean>	YES	
ARROWHEADSIZE	<op> <dexp>	= 1.2	
ARROWHEADSPACING	<op> <double>	= 10	Distance between arrowheads in frame units.
CELLFRACTION	<op> <dexp>	= 0.25	Maximum fraction of the distance across a cell a streamtrace moves in one step. A streamtrace adjusts its step-size between CELLFRACTION and MINCELLFRACTION depending on local curvature of the streamtrace.
COLOR	= <color>	BLACK	
LINETHICKNESS	<op> <dexp>	= 0.1	
MAXSTEPS	<op> < integer>	= 10,000	
MINCELLFRACTION	<op> <dexp>	= 1 x 10^{-005}	Minimum fraction of the distance across a cell a streamtrace moves in one step.
OBEYSOURCEZONEBLANKING	= <boolean>	NO	
OBEYCLIPPLANES	= <boolean>	NO	Clip streamtraces by any clipping planes that intersect the streamtraces.
RODRIBBON			

Parameter	Syntax	Default	Notes
{			
WIDTH	<op> <dexp>	= 0.01	Value is grid units.
NUMRODPOINTS	<op> < integer>	= 3	Number of points defining the streamrod cross-section
MESH			
{			
SHOW	= <boolean>	NO	
}			
CONTOUR			
{			
SHOW	= <boolean>	NO	
USELIGHTINGEFFECT	= <boolean>	YES	
FLOODCOLORING	= <contourcoloring>	GROUP1	
}			
SHADE			
{			
SHOW	= <boolean>	YES	
COLOR	= <color>		
USELIGHTINGEFFECT	= <boolean>	YES	
}			
EFFECTS			
{			
LIGHTINGEFFECT	= <lightingeffect>	GOURAUD	
SURFACETRANSLUCENCY	= <translucency>	50	
USETRANSLUCENCY	= <boolean>	NO	
}			
}			

Parameter	Syntax	Default	Notes
SHOWPATHS	= <boolean>	YES	
STREAMTIMING			
{			
SHOWDASHES	= <boolean>	NO	
SHOWMARKERS	= <boolean>	NO	
MARKCOLOR	= <color>	BLACK	
MARKSIZE	<op> <dexp>	= 1	
DASHSKIP	<op> < integer>	= 1	
MARKSYMBOL	<<symbolshape>>	ISASCII=NO, GEOMSHAPE= SQUARE	
TIMESTART	= <double>	-1 x 10^{150}	
TIMEEND	= <double>	1 x 10^{150}	
TIMEANCHOR	= <double>	0	
TIMEDELTA	= <double>	1 x 10^{150}	
}			
TERMLINE			
{			
ISACTIVE	= <boolean>	NO	
SHOW	= <boolean>	YES	
COLOR	= <color>	BLACK	
LINEPATTERN	= < linepattern>	SOLID	
PATTERNLENGTH	<op> <dexp>	= 2	
LINETHICKNESS	<op> <dexp>	= 0.1	
}			

$!STREAMTRACE [Required-Control Option]

Description:

The different commands in the STREAMTRACE compound function family are described separately in the following sections.

The STREAMTRACE compound function family is:

```
$!STREAMTRACE ADD
$!STREAMTRACE DELETALL
$!STREAMTRACE DELETERANGE
$!STREAMTRACE RESETDELTATIME
$!STREAMTRACE SETTERMINATIONLINE
```

$!STREAMTRACE ADD

Syntax:

```
$!STREAMTRACE ADD
  [Optional Parameters]
```

Description:

Add a single streamtrace or a rake of streamtraces to the active frame. The frame must be a 2D or 3D field plot.

Optional Parameters

Parameters	Syntax	Default	Notes
ALTSTARTPOS			This is required if NUMPTS is greater than 1 or if the streamtype is a volume rod or volume ribbon.
{			
X	= <dexp>	0.0	
Y	= <dexp>	0.0	
Z	= <dexp>	0.0	
}			
DIRECTION	= <streamdirection>	FORWARD	
DISTRIBUTIONREGION	= <streamdistributionregion>		If not present, use NUMPTS to decide whether to add a single streamtrace or a rake of streamtraces.
NUMPTS	= <integer>	1	Use 1 to add a single streamtrace. Use n, n>1 for a rake of streamtraces.

Parameters	Syntax	Default	Notes
STARTPOS			Z is necessary only if dealing with a 3D streamtrace.
{			
X	= <dexp>	0.0	
Y	= <dexp>	0.0	
Z	= <dexp>	0.0	
}			
STREAMTYPE	<streamtype>		Tecplot 360 determines the default STREAMTYPE based on a number of factors. It is best to always specify this parameter explicitly.

Example 1:

Add a rake of 5 streamtraces in a 2D field plot:

```
$!STREAMTRACE ADD
  NUMPTS     = 5
  STREAMTYPE = TWODLINE
  STARTPOS
  {
    X = 0.5
    Y = 0.5
  }
  ALTSTARTPOS
  {
    X = 0.5
    Y = 1.5
  }
```

Example 2:

Add a single volume ribbon. Start the ribbon oriented parallel to the Z-axis:

```
$!STREAMTRACE ADD
  STREAMTYPE = VOLUMERIBBON
  STARTPOS
  {
    X = 3.0
    Y = 4.0
    Z = 1.0
```

```
  }
  ALTSTARTPOS
  {
    X = 3.0
    Y = 4.0
    Z = 8.0
  }
```

$!STREAMTRACE DELETEALL

Syntax:

```
$!STREAMTRACE DELETEALL
  [No Parameters]
```

Description:

Deletes all streamtraces in the active frame. If the frame mode is 2D, all 2D streamtraces are deleted. If the frame mode is 3D, all 3D streamtraces are deleted.

Example:

```
$!STREAMTRACE DELETEALL
```

$!STREAMTRACE DELETERANGE

Syntax:

```
$!STREAMTRACE DELETERANGE
  [Optional Parameters]
```

Description

Delete a range of streamtraces. Streamtraces are numbered sequentially in the order they were created.

Optional Parameters

Parameters	Syntax	Default	Notes
RANGESTART	= <integer>	1	
RANGEEND	= <integer>	1	

Example:

Delete streamtraces 3-5:

```
$!STREAMTRACE DELETERANGE
  RANGESTART = 3
  RANGEEND    = 5
```

$!STREAMTRACE RESETDELTATIME

Syntax:

```
$!STREAMTRACE RESETDELTATIME
  [No Parameters]
```

Description:

Reset the time delta for dashed streamtraces. The delta time is reset such that a stream dash in the vicinity of the maximum vector magnitude will have a length approximately equal to 10 percent of the frame width.

Example:

```
$!STREAMTRACE RESETDELTATIME
```

$!STREAMTRACE SETTERMINATIONLINE

Syntax:

```
$!STREAMTRACE SETTERMINATIONLINE
  <xyrawdata>
```

Description

Set the position of the termination line for streamtraces.

Required Parameter

Parameters	Syntax	Default	Notes
<xyrawdata>			In 3D, the termination line is defined in the eye coordinate system.

Example

Set the termination line using 3 points:

```
 $!STREAMTRACE SETTERMINATIONLINE
 RAWDATA
3
4  0   7  0
5  0   9  0
5  0   3  0
```

$!STREAMTRACELAYERS

Syntax:

```
$!STREAMTRACELAYERS
  SHOW = <boolean>
```

Description:

Turn streamtraces on or off.

Required Parameters

Parameter	Syntax	Default	Notes
SHOW	= <boolean>		

$!SYSTEM

Syntax:

```
$!SYSTEM <string>
  [Optional Parameters]
```

Description:

Instruct Tecplot 360 to submit a command to the operating system. For security reasons, execution of the $!SYSTEM command can be disabled to prevent unauthorized execution of system commands via macros. Use the OKTOEXECUTESYSTEMCOMMAND option to the $!INTERFACE macro command.

Optional Parameters

Parameter	Syntax	Default	Notes
WAIT	= <boolean>	YES	If YES, Tecplot 360 will wait until the execution of the system command has completed before continuing.

Example:

Submit the system command to copy the file t7.plt to xxx.plt (Linux/Mac):

```
$!SYSTEM "cp t7.plt xxx.plt"
```

Example:

Submit the system command to copy the file t7.plt to xxx.plt (Windows):

```
$!SYSTEM "cmd /c copy t7.plt xxx.plt"
```

$!THREEDAXIS

Syntax:

```
$!THREEDAXIS
  [Optional Parameters]
```

Description:

A SetValue command that assigns attributes for axes in a 3D frame.

Optional Parameters

Parameter	Syntax	Default	Notes
ASPECTRATIOLIMIT	<op> <dexp>	= 25	Restrict the aspect ratio of the data.
ASPECTRATIORESET	<op> <dexp>	= 2	Set aspect ratio for the data to this value when ASPECTRATIOLIMIT is exceeded.
AXISMODE	= <axismode>		Set to INDEPENDENT, XYDEPENDENT, or XYZDEPENDENT.
BOXASPECTRATIOLIMIT	<op> <dexp>	= 25	Restrict the aspect ratio of the axis box.
BOXASPECTRATIORESET	<op> <dexp>	= 2	Set aspect ratio for the axis box to this value when ASPECTRATIOLIMIT is exceeded.

Parameter	Syntax	Default	Notes
DEPXTOYRATIO	<op> <dexp>		AXISMODE must be DEPENDENT to use this.
DEPXTOZRATIO	<op> <dexp>		AXISMODE must be DEPENDENT to use this.
EDGEAUTORESET	= <boolean>		Make Tecplot 360 automatically choose edges to label.
FRAMEAXIS			
{			
SHOW	= <boolean>		
SHOWVARIABLENAME	= <boolean>		Shows variable name instead of axis name on the orientation axis
SIZE	<op> <dexp>		
LINETHICKNESS	<op> <dexp>		
COLOR	= <color>		
XYPOS	<<xy>>		
}			
PRESERVEAXISSCALE	= <boolean>		
XDETAIL	<<axisdetail>>		
XYDEPXTOYRATIO	<op> <dexp>		AXISMODE must be XYDEPENDENT to use this.
YDETAIL	<<axisdetail>>		
ZDETAIL	<<axisdetail>>		

Example:

This example does the following:

1. Changes the variable assigned to the Z-axis to be variable number 2.
2. Turns off auto edge assignment and make axis labeling for the Y-axis occur on edge 2.

```
$!THREEDAXIS
  ZDetail {Var = 2}
  EDGEAUTORESET = NO
  YEDGE = 2
```

$!THREEDVIEW

Syntax:

```
$!THREEDVIEW
  [Optional Parameters]
```

Description:

A SetValue command that changes global attributes associated with the 3D view.

Optional Parameters

Parameter	Syntax	Default	Notes
ALPHAANGLE	<op> <dexp>	= 0	Angle is in degrees.
DRAWINPERSPECTIVE	= <boolean>	NO	
FIELDOFVIEW	<op> <dexp>		
PSIANGLE	<op> <dexp>	= 60	Angle is in degrees.
THETAANGLE	<op> <dexp>	= 240	Angle is in degrees.
VIEWERPOSITION	<<xyz>>	See Notes	X = 8.073, Y = 4.873, Z = 5.549
VIEWWIDTH	<op> <dexp>	= 1.74267	

Example:

This example does the following:

1. Switches to perspective.
2. Changes the field of view.
3. Rotates around psi by 20 degrees.
4. Changes the viewer position.

```
$!THREEDVIEW
  DRAWNINPERSPECTIVE = YES
  FIELDOFVIEW = 100
  PSIANGLE += 20
  VIEWERPOSITION
  {
    X = 1.26
    Y = 1.25
    Z = 0.74
  }
```

$!THREEDWATERMARK

Syntax:

```
$!THREEDWATERMARK
  [Optional Parameters]]
```

Description

Set the size and placement of watermarks for 3D Cartesian plots for a specific frame.

Optional Parameters

Parameter	Syntax	Default	Notes
ANCHORALIGNMENT	= <anchoralignment>	BOTTOMRIGHT	Sets the origin location on the watermark itself. Similar to legends.
XYPOS	<<xy>>	X = 99, Y = 1	Location of watermark origin.
WIDTH	= <double>	15	Width of the watermark in X Frame units (0-100).

$!TRANSFORMCOORDINATES

Syntax:

```
$!TRANSFORMCOORDINATES
  TRANSFORMATION = <transformation>
  [Optional Parameters]
```

Description:

Transforms all points in one or more zones from one coordinate system to another.

Tecplot 360 versions 2006 and earlier incorrectly recorded the $!TRANSFORMCOORDINATES command. In these versions, the variable number options in this command were recorded as zero-based values instead of one-based values. Macros or layout files created with any of these versions and containing $!TRANSFORMCOORDINATES should increment each variable sub-command option by one in order to operate correctly with Tecplot 360 versions 2008 and newer.

Required Parameter

Parameters	Syntax	Default	Notes
TRANSFORMATION	= <transformation>		Transformation.

Optional Parameters

Parameter	Syntax	Default	Notes
ANGLESPEC	= <anglespec>	RADIANS	Specifies whether data is in degrees or radians
CREATENEW VARIABLES	= <boolean>	NO	If YES, then new variables X,Y,Z will be created if converting to rectangular coordinates, or R,THETA,PHI if converting to spherical. If NO, then you must specify the output variables.
PSIVAR	= <varref>		PSI variable reference. REQUIRED if the transformation is spherical to rectangular or if CREATENEWVARIABLES is NO.
RVAR	= <varref>		R variable reference. REQUIRED if the transformation is polar to rectangular or spherical to rectangular or if CREATENEWVARIABLES is NO.
THETAVAR	= <varref>	NONE	Theta variable reference. REQUIRED if the transformation is polar to rectangular or spherical to rectangular or if CREATENEWVARIABLES is NO.
XVAR	= <varref>		X variable reference. REQUIRED if the transformation is rectangular to polar or rectangular to spherical or CREATENEWVARIABLES is NO.
YVAR	= <varref>		Y variable reference. REQUIRED if the transformation is rectangular to polar or rectangular to spherical or CREATENEWVARIABLES is NO.
ZONELIST	= <set>	all zones	Set of zones to operate on.
ZVAR	= <varref>		Z variable reference. REQUIRED if the transformation or rectangular to spherical or CREATENEWVARIABLES is NO.

Example

Transform data from rectangular coordinates to polar coordinates specifying angles in degrees and

creating new variables.

```
$!TRANSFORMCOORDINATES
  TRANSFORMATION = RECTTOPOLAR
  ANGLESPEC = DEGREES
  CREATENEWVARIABLES = YES
  XVAR = 2
  YVAR = 3
```

$!TRIANGULATE

Syntax:

```
$!TRIANGULATE
  [Optional Parameters]
```

Description:

Create a new zone by forming triangles from data points in existing zones.

Optional Parameters

Parameters	Syntax	Default	Notes
AUTOSTRANDTRANSIENTDATA	= <boolean>	YES	If set to YES, time strands are automatically created for transient data in the new zone.
BOUNDARYZONES	= <set>		Required if USEBOUNDARY is YES.
INCLUDEBOUNDARYPTS	= <boolean>	NO	Set to YES if you also want the boundary points to be used to create triangles.
SOURCEZONES	= <set>	All zones.	
TRIANGLEKEEPFACTOR	= <dexp>	0.25	
USEBOUNDARY	= <boolean>	NO	Specify one or more I-ordered zones that define boundaries across which no triangles can be created.

Example

Create a zone by triangulating data points from zones 1 and 2:

```
$!TRIANGULATE
  SOURCEZONES   = [1,2]
```

$!TWODAXIS

Syntax:

```
$!TWODAXIS
  [Optional Parameters]
```

Description:

A SetValue command that assigns attributes for axes in a 2D frame.

Optional Parameters

Parameter	Syntax	Default	Notes
AUTOADJUSTRANGESTONICEVALUES	= <boolean>		
AXISMODE	<axismode>		Set to INDEPENDENT or XYDEPENDENT
DEPXTOYRATIO	<op> <dexp>		AXISMODE must be XYDEPENDENT to use.
GRIDAREA	<<gridarea>>		
PRECISEGRID	<<precisegrid>>		
PRESERVEAXISSCALE	= <boolean>		
VIEWPORTNICEFITBUFFER	= <double>		
VIEWPORTPOSTITION	<<rect>>		
VIEWPORTTOPSNAPTARGET	= <integer>	100	
VIEWPORTTOPSNAPTOLERANCE	= <integer>	10	
XDETAIL	<<axisdetail>>		
YDETAIL	<<axisdetail>>		

Example:

Set the X-axis to use the Z variable for a 2D plot:

```
$!TWODAXIS
  XDETAIL {VAR = "Z"}
```

$!TWODWATERMARK

Syntax:

```
$!TWODWATERMARK
  [Optional Parameters]]
```

Description

Set the size and placement of watermarks for 2D Cartesian plots for a specific frame.

Optional Parameters

Parameter	Syntax	Default	Notes
ANCHORALIGNMENT	= <anchoralignment>	BOTTOMRIGHT	Sets the origin location on the watermark itself. Similar to legends.
XYPOS	<<xy>>	X = 99, Y = 1	Location of watermark origin.
WIDTH	= <double>	15	Width of the watermark in X Frame units (0-100).

V

$!VARSET

Syntax:

```
$!VARSET <macrovar> <op> <dexp>
  [no parameters]
or
$!VARSET <macrovar> = <string>
  [no parameters]
```

Description:

Assign a value to a macro variable. If the macro variable did not exist prior to this command, then it is defined here. A macro variable can be assigned a numeric value or a string. If the value is to be calculated from an expression, the expression must be enclosed in parentheses.

Examples

Example 1:

Set the macro variable |myvar| to 3:

```
$!VARSET |myvar| = 3
```

Example 2:

Add 2 to the macro variable |myvar|:

```
$!VARSET |myvar| += 2
```

Example 3:

Set the macro variable |File1| to be myfile.plt:

```
$!VARSET |File1| = "myfile.plt"
```

Example 4:

Set the macro variable |F1| to equal |V2| + |V3|, where |V2| and |V3| are predefined variables:

```
$!VARSET|V2| = 4
$!VARSET|V3| = 5
$!VARSET|F1| = (|V2| + |V3|)
```

$!VIEW [Required-Control Option]

Description:

The different commands in the VIEW compound function family are described separately in the following sections.

