EnSight
Command Language Manual
for Version 7.6

Computational Engineering International, Inc.
2166 N. Salem Street, Suite 101, Apex, NC 27523
USA • 919-363-0883 • 919-363-0833 FAX
http://www.ceintl.com or http://www.ensight.com

EN-CL Revision History

EN-CL:1.0-1	September 2001
EN-CL:1.1-1	March 2002
EN-CL:1.1-2	October 2002
EN-CL:1.2-1	May 2003

This document has been reviewed and approved in accordance with Computational Engineering International, Inc. Documentation Review and Approval Procedures.

This document should be used only for Version 7.6 and greater of the EnSight program.

CEI's World Wide Web addresses:
http://www.ceintl.com
or
http://www.ensight.com

Restricted Rights Legend

Major Sections

Major Sections

Introduction

Purpose

The purpose of this manual is to document the usage and syntax of EnSight's command language in order to assist users interested in modifying and, in some circumstances, writing command files. The manual is organized alphabetically by command. Each command is documented in terms of syntax followed by any applicable notes and an example.

This manual is not intended for users not well acquainted with EnSight. The manual does not contain a thorough description of functionality, philosophies, usage, etc. and should be used in close reference to the EnSight User Manual for Version 7.6 and EnSight How To Manual. To aid in this, each command is cross referenced back to the section index which in turn references appropriate sections in EnSight's User and How To manuals.

Warning

EnSight's command language is really a journaling language. It is NOT intended to be generated externally to the software. Accordingly, users are encouraged to modify existing command files with care and discouraged from writing all but the simplest files from scratch.

Since the language is intended for journaling, EnSight's command interpreter is very unforgiving. While command syntax is checked by the interpreter the appropriateness of the command is not. The user can, for example, issue commands to select a clip part and modify the parts isosurface value. The result of inappropriate commands can be program termination.

Compatibility

The current form of EnSight's command language was first introduced in the 5.0 release in 1993. Previous to this a non user readable language existed back to the first release in 1989.

The command language introduced in 1993 has remained largely compatible through each new release. In the cases that the command language is changed great effort is taken to handle backward compatibility within EnSight's command interpreter. EnSight's release notes (found on the top directory of the release CD and as a pdf document under the doc directory) document any command language incompatibilities that might exist with previous releases. This chapter also contains an Incompatibility Section which describes command language incompatibilities starting with EnSight 7.4.

Organization and Notation

Each documented command begins on a new page with the name of the command as the page header. The command is first described together with it's arguments. The arguments to the command are shown within <>'s. For example:

```
plot: axis_x_origin <origin>
```

indicates that the command must be issued as "`plot: axis_x_origin`" followed by a single argument called "`origin`".

Each argument is shown in the Parameter section which describes the argument type and purpose.

Argument types are as follows:

int	An integral value, i.e., values do not contain decimal places. For example, -10, 0, 123
float	Any numeric input which may or may not contain a decimal or exponent. For example, 10.2, -123, 1.e3
string	Any combination of printable characters. For example, Pressure, /tmp/myfilename
constant	Indicates that the argument must be a character string as documented. For example, ON, OFF, mesh

Required

Any command file (or macro file) read by EnSight must begin with the following line:

```
VERSION x.xx
```

where x.xx is the version for the command language. In all cases the version number corresponds to the version of EnSight in use. For example, "`VERSION 7.01`"

The version number is used by the command interpreter so that any language incompatibilities may be taken into account.

Examples

Three complete examples are shown below. The first and third examples use data from the "data" directory of the EnSight release CD, while the second example uses data from the "other_data" directory.

Example 1

```
VERSION 7.40
prefs: icon_help_labels OFF
prefs: auto_legend_replace ON
prefs: tool_tips OFF
prefs: plane_tool_type line
prefs: view_mode_allowed OFF
prefs: frame_mode_allowed OFF
#
# Above: The version number is required.  Preferences are optional.
#
# Below: Read unstructured external flow dataset.
#
data: binary_files_are big_endian
data: format case
data: path /scratch/data/ensight76/data/ami
data: geometry ami.case
data: read
data_partbuild: begin
#
# Load/build external 3D flow field part
# to exist but not be visual on client.
#
part: select_default
part: modify_begin
```

```
part: elt_representation not_loaded
part: modify_end
data_partbuild: data_type unstructured
data_partbuild: select_begin
 1
data_partbuild: select_end
data_partbuild: description
data_partbuild: create
#
# Load/build 2D shell part.
#
part: select_default
part: modify_begin
part: elt_representation 3D_border_2D_full
part: modify_end
data_partbuild: data_type unstructured
data_partbuild: select_begin
 2
data_partbuild: select_end
data_partbuild: description
data_partbuild: create
data_partbuild: end
#
# Turn on shading
#
view: hidden_surface ON
#
# Activate pressure variable, and
# Color 2D part by pressure.
#
variables: activate pressure
part: select_begin
 2
part: select_end
part: modify_begin
part: colorby_palette pressure
part: modify_end
#
# Transform (rotate, zoom, translate) part globally.
#
view_transf: rotate -74.198471 -3.497139 0.000000
view_transf: function global
view_transf: rotate 27.206108 67.268570 0.000000
view_transf: function global
view_transf: zoom 0.519211
view_transf: function global
view_transf: zoom 0.681123
view_transf: function global
view_transf: translate 0.058393 2.262181 0.000000
view_transf: function global
view_transf: rotate -5.770993 2.262856 0.000000
view_transf: function global
view_transf: zoom 1.208323
view_transf: function global
view_transf: translate -2.160550 -1.404112 0.000000
view_transf: function global
#
# Toggle-on pressure legend.
#
legend: select_palette_begin
```

```
pressure
legend: select_palette_end
legend: visible ON
#
# Create iso-surface part from pressure on 3D part.
#
part: select_begin
 1
part: select_end
isos: begin
isos: variable pressure
isos: value 1.0000e+00
isos: type isosurface
isos: end
isos: create
part: select_begin
 3
part: select_end
#
# Modify iso-surface value.
#
part: select_begin
 3
part: select_end
part: modify_begin
isos: value 9.5334e-01
part: modify_end
#
# Color iso-surface part.
#
part: select_begin
 3
part: select_end
part: modify_begin
part: colorby_palette pressure
part: modify_end
#
# Color iso-part by velocity.
# (First need to activate the velocity variable.)
#
variables: activate velocity
part: select_begin
 3
part: select_end
part: modify_begin
part: colorby_palette velocity
part: modify_end
#
# Create text annotating the current iso-value of the iso-part.
#
legend: select_default
view_transf: function global
text: new_text Iso-value = <\\pval "%.2f" 3\\>
text: select_begin
 0
text: select_end
#
# Modify location of text annotation string.
#
text: select_begin
```

```
 0
text: select_end
text: location_x 2.2489e-01
text: location_y 8.8415e-01
text: select_begin
 0
text: select_end
#
# Modify legend location attributes.
#
text: select_default
legend: select_palette_begin
pressure
legend: select_palette_end
legend: location_x 9.5201e-01
legend: location_y 9.0488e-02
legend: width 3.7991e-02
legend: height 3.3171e-01
legend: format %.2e
legend: title below
legend: select_palette_begin
pressure
legend: select_palette_end
#
# Restore global transformation mode.
#
view_transf: function global
#
# Delete (currently selected) iso-part.
#
part: delete
#
# Delete corresponding iso-annotation.
#
view_transf: function global
text: select_begin
 0
text: select_end
legend: select_default
text: select_begin
 0
text: select_end
text: location_x 2.2489e-01
text: location_y 8.8415e-01
text: select_begin
 0
text: select_end
text: select_default
legend: select_palette_begin
pressure
legend: select_palette_end
text: delete
text: select_default
view_transf: function global
#
# Modify pressure palette.
#
function: palette pressure
function: modify_begin
function: type continuous
```

```
function: scale linear
function: limit_fringes no
function: display_undefined by_model_color
function: #_of_levels 5
function: edit_level 1
function: value 8.0982e-01
function: edit_level 2
function: value 8.7701e-01
function: edit_level 3
function: value 9.4421e-01
function: edit_level 4
function: value 1.0114e+00
function: edit_level 5
function: value 1.0786e+00
function: edit_level 1
function: RGB 0.0000e+00 0.0000e+00 1.0000e+00
function: edit_level 2
function: RGB 0.0000e+00 1.0000e+00 1.0000e+00
function: edit_level 3
function: RGB 0.0000e+00 1.0000e+00 0.0000e+00
function: edit_level 4
function: RGB 1.0000e+00 1.0000e+00 0.0000e+00
function: edit_level 5
function: RGB 1.0000e+00 0.0000e+00 0.0000e+00
function: modify_end
#
# Add another level to the pressure palette.
#
function: modify_begin
function: #_of_levels 6
function: edit_level 2
function: value 8.6357e-01
function: edit_level 3
function: value 9.1733e-01
function: edit_level 4
function: value 9.7109e-01
function: edit_level 5
function: value 1.0248e+00
function: edit_level 6
function: value 1.0786e+00
function: edit_level 2
function: RGB 0.0000e+00 8.0000e-01 1.0000e+00
function: edit_level 3
function: RGB 0.0000e+00 1.0000e+00 4.0000e-01
function: edit_level 4
function: RGB 4.0000e-01 1.0000e+00 0.0000e+00
function: edit_level 5
function: RGB 1.0000e+00 8.0000e-01 0.0000e+00
function: edit_level 6
function: RGB 1.0000e+00 0.0000e+00 0.0000e+00
function: modify_end
#
# Toggle-off legend.
#
legend: select_palette_begin
pressure
legend: select_palette_end
legend: visible OFF
#
# Create 2D-clip x-plane part.
```

```
#
part: select_begin
 1
part: select_end
clip: begin
clip: value 1.000000
clip: domain intersect
clip: tool xyz
clip: end
clip: create
part: select_begin
 3
part: select_end
#
# Modify x-plane value.
#
part: select_begin
 3
part: select_end
part: modify_begin
clip: value 26.677700
part: modify_end
#
# Color clip plane by velocity.
#
part: select_begin
 3
part: select_end
part: modify_begin
part: colorby_palette velocity
part: modify_end
#
# Save image to file “image1”.
#
file: image_file image1
file: save_image
```

Example 2

```
VERSION 7.40
prefs: icon_help_labels OFF
prefs: auto_legend_replace ON
prefs: tool_tips OFF
prefs: plane_tool_type line
prefs: view_mode_allowed OFF
prefs: frame_mode_allowed ON
command: part_selection_by number
#
# Above: Required version number followed by optional preferences.
#
# Below: Read transient unstructured dataset.
#        (A structures application, dropping a cylinder on a rail.)
#
data: binary_files_are big_endian
data: format case
data: path /scratch/data/ensight76/other_data/ensight/dyna
data: geometry dyna.case
data: start_time 16
```

```
data: read
data_partbuild: begin
#
# Load/build both 2D parts
# (cylinder-collar part and rail part)
#
data_partbuild: data_type unstructured
data_partbuild: select_all
data_partbuild: description
data_partbuild: create
data_partbuild: end
#
# Shade parts (default part color).
#
view: hidden_surface ON
#
# Color parts by Stress.
#
variables: activate Stress
part: select_all
part: modify_begin
part: colorby_palette Stress
part: modify_end
#
# Displace parts by the displacement variable.
#
variables: activate Displacement
part: select_all
part: modify_begin
part: displace_by Displacement
part: modify_end
#
# Transform parts.
#
view_transf: rotate 12.091602 -32.914276 0.000000
view_transf: function global
view_transf: zoom 0.751079
view_transf: function global
view_transf: translate -0.014558 1.264062 0.000000
view_transf: function global
#
# Change time step.
#
solution_time: current_step 1.0000e+01
solution_time: update_to_current
#
# Return to final time step.
#
solution_time: current_step 1.6000e+01
solution_time: update_to_current
#
# Load/play transient flipbook.
#
anim_flipbook: begin_time_step 0
anim_flipbook: end_time_step 16
anim_flipbook: specify_time_as step
anim_flipbook: load
anim_flipbook: run_type auto
#
# Step thru flipbook page-by-page.
```

```
#
anim_flipbook: run_type step
anim_flipbook: page 1.000000
anim_flipbook: page 2.000000
anim_flipbook: page 3.000000
anim_flipbook: page 4.000000
anim_flipbook: page 5.000000
anim_flipbook: page 6.000000
anim_flipbook: page 7.000000
anim_flipbook: page 8.000000
anim_flipbook: page 9.000000
anim_flipbook: page 10.000000
anim_flipbook: page 11.000000
anim_flipbook: page 12.000000
anim_flipbook: page 13.000000
anim_flipbook: page 14.000000
anim_flipbook: page 15.000000
anim_flipbook: page 16.000000
anim_flipbook: page 0.000000
#
# Modify flipbook settings.
#
anim_flipbook: display_speed 0.500000
anim_flipbook: run_type auto
#
# Sync flipbook with keyframe animation.
# (Keyfame using the loaded flipbook objects.)
#
anim_keyframe: keyframing ON
anim_keyframe: create_keyframe
anim_keyframe: create_keyframe
anim_screens: type full
anim_keyframe: run
#
# Delete flipbook.
#
anim_flipbook: delete_all_pages
#
# Keyframe w/transient data using same settings.
# (Keyframe by loading each transient time step.)
#
anim_keyframe: use_transient_data ON
anim_keyframe: run
#
# Delete keyframe animation segment.
#
anim_keyframe: keyframing OFF
#
# Query maximum stress over time.
#
query_ent_var: modify_begin
query_ent_var: variable_y Stress
query_ent_var: variable_x TIME
query_ent_var: modify_end
part: select_begin
 1
part: select_end
query_ent_var: begin
query_ent_var: description
query_ent_var: query_type generated
```

```
query_ent_var: #_of_sample_pts 17
query_ent_var: begin_simtime 0.0000e+00
query_ent_var: end_simtime 1.6000e+01
query_ent_var: constrain max
query_ent_var: sample_by value
query_ent_var: variable_y Stress
query_ent_var: generate_over time
query_ent_var: variable_x TIME
query_ent_var: end
query_ent_var: query
curve: select_begin
 0
curve: select_end
#
# Plot maximum stress over time.
#
curve: assign rescale Time vs. Maximum Stress
plot: select_begin
 0
plot: select_end
#
# Modify plot settings (size and add grid).
#
view_transf: function global
plot: origin_x 0.0000e+00
plot: width 4.6849e-01
plot: height 5.3018e-01
plot: axis_x_gridtype grid
plot: axis_y_gridtype grid
view_transf: function global
#
# Save image.
#
file: image_file image1
file: save_image
```

Example 3

```
VERSION 7.40
prefs: icon_help_labels OFF
prefs: auto_legend_replace ON
prefs: tool_tips OFF
prefs: plane_tool_type line
prefs: view_mode_allowed OFF
prefs: frame_mode_allowed OFF
command: part_selection_by number
#
# Above: Required version number followed by optional preferences.
# Below: Read PLOT3D structured dataset
#
data: binary_files_are big_endian
data: format plot3d
data: plot3diblank OFF
data: plot3dmulti_zone OFF
data: plot3dread_as c_binary
data: plot3ddimension 3d
data: path /scratch/data/ensight76/data/plot3d
data: geometry shuttle.grid
data: result shuttle.q
```

```
data_partbuild: begin
data: read
#
# Specify external 3D flow field around shuttle
# to load only on the server and not visible on the client.
#
part: select_default
part: modify_begin
part: elt_representation not_loaded
part: modify_end
#
# Load/build external 3D flow field around shuttle.
#
data_partbuild: data_type structured
data_partbuild: select_begin
 1
data_partbuild: select_end
data_partbuild: domain all
data_partbuild: noderange_i 1 80
data_partbuild: noderange_j 1 63
data_partbuild: noderange_k 1 45
data_partbuild: nodestep 1 1 1
data_partbuild: nodedelta 0 0 0
data_partbuild: description flow field
data_partbuild: create
#
# Specify 2D shell part of shuttle
# to load on server and visible on client.
#
part: select_default
part: modify_begin
part: elt_representation 3D_border_2D_full
part: modify_end
#
# Load/build 2D shell part of shuttle.
#
data_partbuild: data_type structured
data_partbuild: select_begin
 1
data_partbuild: select_end
data_partbuild: domain all
data_partbuild: noderange_i 1 52
data_partbuild: noderange_j 1 63
data_partbuild: noderange_k 1 1
data_partbuild: nodestep 1 1 1
data_partbuild: nodedelta 0 0 0
data_partbuild: description shuttle
data_partbuild: create
data_partbuild: end
#
# Turn on shading
#
view: hidden_surface ON
#
# Transform shuttle part into a forward facing view.
#
view_transf: rotate -81.618317 -3.497139 0.000000
view_transf: function global
view_transf: rotate 2.473281 115.199997 0.000000
view_transf: function global
```

```
view_transf: rotate 21.709923 4.525713 0.000000
view_transf: function global
view_transf: zoom 0.779425
view_transf: function global
view_transf: zoom 0.799537
view_transf: function global
view_transf: zoom 0.904418
view_transf: function global
view_transf: translate -0.191782 -0.017080 0.000000
view_transf: function global
#
# Apply visual symmetry to shuttle part.
#
part: select_begin
 2
part: select_end
part: modify_begin
part: mirror_y ON
part: modify_end
#
# Compute area of shuttle part.
#
part: select_begin
 2
part: select_end
variables: evaluate Area = Area(plist)
#
# Enable extended CFD  variables list.
#
varextcfd: modify_begin
varextcfd: freestream_mach 3.0000e-01
varextcfd: modify_end
varextcfd: show_extended ON
#
# Color shuttle part by velocity
# (which first activates velocity prior to coloring).
#
variables: activate Velocity
part: select_begin
 2
part: select_end
part: modify_begin
part: colorby_palette Velocity
part: modify_end
#
# Set cursor location for particle trace.
#
view: pick_mode cursor
view_transf: cursor 0.179618 0.084309 0.007263
tools: cursor ON
view_transf: function cursor
view_transf: translate 0.008001 0.033193 -0.027716
view_transf: function global
#
# Create a streamline emitted from the cursor.
#
ptrace: select_default
part: modify_begin
ptrace: variable Velocity
part: modify_end
```

```
part: select_begin
 1
part: select_end
ptrace: create_pt
part: select_begin
 3
part: select_end
#
# Apply visual symmetry to streamline part.
#
part: modify_begin
part: mirror_y ON
part: modify_end
#
# Animate streamline.
#
part: modify_begin
ptrace: animate ON
part: modify_end
#
# Toggle-off visibility of streamline part
# to help see animated tracers better.
#
part: modify_begin
part: visible OFF
part: modify_end
#
# Add multiple pulses and represent tracer head as sphere.
#
anim_traces: multiple_pulses ON
anim_traces: head_type sphere
#
# Size head of tracer by velocity.
#
anim_traces: head_size_by vector_mag
anim_traces: head_variable Momentum
anim_traces: head_variable Momentum
anim_traces: head_variable Velocity
anim_traces: head_variable Velocity
#
# Color tracers by velocity.
#
part: modify_begin
part: colorby_palette Velocity
part: modify_end
#
# Toggle-off cursor tool.
#
tools: cursor OFF
#
# Color shuttle by pressure.
#
variables: activate Pres
part: select_begin
 2
part: select_end
part: modify_begin
part: colorby_palette Pres
part: modify_end
#
```

```
# Toggle-on pressure legend.
#
legend: select_palette_begin
Pres
legend: select_palette_end
legend: visible ON
#
# Toggle-off animated traces.
#
part: select_begin
 3
part: select_end
part: modify_begin
ptrace: animate OFF
part: modify_end
#
# Redisplay streamline part,
# and color them white.
#
part: modify_begin
part: visible ON
part: colorby_palette none
part: colorby_rgb 1.0000e+00 1.0000e+00 1.0000e+00
part: modify_end
#
# Save a JPEG image to file "image1".
#
file: image_format jpeg
file: image_file image1
file: save_image
```

Portability

Command language is portable between computer architectures. This is also true between UNIX and Windows except for differences in file pathname conventions between the two systems, i.e., a filename under UNIX might be /mysystem/scratch/tmp/xyz.geo while under windows the file system might be mounted on the G drive such that the file would be G:\tmp\xyz.geo.

Nearly all command files will at some point need to select one or more parts for subsequent operation. By default this is done by part number. For example, the following selects parts 1, 2, and 3 and makes them invisible:

```
part: select_begin
1 2 3
part: select_end
part: modify_begin
part: visible OFF
part: modify_end
```

Using part numbers, however, decreases portability since future data files that could take advantage of an existing command file might not contain the same number of parts. To eliminate this problem consider using part selection by name (you can also set this up to be the default recording behavior in EnSight by going to Edit->Preferences->General User Interface and changing from Number to Name). The above example would be equivalent to:

```
part: select_partname_begin
```

```
"(CASE:Case 1)3d space frame"
"(CASE:Case 1)green arrow"
"(CASE:Case 1)blue arrow"
part: select_partname_end
part: modify_begin
part: visible OFF
part: modify_end
```

While this produces a larger, more verbose command file it will be more portable.

Begin/End Constructs

All commands which modify attributes prior to part creation, a part's default attributes, or modify an existing part must be bracketed by a begin/end construct. If an existing part is being modified or the default attribute is being changed a "part: modify_begin", "part: modify_end" pair is used. When used prior to part creation a "xxxx: begin", "xxxx: end" is used, where xxxx is the name of the part type (for example "clip: begin"). The two examples below are actually equivalent. The first modifies the default attributes for clips and then creates a part while the second example creates the clip directly without modifying the clip defaults.

Example 1

```
#
# modify the default attributes for clips
#
clip: select_default
part: modify_begin
clip: tool xyz
clip: mesh_plane X
clip: value 2.0
part: modify_end
#
# select the parent parts
#
part: select_begin
1 2 3 4
part: select_end
clip: create
```

Example 2

```
#
# select the parent parts
#
part: select_begin
1 2 3 4
part: select_end
clip: begin
clip: tool xyz
clip: mesh_plane X
clip: value 2.0
clip: end
clip: create
```

Other commands

The three commands below are not documented elsewhere in this manual but can prove useful for debugging purposes.

Command	Purpose
interrupt:	Stops the playback of a command file. (It can be continued from the GUI)
play: filename	Plays the command file indicated by filename
exit:	Terminates the program

There are also a number of test commands which may be executed by EnSight. To learn more about these type the command “test: list” in EnSight’s command dialog.

Incompatibilities

In the future this section may contain discussions on incompatibilities between the command language defined in EnSight 7.4 and the then current release.

Commands in this Section:

anim_flipbook

anim_flipbook: begin_simtime <time_value>
anim_flipbook: begin_time_step <step_value>
anim_flipbook: cycle <toggle>
anim_flipbook: delete_all_pages
anim_flipbook: display_speed <speed_value>
anim_flipbook: dynamic_line <part> <end> <X0> <Y0> <Z0> <Xn> <Yn> <Zn>
anim_flipbook: dynamic_plane <part> <cid> <X0> <Y0> <Z0> <Xn> <Yn> <Zn>
anim_flipbook: dynamic_value <part> <start_value> <end_value>
anim_flipbook: end_simtime <time_value>
anim_flipbook: end_time_step <step_value>
anim_flipbook: interactive_part <option>
anim_flipbook: load
anim_flipbook: load_as <option>
anim_flipbook: load_type <option>
anim_flipbook: move_dynamic_plane <part> <transform> <dx> <dy> <dz>
anim_flipbook: number_to_create <number>
anim_flipbook: page <page_value>
anim_flipbook: regen_all_pages <toggle>
anim_flipbook: run_type <option>
anim_flipbook: save_images <filename_prefix>
anim_flipbook: save_multiple_images <toggle>
anim_flipbook: save_win_size <x_size> <y_size>
anim_flipbook: save_win_type <size>
anim_flipbook: select_all_timesteps
anim_flipbook: show_from_page <page_value>
anim_flipbook: show_to_page <page_value>
anim_flipbook: specify_time_as <option>
anim_flipbook: step_by <step_increment>
anim_flipbook: step_simtime <time_value>
anim_flipbook: time <time_value>

See Also:

User Manual - Section 7.14, Flipbook Animation

How To Create a Flipbook Animation

How To Animate Transient Data

anim_flipbook: begin_simtime

Command:

```
anim_flipbook: begin_simtime <time_value>
```

Purpose:

Specify a beginning simulation time value to start the flipbook animation.

Parameters:

Type	Parameter	Description
float	`time_value`	beginning simulation time value (which must correspond to the same simulation time at a time step)

Notes:

Not applicable if dataset is not transient.

Used only if "`anim_flipbook: specify_time_as simulation`" has been specified.

Example Usage:

```
#
# Specify a transient_data flipbook
#
anim_flipbook: load_type transient_data
#
# Set begin/end solution timestep values
#
solution_time: show_as time
#
# Set begin/end flipbook time info
#
anim_flipbook: step_simtime 1.0000e+00
anim_flipbook: specify_time_as simulation
anim_flipbook: begin_simtime 3.4900e+02
anim_flipbook: end_simtime 3.6900e+02
#
# Load flipbook
#
anim_flipbook: load
#
# Run loaded flipbook
#
anim_flipbook: run_type auto
```

See Also:

Associated anim_flipbook commands

Command:

```
anim_flipbook: begin_time_step <step_value>
```

Purpose:

Specify a beginning time step value to start the flipbook animation.

Parameters:

Type	Parameter	Description
int	`step_value`	beginning time step value in the range of 0 to n - 1, where n = total # of time steps

Notes:

Not applicable if dataset is not transient.

Used only if "`anim_flipbook: specify_time_as step`" has been specified.

Example Usage:

```
#
# Specify a transient_data flipbook
#
anim_flipbook: load_type transient_data
#
# Set begin/end solution timestep values
#
solution_time: begin_step 30
solution_time: end_step 50
solution_time: show_as step
#
# Set begin/end flipbook time info
#
# Note: both step and simtime commands are reset
#       because solution time commands were set above
#
anim_flipbook: specify_time_as step
anim_flipbook: step_by 1.0000e+00
anim_flipbook: begin_time_step 30
anim_flipbook: end_time_step 50
#
# Load flipbook
#
anim_flipbook: load
#
# Run loaded flipbook
#
anim_flipbook: run_type auto
```

See Also:

Associated anim_flipbook commands

anim_flipbook: cycle

Command:

```
anim_flipbook: cycle <toggle>
```

Purpose:

Specify the automatic playback sequence of the flipbook.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	alternate playing the flipbook forward and backward
		`OFF`	(default) always replay the flipbook from beginning to end

Notes:

Not applicable if flipbook has not been loaded.

Example Usage:

```
anim_flipbook: cycle ON
```

See Also:

Associated anim_flipbook commands

Command:

```
anim_flipbook: delete_all_pages
```

Purpose:

Specify to delete the currently loaded flipbook.

Parameters:

None.

Notes:

Not applicable if flipbook has not been loaded.

Example Usage:

```
anim_flipbook: delete_all_pages
```

See Also:

Associated anim_flipbook commands

anim_flipbook: display_speed

Command:

```
anim_flipbook: display_speed <speed_value>
```

Purpose:

Specifies the playback-speed factor, or the display speed at which the flipbook animates.

Parameters:

Type	Parameter	Description
float	`speed_value`	set the display speed value in the range [0.,1.] where 1. is the fastest speed (of your hardware) and 0. is stopped (a still page)

Notes:

Not applicable if flipbook has not been loaded.

Example Usage:

```
#
# Halve the display speed
#
anim_flipbook: display_speed 0.500000
```

See Also:

Associated anim_flipbook commands

Command:

```
anim_flipbook: dynamic_line <part> <end> <X0> <Y0> <Z0> <Xn> <Yn> <Zn>
```

Purpose:

Specify dynamic flipbook animation of the selected 1d-clip line part.

Parameters:

Type	Parameter	Description
int	part	a valid GUI part number, for a line clip
int	end	1 - beginning line position
		2 - ending line position
float	X0	X coordinate of first end point of specified line position
	YO	Y coordinate of first end point of specified line position
	ZO	Z coordinate of first end point of specified line position
	Xn	Y coordinate of second end point of specified line position
	Yn	X coordinate of second end point of specified line position
	Zn	Z coordinate of second end point of specified line position

Notes:

Only used for flipbook animations of interactive clip line parts.

Must be bracketed by "`anim_flipbook: interactive_part start`" and "`anim_flipbook: interactive_part end`".

Example Usage:

```
#
# Create the line clip part from the parent field
#
part: select_all
clip: begin
clip: domain intersect
clip: tool line
clip: line 1 5.8348e-01 1.3165e+00 1.2670e+00
clip: line 2 3.0835e+00 1.3165e+00 1.2670e+00
clip: end
clip: create
part: select_begin
 2
part: select_end
#
# Load and animate the clip-line part
# between the 1st and 2nd dynamic_line positions.
#
anim_flipbook: interactive_part start
tools: line ON
part: select_begin
 2
part: select_end
view_transf: function line
view_transf: translate 1.33481 0.816460 0.266963
view_transf: function global
```

```
anim_flipbook: dyanmic_line 2 1 -0.750000 0.500000 1.000000 0.583581
1.316460 1.266963
anim_flipbook: dyanmic_line 2 2  1.750000 0.500000 1.000000 3.083481
1.316460 1.266963
anim_flipbook: interactive_part stop
anim_flipbook: number_to_create 10
anim_flipbook: load
anim_flipbook: run_type auto
tools: line OFF
```

See Also:

Associated anim_flipbook commands

Command:

```
anim_flipbook: dynamic_plane <part> <cid> <X0> <Y0> <Z0> <Xn> <Yn> <Zn>
```

Purpose:

Specify dynamic flipbook animation of the selected 2d-clip plane part.

Parameters:

Type	Parameter	Description
int	part	GUI part number of the 2D-clip plane part to be animated
	cid	corner id (1, 2, 3, or 4) of the plane tool
float	X0	initial x coordinate of the specified corner of the plane
	Y0	initial y coordinate of the specified corner of the plane
	Z0	initial z coordinate of the specified corner of the plane
	Xn	final x coordinate of the specified corner of the plane
	Yn	final y coordinate of the specified corner of the plane
	Zn	final z coordinate of the specified corner of the plane

Notes:

Used for flipbook animations of interactive isosurface, ijk-clip, and xyz-clip parts. Only applicable between commands:

```
anim_flipbook: interactive_part start
anim_flipbook: interactive_part stop
```

Example Usage:

```
#
# From the flow field part (number 1),
# create the 2D-clip plane part (number 2)
# and color it by temperature.
#
part: select_begin
 1
part: select_begin
clip: begin
clip: domain intersect
clip: tool plane
clip: plane 1 -7.5000e-01 -7.5000e-01 1.0000e+00
clip: plane 2 1.7500e+00 -7.5000e-01 1.0000e+00
clip: plane 3 1.7500e+00 1.7500e+00 1.0000e+00
clip: end
clip: create
part: select_begin
 2
part: select_end
part: modify_begin
part: colorby_palette temperature
part: modify_end
#
# Specify interactive flipbook animation
# of the 2D-clip part.
#
anim_flipbook: interactive_part start
tools: plane line
```

```
view_transf: function plane
view_transf: action translate
view_transf: translate 0.0000e+00 0.0000e+00 1.0000e+00
part: select_begin
 2
part: select_end
part: modify_begin
clip: plane 1 -7.5000e-01 -7.5000e-01 2.0000e+00
clip: plane 2 1.7500e+00 -7.5000e-01 2.0000e+00
clip: plane 3 1.7500e+00 1.7500e+00 2.0000e+00
part: modify_end
tools: plane line
tools: plane line
view_transf: function global
view_transf: function global
#
# Track the dynamic position of the four corners
# of the clip plane tool.
#
anim_flipbook: dynamic_plane 2 1 -0.750000 -0.750000 1.000000 -0.750000
-0.750000 2.000000
anim_flipbook: dynamic_plane 2 2 1.750000 -0.750000 1.000000 1.750000 -
0.750000 2.000000
anim_flipbook: dynamic_plane 2 3 1.750000 1.750000 1.000000 1.750000
1.750000 2.000000
anim_flipbook: dynamic_plane 2 4 -0.750000 1.750000 1.000000 -0.750000
1.750000 2.000000
anim_flipbook: move_dynamic_plane 2 5 0.000000 0.000000 1.000000
anim_flipbook: interactive_part stop
#
anim_flipbook: number_to_create 10
anim_flipbook: load
tools: plane OFF
anim_flipbook: run_type auto
anim_flipbook: run_type off
```

See Also:

Associated anim_flipbook commands

Command:

```
anim_flipbook: dynamic_value <part> <start_value> <end_value>
```

Purpose:

Specify dynamic flipbook animation of the selected isosurface, xyz-clip, or ijk-clip part, within the appropriate specified dynamic ranges.

Parameters:

Type	Parameter	Description
int	part	GUI part number of the part to be animated
float	start_value	the starting dynamic value of the animation
float	end_value	final dynamic value of the animation, i.e. the ending isosurface value; x, y, or z clip value; or i, j, or k value of the respective isosurface, xyz clip, or ijk clip part

Notes:

Used for flipbook animations of interactive isosurface, ijk-clip, and xyz-clip parts. Only applicable between commands:

```
anim_flipbook: interactive_part start
anim_flipbook: interactive_part stop
```

Example Usage:

```
#
# (Note: A clip part has already been generated via the
#        XYZ clip part feature and colored by a variable.)
#
# Update the 2D clip part clip at
#
part: select_begin
 2
part: select_end
part: modify_begin
clip: mesh_plane X
clip: tool xyz
clip: value 1.000000
part: modify_end
#
# Specify type of flipbook
#
anim_flipbook: load_type create_data
#
# Setup interactive flipbooking of the clip part number 2
# from values 1. to 3.
#
anim_flipbook: interactive_part start
part: select_begin
 2
part: select_end
part: modify_begin
clip: value 3.000000
part: modify_end
anim_flipbook: dynamic_value 2 1.000000 3.000000
anim_flipbook: interactive_part stop
#
```

```
# Load 10 flipbook pages
#
anim_flipbook: number_to_create 10
anim_flipbook: load
#
# Run loaded flipbook pages
#
anim_flipbook: run_type auto
```

See Also:

Associated anim_flipbook commands

Command:

```
anim_flipbook: end_simtime <time_value>
```

Purpose:

Specify an ending simulation time value to start the flipbook animation.

Parameters:

Type	Parameter	Description
float	`time_value`	ending simulation time value (which must correspond to the simulation time at a time step)

Notes:

Not applicable if dataset is not transient. An ending simulation time must correspond to the simulation time of a time step.

Used only if "`anim_flipbook: specify_time_as step`" has been specified.

Example Usage:

```
#
# Specify a transient_data flipbook
#
anim_flipbook: load_type transient_data
#
# Set begin/end solution timestep values
#
solution_time: begin_step 30
solution_time: end_step 50
solution_time: show_as time
#
# Set begin/end flipbook time info
#
anim_flipbook: step_simtime 1.0000e+00
anim_flipbook: specify_time_as simulation
anim_flipbook: begin_simtime 3.4900e+02
anim_flipbook: end_simtime 3.6900e+02
#
# Load flipbook
#
anim_flipbook: load
#
# Run loaded flipbook
#
anim_flipbook: run_type auto
```

See Also:

Associated anim_flipbook commands

anim_flipbook: end_time_step

Command:

```
anim_flipbook: end_time_step <step_value>
```

Purpose:

Specify an ending time step value to start the flipbook animation.

Parameters:

Type	Parameter	Description
int	`step_value`	ending time step value in the range of 0 to n - 1, where n = total # of timesteps

Notes:

Not applicable if dataset is not transient.

Used only if "`anim_flipbook: specify_time_as step`" has been specified.

Example Usage:

```
#
# Specify a transient_data flipbook
#
anim_flipbook: load_type transient_data
#
# Set begin/end solution timestep values
#
solution_time: begin_step 30
solution_time: end_step 50
solution_time: show_as step
#
# Set begin/end flipbook time info
#
# Note: both step and simtime commands are reset
#       because solution time commands were set above
#
anim_flipbook: step_by 1.0000e+00
anim_flipbook: specify_time_as step
anim_flipbook: begin_time_step 30
anim_flipbook: end_time_step 50
#
# Load flipbook
#
anim_flipbook: load
#
# Run loaded flipbook
#
anim_flipbook: run_type auto
```

See Also:

Associated anim_flipbook commands

Command:

```
anim_flipbook: interactive_part <option>
```

Purpose:

Specify the start and stop of the recording of interactive movement of any specified isosurface or clip parts during a flipbook load.

Parameters:

Type	Parameter	Description	
string	option	start	begin (turn-on) recording of interactive movements of specified parts
		stop	(default) end (turn-off) recording of interactive movement of specified parts

Notes:

Currently this operation only works for interactive manipulation of isosurface and 2D clip parts. Used in conjunction with the following commands:

```
part: select_begin
part: select_end
part: modify_begin
part: modify_end
anim_flipbook: dynamic_value
anim_flipbook: dynamic_line
anim_flipbook: dynamic_plane
anim_flipbook: dynamic_plane
anim_flipbook: move_dynamic_plane
```

Example Usage:

```
#
# (Note: A clip part has already been generated via the
#        XYZ clip part feature and colored by a variable.)
#
# Update the 2D clip part clip at
#
part: select_begin
 2
part: select_end
part: modify_begin
clip: mesh_plane X
clip: tool xyz
clip: value 1.000000
part: modify_end
#
# Specify type of flipbook
#
anim_flipbook: load_type create_data
#
# Setup interactive flipbooking of the clip part
#
anim_flipbook: interactive_part start
part: select_begin
 2
part: select_end
part: modify_begin
```

```
clip: value 3.000000
part: modify_end
anim_flipbook: dynamic_value 2 1.000000 3.000000
anim_flipbook: interactive_part stop
#
# Load 10 flipbook pages
#
anim_flipbook: number_to_create 10
anim_flipbook: load
#
# Run loaded flipbook pages
#
anim_flipbook: run_type auto
```

See Also:

Associated anim_flipbook commands

Command:

```
anim_flipbook: load
```

Purpose:

Specify the loading of the flipbook pages, according to the selected part(s) and specified flipbook options.

Parameters:

None.

Example Usage:

```
#
# Note: transient dataset w/83 time steps...
#
# Specify a transient_data flipbook
#
anim_flipbook: load_type transient_data
#
# Specify page increment
#
anim_flipbook: step_by 1.0000e+01
#
# Set begin/end flipbook time info
#
anim_flipbook: begin_time_step 0
anim_flipbook: end_time_step 83
anim_flipbook: specify_time_as step
#
# Load flipbook pages {0,10,20,30,40,50,60,70,80}
#
anim_flipbook: load
#
# Run loaded flipbook
#
anim_flipbook: run_type auto
```

See Also:

Associated anim_flipbook commands

anim_flipbook: load_as

Command:

```
anim_flipbook: load_as <option>
```

Purpose:

Specify whether to load the flipbook as graphic objects or images.

Parameters:

Type	Parameter	Description	
string	`option`	`graphic_objects`	(default) record the flipbook animation as a collection of graphic objects which may be transformed while running the flipbook. (Playback performance depends on the visual complexity of the model.)
		`graphic_images`	record the flipbook animation as a collection of graphic images which may not be transformed while running the flipbook. (Playback performance depends on window size.)

Notes:

Since `graphic_images` record the image on the screen, make sure there is no overlaying dialog or image overlapping the main graphics window.

Example Usage:

```
anim_flipbook: load_as graphic_images
```

See Also:

Associated anim_flipbook commands

Command:

```
anim_flipbook: load_type <option>
```

Purpose:

Specifies the type of flipbook animation to load.

Parameters:

Type	Parameter	Description	
string	option	transient_data	(default) animation changes in coloration and/or shape, resulting from changes in the transient nature of the data
		mode_shapes	animation changes in the mode shape of the corresponding part(s), resulting from a displacement variable
		create_data	animation changes in isosurface and/or clip parts, resulting from their change in values or corresponding movement
		linear_load	animation changes in the displacement (vector) variable of corresponding parts, resulting from linear interpolating the displacement field from its zero to maximum value. (The variable values or colors also update according to the linearly displaced values.

Example Usage:

```
#
# Example of <transient_data>
#
# Note: transient dataset w/83 time steps...
#
# Specify a transient_data flipbook
#
anim_flipbook: load_type transient_data
#
# Specify page increment
#
anim_flipbook: step_by 1.0000e+01
#
# Set begin/end flipbook time info
#
anim_flipbook: begin_time_step 0
anim_flipbook: end_time_step 83
anim_flipbook: specify_time_as step
#
# Load flipbook pages {0,10,20,30,40,50,60,70,80}
#
anim_flipbook: load
#
# Run loaded flipbook
#
anim_flipbook: run_type auto
```

```
#
# Example of <create_data>
#
# (Note: A clip part has already been generated via the
#        XYZ clip part feature and colored by a variable.)
#
# Update the 2D clip part clip at
#
part: select_begin
 2
part: select_end
part: modify_begin
clip: mesh_plane X
clip: tool xyz
clip: value 1.000000
part: modify_end
#
# Specify type of flipbook
#
anim_flipbook: load_type create_data
#
# Setup interactive flipbooking of the clip part
#
anim_flipbook: interactive_part start
part: select_begin
 2
part: eelect_end
part: modify_begin
clip: value 3.000000
part: modify_end
anim_flipbook: dynamic_value 2 1.000000 3.000000
anim_flipbook: interactive_part stop
#
# Load 10 flipbook pages
#
anim_flipbook: number_to_create 10
anim_flipbook: load
#
# Run loaded flipbook pages
#
anim_flipbook: run_type auto


#
# Example of <mode_shapes>
#
# Note:  Prior to animating modes shapes ... activate and
#        displace displacement vector variable on selected parts.
#
variables: activate 3D_Displacement
part: select_all
part: modify_begin
part: displace_by 3D_Displacement
part: modify_end
#
#
#
anim_flipbook: load_type mode_shapes
anim_flipbook: number_to_create 10
anim_flipbook: load
```

```
anim_flipbook: run_type auto


#
# Followed by example of <linear_load>
#
# Delete current flipbook
#
anim_flipbook: delete_all_pages
#
# Load and run 10 linear_load flipbook pages
#
anim_flipbook: load_type linear_load
anim_flipbook: number_to_create 10
anim_flipbook: load
anim_flipbook: run_type auto
```

See Also:

Associated anim_flipbook commands

anim_flipbook: move_dynamic_plane

Command:

```
anim_flipbook: move_dynamic_plane <part> <transform> <dx> <dy> <dz>
```

Purpose:

Specify the vector in which the 2d-clip plane part is to be transformed.

Parameters:

Type	Parameter	Description
int	`part`	GUI part number of the part to be animated
	`transform`	1 - rotation transformation 3 - scale transformation 5 - translation transformation
float	`dx`	x-component of the direction vector for the transformation to follow
	`dy`	y-component of the direction vector for the transformation to follow
	`dz`	z-component of the direction vector for the transformation to follow

Notes:

Only currently applicable for "`anim_flipbook: dynamic_plane`" commands, i.e. for flipbook animations of interactive clip parts via the plane tool.

Example Usage:

```
#
# From the flow field part (number 1),
# create the 2D-clip plane part (number 2)
# and color it by temperature.
#
part: select_begin
 1
part: select_begin
clip: begin
clip: domain intersect
clip: tool plane
clip: plane 1 -7.5000e-01 -7.5000e-01 1.0000e+00
clip: plane 2 1.7500e+00 -7.5000e-01 1.0000e+00
clip: plane 3 1.7500e+00 1.7500e+00 1.0000e+00
clip: end
clip: create
part: select_begin
 2
part: select_end
part: modify_begin
part: colorby_palette temperature
part: modify_end
#
# Specify interactive flipbook animation
# of the 2D-clip part.
#
anim_flipbook: interactive_part start
tools: plane line
view_transf: function plane
view_transf: action translate
view_transf: translate 0.0000e+00 0.0000e+00 1.0000e+00
part: select_begin
```

```
 2
part: select_end
part: modify_begin
clip: plane 1 -7.5000e-01 -7.5000e-01 2.0000e+00
clip: plane 2 1.7500e+00 -7.5000e-01 2.0000e+00
clip: plane 3 1.7500e+00 1.7500e+00 2.0000e+00
part: modify_end
tools: plane line
tools: plane line
view_transf: function global
view_transf: function global
#
# Track the dynamic position of the four corners
# of the clip plane tool.
#
anim_flipbook: dynamic_plane 2 1 -0.750000 -0.750000 1.000000 -0.750000
-0.750000 2.000000
anim_flipbook: dynamic_plane 2 2 1.750000 -0.750000 1.000000 1.750000 -
0.750000 2.000000
anim_flipbook: dynamic_plane 2 3 1.750000 1.750000 1.000000 1.750000
1.750000 2.000000
anim_flipbook: dynamic_plane 2 4 -0.750000 1.750000 1.000000 -0.750000
1.750000 2.000000
#
# Indicate the 2d-clip plane's translation
#
anim_flipbook: move_dynamic_plane 2 5 0.000000 0.000000 1.000000
anim_flipbook: interactive_part stop
#
anim_flipbook: number_to_create 10
anim_flipbook: load
tools: plane OFF
anim_flipbook: run_type auto
anim_flipbook: run_type off
```

See Also:

Associated anim_flipbook commands

anim_flipbook: number_to_create

Command:

```
anim_flipbook: number_to_create <number>
```

Purpose:

Specify the number of pages to create in the flipbook animation.

Parameters:

Type	Parameter	Description
int	number	(positive) number of pages to create

Example Usage:

```
#
# (Note: A clip part has already been generated via the
#        XYZ clip part feature and colored by a variable.)
#
# Update the 2D clip part clip at
#
part: select_begin
 2
part: select_end
part: modify_begin
clip: mesh_plane X
clip: tool xyz
clip: value 1.000000
part: modify_end
#
# Specify type of flipbook
#
anim_flipbook: load_type create_data
#
# Setup interactive flipbooking of the clip part
#
anim_flipbook: interactive_part start
part: select_begin
 2
part: eelect_end
part: modify_begin
clip: value 3.000000
part: modify_end
anim_flipbook: dynamic_value 2 1.000000 3.000000
anim_flipbook: interactive_part stop
#
# Load 10 flipbook pages
#
anim_flipbook: number_to_create 10
anim_flipbook: load
#
# Run loaded flipbook pages
#
anim_flipbook: run_type auto
```

See Also:

Associated anim_flipbook commands

Command:

```
anim_flipbook: page <page_value>
```

Purpose:

Specify which page of the flipbook animation to display.

Parameters:

Type	Parameter	Description
float	page_value	page value of the flipbook animation to display, value must must be given in the appropriate corresponding increment

Notes:

Only applicable with `run_type` set to `step`.

Example Usage:

```
anim_flipbook: run_type step
anim_flipbook: page 1.000000
anim_flipbook: page 2.000000
anim_flipbook: page 3.000000
```

See Also:

Associated anim_flipbook commands

anim_flipbook: regen_all_pages

Command:

```
anim_flipbook: regen_all_pages <toggle>
```

Purpose:

Toggle indicating whether to regenerate existing flipbook pages when loading a flipbook.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	(default) create all pages (existing pages are also overwritten)
		`OFF`	existing pages are not replaced by new pages having the same time value, and, if loading transient data, new pages can be interleaved according to their solution-time value

Notes:

No need to regenerate all pages if extending the number of time steps or filling-in skipped time steps.

Example Usage:

```
anim_flipbook: regen_all_pages OFF
```

See Also:

Associated anim_flipbook commands

Command:

```
anim_flipbook: run_type <option>
```

Purpose:

Specify how to play the flipbook animation.

Parameters:

Type	Parameter	Description	
string	option	auto	makes the flipbook play continuously
		step	makes the flipbook play page-by-page, according to the "anim_flipbook: page" command
		off	(default) deactivates the flipbook animation

Example Usage:

```
anim_flipbook: run_type auto

or

anim_flipbook: run_type step
anim_flipbook: page 1.000000
anim_flipbook: page 2.000000
anim_flipbook: page 3.000000
or
anim_flipbook: run_type off
```

See Also:

Associated anim_flipbook commands

anim_flipbook: save_images

Command:

```
anim_flipbook: save_images <filename_prefix>
```

Purpose:

Specify the file name prefix into which each flipbook animation page (or each image) will be recorded.

Parameters:

Type	Parameter	Description
string	`filename_prefix`	file name prefix of the recorded media

Notes:

Each page image will be saved into a file of the form "`fileprefix_ipage.ext`" where "`filename_prefix`" is the path and name of the file, "`ipage`" is the corresponding page number, and "`ext`" is the appropriate image format extension name.

Example Usage:

```
#
# Set the recording image format to sgi RGB
#
file: image_format sgi_rgb
#
# Record each page of the flipbook of main graphics window size
# into a file named "demo_ipage.rgb" where "ipage" is the page
# number of the animation.
#
anim_flipbook: save_win_type normal
anim_flipbook: save_images demo
```

See Also:

Associated anim_flipbook commands

Command:

```
anim_flipbook: save_multiple_images <toggle>
```

Purpose:

When recording an animation flipbook with a detached display this option will save a file for each display if on. If off will save a single file.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	record a file per detached display
		`OFF`	record a single file for detached display

Notes:

When saving a flat-wall detached display, the user can choose to write one large image for the wall or one image per ‘pipe’ from the dconfig file. Only used if the anim_flipbook: save_win_type is set to “detached_display”.

Example Usage:

```
#
# Set the recording image format to be envideo
#
file: image_format envideo
#
# Record flipbook images from the detached display
#
anim_flipbook: save_win_type detached_display
anim_flipbook: save_multiple_images ON
anim_flipbook: save_images /tmp/test_movie
```

See Also:

Associated anim_flipbook commands

anim_flipbook: save_win_size

Command:

```
anim_flipbook: save_win_size <x_size> <y_size>
```

Purpose:

Specify the x and y window dimensions of the flipbook animation images.

Parameters:

Type	Parameter	Description
int	x_size	x window size, or width of the window, in screen coordinates
	y_size	y window size, or height of the window, in screen coordinates

Notes:

Only applicable with command option "`anim_flipbook: save_win_type user_defined`".

Example Usage:

```
#
# Set the recording image format to sgi RGB
#
file: image_format sgi_rgb
#
# Record each page of the flipbook of main graphics window size
# into a file named "demo_ipage.rgb" where "ipage" is the page
# number of the animation.
#
anim_flipbook: save_win_type user_defined
anim_flipbook: save_win_size 320 240
anim_flipbook: save_images demo
```

See Also:

Associated anim_flipbook commands

Command:

```
anim_flipbook: save_win_type <size>
```

Purpose:

Specify the size of the window/image into which to play/record the flipbook animation.

Parameters:

Type	Parameter	Description	
string	`size`	`normal`	(default) corresponds to the current size of the main view graphics window
		`user_defined`	corresponds to the window size specified by the `"anim_flipbook: save_win_size"` command
		`detached_display`	use the detached display, as specified by the -dconfig command line argument

Example Usage:

```
#
# Set the recording image format to sgi RGB
#
file: image_format sgi_rgb
#
# Record each page of the flipbook of main graphics window size
# into a file named "demo_ipage.rgb" where "ipage" is the page
# number of the animation.
#
anim_flipbook: save_win_type user_defined
anim_flipbook: save_win_size 320 240
anim_flipbook: save_images demo
```

See Also:

Associated anim_flipbook commands

anim_flipbook: select_all_timesteps

Command:

```
anim_flipbook: select_all_timesteps
```

Purpose:

When loading a flipbook load all timesteps that exist.

Parameters:

None.

Notes:

This command is useful in batch scripts since you do not need to know how many timesteps exist. The command is not generated by interactive use of EnSight.

Example Usage:

```
anim_flipbook: select_all_timesteps
anim_flipbook: load
```

See Also:

Associated anim_flipbook commands

Command:

```
anim_flipbook: show_from_page <page_value>
```

Purpose:

Specify the starting page from which to run the flipbook animation.

Parameters:

Type	Parameter	Description
float	page_value	a page value which lies within the range of loaded pages

Notes:

This page value must not be greater than the "anim_flipbook: show_to_page" value.

Example Usage:

```
#
# Preliminary flipbook animation of 21 time steps
#
anim_flipbook: begin_time_step 30
anim_flipbook: end_time_step 50
anim_flipbook: specify_time_as step
anim_flipbook: load
anim_flipbook: run_type auto
#
# Reduce flipbook animation to 16 time steps
#
anim_flipbook: show_from_page 35
```

See Also:

Associated anim_flipbook commands

anim_flipbook: show_to_page

Command:

```
anim_flipbook: show_to_page <page_value>
```

Purpose:

Specify the ending page to which to run the flipbook animation.

Parameters:

Type	Parameter	Description
float	page_value	A page value which lies within the range of loaded pages

Notes:

This page value must not be less than the "anim_flipbook: show_from_page" value.

Example Usage:

```
#
# Preliminary flipbook animation of 21 time steps
#
anim_flipbook: begin_time_step 30
anim_flipbook: end_time_step 50
anim_flipbook: specify_time_as step
anim_flipbook: load
anim_flipbook: run_type auto
#
# Reduce flipbook animation to 11 time steps
#
anim_flipbook: show_from_page 35
anim_flipbook: show_to_page 45
```

See Also:

Associated anim_flipbook commands

Command:

```
anim_flipbook: specify_time_as <option>
```

Purpose:

Specify the paging units for the flipbook animation.

Parameters:

Type	Parameter	Description	
string	option	simulation	the flipbook pages will be loaded according to simulation time settings
		step	(default) the flipbook pages will be loaded according to time step settings

Notes:

Not applicable if not transient data.

The default is "step"

Example Usage:

```
#
# Change time sequence from time step to simulation time
#
solution_time: show_as time
#
# Set begin/end flipbook time info
#
anim_flipbook: specify_time_as simulation
anim_flipbook: begin_simtime 3.5400e+02
anim_flipbook: end_simtime 3.6900e+02
anim_flipbook: load
anim_flipbook: run_type auto
```

See Also:

Associated anim_flipbook commands

anim_flipbook: step_by

Command:

```
anim_flipbook: step_by <step_increment>
```

Purpose:

Specify the time step increment by which to run the `transient_data` flipbook animation.

Parameters:

Type	Parameter	Description
float	`step_increment`	increment value in time-step units that preferably lies within the range set by the `anim_flipbook: begin_time_step` and `anim_flipbook: end_time_step`

Notes:

Not applicable if not transient data. Only pages contained in the valid time-step range will be generated.

Example Usage:

```
#
# Note: transient dataset w/83 time steps...
#
# Specify a transient_data flipbook
#
anim_flipbook: load_type transient_data
#
# Specify page increment
#
anim_flipbook: step_by 1.0000e+01
#
# Set begin/end flipbook time info
#
anim_flipbook: begin_time_step 0
anim_flipbook: end_time_step 83
anim_flipbook: specify_time_as step
#
# Load and run flipbook pages {0,10,20,30,40,50,60,70,80}
#
anim_flipbook: load
anim_flipbook: run_type auto
```

See Also:

Associated anim_flipbook commands

Command:

```
anim_flipbook: step_simtime <time_value>
```

Purpose:

Specify the simulation-time increment by which to run the `transient_data` flipbook animation.

Parameters:

Type	Parameter	Description
float	`time_value`	increment value in simulation-time units that preferably lies within the range set by the `anim_flipbook: begin_simtime` and `anim_flipbook: end_simtime`

Notes:

Not applicable if not transient data. Only pages contained in the valid simulation-time range will be generated.

Example Usage:

```
#
# Specify time and flipbook sequencing as simulation time
#
solution_time: show_as time
anim_flipbook: specify_time_as simulation
#
# Specify time increment
#
anim_flipbook: step_simtime 2.0000e+00
#
# Load and run transient data flipbook
#
anim_flipbook: begin_simtime 3.5400e+02
anim_flipbook: end_simtime 3.6400e+02
#
# Load and run flipbook pages with t=354.0,356.0,358.0,360.0,362.0,364.0
#
anim_flipbook: load
anim_flipbook: run_type auto
```

See Also:

Associated anim_flipbook commands

anim_flipbook: time

Command:

```
anim_flipbook: time <time_value>
```

Purpose:

Specify which page of the flipbook animation to display.

Parameters:

Type	Parameter	Description
float	`time_value`	simulation-time value that corresponds to a loaded page

Notes:

Only applicable with `run_type` is set to `step`.

Example Usage:

```
#
# Manually display 3 pages of the animated flipbook sequences
# that correspond to the simulation time values of 1.1, 2.1, and 3.1.
#
anim_flipbook: run_type step
anim_flipbook: time 1.100000
anim_flipbook: time 1.200000
anim_flipbook: time 1.300000
```

See Also:

Associated anim_flipbook commands

Commands in this Section:

anim_keyframe

anim_keyframe: acceleration <keyframe> <toggle>
anim_keyframe: animate_transparency <toggle>
anim_keyframe: begin_simtime <time_value>
anim_keyframe: command <key_num> <cmd_num> <command>
anim_keyframe: begin_time_step <step_value>
anim_keyframe: create_keyframe
anim_keyframe: cycle <toggle>
anim_keyframe: delete_back_to <keyframe_num>
anim_keyframe: end_simtime <time_value>
anim_keyframe: end_time_step <step_value>
anim_keyframe: hold <keyframe> <num_frames>
anim_keyframe: keyframing <toggle>
anim_keyframe: restore <filename>
anim_keyframe: run
anim_keyframe: run_from <keyframe_num>
anim_keyframe: run_to <keyframe_num>
anim_keyframe: save <filename>
anim_keyframe: specify_time_as <method>
anim_keyframe: step_by <increment>
anim_keyframe: step_simtime <increment>
anim_keyframe: sub_frames <key1> <key2> <subframes>
anim_keyframe: timeline_arrive_type <action>
anim_keyframe: timeline_delete <timeline>
anim_keyframe: timeline_end_keyframe <keyframe_num>
anim_keyframe: timeline_end_step <time_step>
anim_keyframe: timeline_end_time <time_value>
anim_keyframe: timeline_end_type <time_option>
anim_keyframe: timeline_new
anim_keyframe: timeline_select <timeline>
anim_keyframe: timeline_specify_incr <toggle>
anim_keyframe: timeline_start_keyframe <keyframe_num>
anim_keyframe: timeline_start_step <time_step>
anim_keyframe: timeline_start_time <time_value>
anim_keyframe: timeline_start_type <time_option>
anim_keyframe: timeline_step_increment <delta_step>
anim_keyframe: timeline_time_increment <delta_time>
anim_keyframe: use_interactive_parts <toggle>
anim_keyframe: use_spline <toggle>
anim_keyframe: use_transient_data <toggle>

See Also:

User Manual - Section 7.15, Keyframe Animation
How To Create a Keyframe Animation
How To Animate Transient Data

anim_keyframe: acceleration

Command:

```
anim_keyframe: acceleration <keyframe> <toggle>
```

Purpose:

Toggle acceleration into/out of a keyframe.

Parameters:

Type	Parameter	Description	
int	`keyframe`	The keyframe number to apply the acceleration	
constant	`toggle`	`ON`	to apply acceleration
		`OFF`	for no acceleration

Example Usage:

```
anim_keyframe: keyframing ON
anim_keyframe: create_keyframe
view_transf: rotate 20 30 5
view_transf: translate 5 0 0
anim_keyframe: create_keyframe
#
# accelerate out of keyframe 1
#
anim_keyframe: acceleration 1 ON
anim_keyframe: use_spline ON
#
# deaccelerate into keyframe 2
#
anim_keyframe: acceleration 2 ON
anim_keyframe: run
```

See Also:

Associated anim_keyframe commands

Command:

```
anim_keyframe: animate_transparency <toggle>
```

Purpose:

Toggles use of animated transparency.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	turns on use of transparency changes during animation
		OFF	turns off use of transparency changes during animation

Notes:

This command can be turned on at any time during keyframe recording. If it is on, any transparency modifications made to parts during keyframe specification will be animated during playback. This can be used to create video effects such as fades.

Example Usage:

```
anim_keyframe: keyframing ON
anim_keyframe: create_keyframe
part: select_partname_begin
"(CASE:Case 1)Isosurface part"
part: select_partname_end
part: modify_begin
part: opaqueness 0.0
part: modify_end
anim_keyframe: create_keyframe
part: modify_begin
part: opaqueness 1.0
part: modify_end
anim_keyframe: create_keyframe
anim_keyframe: animate_transparency ON
anim_keyframe: run
```

See Also:

Associated anim_keyframe commands

anim_keyframe: begin/end_simtime

Command:

```
anim_keyframe: begin_simtime <time_value>
anim_keyframe: end_simtime <time_value>
```

Purpose:

Sets the starting simulation time value to use during transient keyframe animations.

Parameters:

Type	Parameter	Description
float	`time_value`	beginning/ending simulation time

Notes:

The beginning/ending time step must be a valid value from the simulation. An interpolated value cannot be used.

Example Usage:

```
anim_keyframe: keyframing ON
anim_keyframe: create_keyframe
view_transf: zoom 0.561365
anim_keyframe: create_keyframe
anim_keyframe: use_transient_data ON
anim_keyframe: specify_time_as simulation
anim_keyframe: begin_simtime 4.7000e-02
anim_keyframe: end_simtime 1.1750e-01
anim_keyframe: step_simtime 2.0000e-02
anim_keyframe: run
```

See Also:

Associated anim_keyframe commands

Command:

```
anim_keyframe: begin_time_step <step_value>
anim_keyframe: end_time_step <step_value>
```

Purpose:

Sets the starting/ending time step to use during transient keyframe animations.

Parameters:

Type	Parameter	Description
int	`step_value`	beginning/ending time step

Example Usage:

```
anim_keyframe: use_transient_data ON
anim_keyframe: step_by 2.0000e+00
anim_keyframe: cycle ON
anim_keyframe: begin_time_step 2
anim_keyframe: end_time_step 15
anim_keyframe: specify_time_as step
anim_keyframe: run
```

See Also:

Associated anim_keyframe commands

anim_keyframe: command key

Command:

```
anim_keyframe: command <key_num> <cmd_num> <command>
```

Purpose:

Specifies a command to execute for at a particular keyframe.

Parameters:

Type	Parameter	Description
int	key_num	specifies the keyframe
	cmd_num	specifies the command number
string	command	specifies the command to execute

Notes:

During animation playback, when keyframe 'key' is encountered, its commands, if any, will be executed. Up to five commands can be specified for a specific keyframe. If more than five commands are needed, consider using the "`play: file.cmd`" command to play an external command file. The command string format is the same for any command. Also, consider using the "`shell: external_app`" command to execute an external application during animation.

Example Usage:

```
anim_keyframe: keyframing ON
anim_keyframe: create_keyframe
view_transf: zoom 0.561365
anim_keyframe: create_keyframe
view_transf: zoom 0.561365
anim_keyframe: create_keyframe
anim_keyframe: command 1 1 view: full_screen on
anim_keyframe: command 1 2 view: hidden_surface on
anim_keyframe: command 2 1 play: /usr/tmp/ensight_cmds.cmd
anim_keyframe: run
```

See Also:

Associated anim_keyframe commands

Command:

```
anim_keyframe: create_keyframe
```

Purpose:

Records the current set of transformations/attributes as a keyframe.

Parameters:

none

Example Usage:

```
anim_keyframe: keyframing ON
anim_keyframe: create_keyframe
view_transf: zoom 0.561365
anim_keyframe: create_keyframe
anim_keyframe: sub_frames 1 2 30
anim_keyframe: run
```

See Also:

Associated anim_keyframe commands

anim_keyframe: cycle

Command:

```
anim_keyframe: cycle <toggle>
```

Purpose:

Toggles whether transient data will cycle during the keyframe animation.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	cycles (i.e., transient data plays in reverse when arrive at last time value) transient data during the animation
		`OFF`	turns cycling off (i.e., transient data starts playing from `begin_time_step` when arrive at last time value)

Example Usage:

```
anim_keyframe: use_transient_data ON
anim_keyframe: step_by 2.0000e+00
anim_keyframe: cycle ON
anim_keyframe: begin_time_step 2
anim_keyframe: end_time_step 15
anim_keyframe: specify_time_as step
anim_keyframe: run
```

See Also:

Associated anim_keyframe commands

Command:

```
anim_keyframe: delete_back_to <keyframe_num>
```

Purpose:

Deletes keyframes back to, but not including, the keyframe specified.

Parameters:

Type	Parameter	Description
int	keyframe_num	keyframe to delete back to

Example Usage:

```
anim_keyframe: keyframing ON
anim_keyframe: create_keyframe
view_transf: zoom 1.5
anim_keyframe: create_keyframe
view_transf: zoom 5.0
anim_keyframe: create_keyframe
anim_keyframe: delete_back_to 2
```

See Also:

Associated anim_keyframe commands

anim_keyframe: hold

Command:

```
anim_keyframe: hold <keyframe> <num_frames>
```

Purpose:

Sets the number of frames to display for a keyframe.

Parameters:

Type	Parameter	Description
int	keyframe	keyframe to control
int	num_frames	display this many frames

Example Usage:

```
anim_keyframe: keyframing ON
anim_keyframe: create_keyframe
view_transf: rotate -84.910713 -3.917774 0.000000
view_transf: function global
view_transf: rotate 13.660716 39.395405 0.000000
view_transf: function global
anim_keyframe: create_keyframe
anim_keyframe: hold 1 10
anim_keyframe: run
```

See Also:

Associated anim_keyframe commands

Command:

```
anim_keyframe: keyframing <toggle>
```

Purpose:

Toggles keyframe animation recording.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	turns on keyframe animation recording
		OFF	turns off recording and removes any keyframe information

Notes:

Toggling off keyframe animation will delete the previously saved keyframes.

Example Usage:

```
anim_keyframe: keyframing ON
anim_keyframe: create_keyframe
view_transf: rotate 28.489212 79.540230 0.000000
anim_keyframe: create_keyframe
anim_keyframe: run
anim_keyframe: keyframing OFF
```

See Also:

Associated anim_keyframe commands

anim_keyframe: restore/save

Command:

```
anim_keyframe: restore <filename>
anim_keyframe: save <filename>
```

Purpose:

Restores/saves keyframe animation data from/to a file.

Parameters:

Type	Parameter	Description
string	`filename`	file name to use

Example Usage:

```
anim_keyframe: save ./my_anim
anim_keyframe: restore ./my_anim
```

See Also:

Associated anim_keyframe commands

Command:

```
anim_keyframe: run
```

Purpose:

Plays the keyframe animation.

Parameters:

none

Example Usage:

```
anim_keyframe: keyframing ON
anim_keyframe: create_keyframe
view_transf: zoom 0.561365
anim_keyframe: create_keyframe
anim_keyframe: sub_frames 1 2 30
anim_keyframe: run
```

See Also:

Associated anim_keyframe commands

anim_keyframe: run_from/to

Command:

```
anim_keyframe: run_from <keyframe_num>
anim_keyframe: run_to <keyframe_num>
```

Purpose:

Sets the first/last keyframe to use for playback.

Parameters:

Type	Parameter	Description
int	`keyframe_num`	beginning/ending keyframe

Notes:

When creating an animation with many keyframes, it is often helpful to play back a smaller range of keyframes instead of the entire animation. The value must be between 1 and the number of keyframes. The "`run_from`" frame must be less than or equal to the "`run_to'` frame.

Example Usage:

```
anim_keyframe: run_from 2
anim_keyframe: run_to 3
anim_keyframe: run
```

See Also:

Associated anim_keyframe commands

Command:

```
anim_keyframe: specify_time_as <method>
```

Purpose:

Specifies whether time values during transient data animations use simulation time or time step number.

Parameters:

Type	Parameter	Description	
constant	method	simulation	times are simulation values
		step	times are step values

Example Usage:

```
anim_keyframe: use_transient_data ON
anim_keyframe: specify_time_as step
anim_keyframe: step_by 2.0000e+00
anim_keyframe: cycle ON
anim_keyframe: begin_time_step 2
anim_keyframe: end_time_step 15
anim_keyframe: run
```

See Also:

Associated anim_keyframe commands

anim_keyframe: step_by

Command:

```
anim_keyframe: step_by <increment>
```

Purpose:

Specifies the time increment for transient data during keyframe animations.

Parameters:

Type	Parameter	Description
float	`increment`	time step increment

Notes:

For each frame of the animation, the time step will be incremented by this value. If needed and possible, interpolation will be used.

Example Usage:

```
anim_keyframe: use_transient_data ON
anim_keyframe: specify_time_as step
anim_keyframe: step_by 2.0000e+00
anim_keyframe: begin_time_step 2
anim_keyframe: end_time_step 15
anim_keyframe: run
```

See Also:

Associated anim_keyframe commands

Command:

```
anim_keyframe: step_simtime <increment>
```

Purpose:

Sets the simulation time increment to use during transient keyframe animations.

Parameters:

Type	Parameter	Description
float	`increment`	simulation time increment

Notes:

For each frame in the animation, simulation time will increment by this value. If needed and possible, interpolation will be used.

Example Usage:

```
anim_keyframe: keyframing ON
anim_keyframe: create_keyframe
view_transf: zoom 0.561365
anim_keyframe: create_keyframe
anim_keyframe: use_transient_data ON
anim_keyframe: cycle ON
anim_keyframe: specify_time_as simulation
anim_keyframe: begin_simtime 4.7000e-02
anim_keyframe: end_simtime 1.1750e-01
anim_keyframe: step_simtime 2.0000e-02
anim_keyframe: run
```

See Also:

Associated anim_keyframe commands

anim_keyframe: sub_frames

Command:

```
anim_keyframe: sub_frames <key1> <key2> <subframes>
```

Purpose:

Sets the number of in-between frames between keyframes key1 and key2.

Parameters:

Type	Parameter	Description
int	`key1`	specifies the starting keyframe
int	`key2`	specifies the ending keyframe (must be `key1` + 1)
int	`subframes`	specifies the number of in-between frames between key1 and key2

Notes:

key1 and key2 must be consecutively keyframes.

Example Usage:

```
anim_keyframe: keyframing ON
anim_keyframe: create_keyframe
view_transf: zoom 0.561365
anim_keyframe: create_keyframe
anim_keyframe: sub_frames 1 2 30
anim_keyframe: run
```

See Also:

Associated anim_keyframe commands

Command:

```
anim_keyframe: timeline_new
```

Purpose:

Create a new timeline.

Parameters:

none

Notes:

Is only possible if the previous (if exists) timeline does not completely span the available key-frames

Example Usage:

```
#
# turn on keyframing and create 4 keyframes
#
anim_keyframe: keyframing ON
anim_keyframe: create_keyframe
anim_keyframe: create_keyframe
anim_keyframe: create_keyframe
anim_keyframe: create_keyframe
#
# transient data (and timelines) will be used during the animation
#
anim_keyframe: use_transient_data ON
#
#
anim_keyframe: timeline_select 1
#
# the first timeline goes from keyframe 2 to 3 (i.e, no transient
# data is used for keyframe 1 to 2)
#
anim_keyframe: timeline_start_keyframe 2
anim_keyframe: timeline_end_keyframe 3
#
# At keyframe 2 the "Begin" (as defined in the solution time dialog)
# time is used.  At keyframe 3 the "End" time is used.
#
anim_keyframe: timeline_start_type use_begin
anim_keyframe: timeline_end_type use_end
#
# Create a new timeline and select it for editing
#
anim_keyframe: timeline_new
anim_keyframe: timeline_select 2
#
# This timeline goes from keyframe 3 to 4
#
anim_keyframe: timeline_start_keyframe 3
anim_keyframe: timeline_end_keyframe 4
#
# At keyframe 3 the current time will be used (which will
# be the "End" time since the first timeline ends with this time
# We do not specify the time at keyframe 4 but rather increment time
# by 1.4 unit for each frame.  If we arrive at the begin/end time we will
```

```
# "swing"
#
anim_keyframe: timeline_start_type use_current
anim_keyframe: timeline_specify_incr ON
anim_keyframe: timeline_time_increment 1.4
anim_keyframe: timeline_arrive_type swing
```

See Also:

Associated anim_keyframe commands

Command:

```
anim_keyframe: timeline_select <timeline>
```

Purpose:

Selects an existing timeline for editing.

Parameters:

Type	Parameter	Description
int	timeline	existing timeline to edit

Example Usage:

```
#
# turn on keyframing and create 4 keyframes
#
anim_keyframe: keyframing ON
anim_keyframe: create_keyframe
anim_keyframe: create_keyframe
anim_keyframe: create_keyframe
anim_keyframe: create_keyframe
#
# transient data (and timelines) will be used during the animation
#
anim_keyframe: use_transient_data ON
#
#
anim_keyframe: timeline_select 1
#
# the first timeline goes from keyframe 2 to 3 (i.e, no transient
# data is used for keyframe 1 to 2)
#
anim_keyframe: timeline_start_keyframe 2
anim_keyframe: timeline_end_keyframe 3
#
# At keyframe 2 the "Begin" (as defined in the solution time dialog)
# time is used.  At keyframe 3 the "End" time is used.
#
anim_keyframe: timeline_start_type use_begin
anim_keyframe: timeline_end_type use_end
#
# Create a new timeline and select it for editing
#
anim_keyframe: timeline_new
anim_keyframe: timeline_select 2
#
# This timeline goes from keyframe 3 to 4
#
anim_keyframe: timeline_start_keyframe 3
anim_keyframe: timeline_end_keyframe 4
#
# At keyframe 3 the current time will be used (which will
# be the "End" time since the first timeline ends with this time
# We do not specify the time at keyframe 4 but rather increment time
# by 1. unit for each frame.  If we arrive at the begin/end time we will
# "swing"
#
anim_keyframe: timeline_start_type use_current
```

```
anim_keyframe: timeline_specify_incr ON
anim_keyframe: timeline_step_increment 1.
anim_keyframe: timeline_arrive_type swing
```

See Also:

Associated anim_keyframe commands

Command:

```
anim_keyframe: timeline_specify_incr <toggle>
```

Purpose:

When ON specifies that a time increment is to be used for the selected timeline.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	to specify a time increment
		OFF	to not specify a time increment

Example Usage:

```
#
# turn on keyframing and create 4 keyframes
#
anim_keyframe: keyframing ON
anim_keyframe: create_keyframe
anim_keyframe: create_keyframe
anim_keyframe: create_keyframe
anim_keyframe: create_keyframe
#
# transient data (and timelines) will be used during the animation
#
anim_keyframe: use_transient_data ON
#
#
anim_keyframe: timeline_select 1
#
# the first timeline goes from keyframe 2 to 3 (i.e, no transient
# data is used for keyframe 1 to 2)
#
anim_keyframe: timeline_start_keyframe 2
anim_keyframe: timeline_end_keyframe 3
#
# At keyframe 2 the "Begin" (as defined in the solution time dialog)
# time is used.  At keyframe 3 the "End" time is used.
#
anim_keyframe: timeline_start_type use_begin
anim_keyframe: timeline_end_type use_end
#
# Create a new timeline and select it for editing
#
anim_keyframe: timeline_new
anim_keyframe: timeline_select 2
#
# This timeline goes from keyframe 3 to 4
#
anim_keyframe: timeline_start_keyframe 3
anim_keyframe: timeline_end_keyframe 4
#
# At keyframe 3 the current time will be used (which will
# be the "End" time since the first timeline ends with this time
# We do not specify the time at keyframe 4 but rather increment time
# by 1. unit for each frame.  If we arrive at the begin/end time we will
# "swing"
#
```

```
anim_keyframe: timeline_start_type use_current
anim_keyframe: timeline_specify_incr ON
anim_keyframe: timeline_step_increment 1.
anim_keyframe: timeline_arrive_type swing
```

See Also:

Associated anim_keyframe commands

Command:

```
anim_keyframe: timeline_arrive_type <action>
```

Purpose:

Specifies the action to be taken when the min/max time is encountered.

Parameters:

Type	Parameter	Description	
constant	action	swing	play time in reverse
		loop	set time to the begin/end

Notes:

Only used if "anim_keyframe: timeline_specify_incr" is ON

Example Usage:

```
#
# turn on keyframing and create 4 keyframes
#
anim_keyframe: keyframing ON
anim_keyframe: create_keyframe
anim_keyframe: create_keyframe
anim_keyframe: create_keyframe
anim_keyframe: create_keyframe
#
# transient data (and timelines) will be used during the animation
#
anim_keyframe: use_transient_data ON
#
#
anim_keyframe: timeline_select 1
#
# the first timeline goes from keyframe 2 to 3 (i.e, no transient
# data is used for keyframe 1 to 2)
#
anim_keyframe: timeline_start_keyframe 2
anim_keyframe: timeline_end_keyframe 3
#
# At keyframe 2 the "Begin" (as defined in the solution time dialog)
# time is used.  At keyframe 3 the "End" time is used.
#
anim_keyframe: timeline_start_type use_begin
anim_keyframe: timeline_end_type use_end
#
# Create a new timeline and select it for editing
#
anim_keyframe: timeline_new
anim_keyframe: timeline_select 2
#
# This timeline goes from keyframe 3 to 4
#
anim_keyframe: timeline_start_keyframe 3
anim_keyframe: timeline_end_keyframe 4
#
# At keyframe 3 the current time will be used (which will
# be the "End" time since the first timeline ends with this time
```

```
# We do not specify the time at keyframe 4 but rather increment time
# by 1.4 unit for each frame.  If we arrive at the begin/end time we will
# "swing"
#
anim_keyframe: timeline_start_type use_current
anim_keyframe: timeline_specify_incr ON
anim_keyframe: timeline_time_increment 1.4
anim_keyframe: timeline_arrive_type swing
```

See Also:

Associated anim_keyframe commands

Command:

```
anim_keyframe: timeline_delete <timeline>
```

Purpose:

Delete a timeline.

Parameters:

Type	Parameter	Description
int	timeline	timeline number to delete

Example Usage:

```
#
# turn on keyframing and create 4 keyframes
#
anim_keyframe: keyframing ON
anim_keyframe: create_keyframe
anim_keyframe: create_keyframe
anim_keyframe: create_keyframe
anim_keyframe: create_keyframe
#
# transient data (and timelines) will be used during the animation
#
anim_keyframe: use_transient_data ON
#
#
anim_keyframe: timeline_new
anim_keyframe: timeline_select 2
#
# This timeline goes from keyframe 3 to 4
#
anim_keyframe: timeline_start_keyframe 3
anim_keyframe: timeline_end_keyframe 4
#
# delete the first timeline
#
anim_keyframe: timeline_delete 1
```

See Also:

Associated anim_keyframe commands

anim_keyframe: timeline_start/end_keyframe

Command:

```
anim_keyframe: timeline_start_keyframe <keyframe_num>
anim_keyframe: timeline_end_keyframe <keyframe_num>
```

Purpose:

Sets the begin/end keyframes for the timeline.

Parameters:

Type	Parameter	Description
int	keyframe_num	beginning/ending keyframe for the timeline

Example Usage:

```
#
# turn on keyframing and create 4 keyframes
#
anim_keyframe: keyframing ON
anim_keyframe: create_keyframe
anim_keyframe: create_keyframe
anim_keyframe: create_keyframe
anim_keyframe: create_keyframe
#
# transient data (and timelines) will be used during the animation
#
anim_keyframe: use_transient_data ON
#
#
anim_keyframe: timeline_select 1
#
# the first timeline goes from keyframe 2 to 3 (i.e, no transient
# data is used for keyframe 1 to 2)
#
anim_keyframe: timeline_start_keyframe 2
anim_keyframe: timeline_end_keyframe 3
#
# At keyframe 2 the "Begin" (as defined in the solution time dialog)
# time is used.  At keyframe 3 the "End" time is used.
#
anim_keyframe: timeline_start_type use_begin
anim_keyframe: timeline_end_type use_end
#
# Create a new timeline and select it for editing
#
anim_keyframe: timeline_new
anim_keyframe: timeline_select 2
#
# This timeline goes from keyframe 3 to 4
#
anim_keyframe: timeline_start_keyframe 3
anim_keyframe: timeline_end_keyframe 4
#
# At keyframe 3 the current time will be used (which will
# be the "End" time since the first timeline ends with this time
# We do not specify the time at keyframe 4 but rather increment time
# by 1. unit for each frame.  If we arrive at the begin/end time we will
# "swing"
#
```

```
anim_keyframe: timeline_start_type use_current
anim_keyframe: timeline_specify_incr ON
anim_keyframe: timeline_step_increment 1.
anim_keyframe: timeline_arrive_type swing
```

See Also:

Associated anim_keyframe commands

anim_keyframe: timeline_start/end_step

Command:

```
anim_keyframe: timeline_start_step <time_step>
anim_keyframe: timeline_end_step <time_step>
```

Purpose:

Sets the start/end timeline step value if being specified.

Parameters:

Type	Parameter	Description
float	`time_value`	The time value (step) for the start/end of the timeline

Notes:

Only used if "anim_keyframe: timeline_start/end_type specify" is also set.

Example Usage:

```
#
# turn on keyframing and create 4 keyframes
#
anim_keyframe: keyframing ON
anim_keyframe: create_keyframe
anim_keyframe: create_keyframe
anim_keyframe: create_keyframe
anim_keyframe: create_keyframe
#
# transient data (and timelines) will be used during the animation
#
anim_keyframe: use_transient_data ON
#
#
anim_keyframe: timeline_select 1
#
# the first timeline goes from keyframe 2 to 3 (i.e, no transient
# data is used for keyframe 1 to 2)
#
anim_keyframe: timeline_start_keyframe 2
anim_keyframe: timeline_end_keyframe 3
#
# At keyframe 2 the "Begin" (as defined in the solution time dialog)
# time is used.  At keyframe 3 the "End" time is used.
#
anim_keyframe: timeline_start_type use_begin
anim_keyframe: timeline_end_type use_end
#
# Create a new timeline and select it for editing
#
anim_keyframe: timeline_new
anim_keyframe: timeline_select 2
#
# This timeline goes from keyframe 3 to 4
#
anim_keyframe: timeline_start_keyframe 3
anim_keyframe: timeline_end_keyframe 4
#
# At keyframe 3 the current time will be used (which will
# be the "End" time since the first timeline ends with this time
```

```
# At keyframe 4 we set time step to 4.
#
anim_keyframe: timeline_start_type use_current
anim_keyframe: timeline_end_type specify
anim_keyframe: timeline_end_time 4.
```

See Also:

Associated anim_keyframe commands

anim_keyframe: timeline_start/end_time

Command:

```
anim_keyframe: timeline_start_time <time_value>
anim_keyframe: timeline_end_time <time_value>
```

Purpose:

Sets the start/end timeline value if being specified.

Parameters:

Type	Parameter	Description
float	`time_value`	The time value (simulation) for the start/end of the timeline

Notes:

Only used if "anim_keyframe: timeline_start/end_type specify" is also set.

Example Usage:

```
#
# turn on keyframing and create 4 keyframes
#
anim_keyframe: keyframing ON
anim_keyframe: create_keyframe
anim_keyframe: create_keyframe
anim_keyframe: create_keyframe
anim_keyframe: create_keyframe
#
# transient data (and timelines) will be used during the animation
#
anim_keyframe: use_transient_data ON
#
#
anim_keyframe: timeline_select 1
#
# the first timeline goes from keyframe 2 to 3 (i.e, no transient
# data is used for keyframe 1 to 2)
#
anim_keyframe: timeline_start_keyframe 2
anim_keyframe: timeline_end_keyframe 3
#
# At keyframe 2 the "Begin" (as defined in the solution time dialog)
# time is used.  At keyframe 3 the "End" time is used.
#
anim_keyframe: timeline_start_type use_begin
anim_keyframe: timeline_end_type use_end
#
# Create a new timeline and select it for editing
#
anim_keyframe: timeline_new
anim_keyframe: timeline_select 2
#
# This timeline goes from keyframe 3 to 4
#
anim_keyframe: timeline_start_keyframe 3
anim_keyframe: timeline_end_keyframe 4
#
# At keyframe 3 the current time will be used (which will
# be the "End" time since the first timeline ends with this time
```

```
# At keyframe 4 we set time to be 40.
#
anim_keyframe: timeline_start_type use_current
anim_keyframe: timeline_end_type specify
anim_keyframe: timeline_end_time 40.
```

See Also:

Associated anim_keyframe commands

anim_keyframe: timeline_start/end_type

Command:

```
anim_keyframe: timeline_start_type <time_option>
anim_keyframe: timeline_end_type <time_option>
```

Purpose:

Sets the begin/end time for the timeline.