The VIEW compound function family is:

```
$!VIEW AXISFIT
$!VIEW AXISMAKECURRENTVALUESNICE
$!VIEW AXISNICEFIT
$!VIEW CENTER
$!VIEW COPY
$!VIEW DATAFIT
$!VIEW FIT
$!VIEW FITSURFACES
```

```
$!VIEW LAST
$!VIEW MAKECURRENTVIEWNICE
$!VIEW NICEFIT
$!VIEW PASTE
$!VIEW PUSH
$!VIEW RESETTOENTIRECIRCLE
$!VIEW SETMAGNIFICATION
$!VIEW TRANSLATE
$!VIEW ZOOM
```

$!VIEW AXISFIT

Syntax:

```
$!VIEW AXISFIT
  [Optional Parameters]
```

Description:

Reset the range on a specific axis so that it equals the minimum and maximum of the data being plotted. If the axis dependency is not independent then this action may also affect the range on another axis.

Optional Parameters

Parameter	Syntax	Default	Notes
AXIS	= <xyzaxis>	'X'	Default is 'T' for polar plot type.
AXISNUM	= <integer>	1	Only XY frame mode allows for this to be a number greater than 1.
CONSIDERBLANKING	= <boolean>	NO	If CONSIDERBLANKING is YES, and blanking is enabled, the resulting view excludes blanked cells at the edges of the plot.

Example:

Reset the range on the Y-axis to fit the data being plotted:

```
$!VIEW AXISFIT
  AXIS ='Y'
```

$!VIEW AXISMAKECURRENTVALUESNICE

Syntax:

```
$!VIEW AXISMAKECURRENTVALUESNICE
  [Optional Parameters]
```

Description:

Reset the axis-line label values such that all currently displayed values are set to have the smallest number of significant digits possible.

Optional Parameters

Parameter	Syntax	Default	Notes
AXIS	= <xyzaxis>	'X'	Default is 'T' for polar plot type.
AXISNUM	= <integer>	1	Only XY line plots allow for this to be a number greater than 1.

Example:

Set the range on the Z-axis to have nice values for the axis labels:

```
$!VIEW AXISMAKECURRENTVALUESNICE
  AXIS = 'Z'
```

$!VIEW AXISNICEFIT

Syntax:

```
$!VIEW AXISNICEFIT
  [Optional Parameters]
```

Description:

Reset the range on a specific axis so that it equals the minimum and maximum of the data being plotted, but makes the axis values "nice" by setting labels to have the smallest number of significant digits possible. If the axis dependency is not independent then this action may also affect the range on another axis.

Optional Parameters

Parameter	Syntax	Default	Notes
AXIS	= <xyzaxis>	'X'	Default is 'T' for polar plot type.
AXISNUM	= <integer>	1	Only XY frame mode allows for this to be a number greater than 1.
CONSIDERBLANKING	= <boolean>	NO	If CONSIDERBLANKING is YES, and blanking is enabled, the resulting view excludes blanked cells at the edges of the plot.

Example:

Reset the range on the Y-axis to fit the data being plotted, with nice values on the axis-line:

```
$!VIEW AXISNICEFIT
  AXIS ='Y'
```

$!VIEW CENTER

Syntax:

```
$!VIEW CENTER
[optional parameter]
```

Description:

Center the data within the axis grid area.

Optional Parameter

Parameter	Syntax	Default	Notes
CONSIDERBLANKING	= <boolean>	NO	If CONSIDERBLANKING is YES, and blanking is enabled, the resulting view excludes blanked cells at the edges of the plot.

$!VIEW COPY

Syntax:

```
$!VIEW COPY
  [No Parameters]
```

Description:

Copy the current view to the view paste buffer. See also `$!VIEW PASTE`.

`$!VIEW DATAFIT`

Syntax:

```
$!VIEW DATAFIT
[optional parameter]
```

Description:

Fit the current set of data zones or line mappings being plotted within the grid area. This does not take into consideration text or geometries.

Optional Parameter

Parameter	Syntax	Default	Notes
CONSIDERBLANKING	= <boolean>	NO	If CONSIDERBLANKING is YES, and blanking is enabled, the resulting view excludes blanked cells at the edges of the plot.

`$!VIEW FIT`

Syntax:

```
$!VIEW FIT
[optional parameter]
```

Description:

Fit the entire plot to the grid area. This also takes into consideration text and geometries that are plotted using the grid coordinate system. In 3D, this also includes the axes.

Optional Parameter

Parameter	Syntax	Default	Notes
CONSIDERBLANKING	= <boolean>	NO	If CONSIDERBLANKING is YES, and blanking is enabled, the resulting view excludes blanked cells at the edges of the plot.

$!VIEW FITSURFACES

Syntax:

```
$!VIEW FITSURFACES
  [Optional Parameter]
```

Description:

Fits active plot surfaces to the grid area. 3D volume zones are excluded when surfaces to plot are set to none. See $!FIELDMAP for more information on setting surfaces to plot.

Optional Parameter

Parameter	Syntax	Default	Notes
CONSIDERBLANKING	= <boolean>	NO	If CONSIDERBLANKING is YES, and blanking is enabled, the resulting view excludes blanked cells at the edges of the plot.

$!VIEW LAST

Syntax:

```
$!VIEW LAST
  [No Parameters]
```

Description:

Retrieve the previous view from the view stack. Each frame mode within each frame maintains its own view stack. $!VIEW LAST will not reverse alterations to data.

$!VIEW MAKECURRENTVIEWNICE

Syntax:

```
$!VIEW MAKECURRENTVIEWNICE
  [No Parameters]
```

Description:

Shifts axis to make axis-line values nice without changing the extents of the window. Only works in Sketch/XY/2D.

$!VIEW NICEFIT

Syntax:

```
$!VIEW NICEFIT
[optional parameter]
```

Description:

Change view to make the extents of the frame neatly hold the plot with integer values for axis labels. Only works in Sketch/XY/2D.

Optional Parameter

Parameter	Syntax	Default	Notes
CONSIDERBLANKING	= <boolean>	NO	If CONSIDERBLANKING is YES, and blanking is enabled, the resulting view excludes blanked cells at the edges of the plot.

$!VIEW PASTE

Syntax:

```
$!VIEW PASTE
  [No Parameters]
```

Description:

Retrieve the view from the view paste buffer and assign it to the active frame.

$!VIEW PUSH

Syntax:

```
$!VIEW PUSH
  [No Parameters]
```

Description:

Instruct Tecplot 360 to push the current view onto the view stack. A view will not be pushed if the current view is the same as the top view on the stack. Note that commands `VIEW AXISFIT`, `VIEW CENTER`, `VIEW DATAFIT`, `VIEW FIT`, and `VIEW ZOOM` automatically push a view onto the stack. Tecplot 360 automatically pushes the current view onto the stack when a `$!REDRAW` command is issued and the

current view is different from the top view on the view stack.

$!VIEW RESETTOENTIRECIRCLE

Syntax:

```
$!VIEW RESETTOENTIRECIRCLE
  [No Parameters]
```

Description:

Reset the Theta-R Axis to initial settings. For Polar plots only.

$!VIEW SETMAGNIFICATION

Syntax:

```
$!VIEW SETMAGNIFICATION
MAGNIFICATION = <dexp>
```

Description:

Set the magnification for the data being plotted. A magnification of 1 will size the plot so it can fit within the grid area.

Required Parameter

Parameter	Syntax	Default	Notes
MAGNIFICATION	= <dexp>		

Example

Make the plot to be drawn one-half as big as when it fits within the grid area:

```
$!VIEW SETMAGNIFICATION
  MAGNIFICATION = 0.5
```

$!VIEW TRANSLATE

Syntax:

```
$!VIEW TRANSLATE
  X = <dexp>
```

```
  Y = <dexp>
  [no optional parameters]
```

Description:

Shift the data being plotted in the X- and/or Y-direction. The amount translated is in frame units.

Required Parameters

Parameter	Syntax	Default	Notes
X	= <dexp>	0.0	Amount to translate in X-frame units.
Y	= <dexp>	0.0	Amount to translate in Y-frame units.

Example:

Translate the view 10 percent of the frame width to the right:

```
$!VIEW TRANSLATE
  X = 10
  Y = 0
```

$!VIEW ZOOM

Syntax:

```
$!VIEW ZOOM
  X1 = <dexp>
  Y1 = <dexp>
  X2 = <dexp>
  Y2 = <dexp>
[no optional parameters]
```

Description:

Change the view by "zooming" into the data. In Sketch, XY, and 2D frame mode plots, Tecplot 360 will adjust the ranges on the axis to view the region defined by the rectangle with corners at (X1, Y1) and (X2, Y2). For 3D orthographic plots, the view is translated and scaled to fit the region. For 3D perspective plots, the view is rotated about the viewer and scaled to fit the region. X1 and so forth are measured in grid coordinates.

Required Parameters

Parameter	Syntax	Default	Notes
X1	= <dexp>		
Y1	= <dexp>		
X2	= <dexp>		
Y2	= <dexp>		

Example:

Zoom so the rectangular region with corners at (1, 0) and (7, 9) are in view:

```
$!VIEW ZOOM
  X1 = 1
  Y1 = 0
  X2 = 7
  Y2 = 9
```

$!WATERMARK

Syntax:

```
$!WATERMARK
  [Optional Parameters]]
```

Description:

Change the overall state of watermarks in 360. In particular, the assignment of watermark images and the ability to turn watermarks on or off.

Optional Parameters:

Parameter	Syntax	Default	Notes
SHOW	= <boolean>	YES	

Parameter	Syntax	Default	Notes
IMAGEFILENAMES	= <rawstring>	NULL	Rawstring of filenames representing the image files for your watermark. If filenames are not absolute paths then files are assumed to be relative to the Tecplot startup directory. These files can be png, jpeg, or bmp files. All files should represent the same image but at different resolutions. For example resolutions of 50, 100, 200, 300 pixels wide would be a reasonable set of images. 360 will then pick the best resolution image for a given context.
RESIZEFILTER	= <resizefilter>	TEXTUREFILTER	The Resize filter determines how the image is resized to fit the screen. The following filters are available: • TextureFilter (Fast) - default- Tecplot 360 uses OpenGL textures to resize the image. This is the fastest option (given sufficient graphics space). However, the accuracy of the image may suffer, especially when reducing an image to a size much smaller than it was before. • Pixelated - Choose this option when the image is much larger than its original size and you want to see the individual pixels. This option is slower than the Fast (textures) for increasing the size of images. • Smooth - There are seven smooth options, all producing slightly different effects. These options are slower than the Fast (textures), but produce better effects for highly reduced images. In general, use the Smooth (Lanczos2) option unless you have specific image processing needs.

$!WHILE···$!ENDWHILE

Syntax:

```
$!WHILE <conditionalexp>
```

```
...
$!ENDWHILE
```

Description:

Continue to execute a set of commands until a conditional expression is NO.

Example:

Execute a set of commands until the macro variable `|myvar|` is greater than or equal to 1.0:

```
$!VARSET |myvar| = 0.0
$!WHILE |myvar| < 1.0
  $!VARSET |myvar| += 0.01
$!ENDWHILE
```

$!WORKSPACEVIEW [Required-Control Option]

Description:

The different commands in the `WORKSPACEVIEW` compound function family are described separately in the following sections.

The `WORKSPACEVIEW` compound functions are:

```
$!WORKSPACEVIEW FITALLFRAMES
$!WORKSPACEVIEW FITPAPER
$!WORKSPACEVIEW FITSELECTEDFRAMES
$!WORKSPACEVIEW LASTVIEW
$!WORKSPACEVIEW MAXIMIZE
$!WORKSPACEVIEW TRANSLATE
$!WORKSPACEVIEW UNMAXIMIZE
$!WORKSPACEVIEW ZOOM
```

$!WORKSPACEVIEW FITALLFRAMES

Syntax:

```
$!WORKSPACEVIEW FITALLFRAMES
  [No Parameters]
```

Description:

Change the view in the workspace so all frames are fit just inside the edges of the workspace.

$!WORKSPACEVIEW FITPAPER

Syntax:

```
$!WORKSPACEVIEW FITPAPER
  [no parameters]
```

Description:

Change the view in the workspace so the entire paper is fit just inside the edges of the workspace.

$!WORKSPACEVIEW FITSELECTEDFRAMES

Syntax:

```
$!WORKSPACEVIEW FITSELECTEDFRAMES
  [No Parameters]
```

Description:

Change the view in the workspace so the currently selected frames (that is, the frames with pick handles) are fit just inside the edges of the workspace.

$!WORKSPACEVIEW LASTVIEW

Syntax:

```
$!WORKSPACEVIEW LASTVIEW
  [No Parameters]
```

Description:

Return to the previous workspace view.

$!WORKSPACEVIEW MAXIMIZE

Syntax:

```
$!WORKSPACEVIEW MAXIMIZE
```

```
  [No Parameters]
```

Description:

Temporarily expand the work area as large as possible. The maximized work area occupies the entire Tecplot 360 process window.

$!WORKSPACEVIEW TRANSLATE

Syntax:

```
$!WORKSPACEVIEW TRANSLATE
  X = <dexp>
  Y = <dexp>
  [no optional parameters]
```

Description:

Shift the view of the workspace. This has no effect on the local view within any frame in your layout.

Required Parameters

Parameter	Syntax	Default	Notes
X	= <dexp>	0	Value is in inches.
Y	= <dexp>	0	Value is in inches.

Example:

Shift the workspace view to the left by 2 inches (as measured by the workspace ruler):

```
$!WORKSPACEVIEW TRANSLATE
  X = -2
  Y = 0
```

$!WORKSPACEVIEW UNMAXIMIZE

Syntax:

```
$!WORKSPACEVIEW UNMAXIMIZE
  [no parameters]
```

Description:

Returns the workspace to its normal size after it has been expanded after `$!WORKSPACE MAXIMIZE` has been used.

$!WORKSPACEVIEW ZOOM

Syntax:

```
$!WORKSPACEVIEW ZOOM
  X1 = <dexp>
  Y1 = <dexp>
  X2 = <dexp>
  Y2 = <dexp>
  [no optional parameters]
```

Description:

Change the view into the work area. This has no effect on the local view within any frame in your layout.

Required Parameters

Parameter	Syntax	Default	Notes
X1	= <dexp>		
Y1	= <dexp>		
X2	= <dexp>		
Y2	= <dexp>		

Example:

Make the region in the lower left corner of an 8.5 by 11 paper be viewable in the work area. The paper is in portrait orientation:

```
$!WORKSPACEVIEW ZOOM
  X1 = 0
  Y1 = 5.5
  X2 = 4.25
  Y2 = 9.75
```

$!WRITECOLORMAP

Syntax:

```
$!WRITECOLORMAP <string>
  [no parameters]
```

Description:

Write all custom color maps to a file. The <string> is the name of the file to write to.

Example:

```
$!WRITECOLORMAP "mycolors.map"
```

$!WRITECURVEINFO

Syntax:

```
$!WRITECURVEINFO <string>
  SOURCEMAP = <integer>
  [Optional Parameters]
```

Description:

Write out the curve details or the calculated data points for the equation(s) used to draw the curve for a selected line mapping. The <string> is the name of the file to write to.

Required Parameter

Parameter	Syntax	Default	Notes
SOURCEMAP	= <integer>		This must be the number of a line mapping that does some type of curve fit or spline.

Optional Parameter

Parameters	Syntax	Default	Notes
CURVEINFOMODE	= <curveinfomode>	CURVEDETAILS	Use CURVE DETAILS or CURVEPOINTS.

Example:

Write out the coefficients for XY line mapping number 3 to `map3.out`:

```
$!WRITECURVEINFO "map3.out"
  SOURCEMAP      = 3
  CURVEINFOMODE  = CURVE DETAILS
```

$!WRITEDATASET

Syntax:

```
$!WRITEDATASET <string>
  [Optional Parameters]
```

Description:

Write the data set attached to the active frame to a file. The <string> is the name of the file to write to.

Optional Parameters

Parameters	Syntax	Default	Notes
ASSOCIATELAYOUTWITHDATAFILE	= <boolean>	YES	
BINARY	= <boolean>	YES	If NO, can use PRECISION and USEPOINTFORMAT.
INCLUDEAUTOGENFACENEIGHBORS	= <boolean>	NO	
INCLUDECUSTOMLABELS	= <boolean>	YES	
INCLUDEDATA	= <boolean>	YES	
INCLUDEDATASHARELINKAGE	= <boolean>	NO	
INCLUDEGEOM	= <boolean>	YES	
INCLUDETEXT	= <boolean>	YES	
PRECISION	= <integer>	12	Only used if ASCII (that is, BINARY is NO).
TECPLOTVERSIONTOWRITE	= <string>	TecplotCurrent	Optional designation of binary file version. Possible values are TecplotCurrent, Tecplot2009, Tecplot2008, and Tecplot2006.
USEPOINTFORMAT	= <boolean>	NO	Only used if ASCII (that is, BINARY is NO).
VARLIST	= <varset>	All vars.	Use this to limit the number of variables written out.

Parameters	Syntax	Default	Notes
ZONELIST	= <set>	All zones.	Use this to limit the number of zones written out.

Example:

Write out only zones 1, 2, 3, and 6 to a file called `zones.plt`:

```
$!WRITEDATASET "zones.plt"
  INCLUDETEXT         = NO
  INCLUDEGEOM         = NO
  INCLUDECUSTOMLABELS = NO
  ZONELIST            = [1-3, 6]
```

$!WRITESTYLESHEET

Syntax:

```
$!WRITESTYLESHEET <string>
  [Optional Parameters]
```

Description:

Write the style for the active frame to a file. The <string> is the name of the file to write to. Note that variable assignments will use either integers or variable names depending on the current value of USENAMESFORVARIABLEASSIGNMENTS in `$!COMPATIBILITY` (if set to TRUE then variable names will be used otherwise integers).

Optional Parameters

Parameters	Syntax	Default	Notes
INCLUDECONTOURLEVELS	= <boolean>	YES	
INCLUDETEXT	= <boolean>	YES	
INCLUDEGEOM	= <boolean>	YES	
INCLUDEPLOTSTYLE	= <boolean>	YES	
INCLUDESTREAMPOSITIONS	= <boolean>	YES	
INCLUDEFACTORYDEFAULTS	= <boolean>	NO	

Parameters	Syntax	Default	Notes
INCLUDEAUXDATA	= <boolean>	YES	

Example:

Write out a stylesheet for the active frame to f1.sty##:

```
$!WRITESTYLESHEET "f1.sty"
  INCLUDEFACTORYDEFAULTS = YES
```

$!XYLINEAXIS

Syntax:

```
$!XYLINEAXIS
  [Optional Parameters]
```

Description:

A SetValue command that assigns attributes for axes in an XY Line plot.