Parameters:

Type	Parameter	Description	
constant	time_option	use_begin	Time is set to the "Beg" time as defined in the solution time dialog
		use_end	Time is set to the "End" time as defined in the solution time dialog
		use_current	Time is set to the current time value
		specify	Time is set to the time value specified by "timeline_end_time", "timeline_end_step", "timeline_start_time" or "timeline"end_time"

Example Usage:

```
#
# turn on keyframing and create 4 keyframes
#
anim_keyframe: keyframing ON
anim_keyframe: create_keyframe
anim_keyframe: create_keyframe
anim_keyframe: create_keyframe
anim_keyframe: create_keyframe
#
# transient data (and timelines) will be used during the animation
#
anim_keyframe: use_transient_data ON
#
#
anim_keyframe: timeline_select 1
#
# the first timeline goes from keyframe 2 to 3 (i.e, no transient
# data is used for keyframe 1 to 2)
#
anim_keyframe: timeline_start_keyframe 2
anim_keyframe: timeline_end_keyframe 3
#
# At keyframe 2 the "Begin" (as defined in the solution time dialog)
# time is used.  At keyframe 3 the "End" time is used.
#
anim_keyframe: timeline_start_type use_begin
anim_keyframe: timeline_end_type use_end
#
# Create a new timeline and select it for editing
#
anim_keyframe: timeline_new
anim_keyframe: timeline_select 2
#
# This timeline goes from keyframe 3 to 4
#
```

```
anim_keyframe: timeline_start_keyframe 3
anim_keyframe: timeline_end_keyframe 4
#
# At keyframe 3 the current time will be used (which will
# be the "End" time since the first timeline ends with this time
# We do not specify the time at keyframe 4 but rather increment time
# by 1. unit for each frame.  If we arrive at the begin/end time we will
# "swing"
#
anim_keyframe: timeline_start_type use_current
anim_keyframe: timeline_specify_incr ON
anim_keyframe: timeline_step_increment 1.
anim_keyframe: timeline_arrive_type swing
```

See Also:

Associated anim_keyframe commands

anim_keyframe: timeline_step_increment

Command:

```
anim_keyframe: timeline_step_increment <delta_step>
```

Purpose:

Specifies the time step increment when timeline_specify_incr is ON.

Parameters:

Type	Parameter	Description
float	`delta_step`	The time step increment between frames during the timeline

Notes:

Only used if "anim_keyframe: timeline_specify_incr" is ON

See "anim_keyframe: timeline_time_increment" to specify the time increment in time units instead of step units.

Example Usage:

```
#
# turn on keyframing and create 4 keyframes
#
anim_keyframe: keyframing ON
anim_keyframe: create_keyframe
anim_keyframe: create_keyframe
anim_keyframe: create_keyframe
anim_keyframe: create_keyframe
#
# transient data (and timelines) will be used during the animation
#
anim_keyframe: use_transient_data ON
#
#
anim_keyframe: timeline_select 1
#
# the first timeline goes from keyframe 2 to 3 (i.e, no transient
# data is used for keyframe 1 to 2)
#
anim_keyframe: timeline_start_keyframe 2
anim_keyframe: timeline_end_keyframe 3
#
# At keyframe 2 the "Begin" (as defined in the solution time dialog)
# time is used.  At keyframe 3 the "End" time is used.
#
anim_keyframe: timeline_start_type use_begin
anim_keyframe: timeline_end_type use_end
#
# Create a new timeline and select it for editing
#
anim_keyframe: timeline_new
anim_keyframe: timeline_select 2
#
# This timeline goes from keyframe 3 to 4
#
anim_keyframe: timeline_start_keyframe 3
anim_keyframe: timeline_end_keyframe 4
#
```

```
# At keyframe 3 the current time will be used (which will
# be the "End" time since the first timeline ends with this time
# We do not specify the time at keyframe 4 but rather increment time
# by 1. unit for each frame.  If we arrive at the begin/end time we will
# "swing"
#
anim_keyframe: timeline_start_type use_current
anim_keyframe: timeline_specify_incr ON
anim_keyframe: timeline_step_increment 1.
anim_keyframe: timeline_arrive_type swing
```

See Also:

Associated anim_keyframe commands

anim_keyframe: timeline_time_increment

Command:

```
anim_keyframe: timeline_time_increment <delta_time>
```

Purpose:

Specifies the time increment when timeline_specify_incr is ON.

Parameters:

Type	Parameter	Description
float	`delta_time`	The time increment between frames during the timeline

Notes:

Only used if "anim_keyframe: timeline_specify_incr" is ON

See "anim_keyframe: timeline_time_increment" to specify the time increment in step units instead of time units.

Example Usage:

```
#
# turn on keyframing and create 4 keyframes
#
anim_keyframe: keyframing ON
anim_keyframe: create_keyframe
anim_keyframe: create_keyframe
anim_keyframe: create_keyframe
anim_keyframe: create_keyframe
#
# transient data (and timelines) will be used during the animation
#
anim_keyframe: use_transient_data ON
#
#
anim_keyframe: timeline_select 1
#
# the first timeline goes from keyframe 2 to 3 (i.e, no transient
# data is used for keyframe 1 to 2)
#
anim_keyframe: timeline_start_keyframe 2
anim_keyframe: timeline_end_keyframe 3
#
# At keyframe 2 the "Begin" (as defined in the solution time dialog)
# time is used.  At keyframe 3 the "End" time is used.
#
anim_keyframe: timeline_start_type use_begin
anim_keyframe: timeline_end_type use_end
#
# Create a new timeline and select it for editing
#
anim_keyframe: timeline_new
anim_keyframe: timeline_select 2
#
# This timeline goes from keyframe 3 to 4
#
anim_keyframe: timeline_start_keyframe 3
anim_keyframe: timeline_end_keyframe 4
#
```

```
# At keyframe 3 the current time will be used (which will
# be the "End" time since the first timeline ends with this time
# We do not specify the time at keyframe 4 but rather increment time
# by 1.4 unit for each frame.  If we arrive at the begin/end time we will
# "swing"
#
anim_keyframe: timeline_start_type use_current
anim_keyframe: timeline_specify_incr ON
anim_keyframe: timeline_time_increment 1.4
anim_keyframe: timeline_arrive_type swing
```

See Also:

Associated anim_keyframe commands

anim_keyframe: use_interactive_parts

Command:

```
anim_keyframe: use_interactive_parts <toggle>
```

Purpose:

Toggles use of animated isosurfaces or clip planes.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	turns on use of interactive part modifications during animation
		`OFF`	turns off use of interactive part modifications during animation

Notes:

This command can be turned on at any time during keyframe recording. If it is on, any isosurfaces or clips that were interactively changed during recording will be animated.

Example Usage:

```
anim_keyframe: keyframing ON
anim_keyframe: create_keyframe
part: select_partname_begin
"(CASE:Case 1)Isosurface part"
part: select_partname_end
part: modify_begin
isos: value 9.9000e-01
part: modify_end
anim_keyframe: create_keyframe
anim_keyframe: use_interactive_parts ON
anim_keyframe: run
```

See Also:

Associated anim_keyframe commands

Command:

```
anim_keyframe: use_spline <toggle>
```

Purpose:

Enable spline interpolation for keyframe animation translations and look at/from changes.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	turns on spline interpolation
		`OFF`	turns off spline interpolation

Notes:

You must have more than 2 keyframes for spline interpolation.

Example Usage:

```
anim_keyframe: keyframing ON
anim_keyframe: create_keyframe
view_transf: translate 5 0 0
anim_keyframe: create_keyframe
view_transf: translate 0 5 0
anim_keyframe: create_keyframe
anim_keyframe: use_spline ON
anim_keyframe: acceleration 3 ON
anim_keyframe: run
```

See Also:

Associated anim_keyframe commands

anim_keyframe: use_transient_data

Command:

```
anim_keyframe: use_transient_data <toggle>
```

Purpose:

Toggles whether transient data will be used during the keyframe animation.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	turns on use of transient data
		`OFF`	turns off use of transient data

Notes:

When using transient data, each frame (keyframes and in-betweens) will use the next transient time value.

Example Usage:

```
anim_keyframe: use_transient_data ON
anim_keyframe: specify_time_as step
anim_keyframe: step_by 2.0000e+00
anim_keyframe: begin_time_step 2
anim_keyframe: end_time_step 15
anim_keyframe: run
```

See Also:

Associated anim_keyframe commands

Commands in this Section:

anim_quick

anim_quick: accelerate_first <toggle>
anim_quick: accelerate_last <toggle>
anim_quick: create_keyframes
anim_quick: explode_direction <direction>
anim_quick: explode_distance <distance>
anim_quick: explode_origin <x_ori> <y_ori> <z_ori>
anim_quick: explode_view <toggle>
anim_quick: fly_around <toggle>
anim_quick: fly_direction <direction>
anim_quick: fly_revolutions <many_revs>
anim_quick: rotate_direction <axis> <direction>
anim_quick: rotate_objects <toggle>
anim_quick: rotate_revolutions <many_revs>
anim_quick: total_frames <num_frames>

See Also:

User Manual - Section 7.15, Keyframe Animation
How To Create a Keyframe Animation

anim_quick: accelerate_first / last

Command:

```
anim_quick: accelerate_first <toggle>
anim_quick: accelerate_last <toggle>
```

Purpose:

Controls the acceleration for the first/last keyframes created for quick animations.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	accelerate at the first/last keyframe
		`OFF`	do not accelerate at the first/last keyframe

Example Usage:

```
#
# turn on keyframe animation
#
anim_keyframe: keyframing ON
#
# move the camera to the right one half revolution
#
anim_quick: fly_around ON
anim_quick: fly_direction right
anim_quick: fly_revolutions 0.5
#
# accelerate at the first keyframe but not at the last
#
anim_quick: accelerate_first ON
anim_quick: accelerate_last OFF
#
# create a total of 125 frames
#
anim_quick: total_frames 125
anim_quick: create_keyframes
```

See Also:

Associated anim_quick commands

Command:

```
anim_quick: create_keyframes
```

Purpose:

Creates the keyframes for the quick_animation setup.

Parameters:

none

Example Usage:

```
#
# turn on keyframe animation
#
anim_keyframe: keyframing ON
#
# create an exploded view about 2,0,0 in the x direction.
# The exploded distance is set to 200 units
#
anim_quick: explode_view ON
anim_quick: explode_origin 2 0 0
anim_quick: explode_direction x
anim_quick: explode_distance 200
#
# while rotating the scene about the x axis 2 revolutions positive direc-
tion
#
anim_quick: rotate_objects ON
anim_quick: rotate_direction x positively
anim_quick: rotate_revolutions x 2.000000
#
# accelerate at the first keyframe but not at the last
#
anim_quick: accelerate_first ON
anim_quick: accelerate_last OFF
#
# create a total of 125 frames
#
anim_quick: total_frames 125
anim_quick: create_keyframes
```

See Also:

Associated anim_quick commands

anim_quick: explode_direction

Command:

```
anim_quick: explode_direction <direction>
```

Purpose:

Sets up an exploded view quick animation.

Parameters:

Type	Parameter	Description	
constant	direction	x	exploded transform will be in global x direction
		y	exploded transform will be in global y direction
		z	exploded transform will be in global z direction
		xyz	exploded transform will be in x, y, or z direction depending on the part centroid position relative to the explode_origin.
		radial	explode along the vector from the explode origin through the part centroid.

Example Usage:

```
#
# turn on keyframe animation
#
anim_keyframe: keyframing ON
#
# create an exploded view about 2,0,0 in the x direction.
# The exploded distance is set to 200 units
#
anim_quick: explode_view ON
anim_quick: explode_origin 2 0 0
anim_quick: explode_direction x
anim_quick: explode_distance 200
#
# while rotating the scene about the x axis 2 revolutions positive direc-
tion
#
anim_quick: rotate_objects ON
anim_quick: rotate_direction x positively
anim_quick: rotate_revolutions x 2.000000
#
# accelerate at the first keyframe but not at the last
#
anim_quick: accelerate_first ON
anim_quick: accelerate_last OFF
#
# create a total of 125 frames
#
anim_quick: total_frames 125
anim_quick: create_keyframes
```

See Also:

Associated anim_quick commands

Command:

```
anim_quick: explode_distance <distance>
```

Purpose:

Sets the exploded view distance.

Parameters:

Type	Parameter	Description
float	distance	the maximum distance a part will be moved

Example Usage:

```
#
# turn on keyframe animation
#
anim_keyframe: keyframing ON
#
# create an exploded view about 2,0,0 in the x direction.
# The exploded distance is set to 200 units
#
anim_quick: explode_view ON
anim_quick: explode_origin 2 0 0
anim_quick: explode_direction x
anim_quick: explode_distance 200
#
# while rotating the scene about the x axis 2 revolutions positive direc-
tion
#
anim_quick: rotate_objects ON
anim_quick: rotate_direction x positively
anim_quick: rotate_revolutions x 2.000000
#
# accelerate at the first keyframe but not at the last
#
anim_quick: accelerate_first ON
anim_quick: accelerate_last OFF
#
# create a total of 125 frames
#
anim_quick: total_frames 125
anim_quick: create_keyframes
```

See Also:

Associated anim_quick commands

anim_quick: explode_origin

Command:

```
anim_quick: explode_origin <x_ori> <y_ori> <z_ori>
```

Purpose:

Sets the exploded view origin.

Parameters:

Type	Parameter	Description
float	x_ori	x coordinate of the explode origin
	y_ori	y coordinate of the explode origin
	z_ori	z coordinate of the explode origin

Example Usage:

```
#
# turn on keyframe animation
#
anim_keyframe: keyframing ON
#
# create an exploded view about 2,0,0 in the x direction.
# The exploded distance is set to 200 units
#
anim_quick: explode_view ON
anim_quick: explode_origin 2 0 0
anim_quick: explode_direction x
anim_quick: explode_distance 200
#
# while rotating the scene about the x axis 2 revolutions positive direc-
tion
#
anim_quick: rotate_objects ON
anim_quick: rotate_direction x positively
anim_quick: rotate_revolutions x 2.000000
#
# accelerate at the first keyframe but not at the last
#
anim_quick: accelerate_first ON
anim_quick: accelerate_last OFF
#
# create a total of 125 frames
#
anim_quick: total_frames 125
anim_quick: create_keyframes
```

See Also:

Associated anim_quick commands

Command:

```
anim_quick: explode_view <toggle>
```

Purpose:

Sets up an exploded view quick animation.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	add exploded view animation
		OFF	do not add exploded view animation

Notes:

The exploded view is not added until a “anim_quick: create_keyframes” command.

The exploded view can be used in combination with “rotate_objects” and “explode_view”

Example Usage:

```
#
# turn on keyframe animation
#
anim_keyframe: keyframing ON
#
# create an exploded view about 2,0,0 in the x direction.
# The exploded distance is set to 200 units
#
anim_quick: explode_view ON
anim_quick: explode_origin 2 0 0
anim_quick: explode_direction x
anim_quick: explode_distance 200
#
# while rotating the scene about the x axis 2 revolutions positive direc-
tion
#
anim_quick: rotate_objects ON
anim_quick: rotate_direction x positively
anim_quick: rotate_revolutions x 2.000000
#
# accelerate at the first keyframe but not at the last
#
anim_quick: accelerate_first ON
anim_quick: accelerate_last OFF
#
# create a total of 125 frames
#
anim_quick: total_frames 125
anim_quick: create_keyframes
```

See Also:

Associated anim_quick commands

anim_quick: fly_around

Command:

```
anim_quick: fly_around <toggle>
```

Purpose:

Sets up a fly-around quick animation.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	add fly-around animation
		`OFF`	do not add fly-around animation

Notes:

The fly-around is not added until a “anim_quick: create_keyframes” command.

The fly-around can be used in combination with “rotate_objects” and “explode_view”

Example Usage:

```
#
# turn on keyframe animation
#
anim_keyframe: keyframing ON
#
# move the camera to the right one half revolution
#
anim_quick: fly_around ON
anim_quick: fly_direction right
anim_quick: fly_revolutions 0.5
#
# accelerate at the first keyframe but not at the last
#
anim_quick: accelerate_first ON
anim_quick: accelerate_last OFF
#
# create a total of 125 frames
#
anim_quick: total_frames 125
anim_quick: create_keyframes
```

See Also:

Associated anim_quick commands

Command:

```
anim_quick: fly_direction <direction>
```

Purpose:

Sets up the direction for a fly-around animation.

Parameters:

Type	Parameter	Description	
constant	direction	right	move look-from position to the right
		left	move look-from position to the left

Notes:

A fly-around moves the look-from position in an arc.

Example Usage:

```
#
# turn on keyframe animation
#
anim_keyframe: keyframing ON
#
# move the camera to the right one half revolution
#
anim_quick: fly_around ON
anim_quick: fly_direction right
anim_quick: fly_revolutions 0.5
#
# accelerate at the first keyframe but not at the last
#
anim_quick: accelerate_first ON
anim_quick: accelerate_last OFF
#
# create a total of 125 frames
#
anim_quick: total_frames 125
anim_quick: create_keyframes
```

See Also:

Associated anim_quick commands

anim_quick: fly_revolutions

Command:

```
anim_quick: fly_revolutions <many_revs>
```

Purpose:

Sets the number of revolutions for the fly-around quick animation.

Parameters:

Type	Parameter	Description
float	many_revs	The number of revolutions to add

Example Usage:

```
#
# turn on keyframe animation
#
anim_keyframe: keyframing ON
#
# move the camera to the right one half revolution
#
anim_quick: fly_around ON
anim_quick: fly_direction right
anim_quick: fly_revolutions 0.5
#
# accelerate at the first keyframe but not at the last
#
anim_quick: accelerate_first ON
anim_quick: accelerate_last OFF
#
# create a total of 125 frames
#
anim_quick: total_frames 125
anim_quick: create_keyframes
```

See Also:

Associated anim_quick commands

Command:

```
anim_quick: rotate_direction <axis> <direction>
```

Purpose:

Specifies the direction of the rotation.

Parameters:

Type	Parameter	Description	
constant	axis	x	specifies direction for x axis
		y	specifies direction for y axis
		z	specifies direction for z axis
constant	direction	positively	positive rotation
		negatively	negative rotation

Example Usage:

```
#
# turn on keyframe animation
#
anim_keyframe: keyframing ON
#
# move the camera to the right one half revolution
#
anim_quick: fly_around ON
anim_quick: fly_direction right
anim_quick: fly_revolutions 0.5
#
# while rotating the scene about the x axis 2 revolutions negative direc-
tion
#
anim_quick: rotate_objects ON
anim_quick: rotate_direction x negatively
anim_quick: rotate_revolutions x 2.000000
#
# accelerate at the first keyframe but not at the last
#
anim_quick: accelerate_first ON
anim_quick: accelerate_last OFF
#
# create a total of 125 frames
#
anim_quick: total_frames 125
anim_quick: create_keyframes
```

See Also:

Associated anim_quick commands

anim_quick: rotate_objects

Command:

```
anim_quick: rotate_objects <toggle>
```

Purpose:

Sets up a scene rotation for quick animation.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	add rotation
		`OFF`	do not add rotation

Notes:

The rotation is not added until a “anim_quick: create_keyframes” command.

The rotation can be used in combination with “fly_around” and “explode_view”

Example Usage:

```
#
# turn on keyframe animation
#
anim_keyframe: keyframing ON
#
# move the camera to the right one half revolution
#
anim_quick: fly_around ON
anim_quick: fly_direction right
anim_quick: fly_revolutions 0.5
#
# while rotating the scene about the x axis 2 revolutions negative direc-
tion
#
anim_quick: rotate_objects ON
anim_quick: rotate_direction x negatively
anim_quick: rotate_revolutions x 2.000000
#
# accelerate at the first keyframe but not at the last
#
anim_quick: accelerate_first ON
anim_quick: accelerate_last OFF
#
# create a total of 125 frames
#
anim_quick: total_frames 125
anim_quick: create_keyframes
```

See Also:

Associated anim_quick commands

Command:

```
anim_quick: rotate_revolutions <many_revs>
```

Purpose:

Sets the number of revolutions for rotate_objects quick animation.

Parameters:

Type	Parameter	Description
float	many_revs	The number of revolutions to add

Example Usage:

```
#
# turn on keyframe animation
#
anim_keyframe: keyframing ON
#
# move the camera to the right one half revolution
#
anim_quick: fly_around ON
anim_quick: fly_direction right
anim_quick: fly_revolutions 0.5
#
# while rotating the scene about the x axis 2 revolutions negative direc-
tion
#
anim_quick: rotate_objects ON
anim_quick: rotate_direction x negatively
anim_quick: rotate_revolutions x 2.000000
#
# accelerate at the first keyframe but not at the last
#
anim_quick: accelerate_first ON
anim_quick: accelerate_last OFF
#
# create a total of 125 frames
#
anim_quick: total_frames 125
anim_quick: create_keyframes
```

See Also:

Associated anim_quick commands

anim_quick: total_frames

Command:

```
anim_quick: total_frames <num_frames>
```

Purpose:

Sets the total number of frames (keyframes plus subframes) to be created when a “anim_quick: create_keyframes” command is executed.

Parameters:

Type	Parameter	Description
int	num_frames	The total number of frames that will be created

Example Usage:

```
#
# turn on keyframe animation
#
anim_keyframe: keyframing ON
#
# move the camera to the right one half revolution
#
anim_quick: fly_around ON
anim_quick: fly_direction right
anim_quick: fly_revolutions 0.5
#
# accelerate at the first keyframe but not at the last
#
anim_quick: accelerate_first ON
anim_quick: accelerate_last OFF
#
# create a total of 125 frames
#
anim_quick: total_frames 125
anim_quick: create_keyframes
```

See Also:

Associated anim_quick commands

Commands in this Section:

anim_recorders

anim_recorders: file_prefix <filename>
anim_recorders: record <toggle>
anim_recorders: render_offscreen <toggle>

See Also:

User Manual - Section 7.15, Keyframe Animation
How To Create a Keyframe Animation

anim_recorders: file_prefix filename

Command:

```
anim_recorders: file_prefix <filename>
```

Purpose:

Specifies the external file name to use for keyframe animation recording.

Parameters:

Type	Parameter	Description
string	`filename`	animation file name prefix

Notes:

The filename can also include an optional directory path. The name is a prefix. A suffix will automatically be added that is based on the animation file format (e.g. '.evo' for EnVideo files)

If a single image format is specified (i.e. PostScript), then a four digit sequence number will be added between the prefix and suffix.

Example Usage:

```
anim_keyframe: keyframing ON
anim_keyframe: create_keyframe
view_transf: rotate 0.0 90.0 0.0
anim_keyframe: create_keyframe
file: image_format envideo
# record envideo file to /tmp/my_animation.evo
anim_recorders: file_prefix /tmp/my_animation
anim_recorders: record ON
anim_keyframe: run
```

See Also:

Associated anim_recorders commands

Command:

```
anim_recorders: record <toggle>
```

Purpose:

Records the keyframe animation to an external file(s).

Parameters:

Type	Parameter	Description	
constant	toggle	ON	turn on recording playback
		OFF	turn off recording

Example Usage:

```
anim_keyframe: keyframing ON
anim_keyframe: create_keyframe
view_transf: rotate 0.0 90.0 0.0
anim_keyframe: create_keyframe
file: image_format envideo
anim_recorders: file_prefix ./my_animation
anim_recorders: record ON
anim_keyframe: run
```

See Also:

Associated anim_recorders commands

Command:

```
anim_recorders: render_offscreen <toggle>
```

Purpose:

Specifies using hardware assisted offscreen rendering for keyframe animation on platforms that support it.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	turn on offscreen rendering
		OFF	turn off offscreen rendering

Notes:

This depends on an OpenGL feature called pbuffers. It is currently supported on SGI platforms. However, due to bugs in SGI's implementation, it can be problematic. You may need to try different video formats (i.e., use ircombine) to get pbuffer rendering to work correctly.

Example Usage:

```
anim_keyframe: keyframing ON
anim_keyframe: create_keyframe
view_transf: rotate 0.0 90.0 0.0
anim_keyframe: create_keyframe
file: image_format envideo
anim_recorders: file_prefix ./my_animation
anim_recorders: record ON
anim_recorders: render_offscreen ON
anim_keyframe: run
```

See Also:

Associated anim_recorders commands

Commands in this Section:

anim_screens

anim_screens: save_multiple_images <toggle>
anim_screens: type <size>
anim_screens: x_max <coord>
anim_screens: x_min <coord>
anim_screens: y_max <coord>
anim_screens: y_min <coord>

See Also:

User Manual - Section 7.15, Keyframe Animation
How To Create a Keyframe Animation

anim_screens: save_multiple_images

Command:

```
anim_screens: save_multiple_images <toggle>
```

Purpose:

Option to save a single image or multiple images fro a flat-wall detached display.

Parameters:

Type	Parameter	Description	
constant	size	ON	to save one image per pipe
		OFF	to save one image per wall

Notes:

When saving a flat-wall detached display, the user can choose to write one large image for the wall or one image per ‘pipe’ from the dconfig file. This option is only valid when the animation size is `detached_display`.

Example Usage:

```
anim_keyframe: create_keyframe
view_transf: rotate -4.500000e+01 7.500000e+01 0.000000e+00
anim_keyframe: create_keyframe
anim_screens: type detached_display
anim_screens: save_multiple_images ON
```

See Also:

Associated anim_screens commands

Command:

```
anim_screens: type <size>
```

Purpose:

Specifies the size of the graphics window to use during keyframe animation playback.

Parameters:

Type	Parameter	Description	
constant	size	detached_display	use the detached display, as specified by the -dconfig command-line argument
		full	use a full screen graphics window
		normal	use the current graphics window
		NTSC	use an NTSC-sized graphics window
		PAL	use a PAL-sized graphics window
		user_defined	use a user specified window size

Notes:

The NTSC window size is 640x480 and corresponds to the size used by US video recording equipment. The PAL window size is 720x576. It is used by most foreign recording equipment. NTSC and PAL sizes have their window origins in the lower left corner of the monitor. user_defined windows can be located anywhere on the monitor.

Example Usage:

```
#
# This plays a keyframe animation in a 320x200 window.
#
anim_keyframe: keyframing ON
anim_keyframe: create_keyframe
view_transf: zoom 0.561365
anim_keyframe: create_keyframe
anim_screens: type user_defined
anim_screens: x_min 0
anim_screens: x_max 319
anim_screens: y_min 0
anim_screens: y_max 199
anim_keyframe: run
```

See Also:

Associated anim_screens commands

anim_screens: x/y_max/min

Command:

```
anim_screens: x_max <coord>
anim_screens: x_min <coord>
anim_screens: y_max <coord>
anim_screens: y_min <coord>
```

Purpose:

Specifies the right x/y coordinate of the keyframe animation window during playback.

Parameters:

Type	Parameter	Description
int	`coord`	screen coordinate of the left (x_min), right (x_max), bottom (y_min) and top (y_max)

Notes:

The range is 0 to maximum graphics resolution - 1.

Example Usage:

```
#
# This plays a keyframe animation in a 320x200 window.
#
anim_keyframe: keyframing ON
anim_keyframe: create_keyframe
view_transf: zoom 0.561365
anim_keyframe: create_keyframe
anim_screens: type user_defined
anim_screens: x_min 0
anim_screens: x_max 319
anim_screens: y_min 0
anim_screens: y_max 199
anim_keyframe: run
```

See Also:

Associated anim_screens commands

Commands in this Section:

anim_traces

anim_traces: color_by <option>
anim_traces: delta_time <time_value>
anim_traces: head_detail <detail_value>
anim_traces: head_scale <scale_value>
anim_traces: head_size_by <option>
anim_traces: head_type <option>
anim_traces: head_variable <variable_name>
anim_traces: line_width <width_value>
anim_traces: max_time <time_value>
anim_traces: multiple_pulses <toggle>
anim_traces: particle_time <time_value>
anim_traces: pulse_interval <time_value>
anim_traces: RGB <red_val> <grn_val> <blu_val>
anim_traces: set_max_time <toggle>
anim_traces: start_time <time_value>
anim_traces: sync_to_transient <toggle>

See Also:

User Manual - Section 7.4, Particle Trace Create/Update

How To Animate Particle Traces

anim_traces: color_by

Command:

```
anim_traces: color_by <option>
```

Purpose:

Specify the option by which to color the animated traces on the selected part(s).

Parameters:

Type	Parameter	Description	
string	option	constant_color	color the animated traces by a constant color
		calculated_color	(default) color the animated traces the same as the part(s)

Notes:

The color of the constant_color option is specified by the "anim_traces: RGB" command.

Example Usage:

```
#
# Select the particle trace part(s) to animate
#
part: select_begin
2
part: select_end
#
# Toggle off the visibility of the selected particle trace part(s), and
# animate them.
#
part: modify_begin
part: visible OFF
part: animate ON
part: modify_end
#
# Color by constant color of magenta
#
anim_traces: color_by constant_color
anim_traces: RGB 1.0000e+00 0.0000e+00 1.0000e+00
```

See Also:

Associated anim_traces commands

Command:

```
anim_traces: delta_time <time_value>
```

Purpose:

Specify how fast (the speed) the animated traces move for the selected particle trace part(s).

Parameters:

Type	Parameter	Description
float	`time_value`	the time that the traces are incremented by between display frames

Notes:

This parameter is not applicable when using "`anim_traces: sync_to_transient ON`", and displaying transient data through flipbook or keyframe animation.

Example Usage:

```
#
# Select the particle trace part(s) to animate
#
part: select_begin
2
part: select_end
#
# Toggle off the visibility of the selected particle trace part(s), and
# animate them.
#
part: modify_begin
part: visible OFF
part: animate ON
part: modify_end
#
# Specify a delta time
#
anim_traces: delta_time 1.0000e-05
```

See Also:

Associated anim_traces commands

anim_traces: head_detail

Command:

```
anim_traces: head_detail <detail_value>
```

Purpose:

Specifies the detail-level of the head representation of all animated traces of the selected particle trace part(s).

Parameters:

Type	Parameter	Description
float	`detail_value`	The detail-level of the head of all animated traces in range I[2,10] with 10 being the most detailed (default = 3.). The higher the detail level, the rounder the sphere head

Notes:

Not applicable when head type is none. Higher detail levels take longer to draw; thus, slowing performance.

Example Usage:

```
#
# Select the particle trace part(s) to animate
#
part: select_begin
2
part: select_end
#
# Toggle off the visibility of the selected particle trace part(s), and
# animate them.
#
part: modify_begin
part: visible OFF
part: animate ON
part: modify_end
#
# Toggle animated trace segment head type to spheres
#
anim_traces: head_type sphere
anim_traces: head_detail 4.0000+e00
```

See Also:

Associated anim_traces commands

Command:

```
anim_traces: head_scale <scale_value>
```

Purpose:

Specify the scaling factor for the head size.

Parameters:

Type	Parameter	Description
float	scale_value	scale factor to apply to the head size.

Notes:

Not applicable when head type is none.

If head_size_by is constant, the scale factor is the diameter of the sphere. If head_size_by is set to a scalar or vector the factor scales the variable selected.

Example Usage:

```
#
# Select the particle trace part(s) to animate
#
part: select_begin
2
part: select_end
#
# Toggle off the visibility of the selected particle trace part(s), and
# animate them.
#
part: modify_begin
part: visible OFF
part: animate ON
part: modify_end
#
# Toggle animated trace segment head type to spheres
#
anim_traces: head_type sphere
anim_traces: head_scale 1.0000e-01
```

See Also:

Associated anim_traces commands

anim_traces: head_size_by

Command:

```
anim_traces: head_size_by <option>
```

Purpose:

Specify which variable type to use to size the head of each animated trace segment of the selected particle trace part(s).

Parameters:

Type	Parameter	Description	
string	option	constant	(default) sizes the head using just the scale factor value (anim_traces: head_scale <scale value>)
		scalar	sizes the head using the specified scalar variable
		vector_mag	sizes the head using the magnitude of the specified vector variable
		vector_xcomp	sizes the head using the x component of the specified vector variable
		vector_ycomp	sizes the head using the x component of the specified vector variable
		vector_zcomp	sizes the head using the x component of the specified vector variable

Notes:

Not applicable when head type is none. The head size is determined by multiplying the scale factor (`anim_traces: head_scale`) by the specified variable (`anim_traces: head_variable`) value.

Example Usage:

```
#
# Select the particle trace part(s) to animate
#
part: select_begin
2
part: select_end
#
# Toggle off the visibility of the selected particle trace part(s), and
# animate them.
#
part: modify_begin
part: visible OFF
part: animate ON
part: modify_end
#
# Toggle animated trace segment head type to spheres
# and size by velocity magnitude
#
anim_traces: head_type sphere
anim_traces: head_size_by vector_mag
anim_traces: head_variable Velocity
```

See Also:

Associated anim_traces commands

Command:

```
anim_traces: head_type <option>
```

Purpose:

Specify the type of head for each animated trace segment for the selected particle trace part(s).

Parameters:

Type	Parameter	Description	
string	option	none	(default) no head will appear for each animated trace segment, just a plain line
		arrow	Use an arrow to represent the head, or leading edge, of each animated trace segment
		sphere	use a sphere to represent the head, or leading edge, of each animated trace segment

Notes:

The following commands help control the size of the sphere at the head of each animated particle trace.

```
anim_traces: head_detail
anim_traces: head_scale
anim_traces: head_size_by
```

Example Usage:

```
#
# Select the particle trace part(s) to animate
#
part: select_begin
2
part: select_end
#
# Toggle off the visibility of the selected particle trace part(s), and
# animate them.
#
part: modify_begin
part: visible OFF
part: animate ON
part: modify_end
#
# Toggle animated trace segment head type to spheres
#
anim_traces: head_type sphere
```

See Also:

Associated anim_traces commands

anim_traces: head_variable

Command:

```
anim_traces: head_variable <variable_name>
```

Purpose:

Specify the variable to use in scaling the head of the animated trace segments of the selected particle trace part(s).

Parameters:

Type	Parameter	Description
string	`variable_name`	variable name to use in scaling the heads of the animated trace segments

Notes:

Not applicable for "`anim_traces: head_sized_by constant`". The scale factor (`anim_traces: head_scale`) will be applied to the value of the specified variable at the location of the head of the animated trace. Variable must be active.

Example Usage:

```
#
# Select the particle trace part(s) to animate
#
part: select_begin
2
part: select_end
#
# Toggle off the visibility of the selected particle trace part(s), and
# animate them.
#
part: modify_begin
part: visible OFF
part: animate ON
part: modify_end
#
# Toggle animated trace segment head type to spheres
# and size by velocity magnitude
#
anim_traces: head_type sphere
anim_traces: head_size_by vector_mag
anim_traces: head_variable Velocity
```

See Also:

Associated anim_traces commands

Command:

```
anim_traces: line_width <width_value>
```

Purpose:

Specify line width of the animated trace segments for the selected particle trace part(s).

Parameters:

Type	Parameter	Description
float	width_value	width of the animated trace line segments (default = 2)

Example Usage:

```
#
# Select the particle trace part(s) to animate
#
part: select_begin
2
part: select_end
#
# Toggle off the visibility of the selected particle trace part(s), and
# animate them.
#
part: modify_begin
part: visible OFF
part: animate ON
part: modify_end
#
# Specify line width to be 3
#
anim_traces: line_width 3.0000e+01
```

See Also:

Associated anim_traces commands

anim_traces: max_time

Command:

```
anim_traces: max_time <time_value>
```

Purpose:

Specify the maximum particle duration.

Parameters:

Type	Parameter	Description
float	`time_value`	time at which to end the animated trace

Notes:

Used in conjunction with "`anim_traces: set_max_time ON`".

Example Usage:

```
#
# Select the particle trace part(s) to animate
#
part: select_begin
2
part: select_end
#
# Toggle off the visibility of the selected particle trace part(s), and
# animate them.
#
part: modify_begin
part: visible OFF
part: animate ON
part: modify_end
#
# End trace animation at .5
#
anim_traces: set_max_time ON
anim_traces: max_time 5.0000e-01
```

See Also:

Associated anim_traces commands

Command:

```
anim_traces: multiple_pulses <toggle>
```

Purpose:

Toggles on/off multiple emission of animated traces for the selected particle trace part(s).

Parameters:

Type	Parameter	Description	
constant	toggle	ON	enables additional animated traces to appear after each specified pulse interval (anim_traces: pulse_interval <interval value>)
		OFF	(default) Only one animated trace appears at the specified start time (anim_traces: start_time <time value>) for each particle trace

Notes:

Not applicable to pathlines. The pulse interval is controlled via the "anim_traces: pulse_interval" command.

When ON, will start a set of animated traces every <time_value> time units (as specified via the "pulse_interval" command).

Example Usage:

```
#
# Select the particle trace part(s) to animate
#
part: select_begin
2
part: select_end
#
# Toggle off the visibility of the selected particle trace part(s), and
# animate them.
#
part: modify_begin
part: visible OFF
part: animate ON
part: modify_end
#
# Toggle on multiple animated trace pulses
#
anim_traces: multiple_pulses ON
anim_traces: pulse_interval 1.0000e-03
```

See Also:

Associated anim_traces commands

anim_traces: particle_time

Command:

```
anim_traces: particle_time <time_value>
```

Purpose:

Specify the time duration (length) of the animated trace segments.

Parameters:

Type	Parameter	Description
float	`time_value`	length in units of time of the animated trace segment

Notes:

The particle trace length parameter scales the length of all animated traces at all times.

The animated trace segments will be short when speed is low and long when speed is high.

Example Usage:

```
#
# Select the particle trace part(s) to animate
#
part: select_begin
2
part: select_end
#
# Toggle off the visibility of the selected particle trace part(s), and
# animate them.
#
part: modify_begin
part: visible OFF
part: animate ON
part: modify_end
#
# Specify a animated trace length
#
anim_traces: particle_time 1.0000e-04
```

See Also:

Associated anim_traces commands

Command:

```
anim_traces: pulse_interval <time_value>
```

Purpose:

Specifies the time delay when using multiple animated trace pulses;

Parameters:

Type	Parameter	Description
float	time_value	time delay interval between trace pulses

Notes:

Not applicable when "anim_traces: multiple_pulses OFF".

Example Usage:

```
#
# Select the particle trace part(s) to animate
#
part: select_begin
2
part: select_end
#
# Toggle off the visibility of the selected particle trace part(s), and
# animate them.
#
part: modify_begin
part: visible OFF
part: animate ON
part: modify_end
#
# Toggle on multiple animated trace pulses
#
anim_traces: multiple_pulses ON
anim_traces: pulse_interval 1.0000e-03
```

See Also:

Associated anim_traces commands

anim_traces: RGB

Command:

```
anim_traces: RGB <red_val> <grn_val> <blu_val>
```

Purpose:

Specify the color for animated traces.

Parameters:

Type	Parameter	Description
float	red_val	red, green, blue color component values given in I[0.,1.]
	grn_val	
	blu_val	

Notes:

Used in conjunction with the "amim_traces: color_by constant" command.

Example Usage:

```
#
# Select the particle trace part(s) to animate
#
part: select_begin
2
part: select_end
#
# Toggle off the visibility of the selected particle trace part(s), and
# animate them.
#
part: modify_begin
part: visible OFF
part: animate ON
part: modify_end
#
# Color by constant color of magenta
#
anim_traces: color_by constant
anim_traces: RGB 1.0000e+00 0.0000e+00 1.0000e+00
```

See Also:

Associated anim_traces commands

Command:

```
anim_traces: set_max_time <toggle>
```

Purpose:

Toggles on/off the maximum lifetime for all animated traces.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	traces animate until the until maximum time is reached
		`OFF`	(default) traces animate until the end of the particle trace is reached

Notes:

Used in conjunction with "`anim_traces: max_time`".

Example Usage:

```
#
# Select the particle trace part(s) to animate
#
part: select_begin
2
part: select_end
#
# Toggle off the visibility of the selected particle trace part(s), and
# animate them.
#
part: modify_begin
part: visible OFF
part: animate ON
part: modify_end
#
# End trace animation at simulation time .5
#
anim_traces: set_max_time ON
anim_traces: max_time 5.0000e-01
```

See Also:

Associated anim_traces commands

anim_traces: start_time

Command:

```
anim_traces: start_time <time_value>
```

Purpose:

Specifies the time value to start the particle animation.

Parameters:

Type	Parameter	Description
float	`time_value`	time at which to begin the animated traces

Example Usage:

```
#
# Select the particle trace part(s) to animate
#
part: select_begin
2
part: select_end
#
# Toggle off the visibility of the selected particle trace part(s), and
# animate them.
#
part: modify_begin
part: visible OFF
part: animate ON
part: modify_end
#
# Start animated traces at .1 rather than 0.
#
anim_traces: start time 1.0000e-01
```

See Also:

Associated anim_traces commands

Command:

```
anim_traces: sync_to_transient <toggle>
```

Purpose:

Toggles on/off synchronization of animated tracer position to solution time of transient data.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	when transient data is in use, each tracer is displayed with its leading-end at the correct location along the particle trace corresponding to the current solution time
		OFF	animated traces not synchronized with transient data

Notes:

Only applicable with transient data cases. (See "solution_time: current_step" and "solution_time: update_to_current".)

Example Usage:

```
#
# Set the
#
solution_time: current_step 4.0000e+01
solution_time: update_to_current
#
# Select the particle trace part(s) to animate
#
part: select_begin
2
part: select_end
#
# Toggle off the visibility of the selected particle trace part(s), and
# animate them.
#
part: modify_begin
part: visible OFF
part: animate ON
part: modify_end
#
# Toggle off synchronization with transient data
#
anim_traces: sync_to_transient OFF
```

See Also:

Associated anim_traces commands

Commands in this Section:

annot_entlbl

annot_entlbl: element_rgb <red_val> <grn_val> <blu_val>
annot_entlbl: element_threshold_high <element_id>
annot_entlbl: element_threshold_low <element_id>
annot_entlbl: element_thresholds <filter_setting>
annot_entlbl: node_rgb <red_val> <grn_val> <blu_val>
annot_entlbl: node_threshold_high <node_id>
annot_entlbl: node_threshold_low <node_id>
annot_entlbl: node_thresholds <filter_setting>

See Also:

User Manual - Section 8.1, Part Mode

annot_entlbl: element_rgb

Command:

```
annot_entlbl: element_rgb <red_val> <grn_val> <blu_val>
```

Purpose:

Sets the color of element id labels.

Parameters:

Type	Parameter	Description
float	red_val	red, green, blue component of the color (0.0 to 1.0)
	grn_val	
	blu_val	

Notes:

This is a normal rgb color triple. The effect of this command is global - namely, all element labels are affected.

Example Usage:

```
annot_entlbl: element_rgb 1.000000 0.400000 0.800000
```

See Also:

Associated annot_entlbl commands

Command:

```
annot_entlbl: element_threshold_high <element_id>
annot_entlbl: element_threshold_low <element_id>
```

Purpose:

Sets the high/low threshold value for element label filtering.

Parameters:

Type	Parameter	Description
int	`element_id`	high/low threshold element id

Notes:

The threshold high/low value is used according to the filter method set in the `annot_entlbl: element_thresholds` associated command.

Example Usage:

```
annot_entlbl: element_thresholds band
annot_entlbl: element_threshold_low 30
annot_entlbl: element_threshold_high 1200
```

See Also:

Associated annot_entlbl commands

annot_entlbl: element_thresholds

Command:

```
annot_entlbl: element_thresholds <filter_setting>
```

Purpose:

Sets method to use for filtering element labels.

Parameters:

Type	Parameter	Description	
constant	`filter_setting`	`low`	Element numbers below the low threshold are not displayed.
		`high`	Element numbers above the high threshold are not displayed.
		`band`	Element numbers between the low and high threshold are not displayed.
		`low_high`	Element numbers below the low and above the high thresholds are not displayed.
		`none`	No element labels are displayed.

Notes:

This command sets the method. The threshold low and high values are set in associated commands

Example Usage:

```
annot_entlbl: element_thresholds band
annot_entlbl: element_threshold_low 30
annot_entlbl: element_threshold_high 1200
```

See Also:

Associated annot_entlbl commands

Command:

```
annot_entlbl: node_rgb <red_val> <grn_val> <blu_val>
```

Purpose:

Sets the color of node id labels.

Parameters:

Type	Parameter	Description
float	red_val	red, green, blue components of the color (0.0 to 1.0)
	grn_val	
	blu_val	

Notes:

This is a normal rgb color triple. The effect of this command is global - namely, all node labels are affected.

Example Usage:

```
annot_entlbl: node_rgb 0.000000 1.000000 0.800000
```

See Also:

Associated annot_entlbl commands

annot_entlbl: node_threshold_high/low

Command:

```
annot_entlbl: node_threshold_high <node_id>
annot_entlbl: node_threshold_low <node_id>
```

Purpose:

Sets the high/low threshold value for node label filtering.

Parameters:

Type	Parameter	Description
int	`node_id`	high/low threshold node id

Notes:

The threshold high/low value is used according to the filter method set in the `annot_entlbl: node_thresholds` associated command.

Example Usage:

```
annot_entlbl: node_thresholds low_high
annot_entlbl: node_threshold_low 2
annot_entlbl: node_threshold_high 20
```

See Also:

Associated annot_entlbl commands

Command:

```
annot_entlbl: node_thresholds <filter_setting>
```

Purpose:

Sets method to use for filtering node labels.

Parameters:

Type	Parameter	Description	
constant	filter_setting	low	node numbers below the low threshold are not displayed.
		high	node numbers above the high threshold are not displayed.
		band	node numbers between the low and high threshold are not displayed.
		low_high	node numbers below the low and above the high thresholds are not displayed.
		none	no node labels are displayed.

Notes:

This command sets the method. The threshold low and high values are set in associated commands

Example Usage:

```
annot_entlbl: node_thresholds low_high
annot_entlbl: node_threshold_low 2
annot_entlbl: node_threshold_high 20
```

See Also:

Associated annot_entlbl commands

Commands in this Section:

annotation

See Also:

User Manual - Section 8.2, Annot Mode

How To Create Color Legends

How To Create Lines and Arrows

How To Create Text Annotation

How To Load Custom Logos

annotation: axis_global

Command:

```
annotation: axis_global <toggle>
```

Purpose:

Turns the global axis triad on or off.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	turn the triad on
		`OFF`	turn the triad off

Notes:

The global axis' origin is located at the centroid of the visible geometry. Its location is recomputed when the viewport is re initialized. The global axis will have 'G' at its origin to distinguish it from other axes.

Example Usage:

```
annotation: axis_global ON
```

See Also:

Associated annotation commands

Command:

```
annotation: axis_local <toggle>
```

Purpose:

Toggles the global attribute controlling local frame visibility.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	turn the global toggle on
		`OFF`	turn the global toggle off

Notes:

Local coordinate frames are visible only if their axis visibility attribute are on AND the global toggle described here is `ON`.

Example Usage:

```
frame: create
frame: select_begin
1
frame: select_end
annotation: axis_local ON
frame: x_labels ON
frame: len_x 1.0000e+03
frame: len_y 7.4565e+02
frame: len_z 7.4565e+02
frame: rgb 0.0000e+00 0.0000e+00 1.0000e+00
frame: visible ON
```

See Also:

Associated annotation commands

Associated frame commands

annotation: axis_model

Command:

```
annotation: axis_model <toggle>
```

Purpose:

Turns the model axis triad on or off.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	turn the triad on
		`OFF`	turn the triad off

Notes:

The model axis' origin is fixed in the lower left corner of the viewport. It is intended to be a simple indicator of which way the X, Y, and Z axes lie.

Example Usage:

```
annotation: axis_model ON
```

See Also:

Associated annotation commands

Command:

```
annotation: deselect_all
```

Purpose:

Deselects all annotations.

Parameters:

none

Example Usage:

```
text: new_text A Test
text: select_begin
 0
text: select_end
text: new_text A Second Test
text: select_begin
 1
text: select_end
text: select_all
text: select_begin
 0 1
text: select_end
annotation: deselect_all
```

See Also:

Associated annotation commands

annotation: element_labeling

Command:

```
annotation: element_labeling <toggle>
```

Purpose:

Turns element labels on or off globally.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	turn labels on globally
		`OFF`	turn labels off globally

Notes:

Element labels for individual parts can be toggled on or off; however, for any labels to be visible, the global setting must be on. A threshold can also be enabled to limit the range of visible labels.

Example Usage:

```
annotation: element_labeling ON
view_transf: function global
part: select_partname_begin
"(CASE:Case 1)Subset of parts: 2"
part: select_partname_end
part: modify_begin
part: entity_label_elt ON
part: modify_end
view_transf: function global
annot_entlbl: node_thresholds high
annot_entlbl: node_threshold_low 1
annot_entlbl: node_threshold_high 10
annot_entlbl: node_rgb 0.000000 0.000000 1.000000
```

See Also:

Associated annotation commands

Command:

```
annotation: legend <toggle>
```

Purpose:

Toggles visibility of all visible legends.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	turn all visible legends on
		OFF	turn all visible legends off

Example Usage:

```
legend: select_palette_begin
pressure
legend: select_palette_end
legend: visible ON
legend: select_palette_begin
velocity
legend: select_palette_end
legend: visible ON
legend: select_palette_begin
pressure
legend: select_palette_end
legend: select_palette_begin
pressure
legend: select_palette_end
legend: location_x 3.2353e-01
legend: location_y 5.3681e-02
legend: select_palette_begin
pressure
legend: select_palette_end
annotation: legend off
```

See Also:

Associated annotation commands

annotation: node_labeling

Command:

```
annotation: node_labeling <toggle>
```

Purpose:

Turns node labels on or off globally.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	turn labels on globally
		`OFF`	turn labels off globally

Notes:

Node labels for individual parts can be toggled on or off; however, for any labels to be visible, the global setting must be on. A threshold can also be enabled to limit the range of visible labels.

Example Usage:

```
annotation: node_labeling ON
view_transf: function global
view_transf: function global
part: modify_begin
part: entity_label_node ON
part: modify_end
view_transf: function global
annot_entlbl: element_thresholds band
annot_entlbl: element_threshold_low 10
annot_entlbl: element_threshold_high 20
annot_entlbl: element_rgb 0.000000 1.000000 0.000000
```

See Also:

Associated annotation commands

Command:

```
annotation: text_display <toggle>
```

Purpose:

Toggles the visibility of all text, line, and logo annotations.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	turns on all annotations
		`OFF`	turns off all annotations

Notes:

This command does not affect the visibility of legends.

Example Usage:

```
text: new_text My Test Data
# turn off all annotations
annotation: text_display OFF
```

See Also:

Associated annotation commands

Commands in this Section:

boundarylayer

boundarylayer: create_update
boundarylayer: density <variable_name>
boundarylayer: determine_velocity_by <option>
boundarylayer: dynamic_viscosity <value>
boundarylayer: freestream_density <density_value>
boundarylayer: freestream_velocity <velocity_value>
boundarylayer: modify_begin
boundarylayer: modify_end
boundarylayer: momentum <variable_name>
boundarylayer: normal_distance <distance_value>
boundarylayer: velocity <variable_name>
boundarylayer: velocity_magnitude <velocity_value>

See Also:

User Manual - Section 7.22, Boundary Layer Variables Create/Update
How To Extract Boundary Layer Variables

boundarylayer: create_update

Command:

```
boundarylayer: create_update
```

Purpose:

Specify to create or update the boundary-layer variables on the selected 2D part(s).

Parameters:

None.

Notes:

The following boundary-layer variables will be created or updated at the nodes of the selected 2D parts:

`bl_thickness` - boundary-layer thickness

`bl_displ_thickness` - displacement thickness

`bl_momen_thickness` - momentum thickness

`bl_shape_parameter` - shape parameter

`bl_skin_friction_Cf` - skin-friction coefficient

Example Usage:

```
part: select_begin
 2
part: select_end
boundarylayer: create_update
```

See Also:

Associated boundarylayer commands

Command:

```
boundarylayer: density <variable_name>
```

Purpose:

Specify the density variable to use in creating/updating the boundary-layer variables.

Parameters:

Type	Parameter	Description
string	`variable_name`	name of the density variable (the default name `"none"` means to use nothing)

Notes:

This variable gets assigned automatically if the variable name `"Density"` exists. Since the computation of separation and attachment lines depend on velocity, either the velocity variable must be defined or both the density and momentum variables must be defined to obtain the velocity variable.

Example Usage:

```
boundarylayer: modify_begin
boundarylayer: density Density
boundarylayer: modify_end
```

See Also:

Associated boundarylayer commands

boundarylayer: determine_velocity_by

Command:

```
boundarylayer: determine_velocity_by <option>
```

Purpose:

Specify how to assign the velocity just outside the boundary layer in calculating the boundary-layer variables.

Parameters:

Type	Parameter	Description	
constant	option	convergence_criteria	(default) determines the velocity just outside the boundary layer to be determined via convergence criteria. That is, when either the velocity magnitude becomes constant or the gradient of the velocity magnitude approaches zero or becomes negative.
		distance_from_surface	determines the velocity magnitude just outside the boundary layer for each node to be that velocity at a specified distance normal from the 2D surface
		velocity_magnitude	determines the velocity magnitude just outside the boundary layer to be a specified velocity magnitude

Example Usage:

```
boundarylayer: determine_velocity_by distance_from_surface
boundarylayer: normal_distance 1.0000e-03
part: select_begin
 2
part: select_end
boundarylayer: create_update
```

See Also:

Associated boundarylayer commands

Command:

```
boundarylayer: dynamic_viscosity <value>
```

Purpose:

Specify the dynamic viscosity constant value or variable name to use in creating/updating the boundary-layer variables.

Parameters:

Type	Parameter	Description
string	`value`	the scalar field variable name for the dynamic viscosity
float	`value`	a constant value (default = 1.8300e-05) for the dynamic viscosity

Notes:

The `<value>` can be multiple types depending on variable definition. This variable is required to compute the fluid shear stress of the fluid for the skin-friction coefficient variable.

Example Usage:

```
boundarylayer: modify_begin
boundarylayer: dynamic_viscosity 1.8300e-05
boundarylayer: modify_end
```

See Also:

Associated boundarylayer commands

boundarylayer: freestream_density

Command:

```
boundarylayer: freestream_density <density_value>
```

Purpose:

Specify the freestream density constant value to use in creating/updating the boundary-layer variables.

Parameters:

Type	Parameter	Description
float	`density_value`	constant value (default = 1.) for the freestream density

Notes:

This variable is required to compute the skin-friction coefficient variable.

Example Usage:

```
boundarylayer: modify_begin
boundarylayer: freestream_density 9.0000e-01
boundarylayer: modify_end
```

See Also:

Associated boundarylayer commands

Command:

```
boundarylayer: freestream_velocity <velocity_value>
```

Purpose:

Specify the freestream velocity constant value to use in creating/updating the boundary-layer variables.

Parameters:

Type	Parameter	Description
float	velocity_value	a constant value (default = 1.) for the freestream velocity

Notes:

This variable is required to compute the skin-friction coefficient variable.

Example Usage:

```
boundarylayer: modify_begin
boundarylayer: freestream_velocity 8.8500e-01
boundarylayer: modify_end
```

See Also:

Associated boundarylayer commands

boundarylayer: modify_begin/end

Command:

```
boundarylayer: modify_begin
<command>
boundarylayer: modify_end
```

Purpose:

Specify the execute construct within which to define a dependent variable used in computing the boundary-layer variables.

Parameters:

Type	Parameter	Description
string	`command`	boundary layer variable command to define or update

Example Usage:

```
boundarylayer: modify_begin
boundarylayer: freestream_velocity 8.8500e-01
boundarylayer: modify_end
```

See Also:

Associated boundarylayer commands

Command:

```
boundarylayer: momentum <variable_name>
```

Purpose:

Specify the momentum variable to use in creating/updating the boundary-layer variables.

Parameters:

Type	Parameter	Description
string	`variable_name`	name of the momentum variable (the default name `<none>` means to use nothing)

Notes:

This variable gets assigned automatically if the variable name `<momentum>` exists. Since the computation of boundary-layer variables depend on velocity, either the velocity variable must be defined or both the density and momentum variables must be defined to obtain the velocity variable.

Example Usage:

```
boundarylayer: modify_begin
boundarylayer: momentum none
boundarylayer: modify_end
```

See Also:

Associated boundarylayer commands

boundarylayer: normal_distance

Command:

```
boundarylayer: normal_distance <distance_value>
```

Purpose:

Specify the distance normal from the surface in which to determine the velocity just outside the boundary layer in order to compute the boundary-layer variables at each node on the surface.

Parameters:

Type	Parameter	Description
float	`distance_value`	distance from the surface at which to compute the velocity just outside the boundary layer

Notes:

Only used for the `boundarylayer: determine_velocity_by distance_from_surface` option.

Example Usage:

```
boundarylayer: determine_velocity_by distance_from_surface
boundarylayer: normal_distance 1.0000e+00
part: select_begin
 2
part: select_end
boundarylayer: create_update
```

See Also:

Associated boundarylayer commands

Command:

```
boundarylayer: velocity <variable_name>
```

Purpose:

Specify the velocity variable to use in creating/updating the boundary-layer variables.

Parameters:

Type	Parameter	Description
string	`variable_name`	name of the velocity variable (the default name `<none>` means to use nothing)

Notes:

This variable gets assigned automatically if the variable name `<Velocity>` exists. Since the computation of boundary-layer variables depend on velocity, either the velocity variable must be defined or both the density and momentum variables must be defined to obtain the velocity variable.

Example Usage:

```
boundarylayer: modify_begin
boundarylayer: velocity Velocity
boundarylayer: modify_end
```

See Also:

Associated boundarylayer commands

boundarylayer: velocity_magnitude

Command:

```
boundarylayer: velocity_magnitude <velocity_value>
```

Purpose:

Specify the velocity magnitude normal at which to determine the velocity just outside the boundary layer in order to compute the boundary-layer variables at each node on the surface.

Parameters:

Type	Parameter	Description
float	`velocity_value`	velocity magnitude at which to assign the velocity just outside the boundary layer

Notes:

Only used for the boundarylayer: `determine_velocity_by distance_from_surface` option.

Example Usage:

```
boundarylayer: determine_velocity_by velocity_magnitude
boundarylayer: velocity_magnitude 1.0000e+00
part: select_begin
 2
part: select_end
boundarylayer: create_update
```

See Also:

Associated boundarylayer commands

Commands in this Section:

case

case: add <case_id>
case: apply_context <toggle>
case: create_viewport <toggle>
case: delete <case_id>
case: reflect_model_in <option>
case: reflect_model_origin <x_comp> <y_comp> <z_comp>
case: replace <old_value> <new_value>
case: select <case_id>
case: viewports <vp# vp# ... vp#>

See Also:

User Manual - Section 6.6, Case Menu Functions
How To Load Multiple Datasets (Cases)

case: add

Command:

```
case: add <case_id>
```

Purpose:

Add another case dataset to your current EnSight session.

Parameters:

Type	Parameter	Description
string	`case_id`	id of the new case (see Notes below)

Notes:

By default the new case names are called "`Case #`" where the # is the next available case # - 1 based.

Example Usage:

```
case: create_viewport ON
case: apply_context OFF
case: relfect_model_in 'none'
case: add Case 2
case: select Case 2
viewport: select_begin
 1
viewport: select_end
```

See Also:

Associated case commands

Command:

```
case: apply_context <toggle>
```

Purpose:

Specify whether to apply context from the initial case when adding another new case.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	to apply context
		OFF	otherwise

Example Usage:

```
case: create_viewport ON
case: apply_context ON
case: relfect_model_in 'none'
case: add Case 2
case: select Case 2
viewport: select_begin
 1
viewport: select_end
```

See Also:

Associated case commands

case: create_viewport

Command:

```
case: create_viewport <toggle>
```

Purpose:

Specify whether to create another viewport in which to display the new case when adding a new case.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	to create another viewport
		`OFF`	otherwise

Example Usage:

```
case: create_viewport ON
case: apply_context OFF
case: relfect_model_in 'none'
case: add Case 2
case: select Case 2
viewport: select_begin
 1
viewport: select_end
```

See Also:

Associated case commands

Command:

```
case: delete <case_id>
```

Purpose:

Delete the currently selected case dataset from your current EnSight session.

Parameters:

Type	Parameter	Description
string	`case_id`	id of the new case (see Notes below)

Notes:

By default the new case names are called "`Case #`" where the # is the next available case # - 1 based.

Example Usage:

```
#
# select case 2 for deletion
#
case: select 2
#
# delete case 2
#
case: delete 2
#
# select case 1 as current, or default
#
case: select 1
```

See Also:

Associated case commands

Command:

```
case: reflect_model_in <option>
```

Purpose:

Specify whether to apply a mirror transform when adding a new case.

Parameters:

Type	Parameter	Description	
string	option	Includes the following options:	
		'none'	no reflection (default)
		'X'	reflect model about the x axis
		'Y'	reflect model about the y axis
		'Z'	reflect model about the z axis
		'X''Y'	or any combination of 'X', 'Y', 'Z'.

Notes:

The reflection transform occurs about the point specifies via the "reflect_model_origin" command.

Example Usage:

```
case: create_viewport ON
case: apply_context OFF
case: relfect_model_in 'Z'
case: add Case 2
case: select Case 2
viewport: select_begin
 1
viewport: select_end
```

See Also:

Associated case commands

Command:

```
case: reflect_model_origin <x_comp> <y_comp> <z_comp>
```

Purpose:

Specify the model origin about which to reflect the newly added case

Parameters:

Type	Parameter	Description
float	x_comp	x component of the origin (in model coordinates) about which to reflect the new case dataset
	y_comp	y component of the origin (in model coordinates) about which to reflect the new case dataset
	z_comp	z component of the origin (in model coordinates) about which to reflect the new case dataset

Notes:

Default origin is 0., 0., 0.

Used with the "reflect_model_in" command.

Example Usage:

```
case: create_viewport ON
case: apply_context OFF
case: reflect_model_in 'X'
case: reflect_model_in 1.0000e+00 0.0000e+00 0.0000e+00
case: add Case 2
case: select Case 2
viewport: select_begin
 1
viewport: select_end
```

See Also:

Associated case commands

case: replace

Command:

```
case: replace <old_value> <new_value>
```

Purpose:

Replace the current selected case dataset with another case dataset.

Parameters:

Type	Parameter	Description
string	old_value	name of the case to be replaced
	new_value	name (id) of the case replacing the old string value

Notes:

By default the new case names are called "Case #" where the # is the next available case # - 1 based.

Example Usage:

```
#
# select 2nd case
#
case: select Case 2
#
# replace 2nd case w/2nd case
#
case: replace 'Case 2'
#
# select 2nd case‘Case 2'
#
case: select Case 2
```

or

```
#
# select 2nd case
#
case: select Case 2
#
# replace 2nd case w/3rd case
#
case: replace 'Case 2''Case 3'
#
# select 2nd case
#
case: select Case 3
```

See Also:

Associated case commands

Command:

```
case: select <case_id>
```

Purpose:

Specify current (active) case.

Parameters:

Type	Parameter	Description
string	case_id	id of the active case

Notes:

Many operations depend on the active case, such as dataset queries.

Example Usage:

```
case: select Case 2
```

See Also:

Associated case commands

case: viewports

Command:

```
case: viewports <vp# vp# ... vp#>
```

Purpose:

Specify which viewport in which to display the data from a case.

Parameters:

Type	Parameter	Description
int	vp#	viewport integer # in I[1,N] which correspond to the respective viewports in which case data is to be displayed. N is the number of active viewports.

Notes:

The active case is modified (see "`case: select`").

Example Usage:

```
#
# Preliminary setup (for reference) - given two viewports
#
viewport: select_begin
 0
viewport: select_end
view_transf: function global
viewport: viewport_layout two_horizontal
viewport: select_begin
 1
viewport: select_end
#
# Displaying a case in both viewports
#
case: viewports 1 2
#
# Displaying a case in only the first viewport
case viewports 1
```

See Also:

Associated case commands

Commands in this Section:

clip

clip: angle <cone_angle>
clip: axis <x_comp> <y_comp> <z_comp>
clip: begin
clip: end
clip: box_axis <axis> <x_vector> <y_vector> <z_vector>
clip: box_length <x_len> <y_len> <z_len>
clip: box_min <xmin> <ymin> <zmin>
clip: box_max <xmax> <ymax> <zmax>
clip: box_origin <x_ori> <y_ori> <z_ori>
clip: create
clip: delta <dx> <dy> <dz>
clip: dimension2 <min> <max>
clip: dimension3 <min> <max>
clip: dimension2_step <step>
clip: dimension3_step <step>
clip: domain <domain>
clip: general_equation <A> <B> <C> <D> <E> <F> <G> <H> <I> <J>
clip: grid_pts <x_pts> <y_pts>
clip: ijk_axis <toggle>
clip: ijk_axis_scale <scale>
clip: interactive_type <type>
clip: line <end> <x_coord> <y_coord> <z_coord>
clip: mesh_plane <plane>
clip: origin <x_coord> <y_coord> <z_coord>
clip: plane <corner> <x_coord> <y_coord> <z_coord>
clip: plane_extents <extent>
clip: plane_type <type>
clip: point <pt_num> <location> <radius>
clip: pts_on_line <line_pts>
clip: radius <value>
clip: revolution_points <number>
clip: revolve_part <part_id>
clip: revolve_partbyname <part_name>
clip: rtz_axis <toggle>
clip: select_default
clip: slider_range_ijk <beg_ijk> <end_ijk>
clip: slider_step_ijk <step_increment>
clip: slider_step_ijk <step_increment>
clip: slider_x <min_value> <max_value>
clip: slider_y <min_value> <max_value>
clip: slider_z <min_value> <max_value>
clip: tool <tool_type>
clip: value <plane_value>
clip: x_min_infinite <toggle>
clip: y_min_infinite <toggle>
clip: z_min_infinite <toggle>

clip: x_max_infinite <toggle>
clip: y_max_infinite <toggle>
clip: z_max_infinite <toggle>

See Also:

User Manual - Section 7.5, Clip Create/Update
How To Create Clips
How To Cut Parts

Command:

```
clip: angle <cone_angle>
```

Purpose:

Set the angle for a cone clip.

Parameters:

Type	Parameter	Description
float	cone_angle	the cone angle in degrees

Example Usage:

```
#
# Select the parent part
#
part: select_begin
 1
part: select_end
clip: begin
clip: domain intersect
clip: tool cone
clip: origin 0 0 0
clip: axis 1 0 0
#
# Set the cone to 20 degrees
#
clip: angle 20.
clip: end
clip: create
```

See Also:

Associated clip commands

clip: axis

Command:

```
clip: axis <x_comp> <y_comp> <z_comp>
```

Purpose:

Set the direction vector for quadric and revolution clips.

Parameters:

Type	Parameter	Description
float	x_comp	x, y, z components of the direction vector
	y_comp	
	z_comp	

Notes:

The origin together with the axis defines the location and orientation of the quadric and revolution tools.

The origin and axis define the point and axis about which a 1D part is revolved.

Example Usage:

```
#
# Select the parent part
#
part: select_begin
 1
part: select_end
clip: begin
clip: domain intersect
clip: tool cylinder
clip: origin 0 0 0
#
# Set the cone axis to lie along the x axis
#
clip: axis 1 0 0
clip: radius .5
clip: end
clip: create
```

See Also:

Associated clip commands

Command:

```
clip: begin
clip: end
```

Purpose:

Delimit the modifications for clip attributes.

Parameters:

none

Notes:

These commands are used to set the attributes for a clip part creation.

Example Usage:

```
clip: begin
#
# Change attributes
#
clip: domain inside
clip: tool plane
clip: plane 1 -1 -1 0
clip: plane 2 1 -1 0
clip: plane 3 1 1 0
#
# End change attributes
#
clip: end
clip: create
```

See Also:

Associated clip commands

clip: box_axis

Command:

```
clip: box_axis <axis> <x_vector> <y_vector> <z_vector>
```

Purpose:

Specify the direction vectors for a box clip.

Parameters:

Type	Parameter	Description	
constant	axis	x	specifies which axis is being defined
		y	
		z	
float	x_vector	The x component of the direction vector	
	y_vector	The y component of the direction vector	
	z_vector	The z component of the direction vector	

Notes:

The direction vectors are in reference to the parent part's frame.

Example Usage:

```
#
# Select a parent part
#
part: select_begin
 2
part: select_end
clip: begin
#
# Create a xyz box clip with the box aligned with the
# global axis system and with an origin -.75, -.75, -.25
# The box sides are of length 1., 1.5, 2.
#
clip: tool xyz_box
clip: box_origin -7.500000e-01 -7.500000e-01 -2.500000e-01
clip: box_axis x 1.000000e+00 0.000000e+00 0.000000e+00
clip: box_axis y 0.000000e+00 1.000000e+00 0.000000e+00
clip: box_axis z 0.000000e+00 0.000000e+00 1.000000e+00
clip: box_length 1.000000e+00 1.500000e+00 2.000000e+00
clip: domain intersect
clip: end
clip: create
```

See Also:

Associated clip commands

Command:

```
clip: box_length <x_len> <y_len> <z_len>
```

Purpose:

Specify the size of the box clip in the x, y, and z directions

Parameters:

Type	Parameter	Description
float	x_len	The length of box clip in the x direction
	y_len	The length of box clip in the y direction
	z_len	The length of box clip in the z direction

Notes:

The lengths are in relation to the box axis directions.

Example Usage:

```
#
# Select a parent part
#
part: select_begin
 2
part: select_end
clip: begin
#
# Create a xyz box clip with the box aligned with the
# global axis system and with an origin -.75, -.75, -.25
# The box sides are of length 1., 1.5, 2.
#
clip: tool xyz_box
clip: box_origin -7.500000e-01 -7.500000e-01 -2.500000e-01
clip: box_axis x 1.000000e+00 0.000000e+00 0.000000e+00
clip: box_axis y 0.000000e+00 1.000000e+00 0.000000e+00
clip: box_axis z 0.000000e+00 0.000000e+00 1.000000e+00
clip: box_length 1.000000e+00 1.500000e+00 2.000000e+00
clip: domain intersect
clip: end
clip: create
```

See Also:

Associated clip commands

clip: box_min/max

Command:

```
clip: box_min <xmin> <ymin> <zmin>
clip: box_max <xmax> <ymax> <zmax>
```

Purpose:

set the bounds for the XYZ Box extraction.

(Note, these commands are no longer in use as of EnSight7.6)

Parameters:

Type	Parameter	Description
float	xmin	minimum X value for extract
	ymin	minimum Y value for extract
	zmin	minimum Z value for extract
	xmax	maximum X value for extract
	ymax	maximum Y value for extract
	zmax	maximum Z value for extract

Notes:

The X/Y/Z min/max values are used only in the cases that the `x/y/z_min/max_infinite` toggles are set off.

Example Usage:

```
#
# Select an existing clip part
#
part: select_begin
 3
part: select_end
part: modify_begin
clip: tool xyz_box
clip: y_max_infinite OFF
#
# Set the minimum box elements
#
clip: box_min -1 -1 -1
part: modify_end
```

See Also:

Associated clip commands

Command:

```
clip: box_origin <x_ori> <y_ori> <z_ori>
```

Purpose:

Specify the origin of the box clip in reference to the parent part's frame

Parameters:

Type	Parameter	Description
float	x_ori	The x-origin of the box clip
	y_ori	The y-origin of the box clip
	z_ori	The z-origin of the box clip

Notes:

The origin is in reference to the parent part's frame.

Example Usage:

```
#
# Select a parent part
#
part: select_begin
 2
part: select_end
clip: begin
#
# Create a xyz box clip with the box aligned with the
# global axis system and with an origin -.75, -.75, -.25
# The box sides are of length 1., 1.5, 2.
#
clip: tool xyz_box
clip: box_origin -7.500000e-01 -7.500000e-01 -2.500000e-01
clip: box_axis x 1.000000e+00 0.000000e+00 0.000000e+00
clip: box_axis y 0.000000e+00 1.000000e+00 0.000000e+00
clip: box_axis z 0.000000e+00 0.000000e+00 1.000000e+00
clip: box_length 1.000000e+00 1.500000e+00 2.000000e+00
clip: domain intersect
clip: end
clip: create
```

See Also:

Associated clip commands

clip: create

Command:

```
clip: create
```

Purpose:

Create a clip part with the currently defined attributes using the currently selected parts as parents.

Parameters:

none

Notes:

The clip part is created using the currently selected parts as parents. The clip part is created using the currently set clip attributes.

Example Usage:

```
#
# Select a parent part
#
part: select_begin
2
part: select_end
clip: begin
clip: domain inside
clip: tool cylinder
clip: origin -1. 0 1.
clip: axis 1 0 0
clip: radius .75
clip: end
clip: create
```

See Also:

Associated clip commands

Command:

```
clip: delta <dx> <dy> <dz>
```

Purpose:

Set a delta transform for use with the keyframe animator.

Parameters:

Type	Parameter	Description
float	dx	amount to translate in x direction for each animation frame
	dy	amount to translate in y direction for each animation frame
	dz	amount to translate in z direction for each animation frame

Notes:

Applies only to line and plane clip types.

Example Usage:

```
#
# Select an existing clip part
#
part: select_begin
 3
part_select_end
part: modify_begin
#
# Increment the clip position by 0.2 in the x-direction
#
clip: delta .2 0 0
part: modify_end
anim_keyframe: keyframing ON
anim_keyframe: create_keyframe
anim_keyframe: create_keyframe
anim_keyframe: run
```

See Also:

Associated clip commands

clip: dimension2/dimension3

Command:

```
clip: dimension2 <min> <max>
clip: dimension3 <min> <max>
```

Purpose:

Specify the minimum/maximum range for the non-constant I/J/K extract.

Parameters:

Type	Parameter	Description
int	min	minimum plane value integer
	max	maximum plane value integer

Notes:

For an I/J/K extract the user specifies that he wants to create a part of constant I, J, or K and an extract value. These parameters specify the range for the non-constant plane.

Example Usage:

```
#
# Select an existing I plane
# extract an modify the J range = 10 to 60
# and the K range to be 1 to 45
#
part: select_begin
 3
part: select_end
part: modify_begin
clip: dimension2 10 60
clip: dimension3 1 45
part: modify_end
```

See Also:

Associated clip commands

Command:

```
clip: dimension2_step <step>
clip: dimension3_step <step>
```

Purpose:

Specify the step increment for the non-constant I/J/K extract.

Parameters:

Type	Parameter	Description
int	`step`	step value

Notes:

For an I/J/K extract the user specifies that he wants to create a part of constant I, J, or K and a extract value. These parameters specify the step increment for the non-constant plane.

Example Usage:

```
#
# Select an existing I plane extract
# and modify the J step by factor
# to 2

part: select_begin
 3
part: select_end
part: modify_begin
clip: dimension2 10 60
clip: dimension2_step 2
clip: dimension3 1 45
part: modify_end
```

See Also:

Associated clip commands

clip: domain

Command:

```
clip: domain <domain>
```

Purpose:

Specify what type of clip will be created.

Parameters:

Type	Parameter	Description	
constant	domain	intersect	to create a part which is the intersection between the clip tool and the parent parts
		inside	to create a part which contains the geometry (including intersected elements) in the positive half space of the clip tool
		outside	to create a part which contains the geometry (including intersected elements) in the negative half space of the clip tool
		in_out	to create two parts - one for the inside and one for the outside half spaces
		crinkly	to create a part which contains the geometry of the elements which intersect the clip tool

Notes:

`Intersect` applies to all tools except XYZ Box.

`Inside/outside/in_out` apply to all tools except XYZ, IJK, and Line.

When domain is set to `inside/outside/in_out` the parent parts will be made invisible.

The default is `intersect`.

Example Usage:

```
#
# Select the part
#
part: select_begin
2
part: select_end
clip: begin
#
# Create a new part inside the
# cylinder tool
#
clip: domain inside
clip: tool cylinder
clip: origin -1. 0 1.
clip: axis 1 0 0
clip: radius .75
clip: end
clip: create
```

See Also:

Associated clip commands

Command:

```
clip: general_equation <A> <B> <C> <D> <E> <F> <G> <H> <I> <J>
```

Purpose:

Clip to a specified quadric equation of the form

Ax^2 + By^2 + Cz^2 + Dxy + Eyz + Fxz + Gx + Hy + Iz = J

Parameters:

Type	Parameter	Description
float	A	The coefficient of the x^2 term
	B	The coefficient of the y^2 term
	C	The coefficient of the z^2 term
	D	The coefficient of the xy term
	E	The coefficient of the yz term
	F	The coefficient of the xz term
	G	The coefficient of the x term
	H	The coefficient of the y term
	I	The coefficient of the z term
	J	The constant term

Example Usage:

```
#
# Select the parent part
#
part: select_begin
 1
part: select_end
clip: begin
#
# Create a clip of the form
# x^2 + y^2 + z^2 = .5
#
clip: domain intersect
clip: tool general_quadric
clip: general_equation 1 1 1 0 0 0 0 0 0 .5
clip: end
clip: create
```

See Also:

Associated clip commands

clip: grid_pts

Command:

```
clip: grid_pts <x_pts> <y_pts>
```

Purpose:

Set the number of sample points for a planar grid clip.

Parameters:

Type	Parameter	Description
int	x_pts	number of points to sample along the x plane tool direction
	y_pts	number of points to sample along the y plane tool direction

Example Usage:

```
#
# Modify an existing
# grid clip part
#
part: select_begin
 3
part: select_end
part: modify_begin
clip: grid_pts 10 12
part: modify_end
```

See Also:

Associated clip commands

Command:

```
clip: ijk_axis <toggle>
```

Purpose:

Display an axis system for a IJK extract.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	to show the axis
		OFF	for no axis

Notes:

Is only valid for I/J/K plane clips.

Example Usage:

```
#
# Turn on an axis showing the
# IJK layout of an existing
# clip part
#
part: select_begin
 3
part: select_end
part: modify_begin
clip: ijk_axis ON
clip: ijk_axis_scale 1.5
part: modify_end
```

See Also:

Associated clip commands

clip: ijk_axis_scale

Command:

```
clip: ijk_axis_scale <scale>
```

Purpose:

Specify the length of the IJK axis for IJK extracts.

Parameters:

Type	Parameter	Description
float	`scale`	length of the axis in world coordinates

Notes:

Is only valid for I/J/K plane clips.

Example Usage:

```
part: select_begin
 3
part: select_end
part: modify_begin
clip: ijk_axis ON
clip: ijk_axis_scale 1.5
part: modify_end
```

See Also:

Associated clip commands

Command:

```
clip: interactive_type <type>
```

Purpose:

Specify the interactive setting for a clip part.

Parameters:

Type	Parameter	Description	
constant	type	off	the clip is not in interactive mode
		manual	set the interactive state to manual such that the clip value can be updated via the user interface slider
		auto	set the interactive state to auto such that the clip is incremented by the interactive_auto_delta value
		auto_cycle	same as "auto" except that when the max is reached the interactive_auto_delta value is decremented back to the min value.

Notes:

The min and max values are constrained via the `slider_range_ijk` commands.

Example Usage:

```
part: select_begin
  2
part: select_end
isos: interactive_type auto
```

See Also:

Associated clip commands

clip: line

Command:

```
clip: line <end> <x_coord> <y_coord> <z_coord>
```

Purpose:

Set an end point for a clip line.

Parameters:

Type	Parameter	Description
constant	`end`	`1` or `2`, to set specified line end point
float	`x_coord`	x coordinate of end point
	`y_coord`	y coordinate of end point
	`z_coord`	z coordinate of end point

Notes:

Two end points are necessary to properly define the line location.

Example Usage:

```
#
# Select a parent part
#
part: select_begin
 2
part: select_end
clip: begin
#
# Create a line clip from
# {-1, -1, 0} to {1, -1, 0} with
# 25 evenly spaced points
#
clip: tool line
clip: pts_on_line 25
clip: line 1 -1 -1 0
clip: line 2 1 -1 0
clip: end
clip: create
```

See Also:

Associated clip commands

Command:

```
clip: mesh_plane <plane>
```

Purpose:

Set the plane for XYZ or IJK clips

Parameters:

Type	Parameter	Description
constant	plane	X - Clip of constant X
		Y - Clip of constant Y
		Z - Clip of constant Z
		I - Clip of constant I
		J - Clip of constant J
		K - Clip of constant K
		R - Clip of constant radius
		T - Clip of constant theta angle (in degrees)

Notes:

IJK clips can only be created on block structured parts.

Example Usage:

```
#
# Select a parent part
#
part: select_begin
  2
part: select_end
clip: begin
#
# Create a clip at y=1.5
#
clip: value 1.5
clip: domain intersect
clip: tool xyz
clip: mesh_plane Y
clip: end
clip: create
```

See Also:

Associated clip commands

clip: origin

Command:

```
clip: origin <x_coord> <y_coord> <z_coord>
```

Purpose:

Set origin for quadric and revolution clips.

Parameters:

Type	Parameter	Description
float	x_coord	x, y, z coordinates of origin
	y_coord	
	z_coord	

Notes:

The origin together with the axis defines the location and orientation of the quadric and revolution tools.

The origin and axis define the point and axis about which a 1D part is revolved.

Example Usage:

```
#
# Select the parent part
#
part: select_begin
1
part: select_end
clip: begin
#
# Create a cylinder clip
# defined by the origin at {0. 0. 0}
# a direction vector of {1, 0, 0}
# and a radius of 0,5
#
clip: domain intersect
clip: tool cylinder
clip: origin 0 0 0
clip: axis 1 0 0
clip: radius .5
clip: end
clip: create
```

See Also:

Associated clip commands

Command:

```
clip: plane <corner> <x_coord> <y_coord> <z_coord>
```

Purpose:

Set a corner for a clip plane.

Parameters:

Type	Parameter	Description
constant	corner	1, 2 or 3, to set specified plane corner
float	x_coord	x, y, z coordinates of corner
	y_coord	
	z_coord	

Notes:

Three corners are necessary to properly define the plane location.

Example Usage:

```
#
# Select the parent part
#
part: select_begin
 2
part: select_end
clip: begin
#
# Create a finite clip plane
# with corners as defined
#
clip: tool plane
clip: plane_extents finite
clip: plane 1 -1 -1 0
clip: plane 2 1 -1 0
clip: plane 3 1 1 0
clip: end
clip: create
```

See Also:

Associated clip commands

clip: plane_extents

Command:

```
clip: plane_extents <extent>
```

Purpose:

Set the extents to infinite or finite for a plane clip.

Parameters:

Type	Parameter	Description	
constant	`extent`	`finite`	to create a clip plane which is bound by the plane tool
		`infinite`	to create an infinite clip in the plane of the plane tool

Notes:

For planes that are clipped using a grid `plane_type` the extents are always finite.

The default is `infinite` (except grid clips which are `finite`).

Quadric clips are always `infinite`.

Example Usage:

```
#
# Select the parent part
#
part: select_begin
 2
part: select_end
clip: begin
#
# Create a finite clip plance
# with bounds as defined
#
clip: tool plane
clip: domain intersect
clip: plane_extents finite
clip: plane 1 -1 -1 0
clip: plane 2 1 -1 0
clip: plane 3 1 1 0
clip: end
clip: create
```

See Also:

Associated clip commands

Command:

```
clip: plane_type <type>
```

Purpose:

For a clip using the plane tool specifies that a true cross-section (mesh) should be created or a uniform sample (grid).

Parameters:

Type	Parameter	Description	
constant	type	mesh	(default) to create a true cross-section
		grid	to sample along the x and y plane directions

Notes:

Applies only to clips created with the plane tool.

Grid clips are by definition finite in nature.

Example Usage:

```
#
# Select the parent part
# part: select_begin
2
part: select_end
clip: begin
clip: tool plane
clip: domain intersect
clip: plane_type mesh
clip: plane 1 -1 -1 0
clip: plane 2 1 -1 0
clip: plane 3 1 1 0
clip: end
clip: create
```

See Also:

Associated clip commands

clip: point

Command:

```
clip: point <pt_num> <location> <radius>
```

Purpose:

Defines points for the surface of revolution.

Parameters:

Type	Parameter	Description
int	`pt_num`	point number
	`location`	location along the line
	`radius`	radius at the point location

Notes:

There can be at most 10 points in the surface of revolution.