Optional Parameters

Parameter	Syntax	Default	Notes
AUTOADJUSTRANGESTONICEVALUES	= <boolean>	NO	
AXISMODE	<axismode>	INDEPENDENT	INDEPENDENT or XYDEPENDENT.
DEPXTOYRATIO	<op> <dexp>	= 1	AXISMODE must be XYDEPENDENT to use this. This applies only to the X1- and Y1-axes.
GRIDAREA	<<gridarea>>	See Notes	DRAWBORDER=N0, COLOR=BLACK, LINE THICKNESS=0.4
PRECISEGRID	<<precisegrid>>	See Notes	INCLUDE=NO, SIZE=0.0045, COLOR=BLACK, ISFILLED=NO, FILLCOLOR=WHITE, DRAWGRIDLAST=NO
PRESERVEAXISSCALE	= <boolean>	NO	
VIEWPORTNICEFITBUFFER	= <double>		Between 1 and 100.
VIEWPORTPOSITION	<<rect>>	See Notes	X1 = 13, Y1 = 11, X2 = 8, Y = 88

Parameter	Syntax	Default	Notes
VIEWPORTTOPSNAPTARGET	= <double>	100	
VIEWPORTTOPSNAPTOLERANCE	= <double>	10	
XDETAIL	<integer> <<axisdetail>>		The <integer> specifies which axis to operate on, 1 < n < 5.
YDETAIL	<integer> <<axisdetail>>		The <integer> specifies which axis to operate on, 1 < n < 5.

Example:

Set the axis mode to be independent for the XY-axes (note that this affects only X1 versus Y1):

```
$!XYLINEAXIS
  AXISMODE = INDEPENDENT
```

$!XYLINEWATERMARK

Syntax:

```
$!XYLINEWATERMARK
  [Optional Parameters]]
```

Description

Set the size and placement of watermarks for XY Line plots for a specific frame.

Optional Parameters

Parameter	Syntax	Default	Notes
ANCHORALIGNMENT	= <anchoralignment>	BOTTOMRIGHT	Sets the origin location on the watermark itself. Similar to legends.
XYPOS	<<xy>>	X = 99, Y = 1	Location of watermark origin.
WIDTH	= <double>	15	Width of the watermark in X Frame units (0-100).

Extended Macro Commands

Tecplot 360 add-ons can and often do extend the Tecplot macro language. Macro commands associated with add-ons are exposed to the macro language using Tecplot 360's `$!EXTENDEDCOMMAND` command, which delegates the processing of the command to a named add-on. The syntax of this command is shown below:

```
$!EXTENDEDCOMMAND
  COMMANDPROCESSORID = <string>
  COMMAND = <string>
```

`COMMANDPROCESSORID` is set to a string that identifies the add-on, while `COMMAND` is set to a string that is passed to the add-on for processing.

Documentation for the macro commands supported by most add-ons can be found in the User's Manual. Two add-ons are also documented here:

- CFD Analyzer which provides the Analyze menu in Tecplot 360
- Extend Macro which provides additional functionality useful for scripting

CFD Analyzer

The Analyze menu is technically implemented as an add-on. Its `COMMANDPROCESSORID` is "CFDAnalyzer3". The command string is set to one of the commands listed below.

Summary of Analyze Macro Commands

ANIMATESTREAKLINES may be used following a streakline calculation to animate the streaklines, either to the screen or to a file.

ATTACHINTEGRATIONRESULTS is used following an integration to create a text field and attach it to the current Tecplot 360 frame. This macro has the same effect as clicking Make Text on the Integration Results text dialog.

It is not necessary to direct the macro to display the Integration Results dialog in order to attach or save the results.

CALCPARTICLEPATH calculates particle paths or streaklines for steady or unsteady flow solutions, using the location of any existing streamtraces as starting locations for the particles. Particles may have mass or be massless.

CALCTURBULENCEFUNCTION calculates any of four turbulence-related functions, given any two in your data set.

CALCULATE calculates a PLOT3D function. The name of this function must be specified in the shortened form listed in Parameter Assignment Values.

CALCULATEACCURACY uses Richardson extrapolation to estimate the order accuracy of the solution, given the solution on three grids of successively finer resolution. If either of the plotting options are set to `TRUE`, the resulting Tecplot 360 frames will be in front after executing this command.

DISPLAYBOUNDARIES displays zone boundaries in a new frame according to settings made by the SETGEOMETRYANDBOUNDARIES macro. Each boundary of each 3D zone (in 3D Cartesian plots) or 2D zone (in 2D Cartesian plots) is displayed and named according to the boundary condition applied to it. Boundaries that are connected to the boundaries of adjacent zones are named as such.

EXTRACTFLOWFEATURE displays shock surfaces, vortex cores, or separation and attachment lines for 3D flow solutions. Separation and attachment lines are only calculated on no-slip wall boundaries identified by the SETGEOMETRYANDBOUNDARIES macro. Shock surfaces are displayed as iso-surfaces of a new variable, ShockFeature, while the remaining features are displayed as new zones.

EXTRAPOLATESOLUTION performs Richardson extrapolation to estimate the true solution from three input solutions on grids of successively finer resolution. It saves the extrapolated solution as a new zone in the current data set. It also saves an additional zone containing the difference between this solution and the original solution.

INTEGRATE performs an integration. All Integrate dialog options are available to this macro, including the display options. If the TRUE, then the Tecplot 360 frame showing the integration results is the active frame following this command.

SAVEINTEGRATIONRESULTS has the same effect as clicking Save on the Integration Results dialog and selecting a file. The results are saved to the file named by the FILE parameter.

SETFIELDVARIABLES identifies variables in your data, such as velocity, pressure and temperature, for use in analysis.

SETFLUIDPROPERTIES sets the properties of the fluid, such as viscosity. These are used by some actions of the `CALCULATE` and `INTEGRATE` commands.

SETGEOMETRYANDBOUNDARIES identifies boundaries of zones in a flow solution and the boundary conditions applied to them. It also specifies whether zones with coincident boundary nodes should be considered connected at those points, as well as whether 2D solutions should be regarded as axisymmetric.

SETREFERENCEVALUES sets the reference (free-stream) properties of the solution. This information is used by other calculations.

SETUNSTEADYFLOWOPTIONS identifies solution time levels for unsteady flow solutions. This information is used for particle path and streakline calculations.

Macro Command Description

The syntax, mandatory and optional parameters for each of the macro commands listed in Summary of Analyze Macro Commands are described below. Items within single angle brackets <> are defined in Parameter Assignment Values.

The COMMAND strings below must be contained on a single line n your macro command file, although they appear on multiple lines below.

ANIMATESTREAKLINES

Syntax:

```
$!EXTENDEDCOMMAND
  COMMANDPROCESSORID = "CFDANALYZER3"
  COMMAND = 'ANIMATESTREAKLINES [optional parameters]'
```

Description:

Animates previously calculated streaklines to the screen or to a file.

Optional Parameters

Parameter	Syntax	Default	Notes
DESTINATION	= <string>	SCREEN	Specifies the destination of the animation. May be SCREEN1, `AVIFILE or RASTERMETAFILE.
FILENAME	= <string>	""	The name of the file to which to save the animation. Must be specified for DESTINATION values of AVIFILE or RASTERMETAFILE.
WIDTH	= <integer>	300	The width of the animation when saved to a file.
SPEED	= <double>	10.0	The speed in frames per second of the animation. Only used for animations saved to an AVI file.
USEMULTIPLECOLOR TABLES	= <boolean>	FALSE	Specifies whether animations saved to a file should include one color table for each frame. The default is to use a single color table.
INCLUDEZONEANIMA TION	= <boolean>	FALSE	

ATTACHINTEGRATIONRESULTS

Syntax:

```
$!EXTENDEDCOMMAND
COMMANDPROCESSORID = "CFDANALYZER3"
COMMAND = 'ATTACHINTEGRATIONRESULTS'
```

Description:

Attach the text results of the previous integration as a text field in the active frame.

CALCPARTICLEPATH

Syntax:

```
$!EXTENDEDCOMMAND
  COMMANDPROCESSORID = "CFDAnalyzer3"
  COMMAND = 'CALCPARTICLEPATH [optional parameters]'
```

Description:

Calculate particle paths or streaklines, starting from existing Tecplot 360 streamtraces.

Optional Parameters

Parameter	Syntax	Default	Notes
FUNCTION	= <particlefunction>	PARTICLEPATH	Can be PARTICLEPATH or STREAKLINE.
TIMESTEP	= <double>	1	The integration time step for the calculation.
MAXTIMESTEPS	= <integer>	1000	For steady-state calculations only.
RELEASEFREQ	= <double>	1	For `FUNCTION = STREAKLINE`. Indicates the number of particles to release in the indicated time period (see the next parameter).
RELEASEOPTION	= <releaseoption>	TIMELEVEL	For FUNCTION = STREAKLINE. If TIMELEVEL, indicates that RELEASEFREQ particles should be released every solution time level. If UNITTIME, indicates that this number of particles should be released in a unit amount of solution time.

Parameter	Syntax	Default	Notes
HAVEMASS	= <boolean>	FALSE	If TRUE, particles have mass; specify the particle mass options below.
CREATESINGLEZONE	= <boolean>	FALSE	For FUNCTION = PARTICLEPATH only, specifies that all particle paths should be combined into a single I-J ordered zone.
STOREOPTION	= <storeoption>	PARTICLEVALUES	If PARTICLEVALUES, the particle's velocity, mass and temperature (if calculated) will be stored in place of appropriate fluid values in the particle path's zone. If FLUIDVALUES, all fluid values the particle passed through will be stored in the zone.
COEFFS	= <coeffsoption>	GENERAL	If GENERAL, specify BALLISTICCOEFF, plus TEMPTIMECONST if calculating particle temperature. If DETAILED, specify MASS, RADIUS, and DRAGCOEFF, plus SPECIFICHEAT and NUSSELT if calculating temperature. Only applies if HAVEMASS = TRUE.
CALCTEMPERATURE	= <boolean>	FALSE	If TRUE, particle temperature will be calculated. Only applies if HAVEMASS = TRUE.
GRAVITYCONSTANT	= <double>	0.0	The acceleration due to gravity. Only applies if `HAVEMASS = TRUE`.
GRAVITYDIRECTION	= <gravitydirection>	MINUSX	The axis direction in which gravity acts. Only applies if `HAVEMASS = TRUE`.
INITIALVELOCITYOPTION	= <initialvelocityoption>	LOCALFLUIDVELOCITY	The initial velocity of particles. Options are `LOCALFLUIDVELOCITY` and `ZEROVELOCITY`. Only applies if `HAVEMASS = TRUE`.
BALLISTICCOEFF	= <double>	1.0	For `GENERAL` coefficients only, the ballistic coefficient of the particle. Only applies if `HAVEMASS = TRUE`.
TEMPTIMECONST	= <double>	1.0	For `GENERAL` coefficients with `CALCTEMPERATURE = TRUE` only, the temperature relaxation factor of the particle. Only applies if `HAVEMASS = TRUE`.
MASS	= <double>	1.0	For `DETAILED` coefficients only, the particle mass. Only applies if `HAVEMASS = TRUE`.
RADIUS	= <double>	1.0	For `DETAILED` coefficients only, the particle initial radius. Only applies if `HAVEMASS = TRUE`.

Parameter	Syntax	Default	Notes
DRAGCOEFFOPTION	= <specifyoption>	SPECIFY	For DETAILED coefficients only. If SPECIFY, specify DRAGCOEFF. If CALCULATE, Tecplot 360 will calculate the drag coefficient. Only applies if HAVEMASS = TRUE.
DRAGCOEFF	= <double>	1.0	For DETAILED coefficients only, with DRAGCOEFFOPTION = SPECIFY, the particle drag coefficient. Only applies if HAVEMASS = TRUE.
SPECIFICHEAT	= <double>	1.0	For DETAILED coefficients with CALCTEMPERATURE = TRUE only, the particle specific heat. Only applies if HAVEMASS = TRUE.
NUSSELTOPTION	= <specifyoption>	SPECIFY	For DETAILED coefficients with CALCTEMPERATURE = TRUE only. If SPECIFY, specify NUSSELT. If CALCULATE, Tecplot 360 will calculate the Nusselt number. Only applies if HAVEMASS = TRUE.
NUSSELT	= <double>	1.0	For DETAILED coefficients with CALCTEMPERATURE = TRUE and NUSSELTOPTION = SPECIFY only, the particle Nusselt number. Only applies if HAVEMASS = TRUE.
TERMOPTION	= <terminationoption>	TEMPERATURE	For DETAILED coefficients with CALCTEMPERATURE = TRUE only (is always TEMPERATURE for general coefficients), the particle termination option. May be TEMPERATURE or ABLATE. Only applies if HAVEMASS = TRUE.
TEMPERATURE	= <double>	1.0	If TERMOPTION = TEMPERATURE, the particle termination temperature. If TERMOPTION = ABLATE, the ablation temperature. Only applies if HAVEMASS = TRUE.
LATENTHEAT	= <double>	1.0	For TERMOPTION = ABLATE only, the latent heat of the ablative process. Only applies if HAVEMASS = TRUE.

Example 1:

Calculate streaklines with an integration time step of 0.1, releasing eight particles per unit solution time:

```
$!EXTENDEDCOMMAND
COMMANDPROCESSORID = "CFDAnalyzer3"
COMMAND = 'CALCPARTICLEPATH
```

```
   FUNCTION=STREAKLINE
   TIMESTEP=0.1    RELEASEFREQ=8
   RELEASEOPTION=UNITTIME'
```

Example 2:

Calculate particle paths, including temperature with ablation, in a steady-state flow for particles with an initial mass of 3E-14, an initial radius of 1.5E-6 and a specific heat of 703. Use a time step of 1E-6. Have Tecplot 360 calculate the drag coefficient and the Nusselt number. Use an ablation temperature of 2,250 and a latent heat of 1.5E5:

```
$!EXTENDEDCOMMAND
COMMANDPROCESSORID = "CFDAnalyzer3"
COMMAND = 'CALCPARTICLEPATH
   TIMESTEP = 1.0e-6
   HAVEMASS = TRUE
   COEFFS = DETAILED
   CALCTEMPERATURE = TRUE
   MASS = 3e-14
   RADIUS = 1.5e-6
   DRAGCOEFFOPTION = CALCULATE
   SPECIFICHEAT = 703
   NUSSELTOPTION = CALCULATE
   TERMOPTION = ABLATE
   TEMPERATURE = 2250
   LATENTHEAT = 1.5e5'
```

CALCTURBULENCEFUNCTION

Syntax:

```
$!EXTENDEDCOMMAND
  COMMANDPROCESSORID = "CFDAnalyzer3"
  COMMAND = 'CALCTURBULENCEFUNCTION [optional parameters]'
```

Description:

Calculate a turbulence-related function from two variables in the current data set. Add the result to the data set as a new variable using the function's name, or overwrite the variable if it already exists.

Optional Parameters

Parameter	Syntax	Default	Notes
CALCULATEONDEMAND	= <boolean>	FALSE	
FUNCTION	= <turbulencefunction>	FREQUENCY	May be ENERGY, DISSIPATIONRATE, FREQUENCY, or VISCOSITY.
ID1	= <turbulencefunction>	ENERGY	The turbulence quantity the first data set variable represents.
VARIABLE1	= <varref>	1	The reference of the first data set variable.
ID2	= <turbulencefunction>	DISSIPATIONRATE	The turbulence quantity the second data set variable represents.
VARIABLE2	= <varref>	2	The reference of the second data set variable.
VALUELOCATION	= <valuelocation>	NODAL	The location of new variables added to the data set. Can be NODAL or CELLCENTERED.

Example:

Calculate turbulent kinematic viscosity from turbulent kinetic energy, variable 5, and turbulent frequency, variable 6:

```
$!EXTENDEDCOMMAND
COMMANDPROCESSORID = "CFDAnalyzer3"
COMMAND = 'CALCTURBULENCEFUNCTION
   FUNCTION=VISCOSITY
   VARIABLE1=5
   ID2=FREQUENCY VARIABLE2=6'
```

CALCULATE

Syntax:

```
$!EXTENDEDCOMMAND
  COMMANDPROCESSORID = "CFDAnalyzer3"
  COMMAND = 'CALCULATE FUNCTION = <functionname> [optional parameters]'
```

Description:

Calculate a Tecplot 360 variable using the specified function and add it to the current data set. If the variable already exists in the current data set, it will be recalculated.

Required Parameter

Parameter	Syntax	Default	Notes
FUNCTION	= <functionname>		Indicates the function to be used to calculate the variable. If it is a vector function, the components will be stored as X name, Y name, and Z name, where name is the function name appearing in the interface.

Optional Parameters

Parameter	Syntax	Default	Notes
NORMALIZATION	= <normalizationoption>	NONE	May be `NONE`, `MAXIMUMMAGNITUDE` or `REFERENCEVALUES`.
VALUELOCATION	<valuelocation>	NODAL	The location of new variables added to the data set. Can be `NODAL` or `CELLCENTERED`.
CALCULATEONDEMAND	= <boolean>	FALSE	

Example 1:

Calculate the Jacobian for the grid of the current data set:

```
$!EXTENDEDCOMMAND
COMMANDPROCESSORID = "CFDAnalyzer3"
COMMAND = 'CALCULATE FUNCTION = JACOBIAN'
```

Example 2:

Calculate the pressure coefficient for the current data set. The freestream density and speed of sound are 1.0 (the defaults):

```
$!EXTENDEDCOMMAND
  COMMANDPROCESSORID = "CFDAnalyzer3"
  COMMAND = 'CALCULATE FUNCTION = PRESSURECOEF'
```

CALCULATEACCURACY

Syntax:

```
$!EXTENDEDCOMMAND
  COMMANDPROCESSORID = "CFDAnalyzer3"
  COMMAND = 'CALCULATEACCURACY ZONES = [<set>]  [optional parameters]'
```

Description:

Calculate the order accuracy of the solution contained in the listed zones. Optionally, plot the overall accuracy versus grid spacing and plot the accuracy at each grid node.

Required Parameter

Parameter	Syntax	Default	Notes
ZONES	= <set>		Indicates the three zones from which to perform the accuracy calculation.

Optional Parameters

Parameter Syntax	Syntax	Default	Notes
MAXACCURACY	= <double>	2.0	The maximum theoretical accuracy of the solver which generated the solution. Used to limit the calculated accuracy.
DATASETVAR	= <varref>	1	The data set variable with which to perform the accuracy calculation.
PLOTDETAILEDACCURACY	= <boolean>	FALSE	If TRUE, a new frame will be created containing the accuracy calculated at each grid node.
PLOTOVERALLACCURACY	= <boolean>	FALSE	If TRUE, a new frame will be created containing the 1-norm and max-norm of the estimated error for each solution zone plotted versus grid resolution.