Is only used when "`clip: tool`" is set to "`revolution`".

Example Usage:

```
#
# Select the parent part
#
part: select_begin
2
part: select_end
clip: begin
#
# Create a clip using a tool
# of revolution with origin at
# {.5, .5, 0} and lined up with
# the x-axis
#
clip: domain intersect
clip: revolution_points 5
clip: tool revolution
clip: origin .5 .5 0
clip: axis 1 0 0
clip: point 1 -1 .5
clip: point 2 0 .7
clip: point 3 1 .5
clip: end
clip: create
```

See Also:

Associated clip commands

Command:

```
clip: pts_on_line <line_pts>
```

Purpose:

Set the number of sample points for a line clip.

Parameters:

Type	Parameter	Description
int	line_pts	number of linearly spaced sample points for the line clip

Example Usage:

```
#
# Select an existing line clip
# and set the number of
# sample points to 20
#
part: select_begin
 3
part: select_end
part: modify_begin
clip: pts_on_line 20
part: modify_end
```

See Also:

Associated clip commands

clip: radius

Command:

```
clip: radius <value>
```

Purpose:

Set the radius for a sphere or cylinder clip.

Parameters:

Type	Parameter	Description
float	`value`	radius for the sphere or cylinder clip

Example Usage:

```
#
# Select the parent part
#
part: select_begin
 1
part: select_end
clip: begin
#
# Create a cylinder clip of
# radius=0.5 with origin and
# axis as shown
#
clip: domain intersect
clip: tool cylinder
clip: origin 0 0 0
clip: axis 1 0 0
clip: radius .5
clip: end
clip: create
```

See Also:

Associated clip commands

Command:

```
clip: revolution_points <number>
```

Purpose:

Set the number of points used in the tool of revolution for clipping.

Parameters:

Type	Parameter	Description
int	number	number of points used for the revolution tool

Example Usage:

```
#
# Select the parent part
#
part: select_begin
 2
part: select_end
clip: begin
#
# Create a clip of revolution
# with five defined points
#
clip: domain intersect
clip: revolution_points 5.0000e+00
clip: tool revolution
clip: origin 5.0000e-01 5.0000e-01 1.0000e+00
clip: axis 1.0000e+00 0.0000e+00 0.0000e+00
clip: point 1 -1.2500e+00 5.0000e-01
clip: point 2 -6.2500e-01 5.0000e-01
clip: point 3 0.0000e+00 1.0000e+00
clip: point 4 6.2500e-01 5.0000e-01
clip: point 5 1.2500e+00 5.0000e-01
clip: end
clip: create
```

See Also:

Associated clip commands

clip: revolve_part

Command:

```
clip: revolve_part <part_id>
```

Purpose:

Clip by revolving a 1D part about an axis.

Parameters:

Type	Parameter	Description
int	part_id	part ID number to revolve

Notes:

The end points for the 1D part are effectively extended to infinity during the clip.

The part is revolved around the defined origin and axis.

Example Usage:

```
#
# Select an existing clip part
#
part: select_begin
 3
part: select_end
part: modify_begin
#
# Revolve 1D part 4
# about an axis
#
clip: tool 1d_part
clip: revolve_part 4
clip: origin -2.4838e-01 -2.3340e-01 1.8757e+00
clip: axis 0.0000e+00 1.0000e+00 0.0000e+00
part: modify_end
```

See Also:

Associated clip commands

Command:

```
clip: revolve_partbyname <part_name>
```

Purpose:

Clip by revolving a 1D part about an axis.

Parameters:

Type	Parameter	Description
string	part_name	part name to revolve

Notes:

This command is functionally the same as "clip: revolve_part" except it takes a part name as the argument instead of a part id.

The end points for the 1D part are effectively extended to infinity during the clip.

The part is revolved around the defined origin and axis.

Example Usage:

```
#
# Select an existing clip part
#
part: select_begin
 3
part: select_end
part: modify_begin
#
# Revolve part “clip line” about an axis
#
clip: tool 1d_part
clip: revolve_part clip line
clip: origin -2.4838e-01 -2.3340e-01 1.8757e+00
clip: axis 0.0000e+00 1.0000e+00 0.0000e+00
part: modify_end
```

See Also:

Associated clip commands

clip: rtz_axis

Command:

```
clip: rtz_axis <toggle>
```

Purpose:

Specify the axis to be used for RTZ clips. Theta is about this axis.

Parameters:

Type	Parameter	Description	
constant	axis	X	The x frame axis defines the RTZ system.
		Y	The y frame axis defines the RTZ system.
		Z	The z frame axis defines the RTZ system.

Notes:

The axis is in reference to the frame associated with the parent part(s).

Example Usage:

```
#
# Select a parent part
#
part: select_begin
 2
part: select_end
clip: begin
#
# Create a Z RTZ clip about the Z axis at Z=0.
#
clip: tool rtz
clip: rtz_axis Z
clip: value 0.
clip: domain intersect
clip: mesh_plane Z
clip: end
clip: create
```

See Also:

Associated clip commands

Command:

```
clip: select_default
```

Purpose:

Sets the target to the default settings for subsequent attribute modifications.

Parameters:

none

Example Usage:

```
#
# Set the default clip tool to cylinder
#
clip: select_default
part: modify_begin
clip: tool cylinder
part: modify_end
```

See Also:

Associated clip commands

clip: slider_range_ijk

Command:

```
clip: slider_range_ijk <beg_ijk> <end_ijk>
```

Purpose:

Set the I, J, or K range through which an IJK clip part will be modified when in interactive mode, whether using the slider in Manual mode or in Auto or Auto Cycle mode.

Parameters:

Type	Parameter	Description
int	`beg_ijk`	begining I, J, or K plane of range.
int	`end_ijk`	ending I, J, or K plane of range.

Notes:

Is only used for interactive clips.

Example Usage:

```
#
# Set the slider ijk range for an
# existing IJK clip plane to be within 5 and 35
#
part: select_begin
 2
part: select_end
part: modify_begin
clip: tool ijk
clip: slider_range_ijk 5 35
part: modify_end
```

See Also:

Associated clip commands

Command:

```
clip: slider_step <step_increment>
```

Purpose:

Set the value that a X/Y/Z clip part is modified when in interactive mode and the user interface less/greater buttons are selected.

Parameters:

Type	Parameter	Description
float	`step_increment`	step increment of slider

Notes:

A negative step value is used when the less button is selected while a positive step value is used when the greater button is selected.

Is only used for interactive clips.

Example Usage:

```
#
# Set the step increment for an
# existing clip plane to 1.0
#
part: select_begin
 3
part: select_end
part: modify_begin
clip: tool xyz
clip: slider_step 1.0
part: modify_end
```

See Also:

Associated clip commands

clip: slider_step_ijk

Command:

```
clip: slider_step_ijk <step_increment>
```

Purpose:

Set the value that an IJK clip part is modified when in interactive mode and the user interface less/greater buttons are selected.

Parameters:

Type	Parameter	Description
int	`step_increment`	step increment of slider

Notes:

A negative step value is used when the less button is selected while a positive step value is used when the greater button is selected.

Is only used for interactive clips.

Example Usage:

```
#
# Set the step increment for an
# existing IJK clip plane to 3
#
part: select_begin
 2
part: select_end
part: modify_begin
clip: tool ijk
clip: slider_step)ijk 3
part: modify_end
```

See Also:

Associated clip commands

Command:

```
clip: slider_x <min_value> <max_value>
clip: slider_y <min_value> <max_value>
clip: slider_z <min_value> <max_value>
```

Purpose:

Set the min/max X/Y/Z extents in the user interface for a clip of constant X/Y/Z.

Parameters:

Type	Parameter	Description
float	min_value	minimum slider value
	max_value	maximum slider value

Notes:

Is only used for interactive clips.

Example Usage:

```
#
# Set the min/max slider value for
# a clip of constant X
#
part: select_begin
 3
part: select_end
part: modify_begin
clip: slider_x -1 4
part: modify_end
```

See Also:

Associated clip commands

clip: tool

Command:

```
clip: tool <tool_type>
```

Purpose:

Set the clipping tool.

Parameters:

Type	Parameter	Description	
constant	tool_type	1d_part	Clip by rotating a 1D part about an axis
		cone	Clip using a cone
		cylinder	Clip using a cylinder
		general_quadric	Clip by specifying a quadric equation
		ijk	Extract a constant I, J, or K plane from a block structured part
		line	Create a line with linearly spaced sample points
		plane	Clip using a plane
		revolution	Clip by a tool of revolution
		rtz	Clip to a constant radial, theta, or z coordinate
		sphere	Clip using a sphere
		xyz	Clip to a constant x, y, or z coordinate
		xyz_box	Cut to the boundary of a box

Notes:

`ijk` extract can only occur using a single parent part.

Example Usage:

```
#
# Select the parent part
#
part: select_begin
 2
part: select_end
clip: begin
clip: domain inside
clip: tool cylinder
clip: origin -1. 0 1.
clip: axis 1 0 0
clip: radius .75
clip: end
clip: create
```

See Also:

Associated clip commands

Command:

```
clip: value <plane_value>
```

Purpose:

Specify the constant plane value for XYZ or IJK clips.

Parameters:

Type	Parameter	Description
float	`plane_value`	value for the plane

Notes:

For IJK clips the `plane_value` must be an integer.

Example Usage:

```
#
# Select the parent part
#
part: select_begin
 2
part: select_end
clip: begin
#
# Create a clip of y=1.5
#
clip: value 1.5
clip: domain intersect
clip: tool xyz
clip: mesh_plane Y
clip: end
clip: create
```

See Also:

Associated clip commands

clip: x/y/z_min/max_infinite

Command:

```
clip: x_min_infinite <toggle>
clip: y_min_infinite <toggle>
clip: z_min_infinite <toggle>
clip: x_max_infinite <toggle>
clip: y_max_infinite <toggle>
clip: z_max_infinite <toggle>
```

Purpose:

Specify if the x/y/z min/max bounds of a xyz_box clip is infinite.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	for infinite bound
		`OFF`	for finite bound

Notes:

If finite bound the value is taken from the "`clip: box_min/max`" command.

Example Usage:

```
#
# Select an existing box clip
#
part: select_begin
 3
part: select_end
part: modify_begin
clip: tool xyz_box
#
# Set the minimum y value to -1.
#
clip: y_min_infinite OFF
clip: box_min 0. -1. 0.
part: modify_end
```

See Also:

Associated clip commands

Commands in this Section:

command

command: delay <value>
command: delay_refresh <toggle>
command: part_selection_by <method>

See Also:

User Manual - Section 2.4, Command Files
How To Record and Play Command Files

command: delay

Command:

```
command: delay <value>
```

Purpose:

Causes a delay between each command while playing commands from a command file.

Parameters:

Type	Parameter	Description
int	`value`	time in seconds

Notes:

This command is useful for slowing down the playback speed of command files. It is typically used when playing command files for demos.

Example Usage:

```
command delay_refresh OFF
command: delay 2
```

See Also:

Associated command commands

Command:

```
command: delay_refresh <toggle>
```

Purpose:

Causes a delay between each command while playing commands from a command file.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	updates display after last command
		`OFF`	updates display after each command

Notes:

If delay refresh is on, the graphics are not redrawn after each command.

Playback of command files can be sped up toggling on `delay_refresh` at the beginning of the command file and then off towards the end (i.e. just before recording of an animation).

It is advantageous to use this technique for batch animations.

Example Usage:

```
#
# Save time by not drawing the image after each command
#
command: delay_refresh ON
#
# Other commands to set up the animation
#
# restore graphics updates
#
command: delay_refresh OFF
```

See Also:

Associated command commands

command: part_selection_by

Command:

```
command: part_selection_by <method>
```

Purpose:

Determines if command language will record part selections by name or by number as it appears in the part list.

Parameters:

Type	Parameter	Description	
constant	method	number	record part selections by part list #
		name	record part selections by part list name

Notes:

This setting is typically controlled in the Preference dialog. Creating command files that reference names instead of numbers may be more portable if the names used in the data files are relatively generic.

Example Usage:

```
#
# Turn visibility of part
# number 2 off
#
command: part_selection_by number
part: select_begin
2
part: select_end
part: modify_begin
part: visible OFF
part: modify_end
command: part_selection_by name
#
# Turn visibility of part named "external flow field"
# off
#
part: select_partname_begin
"external flow field"
part: select_partname_end
part: modify_begin
part: visible ON
part: modify_end
```

See Also:

Associated command commands

Commands in this Section:

connect

See Also:

User Manual - Section 6.1, File Menu Functions

How To Connect Automatically

connect: backend

Command:

```
connect: backend
```

Purpose:

Indicates that the following '`connect:`' commands will apply to new server connections.

Parameters:

none

Notes:

To change the default server connection settings, use the '`connect: default`' command.

Example Usage:

```
connect: backend
connect: machine localhost
connect: executable ensight7.server
connect: directory
connect: type auto
connect: connect
```

See Also:

Associated connect commands

Command:

```
connect: connect
```

Purpose:

Cause the client to wait for a connection to the EnSight server.

Parameters:

none

Notes:

The client will try to start the server if an auto connection is specified.

The client will wait and listen for a connection from the server. By default it will listen on TCP port 1106. This port number can be modified with the '`-ports #`' argument to the client and server.

Example Usage:

```
connect: backend
connect: machine localhost
connect: executable ensight7.server
connect: directory
connect: type auto
connect: connect
```

See Also:

Associated connect commands

connect: default

Command:

```
connect: default
```

Purpose:

Indicates that the following '`connect:`' commands will change the EnSight defaults for future sessions.

Parameters:

none

Notes:

To temporarily change server connection settings, use the '`connect: backend`' command.

Example Usage:

```
connect: default
connect: machine localhost
connect: executable ensight7.server
connect: directory /tmp
connect: type auto
```

See Also:

Associated connect commands

Command:

```
connect: directory <pathname>
```

Purpose:

Specifies the directory shown in the Data Reader dialog.

Parameters:

Type	Parameter	Description
string	pathname	name of the directory

Example Usage:

```
# Show the /tmp director on 'slim' in the Data Reader dialog
connect: backend
connect: machine slim
connect: executable ensight7.server
connect: directory /tmp
connect: type auto
connect: connect
```

See Also:

Associated connect commands

connect: executable filename

Command:

```
connect: executable <filename>
```

Purpose:

Specifies the file name of the EnSight server for automatic connections initiated by the client.

Parameters:

Type	Parameter	Description
string	`filename`	name of the server

Notes:

The filename can be a complete pathname to the server otherwise the server executable must be found in your default path on the host running the server.

Example Usage:

```
# use server found in /usr/local/bin/
connect: backend
connect: machine slim
connect: executable /usr/local/bin/ensight7.server
connect: directory /tmp
connect: type auto
connect: connect
```

See Also:

Associated connect commands

Command:

```
connect: login_id <username>
```

Purpose:

Specifies a different user name to use on the computer where the server will run.

Parameters:

Type	Parameter	Description
string	`username`	login name

Notes:

For auto connections the client will try to start the server on a remote computer using the '`rsh`' command. If a user name is specified, it will be used as part of the rsh command. In lieu of the rsh command, a different command can be used as specified by the '`-rsh command`' argument to the client.

Example Usage:

```
# log in to host slim as user 'joe'
connect: backend
connect: machine slim
connect: login_id joe
connect: executable ensight7.server
connect: directory
connect: type auto
connect: connect
```

See Also:

Associated connect commands

connect: machine hostname

Command:

```
connect: machine <hostname>
```

Purpose:

Specifies the hostname where EnSight server should be run for automatic connections initiated by the client.

Parameters:

Type	Parameter	Description
string	`hostname`	name of the computer where the server should run

Notes:

The name specified must be an Internet accessible hostname or IP address. You must be able to rlogin (or ssh) into this machine.

Example Usage:

```
# use an auto connection to host 'slim'
connect: backend
connect: machine slim
connect: executable ensight7.server
connect: directory /tmp
connect: type auto
connect: connect
```

See Also:

Associated connect commands

Command:

```
connect: type <method>
```

Purpose:

Specifies whether the EnSight server should be started by the client or by the user.

Parameters:

Type	Parameter	Description	
constant	method	auto	client starts the server
		manual	user starts the server

Example Usage:

```
# use a manual connection
connect: backend
connect: type manual
connect: connect


# use an auto connection
connect: backend
connect: machine slim
connect: executable ensight7.server
connect: directory /tmp
connect: type auto
connect: connect
```

See Also:

Associated connect commands

Commands in this Section:

contour

See Also:

User Manual - Section 7.2, Contour Create/Update

How To Create Contours

contour: begin/end

Command:

```
contour: begin
contour: end
```

Purpose:

Delimit the modifications for contour attributes.

Parameters:

none

Example Usage:

```
part: select_begin
 2
part: select_end
contour: begin
contour: variable temperature
contour: end
contour: create
```

See Also:

Associated contour commands

Command:

```
contour: component <vector_component>
```

Purpose:

If contouring by a vector specifies component or magnitude.

Parameters:

Type	Parameter	Description	
constant	vector_component	One of the following:	
		x	use x component
		y	use y component
		z	use z component
		magnitude	use magnitude

Notes:

If not specified "magnitude" is used.

Example Usage:

```
#
# Create contours of the x-component
# of velocity on part 2
#
part: select_begin
 2
part: select_end
contour: begin
contour: variable velocity
contour: component x
contour: end
contour: create
```

See Also:

Associated contour commands

contour: create

Command:

```
contour: create
```

Purpose:

Create a contour part with the currently defined attributes using the currently selected parts as parent parts.

Parameters:

none

Notes:

The contour part is created using the currently selected parts as the parent part.

The contour part is created using the currently set contour attributes.

Example Usage:

```
part: select_begin
 2
part: select_end
contour: begin
contour: variable temperature
contour: end
contour: create
```

See Also:

Associated contour commands

Command:

```
contour: display_offset <offset>
```

Purpose:

Translate the contour geometry in the surface normal direction by `<offset>` units.

Parameters:

Type	Parameter	Description
float	`offset`	number of units in world coordinates to translate the contour information

Notes:

Useful option when displaying contour parts on top of shaded objects in order to lift the contour line objects closer to the user to properly view them.

Works only if the surface normals are consistent.

Positive values are offsets in the surface normal direction while negative values are offsets in the opposite direction.

Example Usage:

```
#
# Create a contour part of velocity magnitude
# displaying it offset from the surface by .01 in the
# negative surface normal direction
#
part: select_begin
 2
part: select_end
contour: begin
contour: variable velocity
contour: display_offset -.01
contour: end
contour: create
```

See Also:

Associated contour commands

contour: distribution

Command:

```
contour: distribution <distribution>
```

Purpose:

Specifies the minimum and maximum contour levels when contours are not tied to the variable's palette.

Parameters:

Type	Parameter	Description	
constant	distribution	linear	linear distribution between min and max
		logarithmic	logarithmic distribution between min and max
		quadratic	quadratic distribution between min and max

Notes:

Only used when "contour: sync_to_palette OFF"

Example Usage:

```
#
# Create contours on part 2
#
part: select_begin
 2
part: select_end
contour: begin
contour: variable temperature
contour: sync_to_palette OFF
#
# create 8 contours from 0 to 4.5 using a quadratic distribution
#
contour: distribution quadratic
contour: range 0. 4.5
contour: levels 8
#
# turn on labels spaced every 2 units
# using a floating point representation with 2 decimal places
#
contour: label_visible ON
contour: label_spacing 2.0
contour: label_format floating_point
contour: label_decimal_places 2
contour: end
contour: create
```

See Also:

Associated contour commands

Command:

```
contour: label_decimal_places <num>
```

Purpose:

Specify the number of decimal places to use for contour labels.

Parameters:

Type	Parameter	Description
int	num	The number of decimal places

Notes:

Used only if labels are visible

Example Usage:

```
#
# Create contours on part 2
#
part: select_begin
 2
part: select_end
contour: begin
contour: variable temperature
contour: sync_to_palette OFF
#
# create 8 contours from 0 to 4.5 using a quadratic distribution
#
contour: distribution quadratic
contour: range 0. 4.5
contour: levels 8
#
# turn on labels spaced every 2 units
# using a floating point representation with 2 decimal places
#
contour: levels_visible ON
contour: label_visible ON
contour: label_spacing 2.0
contour: label_format floating_point
contour: label_decimal_places 2
contour: end
contour: create
```

See Also:

Associated contour commands

contour: label_format

Command:

```
contour: label_format <format>
```

Purpose:

Specify the format for contour labels.

Parameters:

Type	Parameter	Description	
constant	format	floating_point	Use floating point format
		exponential	Use exponential format

Example Usage:

```
#
# Create contours on part 2
#
part: select_begin
 2
part: select_end
contour: begin
contour: variable temperature
contour: sync_to_palette OFF
#
# create 8 contours from 0 to 4.5 using a quadratic distribution
#
contour: distribution quadratic
contour: range 0. 4.5
contour: levels 8
#
# turn on labels spaced every 2 units
# using a floating point representation with 2 decimal places
#
contour: levels_visible ON
contour: label_visible ON
contour: label_spacing 2.0
contour: label_format floating_point
contour: label_decimal_places 2
contour: end
contour: create
```

See Also:

Associated contour commands

Command:

```
contour: label_rgb <red_val> <grn_val> <blu_val>
```

Purpose:

Set the color for contour labels.

Parameters:

Type	Parameter	Description
float	red_val	red, green, blue components of color (0 to 1.)
	grn_val	
	blu_val	

Example Usage:

```
#
# Modify an existing contour part to
# have red labels
#
part: select_begin
 3
part: select_end
part: modify_begin
contour: label_visible ON
contour: label_rgb 1. 0. 0.
part: modify_end
```

See Also:

Associated contour commands

contour: label_spacing

Command:

```
contour: label_spacing <distance>
```

Purpose:

Control the spacing of contour labels

Parameters:

Type	Parameter	Description
float	`distance`	distance between labels

Notes:

The labels will be spaced approximately the value specified.

Example Usage:

```
#
# Modify an existing contour part to have
# labels .5 units apart
#
part: select_begin
 3
part: select_end
part: modify_begin
contour: label_visible ON
contour: label_spacing .5
part: modify_end
```

See Also:

Associated contour commands

Command:

```
contour: label_visible <toggle>
```

Purpose:

Turn labels on or off for contour parts.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	to show labels
		OFF	to show no labels

Notes:

The contour labels are shown as integer values indicating the contour level which corresponds to the color palette level.

Level 1 is the lowest level for the color palette.

Subcontours are not labeled.

Example Usage:

```
#
# Modify an existing contour part to show labels
#
part: select_begin
 3
part: select_end
part: modify_begin
contour: label_visible ON
part: modify_end
```

See Also:

Associated contour commands

contour: levels

Command:

```
contour: levels <num_levels>
```

Purpose:

Specifies how many contour levels are created between the min and max values.

Parameters:

Type	Parameter	Description
int	num_levels	The number of contour levels to create

Notes:

Only used when “contour: sync_to_palette OFF”

Example Usage:

```
#
# Create contours on part 2
#
part: select_begin
 2
part: select_end
contour: begin
contour: variable temperature
contour: sync_to_palette OFF
#
# create 8 contours from 0 to 4.5 using a quadratic distribution
#
contour: distribution quadratic
contour: range 0. 4.5
contour: levels 8
#
# turn on labels spaced every 2 units
# using a floating point representation with 2 decimal places
#
contour: label_visible ON
contour: label_spacing 2.0
contour: label_format floating_point
contour: label_decimal_places 2
contour: end
contour: create
```

See Also:

Associated contour commands

Command:

```
contour: levels_visible <toggle>
```

Purpose:

Specifies the visibility of the contour levels created between the min and max values.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	if the contour levels are visible
		OFF	if the contour levels are not visible

Notes:

Only used when “contour: sync_to_palette OFF”

Example Usage:

```
#
# Create contours on part 2
#
part: select_begin
 2
part: select_end
contour: begin
contour: variable temperature
contour: sync_to_palette OFF
#
# create 8 contours from 0 to 4.5 using a quadratic distribution
#
contour: distribution quadratic
contour: range 0. 4.5
contour: levels 8
#
# turn on labels spaced every 2 units
# using a floating point representation with 2 decimal places
#
contour: levels_visible ON
contour: label_visible ON
contour: label_spacing 2.0
contour: label_format floating_point
contour: label_decimal_places 2
contour: end
contour: create
```

See Also:

Associated contour commands

Command:

```
contour: range <min> <max>
```

Purpose:

Specifies the minimum and maximum contour levels when contours are not tied to the variable's palette.

Parameters:

Type	Parameter	Description
float	min	The minimum contour value
	max	The maximum contour value

Notes:

Only used when "contour: sync_to_palette OFF"

Example Usage:

```
#
# Create contours on part 2
#
part: select_begin
 2
part: select_end
contour: begin
contour: variable temperature
contour: sync_to_palette OFF
#
# create 8 contours from 0 to 4.5 using a quadratic distribution
#
contour: distribution quadratic
contour: range 0. 4.5
contour: levels 8
#
# turn on labels spaced every 2 units
# using a floating point representation with 2 decimal places
#
contour: label_visible ON
contour: label_spacing 2.0
contour: label_format floating_point
contour: label_decimal_places 2
contour: end
contour: create
```

See Also:

Associated contour commands

Command:

```
contour: select_default
```

Purpose:

Sets the target to the default settings for subsequent attribute modifications.

Parameters:

none

Example Usage

```
#
# Set the default so two sub-contours will be generated
#
contour: select_default
part: modify_begin
contour: subcontours 2
part: modify_end
```

See Also:

Associated contour commands

contour: subcontours

Command:

```
contour: subcontours <num>
```

Purpose:

Add a specified number of contour loops linearly distributed between the variable palette levels.

(*Note, this command is no longer in use as of EnSight 7.6)*

Parameters:

Type	Parameter	Description
int	num	number of contour values to create between variable palette levels

Notes:

Contours are generated using the color palette for the variable as the contour values. By adding subcontours additional contour levels are generated between the defined levels.

Example Usage:

```
#
# Create contours on part 2 using 5 sub-contours
#
part: select_begin
 2
part: select_end
contour: begin
contour: variable temperature
contour: subcontours 5
contour: end
contour: create
```

See Also:

Associated contour commands

Command:

```
contour: sublevels <num>
```

Purpose:

Add a specified number of contours distributed between the main contour levels.

Parameters:

Type	Parameter	Description
int	num	Number of contours to create between the main contour levels.

Example Usage:

```
#
# Create contours on part 2
#
part: select_begin
 2
part: select_end
contour: begin
contour: variable temperature
contour: sync_to_palette OFF
#
# create 8 contours from 0 to 4.5 using a quadratic distribution
#
contour: distribution quadratic
contour: range 0. 4.5
contour: levels 8
#
# turn off the visiblity of the main contours and turn
# subcontours on - make 2 subcontours between each of the 8 levels.
#
contour: levels_visible OFF
contour: sublevels_visible ON
contour: sublevels 2
contour: end
contour: create
```

See Also:

Associated contour commands

contour: sublevels_visible

Command:

```
contour: sublevels_visible <toggle>
```

Purpose:

Specifies the visibility of the contour sub-levels created between the min and max values.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	if the contour sub-levels are visible
		`OFF`	if the contour sub-levels are not visible

Notes:

Only used when “contour: sync_to_palette OFF”

Example Usage:

```
#
# Create contours on part 2
#
part: select_begin
 2
part: select_end
contour: begin
contour: variable temperature
contour: sync_to_palette OFF
#
# create 8 contours from 0 to 4.5 using a quadratic distribution
#
contour: distribution quadratic
contour: range 0. 4.5
contour: levels 8
#
# turn off the visiblity of the main contours and turn
# subcontours on
#
contour: levels_visible OFF
contour: sublevels_visible ON
contour: sublevels 2
contour: end
contour: create
```

See Also:

Associated contour commands

Command:

```
contour: sync_to_palette <toggle>
```

Purpose:

Specifies contour part creation to be tied to the variable palette or by user input.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	if the contour part is created using the variable's palette
		OFF	if the contour part is created using the defined min/max information

Notes:

Example Usage:

```
#
# Create contours on part 2
#
part: select_begin
 2
part: select_end
contour: begin
contour: variable temperature
contour: sync_to_palette OFF
#
# create 8 contours from 0 to 4.5 using a quadratic distribution
#
contour: distribution quadratic
contour: range 0. 4.5
contour: levels 8
#
# turn on labels spaced every 2 units
# using a floating point representation with 2 decimal places
#
contour: label_visible ON
contour: label_spacing 2.0
contour: label_format floating_point
contour: label_decimal_places 2
contour: end
contour: create
```

See Also:

Associated contour commands

contour: variable

Command:

```
contour: variable <var_name>
```

Purpose:

Use this variable for creating a contour part.

Parameters:

Type	Parameter	Description
string	var_name	create contours from this variable

Notes:

The variable must be active before the contour part can be created.

Example Usage:

```
#
# Create contours of velocity magnitude of part 2
#
part: select_begin
 2
part: select_end
contour: begin
contour: variable velocity
contour: end
contour: create
```

See Also:

Associated contour commands

Commands in this Section:

curve

curve: assign <option> <description>
curve: delete
curve: desc <description>
curve: line_style <option>
curve: line_type <option>
curve: line_width <width>
curve: load <filename>
curve: marker <option>
curve: marker_scale <size>
curve: normalize_x <toggle>
curve: normalize_y <toggle>
curve: rgb <red_val> <grn_val> <blu_val>
curve: save <type> <filename>
curve: scale <x_scale> <y_scale>
curve: select_all
curve: select_begin
curve: select_end
curve: select_default
curve: smooth_subpoints <num>
curve: visible <toggle>

See Also:

User Manual - Section 8.3, Plot Mode
How To Change Plot Attributes

Command:

```
curve: assign <option> <description>
```

Purpose:

Assign a selected curve to the selected plot.

Parameters:

Type	Parameter	Description	
constant	option	rescale	rescale the plot according to the value extents of the curve(s) (default).
		no_rescale	do not rescale the plot according to the values of the curve(s).
string	description	description of the plotter	

Notes:

String description not needed if assigning a new curve to an existing plot.

Example Usage:

```
curve: select_begin
 0
curve: select_end
curve: assign rescale Velocity vs. Distance for part 2
plot: select_begin
 0
plot: select_end
```

See Also:

Associated curve commands

Command:

```
curve: delete
```

Purpose:

Delete the selected curve(s).

Parameters:

none.

Example Usage:

```
curve: select_begin
 0
curve: select_end
curve: delete
```

See Also:

Associated curve commands

curve: desc

Command:

```
curve: desc <description>
```

Purpose:

Specify curve description.

Parameters:

Type	Parameter	Description
string	`description`	description of the selected curve

Example Usage:

```
curve: select_begin
 0
curve: select_end
curve: desc Velocity vs. Distance for part 2
```

See Also:

Associated curve commands

Command:

```
curve: line_style <option>
```

Purpose:

Specifies the line style to use for the selected curves.

Parameters:

Type	Parameter	Description	
constant	option	solid	curve line segments solid
		dotted	curve line segments dotted
		dash	curve line segments dashed

Notes:

Only works when line_type is connect_dots.

Example Usage:

```
curve: select_begin
 1
curve: select_end
curve: line_type connect_dots
curve: line_style dotted
```

See Also:

Associated curve commands

curve: line_type

Command:

```
curve: line_type <option>
```

Purpose:

Specifies the line type to use for the selected curves.

Parameters:

Type	Parameter	Description	
constant	option	none	curve points not connected, represented by marker type selection
		connect_dots	curve points connected with linear line segments (default)
		smooth	curve points connected with smoothed line segments

Example Usage:

```
curve: select_begin
 1
curve: select_end
curve: line_type none
```

See Also:

Associated curve commands

Command:

```
curve: line_width <width>
```

Purpose:

Specify the (pixel) line width of the selected curve(s).

Parameters:

Type	Parameter	Description
int	`width`	pixel line width of 1, 2 (default), 3, or 4.

Example Usage:

```
curve: select_begin
 1
curve: select_end
curve: line_width 3
```

See Also:

Associated curve commands

curve: load

Command:

```
curve: load <filename>
```

Purpose:

Load the specified (saved) curve file.

Parameters:

Type	Parameter	Description
string	`filename`	file (path and) name of the saved curve file

Notes:

See the curve file formats under "`curve: save`".

Example Usage:

```
curve: select_begin
 2
curve: select_end
curve: load Curve0
```

See Also:

Associated curve commands

Command:

```
curve: marker <option>
```

Purpose:

Specify the type of marker in which to represent the points of the selected curve(s).

Parameters:

Type	Parameter	Description	
constant	option	none	curve points represented by no symbol
		dot	curve points represented as dots
		circle	curve points represented as circles
		triangle	curve points represented as triangles
		square	curve points represented as squares

Example Usage:

```
curve: select_begin
  0 1
curve: select_end
curve: marker_type circle
```

See Also:

Associated curve commands

curve: marker_scale

Command:

```
curve: marker_scale <size>
```

Purpose:

Specify the size of the marker type which represents the points of the selected curve(s).

Parameters:

Type	Parameter	Description
float	`size`	size of the curve point markers (default = 1.)

Example Usage:

```
curve: select_begin
 0 1
curve: select_end
curve: marker_scale 2.0000e+00
```

See Also:

Associated curve commands

Command:

```
curve: normalize_x <toggle>
curve: normalize_y <toggle>
```

Purpose:

Specify whether to normalize the abscissa range of the selected curve(s) to the interval I[0.,1.].

Parameters:

Type	Parameter	Description	
constant	toggle	ON	represent the curve abscissa coordinate values in the normalized range I[0.,1.]
		OFF	represent the curve abscissa coordinate values in the actual range of values (default)

Notes:

This means to divide all abscissa coordinates of the curve by the maximum abscissa coordinate value.

Example Usage:

```
curve: select_begin
 0 1
curve: select_end
curve: normalize_x ON
curve: normalize_y OFF
```

See Also:

Associated curve commands

curve: rgb

Command:

```
curve: rgb <red_val> <grn_val> <blu_val>
```

Purpose:

Specify the color of the selected curve(s) via red, green, and blue component values.

Parameters:

Type	Parameter	Description
float	red_val	red, green, blue component of color (0 to 1.)
	grn_val	
	blu_val	

Example Usage:

```
#
# Set the curve color to cyan
#
curve: select_begin
 1
curve: select_end
curve: rgb 0.0000e+00 1.0000e+00 1.0000e+00
```

See Also:

Associated curve commands

Command:

```
curve: save <type> <filename>
```

Purpose:

Specify how to save the selected curve(s) data to a file.

Parameters:

Type	Parameter	Description	
constant	type	formatted	See Notes
		xy_data	See Notes
string	filename	name of the file in which to save the curve points	

Notes:

formatted - curve data saved in the following format:

```
velocity vs. Distance for line tool

  Distance       Value           X             Y             Z
 0.00000e+00   5.56944e-01  -1.84272e+00   5.00623e-01   1.02349e+00
 1.16535e+00   5.06148e-01  -6.77367e-01   5.00623e-01   1.02349e+00
 2.33070e+00   4.33805e-02   4.87984e-01   5.00623e-01   1.02349e+00
 3.49605e+00   3.41879e-01   1.65334e+00   5.00623e-01   1.02349e+00
 4.66140e+00   5.29504e-01   2.81869e+00   5.00623e-01   1.02349e+00

Minimum value overall
At Point =  4.87984e-01  5.00623e-01  1.02349e+00,
Dist. =  2.33070e+00, Value = 4.33805e-02

Maximum value overall
At Point = -1.84272e+00  5.00623e-01  1.02349e+00,
Dist. =  0.00000e+00, Value = 5.56944e-01
```

xy_data - curve data saved in the following (x,y) format for importing into a spreadsheets:

```
velocity vs. Distance for line tool
Distance
velocity
1
5
0 0.556944
1.16535 0.506148
2.3307 0.0433805
3.49605 0.341879
4.6614 0.529504
```

Example Usage:

```
curve: select_begin
 0
curve: select_end
curve: save xy_data Curve0
```

See Also:

Associated curve commands

Command:

```
curve: scale <x_scale> <y_scale>
```

Purpose:

Scales the x and y query by the values given.

Parameters:

Type	Parameter	Description
float	x_scale	query x scale factor
	y_scale	query y scale factor

Notes:

Only scales the curve visually, does not modify the query information.

Example Usage:

```
curve: select_begin
 1
curve: select_end
curve: scale 60.0 1.
```

See Also:

Associated curve commands

Command:

```
curve: select_all
```

Purpose:

Select all curves for subsequent operations.

Parameters:

none.

Example Usage:

```
curve: select_all
```

See Also:

Associated curve commands

curve: select_begin/end

Command:

```
curve: select_begin
curve: select_end
```

Purpose:

Curve(s) selection construct for subsequent curve operation.

Parameters

none

Notes:

Curve id's to be selected must be specified betweeen `curve: select_begin` and `curve: select_end`.

Example Usage:

```
curve: select_begin
 0
curve: select_end
curve: assign rescale Velocity vs. Distance for part 2
plot: select_begin
 0
plot: select_end
```

See Also:

Associated curve commands

Command:

```
curve: select_default
```

Purpose:

Specify the default curve to be selected - or unselect all curves.

Parameters:

none.

Notes:

This is used to set the default (unselected) curve attributes.

Example Usage:

```
curve: select_default
```

See Also:

Associated curve commands

curve: smooth_subpoints

Command:

```
curve: smooth_subpoints <num>
```

Purpose:

Specify the number of smooth subpoints to use between curve points of the selected curve.

Parameters:

Type	Parameter	Description
float	num	number of subpoints in which to display the smoothed line type between curve points (default is 3)

Notes:

The curve line type must be smooth.

Example Usage:

```
curve: select_begin
 1
curve: select_end
curve: line_type smooth
curve: smooth_subpoints 4.0000e+00
```

See Also:

Associated curve commands

Command:

```
curve: visible <toggle>
```

Purpose:

Toggle the selected curve(s) visibility.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	curve visible (default)
		OFF	curve invisible

Example Usage:

```
curve: select_begin
 1
curve: select_end
curve: visible OFF
```

See Also:

Associated curve commands

Commands in this Section:

data

data: binary_files_are <endian_order>
data: boundary <filename>
data: format <format_type>
data: geometry <filename>
data: measured <filename>
data: new_static_done <num>
data: new_static_ready
data: new_time <time>
data: new_transient_time <time>
data: path <data_path>
data: plot3ddimension <dimension>
data: plot3diblank <toggle>
data: plot3dmulti_zone <toggle>
data: plot3dread_as <binary_flag>
data: read
data: result <filename>
data: start_time <step_number>

See Also:

User Manual - Section 2.1, Internal Readers
How To Read Data
How To Display Discrete or Experimental Data

data: binary_files_are

Command:

```
data: binary_files_are <endian_order>
```

Purpose:

Sets the byte (endian) order for binary geometry, result, and measured data files.

Parameters:

Type	Parameter	Description	
constant	endian_order	The endian order for the binary files and must be one of:	
		little_endian	byte order of Intel and Alpha CPU's
		big_endian	byte order of SGI, SUN, HP, IBM
		native	byte order of machine running the EnSight server

Notes:

EnSight assumes that the geometry, result, and measured data files are of the same byte order.

Must be specified before a "data: read" command.

Example Usage:

```
VERSION 7.40
data: binary_files_are big_endian
data: format ensight
data: path /usr/tmp
data: geometry ski_c.geo
data: result ski_c.res
data: read
```

See Also:

Associated data commands

Command:

```
data: boundary <filename>
```

Purpose:

Specify the name for the boundary file. Boundary files provide a way to externally specify the ijk extents of bounding surfaces, etc. for structured data. The resultant is an unstructured part.

Parameters:

Type	Parameter	Description
string	`filename`	boundary file name

Notes:

The filename is concatenated with the path defined via the "`data: path`" command.

Since the boundary file depends on the structured block/zone definitions in the geometry file, the use of a boundary file requires the specification of a geometry file with structured data as well. For EnSight case format, the boundary file is contained within the casefile and this command is not needed. It must be used for other structured data formats - such as PLOT3D.

When specified, unstructured parts can be created according to the boundary file defintions in the unstructured portion of the Data Part Loader dialog.

Example Usage:

Read a structured geometry file and a boundary file.

```
VERSION 7.40
data: binary_files_are_big_endian
data: format plot3d
data: plot3diblank OFF
data: plot3dmulti_zone ON
data: plot3dread_as ascii
data: plot3ddimension 3d
data: path /usr/tmp
data: geometry single.grid
data: boundary single.bnd
data: result single.q
data: read
```

See Also:

Associated data commands

data: format

Command:

```
data: format <format_type>
```

Purpose:

Set the format type for the geometry and result files.

Parameters:

Type	Parameter	Description
constant	format_type	The format type which must be one of the following (or the name of a user defined reader):
		ensight
		plot3d
		fidap
		n3s
		estet
		mpgs4
		movie
		fluent
		ansys
		abaqus
		case
		fast-unstructured

Example Usage:

```
#
# Read a datafile in case format and load a part from it.
#
VERSION 7.40
data: binary_files_are big_endian
data: format case
data: path /usr/tmp
data: geometry cube.case
data: read
data_partbuild: begin
data_partbuild: data_type unstructured
data_partbuild: select_begin
 1
data_partbuild: select_end
data_partbuild: create
data_partbuild: end
```

See Also:

Associated data commands

Command:

```
data: geometry <filename>
```

Purpose:

Set the name for the file containing geometric information.

Parameters:

Type	Parameter	Description
string	`filename`	geometry file name

Notes:

The filename is concatenated with the path defined via the "`data: path`" command.

The meaning of "`geometry`" is different depending on the data format. For example, if the format is `CASE`, the "`geometry`" file is the case file, while if the format is `ANSYS`, the file refers to the ANSYS result file.

Example Usage:

```
VERSION 7.40
data: binary_files_are big_endian
data: format case
data: geometry cube.case
data: read
data_partbuild: begin
data_partbuild: data_type unstructured
data_partbuild: select_begin 1
data_partbuild: select_end
data_partbuild: create
data_partbuild: end
```

See Also:

Associated data commands

data: measured

Command:

```
data: measured <filename>
```

Purpose:

Specify the name for the measured particle file.

Parameters:

Type	Parameter	Description
string	`filename`	measured file name

Notes:

The filename is concatenated with the path defined via the "`data: path`" command.

It is not possible to read a measured file without also specifying a geometry file, or having specified a geometry file previously.

When specified, the part containing the measured particle data will be generated upon a "`data: read`" command, i.e., this part type will not be shown in the various part loaders.

Example Usage:

Read a geometry file and a measured particle file and create a part.

```
VERSION 7.40
data: format ensight
data: path /usr/tmp
data: geometry cube.geom
data: measured cube.mres
data: start_time 1
data: read
data_partbuild: begin
data_partbuild: data_type unstructured
data_partbuild: select_begin
 1
data_partbuild: select_end
data_partbuild: create
data_partbuild: end
```

See Also:

Associated data commands

Command:

```
data: new_static_done <num>
```

Purpose:

Informs the client that the static data files have been modified.

Parameters:

Type	Parameter	Description	
constant	num	0	geometry is not changing
		1	geometry is changing

Example Usage:

Internal command used only by the dynamic data interface.

See Also:

Associated data commands

data: new_static_ready

Command:

```
data: new_static_ready
```

Purpose:

Informs the client that the static data files are about to be modified.

Parameters:

none

Example Usage:

Internal command used only by the dynamic data interface.

See Also:

Associated data commands

Command:

```
data: new_time <time>
```

Purpose:

Informs client that a new transient time step is available

Parameters:

Type	Parameter	Description
float	time	simulation time

Notes:

Each time the command is called, it's value must be increasing.

Example Usage:

Internal command used only by the dynamic data interface.

See Also:

Associated data commands

data: new_transient_time

Command:

```
data: new_transient_time <time>
```

Purpose:

Informs client of last transient time step available

Parameters:

Type	Parameter	Description
float	`time`	simulation time

Notes:

Each time the command is called, it's value must be increasing.

Example Usage:

Internal command used only by the dynamic data interface.

See Also:

Associated data commands

Command:

```
data: path <data_path>
```

Purpose:

Specify the directory under which the data files are located.

Parameters:

Type	Parameter	Description
string	data_path	path to the data files

Notes:

The path can be fully or partially qualified. All (geometry, result, and measured) file names are post concatenated with the path given to arrive at the full file name.

Example Usage:

Read a data file located in /usr/tmp and load a part from it.

```
VERSION 7.40
data: binary_files_are big_endian
data: format case
data: path /usr/tmp
data: geometry cube.case
data: read
data_partbuild: begin
data_partbuild: data_type unstructured
data_partbuild: select_begin
 1
data_partbuild: select_end
data_partbuild: create
data_partbuild: end
```

See Also:

Associated data commands

data: plot3ddimension

Command:

```
data: plot3ddimension <dimension>
```

Purpose:

When the format is plot3d, specifies the dimensionality of the data.

Parameters:

Type	Parameter	Description
constant	`dimension`	The dimensionality of the data and must be one of:
		`1d`
		`2d`
		`3d`

Notes:

EnSight attempts to determine the dimensionality of plot3d data, if this command is not given.

It must be specified before a "`data: read`" command is given.

Example Usage:

```
#
# Read a 3D plot3d file and extract a part.
#
VERSION 7.40
data: format plot3d
data: plot3diblank OFF
data: plot3dmulti_zone OFF
data: plot3dread_as c_binary
data: plot3ddimension 3d
data: path /usr/tmp
data: geometry shuttle.xyz
data: result shuttle.q
data: read
data_partbuild: begin
data_partbuild: data_type structured
data_partbuild: select_begin
 1
data_partbuild: select_end
data_partbuild: domain all
data_partbuild: noderange_i 1 52
data_partbuild: noderange_j 1 63
data_partbuild: noderange_k 1 1
data_partbuild: nodestep 1 1 1
data_partbuild: nodedelta 0 0 0
data_partbuild: create
```

See Also:

Associated data commands

Command:

```
data: plot3diblank <toggle>
```

Purpose:

When the format is plot3d, specifies the iblanking value for the geometry file.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	for iblanking
		OFF	for no Iblanking

Notes:

Must be specified before a "data: read" command is given.

Example Usage:

```
#
# Read a plot3d file with no iblanking and extract a part.
#
VERSION 7.40
data: format plot3d
data: plot3diblank OFF
data: plot3dmulti_zone OFF
data: plot3dread_as c_binary
data: plot3ddimension 3d
data: path /usr/tmp
data: geometry shuttle.xyz
data: result shuttle.q
data: read
data_partbuild: begin
data_partbuild: data_type structured
data_partbuild: select_begin
 1
data_partbuild: select_end
data_partbuild: domain all
data_partbuild: noderange_i 1 52
data_partbuild: noderange_j 1 63
data_partbuild: noderange_k 1 1
data_partbuild: nodestep 1 1 1
data_partbuild: nodedelta 0 0 0
data_partbuild: create
```

See Also:

Associated data commands

data: plot3dmulti_zone

Command:

```
data: plot3dmulti_zone <toggle>
```

Purpose:

When the format is plot3d, specifies that the file is single or multi zone.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	if the file is multi-zone
		`OFF`	if single zone

Notes:

EnSight will attempt to determine if the file is single or multi-zone if this command is not specified.

Must be specified before a "`data: read`" command is given.

Example Usage:

```
#
# Read a single zone plot3d file and extract a part.
#
VERSION 7.40
data: format plot3d
data: plot3diblank OFF
data: plot3dmulti_zone OFF
data: plot3dread_as c_binary
data: plot3ddimension 3d
data: path /usr/tmp
data: geometry shuttle.xyz
data: result shuttle.q
data: read
data_partbuild: begin
data_partbuild: data_type structured
data_partbuild: select_begin
 1
data_partbuild: select_end
data_partbuild: domain all
data_partbuild: noderange_i 1 52
data_partbuild: noderange_j 1 63
data_partbuild: noderange_k 1 1
data_partbuild: nodestep 1 1 1
data_partbuild: nodedelta 0 0 0
data_partbuild: create
```

See Also:

Associated data commands

Command:

```
data: plot3dread_as <binary_flag>
```

Purpose:

When the format is plot3d, specifies how the file is formatted.

Parameters:

Type	Parameter	Description	
constant	binary_flag	The format flag which must be on of:	
		ascii	the file is ASCII
		c_binary	the file was written in C binary
		fortran_binary	the file was written in Fortran binary

Notes:

If the user does not specify, EnSight will attempt to determine the binary nature of the data files.

It must be specified before a "data: read" command is given.

All files (geometry, result, and measured) must be of the same binary nature.

Example Usage:

```
#
# Read a c binary plot3d file and extract a part.
#
VERSION 7.40
data: format plot3d
data: plot3diblank OFF
data: plot3dmulti_zone OFF
data: plot3dread_as c_binary
data: plot3ddimension 3d
data: path /usr/tmp
data: geometry shuttle.xyz
data: result shuttle.q
data: read
data_partbuild: begin
data_partbuild: data_type structured
data_partbuild: select_begin
 1
data_partbuild: select_end
data_partbuild: domain all
data_partbuild: noderange_i 1 52
data_partbuild: noderange_j 1 63
data_partbuild: noderange_k 1 1
data_partbuild: nodestep 1 1 1
data_partbuild: nodedelta 0 0 0
data_partbuild: create
```

See Also:

Associated data commands

data: read

Command:

```
data: read
```

Purpose:

Read the `geometry`, `result`, and/or `measured` data files.

Parameters:

none

Notes:

The `geometry` and/or `result` and/or `measured` data file names must be set before making this call.

Example Usage:

```
#
# Read a geometry and associated result file and create a part.
#
VERSION 7.40
data: format ensight
data: path /usr/tmp
data: geometry cube.geom
data: result cube.res
data: read
data_partbuild: begin
data_partbuild: data_type unstructured
data_partbuild: select_begin
 1
data_partbuild: select_end
data_partbuild: create
data_partbuild: end
```

See Also:

Associated data commands

Command:

```
data: result <filename>
```

Purpose:

Set the name for the file containing result information.

Parameters:

Type	Parameter	Description
string	`filename`	result file name

Notes:

The usage of a result file depends on the data format. For some formats the result file is not necessary, while for others the result file either directly contains result information or may be another file containing information pointing to the various files that may exist.

The filename is concatenated with the path defined via the "`data: path`" command.

It is not possible to read a result file without also specifying a geometry file, or having specified a geometry file previously.

Example Usage:

```
#
# Read a geometry and result file,
# create a part,
# rotate the scene,
# and color the part by a variable
#
VERSION 7.40
data: format ensight
data: path /usr/tmp
data: geometry cube.geom
data: result cube.res
data: read
data_partbuild: begin
data_partbuild: data_type unstructured
data_partbuild: select_begin
 1
data_partbuild: select_end
data_partbuild: create
data_partbuild: end
view_transf: rotate -91.339279 2.611849 0.000000
view_transf: rotate 18.749998 42.660221 0.000000
variables: activate temperature
part: select_all
part: modify_begin
part: colorby_palette temperature
part: modify_end
```

See Also:

Associated data commands

data: start_time

Command:

```
data: start_time <step_number>
```

Purpose:

When reading transient data load the timestep indicated as the initial time step.

Parameters:

Type	Parameter	Description
int	`step_number`	timestep to use for the initial data load. Values range from 1 (the first timestep) to N where N is the total number of timesteps supplied.

Notes:

It is not possible to specify a fractional timestep.

The value specified is a step number, not a time value.

Example Usage:

```
#
# Read a geometry file and a measured particle file
# at the first timestep, then create a part.
#
VERSION 7.40
data: format ensight
data: path /usr/tmp
data: geometry cube.geom
data: measured cube.mres
data: start_time 1
data: read
data_partbuild: begin
data_partbuild: data_type unstructured
data_partbuild: select_begin
 1
data_partbuild: select_end
data_partbuild: create
data_partbuild: end
```

See Also:

Associated data commands

Commands in this Section:

data_partbuild

See Also:

User Manual - Section 2.1, Internal Readers

How To Read EnSight Gold Data

How To Read EnSight 5 Data

How To Read EnSight 6 Data

How To Read ABAQUS Data

How To Read ANSYS Data

How To Read ESTET Data

How To Read FAST Unstructured Data

How To Read PLOT3D Data

How To Read FIDAP NEUTRAL Data

How To Read Fluent Universal Data

Command:

```
data_partbuild: begin
<commands>
data_partbuild: end
```

Purpose:

Delineates commands issued while using the Data Part Builder dialogs.

Parameters:

Type	Parameter	Description
string	commands	valid command language for part building (can be one or many)

Notes:

These commands exist for memory management reasons. While doing part building it is often necessary to have information in memory that will only be used during the part building process. Thus, the begin indicates the need for this memory, and the end indicates that such memory could be freed for other purposes. All data formats that use the Data Part Builder method use this command (n3s, estet, mpgs4, plot3d, userd, ensight formats, etc).

Example Usage:

```
#
# Starting part building for N3S parts
#
data_partbuild: begin
#
# Building a part entitled "part_all"
#
data_partbuild: description part_all
data_partbuild: n3sall_elements
data_partbuild: create
#
# Building a part entitled "part_color"
#
data_partbuild: description part_color
data_partbuild: n3scolor_number 7
data_partbuild: create
#
# Building a part entitled "part_color"
#
data_partbuild: description part_bc
data_partbuild: n3sboundary
data_partbuild: n3scondition VITESSE_UX
data_partbuild: n3scode Free Node
data_partbuild: create
#
# Stopping part building for N3S parts
#
data_partbuild: end
```

See Also:

Associated data_partbuild commands

data_partbuild: create

Command:

```
data_partbuild: create
```

Purpose:

Causes the part to be built according to previously defined attributes or defaults.

Parameters:

none

Notes:

All data formats that use the Data Part Builder method use this command (n3s, estet, mpgs4, plot3d, userd, ensight formats, etc).

Example Usage:

```
#
# Starting part building for N3S parts
#
data_partbuild: begin
#
# Building a part entitled "part_all"
#
data_partbuild: description part_all
data_partbuild: n3sall_elements
data_partbuild: create
#
# Building a part entitled "part_color"
#
data_partbuild: description part_color
data_partbuild: n3scolor_number 7
data_partbuild: create
#
# Building a part entitled "part_color"
#
data_partbuild: description part_bc
data_partbuild: n3sboundary
data_partbuild: n3scondition VITESSE_UX
data_partbuild: n3scode Free Node
data_partbuild: create
#
# Stopping part building for N3S parts
#
data_partbuild: end
```

See Also:

Associated data_partbuild commands

Command:

```
data_partbuild: create_iblank
```

Purpose:

Causes unstructured part(s) to be built from structured part(s) according to iblanking values available.

Parameters:

none

Notes:

All data formats that have structured iblanking use this command (estet, plot3d, EnSight6, EnSight Gold, some userd formats, etc). It is required that a structured part be selected as parent before this command.

Example Usage:

```
#
# First we create a structured estet part entitled "field"
#
data_partbuild: begin
data_partbuild: estetdomain inside
data_partbuild: description field
data_partbuild: estetnoderange_i 1 39
data_partbuild: estetnoderange_j 1 15
data_partbuild: estetnoderange_k 1 28
data_partbuild: estetstep 1 1 1
data_partbuild: estetdelta 0 0 0
data_partbuild: create
#
# Then we select this as the parent for our ibanked part
#
data_partbuild: select_begin
1
data_partbuild: select_end
#
# Here we set the description desired and the ibanking type
#
data_partbuild: description symm
data_partbuild: iblank_type Symmetry (4)
#
# And create it
#
data_partbuild: create_iblank
data_partbuild: end
```

See Also:

Associated data_partbuild commands

data_partbuild: data_type

Command:

```
data_partbuild: data_type <type>
```

Purpose:

For those formats that contain both structured and unstructured data (EnSight6, EnSight Gold, and some user defined formats), this command indicates which type to create.

Parameters:

Type	Parameter	Description	
constant	type	unstructured	to create unstructured parts
		structured	to create structured parts

Example Usage:

```
data_partbuild: begin
#
# Doing an unstructured ensight part
#
data_partbuild: data_type unstructured
#
# Selecting the first part in the
# unstructured part building list
#
data_partbuild: select_begin
1
data_partbuild: select_end
#
# Naming it "my_unstructured_part"
# and creating it
#
data_partbuild: description my_unstructured_part
data_partbuild: create
#
# Now doing a structured ensight part
#
data_partbuild: data_type structured
#
# Selecting the first part in the
# structured part building list
# (Note this is not the same part as the
# unstructured one selected above)
#
data_partbuild: select_begin
1
data_partbuild: select_end
#
# Setting ijk ranges, step_by, delta values, etc.
# and creating it.
#
data_partbuild: domain all
data_partbuild: noderange_i 1 3
data_partbuild: noderange_j 1 3
data_partbuild: noderange_k 1 3
data_partbuild: nodestep 1 1 1
data_partbuild: nodedelta 0 0 0
data_partbuild: description my_structured_part
data_partbuild: create
```

```
data_partbuild: end
```

See Also:

Associated data_partbuild commands

data_partbuild: description

Command:

```
data_partbuild: description <desc>
```

Purpose:

Specifies the textual description for a part.

Parameters:

Type	Parameter	Description
string	desc	description for the part (it is what is displayed in the user interface)

Example Usage:

```
data_partbuild: begin
#
# Doing an unstructured ensight part
#
data_partbuild: data_type unstructured
#
# Selecting the first part in the
# unstructured part building list
#
data_partbuild: select_begin
1
data_partbuild: select_end
#
# Naming it "my_unstructured_part"
# and creating it
#
data_partbuild: description my_unstructured_part
data_partbuild: create
#
# Now doing a structured ensight part
#
data_partbuild: data_type structured
#
# Selecting the first part in the
# structured part building list
# (Note this is not the same part as the
# unstructured one selected above)
#
data_partbuild: select_begin
1
data_partbuild: select_end
#
# Setting ijk ranges, step_by, delta values, etc.
# and creating it.
#
data_partbuild: domain all
data_partbuild: noderange_i 1 3
data_partbuild: noderange_j 1 3
data_partbuild: noderange_k 1 3
data_partbuild: nodestep 1 1 1
data_partbuild: nodedelta 0 0 0
data_partbuild: description my_structured_part
data_partbuild: create
data_partbuild: end
```

See Also:

Associated data_partbuild commands

data_partbuild: domain

Command:

```
data_partbuild: domain <option>
```

Purpose:

Specify the iblanking domain to use for EnSight6, EnSight Gold and User-defined models that have structured iblanked data.

Parameters:

Type	Parameter	Description	
constant	option	inside	for domains that have iblank value of 1
		outside	for domains that have iblank value of 0
		all	for all domains regardless of iblank value. (Basically ignores iblanking)

Example Usage:

```
data_partbuild: begin
#
# Now doing a structured ensight part
#
data_partbuild: data_type structured
#
# Selecting the first part in the
# structured part building list
#
data_partbuild: select_begin
1
data_partbuild: select_end
#
# Setting ijk ranges, step_by, delta values, etc.
# and creating it. Note the domain used.
#
data_partbuild: domain inside
data_partbuild: noderange_i 1 3
data_partbuild: noderange_j 1 3
data_partbuild: noderange_k 1 3
data_partbuild: nodestep 1 1 1
data_partbuild: nodedelta 0 0 0
data_partbuild: description my_inside_part
data_partbuild: create
data_partbuild: end
```

See Also:

Associated data_partbuild commands

Command:

```
data_partbuild: estet_domain <option>
```

Purpose:

Specify the iblanking domain to use for estet models.

Parameters:

Type	Parameter	Description	
constant	option	inside	for domains that have iblank value of INSIDE
		outside	for domains that have iblank value of OUTSIDE
		all	for all domains regardless of iblank value. (Basically ignores iblanking)

Example Usage:

```
data_partbuild: begin
data_partbuild: estetdomain inside
data_partbuild: description my_inside_part
data_partbuild: estetnoderange_i 1 39
data_partbuild: estetnoderange_j 1 15
data_partbuild: estetnoderange_k 1 28
data_partbuild: estetstep 1 1 1
data_partbuild: estetdelta 0 0 0
data_partbuild: create
data_partbuild: end
```

See Also:

Associated data_partbuild commands

data_partbuild: estetbuildvector

Command:

```
data_partbuild: estetbuildvector
```

Purpose:

Combines estet vector components (given as scalars) into a vector variable.

Parameters:

none

Notes:

Requires the use of "`data_partbuild: estetvectordesc`" and "`data_partbuild: estetvector_x_*`" commands before being issued.

Example Usage:

```
data_partbuild: estetvectordesc velocity
data_partbuild: estetvector_x VITESSE_U
data_partbuild: estetvector_y VITESSE_V
data_partbuild: estetvector_z VITESSE_W
data_partbuild: estetbuildvector
```

See Also:

Associated data_partbuild commands

Command:

```
data_partbuild: estetdelta <i_del> <j_del> <k_del>
```

Purpose:

Specify ijk delta value to use when creating more than one ijk surface from the same ijk block. Only one of the deltas may be non-zero.

Parameters:

Type	Parameter	Description
int	i_del	I direction delta value
	j_del	J direction delta value
	k_del	K direction delta value

Example Usage:

```
#
# This will create a slice at i = 1, i = 6, i = 11, ...
# of the inside domain.
#
data_partbuild: begin
data_partbuild: estetdomain inside
data_partbuild: description multiple_iplane_slice_part
data_partbuild: estetnoderange_i 1 1
data_partbuild: estetnoderange_j 1 15
data_partbuild: estetnoderange_k 1 28
data_partbuild: estetstep 1 1 1
data_partbuild: estetdelta 5 0 0
data_partbuild: create
data_partbuild: end
```

See Also:

Associated data_partbuild commands

data_partbuild: estetnoderange_i/j/k

Command:

```
data_partbuild: estetnoderange_i <beg> <end>
data_partbuild: estetnoderange_j <beg> <end>
data_partbuild: estetnoderange_k <beg> <end>
```

Purpose:

Specify i, j, or k range values to use when creating a part.

Parameters:

Type	Parameter	Description
int	beg	I, J, or K range beginning node plane
	end	I, J, or K range ending node plane

Example Usage:

```
#
# This will create a slice at i = 1, i = 6, i = 11, ...
# of the inside domain.
# (Note the ranges for each of the ijk directions)
#
data_partbuild: begin
data_partbuild: estetdomain inside
data_partbuild: description multiple_iplane_slice_part
data_partbuild: estetnoderange_i 1 1
data_partbuild: estetnoderange_j 1 15
data_partbuild: estetnoderange_k 1 28
data_partbuild: estetstep 1 1 1
data_partbuild: estetdelta 5 0 0
data_partbuild: create
data_partbuild: end
```

See Also:

Associated data_partbuild commands

Command:

```
data_partbuild: estetstep <i_step> <j_step> <k_step>
```

Purpose:

Specify i, j, k step-by values to use in creating an estet part. Values greater than 1 will cause original cells to be combined appropriately into larger elements - producing a coarser grid.

Parameters:

Type	Parameter	Description
int	i_step	step-by value for the I direction
	j_step	step-by value for the J direction
	k_step	step-by value for the K direction

Notes:

This is used when a coarser approximation of the original data is desired. Typically this reduction in the apparent resolution of the model is desired in order to be able to reasonably display very large models.

Example Usage:

```
#
# This example causes the i and j directions to have cells
# that are made up of 4 original cells each. However, in
# the k direction, the original thickness is retained.
#
data_partbuild: begin
data_partbuild: estetdomain inside
data_partbuild: description coarse_in_ij_part
data_partbuild: estetnoderange_i 1 39
data_partbuild: estetnoderange_j 1 15
data_partbuild: estetnoderange_k 1 28
data_partbuild: estetstep 2 2 1
data_partbuild: estetdelta 0 0 0
data_partbuild: create
data_partbuild: end
```

See Also:

Associated data_partbuild commands

data_partbuild: estetvector_x/y/z

Command:

```
data_partbuild: estetvector_x <scalar_name>
data_partbuild: estetvector_y <scalar_name>
data_partbuild: estetvector_z <scalar_name>
```

Purpose:

Specifies the name of the estet scalar variable to use for the vector variable component when build vector variables.

Parameters:

Type	Parameter	Description
string	`scalar_name`	name of the estet variable that represents a vector component

Example Usage:

```
data_partbuild: estetvectordesc velocity
data_partbuild: estetvector_x VITESSE_U
data_partbuild: estetvector_y VITESSE_V
data_partbuild: estetvector_z VITESSE_W
data_partbuild: estetbuildvector
```

See Also:

Associated data_partbuild commands

Command:

```
data_partbuild: estetvectordesc <desc>
```

Purpose:

Specifies the name for the vector variable will be created with a "`data_partbuild: estet-buildvector`" command.

Parameters:

Type	Parameter	Description
string	`desc`	desired name for the vector variable

Example Usage:

```
data_partbuild: estetvectordesc velocity
data_partbuild: estetvector_x VITESSE_U
data_partbuild: estetvector_y VITESSE_V
data_partbuild: estetvector_z VITESSE_W
data_partbuild: estetbuildvector
```

See Also:

Associated data_partbuild commands

data_partbuild: iblank_type

Command:

```
data_partbuild: iblank_type <desc>
```

Purpose:

Selects the iblanking type to use when creating unstructured parts from the iblanking of structured parts.

Parameters:

Type	Parameter	Description
string	`desc`	description for the iblanking type that is provided to the part building process. (varies for each data format)

Notes:

All data formats that have structured iblanking use this command (estet, plot3d, EnSight6, EnSight Gold, some userd formats, etc). This process requires that a structured part be selected as parent.

Example Usage:

```
#
# First we create a structured estet part entitled "field"
#
data_partbuild: begin
data_partbuild: estetdomain inside
data_partbuild: description field
data_partbuild: estetnoderange_i 1 39
data_partbuild: estetnoderange_j 1 15
data_partbuild: estetnoderange_k 1 28
data_partbuild: estetstep 1 1 1
data_partbuild: estetdelta 0 0 0
data_partbuild: create
#
# Then we select this as the parent for our ibanked part
#
data_partbuild: select_begin
1
data_partbuild: select_end
#
# Here we set the description desired and the ibanking type
#
data_partbuild: description symm
data_partbuild: iblank_type Symmetry (4)
#
# And create it
#
data_partbuild: create_iblank
data_partbuild: end
```

See Also:

Associated data_partbuild commands

Command:

```
data_partbuild: mpgsload <option>
```

Purpose:

Specifies the option to use when reading an mpgs data file. All parts will be read. This option controls which parts will be loaded to the client..

Parameters:

Type	Parameter	Description	
constant	option	all_parts	all parts will be downloaded
		part1_ony	part one will be downloaded
		all_but_part1	all but part 1 will be downloaded

Example Usage:

```
#
# This one is pretty easy, after you specified the data,
# issue the following (or one of the other options)
#
data_partbuild: mpgsload all_parts
```

See Also:

Associated data_partbuild commands

data_partbuild: n3sall_elements

Command:

```
data_partbuild: n3sall_elements
```

Purpose:

Specifies the "`All Elements`" process when creating an n3s part.

Parameters:

none

Notes:

This process uses all elements, regardless of color or boundary condition tags applied.

Example Usage:

```
data_partbuild: begin
#
# Creating a part composed of all elements
#
data_partbuild: description part_all
data_partbuild: n3sall_elements
data_partbuild: create
#
# Creating a part based on color number 7
#
data_partbuild: description part_color
data_partbuild: n3scolor_number 7
data_partbuild: create
#
# Creating a part based on boundary condition VITESSE_UX
#
data_partbuild: description part_bc
data_partbuild: n3sboundary
data_partbuild: n3scondition VITESSE_UX
data_partbuild: n3scode Free Node
data_partbuild: create
data_partbuild: end
```

See Also:

Associated data_partbuild commands

Command:

```
data_partbuild: n3sboundary
```

Purpose:

Specifies the "`boundary condition`" process when creating an n3s part.

Parameters:

none

Notes:

Needs the "`data_partbuild: n3scondition`" and "`data_partbuild: n3scode`" commands as well.

Example Usage:

```
data_partbuild: begin
#
# Creating a part composed of all elements
#
data_partbuild: description part_all
data_partbuild: n3sall_elements
data_partbuild: create
#
# Creating a part based on color number 7
#
data_partbuild: description part_color
data_partbuild: n3scolor_number 7
data_partbuild: create
#
# Creating a part based on boundary condition VITESSE_UX
#
data_partbuild: description part_bc
data_partbuild: n3sboundary
data_partbuild: n3scondition VITESSE_UX
data_partbuild: n3scode Free Node
data_partbuild: create
data_partbuild: end
```

See Also:

Associated data_partbuild commands

data_partbuild: n3scode

Command:

```
data_partbuild: n3scode <desc>
```

Purpose:

Specifies the boundary condition code to use in the "`boundary condition`" process when creating an N3S part.

Parameters:

Type	Parameter	Description

string	`desc`	The n3s boundary condition code. (the available codes can vary per model) They are typically things like:
		`Free Node`
		`Homogeneous Dirichlet`
		`Non-homogeneous Dirichlet`

Example Usage:

```
data_partbuild: begin
#
# Creating a part composed of all elements
#
data_partbuild: description part_all
data_partbuild: n3sall_elements
data_partbuild: create
#
# Creating a part based on color number 7
#
data_partbuild: description part_color
data_partbuild: n3scolor_number 7
data_partbuild: create
#
# Creating a part based on boundary condition VITESSE_UX
#
data_partbuild: description part_bc
data_partbuild: n3sboundary
data_partbuild: n3scondition VITESSE_UX
data_partbuild: n3scode Free Node
data_partbuild: create
data_partbuild: end
```

See Also:

Associated data_partbuild commands

Command:

```
data_partbuild: n3scolor_number <num>
```

Purpose:

Specifies the "`color number`" process when creating an N3S part.

Parameters:

Type	Parameter	Description
int	`num`	node color number to use when creating an N3S part

Example Usage:

```
data_partbuild: begin
#
# Creating a part composed of all elements
#
data_partbuild: description part_all
data_partbuild: n3sall_elements
data_partbuild: create
#
# Creating a part based on color number 7
#
data_partbuild: description part_color
data_partbuild: n3scolor_number 7
data_partbuild: create
#
# Creating a part based on boundary condition VITESSE_UX
#
data_partbuild: description part_bc
data_partbuild: n3sboundary
data_partbuild: n3scondition VITESSE_UX
data_partbuild: n3scode Free Node
data_partbuild: create
data_partbuild: end
```

See Also:

Associated data_partbuild commands

data_partbuild: n3scondition

Command:

```
data_partbuild: n3scondition <desc>
```

Purpose:

Specifies the boundary condition to use in the "`boundary condition`" process when creating an N3S part.

Parameters:

Type	Parameter	Description
string	`desc`	N3S boundary condition. (available conditions can vary per model)

Example Usage:

```
data_partbuild: begin
#
# Creating a part composed of all elements
#
data_partbuild: description part_all
data_partbuild: n3sall_elements
data_partbuild: create
#
# Creating a part based on color number 7
#
data_partbuild: description part_color
data_partbuild: n3scolor_number 7
data_partbuild: create
#
# Creating a part based on boundary condition VITESSE_UX
#
data_partbuild: description part_bc
data_partbuild: n3sboundary
data_partbuild: n3scondition VITESSE_UX
data_partbuild: n3scode Free Node
data_partbuild: create
data_partbuild: end
```

See Also:

Associated data_partbuild commands

Command:

```
data_partbuild: nodedelta <i_del> <j_del> <k_del>
```

Purpose:

Specify ijk delta value to use when creating more than one ijk surface from the same ijk block. Only one of the deltas may be non-zero.

Parameters:

Type	Parameter	Description
int	i_del	I, J, K direction delta values
	j_del	
	k_del	

Notes:

This command is used for EnSight6, EnSight Gold, and User-Defined data formats.

Example Usage:

```
data_partbuild: begin
#
# Now doing a structured ensight part
#
data_partbuild: data_type structured
#
# Selecting the first part in the
# structured part building list
#
data_partbuild: select_begin
1
data_partbuild: select_end
#
# Setting ijk ranges, step_by, delta values, etc.
# and creating it. Note the nodedelta and noderange
# combination used
#
data_partbuild: domain inside
data_partbuild: noderange_i 1 30
data_partbuild: noderange_j 1 1
data_partbuild: noderange_k 1 30
data_partbuild: nodestep 1 1 1
data_partbuild: nodedelta 0 3 0
data_partbuild: description multi_jslice_part
data_partbuild: create
data_partbuild: end
```

See Also:

Associated data_partbuild commands

data_partbuild: noderange_i/j/k

Command:

```
data_partbuild: noderange_i <beg> <end>
data_partbuild: noderange_j <beg> <end>
data_partbuild: noderange_k <beg> <end>
```

Purpose:

Specify i, j, or k range values to use when creating a part.

Parameters:

Type	Parameter	Description
int	beg	I, J, or K range beginning node plane
	end	I, J, or K range ending node plane

Notes:

This command is used for EnSight6, EnSight Gold, and User-Defined data formats.

Example Usage:

```
data_partbuild: begin
#
# Now doing a structured ensight part
#
data_partbuild: data_type structured
#
# Selecting the first part in the
# structured part building list
#
data_partbuild: select_begin
1
data_partbuild: select_end
#
# Setting ijk ranges, step_by, delta values, etc.
# and creating it. Note the nodedelta and noderange
# combination used
#
data_partbuild: domain inside
data_partbuild: noderange_i 1 30
data_partbuild: noderange_j 1 1
data_partbuild: noderange_k 1 30
data_partbuild: nodestep 1 1 1
data_partbuild: nodedelta 0 3 0
data_partbuild: description multi_jslice_part
data_partbuild: create
data_partbuild: end
```

See Also:

Associated data_partbuild commands

Command:

```
data_partbuild: nodestep <i_step> <j_step> <k_step>
```

Purpose:

Specify i, j, k step-by values to use in creating a structured part. Values greater than 1 will cause original cells to be combined appropriately into larger elements - producing a coarser grid.

Parameters:

Type	Parameter	Description
int	i_step	step-by value for the I, J, K directions
	j_step	
	k_step	

Notes:

This command is used for EnSight6, EnSight Gold, and User-Defined data formats. This is used when a coarser approximation of the original data is desired. Typically this reduction in the apparent resolution of the model is desired in order to be able to reasonably display very large models.

Example Usage:

```
#
# This example causes the i and j directions to have cells
# that are made up of 4 original cells each. However, in
# the k direction, the original thickness is retained.
#
data_partbuild: begin
data_partbuild: domain all
data_partbuild: description coarse_in_ij_part
data_partbuild: noderange_i 1 39
data_partbuild: noderange_j 1 15
data_partbuild: noderange_k 1 28
data_partbuild: nodestep 2 2 1
data_partbuild: nodedelta 0 0 0
data_partbuild: create
data_partbuild: end
```

See Also:

Associated data_partbuild commands

data_partbuild: select_begin/end

Command:

```
data_partbuild: select_begin
<part1, part2, ... partN>
data_partbuild: select_end
```

Purpose:

Specify parts, from the part build lists, to use in the part build process.

Parameters:

Type	Parameter	Description
int	partN	number of the part in the list of available parts in the model data file(s)

Notes:

These commands are used by all iblanked structured formats when doing create_iblank process. They are also used by EnSight6, EnSight Gold, and User-Defined formats when selecting parent parts.

Example Usage:

```
data_partbuild: begin
#
# Doing a structured ensight part
#
data_partbuild: data_type structured
#
# Selecting the first part in the
# structured part building list
#
data_partbuild: select_begin
1
data_partbuild: select_end
#
# Setting ijk ranges, step_by, delta values, etc.
# and creating it.
#
data_partbuild: domain inside
data_partbuild: noderange_i 1 3
data_partbuild: noderange_j 1 3
data_partbuild: noderange_k 1 3
data_partbuild: nodestep 1 1 1
data_partbuild: nodedelta 0 0 0
data_partbuild: description my_inside_part
data_partbuild: create
data_partbuild: end
```

See Also:

Associated data_partbuild commands

Commands in this Section:

devsrf

devsrf: axis <dx> <dy> <dz>
devsrf: begin
devsrf: end
devsrf: create
devsrf: origin <x_comp> <y_comp> <z_comp>
devsrf: projection <type>
devsrf: scale_factor_u <factor>
devsrf: scale_factor_v <factor>
devsrf: seam_vector <x_comp> <y_comp> <z_comp>
devsrf: select_default

See Also:

User Manual - Section 7.9, Developed Surface Create/Update
How To Create Developed (Unrolled) Surfaces

devsrf: axis

Command:

```
devsrf: axis <dx> <dy> <dz>
```

Purpose:

Specifies the direction of the axis of revolution, or longitudinal axis, of the revolved part.

Parameters:

Type	Parameter	Description
float	dx	cartesian vector x, y, z components of the developing cylinder's longitudinal axis
	dy	
	dz	

Notes:

This vector corresponds to the direction of the axis of revolution of the cylinder, cone, sphere, or revolution tool for any revolved surface created via these quadric tools. Otherwise, this vector should be specified to coincide with the axis of revolution of the 2D parent part.

This is the longitudinal axis about which the nodes of the selected part(s) are mapped to the computational developing cylinder; which cylinder is subsequently unrolled yielding the developed (unrolled) surface.

Example Usage:

```
devsrf: select_default
part: modify_begin
devsrf: origin 5.0000e-01 5.0000e-01 0.0000e+00
devsrf: axis 0.0000e+00 0.0000e+00 1.0000e+00
devsrf: seam_vector 1.0000e+00 0.0000e+00 0.0000e+00
part: modify_end
```

See Also:

Associated devsrf commands

Command:

```
devsrf: begin
<command list>
devsrf: end
```

Purpose:

A wrapper command construct that contains the appropriate developed surface attribute commands used to create a (2D) developed surface part.

Parameters:

Type	Parameter	Description
command	`command list`	appropriate devsrf commands that affect the creation of the developed surface part - see Example Usage below

Example Usage:

```
part: select_begin
 2
part: select_end
devsrf: begin
devsrf: origin 5.0000e-01 5.0000e-01 0.0000e+00
devsrf: axis 0.0000e+00 0.0000e+00 1.0000e+00
devsrf: seam_vector 1.0000e+00 0.0000e+00 0.0000e+00
devsrf: end
devsrf: create
```

See Also:

Associated devsrf commands

devsrf: create

Command:

```
devsrf: create
```

Purpose:

Specifies the creation of a developed, or unrolled surface. This unrolls the selected 2D (parent) part into a flat2D developed part.

Parameters:

none

Notes:

In the example below, the 2 in the 2D parent part 2 is unrolled into the 2D developed part 3.

Example Usage:

```
#
# Align default settings according to parent part
#
part: select_begin
 2
part: select_end
devsrf: select_default
part: modify_begin
devsrf: origin 5.0000e-01 5.0000e-01 0.0000e+00
devsrf: axis 0.0000e+00 0.0000e+00 1.0000e+00
devsrf: seam_vector 1.0000e+00 0.0000e+00 0.0000e+00
part: modify_end
#
# Create developed part 3 from parent part 2
#
part: select_begin
 2
part: select_end
devsrf: begin
devsrf: origin 5.0000e-01 5.0000e-01 0.0000e+00
devsrf: axis 0.0000e+00 0.0000e+00 1.0000e+00
devsrf: seam_vector 1.0000e+00 0.0000e+00 0.0000e+00
devsrf: end
devsrf: create
```

See Also:

Associated devsrf commands

Command:

```
devsrf: origin <x_comp> <y_comp> <z_comp>
```

Purpose:

Specifies an origin which lies on the axis of revolution, or the longitudinal axis of the developing cylinder.

Parameters:

Type	Parameter	Description
float	x_comp	x, y, z components of the origin of the developing cylinder (located on the longitudinal axis) in model coordinates
	y_comp	
	z_comp	

Notes:

This point corresponds to the origin of the cylinder, cone, sphere, or revolution tool for any revolved surface created via these quadric tools. Otherwise, this point locates the axis of revolution of the 2D parent part.

Example Usage:

```
#
# Modify the default
#
devsrf: select_default
part: modify_begin
devsrf: origin 5.0000e-01 5.0000e-01 0.0000e+00
devsrf: axis 0.0000e+00 0.0000e+00 1.0000e+00
devsrf: seam_vector 1.0000e+00 0.0000e+00 0.0000e+00
part: modify_end
```

See Also:

Associated devsrf commands

devsrf: projection

Command:

```
devsrf: projection <type>
```

Purpose:

Specifies the type of coordinate projection, or mapping, that the developed surface part represents.

Parameters:

Type	Parameter	Description	
constant	`type`	Includes the following types of coordinate projections, or mapping, in (abscissa, ordinate) frame of reference:	
		`r,z`	a (radial, z-direction) coordinate system
		`theta,z`	a (theta, z-direction) coordinate system
		`m,theta`	a (meridian, theta) coordinate system
		`m,r`	a (meridian, radial) coordinate system
		`theta,r`	a (theta, radial) coordinate system

Notes:

The meridianal component is defined as m = SQRT(r^2 + z^2).

The default projection is (theta,z).

Example Usage:

```
part: select_begin
 3
part: select_end
part: modify_begin
devsrf: projection m,theta
part: modify_end
viewport: select_begin
 1
viewport: select_end
```

See Also:

Associated devsrf commands

Command:

```
devsrf: scale_factor_u <factor>
devsrf: scale_factor_v <factor>
```

Purpose:

Specifies the scale factor applied to the abscissa/ordinate component of the projection type of the developed part.

Parameters:

Type	Parameter	Description
float	`factor`	scale factor applied to the abscissa/ordinate of the developed surface projection

Notes:

Default scale factor for both u and v is 1.

Example Usage:

```
part: select_begin
 1
part: select_end
part: modify_begin
devsrf: scale_factor_u 5.0000e-01
devsrf: scale_factor_v 2.0000e+00
part: modify_end
```

See Also:

Associated devsrf commands

devsrf: seam_vector

Command:

```
devsrf: seam_vector <x_comp> <y_comp> <z_comp>
```

Purpose:

Specifies the vector perpendicular to the axis of revolution which points in the direction of the seam line.

Parameters:

Type	Parameter	Description
float	x_comp	cartesian x, y, z vector components of a vector perpendicular to and starting from the axis of revolution and pointing to the seam line
	y_comp	
	z_comp	

Notes:

This vector does not specify the actual direction of the seam line by which the surface is split. Rather it indirectly specifies the location of the seam line via defining a vector perpendicular to and emanating from the axis of revolution which points to the seam (and thus, is consequently perpendicular to the seam).

Example Usage:

```
#
# Modify the default
#
devsrf: select_default
part: modify_begin
devsrf: origin 5.0000e-01 5.0000e-01 0.0000e+00
devsrf: axis 0.0000e+00 0.0000e+00 1.0000e+00
devsrf: seam_vector 1.0000e+00 0.0000e+00 0.0000e+00
part: modify_end
```

See Also:

Associated devsrf commands

Command:

```
devsrf: select_default
```

Purpose:

Specifies the default origin, axis (of rotation), and seam vector to align according to the longitudinal axis of the selected part.

Parameters:

none

Notes:

This command facilitates a method for the origin, axis, and seam vector to inherit the position and direction of the longitudinal axis of the 2D parent part.

Example Usage:

```
devsrf: select_default
part: modify_begin
devsrf: origin 5.0000e-01 5.0000e-01 0.0000e+00
devsrf: axis 0.0000e+00 0.0000e+00 1.0000e+00
devsrf: seam_vector 1.0000e+00 0.0000e+00 0.0000e+00
part: modify_end
```

See Also:

Associated devsrf commands

Commands in this Section:

elevsurf

elevsurf: begin
elevsurf: end
elevsurf: component <x_scal> <y_scal> <z_scal>
elevsurf: create
elevsurf: offset <offset_val>
elevsurf: scale_factor <scale>
elevsurf: select_default
elevsurf: sidewall <toggle>
elevsurf: surface <toggle>
elevsurf: variable <variable_name>

See Also:

User Manual - Section 7.7, Elevated Surface Create/Update
How To Create Elevated Surfaces

elevsurf: begin/end

Command:

```
elevsurf: begin
elevsurf: end
```

Purpose:

Delimit the modifications for elevated surface. parts.

Parameters:

none

Notes:

These commands are used to set the attributes for an elevated surface part creation.

Example Usage:

```
#
# Select parent
#
part: select_begin
 2
part: select_end
#
# Set the attributes
#
elevsurf: begin
elevsurf: variable N_temperature
elevsurf: offset 0.0000e+00
elevsurf: scale_factor 2.0750e-02
elevsurf: sidewall ON
elevsurf: surface ON
elevsurf: end
elevsurf: create
```

See Also:

Associated elevsurf commands

Command:

```
elevsurf: component <x_scal> <y_scal> <z_scal>
```

Purpose:

Elevated surfaces are always created with a scalar. If creation variable is a vector this command specifies how the scalar will be created.

Parameters:

Type	Parameter	Description
float	x_scal	scale factor to apply to the x, y, z components of the vector
	y_scal	
	z_scal	

Notes:

If all the scale factors are set to 0.0, the magnitude of the vector will be used (this is the default).

Example Usage:

```
#
# Select existing elevated surfce part
#
part: select_begin
 3
part: select_end
part: modify_begin
elevsurf: variable N_velocity
#
# Use x-component of the vector
#
elevsurf: component 1.0000e+00 0.0000e+00 0.0000e+00
elevsurf: scale_factor 9.9660e-01
part: modify_end
```

See Also:

Associated elevsurf commands

elevsurf: create

Command:

```
elevsurf: create
```

Purpose:

Create an elevated surface according to specified or default attributes.

Parameters:

none

Example Usage:

```
#
# Select parent part
#
part: select_begin
 2
part: select_end
elevsurf: begin
elevsurf: variable N_temperature
elevsurf: offset 0.0000e+00
elevsurf: scale_factor 2.0750e-02
elevsurf: sidewall ON
elevsurf: surface ON
elevsurf: end
elevsurf: create
```

See Also:

Associated elevsurf commands

Command:

```
elevsurf: offset <offset_val>
```

Purpose:

Specifies a value that is added to the variable values before the scale factor is applied. Has the effect of moving the surface basis relative to the model.

Parameters:

Type	Parameter	Description
float	offset_val	value that is applied to the variable values before scaling.

Example Usage:

```
#
# Select existing elevated surface part
#
part: select_begin
 3
part: select_end
part: modify_begin
elevsurf: variable N_velocity
elevsurf: component 1.0000e+00 0.0000e+00 0.0000e+00
elevsurf: scale_factor 9.9660e-01
elevsurf: offset 1.5000e+00
part: modify_end
```

See Also:

Associated elevsurf commands

elevsurf: scale_factor

Command:

```
elevsurf: scale_factor <scale>
```

Purpose:

Specifies the scale factor to apply to the variable for the elevated surface.

Parameters:

Type	Parameter	Description
float	`scale`	scale factor to apply to variable values

Example Usage:

```
#
# Select existing elevated surface part
#
part: select_begin
 3
part: select_end
part: modify_begin
elevsurf: variable N_velocity
elevsurf: component 1.0000e+00 0.0000e+00 0.0000e+00
elevsurf: scale_factor 9.9660e-01
elevsurf: offset 1.5000e+00
part: modify_end
```

See Also:

Associated elevsurf commands

Command:

```
elevsurf: select_default
```

Purpose:

Set elevated surface selection to none, so that subsequent elevated surface attribute commands will apply to the defaults.

Parameters:

none

Notes:

The defaults are used for future elevated surface creations.

Example Usage:

```
elevsurf: select_default
part: modify_begin
elevsurf: scale_factor 3.0750e-02
part: modify_end
```

See Also:

Associated elevsurf commands

elevsurf: sidewall

Command:

```
elevsurf: sidewall <toggle>
```

Purpose:

Specifies whether the elevated sidewalls will be created.

Parameters:

Type	Parameter	Description
constant	toggle	ON
		OFF

Notes:

This command is used in conjunction with "elevsurf: surface", to achieve the combinations of surface and sidewall, surface only, or sidewalls only.

Example Usage:

```
#
# Select parent part
#
part: select_begin
 2
part: select_end
elevsurf: begin
elevsurf: variable N_temperature
elevsurf: offset 0.0000e+00
elevsurf: scale_factor 2.0750e-02
elevsurf: sidewall ON
elevsurf: surface ON
elevsurf: end
elevsurf: create
```

See Also:

Associated elevsurf commands

Command:

```
elevsurf: surface <toggle>
```

Purpose:

Specifies whether the elevated surface itself will be displayed.

Parameters:

Type	Parameter	Description
constant	toggle	ON
		OFF

Notes:

This command is used in conjunction with "elevsurf: sidewall", to achieve the combinations of surface and sidewall, surface only, or sidewalls only.

Example Usage:

```
#
# Select parent part
#
part: select_begin
 2
part: select_end
elevsurf: begin
elevsurf: variable N_temperature
elevsurf: offset 0.0000e+00
elevsurf: scale_factor 2.0750e-02
elevsurf: sidewall ON
elevsurf: surface ON
elevsurf: end
elevsurf: create
```

See Also:

Associated elevsurf commands

elevsurf: variable

Command:

```
elevsurf: variable <variable_name>
```

Purpose:

Assigns the variable to use for the creation of an elevated surface.

Parameters:

Type	Parameter	Description
string	variable_name	name of the variable to use

Example Usage:

```
#
# Select parent part
#
part: select_begin
 2
part: select_end
elevsurf: begin
elevsurf: variable N_temperature
elevsurf: offset 0.0000e+00
elevsurf: scale_factor 2.0750e-02
elevsurf: sidewall ON
elevsurf: surface ON
elevsurf: end
elevsurf: create
```

See Also:

Associated elevsurf commands

Commands in this Section:

file

file: image_color_type <color_type>
file: image_convert <toggle>
file: image_encapsulated <toggle>
file: image_envideo_type <compression_type>
file: image_file <filename>
file: image_format <image_format>
file: image_gamma <gamma_value>
file: image_orientation <orientation>
file: image_plot_only <toggle>
file: image_postscript_cull <toggle>
file: image_postscript_prev <toggle>
file: image_postscript_subd <toggle>
file: image_postscript_type <ps_type>
file: image_printer_command <command>
file: image_printer_model <printer>
file: image_quality <quality>
file: image_saturation <sat_value>
file: image_scale_factor <scale_value>
file: image_to_file <toggle>
file: image_to_printer <toggle>
file: image_window_size <window_sz>
file: image_window_xy <width> <height>
file: restore_context <filename>
file: restore_full_backup <filename>
file: save_context <filename>
file: save_context_type <type>
file: save_full_backup <filename>
file: save_image <filename>
file: save_multiple_images <toggle>
file: save_scenario_file <filename>
file: save_scenario_project <directory_name>
file: save_scenario_view <filename> <viewname>
file: scenario_desc_begin
file: scenario_desc_end
file: scenario_flipbook_anim <toggle>
file: scenario_keyframe_anim <toggle>
file: scenario_trace_anim <toggle>

See Also:

User Manual - Section , Section 6.1, File Menu Functions

How To Save and Restore an Archive

How To Print/Save and Image

How To Save/Restore Context

How To Save Scenario

file: image_color_type

Command:

```
file: image_color_type <color_type>
```

Purpose:

Specify that the image will be printed in color or black and white

Parameters:

Type	Parameter	Description	
constant	color_type	color	if the image should be printed in color
		black&white	if image should be printed in grayscale

Notes:

All formats support this parameter except TIFF and EnVideo

Example Usage:

```
file: image_format jpeg
file: image_color_type black&white
file: image_file /tmp/test_image
file: save_image
```

See Also:

Associated file commands

Command:

```
file: image_convert <toggle>
```

Purpose:

Change black backgrounds to white and change any white objects (geometry, plotters, annotations, etc.) to black.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	to change to white background with black foreground.
		OFF	to change back to the original colors.

Example Usage:

```
file: image_format sgi_rgb
file: image_file /tmp/test_image
file: image_convert ON
file: save_image
file: image_convert OFF
```

See Also:

Associated file commands

file: image_encapsulated

Command:

```
file: image_encapsulated <toggle>
```

Purpose:

Turn on/off encapsulation for postscript output

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	to include encapsulation information in the print output.
		`OFF`	(default) will produce print output without the encapsulation information.

Notes:

Encapsulation should only be used when the resulting image file will be used as part of another document. Printers do not normally interpret encapsulated image files.

Example Usage:

```
file: image_format postscript
file: image_color_type black&white
file: image_file /tmp/test_image
file: image_orientation landscape
file  image_encapsulated ON
file: save_image
```

See Also:

Associated file commands

Command:

```
file: image_envideo_type <compression_type>
```

Purpose:

Set the envideo format compression scheme.

Parameters:

Type	Parameter	Description	
constant	compression_type	rle	run length encoding
		jpeg	jpeg encoding

Notes:

rle is a lossless compression scheme, though it produces larger files than jpeg. If the compression scheme is jpeg, "`file: image_quality`" controls the jpeg compression/quality.

Example Usage:

```
file: image_format envideo
file: image_envideo_type jpeg
file: image_quality 80
file: image_file /tmp/test_image
file: save_image
```

See Also:

Associated file commands

file: image_file

Command:

```
file: image_file <filename>
```

Purpose:

Specify the filename to be used for image printing.

Parameters:

Type	Parameter	Description
string	`filename`	name of the image file to be saved

Notes:

Must be specified before a "`file: save_image`" if "`file: image_to_file`" is ON.

Example Usage:

```
file: image_format jpeg
file: image_file /tmp/test_image
file: save_image
```

See Also:

Associated file commands

Command:

```
file: image_format <image_format>
```

Purpose:

Specifies the file format to be used when an image is created.

Parameters:

Type	Parameter	Description
constant	image_format	The image format which must be one of:
		pict
		jpeg
		tiff
		pcl
		postscript
		sgi_rgb
		targa
		cei_rgbz
		envideo
		mpeg
		avi

Notes:

If the image is saved to a disk file the filename is concatenated with an extension as follows:

```
pict-.pct
jpeg-.jpg
tiff-.tif
pcl-.pcl
postscript-.ps
encapsulated ps -.eps
sgi_rgb-.rgb
targa-.tga
cei_rgbz-.rgbz
envideo-.evo
mpeg-.mpg
avi-.avi
```

Example Usage:

```
file: image_format jpeg
file: image_file /tmp/test_image
file: save_image
```

See Also:

Associated file commands

file: image_gamma

Command:

```
file: image_gamma <gamma_value>
```

Purpose:

Set gamma correction value for PCL image files.

Parameters:

Type	Parameter	Description
float	`gamma_value`	The range is dependent on the system. Typically values are on the order 1<= `gamma_value` <=3.

Notes:

Gamma corrects for monitor/printer non-linearities for color saturation/brightness.For example, without gamma correction, a full intensity red pixel will not be twice as bright as a half intensity pixel. 1.0 is the default and is used if not specified.

Example Usage:

```
file: image_format pcl
file: image_gamma 1.7
file: image_printer_model DeskJet_1200c
file: image_file /tmp/test_image
file: save_image
```

See Also:

Associated file commands

Command:

```
file: image_orientation <orientation>
```

Purpose:

Sets the image orientation for PCL and Postscript image formats

Parameters:

Type	Parameter	Description	
constant	`orientation`	`portrait`	to print image in portrait layout
		`landscape`	to print image in landscape layout

Notes:

"`portrait`" is used if not specified.

Example Usage:

```
file: image_format postscript
file: image_color_type black&white
file: image_file /tmp/test_image
file: image_orientation landscape
file: save_image
```

See Also:

Associated file commands

file: image_plot_only

Command:

```
file: image_plot_only <toggle>
```

Purpose:

When printing an image will turn invisible everything except for plotters.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	to save a view with plotters only
		OFF	for normal “show everything” operation

Notes:

If turned ON, should be turned OFF after a "file: save_image"

Example Usage:

```
file: image_format jpeg
file: image_quality 80
file: image_file /tmp/test_image
file: image_plot_only ON
file: save_image
file: image_plot_only OFF
```

See Also:

Associated file commands

Command:

```
file: image_postscript_cull <toggle>
```

Purpose:

For postscript format move-draw images turns on/off the removal of hidden geometry.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	to remove hidden geometry
		OFF	to keep all geometry

Notes:

Culling hidden geometry can vastly improve printer performance and reduce file size. However, processing time is significantly higher.

When OFF the geometry will be drawn from back to front.

The default is ON.

Example Usage:

```
file: image_format postscript
file: image_postscript_type move_draw
file: image_postscript_cull OFF
file: image_file /tmp/test_image
file: save_image
```

See Also:

Associated file commands

file: image_postscript_prev

Command:

```
file: image_postscript_prev <toggle>
```

Purpose:

Include a windows preview image in an encapsulated postscript (eps) file.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	if file will include a preview image
		`OFF`	if file will not include a preview image

Notes:

If a preview image is not included a EPS file will be shown as a grey rectangle in Windows applications. Many applications which can normally read a EPS file will not be able to read one which includes a windows preview image.

Example Usage:

```
file: image_format postscript
file: image_encapsulated ON
file: image_postscript_prev ON
file: image_file /tmp/test_image
file: save_image
```

See Also:

Associated file commands

Command:

```
file: image_postscript_subd <toggle>
```

Purpose:

For postscript format move-draw images turns on/off the subdivision of polygons for better shading.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	to turn on polygon subdivision
		OFF	to turn off polygon subdivision

Notes:

The postscript format supports only a single color per polygon. Lighting and false color mapping will look poor unless the polygon is subdivided into smaller (single color) regions. The default is ON.

Example Usage:

```
file: image_format postscript
file: image_postscript_type move_draw
file: image_postscript_subd ON
file: image_file /tmp/test_image
file: save_image
```

See Also:

Associated file commands

file: image_postscript_type

Command:

```
file: image_postscript_type <ps_type>
```

Purpose:

Set the postscript output to move-draw or bitmap.

Parameters:

Type	Parameter	Description	
constant	ps_type	move_draw	for move-draw mode
		image_pixels	for bitmap mode

Example Usage:

```
file: image_format postscript
file: image_file /tmp/test_image
file: image_postscript_type image_pixels
file: save_image
```

See Also:

Associated file commands

Command:

```
file: image_printer_command <command>
```

Purpose:

Specify the printer command to use when printing an image to a printer

Parameters:

Type	Parameter	Description
string	command	printer command used to print the image

Notes:

You can print an image straight to a printer only if the printer can properly interpret the image file. Many printers will, for example, properly print a postscript file. Printing a PICT file, however, will usually result in multiple pages of non-printable characters.

Example Usage:

```
file: image_format postscript
file: image_to_file OFF
file: image_to_printer ON
file: image_printer_command lpr -Pcolor
file: save_image
```

See Also:

Associated file commands

file: image_printer_model

Command:

```
file: image_printer_model <printer>
```

Purpose:

Set the target printer for a PCL image file.

Parameters:

Type	Parameter	Description
constant	printer	One of:
		LaserJet
		LaserJet_+
		LaserJet_500+
		Laser Jet_II
		LaserJet_IIp
		LaserJet_III
		LaserJet_4
		LaserJet_4L
		DeskJet_500
		DeskJet_500c
		DeskJet_550
		DeskJet_550c
		DeskJet_1200c

Example Usage:

```
file: image_format pcl
file: image_printer_model DeskJet_1200c
file: image_file /tmp/test_image
file: save_image
```

See Also:

Associated file commands

Command:

```
file: image_quality <quality>
```

Purpose:

Sets the quality for JPEG images.

Parameters:

Type	Parameter	Description
int	quality	0 <= quality <= 100
		100 is the highest quality (no compression)
		0 is the lowest quality (highest compression)

Notes:

JPEG is a lossy format, i.e., it trades off file size for quality. Normally a quality factor of around 80 will produce a file that will be on order 2X smaller than one produced with a quality factor set to 100 with negligible impact on quality. JPEG format is best used with shaded images

Example Usage:

```
file: image_format jpeg
file: image_quality 80
file: image_file /tmp/test_image
file: save_image
```

See Also:

Associated file commands

file: image_saturation

Command:

```
file: image_saturation <sat_value>
```

Purpose:

Desaturate color when saving an image to disk or printer.

Parameters:

Type	Parameter	Description
float	`sat_value`	A value from 0 (black and white) to 1 (full color saturation)

Notes:

Desaturating images is especially useful when saving images to a format which will be played back on a television monitor as fully saturated colors have a tendency to "bleed". 1.0 is the default and is used if not specified.

Example Usage:

```
file: image_format sgi_rgb
file: image_saturation 0.8
file: image_file /tmp/test_image
file: save_image
```

See Also:

Associated file commands

Command:

```
file: image_scale_factor <scale_value>
```

Purpose:

To scale a postscript image for the page size.

Parameters:

Type	Parameter	Description
float	scale_value	a value from 0 to 1 specifying portion of the page to use for printing

Example Usage:

```
file: image_format postscript
file: image_color_type black&white
file: image_file /tmp/test_image
file: image_orientation landscape
file: image_scale_factor .5
file: save_image
```

See Also:

Associated file commands

file: image_to_file

Command:

```
file: image_to_file <toggle>
```

Purpose:

Specifies if the image should be saved to a disk file.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	if the file should be saved to a file
		OFF	if the file should not be saved to a file

Notes:

If flag is set to OFF, the other choice, printing the image directly to a printer should be specified. If ON the "file: image_file" command should be specified. The default is ON, i.e., this command does not have to be specified if printing an image to a file.

Example Usage:

```
file: image_format jpeg
file: image_to_file ON
file: image_file /tmp/test_image
file: save_image
```

See Also:

Associated file commands

Command:

```
file: image_to_printer <toggle>
```

Purpose:

Specifies if the image should be printed directly to a printer.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	if the image should be printed directly to a printer
		OFF	if the image should not be printed to a printer

Notes:

If flag is set to OFF, the other choice, printing the image to a disk file should be specified. If ON, the "file: image_printer_command" should be specified.

Example Usage:

```
file: image_format jpeg
file: image_to_file OFF
file: image_to_printer ON
file: image_printer_command lpr -Pprinter
file: save_image
```

See Also:

Associated file commands

file: image_window_size

Command:

```
file: image_window_size <window_sz>
```

Purpose:

When printing image based images sets the window (and thus image) size.

Parameters:

Type	Parameter	Description	
constant	window_sz	normal	to use the current graphics window size
		full	to use a full screen size window
		User_defined	to specify the size of the window
		detached_display	use the detached display, as specified by the -dconfig command line argument

Notes:

When an image is saved a new window will be created of the size specified. The default is normal.

Example Usage:

```
file: image_format sgi_rgb
file: image_window_size full
file: image_file /tmp/test_image
file: save_image
```

See Also:

Associated file commands

Command:

```
file: image_window_xy <width> <height>
```

Purpose:

Sets the image width and height if the user has specified a `user_defined` image size.

Parameters:

Type	Parameter	Description
float	`width`	pixels in x direction
	`height`	pixels in y direction

Notes:

For interactive processing width and height are limited to the size of the managed display area. For batch rendering width and height are limited to 15360 x 4096.

If the aspect ratio of EnSight's graphics area does not match the specified values the resulting image will not match what is currently displayed. To avoid this problem the specified aspect ratio (width/height) must match what is currently in use. To find the currently in use values enter "`test: size_of_window`" in the command dialog. The currently in use values will be printed in the EnSight shell window.

Example Usage:

```
file: image_format sgi_rgb
file: image_window_size user_defined
file: image_window_xy 800 600
file: image_file /tmp/test_image
file: save_image
```

See Also:

Associated file commands

file: restore_context

Command:

```
file: restore_context <filename>
```

Purpose:

Restore a context file.

Parameters:

Type	Parameter	Description
string	`filename`	name of the context file to restore

Notes:

If a context file is restored before data is read the data used to create the context file is also restored together with the parts that existed at the time the context file was saved.

If a context file is restored after data is read but before parts are created the same parts that existed in the context file will be restored.

If a context file is restored after data and parts have been loaded, the context content is applied towards the loaded parts. In this case the part names must match. If part names do not match the user must manually match up the parts via a pop-up dialog.

Example Usage:

```
ensight7 -ctx /tmp/test.ctx
```

See Also:

Associated file commands

Command:

```
file: restore_full_backup <filename>
```

Purpose:

Restore an EnSight archive (restart) file.

Parameters:

Type	Parameter	Description
string	`filename`	name of the EnSight archive file

Notes:

The file specified must be located on the EnSight client machine.

The file specified contains the file names of the archive files for both the client and server machines.

When an EnSight archive file is restored, the EnSight command file used to generate the archive file is also restored. The command file is named ensig********.restore and is usually located in `/usr/tmp` (UNIX) or the user's home directory (windows). The restored command file contains the commands from the last EnSight session. In the case where the session was started with an archive restore, the command file will not be complete.

Archive files are usually not compatible between major EnSight releases and are usually not portable between machines. Should the restore of an archive fail, the user should consider using the command file mentioned above.

Example Usage:

```
file: restore_full_backup /tmp/test.ar
```

See Also:

Associated file commands

file: save_context

Command:

```
file: save_context <filename>
```

Purpose:

Save a context file.

Parameters:

Type	Parameter	Description
string	`filename`	name of the context file

Notes:

A context file provides a data independent way to apply a visualization to a new dataset.

The context file consists, at minimum, of a file containing the commands to recreate the visuals on the screen that existed at the time the context was saved.

The filename is used as a root name as multiple files are saved including the palette, view, and keyframe animation state.

Context files are not portable between big/little endian machines at this time.

Example Usage:

```
file: save_context /tmp/test.ctx
```

See Also:

Associated file commands

Command:

```
file: save_context_type <type>
```

Purpose:

Can specify the desired case to use, or all cases, when saving a context file.

Parameters:

Type	Parameter	Description
string	all_cases	A context file will be saved for each case
	current_case	A context file will be saved for the current case

Example Usage:

```
file: save_context_type all_cases
file: save_context myfile.ctx
```

See Also:

Associated file commands

file: save_full_backup

Command:

```
file: save_full_backup <filename>
```

Purpose:

Create an EnSight archive (restart) file.

Parameters:

Type	Parameter	Description
string	`filename`	name of the EnSight archive file

Notes:

Three files are actually written. The file name specified contains the name of the other two files and is written on the EnSight client machine. Besides the file specified, one file is written on the EnSight client and server machines containing the archive information.

Archive files are usually not compatible between major EnSight releases and are usually not portable between machines.

Example Usage:

```
file: save_full_backup /tmp/test.ar
```

See Also:

Associated file commands

Command:

```
file: save_image <filename>
```

Purpose:

Save an image to a disk file.

Parameters:

Type	Parameter	Description
string	`filename`	filename to save the image to

Example Usage:

```
file: image_format jpeg
file: image_file /tmp/test_image
file: save_image
```

See Also:

Associated file commands

file: save_multile_images

Command:

```
file: save_multiple_images <toggle>
```

Purpose:

Option to save a single image or multiple images for a flat-wall detached display.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	to save an image per pipe
		`OFF`	to save a single image for the wall

Notes:

When saving a flat-wall detached display, the user can choose to write one large image for the wall or one image per 'pipe' from the dconfig file. This option is only valid when the image size is `detached_display`.

Example Usage:

```
file: image_window_size detached_display
file: save_multiple_images ON
file: image_file prefix
```

See Also:

Associated file commands

Command:

```
file: save_scenario_file <filename>
```

Purpose:

Save a scenario file.

Parameters:

Type	Parameter	Description
string	`filename`	name of the file to use for the scenario information

Notes:

A scenario can be restored in EnLiten.

A scenario file contains information to completely restore the current visuals in EnLiten.

A scenario project contains additional information.

Example Usage:

```
file: save_scenario_file /tmp/test_scenario_file
```

See Also:

Associated file commands

file: save_scenario_project

Command:

```
file: save_scenario_project <directory_name>
```

Purpose:

Save a scenario project.

Parameters:

Type	Parameter	Description
string	`directory_name`	name of the directory to save the scenario information

Notes:

A scenario can be restored in EnLiten.

A scenario file contains information to completely restore the current visuals in EnLiten.

A scenario project contains a scenario file plus (a) a JPEG image file of what the project looks like, (b) a file containing the scenario description, and (c) an EnSight context file (can be multiple files).

The scenario file and associated project files will be named using the `directory_name` as the root filename.

Example Usage:

```
file: save_scenario_project /tmp/test_scenario_project
```

See Also:

Associated file commands

Command:

```
file: save_scenario_view <filename> <viewname>
```

Purpose:

Save a view as part of an existing scenario file.

Parameters:

Type	Parameter	Description
string	`filename`	name of the existing scenario file
	`viewname`	a description for the view

Example Usage:

```
file: save_scenario_project /tmp/test_scenario_file
file: save_scenario_view /tmp/test_scenario_file top_view
```

See Also:

Associated file commands

file: scenario_desc_begin/end

Command:

```
file: scenario_desc_begin
file: scenario_desc_end
```

Purpose:

Delimit the description for a scenario project.

Parameters:

none

Notes:

A scenario project contains descriptions of the project suitable for inclusion in HTML documents.

Example Usage:

```
file: scenario_desc_begin
This project contains multiple studies from 2001
file: scenario_desc_end
file save_scenario_project /tmp/test_project
```

See Also:

Associated file commands

Command:

```
file: scenario_flipbook_anim <toggle>
```

Purpose:

When a scenario file/project is saved will determine if flipbook animation (if defined) will be included.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	if flipbook animations are to be included
		OFF	if flipbook animations are not to be included

Notes:

The default is ON

Example Usage:

```
file: scenario_flipbook_anim ON
file: save_scenario_file /tmp/test_scenario
```

See Also:

Associated file commands

file: scenario_keyframe_anim

Command:

```
file: scenario_keyframe_anim <toggle>
```

Purpose:

When a scenario file/project is saved will determine if keyframe animation (if defined) will be included.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	if keyframe animations are to be included
		`OFF`	if keyframe animations are not to be included

Notes:

The default is `ON`.

Example Usage:

```
file: scenario_keyframe_anim ON
file: save_scenario_file /tmp/test_scenario
```

See Also:

Associated file commands

Command:

```
file: scenario_trace_anim <toggle>
```

Purpose:

When a scenario file/project is saved will determine if particle trace animation (if defined) will be included.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	if particle trace animations are to be included
		OFF	if particle trace animations are not to be included

Notes:

The default is ON.

Example Usage:

```
file: scenario_trace_anim ON
file: save_scenario_file /tmp/test_scenario
```

See Also:

Associated file commands

Commands in this Section:

format

format: replace <index> <format_string>

See Also:

User Manual - Section 8.2, Annot Mode

How To Create Color Legends

format: replace index

Command:

```
format: replace <index> <format_string>
```

Purpose:

Replaces a default legend format string with the specified string.

Parameters:

Type	Parameter	Description
int	`index`	which format choice to replace
string	`format_string`	printf() style format to use

Notes:

Instead of using the default legend formats, custom ones can be specified to appear in the legend format dialog.

Example Usage:

```
#
# I'd rather see printf() style %.3f formatting as the
# first choice in the legend format dialog format:
#
replace 1 %.3f
```

See Also:

Associated format commands

Commands in this Section:

frame

frame: #_of_labels_x <num>
frame: #_of_labels_y <num>
frame: #_of_labels_z <num>
frame: assign <num>
frame: create
frame: delete
frame: len_x <len>
frame: len_y <len>
frame: len_z <len>
frame: line_width <width>
frame: modify_begin
frame: modify_end
frame: orientation_x <x_comp> <y_comp> <z_comp>
frame: orientation_y <x_comp> <y_comp> <z_comp>
frame: orientation_z <x_comp> <y_comp> <z_comp>
frame: origin <x_coord> <y_coord> <z_coord>
frame: rgb <red_val> <grn_val> <blu_val>
frame: select_all
frame: select_begin
frame: select_end
frame: select_default
frame: symmetry_angle <angle>
frame: symmetry_axis <axis>
frame: symmetry_delta <del_x> <del_y> <del_z>
frame: symmetry_file <filename>
frame: symmetry_mirror_x <toggle>
frame: symmetry_mirror_y <toggle>
frame: symmetry_mirror_z <toggle>
frame: symmetry_mirror_xy <toggle>
frame: symmetry_mirror_yz <toggle>
frame: symmetry_mirror_xz <toggle>
frame: symmetry_mirror_xyz <toggle>
frame: symmetry_rinstances <num>
frame: symmetry_tinstances <num>
frame: symmetry_type <options>
frame: symmetry_use_file <toggle>
frame: type <coord_system>
frame: visible <toggle>
frame: x_labels <toggle>
frame: y_labels <toggle>
frame: z_labels <toggle>

See Also:

User Manual - Section 8.5, Frame Mode
How To Create and Manipulate Frames

frame: #_of_labels_x/y/z

Command:

```
frame: #_of_labels_x <num>
frame: #_of_labels_y <num>
frame: #_of_labels_z <num>
```

Purpose:

Sets the number of labels that will be displayed along the frame x, y, z axes.

Parameters:

Type	Parameter	Description
int	num	number of labels

Example Usage:

```
frame: select_begin
 1
frame: select_end
frame: x_labels ON
frame: y_labels ON
frame: z_labels ON
frame: #_of_labels_x 3
frame: #_of_labels_y 4
frame: #_of_labels_z 3
```

See Also:

Associated frame commands

Command:

```
frame: assign <num>
```

Purpose:

Assigns selected parts to the specified frame.

Parameters:

Type	Parameter	Description
int	num	frame to assign the selected parts to

Notes:

Before this command is issued, you need to have one (and one only) frame selected, as well as the parts you will be assigning to the frame.

Example Usage:

```
frame: create
frame: select_begin
 1
frame: select_end
part: select_begin
 2
part: select_end
frame: assign 1
```

See Also:

Associated frame commands

frame: create

Command:

```
frame: create
```

Purpose:

Creates a new frame.

Parameters:

none

Notes:

The new frame will be created with the current default attributes.

Example Usage:

```
frame: create
```

See Also:

Associated frame commands

Command:

```
frame: delete
```

Purpose:

Deletes selected frames.

Parameters:

none.

Notes:

Before this command is issued, you need to select the frames you want deleted.

Example Usage:

```
frame: select_begin
 1 2
frame: select_end
frame: delete
```

See Also:

Associated frame commands

frame: len_x/y/z

Command:

```
frame: len_x <len>
frame: len_y <len>
frame: len_z <len>
```

Purpose:

Sets the display length of the frame x, y, z axes.

Parameters:

Type	Parameter	Description
float	`len`	length of the displayed frame axis (in model coordinates)

Example Usage:

```
frame: select_begin
 1
frame: select_end
frame: len_x 8.0000e-01
frame: len_y 8.3333e-01
frame: len_z 8.3333e-01
```

See Also:

Associated frame commands

Command:

```
frame: line_width <width>
```

Purpose:

Sets the line width of the displayed frame axes.

Parameters:

Type	Parameter	Description
int	`width`	number of pixels wide

Example Usage:

```
frame: select_begin
 1
frame: select_end
frame: line_width 3
```

See Also:

Associated frame commands

frame: modify_begin/end

Command:

```
frame: modify_begin
<frame_command>
<frame_command>
.
.
.
<frame_command>
frame: modify_end
```

Purpose:

Grouping several frame attributes together that all need to be set before application.

Parameters:

Type	Parameter	Description
command	`frame_command`	any of the attribute changing `frame:` commands

Notes:

Since modification of frame symmetry attributes typically need more than one attribute change before being able to properly perform the option, one typically should encase the attribute lines between `frame: modify_begin` and `frame: modify_end` commands.

Example Usage:

```
frame: select_begin
 1
frame: select_end
frame: modify_begin
frame: symmetry_type translational
frame: symmetry_use_file OFF
frame: symmetry_tinstances 2
frame: symmetry_delta 1.0000e+00 0.0000e+00 0.0000e+00
frame: modify_end
```

See Also:

Associated frame commands

Command:

```
frame: orientation_x <x_comp> <y_comp> <z_comp>
frame: orientation_y <x_comp> <y_comp> <z_comp>
frame: orientation_z <x_comp> <y_comp> <z_comp>
```

Purpose:

Sets the orientation of the x, y, z axes of the frame.

Parameters:

Type	Parameter	Description
float	x_comp	x, y, z components of vector representing the frame axis.
	y_comp	
	z_comp	

Notes:

Frame 0 is the model coordinate system. It is always cartesian and it cannot be modified. Other frames are always defined in terms of this cartesian coordinate system using the "`frame: origin`" command and this command (and it's two other sister commands).

Example Usage:

```
frame: select_begin
 1
frame: select_end
frame: origin 1.0000e+00 1.0000e+00 5.0000e-01
frame: orientation_x 1.0000e+00 0.0000e+00 0.0000e+00
frame: orientation_y 0.0000e+00 1.0000e+00 0.0000e+00
frame: orientation_z 0.0000e+00 0.0000e+00 1.0000e+00
```

See Also:

Associated frame commands

frame: origin

Command:

```
frame: origin <x_coord> <y_coord> <z_coord>
```

Purpose:

Sets the location of the origin of a frame in model coordinates.

Parameters:

Type	Parameter	Description
float	x_coord	model x, y, z coordinates of the origin of the frame
	y_coord	
	z_coord	

Notes:

Frame 0 is the model coordinate system. It is always cartesian and cannot be moved. Other frames are always defined in terms of this cartesian coordinate system using this command and the "`frame: orientation`" commands.

Example Usage:

```
frame: select_begin
 1
frame: select_end
frame: origin 1.0000e+00 1.0000e+00 5.0000e-01
frame: orientation_x 1.0000e+00 0.0000e+00 0.0000e+00
frame: orientation_y 0.0000e+00 1.0000e+00 0.0000e+00
frame: orientation_z 0.0000e+00 0.0000e+00 1.0000e+00
```

See Also:

Associated frame commands

Command:

```
frame: rgb <red_val> <grn_val> <blu_val>
```

Purpose:

Sets the color of the axes and labels of the selected frame.

Parameters:

Type	Parameter	Description
float	red_val	red, green, and blue components between 0.0 and 1.0
	grn_val	
	blu_val	

Example Usage:

```
frame: select_begin
 1
frame: select_end
frame: rgb 1.0000e+00 0.0000e+00 3.9608e-01
```

See Also:

Associated frame commands

frame: select_all

Command:

```
frame: select_all
```

Purpose:

Selection of all visible frames.

Parameters:

none.

Notes:

Can be used to select all visible frames that subsequent frame commands will apply to.

Example Usage:

```
frame: select_all
```

See Also:

Associated frame commands

Command:

```
frame: select_begin
<frame1 frame2 ... frameN>
frame: select_end
```

Purpose:

Selection of frames.

Parameters:

Type	Parameter	Description
int	`frameN`	frame numbers

Notes:

These commands are needed to select frames before modifying any attributes, doing assignments, etc.

Example Usage:

```
frame: select_begin
 1 2
frame: select_end
frame: delete
```

See Also:

Associated frame commands

frame: select_default

Command:

```
frame: select_default
```

Purpose:

Deselects all frames, such that subsequent frame attribute commands will modify default frame attributes.

Parameters:

none.

Notes:

When a new frame is created, it uses default attributes. This command allows you to change the defaults.

Example Usage:

```
frame: select_default
```

See Also:

Associated frame commands

Command:

```
frame: symmetry_angle <angle>
```

Purpose:

Sets the angle between instances for rotational symmetry.

Parameters:

Type	Parameter	Description
float	`angle`	angle, in degrees, between instances

Notes:

Since modification of frame symmetry attributes typically need more than one attribute change before being able to properly perform the option, one typically should encase the attribute lines between `frame: modify_begin` and `frame: modify_end` commands.

Example Usage:

```
frame: select_begin
 1
frame: select_end
frame: modify_begin
frame: symmetry_type rotational
frame: symmetry_angle 3.5000e+01
frame: symmetry_rinstances 2
frame: symmetry_use_file OFF
frame: modify_end
```

See Also:

Associated frame commands

frame: symmetry_axis

Command:

```
frame: symmetry_axis <axis>
```

Purpose:

Sets the major axis to use for rotational symmetry.

Parameters:

Type	Parameter	Description
float	`axis`	major axis: x, y, or z

Notes:

Since modification of frame symmetry attributes typically need more than one attribute change before being able to properly perform the option, one typically should encase the attribute lines between `frame: modify_begin` and `frame: modify_end` commands.

Example Usage:

```
frame: select_begin
 1
frame: select_end
frame: modify_begin
frame: symmetry_type rotational
frame: symmetry_angle 3.5000e+01
frame: symmetry_rinstances 2
frame: symmetry_use_file OFF
frame: symmetry_axis y
frame: modify_end
```

See Also:

Associated frame commands

Command:

```
frame: symmetry_delta <del_x> <del_y> <del_z>
```

Purpose:

Sets the delta values in the xyz directions between instances for translational symmetry.

Parameters:

Type	Parameter	Description
float	del_x	x, y, z delta values (in model coordinates) between instances
	del_y	
	del_z	

Notes:

Since modification of frame symmetry attributes typically need more than one attribute change before being able to properly perform the option, one typically should encase the attribute lines between frame: `modify_begin` and `frame: modify_end` commands.

Example Usage:

```
frame: select_begin
 1
frame: select_end
frame: modify_begin
frame: symmetry_type translational
frame: symmetry_use_file OFF
frame: symmetry_tinstances 2
frame: symmetry_delta 1.0000e+00 0.0000e+00 0.0000e+00
frame: modify_end
```

See Also:

Associated frame commands

frame: symmetry_file

Command:

```
frame: symmetry_file <filename>
```

Purpose:

Sets the name of the periodic matchfile for the computational symmetry.

Parameters:

Type	Parameter	Description
string	`filename`	filename of periodic symmetry match file - should be a full path name.

Notes:

Since modification of frame symmetry attributes typically need more than one attribute change before being able to properly perform the option, one typically should encase the attribute lines between `frame: modify_begin` and `frame: modify_end` commands.

If no matchfile is used, the rotational angle and number of instances (for rotational symmetry) or the translational deltas and number of instances (for translational symmetry) must be set in the user interface. Also, node hashing will be done in an attempt to eliminate duplicate nodes at the boundaries of instances. If a matchfile is used, the matchfile must follow the prescribed format, which contains the angle or delta, and the list of paired nodes that match at the instance boundaries. The number of instances still comes from the user interface.

Example Usage:

```
frame: select_begin
 1
frame: select_end
frame: modify_begin
frame: symmetry_type rotational
frame: symmetry_rinstances 5
frame: symmetry_use_file ON
frame: symmetry_file /usr/tmp/model.trn_match
frame: modify_end
```

See Also:

Associated frame commands

Command:

```
frame: symmetry_mirror_x <toggle>
frame: symmetry_mirror_y <toggle>
frame: symmetry_mirror_z <toggle>
frame: symmetry_mirror_xy <toggle>
frame: symmetry_mirror_yz <toggle>
frame: symmetry_mirror_xz <toggle>
frame: symmetry_mirror_xyz <toggle>
```

Purpose:

Sets the flag for mirror symmetry in the x / y / z / xy / yz / xz / xyz directions.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	to mirror
		`OFF`	otherwise

Notes:

Since modification of frame symmetry attributes typically need more than one attribute change before being able to properly perform the option, one typically should encase the attribute lines between `frame: modify_begin` and `frame: modify_end` commands.

Example Usage:

```
frame: select_begin
 1
frame: select_end
frame: modify_begin
frame: symmetry_type mirror
frame: symmetry_mirror_x ON
frame: symmetry_mirror_z ON
frame: symmetry_mirror_y ON
frame: symmetry_mirror_xy OFF
frame: symmetry_mirror_yz OFF
frame: symmetry_mirror_xz OFF
frame: symmetry_mirror_xyz OFF
frame: modify_end
```

See Also:

Associated frame commands

frame: symmetry_rinstances

Command:

```
frame: symmetry_rinstances <num>
```

Purpose:

Sets the number of instances for rotational symmetry.

Parameters:

Type	Parameter	Description
int	num	number of instances

Notes:

Since modification of frame symmetry attributes typically need more than one attribute change before being able to properly perform the option, one typically should encase the attribute lines between frame: `modify_begin` and `frame: modify_end` commands.

Example Usage:

```
frame: select_begin
 1
frame: select_end
frame: modify_begin
frame: symmetry_type rotational
frame: symmetry_angle 3.5000e+01
frame: symmetry_rinstances 5
frame: symmetry_use_file OFF
frame: modify_end
```

See Also:

Associated frame commands

Command:

```
frame: symmetry_tinstances <num>
```

Purpose:

Sets the number of instances for translational symmetry.

Parameters:

Type	Parameter	Description
int	num	number of instances

Notes:

Since modification of frame symmetry attributes typically need more than one attribute change before being able to properly perform the option, one typically should encase the attribute lines between `frame: modify_begin` and `frame: modify_end` commands.

Example Usage:

```
frame: select_begin
 1
frame: select_end
frame: modify_begin
frame: symmetry_type translational
frame: symmetry_use_file OFF
frame: symmetry_tinstances 2
frame: symmetry_delta 1.0000e+00 0.0000e+00 0.0000e+00
frame: modify_end
```

See Also:

Associated frame commands

frame: symmetry_type

Command:

```
frame: symmetry_type <options>
```

Purpose:

Sets the type of periodic symmetry to apply to parts attached to the frame.

Parameters:

Type	Parameter	Description	
constant	options	mirror	instances will reflect into appropriate quadrants
		rotational	instances will repeat at specified angles
		translational	instances will repeat in a given direction
		non	single instance only (default)

Notes:

Since modification of frame symmetry attributes typically need more than one attribute change before being able to properly perform the option, one typically should encase the attribute lines between `frame: modify_begin` and `frame: modify_end` commands.

Example Usage:

```
frame: select_begin
 1
frame: select_end
frame: modify_begin
frame: symmetry_type rotational
frame: symmetry_angle 3.5000e+01
frame: symmetry_rinstances 2
frame: symmetry_use_file OFF
frame: modify_end
```

See Also:

Associated frame commands

Command:

```
frame: symmetry_use_file <toggle>
```

Purpose:

Sets whether a periodic matchfile will be used for specification of the rotational or translational symmetry.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	if matchfile is to be used
		OFF	otherwise

Notes:

Since modification of frame symmetry attributes typically need more than one attribute change before being able to properly perform the option, one typically should encase the attribute lines between `frame: modify_begin` and `frame: modify_end` commands.

If no matchfile is used, the rotational angle and number of instances (for rotational symmetry) or the translational deltas and number of instances (for translational symmetry) must be set in the user interface. Also, node hashing will be done in an attempt to eliminate duplicate nodes at the boundaries of instances. If a matchfile is used, the matchfile must follow the prescribed format, which contains the angle or delta, and the list of paired nodes that match at the instance boundaries. The number of instances still comes from the user interface.

Example Usage:

```
frame: select_begin
 1
frame: select_end
frame: modify_begin
frame: symmetry_type rotational
frame: symmetry_rinstances 5
frame: symmetry_use_file ON
frame: symmetry_file /usr/tmp/model.trn_match
frame: modify_end
```

See Also:

Associated frame commands

frame: type

Command:

```
frame: type <coord_system>
```

Purpose:

Set the coordinate system type for the frame.

Parameters:

Type	Parameter	Description	
constant	coord_system	rectangular	for cartesian coordinate system (xyz)
		cylindrical	for cylindrical coordinate system (rtz)
		spherical	for spherical coordinate system (rpt)

Example Usage:

```
frame: select_begin
 1
frame: select_end
frame: type cylindrical
```

See Also:

Associated frame commands

Command:

```
frame: visible <toggle>
```

Purpose:

Set the frame axis visibility on or off

Parameters:

Type	Parameter	Description
constant	toggle	ON
		OFF

Example Usage:

```
frame: select_begin
 1
frame: select_end
frame: visible ON
```

See Also:

Associated frame commands

frame: x/y/z_labels

Command:

```
frame: x_labels <toggle>
frame: y_labels <toggle>
frame: z_labels <toggle>
```

Purpose:

Sets the visibility of labels along the frame x/y/z axis.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	for labels
		`OFF`	for no labels

Example Usage:

```
frame: select_begin
 1
frame: select_end
frame: x_labels ON
frame: y_labels ON
frame: z_labels ON
frame: #_of_labels_x 3
frame: #_of_labels_y 4
frame: #_of_labels_z 3
```

See Also:

Associated frame commands

Commands in this Section:

function

See Also:

User Manual - Section 4.2, Variable Summary & Palette

How To Edit Color Palettes

function: #_of_colors

Command:

```
function: #_of_colors <num>
```

Purpose:

Sets the number of colors per level in a texture palette.

Parameters:

Type	Parameter	Description
int	num	number of colors per level in the texture palette

Notes:

This command is only useful when doing texture palettes.

Example Usage:

```
function: palette temperature
function: #_of_colors 4
```

See Also:

Associated function commands

Command:

```
function: #_of_levels <num>
```

Purpose:

Sets the number of levels in the palette.

Parameters:

Type	Parameter	Description
int	num	number of levels in the palette

Notes:

Each level has a value and a color associated with it. Interpolations occur between the levels according to the type and scale used.

Example Usage:

```
function: palette N_temperature
function: modify_begin
function: #_of_levels 6
function: edit_level 2
function: value 9.6464e+00
function: edit_level 3
function: value 1.9283e+01
function: edit_level 4
function: value 2.8919e+01
function: edit_level 5
function: value 3.8556e+01
function: edit_level 6
function: value 4.8192e+01
function: edit_level 1
function: RGB 5.0000e-01 0.0000e+00 1.0000e+00
function: edit_level 2
function: RGB 0.0000e+00 0.0000e+00 1.0000e+00
function: edit_level 3
function: RGB 0.0000e+00 1.0000e+00 1.0000e+00
function: edit_level 4
function: RGB 0.0000e+00 1.0000e+00 0.0000e+00
function: edit_level 5
function: RGB 1.0000e+00 1.0000e+00 0.0000e+00
function: edit_level 6
function: RGB 1.0000e+00 0.0000e+00 0.0000e+00
function: modify_end
```

See Also:

Associated function commands

function: display_undefined

Command:

```
function: display_undefined <option>
```

Purpose:

Selects how elements/nodes, at which variables are undefined, will be displayed.

Parameters:

Type	Parameter	Description	
constant	option	One of the following:	
		by_part_color	elements/nodes with undefined variables are colored by the part color
		by_invisible	elements/nodes with undefined variables are not displayed

See Also:

Associated function commands

Command:

```
function: edit_level <num>
```

Purpose:

Sets the level in the palette for which value or color will be set/modified.

Parameters:

Type	Parameter	Description
int	num	level number (1 based, with 1 being the lowest level)

Notes:

Each level has a value and a color associated with it. Interpolations occur between the levels according to the type and scale used.

Example Usage:

```
function: palette N_temperature
function: modify_begin
function: #_of_levels 6
function: edit_level 2
function: value 9.6464e+00
function: edit_level 3
function: value 1.9283e+01
function: edit_level 4
function: value 2.8919e+01
function: edit_level 5
function: value 3.8556e+01
function: edit_level 6
function: value 4.8192e+01
function: edit_level 1
function: RGB 5.0000e-01 0.0000e+00 1.0000e+00
function: edit_level 2
function: RGB 0.0000e+00 0.0000e+00 1.0000e+00
function: edit_level 3
function: RGB 0.0000e+00 1.0000e+00 1.0000e+00
function: edit_level 4
function: RGB 0.0000e+00 1.0000e+00 0.0000e+00
function: edit_level 5
function: RGB 1.0000e+00 1.0000e+00 0.0000e+00
function: edit_level 6
function: RGB 1.0000e+00 0.0000e+00 0.0000e+00
function: modify_end
```

See Also:

Associated function commands

function: limit_fringes

Command:

```
function: limit_fringes <option>
```

Purpose:

Selects how shaded elements above or below the min and max palette levels will be displayed.

Parameters:

Type	Parameter	Description	
constant	option	One of the following:	
		No	elements/nodes below the min level are colored by min level and elements/nodes above the max level are colored by the max level.
		by_part_color	elements/nodes below the min and above the max are colored by the part color
		by_invisible	elements/nodes below the min and above the max are not displayed

Example Usage:

```
function: palette N_temperature
function: modify_begin
function: type continuous
function: limit_fringes by_invisible
function: modify_end
```

See Also:

Associated function commands

Command:

```
function: modify_begin
<function_command1>
<function_command2>
.
.
.
<function_commandN>
function: modify_end
```

Purpose:

Groups function attribute commands together before applying them.

Parameters:

Type	Parameter	Description
string	function_commandN	any of the function commands which modify attributes

Example Usage:

```
function: palette Stress
function: modify_begin
function: edit_level 1
function: RGB 1.0000e+00 0.0000e+00 0.0000e+00
function: edit_level 2
function: RGB 1.0000e+00 1.0000e+00 0.0000e+00
function: edit_level 3
function: RGB 0.0000e+00 1.0000e+00 0.0000e+00
function: edit_level 4
function: RGB 0.0000e+00 1.0000e+00 1.0000e+00
function: edit_level 5
function: RGB 0.0000e+00 0.0000e+00 1.0000e+00
function: edit_level 6
function: RGB 5.0000e-01 0.0000e+00 1.0000e+00
function: modify_end
```

See Also:

Associated function commands

function: over_time

Command:

```
function: over_time <toggle> [<beg_step> <end_step>]
```

Purpose:

Sets the option to include variable ranges over a time range on or off.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	to include variable ranges over time. For this option, the `beg_step` and `end_step` must be included
		`OFF`	variable ranges at current time only
int	`beg_step`	The beginning time step to consider for the time range. Only needed if `toggle` is `ON`	
int	`end_step`	The ending time step to consider for the time range. Only needed if `toggle` is `ON`	

Notes:

The minimum and maximum values of a variable's range, and thus the resulting palette levels will be based on the current time when a variable is activated, unless this command has been used.

Example Usage:

```
function: palette Stress
function: over_time ON 0 16
function: Displacement
function: over_time OFF
```

See Also:

Associated function commands

Command:

```
function: palette <palette_name>
```

Purpose:

Selects the palette that will be modified.

Parameters:

Type	Parameter	Description
string	palette_name	name of the palette (It will be the same as the variable, or variable component that it represents).

Notes:

Only one palette at a time can be modified.

Example Usage:

```
function: palette N_temperature
function: modify_begin
function: #_of_levels 4
function: edit_level 2
function: value 1.6064e+01
function: edit_level 3
function: value 3.2128e+01
function: edit_level 4
function: value 4.8192e+01
function: edit_level 2
function: RGB 0.0000e+00 1.0000e+00 6.6667e-01
function: edit_level 3
function: RGB 6.6667e-01 1.0000e+00 0.0000e+00
function: edit_level 4
function: RGB 1.0000e+00 0.0000e+00 0.0000e+00
function: modify_end
```

See Also:

Associated function commands

function: restore

Command:

```
function: restore <palette_filename>
```

Purpose:

Restore a previously saved palette from a file.

Parameters:

Type	Parameter	Description
string	`palette_filename`	name of the file containing a palette definition

Example Usage:

```
function: restore /usr/tmp/mystress.pal
function: palette Stress
```

See Also:

Associated function commands

Command:

```
function: restore_predefinedpal <level_option> <palette_name>
```

Purpose:

Restores predefined palettes that are in the user's default palette directory.

Parameters:

Type	Parameter	Description	
constant	level_option	One of the following:	
		use_new_levels	changes the number of levels to match that of the predefined palette.
		use_existing_levels	keeps the number of current levels and interpolates the predefined palette values and colors to this number of levels
string	palette_name	name of the predefined palette to restore	

Notes:

Predefined palettes reside in a user's .ensight7/palettes directory.

Example Usage:

```
function: restore_predefinedpal use_new_levels mypal
```

See Also:

Associated function commands

function: RGB

Command:

```
function: RGB <red_val> <grn_val> <blu_val>
```

Purpose:

Sets the color to associate with a level in the palette.

Parameters:

Type	Parameter	Description
float	red_val	red, green, blue components of color (0.0 to 1.0)
	grn_val	
	blu_val	

Notes:

Each level has a value and a color associated with it. Interpolations occur between the levels according to the type and scale used.

Example Usage:

```
function: palette N_temperature
function: modify_begin
function: #_of_levels 6
function: edit_level 2
function: value 9.6464e+00
function: edit_level 3
function: value 1.9283e+01
function: edit_level 4
function: value 2.8919e+01
function: edit_level 5
function: value 3.8556e+01
function: edit_level 6
function: value 4.8192e+01
function: edit_level 1
function: RGB 5.0000e-01 0.0000e+00 1.0000e+00
function: edit_level 2
function: RGB 0.0000e+00 0.0000e+00 1.0000e+00
function: edit_level 3
function: RGB 0.0000e+00 1.0000e+00 1.0000e+00
function: edit_level 4
function: RGB 0.0000e+00 1.0000e+00 0.0000e+00
function: edit_level 5
function: RGB 1.0000e+00 1.0000e+00 0.0000e+00
function: edit_level 6
function: RGB 1.0000e+00 0.0000e+00 0.0000e+00
function: modify_end
```

See Also:

Associated function commands

Command:

```
function: save <palette_filename>
```

Purpose:

Save the current palette to a file.

Parameters:

Type	Parameter	Description
string	palette_filename	name of the file containing a palette definition

Notes:

A saved palette can later be restored.

Example Usage:

```
function: palette Stress
function: save /usr/tmp/mystress.pal
```

See Also:

Associated function commands

function: save_all

Command:

```
function: save_all <palette_filename>
```

Purpose:

Save all palettes to a file.

Parameters:

Type	Parameter	Description
string	`palette_filename`	name of the file containing palette definitions

Notes:

Palettes saved in a file can later be restored.

Example Usage:

```
function: save_all /usr/tmp/allmy.pal
```

See Also:

Associated function commands

Command:

```
function: save_current_case <palette_filename>
```

Purpose:

Save palettes, of active variables of the currently selected case, to a file.

Parameters:

Type	Parameter	Description
string	`palette_filename`	name of the file containing palette definitions

Notes:

Palettes saved in a file can later be restored.

Note that this command is not generated by EnSight, but can be placed in a command file manually if desired.

Example Usage:

```
function: save_current_case /usr/tmp/allmy.pal
```

See Also:

Associated function commands

function: scale

Command:

```
function: scale <scale_type>
```

Purpose:

Sets the type of scale for value-separation of levels.

Parameters:

Type	Parameter	Description	
constant	scale_type	Can be one of the following types:	
		linear	value-separation or levels is uniform and values map linearly to the colors
		quadratic	value-separation of levels is not uniform, but are based on the second order of the variable
		logarithmic	value-separation of levels in not uniform, but are based on the base-10 logarithm of the variable

Example Usage:

```
function: palette N_temperature
function: type continuous
function: modify_begin
function: scale quadratic
function: edit_level 2
function: value 1.9373e+00
function: edit_level 3
function: value 7.7191e+00
function: edit_level 4
function: value 1.7356e+01
function: edit_level 5
function: value 3.0846e+01
function: edit_level 6
function: value 4.8192e+01
function: modify_end
```

See Also:

Associated function commands

Command:

```
function: select_default
```

Purpose:

Set palette selection to none, so that subsequent function attribute commands will apply to the default.

Parameters:

none

Example Usage:

```
function: select_default
function: #_of_levels 4
function: modify_begin
function: edit_level 1
function: value 1.0000e-02
function: edit_level 2
function: value 1.6071e+01
function: edit_level 3
function: value 3.2131e+01
function: edit_level 4
function: value 4.8192e+01
function: modify_end
```

See Also:

Associated function commands

function: type

Command:

```
function: type <type>
```

Purpose:

Sets the type of color gradation that will be applied across elements.

Parameters:

Type	Parameter	Description	
constant	`type`	Can be one of the following types:	
		`continuous`	legend will be smoothly varied, and an element's color will vary across the element as needed
		`banded`	legend will show lines or demarcation, as will elements as needed
		`constant`	legend will appear as continuous, but elements will be colored a constant color over the entire element

Example Usage:

```
function: palette N_temperature
function: type banded
```

See Also:

Associated function commands

Command:

```
function: value <var_value>
```

Purpose:

Sets the value to associate with a level in the palette.

Parameters:

Type	Parameter	Description
float	var_value	variable value to assign to the level

Notes:

Each level has a value and a color associated with it. Interpolations occur between the levels according to the type and scale used.

Example Usage:

```
function: palette N_temperature
function: modify_begin
function: #_of_levels 6
function: edit_level 2
function: value 9.6464e+00
function: edit_level 3
function: value 1.9283e+01
function: edit_level 4
function: value 2.8919e+01
function: edit_level 5
function: value 3.8556e+01
function: edit_level 6
function: value 4.8192e+01
function: edit_level 1
function: RGB 5.0000e-01 0.0000e+00 1.0000e+00
function: edit_level 2
function: RGB 0.0000e+00 0.0000e+00 1.0000e+00
function: edit_level 3
function: RGB 0.0000e+00 1.0000e+00 1.0000e+00
function: edit_level 4
function: RGB 0.0000e+00 1.0000e+00 0.0000e+00
function: edit_level 5
function: RGB 1.0000e+00 1.0000e+00 0.0000e+00
function: edit_level 6
function: RGB 1.0000e+00 0.0000e+00 0.0000e+00
function: modify_end
```

See Also:

Associated function commands

Commands in this Section:

isos

See Also:

User Manual - Section 7.3, Isosurface Create/Update

How To Create Isosurfaces

isos: begin/end

Command:

```
isos: begin
isos: end
```

Purpose:

Delimit the modifications for iso-surface attributes

Parameters:

none

Notes:

These commands are used to set the attributes for a isosurface part creation.

Example Usage:

```
isos: begin
isos: variable temperature
isos: value 1.1
isos: type isosurface
isos: end
isos: create
```

See Also:

Associated isos commands

Command:

```
isos: component <x_scale> <y_scale> <z_scale>
```

Purpose:

The isosurface algorithim works only on scalar variables. For a vector, this command specifies how a scalar will be created from the vector.

Parameters:

Type	Parameter	Description
float	x_scale	factor to scale the x, y, z components of the vector
	y_scale	
	z_scale	

Notes:

The scale factors apply only if the isosurface variable is a vector.

Usually used with a scale factor of 0 or 1. in order to use only a component of a vector.

A special condition exists when `x_scale` = `y_scale` = `z_scale` = 0, in which case the vector magnitude is used.

The default is `x_scale` = `y_scale` = `z_scale` = 0.

Example Usage:

```
part: select_begin
 3
part: select_end
#
# Use the y_component of the vector
#
part: modify_begin
isos: component 0 1 0
part: modify_end
```

See Also:

Associated isos commands

isos: constraint

Command:

```
isos: constraint <isovolume_bound>
```

Purpose:

Set the bounding condition for an isovolume.

Parameters:

Type	Parameter	Description	
constant	isovolume_bound	low	isovolume will contain geometry with variable values <= to the value specified with "isos: min"
		high	isovolume will contain geometry with variable values >= to the value specified with "isos: max"
		band	isovolume will contain geometry with variable values >= the value specified with "isos: min" and <= the value specified with "isos: max"

Notes:

Is used only when "isos: type" is set to isovolume.

Example Usage:

```
isos: begin
isos: variable temperature
isos: type isovolume
isos: constraint band
isos: min 1.
isos: max 2.
isos: end
isos: create
```

See Also:

Associated isos commands

Command:

```
isos: create
```

Purpose:

Create a isosurface part with the currently defined attributes using the currently selected parts as parent parts.

Parameters:

none

Notes:

The isosurface part is created using the currently selected parts as parents.

The isosurface part is created using the currently set isosurface attributes.

Example Usage:

```
isos: begin
isos: variable temperature
isos: type isovolume
isos: constraint band
isos: min 1.
isos: max 2.
isos: end
isos: create
```

See Also:

Associated isos commands

isos: interactive_auto_delta

Command:

```
isos: interactive_auto_delta <delta>
```

Purpose:

Set the isosurface animation delta for use when surface is set to interactive mode via "`isos: interactive_type auto/auto_cycle`".

Parameters:

Type	Parameter	Description
float	`delta`	delta value that will be added to the isosurface value for each refresh of the screen

Notes:

The currently selected part must be an isosurface part. Not used for isovolume parts.

Example Usage:

```
part: select_begin
 3
part: select_end
isos: interactive_type auto
isos: interactive_rangemin 0.
isos: interactive_rangemax 5.
isos: interactive_auto_delta .5
```

See Also:

Associated isos commands

Command:

```
isos: interactive_rangemin <min_range>
isos: interactive_rangemax <max_range>
```

Purpose:

Set the minimum/maximum isosurface value when isosurface is in an interactive mode.

Parameters:

Type	Parameter	Description
float	min_range	minimum isosurface value when displaying interactive
	max_range	maximum isosurface value when displaying interactive

Notes:

The currently selected part must be an isosurface part. Not used for isovolume parts.

Example Usage:

```
part: select_begin
  3
 part: select_end
 isos: interactive_type auto
 isos: interactive_rangemin 0.
 isos: interactive_rangemax 5.
 isos: interactive_auto_delta .5
```

See Also:

Associated isos commands

isos: interactive_type

Command:

```
isos: interactive_type <type>
```

Purpose:

Specify the interactive setting for an isosurface.

Parameters:

Type	Parameter	Description	
constant	`type`	`off`	the isosurface is not in interactive mode
		`manual`	set the interactive state to manual such that the isosurface value can be updated via the user interface slider
		`auto`	set the interactive state to auto such that the iso value is incremented by the interactive_auto_delta value
		`auto_cycle`	same as "auto" except that when the max is reached the interactive_auto_delta value is decremented back to the min value.

Notes:

The min and max values are constrained via the `interactive_rangemin/max commands`. Only isosurfaces can be in interactive mode (i.e., isovolumes can not).

Example Usage:

```
part: select_begin
  3
part: select_end
isos: interactive_type auto
isos: interactive_rangemin 0.
isos: interactive_rangemax 5.
isos: interactive_auto_delta .5
```

See Also:

Associated isos commands

Command:

```
isos: min <min_value>
isos: max <max_value>
```

Purpose:

Set the min/max constraint values for an isovolume.

Parameters:

Type	Parameter	Description
float	min_value	minimum iso constraint
	max_value	maximum iso constraint

Notes:

Is used only when "isos: type" is set to isovolume.

Example Usage:

```
isos: begin
isos: variable temperature
isos: type isovolume
isos: constraint band
isos: min 1.
isos: max 2.
isos: end
isos: create
```

See Also:

Associated isos commands

isos: select_default

Command:

```
isos: select_default
```

Purpose:

Sets the target to the default settings for subsequent attribute modifications.

Parameters:

none

Example Usage:

```
isos: select_default
part: modify_begin
isos: variable temperature
isos: value 1.0
part: modify_end
```

See Also:

Associated isos commands

Command:

```
isos: type <iso_type>
```

Purpose:

Set the isosurface part to represent a surface or a volume.

Parameters:

Type	Parameter	Description	
constant	iso_type	isosurface	for a surface
		isovolume	for a volume

Notes:

The default is isosurface.

Example Usage:

```
isos: begin
isos: variable temperature
isos: value 1.1
isos: type isovolume
isos: constraint band
isos: min 1.
isos: max 2.
isos: end
isos: create
```

See Also:

Associated isos commands

isos: value

Command:

```
isos: value <iso_value>
```

Purpose:

Specify the isosurface value.

Parameters:

Type	Parameter	Description
float	`iso_value`	iso value For a scalar variable, the value is the scalar value at which the iso-surface (or line) will be generated. For a vector variable, the value represents the magnitude or a scaling of vector components (see Notes below).

Notes:

Is not used for isovolumes.

For a vector: If the following is defined:
`X_var` = The vector X component
`Y_var` = The vector Y component
`Z_var` = The vector Z component
`X_com` = The X vector scale factor as specified with "`isos: component`"
`Y_com` = The Y vector scale factor as specified with "`isos: component`"
`Z_com` = The Z vector scale factor as specified with "`isos: component`"

then the isosurface (or line) created represents a uniform scalar value where

```
X_var*X_com + Y_var*Y_com + Z_var*Z_com = iso_value,
```

except in the special case where `X_com = Y_com = Z_com = 0`. in which case the isosurface (or line) created represents a uniform scalar value where

```
sqrt(X_var*X_var + Y_var*Y_var + Z_var*Z_var) = iso_value.
```

Example Usage:

```
isos: begin
isos: variable temperature
isos: value 1.1
isos: type isosurface
isos: end
isos: create
```

See Also:

Associated isos commands

Command:

```
isos: variable <var_name>
```

Purpose:

Set the isosurface variable.

Parameters:

Type	Parameter	Description
string	var_name	name of an active variable

Notes:

A variable must be active before it can be used for isosurfaces.

Example Usage:

```
isos: begin
isos: variable temperature
isos: value 1.1
isos: type isosurface
isos: end
isos: create
```

See Also:

Associated isos commands

Commands in this Section:

legend

See Also:

User Manual - Section 8.2, Annot Mode

How To Create Color Legends

legend: format

Command:

```
legend: format <number_format>
```

Purpose:

Set the number format for legend labels.

Parameters:

Type	Parameter	Description
string	number_format	This can be any valid C language number format, such as:
		%.2e
		%12.5e
		%f
		%g
		%.4f

Notes:

If you are not familiar with the C language, any C language manual will have a description of valid number formats and how they affect the displayed appearance of a number.

Example Usage:

```
legend: select_palette_begin
pressure
legend: select_palette_end
legend: format %.2e
```

See Also:

Associated legend commands

Command:

```
legend: height <num>
```

Purpose:

Set the height of the legend bar

Parameters:

Type	Parameter	Description
float	num	height of the legend bar

Notes:

The height is in terms of screen coordinates, which are 0.0 to 1.0 in both x and y directions.

Example Usage:

```
legend: select_palette_begin
pressure
legend: select_palette_end
legend: location_x 2.1362e-01
legend: location_y 4.5270e-02
legend: width 5.7778e-02
legend: height 8.5655e-01
```

See Also:

Associated legend commands

legend: location_x/y

Command:

```
legend: location_x <num>
legend: location_y <num>
```

Purpose:

Set the horizontal/vertical location of the legend on the screen.

Parameters:

Type	Parameter	Description
float	num	horizontal/vertical screen position of lower left of legend bar

Notes:

The range of valid x/y screen position values is 0.0 to 1.0.

Example Usage:

```
legend: select_palette_begin
pressure
legend: select_palette_end
legend: location_x .2
legend: location_y .05
```

See Also:

Associated legend commands

Command:

```
legend: orientation <layout>
```

Purpose:

Set the legend bar to be either vertical or horizontal.

Parameters:

Type	Parameter	Description
constant	layout	vertical
		horizontal

Notes:

The normal orientation is vertical. If orientation is set to horizontal, be aware that width is now in the y-direction, height in the x-direction, and that label sizes become more important to avoid overlaps (even though it will attempt to alleviate this by alternating sides if needed).

Example Usage:

```
legend: select_palette_begin
velocity
legend: select_palette_end
legend: visible ON
legend: orientation horizontal
legend: select_palette_begin
pressure
legend: select_palette_end
legend: orientation vertical
```

See Also:

Associated legend commands

legend: select_all

Command:

```
legend: select_all
```

Purpose:

Selects all legends that are currently visible, for which other attributes will be set or modified.

Parameters:

none

Notes:

If any legends are currently visible, but not currently selected, this command will cause them to become selected. Thus, subsequent attribute modifications will apply to all visible legends.

Example Usage:

```
legend: select_all
legend: type continuous
legend: title below
```

See Also:

Associated legend commands

Command:

```
legend: select_default
```

Purpose:

Sets the current legend selection to the default, so default attributes can be modified.

Parameters:

none

Notes:

Any legend attributes set or modified after this command (and before another `legend: select_palette_begin` and `legend: select_palette_end pair`) will be applied to the "`default`" legend. The `default` legend attributes will be applied for any subsequent legends which are made active (made visible for the first time).

Example Usage:

```
legend: select_default
legend: orientation horizontal
legend: title below
```

See Also:

Associated legend commands

legend: select_palette_begin/end

Command:

```
legend: select_palette_begin
palette_name
palette_name
.
.
.
palette_name
legend: select_palette_end
```

Purpose:

Selection of legends, for which attributes will be set or modified.

Parameters:

Type	Parameter	Description
string	`palette_name`	palette name

Notes:

The legends bracketed by these two commands are the ones to which subsequent legend commands will be applied.

Example Usage:

```
legend: select_palette_begin
pressure
velocity
legend: select_palette_end
legend: visible ON
```

See Also:

Associated legend commands

Command:

```
legend: text_position <location>
```

Purpose:

Set the visibility and location of legend labels in relation to the legend color bar.

Parameters:

Type	Parameter	Description	
constant	location	left	to have labels on the left of the bar
		right	to have labels on the right of the bar
		none	to have no labels

Example Usage:

```
legend: select_palette_begin
pressure
legend: select_palette_end
legend: text_position left
legend: select_palette_begin
velocity
legend: select_palette_end
legend: text_position right
legend: select_palette_begin
temperature
legend: select_palette_end
legend: text_position none
```

See Also:

Associated legend commands

legend: text_rgb

Command:

```
legend: text_rgb <red_val> <grn_val> <blu_val>
```

Purpose:

Sets the color of the legend text for title and labels.

Parameters:

Type	Parameter	Description
float	red_val	red, green, blue contributions of the color (0.0 to 1.0)
	grn_val	
	blu_val	

Example Usage:

```
legend: select_palette_begin
velocity
legend: select_palette_end
legend: text_size 35
legend: format %.2f
legend: text_rgb 1.0000e+00 0.0000e+00 3.9608e-01
```

See Also:

Associated legend commands

Command:

```
legend: text_size <size>
```

Purpose:

Sets the size of font to use for legend title and labels.

Parameters:

Type	Parameter	Description
int	size	font size to use

Notes:

This size is a relative size based on the graphics window. It will change appropriately as the graphics window changes.

Example Usage:

```
legend: select_palette_begin
velocity
legend: select_palette_end
legend: text_size 35
legend: format %.2f
legend: text_rgb 1.0000e+00 0.0000e+00 3.9608e-01
```

See Also:

Associated legend commands

legend: title

Command:

```
legend: title <location>
```

Purpose:

Set the visibility and location of a legend title in relation to the legend color bar.

Parameters:

Type	Parameter	Description	
constant	`location`	`above`	to have the title at the top of the legend
		`below`	to have the title at the bottom of the legend
		`none`	to have no title

Example Usage:

```
legend: select_palette_begin
pressure
legend: select_palette_end
legend: type continuous
legend: title below
legend: select_palette_begin
velocity
legend: select_palette_end
legend: title none
```

See Also:

Associated legend commands

Command:

```
legend: type <bar_type>
```

Purpose:

Set the type of bar to use for color representation of legend values.

Parameters:

Type	Parameter	Description	
constant	bar_type	continuous	to have a single continuous color bar.
		discrete	to have a small color rectangle per palette level

Notes:

A `continuous` bar shows all the variations as one color changes to another, while a `discrete` bar shows the color at the levels only.

Example Usage:

```
legend: select_palette_begin
pressure
legend: select_palette_end
legend: type discrete
legend: select_palette_begin
velocity
legend: select_palette_end
legend: type continuous
```

See Also:

Associated legend commands

legend: visible

Command:

```
legend: visible <toggle>
```

Purpose:

To turn the visibility of a legend on or off.

Parameters:

Type	Parameter	Description
constant	toggle	ON
		OFF

Notes:

The command applies to the currently selected legend(s)

Example Usage:

```
legend: select_palette_begin
pressure
legend: select_palette_end
legend: visible ON
legend: select_palette_begin
velocity
legend: select_palette_end
legend: visible OFF
```

See Also:

Associated legend commands

Command:

```
legend: width <num>
```

Purpose:

Set the width of the legend bar.

Parameters:

Type	Parameter	Description
float	num	width of the legend bar

Notes:

The width is in terms of screen coordinates, which are 0.0 to 1.0 in both x and y directions.

Example Usage:

```
legend: select_palette_begin
pressure
legend: select_palette_end
legend: location_x 2.1362e-01
legend: location_y 4.5270e-02
legend: width 5.7778e-02
legend: height 8.5655e-01
```

See Also:

Associated legend commands

Commands in this Section:

line

line: arrowhead <head_type>
line: delete
line: location_x <index> <coord>
line: location_y <index> <coord>
line: modify_begin
line: modify_end
line: new_line
line: rgb <red_val> <grn_val> <blu_val>
line: select_all
line: select_begin
line: select_end
line: select_default
line: visible <toggle>
line: width <width>

See Also:

User Manual - Section 8.2, Annot Mode
How To Create Lines and Arrows

line: arrowhead

Command:

```
line: arrowhead <head_type>
```

Purpose:

Controls the display of arrowheads on the currently selected line in the graphics window.

Parameters:

Type	Parameter	Description
constant	head_type	can be any of the following values:
		none
		on_first_end
		on_second_end
		on_both_ends

Example Usage:

```
line: select_begin
1
line: select_end
line: arrowhead on_first_end
```

See Also:

Associated line commands

Command:

```
line: delete
```

Purpose:

Deletes the currently selected line in the graphics window.

Parameters:

none

Example Usage:

```
line: select_begin
1 2 3
line: select_end
line: delete
```

See Also:

Associated line commands

line: location_x/y

Command:

```
line: location_x <index> <coord>
line: location_y <index> <coord>
```

Purpose:

Specifies the x/y coordinate of an endpoint for the selected line in the graphics window.

Parameters:

Type	Parameter	Description
int	`index`	Must be either 1 or 2, depending on which endpoint is being specified.
float	`coord`	Specifies the x/y coordinate. A value of 0.0 indicates the left/bottom side of the window, while a value of 1.0 indicates the right/top side.

Example Usage:

```
line: select_begin
 1
line: select_end
line: location_x  1 0.5
line: location_y  1 0.6
line: location_x  2 0.1
line: location_y  2 0.3
```

See Also:

Associated line commands

Command:

```
line: modify_begin
line: modify_end
```

Purpose:

These optional commands can be used to demarcate a series of `line`: commands which apply to the same line or lines so that they can be executed at one time for better performance.

Parameters:

none

Example Usage:

```
line: select_begin
 1
line: select_end
line: modify_begin
line: rgb 1 0 0
line location_x 1 100
line location_y 1 20
line location_x 2 10
line location_y 2 10
line modify_end
```

See Also:

Associated line commands

line: new_line

Command:

```
line: new_line
```

Purpose:

Creates a new line in the graphics window.

Parameters:

none

Example Usage:

```
line: new_line
line: location_x 1 0.5
line: location_y 1 0.6
line: location_x 2 0.6
line: location_y 2 0.5
```

See Also:

Associated line commands

Command:

```
line: rgb <red_val> <grn_val> <blu_val>
```

Purpose:

Specifies the RGB color value of the selected line(s) .

Parameters:

Type	Parameter	Description
float	red_val	red, green, blue color components. Must be between 0.0 and 1.0
	grn_val	
	blu_val	

Example Usage:

```
line: select_begin
 1
line: select_end
line: rgb 1 0 0
```

See Also:

Associated line commands

line: select_all

Command:

```
line: select_all
```

Purpose:

Selects all of the annotation lines as the target for future `line:` commands.

Parameters:

none

Example Usage:

```
line: select_all
line: delete
```

See Also:

Associated line commands

Command:

```
line: select_begin
<index1  index2 ... indexN>
line: select_end
```

Purpose:

Selects one or more lines in the graphics window as the target for future `line:` commands.

Parameters:

Type	Parameter	Description
int	`indexN`	up to a maximum of 10 values per line

Notes:

Using these commands is equivalent to selecting lines by clicking on them in the graphics window with the CTRL key held down.

Example Usage:

```
line: select_begin
 1 2
line: select_end
line: rgb 1 0 0
```

See Also:

Associated line commands

line: select_default

Command:

```
line: select_default
```

Purpose:

Causes any following `line:` commands to affect the default settings, rather than a particular line.

Parameters:

none

Notes:

All `line:` commands following `line: select_default` will affect default settings until a new line is selected using `line: select_begin/select_end`.

Example Usage:

```
line: select_default
line: visible OFF
line: rgb 1 0 0
line: select_begin
 1
line: select_end
```

See Also:

Associated line commands

Command:

```
line: visible <toggle>
```

Purpose:

Controls the visibility of the selected lines in the graphics window.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	to have lines displayed
		OFF	to not display lines

Notes:

Lines are visible by default.

Example Usage:

```
line: select_begin
 1
line: select_end
line: visible ON
```

See Also:

Associated line commands

line: width

Command:

```
line: width  <width>
```

Purpose:

Sets the width of the currently selected line.

Parameters:

Type	Parameter	Description
int	`width`	width (in pixels) can be any value from 1 to 4

Example Usage:

```
line: select_begin
 1
line: select_end
line: width  2
```

See Also:

Associated line commands

Commands in this Section:

logo

logo: delete
logo: location_x <coord>
logo: location_y <coord>
logo: modify_begin
logo: modify_end
logo: new_logo <filename>
logo: scale <x_val> <y_val>
logo: select_all
logo: select_begin
logo: select_end
logo: select_default
logo: visible <toggle>

See Also:

User Manual - Section 8.2, Annot Mode
How To Load Custom Logos

logo: delete

Command:

```
logo: delete
```

Purpose:

Deletes the currently selected logo(s) in the graphics window.

Parameters:

none

Example Usage:

```
logo: select_begin
 1 2 3
logo: select_end
logo: delete
```

See Also:

Associated logo commands

Command:

```
logo: location_x <coord>
logo: location_y <coord>
```

Purpose:

Specifies the x/y coordinate of the selected logo in the graphics window.

Parameters:

Type	Parameter	Description
float	`coord`	A value of 0.0 indicates the left/bottom of the window, while a value of 1.0 indicates the right/top side

Notes:

The lower left corner of the logo will be positioned at this point (x,y).

Example Usage:

```
logo: select_begin
 1
logo: select_end
logo: location_x  0.5
logo: location_y  0.5
```

See Also:

Associated logo commands

logo: modify_begin/end

Command:

```
logo: modify_begin
logo: modify_end
```

Purpose:

These optional commands can be used to demarcate a series of `logo:` commands which apply to the same logo or logos, so that they can be executed at one time for better performance.

Parameters:

none

Example Usage:

```
logo: select_begin
 1
logo: select_end
logo: modify_begin
logo: rgb 1 0 0
logo: location_x 100
logo: location_y 20
logo: modify_end
```

See Also:

Associated logo commands

Command:

```
logo: new_logo  <filename>
```

Purpose:

Creates a new logo in the graphics window.

Parameters:

Type	Parameter	Description
string	filename	path to the logo file

Notes:

The logo file can be any valid XPM file. The filename does not have to include an ".xpm" extension.

Example Usage:

```
logo: new_logo /usr/tmp/mylogo.xpm
logo: location_x  0.5
logo: location_y  0.6
```

See Also:

Associated logo commands

logo: scale

Command:

```
logo: scale  <x_val> <y_val>
```

Purpose:

Scales the selected logo up or down in x and y dimensions

Parameters:

Type	Parameter	Description
float	x_val	Can be any value between 0 and infinity, with a value of one specifying no change, values less than one specifying shrinkage and values greater than one specifying expansion.
	y_val	Can be any value between 0 and infinity, with a value of one specifying no change, values less than one specifying shrinkage and values greater than one specifying expansion.

Example Usage:

```
logo: new_logo /usr/tmp/mylogo.xpm
logo: location_x  0.5
logo: location_y  0.6
logo: scale 2.5 2
```

See Also:

Associated logo commands

Command:

```
logo: select_all
```

Purpose:

Selects all of the logos in the graphics window as the target for future `logo:` commands.

Parameters:

none

Example Usage:

```
logo: select_all
logo: delete
```

See Also:

Associated logo commands

logo: select_begin/end

Command:

```
logo: select_begin
<index1  index2 …indexN>
logo: select_end
```

Purpose:

Selects one or more logos in the graphics window as the target for future `logo:` commands

Parameters:

Type	Parameter	Description
int	`indexN`	up to a maximum of 10 values per line

Notes:

Using these commands is equivalent to selecting logos by clicking on them in the graphics window with the CTRL key held down.

Example Usage:

```
logo: select_begin
 1 2
logo: select_end
logo: rgb 1 0 0
```

See Also:

Associated logo commands

Command:

```
logo: select_default
```

Purpose:

Causes any following `logo:` commands to affect the default settings, rather than a particular logo.

Parameters:

none

Notes:

All `logo:` commands following `logo: select_default` will affect default settings until a new logo is selected using `logo: select_begin`.

Example Usage:

```
logo: select_default
logo: visible OFF
logo: rgb 1 0 0
logo: select_begin
 1
logo: select_end
```

See Also:

Associated logo commands

logo: visible

Command:

```
logo: visible <toggle>
```

Purpose:

Controls the visibility of the selected logo in the graphics window.

Parameters:

Type	Parameter	Description
constant	toggle	ON
		OFF

Notes:

Logos are visible by default.

Example Usage:

```
logo: select_begin
 1
logo: select_end
logo: visible ON
```

See Also:

Associated logo commands

Commands in this Section:

material

material: begin
material: create
material: end
material: select_byname_begin
material: select_byname_end
material: select_default
material: subdivide <level>
material: type <type>

See Also:

User Manual - Section 7.18, Material Parts Create/Update

How To Create Material Parts

material: begin/end

Command:

```
material: begin
material: end
```

Purpose:

Delimit the modifications for material part attributes.

Parameters:

none

Example Usage:

```
#
# Select the parent part
#
part: select_begin
 1
part: select_end
#
material: begin
#
# create a domain type material part
#
material: type domain
material: subdivide 2
#
# use the two materials shown
#
material: select_byname_begin
matl_02
matl_03
material: select_byname_end
material: end
material: create
```

See Also:

Associated material commands

Command:

```
material: create
```

Purpose:

Create a material part with the currently defined attributes using the currently selected parts as parent parts.

Parameters:

none

Example Usage:

```
#
# Select the parent part
#
part: select_begin
 1
part: select_end
#
material: begin
#
# create a domain type material part
#
material: type domain
material: subdivide 2
#
# use the two materials shown
#
material: select_byname_begin
matl_02
matl_03
material: select_byname_end
material: end
material: create
```

See Also:

Associated material commands

material: select_byname_begin/end

Command:

```
material: select_byname_begin
material: select_byname_end
```

Purpose:

Delimit material name selection.

Parameters:

none

Example Usage:

```
#
# Select the parent part
#
part: select_begin
 1
part: select_end
#
material: begin
#
# create a domain type material part using materials matl_02 and matl_03
#
material: type domain
material: subdivide 2
#
# use the two materials shown
#
material: select_byname_begin
matl_02
matl_03
material: select_byname_end
material: end
material: create
```

See Also:

Associated material commands

Command:

```
material: select_default
```

Purpose:

Sets the target to the default settings for subsequent attribute modifications.

Parameters:

none

Example Usage:

```
material: select_default
part: modify_begin
material: type domain
part: modify_end
```

See Also:

Associated material commands

material: subdivide

Command:

```
material: subdivide <level>
```

Purpose:

Set the accuracy of the resulting material part - higher values result in higher fidelity.

Parameters:

Type	Parameter	Description	
constant	level	1	higher values result in higher accuracy
		2	
		3	

Example Usage:

```
#
# Select the parent part
#
part: select_begin
 1
part: select_end
#
material: begin
#
# create a domain type material part
#
material: type domain
material: subdivide 2
#
# use the two materials shown
#
material: select_byname_begin
matl_02
matl_03
material: select_byname_end
material: end
material: create
```

See Also:

Associated material commands

Command:

```
material: type <type>
```

Purpose:

Set the resulting material part to be a domain or interface through/between the material selected.

Parameters:

Type	Parameter	Description	
constant	type	domain	create a domain interface part
		interface	create the interfaces between the material

Example Usage:

```
#
# Select the parent part
#
part: select_begin
 1
part: select_end
#
material: begin
#
# create a domain type material part
#
material: type domain
material: subdivide 2
#
# use the two materials shown
#
material: select_byname_begin
matl_02
matl_03
material: select_byname_end
material: end
material: create
```

See Also:

Associated material commands

Commands in this Section:

message-window

message-window: clear
message-window: save <filename>

See Also:

User Manual - Section 2.13, Saving Query Text Information

message-window: clear

Command:

```
message-window: clear
```

Purpose:

Clears the EnSight Message Window dialog

Parameters:

none

Example Usage:

message-window: clear

See Also:

Associated message-window commands

Command:

```
message-window: save <filename>
```

Purpose:

Saves the contents of the EnSight Message Window dialog to a file.