Example:

Calculate the accuracy using zones 3, 4 and 5, along with data set variable 7, plotting the overall accuracy:

```
$!EXTENDEDCOMMAND
  COMMANDPROCESSORID = "CFDAnalyzer3"
  COMMAND = 'CALCULATEACCURACY ZONES=[3-5]   DATASETVAR=7   PLOTOVERALLACCURACY=TRUE'
```

DISPLAYBOUNDARIES

Syntax:

```
$!EXTENDEDCOMMAND
  COMMANDPROCESSORID = "CFDAnalyzer3"
  COMMAND = 'DISPLAYBOUNDARIES [optional parameters]'
  [RAWDATA [<boundaryrawdata>]]
```

Description:

Displays boundaries corresponding to a geometry and boundaries specification without actually setting the geometry and boundaries. This macro is generally not useful for those writing macro files, but is recorded when the user clicks the Display Boundaries button in the Geometry and Boundaries dialog in order to duplicate the actions of Tecplot 360 that happen in response to that action. See SETGEOMETRYANDBOUNDARIES for a description of the parameters for this macro.

EXTRACTFLOWFEATURE

Syntax:

```
$!EXTENDEDCOMMAND
  COMMANDPROCESSORID = "CFDAnalyzer3"
  COMMAND = 'EXTRACTFLOWFEATURE [optional parameters]'
```

Description:

Extract and display shock surfaces, vortex cores, or separation and attachment lines. Shock surfaces are displayed as iso-surfaces of a new variable, ShockSurface, while vortex cores and separation and attachment lines are displayed as new zones.

Optional Parameters

Parameter	Syntax	Default	Notes
Feature	= <flowfeature>	SHOCKSURFACES	Can be SHOCKSURFACES, VORTEXCORES, or SEPATTACHLINES.
VCOREMETHOD	= <vcoremethod>	EIGENMODES	The vortex core extraction method. Can be VORTICITY or EIGENMODES.
EXCLUDEBLANKED	= <boolean>	FALSE	If TRUE, vortex cores and separation/attachment lines will not be calculated in blanked regions.

Example:

Extract vortex cores using the eigenmodes method:

```
$!EXTENDEDCOMMAND
  COMMANDPROCESSORID = "CFDAnalyzer3"
  COMMAND = 'EXTRACTFLOWFEATURE   FEATURE = VORTEXCORES   VCOREMETHOD = EIGENMODES'
```

EXTRAPOLATESOLUTION

Syntax:

```
$!EXTENDEDCOMMAND
  COMMANDPROCESSORID = "CFDAnalyzer3"
  COMMAND = 'EXTRAPOLATESOLUTION ZONES = <set> [optional parameters]'
```

Description:

Perform Richardson extrapolation to estimate the true solution from three input solutions on grids of successively finer resolution. Two new zones are added to the current data set. The first contains the extrapolated solution, while the second contains the estimated error.

Required Parameter

Parameter	Syntax	Default	Notes
ZONES	= <set>		Indicates the three zones from which to perform the accuracy calculation.

Optional Parameters

Parameter	Syntax	Default	Notes
MAXACCURACY	= <double>	2.0	The maximum theoretical accuracy of the solver which generated the solution. Used to limit the calculated accuracy.

Example:

Extrapolate zones 3, 4, and 5, which were calculated with a second order accurate solver:

```
$!EXTENDEDCOMMAND
  COMMANDPROCESSORID = "CFDAnalyzer3"
  COMMAND = 'EXTRAPOLATESOLUTION ZONES=[3-5]    MAXACCURACY = 2'
```

INTEGRATE

Syntax:

```
$!EXTENDEDCOMMAND
  COMMANDPROCESSORID = "CFDAnalyzer3"
  COMMAND = 'INTEGRATE [<set>] [optional parameters]'
```

Description:

Perform an integration over the specified zones (or time strands, if INTEGRATEBY is set to TIMESTRANDS). If <set> is not specified, the integration will be performed over all zones (or time strands). If PLOTAS is set to TRUE, the integration results will be plotted in a new frame.

Optional Parameters

Parameter	Syntax	Default	Notes
VARIABLEOPTION	= <variableoption>	SCALAR	
XORIGIN	= <double>	0.0	For VARIABLEOPTION = FORCESANDMOMENTS, indicates the origin X-location for moment calculations.
YORIGIN	= <double>	0.0	For VARIABLEOPTION = FORCESANDMOMENTS, indicates the origin Y-location for moment calculations.
ZORIGIN	= <double>	0.0	For VARIABLEOPTION = FORCESANDMOMENTS, indicates the origin Z-location for moment calculations.
SCALARVAR	= <varref>	1	For when VARIABLEOPTION = AVERAGE, MASSWEIGHTEDAVERAGE, WEIGHTEDAVERAGE, MASSFLOWWEIGHTEDAVERAGE, or VECTORAVERAGE. Indicates which variable's average will be calculated.
ABSOLUTE	= <boolean>	FALSE	If TRUE, the absolute value of cell volumes will be used for integration.
EXCLUDEBLANKED	= <boolean>	FALSE	If TRUE, integration will only include non-blanked regions.
XVARIABLE	= <varref>	0	Data set position or name of the scalar variable or X-component of the vector variable to be integrated.

Parameter	Syntax	Default	Notes
YVARIABLE	= <varref>	0	Only required for vector integrations. Indicates the Y-component of the vector variable to be integrated.
ZVARIABLE	= <varref>	0	Only required for vector integrations. Indicates the Z-component of the vector variable to be integrated.
INTEGRATEBY	= <integratebyoption>	ZONES	Indicates whether the integration is performed by zones or by time strands.
INTEGRATEOVER	= <integrateoveroption>	CELLS	Specifies cell volumes, planes, or lines.
IRANGE			
{			
MIN	= <integer>	1	
MAX	= <integer>	0	
SKIP	= <integer>	1	
}			
JRANGE			
{			
MIN	= <integer>	1	
MAX	= <integer>	0	
SKIP	= <integer>	1	
}			
KRANGE			
{			
MIN	= <integer>	1	
MAX	= <integer>	0	
SKIP	= <integer>	1	
}			
PLOTRESULTS	= <boolean>	FALSE	Indicated whether the results of the integration will be plotted in a Tecplot 360 frame.

Parameter	Syntax	Default	Notes
PLOTAS	= <string>	Results	The variable name used to plot integration results. If it contains spaces, surround it with quotes preceded by a backslash (\'). Ignored for forces and moments.
TIMEMAX	= <integer>	Maximum SolutionTime	Can only be used if IntegrateBy is set to "TimeStrands"
TIMEMIN	= <integer>	Minimum SolutionTime	Can only be used if IntegrateBy is set to "TimeStrands"

Range Parameters:

The I-range, J-range and K-range parameters are used to limit the data altered by the equation. The specification of range indices follow the rules below.

- All indices start with one and go to some maximum index m.
- Zero can be used to represent the maximum index *m*; specifying zero tells the command to go to the very last position of the range, that is, the maximum index value m. If the maximum index m = 15, specifying zero sets the range index to 15.
- Negative values represent the offset from the maximum index. If a value of -2 is specified, and the maximum index m is 14, the value used is 14–2, or 12.

You an access your integration results in macros through a variety of specific environment variables. For a list of the variables and how to access them, refer to User's Manual.

Examples

Example 1:

The following command calculates the mass for all zones by integrating density (variable 4) over cell volumes:

```
$!EXTENDEDCOMMAND
  COMMANDPROCESSORID = "CFDAnalyzer3"
  COMMAND = 'INTEGRATE  SCALARVAR = 4'
```

Example 2:

Calculate the mass flux across a series of I = constant planes for zones 1, 2, and 3 and plots the results as "Mass Flux." Since the `COMMAND` string is surrounded by single quotation marks, the quotes surrounding the `PLOTAS` parameter must be preceded by a backslash to avoid a syntax error:

```
$!EXTENDEDCOMMAND
  COMMANDPROCESSORID = "CFDAnalyzer3"
  COMMAND = 'INTEGRATE  [1-3]  VARIABLEOPTION = MASSFLOWRATE INTEGRATEOVER = IPLANES
PLOTRESULTS = TRUE PLOTAS = \'Mass Flux\' '
```

Example 3:

Calculate the "mass-weighted average" (actually the mass flow-weighted average) of total pressure, variable 7:

```
$!EXTENDEDCOMMAND
  COMMANDPROCESSORID = "CFDAnalyzer3"
  COMMAND = 'INTEGRATE   [1-3]   VARIABLEOPTION = ASSFLOWWEIGHTEDAVERAGE SCALARVAR = 7
INTEGRATEOVER = IPLANES PLOTRESULTS = TRUE PLOTAS = \'Mass Weighted Avg Pt\'  '
```

SAVEINTEGRATIONRESULTS

Syntax:

```
$!EXTENDEDCOMMAND
COMMANDPROCESSORID = "CFDAnalyzer3"
COMMAND = 'SAVEINTEGRATIONRESULTS FILENAME = <string>'
```

Description:

Saves the most recently calculated integration results to a text file.

Required Parameter

Parameter	Syntax		Notes
FILENAME	= <string>		Indicates the name of the file to which to save the results. It may be a new or existing file.

Example:

Save the most recent integration results to file E:\users\dave\results.txt. The backslash characters (\) must be escaped with a second backslash character, and the file name is surrounded by quotes (''):

```
$!EXTENDEDCOMMAND
  COMMANDPROCESSORID = "CFDAnalyzer3"
  COMMAND = 'SAVEINTEGRATIONRESULTS FILENAME = "E:\\users\\dave\\results.txt"'
```

SETFIELDVARIABLES

Syntax:

```
$!EXTENDEDCOMMAND
  COMMANDPROCESSORID = "CFDAnalyzer3"
  COMMAND = 'SETFIELDVARIABLES [optional parameters]'
```

Description:

Identifies variables in your data, such as velocity, pressure and temperature, for use in analysis.

Optional Parameters

Parameter	Syntax	Default	Notes
CONVECTIONVARSAREMOMENTUM	= <boolean>	TRUE	Indicates whether the variables designated for Tecplot 360 vector plots are momentum variables (density * velocity). If **FALSE**, then the vector variables must represent velocity values.
UVar	= <varref>	0	Specify the variable to use for the first Vector/Momentum variable.
VVar	= <varref>	0	Specify the variable to use for the second Vector/Momentum variable.
WVar	= <varref>	0	Specify the variable to use for the third Vector/Momentum variable.
ID1	= <varid>	NOTUSED	Identification of the first data set variable from which the function will be calculated.
ID2	= <varid>	NOTUSED	Identification of the second data set variable from which the function will be calculated.
VARIABLE1	= <varref>	0	Position or name of the first variable in the data set.
VARIABLE2	= <varref>	0	Position or name of the second variable in the data set.

SETFLUIDPROPERTIES

Syntax:

```
$!EXTENDEDCOMMAND
COMMANDPROCESSORID = "CFDAnalyzer3"
```

```
COMMAND = 'SETFLUIDPROPERTIES [optional parameters]'
```

Description:

Set the fluid properties for use by other commands.

Optional Parameters

Parameter	Syntax	Default	Notes
INCOMPRESSIBLE	= <boolean>	FALSE	If TRUE, indicates an incompressible fluid.
DENSITY	= <double>	1.0	For INCOMPRESSIBLE = TRUE, indicates the density of the fluid.
SPECIFICHEAT	= <double>	2.5	For INCOMPRESSIBLE = TRUE. The value of the fluid's specific heat.
USESPECIFICHEATVAR	= <boolean>	FALSE	For INCOMPRESSIBLE = TRUE.
SPECIFICHEATVAR	= <varref>	1	For INCOMPRESSIBLE = TRUE and USESPECIFICHEATVAR = TRUE. The data set variable that holds the fluid's specific heat.
GASCONSTANT	= <double>	1.0	For INCOMPRESSIBLE = FALSE. The value of the fluid's specific gas constant.
USEGASCONSTANTVAR	= <boolean>	FALSE	For INCOMPRESSIBLE = FALSE.
GASCONSTANTVAR	= <varref>	1	For INCOMPRESSIBLE = FALSE and USEGASCONSTANTVAR = TRUE. The data set variable which holds the fluid's specific gas constant.
GAMMA	= <double>	1.4	For INCOMPRESSIBLE = FALSE. The value of the fluid's ratio of specific heats. Must be between 1 and 5/3.
USEGAMMAVAR	= <boolean>	FALSE	For INCOMPRESSIBLE = FALSE.
GAMMAVAR	= <varref>	1	For INCOMPRESSIBLE = FALSE and USEGAMMAVAR = TRUE. The data set variable that holds the fluid's ratio of specific heats.
VISCOSITY	= <double>	1.0	The value of the fluid's dynamic viscosity.
USEVISCOSITYVAR	= <boolean>	FALSE	
VISCOSITYVAR	= <varref>	1	For USEVISCOSITYVAR = TRUE. The data set variable which holds the fluid's dynamic viscosity.

Parameter	Syntax	Default	Notes
CONDUCTIVITY	= <double>	1.0	The value of the fluid's conductivity.
USECONDUCTIVITYVAR	= <boolean>	FALSE	
CONDUCTIVITYVAR	= <varref>	1	For USECONDUCTIVITYVAR = TRUE. The data set variable which holds the fluid's conductivity.

Example 1:

Set the fluid properties to standard air values in meters/kilograms/seconds units:

```
$!EXTENDEDCOMMAND
  COMMANDPROCESSORID = "CFDAnalyzer3"
  COMMAND = 'SETFLUIDPROPERTIES    GASCONSTANT=287    VISCOSITY=17.8E-6
CONDUCTIVITY=2.48E-2'
```

Example 2:

Set the fluid properties to incompressible with density equal to 1.0 (the default) and specific heat, viscosity and conductivity taken from data set variables 5, 6, and 7:

```
$!EXTENDEDCOMMAND
COMMANDPROCESSORID = "CFDAnalyzer3"
COMMAND = 'SETFLUIDPROPERTIES INCOMPRESSIBLE=TRUE SPECIFICHEATOPTION=DATASETVAR
SPECIFICHEATVAR=5 VISCOSITYOPTION=DATASETVAR VISCOSITYVAR=6 CONDUCTIVITYOPTION=DATASETVAR
CONDUCTIVITYVAR=7'
```

SETGEOMETRYANDBOUNDARIES

Syntax:

```
$!EXTENDEDCOMMAND
  COMMANDPROCESSORID = "CFDAnalyzer3"
  COMMAND = 'SETGEOMETRYANDBOUNDARIES [optional parameters]'
  [RAWDATA <boundaryrawdata>]
```

Description:

Specify whether the data represent an axisymmetric flow solution (2D Cartesian plots only), whether adjacent zones should be considered to be connected at coincident faces, and specify zone boundaries and their corresponding boundary conditions. Each line of the RAWDATA describes one boundary, and appears in the same format as on the Geometry and Boundaries dialog. For all boundaries, list the

boundary condition and the set of zones, separated by a comma. The index range-type boundary follows this with the boundary face, the first starting index, the first ending index, the second starting index and the second ending index. All entries are separated by commas. The boundary condition is one of INFLOW, OUTFLOW, WALL, SLIPWALL, SYMMETRY, EXTRAPOLATED. The boundary face is one of I=1, I=IMAX, J=1, J=JMAX, K=1, and K=KMAX`. Refer to the User's Manual for more information on boundaries.

Optional Parameters

Parameter	Syntax	Default	Notes
AXISYMMETRIC	= <boolean>	FALSE	Can only be TRUE if the current plot type is 2D Cartesian. If TRUE, indicates that the data represents an axisymmetric solution.
SYMMETRYVAR	= <xory>	Y	For AXISYMMETRIC = TRUE. Can be X or Y. Indicates which axis variable is constant along the axis of symmetry.
SYMMETRYVALUE	= <double>	0.0	For AXISYMMETRIC = TRUE. Indicates the value of the SYMMETRYVAR along the axis of symmetry.
CONNECTZONES	= <boolean>	TRUE	If TRUE, indicates that adjacent zones should be connected where boundary faces coincide.
NODETOLERANCE	= <double>	1.0E-6	Indicates how close two nodes must be before they will be considered coincident for the purpose of matching zone faces.
DEFAULTBC	= <string>	EXTRAPOLATED	Indicates the boundary condition that will be applied to all zone boundary faces not connected to adjacent zones or covered by zone boundaries defined by the RAWDATA section.

Example:

Specify that the solution data represents an axisymmetric solution about X = 1. Do not allow adjacent zones to be connected. Identify two zone-type boundaries and one zone, face and index-range-type boundary:

```
$!EXTENDEDCOMMAND
  COMMANDPROCESSORID = "CFDAnalyzer3"
  COMMAND = 'SETGEOMETRYANDBOUNDARIES AXISYMMETRIC = TRUE SYMMETRYVAR = X SYMMETRYVALUE =
1 CONNECTZONES = FALSE'
RAWDATA
WALL,[2-3]
INFLOW,[4]
OUTFLOW,[1],I=IMAX,1,10,1,20
```

SETREFERENCEVALUES

Syntax:

```
$!EXTENDEDCOMMAND
  COMMANDPROCESSORID = "CFDAnalyzer3"
  COMMAND = 'SETREFERENCEVALUES [optional parameters]'
```

Description:

Specify the reference (free-stream) properties of the solution, identify two variables in the current data set for use with other commands.

Optional Parameters

Parameter	Syntax	Default	Notes
RVELOCITY1ID	= <string>	MACHNUMBER	Identification of the first free-stream velocity component. May be UVELOCITY or MACHNUMBER.
RVELOCITY1	= <double>	0.0	The value of the first free-stream velocity component.
RVELOCITY2ID	= <string>	ANGLEOFATTACK	Identification of the second free-stream velocity component. May be VVELOCITY` or ANGLEOFATTACK.
RVELOCITY2	= <double>	0.0	The value of the second free-stream velocity component. NOTE: RVELOCITY1 must be defined before using RVELOCITY2.
RTHERMO1ID	= <string>	DENSITY	Identification of the first free-stream thermodynamic variable. May be PRESSURE or DENSITY.
RTHERMO1	= <double>	1.0	The value of the first free-stream thermodynamic variable.
RTHERMO2ID	= <string>	SPEEDOFSOUND	Identification of the second free-stream thermodynamic variable. May be TEMPERATURE or SPEEDOFSOUND.
RTHERMO2	= <double>	1.0	The value of the second free-stream thermodynamic variable.