Parameters:

Type	Parameter	Description
string	`filename`	filename into which the contents of the message window is to be saved.

Example Usage:

message-window: save /home/users/me/myfile.txt

See Also:

Associated message-window commands

Commands in this Section:

model

model: ijk_axis <toggle>
model: ijk_axis_scale <scale_value>
model: noderangestep_i <val1> <val2> <val3>
model: noderangestep_j <val1> <val2> <val3>
model: noderangestep_k <val1> <val2> <val3>
model: select_default

See Also:

User Manual - Section 2.1, Internal Readers

How To Read EnSight Gold Data

How To Read EnSight 6 Data

How To Read ESTET Data

How To Read PLOT3D Data

model: ijk_axis visibility

Command:

```
model: ijk_axis <toggle>
```

Purpose:

Toggles the visibility of an IJK axis attached to structured parts.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	axis visible
		`OFF`	axis invisible

Example Usage:

```
part: select_begin
 1
part: select_end
part: modify_begin
model: ijk_axis ON
part: modify_end
part: modify_begin
model: ijk_axis_scale 1.0000e+00
part: modify_end
```

See Also:

Associated model commands

Command:

```
model: ijk_axis_scale <scale_value>
```

Purpose:

Scales an IJK axis attached to structured parts.

Parameters:

Type	Parameter	Description
float	scale_value	scale factor

Example Usage:

```
part: select_begin
 1
part: select_end
part: modify_begin
model: ijk_axis ON
part: modify_end
part: modify_begin
model: ijk_axis_scale 1.0000e+00
part: modify_end
```

See Also:

Associated model commands

model: noderangestep_i/j/k

Command:

```
model: noderangestep_i <val1> <val2> <val3>
model: noderangestep_j <val1> <val2> <val3>
model: noderangestep_k <val1> <val2> <val3>
```

Purpose:

Sets the IJK range and step for a structured part.

Parameters:

Type	Parameter	Description
int	val1	minimum I, J, K coordinate
	val2	maximum I, J, K coordinate
	val3	I, J, K stride

Notes:

To speed interactive operations on structured data sets, only a portion of the model can be loaded. At a later point the entire model can be loaded.

Example Usage:

```
# partial load a structured Plot3D part
data_partbuild: data_type structured
data_partbuild: select_begin
 1
data_partbuild: select_end
data_partbuild: domain all
data_partbuild: noderange_i 1 40
data_partbuild: noderange_j 1 63
data_partbuild: noderange_k 1 45
data_partbuild: nodestep 10 1 1
data_partbuild: nodedelta 0 0 0
data_partbuild: description partial grid
data_partbuild: create
data_partbuild: end
part: select_all
part: modify_begin
# load every other i coordinate
model: noderangestep_i 1 40 2
model: noderangestep_j 1 63 1
model: noderangestep_k 1 45 1
part: modify_end
part: modify_begin
# load all i coordinates
model: noderangestep_i 1 80 1
model: noderangestep_j 1 63 1
model: noderangestep_k 1 45 1
part: modify_end
```

See Also:

Associated model commands

Command:

```
model: select_default
```

Purpose:

Sets default attribute values for model parts

Parameters:

none

Notes:

model commands that follow this one set default values for new model parts loaded at a latter time during the current session.

Example Usage:

```
model: select_default
part: modify_begin
part: bounding_rep points
part: modify_end
part: modify_begin
model: ijk_axis ON
part: modify_end
data: binary_files_are big_endian
data: format plot3d
data: plot3diblank OFF
data: plot3dmulti_zone OFF
data: plot3dread_as c_binary
data: plot3ddimension 3d
data: path /usr/local/bin/ensight76/data/e7data/plot3d
data: geometry shuttle.xyz
data: result shuttle.q
data_partbuild: begin
data: read
data_partbuild: data_type structured
data_partbuild: select_begin
 1
data_partbuild: select_end
data_partbuild: domain all
data_partbuild: noderange_i 1 80
data_partbuild: noderange_j 1 63
data_partbuild: noderange_k 1 45
data_partbuild: nodestep 1 1 1
data_partbuild: nodedelta 0 0 0
data_partbuild: description
data_partbuild: create
data_partbuild: end
view_transf: rotate -2.848098 21.765417 0.000000
view_transf: function global
```

See Also:

Associated model commands

Commands in this Section:

part

part: reduce_polygons_factor <factor>
part: ref_frame <frame#>
part: select_all
part: select_all_case
part: select_all_default
part: select_begin
part: select_end
part: select_default
part: select_lastcreatedpart
part: select_lastonecreated
part: select_mainpartlist_all
part: select_partexpr_begin
part: select_partexpr_end
part: select_partname_begin
part: select_partname_end
part: shading <type>
part: symmetry_angle <angle>
part: symmetry_axis <axis>
part: symmetry_mirror_x <toggle>
part: symmetry_mirror_y <toggle>
part: symmetry_mirror_z <toggle>
part: symmetry_mirror_xy <toggle>
part: symmetry_mirror_yz <toggle>
part: symmetry_mirror_xz <toggle>
part: symmetry_mirror_xyz <toggle>
part: symmetry_rinstances <num>
part: symmetry_type <sym_type>
part: ungroup
part: updtparts <parent>
part: viewport_vis <toggle> <vport1>...<vportN>
part: visibility_elt <toggle>
part: visibility_line <toggle>
part: visibility_node <toggle>
part: visible <toggle>

See Also:

User Manual - Section 8.1, Part Mode
User Manual - Chapter 3, Parts
How To Set Attributes
How To Display Displacements
How To Change Color
How To Copy a Part
How To Group Parts
How To Merge Parts
How To Extract Part Representation
How To Delete a Part
How To Change the Visual Representation
How To Display Labels

How To Set Transparency
How To Set Symmetry
How To Select Parts

part: active

Command:

```
part: active <toggle>
```

Purpose:

Activates or deactivates the currently selected part(s).

Parameters:

Type	Parameter	Description
constant	toggle	ON
		OFF

Notes:

When a part is deactivated, the data for that part will not be updated from the server on any future timestep changes, until the part is activated again

Example Usage:

```
part: select_begin
1
part: select_end
part: modify_begin
part: active OFF
part: modify_end
```

See Also:

Associated part commands

Command:

```
part: allow_default_editing <toggle>
```

Purpose:

Allow or disallow editing of part defaults.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	allow editing of part defaults
		OFF	do not allow editing of part defaults

Notes:

This is a command for preferences and is written out in the ensight_allparts_prefs.def file.

If the user edits a part attribute when no parts are selected the default is normally edited. With this toggle OFF the user will no be able to edit part defaults.

Example Usage:

```
part: allow_default_editing OFF
```

See Also:

Associated part commands

part: bounding_rep

Command:

```
part: bounding_rep  <type>
```

Purpose:

Specifies the type of fast-display representation used to for the currently selected part(s).

Parameters:

Type	Parameter	Description	
constant	type	Can have any of the following values:	
		OFF	specifies display according to the currently specified Element Representation
		box	specifies a bounding (Cartesian extent) box of all part elements (default)
		points	specifies point cloud representation of the part
		sparse_model	specifies sparse model representation of the part

Example Usage:

```
part: select_begin
1
part: select_end
part: modify_begin
part: bounding_rep sparse_model
part: modify_end
```

See Also:

Associated part commands

Command:

```
part: clip <toggle>
```

Purpose:

Turns auxiliary clipping on or off for the currently selected part(s).

Parameters:

Type	Parameter	Description
constant	`toggle`	`ON`
		`OFF`

Notes:

The global toggle for auxilliary clipping must be on - See `view: clipping`

Example Usage:

```
view: clipping ON
part: select_begin
1
part: select_end
part: modify_begin
part: clip ON
part: modify_end
```

See Also:

Associated part commands

part: colorby_palette

Command:

```
part: colorby_palette <varname>
```

Purpose:

Colors the currently selected part(s) based on the value of the specified variable at each node or element (depending on whether data is node or element based)

Parameters:

Type	Parameter	Description
string	varname	the name of an existing variable (or the word “none”)

Example Usage:

```
part: select_begin
1
part: select_end
part: modify_begin
part: colorby_palette pressure
part: modify_end
```

See Also:

Associated part commands

Command:

```
part: colorby_rgb <red_val> <grn_val> <blu_val>
```

Purpose:

Colors the currently selected part(s) using the specified rgb value.

Parameters:

Type	Parameter	Description
float	red_val	red, green, blue color components (must be between 0.0 and 1.0)
	grn_val	
	blu_val	

Example Usage:

```
part: select_begin
1
part: select_end
part: modify_begin
part: colorby_rgb 1.0  0.4  0.0
part: modify_end
```

See Also:

Associated part commands

part: colorby_selected_var

Command:

```
part: colorby_selected_var
```

Purpose:

Colors a part by the palette tied to the variable selected in the main variable list.

Parameters:

None.

Notes:

Used with the HUM to tie parts selected to variables selected.

Example Usage:

```
variables: get_mainvarlist_select
part: get_mainpartlist_select
part: modify_begin
part: colorby_selected_var
part: modify_end
```

See Also:

Associated part commands

Command:

```
part: copy
```

Purpose:

Creates a copy of the currently selected part(s).

Parameters:

None.

Notes:

The Copy operation creates a dependent copy of the original part.

The copy is created on the client and its existence is not known to the EnSight Server process. A copy shares geometric data and variable data with the original part.

Example Usage:

```
part: select_begin
1
part: select_end
part: copy
```

See Also:

Associated part commands

part: create_assign_viewport

Command:

```
part: create_assign_viewport
```

Purpose:

Creates a new viewport and assigns the currently selected part(s) to it.

Parameters:

None.

Example Usage:

```
part: select_begin
3 5
part: select_end
part: create_assign_viewport
```

See Also:

Associated part commands

Command:

```
part: create_assign_viewports
```

Purpose:

Creates a new viewport for each of the currently selected part(s) and assigns one part to each viewport.

Parameters:

None.

Example Usage:

```
part: select_begin
3 5
part: select_end
part: create_assign_viewports
```

See Also:

Associated part commands

part: delete

Command:

```
part: delete
```

Purpose:

Deletes the currently selected part(s).

Parameters:

None

Example Usage:

```
part: select_begin
1
part: select_end
part: delete
```

See Also:

Associated part commands

Command:

```
part: description <desc>
```

Purpose:

Specifies a description string for the currently selected part. This string will appear in lists that include the part.

Parameters:

Type	Parameter	Description
string	desc	name that will show up in the user interface; can be up to 50 characters in length

Example Usage:

```
part: select_begin
1
part: select_end
part: modify_begin
part: description compressor inlet
part: modify_end
```

See Also:

Associated part commands

part: displace_factor

Command:

```
part: displace_factor <disp_factor>
```

Purpose:

Specifies the displacement factor to be applied to the source variable to calculate displacement for the currently selected part(s).

Parameters:

Type	Parameter	Description
float	`disp_factor`	the scaling factor that will be applied to the displacement values.

Notes:

The displacement factor can be changed before or after the `part: displace_by` command is issued.

Example Usage:

```
part: select_begin
1
part: select_end
part: modify_begin
part: displace_factor 0.9
part: displace_by pressure
part: modify_end
```

See Also:

Associated part commands

Command:

```
part: displace_by <variable_name>
```

Purpose:

Specifies the variable to be used to displace the nodes and elements of the currently selected part(s).

Parameters:

Type	Parameter	Description
string	variable_name	name of an activated variable

Notes:

The displacement factor can be changed before or after the `part: displace_by` command is issued.

Example Usage:

```
part: select_begin
1
part: select_end
part: modify_begin
part: displace_factor 0.9
part: displace_by pressure
part: modify_end
```

See Also:

Associated part commands

part: elt_feature_angle

Command:

```
part: elt_feature_angle <angle>
```

Purpose:

Specifies the angle to be used in feature angle element representation for the currently selected part(s).

Parameters:

Type	Parameter	Description
float	angle	angle between adjacent polygons, above which the edge will be drawn, below which the edge will not be drawn. the range is 0.0 to 180.0 degrees

Example Usage:

```
part: select_begin
1
part: select_end
part: modify_begin
part: elt_representation  feature_angle
part: elt_feature_angle  20.0
part: modify_end
```

See Also:

Associated part commands

Command:

```
part: elt_representation <type>
```

Purpose:

Specifies the type of element representation used for the currently selected part(s).

Parameters:

Type	Parameter	Description	
constant	type	can have any of the following values:	
		3D_border_2D_full	represents the Part's 3D elements in Border representation, the Part's 1 and 2D elements in Full representation. The outside surfaces of the Part are displayed along with all bar elements
		border	represents the Part's 3D elements with 2D elements corresponding to unshared element faces, the Part's 2D elements with 1D elements corresponding to the unshared edges, and the Part's 1D elements as 1D elements. The outside faces and edges of the Part's elements are displayed.
		feature angle	first runs the 3D border, 2D full representation to get a list of 1 and 2D elements. The 1D elements and all non-shared 2D edges will be shown, but only the shared edges above the Angle value will be shown. The result consists of 1D elements visualizing the sharp edges of the Part.
		full	represents all faces of the part's 3D elements, and all the 1 and 2D elements
		not_loaded	part exists on the server, but is not loaded on the client. Not loaded parts may be used as parent parts, but do not exist on the client
		bounding_box	represents all part elements as a bounding box surrounding the Cartesian extent of the elements of the part

Example Usage:

```
part: select_begin
1
part: select_end
part: modify_begin
part: elt_representation  full
part: modify_end
```

See Also:

Associated part commands

part: elt_shrink_factor

Command:

```
part: elt_shrink_factor <scale_factor>
```

Purpose:

Specifies a scaling factor (fraction) by which to shrink every element toward its centroid.

Parameters:

Type	Parameter	Description
float	`scale_factor`	shrinkage factor which must be within the range of 0.0 to 1.0. The default is 0.0 for no shrinkage. If set to 1.0, elements will completely disappear.

Example Usage:

```
part: select_begin
1
part: select_end
part: modify_begin
part: elt_shrink_factor 0.5
part: modify_end
```

See Also:

Associated part commands

Command:

```
part: entity_label_elt <toggle>
```

Purpose:

Turns element labeling on or off for the currently selected part(s).

Parameters:

Type	Parameter	Description
constant	toggle	ON
		OFF

Example Usage:

```
part: select_begin
1
part: select_end
part: modify_begin
part: entity_label_elt ON
part: modify_end
```

See Also:

Associated part commands

part: entity_label_node

Command:

```
part: entity_label_node <toggle>
```

Purpose:

Turns node labeling on or off for the currently selected part(s).

Parameters:

Type	Parameter	Description
constant	toggle	ON
		OFF

Example Usage:

```
part: select_begin
1
part: select_end
part: modify_begin
part: entity_label_node ON
part: modify_end
```

See Also:

Associated part commands

Command:

```
part: extract
```

Purpose:

Creates an extract of the currently selected part(s).

Parameters:

None.

Notes:

Extracts a new part from the selected part(s) using the part representation in effect at the time (full, border, or feature). If more than one part is selected, then they are joined into a single part which is assigned to Frame 0.

The `part: extract` command creates a new part using the geometry of the current representation (what you see is what you get). Extracted parts which are in Full Representation are actual copies of the original, but extracted parts which are in Border Representation are only the shell or boundary of the original.

Example Usage:

```
part: select_begin
1
part: select_end
part: extract
```

See Also:

Associated part commands

part: fill_pattern

Command:

```
part: fill_pattern <pattern_num>
```

Purpose:

Specifies a fill pattern of variable transparency to be used on the surfaces of the currently selected part(s).

Parameters:

Type	Parameter	Description
integer value	`pattern_num`	must be within the range of 0 to 3. transparency value of 0 specifies an opaque surface, while a value of 3 specifies a sparse (transparent) fill pattern

Notes:

This command provides a means of simulating surface transparency which requires less computation than true transparency. Part opaqueness (controlled by the `part: opaqueness` command) must be 1.0 in order for `part: fill_pattern` to take effect.

Example Usage:

```
view: hidden_surface ON
part: select_begin
1
part: select_end
part: modify_begin
part: hidden_surface ON
part: fill_pattern  2
part: modify_end
```

See Also:

Associated part commands

Command:

```
part: get_mainpartlist_select
```

Purpose:

Causes the parts highlighted in the main part list to become the currently selected parts.

Parameters:

None.

Example Usage:

```
part: get_mainpartlist_select
part: modify_begin
part: mirror_original OFF
part: colorby_rgb 1 0 0
part: modify_end
```

See Also:

Associated part commands

part: group

Command:

```
part: group <groupname>
```

Purpose:

Groups the currently selected parts into a group with the specified name.

Parameters:

Type	Parameter	Description
string value	`groupname`	name of the group

Example Usage:

```
part: select_begin
3 5
part: select_end
part: group yinyang
```

See Also:

Associated part commands

Command:

```
part: hidden_line <toggle>
```

Purpose:

Causes lines which are behind any portion of the currently selected part(s) to not be drawn.

Parameters:

Type	Parameter	Description
constant	toggle	ON
		OFF

Notes:

The `part: hidden_line` command also causes the visible drawing lines to be rendered on shaded surfaces of the currently selected part(s). `View: hidden_line ON` must be issued in order for `part: hidden_line ON` to take effect.

Example Usage:

```
view: hidden_line ON
part: select_begin
1
part: select_end
part: modify_begin
part: hidden_line ON
part: modify_end
```

See Also:

Associated part commands

part: hidden_surface

Command:

```
part: hidden_surface <toggle>
```

Purpose:

Causes the currently selected part(s) to be drawn with shaded surfaces.

Parameters:

Type	Parameter	Description
constant	toggle	ON
		OFF

Notes:

`View: hidden_surface ON` must be issued in order for `part: hidden_surface ON` to take effect.

Example Usage:

```
view: hidden_surface ON
part: select_begin
1
part: select_end
part: modify_begin
part: hidden_surface ON
part: modify_end
```

See Also:

Associated part commands

Command:

```
part: light_diff <diffuse>
```

Purpose:

Specifies the lighting diffusion value used in shading the surfaces of the currently selected part(s).

Parameters:

Type	Parameter	Description
float	diffuse	diffuse light value, must be within the range of 0.0 to 1.0

Example Usage:

```
part: select_begin
1
part: select_end
part: modify_begin
part: light_diff  0.9
part: modify_end
```

See Also:

Associated part commands

part: light_h_int

Command:

```
part: light_h_int <intensity>
```

Purpose:

Specifies the highlight intensity value used in shading the surfaces of the currently selected part(s).

Parameters:

Type	Parameter	Description
float	`intensity`	highlight intensity, must be within the range of 0.0 to 1.0

Example Usage:

```
part: select_begin
1
part: select_end
part: modify_begin
part: light_h_int  0.5
part: modify_end
```

See Also:

Associated part commands

Command:

```
part: light_shin <shiny>
```

Purpose:

Specifies the "shinyness" value used in shading the surfaces of the currently selected part(s).

Parameters:

Type	Parameter	Description
float	`shiny`	“shinyness” value, must be within the range of 1.0 to 100.0 . Lower values specify less shine

Notes:

Highlight intensity (`part: light_h_int`) must be non-zero for shinyness to have any effect.

Example Usage:

```
part: select_begin
1
part: select_end
part: modify_begin
part: light_h_int  0.9
part: light_shin  50.0
part: modify_end
```

See Also:

Associated part commands

part: line_style

Command:

```
part: line_style  <style>
```

Purpose:

Specifies the line style used to draw the currently selected part(s).

Parameters:

Type	Parameter	Description
constant	style	can have any of the following values:
		solid
		dotted
		dot-dash

Example Usage:

```
part: select_begin
1
part: select_end
part: modify_begin
part: line_style dotted
part: modify_end
```

See Also:

Associated part commands

Command:

```
part: line_width <width>
```

Purpose:

Specifies the line width in pixels used to draw the currently selected part(s).

Parameters:

Type	Parameter	Description
int	width	pixel width of line; must be within the range of 1 to 4.

Example Usage:

```
part: select_begin
1
part: select_end
part: modify_begin
part: line_width  2
part: modify_end
```

See Also:

Associated part commands

part: merge

Command:

```
part: merge
```

Purpose:

Merges the currently selected parts into a single part.

Parameters:

None.

Example Usage:

```
part: select_begin
3 5
part: select_end
part: merge
```

See Also:

Associated part commands

Command:

```
part: mirror_original <toggle>
```

Purpose:

Turns visibility of the non-mirrored currently selected parts on or off.

Parameters:

Type	Parameter	Description
constant	toggle	ON
		OFF

Notes:

This command is normally used to remove the original image of parts which have been mirrored in one or more dimensions

Example Usage:

```
part: select_begin
1
part: select_end
part: modify_begin
part: mirror_z ON
part: mirror_original OFF
part: modify_end
```

See Also:

Associated part commands

part: mirror_x/y/z/xy/yz/xz/xyz

Command:

```
part: mirror_x <toggle>
part: mirror_y <toggle>
part: mirror_z <toggle>
part: mirror_xy <toggle>
part: mirror_yz <toggle>
part: mirror_xz <toggle>
part: mirror_xyz <toggle>
```

Purpose:

Set the flag for mirror images of currently selected parts in the x / y / z / xy / yz / xz / xyz directions.
(Note: These commands have been superceded by: part: symmetry_mirror_ ...)

Parameters:

Type	Parameter	Description
constant	`toggle`	`ON`
		`OFF`

Notes:

The part is reflected about the frame that it is assigned to (`part: ref_frame`)

Example Usage:

```
part: select_begin
1
part: select_end
part: modify_begin
part: mirror_x ON
part: mirror_y ON
part: mirror_xy ON
part: mirror_xyz ON
part: modify_end
```

See Also:

Associated part commands

Command:

```
part: modify_begin
...
part: modify_end
```

Purpose:

These commands are used to demarcate part attribute changing commands. The attributes within them will be changed when the modify_end is reached.

Parameters:

None

Example Usage:

```
part: select_begin
1
part: select_end
part: modify_begin
part: colorby_rgb 1 0 0
part: visibility_node ON
part: node_type sphere
part: modify_end
```

See Also:

Associated part commands

part: node_detail

Command:

```
part: node_detail <level>
```

Purpose:

Sets the detail level for drawing spherical nodes in the currently selected part(s).

Parameters:

Type	Parameter	Description
int	`level`	detail level for the spheres used to represent the nodes; must be within the range of 2 to 10.

Notes:

Higher values of level result in more polygons being used to draw the spheres.

Example Usage:

```
part: select_begin
1
part: select_end
part: modify_begin
part: node_type sphere
part: node_detail  5
part: modify_end
```

See Also:

Associated part commands

Command:

```
part: node_scale <scale>
```

Purpose:

Sets the scale factor used to determine node size for the currently selected part(s).

Parameters:

Type	Parameter	Description
float	`scale`	scaling factor to be applied to the node cross or sphere representation; can be any valid floating point value

Example Usage:

```
part: select_begin
1
part: select_end
part: modify_begin
part: node_scale 1.5
part: modify_end
```

See Also:

Associated part commands

part: node_sizeby

Command:

```
part: node_sizeby  <source_type>
```

Purpose:

Selects the source for calculating node size in the currently selected part(s).

Parameters:

Type	Parameter	Description	
constant	source_type	Can have any of the following values:	
		constant	size will be constant, controlled by the scale factor
		scalar	size will be controlled by a scalar variable and the scale factor
		vector_mag	size will be controlled by the magnitude of a vector variable and the scale factor
		vector_xcomp	size will be controlled by the x component of a vector variable and the scale factor
		vector_ycomp	size will be controlled by the y component of a vector variable and the scale factor
		vector_zcomp	size will be controlled by the z component of a vector variable and the scale factor

Example Usage:

```
part: select_begin
1
part: select_end
part: modify_begin
part: node_type sphere
part: node_sizeby vector_mag
part: node_variable velocity
part: modify_end
```

See Also:

Associated part commands

Command:

```
part: node_type <type>
```

Purpose:

Selects the node representation mode for the currently selected part(s).

Parameters:

Type	Parameter	Description
constant	type	dot
		cross
		sphere

Example Usage:

```
part: select_begin
1
part: select_end
part: modify_begin
part: node_type sphere
part: modify_end
```

See Also:

Associated part commands

part: node_variable

Command:

```
part: node_variable <variable_name>
```

Purpose:

Specifies the variable to be used in calculating the size of spherical nodes in the currently selected part(s).

Parameters:

Type	Parameter	Description
string	`variable_name`	the name of an activated variable

Example Usage:

```
part: select_begin
1
part: select_end
part: modify_begin
part: node_type sphere
part: node_sizeby vector_mag
part: node_variable velocity
part: modify_end
```

See Also:

Associated part commands

Command:

```
part: opaqueness <value>
```

Purpose:

Specifies the opaqueness of the currently selected part(s).

Parameters:

Type	Parameter	Description
float	value	opaqueness; must be within the range of 0.0 to 1.0

Notes:

A value of 1.0 is fully opaque.

Example Usage:

```
part: select_begin
1
part: select_end
part: modify_begin
part: opaqueness  0.4
part: modify_end
```

See Also:

Associated part commands

part: reduce_polygons

Command:

```
part: reduce_polygons <toggle>
```

Purpose:

Reduce the number of polygons for display.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	turn on polygon reduction
		OFF	turn off polygon reduction

Example Usage:

```
part: select_begin
1
part: select_end
part: modify_begin
part: reduce_polygons_factor 1
part: reduce_polygons ON
part: modify_end
```

See Also:

Associated part commands

Command:

```
part: reduce_polygons_factor <factor>
```

Purpose:

Set the polygon reduction factor.

Parameters:

Type	Parameter	Description
int	`factor`	0 (less) to 10 (more) controlling the number of polygons

Notes:

Used only if reduce_polygons is ON

If no polygons were removed during the process the original geometry will be shown.

Example Usage:

```
part: select_begin
1
part: select_end
part: modify_begin
part: reduce_polygons_factor 1
part: reduce_polygons ON
part: modify_end
```

See Also:

Associated part commands

part: ref_frame

Command:

```
part: ref_frame <frame#>
```

Purpose:

Assigns the currently selected part(s) to the specified reference frame.

Parameters:

Type	Parameter	Description
int	`frame#`	an existing reference frame number

Example Usage:

```
part: select_begin
1 2 5
part: select_end
part: modify_begin
part: ref_frame 2
part: modify_end
```

See Also:

Associated part commands

Command:

```
part: select_all
```

Purpose:

Selects all parts as the target for future `part:` commands.

Parameters:

None.

Example Usage:

```
part: select_all
part: modify_begin
part: colorby_rgb 1 0 0
part: modify_end
```

See Also:

Associated part commands

part: select_all_case

Command:

```
part: select_all_case
```

Purpose:

Selects all parts in the current case as the target for future part: commands.

Parameters:

None.

Example Usage:

```
part: select_all_case
part: modify_begin
part: colorby_rgb 1 0 0
part: modify_end
```

See Also:

Associated part commands

Command:

```
part: select_all_default
```

Purpose:

Causes any following part editing commands to affect the default settings for all parts of any type.

Parameters:

None

Notes:

All part: commands following `part: select_all_default` will affect default settings until a new part is selected using `part: select_begin/end`.

Example Usage:

```
part: select_all_default
part: modify_begin
part: mirror_original OFF
part: colorby_rgb 1 0 0
part: modify_end
```

See Also:

Associated part commands

part: select_begin/end

Command:

```
part: select_begin
<pn1  pn2 ...pnN>
part: select_end
```

Purpose:

Selects one or more parts as the target for future `part:` commands

Parameters:

Type	Parameter	Description
int	`pnN`	part number(s). Up to a maximum of 10 per line, but can use as many lines as needed.

Example Usage:

```
part: select_begin
1 2
part: select_end
part: modify_begin
part: colorby_rgb 1 0 0
part: modify_end
```

See Also:

Associated part commands

Command:

```
part: select_default
```

Purpose:

Causes any following part editing commands to affect the default settings for all parts belonging to the part type of the next edit command.

Parameters:

None

Notes:

All part editing commands following `part: select_default` will affect default settings until a new part is selected using `part: select_begin/select_end`.

Example Usage:

```
part: select_default
part: modify_begin
part: mirror_original OFF
part: colorby_rgb 1 0 0
part: modify_end
```

See Also:

Associated part commands

part: select_lastcreatedpart

Command:

```
part: select_lastcreatedpart
```

Purpose:

Selects all of the parts created in the last part-creating operation.

Parameters:

None.

Example Usage:

```
part: select_lastcreatedpart
part: modify_begin
part: colorby_rgb 1 0 0
part: modify_end
```

See Also:

Associated part commands

Command:

```
part: select_lastonecreated
```

Purpose:

Selects the last single part from the last part-creating operation.

Parameters:

None.

Example Usage:

```
part: select_lastonecreated
part: modify_begin
part: colorby_rgb 1 0 0
part: modify_end
```

See Also:

Associated part commands

part: select_mainpartlist_all

Command:

```
part: select_mainpartlist_all
```

Purpose:

Causes all parts in the main part list to be highlighted. Note that this does NOT cause the parts to be selected as the target for future `part:` commands.

Parameters:

None.

Notes:

To make the highlighted parts the currently selected parts, use `part: get_mainpartlist_select` or `part: select_all`.

Example Usage:

```
part: select_mainpartlist_all
part: get_mainpartlist_select
part: modify_begin
part: colorby_rgb 1 0 0
part: modify_end
```

See Also:

Associated part commands

Command:

```
part: select_partexpr_begin
<expr1> <expr2> ...<exprN>
part: select_partexpr_end
```

Purpose:

Selects parts using an expression.

Parameters:

Type	Parameter	Description
string	exprN	part expression (partial name with asterisk(s))

Notes:

Is used in command files only, i.e., EnSight does not generate these commands

Example Usage:

```
part: select_partexpr_begin
flui*
soli*
part: select_partexpr_end
part: modify_begin
part: reduce_polygons_factor 1
part: reduce_polygons ON
part: modify_end
```

See Also:

Associated part commands

part: select_partname_begin/end

Command:

```
part: select_partname_begin
<name1> <name2> ...<nameN>
part: select_partname_end
```

Purpose:

Selects parts by name.

Parameters:

Type	Parameter	Description
string	nameN	part name (as opposed to number)

Example Usage:

```
part: select_partname_begin
“impeller”
“main chamber”
“exhaust port”
part: select_partname_end
part: modify_begin
part: colorby_rgb 1 0 0
part: modify_end
```

See Also:

Associated part commands

Command:

```
part: shading <type>
```

Purpose:

Specifies the type of surface shading used on the currently selected part(s).

Parameters:

Type	Parameter	Description	
constant	type	Can have any of the following values:	
		flat	each element is a constant shade, no variation across it.
		gouraud	shading varies across each element, using Gouraud method, but does not have to match its neighbors.
		smooth	shading varies across each element, but additionally will match its neighbors at boundaries.

Example Usage:

```
view: hidden_surface ON
part: select_begin
1
part: select_end
part: modify_begin
part: hidden_surface ON
part: shading gouraud
part: modify_end
```

See Also:

Associated part commands

part: symmetry_angle

Command:

```
part: symmetry_angle  <angle>
```

Purpose:

Specifies the rotational symmetry angle.

Parameters:

Type	Parameter	Description
float	`angle`	The angle (in degrees) for the rotation

Notes:

Used only when part: symmetry_type is set to rotational.

Example Usage:

```
part: select_begin
 1
part: select_end
part: modify_begin
#
# turn on rotational symmetry about the x axis
# use 4 instances rotated 90 degrees
#
part: symmetry_type rotational
part: symmetry_axis x
part: symmetry_rinstances 4
part: symmetry_angle 90.
part: modify_end
```

See Also:

Associated part commands

Command:

```
part: symmetry_axis  <axis>
```

Purpose:

Specifies the rotational symmetry axis.

Parameters:

Type	Parameter	Description	
constant	axis	x	rotation about the x axis
		y	rotation about the y axis
		z	rotation about the z axis

Notes:

Used only when part: symmetry_type is set to rotational.

The axis is in reference to the part's coordinate frame.

Example Usage:

```
part: select_begin
 1
part: select_end
part: modify_begin
#
# turn on rotational symmetry about the x axis
# use 4 instances rotated 90 degrees
#
part: symmetry_type rotational
part: symmetry_axis x
part: symmetry_rinstances 4
part: symmetry_angle 90.
part: modify_end
```

See Also:

Associated part commands

part: symmetry_mirror_x/y/z/xy/yz/xz/xyz

Command:

```
part: symmetry_mirror_x <toggle>
part: symmetry_mirror_y <toggle>
part: symmetry_mirror_z <toggle>
part: symmetry_mirror_xy <toggle>
part: symmetry_mirror_yz <toggle>
part: symmetry_mirror_xz <toggle>
part: symmetry_mirror_xyz <toggle>
```

Purpose:

Set the flag for visual symmetry mirror images of currently selected parts in the x / y / z / xy / yz / xz / xyz directions.

Parameters:

Type	Parameter	Description
constant	`toggle`	`ON`
		`OFF`

Notes:

The part is reflected about the frame that it is assigned to (`part: ref_frame`)

Example Usage:

```
part: select_begin
1
part: select_end
part: modify_begin
part: symmetry_mirror_x ON
part: symmetry_mirror_y ON
part: symmetry_mirror_xy ON
part: symmetry_mirror_xyz ON
part: modify_end
```

See Also:

Associated part commands

Command:

```
part: symmetry_rinstances  <num>
```

Purpose:

Specifies the rotational instances.

Parameters:

Type	Parameter	Description
int	num	The number of rotational instances

Notes:

Used only when part: symmetry_type is set to rotational.

Example Usage:

```
part: select_begin
 1
part: select_end
part: modify_begin
#
# turn on rotational symmetry about the x axis
# use 4 instances rotated 90 degrees
#
part: symmetry_type rotational
part: symmetry_axis x
part: symmetry_rinstances 4
part: symmetry_angle 90.
part: modify_end
```

See Also:

Associated part commands

part: symmetry_type

Command:

```
part: symmetry_type  <sym_type>
```

Purpose:

Turn on symmetry display.

Parameters:

Type	Parameter	Description	
constant	sym_type	mirror	turn on mirror symmetry
		rotational	turn on rotational symmetry
		none	turn off symmetry

Example Usage:

```
part: select_begin
 1
part: select_end
part: modify_begin
#
# turn on rotational symmetry about the x axis
# use 4 instances rotated 90 degrees
#
part: symmetry_type rotational
part: symmetry_axis x
part: symmetry_rinstances 4
part: symmetry_angle 90.
part: modify_end
```

See Also:

Associated part commands

Command:

```
part: ungroup
```

Purpose:

Ungroups the currently selected (grouped) parts.

Parameters:

None.

Example Usage:

```
part: select_begin
3 5
part: select_end
part: ungroup
```

See Also:

Associated part commands

part: updtparts

Command:

```
part: updtparts <parent>
```

Purpose:

Updates the parent part for the currently selected part(s)

Parameters:

Type	Parameter	Description
int	`parent`	part number of the new parent(s)

Example Usage:

```
part: select_begin
3 5
part: select_end
part: updtparts  2
```

See Also:

Associated part commands

Command:

```
part: viewport_vis <toggle> <vport1>...<vportN>
```

Purpose:

Controls visibility of the currently selected part(s) in the specified viewport(s).

Parameters:

Type	Parameter	Description
constant	toggle	ON
		OFF
	vportN	viewport numbers in which to modify part visibility

Example Usage:

```
part: select_begin
1 2 5
part: select_end
part: modify_begin
part: viewport_vis OFF 1 3
part: modify_end
```

See Also:

Associated part commands

part: visibility_elt

Command:

```
part: visibility_elt <toggle>
```

Purpose:

Turns element visibility on or off for the currently selected part(s).

Parameters:

Type	Parameter	Description
constant	toggle	ON
		OFF

Example Usage:

```
part: select_begin
1
part: select_end
part: modify_begin
part: visibility_elt  OFF
part: modify_end
```

See Also:

Associated part commands

Command:

```
part: visibility_line <toggle>
```

Purpose:

Turns line visibility on or off for the currently selected part(s).

Parameters:

Type	Parameter	Description
constant	toggle	ON
		OFF

Notes:

"Line" refers to the 1D bar elements.

Example Usage:

```
part: select_begin
1
part: select_end
part: modify_begin
part: visibility_line OFF
part: modify_end
```

See Also:

Associated part commands

part: visibility_node

Command:

```
part: visibility_node <toggle>
```

Purpose:

Turns node visibility on or off for the currently selected part(s).

Parameters:

Type	Parameter	Description
constant	toggle	ON
		OFF

Example Usage:

```
part: select_begin
1
part: select_end
part: modify_begin
part: visibility_node ON
part: modify_end
```

See Also:

Associated part commands

Command:

```
part: visible <toggle>
```

Purpose:

Controls the visibility of the currently selected part(s).

Parameters:

Type	Parameter	Description
constant	toggle	ON
		OFF

Example Usage:

```
part: select_begin
1 2 5
part: select_end
part: modify_begin
part: visible OFF
part: modify_end
```

See Also:

Associated part commands

Commands in this Section:

plot

plot: origin_x <coord>
plot: origin_y <coord>
plot: plot_title <title>
plot: select_all
plot: select_begin
plot: select_end
plot: select_default
plot: subgrid_linetype <line_type>
plot: subgrid_linewidth <width>
plot: subgrid_rgb <red_val> <grn_val> <blu_val>
plot: title_rgb <red_val> <grn_val> <blu_val>
plot: title_textsize <size>
plot: visible <toggle>
plot: width <width>

See Also:

User Manual - Section 8.3, Plot Mode

How To Change Plot Attributes

Command:

```
plot: auto_scale <toggle>
```

Purpose:

Turns auto scaling on or off for the currently selected plot(s).

Parameters:

Type	Parameter	Description
constant	toggle	ON (default)
		OFF

Notes:

With auto scaling ON, the x and y min/max and the # of tick marks will be rounded based on user supplied attributes.

Example Usage:

```
plot: select_begin
 1
plot: select_end
plot: auto_scale ON
```

See Also:

Associated plot commands

plot: axis_linewidth

Command:

```
plot: axis_linewidth  <width>
```

Purpose:

Sets the axis line width of the currently selected plot(s).

Parameters:

Type	Parameter	Description
int	`width`	specifies the width in pixels and can be any value from 1 to 4

Example Usage:

```
plot: select_begin
 1
plot: select_end
plot: axis_linewidth  2
```

See Also:

Associated plot commands

Command:

```
plot: axis_rgb <red_val> <grn_val> <blu_val>
```

Purpose:

Specifies the RGB color value of the axis for the currently selected plot(s).

Parameters:

Type	Parameter	Description
float	red_val	red, green, blue color components, must be between 0.0 and 1.0
	grn_val	
	blu_val	

Example Usage:

```
plot: select_begin
 1
plot: select_end
#
# Set the plot axis to blue
#
plot: axis_rgb 0 0 1
```

See Also:

Associated plot commands

plot: axis_swap_minmax

Command:

```
plot: axis_swap_minmax <axis>
```

Purpose:

Swaps the axis min and max values for the specified axis of the currently selected plot.

Parameters:

Type	Parameter	Description
constant	`axis`	must be either `x` or `y`

Notes:

By default the values increase along the axis.

Example Usage:

```
plot: select_begin
 1
plot: select_end
plot: axis_swap_minmax  x
```

See Also:

Associated plot commands

Command:

```
plot: axis_x_gridtype <grid_type>
plot: axis_y_gridtype <grid_type>
```

Purpose:

Specifies the type of grid that will be drawn for the x/y axis of the currently selected plot.

Parameters:

Type	Parameter	Description	
constant	grid_type	Can be any of the following values:	
		none	specifies no grid lines
		grid	specifies solid grid lines
		tick	specifies tick marks only

Notes:

The default grid type is tick.

Example Usage:

```
plot: select_begin
 1
plot: select_end
plot: axis_x_gridtype tick
plot: axis_y_gridtype grid
```

See Also:

Associated plot commands

plot: axis_x/y_labelformat

Command:

```
plot: axis_x_labelformat  <format>
plot: axis_y_labelformat  <format>
```

Purpose:

Sets the format to be used for x/y axis value labels in the currently selected plot.

Parameters:

Type	Parameter	Description
string	`format`	may be any valid C language print format string

Example Usage:

```
plot: select_begin
 1
plot: select_end
plot: axis_x_labelformat  %.2e
plot: axis_y_labelformat  %.3e
```

See Also:

Associated plot commands

Command:

```
plot: axis_x_labelrgb <red_val> <grn_val> <blu_val>
plot: axis_y_labelrgb <red_val> <grn_val> <blu_val>
```

Purpose:

Specifies the RGB color value of the x/y axis label for the currently selected plot(s).

Parameters:

Type	Parameter	Description
float	red_val	red, green, blue color components, must be between 0.0 and 1.0.
	grn_val	
	blu_val	

Example Usage:

```
plot: select_begin
 1
plot: select_end
plot: axis_x_labelrgb 0 0 1
plot: axis_y_labelrgb 1 0 0
```

See Also:

Associated plot commands

plot: axis_x/y_labelsize

Command:

```
plot: axis_x_labelsize <size>
plot: axis_y_labelsize <size>
```

Purpose:

Sets the size of the x/y axis label text in the currently selected plot.

Parameters:

Type	Parameter	Description
float	`size`	label text size; can be any value from 1 to 100.0, with a value of one specifying the smallest text and 100 the largest.

Example Usage:

```
plot: select_begin
 1
plot: select_end
plot: axis_x_labelsize 10
plot: axis_y_labelsize 10
```

See Also:

Associated plot commands

Command:

```
plot: axis_x_labeltype <label_type>
plot: axis_y_labeltype <label_type>
```

Purpose:

Specifies how the x/y axis is labeled in the currently selected plot.

Parameters:

Type	Parameter	Description	
constant	label_type	Can be any of the following values:	
		none	no labels are drawn along axis
		all	a label is drawn for each tick along axis
		beg_end	a label is drawn at the begin and end of axis

Notes:

The default value for x/y axis label type is `all`.

Example Usage:

```
plot: select_begin
 1
plot: select_end
plot: axis_x_labeltype none
plot: axis_y_labeltype beg_end
```

See Also:

Associated plot commands

plot: axis_x/y_max

Command:

```
plot: axis_x_max <max>
plot: axis_y_max <max>
```

Purpose:

Sets the maximum value for the x/y axis labels in the currently selected plot.

Parameters:

Type	Parameter	Description
float	`max`	max label number on the axis; can be any valid floating-point value

Example Usage:

```
plot: select_begin
 1
plot: select_end
plot: axis_x_max  9.5
plot: axis_y_max  5.3
```

See Also:

Associated plot commands

Command:

```
plot: axis_x_min <min>
plot: axis_y_min <min>
```

Purpose:

Sets the minimum value for the x/y axis labels in the currently selected plot.

Parameters:

Type	Parameter	Description
float	`min`	min label number on the axis; can be any valid floating-point value

Example Usage:

```
plot: select_begin
 1
plot: select_end
plot: axis_x_min 1.5
plot: axis_y_min 1.5
```

See Also:

Associated plot commands

plot: axis_x/y_numgrid

Command:

```
plot: axis_x_numgrid <num>
plot: axis_y_numgrid <num>
```

Purpose:

Sets the number of grid lines to be used along the x/y axis of the currently selected plot(s).

Parameters:

Type	Parameter	Description
int	num	number of grid lines along the axis; can be any whole number

Example Usage:

```
plot: select_begin
 1
plot: select_end
plot: axis_x_numgrid  10
plot: axis_y_numgrid   6
```

See Also:

Associated plot commands

Command:

```
plot: axis_x_numsubgrid <num>
plot: axis_y_numsubgrid <num>
```

Purpose:

Sets the number of subgrid lines to be used along the x/y axis of the currently selected plot(s).

Parameters:

Type	Parameter	Description
float	num	number of subgrid lines (or ticks) to be used between grid lines along an axis; can be any whole number

Notes:

While the argument to this command is a floating-point value, the actual number of grid lines drawn will be truncated to an integer.

Example Usage:

```
plot: select_begin
 1
plot: select_end
plot: axis_x_numsubgrid  2
plot: axis_y_numsubgrid  1
```

See Also:

Associated plot commands

plot: axis_x/y_origin

Command:

```
plot: axis_x_origin <origin>
plot: axis_y_origin <origin>
```

Purpose:

Sets the value of the x/y axis origin for the currently selected plot(s).

Parameters:

Type	Parameter	Description
float	`origin`	x or y location of origin within the plotter; range is the normalized values 0.0 to 1.0

Notes:

The default x/y axis origin is 0.

Example Usage:

```
plot: select_begin
 1
plot: select_end
plot: axis_x_origin  0.25
plot: axis_y_origin  0.18
```

See Also:

Associated plot commands

Command:

```
plot: axis_x_scale <scale_type>
plot: axis_y_scale <scale_type>
```

Purpose:

Selects the type of scale used for the x/y axis of the currently selected plot(s).

Parameters:

Type	Parameter	Description
constant	scale_type	one of the following:
		linear
		logarithmic

Notes:

The default scale is linear.

Example Usage:

```
plot: select_begin
 1
plot: select_end
plot: axis_x_scale logarithmic
plot: axis_y_scale linear
```

See Also:

Associated plot commands

plot: axis_x/y_sgridtype

Command:

```
plot: axis_x_sgridtype <grid_type>
plot: axis_y_sgridtype <grid_type>
```

Purpose:

Specifies the type of subgrid that will be drawn for the x/y axis of the currently selected plot.

Parameters:

Type	Parameter	Description	
constant	grid_type	Can be any of the following values:	
		none	specifies no grid lines
		grid	specifies solid grid lines
		tick	specifies tick marks only

Notes:

The default subgrid type is none.

Example Usage:

```
plot: select_begin
 1
plot: select_end
plot: axis_x_sgridtype tick
plot: axis_y_sgridtype grid
```

See Also:

Associated plot commands

Command:

```
plot: axis_x_title <title>
plot: axis_y_title <title>
```

Purpose:

Sets the title string for the x/y axis of the currently selected plot.

Parameters:

Type	Parameter	Description
string	`title`	x/y axis title

Example Usage:

```
plot: select_begin
 1
plot: select_end
plot: axis_x_title Time in Seconds
plot: axis_y_title Pressure
```

See Also:

Associated plot commands

plot: axis_x/y_titlergb

Command:

```
plot: axis_x_titlergb <red_val> <grn_val> <blu_val>
plot: axis_y_titlergb <red_val> <grn_val> <blu_val>
```

Purpose:

Specifies the RGB color value of the x/y axis title for the currently selected plot(s).

Parameters:

Type	Parameter	Description
float	red_val	red, green, blue color components, must be between 0.0 and 1.0
	grn_val	
	blu_val	

Example Usage:

```
plot: select_begin
 1
plot: select_end
plot: axis_x_titlergb 0 0 1
plot: axis_y_titlergb 0 1 0
```

See Also:

Associated plot commands

Command:

```
plot: axis_x_titlesize <size>
plot: axis_y_titlesize <size>
```

Purpose:

Sets the size of the x/y axis title in the currently selected plot.

Parameters:

Type	Parameter	Description
float	`size`	axis title text size; can be any value from 1 to 100.0, with a value of one specifying the smallest text and 100 the largest.

Example Usage:

```
plot: select_begin
 1
plot: select_end
plot: axis_x_titlesize 10
plot: axis_y_titlesize 12
```

See Also:

Associated plot commands

plot: axis_x/y_visible

Command:

```
plot: axis_x_visible <toggle>
plot: axis_y_visible <toggle>
```

Purpose:

Controls the visibility of the x/y axis of the currently selected plot(s).

Parameters:

Type	Parameter	Description
constant	toggle	ON
		OFF

Notes:

Plot axes are visible by default.

Example Usage:

```
plot: select_begin
 1
plot: select_end
plot: axis_x_visible OFF
plot: axis_y_visible ON
```

See Also:

Associated plot commands

Command:

```
plot: background_rgb <red_val> <grn_val> <blu_val>
```

Purpose:

Specifies the background RGB color value for the currently selected plot(s).

Parameters:

Type	Parameter	Description
float	red_val	red, green, blue color components, must be between 0.0 and 1.0
	grn_val	
	blu_val	

Notes:

Used only if `plot: background_type` is set to solid.

Example Usage:

```
plot: select_begin
  1
plot: select_end
plot: background_rgb 0 0 1
```

See Also:

Associated plot commands

plot: background_type

Command:

```
plot: background_type <type>
```

Purpose:

Specifies the background type of the currently selected plot.

Parameters:

Type	Parameter	Description	
constant	`type`	Can be any of the following values:	
		`none`	plot background is transparent, allowing things under it to be seen
		`solid`	plot background is opaque

Notes:

The default for plot background type is none.

Example Usage:

```
plot: select_begin
 1
plot: select_end
plot: background_type solid
```

See Also:

Associated plot commands

Command:

```
plot: border_rgb <red_val> <grn_val> <blu_val>
```

Purpose:

Specifies the border RGB color value for the currently selected plot(s).

Parameters:

Type	Parameter	Description
float	red_val	red, green, blue color components, must be between 0.0 and 1.0
	grn_val	
	blu_val	

Notes:

Used only if `plot: border_visible` is set to `ON`.

Example Usage:

```
plot: select_begin
 1
plot: select_end
plot: border_rgb 0 0 1
```

See Also:

Associated plot commands

plot: border_visible

Command:

```
plot: border_visible <toggle>
```

Purpose:

Controls the visibility of the border of the currently selected plot(s).

Parameters:

Type	Parameter	Description
constant	`toggle`	`ON`
		`OFF`

Notes:

Plot borders are visible by default.

The border will be drawn according to `plot: border_rgb`

Example Usage:

```
plot: select_begin
 1
plot: select_end
plot: border_visible ON
```

See Also:

Associated plot commands

Command:

```
plot: delete_plot
```

Purpose:

Deletes the currently selected plot(s).

Parameters:

none

Example Usage:

```
plot: select_begin
 1 2 3
plot: select_end
plot: delete_plot
```

See Also:

Associated plot commands

plot: grid_linetype

Command:

```
plot: grid_linetype <line_type>
```

Purpose:

Specifies the type of grid lines that will be drawn for the currently selected plot.

Parameters:

Type	Parameter	Description
constant	`line_type`	Can be any of the following values:
		`solid`
		`dotted`
		`dash`

Notes:

The grid `line_type` is `solid` by default.

Example Usage:

```
plot: select_begin
 1
plot: select_end
plot: grid_linetype solid
```

See Also:

Associated plot commands

Command:

```
plot: grid_linewidth <width>
```

Purpose:

Sets the grid line width of the currently selected plot(s).

Parameters:

Type	Parameter	Description
int	`width`	specifies the line width in pixels of the grid lines, and can be any value from 1 to 4

Example Usage:

```
plot: select_begin
 1
plot: select_end
plot: grid_linewidth  2
```

See Also:

Associated plot commands

plot: grid_rgb

Command:

```
plot: grid_rgb <red_val> <grn_val> <blu_val>
```

Purpose:

Specifies the RGB color value of the grid for the currently selected plot(s).

Parameters:

Type	Parameter	Description
float	red_val	red, green, blue color components, must be between 0.0 and 1.0
	grn_val	
	blu_val	

Example Usage:

```
plot: select_begin
 1
plot: select_end
plot: grid_rgb 0 0 1
```

See Also:

Associated plot commands

Command:

```
plot: height  <height>
```

Purpose:

Sets the height of the currently selected plot.

Parameters:

Type	Parameter	Description
float	`height`	height of plot viewport in normalized coordinates, namely any value between 0 and 1.0, with a value of one corresponding to the height of the graphics window

Notes:

Plots are drawn in reference to the graphics window. A value of 1.0 is the height of the window.

Example Usage:

```
plot: select_begin
 1
plot: select_end
plot: width  0.5
plot: height 0.4
```

See Also:

Associated plot commands

plot: legend_rgb

Command:

```
plot: legend_rgb <red_val> <grn_val> <blu_val>
```

Purpose:

Specifies the RGB color value of the legend for the currently selected plot(s).

Parameters:

Type	Parameter	Description
float	red_val	red, green, blue color components, must be between 0.0 and 1.0
	grn_val	
	blu_val	

Notes:

Used only if `plot: legend_visible` is `ON`.

Example Usage:

```
plot: select_begin
 1
plot: select_end
plot: legend_rgb 0 0 1
```

See Also:

Associated plot commands

Command:

```
plot: legend_textsize <size>
```

Purpose:

Sets the size of legend text in the currently selected plot.

Parameters:

Type	Parameter	Description
float	`size`	size of legend text; any value from 1 to 100.0, with a value of one specifying the smallest text and 100 the largest

Example Usage:

```
plot: select_begin
 1
plot: select_end
plot: legend_textsize  10
```

See Also:

Associated plot commands

plot: legend_visible

Command:

```
plot: legend_visible <toggle>
```

Purpose:

Controls the visibility of the legend for the currently selected plot(s).

Parameters:

Type	Parameter	Description
constant	toggle	ON
		OFF

Notes:

Plot legends are visible by default.

Example Usage:

```
plot: select_begin
 1
plot: select_end
plot: legend_visible OFF
```

See Also:

Associated plot commands

Command:

```
plot: origin_x <coord>
plot: origin_y <coord>
```

Purpose:

Specifies the x/y coordinate of the origin of the selected plot in the graphics window.

Parameters:

Type	Parameter	Description
float	coord	normalized coordinate of lower left corner of the plot viewport. A value of 0.0 indicates the left/botton side of the window, while a value of 1.0 indicates the right/top side.

Notes:

The origin of the plot will be positioned at this point (x,y) in the graphics window

Example Usage:

```
plot: select_begin
 1
plot: select_end
plot: origin_x  0.5
plot: origin_y  0.5
```

See Also:

Associated plot commands

plot: plot_title

Command:

```
plot: plot_title <title>
```

Purpose:

Sets the title string for the currently selected plot.

Parameters:

Type	Parameter	Description
string	`title`	plot title

Example Usage:

```
plot: select_begin
 1
plot: select_end
plot: plot_title My Plot Title
```

See Also:

Associated plot commands

Command:

```
plot: select_all
```

Purpose:

Selects all of the plots in the graphics window as the target for future `plot:` commands.

Parameters:

none

Example Usage:

```
plot: select_all
plot: delete_plot
```

See Also:

Associated plot commands

plot: select_begin/end

Command:

```
plot: select_begin
<index1  index2 ... indexN>
plot: select_end
```

Purpose:

Selects one or more plots in the graphics window as the target for future `plot:` commands

Parameters:

Type	Parameter	Description
int	`indexN`	plot number (zero based). Up to 10 numbers per line are allowed, but as many lines as needed can be used

Example Usage:

```
plot: select_begin
 1 2
plot: select_end
plot: delete_plot
```

See Also:

Associated plot commands

Command:

```
plot: select_default
```

Purpose:

Causes any following `plot:` commands to affect the default settings, rather than a particular plot.

Parameters:

none

Notes:

All `plot:` commands following `plot: select_default` will affect default settings until a new plot is selected using `plot: select_begin/select_end`.

Example Usage:

```
plot: select_default
plot: visible OFF
plot: select_begin
 1
plot: select_end
```

See Also:

Associated plot commands

plot: subgrid_linetype

Command:

```
plot: subgrid_linetype <line_type>
```

Purpose:

Specifies the type of subgrid lines that will be drawn for the currently selected plot.

Parameters:

Type	Parameter	Description
constant	`line_type`	Can be any of the following values:
		`solid`
		`dotted`
		`dash`

Notes:

The subgrid `line_type` is `solid` by default

Example Usage:

```
plot: select_begin
 1
plot: select_end
plot: subgrid_linetype solid
```

See Also:

Associated plot commands

Command:

```
plot: subgrid_linewidth <width>
```

Purpose:

Sets the subgrid line width of the currently selected plot(s).

Parameters:

Type	Parameter	Description
int	`width`	the line width in pixels for subgrid lines (or tics); can be any value from 1 to 4

Example Usage:

```
plot: select_begin
 1
plot: select_end
plot: subgrid_linewidth  2
```

See Also:

Associated plot commands

plot: subgrid_rgb

Command:

```
plot: subgrid_rgb <red_val> <grn_val> <blu_val>
```

Purpose:

Specifies the RGB color value of the subgrid for the currently selected plot(s).

Parameters:

Type	Parameter	Description
float	red_val	red, green, blue color components, must be between 0.0 and 1.0
	grn_val	
	blu_val	

Example Usage:

```
plot: select_begin
 1
plot: select_end
plot: subgrid_rgb 0 0 1
```

See Also:

Associated plot commands

Command:

```
plot: title_rgb <red_val> <grn_val> <blu_val>
```

Purpose:

Specifies the RGB color value of the title for the currently selected plot(s).

Parameters:

Type	Parameter	Description
float	red_val	red, green, blue color components, must be between 0.0 and 1.0
	grn_val	
	blu_val	

Example Usage:

```
plot: select_begin
 1
plot: select_end
plot: title_rgb 0 0 1
```

See Also:

Associated plot commands

plot: title_textsize

Command:

```
plot: title_textsize <size>
```

Purpose:

Sets the size of title text in the currently selected plot.

Parameters:

Type	Parameter	Description
float	`size`	size of title text; any value from 1 to 100.0, with a value of one specifying the smallest text and 100 the largest

Example Usage:

```
plot: select_begin
 1
plot: select_end
plot: title_textsize  10
```

See Also:

Associated plot commands

Command:

```
plot: visible <toggle>
```

Purpose:

Controls the visibility of the selected plot(s) in the graphics window.

Parameters:

Type	Parameter	Description
constant	toggle	ON
		OFF

Notes:

Plots are visible by default.

Example Usage:

```
plot: select_begin
 1
plot: select_end
plot: visible OFF
```

See Also:

Associated plot commands

plot: width

Command:

```
plot: width  <width>
```

Purpose:

Sets the width of the currently selected plot.

Parameters:

Type	Parameter	Description
float	`width`	width of plot viewport in normalized coordinates, namely any value between 0 and 1.0, with a value of 1.0 corresponding to the width of the graphics window

Example Usage:

```
plot: select_begin
 1
plot: select_end
plot: width  0.5
plot: height 0.4
```

See Also:

Associated plot commands

Commands in this Section:

prefs

See Also:

User Manual - Section 6.2, Edit Menu Functions

User Manual - Chapter 10, Preference File Formats

How To Set or Modify Preferences

How To Customize Icon Bars

How To Customize Mouse Button Actions

How To Save GUI Settings

How To Define and Use Macros

How To Enable User Defined Input Devices

prefs: auto_legend_display

Command:

```
prefs: auto_legend_display <toggle>
```

Purpose:

Will cause the legend to automatically appear when you color a part by a variable.

Parameters:

Type	Parameter	Description
constant	toggle	ON
		OFF

Example Usage:

```
prefs: auto_legend_replace ON
```

See Also:

Associated prefs commands

Command:

```
prefs: auto_legend_replace <toggle>
```

Purpose:

Preference that will cause legends to be automatically replaced when the current legend is no longer in use (i.e. no parts are colored by the variable) and a new variable is in use.

Parameters:

Type	Parameter	Description
constant	toggle	ON
		OFF

Example Usage:

```
prefs: auto_legend_replace ON
```

See Also:

Associated prefs commands

prefs: auto_load_parts

Command:

```
prefs: auto_load_parts <method>
```

Purpose:

Allows you to have EnSight automatically load parts when the data file is read.

Parameters:

Type	Parameter	Description	
constant	method	Must be one of:	
		all	all parts defined in the file
		first	the first part defined in the file
		last	the last part defined in the file
		none	no parts will be read

Example Usage:

```
prefs: starting_time_step last
prefs: auto_load_parts all
data: binary_files_are big_endian
data: format case
data: path /usr/tmp
data: geometry car_crash.case
data: read
```

See Also:

Associated prefs commands

Command:

```
prefs: cull_lines <toggle>
```

Purpose:

Will only draw shared lines between polygons once. May help performance.

Parameters:

Type	Parameter	Description
constant	toggle	ON
		OFF

Example Usage:

```
prefs: cull_lines ON
```

See Also:

Associated prefs commands

prefs: default_data_directory

Command:

```
prefs: default_data_directory <directory_name>
```

Purpose:

Preference to set the default directory where the user's data resides.

Parameters:

Type	Parameter	Description
string	`directory_name`	name of the default directory

Example Usage:

```
prefs: default_data_directory /usr/local/bin/ensight76/data/ami
```

See Also:

Associated prefs commands

Command:

```
prefs: default_palette <predefined_palette_name>
```

Purpose:

Preference to override the default EnSight color palette with a user predefined palette.

Parameters:

Type	Parameter	Description
string	predefined_palette_name	palette name

Notes:

When EnSight starts, it looks for user defined function color palettes located under `$CEI_HOME/ensight76/site_preferences/palettes` and in the `.ensight7/palettes` directory found in the user's home directory. These files must be named `palette_name.cpal`, where the `palette_name` will be the name of the color palette in the Simple Interface area of the function dialog.

Example Usage:

```
prefs: default_palette FASTDefaultPalette
```

See Also:

Associated prefs commands

prefs: fastdisplay_point_res

Command:

```
prefs: fastdisplay_point_res <value>
```

Purpose:

Allows specification of fraction of nodes to display in fast display-point representation. The default is 1 (indicating all nodes), 2 (would be every other node), 3 (every third node), etc.

Parameters:

Type	Parameter	Description
int	`value`	>= 1

Notes:

May help performance.

Example Usage:

```
view: fast_display ON
prefs: static_fast_display ON
prefs: fastdisplay_point_res 2
part: select_begin
1 3 4
part: select_end
part: modify_begin
part: bounding_rep points
part: modify_end
```

See Also:

Associated prefs commands

Command:

```
prefs: fastdisplay_sparse_res <value>
```

Purpose:

Allows specification of the percentage of the model geometry that will be displayed. The default is 50. The range is 1-100 where 1 would be sparse and 100 would be full.

Parameters:

Type	Parameter	Description
int	value	(1-100)

Notes:

Will help performance but will degrade quality in the graphics window while interactively transforming the model.

Example Usage:

```
view: fast_display ON
prefs: static_fast_display ON
prefs: fastdisplay_sparse_res 75
part: select_begin
1 3 4
part: select_end
part: modify_begin
part: bounding_rep sparse_model
part: modify_end
```

See Also:

Associated prefs commands

prefs: frame_mode_allowed

Command:

```
prefs: frame_mode_allowed <toggle>
```

Purpose:

Will display Frame as one of the modes along the left side of the EnSight main interface.

Parameters:

Type	Parameter	Description
constant	toggle	ON
		OFF

Notes:

As EnSight reads in model Parts, they are all initially assigned to the same Frame of reference: Frame 0. Frame 0 corresponds to the model coordinate system (defined when the model was created). Using the Frame mode, you can create additional frames, reassign parts to different frames, and specify various attributes of the frames. By default frame mode is not available because frame mode is an advanced feature not normally used by first time users.

Example Usage:

```
prefs: frame_mode_allowed ON
```

See Also:

Associated prefs commands

Command:

```
prefs: function_editor_type <interface_type>
```

Purpose:

Preference to set the default type of interface used for editing in the variable function editor.

Parameters:

Type	Parameter	Description	
constant	`interface_type`	must be one of:	
		`simple`	simple interface
		`advanced`	advanced interface

Example Usage:

```
prefs: function_editor simple
```

See Also:

Associated prefs commands

prefs: icon_help_labels

Command:

```
prefs: icon_help_labels <toggle>
```

Purpose:

Will cause a help label to appear below the mode icons along the left side of the main EnSight interface.

Parameters:

Type	Parameter	Description
constant	toggle	ON
		OFF

Example Usage:

```
prefs: icon_help_labels ON
```

See Also:

Associated prefs commands

Command:

```
prefs: large_parts_list <toggle>
```

Purpose:

Will cause a separate, larger parts list dialog (which can be expanded) to be used in place of the normal parts list.

Parameters:

Type	Parameter	Description
constant	toggle	ON
		OFF

Example Usage:

```
prefs: large_parts_list ON
```

See Also:

Associated prefs commands

prefs: legend_colorby

Command:

```
prefs: legend_colorby <option>
```

Purpose:

Specifies whether color palettes will be RGB or textures.

Parameters:

Type	Parameter	Description	
constant	option	RGB	color using rgb values
		textures	color using textures

Example Usage:

```
prefs: legend_colorby textures
```

See Also:

Associated prefs commands

Command:

```
prefs: macro_panel <toggle>
```

Purpose:

Toggles on/off the user defined macro panel. This is normally used with VR devices and user defined input. The macro panel will be displayed in the main graphics window.

Parameters:

Type	Parameter	Description
constant	toggle	ON
		OFF

Notes:

Users define the macro panel layout and actions in the `~/.ensight7/macros/hum.define` file. An example hum.define file is located at `$CEI_HOME/ensight76/site_preferences/macros/hum.define` on your client system.

Example Usage:

```
prefs: macro_panel ON
```

See Also:

Associated prefs commands

prefs: new_data_notification

Command:

```
prefs: new_data_notification <type>
```

Purpose:

Tells client how to handle dynamic data updates.

Parameters:

Type	Parameter	Description	
constant	type	auto_update	cause the client to automatically load new data as it becomes available
		ignore	cause the client to not update to the new data.
		prompt	cause a dialog box to appear asking the user what to do as new data becomes available.

Notes:

Used in conjunction with the dynamic data interface

Example Usage:

```
prefs: new_data_notification auto_update
```

See Also:

Associated prefs commands

Command:

```
prefs: part_panel <toggle>
```

Purpose:

Toggles on/off the part panel which allows users to select parts to operate on. Normally used with VR devices and user defined input when the main part list is not visible (ie. Full screen graphics window). The part panel will be displayed in the main graphics window.

Parameters:

Type	Parameter	Description
constant	`toggle`	`ON`
		`OFF`

Example Usage:

```
prefs: part_panel ON
```

See Also:

Associated prefs commands

prefs: resetlegend_ontimechange

Command:

```
prefs: resetlegend_ontimechange <toggle>
```

Purpose:

Toggles on/off the preference of having the legend ranges update when time is changed or not.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	causes the legend ranges to update on a time change
		`OFF`	the legend ranges will not be updated on a time change

Example Usage:

```
prefs: resetlegend_ontimechange OFF
```

See Also:

Associated prefs commands

Command:

```
prefs: starting_time_step <value>
```

Purpose:

Preference to set the default starting time step for transient data. When the data is read the first or the last step will be loaded as specified.

Parameters:

Type	Parameter	Description	
constant	value	must be one of:	
		first	the first time step
		last	the last time step

Example Usage:

```
prefs: starting_time_step last
data: binary_files_are big_endian
data: format case
data: path /usr/tmp
data: geometry car_crash.case
data: read
```

See Also:

Associated prefs commands

prefs: static_fast_display

Command:

```
prefs: static_fast_display <toggle>
```

Purpose:

Will cause the fast display representation to always be displayed when the fast display toggle is on. If `static_fast_display` is off (the default), fast display will only be active during a transformation.

Parameters:

Type	Parameter	Description
constant	`toggle`	`ON`
		`OFF`

Notes:

This is a performance enhancing attribute. This attribute can be used to work with very large models. Large models can be too slow in full element representation to manipulate (ie. transforming, positioning tools, etc.)

Example Usage:

```
view: fast_display ON
prefs: static_fast_display ON
part: select_begin
1 3 4
part: select_end
part: modify_begin
part: bounding_rep points
part: modify_end
```

See Also:

Associated prefs commands

Command:

```
prefs: tool_tips <toggle>
```

Purpose:

Will cause pop-up help information to appear when the mouse is placed over certain icons while running EnSight.

Parameters:

Type	Parameter	Description
constant	toggle	ON
		OFF

Example Usage:

```
prefs: tool_tips ON
```

See Also:

Associated prefs commands

prefs: transparency_resort

Command:

```
prefs: transparency_resort <toggle>
```

Purpose:

Causes polygons to be resorted with each transformation - so the image is always correct. If not on, the polygons will not be resorted while the mouse is down during transformations, but will be resorted when the mouse is released.

Parameters:

Type	Parameter	Description
constant	`toggle`	`ON`
		`OFF`

Notes:

Increases performance, but only when interactively manipulating the model.

Example Usage:

```
prefs: transparency_resort ON
```

See Also:

Associated prefs commands

Command:

```
prefs: use_continuous_perelem <toggle>
```

Purpose:

If on, display element variables as per node for the purpose of creating contour (filled or line) images.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	display per element variables as per node
		OFF	display per element variables as per element

Notes:

When on, the per element variable is averaged to the nodes using node count weighted algorithm.

Example Usage:

```
#
# set the preference to show the per element colored as a per node vari-
able
prefs: use_continuous_perelem ON
```

See Also:

Associated prefs commands

prefs: user_defined_input

Command:

```
prefs: user_defined_input <toggle>
```

Purpose:

Toggles on/off the User Defined Input Device that is linked via a runtime library. This is normally used with VR devices.

Parameters:

Type	Parameter	Description
constant	toggle	ON
		OFF

Notes:

Steps outlining the implementation of this library and input device are found in the file: $CEI_HOME/ensight76/src/input/README.

Example Usage:

```
prefs: user_defined_input ON
```

See Also:

Associated prefs commands

Command:

```
prefs: user_input_rotatesens <value>
```

Purpose:

Specifies a scalar value that adjusts the sensitivity of rotates when using a User Defined Input Device (i.e. values < 1 are slower, and values > 1 are faster).

Parameters:

Type	Parameter	Description
float	`value`	must be > 0.0

Notes:

Steps outlining the implementation of the library for input devices are found in the file: `$CEI_HOME/ensight76/src/input/README`.

Example Usage:

```
prefs: user_defined_input ON
prefs: user_input_rotateusing Position
prefs: user_input_rotatesens 0.5
```

See Also:

Associated prefs commands

prefs: user_input_rotateusing

Command:

```
prefs: user_input_rotateusing <rotate_mode>
```

Purpose:

Allows selection of the type of rotation mode used for rotate transformations for the User Defined Input Device.

Parameters:

Type	Parameter	Description	
constant	rotate_mode	must be one of:	
		mixed_mode	A device that returns virtual angle values where the Z rotations correspond to (literal) movement of the input device about its local Z (or roll) axis; and where the X and Y rotations correspond to translational movements of the input device with respect to its local X and Y axes.
		direct_mode	A device that returns virtual angle values that correspond to (literal) rotational movements of the input device about its local X, Y, and Z axes.

Notes:

Steps outlining the implementation of the library for input devices are found in the file: `$CEI_HOME/ensight76/src/input/README`.

Example Usage:

```
prefs: user_defined_input ON
prefs: user_input_rotateusing direct_mode
```

See Also:

Associated prefs commands

Command:

```
prefs: user_input_zoomsens <value>
```

Purpose:

Specifies a value that adjusts the sensitivity of the zoom operation for the User Defined Input Device.

Parameters:

Type	Parameter	Description
float	value	must be > 0.0 (< 1 slower, > 1 faster)

Notes:

Steps outlining the implementation of the library for input devices are found in the file: `$CEI_HOME/ensight76/src/input/README`.

Example Usage:

```
prefs: user_defined_input ON
prefs: user_input_zoomusing position
prefs: user_input_zoomsens 1.5
```

See Also:

Associated prefs commands

prefs: user_input_zoomusing

Command:

```
prefs: user_input_zoomusing <mode>
```

Purpose:

Allows selection of the mode used for zoom transformations for the User Defined Input Device.

Parameters:

Type	Parameter	Description	
constant	mode	must be one of:	
		valuator	a device that returns a value (like a virtual joystick).
		position	a device that returns delta movement in the Z direction (like a wand)

Notes:

Steps outlining the implementation of the library for input devices are found in the file: `$CEI_HOME/ensight76/src/input/README`.

Example Usage:

```
prefs: user_defined_input ON
prefs: user_input_zoomusing position
```

See Also:

Associated prefs commands

Command:

```
prefs: view_mode_allowed <toggle>
```

Purpose:

Will display View as one of the modes along the left side of the EnSight main interface.

Parameters:

Type	Parameter	Description
constant	toggle	ON
		OFF

Notes:

View Mode is used to adjust the appearance of Parts in the Graphics Window (Perspective, Static Color, etc.), the visibility and appearance of Labels, to adjust Auxiliary Clipping status, and to toggle visibility of the Global Axis triad. By default, this mode is not available because the attributes are available under the View menu of the main interface.

Example Usage:

```
prefs: view_mode_allowed ON
```

See Also:

Associated prefs commands

Commands in this Section:

profile

profile: begin
profile: end
profile: component <x_scal> <y_scal> <z_scal>
profile: create
profile: in_plane <point>
profile: offset <value>
profile: scale_factor <value>
profile: select_default
profile: variable <name>

See Also:

User Manual - Section 7.8, Profile Create/Update
How To Create Profile Plots

profile: begin/end

Command:

```
profile: begin
<profile_commands>
profile: end
```

Purpose:

Groups profile attribute commands to be applied together before a "`profile: create`" command.

Parameters:

Type	Parameter	Description
string	`profile_commands`	any of the profile attribute commands

Example Usage:

```
profile: begin
profile: variable VITESSE
profile: offset 0.0000e+00
profile: scale_factor 1.3603e+00
profile: end
profile: create
```

See Also:

Associated profile commands

Command:

```
profile: component <x_scal> <y_scal> <z_scal>
```

Purpose:

Specifies the contribution of a vector variable to be applied for profile generation.

Parameters:

Type	Parameter	Description
float	x_scal	scale factor to apply to the x, y, z components of the vector
	y_scal	
	z_scal	

Notes:

If all the scale factors are set to 0.0, the magnitude of the vector will be used (this is the default).

Part selection and modify commands are used as needed since profiles are a part type based on a parent part.

Example Usage:

```
profile: variable VITESSE
part: select_begin
 3
part: select_end
part: modify_begin
#
# Use the y-component of the vector
#
profile: component 0.0000e+00 1.0000e+00 0.0000e+00
part: modify_end
```

See Also:

Associated profile commands

profile: create

Command:

```
profile: create
```

Purpose:

Create a profile part with the currently defined attributes using the currently selected parts as parents.

Parameters:

none

Notes:

The profile part is created using the currenlty selected parts as parents. The profile part is creates using the currently set profile attributes.

Example Usage:

```
#
# Select parent part
#
part: select_begin
 2
part: select_end
profile: begin
profile: variable VITESSE
profile: offset 0.0000e+00
profile: scale_factor 1.3603e+00
profile: end
profile: create
```

See Also:

Associated profile commands

Command:

```
profile: in_plane <point>
```

Purpose:

Specify the location of the orientation points for profile generation.

Parameters:

Type	Parameter	Description
int	point	must be 1, 2, or 3 (indicating which point is being specified)

Notes:

Three points define the plane in which the profile will be projected. The corresponding 4th point is generated and the center of these 4 points determines which direction the profile will projected in the plane.

Example Usage:

```
#
#  Select existing profile part
#
part: select_begin
 3
part: select_end
part: modify_begin
profile: in_plane 1 1.1977e+00 2.5928e+00 -9.6950e+00
profile: in_plane 2 2.0198e+01 2.5928e+00 -9.6950e+00
profile: in_plane 3 2.0198e+01 9.5655e+00 7.9793e+00
part: modify_end
```

See Also:

Associated profile commands

profile: offset

Command:

```
profile: offset <value>
```

Purpose:

Add a value to the variable values before scaling the profile.

Parameters:

Type	Parameter	Description
float	`value`	value to add to the variable values before scaling

Notes:

This command has the effect of shifting the profile up or down relative to the parent part(s).

Example Usage:

```
#
# Select existing profile part
#
part: select_begin
 3
part: select_end
part: modify_begin
profile: offset 1.0000e+00
profile: scale_factor 2.0000e+00
part: modify_end
```

See Also:

Associated profile commands

Command:

```
profile: scale_factor <value>
```

Purpose:

Apply a scale factor to the variable values of the profile.

Parameters:

Type	Parameter	Description
float	value	scale factor to apply to the variable values

Example Usage:

```
#
# Select existing profile part
#
part: select_begin
 3
part: select_end
part: modify_begin
profile: offset 3.0000e+00
profile: scale_factor 1.5000e+01
part: modify_end
```

See Also:

Associated profile commands

profile: select_default

Command:

```
profile: select_default
```

Purpose:

Sets the current profile to none, so that subsequent commands will modify the default profile attributes.

Parameters:

none

Notes:

When profiles are created, the default attributes are used (unless overridden by attribute commands between "`profile: begin / end`" commands. This command allows for changing of defaults.

Example Usage:

```
profile: select_default
profile: offset 0.0000e+00
profile: scale_factor 1.0000e+00
```

See Also:

Associated profile commands

Command:

```
profile: variable <name>
```

Purpose:

Sets the variable to use for profile creation or modification.

Parameters:

Type	Parameter	Description
string	name	variable name

Example Usage:

```
#
# Select existing profile part
#
part: select_begin
 2
part: select_end
profile: begin
profile: variable VITESSE
profile: end
profile: create
```

See Also:

Associated profile commands

Commands in this Section:

ptrace

ptrace: min_steps <num>
ptrace: modify_emit_totoolloc <part_num> <emitter>
ptrace: modify_fileemit_name <emitter> <filename>
ptrace: modify_netemit_points <emitter> <new_num_in_x> <new_num_in_y>
ptrace: modify_partemit_byname <emitter> <new_part_name>
ptrace: modify_partemit_num <emitter> <new_part_num>
ptrace: modify_rakeemit_points <emitter> <new_num_pts>
ptrace: representation <rep>
ptrace: ribbon_width <width>
ptrace: rot_angle <value>
ptrace: select_default
ptrace: select_emitter_begin
ptrace: select_emitter_end
ptrace: start_time <emit_time>
ptrace: surface_restricted <toggle>
ptrace: total_time <limit>
ptrace: trace_direction <direction>
ptrace: type <option>
ptrace: variable <name>
ptrace: variable_offset <offset_distance>

See Also:

User Manual - Section 7.4, Particle Trace Create/Update

How To Create Particle Traces

Command:

```
ptrace: add_file <filename>
```

Purpose:

Adds emitters (and thus more traces) to the current particle trace part(s) according to the time and locations as defined in the file.

Parameters:

Type	Parameter	Description
string	`filename`	The name of the emitter file

Example Usage:

```
#
# Add emitters to a currently defined particle trace part 3
#
#
part: select_begin
 3
part: select_end
ptrace: add_file /tmp/emitter.file
```

See Also:

EnSight Particle Emitter File Format

Associated ptrace commands

ptrace: add_net

Command:

```
ptrace: add_net <num_in_x> <num_in_y>
```

Purpose:

Add emitters (and thus more traces) to the current particle trace part(s) from

Parameters:

Type	Parameter	Description
int	num_in_x	number of equally spaced points in the x direction of the plane tool
	num_in_y	number of equally spaced points in the y direction of the plane tool

Notes:

Any existing particle trace parts can have emitters added to them. These trace parts must be selected, and the plane tool must be at the new position, before the addition. The number of additional emitter points will be the product of `num_in_x * num_in_y`.

Example Usage:

```
#
# Getting the plane tool to the desired location
#
view_transf: plane 1 1.750000 2.700000 0.000000
view_transf: plane 2 13.000000 2.700000 0.000000
view_transf: plane 3 13.000000 9.500000 0.000000
#
# Selecting the parent part(s)
#
part: select_begin
 1
part: select_end
#
# Setting some attributes before creation
# (others are the current defaults)
#
ptrace: begin
ptrace: variable VITESSE
ptrace: type streamline
ptrace: component 1.0000e+00 1.0000e+00 1.0000e+00
ptrace: representation line
ptrace: emit_at_current_time ON
ptrace: end
#
# Creating 35 traces from the plane tool (becomes part 2)
#
ptrace: create_net 5 7
#
# Moving the plane tool to a new location
#
tools: plane ON
view_transf: function plane
view_transf: scale -0.522802 -1.195307 1.000000
view_transf: translate -10.588167 -0.218373 0.000000
view_transf: function global
#
# Selecting the particle trace part to add to
# and adding the 9 new emitters from which
```

```
# more traces are made
#
part: select_begin
 2
part: select_end
ptrace: add_net 3 3
```

See Also:

Associated ptrace commands

ptrace: add_part

Command:

```
ptrace: add_part <part_num>
```

Purpose:

Add emitters (and thus more traces) to the current particle trace part(s) from the nodes of additional parts.

Parameters:

Type	Parameter	Description
int	`part_num`	number of the part whose nodes will be used as an emitter

Notes:

Any existing particle trace parts can have emitters added to them. These trace parts must be selected before the addition. The number of additional emitter points will be the number of nodes in the part multiplied by the "`ptrace: emit_density`" value.

Example Usage:

```
#
# Selecting the parent part(s)
#
part: select_begin
 1
part: select_end
#
# Setting some attributes before creation
# (others are the current defaults)
#
ptrace: begin
ptrace: variable VITESSE
ptrace: type streamline
ptrace: component 1.0000e+00 1.0000e+00 1.0000e+00
ptrace: representation line
ptrace: emit_at_current_time ON
ptrace: emit_density 1.0000e+00
ptrace: end
#
# Creating traces from all the nodes of part 2 (becomes part 7)
#
ptrace: create_part 2
#
# Selecting the particle trace part to add to
# and adding the new emitters from all the nodes
* of part 3, form which more traces are made
#
part: select_begin
 7
part: select_end
ptrace: add_part 3
```

See Also:

Associated ptrace commands

Command:

```
ptrace: add_partbyname <part_name>
```

Purpose:

Add emitters (and thus more traces) to the current particle trace part(s) from the nodes of additional parts. (Used instead of "`ptrace: add_part`" if doing parts by name)

Parameters:

Type	Parameter	Description
string	`part_name`	name of the part whose nodes will be used as an emitter

Notes:

Any existing particle trace parts can have emitters added to them. These trace parts must be selected before the addition. The number of additional emitter points will be the number of nodes in the part multiplied by the "`ptrace: emit_density`" value.

Example Usage:

```
#
# Selecting the particle trace part
#
part: select_partname_begin
"(CASE:Case 1)Particle trace part"
part: select_partname_end
#
# Creating traces from all the nodes
# of a clip line part
#
ptrace: add_partbyname (CASE:Case 1)Clip_line
```

See Also:

Associated ptrace commands

ptrace: add_pt

Command:

```
ptrace: add_pt
```

Purpose:

Adds emitters (and thus more traces) to the current particle trace part(s) from the current location of the cursor (a point).

Parameters:

none

Notes:

Any existing particle trace part can have emitters added to them. These trace parts must be selected, and the cursor must be at the new location, before the addition.

Example Usage:

```
#
# Getting the cursor to the desired location
#
view_transf: cursor -0.927830 1.474020 0.000000
#
# Selecting the parent part(s)
#
part: select_begin
 1
part: select_end
#
# Setting some attributes before creation
# (others are the current defaults)
#
ptrace: begin
ptrace: variable VITESSE
ptrace: type streamline
ptrace: component 1.0000e+00 1.0000e+00 1.0000e+00
ptrace: representation line
ptrace: emit_at_current_time ON
ptrace: end
#
# Creating trace from a point (becomes part 2)
#
ptrace: create_pt
#
# Moving the cursor to a new location
#
tools: cursor ON
view_transf: function cursor
view_transf: translate -7.368053 -0.491341 0.000000
view_transf: function global
#
# Selecting the particle trace part to add to
# and adding the new emitter from which another
# trace is made
#
part: select_begin
 2
part: select_end
ptrace: add_pt
```

See Also:

Associated ptrace commands

ptrace: add_rake

Command:

```
ptrace: add_rake <num_points>
```

Purpose:

Add emitters (and thus more traces) to the current particle trace part(s) from the current line tool.

Parameters:

Type	Parameter	Description
int	num_points	number of equally spaced points on the line to use as emitters

Notes:

Any existing particle trace part can have emitters added to them. These trace parts must be selected, and the line tool must be at the new position, before the addition.

Example Usage:

```
#
# Getting the line tool to the desired location
#
view_transf: line 1 9.000000 6.000000 0.000000
view_transf: line 2 30.000000 6.000000 0.000000
#
# Selecting the parent part(s)
#
part: select_begin
 1
part: select_end
#
# Setting some attributes before creation
# (others are the current defaults)
#
ptrace: begin
ptrace: variable VITESSE
ptrace: type streamline
ptrace: component 1.0000e+00 1.0000e+00 1.0000e+00
ptrace: representation line
ptrace: emit_at_current_time ON
ptrace: end
#
# Creating 10 traces from a line tool (becomes part 2)
#
ptrace: create_rake 10
#
# Moving the line tool to a new location
#
tools: line ON
view_transf: function line
view_transf: translate -4.202520 4.531246 0.000000
view_transf: function global
#
# Selecting the particle trace part to add to
# and adding the 6 new emitters from which more
# traces are made
#
part: select_begin
 2
```

```
part: select_end
ptrace: add_rake 6
```

See Also:

Associated ptrace commands

ptrace: add_surface_net

Command:

```
ptrace: add_surface_net <num_in_x> <num_in_y> <x1> <y1> <x2> <y2>
```

Purpose:

Add emitters (and thus more traces) to the current particle trace part(s) from the grid of a screen box pick.

Parameters:

Type	Parameter	Description
int	num_in_x	number of equally spaced points in the x direction of the plane tool
	num_in_y	number of equally spaced points in the y direction of the plane tool
float	x1	screen x coordinate of the first point on the diagonal of the box (0.0 to 1.0)
	y1	screen y coordinate of the first point on the diagonal of the box (0.0 to 1.0)
	x2	screen x coordinate of the second point on the diagonal of the box (0.0 to 1.0)
	y2	screen y coordinate of the second point on the diagonal of the box (0.0 to 1.0)

Notes:

The "ptrace: surface_restricted" command must be ON prior to this command.

Any existing particle trace parts can have emitters added to them. These trace parts must be selected before the addition.

The number of additional emitter points will be the product of num_in_x * num_in_y.

Example Usage:

```
#
# Selecting the parent part(s)
#
part: select_begin
 1
part: select_end
#
# Setting some attributes before creation
# (surface_restricted must be on)
#
ptrace: begin
ptrace: surface_restricted ON
ptrace: variable VITESSE
ptrace: type streamline
ptrace: component 1.0000e+00 1.0000e+00 1.0000e+00
ptrace: representation line
ptrace: emit_at_current_time ON
ptrace: end
#
# Creating 35 surface restricted traces from
# the box selection (becomes part 2)
#
ptrace: create_surface_net 5 7
#
```

```
# Selecting the surface restricted particle trace
# part to add to, and adding the 16 new emitters
# from which more traces are made
#
part: select_begin
 2
part: select_end
ptrace: create_surface_net 4 4 0.114734 0.424963 0.166667 0.362556
```

See Also:

Associated ptrace commands

ptrace: add_surface_pt

Command:

```
ptrace: add_surface_pt <screen_x> <screen_y>
```

Purpose:

Adds emitters (and thus more traces) to the current surface restricted particle trace part(s) from a screen point pick location.

Parameters:

Type	Parameter	Description
float	screen_x	screen x coordinate (0.0 to 1.0)
	screen_y	screen y coordinate (0.0 to 1.0)

Notes:

The "ptrace: surface_restricted" command must be ON prior to this command. Any existing surface restricted particle trace part can have emitters added to them. These trace parts must be selected before the addition.

Example Usage:

```
#
# Getting the cursor to the desired location
#
view_transf: cursor -0.927830 1.474020 0.000000
#
# Selecting the parent part(s)
#
part: select_begin
 1
part: select_end
#
# Setting some attributes before creation
# (surface_restricted must be on)
#
ptrace: begin
ptrace: surface_restricted ON
ptrace: variable VITESSE
ptrace: type streamline
ptrace: component 1.0000e+00 1.0000e+00 1.0000e+00
ptrace: representation line
ptrace: emit_at_current_time ON
ptrace: end
#
# Creating surface restricted trace from a
# screen location (becomes part 2)
#
ptrace: create_surface_pt 0.148551 0.484398
#
# Selecting the particle trace part to add to
# and adding a couple of new emitters from which
# more traces are made
#
part: select_begin
 2
part: select_end
ptrace: add_surface_pt 0.136473 0.646360
ptrace: add_surface_pt 0.193237 0.546805
```

See Also:

Associated ptrace commands

ptrace: add_surface_rake

Command:

```
ptrace: add_surface_rake <num_points> <x1> <y1> <x2> <y2>
```

Purpose:

Add emitters (and thus more traces) to the current surface restricted particle trace part(s) from a screen pick rake location.

Parameters:

Type	Parameter	Description
int	num_points	number of equally spaced points on the line to use as emitters
float	x1	screen x coordinate of the first end of the rake (0.0 to 1.0)
	y1	screen y coordinate of the first end of the rake (0.0 to 1.0)
	x2	screen x coordinate of the second end of the rake (0.0 to 1.0)
	y2	screen y coordinate of the second end of the rake (0.0 to1.0)

Notes:

The "ptrace: surface_restricted" command must be ON prior to this command. Any existing particle trace parts can have emitters added to them. These trace parts must be selected before the addition.

Example Usage:

```
#

# Getting the line tool to the desired location
#
view_transf: line 1 9.000000 6.000000 0.000000
view_transf: line 2 30.000000 6.000000 0.000000
#
# Selecting the parent part(s)
#
part: select_begin
 1
part: select_end
#
# Setting some attributes before creation
# (surface_restricted must be on)
#
ptrace: begin
ptrace: surface_restricted ON
ptrace: variable VITESSE
ptrace: type streamline
ptrace: component 1.0000e+00 1.0000e+00 1.0000e+00
ptrace: representation line
ptrace: emit_at_current_time ON
ptrace: end
#
# Creating 10 surface restricted traces from
# a rake pick location (becomes part 2)
#
ptrace: create_surface_rake 10 0.117150 0.540862 0.117150 0.442793
#
# Selecting the particle trace part to add to
# and adding 6 new emitters from which more
```

```
# traces are made
#
part: select_begin
 2
part: select_end
ptrace: add_surface_rake 6 0.129227 0.594354 0.188406 0.592868
```

See Also:

Associated ptrace commands

ptrace: animate

Command:

```
ptrace: animate <toggle>
```

Purpose:

Toggles particle trace animation on or off for selected traces.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	to animate the trace
		OFF	otherwise

Example Usage:

```
#
# Selecting the trace(s) to animate
#
part: select_begin
 7 8
part: select_end
#
# Turning on the animation for the two selected trace parts
#
part: modify_begin
ptrace: animate ON
part: modify_end
```

See Also:

Associated ptrace commands

Command:

```
ptrace: begin
<ptrace_command>
<ptrace_command>
.
.
.
<ptrace_command>
ptrace: end
```

Purpose:

Delimit the modifications for particle trace attributes.

Parameters:

Type	Parameter	Description
string	ptrace_command	any of the various attribute changing "ptrace:" commands

Notes:

These commands are used to set the attributes for particle trace part creation.

Example Usage:

```
#
# Selecting the parent part
#
part: select_begin
 1
part: select_end
#
# Setting some attributes before creation
#
ptrace: begin
ptrace: type streamline
ptrace: component 1.0000e+00 1.0000e+00 1.0000e+00
ptrace: representation line
ptrace: end
#
# Creating trace from a point
#
ptrace: create_pt
```

See Also:

Associated ptrace commands

ptrace: component

Command:

```
ptrace: component <x_scal> <y_scal> <z_scal>
```

Purpose:

Specifies the contribution of the components of the flow field vector variable through which the particle trace will be integrated.

Parameters:

Type	Parameter	Description
float	x_scal	scale factor to apply to the x, y, z components of the vector
	y_scal	
	z_scal	

Notes:

The normal situation is to have all three set to 1.0, to get the true vector field.

Example Usage:

```
#
# Selecting the parent part
#
part: select_begin
 1
part: select_end
#
# Setting some attributes before creation
# (the components are being set here)
#
ptrace: begin
ptrace: type streamline
#
# Create the particle trace with
# the x and y components of the vector
#
ptrace: component 1.0000e+00 1.0000e+00 0.0000e+00
ptrace: representation line
ptrace: end
#
# Creating trace from a point
#
ptrace: create_pt
```

See Also:

Associated ptrace commands

Command:

```
ptrace: create_file <filename>
```

Purpose:

Create particle traces from the current location of the line tool.

Parameters:

Type	Parameter	Description
string	`filename`	The name of the emitter file

Example Usage:

```
#
# Create particle traces through part 1 by releasing
# emitters as defined in the given file
#
part: select_begin
 1
part: select_end
ptrace: begin
ptrace: variable VITESSE
ptrace: type streamline
ptrace: create_file /tmp/emitter.file
```

See Also:

EnSight Particle Emitter File Format

Associated ptrace commands

ptrace: create_net

Command:

```
ptrace: create_net <num_in_x> <num_in_y>
```

Purpose:

Create particle traces from the plane tool.

Parameters:

Type	Parameter	Description
int	num_in_x	number of equally spaced points in the x direction of the plane tool
	num_in_y	number of equally spaced points in the y direction of the plane tool

Notes:

The default attributes and/or any "ptrace:" attribute commands between "ptrace: begin" and "ptrace: end" commands are used for the trace creation.

A parent part or parts (containing the flow field) must be selected prior to this command. You need the plane tool to be at the desired location.

The number of emitter points will be the product of num_in_x * num_in_y.

Example Usage:

```
#
# Getting the plane tool to the desired location
#
view_transf: plane 1 1.750000 2.700000 0.000000
view_transf: plane 2 13.000000 2.700000 0.000000
view_transf: plane 3 13.000000 9.500000 0.000000
#
# Selecting the parent part(s)
#
part: select_begin
 1 3 4
part: select_end
#
# Setting some attributes before creation
# (others are the current defaults)
#
ptrace: begin
ptrace: variable VITESSE
ptrace: type streamline
ptrace: component 1.0000e+00 1.0000e+00 1.0000e+00
ptrace: representation line
ptrace: emit_at_current_time ON
ptrace: end
#
# Creating 35 traces from the plane tool
#
ptrace: create_net 5 7
```

See Also:

Associated ptrace commands

Command:

```
ptrace: create_part <part_num>
```

Purpose:

Create particle traces from the nodes of a part.

Parameters:

Type	Parameter	Description
int	part_num	number of the part whose nodes will be used as an emitter

Notes:

The default attributes and/or any "ptrace:" attribute commands between "ptrace: begin" and "ptrace: end" commands are used for the trace creation. A parent part or parts (containing the flow field) must be selected prior to this command. The number of emitter points will be the number of nodes in the part multiplied by the "ptrace: emit_density" value.

Example Usage:

```
#
# Selecting the parent part(s)
#
part: select_begin
 1 3 4
part: select_end
#
# Setting some attributes before creation
# (others are the current defaults)
#
ptrace: begin
ptrace: variable VITESSE
ptrace: type streamline
ptrace: component 1.0000e+00 1.0000e+00 1.0000e+00
ptrace: representation line
ptrace: emit_at_current_time ON
ptrace: emit_density 1.0000e+00
ptrace: end
#
# Creating traces from all the nodes of part 2
#
ptrace: create_part 2
```

See Also:

Associated ptrace commands

ptrace: create_partbyname

Command:

```
ptrace: create_partbyname <part_name>
```

Purpose:

Create a particle trace from the nodes of a part, where the part is referred to by name.

Parameters:

Type	Parameter	Description
string	part_name	name of the part whose nodes will be used as an emitter

Notes:

The default attributes and/or any "ptrace:" attribute commands between "ptrace: begin" and "ptrace: end" commands are used for the trace creation. A parent part or parts (containing the flow field) must be selected prior to this command. The number of emitter points will be the number of nodes in the part multiplied by the "ptrace: emit_density" value.

Example Usage:

```
#
# Selecting the parent part(s)
#
part: select_begin
 1 3 4
part: select_end
#
# Setting some attributes before creation
# (others are the current defaults)
#
ptrace: begin
ptrace: variable VITESSE
ptrace: type streamline
ptrace: component 1.0000e+00 1.0000e+00 1.0000e+00
ptrace: representation line
ptrace: emit_at_current_time ON
ptrace: emit_density 1.0000e+00
ptrace: end
#
# Creating traces from all the nodes of the part
# entitled "(CASE:Case 1)Clip_line"
#
ptrace: create_partbyname (CASE:Case 1)Clip_line
```

See Also:

Associated ptrace commands

Command:

`ptrace: create_pt`

Purpose:

Create a particle trace from the current location of the cursor (a point).

Parameters:

none

Notes:

The default attributes and/or any "`ptrace:`" attribute commands between "`ptrace: begin`" and "`ptrace: end`" commands are used for the trace creation. A parent part or parts (containing the flow field) must be selected prior to this command, and you need the cursor to be at the desired location.

Example Usage:

```
#
# Getting the cursor to the desired location
#
view_transf: cursor -0.927830 1.474020 0.000000
#
# Selecting the parent part(s)
#
part: select_begin
 1 3 4
part: select_end
#
# Setting some attributes before creation
# (others are the current defaults)
#
ptrace: begin
ptrace: variable VITESSE
ptrace: type streamline
ptrace: component 1.0000e+00 1.0000e+00 1.0000e+00
ptrace: representation line
ptrace: emit_at_current_time ON
ptrace: end
#
# Creating trace from a point
#
ptrace: create_pt
```

See Also:

Associated ptrace commands

ptrace: create_rake

Command:

```
ptrace: create_rake <num_points>
```

Purpose:

Create particle traces from the current location of the line tool.

Parameters:

Type	Parameter	Description
int	num_points	number of equally spaced points on the line to use as emitters

Notes:

The default attributes and/or any "ptrace:" attribute commands between "ptrace: begin" and "ptrace: end" commands are used for the trace creation. A parent part or parts (containing the flow field) must be selected prior to this command, and you need the line tool to be at the desired location.

Example Usage:

```
#
# Getting the line tool to the desired location
#
view_transf: line 1 9.000000 6.000000 0.000000
view_transf: line 2 30.000000 6.000000 0.000000
#
# Selecting the parent part(s)
#
part: select_begin
 1 3 4
part: select_end
#
# Setting some attributes before creation
# (others are the current defaults)
#
ptrace: begin
ptrace: variable VITESSE
ptrace: type streamline
ptrace: component 1.0000e+00 1.0000e+00 1.0000e+00
ptrace: representation line
ptrace: emit_at_current_time ON
ptrace: end
#
# Creating 10 traces from a line tool
#
ptrace: create_rake 10
```

See Also:

Associated ptrace commands

Command:

```
ptrace: create_surface_net <num_in_x> <num_in_y> <x1> <y1> <x2> <y2>
```

Purpose:

Create surface restricted particle traces from a screen box pick.

Parameters:

Type	Parameter	Description
int	num_in_x	number of equally spaced points in the x direction of the box pick
	num_in_y	number of equally spaced points in the y direction of the box pick
float	x1	screen x coordinate of the first point on the diagonal of the box (0.0 to 1.0)
	y1	screen y coordinate of the first point on the diagonal of the box (0.0 to 1.0)
	x2	screen x coordinate of the second point on the diagonal of the box (0.0 to 1.0)
	y2	screen y coordinate of the second point on the diagonal of the box (0.0 to 1.0)

Notes:

The "ptrace: surface_restricted" command must be ON prior to this command.

The default attributes and/or any "ptrace:" attribute commands between "ptrace: begin" and "ptrace: end" commands are used for the trace creation. A parent part or parts (containing the flow field) must be selected prior to this command. The number of emitter points will be the product of num_in_x * num_in_y.

Example Usage:

```
part: select_begin
 1 3 4
part: select_end
#
# Setting some attributes before creation
# (surface_restricted must be on)
#
ptrace: begin
ptrace: surface_restricted ON
ptrace: variable VITESSE
ptrace: type streamline
ptrace: component 1.0000e+00 1.0000e+00 1.0000e+00
ptrace: representation line
ptrace: emit_at_current_time ON
ptrace: end
#
# Creating 35 traces from the box selection
#
ptrace: create_surface_net 5 7
```

See Also:

Associated ptrace commands

ptrace: create_surface_pt

Command:

```
ptrace: create_surface_pt <screen_x> <screen_y>
```

Purpose:

Create a surface restricted particle trace from a screen point pick location.

Parameters:

Type	Parameter	Description
float	screen_x	screen x coordinate (0.0 to 1.0)
	screen_y	screen y coordinate (0.0 to 1.0)

Notes:

The "ptrace: surface_restricted" command must be ON prior to this command.

The default attributes and/or any "ptrace:" attribute commands between "ptrace: begin" and "ptrace: end" commands are used for the trace creation. A parent part or parts (containing the flow field) must be selected prior to this command.

Example Usage:

```
#
# Selecting the parent part(s)
#
part: select_begin
 1 3 4
part: select_end
#
# Setting some attributes before creation
# (surface_restricted must be on)
#
ptrace: begin
ptrace: surface_restricted ON
ptrace: variable VITESSE
ptrace: type streamline
ptrace: component 1.0000e+00 1.0000e+00 1.0000e+00
ptrace: representation line
ptrace: emit_at_current_time ON
ptrace: end
#
# Creating surface restricted trace from a screen location
#
ptrace: create_surface_pt 0.148551 0.484398
```

See Also:

Associated ptrace commands

Command:

```
ptrace: create_surface_rake <num_points> <x1> <y1> <x2> <y2>
```

Purpose:

Create surface restricted particle traces from a screen pick rake location.

Parameters:

Type	Parameter	Description
int	num_points	number of equally spaced points along the rake to use as emitters
float	x1	screen x coordinate of the first end of the rake (0.0 to 1.0)
	y1	screen y coordinate of the first end of the rake (0.0 to 1.0)
	x2	screen x coordinate of the second end of the rake (0.0 to 1.0)
	y2	screen y coordinate of the second end of the rake (0.0 to 1.0)

Notes:

The "ptrace: surface_restricted" command must be ON prior to this command.

The default attributes and/or any "ptrace:" attribute commands between "ptrace: begin" and "ptrace: end" commands are used for the trace creation. A parent part or parts (containing the flow field) must be selected prior to this command.

Example Usage:

```
#
# Selecting the parent part(s)
#
part: select_begin
 1 3 4
part: select_end
#
# Setting some attributes before creation
# (surface_restricted must be on)
#
ptrace: begin
ptrace: surface_restricted ON
ptrace: variable VITESSE
ptrace: type streamline
ptrace: component 1.0000e+00 1.0000e+00 1.0000e+00
ptrace: representation line
ptrace: emit_at_current_time ON
ptrace: end
#
# Creating 10 traces from a screen rake location
#
ptrace: create_surface_rake 10 0.117150 0.540862 0.117150 0.442793
```

See Also:

Associated ptrace commands

ptrace: default_emit_file

Command:

```
ptrace: default_emit_file <filename>
```

Purpose:

Set the default filename for file emitters and sets the user interface Emit From option to be File.

Parameters:

Type	Parameter	Description
string	`filename`	The default emitter file name

Example Usage:

```
#
# Set the default file emitter
#
ptrace: default_emit_file /tmp/emitter.file
```

See Also:

EnSight Particle Emitter File Format

Associated ptrace commands

Command:

```
ptrace: default_emit_net <num_in_x> <num_in_y>
```

Purpose:

Set the default number of x and y grid points for net emitters, and sets the user interface Emit From option to be Plane.

Parameters:

Type	Parameter	Description
int	num_in_x	number of emission points to be the default for x direction of net
	num_in_y	number of emission points to be the default for y direction of net

Notes:

Mainly used to set the default "Emit From" option in the user interface, but also sets the default number of x and y grid points. Any subsequent particle traces created using a net will use this unless changed specifically.

Example Usage:

```
#
# Set the default number of emission points
# for nets to be 10 x 12 in this case
#
ptrace: default_emit_net 10 12
```

See Also:

Associated ptrace commands

ptrace: default_emit_part

Command:

```
ptrace: default_emit_part <part_num>
```

Purpose:

Set the default part number for part emitters, and sets the user interface `Emit From` option to be Part.

Parameters:

Type	Parameter	Description
int	`part_num`	default part number to use for part emitters

Notes:

Mainly used to set the default "`Emit From`" in the user interface, but also set the default part number. Any subsequent particle traces created using a part emitter will use this unless changed specifically. This command does not apply if doing parts by name instead of number. (Use "`ptrace: default_emit_partbyname`" if doing parts by name).

Example Usage:

```
#
# Set the default part number for part emitters
# (int this case, setting to part 5)
#
ptrace: default_emit_part 5
```

See Also:

Associated ptrace commands

Command:

```
ptrace: default_emit_partbyname <part_name>
```

Purpose:

Set the default part name for part emitters, and sets the user interface Emit From option to be Part.

Parameters:

Type	Parameter	Description
int	`part_name`	the default part name to use for part emitters

Notes:

Mainly used to set the default "`Emit From`" in the user interface, but also sets the default part name. Any subsequent particle traces created using a part emitter will use this unless changed specifically. This command does not apply if doing parts by number instead of name. (Use "`ptrace: default_emit_part`" if doing parts by number).

Example Usage:

```
#
# Set the default part name for part emitters
# (int this case, setting to a particular clip line part)
#
ptrace: default_emit_partbyname (CASE:Case 1)Clip_line
```

See Also:

Associated ptrace commands

ptrace: default_emit_pt

Command:

```
ptrace: default_emit_pt
```

Purpose:

Set the user interface Emit From option to be Cursor.

Parameters:

none

Notes:

Sets the default "`Emit From`" in the user interface.

Example Usage:

```
ptrace: default_emit_pt
```

See Also:

Associated ptrace commands

Command:

```
ptrace: default_emit_rake <num>
```

Purpose:

Set the default number of emission points for rake emitters, and sets the user interface Emit From option to be Line.

Parameters:

Type	Parameter	Description
int	num	the number of emission points to be the default for rakes

Notes:

Mainly used to set the default "Emit From" option in the user interface, but also sets the default number of rake emission points. Any subsequent particle traces created using a rake will use this unless changed specifically.

Example Usage:

```
#
# Set the default number of emission points
# for rakes to be 10 in this case
#
ptrace: default_emit_rake 10
```

See Also:

Associated ptrace commands

ptrace: delete_emitter

Command:

```
ptrace: delete_emitter
```

Purpose:

Remove emitters from a particle trace part (and their associated traces).

Parameters:

none

Notes:

The desired emitter for the particle trace part must be selected before this command is issued.

Example Usage:

```
#
# Selecting the parent part(s)
#
part: select_begin
 2
part: select_end
#
# Selecting the emitter within the part
# (in this case we will delete the first 3 emitters)
#
ptrace: select_emitter_begin
 0 1 2
ptrace: select_emitter_end
ptrace: delete_emitter
```

See Also:

Associated ptrace commands

Command:

```
ptrace: delta_time <emit_delta>
```

Purpose:

Specifies the time interval between emissions from the emitters.

Parameters:

Type	Parameter	Description
float	`emit_delta`	the time interval between emissions. (The default is 0.0, which will cause only the initial emission to take place

Notes:

Particles will be traced starting at the current time step or the time specified with `ptrace: start_time`. Additional particles will be released every "`emit_delta`" time units.

Most useful for pathline generation.

Example Usage:

```
#
# Selecting the parent part
#
part: select_begin
 1
part: select_end
#
# Setting some attributes before creation
# (the delta_time is being set here, to emit
#  every 2.5 time units)
#
ptrace: begin
ptrace: variable VITESSE
ptrace: type streamline
ptrace: component 1.0000e+00 1.0000e+00 1.0000e+00
ptrace: representation line
ptrace: start_time 1.5000e+01
ptrace: delta_time 2.5000e+00
ptrace: end
#
# Creating trace from a point
#
ptrace: create_pt
```

See Also:

Associated ptrace commands

ptrace: display_offset

Command:

```
ptrace: display_offset <offset_distance>
```

Purpose:

Sets the distance normal to the surface at which the surface restricted trace will be displayed. This can be used to avoid z-buffer display problems that arise because the trace lies in the same plane as the surface.

Parameters:

Type	Parameter	Description
float	`offset_distance`	the distance normal to the surface at which to display the trace

Notes:

A positive number moves the trace in the direction of the positive surface normal, while a negative number moves in the opposite direction.

Example Usage:

```
#
# Selecting the parent part
#
part: select_begin
 1
part: select_end
#
# Setting some attributes before creation
# (the  display_offset for the surface restricted trace
#  is being set here)
#
ptrace: begin
part: description Particle trace part
ptrace: surface_restricted ON
ptrace: variable_offset 1.0000e-01
ptrace: display_offset -0.5000e+00
ptrace: total_time 2.0000e+02
ptrace: emit_at_current_time ON
ptrace: end
#
# Creating trace from a point
#
ptrace: create_pt
```

See Also:

Associated ptrace commands

Command:

```
ptrace: emit_at_current_time <toggle>
```

Purpose:

Sets the emission time for a particle trace to be current time when the particle trace is created. The particle trace is also updated whenever time is changed.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	to emit and update at the current time
		`OFF`	to emit at the set `start_time` and not update when time is changed

Notes:

While this flag is `ON`, the "`ptrace: start_time`" is not used.

Example Usage:

```
#
# Selecting the parent part
#
part: select_begin
 1
part: select_end
#
# Setting some attributes before creation
# (the emit_at_current_time flag is being set here)
#
ptrace: begin
ptrace: variable VITESSE
ptrace: type streamline
ptrace: component 1.0000e+00 1.0000e+00 1.0000e+00
ptrace: representation line
ptrace: emit_at_current_time ON
ptrace: end
#
# Creating trace from a point
#
ptrace: create_pt
```

See Also:

Associated ptrace commands

ptrace: emit_density

Command:

```
ptrace: emit_density <value>
```

Purpose:

Allows for particle trace emissions from a random subset of the nodes in an emitter part instead of all from all nodes.

(Note: this command has been superceded by ptrace: emit_number)

Parameters:

Type	Parameter	Description
float	`value`	A number between 0.0 and 1.0. A value of 1.0 will cause all nodes to emit, 0.5 will cause a random half of the nodes to emit, etc.

Example Usage:

```
#
# Selecting the parent part
#
part: select_begin
 1
part: select_end
#
# Setting some attributes before creation
# (the emit_density value is being set here)
#
ptrace: begin
ptrace: variable VITESSE
ptrace: type streamline
ptrace: component 1.0000e+00 1.0000e+00 1.0000e+00
ptrace: representation line
ptrace: emit_at_current_time ON
ptrace: emit_density 5.0000e-01
ptrace: end
#
# Creating trace with the nodes from part 2 as emitters
# (due to the emit_density command above, only half the
#  nodes will emit)
#
ptrace: create_part 2
```

See Also:

Associated ptrace commands

Command:

```
ptrace: emit_number <value>
```

Purpose:

Specify the number of emissions from a random subset of nodes in an emitter part.

Parameters:

Type	Parameter	Description
int	value	The number of emission points to use

Notes:

The nodes of the emitter part are used. The maximum number of emitter locations that can be created is limited by the number of nodes in the emitter part.

Example Usage:

```
#
# Create particle traces through part 1 by releasing
# 25 emitters off of part 2
#
part: select_begin
 1
part: select_end
ptrace: begin
ptrace: variable VITESSE
ptrace: type streamline
ptrace: emit_number 25
ptrace: create_part 2
```

See Also:

Associated ptrace commands

ptrace: massed_drag

Command:

```
ptrace: massed_drag <toggle>
```

Purpose:

Specify whether the drag force term is used in the massed-particle calculation.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	use drag as part of the forces on the particle
		OFF	do not use drag as part of the forces on the particle

Notes:

The default setting is ON. The "ptrace: massed_particles" command does not have to be ON in order to specify this command. Drag force term in the massed-particle momentum balance equation is computed according to the ptrace: massed_pressure_gradient setting.

Example Usage:

```
#
# Create a massless point trace
#
part: select_all
ptrace: select_default
part: modify_begin
ptrace: variable velocity
part: modify_end
part: select_all
ptrace: create_pt
part: select_begin
 2
part: select_end
#
# Toggle on massed-particle drag force term
# Specifiy default drag coefficient function table
#
part: modify_begin
ptrace: massed_drag ON
ptrace: massed_drag_coefficient none
part: modify_end
#
# Change part to massed trace using current settings
#
part: modify_begin
ptrace: massed_particles ON
part: modify_end
```

See Also:

Associated ptrace commands

Command:

```
ptrace: massed_drag_coefficient <table_name>
```

Purpose:

Specify the name of the drag coefficient function table to be used in the drag force term of the massed-particle calculation.

Parameters:

Type	Parameter	Description	
string	table_name	none	defaults to hard coded DragCoefTable1
		DragCoefTable1	drag coefficient computed by table
		DragCoefPower	drag coefficient computed by power function
		DragCoefPoly	drag coefficient computed by polynomial function
		any User-Defined Math Function	drag coefficient computed by a user supplied function

Notes:

The default setting is "none". The "ptrace: massed_particles" command does not have to be ON in order to specify this command. This parameter is only used when the drag force term "ptrace: massed_drag" is toggled ON.

Also the three options "DragCoefPoly", "DragCoefPower", and "DragCoefTable1", are EnSight User-Defined Math Functions. These math functions are typically located under $CEI_HOME/ensight76/src/math_functions, and are compiled as libraries by the user, and then dynamically liked via the ENSIGHT7_UDMF environment variable.

Example Usage:

```
#
# Create a massless point trace
#
part: select_all
ptrace: select_default
part: modify_begin
ptrace: variable velocity
part: modify_end
part: select_all
ptrace: create_pt
part: select_begin
 2
part: select_end
#
# Toggle on massed-particle drag force term
# Specifiy default drag coefficient function table
#
part: modify_begin
ptrace: massed_drag ON
ptrace: massed_drag_coefficient none
part: modify_end
#
# Change part to massed trace using current settings
#
part: modify_begin
ptrace: massed_particles ON
```

```
part: modify_end
#
# Change the drag coefficient funtion to use the power formulation
#
part: modify_begin
ptrace: massed_drag_coefficient DragCoefPower
part: modify_end
```

See Also:

ptrace: massed_drag <toggle>

Associated ptrace commands

Command:

```
ptrace: massed_fluid_density <density>
```

Purpose:

Specify the density of the corresponding fluid field where the massed-particles are traced.

Parameters:

Type	Parameter	Description
float	density	the fluid density constant value of the corresponding fluid field
or		
string	density	the fluid density variable name of the corresponding fluid field - defaults to "none" when constant value used

Notes:

The default setting for this parameter is the constant value of "1.0", with the Fluid Density variable name set to "none". This indicates that no variable name is used; rather the constant density is used in the calculations instead. The "ptrace: massed_particles" command does not have to be ON in order to specify this command. This parameter is figured only when the drag and gravity force terms of the massed-particle momentum balance equation are used.

Example Usage:

```
#
# Create a massless point trace
#
part: select_all
ptrace: select_default
part: modify_begin
ptrace: variable velocity
part: modify_end
part: select_all
ptrace: create_pt
part: select_begin
 2
part: select_end
#
# Specifiy density scalar field variable name of the fluid.
#
part: modify_begin
ptrace: massed_fluid_density density
part: modify_end
#
# Change part to massed trace using current settings
#
part: modify_begin
ptrace: massed_particles ON
part: modify_end
```

See Also:

Associated ptrace commands

ptrace: massed_fluid_viscosity

Command:

```
ptrace: massed_fluid_viscosity <viscosity>
```

Purpose:

Specify the dynamic viscosity of the corresponding fluid field where the massed-particles are traced.

Parameters:

Type	Parameter	Description
float	viscosity	the dynamic viscosity constant value of the corresponding fluid field
or		
string	viscosity	the dynamic viscosity variable name of the corresponding fluid field - defaults to "none" when constant value used

Notes:

The default setting for this parameter is the constant value of "1.9620e+05", with the Fluid Dynamic Viscosity variable name set to "none". This indicates that no variable name is used; rather the constant dynamic viscosity is used in the calculations instead. The "ptrace: massed_particles" command does not have to be ON in order to specify this command. This parameter is figured only when the drag force term of the massed-particle momentum balance equation is used.

Example Usage:

```
#
# Create a massless point trace
#
part: select_all
ptrace: select_default
part: modify_begin
ptrace: variable velocity
part: modify_end
part: select_all
ptrace: create_pt
part: select_begin
 2
part: select_end
#
# Specifiy dynamic viscosity scalar field variable name of the fluid.
#
part: modify_begin
ptrace: massed_fluid_viscosity dynamic_viscosity
part: modify_end
#
# Change part to massed trace using current settings
#
part: modify_begin
ptrace: massed_particles ON
part: modify_end
```

See Also:

ptrace: massed_drag <toggle>

Associated ptrace commands

Command:

```
ptrace: massed_gravity <toggle>
```

Purpose:

Specify whether the gravity force term is used in the massed-particle calculation.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	use gravity as part of the forces on the particle
		OFF	do not use gravity as part of the forces on the particle

Notes:

The default setting is ON. The "ptrace: massed_particles" command does not have to be ON in order to specify this command. If this command is ON, then the gravity force term in the massed-particle momentum balance equation is computed according to the "ptrace: massed_gravity_vector" setting.

Example Usage:

```
#
# Create a massless point trace
#
part: select_all
ptrace: select_default
part: modify_begin
ptrace: variable velocity
part: modify_end
part: select_all
ptrace: create_pt
part: select_begin
 2
part: select_end
#
# Toggle on massed-particle gravity force term
# Specifiy gravity vector
#
part: modify_begin
ptrace: massed_gravity ON
ptrace: massed_gravity_vector 0.00000e+00-9.81000e+00 0.00000e+00
part: modify_end
#
# Change part to massed trace using current settings
#
part: modify_begin
ptrace: massed_particles ON
part: modify_end
```

See Also:

Associated ptrace commands

ptrace: massed_gravity_vector

Command:

```
ptrace: massed_gravity_vector <gx> <gy> <gz>
```

Purpose:

Specify the name of the gravity vector used in the pressure force term of the massed-particle calculation.

Parameters:

Type	Parameter	Description
float	gx	the x-component of the gravity vector
	gy	the y-component of the gravity vector
	gz	the z-component of the gravity vector

Notes:

The default setting is <0., -9.81, 0.>. The "`ptrace: massed_particles`" command does not have to be `ON` in order to specify this command.

Example Usage:

```
#
# Create a massless point trace
#
part: select_all
ptrace: select_default
part: modify_begin
ptrace: variable velocity
part: modify_end
part: select_all
ptrace: create_pt
part: select_begin
 2
part: select_end
#
# Toggle on massed-particle gravity force term
# Specifiy the gravity vector
#
part: modify_begin
ptrace: massed_gravity ON
ptrace: massed_gravity_vector 0.00000e+00-9.81000e+00 0.00000e+00
part: modify_end
#
# Change part to massed trace using current settings
#
part: modify_begin
ptrace: massed_particles ON
part: modify_end
```

See Also:

ptrace: massed_gravity <toggle>

Associated ptrace commands

Command:

```
ptrace: massed_initial_use_fluid <toggle>
```

Purpose:

Specify whether to use the velocity of the fluid at the particle emitter location as the initial velocity of the massed-particle.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	use the fluid velocity at the start of each trace
		OFF	use the specified velocity at the start of each trace

Notes:

The default value is ON. The "ptrace: massed_particles" command does not have to be ON in order to specify this command. This parameter is figured in all terms of the massed-particle momentum balance equation. If the value of this command is OFF, then the massed-particle emitter will use the velocity specified via the "ptrace: massed_initial_velocity" command.

Example Usage:

```
#
# Create a massless point trace
#
part: select_all
ptrace: select_default
part: modify_begin
ptrace: variable velocity
part: modify_end
part: select_all
ptrace: create_pt
part: select_begin
 2
part: select_end
#
# Do not use the field velocity as the initial particle velocity
# Specify the initial velocity value for all the massed-particles
#
part: modify_begin
ptrace: massed_initial_use_fluid OFF
ptrace: massed_initial_velocity 1.0000e+00 1.0000e+00 1.0000e+00
part: modify_end
#
# Change part to massed trace using current settings
#
part: modify_begin
ptrace: massed_particles ON
part: modify_end
```

See Also:

Associated ptrace commands

ptrace: massed_initial_velocity

Command:

```
ptrace: massed_initial_velocity <vx> <vy> <vz>
```

Purpose:

Specify the initial emitter velocity vector to be used on all the massed-particles.

Parameters:

Type	Parameter	Description
float	vx	the x-component of the initial velocity vector
	vy	the y-component of the initial velocity vector
	vz	the z-component of the initial velocity vector

Notes:

The default vector value is <1,1,1>. The "`ptrace: massed_particles`" command does not have to be `ON` in order to specify this command. This parameter is figured in all terms of the massed-particle momentum balance equation. The values of this command take effect only if "`ptrace: massed_initial_use_fluid`" is toggled `OFF`.

Example Usage:

```
#
# Create a massless point trace
#
part: select_all
ptrace: select_default
part: modify_begin
ptrace: variable velocity
part: modify_end
part: select_all
ptrace: create_pt
part: select_begin
 2
part: select_end
#
# Do not use the field velocity as the initial particle velocity
# Specify the initial velocity value for all the massed-particles
#
part: modify_begin
ptrace: massed_initial_use_fluid OFF
ptrace: massed_initial_velocity 1.0000e+00 1.0000e+00 1.0000e+00
part: modify_end
#
# Change part to massed trace using current settings
#
part: modify_begin
ptrace: massed_particles ON
part: modify_end
```

See Also:

Associated ptrace commands

Command:

```
ptrace: massed_particles <toggle>
```

Purpose:

Enable or disable massed-particle trace calculation.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	turn on massed force terms
		OFF	do not use massed force terms

Notes:

The default value is OFF. The "ptrace: massed_particles" command does not have to be ON in order to specify the other massed-particle commands.

Example Usage:

```
#
# Create a massless point trace
#
part: select_all
ptrace: select_default
part: modify_begin
ptrace: variable velocity
part: modify_end
part: select_all
ptrace: create_pt
part: select_begin
 2
part: select_end
#
# Change part to massed trace using current settings
#
part: modify_begin
ptrace: massed_particles ON
part: modify_end
```

See Also:

Associated ptrace commands

ptrace: massed_particle_density

Command:

```
ptrace: massed_particle_density <density>
```

Purpose:

Specify the density of all the massed-particles.

Parameters:

Type	Parameter	Description
float	density	the density value of all the particles

Notes:

The default value is 1.0e+03. The "ptrace: massed_particles" command does not have to be ON in order to specify this command. This parameter is figured in all terms of the massed-particle momentum balance equation. Water particle densities are on the order of 1.0e+3, and air partilce densities are on the order of 1.0.

Example Usage:

```
#
# Create a massless point trace
#
part: select_all
ptrace: select_default
part: modify_begin
ptrace: variable velocity
part: modify_end
part: select_all
ptrace: create_pt
part: select_begin
 2
part: select_end
#
# Specifiy density of all the massed particles
#
part: modify_begin
ptrace: massed_particle_density 1.0000e+03
part: modify_end
#
# Change part to massed trace using current settings
#
part: modify_begin
ptrace: massed_particles ON
part: modify_end
```

See Also:

Associated ptrace commands

Command:

```
ptrace: massed_particle_diameter <diameter>
```

Purpose:

Specify the diameter of all the massed-particles.

Parameters:

Type	Parameter	Description
float	diameter	the diameter value of all the particles

Notes:

The default value is 1.0e-03. The "ptrace: massed_particles" command does not have to be ON in order to specify this command. This parameter is figured in all terms of the massed-particle momentum balance equation. Water particle diameters are on the order of 1.0e-3, and air partilce diameters are on the order of 1.0e-6.

Example Usage:

```
#
# Create a massless point trace
#
part: select_all
ptrace: select_default
part: modify_begin
ptrace: variable velocity
part: modify_end
part: select_all
ptrace: create_pt
part: select_begin
 2
part: select_end
#
# Specifiy diameter of the massed particle
#
part: modify_begin
ptrace: massed_particle_diameter 1.0000e-03
part: modify_end
#
# Change part to massed trace using current settings
#
part: modify_begin
ptrace: massed_particles ON
part: modify_end
```

See Also:

Associated ptrace commands

ptrace: massed_pressure

Command:

```
ptrace: massed_pressure <toggle>
```

Purpose:

Specify whether the pressure force term is used in the massed-particle calculation.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	use pressure as part of the forces on the particle
		OFF	do not use pressure as part of the forces on the particle

Notes:

The default setting is OFF. The "ptrace: massed_particles" command does not have to be ON in order to specify this command. If this command is ON, then the pressure force term in the massed-particle momentum balance equation is computed according to the "ptrace: massed_pressure_gradient" setting.

Example Usage:

```
#
# Create a massless point trace
#
part: select_all
ptrace: select_default
part: modify_begin
ptrace: variable velocity
part: modify_end
part: select_all
ptrace: create_pt
part: select_begin
 2
part: select_end
#
# Toggle on massed-particle pressure force term
# Specifiy pressure gradient variable
#
part: modify_begin
ptrace: massed_pressure ON
ptrace: massed_pressure_gradient grad_pressure
part: modify_end
#
# Change part to massed trace using current settings
#
part: modify_begin
ptrace: massed_particles ON
part: modify_end
```

See Also:

Associated ptrace commands

Command:

```
ptrace: massed_pressure_gradient <variable_name>
```

Purpose:

Specify the name of the pressure gradient variable used in the pressure force term of the massed-particle calculation.

Parameters:

Type	Parameter	Description

string	`variable_name`
	or
	`none`
		the name of the pressure gradient variable

Notes:

The "`ptrace: massed_particles`" command does not have to be `ON` in order to specify this command. If "`none`" is specified, then the pressure force term is ignored in the massed-particle calculation. The default setting is "`none`".

Example Usage:

```
#
# Create a massless point trace
#
part: select_all
ptrace: select_default
part: modify_begin
ptrace: variable velocity
part: modify_end
part: select_all
ptrace: create_pt
part: select_begin
 2
part: select_end
#
# Toggle on massed-particle pressure force term
# Specifiy pressure gradient variable
#
part: modify_begin
ptrace: massed_pressure ON
ptrace: massed_pressure_gradient grad_pressure
part: modify_end
#
# Change part to massed trace using current settings
#
part: modify_begin
ptrace: massed_particles ON
part: modify_end
```

See Also:

ptrace: massed_pressure <toggle>

Associated ptrace commands

ptrace: max_angle

Command:

```
ptrace: max_angle <value>
```

Purpose:

Sets the maximum angle between successive particle trace line segments that will trigger a halving of the integration step.

Parameters:

Type	Parameter	Description
float	`value`	desired maximum angle (in degrees). If the angle between successive segments is greater than this, the integration step will be halved. The range is 1.0 to 90.0 degrees

Notes:

When a particle trace starts curving, the integration step typically needs to be decreased. This attribute is one of the controls for such adaptation.

Example Usage:

```
#
# Selecting the trace(s)
#
part: select_begin
 7 8
part: select_end
#
# Modifying attributes for the two selected trace parts
# (max_angle is modified in here)
#
part: modify_begin
ptrace: min_steps 3
ptrace: min_angle 3.0000e+00
ptrace: max_angle 2.0000e+01
ptrace: rot_angle 3.2000e+01
#
# Updates the traces with the modified attributes
#
part: modify_end
```

See Also:

Associated ptrace commands

Command:

```
ptrace: min_angle <value>
```

Purpose:

Sets the minimum angle between successive particle trace line segments that will trigger a doubling of the integration step.

Parameters:

Type	Parameter	Description
float	`value`	desired minimum angle (in degrees). If the angle between successive segments is less than this, the integration step will be doubled. The range is 1.0 to 90.0 degrees

Notes:

When a particle trace is not curving, the integration step can typically be increased. This attribute is one of the controls for such adaptation.

Example Usage:

```
#
# Selecting the trace(s)
#
part: select_begin
 7 8
part: select_end
#
# Modifying attributes for the two selected trace parts
# (min_angle is modified in here)
#
part: modify_begin
ptrace: min_steps 3
ptrace: min_angle 3.0000e+00
ptrace: max_angle 2.0000e+01
ptrace: rot_angle 3.2000e+01
#
# Updates the traces with the modified attributes
#
part: modify_end
```

See Also:

Associated ptrace commands

ptrace: min_steps

Command:

```
ptrace: min_steps <num>
```

Purpose:

Sets the minimum number of integration steps to perform in each element

Parameters:

Type	Parameter	Description
int	num	desired minimum number of integration steps per element (range is 2 to 20)

Notes:

A higher number increases the accuracy of the integration, but is more compute intensive and produces more segments to display.

Example Usage:

```
#
# Selecting the trace(s)
#
part: select_begin
 7 8
part: select_end
#
# Modifying attributes for the two selected trace parts
# (min_steps being modified in here)
#
part: modify_begin
ptrace: min_steps 3
ptrace: min_angle 3.0000e+00
ptrace: max_angle 2.0000e+01
ptrace: rot_angle 3.2000e+01
#
# Updates the traces with the modified attributes
#
part: modify_end
```

See Also:

Associated ptrace commands

Command:

```
ptrace: modify_emit_totoolloc <part_num> <emitter>
```

Purpose:

Change the location of the emitter in a particle trace part.

Parameters:

Type	Parameter	Description
int	part_num	particle trace part number
	emitter	emitter number (0-based)

Example Usage:

```
#
# Change the location of the tool used for the emitter
#
view_transf: cursor 7.422632 -0.327560 0.000000
#
# modify the trace emitter based on this tool
# (in this case, it is emitter 0 in part 2
#
ptrace: modify_emit_totoolloc 2 0
```

See Also:

Associated ptrace commands

ptrace: modify_fileemit_name

Command:

```
ptrace: modify_fileemit_name <emitter> <filename>
```

Purpose:

Change the emitter file name for a file emitter.

Parameters:

Type	Parameter	Description
int	`emitter`	Emitter number (0 based)
string	`filename`	The new emitter file name

Example Usage:

```
#
# Select an existing particle trace part 3
#
#
part: select_begin
 3
part: select_end
#
# Change the emitter name for emitter 0
#
ptrace: modify_fileemit_name 0 /tmp/new_emitter.file
```

See Also:

EnSight Particle Emitter File Format

Associated ptrace commands

Command:

```
ptrace: modify_netemit_points <emitter> <new_num_in_x> <new_num_in_y>
```

Purpose:

Change the number of emission points for a particular net emitter in a particle trace part.

Parameters:

Type	Parameter	Description
int	emitter	emitter number (0-based)
	new_num_in_x	desired new number of emission points for the grid x direction.
	new_num_in_y	desired new number of emission points for the grid y direction

Example Usage:

```
#
# Selecting an existing particle trace part
# (which contains some net emitters)
#
part: select_begin
 3
part: select_end
#
# Changing the number of points in the
# first 2 net emitters (emitters 0 and 1)
# to 5 x 6, and 9 x 15 respectively
#
ptrace: modify_netemit_points 0 5 6
ptrace: modify_netemit_points 1 9 15
```

See Also:

Associated ptrace commands

ptrace: modify_partemit_byname

Command:

```
ptrace: modify_partemit_byname <emitter> <new_part_name>
```

Purpose:

Change the part name for a particular part emitter in a particle trace part.

Parameters:

Type	Parameter	Description
int	`emitter`	the emitter number (0-based)
	`new_part_name`	name of the new part from which emissions from nodes will be made

Example Usage:

```
#
# Selecting an existing particle trace part
# (which contains some part emitters)
#
part: select_partname_begin
"(CASE:Case 1)Particle trace part"
part: select_partname_end
#
# Changing emitter 1 to use the nodes
# from clip line2 part
#
ptrace: modify_partemit_byname 1 (CASE:Case 1)Clip_line2
```

See Also:

Associated ptrace commands

Command:

```
ptrace: modify_partemit_num <emitter> <new_part_num>
```

Purpose:

Change the part number for a particular part emitter in a particle trace part.

Parameters:

Type	Parameter	Description
int	emitter	emitter number (0-based)
	new_part_num	new part number from which emissions from nodes will be made

Example Usage:

```
#
# Selecting an existing particle trace part
# (which contains some part emitters)
#
part: select_begin
 3
part: select_end
#
# Changing emitter 3 to use the nodes
# from part 12
#
ptrace: modify_partemit_num 3 12
```

See Also:

Associated ptrace commands

ptrace: modify_rakeemit_points

Command:

```
ptrace: modify_rakeemit_points <emitter> <new_num_pts>
```

Purpose:

Change the number of emission points for a particular rake emitter in a particle trace part.

Parameters:

Type	Parameter	Description
int	emitter	emitter number (0-based
	new_num_pts	desired new number of emission points along the rake

Example Usage:

```
#
# Selecting an existing particle trace part
# (which contains a rake emitter)
#
part: select_begin
 3
part: select_end
#
# Changing the number of points in the
# first emitter (emitter 0), to 8
#
ptrace: modify_rakeemit_points 0 8
```

See Also:

Associated ptrace commands

Command:

```
ptrace: representation <rep>
```

Purpose:

Sets the display representation for a particle trace.

Parameters:

Type	Parameter	Description	
constant	rep	line	the trace will be drawn as a line
		ribbon	the trace will be drawn as a ribbon
		square_tubes	the trace be drawn as a square tube

Notes:

Use the ribbon or square_tubes representations whenever rotational effects need to be seen. Surface restricted traces can only be displayed in line representation.

Example Usage:

```
#
# Selecting the trace(s)
#
part: select_begin
 7 8
part: select_end
#
# Modifying attributes for the two selected trace parts
# (representation is modified in here)
#
part: modify_begin
ptrace: representation ribbon
ptrace: ribbon_width 2.0000e+00
#
# Updates the traces with the modified attributes
#
part: modify_end
```

See Also:

Associated ptrace commands

ptrace: ribbon_width

Command:

```
ptrace: ribbon_width <width>
```

Purpose:

Sets the width for ribbon or square tube particle trace representations.

Parameters:

Type	Parameter	Description
float	`width`	the width of the ribbon or square tube

Notes:

This attribute has no effect when using the "line" trace representation.

Example Usage:

```
#
# Selecting the trace(s)
#
part: select_begin
 7 8
part: select_end
#
# Modifying attributes for the two selected trace parts
# (ribbon_width is modified in here)
#
part: modify_begin
ptrace: representation square_tubes
ptrace: ribbon_width 2.0000e+00
#
# Updates the traces with the modified attributes
#
part: modify_end
```

See Also:

Associated ptrace commands

Command:

```
ptrace: rot_angle <value>
```

Purpose:

Sets the angle between successive particle trace line segments that will trigger a halving of the integration step.

Parameters:

Type	Parameter	Description
float	value	desired rotational angle (in degrees). If the change in rotation angle is greater than this, the integration step will be halved. range is 1.0 to 90.0 degrees

Notes:

When a particle begins experiencing rotational effects, the integration step typically needs to be decreased. This attribute is one of the controls for such adaptation.

Example Usage:

```
#
# Selecting the trace(s)
#
part: select_begin
 7 8
part: select_end
#
# Modifying attributes for the two selected trace parts
# (rot_angle is modified in here)
#
part: modify_begin
ptrace: min_steps 3
ptrace: min_angle 3.0000e+00
ptrace: max_angle 2.0000e+01
ptrace: rot_angle 3.2000e+01
#
# Updates the traces with the modified attributes
#
part: modify_end
```

See Also:

Associated ptrace commands

ptrace: select_default

Command:

```
ptrace: select_default
```

Purpose:

Sets the currently selected particle trace parts to none, so any following attribute modifications will change the particle trace defaults.

Parameters:

none

Notes:

When particle trace parts are created, the defaults, and any "`ptrace:`" attribute commands between "`ptrace: begin`" and "`ptrace: end`", are used in the creation.

Example Usage:

```
#
# Setting to default particle traces
#
ptrace: select_default
#
# modifying some attributes
#
part: modify_begin
ptrace: variable VITESSE
ptrace: type pathline
part: modify_end
```

See Also:

Associated ptrace commands

Command:

```
ptrace: select_emitter_begin
 <emitter0 emitter1 ... emitterN>
ptrace: select_emitter_end
```

Purpose:

Select emitters within a particle trace part for the purpose of modifying emitter attributes, deleting, etc.

Parameters:

Type	Parameter	Description
int	`emitterN`	emitter numbers (0-based)

Example Usage:

```
#
# Selecting the parent part(s)
#
part: select_begin
 2
part: select_end
#
# Selecting the first 3 emitters within the part
# for the purpose of deleting them
#
ptrace: select_emitter_begin
 0 1 2
ptrace: select_emitter_end
ptrace: delete_emitter
```

See Also:

Associated ptrace commands

ptrace: start_time

Command:

```
ptrace: start_time <emit_time>
```

Purpose:

Specifies the solution time at which to begin particle emission.

Parameters:

Type	Parameter	Description
float	`emit_time`	solution time for particle emission. Must be in the range of valid solution times

Example Usage:

```
#
# Selecting the parent part
#
part: select_begin
 1
part: select_end
#
# Setting some attributes before creation
# (the start_time is being set here)
#
ptrace: begin
ptrace: variable VITESSE
ptrace: type streamline
ptrace: component 1.0000e+00 1.0000e+00 1.0000e+00
ptrace: representation line
ptrace: start_time 1.5000e+01
ptrace: delta_time 1.0000e+00
ptrace: end
#
# Creating trace from a point
#
ptrace: create_pt
```

See Also:

Associated ptrace commands

Command:

```
ptrace: surface_restricted <toggle>
```

Purpose:

Toggles trace creation to be surface restricted or not.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	for surface restricted traces
		OFF	for regular traces

Example Usage:

```
#
# Selecting the parent part
#
part: select_begin
 1
part: select_end
#
# Setting some attributes before creation
# (the surface_restricted flag is being set here)
#
ptrace: begin
part: description Particle trace part
ptrace: surface_restricted ON
ptrace: variable_offset 1.0000e-01
ptrace: display_offset 0.5000e+00
ptrace: total_time 2.0000e+02
ptrace: emit_at_current_time ON
ptrace: end
#
# Creating trace from a point
#
ptrace: create_pt
```

See Also:

Associated ptrace commands

ptrace: total_time

Command:

```
ptrace: total_time <limit>
```

Purpose:

Sets the maximum length of time that the particle trace is allowed to continue.

Parameters:

Type	Parameter	Description
float	`limit`	total time limit

Notes:

A particle trace may terminate earlier than this for other reasons, but this limit avoids infinite calculations in recirculation zones.

Example Usage:

```
#
# Selecting the trace(s)
#
part: select_begin
 7 8
part: select_end
#
# Modifying attributes for the two selected trace parts
# (total_time is modified in here)
#
part: modify_begin
ptrace: total_time 2.0000e+02
ptrace: trace_direction +/-
#
# Updates the traces with the modified attributes
#
part: modify_end
```

See Also:

Associated ptrace commands

Command:

```
ptrace: trace_direction <direction>
```

Purpose:

Sets the integration direction for a particle trace - forward in time, backward in time, or both.

Parameters:

Type	Parameter	Description	
constant	direction	+	to trace forward in time
		-	to trace backward in time
		+/-	to trace both forward and backward in time

Notes:

By setting this option appropriately, one can see where a particle will go if released from the emitter location (the + direction), or one can see the path that a particle took to get to the emitter location (the - direction).

Example Usage:

```
#
# Selecting the trace(s)
#
part: select_begin
 7 8
part: select_end
#
# Modifying attributes for the two selected trace parts
# (trace_direction is modified in here)
#
part: modify_begin
ptrace: total_time 2.0000e+02
ptrace: trace_direction +/-
#
# Updates the traces with the modified attributes
#
part: modify_end
```

See Also:

Associated ptrace commands

ptrace: type

Command:

```
ptrace: type <option>
```

Purpose:

Sets the type of particle trace that will be created.

Parameters:

Type	Parameter	Description	
constant	option	streamline	trace will be integrated in the flow field at a specific time
		pathline	trace will be integrated in the changing flow field

Notes:

Pathline is only available for transient data cases.

Example Usage:

```
#
# Setting default particle trace variable
#
ptrace: select_default
part: modify_begin
ptrace: variable VITESSE
part: modify_end
#
# Selecting the parent part
#
part: select_begin
 1
part: select_end
#
# Setting some attributes before creation
# (the type is being set here)
#
ptrace: begin
ptrace: type streamline
ptrace: component 1.0000e+00 1.0000e+00 1.0000e+00
ptrace: representation line
ptrace: end
#
# Creating trace from a point
#
ptrace: create_pt
```

See Also:

Associated ptrace commands

Command:

```
ptrace: variable <name>
```

Purpose:

Specifies the vector variable to use for the flow field for the particle trace.

Parameters:

Type	Parameter	Description
string	name	name of the vector variable

Example Usage:

```
#
# Selecting the parent part
#
part: select_begin
 1
part: select_end
#
# Setting some attributes before creation
# (the variable is being set here)
#
ptrace: begin
ptrace: variable VITESSE
ptrace: type streamline
ptrace: component 1.0000e+00 1.0000e+00 1.0000e+00
ptrace: representation line
ptrace: end
#
# Creating trace from a point
#
ptrace: create_pt
```

See Also:

Associated ptrace commands

ptrace: variable_offset

Command:

```
ptrace: variable_offset <offset_distance>
```

Purpose:

Sets the distance from the surface into the flow field at which the velocity variable will be sampled. The tangential component of the variable at this offset distance is the field that will be used to trace through for surface restricted traces.

Parameters:

Type	Parameter	Description
float	`offset_distance`	distance into the field

Notes:

A positive number moves the distance in the direction of the positive normal of the surface, while a negative number moves in the opposite direction. If there is only flow on one side of the surface, the program will flip the sign internally.

Used only for surface restricted particle traces.

Example Usage:

```
#
# Selecting the parent part
#
part: select_begin
 1
part: select_end
#
# Setting some attributes before creation
# (the variable_offset for the surface_restricted trace
#  is being set here)
#
ptrace: begin
part: description Particle trace part
ptrace: surface_restricted ON
ptrace: variable_offset 1.0000e-01
ptrace: display_offset 0.5000e+00
ptrace: total_time 2.0000e+02
ptrace: emit_at_current_time ON
ptrace: end
#
# Creating trace from a point
#
ptrace: create_pt
```

See Also:

Associated ptrace commands

Commands in this section:

query_ent_var

query_ent_var: #_of_sample_pts <total_points>
query_ent_var: auto_plot_checkexisting <toggle>
query_ent_var: auto_plot_queries <toggle>
query_ent_var: begin
query_ent_var: end
query_ent_var: modify_begin
query_ent_var: modify_end
query_ent_var: begin_simtime <time_value>
query_ent_var: end_simtime <time_value>
query_ent_var: constrain <option>
query_ent_var: cursor_loc <x_coord> <y_coord> <z_coord>
query_ent_var: description <desc>
query_ent_var: distance <option>
query_ent_var: elem_id <element_id_#>
query_ent_var: generate_over <type>
query_ent_var: ijk <i-plane_#> <j-plane_#> <k-plane_#>
query_ent_var: line_loc <end_point_id> <x_comp> <y_comp> <z_comp>
query_ent_var: marker_rgb <red_val> <grn_val> <blu_val>
query_ent_var: marker_size_normalized <size_value>
query_ent_var: marker_visible <toggle>
query_ent_var: modify_begin
query_ent_var: modify_end
query_ent_var: multiple_segments_by <option>
query_ent_var: node_id <node_id_#>
query_ent_var: operation_factor1 <scale>
query_ent_var: operation_factor2 <scale>
query_ent_var: operation_query1byname <name>
query_ent_var: operation_query2byname <name>
query_ent_var: origin <x_coord> <y_coord> <z_coord>
query_ent_var: part_id <part_id_#>
query_ent_var: part_idbyname <part_name>
query_ent_var: query
query_ent_var: query_type <type>
query_ent_var: sample_by <option>
query_ent_var: select_all_timesteps
query_ent_var: type <domain>
query_ent_var: variable_1 <name>
query_ent_var: variable_2 <name>

See Also:

User Manual - Section 7.11, Query/Plot

How To Query/Plot

How To Change Plot Attributes

query_ent_var: #_of_sample_pts

Command:

```
query_ent_var: #_of_sample_pts <total_points>
```

Purpose:

Specify the number of evenly spaced points to sample either along the line tool or over the specified range of time steps.

Parameters:

Type	Parameter	Description
int	`total_points`	For queries over Distance using the Line Tool, this specifies the number of equally spaced points to query along the line (default = 20). For queries Over Time, this specifies how many evenly timed moments over the specified range of time steps at which to query - if left blank, you get a sample point at each time step (default = # of time steps).

Notes:

This command must lie between either a "`query_ent_var: begin`" and "`query_ent_var: end`" pair, or a "`query_ent_var: modify_begin`" and "`query_ent_var: modify_end`" pair.

If you specify more or fewer sample points than the number of time steps, EnSight linearly interpolates between the adjoining time steps.

If the query is an FFT sampling, the number of frequencies output will be (a power of two) less than or equal to the number of sample points.

Example Usage:

```
#
# Creation example...
#
part: select_begin
1
part: select_end
query_ent_var: begin
query_ent_var: description
query_ent_var: type distance
query_ent_var: #_of_sample_pts 20
query_ent_var: constrain line_tool
query_ent_var: line_loc 1 2.7770e-01 7.7363e-02 1.1348e-02
query_ent_var: line_loc 2 7.6144e-01 1.7007e-01 5.5697e-02
query_ent_var: variable Density
query_ent_var: end
query_ent_var: query
curve: select_begin
0
curve: select_end
#
# Modification example...
#
query_ent_var: modify_begin
query_ent_var: type distance
query_ent_var: #_of_sample_pts 20
query_ent_var: constrain line_tool
```

```
query_ent_var: modify_end
```

See Also:

Associated query_ent_var commands

query_ent_var: auto_plot_checkexisting

Command:

```
query_ent_var: auto_plot_checkexisting <toggle>
```

Purpose:

Specifies to check for existing plotters with the same abscissa when auto plotting query information.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	assign the query to any existing plotter with same abscissa
		`OFF`	assign to a new plotter

Notes:

Only applicable with the "`query_ent_var: auto_plot_queries ON`" command.

Example Usage:

```
query_ent_var: auto_plot_checkexisting ON
```

See Also:

Associated query_ent_var commands

Command:

```
query_ent_var: auto_plot_queries <toggle>
```

Purpose:

Specify that any created query to be plotted automatically.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	plot the query automatically
		OFF	plot the query via manual specification

Notes:

See also the "query_ent_var: auto_plot_checkexisting" command.

Example Usage:

```
query_ent_var: auto_plot_queries ON
```

See Also:

Associated query_ent_var commands

query_ent_var: begin/end

Command:

```
query_ent_var: begin
query_ent_var: end
```

Purpose:

Delimit the modifications for query attributes.

Parameters:

None.

Notes:

These commands are used to set the attributes for a query creation.

Example Usage:

```
part: select_begin
1
part: select_end
query_ent_var: begin
query_ent_var: description
query_ent_var: type distance
query_ent_var: #_of_sample_pts 20
query_ent_var: constrain line_tool
query_ent_var: line_loc 1 2.7770e-01 7.7363e-02 1.1348e-02
query_ent_var: line_loc 2 7.6144e-01 1.7007e-01 5.5697e-02
query_ent_var: variable Density
query_ent_var: end
query_ent_var: query
curve: select_begin
0
curve: select_end
```

See Also:

Associated query_ent_var commands

Command:

```
query_ent_var: begin_simtime <time_value>
```

Purpose:

Specify the beginning simuation time to start the query over time.

Parameters:

Type	Parameter	Description
floar	`time_value`	beginning simulation time value (that corresponds to a valid time step) in which to start the query over time

Notes:

This command must lie between either a "`query_ent_var: begin`" and "`query_ent_var: end`" pair, or a "`query_ent_var: modify_begin`" and "`query_ent_var: modify_end`" pair.

Only applicable with the "`query_ent_var: type time`" command.

Example Usage:

```
#
# Select part
#
part: select_begin
2
part: select_end
#
# Query over time
#
query_ent_var: begin
query_ent_var: description
query_ent_var: type time
query_ent_var: #_of_sample_pts 84
query_ent_var: begin_simtime 2.2300e+02
query_ent_var: end_simtime 4.8800e+02
query_ent_var: constrain node
query_ent_var: sample_by value
query_ent_var: node_id 1
query_ent_var: variable temp
query_ent_var: end
query_ent_var: query
#
# Select resulting plot curve
#
curve: select_begin
0
curve: select_end
```

See Also:

Associated query_ent_var commands

query_ent_var: constrain

Command:

```
query_ent_var: constrain <option>
```

Purpose:

Specify the manner in which to constrain the query of the variable.

Parameters:

Type	Parameter	Description	
constant	option	query type Over Time choices are:	
		cursor	to query the variable (or FFTs) at the point specified by the cursor tool. (This option is dependent on the "query_ent_var: cursor_loc" command.)
		element	to query the variable (or FFTs) at the element specified by the Element Id. (This option is dependent on the "query_ent_var: elem_id" command.)
		ijk	to query the variable values (or FFTs) at a specified IJK node (only applicable if the selected part is structured). (This option is dependent on the "query_ent_var: ijk" command.)
		max	to query for the maximum variable at each time step
		min	to query for the minimum variable at each time step
		node	to query the variable (or FFTs) at the node specfied by the Node Id. (This option is dependent on the "query_ent_var: node_id" command.)
		query type Distance choices are:	
		1d_part	query the varialbe values at the ends of the bar elements of a Part. (This option is dependent on the "query_ent_var: part_id" command.)
		line_tool	query the variable values at evenly spaced steps of the Line Tool

Notes:

This command must lie between either a "query_ent_var: begin" and "query_ent_var: end" pair, or a "query_ent_var: modify_begin" and "query_ent_var: modify_end" pair.

Example Usage:

```
#
# Modification change:
#
query_ent_var: modify_begin
query_ent_var: type distance
query_ent_var: #_of_sample_pts 20
query_ent_var: constrain line_tool
query_ent_var: modify_end
```

See Also:

Associated query_ent_var commands

query_ent_var: cursor_loc

Command:

```
query_ent_var: cursor_loc <x_coord> <y_coord> <z_coord>
```

Purpose:

Specify the x,y,z coordinates of the cursor location to be queried.

Parameters:

Type	Parameter	Description
float	x_coord	specify the x, y, z coordinate locations of the cursor in model space
	y_coord	
	z_coord	

Notes:

This command must lie between either a "query_ent_var: begin" and "query_ent_var: end" pair, or a "query_ent_var: modify_begin" and "query_ent_var: modify_end" pair.

Used in conjuction with the "query_ent_var: type time" and the "query_ent_var: constrain cursor" commands.

Example Usage:

```
#
# Specify part on which to perform query
#
part: select_begin
2
part: select_end
query_ent_var: begin
query_ent_var: description
query_ent_var: type time
query_ent_var: #_of_sample_pts 84
query_ent_var: begin_simtime
query_ent_var: end_simtime
query_ent_var: constrain cursor
query_ent_var: sample_by value
query_ent_var: cursor_loc 1.4870e+00 4.3663e-01 1.1934e+01
query_ent_var: variable temp
query_ent_var: end
query_ent_var: query
#
# Specify new curve to be current
#
curve: select_begin
1
curve: select_begin
```

See Also:

Associated query_ent_var commands

Command:

```
query_ent_var: description <desc>
```

Purpose:

Specify the text description for the query entity.

Parameters:

Type	Parameter	Description
string	desc	text description for the query entity

Notes:

This command must lie between a "query_ent_var: begin" and "query_ent_var: end" pair.

Example Usage:

```
part: select_begin
2
part: select_end
query_ent_var: begin
query_ent_var: description Temp Vs. Time
query_ent_var: type time
query_ent_var: #_of_sample_pts 10
query_ent_var: begin_simtime 2.2300e+02
query_ent_var: end_simtime 4.8800e+02
query_ent_var: constrain cursor
query_ent_var: sample_by value
query_ent_var: cursor_loc 1.4870e+00 4.3663e-01 1.1934e+01
query_ent_var: variable temp
query_ent_var: end
query_ent_var: query
curve: select_begin
2
curve: select_end
```

See Also:

Associated query_ent_var commands

query_ent_var: distance

Command:

```
query_ent_var: distance <option>
```

Purpose:

Specify how to parameterize the distance of the computed query (also, how the Distance is labeled on the plot).

Parameters:

Type	Parameter	Description	
constant	option	arc_length	(default) measures from the starting endpoint (or Origin) of the Line Tool (or 1D Part) to its final endpoint; all the RMS sums of the line segments in the range 0 to the arc length
		x_arc_length	measures from the starting endpoint (or Origin) of the Line Tool (or 1D Part) to its final endpoint; all the RMS sums of the x, y, or z components of the line segments in the range 0 to sum of the x, y, or z components of the arc length
		y_arc_length	
		z_arc_length	
		from_origin	measures from the starting endpoint (or Origin) of the Line Tool (or 1D Part) to its final endpoint; all the RMS sums of the line segments in the range determined by the position vector of the starting endpoint to the position vector of its final endpoint
		x_from_origin	measures from the starting endpoint (or Origin) of the Line Tool (or 1D Part) to its final endpoint; all the RMS sums of just the x, y, or z components of the line segments in the range determined by the position vector of the starting endpoint to the position vector of its final endpoint
		y_from_origin	
		z_from_origin	

Notes:

This command must lie between either a "query_ent_var: begin" and "query_ent_var: end" pair, or a "query_ent_var: modify_begin" and "query_ent_var: modify_end" pair.

Only applicable with the "query_ent_var: type distance" command.

Example Usage:

```
#
# Modify to parameterize using only the x-component of the arc length
#
query_ent_var: modify_begin
query_ent_var: type distance
query_ent_var: constrain line_tool
query_ent_var: distance x_arc_length
query_ent_var: modify_end
```

See Also:

Associated query_ent_var commands

Command:

```
query_ent_var: elem_id <element_id_#>
```

Purpose:

Specify the element id used to perform the specified element constrained query.

Parameters:

Type	Parameter	Description
int	element_id_#	valid element id number (0-bias

Notes:

This command must lie between either a "query_ent_var: begin" and "query_ent_var: end" pair, or a "query_ent_var: modify_begin" and "query_ent_var: modify_end" pair.

Only applicable with the "query_ent_var: type time" and "query_ent_var: constrain element" commands.

Only applicable on element-based variables.

Only applicable with the "query_ent_var: constrain element" and "query_ent_var: type time" commands.

Example Usage:

```
part: select_begin
4
part: select_end
query_ent_var: begin
query_ent_var: description
query_ent_var: type time
query_ent_var: #_of_sample_pts 10
query_ent_var: begin_simtime 2.2300e+02
query_ent_var: end_simtime 4.8800e+02
query_ent_var: constrain element
query_ent_var: sample_by value
query_ent_var: variable NodeToElem
query_ent_var: elem_id 14
query_ent_var: end
query_ent_var: query
curve: select_begin
0
curve: select_end
```

See Also:

Associated query_ent_var commands

query_ent_var: end_simtime

Command:

```
query_ent_var: end_simtime <time_value>
```

Purpose:

Specify the ending simulation time used to stop the query over time.

Parameters:

Type	Parameter	Description
float	time_value	ending simulation time value (that corresponds to a time step) at which to stop the computed query over time

Notes:

This command must lie between either a "query_ent_var: begin" and "query_ent_var: end" pair, or a "query_ent_var: modify_begin" and "query_ent_var: modify_end" pair.

Only applicable with the "query_ent_var: type time" command.

Example Usage:

```
#
# Select part
#
part: select_begin
2
part: select_end
#
# Query over time
#
query_ent_var: begin
query_ent_var: description
query_ent_var: type time
query_ent_var: #_of_sample_pts 84
query_ent_var: begin_simtime 2.2300e+02
query_ent_var: end_simtime 4.8800e+02
query_ent_var: constrain node
query_ent_var: sample_by value
query_ent_var: node_id 1
query_ent_var: variable temp
query_ent_var: end
query_ent_var: query
#
# Select resulting plot curve
#
curve: select_begin
0
curve: select_end
```

See Also:

Associated query_ent_var commands

Command:

```
query_ent_var: generate_over <type>
```

Purpose:

Specify whether the query is over time or distance.

Parameters:

Type	Parameter	Description	
constant	type	distance	query is over distance
		time	query is over time

Notes:

This command must lie between either a "query_ent_var: begin" and "query_ent_var: end" pair, or a "query_ent_var: modify_begin" and "query_ent_var: modify_end" pair.

Example Usage:

```
part: select_all
query_ent_var: begin
query_ent_var: description
query_ent_var: query_type generated
query_ent_var: #_of_sample_pts 20
query_ent_var: begin_simtime 0.0000e+00
query_ent_var: end_simtime 0.1900e+02
query_ent_var: constrain node
query_ent_var: sample_by value
query_ent_var: node_id 139
query_ent_var: variable_1 temperature
query_ent_var: generate_over time
query_ent_var: variable_2 TIME
query_ent_var: end
query_ent_var: query
```

See Also:

Associated query_ent_var commands

query_ent_var: ijk

Command:

```
query_ent_var: ijk <i-plane_#> <j-plane_#> <k-plane_#>
```

Purpose:

Specify the ijk location used to perform the specified ijk constrained query.

Parameters:

Type	Parameter	Description
int	i-plane_#	i, j, and k component values which must lie within the min/max of the i, j, and k range
	j-plane_#	
	k-plane_#	

Notes:

This command must lie between either a "query_ent_var: begin" and "query_ent_var: end" pair, or a "query_ent_var: modify_begin" and "query_ent_var: modify_end" pair.

Only applicable with the "query_ent_var: constrain ijk" and "query_ent_var: type time" commands.

Example Usage:

```
part: select_begin
4
part: select_end
query_ent_var: begin
query_ent_var: description
query_ent_var: type time
query_ent_var: #_of_sample_pts 10
query_ent_var: begin_simtime 2.2300e+02
query_ent_var: end_simtime 4.8800e+02
query_ent_var: constrain ijk
query_ent_var: sample_by value
query_ent_var: variable temp
query_ent_var: ijk 2 4 14
query_ent_var: end
query_ent_var: query
curve: select_begin
0
curve: select_end
```

See Also:

Associated query_ent_var commands

Command:

```
query_ent_var: line_loc <end_point_id> <x_comp> <y_comp> <z_comp>
```

Purpose:

Specify the location of the line used for distance queries when the constraint is set to `line_tool`.

Parameters:

Type	Parameter	Description
int	`end_point_id`	endpoints of the line, either 1 or 2
float	`x_comp`	x , y, or z component of the corresponding line endpoint in model coordinates
	`y_comp`	
	`z_comp`	

Notes:

This command must lie between either a "`query_ent_var: begin`" and "`query_ent_var: end`" pair, or a "`query_ent_var: modify_begin`" and "`query_ent_var: modify_end`" pair.

Example Usage:

```
part: select_begin
1
part: select_end
query_ent_var: begin
query_ent_var: description
query_ent_var: type distance
query_ent_var: #_of_sample_pts 20
query_ent_var: constrain line_tool
query_ent_var: line_loc 1 2.7770e-01 7.7363e-02 1.1348e-02
query_ent_var: line_loc 2 7.6144e-01 1.7007e-01 5.5697e-02
query_ent_var: variable Density
query_ent_var: end
query_ent_var: query
curve: select_begin
0
curve: select_end
```

See Also:

Associated query_ent_var commands

query_ent_var: marker_rgb

Command:

```
query_ent_var: marker_rgb <red_val> <grn_val> <blu_val>
```

Purpose:

Specify the color for the visible markers indicating query location.

Parameters:

Type	Parameter	Description
float	red_val	red, green, or blue component of color (0. to 1.)
	grn_val	
	blu_val	

Notes:

This command must lie between either a "query_ent_var: begin" and "query_ent_var: end" pair, or a "query_ent_var: modify_begin" and "query_ent_var: modify_end" pair.

Used only if query_ent_var: marker_visible is on.

Example Usage:

```
query_ent_var: modify_begin
query_ent_var: type distance
query_ent_var: constrain 1d_part
query_ent_var: marker_rgb 1.0000e+00 0.0000e+00 1.0000e+00
query_ent_var: modify_end
```

See Also:

Associated query_ent_var commands

Command:

```
query_ent_var: marker_size_normalized <size_value>
```

Purpose:

Specify the size of a query marker.

Parameters:

Type	Parameter	Description
float	size_value	size of the marker

Notes:

This command must lie between either a "query_ent_var: begin" and "query_ent_var: end" pair, or a "query_ent_var: modify_begin" and "query_ent_var: modify_end" pair.

Used only if query_ent_var:marker_visible is on.

A size of 1.0 indicates to use the computed size.

Values >1 scale the computed size larger while values <1 (but >0) shrink the marker.

Example Usage:

```
query_ent_var: modify_begin
query_ent_var: type distance
query_ent_var: constrain 1d_part
query_ent_var: marker_size_normalized 2.0000e+00
query_ent_var: modify_end
```

See Also:

Associated query_ent_var commands

query_ent_var: marker_visible

Command:

```
query_ent_var: marker_visible <toggle>
```

Purpose:

The begin/end construct in which to modify query entity attributes.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	(default for distance queries) toggle-on the marker
		`OFF`	(default for time queries) toggle-off the marker

Notes:

This command must lie between either a "`query_ent_var: begin`" and "`query_ent_var: end`" pair, or a "`query_ent_var: modify_begin`" and "`query_ent_var: modify_end`" pair.

If the query type is distance, then the marker is placed at the location of the origin (starting end-point).

Example Usage:

```
query_ent_var: modify_begin
query_ent_var: type time
query_ent_var: constrain element
query_ent_var: marker_type OFF
query_ent_var: modify_end
```

See Also:

Associated query_ent_var commands

Command:

```
query_ent_var: modify_begin
query_ent_var: modify_end
```

Purpose:

Delimit the modifications for query entity attributes.

Notes:

These commands are used to set the attributes for a query entity.

Parameters:

None.

Example Usage:

```
#
# Modification change:
#
query_ent_var: modify_begin
query_ent_var: type distance
query_ent_var: #_of_sample_pts 20
query_ent_var: constrain line_tool
query_ent_var: modify_end
```

See Also:

Associated query_ent_var commands

query_ent_var: multiple_segments_by

Command:

```
query_ent_var: multiple_segments_by <option>
```

Purpose:

Treat multiple line segments of a distance query as multiple queries, (i.e., reset the distance to 0. for each segment) or as a continuous line.

Parameters:

Type	Parameter	Description	
constant	option	accumulation	treat multiple line segments as a continuous line
		reset_each	treat multiple line segments as separate queries

Notes:

This command must lie between either a "query_ent_var: begin" and "query_ent_var: end" pair, or a "query_ent_var: modify_begin" and "query_ent_var: modify_end" pair.

Only applicable with the "query_ent_var: type distance" command.

Example Usage:

```
query_ent_var: modify_begin
query_ent_var: type distance
query_ent_var: constrain 1d_part
query_ent_var: multiple_segments_by reset_each
query_ent_var: modify_end
```

See Also:

Associated query_ent_var commands

Command:

```
query_ent_var: node_id <node_id_#>
```

Purpose:

Specify the node id in which to perform the specified node constrained query.

Parameters:

Type	Parameter	Description
int	node_id_#	node id, or label, number

Notes:

This command must lie between either a "query_ent_var: begin" and "query_ent_var: end" pair, or a "query_ent_var: modify_begin" and "query_ent_var: modify_end" pair.

Only applicable with the "query_ent_var: constrain node" command. Only applicable on node-based variables.

Example Usage:

```
#
# Select part
#
part: select_begin
2
part: select_end
#
# Query over time
#
query_ent_var: begin
query_ent_var: description
query_ent_var: type time
query_ent_var: #_of_sample_pts 84
query_ent_var: begin_simtime 2.2300e+02
query_ent_var: end_simtime 4.8800e+02
query_ent_var: constrain node
query_ent_var: sample_by value
query_ent_var: node_id 1
query_ent_var: variable temp
query_ent_var: end
query_ent_var: query
```

See Also:

Associated query_ent_var commands

query_ent_var: operation_factor1/_factor2

Command:

```
query_ent_var: operation_factor1 <scale>
query_ent_var: operation_factor2 <scale>
```

Purpose:

Specify the scale factor to apply to the one or two existing queries of an operation query.

Parameters:

Type	Parameter	Description
float	`scale`	value to scale the respective query by

Notes:

This command must lie between either a "`query_ent_var: begin`" and "`query_ent_var: end`" pair, or a "`query_ent_var: modify_begin`" and "`query_ent_var: modify_end`" pair.

Example Usage:

```
query_ent_var: begin
query_ent_var: description
query_ent_var: query_type operation
query_ent_var: operation_factor1 2.0000e+00
query_ent_var: operation_factor2 -1.5000e+01
query_ent_var: operation_query1byname Distance vs. temp for line tool
query_ent_var: operation_query2byname Time vs. temp for Node 1
query_ent_var: end
query_ent_var: query
```

See Also:

Associated query_ent_var commands

Command:

```
query_ent_var: operation_query1byname <name>
query_ent_var: operation_query2byname <name>
```

Purpose:

Specify the name of the one or two existing queries of an operation query.

Parameters:

Type	Parameter	Description
string	`name`	existing query name

Notes:

This command must lie between either a "`query_ent_var: begin`" and "`query_ent_var: end`" pair, or a "`query_ent_var: modify_begin`" and "`query_ent_var: modify_end`" pair.

Example Usage:

```
query_ent_var: begin
query_ent_var: description
query_ent_var: query_type operation
query_ent_var: operation_factor1 2.0000e+00
query_ent_var: operation_factor2 -1.5000e+01
query_ent_var: operation_query1byname Distance vs. temp for line tool
query_ent_var: operation_query2byname Time vs. temp for Node 1
query_ent_var: end
query_ent_var: query
```

See Also:

Associated query_ent_var commands

query_ent_var: origin

Command:

```
query_ent_var: origin <x_coord> <y_coord> <z_coord>
```

Purpose:

Specify a location which will be used to select the origin of a distance plot when performing the query on a 1D part.

Parameters:

Type	Parameter	Description
float	x_coord	x, y, or z coordinate value in model coordinates
	y_coord	
	z_coord	

Notes:

This command must lie between either a "query_ent_var: begin" and "query_ent_var: end" pair, or a "query_ent_var: modify_begin" and "query_ent_var: modify_end" pair.

Only applicable with the "querey_ent_var: type distance" and "query_ent_var: constrain 1d_part" commands.

The end point closest to the origin specified will be used as the “origin” of the query, i.e., where distance = 0.

If the ID part is a closed loop (i.e., there are no end points), the closest point on the loop is used as the “origin”.

Example Usage:

```
query_ent_var: modify_begin
query_ent_var: origin 2.3596e+00 1.8494e+00 1.1773e+01
query_ent_var: modify_end
```

See Also:

Associated query_ent_var commands

Command:

```
query_ent_var: part_id <part_id_#>
```

Purpose:

Specify the 1D part number to query when performing a query over distance with a part constraint.

Parameters:

Type	Parameter	Description
int	part_id_#	GUI number of the part

Notes:

This command must lie between either a "query_ent_var: begin" and "query_ent_var: end" pair, or a "query_ent_var: modify_begin" and "query_ent_var: modify_end" pair.

Example Usage:

```
part: select_begin
4
part: select_end
query_ent_var: begin
query_ent_var: description
query_ent_var: type distance
query_ent_var: part_id 4
query_ent_var: constrain 1d_part
query_ent_var: distance arc_length
query_ent_var: variable temp
query_ent_var: end
query_ent_var: query
curve: select_begin
0
curve: select_end
```

See Also:

Associated query_ent_var commands

query_ent_var: part_idbyname

Command:

```
query_ent_var: part_idbyname <part_name>
```

Purpose:

Specify the 1D part name to query when performing a query over distance with a part constraint.

Parameters:

Type	Parameter	Description
string	part_name	GUI name of the part

Notes:

This command must lie between either a "query_ent_var: begin" and "query_ent_var: end" pair, or a "query_ent_var: modify_begin" and "query_ent_var: modify_end" pair.