SETUNSTEADYFLOWOPTIONS

Syntax:

```
$!EXTENDEDCOMMAND
  COMMANDPROCESSORID = "CFDANALYZER3"
  COMMAND = 'SETUNSTEADYFLOWOPTIONS [optional parameters]'
[RAWDATA
<timelevelrawdata>]
```

Description:

Identifies time levels for unsteady flow, or specifies that the solution is steady-state. If the flow is unsteady, the solution time levels are specified in the RAWDATA section. The first line of the RAWDATA section must consist of a single integer indicating the number of solution time levels. This must be followed by the time levels themselves. Each time level must be on a separate line and must consist of a floating-point number (the solution time), as well as one or more integers (the zone numbers for that solution time).

Optional Parameters

Parameter	Syntax	Default	Notes
STEADYSTATE	= <boolean>	TRUE	If TRUE, indicates that the solution is steady-state, and the RAWDATA, if any, is ignored. If FALSE, indicates that the solution is unsteady, with time levels identified in the RAWDATA section.

Example:

The unsteady solution contains three solution time levels of two zones each, representing solution times 0.5, 1.0 and 1.5:

```
$!EXTENDEDCOMMAND
  COMMANDPROCESSORID = "CFDAnalyzer3"
  COMMAND = 'SETUNSTEADYFLOWOPTIONS STEADYSTATE = FALSE'
RAWDATA
3
.5   1   2
1.0   3   4
1.5   5   6
```

Parameter Assignment Values

Parameter assignments referenced in the previous section using single angle brackets (<>) not defined in Parameter Assignment Values, Expressions, and Operators are defined here. Note that case is not important.

Value Identifier	Allowable Values
<coeffsoption>	GENERAL, DETAILED
<functionname>	IASPECTRATIO, JASPECTRATIO, KASPECTRATIO, ISTRETCHRATIO, JSTRETCHRATIO, KSTRETCHRATIO, IFACESKEWNESS, JFACESKEWNESS, KFACESKEWNESS, CELLDIAGONAL1SKEWNESS, CELLDIAGONAL2SKEWNESS, IJNORMALSSKEWNESS, JKNORMALSSKEWNESS, KINORMALSSKEWNESS, MAXNORMALSSKEWNESS, IORTHOGONALITY, JORTHOGONALITY, KORTHOGONALITY, MINORTHOGONALITY, INONPLANARITY, JNONPLANARITY, KNONPLANARITY, MINNOPLANARITY, JACOBIAN, CELLVOLUME, GRIDIUNITNORMAL, GRIDJUNITNORMAL, GRIDKUNITNORMAL, DENSITY, STAGDENSITY, PRESSURE, STAGPRESSURE, PRESSURECOEF, STAGPRESSURECOEF, PITOTPRESSURE, PITOTPRESSURERATIO, DYNAMICPRESSURE, TEMPERATURE, STAGTEMPERATURE, ENTHALPY, STAGENTHALPY, INTERNALENERGY, STAGENERGY, STAGENERGYPERUNITVOL, KINETICENERGY, UVELOCITY, VVELOCITY, WVELOCITY, VELOCITYMAG, MACHNUMBER, SPEEDOFSOUND, CROSSFLOWVELOCITY, EQUIVALENTPOTENTIALVELRAT, XMOMENTUM, YMOMENTUM, ZMOMENTUM, ENTROPY, ENTROPYMEASURES1, XVORTICITY, YVORTICITY, ZVORTICITY, VORTICITYMAG, SWIRL, VELOCITYCROSSVORTICITYMAG, HELICITY, RELATIVEHELICITY, FILTEREDRELATIVEHELICITY, SHOCK, FILTEREDSHOCK, PRESSUREGRADIENTMAG, DENSITYGRADIENTMAG, XDENSITYGRADIENT, YDENSITYGRADIENT, ZDENSITYGRADIENT, SHADOWGRAPH, DIVERGENCEOFVELOCITY, SUTHERLANDSLAW, ISENTROPICDENSRAT, ISENTROPICPRESRAT, ISENTROPICTEMPRAT, VELOCITY, VORTICITY, MOMENTUM, PERTURBATIONVELOCITY, VELOCITYCROSSVORTICITY, PRESSUREGRADIENT, DENSITYGRADIENT, VELOCITYGRADIENT
<gravitydirection>	MINUSX, MINUSY, MINUSZ, PLUSX, PLUSY, PLUSZ
<integratebyoption>	ZONES, TIMESTRANDS
<integrateoveroption>	CELLS, IPLANES, JPLANES, KPLANES, ILINES, JLINES, KLINES
<normalizationoption>	NONE, MAXIMUMMAGNITUDE, REFERENCEVALUES
<particlefunction>	PARTICLEPATH, STREAKLINE
<releaseoption>	TIMELEVEL, UNITTIME
<specifyoption>	SPECIFY, CALCULATE
<storeoption>	PARTICLEVALUES, FLUIDVALUES
<terminationoption>	TEMPERATURE, ABLATE

Value Identifier	Allowable Values
<turbulencefunction>	ENERGY, DISSIPATIONRATE, DYNAMICVISCOSITY, FREQUENCY, VISCOSITY
<variableoption>	LENGTHAREAVOLUME, SCALAR, AVERAGE, MASSWEIGHTEDSCALAR, MASSWEIGHTEDAVERAGE, WEIGHTEDAVERAGE, SCALARFLOWRATE, MASSFLOWRATE, MASSWEIGHTEDFLOWRATE, MASSFLOWWEIGHTEDAVERAGE, FORCESANDMOMENTS, VECTORDOTNORMAL, VECTORAVERAGE, VECTORDOTTANGENTIAL
<varid>	PRESSURE, TEMPERATURE, DENSITY, STAGNATIONENERGY, MACHNUMBER, NOTUSED
<xory>	X, Y

Extend Macro

The Extend Macro `COMMANDPROCESSORID` is `extendmcr`. The commands supported by the add-on are listed in the following table

When specifying a macro variable name in an Extend Macro command, you should not surround the variable name with vertical bars. For example, ZONENUM rather than |ZONENUM|.

Table 3. Command Options for Extend Macro

Command	Notes
FILE.APPENDLASTERROR "filename"	Appends the last error message, if any, to the indicated file, creating the file if necessary. FileName may be a relative path from Tecplot's working directory or an absolute path. If no Tecplot errors have occurred, this command will have no effect.
FILE.APPENDSTRING "filename" "string"	Appends the given string to the indicated file, creating the file if necessary. FileName may be a relative path from Tecplot's working directory or an absolute path.
FILE.READSTRING "filename" *nnn VVV*	Read the *nnnth* line from file FileName and stuff the entire string into variable *VVV* .

Command	Notes
QUERY.ACTIVELINEMAPS *VVV*	Get the set of active line maps and put the result in *VVV* . Note: The set string does not include any blank spaces. If linemaps 2, 4, 6, 7, and 8 are active, *VVV* would have the string "2,4,6-8."
QUERY.ACTIVEZONES *VVV*	Get the set of active zones and put the result in *VVV*. Note: The set string does not include any blank spaces. If zones 2, 4, 6, 7 and 8 are active, *VVV* would have the string "2, 4, 6-8."
QUERY.DATASETTITLE *VVV*	Get the string for the dataset title and assign to variable *VVV*
QUERY.FILEEXISTS "filename" *VVV*	If the file exists, *VVV* will be "YES" otherwise *VVV* will be "NO"
QUERY.FIRSTZONEINFIELDMAP *nnn* ZONENUM	Query to find the first zone assigned to the *nnnth* fieldmap.
QUERY.ISADDONLOADED COMMANDPROCESSORID *VVV*	Return "YES" if Add-on `COMMANDPROCESSORID` is loaded, otherwise return "NO"
QUERY.ISLINEMAPACTIVE *nnn VVV*	Test to see if line map *nnn* is active. If so, *VVV* is set to "YES", otherwise it is set to "NO". *VVV* is set to "INVALID" if *nnn* ≤ 0.
QUERY.ISZONEACTIVE ZZZ VVV	Test to see if zone *ZZZ* is active. If so, *VVV* is set to "YES" otherwise it is set to "NO".
QUERY.LINEMAPZONEASSIGNMENT *nnn VVV*	Get the zone number assigned to line map *nnn* and put the result in *VVV*.
QUERY.MAPNAMEBYNUM *nnn VVV*	Returns a string (the name of the map) and places it in variable *VVV* . The active plot must be XY-Line or Polar-Line.

Command	Notes
QUERY.STYLE S1 S2 S3 S4 S5 S6 : *Offset1 Offset2 RESULTSTRVAR* [*RESULTTYPEVAR*] [*RETURNCODEVAR*]	Query for a style setting in Tecplot 360 where: • S1, S2,…S6 are SV parameters • ":" marks the end of the SV parameters. • Offset1 and Offset 2 must always be supplied. Use "1 1" if not used. • *RESULTTYPEVAR* is the name of the variable into which to deposit the result type. This is optional but must be supplied if *RETURNCODEVAR* is supplied. • *RETURNCODEVAR* is the name of the variable into which the return code is deposited. This is optional. It returns the "Get Value" return code for the operation.
QUERY.VARMINMAXINDEX *nnn VZONEMIN VMININDEX VZONEMAX VMAXINDEX*	Query to find the zone, index where the minimum value for variable *nnn* occurs and the zone, index where the maximum value occurs. Results are stored in supplied *Vxxxxx* variable names.
QUERY.VARNAMEBYNUM *nnn VVV*	Get the string for variable *nnn* and assign to variable *VVV*
QUERY.VARNUMBYASSIGNMENT *assignment VVV*	Get the number of variable by assignment and assign to variable *VVV*. *assignment* may have any of the following values: • X, Y or Z - Variable assigned to the X, Y or Z-axis. • U, V or W - Variable assigned to be the U, V or W-vector component. • C -Variable assigned to contours. • S - Variable assigned to scatter sizing. • B - Variable assigned to the first constraint for value-blanking.
QUERY.ZONENAMEBYNUM *nnn VVV*	Get the string for zone *nnn* and assign to variable *VVV*.
QUERY.ZONENUMBYNAME *zonename VVV*	Get the number of zone named *zonename* and assign to variable *VVV*

Command	Notes
STRING.FINDPATTERN "strsource" "strpattern" *VVV*	Get the sub-string from `StrSource` starting at pattern `StrPattern` and going to the end of `StrSource`. Returns "NOTFOUND" if not found.
STRING.LENGTH "strsource" *VVV*	Get the length of string `strsource` and assign to variable *VVV*.
STRING.SUBSTRING "strsource" start end *VVV*	Get the sub-string from `StrSource` starting at position start and ending at position end. Put the result in *VVV*.
STRING.TOKEN "strsource" *nnn VVV*	Get the *nnnth* token from StrSource. Tokens are space or tab delimited.

If you have declared macro variables and would like to use them with the Extend Macro add-on, you can do so by surrounding the command call with single quotes and the macro variable with double-quotes.

For example:

```
$!VarSet |ZoneName| = "Unknown"
$!EXTENDEDCOMMAND
  COMMANDPROCESSORID = "extendmcr"
  Command = 'query.zonenamebynum 1 "ZoneName"'
$!RemoveVar |ZoneName|
```

QUERY.DATASETTITLE

The following example, uses the QUERY.DATASETITLE command to place the title of the dataset at a specific position on the plot.

```
$!VARSET |ZNUM| = "blank"
$!EXTENDEDCOMMAND
  COMMANDPROCESSORID='extendmcr'
  COMMAND='QUERY.DATASETTITLE ZNUM'

$!ATTACHTEXT
  XYPOS
  {
    X = 5
    Y = 90
  }
  TEXT = "Title is: |ZNUM|"
```

QUERY.VARNAMEBYNUM

The following example uses QUERY.VARNAMEBYNUM to place the name of variable 2 at a specific position on the plot.

```
$!VARSET |VNAME| = "X"
$!EXTENDEDCOMMAND
  COMMANDPROCESSORID = 'extendmcr'
  COMMAND='QUERY.VARNAMEBYNUM 2 VNAME'

$!ATTACHTEXT
  XYPOS
  {
    X = 5
    Y = 85
  }
  TEXT = "Var 2 is: |VNAME|"
```

QUERY.ZONENAMEBYNUM

The following example uses QUERY.ZONENAMEBYNUM to place the title of Zone 1 at a specific position on the plot.QUERY.ACTIVEZONES

```
$!VARSET |ZNAME| = "HELLO"
$!EXTENDEDCOMMAND
  COMMANDPROCESSORID = 'extendmcr'
  COMMAND='QUERY.ZONENAMEBYNUM 1 ZNAME'
$!ATTACHTEXT
  XYPOS
  {
    X = 5
    Y = 80
  }
  TEXT = "Zone is:  |ZNAME|"
```

The follow example uses QUERY.ACTIVEZONES to display a list of the active zones.

```
$!VARSET |ZNUMS| = "blank"
$!EXTENDEDCOMMAND
  COMMANDPROCESSORID='extendmcr'
  COMMAND='QUERY.ACTIVEZONES ZNUMS'

$!ATTACHTEXT
  XYPOS
  {
```

```
    X = 5
    Y = 70
  }
  TEXT = "Active zones are: |ZNUMS|"
```

QUERY.STYLE

The following examples use the following "TestQueryStyle" macro function:

```
$!MacroFunction Name="TestQueryStyle"
  $!ExtendedCommand COMMANDPROCESSORID='extendmcr'
    COMMAND='QUERY.STYLE |1| RESULTSTRVAR RESULTTYPEVAR RESULTCODEVAR'
  $!Pause "|1| : |ResultStrVar| : |ResultTypeVar| : |ResultCodeVar|"
$!EndMacroFunction

#
# Prior to running the following commands open a simple layout with a
# field plot...

$!OpenLayout "somefieldplot.lay"

$!RunMacroFunction "TestQueryStyle" ("STREAMATTRIBUTES COLOR : 1 1")
$!RunMacroFunction "TestQueryStyle" ("GLOBALTHREEDVECTOR USERELATIVE : 1 1")
$!RunMacroFunction "TestQueryStyle" ("FIELDMAP MESH COLOR : 2 1")
$!RunMacroFunction "TestQueryStyle" ("GLOBALTWODVECTOR USERELATIVE : 1 1")
$!RunMacroFunction "TestQueryStyle" ("GLOBALTHREEDVECTOR UVAR : 1 1")
$!RunMacroFunction "TestQueryStyle" ("GLOBALTHREEDVECTOR VVAR : 1 1")
$!RunMacroFunction "TestQueryStyle" ("GLOBALTHREEDVECTOR WVAR : 1 1")
$!RunMacroFunction "TestQueryStyle" ("GLOBALTHREEDVECTOR RelativeLength : 1 1")
$!RunMacroFunction "TestQueryStyle" ("ACTIVEFIELDMAPS : 1 1")

# test invalid command name ....
$!RunMacroFunction "TestQueryStyle" ("AACTIVEFIELDMAPS : 1 1")

#.... Open a simple layout with an XY plot
$!OpenLayout "somelineplot.lay"
$!RunMacroFunction "TestQueryStyle" ("LINEMAP NAME : 1 1")

# Test invalid linemap number
$!RunMacroFunction "TestQueryStyle" ("LINEMAP NAME : 4 1")
```

Parameter Subcommands

This chapter details secondary or common macro parameter subcommands in Tecplot 360. These subcommands provide a means to access the lower level variables of commands defined in the

previous chapter of this manual. Each subcommand can expand to contain one or more parameters or subcommands. All parameters within a subcommand are optional.

Items within single angle brackets (<>) are defined in Parameter Assignment Values, Expressions, and Operators.

A-D

<<anchorpos>>

Description:

Assign attributes for positioning of objects.

Expands to:

Parameter	Syntax	Default	Notes
{			
X	= <double>		Sets X-value (and THETA-value)
Y	= <double>		Sets Y-value (and R-value)
Z	= <double>		Sets Z-value
THETA	= <double>		Sets THETA-value (and X-value)
R	= <double>		Sets R-value (and Y-value)
}			

Example:

Make a square geometry and place it at a certain XY location:

```
  $!ATTACHGEOM
  GEOMTYPE = SQUARE
  POSITIONCOORDSYS = FRAME
  ANCHORPOS
    {
    X = 2.89124668435
    Y = 88.7359084881
    }
  RAWDATA
 5.23430593312
```

Description:

Change settings for the axis grid area.

Expands to:

Parameter	Syntax	Default	Notes
{			
DRAWGRIDLAST	= <boolean>		Not available in 3D frame mode.
DRAWBORDER	= <boolean>		
LINETHICKNESS	<op> <dexp>		
COLOR	= <color>		
ISFILLED	= <boolean>		
FILLCOLOR	= <color>		
USELIGHTSOURCETOFIL	= <boolean>		Only available for 3D frame mode.
}			

Example:

Turn on the grid area border for a 2D plot and change the line thickness to be 2 percent:

```
$!TWODAXIS
  AREASTYLE
  {
    DRAWBORDER = YES
    LINETHICKNESS = 2
  }
```

<<axisdetail>>

Description:

Assign attributes for axes.

Expands to:

Parameter	Syntax	Default	Notes
{			
SHOWAXIS	= <boolean>		
AUTOGRID	= <boolean>		

Parameter	Syntax	Default	Notes
ISREVERSED	= <boolean>		
GRANCHOR	= <double>		
GRSPACING	= <double>		
RANGEMIN	= <double>		
RANGEMAX	= <double>		
COORDSCALE	= <coordscale>		2D, XY, and Polar Line plots only.
CLIPDATA	= <boolean>		
VALUEATORIGIN	= <double>		
VAR	= <varref>		Available for 2D and 3D plot types only. Refer to `$!LINEMAP` for information on accessing variable references for XY and Polar Line plots.
TICKLABEL	<<ticklabeldetail>>		
GRIDLINES	<<gridlinedetail>>		
MINORGRIDLINES	<<gridlinedetail>>		
MARKERGRIDLINE	<<gridlinedetail>>		
TICKS	<<tickmarkdetail>>		
TITLE	<<axistitle>>		
AXISLINE	<<axisline>>		
}			

Example:

Turn on the axis line, reverse the axis direction, and set the range to go from 0.5 to 1.5 for the X-axis in a 2D plot:

```
$!TWODAXIS
  SHOWAXISLINE = TRUE
  XDETAIL
  {
    ISREVERSED = TRUE
    RANGEMIN   = 0.5
```

```
    RANGEMAX   = 1.5
  }
```

<<axisline>>

Description:

Assign attributes for axis lines.