Example Usage:

```
part: select_begin
Flow Field
part: select_end
query_ent_var: begin
query_ent_var: description
query_ent_var: type distance
query_ent_var: part_idbyname Line Clip
query_ent_var: constrain 1d_part
query_ent_var: distance arc_length
query_ent_var: variable temp
query_ent_var: end
query_ent_var: query
curve: select_begin
0
curve: select_end
```

See Also:

Associated query_ent_var commands

Command:

```
query_ent_var: query
```

Purpose:

Specify the creation of a query entity.

Parameters:

None.

Notes:

The query is created using the current query attributes.

Example Usage:

```
part: select_begin
1
part: select_end
query_ent_var: begin
query_ent_var: description
query_ent_var: type distance
query_ent_var: #_of_sample_pts 20
query_ent_var: constrain line_tool
query_ent_var: line_loc 1 2.7770e-01 7.7363e-02 1.1348e-02
query_ent_var: line_loc 2 7.6144e-01 1.7007e-01 5.5697e-02
query_ent_var: variable Density
query_ent_var: end
query_ent_var: query
curve: select_begin
0
curve: select_end
```

See Also:

Associated query_ent_var commands

query_ent_var: query_type

Command:

```
query_ent_var: query_type <type>
```

Purpose:

Specify the type of query to be created.

Parameters:

Type	Parameter	Description	
constant	type	generated	all distance and time queries
		operation	scaling and algebraic sum operation on existing queries
		external	loading externally created or previously saved queries

Notes:

This command must lie between either a "query_ent_var: begin" and "query_ent_var: end" pair, or a "query_ent_var: modify_begin" and "query_ent_var: modify_end" pair.

Example Usage:

```
part: select_all
#
# At line tool over distance query
#
query_ent_var: begin
query_ent_var: description
query_ent_var: query_type generated
query_ent_var: #_of_sample_pts 20
query_ent_var: constrain line_tool
query_ent_var: constrain line_tool
query_ent_var: line_loc 1 -7.5000e-01 1.1561e+00 1.0000e+00
query_ent_var: line_loc 2 1.7500e+00 1.1561e+00 1.0000e+00
query_ent_var: variable_1 temperature
query_ent_var: generate_over distance
query_ent_var: variable_2 DISTANCE
query_ent_var: end
query_ent_var: query
```

See Also:

Associated query_ent_var commands

Command:

```
query_ent_var: sample_by <option>
```

Purpose:

Specify how to report values queried over time.

Parameters:

Type	Parameter	Description	
constant	option	value	(default) reports "Values versus Time"
		fft	"FFT spectral Values vs. (postitive) Frequencies"

Notes:

This command must lie between either a "query_ent_var: begin" and "query_ent_var: end" pair, or a "query_ent_var: modify_begin" and "query_ent_var: modify_end" pair.

Only applicable with the "query_ent_var: type time" command.

Example Usage:

```
#
# Select part
#
part: select_begin
2
part: select_end
#
# Query over time
#
query_ent_var: begin
query_ent_var: description
query_ent_var: type time
query_ent_var: #_of_sample_pts 84
query_ent_var: begin_simtime 2.2300e+02
query_ent_var: end_simtime 4.8800e+02
query_ent_var: constrain node
query_ent_var: sample_by value
query_ent_var: node_id 1
query_ent_var: variable temp
query_ent_var: end
query_ent_var: query
#
# Select resulting plot curve
#
curve: select_begin
0
curve: select_end
#
# Modified to sample by FFT
#
query_ent_var: modify_begin
query_ent_var: type time
query_ent_var: constrain node
query_ent_var: sample_by fft
query_ent_var: modify_end
```

See Also:

Associated query_ent_var commands

Command:

```
query_ent_var: select_all_timesteps
```

Purpose:

Causes all current timesteps to be selected for querying.

Parameters:

None.

Notes:

This command is not generated by EnSight, but can be placed in a command file to generalize it so you don't have to know how many time steps are present. It can replace the `query_ent_var: begin_simtime` and `query_ent_var: end_simtime` commands.

Example Usage:

```
part: select_begin
1
part: select_end
query_ent_var: begin
query_ent_var: description
query_ent_var: query_type generated
query_ent_var: #_of_sample_pts 20
query_ent_var: select_all_timesteps
query_ent_var: constrain node
query_ent_var: sample_by value
query_ent_var: node_id 100
query_ent_var: variable_1 Stress
query_ent_var: generate_over time
query_ent_var: variable_2 TIME
query_ent_var: end
query_ent_var: query
curve: select_begin
0
curve: select_end
```

See Also:

Associated query_ent_var commands

query_ent_var: type

Command:

```
query_ent_var: type <domain>
```

Purpose:

Specify the domain over which to perform the query.

Parameters:

Type	Parameter	Description	
constant	`domain`	`distance`	to query along a line using the Line Tool or 1D Part at the current time step
		`time`	to query variable values or FFT's over a specified range of time steps. (This outputs for the selected variable a value versus time or FFT frequency versus FFT spectral values

Notes:

This command must lie between either a "`query_ent_var: begin`" and "`query_ent_var: end`" pair, or a "`query_ent_var: modify_begin`" and "`query_ent_var: modify_end`" pair.

Example Usage:

```
#
# Modification example of a distance query
#
query_ent_var: modify_begin
query_ent_var: type distance
query_ent_var: #_of_sample_pts 20
query_ent_var: constrain line_tool
query_ent_var: modify_end
```

See Also:

Associated query_ent_var commands

Command:

```
query_ent_var: variable_1 <name>
query_ent_var: variable_2 <name>
```

Purpose:

Specify the variables to be queried. The first variable can be any active variable. The second variable can be distance, time, or a second active variable.

Parameters:

Type	Parameter	Description
string or constant	`name`	name of the active variable. For variable_1 it can be any active variable. For variable_2, it can be `DISTANCE` if a non-scatter query over distance is being created, `TIME` if a non-scatter query over time is being created, or any active variable if a scatter query is being created. If plotted, variable_1 will be the Y-axis variable and varibe_2 will be the X-axis variable.

Notes:

This command must lie between either a "`query_ent_var: begin`" and "`query_ent_var: end`" pair, or a "`query_ent_var: modify_begin`" and "`query_ent_var: modify_end`" pair.

Example Usage:

```
part: select_all
#
# At line tool over distance query
#
query_ent_var: begin
query_ent_var: description
query_ent_var: query_type generated
query_ent_var: #_of_sample_pts 20
query_ent_var: constrain line_tool
query_ent_var: constrain line_tool
query_ent_var: line_loc 1 -7.5000e-01 1.1561e+00 1.0000e+00
query_ent_var: line_loc 2 1.7500e+00 1.1561e+00 1.0000e+00
query_ent_var: variable_1 temperature
query_ent_var: generate_over distance
query_ent_var: variable_2 DISTANCE
query_ent_var: end
query_ent_var: query
```

See Also:

Associated query_ent_var commands

Commands in this Section:

query_interact

query_interact: #_displayed <num>
query_interact: create <parameters>
query_interact: display_id <toggle>
query_interact: label_always_on_top <toggle>
query_interact: label_rgb <red_val> <grn_val> <blu_val>
query_interact: marker_size_normalized <scale>
query_interact: marker_visible <toggle>
query_interact: query <query_type>
query_interact: request_method <method>
query_interact: save <filename>
query_interact: search <search_mode>
query_interact: select_default
query_interact: select_varname_begin
query_interact: select_varname_end

See Also:

User Manual - Section 7.12, Interactive Probe Query
How To Probe Interactively

query_interact: #_displayed

Command:

```
query_interact: #_displayed  <num>
```

Purpose:

Specifies how many query markers and report lines will be displayed.

Parameters:

Type	Parameter	Description
int	num	number of queries

Notes:

This defines a stack size. Once full, the oldest query is popped off the stack.

Example Usage:

```
variable: activate pressure
query_interact: select_varname_begin
pressure
query_interact:  select_varname_end
query_interact: query xyz
query_interact: #_displayed 3
query_interact: create 12.0 15.1 10.3
query_interact: create 11.0 15.0 10.1
query_interact: create 10.0 15.0 10.3
```

See Also:

Associated query_interact commands

Command:

```
query_interact: create <parameters>
```

Purpose:

Creates a query using the current settings.

Parameters:

Surface Query		
Type	**Parameter**	**Description**
float	screen_x	xy normalized screen coordinates (0.0 to 1.0) query occurs on closest surface
	screen_y	

Cursor Query		
Type	**Parameter**	**Description**
float	x_coord	xyz model coordinates query occurs on point specified
	y_coord	
	z_coord	

Node Query		
Type	**Parameter**	**Description**
int	node_id	node id for query

IJK Query		
Type	**Parameter**	**Description**
int	i_coord	ijk coordinates for query
	j_coord	
	k_coord	

Element Query		
Type	**Parameter**	**Description**
int	element_id	element id for query

XYZ Query		
Type	**Parameter**	**Description**
float	x_coord	xyz model coordinates query occurs on point specified
	y_coord	
	z_coord	

Notes:

The parameters vary based on the type of query being used. (See `query_interact: query`)

Query is performed at location specified or at closest node if `query_interact: search closest_node` is specified.

Example Usage:

```
variable: activate pressure
query_interact: select_varname_begin
pressure
query_interact:  select_varname_end
query_interact: query xyz
query_interact: create 12.0 15.1 10.3
```

See Also:

Associated query_interact commands

Command:

```
query_interact: display_id <toggle>
```

Purpose:

Causes the node located for the query to be reported.

Parameters:

Type	Parameter	Description
constant	`toggle`	`ON`
		`OFF`

Notes:

This command is used with `query_interact: search closest_node` to get reporting of the actual node that was used for the query.

Example Usage:

```
variable: activate pressure
query_interact: select_varname_begin
pressure
query_interact:  select_varname_end
query_interact: query cursor
query_interact: search closest_node
query_interact: display_id ON
query_interact: create 12.0 15.1 10.3
```

See Also:

Associated query_interact commands

query_interact: label_always_on_top

Command:

```
query_interact: label_always_on_top <toggle>
```

Purpose:

Causes the label written beside the query marker in the graphics window to be rendered on top of any surface that would normally hide or obscure it.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	do not obscure by geometry
		`OFF`	draw only if label is not obscured by geometry

Notes:

This attribute is `ON` by default.

Example Usage:

```
variable: activate pressure
query_interact: select_varname_begin
pressure
query_interact:  select_varname_end
query_interact: query cursor
query_interact: label_always_on_top ON
query_interact: create 12.0 15.1 10.3
```

See Also:

Associated query_interact commands

Command:

```
query_interact: label_rgb  <red_val> <grn_val> <blu_val>
```

Purpose:

Specifies the RGB color value of the label written next to the query marker in the graphics window.

Parameters:

Type	Parameter	Description
float	red_val	red, green, blue contributions of the color within range of 0.0 to 1.0
	grn_val	
	blu_val	

Example Usage:

```
variable: activate pressure
query_interact: select_varname_begin
pressure
query_interact:  select_varname_end
query_interact: query cursor
query_interact: label_rgb 0 1 1
query_interact: create 12.0 15.1 10.3
```

See Also:

Associated query_interact commands

query_interact: marker_size_normalized

Command:

```
query_interact: marker_size_normalized <scale>
```

Purpose:

Specify the size of a query marker.

Parameters:

Type	Parameter	Description
float	`scale`	size of marker

Notes:

The default marker scale value is 1.0.

A size of 1.0 indicates to use the computed size

Values >1 scale the computed size larger while values <1 (but >0) shrink the marker.

Example Usage:

```
variable: activate pressure
query_interact: variable pressure
query_interact: query cursor
query_interact: label_visible OFF
query_interact: marker_size_normalized  0.5
query_interact: create 12.0 15.1 10.3
```

See Also:

Associated query_interact commands

Command:

```
query_interact: marker_visible <toggle>
```

Purpose:

Sets the visibility of the query marker in the graphics window.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	make markers visible
		OFF	do not display markers

Notes:

Marker visibility is ON by default.

Example Usage:

```
variable: activate pressure
query_interact: select_varname_begin
pressure
query_interact:  select_varname_end
query_interact: query cursor
query_interact: label_visible OFF
query_interact: marker_visible OFF
query_interact: create 12.0 15.1 10.3
```

See Also:

Associated query_interact commands

Command:

```
query_interact: query <query_type>
```

Purpose:

Sets the type of query to be performed.

Parameters:

Type	Parameter	Description	
constant	query_type	may be any of the following values:	
		surface	causes the query to occur at a point defined on the first surface encountered by a projection of a given screen x/y coordinate into the z dimension. The x/y coordinates will be specified as arguments to the query_interact: create command
		cursor	causes the query to occur at the cursor location specified as an argument to the query_interact: create command
		node	causes the query to occur at a specified node. The node id will be specified as an argument to the query_interact: create command
		ijk	causes the query to occur at a specific location in an ijk space. The ijk coordinates will be specified as arguments to the query_interact: create command
		element	element – causes the query to occur at a specified element. The element id will be specified as an argument to the query_interact: create command
		xyz	causes the query to occur at a specific location in an xyz space. The xyz coordinates will be specified as arguments to the query_interact: create command
		none	turns query off

Notes:

The exact location of the query, as specified above, may be modified by the query_interact: search command; i.e. if an exact set of coordinates or a cursor position is specified, the actual query may be done at the closest node to this point if query_interact: search closest_node has been specified.

Example Usage:

```
variable: activate pressure
query_interact: select_varname_begin
pressure
query_interact:  select_varname_end
query_interact: query cursor
query_interact: searchclosest_node
query_interact: create 12.0 15.1 10.3
```

See Also:

Associated query_interact commands

query_interact: request_method

Command:

```
query_interact: request_method  <method>
```

Purpose:

Sets the request method for surface pick or cursor-based queries to be on a request or continuous basis.

Parameters:

Type	Parameter	Description	
constant	method	may be any of the following values:	
		continuous	automatically update as mouse/cursor moves
		pick	create the query only when requested via a pick action

Notes:

In continuous mode, the query will be updated continuously while the mouse or cursor is being moved.

Example Usage:

```
variable: activate pressure
query_interact: select_varname_begin
pressure
query_interact:  select_varname_end
query_interact: query cursor
query_interact: request_method pick
query_interact: create 12.0 15.1 10.3
```

See Also:

Associated query_interact commands

Command:

```
query_interact: save <filename>
```

Purpose:

Specifies the filename into which the Interactive Probe Query Results Table will be saved.

Parameters:

Type	Parameter	Description
string	`filename`	filename to receive the results table information

Example Usage:

```
variable: activate pressure
query_interact: select_varname_begin
pressure
query_interact:  select_varname_end
query_interact: query xyz
query_interact: #_displayed 3
query_interact: create 12.0 15.1 10.3
query_interact: create 11.0 15.0 10.1
query_interact: create 10.0 15.0 10.3
query_interact: save /home/users/me/probe.txt
```

See Also:

Associated query_interact commands

query_interact: search

Command:

```
query_interact: search <search_mode>
```

Purpose:

Sets the search mode for surface pick or cursor queries.

Parameters:

Type	Parameter	Description	
constant	search_mode	may be any of the following values:	
		exact	query to be done at the exact point specified
		closest_node	query to occur at the node closest to the specified point

Example Usage:

```
variable: activate pressure
query_interact: select_varname_begin
pressure
query_interact:  select_varname_end
query_interact: query cursor
query_interact: search closest_node
query_interact: create 12.0 15.1 10.3
```

See Also:

Associated query_interact commands

Command:

```
query_interact: select_default
```

Purpose:

Causes any following query_interact: commands to affect the default query settings.

Parameters:

none

Notes:

All `query_interact:` commands following `query_interact: select_default` will affect default settings until a new variable is selected to query using `query_interact: select_varname_begin`.

Example Usage:

```
query_interact: select_default
query_interact: marker_visible OFF
query_interact: marker_rgb 1 0 0
query_interact: select_varname_begin
pressure
query_interact: select_varname_end
```

See Also:

Associated query_interact commands

query_interact: select_varname_begin/end

Command:

```
query_interact: select_varname_begin
<varname1  varname2 …varnameN>
query_interact:  select_varname_end
```

Purpose:

Specifies the variables to be queried.

Parameters:

Type	Parameter	Description
string	`varnameN`	names of variables to be queried

Example Usage:

```
variable: activate pressure
variable: activate temperature
query_interact: select_varname_begin
pressure
temperature
query_interact:  select_varname_end
query_interact: query cursor
query_interact: create 12.0 15.1 10.3
```

See Also:

Associated query_interact commands

Commands in this Section:

savegeom

savegeom: begin_step <step>
savegeom: binary <toggle>
savegeom: brick_format <option>
savegeom: end_step <step>
savegeom: format <option>
savegeom: maximum_file_size <size>
savegeom: parameters <input_string>
savegeom: resolution <x> <y> <z>
savegeom: save_geometric_entities <filename_prefix>
savegeom: select_all_steps
savegeom: single_file <toggle>
savegeom: step_by <increment>
savegeom: undefined <undef_value>
savegeom: variable <var_name>

See Also:

User Manual - Section 6.1, File Menu Functions
How To Save Geometric Entities

savegeom: brick_format

Command:

```
savegeom: brick_format <option>
```

Purpose:

Specify the format of the values to write out when using brick format.

Parameters:

Type	Parameter	Description	
constant	option	bytes	for brick of bytes
		floats	for brick of floats

Notes:

Used in conjunction with `savegeom: format brick`

Example Usage:

```
part: select_all
savegeom: format brick
savegeom: brick_format floats
savegeom: resolution 128 128 128
savegeom: undefined -1.2345e-10
savegeom: variable pressure
savegeom: save_geometric_entities /tmp/file_prefix
```

See Also:

Associated savegeom commands

Command:

```
savegeom: begin_step <step>
savegeom: end_step <step>
```

Purpose:

Specify the beginning/ending time step for exporting the selected part(s) and active variable(s).

Parameters:

Type	Parameter	Description
int	`step`	the beginning/ending time step in the range I[0,n], where n is the last time step

Example Usage:

```
part: select_all
savegeom: format case
savegeom: binary ON
savegeom: single_file OFF
savegeom: begin_step 0
savegeom: end_step 16
savegeom: step_by 1
savegeom: save_geometric_entities /tmp/file_prefix
```

See Also:

Associated savegeom commands

savegeom: binary

Command:

```
savegeom: binary <toggle>
```

Purpose:

Specify either binary or ASCII as the format for the exported files.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	Binary format
		`OFF`	ASCII format

Example Usage:

```
part: select_all
savegeom: format case
savegeom: binary ON
savegeom: single_file OFF
savegeom: select_all_steps
savegeom: save_geometric_entities /tmp/file_prefix
```

See Also:

Associated savegeom commands

Command:

```
savegeom: format <option>
```

Purpose:

Specify the export data file format for the selected part(s) and active variable(s).

Parameters:

Type	Parameter	Description	
constant	option	Include one of the following:	
		brick	EnSight volume rendering format
		case	EnSight Gold case file format
		vrml	VRML (version 2.0) file format
		UDWName	Where UDWName can be any one of the user-defined writer names that appear in the Save Geometric Entities dialog.

Notes:

The option is the format of an included internal format usch as case or vrml, or a user-defined format that can be site specfic.

Example Usage:

```
part: select_all
savegeom: format case
savegeom: binary ON
savegeom: single_file OFF
savegeom: select_all_steps
savegeom: save_geometric_entities /tmp/file_prefix
```

See Also:

Associated savegeom commands

savegeom: maximum_file_size

Command:

```
savegeom: maximum_file_size <size>
```

Purpose:

Specify the maximum file size (in MBs) for exporting time step data into single file format.

Parameters:

Type	Parameter	Description
float	`size`	file size limit in MB (Mega-Bytes)

Notes:

This default value is the maximum file size limit of either 2 Giga-Bytes for 32-bit architecture, or 4 Tera-Bytes for 64-bit architecture machines.

Example Usage:

```
part: select_all
savegeom: format case
savegeom: binary ON
savegeom: single_file ON
savegeom: maximum_file_size 2147.479980
savegeom: select_all_steps
savegeom: save_geometric_entities /tmp/file_prefix
```

See Also:

Associated savegeom commands

Command:

```
savegeom: parameters <input_string>
```

Purpose:

Pass into the User-defined writer a string that can be used to modify the behavior, or supply additional parameters.

Parameters:

Type	Parameter	Description
string	`input_string`	up to 200 characters of information that will be passed into the user-defined writer selected.

Notes:

This can be used to provide parameters to a user-defined writer.

Example Usage:

```
part: select_all
savegeom: format STL
savegeom: parameters test string
savegeom: save_geometric_entities /tmp/file_prefix
```

See Also:

Associated savegeom commands

savegeom: resolution

Command:

```
savegeom: resolution <x> <y> <z>
```

Purpose:

Specify the Brick of Values resolution in the x, y, and z directions.

Parameters:

Type	Parameter	Description
int	x	x, y, z direction resolution
	y	
	z	

Notes:

Used in conjunction with `savegeom: format brick`

Example Usage:

```
part: select_all
savegeom: format brick
savegeom: brick_format floats
savegeom: resolution 128 128 128
savegeom: undefined -1.2345e-10
savegeom: variable pressure
savegeom: save_geometric_entities /tmp/file_prefix
```

See Also:

Associated savegeom commands

Command:

```
savegeom: save_geometric_entities <filename_prefix>
```

Purpose:

Specify the file name path and prefix for the selected part and active variable data to be exported.

Parameters:

Type	Parameter	Description
string	`filename_prefix`	directory path and filename prefix to be used for the exported data file

Notes:

The exported files will be saved in the current working directory of the server if no path is provided.

If the format (as defined by `savegeom: format`) is VRML, the filename given will be used directly. If the format is CASE, the multiple files (case, geometry, and variables) are written using the supplied name as the prefix.

Example Usage:

```
part: select_all
savegeom: format case
savegeom: binary ON
savegeom: single_file OFF
savegeom: select_all_steps
savegeom: save_geometric_entities /tmp/file_prefix
or:
part: select_begin
 1
part: select_end
savegeom: format case
savegeom: binary ON
savegeom: single_file OFF
savegeom: select_all_steps
savegeom: save_geometric_entities /tmp/file_prefix
```

See Also:

Associated savegeom commands

savegeom: select_all_steps

Command:

```
savegeom: select_all_steps
```

Purpose:

Specify all time step data of selected parts (geometry) and active variables to be exported to an EnSight case or VRML file(s).

Parameters:

none

Example Usage:

```
part: select_all
savegeom: format case
savegeom: binary ON
savegeom: single_file OFF
savegeom: select_all_steps
savegeom: save_geometric_entities /tmp/file_prefix
```

See Also:

Associated savegeom commands

Command:

```
savegeom: single_file <toggle>
```

Purpose:

Specify whether the selected part(s), or geometry, and each active variable will be exported to a single or to separate time step files.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	combine all geometry time step data into a single file; also for each variable, combine all respective variable timestep data into a single file.
		OFF	export data for each geometry time step and each variable time step to a separate file

Example Usage:

```
part: select_all
savegeom: format case
savegeom: binary ON
savegeom: single_file ON
savegeom: maximum_file_size 2147.479980
savegeom: select_all_steps
savegeom: save_geometric_entities /tmp/file_prefix
```

See Also:

Associated savegeom commands

savegeom: step_by

Command:

```
savegeom: step_by <increment>
```

Purpose:

Specify the increment step by which to stride the time steps in exporting the selected part(s) and active variable(s).

Parameters:

Type	Parameter	Description
int	`increment`	increment to stride the time steps in the range of the beginning and ending time steps

Example Usage:

```
part: select_all
savegeom: format case
savegeom: binary ON
savegeom: single_file OFF
savegeom: begin_step 0
savegeom: end_step 16
savegeom: step_by 1
savegeom: save_geometric_entities /tmp/file_prefix
```

See Also:

Associated savegeom commands

Command:

```
savegeom: undefined <undef_value>
```

Purpose:

Specify the value to write out as undefined when using brick format.

Parameters:

Type	Parameter	Description
float	undef_value	any undefined value is written out as this value

Notes:

Used in conjunction with `savegeom: format brick`

Example Usage:

```
part: select_all
savegeom: format brick
savegeom: brick_format floats
savegeom: resolution 128 128 128
savegeom: undefined -1.2345e-10
savegeom: variable pressure
savegeom: save_geometric_entities /tmp/file_prefix
```

See Also:

Associated savegeom commands

savegeom: variable

Command:

```
savegeom: variable <var_name>
```

Purpose:

Specify the name of the variable to save when using brick format.

Parameters:

Type	Parameter	Description
string	`var_name`	variable name

Notes:

Used in conjunction with `savegeom: format brick`

Example Usage:

```
part: select_all
savegeom: format brick
savegeom: brick_format floats
savegeom: resolution 128 128 128
savegeom: undefined -1.2345e-10
savegeom: variable pressure
savegeom: save_geometric_entities /tmp/file_prefix
```

See Also:

Associated savegeom commands

Commands in this Section:

sepattach

sepattach: begin
sepattach: end
sepattach: create
sepattach: density <variable_name>
sepattach: display_offset <offset_value>
sepattach: method <option>
sepattach: momentum <variable_name>
sepattach: ratio_of_spec_heat <name or value>
sepattach: select_default
sepattach: threshold <variable_name>
sepattach: threshold_filter <option>
sepattach: threshold_max <threshold_value>
sepattach: threshold_min <threshold_value>
sepattach: threshold_value <threshold_value>
sepattach: velocity <variable_name>

See Also:

User Manual - Section 7.21, Separation/Attachment Lines Create/Update
How To Extract Separation and Attachment Lines

sepattach: begin/end

Command:

```
sepattach: begin
sepattach: end
```

Purpose:

Delimit the modifications for separation or attachment line attributes.

Parameters:

None.

Notes:

These commands are used to set the attributes for a separation or attachment line part creation.

Example Usage:

```
#
# Specify the 2D parent part(s) from which
# to create the separation and attachment lines
#
part: select_begin
 2 3
part: select_end
#
# Compute the separation and attachment lines parts
#
sepattach: begin
sepattach: end
sepattach: create
#
# Now select the newly created separation and
# attachment line parts for subsequent operations
#
part: select_begin
 6 7
part: select_end
```

See Also:

Associated sepattach commands

Command:

```
sepattach: create
```

Purpose:

Create separation and attachment line parts using the currently selected 2D parts and defined attributes.

Parameters:

None.

Notes:

This feature is different from other features in that it creates two parts, i.e. a separation line part, and an attachment line part. In addition, both of these parts are modified with any command change applied to either one or both of these parts.

Example Usage:

```
#
# Specify the 2D parent part(s) from which
# to create the separation and attachment lines
#
part: select_begin
 2 3
part: select_end
#
# Compute the separation and attachment lines parts
#
sepattach: begin
sepattach: end
sepattach: create
#
# Now select the newly created separation and
# attachment line parts for subsequent operations
#
part: select_begin
 6 7
part: select_end
```

See Also:

Associated sepattach commands

sepattach: density

Command:

```
sepattach: density <variable_name>
```

Purpose:

Specify the density variable to use in creating/updating the default separation and attachment line part attributes and/or the selected separation and attachment line part(s).

Parameters:

Type	Parameter	Description
string	`variable_name`	name of the density variable (the default `<none>` means to use nothing)

Notes:

This variable is assigned automatically if the variable name description `<Density>` exists. Since the computation of separation and attachment lines depend on velocity, either the velocity variable must be defined or both the density and momentum variables must be defined to obtain the velocity variable.

Example Usage:

```
variables: activate Density
sepattach: select_default
part: modify_begin
sepattach: density Density
part: modify_end
```

See Also:

Associated sepattach commands

Command:

```
sepattach: display_offset <offset_value>
```

Purpose:

Specify the offset from the 2D parent part(s) surface from which to display (or lift) the separation and attachment line parts.

Parameters:

Type	Parameter	Description
float	`offset_value`	numerical offset value from the 2D surface, (default = 0.)

Notes:

This offset measures in model coordinate units, in the normal direction from the 2D parent part(s) surface(s); and thus, may need to be specified as either a + or - number.

Example Usage:

```
part: modify_begin
sepattach: display_offset -1.0000e-03
part: modify_end
```

See Also:

Associated sepattach commands

sepattach: method

Command:

```
sepattach: method <option>
```

Purpose:

Specify the method, or algorithm, in which to compute the separation and attachment line parts, from either 2D parent part(s) or in recomputing existing 1D separation and attachment line parts.

Parameters:

Type	Parameter	Description	
constant	`option`	`phase_plane`	an algorithm that computes the separation and attachment lines based on eigen-analysis done in phase plane coordinates

Notes:

Currently only one algorithm offered.

Example Usage:

```
part: select_begin
 6 7
part: select_end
part: modify_begin
sepattach: method phase_plane
part: modify_end
```

See Also:

Associated sepattach commands

Command:

```
sepattach: momentum <variable_name>
```

Purpose:

Specify the momentum variable to use in creating/updating the default separation and attachment line part attributes and/or the selected separation and attachment line part(s).

Parameters:

Type	Parameter	Description
string	`variable_name`	name of the momentum variable (the default `<none>` means to use nothing)

Notes:

This variable is assigned automatically if the variable name description `<Momentum>` exists. Since the computation of separation and attachment lines depend on velocity, either the velocity variable must be defined or both the density and momentum variables must be defined to obtain the velocity variable.

Example Usage:

```
variables: activate Momentum
sepattach: select_default
part: modify_begin
sepattach: momentum Momentum
part: modify_end
```

See Also:

Associated sepattach commands

sepattach: ratio_of_spec_heat

Command:

```
sepattach: ratio_of_spec_heat <name or value>
```

Purpose:

Specify the ratio of specific heat constant value or variable name to use in creating/updating the default separation and attachment line attributes and/or selected separation and attachment part(s).

Parameters:

Type	Parameter	Description
string	`name`	scalar field variable name for the ratio of specific heats
float	`value`	constant value (default = 1.4) for the ratio of specific heats

Notes:

The `<name or value>` can be multiple types depending on variable definition.

Example Usage:

```
sepattach: select_default
part: modify_begin
sepattach: ratio_of_spec_heat 1.4
part: modify_end
```

See Also:

Associated sepattach commands

Command:

```
sepattach: select_default
```

Purpose:

Specify that the default attributes of the separation and attachment line parts are to be set for all subsequent separation and attachment line creates and updates.

Parameters:

None.

Example Usage:

```
sepattach: select_default
part: modify_begin
sepattach: momentum none
part: modify_end
```

See Also:

Associated sepattach commands

sepattach: threshold

Command:

```
sepattach: threshold <variable_name>
```

Purpose:

Specify the threshold variable description by which to filter the segments of the selected separation and attachment line parts.

Parameters:

Type	Parameter	Description
string	`variable_name`	name of the scalar variable by which to filter the vortex core segments

Notes:

The default threshold variable for vortex cores is `"fx_sep_att_strengh"`.

Example Usage:

```
part: select_begin
 5
part: select_end
part: modify_begin
sepattach: threshold Density
part: modify_end
```

See Also:

Associated sepattach commands

Command:

```
sepattach: threshold_filter <option>
```

Purpose:

Specify how to filter out segments of the specified separation and attachment line part, according to the specified threshold variable and value.

Parameters:

Type	Parameter	Description	
constant	option	<	filter out all values of the specified threshold variable less than the specified threshold value (default)
		>	filter out all values of the specified threshold variable greater than the specified threshold value

Example Usage:

```
part: select_begin
 5
part: select_end
part: modify_begin
sepattach: threshold_filter >
part: modify_end
```

See Also:

Associated sepattach commands

sepattach: threshold_max/min

Command:

```
sepattach: threshold_max <threshold_value>
sepattach: threshold_min <threshold_value>
```

Purpose:

Specify the maximum/minimum threshold limit under which to filter the threshold value of the selected separation and attachment line part(s).

Parameters:

Type	Parameter	Description
float	`threshold_value`	maximum/minimum threshold variable range limit below which to threshold the separation and attachment lines parts

Notes:

This max must be greater than the min value, and be a valid threshold variable value.

Example Usage:

```
part: select_begin
 5
part: select_end
part: modify_begin
sepattach: threshold_max 1.0000e+01
part: modify_end
```

See Also:

Associated sepattach commands

Command:

```
sepattach: threshold_value <threshold_value>
```

Purpose:

Specify the threshold value by which to filter the segments from the selected separation and attachment line part(s).

Parameters:

Type	Parameter	Description
float	`threshold_value`	variable value by which to apply the threshold filter to the vortex core segments

Notes:

This value is automatically set to the minimum separation and attachment line strength variable value of the separation and attachment line parts.

Example Usage:

```
part: select_begin
 5
part: select_end
part: modify_begin
sepattach: threshold_value 5.0000e-01
part: modify_end
```

See Also:

Associated sepattach commands

sepattach: velocity

Command:

```
sepattach: velocity <variable_name>
```

Purpose:

Specify the velocity variable to use in creating/updating the default separation and attachment line part attributes and/or the selected separation and attachment line part(s).

Parameters:

Type	Parameter	Description
string	`variable_name`	name of the velocity variable (the default <none> means to use nothing)

Notes:

This variable is assigned automatically if the variable name description `<Velocity>` exists. Since the computation of separation and attachment lines depend on velocity, either the velocity variable must be defined or both the density and momentum variables must be defined to obtain the velocity variable.

Example Usage:

```
variables: activate Velocity
sepattach: select_default
part: modify_begin
sepattach: velocity Velocity
part: modify_end
```

See Also:

Associated sepattach commands

Commands in this Section:

shell

shell: <str>

See Also:

Command:

```
shell: <str>
```

Purpose:

Have EnSight execute a system command.

Parameters:

Type	Parameter	Description
string	`str`	a system command line

Notes:

A typical use of this command might be to move animation images from the local directory to an archive location.

Example Usage:

```
shell: cp image*.rgb /scratch/archive/
```

See Also:

Associated shell commands

Commands in this Section:

shock

shock: begin
shock: end
shock: create
shock: density <variable_name>
shock: enthalpy <variable_name>
shock: gas_constant <value or name>
shock: mach <variable_name>
shock: method <algorithm>
shock: momentum <variable_name>
shock: pressure <variable_name>
shock: ratio_of_spec_heat <name or value>
shock: select_default
shock: temperature <variable_name>
shock: threshold <variable_name>
shock: threshold_filter <option>>
shock: threshold_max <threshold_value>
shock: threshold_min <threshold_value>
shock: threshold_value <threshold_value>
shock: total_energy <variable_name>
shock: variable <variable_name>
shock: velocity <variable_name>

See Also:

User Manual - Section 7.20, Shock Surface/Region Create/Update
How To Extract Shock Surfaces

shock: begin/end

Command:

```
shock: begin
<shock_command>
shock: end
```

Purpose:

Delimit the modifications for shock part attributes.

Parameters:

Type	Parameter	Description
string	shock_command	"shock: variable" commands

Notes:

These commands are used to set the attributtes for a shock part creation.

Example Usage:

```
#
# Specify the 3D parent part(s)
# from which to create the shock part
#
part: select_begin
 1
part: select_end
#
# Compute the shock part by the Surface method
#
shock: begin
shock: variable Pressure
shock: end
shock: create
#
# Now select the newly created shock part
# for subsequent operations
#
part: select_begin
 5
part: select_end
```

Or:

```
#
# Specify the 3D parent part(s)
# from which to create the shock part
#
part: select_begin
 1
part: select_end
#
# Compute the shock part by the Region method
#
shock: begin
shock: variable none
shock: end
shock: create
#
```

```
# Now select the newly created shock part
# for subsequent operations
#
part: select_begin
 5
part: select_end
```

See Also:

Associated shock commands

shock: create

Command:

```
shock: create
```

Purpose:

Create a shock part with the currently defined attributes using the currently selected 3D part(s).

Parameters:

None.

Example Usage:

```
#
# Specify the 3D parent part(s)
# from which to create the shock part
#
part: select_begin
 1
part: select_end
#
# Compute the shock part by the Surface method
#
shock: begin
shock: variable Pressure
shock: end
shock: create
#
# Now select the newly created shock part
# for subsequent operations
#
part: select_begin
 5
part: select_end
```

See Also:

Associated shock commands

Command:

```
shock: density <variable_name>
```

Purpose:

Specify the density variable to use in creating/updating a shock part or the default attributes.

Parameters:

Type	Parameter	Description
string	`variable_name`	name of the density variable (the default name `<none>` means to use nothing)

Notes:

This variable is assigned automatically if the variable name `<Density>` exists.

Since the computation of separation and attachment lines depend on density, either the density variable must be defined or both the pressure and temperature variables must be defined to obtain the density variable.

Example Usage:

```
variables: activate Density
shock: select_default
part: modify_begin
shock: density Density
part: modify_end
```

See Also:

Associated shock commands

shock: enthalpy

Command:

```
shock: enthalpy <variable_name>
```

Purpose:

Specify the enthalpy variable to use in creating/updating a shock part or the default attributes.

Parameters:

Type	Parameter	Description
string	`variable_name`	name of the enthalpy variable (the default name `<none>` means to use nothing).

Notes:

This variable is not needed unless you are computing the shock via the Region method and transient (moving) shocks are specified via the "`test: toggle_moving_shock`" command.

Example Usage:

```
test: toggle_moving_shock
variables: activate Enthalpy
shock: select_default
part: modify_begin
shock: density Enthalpy
part: modify_end
```

See Also:

Associated shock commands

Command:

```
shock: gas_constant <value or name>
```

Purpose:

Specify the gas constant as a constant value or a variable name to use in creating/updating a shock part or the default attributes.

Parameters:

Type	Parameter	Description
string	`name`	scalar field variable name for the gas constant
float	`value`	a constant value (default =1.) for the gas constant

Example Usage:

```
shock: select_default
part: modify_begin
shock: gas_constant 1.
part: modify_end
```

See Also:

Associated shock commands

Command:

```
shock: mach <variable_name>
```

Purpose:

Specify the mach variable to use in creating/updating a shock part or the default attributes.

Parameters:

Type	Parameter	Description
string	variable_name	name of the mach variable (the default name <none> means to use nothing)

Notes:

Although the specification of this variable is optional, if it exists, its specification will save the creation of a temporary variable. If this variable is not specified or does not exist, it will be created on-the-fly temporarily as needed from dependent density, energy or pressure, velocity or momentum, and ratio of specific heats variables and values.

Example Usage:

```
variables: activate Mach
vortexcore: select_default
part: modify_begin
vortexcore: density Mach
part: modify_end
```

See Also:

Associated shock commands

Command:

```
shock: method <algorithm>
```

Purpose:

Specify the algorithm to use for the computation of the shock part.

Parameters:

Type	Parameter	Description	
constant	algorithm	region	shock computed based on the Lovely and Haimes algorithm where the pressure gradient is dotted with the mach vector to compute a shock test value at each node of the 3D mesh
		surface	(default) shock computed based on the Pagendarm algorithm where the shock position in the flow field is given by the maximal gradient of a quantity of density or pressure along the local flow direction

Notes:

The Surface method requires the specification of a shock threshold variable. The Region method does not require the specification of a shock threshold variable.

Example Usage:

```
part: select_begin
 8
part: select_end
part: modify_begin
shock: method region
part: modify_end
```

See Also:

Associated shock commands

shock: momentum

Command:

```
shock: momentum <variable_name>
```

Purpose:

Specify the momentum variable to use in creating/updating a shock part or the default attributes.

Parameters:

Type	Parameter	Description
string	`variable_name`	name of the momentum variable (the default name `<none>` means to use nothing

Notes:

This variable is assigned automatically if the variable name “momentum” exists.

Since the computation of shock depends on velocity, either the velocity variable must be defined or both the density and momentum variables must be defined to obtain the velocity variable.

Example Usage:

```
variables: activate momentum
shock: select_default
part: modify_begin
shock: momentum momentum
part: modify_end
```

See Also:

Associated shock commands

Command:

```
shock: pressure <variable_name>
```

Purpose:

Specify the pressure variable to use in creating/updating a shock part or the default attributes.

Parameters:

Type	Parameter	Description
string	variable_name	name of the pressure variable (the default name <none> means to use nothing

Notes:

This variable is used throughout the shock calculation. Although the specification of this variable is optional, if it exists, its specification will save the creation of a temporary variable. If this variable is not specified or does not exist, it will be created on-the-fly temporarily as needed from dependent density, energy, velocity or momentum, and ratio of specific heats variables and values.

Example Usage:

```
variables: activate Pressure
shock: select_default
part: modify_begin
shock: pressure Pressure
part: modify_end
```

See Also:

Associated shock commands

shock: ratio_of_spec_heat

Command:

```
shock: ratio_of_spec_heat <name or value>
```

Purpose:

Specify the ratio of specific heat constant value or variable name to use in creating/updating a shock part or the default attributes.

Parameters:

Type	Parameter	Description
string	`name`	scalar field variable name for the ratio of specific heats
float	`value`	a constant value (default = 1.4) for the ratio of specific heats

Example Usage:

```
shock: select_default
part: modify_begin
shock: ratio_of_spec_heat 1.4
part: modify_end
```

See Also:

Associated shock commands

Command:

```
shock: select_default
```

Purpose:

Specify that the default attributes of the shock part are to be set for all subsequent shock attribute updates.

Parameters:

None.

Example Usage:

```
vortexcore: select_default
part: modify_begin
shock: momentum none
part: modify_end
```

See Also:

Associated shock commands

shock: temperature

Command:

```
shock: temperature <variable_name>
```

Purpose:

Specify the temperature variable to use in creating/updating a shock part or the default attributes.

Parameters:

Type	Parameter	Description
string	`variable_name`	name of the temperature variable (the default name `<none>` means to use nothing

Notes:

Although the specification of this variable is optional, if it exists, its specification will save the creation of a temporary variable.

This variable is required for the creation of Density if Density does not exist. If Density exists, no need to create this variable, unless you would like to use this as the shock creation variable for the Surface method.

Example Usage:

```
variables: activate Temperature
shock: select_default
part: modify_begin
shock: momentum Temperature
part: modify_end
```

See Also:

Associated shock commands

Command:

```
shock: threshold <variable_name>
```

Purpose:

Specify the threshold variable description by which to filter the segments of the selected shock part.

Parameters:

Type	Parameter	Description
string	`variable_name`	name of the scalar variable by which to filter the vortex core segments

Notes:

The default threshold for the Region shock part is `"SHK_Threshold"`. The default threshold for the Region shock part is `"SHK_xxx"`, where xxx is the name of the variable used as the shock creation variable.

Example Usage:

```
part: select_begin
 8
part: select_end
part: modify_begin
shock: threshold Density
part: modify_end
```

See Also:

Associated shock commands

shock: threshold_filter

Command:

```
shock: threshold_filter <option>>
```

Purpose:

Specify how to filter out segments of the specified shock part, according to the specified threshold variable and value.

Parameters:

Type	Parameter	Description	
constant	option	<	(default) filter out all values of the specified threshold variable less than the specified threshold value
		>	filter out all values of the specified threshold variable greater than the specified threshold value

Example Usage:

```
part: select_begin
 8
part: select_end
part: modify_begin
shock: threshold_filter >
part: modify_end
```

See Also

Associated shock commands

Command:

```
shock: threshold_max <threshold_value>
shock: threshold_min <threshold_value>
```

Purpose:

Specify the maximum/minimum threshold limit under which to filter the threshold value of the selected shock part(s).

Parameters:

Type	Parameter	Description
float	`threshold_value`	maximum/minimum threshold variable range limit below/ above which to threshold the shock part

Notes:

Max must be greater than the min value, and be a valid threshold variable value.

Example Usage:

```
part: select_begin
 8
part: select_end
part: modify_begin
shock: threshold_min 0.1
shock: threshold_max 10.0
part: modify_end
```

See Also:

Associated shock commands

shock: threshold_value

Command:

```
shock: threshold_value <threshold_value>
```

Purpose:

Specify the threshold value by which to filter the elements from the selected shock part(s).

Parameters:

Type	Parameter	Description
float	`threshold_value`	variable value by which to apply the threshold filter to the shock elements

Notes:

This value is automatically set to a value that should approximate the valid region in which the shock part should exist. Although for some models, further refinement and adjustment of the threshold min/max values is required.

This value must lie within the threshold variable interval [`threshold_min`, `threshold_max`].

Example Usage:

```
part: select_begin
 8
part: select_end
part: modify_begin
shock: threshold_value 5.0000e-01
part: modify_end
```

See Also:

Associated shock commands

Command:

```
shock: total_energy <variable_name>
```

Purpose:

Specify the total energy variable to use in creating/updating a shock part or the default attributes.

Parameters:

Type	Parameter	Description
string	`variable_name`	name of the total energy variable (the default `<none>` means to use nothing)

Notes:

Although the specification of this variable is optional, if it exists, its specification will save the creation of a temporary variable.

This variable is required for the creation of Pressure if Pressure does not exist. If Pressure exists, no need to create this variable, unless you would like to use this as the shock creation variable for the Surface method.

Example Usage:

```
variables: activate TotalEnergy
vortexcore: select_default
part: modify_begin
shock: velocity TotalEnergy
part: modify_end
```

See Also:

Associated shock commands

Command:

```
shock: variable <variable_name>
```

Purpose:

Specify the variable (i.e Density, Energy, Pressure, or Temperature) in which to create the Surface method shock part(s).

Parameters:

Type	Parameter	Description
string	`variable_name`	name of the variable in which to use as the creation variable or the variable in which to apply the gradient calculations for the shock function

Notes:

For the Surface method, this variable is either Density, Energy, Pressure, or Temperature; and this variable must have previously been defined via its respective shock: density, energy, pressure, or temperature definition.

For the Region method, this variable is always pressure, and thus the specification is not needed; thus, `<none>` is used as the shock variable.

Example Usage:

```
#
# Specify the 3D parent part(s)
# from which to create the shock part
#
part: select_begin
 1
part: select_end
#
# Compute the shock part by the Surface method
#
shock: begin
shock: variable Pressure
shock: end
shock: create
#
# Now select the newly created shock part
# for subsequent operations
#
part: select_begin
 5
part: select_end
```

Or:

```
#
# Specify the 3D parent part(s)
# from which to create the shock part
#
part: select_begin
 1
part: select_end
#
```

```
# Compute the shock part by the Region method
#
shock: begin
shock: variable none
shock: end
shock: create
#
# Now select the newly created shock part
# for subsequent operations
#
part: select_begin
 5
part: select_end
```

See Also:

Associated shock commands

shock: velocity

Command:

```
shock: velocity <variable_name>
```

Purpose:

Specify the velocity variable to use in creating/updating a shock part or the default attributes.

Parameters:

Type	Parameter	Description
string	variable_name	name of the velocity variable (the default <none> means to use nothing)

Notes:

Since the computation of shock depends on velocity, either the velocity variable must be defined or both the density and momentum variables must be defined to obtain the velocity variable.

Example Usage:

```
variables: activate Velocity
shock: select_default
part: modify_begin
shock: velocity Velocity
part: modify_end
```

See Also:

Associated shock commands

Commands in this Section:

show_info

show_info: cursor <x-coord> <y-coord> <z-coord>
show_info: element <element_id_number>
show_info: ijk <i-plane_value> <j-plane_value> <k-plane_value>
show_info: node <node_id_number>
show_info: part <part_id_number>

See Also:

User Manual - Section 6.3, Query Menu Functions

How To Get Point, Node, Element, Part Information

show_info: cursor

Command:

```
show_info: cursor <x-coord> <y-coord> <z-coord>
```

Purpose:

Show corresponding part and values of all active variables at the specified x, y, z model location.

Parameters:

Type	Parameter	Description
float	x-coord	x, y, z locations of the cursor in model coordinates
	y-coord	
	z-coord	

Notes:

Used in conjuction with the following commands.

```
view_transf: function cursor
view_transf: action translate
view_transf: cursor
```

Example Usage:

```
#
# Toggle-on and position the cursor tool
#
view_transf: function cursor
tools: cursor ON
view_transf: action translate
view_transf: cursor -1.0000e+00 0.500000 0.500000
#
# Revert back to global transformation mode
#
view_transf: function global
#
# Show part and all active variable information at cursor location
#
show_info: cursor -1.0000e+00 0.0000e+00 0.0000e+00
```

Execution of the above example shows the following information in the EnSight Message Window for this dataset:

```
*****************
Point (-1.00000e+00,5.00000e-01,5.00000e-01)(In Frame 0) Query Informa-
tion.
Found in unstructured part # 1.
Found in element # 1698.
Closest node # 1173. (within the element)
Value for Variable temperature is 1.08646e+01.
Values for Variable velocity are:
x=3.64549e-01,y=7.84898e-10,z=2.06609e-02,mag=3.65134e-01.
*****************
```

See Also:

Associated show_info commands

Command:

```
show_info: element <element_id_number>
```

Purpose:

Query/show the associated part connectivity and values of all active variables for the specified element id label.

Parameters:

Type	Parameter	Description
int	element_id_number	a valid element id corresponding to the specified part

Example Usage:

```
#
# Specify which part
#
part: select_begin
1
part: select_end
#
# from which to show the corresponding element information.
#
show_info: element 20
```

Execution of the above example shows the following information in the EnSight Message Window for this dataset:

```
*****************
Element 20 Query Information.
Found in unstructured part # 1.
Type of element is 8 Noded hexa
Number of nodes is 8
Node IDs are: 6185 6186 6522 6521 6201 6202 6538 6537
Neighboring Element Information is:
Element neighbor 19 is of type 8 Noded hexa
Element neighbor 504 is of type 8 Noded hexa
Element neighbor 21 is of type 8 Noded hexa
Element neighbor 46 is of type 8 Noded hexa
No element based variables active to show values at the element.
*****************
```

See Also:

Associated show_info commands

Command:

```
show_info: ijk <i-plane_value> <j-plane_value> <k-plane_value>
```

Purpose:

Query/show the associated part connectivity and values of all active variables for the specified i, j, and k location.

Parameters:

Type	Parameter	Description
float	i-plane_value	corresponding i, j, k-plane values
	j-plane_value	
	k-plane_value	

Notes:

Only applicable for structured parts.

Example Usage:

```
#
# Specify the part ...
#
part: select_begin
1
part: select_end
#
# and active variables ...
#
variables: select_varname_begin
Density
Energy
Momentum
Velo
variables: select_varname_end
#
# at which to show the specified ijk information.
#
show_info: ijk 1 41 1
```

Execution of the above example shows the following information in the EnSight Message Window for this dataset:

```
*****************
IJK 1 45 1 Query Information.
Node id is: 2465.
Found in structured part # 1.
Coordinates (In Frame 0) are: (2.15279e-02,0.00000e+00,-4.94343e-04)
Value for Variable Density is 5.01719e-01.
Value for Variable Energy is 9.46637e-01.
Values for Variable Momentum are:
x=0.00000e+00,y=0.00000e+00,z=0.00000e+00,mag=0.00000e+00.
Values for Variable Velo are:
x=0.00000e+00,y=0.00000e+00,z=0.00000e+00,mag=0.00000e+00.
*****************
```

See Also:

Associated show_info commands

show_info: node

Command:

```
show_info: node <node_id_number>
```

Purpose:

Query/show the associated part connectivity and values of all active variables for the specified node id label.

Parameters:

Type	Parameter	Description
int	node_id_number	valid node id corresponding to the specified part

Example Usage:

```
#
# Specify to show the information of node 8000 of part 1
#
part: select_begin
1
part: select_end
show_info: node 8000
```

Execution of the above example shows the following information in the EnSight Message Window for this part:

```
*****************
Node 8000 Query Information.
Coordinates (In Frame 0) are: (2.32510e+00,1.66670e-01,8.75000e-01)
Found in unstructured part # 1.
Found in element # 3104.
Found in element # 3105.
Found in element # 3124.
Found in element # 3125.
Found in element # 3588.
Found in element # 3589.
Found in element # 3608.
Found in element # 3609.
Value for Variable temperature is 1.31550e+01.
Values for Variable velocity are:
x=4.87360e-01,y=5.10460e-02,z=-3.70030e-02,mag=4.91421e-01.
*****************
```

See Also:

Associated show_info commands

Command:

```
show_info: part <part_id_number>
```

Purpose:

Query/show the associated model information and extent values for the specified part id.

Parameters:

Type	Parameter	Description
int	part_id_number	the part GUI number

Example Usage:

```
#
# First, select the part ...
#
part: select_begin
1
part: select_end
#
# Then, specify to show its corresponding information.
#
show_info: part 1
```

Execution of the above example shows the following information in the EnSight Message Window for this part:

```
*****************
Part 1 Query Information.
Unstructured part.
Number of nodes 8872
Minimum coordinate(In Frame 0) is (-2.00000e+00,-
2.00000e+00,0.00000e+00)
Maximum coordinate(In Frame 0) is (3.00000e+00,3.00000e+00,2.00000e+00)
Element Information is:
Element type: 8 Noded hexa, count = 7512.
*****************
```

See Also:

Associated show_info commands

Commands in this Section:

solution_time

solution_time: #_of_cycles <num>
solution_time: begin_simtime <time>
solution_time: begin_step <step>
solution_time: current_simtime <time>
solution_time: current_step <step>
solution_time: end_simtime <time>
solution_time: end_step <step>
solution_time: increment <step_by>
solution_time: show_as <method>
solution_time: timeset_between <method>
solution_time: timeset_leftof <method>
solution_time: timeset_rightof <method>
solution_time: timeset_select <ts_num>
solution_time: timeset_update
solution_time: update_to_current
solution_time: update_to_first
solution_time: update_to_last
solution_time: update_type <method>
solution_time: zoomout

See Also:

User Manual - Section 7.13, Solution Time

How To Change Time Steps

solution_time: #_of_cycles

Command:

```
solution_time: #_of_cycles <num>
```

Purpose:

Sets the number of cycles of cyclic transient data. If the first and last time steps represent the same data, one can effectively increase the apparent number of computed time steps, by using this command.

Parameters:

Type	Parameter	Description
int	num	number of times to cycle the given transient data (default is 1)

Notes:

This is especially useful for things like pathline generation, (where you can only trace as long as you have transient steps) or transient animation that is synchronized to the data.

Example Usage:

```
#
# This sets several of the attributes using time steps
# (Including having the data cycle twice)
#
solution_time: begin_step 0
solution_time: end_step 16
solution_time: current_step 0.0000e+00
solution_time: update_to_current
solution_time: #_of_cycles 2
solution_time: increment 1.0000e+00
solution_time: show_as step
solution_time: update_type discrete
```

See Also:

Associated solution_time commands

Command:

```
solution_time: begin_simtime <time>
solution_time: end_simtime <time>
```

Purpose:

Sets the solution time to be used as the beginning/ending time for subsequent time use.

Parameters:

Type	Parameter	Description
float	`time`	solution time to use as the beginning/ending time

Notes:

This is used when the "`solution_time: show_as`" command is set to "`time`". Also, it will be truncated to a solution time directly associated with a time step if "`solution_time: update_type`" is set to "`discrete`".

Can be greater/lesser or equal to the smallest/largest time available.

Example Usage:

```
#
# This sets several of the attributes using simulation time
#
solution_time: show_as time
solution_time: update_type continuous
solution_time: begin_simtime 0.0000e+00
solution_time: end_simtime 1.6000e+01
solution_time: current_simtime 9.0000e+00
solution_time: update_to_current
solution_time: #_of_cycles 1
solution_time: increment 2.0000e+00
```

See Also:

Associated solution_time commands

solution_time: begin/end_step

Command:

```
solution_time: begin_step <step>
solution_time: end_step <step>
```

Purpose:

Sets the time step to be used as the beginning/ending time step for subsequent time use.

Parameters:

Type	Parameter	Description
int	`step`	time step to use as the beginning/ending time step

Notes:

This is used when the "`solution_time: show_as`" command is set to "`step`".

Can be greater/lesser or equal to the smallest/largest time step available.

Example Usage:

```
#
# This sets several of the attributes using time steps
#
solution_time: show_as step
solution_time: update_type discrete
solution_time: begin_step 0
solution_time: end_step 16
solution_time: current_step 0
solution_time: update_to_current
solution_time: #_of_cycles 1
solution_time: increment 1.0000e+00
```

See Also:

Associated solution_time commands

Command:

```
solution_time: current_simtime <time>
```

Purpose:

Sets the current simulation time. Must be between the beginning time and the ending time.

Parameters:

Type	Parameter	Description
float	`time`	The simulation time to use as the current time. If "`solution_time: update_type`" is "`continuous`", this can be set to be any time between the beginning and ending times, even if it lies between data points (interpolation will take place). If the "`solution_time: update_type`" is "`discrete`", this should be the time associated with a time step.

Notes:

This is used when the "`solution_time: show_as`" command is set to "`time`".

Example Usage:

```
#
# This sets several of the attributes using time steps,
# but the current is set 40% between steps 2 and 3.
#
solution_time: show_as step
solution_time: update_type continuous
solution_time: begin_step 0
solution_time: end_step 16
solution_time: current_step 2.4
solution_time: update_to_current
solution_time: #_of_cycles 1
solution_time: increment 1.0000e+00
```

See Also:

Associated solution_time commands

solution_time: current_step

Command:

```
solution_time: current_step <step>
```

Purpose:

Sets the current time step. Must be between the beginning time step and the ending time step.

Parameters:

Type	Parameter	Description
int or float	`step`	The time step to use as the current time step. If "`solution_time: update_type`" is "`continuous`", this can be float and thus can be set to be fractionally between time steps (interpolation will take place). If the "`solution_time: update_type`" is "`discrete`", this should be an integer.

Notes:

This is used when the "`solution_time: show_as`" command is set to "`step`".

Example Usage:

```
#
# This sets several of the attributes using simulation time
#
solution_time: show_as time
solution_time: update_type continuous
solution_time: begin_simtime 0.0000e+00
solution_time: end_simtime 1.6000e+01
solution_time: current_simtime 9.0000e+00
solution_time: update_to_current
solution_time: #_of_cycles 1
solution_time: increment 2.0000e+00
```

See Also:

Associated solution_time commands

Command:

```
solution_time: increment <step_by>
```

Purpose:

Sets the number of steps that are moved when the solution time dialog time stepper buttons are clicked.

Parameters:

Type	Parameter	Description
int	step_by	number of time steps to advance by or decrease by with one click of the solution time stepper buttons

Notes:

This is a user interface attribute.

Example Usage:

```
#
# This sets several of the attributes using simulation time
# (but at the end it sets the stepper button increment)
#
solution_time: show_as time
solution_time: update_type continuous
solution_time: begin_simtime 0.0000e+00
solution_time: end_simtime 1.6000e+01
solution_time: current_simtime 9.0000e+00
solution_time: update_to_current
solution_time: #_of_cycles 1
solution_time: increment 2.0000e+00
```

See Also:

Associated solution_time commands

solution_time: show_as

Command:

```
solution_time: show_as <method>
```

Purpose:

Sets method to use for specifying time - either as solution time directly or as time steps of data.

Parameters:

Type	Parameter	Description	
constant	method	step	to use time steps. Inherently this is discrete in nature, equal to the number of discrete times that data is given. EnSight can however deal with fractional time steps by interpolating between them
		time	to use simulation time directly. This will also cause interpolation to occur if a time is specified which is not at the time associated with a time step

Example Usage:

```
#
# This sets several of the attributes using simulation time
#
solution_time: show_as time
solution_time: update_type continuous
solution_time: begin_simtime 0.0000e+00
solution_time: end_simtime 1.6000e+01
solution_time: current_simtime 9.0000e+00
solution_time: update_to_current
solution_time: #_of_cycles 1
solution_time: increment 2.0000e+00
#
# But here we change it to use time steps
#
solution_time: show_as step
solution_time: update_type discrete
solution_time: begin_step 0
solution_time: end_step 16
solution_time: current_step 5
solution_time: update_to_current
solution_time: #_of_cycles 1
solution_time: increment 1.0000e+00
```

See Also:

Associated solution_time commands

Command:

```
solution_time: timeset_between <method>
```

Purpose:

When the current time lies between the steps of a given timeset, this command sets how to handle the situation in the span.

Parameters:

Type	Parameter	Description	
constant	method	undefined	when between steps, treat the variable as undefined
		interpolate	when between steps, interpolate the variable value from the left and right steps
		nearest	when between steps, use the variable value at the nearest step
		left	when between steps, use the variable value at the step to the left
		right	when between steps, use the variable values at the step to the right

Notes:

The geometry and each variable can be specified according to its own timeset, with different beginning and ending times and number of steps. These timesets are combined into the solution time range in the solution dialog. But, with this general capability comes the need to know what to do when a current time lies outside the extents of, or between the steps of a given timeset.

Example Usage:

```
#
# Selecting the timeset
#
solution_time: timeset_select 2
#
# Setting the timeset span handling methods
# (Note that we will interpolate if between)
#
solution_time: timeset_leftof undefined
solution_time: timeset_between interpolate
solution_time: timeset_rightof nearest
#
# Update timesets and model display
#
solution_time: timeset_update
```

See Also:

Associated solution_time commands

solution_time: timeset_leftof

Command:

```
solution_time: timeset_leftof <method>
```

Purpose:

When the current time lies before the beginning step of a given timeset, this command sets how to handle the situation.

Parameters:

Type	Parameter	Description	
constant	method	undefined	when before beginning step, treat the variable as undefined
		nearest	when before beginning step, use the variable value at the nearest step

Notes:

The geometry and each variable can be specified according to its own timeset, with different beginning and ending times and number of steps. These timesets are combined into the solution time range in the solution dialog. But, with this general capability comes the need to know what to do when a current time lies outside the extents of, or between the steps of a given timeset.

Example Usage:

```
#
# Selecting the timeset
#
solution_time: timeset_select 2
#
# Setting the timeset span handling methods
# (Note that we will treat as undefined if left of)
#
solution_time: timeset_leftof undefined
solution_time: timeset_between interpolate
solution_time: timeset_rightof nearest
#
# Update timesets and model display
#
solution_time: timeset_update
```

See Also:

Associated solution_time commands

Command:

```
solution_time: timeset_rightof <method>
```

Purpose:

When the current time lies after the ending step of a given timeset, this command sets how to handle the situation.

Parameters:

Type	Parameter	Description	
constant	method	undefined	after the ending step, treat the variable as undefined
		nearest	after the ending step, use the variable value at the nearest step

Notes:

The geometry and each variable can be specified according to its own timeset, with different beginning and ending times and number of steps. These timesets are combined into the solution time range in the solution dialog. But, with this general capability comes the need to know what to do when a current time lies outside the extents of, or between the steps of a given timeset.

Example Usage:

```
#
# Selecting the timeset
#
solution_time: timeset_select 2
#
# Setting the timeset span handling methods
# (Note that we will treat as nearest if right of)
#
solution_time: timeset_leftof undefined
solution_time: timeset_between interpolate
solution_time: timeset_rightof nearest
#
# Update timesets and model display
#
solution_time: timeset_update
```

See Also:

Associated solution_time commands

solution_time: timeset_select

Command:

```
solution_time: timeset_select <ts_num>
```

Purpose:

Selects the timeset to modify.

Parameters:

Type	Parameter	Description
int	ts_num	timeset number

Notes:

The geometry and each variable can be specified according to its own timeset, with different beginning and ending times and number of steps. These timesets are combined into the solution time range in the solution dialog. But, with this general capability comes the need to know what to do when a current time lies outside the extents of, or between the steps of a given timeset.

Example Usage:

```
#
# Selecting the timeset
#
solution_time: timeset_select 2
#
# Setting the timeset span handling methods
# (Note that we will treat as nearest if right of)
#
solution_time: timeset_leftof undefined
solution_time: timeset_between interpolate
solution_time: timeset_rightof nearest
#
# Update timesets and model display
#
solution_time: timeset_update
```

See Also:

Associated solution_time commands

Command:

```
solution_time: timeset_update
```

Purpose:

Updates the selected timeset and any associated display of the model according to the span handling methods specified.

Parameters:

none

Notes:

The geometry and each variable can be specified according to its own timeset, with different beginning and ending times and number of steps. These timesets are combined into the solution time range in the solution dialog. But, with this general capability comes the need to know what to do when a current time lies outside the extents of, or between the steps of a given timeset.

Example Usage:

```
#
# Selecting the timeset
#
solution_time: timeset_select 2
#
# Setting the timeset span handling methods
# (Note that we will treat as nearest if right of)
#
solution_time: timeset_leftof undefined
solution_time: timeset_between interpolate
solution_time: timeset_rightof nearest
#
# Update timesets and model display
#
solution_time: timeset_update
```

See Also:

Associated solution_time commands

solution_time: update_to_current

Command:

```
solution_time: update_to_current
```

Purpose:

Causes the display and all associated computations of the model to be updated to the currently specified time or step.

Parameters:

none

Example Usage:

```
#
# This sets several of the attributes using simulation time
#
solution_time: show_as time
solution_time: update_type continuous
solution_time: begin_simtime 0.0000e+00
solution_time: end_simtime 1.6000e+01
solution_time: current_simtime 9.0000e+00
solution_time: update_to_current
solution_time: #_of_cycles 1
solution_time: increment 2.0000e+00
#
# But here we change it to use time steps
#
solution_time: show_as step
solution_time: update_type discrete
solution_time: begin_step 0
solution_time: end_step 16
solution_time: current_step 5
solution_time: update_to_current
solution_time: #_of_cycles 1
solution_time: increment 1.0000e+00
```

See Also:

Associated solution_time commands

Command:

```
solution_time: update_to_first
```

Purpose:

Causes the display and all associated computations of the model to be updated to the first time or step of the model.

Parameters:

none

Notes:

This command is not generated by the EnSight GUI, but is intended for use with batch or context operations where the time of the first step could vary from model to model.

Example Usage:

```
solution_time: update_to_first
```

See Also:

Associated solution_time commands

solution_time: update_to_last

Command:

```
solution_time: update_to_last
```

Purpose:

Causes the display and all associated computations of the model to be updated to the last time or step of the model.

Parameters:

none

Notes:

This command is not generated by the EnSight GUI, but is intended for use with batch or context operations where the time of the last step could vary from model to model.

Example Usage:

```
solution_time: update_to_last
```

See Also:

Associated solution_time commands

Command:

```
solution_time: update_type <method>
```

Purpose:

Sets whether times or time steps can be specified between the data steps in the model, thus causing interpolation to be needed.

Parameters:

Type	Parameter	Description	
constant	method	discrete	no in-between steps or times can be specified
		continuous	fractional time steps or simulation times not at a step can be specified

Notes:

This command and "solution_time: show_as" combine to control whether interpolation will be needed.

Example Usage:

```
#
# This sets several of the attributes using simulation time
#
solution_time: show_as time
solution_time: update_type continuous
solution_time: begin_simtime 0.0000e+00
solution_time: end_simtime 1.6000e+01
solution_time: current_simtime 9.0000e+00
solution_time: update_to_current
solution_time: #_of_cycles 1
solution_time: increment 2.0000e+00
#
# But here we change it to use time steps
#
solution_time: show_as step
solution_time: update_type discrete
solution_time: begin_step 0
solution_time: end_step 16
solution_time: current_step 5
solution_time: update_to_current
solution_time: #_of_cycles 1
solution_time: increment 1.0000e+00
```

See Also:

Associated solution_time commands

solution_time: zoomout

Command:

```
solution_time: zoomout
```

Purpose:

Resets the begin and end steps (or begin and end simtimes) to the widest possible settings.

Parameters:

none

Notes:

This may be desirable after one has modified (narrowed) the begin and end steps or time for a given temporary purpose.

Example Usage:

```
solution_time: show_as step
solution_time: update_type discrete
solution_time: begin_step 3
solution_time: end_step 12
solution_time: current_step 5
solution_time: update_to_current
solution_time: #_of_cycles 1
solution_time: increment 1.0000e+00
#
# One could load a flipbook or something like
# that at this point, using the narrowed range
# of 3 to 12.  Then could use the next command
# to get the range back to 0 to 16
#
solution_time: zoomout
```

See Also:

Associated solution_time commands

Commands in this Section:

subset

See Also:

User Manual - Section 7.16, Subset Parts Create/Update

How To Create Subset Parts

subset: begin/end

Command:

```
subset: begin
<subset_commands>
subset: end
```

Purpose:

Delimit the modifications for subset part attributes.

Parameters:

Type	Parameter	Description
string	subset_commands	any of the subset attribute commands

Notes:

These commands are used to set the attributes for a subset part creation.

Example Usage:

```
subset: begin
subset: begin_subsetdesc
part_num 1
nodes
1-10
elements
20-30
subset: end_subsetdesc
subset: end
subset: create
```

See Also:

Associated subset commands

Command:

```
subset: begin_subsetdesc
part_num <num>
nodes
<nodes_string>
elements
<elements_string>
subset: end_subsetdesc
```

Purpose:

Encloses the information for parent part number, nodes, and elements for a subset part.

Parameters:

Type	Parameter	Description
int	num	Parent part number of the subset part. Place it on the line with the keyword "part_num", as shown in the example.
string	nodes_string	List of nodes to include in the subset part. Place the keyword "nodes" on a line, then place this string on subsequent lines. Note that the list is a comma separated list of node ids which can also include dash separated ranges of nodes.
string	elements_string	List of elements to include in the subset part. Place the keyword "elements" on a line, then place this string on subsequent lines. Note that the list is a comma separated list of element ids which can also include dash separated ranges of elements.

Example Usage:

```
subset: begin
subset: begin_subsetdesc
part_num 1
nodes
13,1-10,103,104,200-3000
elements
20-30,100-200,355
subset: end_subsetdesc
subset: end
subset: create
```

See Also:

Associated subset commands

subset: create

Command:

```
subset: create
```

Purpose:

Creates a subset part according to previously defined attributes or defaults.

Parameters:

none

Example Usage:

```
subset: begin
subset: begin_subsetdesc
part_num 1
nodes
13,1-10,103,104,200-3000
elements
20-30,100-200,355
subset: end_subsetdesc
subset: end
subset: create
```

See Also:

Associated subset commands

Command:

```
subset: select_default
```

Purpose:

Sets to default subset part, so subsequent attribute changes will apply to the defaults.

Parameters:

none

Example Usage:

```
subset: select_default
part: modify_begin
part: colorby_rgb 6.0000e-01 0.0000e+00 1.0000e+00
part: modify_end
```

See Also:

Associated subset commands

Commands in this Section:

tensor

tensor: begin
tensor: end
tensor: color_by <method>
tensor: compression_line_width <width>
tensor: compression_rgb <red_val> <grn_val> <blu_val>
tensor: create
tensor: display_compression <toggle>
tensor: display_major <toggle>
tensor: display_middle <toggle>
tensor: display_minor <toggle>
tensor: display_tension <toggle>
tensor: line_width_by <method>
tensor: scale_factor <value>
tensor: select_default
tensor: tension_line_width <width>
tensor: tension_rgb <red_val> <grn_val> <blu_val>
tensor: tip_shape <shape>
tensor: tip_size <size>
tensor: variable <name>

See Also:

User Manual - Section 7.17, Tensor Glyph Parts Create/Update

How To Create Tensor Glyphs

tensor: begin/end

Command:

```
tensor: begin
<tensor_command>
<tensor_command>
.
.
.
<tensor_command>
tensor: end
```

Purpose:

Delimit the modifications for tensor parts.

Parameters:

Type	Parameter	Description
string	`tensor_command`	any of the tensor attribute commands

Notes:

These commands are used to set the attributes for tensor glyph part creation.

Example Usage:

```
#
# Activating a tensor variable
#
variables: activate uniaxial_tensor
#
# Selecting all parts as parents
#
part: select_all
#
# Grouping a bunch of tensor attribute commands
#
tensor: begin
tensor: variable uniaxial_tensor
tensor: display_compression ON
tensor: display_tension ON
tensor: display_major ON
tensor: display_minor ON
tensor: display_middle ON
tensor: tip_shape triangle
tensor: tip_size 1.0000e-01
tensor: color_by direction
tensor: compression_rgb 1.0000e+00 0.0000e+00 4.0000e-01
tensor: tension_rgb 0.0000e+00 8.0000e-01 1.0000e+00
tensor: line_width_by direction
tensor: compression_line_width 2
tensor: tension_line_width 4
tensor: end
#
# Creating the tensor glyph part
#
tensor: create
```

See Also:

Associated tensor commands

Command:

```
tensor: color_by <method>
```

Purpose:

Sets the method by which a tensor glyph part is colored.

Parameters:

Type	Parameter	Description	
constant	method	part	tensor glyph part will be colored according to the "part: colorby_rgb" command, just as any other part
		direction	tensor glyph part will be colored by the rgb values set for tension and compression using the "tensor: tension_rgb" and "tensor: compression_rgb" commands

Example Usage:

```
#
# Activating a tensor variable
#
variables: activate uniaxial_tensor
#
# Selecting all parts as parents
#
part: select_all
#
# Grouping a bunch of tensor attribute commands
#
tensor: begin
tensor: variable uniaxial_tensor
tensor: display_compression ON
tensor: display_tension ON
tensor: display_major ON
tensor: display_minor ON
tensor: display_middle ON
tensor: tip_shape triangle
tensor: tip_size 1.0000e-01
tensor: color_by direction
tensor: compression_rgb 1.0000e+00 0.0000e+00 4.0000e-01
tensor: tension_rgb 0.0000e+00 8.0000e-01 1.0000e+00
tensor: line_width_by direction
tensor: compression_line_width 2
tensor: tension_line_width 4
tensor: end
#
# Creating the tensor glyph part
#
tensor: create
#
# Now selecting the tensor glyph part
#
part: select_begin
 5
part: select_end
#
```

```
# And changing to color by part instead of direction
#
part: modify_begin
tensor: color_by part
part: modify_end
```

See Also:

Associated tensor commands

Command:

```
tensor: compression_line_width <width>
```

Purpose:

Sets the line width (in pixels) that will be used for the compression portion of a tensor glyph if the "`tensor: line_width_by`" command method is set to "`direction`".

Parameters:

Type	Parameter	Description
int	`width`	width in pixels for the compression portion of a tensor glyph (range is 1 to 4)

Notes:

If the "`tensor: line_width_by`" command method is set to "`part`", the tensor glyph part is line width is set in the same manner as any other part.

Example Usage:

```
#
# Activating a tensor variable
#
variables: activate uniaxial_tensor
#
# Selecting all parts as parents
#
part: select_all
#
# Grouping a bunch of tensor attribute commands
# (included is the one to make compression
#  lines 2 pixels wide)
#
tensor: begin
tensor: variable uniaxial_tensor
tensor: display_compression ON
tensor: display_tension ON
tensor: display_major ON
tensor: display_minor ON
tensor: display_middle ON
tensor: tip_shape triangle
tensor: tip_size 1.0000e-01
tensor: color_by direction
tensor: compression_rgb 1.0000e+00 0.0000e+00 4.0000e-01
tensor: tension_rgb 0.0000e+00 8.0000e-01 1.0000e+00
tensor: line_width_by direction
tensor: compression_line_width 2
tensor: tension_line_width 4
tensor: end
#
# Creating the tensor glyph part
#
tensor: create
```

See Also:

Associated tensor commands

tensor: compression_rgb

Command:

```
tensor: compression_rgb <red_val> <grn_val> <blu_val>
```

Purpose:

Sets the color that will be used for the compression portion of a tensor glyph if the "`tensor: color_by`" command method is set to "`direction`".

Parameters:

Type	Parameter	Description
float	`red_val`	red, green, blue color components for the compression line (0.0 to 1.0)
	`grn_val`	
	`blu_val`	

Notes:

If the "`tensor: color_by`" command method is set to "`part`", the tensor glyph part is colored like any other part.

Example Usage:

```
#
# Activating a tensor variable
#
variables: activate uniaxial_tensor
#
# Selecting all parts as parents
#
part: select_all
#
# Grouping a bunch of tensor attribute commands
# (included is the one to make compression
#  lines a particular color)
#
tensor: begin
tensor: variable uniaxial_tensor
tensor: display_compression ON
tensor: display_tension ON
tensor: display_major ON
tensor: display_minor ON
tensor: display_middle ON
tensor: tip_shape triangle
tensor: tip_size 1.0000e-01
tensor: color_by direction
tensor: compression_rgb 1.0000e+00 0.0000e+00 4.0000e-01
tensor: tension_rgb 0.0000e+00 8.0000e-01 1.0000e+00
tensor: line_width_by direction
tensor: compression_line_width 2
tensor: tension_line_width 4
tensor: end
#
# Creating the tensor glyph part
#
tensor: create
```

See Also:

Associated tensor commands

Command:

```
tensor: create
```

Purpose:

Causes a tensor glyph part to be created according to previously set attributes or defaults.

Parameters:

none

Notes:

The selection of parent part(s) is required.

Example Usage:

```
#
# Activating a tensor variable
#
variables: activate uniaxial_tensor
#
# Selecting all parts as parents
#
part: select_all
#
# Grouping a bunch of tensor attribute commands
#
tensor: begin
tensor: variable uniaxial_tensor
tensor: display_compression ON
tensor: display_tension ON
tensor: display_major ON
tensor: display_minor ON
tensor: display_middle ON
tensor: tip_shape triangle
tensor: tip_size 1.0000e-01
tensor: color_by direction
tensor: compression_rgb 1.0000e+00 0.0000e+00 4.0000e-01
tensor: tension_rgb 0.0000e+00 8.0000e-01 1.0000e+00
tensor: line_width_by direction
tensor: compression_line_width 2
tensor: tension_line_width 4
tensor: end
#
# Creating the tensor glyph part
#
tensor: create
```

See Also:

Associated tensor commands

tensor: display_compression

Command:

```
tensor: display_compression <toggle>
```

Purpose:

Specifies whether the compression portion of a tensor glyph will be displayed or not.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	to have the compression portion of the glyph displayed
		OFF	to have the compression portion of the glyph displayed

Example Usage:

```
#
# Activating a tensor variable
#
variables: activate uniaxial_tensor
#
# Selecting all parts as parents
#
part: select_all
#
# Grouping a bunch of tensor attribute commands
# (One of which is to have the compression displayed)
#
tensor: begin
tensor: variable uniaxial_tensor
tensor: display_compression ON
tensor: display_tension ON
tensor: display_major ON
tensor: display_minor ON
tensor: display_middle ON
tensor: tip_shape triangle
tensor: tip_size 1.0000e-01
tensor: color_by direction
tensor: compression_rgb 1.0000e+00 0.0000e+00 4.0000e-01
tensor: tension_rgb 0.0000e+00 8.0000e-01 1.0000e+00
tensor: line_width_by direction
tensor: compression_line_width 2
tensor: tension_line_width 4
tensor: end
#
# Creating the tensor glyph part
#
tensor: create
#
# Selecting the newly created tensor glyph part
#
part: select_begin
 5
part: select_end
#
# And turning compression portion off - to
# show how it can be done
#
part: modify_begin
tensor: display_compression OFF
```

```
part: modify_end
```

See Also:

Associated tensor commands

tensor: display_major/middle/minor

Command:

```
tensor: display_major <toggle>
tensor: display_middle <toggle>
tensor: display_minor <toggle>
```

Purpose:

Specifies whether the major, middle, or minor eigenvectors of a tensor glyph will be displayed.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	to have the specified eigenvector of the glyph displayed
		OFF	to not have the specified eigenvector of the glyph displayed

Example Usage:

```
#
# Activating a tensor variable
#
variables: activate uniaxial_tensor
#
# Selecting all parts as parents
#
part: select_all
#
# Grouping a bunch of tensor attribute commands
# (Note that all eigenvectors are set to be displayed)
#
tensor: begin
tensor: variable uniaxial_tensor
tensor: display_compression ON
tensor: display_tension ON
tensor: display_major ON
tensor: display_minor ON
tensor: display_middle ON
tensor: tip_shape triangle
tensor: tip_size 1.0000e-01
tensor: color_by direction
tensor: compression_rgb 1.0000e+00 0.0000e+00 4.0000e-01
tensor: tension_rgb 0.0000e+00 8.0000e-01 1.0000e+00
tensor: line_width_by direction
tensor: compression_line_width 2
tensor: tension_line_width 4
tensor: end
#
# Creating the tensor glyph part
#
tensor: create
#
# Selecting the newly created tensor glyph part
#
part: select_begin
 5
part: select_end
#
# And turning off the middle eigenvector
```

```
#
part: modify_begin
tensor: display_middle OFF
part: modify_end
```

See Also:

Associated tensor commands

tensor: display_tension

Command:

```
tensor: display_tension <toggle>
```

Purpose:

Specifies whether the tension portion of a tensor glyph will be displayed.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	to have the tension portion of the glyph displayed
		OFF	to not have the tension portion of the glyph displayed

Example Usage:

```
#
# Activating a tensor variable
#
variables: activate uniaxial_tensor
#
# Selecting all parts as parents
#
part: select_all
#
# Grouping a bunch of tensor attribute commands
# (One of which is to have the tension displayed)
#
tensor: begin
tensor: variable uniaxial_tensor
tensor: display_compression ON
tensor: display_tension ON
tensor: display_major ON
tensor: display_minor ON
tensor: display_middle ON
tensor: tip_shape triangle
tensor: tip_size 1.0000e-01
tensor: color_by direction
tensor: compression_rgb 1.0000e+00 0.0000e+00 4.0000e-01
tensor: tension_rgb 0.0000e+00 8.0000e-01 1.0000e+00
tensor: line_width_by direction
tensor: compression_line_width 2
tensor: tension_line_width 4
tensor: end
#
# Creating the tensor glyph part
#
tensor: create
#
# Selecting the newly created tensor glyph part
#
part: select_begin
 5
part: select_end
#
# And turning tension portion off - to
# show how it can be done
#
part: modify_begin
tensor: display_tension OFF
```

```
part: modify_end
```

See Also:

Associated tensor commands

tensor: line_width_by

Command:

```
tensor: line_width_by <method>
```

Purpose:

Sets the method by which a tensor glyph part's line width is controlled.

Parameters:

Type	Parameter	Description	
constant	method	part	tensor glyph part's line width will be set just as any other part
		direction	tensor glyph part's line width will be set by the "tensor: compression_line_width" and "tensor: tension_line_width" command values

Example Usage:

```
#
# Activating a tensor variable
#
variables: activate uniaxial_tensor
#
# Selecting all parts as parents
#
part: select_all
#
# Grouping a bunch of tensor attribute commands
# (Note line width set by direction)
#
tensor: begin
tensor: variable uniaxial_tensor
tensor: display_compression ON
tensor: display_tension ON
tensor: display_major ON
tensor: display_minor ON
tensor: display_middle ON
tensor: tip_shape triangle
tensor: tip_size 1.0000e-01
tensor: color_by direction
tensor: compression_rgb 1.0000e+00 0.0000e+00 4.0000e-01
tensor: tension_rgb 0.0000e+00 8.0000e-01 1.0000e+00
tensor: line_width_by direction
tensor: compression_line_width 2
tensor: tension_line_width 4
tensor: end
#
# Creating the tensor glyph part
#
tensor: create
#
# Now selecting the tensor glyph part
#
part: select_begin
 5
part: select_end
#
```

```
# And changing to line width to be by part instead of direction
#
part: modify_begin
tensor: line_width_by part
part: modify_end
```

See Also:

Associated tensor commands

tensor: scale_factor

Command:

```
tensor: scale_factor <value>
```

Purpose:

Sets the scale factor to be applied to the tensor glyph.

Parameters:

Type	Parameter	Description
float	value	the scale factor that to apply to the tensor glyph

Notes:

The factor is applied to major, middle, and minor eigenvectors.

Example Usage:

```
#
# Activating a tensor variable
#
variables: activate uniaxial_tensor
#
# Selecting all parts as parents
#
part: select_all
#
# Grouping a bunch of tensor attribute commands
# (One of which is the scale factor)
#
tensor: begin
tensor: variable uniaxial_tensor
tensor: scale_factor 1.6700e+00
tensor: display_compression ON
tensor: display_tension ON
tensor: display_major ON
tensor: display_minor ON
tensor: display_middle ON
tensor: tip_shape triangle
tensor: tip_size 1.0000e-01
tensor: color_by direction
tensor: compression_rgb 1.0000e+00 0.0000e+00 4.0000e-01
tensor: tension_rgb 0.0000e+00 8.0000e-01 1.0000e+00
tensor: line_width_by direction
tensor: compression_line_width 2
tensor: tension_line_width 4
tensor: end
#
# Creating the tensor glyph part
#
tensor: create
#
# Now selecting the tensor glyph part
#
part: select_begin
 5
part: select_end
#
# And changing the scale factor back to 1
#
```

```
part: modify_begin
tensor: scale_factor 1.0000e+00
part: modify_end
```

See Also:

Associated tensor commands

tensor: select_default

Command:

```
tensor: select_default
```

Purpose:

Sets default as the current tensor glyph selection, so subsequent attribute modifications will affect the default.