Expands to:

Parameter	Syntax	Default	Notes
{			
SHOW	= <boolean>		
SHOWBOTHDIRECTIONS	= <boolean>	FALSE	Non-3D only.
SHOWPERPENDICULAR	= <boolean>	FALSE	Non-3D only.
SHOWOPPOSITEEDGE	= <boolean>	FALSE	3D only
COLOR	= <color>		
LINETHICKNESS	= <double>		
AXISALIGNMENT	<axisalignment>		
OPPOSINGAXISVALUE	= <double>		
POSITION	= <double>		
ANGLE	= <double>		
OFFSET	= <double>		
EDGE	= <integer>		
}			

Example:

Change the thickness of the Theta-axis line to 0.8 and the color to red.:

```
$!POLARAXIS THETADETAIL{AXISLINE{COLOR = RED}}
$!POLARAXIS THETADETAIL{AXISLINE{LINETHICKNESS = 0.8}}
```

<<axistitle>>

Description:

Assign attributes for titles.

Expands to:

Parameter	Syntax	Default	Notes
{			
SHOWONAXISLINE	= <boolean>	TRUE	
SHOWONGRIDBORDERMIN	= <boolean>	FALSE	Non-3D only.
SHOWONGRIDBORDERMAX	= <boolean>	FALSE	Non-3D only.
SHOWONOPPOSITEEDGE	= <boolean>	FALSE	3D only.
SHOWONALLAXES	= <boolean>	TRUE	Polar R only.
SHOWONVIEWPORTTOP	= <boolean>	TRUE	Polar only.
SHOWONVIEWPORTBOTTOM	= <boolean>	TRUE	Polar only.
SHOWONVIEWPORTLEFT	= <boolean>	TRUE	Polar only.
SHOWONVIEWPORTRIGHT	= <boolean>	TRUE	Polar only.
TITLEMODE	= <axistitlemode>		
TEXT	= <string>		
COLOR	= <color>		
TEXTSHAPE	<<textshape>>		
OFFSET	= <double>		
PERCENTALONGLINE	= <double>	50%	
}			

Example:

Create a R-axis title, saying "Harmonic Motion" in red, times, size 6 font.:

```
$!POLARAXIS RDETAIL{TITLE{TEXT = 'Harmonic Motion'}}
$!POLARAXIS RDETAIL{TITLE{OFFSET = -4}}
$!POLARAXIS RDETAIL{TITLE{COLOR = RED}}
$!POLARAXIS RDETAIL{TITLE{TEXTSHAPE{FONTFAMILY = "Times"}}}
$!POLARAXIS RDETAIL{TITLE{TEXTSHAPE{ISBOLD = NO}}}
$!POLARAXIS RDETAIL{TITLE{TEXTSHAPE{ISITALIC = NO}}}
$!POLARAXIS RDETAIL{TITLE{TEXTSHAPE{HEIGHT = 6}}}
```

<<basiccolorcontrol>>

Description:

Assign name to a basic color.

Expands to:

Parameter	Syntax	Default	Notes
{			
NAME	= <string>	""	May include dynamic text variables
SHOW	= <boolean>	YES	Controls visibility of the basic color in a basic color legend.
}			

Example:

Hide Red from the basic color legend; if Blue is employed in the plot, label it "Water":

```
$!BASICCOLORLEGEND
  BASICCOLORCONTROL
  {
    RED
    {
        SHOW = NO
    }
    BLUE
    {
        SHOW = YES
        NAME = "Water"
    }
  }
```

<<basicsizelist>>

Description:

Assign basic sizes. The units for the values assigned here are dependent on the parent command. Assignments here do not affect the plot. These assignments are used only to configure drop-down menus in the interface so the user can make quick selections.

Expands to:

Parameter	Syntax	Default	Notes
{			
TINY	<op> <dexp>		
SMALL	<op> <dexp>		
MEDIUM	<op> <dexp>		
LARGE	<op> <dexp>		
HUGE	<op> <dexp>		
}			

Example:

Change the medium line pattern length for drop-down menus in the interface to be five percent:

```
$!BASICSIZE
 LINEPATLENGTHS
 {
   MEDIUM = 5
 }
```

<<colormapcontrolpoints>>

Description:

All contour color maps except the Raw user-defined color map make use of control points to determine the color distribution. Each control point has a position and a left and right color.

Expands to:

Parameter	Syntax	Default	Notes
{			

Parameter	Syntax	Default	Notes
CONTROLPOINT	<integer>		Use <integer> to specify which control point to modify. You may provide more than one CONTROLPOINT subcommand within the outer {}'s.
{			
COLORMAPFRACTION	= <dexp>		Positions the control point; 0 sets the position to the lowest index and 1 to the highest index in the color map.
LEADRGB	<<rgb>>		
TRAILRGB	<<rgb>>		
}			
}			

<<colormapoverride>>

Description:

Change settings for a color map override. Color map overrides are used to replace a specific band in a contour color map with one of the 16 basic colors.

Expands to:

Parameter	Syntax	Default	Notes
{			
INCLUDE	= <boolean>		
COLOR	= <color>		
STARTLEVEL	<op> < integer>		
ENDLEVEL	<op> < integer>		
}			

Example:

Set the color used between contour level number 1 to number 3 to be purple. Use color map override number 3:

```
$!GLOBALCONTOUR
  COLORMAPFILTER
```

```
  {
    COLORMAPOVERRIDEACTIVE = YES
    COLORMAPOVERRIDE 3
    {
      INCLUDE = YES
      COLOR = PURPLE
      STARTLEVEL = 1
      ENDLEVEL   = 3
    }
  }
```

<<continuouscolor>>

Description:

Change settings for continuous color.

Expands to:

Parameter	Syntax	Default	Notes
CMIN	= <double>		
CMAX	= <double>		

Example:

Set the continuous color.

```
$!GLOBALCONTOUR VAR = 4
$!FIELDLAYERS SHOWCONTOUR = YES

$!GLOBALCONTOUR COLORMAPFILTER
  {COLORMAPDISTRIBUTION = CONTINUOUS}
$!GLOBALCONTOUR COLORMAPFILTER
  {
   CONTINUOUSCOLOR
    {
     CMIN = 0.5
     CMAX = 2
    }
  }
```

G-N

<<gridarea>>

Description:

Change settings for the axis grid area.

Expands to:

Parameter	Syntax	Default	Notes
{			
DRAWBORDER	= <boolean>		Not available in 3D.
LINETHICKNESS	<op> <dexp>		
COLOR	= <color>		Not available for 3D or Polar Line.
ISFILLED	= <boolean>		
FILLCOLOR	= <color>		
USELIGHTSOURCETOFILL	= <boolean>		Only available for 3D.
LABELSALLSIDES	= <boolean>		
TICKSALLSIDES	= <boolean>		
EXTENTS	<<rect>>		Not available in 3D.
}			

Example:

Turn on the grid area border for a 2D plot and change the line thickness to be 2 percent:

```
$!TWODAXIS
 GRIDAREA
 {
   DRAWBORDER = YES
   LINETHICKNESS = 2
 }
```

<<gridlinedetail>>

Description:

Change settings for axis gridlines.

Expands to:

Parameter	Syntax	Default	Notes
{			
COLOR	= <color>		
SHOW	= <boolean>		
LINEPATTERN	= <linepattern>		
PATTERNLENGTH	<op> <dexp>		
LINETHICKNESS	<op> <dexp>		
CUTOFF	= <double>		Theta only.
DRAWGRIDLAST	= <boolean>		Drawing last places the grid at the front of the plot; drawing first places it in the back
POSITIONMARKERBY	= <markerpos>		Marker gridline only
CONSTANTVALUE	= <double>		Marker gridline only; valid only when POSITIONMARKERBY is CONSTANT
}			

Example:

Set the line pattern for minor gridlines for the X-axis in a 3D plot to be dashed:

```
$!THREEDAXIS
  XDETAIL
  {
    MINORGRIDLINES
    {
      LINEPATTERN = DASHED
    }
  }
```

<<header>>

Description:

Set value commands for the legend header

Expands to:

Parameter	Syntax	Default	Notes
{			

Parameter	Syntax	Default	Notes
SHOW	= <boolean>	YES	
TEXTSHAPE	<<textshape>>	FONTFAMILY ="Helvetica", ISBOLD=NO, ISITALIC=NO, SIZEUNITS=FRAME, HEIGHT=2.5	If TEXTTYPE is LATEX, only HEIGHT and SIZEUNITS will affect the LaTeX text.
TEXTTYPE	= <texttype>	REGULAR	Only applied when USECUSTOMTEXT is set to YES. Supported values are LATEX and REGULAR. REGULAR: Produces normal text annotations. LATEX: Instructs Tecplot to pass the text string through the LaTeX toolchain (see User's Manual for more information) to produce images which are then placed in the plot in the Contour Legend Header.
USECUSTOMTEXT	= <boolean>	NO	If NO, the specified Contour Group variable name from the data set will be used.
CUSTOMTEXT	= <string>	""	Accepts REGULAR or LATEX formatted text strings.
}			

Example:

Set the contour legend header to use a custom header text:

```
$!GLOBALCONTOUR
  LEGEND
  {
    HEADER
    {
      SHOW = YES
      TEXTSHAPE
      {
        FONTFAMILY = "Times"
        ISBOLD = NO
        ISITALIC = NO
      }
      USECUSTOMTEXT = YES
      CUSTOMTEXT = "Parcel Pressure"
    }
```

```
  }
```

Example

Set the contour legend header to use a custom header text:

```
$!GlobalContour 1
  Legend
  {
    Header
    {
      UseCustomText = Yes
      CustomText = '\textbf{Pressure} ($\frac{N}{m^2}$)'
      TextType = LaTeX
      TextShape
      {
        SizeUnits = Point
        Height = 27.5
      }
    }
  }
```

<<ijk>>

Description:

Set an I-, J-, or K-index.

Expands to:

Parameter	Syntax	Default	Notes
{			
I	<op> < integer>		
J	<op> < integer>		
K	<op> < integer>		
}			

Example:

Set the I- and J-index skip for vectors to 2 for all zones:

```
$!FIELDMAP
  VECTOR
  {
    IJKSKIP
    {
      I = 2
      J = 2
    }
  }
```

<<indexrange>>

Description:

Set an index range.

Expands to:

Parameter	Syntax	Default	Notes
{			
MIN	<op> <integer>		
MAX	<op> <integer>		
SKIP	<op> <integer>		
}			

Example:

Change the plot so the data set shows I-planes 3, 5, and 7 for zones 1 to 3:

```
$!FIELDMAP [1-3]
  SURFACES
  {
    SURFACESTOPLOT = IPLANES
    IRANGE
    {
      MIN = 3
      MAX = 7
      SKIP = 2
    }
```

```
}
```

<<numberformat>>

Description:

Set the format used to draw a number.

Expands to:

Parameter	Syntax	Default	Notes
{			
FORMATTING	= <valueformat>		
CUSTOMLABEL	= <integer>		
DYNAMICLABELNAME	= <string>		Name of the dynamic label generator to use when "Formatting" is set to "DynamicLabel"
PRECISION	<op> < integer>		
SHOWDECIMALSONWHOLENUMBERS	= <boolean>	FALSE	
REMOVELEADINGZEROS	= <boolean>	FALSE	
SHOWNEGATIVESIGN	= <boolean>	TRUE	
POSITIVEPREFIX	= <string>		
POSITIVESUFFIX	= <string>		
NEGATIVEPREFIX	= <string>		
NEGATIVESUFFIX	= <string>		
TIMEDATEFORMAT	= <string>		
ZEROPREFIX	= <string>		
ZEROSUFFIX	= <string>		
}			

Example:

Set the number format for axis labels on the X-axis in a 2D field plot to use the "float" format with a precision of 3, and add the phrase "DAYS WITHOUT RAIN" after every positive value:

```
$!TWODAXIS
  XDETAIL
  {
    TICKLABEL
    {
      NUMFORMAT
      {
        FORMATTING = FIXEDFLOAT
        PRECISION = 3
        POSITIVESUFFIX = "DAYS WITHOUT RAIN"
      }
    }
  }
```

Example:

Set the number format for axis labels on the X-axis in a 2D field plot to use the Time/Date format. Add the time and date in format

P-Z

<<papersize>>

Description:

Change dimensions or hardclip offsets for LETTER, DOUBLE, A3, A4, CUSTOM1 and CUSTOM2 paper sizes.

Expands to:

Parameter	Syntax	Default	Notes
{			All values are in inches.
WIDTH	<op> <dexp>		
HEIGHT	<op> <dexp>		
LEFTHARDCLIPOFFSET	<op> <dexp>		
RIGHTHARDCLIPOFFSET	<op> <dexp>		
TOPHARDCLIPOFFSET	<op> <dexp>		
BOTTOMHARDCLIPOFFSET	<op> <dexp>		

Parameter	Syntax	Default	Notes
}			

Example:

Change the left hardclip offset for LETTER size paper to be 0.25 inches:

```
$!PAGE
  PAPERSIZEINFO
  {
    LETTER
    {
      LEFTHARDCLIPOFFSET = 0.25
    }
  }
```

<<precisegrid>>

Description:

Change settings for the precise dot grid.

Expands to:

Parameter	Syntax	Default	Notes
{			
INCLUDE	= <boolean>		
COLOR	= <color>		
SIZE	= <double>		Size is in centimeters.
}			

Example:

Turn on the precise dot grid in an XY-plot:

```
$!XYAXIS
  PRECISEGRID
  {
    INCLUDE = YES
  }
```

<<rect>>

Description:

Change settings for a rectangle. The rectangle is defined using two points (X1,Y1) and (X2,Y2).

Expands to:

Parameter	Syntax	Default	Notes
{			Units are based on the parent command.
X1	<op> <dexp>		
Y1	<op> <dexp>		
X2	<op> <dexp>		
Y2	<op> <dexp>		
}			

Example:

Set the 2D axis grid area to be positioned 10 percent from all edges of the frame:

```
$!TWODAXIS
  AREASTYLE
  {
    EXTENTS
    {
      X1 = 10
      Y1 = 10
      X2 = 90
      Y2 = 90
    }
  }
```

<<refscatsymbol>>

Description:

Set the attributes for the reference scatter symbol.

Expands to:

Parameter	Syntax	Default	Notes
{			
SHOW	= <boolean>		

Parameter	Syntax	Default	Notes
COLOR	= <color>		
LINETHICKNESS	= <dexp>		
ISFILLED	= <boolean>		
FILLCOLOR	= <color>		
MAGNITUDE	= <dexp>		
XYZPOS	<<xyz>>		
SYMBOLSHAPE	<<symbolshape>>		
}			

Example:

Change the fill color of the reference scatter symbol to be green:

```
$!GLOBALSCATTER
  REFSCATSYMBOL
  {
    FILLCOLOR = GREEN
  }
```

<<renderconfig>>

Description:

Set the attributes for OpenGL rendering.

Expands to:

Parameter	Syntax	Default	Notes
{			
POLYGONOFFSETEXTBIASFACTOR	= <double>		
STIPPLEALLLINES	= <stipplemode>		If thin patterned lines are not drawn correctly, set STIPPLEALLLINES to ALL.
DEPTHBUFFERSIZE	= <integer>		For low memory graphics cards, the depth buffer size may need to be reduced.
MINBITSPERRGBPLANE	= <integer>		Specify the minimum number of bits used for each of the planes in the image buffer.

Parameter	Syntax	Default	Notes
DOEXTRADRAWFORLASTPIXEL	= <boolean>		Sometimes the last pixel for stroked font characters is not drawn If so, turn DOEXTRADRAWFORLASTPIXEL on.
MAXMULTISAMPLES	= <integer>		Sets the requested number of multi-samples used for anti-aliasing on the graphics card. This value is only a hint and may be ignored by graphics drivers. The Default value for on-screen and off-screen image rendering is 4.
MAXSTRIPLENGTH	= <integer>		Some graphics cards have problems with long strips. Use MAXSTRIPLENGTH to reduce the strip length.
MAXPRIMATIVESPERBLOCK	= <integer>		Some graphics cards have problems with large numbers of graphics primitives in a single block. Use MAXPRIMATIVESPERBLOCK to reduce the number of primitives delivered to the graphics hardware in a single block.
CONSTANTLYUSESCISSORING	= <boolean>		Turn ConstantlyUseScissoring on if you see lines extending outside the borders of the frame. There is a slight performance penalty when using this option.
USEQUADSTRIPS	= <boolean>		If some shaded or contour flooded quads or triangles do not appear or are black, try turning this off.
USETRIANGLESTRIPS	= <boolean>		As with USEQUADSTRIPS, try turning off USEQUADSTRIPS before turning USETRIANGLESTIPS off. Turning off both options will result in reduced performance, but may help fix errors caused by buggy graphics card drivers.
TRIANGULATEFILLEDPOLYGONS	= <boolean>		As with USEQUADSTRIPS, try turning on TRIANGULATEFILLEDPOLYGONS if you are still experiencing problems even after turning off USETRIANGLESTRIPS and USEQUADSTRIPS.
USEGLCOLORMATERIALFUNCTION	= <boolean>		Some graphics cards have problems with an OpenGL's glColorMaterial function. Higher performance (especially for continuous contour flooded plots) can be achieved when it is used. However, it may need to be turned off if you are experiencing problems.

Parameter	Syntax	Default	Notes
MAXTEXTURESIZE	= <integer>		
FORCESMOOTHSHADINGFORLIGHTING	= <boolean>		
ADJUSTRECTANGLERIGHTANDBOTTOM	= <boolean>		
}			

Example:

Force all line drawing to include the last point in the line. Also, make the size of the depth buffer to be at least 32 bits.

```
$!INTERFACE
  OPENGLCONFIG
  {
   SCREENRENDERING
    {
      DOEXTRADRAWFORLASTPIXEL = TRUE
      DEPTHBUFFERSIZE = 32
    }
  }
```

<<rgb>>

Description:

Set a color value by assigning values to its red, green, and blue components.

Expands to:

Parameter	Syntax	Default	Notes
{			
R	<op> <integer>		
G	<op> <integer>		
B	<op> <integer>		
}			

Example:

Change the CUSTOM3 basic color to be light green:

```
$!BASICCOLOR
  CUSTOM 3
  {
    R =  80
    G = 255
    B =  80
  }
```

<<shademap>>

Description:

Map colors on the screen to shades of gray for monochrome hardcopy output.

Expands to:

Parameter	Syntax	Default	Notes
{			Shade values can range from 0 (black) to 100 (white).
BLACKSHADE	= <dexp>		
REDSHADE	= <dexp>		
GREENSHADE	= <dexp>		
BLUESHADE	= <dexp>		
CYANSHADE	= <dexp>		
YELLOWSHADE	= <dexp>		
PURPLESHADE	= <dexp>		
WHITESHADE	= <dexp>		
CUSTOM1SHADE	= <dexp>		
CUSTOM2SHADE	= <dexp>		
CUSTOM3SHADE	= <dexp>		
CUSTOM4SHADE	= <dexp>		
CUSTOM5SHADE	= <dexp>		
CUSTOM6SHADE	= <dexp>		
CUSTOM7SHADE	= <dexp>		

Parameter	Syntax	Default	Notes
CUSTOM8SHADE	= <dexp>		
}			

Example:

Make blue flooded regions map to 50 percent gray:

```
$!PRINTSETUP
  MONOFLOODMAP
  {
    BLUESHADE = 50
  }
```

<<symbolshape>>

Description:

Set a symbol shape. Symbols can be a geometric shape (circle, square, and so forth) or an ASCII character.

Expands to:

Parameter	Syntax	Default	Notes
{			
ISASCII	= <boolean>		
ASCIISHAPE			
{			
USEBASEFONT	= <boolean>		
FONTOVERRIDE	= <font>		
ASCIICHAR	= <string>		
}			
GEOMSHAPE	= <geomshape>		
}			

Example:

Change the symbol shape for symbols drawn with line map 3 to use circles:

```
$!LINEMAP[3]
  SYMBOLS
  {
    SYMBOLSHAPE
    {
      ISASCII = FALSE
      GEOMSHAPE = CIRCLE
    }
  }
```

<<textbox>>

Description:

Change settings for the optional box around a text label.

Expands to:

Parameter	Syntax	Default	Notes
{			
BOXTYPE	= <textboxtype>		
MARGIN	<op> <dexp>		The margin is the space between the text and box. The margin is measured in terms of the percentage of the text height.
LINETHICKNESS	<op> <dexp>		
COLOR	= <color>		
FILLCOLOR	= <color>		
}			

Example:

See example for <<textbox>> incorporated in the example for <<textshape>>

<<textshape>>

Description:

Change settings related to text font and character height.

Expands to:

Parameter	Syntax	Default	Notes
{			
FONTFAMILY	= <string>		
ISBOLD	= <boolean>		
ISITALIC	= <boolean>		
SIZEUNITS	= <sizeunits>		The following combinations of SIZEUNITS and POSITIONCOORDSYS are allowed: FRAME/FRAME, POINT
HEIGHT	<dexp>		
}			

Example:

Add a text label in the center of the frame using Times Roman font. Make the text height 12 point. Include a box around the text with a line thickness of one percent:

```
$!ATTACHTEXT
  XYPOS {  X = 50 Y = 50 }
  TEXTSHAPE { FONTMFFAMILY = "Times" ISBOLD = NO ISITALIC = NO }
  BOX { BOXTYPE = HOLLOW LINETHICKNESS = 1 }
  TEXT = 'Hi Mom'
```

<<ticklabeldetail>>

Description:

Change settings for the text used to label axis tick marks.

Expands to:

Parameter	Syntax	Default	Notes
{			
SHOWONAXISLINE	= <boolean>	TRUE	
SHOWONGRIDBORDERMIN	= <boolean>	FALSE	Non-3D only.
SHOWONGRIDBORDERMAX	= <boolean>	FALSE	Non-3D only.
SHOWONOPPOSITEEDGE	= <boolean>	FALSE	3D only.

Parameter	Syntax	Default	Notes
SHOWONALLAXES	= <boolean>	TRUE	Polar R only.
SHOWATAXISINTERSECTION	= <boolean>		
SKIP	= <integer>		
ERASEBEHINDLABELS	= <boolean>		
NUMFORMAT	<<numberformat>>		
TEXTSHAPE	<<textshape>>		Not allowed to change size units parameter.
OFFSET	<op> <dexp>		
LABELALIGNMENT	= <labelalignment>		
ANGLE	<op> <dexp>		
COLOR	= <color>		
}			

Example:

Change the color for X-axis tick mark labels in a 2D plot to be red:

```
$!TWODAXIS
  XDETAIL
  {
    TICKLABEL
    {
      COLOR = RED
    }
  }
```

<<tickmarkdetail>>

Description:

Assign attributes for axis tick marks.

Expands to:

Parameter	Syntax	Default	Notes
{			

Parameter	Syntax	Default	Notes
SHOWONAXISLINE	= <boolean>	TRUE	
SHOWONGRIDBORDERMIN	= <boolean>	FALSE	Non-3D only.
SHOWONGRIDBORDERMAX	= <boolean>	FALSE	Non-3D only.
SHOWONOPPOSITEEDGE	= <boolean>	FALSE	3D only.
SHOWONALLAXES	= <boolean>	TRUE	Polar R only.
TICKDIRECTION	= <tickdirection>		
LENGTH	<op> <dexp>		
LINETHICKNESS	<op> <dexp>		
NUMMINORTICKS	= <integer>		
MINORLENGTH	= <double>		
MINORLINETHICKNESS	= <double>		
}			

Example:

Set the tick mark length to 2 percent for the second Y-axis in an XY-plot:

```
$!XYLINEAXIS
  YDETAIL 2
  {
    TICKS
    {
      LENGTH = 2
      SHOWONGRIDBORDERMIN = TRUE
    }
  }
```

<<volumeobjectstoplot>>

Description:

Specifies what volume objects are to be displayed.

Expands to:

Parameter	Syntax	Default	Notes
{			
SHOWISOSURFACES	= <boolean>		
SHOWSLICES	= <boolean>		
SHOWSTREAMTRACES	= <boolean>		
}			

Example:

```
$!FIELD
VOLUMEMODE
 {
  VOLUMEOBJECTSTOPLOT
   {
    SHOWISOSURFACES = NO
    SHOWSLICES = YES
    SHOWSTREAMTRACES = YES
   }
 }
```

<<xy>>

Description:

Change settings for an (X,Y) position.

Expands to:

Parameter	Syntax	Default	Notes
{			
X	<op> <dexp>		
Y	<op> <dexp>		
}			

Example:

See the XYPOS parameter in the example for <<textshape>>.

<<xyz>>

Description:

Change settings for an (X, Y, Z) triplet.

Expands to:

Parameter	Syntax	Default	Notes
{			
X	<op> <dexp>		
Y	<op> <dexp>		
Z	<op> <dexp>		
}			

Example:

Change the scale factor on the Z-axis to be 0.5:

```
$!GLOBALTHREED
  AXISSCALEFACT
  {
    Z = 0.5
  }
```

<<zebrashade>>

Description:

Change zebra shading attributes.

Expands to:

Parameter	Syntax	Default	Notes
{			
INCLUDE	= <boolean>		
ISTRANSPARENT	= <boolean>		
COLOR	= <color>		
}			

Example:

Turn on zebra shading and make the zebra shade color to be black:

```
$!GLOBALCONTOUR
  COLORMAPFILTER
  {
    ZEBRA
    {
      INCLUDE = TRUE
      COLOR   = BLACK
    }
  }
```

Parameter Assignment Values, Expressions, and Operators

Assignment Value Table

Parameter assignments referenced in the previous chapters using single angle brackets (<>) are defined here. (Case is not important.)

Value Identifier	Allowable Values		
<altmousebuttonmode>	REDRAW	REVERTTOSELECT	
<addonstyle>	V7STANDARD	V7ACTIVEX	
<anchoralignment>	TOPLEFT	TOPCENTER	TOPRIGHT
	MIDDLELEFT	MIDDLECENTER	MIDDLERIGHT
	BOTTOMLEFT	BOTTOMRIGHT	BOTTOMCENTER
<anglespec>	RADIANS	DEGREES	
<arrowheadattachment>	NONE	ATBEGINNING	ATEND
	ATBOTHENDS		
<arrowheadstyle>	PLAIN	FILLED	HOLLOW

Value Identifier	Allowable Values		
<auxlocation>	DATASET	FRAME	LAYOUT
	LINEMAP	PAGE	VAR
	ZONE		
<axisalignment>	WITHVIEWPORT	WITHSPECIFICANGLE	
	WITHGRIDMIN	WITHGRIDAREABOTTOM	
	WITHGRIDMAX	WITHGRIDAREARIGHT.	
	WITHGRIDAREATOP	WITHGRIDAREALEFT	
	WITHOPPOSINGAXISVALUE		
<axismode>	INDEPENDENT	XYDEPENDENT	XYZDEPENDENT
<axistitlemode>	USEVARNAME	USETEXT	
<axistitleposition>	LEFT	CENTER	RIGHT
<backingstoremode>	REALTIMEUPDATE	NOTUSED	PERIODICUPDATE
<bitdumpregion>	CURRENTFRAME	ALLFRAMES	WORKAREA
<boolean>	YES	NO	
	TRUE	FALSE	
	ON	OFF	
<borderlocation>	IBORDER	JBORDER	KBORDER
<boundarycondition>	FIXED	ZEROGRADIENT	ZERO2ND
<boundingboxmode>	AUTO	OFF	
<boxtype>	NONE	FILLED	HOLLOW
<charactersequence>	One or more printable characters.		
<clipping>	CLIPTOVIEWPORT	CLIPTOFRAME	
<clipplane>	BELOWPRIMARYSLICE	ABOVEPRIMARYSLICE	NONE

Value Identifier	Allowable Values		
<collectingobjectsmode>	INVERTINGADD	ALWAYSADD	HOMOGENEOUSADD
<color>	BLACK	RED	GREEN
	BLUE	CYAN	YELLOW
	PURPLE	WHITE	CUSTOM1 to CUSTOM56
	MULTI to MULTI8[1]	RGBCOLOR	
<colormap>	<standardcolormap>	WILD	USERDEF
	RAWUSERDEF		
<colormapcontrol>	COPYSTANDARD	REDISTRIBUTECONTROLPOINTS	
	RESETTOFACTORY		
<colormapdistribution>	BANDED	CONTINUOUS	
<compressiontype>	BESTSPEED	SMALLESTSIZE	
<conditionalexp>	<dexp> < relop> <dexp>	<string> <relop> <string>	
<constrainintop2mode>	USEVAR	USECONSTANT	
<contourcoloring>	RGB	GROUP1 to GROUP4	
<contourlabelaction>	ADD	DELETEALL	
<contourlabellocation>	COLORMAPDIVISIONS	INCREMENT	CONTOURLEVELS
<contourlevelaction>	ADD	NEW	DELETERANGE
	DELETENEAREST	RESET	
<contourlinemode>	USEZONELINETYPE	SKIPTOSOLID	DASHNEGATIVE

Value Identifier	Allowable Values		
<contourtype>	LINES	FLOOD	AVERAGECELL
	PRIMARYVALUE	BOTHLINESANDFLOOD	
<coordscale>	LINEAR	LOG	
<coordsys>	GRID	FRAME	GRID3D
<curveinfomode>	CURVEDETAILS	CURVEPOINTS	
<curvetype>	LINESEG	CURVFIT	SPLINE
	PARASPLINE	ETORFIT	POWERFIT
	EXTENDED	POLYNOMIALFIT	
<datatype>	SINGLE	DOUBLE	LONGINT
	SHORTINT	BYTE	BIT
<dataloadstrategy>	AUTO	HEAP	
<derivpos>	SIMPLE	ATPOINT	COMPLEX
	ATPOINTB2		
<dexp>	<double> <expression>		
<double>	Valid floating point value.		
<draworder>	BEFOREDATA	AFTERDATA	
<drift>	NONE	LINEAR	QUAD
<edgesetting>	NONE	MIN	MAX
	BOTH		
<edgetype>	BORDERSANDCREASE	BORDERS	CREASES
<epspreviewimagetype>	NONE	TIFF	EPSIV2
	FRAME		
<errorbartype>	UP	DOWN	
	LEFT	RIGHT	
	VERT.	HORZ	
	CROSS		

Value Identifier	Allowable Values		
<exportformat>	The following options are available when exporting single images (via the $!EXPORT command):		
	XWINDOWS	EPS	PNG
	WMF	TIFF	WINDOWSMETAFILE
	PSIMAGE	TECPLOTVIEWER	SUNRASTER
	PS	X3D	BMP
	JPEG	FLASH	
	The following options are available when exporting animations:		
	RASTERMETAFILE	TIFF	BMP
	PNG	AVI	JPEG
	MPEG4	WMV	
<expression>	See Assignment Value Expressions		
<extractionstrandidassignment>	AUTO	DONOTASSIGNSTRANDIDS	ONESTRANDPERSUBEXTRACTION
	ONESTRANDPERGROUP		
<extractmode>	SINGLEZONE	ONEZONEPERCONNECTEDREGION	
	ONEZONEPERSOURCEZONE		
<federivativemethod>	GREENGAUSS	MOVINGLEASTSQUARES	
<fielddatatype>	FLOAT	DOUBLE	
<fillmode>	NONE	USEBACKGROUNDCOLOR	
	USELINECOLOR	USESPECIFICCOLOR	

Value Identifier	Allowable Values		
<font>	HELV	HELVBOLD	TIMES
	TIMESBOLD	TIMESITALIC	TIMESITALICBOLD
	COURIER	COURIERBOLD	GREEK
	MATH	USERDEF	
<frameaction>	DELETETOP	FITALLTOPAPER	POP
	POPATPOSITION	PUSHTOP	
<framecollection>	ALL	PICKED	
<framecoordmode>	PAPER	WORKSPACE	
<framemode>	THREED	TWOD	XY
	SKETCH		
<functiondependency>	XINDEPENDENT	YINDEPENDENT	RINDEPENDENT
	THETAINDEPENDENDT		
<geomshape>	SQUARE	DEL	GRAD
	RTRI	LTRI	DIAMOND
	CIRCLE	CUBE	OCTAHEDRON
	SPHERE	POINT	
<geomtype>	GEOMIMAGE	LINESEGS	RECTANGLE
	SQUARE	CIRCLE	ELLIPSE
	LINESEGS3D		
<ijkblankmode>	INTERIOR	EXTERIOR	
<ijklines>	I	J	K
<ijkplane>	I	J	K
<imagestyle>	ONEPERFRAME	WORKSPACEONLY	
<imagetype>	LOSSLESS	JPEG	256COLOR

Value Identifier	Allowable Values		
<integer>	Integer constants or variables containing an integer. Expressions that logically result in integer are not currently supported.		
<interpptselection>	NEARESTNPOINTS	ALLPOINTS	OCTANTNPOINTS
<isosurfacesselection>	ALLCONTOURLEVELS	TWOSPECIFICVALUES	
	THREESPECIFICVALUES	ONESPECIFICVALUE	
<krigdrift>	NONE	LINEAR	QUAD
<labelalignment>	BYANGLE	ALONGAXIS	
	PERPENDICULARTOAXIS		
<labeltype>	INDEX	VARVALUE	XANDYVARVALUE[2]
<lightingeffect>	PANELED	GOURAUD	
<linearinterpmode>	DONTCHANGE	SETTOCONST	
<linepattern>	SOLID	DASHED	DASHDOT
	DOTTED	LONGDASH	DASHDOTDOT
<linktype>	WITHINFRAME	BETWEENFRAMES	
<macrofunctionvar>	<integer>		

Value Identifier	Allowable Values		
<macrointrinsic>	AXISMAXX	AXISMAXY	AXISMAXZ
	AXISMINX	AXISMINY	AXISMINZ
	COLORMAPDYNAMIC	ENDSLICEPOS	FRAMEMODE
	IS3DV	LOOP	MACROFILEPATH
	MAXB	MAXC	MAXI
	MAXJ	MAXK	MAXS
	MAXU	MAXV	MAXVnn
	MAXW	MAXX	MAXY
	MAXZ	MINB	MINC
	MINS	MINU	MINV
	MINVnn	MINW	MINX
	MINY	MINZ	NUMFRAMES
	NUMPLANES	NUMVARS	NUMWIN
	NUMXYMAPS	NUMZONES	OPSYS
	PLATFORMNAME	SOLUTIONTIME	SLICEPLANETYPE
	STARTSLICEPOS	TECHOME	TECPLOTVERSION
<macrointrinsicvar>	<macrointrinsic>		
<macroparameter>	<charactersequence> <string>		
<macroparameterlist>	<macroparameter> <macroparameter>		
<macrouserdefvar>	<characterseq uence>		
<macrovar>	<macrointrinsicvar> <macrouserdefvar> <macrofunctionvar>		
<markerpos>	SOLUTIONTIME	CONSTANT	
<meshtype>	WIREFRAME	OVERLAY	HIDDENLINE
<mirrorvar>	'X'	'Y'	'Z'
<mousebuttonclick>	REDRAW	REVERTTOSELECT	NOOP

Value Identifier	Allowable Values		
<mousebuttondrag>	NOOP	ZOOMDATA	ZOOMPAPER
	TRANSLATEDATA	TRANSLATEPAPER	
	ROLLERBALLROTATE	TWISTROTATE	
	SPHERICALXROTATEDATA	SPHERICALYROTATEDATA	SPHERICALZROTATEDATA
	XROTATE	YROTATE	ZROTATE
<mousemode>	ADJUST	ADVANCEADJUST	SELECT
<noncurrentframedrawlevel>	FULL	TRACE	
<objectalign>	BOTTOM	CENTER	TOP
	LEFTJUSTIFY	RIGHTJUSTIFY	
<op>	=	-=	+=
	*=	/=	
<originresetlocation>	DATACENTER	VIEWCENTER	
<palette>	MONOCHROME	PENPLOTTER	COLOR
<papergridspacing>	HALFCENTIMETER	ONECENTIMETER	
	TWOCENTIMETERS	QUARTERINCH	HALFINCH
	ONEINCH	TENPOINTS	TWENTYFOURPOINTS
	THIRTYSIXPOINTS	FIFTYPOINTS	
<paperrulerspacing>	ONEINCH	FIFTYPOINTS	ONECENTIMETER
	TWOCENTIMETERS	SEVENTYTWOPOINTS	
<papersize>	LETTER	DOUBLE	A4
	A3	CUSTOM1	CUSTOM2

Value Identifier	Allowable Values		
<pickaction>	ADD	ADDALL	ADDALLINREGION
	CLEAR	COPY	CUT
	EDIT	MAGNIFY	PASTE
	POP	PUSH	SETMOUSEMODE
	SHIFT		
<placementplaneorientation>	X	Y	Z
<plotapproximationmode>	AUTOMATIC	NONCURRENTALWAYSAPPROX	ALLFRAMESALWAYSAPPROX
<plottype>	CARTESIAN3D	CARTESIAN2D	XYLINE
	POLARLINE	SKETCH	
<pointerstyle>	ALLDIRECTIONS	BOTTOM	LEFT
	LEFTRIGHT	LOWERLEFT	LOWERRIGHT
	RIGHT	TOP	UPDOWN
	UPPERLEFT	UPPERRIGHT	
<pointselection>	NEARESTNPOINTS	ALLPOINTS	OCTANTNPOINTS
<pointstoplot>	SURFACESONLY	ALL	
<positionatanchor>	ONCE	NEVER	ALWAYS
<pretranslatemode>	ON	OFF	AUTO
<printerdriver>	PS	EPS	
<printrendertype>	VECTOR	IMAGE	
<quickcolormode>	LINECOLOR	FILLCOLOR	TEXTCOLOR

Value Identifier	Allowable Values		
<rawstring>	Raw strings are of the form: **R**"*delimiter* **(** *raw_characters* **)** *delimiter*" where: *delimiter* is an optional character sequence composed of zero or more printable characters that does not contain parentheses, backslashes, or whitespace *raw_characters* is any character sequence, that does not match the closing delimiter sequence: **)***delimiter*" For most cases an empty delimiter is sufficient. If the preamble needs to contain the two character sequence,)", then a custom delimiter conforming to the raw string format defined above must be used. For example: `Preamble=R"LaTeX(....)LaTeX"`		
<readdataoption>	NEW	APPEND	REPLACE
<relop>	<	>	⇐
	>=	==	!= (not equal to)
	<> (not equal to).	GREATERTHAN	LESSTHAN
	EQUALTO	NOTEQUALTO	
<resizefilter>	TEXTUREFILTER	BOXFILTER	LANCZOS3FILTER
	LANCZOS2FILTER	BELLFILTER	TRIANGLEFILTER
	BSPLINEFILTER	CUBICFILTER	MITCHELFILTER
	GAUSSIANFILTER		
<resulting1dzonetype>	IORDEREDIFPOSSIBLE	FELINESEGMENT	
<rgblegendorientation>	RGB	GBR	BRG
	RBG	BGR	GRB
<rgbmode>	SPECIFYRGB	SPECIFYRG	SPECIFYRB
	SPECIFYGB		

Value Identifier	Allowable Values		
<rotateaxis>	X	Y	Z
	ALPHA	THETA	PSI
	ABOUTVECTOR	TWIST	HORZROLLERBALL
	VERTROLLERBALL		
<rotateoriginlocation>	VIEWER	DEFINEDORIGIN	
<rotationmode>	XYZAXIS	SPHERICAL	ROLLERBALL
<scope>	LOCAL	GLOBAL	
<set>	[<setspecifier> <setspecifier> ...]		
<setspecifier>	<integer>, <integer> - <integer> *[*:<integer>*]*		
<shadetype>	SOLIDCOLOR	PANELED	GOURAUD
	COLOREDPANELED		
<sidebarsizing>	MAXOFALL	DYNAMIC	
<sizeunits>	GRID	FRAME	POINT
<skipmode>	BYINDEX	BYFRAMEUNITS	
<slicesource>	VOLUMEZONES	SURFACEZONES	LINEARZONES
	SURFACESOFVOLUMEZONES		
<slicesurface>	XPLANES	YPLANES	ZPLANES
	IPLANES	JPLANES	KPLANES
	ARBITRARY		
<sortby>	NONE	BYSPECIFICVAR	BYDEPENDENDTVAR
	BYINDEPENDENTVAR		
<spherescatterrenderquality>	LOW	MEDIUM	HIGH.
<standardcolormap>	SMRAINBOW	LGRAINBOW	MODERN
	GRAYSCALE	TWOCOLOR	

Value Identifier	Allowable Values		
<stipplemode>	ALL	CRITICAL	NONE
<streamdirection>	FORWARD	REVERSE	BOTH
<streamdistributionregion>	POINT	RAKE	SURFACESOFACTIVEZONES
	SURFACESOFSELECTEDOBJECTS		
<streamtype>	SURFACELINE	VOLUMELINE	VOLUMERIBBON
	VOLUMEROD	TWODLINE	
<string>	"<charactersequence>" or '<charactersequence>'[3]		
<stylebase>	FACTORY	CONFIG	
<subboundary>	ADD	ADDONLY	ALL
	REMOVE		
<sunrasterformat>	OLDFORMAT	STANDARD	BYTEENCODED
<surfacegenerationmethod>	AUTO	ALLOWQUADS	ALLTRIANGLES
	ALLPOLYGONS		
<surfacestoplot>	BOUNDARYFACES	ALL	IPLANES
	JPLANES	KPLANES	IJPLANES
	JKPLANES	IKPLANES	IJKPLANES
	EXPOSEDCELLFACES	NONE	
<textanchor>	LEFT	CENTER	RIGHT
	MIDLEFT	MIDCENTER	MIDRIGHT
	HEADLEFT	HEADCENTER	HEADRIGHT
<textboxtype>	NONE	FILLED	HOLLOW
<texttype>	REGULAR	LATEX	
<threedviewchangedrawlevel>	FULL	TRACE	
<thetamode>	DEGREES	RADIANS	ARBITRARY
<tickdirection>	IN	OUT	CENTERED
<tiffbyteorder>	INTEL	MOTOROLA	

Value Identifier	Allowable Values		
<timescaling>	LINEAR	LOGARITHMIC	
<transformation>	POLARTORECT	RECTTOPOLAR	SPHERICALTORECT
	RECTTOSPHERICAL		
<transientoperationmode>	SINGLESOLUTIONTIME	ALLSOLUTIONTIMES	
<transientzonevisibility>	ZonesAtSolutionTime	ZonesAtOrBeforeSolutionTime	
<translucency>	Valid integer from one to 99.		
<twoddraworder>	BYZONE	BYLAYER	
<unloadstrategy>	MINIMIZEMEMORYUSE	NEVERUNLOAD	AUTO
<valueblankcellmode>	ALLCORNERS	ANYCORNER	PRIMARYVALUE
<valueblankrelop>	LESSTHANOREQUAL	NOTEQUALTO	GREATERTHAN
	LESSTHAN	EQUALTO	
	GREATERTHANOREQUAL		
<valueformat>	INTEGER	FLOAT	EXPONENT
	BESTFLOAT	SUPERSCRIPT	RANGEBESTFLOAT
	CUSTOMLABEL	TIMEDATE	DYNAMICLABEL
<valuelocation>	AUTO	NODAL	CELLCENTERED
<varloadmode>	BYNAME	BYPOSITION	
<varref>	<integer> or <string>		
<varset>	<set> or **[** <string> <string> <string> **]**		
<vectortype>	TAILATPOINT	HEADATPOINT	MIDATPOINT
	HEADONLY		

Value Identifier	Allowable Values		
<viewmode>	FIT	ZOOM	DATAFIT
	SETMAGNIFICATION	AXISFIT	CENTER
	TRANSLATE	LAST	COPY
	PASTE	PUSH	
<windowfunction>	HANN	HAMMING	RECTANGULAR
	TRIANGULAR		
<workspaceviewmode>	FITSELECTEDFRAMES	FITALLFRAMES	FITPAPER
	MAXIMIZE	LASTVIEW	ZOOM
	TRANSLATE		
<xyaxis>	'X'	'Y'	
<xyzaxis>	'X'	'Y'	'Z'

[1] In order to color an object using one of the contour variable groups (i.e. assigning the color to MULTI, MULTI2, etc.), you must first set the contour variable via the `$!GLOBALCONTOUR` command.
[2] Available in XY-plots only
[3] The only difference in using single quotes vs. double quotes for strings is that single quotes prevent the processing of the backslash character `\` (that is `\n` inserts a newline `\\` inserts the backslash itself).

Assignment Value Expressions

Simple values are literal constants such as 1, 3, 3.5, 2.5e17. Complex expressions are identified by an equation surrounded by `(` and `)` delimiters.

Expressions can be used within any layout or macro file and support all of the common operators and functions familiar to most C and FORTRAN programmers.

Arithmetic operators include the common multiply, divide, add, and subtract (`*`, `/`, `+` and `-`), as well as a few others (`^` and `**`) that are worth noting. The raise operator (`^`, or `**`) returns the result of raising the first number by the second.

Expressions may also contain macro variables and an assortment of useful functions and constants. Following are tables of supported functions and constants and a short explanation for each:

abs(x)	Absolute value of x.
acos(x)	Arc cosine of x between -1 and 1. Return an angle between 0 and p radians.
asin(x)	Arc sine of x between -1 and 1. Return an angle between -p/2 and p/2 radians.

atan(x)	Arc tangent of x. Return an angle between -p and p radians.
atan2(y,x)	Arc tangent of y/x. Return an angle between -p and p radians.
ceil(x)	Smallest integer larger than or equal to x.
cos(x)	Cosine of x in radians.
cosh(x)	Hyperbolic cosine of x.
exp(x)	Exponential of x.
floor(x)	Largest integer smaller than or equal to x.
frac(x)	Fractional part of x.
int(x)	Integer part of x.
log(x)	Natural logarithm of x.
log10(x)	Logarithm to the base 10 of x.
max(x,y)	Larger of x or y.
min(x,y)	Smaller of x or y.
pow(x,y)	x^y
sin(x)	Sine of x in radians.
sinh(x)	Hyperbolic sine of x.
sqrt(x)	Square root of x.
tan(x)	Tangent of x in radians.
tanh(x)	Hyperbolic tangent of x.

Constants are also supported, as listed in the following table.

BASEe	Natural logarithm base e.
DEG	Degrees per radian.
GAMMA	Euler-Mascheroni constant.
PHI	Golden ratio: $\frac{(\sqrt{5}+1)}{2}$
PI	pi.
RAD	Radians per degree.

The following table shows the operator precedence and associativity for assignment value expressions. Operators with higher precedence are listed in the higher rows of the table, while operators that are in the same row have the same precedence. The associativity describes how an operator associates with its operand.

Operator Type	Operators	Associativity
Expression	( )	Left to right.
Power	^ **	Right to left.
Unary	- + !	Right to left.
Multiplicative	* /	Left to right.
Additive	+ -	Left to right.
Relational	> >= < ⇐ == !=	Left to right.
Logical AND	&&	Left to right.
Logical OR	||	Left to right.
Conditional	? :	Right to left.

Unlike C, relational expressions do not evaluate to 0 or 1, instead, they evaluate to true or false. As such, they may only be used with other logical operators, or with the conditional operator.

Examples of common expressions used in the Tecplot 360 macro language follow (note that all expressions evaluate to a simple, <dexp> value):

```
$!If (|b|^2) > (4*|a|*|c|)
    $!If |a| > 0.0
      $!VarSet |root1| = (-|b| + sqrt(|b|^2 - 4*|a|*|c|) / (2*|a|))
      $!VarSet |root2| = (-|b| - sqrt(|b|^2 - 4*|a|*|c|) / (2*|a|))
   $!EndIf
$!EndIf

$!VarSet |area| = (PI*|r|**2)
```

In addition to the more common operators mentioned above, some relational and logical operators are provided to form compound expressions. A relation, *<relation>*, may be constructed and used in conjunction with the conditional operator (? and :) to form compound expressions. The conditional operator (? and :) has the following syntax:

<relation> ? *<expression if true>* : *<expression if false>*

where:

- *<relation>* is a conditional statement that evaluates to true or false, and is formed by any two subexpressions which are compared to one another with one of the relational operators (>, >=, <, ⇐, ==, !=) in combination with zero or more of the logical operators: logical Not !, logical And &&, and logical Or ||).
- *<expression if true>* is the *<expression>* that is evaluated if the *<relation>* condition evaluates to TRUE.

- <*expression if false*> is the <*expression*> that is evaluated if the <*relation*> condition evaluates to FALSE.

Examples of compound expressions used in the Tecplot 360 macro language follow (note that all compound expressions evaluate to a simple, <dexp> value):

```
$!VarSet |value| = (|stress| > |cutoff| ? |cutoff| : |stress|)
$!VarSet |value| = (|x| < 1.5 && |y| <= 5.5 ? |x|^6 : (|x|+|y|)^3.2)
$!VarSet |root| = (|b|^2 > 4*|a|*|c| && |a| > 0.0 ? -|b| + sqrt(|b|^2 - *|a|*|c|) /
(2*|a|) : 0)
```

It is important not to confuse an expression's relation, <*relation*>, that controls the evaluation of a compound expression, with the conditional expression, <*conditionalexp*>, that controls the execution of control commands such as `$!IF` and `$!WHILE`.

For example, the following is a valid macro command since it has a valid expression syntax and a valid control command syntax:

```
$!If |a| > (PI*|r|^2)
   ...
$!EndIf
```

The following is also a valid macro command because, like the last example, it has a valid expression syntax and a valid control command syntax:

```
$!If (|a|^2) == (|b| > 5 ? 1 : 0)
    ...
$!EndIf
```

The following is not a valid macro command since it has an invalid expression syntax and consequently an invalid control command syntax:

```
$!If (|a| > PI*|r|^2)
    ...
$!EndIf
```

As with the invalid example above, if Tecplot 360 encounters a relation, <*relation*>, within an expression, <*expression*> (enclosed within `(` and `)` delimiters), it expects to find the conditional operator (`?` and `:`) and the two required expressions following the specified relation.

Raw Data

Some macro commands contain a "raw data" section. A raw data section is defined by using the keyword `RAWDATA` followed by the raw data values unique to the macro command. Most raw data sections start with a single count value which represents the number of blocks of raw data followed by the blocks of raw data themselves. The following table lists the raw data sections found in Tecplot 360 macros.

Raw Data Name	**Value Type(s) per Block**	**Notes**
<arbfielddata>	N M p11 p12 p13 ... p1M p21 p22 p23 ....p2M ... pN1 pN2 pN3 ...pNM	Where: N is the number of points M is the number of variables per data point.
<colormaprawdata>	<integer> <integer> <integer>	Red. Green. Blue.
<contourlevelrawdata>	<dexp>	Contour level.
<extendedcommandrawdata>	<string>	Each line of the RAWDATA section contains an arbitrary text string. The only requirement is that the character sequence `$!` (a dollar sign followed by an exclamation mark) cannot appear anywhere in the section. Comments can be inserted by using `#` (the octothorp). If encountered, everything to the right of the `#` (including the `#` itself) will be ignored.
<geometryrawdata> (Line segment geometry)	<xyrawdata>	Each block contains a block of <xyrawdata> which forms single polyline within the geometry.
<geometryrawdata> (3D Line segment)	<xyzrawdata>	Each block contains a block of <xyzrawdata> which forms a single polyline within the geometry.
<geometryrawdata> (circle)	<dexp>[1]	Only one value supplied. Value is the radius.

Raw Data Name	Value Type(s) per Block	Notes
<geometryrawdata> (ellipse)	<dexp>[1] <dexp>[1]	Two values supplied. Values are RX and RY.
<geometryrawdata> (rectangle)	<dexp>[1] <dexp>[1]	Two values supplied. Values are width and height.
<geometryrawdata> (square)	<dexp>[1]	Only one value supplied. Value is the width.
<xyrawdata>	<dexp> <dexp>	X Y
<xyzrawdata>	<dexp> <dexp> <dexp>	X Y Z

[1] A count value does not precede the raw data in this case.

Examples:

Example 1:

Raw data for a circle with radius equal to 1.7:

```
RAWDATA
1.7
```

Example 2:

Raw data for a line segment geometry with two segments. Segment 1 has 4 points and segment 2 has 3 points:

```
RAWDATA
2
4
1.5 2.2
1.7 2.4
1.9 2.8
2.1 3.0
3
1.1 1.7
1.2 1.9
1.3 2.0
```

Example 3:

Raw data to define five contour levels:

```
RAWDATA
5
1.5
2.6
3.7
4.9
5.5
```

Example 4:

Raw data to define three RGB values:

```
RAWDATA
3
0   0   0
45  100 100
90  200 200
```

Example 5:

For greater control of contour levels in a macro, set the levels with RAWDATA. This example allows you to choose the number of levels, then sets new levels based on the minimum and maximum values of the current contour variable.

```
$!FIELDLAYERS SHOWCONTOUR = YES
$!Drawgraphics No
$!GLOBALCONTOUR 1 VAR = 4
$!PromptforTextString |numlevels|
Instructions = "Enter the number of contour levels."
$!Varset |Delta| = ((|maxc| - |minc|)/|numlevels|)

$!CONTOURLEVELS DELETERANGE
 CONTOURGROUP = 1
 RANGEMIN = |minc|
 RANGEMAX = |maxc|
$!Varset |newlevel| = (|minc| + |delta|/2)

$!Loop |numlevels|
$!CONTOURLEVELS ADD
 CONTOURGROUP = 1
```

```
 RAWDATA
 1
 |newlevel|

$!Varset |newlevel| += |Delta|
$!Endloop
$!Drawgraphics Yes
$!REDRAW
```

Macro Language Limitations

The only macro control commands allowed in stylesheets and layout files are:

`$!VARSET` and `$!REMOVEVAR`

The only SetValue command allowed in color map files is:

`$!CREATECOLORMAP`

Layout files, stylesheet files and colormap files cannot contain any of the following commands:

```
$!OPENLAYOUT
$!READSTYLESHEET
$!LOADCOLORMAP
```

Only SetValue macro commands are allowed in the Tecplot 360 configuration file.

The `$!LIMITS` command can be used only in the Tecplot 360 configuration file.

The `$!FIELDMAP` and `$!LINEMAP` commands may be used in the configuration file but they may not specify an individual zone or line map. This special use of `$!FIELDMAP` and `$!LINEMAP` allows you to change the default attributes for all zones and line mappings when they are initialized in Tecplot 360.

The file name referenced in the `$!INCLUDEMACRO` command cannot use Tecplot 360 macro variables.

Size limitations:

Maximum number of nested macro function calls	10
Maximum number of nested macro loops	10
Maximum number of nested While-EndWhile loops	Unlimited.
Maximum number of nested If-EndIf loops	Unlimited.
Maximum number of nested macro includes	5

Maximum number of macro commands	200,000
Maximum number of parameters per macro function	20
Maximum number of characters in macro variable name	31
Maximum number of characters in macro function name	Unlimited.
Maximum number of macro variables	400

Copyright

Tecplot 360 Scripting Guide is for use with Tecplot 360 2025 R1.

Tecplot, Inc.
Post Office Box 52708
Bellevue, WA 98015-2708 U.S.A.

Tel:1.800.763.7005 (within the U.S. or Canada), 00 1 (425) 653-1200 (internationally)

E-mail: sales@tecplot.com, support@tecplot.com
Questions, comments or concerns regarding this document: support@tecplot.com

For more information, visit www.tecplot.com

Part Number: 23-360-07-2 Build Revision {CI_PIPELINE_ID}

Released: 06/2025

For third-party trademark and copyright information, see the User's Manual.