Parameters:

none

Notes:

When certain attributes will be common to future tensor glyph parts, it is useful to set these once as the default.

Example Usage:

```
#
# Setting to default
#
tensor: select_default
#
# Setting a few things as defaults
#
part: modify_begin
tensor: scale_factor 1.5000e+00
tensor: display_middle OFF
tensor: color_by direction
tensor: compression_rgb 1.0000e+00 0.0000e+00 0.0000e+00
tensor: tension_rgb 0.0000e+00 0.0000e+00 1.0000e+00
tensor: line_width_by part
part: modify_end
```

See Also:

Associated tensor commands

Command:

```
tensor: tension_line_width <width>
```

Purpose:

Sets the line width (in pixels) that will be used for the tension portion of a tensor glyph if the "`tensor: line_width_by`" command method is set to "`direction`".

Parameters:

Type	Parameter	Description
int	`width`	width in pixels for the tension portion of a tensor glyph (range is 1 to 4)

Notes:

If the "`tensor: line_width_by`" command method is set to "`part`", the tensor glyph part is line width is set in the same manner as any other part.

Example Usage:

```
#
# Activating a tensor variable
#
variables: activate uniaxial_tensor
#
# Selecting all parts as parents
#
part: select_all
#
# Grouping a bunch of tensor attribute commands
# (included is the one to make tension
#  lines 4 pixels wide)
#
tensor: begin
tensor: variable uniaxial_tensor
tensor: display_compression ON
tensor: display_tension ON
tensor: display_major ON
tensor: display_minor ON
tensor: display_middle ON
tensor: tip_shape triangle
tensor: tip_size 1.0000e-01
tensor: color_by direction
tensor: compression_rgb 1.0000e+00 0.0000e+00 4.0000e-01
tensor: tension_rgb 0.0000e+00 8.0000e-01 1.0000e+00
tensor: line_width_by direction
tensor: compression_line_width 2
tensor: tension_line_width 4
tensor: end
#
# Creating the tensor glyph part
#
tensor: create
```

See Also:

Associated tensor commands

tensor: tension_rgb

Command:

```
tensor: tension_rgb <red_val> <grn_val> <blu_val>
```

Purpose:

Sets the color that will be used for the tension portion of a tensor glyph if the "`tensor: color_by`" command method is set to "`direction`".

Parameters:

Type	Parameter	Description
float	`red_val`	red, green, blue color components for the tension line (0.0 to 1.0)
	`grn_val`	
	`blu_val`	

Notes:

If the "`tensor: color_by`" command method is set to "`part`", the tensor glyph part is colored like any other part.

Example Usage:

```
#
# Activating a tensor variable
#
variables: activate uniaxial_tensor
#
# Selecting all parts as parents
#
part: select_all
#
# Grouping a bunch of tensor attribute commands
# (included is the one to make tension
#  lines a particular color)
#
tensor: begin
tensor: variable uniaxial_tensor
tensor: display_compression ON
tensor: display_tension ON
tensor: display_major ON
tensor: display_minor ON
tensor: display_middle ON
tensor: tip_shape triangle
tensor: tip_size 1.0000e-01
tensor: color_by direction
tensor: compression_rgb 1.0000e+00 0.0000e+00 4.0000e-01
tensor: tension_rgb 0.0000e+00 8.0000e-01 1.0000e+00
tensor: line_width_by direction
tensor: compression_line_width 2
tensor: tension_line_width 4
tensor: end
#
# Creating the tensor glyph part
#
tensor: create
```

See Also:

Associated tensor commands

Command:

```
tensor: tip_shape <shape>
```

Purpose:

Sets the shape of the tips on the tensor glyphs

Parameters:

Type	Parameter	Description	
constant	shape	none	glyphs will have no tips
		normal	glyphs will have typical 2-line arrowheads
		triangle	glyphs will have triangular arrowheads

Notes:

The direction of the arrowheads will be towards each for compression and away from each other for tension.

Example Usage:

```
#
# Activating a tensor variable
#
variables: activate uniaxial_tensor
#
# Selecting all parts as parents
#
part: select_all
#
# Grouping a bunch of tensor attribute commands
# (included is the one to make triangle arrowhead tips)
#
tensor: begin
tensor: variable uniaxial_tensor
tensor: display_compression ON
tensor: display_tension ON
tensor: display_major ON
tensor: display_minor ON
tensor: display_middle ON
tensor: tip_shape triangle
tensor: tip_size 1.0000e-01
tensor: color_by direction
tensor: compression_rgb 1.0000e+00 0.0000e+00 4.0000e-01
tensor: tension_rgb 0.0000e+00 8.0000e-01 1.0000e+00
tensor: line_width_by direction
tensor: compression_line_width 2
tensor: tension_line_width 4
tensor: end
#
# Creating the tensor glyph part
#
tensor: create
```

See Also:

Associated tensor commands

tensor: tip_size

Command:

```
tensor: tip_size <size>
```

Purpose:

Sets the size of the tips on the tensor glyphs

Parameters:

Type	Parameter	Description
float	size	a relative size factor for tensor glyph tips

Example Usage:

```
#
# Activating a tensor variable
#
variables: activate uniaxial_tensor
#
# Selecting all parts as parents
#
part: select_all
#
# Grouping a bunch of tensor attribute commands
# (included is the one to set tip size)
#
tensor: begin
tensor: variable uniaxial_tensor
tensor: display_compression ON
tensor: display_tension ON
tensor: display_major ON
tensor: display_minor ON
tensor: display_middle ON
tensor: tip_shape triangle
tensor: tip_size 1.0000e-01
tensor: color_by direction
tensor: compression_rgb 1.0000e+00 0.0000e+00 4.0000e-01
tensor: tension_rgb 0.0000e+00 8.0000e-01 1.0000e+00
tensor: line_width_by direction
tensor: compression_line_width 2
tensor: tension_line_width 4
tensor: end
#
# Creating the tensor glyph part
#
tensor: create
```

See Also:

Associated tensor commands

Command:

```
tensor: variable <name>
```

Purpose:

Specifies the name of the tensor variable that the tensor glyphs will represent.

Parameters:

Type	Parameter	Description
string	name	name of the tensor variable

Example Usage:

```
#
# Activating a tensor variable
#
variables: activate uniaxial_tensor
#
# Selecting all parts as parents
#
part: select_all
#
# Grouping a bunch of tensor attribute commands
# (included the one that specifies the name)
#
tensor: begin
tensor: variable uniaxial_tensor
tensor: display_compression ON
tensor: display_tension ON
tensor: display_major ON
tensor: display_minor ON
tensor: display_middle ON
tensor: tip_shape triangle
tensor: tip_size 1.0000e-01
tensor: color_by direction
tensor: compression_rgb 1.0000e+00 0.0000e+00 4.0000e-01
tensor: tension_rgb 0.0000e+00 8.0000e-01 1.0000e+00
tensor: line_width_by direction
tensor: compression_line_width 2
tensor: tension_line_width 4
tensor: end
#
# Creating the tensor glyph part
#
tensor: create
```

See Also:

Associated tensor commands

Commands in this Section:

text

See Also:

User Manual - Section 8.2, Annot Mode

How To Create Text Annotation

text: change_text

Command:

```
text: change_text  <revised_text>
```

Purpose:

Replaces the currently selected annotation text.

Parameters:

Type	Parameter	Description
string	`revised_text`	desired revised text

Example Usage:

```
text: select_begin
 1
text: select_end
text: change_text
this is the revised text!
```

See Also:

Associated text commands

Command:

```
text: delete
```

Purpose:

Deletes the currently selected text in the graphics window.

Parameters:

none

Example Usage:

```
text: select_begin
 1 2 3
text: select_end
text: delete
```

See Also:

Associated text commands

text: justification

Command:

```
text: justification <justification_type>
```

Purpose:

Specifies the placement of the selected text in the graphics window, in relation to the point specified in the `text: location_x` and `location_y` commands.

Parameters:

Type	Parameter	Description
constant	`justification_type`	Must have one of the following values:
		`left`
		`center`
		`right`

Notes:

The default value for justification is left.

Also denotes the point about which the text rotates.

Example Usage:

```
text: select_begin
 1
text: select_end
text: location_x  0.5
text: location_y  0.6
text: justification left
```

See Also:

Associated text commands

Command:

```
text: location_x <coord>
text: location_y <coord>
```

Purpose:

Specifies the x/y coordinate of the selected text in the currently selected viewport.

Parameters:

Type	Parameter	Description
float	`coord`	A value of 0.0 indicates the left/bottom side of the window, while a value of 1.0 indicates the right/top side.

Notes:

The text position can also be in reference to a specific viewport.

See "`text: relative_viewport`" command.

Example Usage:

```
text: select_begin
 1
text: select_end
text: relative_viewport  2
text: location_x  0.5
text: location_y  0.5
text: justification center
```

See Also:

Associated text commands

text: modify_begin/end

Command:

```
text: modify_begin
text: modify_end
```

Purpose:

Delimit the modifications for text attributes.

Notes:

These optional commands can be used to execute a number of attribute changes at one time for better performance.

Parameters:

none

Example Usage:

```
text: select_begin
 1
text: select_end
text: modify_begin
text: rgb 1 0 0
text location_x 100
text location_y 20
text modify_end
```

See Also:

Associated text commands

Command:

```
text: new_text  <text>
```

Purpose:

Creates new text in the graphics window.

Parameters:

Type	Parameter	Description
string	text	desired text

Example Usage:

```
text: new_text This is the new text!
text: location_x  0.5
text: location_y  0.6
```

See Also:

Associated text commands

text: relative_viewport

Command:

```
text: relative_viewport <viewport>
```

Purpose:

Specifies the viewport to which location of the selected text will be relative, when specified by the `text: location_x` and `location_y` commands.

Parameters:

Type	Parameter	Description
int	`viewport`	specifies the viewport

Notes:

A value of zero indicates the graphics window.

Example Usage:

```
text: select_begin
 1
text: select_end
text: relative_viewport 0
text: location_x  0.5
text: location_y  0.6
```

See Also:

Associated text commands

Command:

```
text: rgb <red_val> <grn_val> <blu_val>
```

Purpose:

Specifies the RGB color value of the selected text in the graphics window.

Parameters:

Type	Parameter	Description
float	red_val	red, green, blue color components, (0.0 to 1.0).
	grn_val	
	blu_val	

Example Usage:

```
text: select_begin
 1
text: select_end
text: rgb 1.0 0 0
```

See Also:

Associated text commands

text: rotational_angle

Command:

```
text: rotational_angle <angle>
```

Purpose:

Specifies the angle of rotation of the selected text in the graphics window.

Parameters:

Type	Parameter	Description
float	`angle`	must be within the range of 0.0 to 360.0

Notes:

A value of 0 for rotation corresponds to the normal, upright orientation of text, with clockwise rotation at higher values.

The text rotates about the justification point.

Example Usage:

```
text: select_begin
 1
text: select_end
text: rotational_angle 45.0
```

See Also:

Associated text commands

Command:

```
text: select_all
```

Purpose:

Selects all of the defined text annotations as the target for future text commands.

Parameters:

none

Example Usage:

```
text: select_all
text: delete
```

See Also:

Associated text commands

text: select_begin/end

Command:

```
text: select_begin
<index1  index2 ... indexN>
text: select_end
```

Purpose:

Selects one or more text strings in the graphics window as the target for future `text:` commands

Parameters:

Type	Parameter	Description
int	`indexN`	The text annotation ID

Example Usage:

```
text: select_begin
 1 2
text: select_end
text: rgb 1 0 0
```

See Also:

Associated text commands

Command:

```
text: select_default
```

Purpose:

Causes any following text commands to affect the default settings, rather than a particular text object.

Parameters:

none

Notes:

All text commands following `text: select_default` will affect default settings until a new text object is selected using `text: select_begin/select_end`.

Example Usage:

```
text: select_default
text: visible OFF
text: rgb 1 0 0
text: select_begin
 1
text: select_end
```

See Also:

Associated text commands

text: size

Command:

```
text: size <size>
```

Purpose:

Specifies the size of the selected text in the graphics window.

Parameters:

Type	Parameter	Description
int	`size`	must be within the range of 0 to 100

Notes:

Text size is specified in arbitrary units, with 0 being the smallest possible text and 100 the largest.

The text scales with the size of the graphics window.

Example Usage:

```
text: select_begin
 1
text: select_end
text: size 25
```

See Also:

Associated text commands

Command:

```
text: visible <toggle>
```

Purpose:

Controls the visibility of the selected text in the graphics window.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	annotation is visible
		OFF	annotation is not visible

Notes:

Text is visible by default.

Example Usage:

```
text: select_begin
 1
text: select_end
text: visible ON
```

See Also:

Associated text commands

Commands in this Section:

tools

See Also:

User Manual - Section 6.5, Tools Menu Functions

How To Use the Cursor (Point) Tool

How To Use the Line Tool

How To Use the Plane Tool

How To Use the Cylinder Tool

How To Use the Sphere Tool

How To Use the Cone Tool

How To Use the Surface of Revolution Tool

tools: box

Command:

```
tools: box <option>
```

Purpose:

Turns the box tool on or off.

Parameters:

Type	Parameter	Description	
constant	`option`	`ON`	turn the box tool on
		`OFF`	turn the box tool off

Example Usage:

```
tools: box ON
```

See Also:

Associated tools commands

Command:

```
tools: cursor <toggle>
```

Purpose:

Turns the cursor tool on or off.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	turn the tool on
		`OFF`	turn the tool off

Example Usage:

```
tools: cursor ON
```

See Also:

Associated tools commands

tools: line

Command:

```
tools: line <toggle>
```

Purpose:

Turns the line tool on or off.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	turn the tool on
		`OFF`	turn the tool off

Example Usage:

```
tools: line ON
```

See Also:

Associated tools commands

Command:

```
tools: plane <option>
```

Purpose:

Turns the plane tool on or off.

Parameters:

Type	Parameter	Description	
constant	option	ON	turn the tool on using last representation
		OFF	turn the tool off
		LINE	turn the tool on and drawn with lines
		FILLED	turn the tool on and draw filled

Example Usage:

```
tools: plane ON
```

See Also:

Associated tools commands

tools: quadric

Command:

```
tools: quadric <option>
```

Purpose:

turns the quadric tool on or off.

Parameters:

Type	Parameter	Description	
constant	option	CYLINDER	turns the cylinder tool visible
		SPHERE	turn the sphere tool visible
		CONE	turn the cone tool visible
		REVOLUTION	turn the surface of revolution tool visible
		NONE	turn the quadric tool invisible

Example Usage:

```
tools: quadric SPHERE
```

See Also:

Associated tools commands

Commands in this Section:

varextcfd

See Also:

User Manual - Section 4.1, Variable Selection and Activation

User Manual - Section 4.3, Variable Creation

How To Create New Variables

varextcfd: density

Command:

```
varextcfd: density <var>
```

Purpose:

Set the density basis variable.

Parameters:

Type	Parameter	Description
string	`var`	name of the density variable

Notes:

For extended CFD variables to be computed, the basis variables must be identified and the basis constants defined.

Example Usage:

```
#
# Activating all variables in this case
#
variables: activate_all
#
# Making extended variables available
#
varextcfd: show_extended ON
#
# Setting the basis variables and constants
#
varextcfd: modify_begin
varextcfd: density Density
varextcfd: momentum Momentum
varextcfd: total_energy Energy
varextcfd: ratio_of_spec_heat 1.4000e+00
varextcfd: freestream_mach 1.0000e+00
varextcfd: gas_constant 1.0000e+00
varextcfd: freestream_density 1.0000e+00
varextcfd: freestream_speed_sound 1.0000e+00
varextcfd: modify_end
```

See Also:

Associated varextcfd commands

Command:

```
varextcfd: freestream_density <dens>
```

Purpose:

Set the freestream density constant.

Parameters:

Type	Parameter	Description
float	dens	desired value of the freestream density constant

Notes:

For extended CFD variables to be computed, the basis variables must be identified and the basis constants defined.

Example Usage:

```
#
# Activating all variables in this case
#
variables: activate_all
#
# Making extended variables available
#
varextcfd: show_extended ON
#
# Setting the basis variables and constants
#
varextcfd: modify_begin
varextcfd: density Density
varextcfd: momentum Momentum
varextcfd: total_energy Energy
varextcfd: ratio_of_spec_heat 1.4000e+00
varextcfd: freestream_mach 1.0000e+00
varextcfd: gas_constant 1.0000e+00
varextcfd: freestream_density 1.0000e+00
varextcfd: freestream_speed_sound 1.0000e+00
varextcfd: modify_end
```

See Also:

Associated varextcfd commands

varextcfd: freestream_mach

Command:

```
varextcfd: freestream_mach <num>
```

Purpose:

Set the freestream mach number constant.

Parameters:

Type	Parameter	Description
float	num	desired value of the freestream mach number constant

Notes:

For extended CFD variables to be computed, the basis variables must be identified and the basis constants defined.

Example Usage:

```
#
# Activating all variables in this case
#
variables: activate_all
#
# Making extended variables available
#
varextcfd: show_extended ON
#
# Setting the basis variables and constants
#
varextcfd: modify_begin
varextcfd: density Density
varextcfd: momentum Momentum
varextcfd: total_energy Energy
varextcfd: ratio_of_spec_heat 1.4000e+00
varextcfd: freestream_mach 1.0000e+00
varextcfd: gas_constant 1.0000e+00
varextcfd: freestream_density 1.0000e+00
varextcfd: freestream_speed_sound 1.0000e+00
varextcfd: modify_end
```

See Also:

Associated varextcfd commands

Command:

```
varextcfd: freestream_speed_sound <num>
```

Purpose:

Set the freestream speed of sound constant.

Parameters:

Type	Parameter	Description
float	num	desired value of the freestream speed of sound constant

Notes:

For extended CFD variables to be computed, the basis variables must be identified and the basis constants defined.

Example Usage:

```
#
# Activating all variables in this case
#
variables: activate_all
#
# Making extended variables available
#
varextcfd: show_extended ON
#
# Setting the basis variables and constants
#
varextcfd: modify_begin
varextcfd: density Density
varextcfd: momentum Momentum
varextcfd: total_energy Energy
varextcfd: ratio_of_spec_heat 1.4000e+00
varextcfd: freestream_mach 1.0000e+00
varextcfd: gas_constant 1.0000e+00
varextcfd: freestream_density 1.0000e+00
varextcfd: freestream_speed_sound 1.0000e+00
varextcfd: modify_end
```

See Also:

Associated varextcfd commands

varextcfd: gas_constant

Command:

```
varextcfd: gas_constant <num>
```

Purpose:

Set the gas constant.

Parameters:

Type	Parameter	Description
float	num	desired value of the gas constant

Notes:

For extended CFD variables to be computed, the basis variables must be identified and the basis constants defined.

Example Usage:

```
#
# Activating all variables in this case
#
variables: activate_all
#
# Making extended variables available
#
varextcfd: show_extended ON
#
# Setting the basis variables and constants
#
varextcfd: modify_begin
varextcfd: density Density
varextcfd: momentum Momentum
varextcfd: total_energy Energy
varextcfd: ratio_of_spec_heat 1.4000e+00
varextcfd: freestream_mach 1.0000e+00
varextcfd: gas_constant 1.0000e+00
varextcfd: freestream_density 1.0000e+00
varextcfd: freestream_speed_sound 1.0000e+00
varextcfd: modify_end
```

See Also:

Associated varextcfd commands

Command:

```
varextcfd: modify_begin
<varextcfd_command>
<varextcfd_command>
.
.
.
<varextcfd_command>
varextcfd: modify_end
```

Purpose:

Delimit the modifications to the extended CFD attributes.

Parameters:

Type	Parameter	Description
string	varextcfd_command	any of the varextcfd commands

Notes:

For extended CFD variables to be computed, the basis variables must be identified and the basis constants defined.

These are optional commands which increase performance since all changes are made at one time.

Example Usage:

```
#
# Activating all variables in this case
#
variables: activate_all
#
# Making extended variables available
#
varextcfd: show_extended ON
#
# Setting the basis variables and constants
#
varextcfd: modify_begin
varextcfd: density Density
varextcfd: momentum Momentum
varextcfd: total_energy Energy
varextcfd: ratio_of_spec_heat 1.4000e+00
varextcfd: freestream_mach 1.0000e+00
varextcfd: gas_constant 1.0000e+00
varextcfd: freestream_density 1.0000e+00
varextcfd: freestream_speed_sound 1.0000e+00
varextcfd: modify_end
```

See Also:

Associated varextcfd commands

varextcfd: momentum

Command:

```
varextcfd: momentum <var>
```

Purpose:

Set the momentum basis variable.

Parameters:

Type	Parameter	Description
string	`var`	name of the momentum variable

Notes:

For extended CFD variables to be computed, the basis variables must be identified and the basis constants defined. You must define either momentum or velocity for a basis, but not both.

Example Usage:

```
#
# Activating all variables in this case
#
variables: activate_all
#
# Making extended variables available
#
varextcfd: show_extended ON
#
# Setting the basis variables and constants
#
varextcfd: modify_begin
varextcfd: density Density
varextcfd: momentum Momentum
varextcfd: total_energy Energy
varextcfd: ratio_of_spec_heat 1.4000e+00
varextcfd: freestream_mach 1.0000e+00
varextcfd: gas_constant 1.0000e+00
varextcfd: freestream_density 1.0000e+00
varextcfd: freestream_speed_sound 1.0000e+00
varextcfd: modify_end
```

See Also:

Associated varextcfd commands

Command:

```
varextcfd: ratio_of_specific_heat <var or num>
```

Purpose:

Set the ratio of specific heat basis variable, or constant.

Parameters:

Type	Parameter	Description
string or float	`var` or `num`	name of the ratio of specific heat variable, or the value to use as a constant

Notes:

For extended CFD variables to be computed, the basis variables must be identified and the basis constants defined.

Example Usage:

```
#
# Activating all variables in this case
#
variables: activate_all
#
# Making extended variables available
#
varextcfd: show_extended ON
#
# Setting the basis variables and constants
#
varextcfd: modify_begin
varextcfd: density Density
varextcfd: momentum Momentum
varextcfd: total_energy Energy
varextcfd: ratio_of_spec_heat 1.4000e+00
varextcfd: freestream_mach 1.0000e+00
varextcfd: gas_constant 1.0000e+00
varextcfd: freestream_density 1.0000e+00
varextcfd: freestream_speed_sound 1.0000e+00
varextcfd: modify_end
```

See Also:

Associated varextcfd commands

varextcfd: show_extended

Command:

```
varextcfd: show_extended <toggle>
```

Purpose:

Make extended CFD variables available or not.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	to make extended CFD variables available
		`OFF`	to make them unavailable.

Notes:

For extended CFD variables to be computed, the basis variables must be identified and the basis constants defined. Once the basis variables and constants are defined, and this command makes them available - the variables can be computed when activated or used.

Example Usage:

```
#
# Activating all variables in this case
#
variables: activate_all
#
# Making extended variables available
#
varextcfd: show_extended ON
#
# Setting the basis variables and constants
#
varextcfd: modify_begin
varextcfd: density Density
varextcfd: momentum Momentum
varextcfd: total_energy Energy
varextcfd: ratio_of_spec_heat 1.4000e+00
varextcfd: freestream_mach 1.0000e+00
varextcfd: gas_constant 1.0000e+00
varextcfd: freestream_density 1.0000e+00
varextcfd: freestream_speed_sound 1.0000e+00
varextcfd: modify_end
```

See Also:

Associated varextcfd commands

Command:

```
varextcfd: total_energy <var>
```

Purpose:

Set the `total_energy` basis variable.

Parameters:

Type	Parameter	Description
string	`var`	name of the `total_energy` variable

Notes:

For extended CFD variables to be computed, the basis variables must be identified and the basis constants defined.

Example Usage:

```
#
# Activating all variables in this case
#
variables: activate_all
#
# Making extended variables available
#
varextcfd: show_extended ON
#
# Setting the basis variables and constants
#
varextcfd: modify_begin
varextcfd: density Density
varextcfd: momentum Momentum
varextcfd: total_energy Energy
varextcfd: ratio_of_spec_heat 1.4000e+00
varextcfd: freestream_mach 1.0000e+00
varextcfd: gas_constant 1.0000e+00
varextcfd: freestream_density 1.0000e+00
varextcfd: freestream_speed_sound 1.0000e+00
varextcfd: modify_end
```

See Also:

Associated varextcfd commands

varextcfd: velocity

Command:

```
varextcfd: velocity <var>
```

Purpose:

Set the velocity basis variable.

Parameters:

Type	Parameter	Description
string	`var`	name of the velocity variable

Notes:

For extended CFD variables to be computed, the basis variables must be identified and the basis constants defined. You must define either momentum or velocity for a basis, but not both.

Example Usage:

```
#
# Activating all variables in this case
#
variables: activate_all
#
# Making extended variables available
#
varextcfd: show_extended ON
#
# Setting the basis variables and constants
#
varextcfd: modify_begin
varextcfd: density Density
varextcfd: velocity Velocity
varextcfd: total_energy Energy
varextcfd: ratio_of_spec_heat 1.4000e+00
varextcfd: freestream_mach 1.0000e+00
varextcfd: gas_constant 1.0000e+00
varextcfd: freestream_density 1.0000e+00
varextcfd: freestream_speed_sound 1.0000e+00
varextcfd: modify_end
```

See Also:

Associated varextcfd commands

Commands in this Section:

variables

See Also:

User Manual - Section 4.1, Variable Selection and Activation

How To Create New Variables

How To Activate Variables

variables: activate

Command:

```
variables: activate <name>
```

Purpose:

Activates a variable.

Parameters:

Type	Parameter	Description
string	`name`	name of the variable to activate

Notes:

A variable must be activated before it can be used for any purpose.

Example Usage:

```
#
# Activate a variable and use it
# to color a part
#
variables: activate temperature
part: select_begin
1
part: select_end
part: colorby_palette temperature
```

See Also:

Associated variables commands

Command:

```
variables: activate_all
```

Purpose:

Activates all variables.

Parameters:

none

Example Usage:

```
#
# Activating all variables
#
variables: activate_all
```

See Also:

Associated variables commands

variables: deactivate

Command:

```
variables: deactivate <name>
```

Purpose:

Deactivates a variable.

Parameters:

Type	Parameter	Description
string	`name`	name of the variable to deactivate

Notes:

A deactivated variable is not available for use, but can be activated again if needed.

If a computed variable is deactivated it is deleted and must be re-specified if needed at a later time.

Example Usage:

```
variables: deactivate temperature
```

See Also:

Associated variables commands

Command:

```
variables: evaluate <expression>
```

Purpose:

Creates a computed variable.

Parameters:

Type	Parameter	Description
string	`expression`	expression needed to create a computed variable

Notes:

The expression must conform to one of the many supplied general functions or proper math or calculator operations within EnSight. It is subject to all the limitations of such, so please familiarize yourself with these.

Example Usage:

```
part: select_all
variables: activate_all
variables: evaluate Area = Area(plist)
variables: evaluate Grad = Grad(plist,temperature)
variables: evaluate CmplxTransResp = CmplxTransResp(plist,c_scalar,90.0)
```

See Also:

Associated variables commands

variables: modify_parameter

Command:

```
variables: modify_parameter <varname> param <num>=<string or value>
```

Purpose:

Modifies a parameter of a general function computed variable.

Parameters:

Type	Parameter	Description
string	`varname`	computed variable's name
int	`num`	parameter number for the computed variable
string, float or int	`string or value`	proper modified string, or number for the parameter specified

Notes:

Currently only used for modifying the phase angle of a complex transient response variable.

Example Usage:

```
variables: activate_all
part: select_all
variables: evaluate CmplxTransResp = CmplxTransResp(plist,c_scalar,90.0)
function: palette CmplxTransResp
variables: modify_parameter CmplxTransResp param 3=1.1700e+02
```

See Also:

Associated variables commands

Command:

```
variables: notify_before_activate <toggle>
```

Purpose:

A preference that can be set concerning notification when a variable will activated. Since EnSight by default activates variables automatically if needed, this command can cause a confirmation to be requested before a variable is activated.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	user will be asked to confirm a variable activation
		`OFF`	automatic variable activation occurs without any confirmation request (the default)

Example Usage:

```
variables: notify_before_activate ON
```

See Also:

Associated variables commands

variables: save_all_constants

Command:

```
variables: save_all_constants <filename>
```

Purpose:

Saves all active constant variable values to a file.

Parameters:

Type	Parameter	Description
string	`filename`	name of the file in which to save the constants

Example Usage:

```
#
# Activating all variables
#
variables: activate_all
#
# Computing area of one part
#
part: select_begin
1
part: select_end
variables: evaluate Area = Area(plist)
#
# Computing area of another part
#
part: select_begin
2
part: select_end
variables: evaluate Area2 = Area(plist)
variables: save_all_constants /usr/tmp/all_constants
```

See Also:

Associated variables commands

Command:

```
variables: save_constants <filename>
```

Purpose:

Saves selected constant variable values to a file.

Parameters:

Type	Parameter	Description
string	`filename`	name of the file in which to save the constants

Example Usage:

```
#
# Activating all variables
#
variables: activate_all
#
# Computing area of one part
#
part: select_begin
1
part: select_end
variables: evaluate Area = Area(plist)
#
# Computing area of another part
#
part: select_begin
2
part: select_end
variables: evaluate Area2 = Area(plist)
#
# Selecting the computed area constant to write them out
#
variables: select_varname_begin
Area
Area2
variables: select_varname_end
#
# Saving the areas to a file
#
variables: save_constants /usr/tmp/my_constants
```

See Also:

Associated variables commands

variables: select_varname_begin/end

Command:

```
variables: select_varname_begin
<varname>
<varname>
.
.
.
<varname>
variables: select_varname_end
```

Purpose:

Selects variables for future use.

Parameters:

Type	Parameter	Description
string	varname	valid name of a variable

Example Usage:

```
#
# Activating all variables
#
variables: activate_all
#
# Computing area of one part
#
part: select_begin
1
part: select_end
variables: evaluate Area = Area(plist)
#
# Computing area of another part
#
part: select_begin
2
part: select_end
variables: evaluate Area2 = Area(plist)
#
# Selecting the computed area constant to write them out
#
variables: select_varname_begin
Area
Area2
variables: select_varname_end
#
# Saving the areas to a file
#
variables: save_constants /usr/tmp/my_constants
```

See Also:

Associated variables commands

Commands in this Section:

vctarrow

vctarrow: arrow_location <location>
vctarrow: arrow_type <type>
vctarrow: begin
vctarrow: end
vctarrow: component <x_scale> <y_scale> <z_scale>
vctarrow: create
vctarrow: density <value>
vctarrow: display_offset <offset>
vctarrow: projection <proj>
vctarrow: scale_factor <scale/time>
vctarrow: select_default
vctarrow: threshold_low <low>
vctarrow: threshold_high <high>
vctarrow: thresholds <type>
vctarrow: tip_rgb <red_val> <grn_val> <blu_val>
vctarrow: tip_shape <type>
vctarrow: tip_size <size>
vctarrow: tip_size_rep <rep>
vctarrow: variable <var_name>

See Also:

User Manual - Section 7.6, Vector Arrow Create/Update

How To Create Vector Arrows

vctarrow: arrow_location

Command:

```
vctarrow: arrow_location <location>
```

Purpose:

Control the location of the arrows

Parameters:

Type	Parameter	Description	
constant	location	vertices	draw vectors at vertices (corner nodes)
		node	draw vectors at all nodes
		element_center	draw vectors at element centers

Notes:

Discrete particles can only use the 'node' vector location.

For linear elements 'node' and 'vertices' give the same result.

Example Usage:

```
part: select_begin
2
part: select_end
part: modify_begin
vctarrow: arrow_location element_center
part: modify_end
```

See Also:

Associated vctarrow commands

Command:

```
vctarrow: arrow_type <type>
```

Purpose:

Change the visual representation of the arrow shafts

Parameters:

Type	Parameter	Description	
constant	type	rectilinear	straight arrows with the length proportional to the vector magnitude
		rect_fixed	straight arrows of fixed length
		curved	arrow shaft curves with the vector

Example Usage:

```
part: select_begin
2
part: select_end
part: modify_begin
vctarrow: arrow_type curved
part: modify_end
```

See Also:

Associated vctarrow commands

vctarrow: begin/end

Command:

```
vctarrow: begin
vctarrow: end
```

Purpose:

Delimit the modifications for vector arrow attributes

Parameters:

none

Notes:

These commands are used to set the attributes for a vector arrow part creation.

Example Usage:

```
vctarrow: begin
vctarrow: variable VITESSE
vctarrow: end
vctarrow: create
```

See Also:

Associated vctarrow commands

Command:

```
vctarrow: component <x_scale> <y_scale> <z_scale>
```

Purpose:

Specify a scaling factor for each component of the vector variable, to be used in calculating both the magnitude and direction of the vector arrow.

Parameters:

Type	Parameter	Description
float	x_scale	scaling for the x-component
	y_scale	scaling for the y-component
	z_scale	scaling for the z-component

Notes:

Specify 1 to use the full value of a component. Specify 0 to ignore the corresponding vector components (and thus confine all vector arrows to planes perpendicular to that axis).

Values may be negative to reverse the direction of the component.

Example Usage:

```
vctarrow: begin
vctarrow: variable VITESSE
vctarrow: end
vctarrow: create
part: select_begin
2
part: select_end
part: modify_begin
vctarrow: component 2.0000e+00 1.0000e+00 1.0000e+00
part: modify_end
```

See Also:

Associated vctarrow commands

vctarrow: create

Command:

```
vctarrow: create
```

Purpose:

Create a new vector arrow part using the currently defined attributes and selected parent parts.

Parameters:

none

Notes:

The vector variable must be set via the `vctarrow: variable` command before the part can be created.

Example Usage:

```
variables: activate VITESSE
part: select_all
vctarrow: begin
vctarrow: variable VITESSE
vctarrow: end
vctarrow: create
```

See Also:

Associated vctarrow commands

Command:

```
vctarrow: density <value>
```

Purpose:

Control the fraction of the parent's nodes which will display a vector.

Parameters:

Type	Parameter	Description
float	value	density value between 0 and 1

Notes:

A value of 0.0 results in no arrows, while a value of 1.0 results in an arrow at every node. Between these two values the arrows are randomly distributed at the specified density.

Example Usage:

```
vctarrow: begin
vctarrow: variable VITESSE
vctarrow: end
vctarrow: create
part: select_begin
2
part: select_end
part: modify_begin
vctarrow: density 5.0000e-01
part: modify_end
```

See Also:

Associated vctarrow commands

vctarrow: display_offset

Command:

```
vctarrow: display_offset <offset>
```

Purpose:

Specify a visual offset for the vectors from the surface.

Parameters:

Type	Parameter	Description
float	`offset`	offset in world coordinates

Notes:

A positive offset moves the vector away from the surface in the direction of the normal.

Example Usage:

```
vctarrow: begin
vctarrow: variable VITESSE
vctarrow: end
vctarrow: create
part: select_begin
2
part: select_end
part: modify_begin
vctarrow: display_offset 1.0000e-01
part: modify_end
```

See Also:

Associated vctarrow commands

Command:

```
vctarrow: projection <proj>
```

Purpose:

Apply a projection to the vectors prior to display

Parameters:

Type	Parameter	Description	
constant	proj	all	create the vector arrows using all components of the specified vector
		normal	create the vector arrows using the components normal to the surface
		tangential	create the vector arrows using the components tangential to the surface
		component	create two arrows for each target location - one normal and one tangential

Notes:

Vector components are scaled prior to computing the projections.

Example Usage:

```
variables: activate velocity
vctarrow: begin
vctarrow: variable velocity
vctarrow: end
vctarrow: create
part: select_begin
4
part: select_end
part: modify_begin
vctarrow: projection component
part: modify_end
```

See Also:

Associated vctarrow commands

vctarrow: scale_factor

Command:

```
vctarrow: scale_factor <scale/time>
```

Purpose:

Control the length of the vector arrows

Parameters:

Type	Parameter	Description
float	`scale/time`	scale factor or time

Notes:

If the arrow type is "`Rectilinear`" this parameter is a scaling multiple to change the magnitude of the vectors. If the type is "`Rect_fixed`" the parameter specifies the absolute magnitude for each vector. If the type is "`Curved`" this parameter controls the integration time for the streamlines which make up the curved part of the arrow.

Example Usage:

```
vctarrow: begin
vctarrow: variable VITESSE
vctarrow: end
vctarrow: create
part: select_begin
2
part: select_end
part: modify_begin
vctarrow: scale_factor 7.0000e-01
part: modify_end
```

See Also:

Associated vctarrow commands

Command:

```
vctarrow: select_default
```

Purpose:

Sets the target to the default settings for subsequent attribute modifications.

Parameters:

none

Example Usage:

```
vctarrow: select_default
part: modify_begin
vctarrow: variable VITESSE
part: modify_end
```

See Also:

Associated vctarrow commands

vctarrow: threshold_low/high

Command:

```
vctarrow: threshold_low <low>
vctarrow: threshold_high <high>
```

Purpose:

Set the low or high threshold value

Parameters:

Type	Parameter	Description
float	`low`	low filter threshold
	`high`	high filter threshold

Notes:

Used with `vctarrow: thresholds`

Example Usage:

```
vctarrow: begin
vctarrow: variable VITESSE
vctarrow: end
vctarrow: create
part: select_begin
2
part: select_end
part: modify_begin
vctarrow: thresholds band
part: modify_end
part: modify_begin
vctarrow: threshold_low 5.0000e-1
part: modify_end
part: modify_begin
vctarrow: threshold_high 7.5000e-1
part: modify_end
```

See Also:

Associated vctarrow commands

Command:

```
vctarrow: thresholds <type>
```

Purpose:

Apply a threshold filter to the selection of vectors for display.

Parameters:

Type	Parameter	Description	
constant	type	none	display all vectors
		low	display vectors above the low threshold
		band	display vectors below the low and above the high threshold
		high	display vectors below the high threshold
		low_high	display vectors between low and high threshold

Notes:

Threshold filtering occurs after density filtering. When using both forms of filtering in combination the actual vector density may not be exactly as specified.

Example Usage:

```
vctarrow: begin
vctarrow: variable VITESSE
vctarrow: end
vctarrow: create
part: select_begin
2
part: select_end
part: modify_begin
vctarrow: thresholds band
part: modify_end
part: modify_begin
vctarrow: threshold_low 5.0000e-1
part: modify_end
part: modify_begin
vctarrow: threshold_high 7.5000e-1
part: modify_end
```

See Also:

Associated vctarrow commands

vctarrow: tip_rgb

Command:

```
vctarrow: tip_rgb <red_val> <grn_val> <blu_val>
```

Purpose:

Change the color of the arrow tips

Parameters:

Type	Parameter	Description
float	red_val	red, green, blue components of the color tip (range 0.0 to 1.0)
	grn_val	
	blu_val	

Notes:

The color of the tip is only used when the tip shape is 'tipped'

Example Usage:

```
vctarrow: begin
vctarrow: variable VITESSE
vctarrow: end
vctarrow: create
part: select_begin
2
part: select_end
part: modify_begin
vctarrow: tip_shape tipped
part: modify_end
part: modify_begin
vctarrow: tip_rgb 0.0000e00 1.0000e00 0.0000e00
part: modify_end
```

See Also:

Associated vctarrow commands

Command:

```
vctarrow: tip_shape <type>
```

Purpose:

Change the visual representation of the arrow tips

Parameters:

Type	Parameter	Description	
constant	type	none	no arrow tips
		normal	arrow tip displayed as two short lines
		triangle	arrow tip displayed as two orthogonal triangles
		tipped	arrow tip displayed in a different color

Example Usage:

```
vctarrow: begin
vctarrow: variable VITESSE
vctarrow: end
vctarrow: create
part: select_begin
2
part: select_end
part: modify_begin
vctarrow: tip_shape triangle
part: modify_end
```

See Also:

Associated vctarrow commands

vctarrow: tip_size

Command:

```
vctarrow: tip_size <size>
```

Purpose:

Change the size of the arrow tips

Parameters:

Type	Parameter	Description
float	`size`	size of the arrow tips

Notes:

This parameter is only used for '`triangle`' or '`normal`' arrow tips, and only when the tip size is '`fixed`'.

Example Usage:

```
vctarrow: begin
vctarrow: variable VITESSE
vctarrow: end
vctarrow: create
part: select_begin
2
part: select_end
part: modify_begin
vctarrow: tip_shape triangle
part: modify_end
part: modify_begin
vctarrow: tip_size_rep fixed
part: modify_end
part: modify_begin
vctarrow: tip_size 2.0000e-1
part: modify_end
```

See Also:

Associated vctarrow commands

Command:

```
vctarrow: tip_size_rep <rep>
```

Purpose:

Set the method for determining tip size

Parameters:

Type	Parameter	Description	
constant	rep	fixed	length is specified by the vector tip size attribute
		proportional	length is proportional to the magnitude of the vector

Notes:

This parameter is only used for 'triangle' or 'normal' arrow tips.

Example Usage:

```
vctarrow: begin
vctarrow: variable VITESSE
vctarrow: end
vctarrow: create
part: select_begin
2
part: select_end
part: modify_begin
vctarrow: tip_shape triangle
part: modify_end
part: modify_begin
vctarrow: tip_size_rep fixed
part: modify_end
part: modify_begin
vctarrow: tip_size 2.0000e-1
part: modify_end
```

See Also:

Associated vctarrow commands

vctarrow: variable

Command:

```
vctarrow: variable <var_name>
```

Purpose:

Specify the vector variable to be used for vector arrow creation.

Parameters:

Type	Parameter	Description
string	var_name	name of an active vector variable

Notes:

The variable must be activated before it can be used in a vector arrow plot.

Example Usage:

```
variables: activate VITESSE
part: select_all
vctarrow: begin
vctarrow: variable VITESSE
vctarrow: end
vctarrow: create
```

See Also:

Associated vctarrow commands

Commands in this Section:

view

view: bounds <toggle>
view: clipping <toggle>
view: default_orientation <direction>
view: fast_display <toggle>
view: full_screen <toggle>
view: hardware_line_offset <toggle>
view: hidden_line <toggle> [<red_val> <grn_val> <blu_val>]
view: hidden_surface <toggle>
view: perspective <toggle>
view: static_lighting <toggle>
view: pick_mode <value>
view: stereo <toggle>

See Also:

User Manual - Section 8.6, View Mode

How To Set Global Viewing Parameters

How To Set Drawing Mode (Line, Surface, Hidden Line)

How To Enable Stereo Viewing

view: bounds

Command:

```
view: bounds <toggle>
```

Purpose:

Toggles on or off the global bounds display

Parameters:

Type	Parameter	Description	
constant	toggle	ON	turns on bounds display
		OFF	turns off bounds display

Notes:

Bounds display will be visible in a viewport if this toggle is ON and the "viewport_bounds: visible ON" is also set.

Example Usage:

```
#
# turn on bounds display but turn the bounds off for viewport 1
#
view: bounds ON
viewport: select_begin
 1
viewport: select_end
viewport_bounds: visible OFF
```

See Also:

Associated view commands

Command:

```
view: clipping <toggle>
```

Purpose:

Toggles on or off the auxiliary clipping plane

Parameters:

Type	Parameter	Description	
constant	toggle	ON	turns on auxiliary clipping
		OFF	turns off auxiliary clipping

Notes:

When auxiliary clipping is on, objects on the negative side of the plane tool are not drawn. By toggling on or off parts' individual auxiliary clipping attribute (`part: clip`), some parts can be clipped while others are not.

Example Usage:

```
tools: plane ON
view: clipping ON
part: select_partname_begin
"(CASE:Case 1)ami-x hypersonic body"
part: select_partname_end
part: modify_begin
part: clip OFF
part: modify_end
view_transf: function plane
view_transf: rotate 106.516396 0.000000 0.000000
view_transf: function global
```

See Also:

Associated view commands

view: default_orientation

Command:

```
view: default_orientation <direction>
```

Purpose:

Sets the default view orientation.

Parameters:

Type	Parameter	Description	
constant	direction	+x	The viewer is located down the positive x axis looking toward the origin
		-x	The viewer is located down the negative x axis looking toward the origin
		+y	The viewer is located down the positive y axis looking toward the origin
		-y	The viewer is located down the negative y axis looking toward the origin
		+z	The viewer is located down the positive z axis looking toward the origin
		-z	The viewer is located down the negative z axis looking toward the origin

Notes:

Must be executed before data is read.

The default is +z

Example Usage:

```
view: default_orientation +x
```

See Also:

Associated view commands

Command:

```
view: fast_display <toggle>
```

Purpose:

Toggles the fast display setting. When on, parts can be rendered using an alternative representation while being interactively moved.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	turns on fast display
		OFF	turns off fast display

Notes:

Each part has an attribute that indicates how it will be drawn when fast display is on. See 'part: bounding_rep'.

Example Usage:

```
view: fast_display ON
part: select_partname_begin
"(CASE:Case 1)external flow field"
part: select_partname_end
part: modify_begin
part: bounding_rep points
part: modify_end
view_transf: rotate -3.021584 51.034481 0.000000
view_transf: function global
```

See Also:

Associated view commands

view: full_screen

Command:

```
view: full_screen <toggle>
```

Purpose:

Toggles full screen rendering

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	turns on full screen rendering
		`OFF`	turns off full screen rendering

Notes:

During full screen rendering, the graphics window fills the screen hiding the rest of the user interface and other windows.

Example Usage:

```
view: full_screen ON
```

See Also:

Associated view commands

Command:

```
view: hardware_line_offset <toggle>
```

Purpose:

Toggles on or off the use of hardware line offset

Parameters:

Type	Parameter	Description	
constant	toggle	ON	turns on harware line offset
		OFF	turns off hardware line offset

Notes:

This command is only used for view preferences.

Example Usage:

```
view: hardware_line_offset ON
view: default_orientation +Z
view: hidden_surface OFF
```

See Also:

Associated view commands

view: hidden_line

Command:

```
view: hidden_line <toggle> [<red_val> <grn_val> <blu_val>]
```

Purpose:

Toggles global hidden line drawing.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	turns on hidden line drawing
		OFF	turns off hidden line drawing
float	red_val	optional red, green, blue color values (range 0.0 to 1.0) for the hidden lines when hidden surface (See view: hidden_surface) is also ON	
	grn_val		
	blu_val		

Notes:

Each part has a part attribute that determines whether or not it will be drawn with hidden lines if the global setting is on (See part: hidden_line).

If the optional RGB values are specified, they must be values between 0.0 and 1.0. If none are specified, then 0.0, 0.0, 0.0 are used.

Example Usage:

```
VERSION 7.41
view: hidden_line ON 1.0000e+00 0.0000e+00 0.0000e+00
part: modify_begin
part: hidden_line OFF
part: modify_end
```

See Also:

Associated view commands

Command:

```
view: hidden_surface <toggle>
```

Purpose:

Toggles display of shaded surfaces.

Parameters:

Type	Parameter	Description	
constant	toggle	ON	turns on shaded surfaces
		OFF	turns off shaded surfaces (line drawing)

Notes:

Each part has an attribute that determines whether it will be drawn in shaded or in line mode (`part: hidden_surface`).

Example Usage:

```
view: hidden_surface ON
part: select_partname_begin
"(CASE:Case 1)ami-x hypersonic body"
part: select_partname_end
part: modify_begin
part: hidden_surface OFF
part: modify_end
```

See Also:

Associated view commands

view: perspective

Command:

```
view: perspective <toggle>
```

Purpose:

Toggles global perspective rendering.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	turns on perspective rendering
		`OFF`	turns off perspective rendering

Notes:

Perspective rendering causes things further in distance to appear smaller than closer objects. If this setting is off then objects will be rendered in orthographic mode which does not use perspective foreshortening.

Each viewport has a local setting. See `viewport: perspective`

Example Usage:

```
view: perspective ON
```

See Also:

Associated view commands

Command:

```
view: pick_mode <value>
```

Purpose:

Sets what type of pick will be performed when the 'p' key is pressed.

Parameters:

Type	Parameter	Description	
constant	value	center_of_transform	pick a point to set the center of transformation
		cursor	pick a point to specify the cursor tool location
		line	pick two points to specify the line tool location
		look_at	set the look at point
		part	select the part under the mouse
		plane	pick three points to specify the plane tool location
		plane_2pt	draw a line to position the plane tool
		plane_normal	pick the plane tool z-direction
		plane_origin	pick the plane tool origin

Notes:

See the EnSight User's Manual for instructions on how to operate each of these picks.

Example Usage:

```
view: pick_mode part
view: pick_mode cursor
view_transf: cursor -9.941673 -39.746929 3.714395
```

See Also:

Associated view commands

view: static_lighting

Command:

```
view: static_lighting <toggle>
```

Purpose:

Toggles the static lighting model.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	turns on static lighting
		`OFF`	turns off static lighting

Notes:

If static lighting is off (the default), then the light does not move as the scene moves. If it is on, then the light will move as the scene moves. The light direction for static lighting is set when it is toggled on.

Example Usage:

```
view: static_lighting ON
```

See Also:

Associated view commands

Command:

```
view: stereo <toggle>
```

Purpose:

Toggles stereo rendering.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	turns on stereo rendering
		`OFF`	turns off stereo rendering

Notes:

Stereo rendering will only occur if the graphics hardware supports it.

Example Usage:

```
view: stereo ON
test: larger stereo separation
```

See Also:

Associated view commands

Commands in this Section:

view_transf

view_transf: zclip_back <z_loc>
view_transf: zclip_float <toggle>
view_transf: zclip_front <z_loc>
view_transf: zclip_min <z_min>
view_transf: zoom <factor>
view_transf: zoom_style <style>

See Also:

User Manual - Chapter 9, Transformation Control

How To Rotate, Zoom, Translate, Scale

How To Set Z Clipping

How To Set LookFrom / LookAt

How To Set Auxiliary Clipping

How To Define and Change Viewports

How To Save and Restore Viewing Parameters

How To Reset Tools and Viewports

Command:

```
view_transf: action <method>
```

Purpose:

Sets the transformation method for interaction in the user interface.

Parameters:

Type	Parameter	Description	
constant	method	rotate	sets transformation mode to rotate
		translate	sets transformation mode to translate
		scale	sets transformation mode to scale
		zoom	sets transformation mode to zoom
		band_zoom	initialize for a rubber band zoom

Notes:

This is not a necessary command. It is used to change the user interface to reflect the transformation currently attached to the mouse button tied to the transformation action (by default the left mouse button).

Example Usage:

```
view_transf: action translate
```

See Also:

Associated view_transf commands

view_transf: axis

Command:

```
view_transf: axis <which>
```

Purpose:

Sets the axis to use for rotate, translate, and scale transformations.

Parameters:

Type	Parameter	Description	
constant	which	x	action will take place about or in the direction of the x axis
		y	action will take place about or in the direction of the y axis
		z	action will take place about or in the direction of the z axis
		all	action will take place about or in the direction of the all three (or the combined) axes

Notes:

Used in the transformation dialog to control the axis about which transformational values are applied.

The axis is either the global system or, if in Frame mode, the selected frame's axis system.

Example Usage:

```
#
# Set the axis to x
# and rotate about the x axis 15 degrees
#
view_transf: axis x
view_transf: rotate 15.000000 0.000000 0.000000
#
# Set the axis to be y
# and translate one unit in the -negative y direction
#
view_transf: axis y
view_transf: translate 0.0000e+00 -1.0000e+00 0.0000e+00
```

See Also:

Associated view_transf commands

Command:

```
view_transf: box_axis <axis> <x_comp> <y_comp> <z_comp>
```

Purpose:

Specify the direction vectors for a box tool.

Param eters:

Type	Parameter	Description	
constant	axis	x	specifies which axis is being defined
		y	
		z	
float	x_comp	The x component of the direction vector	
	y_comp	The y component of the direction vector	
	z_comp	The z component of the direction vector	

Notes:

The direction vectors are in reference to the selected coordinate frame.

Example Usage:

```
tools: box ON
view_transf: function box
view_transf: box_length 3.000000e+00 4.000000e+00 5.000000e+00
view_transf: box_origin 0.000000e+00 1.000000e+00 0.000000e+00
view_transf: box_axis x 1.000000e+00 0.000000e+00 0.000000e+00
view_transf: box_axis y 0.000000e+00 1.000000e+00 0.000000e+00
view_transf: box_axis z 0.000000e+00 0.000000e+00 1.000000e+00
```

See Also:

Associated view_transf commands

view_transf: box_length

Command:

```
view_transf: box_length <x_len> <y_len> <z_len>
```

Purpose:

Specify the size of the box tool in the x, y, and z directions.

Parameters:

Type	Parameter	Description
float	x_len	The length of box tool in the x, y, z direction
	y_len	
	z_len	

Notes:

The lengths are in relation to the box axis directions.

Example Usage:

```
tools: box ON
view_transf: function box
view_transf: box_length 3.000000e+00 4.000000e+00 5.000000e+00
view_transf: box_origin 0.000000e+00 1.000000e+00 0.000000e+00
view_transf: box_axis x 1.000000e+00 0.000000e+00 0.000000e+00
view_transf: box_axis y 0.000000e+00 1.000000e+00 0.000000e+00
view_transf: box_axis z 0.000000e+00 0.000000e+00 1.000000e+00
```

See Also:

Associated view_transf commands

Command:

```
view_transf: box_origin <x_ori> <y_ori> <z_ori>
```

Purpose:

Specify the origin of the box tool in reference to the selected coordinate frame.

Parameters:

Type	Parameter	Description
float	x_ori	The x, y, z origin of the box tool
	y_ori	
	z_ori	

Example Usage:

```
tools: box ON
view_transf: function box
view_transf: box_length 3.000000e+00 4.000000e+00 5.000000e+00
view_transf: box_origin 0.000000e+00 1.000000e+00 0.000000e+00
view_transf: box_axis x 1.000000e+00 0.000000e+00 0.000000e+00
view_transf: box_axis y 0.000000e+00 1.000000e+00 0.000000e+00
view_transf: box_axis z 0.000000e+00 0.000000e+00 1.000000e+00
```

See Also:

Associated view_transf commands

view_transf: center_of_transform

Command:

```
view_transf: center_of_transform <x_coord> <y_coord> <z_coord>
```

Purpose:

Set the point in space that will be the origin for transformations.

Param eters:

Type	Parameter	Description
float	x_coord	model x coordinate of the desired transformation origin
	y_coord	model y coordinate of the desired transformation origin
	z_coord	model z coordinate of the desired transformation origin

Notes:

Since transformations can be different per viewport, this can be set for each viewport, thus a viewport selection is needed if more than one viewport is being used.

Cannot be set during keyframe animation.

Example Usage:

```
#
# Selecting two viewports
#
viewport: select_begin
2 3
viewport: select_end
#
# Setting the center of transformations to be
# at (.5 .6 1) for each of these viewports
#
view_transf: center_of_transform 5.0000e-01 6.0000e-01 1.0000e+00
```

See Also:

Associated view_transf commands

Command:

```
view_transf: cone_angle <angle>
```

Purpose:

Set the angle for the cone tool.

Parameters:

Type	Parameter	Description
float	`angle`	the cone angle (in degrees)

Notes:

Use with `view_transf: cone_axis` and `view_transf: cone_origin` to specify the cone tool location and orientation

Example Usage:

```
#
# Turns the display of the quadric cone on
# (only one quadric tool at a time can be on)
#
tools: quadric cone
#
# Sets the angle and other attributes of the cone tool
#
view_transf: cone_angle 25.000000
view_transf: cone_axis 1.000000 0.100000 0.000000
view_transf: cone_origin -0.750000 0.700000 1.000000
#
# Turns off the display of the quadric tool
#
tools: quadric none
```

See Also:

Associated view_transf commands

view_transf: cone_axis

Command:

```
view_transf: cone_axis <x_comp> <y_comp> <z_comp>
```

Purpose:

Sets the axis for the cone tool.

Parameters:

Type	Parameter	Description
float	x_comp	x, y, z directional components for the cone tool axis
	y_comp	
	z_comp	

Notes:

Use with `view_transf: cone_angle` and `view_transf: cone_origin` to specify the cone tool location and orientation.

The components are in reference to the selected frame.

Example Usage:

```
#
# Turns display of the quadric cone on
# (only one quadric tool at a time can be on)
#
tools: quadric cone
#
# Sets the axis and other attributes of the cone tool
#
view_transf: cone_angle 25.000000
view_transf: cone_axis 1.000000 0.100000 0.000000
view_transf: cone_origin -0.750000 0.700000 1.000000
#
# Turns off the display of the quadric tool
#
tools: quadric none
```

See Also:

Associated view_transf commands

Command:

```
view_transf: cone_origin <x_coord> <y_coord> <z_coord>
```

Purpose:

Sets the origin point for the tip of the cone tool.

Parameters:

Type	Parameter	Description
float	x_coord	x, y, z coordinates of the cone tool origin
	y_coord	
	z_coord	

Notes:

Use with `view_transf: cone_angle` and `view_transf: cone_axis` to specify the cone tool location and orientation

The components are in reference to the selected frame.

Example Usage:

```
#
# Turns display of the quadric cone on
# (only one quadric tool at a time can be on)
#
tools: quadric cone
#
# Sets the origin and other attributes of the cone tool
#
view_transf: cone_angle 25.000000
view_transf: cone_axis 1.000000 0.100000 0.000000
view_transf: cone_origin -0.750000 0.700000 1.000000
#
# Turns off the display of the quadric tool
#
tools: quadric none
```

See Also:

Associated view_transf commands

view_transf: cursor

Command:

```
view_transf: cursor <x_coord> <y_coord> <z_coord>
```

Purpose:

Sets the location of the cursor tool.

Parameters:

Type	Parameter	Description
float	x_coord	x, y, z coordinates of the cursor
	y_coord	
	z_coord	

Notes:

The components are in reference to the selected frame.

Example Usage:

```
#
# Sets the cursor to the point (.79563 1 .8554)
#
view_transf: cursor 0.795630 1.000000 0.855400
```

See Also:

Associated view_transf commands

Command:

```
view_transf: cylinder_axis <x_comp> <y_comp> <z_comp>
```

Purpose:

Sets the axis for the cylinder tool.

Parameters:

Type	Parameter	Description
float	x_comp	x, y, z directional components for the cylinder tool axis
	y_comp	
	z_comp	

Notes:

Use with `view_transf: cylinder_origin` and `view_transf: cylinder_radius` to fully define the cylinder tool

The components are in reference to the selected frame.

Example Usage:

```
#
# Turns display of the quadric cylinder on
# (only one quadric tool at a time can be on)
#
tools: quadric cylinder
#
# Sets the axis and other attributes of the cylinder tool
#
view_transf: cylinder_axis 1.000000 0.100000 0.000000
view_transf: cylinder_origin -0.750000 0.700000 1.000000
view_transf: cylinder_radius 1.500000
#
# Turns off the display of the quadric tool
#
tools: quadric none
```

See Also:

Associated view_transf commands

view_transf: cylinder_origin

Command:

```
view_transf: cylinder_origin <x_coord> <y_coord> <z_coord>
```

Purpose:

Sets the origin point for one end of the cylinder tool.

Parameters:

Type	Parameter	Description
float	x_coord	x, y, z coordinates of the cylinder tool origin
	y_coord	
	z_coord	

Notes:

Use with `view_transf: cylinder_axis` and `view_transf: cylinder_radius` to fully define the cylinder tool.

The components are in reference to the selected frame.

Example Usage:

```
#
# Turns display of the quadric cylinder on
# (only one quadric tool at a time can be on)
#
tools: quadric cylinder
#
# Sets the origin and other attributes of the cylinder tool
#
view_transf: cylinder_axis 1.000000 0.100000 0.000000
view_transf: cylinder_origin -0.750000 0.700000 1.000000
view_transf: cylinder_radius 1.500000
#
# Turns off the display of the quadric tool
#
tools: quadric none
```

See Also:

Associated view_transf commands

Command:

```
view_transf: cylinder_radius <rad>
```

Purpose:

Set the radius value for the cylinder tool.

Parameters:

Type	Parameter	Description
float	rad	cylinder radius value (in model coordinates)

Notes:

Use with `view_transf: cylinder_origin` and `view_transf: cylinder_axis` to fully define the cylinder tool.

Example Usage:

```
#
# Turns the display of the quadric cylinder on
# (only one quadric tool at a time can be on)
#
tools: quadric cylinder
#
# Sets the radius and other attributes of the cylinder tool
#
view_transf: cylinder_axis 1.000000 0.100000 0.000000
view_transf: cylinder_origin -0.750000 0.700000 1.000000
view_transf: cylinder_radius 1.500000
#
# Turns off the display of the quadric tool
#
tools: quadric none
```

See Also:

Associated view_transf commands

view_transf: distance

Command:

```
view_transf: distance <dist>
```

Purpose:

Sets the distance from the look_from to the look_at point.

Parameters:

Type	Parameter	Description
float	`dist`	distance between the `look_from` and the `look_at` point (in model system)

Notes:

Since transformations can be different per viewport, this can be set for each viewport, thus a viewport selection is needed if more than one viewport is being used.

This command causes the look_from point to move along the line between the `look_from` and the `look_at` point to the desired distance. The `look_at` point does not move.

Example Usage:

```
#
# Selecting two viewports
#
viewport: select_begin
1 3
viewport: select_end
#
# Setting the look_at and look_from points
# for each of these viewports
#
view_transf: look_at 1.0000e+00 5.0000e-01 1.0000e+00
view_transf: look_from 5.0000e-01 5.0000e-01 2.0000e+01
#
# Changing the distance between the look_at and the look_from
# points. The look_at point will not change, but the
# look_from point will be moved appropriately
#
view_transf: distance 2.5000e+01
```

See Also:

Associated view_transf commands

Command:

```
view_transf: function <mode>
```

Purpose:

Sets the transformation mode.

Parameters:

Type	Parameter	Description	
constant	mode	box	box tool mode (rotate, translate or scale the box tool according to the global or local frame selected)
		cone	quadric cone tool mode (translate or scale the cone tool according to the global or local frame selected)
		cursor	cursor tool mode (translate cursor in global or local frame directions according to frame selected)
		cylinder	quadric cylinder tool mode (translate or scale the cylinder tool according to the global or local frame selected)
		frame	frame definition mode (rotates or translates the selected frame coordinate triad directions and/or origin in the model coordinate system
		global	global transformation mode (rotate, translate, zoom, or scale the model about the screen coordinate system)
		line	line tool mode (translate or scale the line tool according to the global or local frame selected)
		local	local transformation mode (rotate, translate, zoom, or scale the model about a local axis frame)
		plane	plane tool mode (rotate, translate or scale the plane tool according to the global or local frame selected)
		revolution	quadric revolution tool mode (translate or scale the revolution tool according to the global or local frame selected)
		sphere	quadric sphere tool mode (translate or scale the sphere tool according to the global or local frame selected)

Example Usage:

```
#
# Sets the function to cursor mode
# and translates the cursor
#
view_transf: function cursor
view_transf: translate 0.295634 -0.014792 -0.144601
#
# Return to global transformation mode
#
view_transf: function global
#
# Does some global transformations
#
```

```
view_transf: zoom 1.431051
view_transf: rotate -1.071430 21.547768 0.000000
view_transf: translate 0.620558 0.000000 0.000000
#
# Sets Frame transform mode (local)
#
view_transf: function local
#
# Selects the viewport and frame to use
#
viewport: select_begin
3
viewport: select_end
frame: select_begin
1
frame: select_end
#
# And translate the parts attached to frame 1
#
view_transf: translate 0.0000E+00 -4.0000e+00 0.0000e+00
#
# Return to global transformation mode
#
view_transf: function global
```

See Also:

Associated view_transf commands

Command:

```
view_transf: initialize_viewports
```

Purpose:

Resets transformations, `look_at` viewing parameters, and z clipping to the original values applied when model was first displayed.

Parameters:

none

Notes:

Since transformations can be different per viewport, this can be applied to each viewport. Thus, a viewport selection is needed if more than one viewport is being used.

Example Usage:

```
#
# Selecting two viewports
#
viewport: select_begin
1 3
viewport: select_end
#
# Reinitializing these 2 viewports
#
view_transf: initialize_viewports
```

See Also:

Associated view_transf commands

view_transf: limit

Command:

```
view_transf: limit <value>
```

Purpose:

Sets value represented by the extreme ends of the transformation slider in the transformation dialog.

Parameters:

Type	Parameter	Description
float	`value`	the value to set as the limit for a complete movement of the cursor in the positive or negative directions

Notes:

This is a control for the user interface. Moving the slider all the way to the right will apply this value once positively. Moving it all the way to the left will apply this value once negatively. The slider is self centering, so one can continue to apply these amounts to the selected transformations.

Associated with this is the Increment, which is the value that is applied (positively or negatively) by clicking the stepper button once.

Example Usage:

```
view_transf: limit 5.000000
```

See Also:

Associated view_transf commands

Command:

```
view_transf: line <end_pt> <x_coord> <y_coord> <z_coord>
```

Purpose:

Sets the location of the line tool by setting two endpoints.

Parameters:

Type	Parameter	Description
constant	end_pt	1 - for the first end of the line
		2 - for the second end of the line
float	x_coord	the x coordinate of the end point
	y_coord	the y coordinate of the end point
	z_coord	the z coordinate of the end point

Notes:

Coordinates are in reference to the selected coordinate frame.

Example Usage:

```
#
# Set the line tool endpoints
#
view_transf: line 1 -0.723300 0.700000 1.000000
view_transf: line 2 1.723300 0.500000 1.000000
```

See Also:

Associated view_transf commands

view_transf: line_length

Command:

```
view_transf: line_length <length>
```

Purpose:

Changes the line tool length.

Parameters:

Type	Parameter	Description
float	`length`	desired length of the line tool

Notes:

When this is applied, the location of endpoint 1 will not change, but endpoint 2 will be moved along the line through endpoint 1 and endpoint 2, until the points are separated by the length desired.

Example Usage:

```
#
# Set the line tool endpoints
#
view_transf: line 1 -0.723300 0.700000 1.000000
view_transf: line 2 1.723300 0.500000 1.000000
#
# Now make the tool 10 units long
# (endpoint 2 will be moved)
#
view_transf: line_length 10.000000
```

See Also:

Associated view_transf commands

Command:

```
view_transf: look_at <x_coord> <y_coord> <z_coord>
```

Purpose:

Sets the look_at point for the selected viewport(s).

Parameters:

Type	Parameter	Description
float	x_coord	model x, y, z coordinates of the look_at point
	y_coord	
	z_coord	

Notes:

Since transformations can be different per viewport, this can be set for each viewport, thus a viewport selection is needed if more than one viewport is being used.

Coordinates are in reference to the model coordinate system.

Example Usage:

```
#
# Selecting two viewports
#
viewport: select_begin
1 3
viewport: select_end
#
# Setting the look_at and look_from points
# for each of these viewports
#
view_transf: look_at 1.0000e+00 5.0000e-01 1.0000e+00
view_transf: look_from 5.0000e-01 5.0000e-01 2.0000e+01
#
# Changing the distance between the look_at and the look_from
# points. The look_at point will not change, but the
# look_from point will be moved appropriately
#
view_transf: distance 2.5000e+01
```

See Also:

Associated view_transf commands

view_transf: look_from

Command:

```
view_transf: look_from <x_coord> <y_coord> <z_coord>
```

Purpose:

Sets the look_from point for the selected viewport(s).

Parameters:

Type	Parameter	Description
float	x_coord	model x, y, z coordinates of the look_from point
	y_coord	
	z_coord	

Notes:

Since transformations can be different per viewport, this can be set for each viewport, thus a viewport selection is needed if more than one viewport is being used.

Coordinates are in reference to the model coordinate system.

Example Usage:

```
#
# Selecting two viewports
#
viewport: select_begin
1 3
viewport: select_end
#
# Setting the look_at and look_from points
# for each of these viewports
#
view_transf: look_at 1.0000e+00 5.0000e-01 1.0000e+00
view_transf: look_from 5.0000e-01 5.0000e-01 2.0000e+01
#
# Changing the distance between the look_at and the look_from
# points. The look_at point will not change, but the
# look_from point will be moved appropriately
#
view_transf: distance 2.5000e+01
```

See Also:

Associated view_transf commands

Command:

```
view_transf: plane <corner_pt> <x_coord> <y_coord> <z_coord>
```

Purpose:

Sets the location of the plane tool by setting three corner points.

Parameters:

Type	Parameter	Description
constant	corner_pt	1 - for the first corner of the plane tool
		2 - for the second corner of the plane tool
		3 - for the third corner of the plane tool
	x_coord	x coordinate of the corner point
	y_coord	y coordinate of the corner point
	z_coord	z coordinate of the corner point

Notes:

These points are best thought of as 1 being the lower left corner, 2 being the lower right corner, and 3 being the upper right corner.

Coordinates are in reference to the selected coordinate frame.

Example Usage:

```
#
# Set the plane tool corner points
#
view_transf: plane 1 -0.750000 -0.750000 0.900000
view_transf: plane 2 1.750000 -0.750000 1.100000
view_transf: plane 3 1.750000 2.000000 1.000000
```

See Also:

Associated view_transf commands

view_transf: plane_equation

Command:

```
view_transf: plane_equation <A> <B> <C> <D>
```

Purpose:

Sets the location of the plane tool by specifying the coefficients of the plane equation (Ax + By + Cz = D).

Parameters:

Type	Parameter	Description
float	A	the x coefficient
	B	the y coefficient
	C	the z coefficient
	D	the constant term

Notes:

Coefficients are in reference to the selected coordinate frame.

Example Usage:

```
#
# Set the plane tool by the plane equation
#
view_transf: plane_equation -1.000000 0.500000 12.499997 11.659089
```

See Also:

Associated view_transf commands

Command:

```
view_transf: plane_normal <x_dir> <y_dir> <z_dir>
```

Purpose:

Specify the plane tool orientation.

Parameters:

Type	Parameter	Description
float	x_dir	x, y, z component of the plane tool normal
	y_dir	
	z_dir	

Notes:

The normal is in reference to the selected coordinate frame.

Example Usage:

```
tools: plane ON
view_transf: function plane
#
# define the plane to be located at 0,1,0 with the z-axis (plane normal)
# pointing in the x direction.  The plane is 3 units in X and 4 units in
Y
# (x and y according to the plane definition)
#
view_transf: plane_origin 0.000000e+00 1.000000e+00 0.000000e+00
view_transf: plane_normal 1.000000e+00 0.000000e+00 0.000000e+00
view_transf: plane_size 3.000000e+00 4.000000e+00
```

See Also:

Associated view_transf commands

view_transf: plane_origin

Command:

```
view_transf: plane_origin <x_ori> <y_ori> <z_ori>
```

Purpose:

Specify the plane tool origin.

Parameters:

Type	Parameter	Description
float	x_ori	The x, y, z origin of the plane tool
	y_ori	
	z_ori	

Notes:

The origin is in reference to the selected coordinate frame.

Example Usage:

```
tools: plane ON
view_transf: function plane
#
# define the plane to be located at 0,1,0 with the z-axis (plane normal)
# pointing in the x direction.  The plane is 3 units in X and 4 units in
Y
# (x and y according to the plane definition)
#
view_transf: plane_origin 0.000000e+00 1.000000e+00 0.000000e+00
view_transf: plane_normal 1.000000e+00 0.000000e+00 0.000000e+00
view_transf: plane_size 3.000000e+00 4.000000e+00
```

See Also:

Associated view_transf commands

Command:

```
view_transf: plane_size <x_size> <y_size>
```

Purpose:

Specify the plane tool size.

Parameters:

Type	Parameter	Description
float	x_size	Size of the plane tool in the x, y direction
	y_size	

Notes:

X and Y directions are according to the plane tool definition.

Example Usage:

```
tools: plane ON
view_transf: function plane
#
# define the plane to be located at 0,1,0 with the z-axis (plane normal)
# pointing in the x direction.  The plane is 3 units in X and 4 units in
Y
# (x and y according to the plane definition)
#
view_transf: plane_origin 0.000000e+00 1.000000e+00 0.000000e+00
view_transf: plane_normal 1.000000e+00 0.000000e+00 0.000000e+00
view_transf: plane_size 3.000000e+00 4.000000e+00
```

See Also:

Associated view_transf commands

view_transf: redo

Command:

```
view_transf: redo
```

Purpose:

Redo the undo.

Parameters:

none

Example Usage:

```
view_transf: function global
view_transf: rotate 2.510301e+01 7.117291e+01 0.000000e+00
view_transf: undo
view_transf: redo
```

See Also:

Associated view_transf commands

Command:

```
view_transf: reset_by_action
```

Purpose:

Initializes the current action transformation, thus removing the effect of that transformation.

Parameters:

none

Notes:

Since transformations can be different per viewport, this command can be used for each viewport, thus a viewport selection is needed if more than one viewport is being used.

Example Usage:

```
#
# Set the action to rotate
# and the axis to be x
#
view_transf: action rotate
view_transf: axis x
#
# Select the viewport(s) desired
#
viewport: select_begin
0 2
viewport: select_end
#
# perform a couple of rotates
#
view_transf: rotate 0.500000 0.000000 0.00000
view_transf: rotate -30.00000 0.000000 0.00000
#
# Now remove both rotates using this command
# because the action is rotate.
#
view_transf: reset_by_action
```

See Also:

Associated view_transf commands

view_transf: reset_rot_trn_scl

Command:

```
view_transf: reset_rot_trn_scl
```

Purpose:

Removes the rotation, translation, and scale transformations in the selected viewports.

Parameters:

none

Notes:

Since transformations can be different per viewport, this command can be used for each viewport, thus a viewport selection is needed if more than one viewport is being used.

In Frame mode initializes the selected frames.

Example Usage:

```
#
# Select the viewport(s) desired
#
viewport: select_begin
0 2
viewport: select_end
#
# perform some rotations, translations and/or scales
#
view_transf: rotate 15.535719 -27.859741 0.000000
view_transf: translate 0.620558 0.000000 0.000000
view_transf: scale 1.2000e+00 1.2000e+00 1.2000e+00
#
# Now remove these transformations using this command
#
view_transf: reset_rot_trn_scl
```

See Also:

Associated view_transf commands

Command:

```
view_transf: reset_tool_constrain <option>
```

Purpose:

Sets scope for the resetting of tool locations. The reset can cause the tool to be placed at its initial default location or at the center of the parts in the current viewport.

Parameters:

Type	Parameter	Description	
constant	option	by_global	causes the tool to be placed at the initial default location.
		by_viewport	causes the tool to be places at the center of the parts in the current viewport

Notes:

The state set here applies to all future tool resets.

Example Usage:

```
#
# Set the tool reset state to be "by viewport"
#
view_transf: reset_tool_constrain by_viewport
#
# Select the viewport(s) desired
#
viewport: select_begin
0
viewport: select_end
#
# Reset the plane tool to the center of parts
# parts in viewport 0
#
view_transf: function plane
view_transf: reset_rot_trn_scl
```

See Also:

Associated view_transf commands

Command:

```
view_transf: restore_view <filename>
```

Purpose:

Restores a previously saved view state file.

Parameters:

Type	Parameter	Description
string	`filename`	name of the file containing the saved view

Notes:

Since transformations are viewport independent, you need to select the desired viewport before restoring a view state (if you are using multiple viewports).

Example Usage:

```
#
# Select the viewport(s) desired
#
viewport: select_begin
0
viewport: select_end
#
# Save the current view state into the file
# /usr/tmp/view1
#
view_transf: save_view /usr/tmp/view1
#
# You could now do other transformations etc...
#
# And then when you want to restore to the view state
# we saved, do the following
#
view_transf: restore_view /usr/tmp/view1
```

See Also:

Associated view_transf commands

Command:

```
view_transf: revolution_#_of_points <number_of>
```

Purpose:

Defines the number of points on the revolution tool.

Parameters:

Type	Parameter	Description
int	number_of	number of points defined for the revolution tool

Example Usage:

```
#
# Turns display of the quadric surface of revolution on
# (only one quadric tool at a time can be on)
#
tools: quadric revolution
#
# Sets the axis and other attributes of the revolution tool
#
view_transf: revolution_origin 0.500000 0.700000 1.000000
view_transf: revolution_axis 0.995040 0.099504 0.000000
view_transf: revolution_#_of_points 7
view_transf: revolution_point 1 -1.250000 0.500000
view_transf: revolution_point 2 -0.625000 0.500000
view_transf: revolution_point 3 0.000000 1.000000
view_transf: revolution_point 4 0.301802 0.720000
view_transf: revolution_point 5 0.319820 0.710000
view_transf: revolution_point 6 0.625000 0.500000
view_transf: revolution_point 7 1.250000 0.500000
#
# Turns off the display of the revolution tool
#
tools: quadric none
```

See Also:

Associated view_transf commands

view_transf: revolution_axis

Command:

```
view_transf: revolution_axis <x_comp> <y_comp> <z_comp>
```

Purpose:

Sets the axis for the revolution tool.

Parameters:

Type	Parameter	Description
float	x_comp	x, y, z directional components of the revolution tool axis
	y_comp	
	z_comp	

Notes:

Coordinates are in reference to the selected coordinate frame.

Example Usage:

```
#
# Turns display of the quadric surface of revolution on
# (only one quadric tool at a time can be on)
#
tools: quadric revolution
#
# Sets the axis and other attributes of the revolution tool
#
view_transf: revolution_origin 0.500000 0.700000 1.000000
view_transf: revolution_axis 0.995040 0.099504 0.000000
view_transf: revolution_#_of_points 7
view_transf: revolution_point 1 -1.250000 0.500000
view_transf: revolution_point 2 -0.625000 0.500000
view_transf: revolution_point 3 0.000000 1.000000
view_transf: revolution_point 4 0.301802 0.720000
view_transf: revolution_point 5 0.319820 0.710000
view_transf: revolution_point 6 0.625000 0.500000
view_transf: revolution_point 7 1.250000 0.500000
#
# Turns off the display of the revolution tool
#
tools: quadric none
```

See Also:

Associated view_transf commands

Command:

```
view_transf: revolution_origin <x_coord> <y_coord> <z_coord>
```

Purpose:

Sets the origin point for the first end of the revolution tool.

Parameters:

Type	Parameter	Description
float	x_coord	x, y, z coordinates of 1st end of the revolution tool
	y_coord	
	z_coord	

Notes:

Coordinates are in reference to the selected coordinate frame.

Example Usage:

```
#
# Turns display of the quadric surface of revolution on
# (only one quadric tool at a time can be on)
#
tools: quadric revolution
#
# Sets the axis and other attributes of the revolution tool
#
view_transf: revolution_origin 0.500000 0.700000 1.000000
view_transf: revolution_axis 0.995040 0.099504 0.000000
view_transf: revolution_#_of_points 7
view_transf: revolution_point 1 -1.250000 0.500000
view_transf: revolution_point 2 -0.625000 0.500000
view_transf: revolution_point 3 0.000000 1.000000
view_transf: revolution_point 4 0.301802 0.720000
view_transf: revolution_point 5 0.319820 0.710000
view_transf: revolution_point 6 0.625000 0.500000
view_transf: revolution_point 7 1.250000 0.500000
#
# Turns off the display of the revolution tool
#
tools: quadric none
```

See Also:

Associated view_transf commands

view_transf: revolution_point

Command:

```
view_transf: revolution_point <pt_number> <x_coord> <y_coord>
```

Purpose:

Defines the location of the given point in relation to the revolution axis.

Parameters:

Type	Parameter	Description
int	pt_number	revolution tool point number for which the coordinates are being given
float	x_coord	x coordinate of the point on the revolution tool. Namely, the distance parallel to the revolution axis from the origin of the tool
	y_coord	y coordinate of the point on the revolution tool. Namely, the distance perpendicular to the revolution axis from the origin of the tool

Example Usage:

```
#
# Turns display of the quadric surface of revolution on
# (only one quadric tool at a time can be on)
#
tools: quadric revolution
#
# Sets the axis and other attributes of the revolution tool
#
view_transf: revolution_origin 0.500000 0.700000 1.000000
view_transf: revolution_axis 0.995040 0.099504 0.000000
view_transf: revolution_#_of_points 7
view_transf: revolution_point 1 -1.250000 0.500000
view_transf: revolution_point 2 -0.625000 0.500000
view_transf: revolution_point 3 0.000000 1.000000
view_transf: revolution_point 4 0.301802 0.720000
view_transf: revolution_point 5 0.319820 0.710000
view_transf: revolution_point 6 0.625000 0.500000
view_transf: revolution_point 7 1.250000 0.500000
#
# Turns off the display of the revolution tool
#
tools: quadric none
```

See Also:

Associated view_transf commands

Command:

```
view_transf: rotate <x_ang> <y_ang> <z_ang>
```

Purpose:

Specifies a rotation (in degrees about each of the axes). Used for global, frame, and tool transformations.

Parameters:

Type	Parameter	Description
float	x_ang	number of degrees to rotate about the x,y,z axes. (sign is according to the right-hand rule)
	y_ang	
	z_ang	

Notes:

Since transformations can be different per viewport, this can be set for each viewport, thus a viewport selection is needed if more than one viewport is being used.

Example Usage:

```
#
# Selecting viewport(s)
#
viewport: select_begin
2
viewport: select_end
#
# Set the axis to x
# and rotate about the x axis 15 degrees
#
view_transf: axis x
view_transf: rotate 15.000000 0.000000 0.000000
#
# Set the axis to be y
# and translate one unit in the -negative y direction
#
view_transf: axis y
view_transf: translate 0.0000e+00 -1.0000e+00 0.0000e+00
#
# Now showing it with a plane tool rotation
#
tools: plane ON
view_transf: function plane
view_transf: rotate 0.000000 -39.575481 0.000000
view_transf: function global
#
# And now with a local transformation
#
part: select_default
frame: create
frame: select_begin
1
frame: select_end
part: select_begin
1
part: select_end
```

```
frame: assign 1
view_transf: function local
view_transf: axis y
view_transf: rotate 0.000000 27.000000 0.000000
```

See Also:

Associated view_transf commands

Command:

```
view_transf: save_view <filename>
```

Purpose:

Save the current view state into a file.

Parameters:

Type	Parameter	Description
string	`filename`	the name (including the path) of the file in which to save the view state

Example Usage:

```
#
# Select the viewport(s) desired
#
viewport: select_begin
0
viewport: select_end
#
# Save the current view state into the file
# /usr/tmp/view1
#
view_transf: save_view /usr/tmp/view1
#
# You could now do other transformations etc...
#
# And then when you want to restore to the view state
# we saved, do the following
#
view_transf: restore_view /usr/tmp/view1
```

See Also:

Associated view_transf commands

view_transf: scale

Command:

```
view_transf: scale <x_scale> <y_scale> <z_scale>
```

Purpose:

Defines a scaling in each of the axis directions. Used for global, frame, and tool transformation.

Parameters:

Type	Parameter	Description
float	x_scale	scale factor to apply to x, y, z coordinates
	y_scale	
	z_scale	

Notes:

Since transformations can be different per viewport, this can be set for each viewport, thus a viewport selection is needed if more than one viewport is being used.

Example Usage:

```
#
# Selecting viewport(s)
#
viewport: select_begin
2
viewport: select_end
#
# Applying a global scaling
# (evenly in all directions)
#
view_transf: scale 1.2000e+00 1.2000e+00 1.2000e+00
#
# Now with a local transformation
# (Scaling only in the z direction)
#
frame: select_begin
1
frame: select_end
view_transf: function local
view_transf: scale 0.0000e+00 0.0000e+00 5.0000e+00
```

See Also:

Associated view_transf commands

Command:

```
view_transf: sphere_axis <x_comp> <y_comp> <z_comp>
```

Purpose:

Sets the axis components for the sphere tool.

Parameters:

Type	Parameter	Description
float	x_comp	x, y, z components of the sphere tool axis
	y_comp	
	z_comp	

Notes:

Coordinates are in reference to the selected coordinate frame.

Example Usage:

```
#
# Turns display of the quadric sphere tool on
# (only one quadric tool at a time can be on)
#
tools: quadric sphere
#
# Sets the axis and other attributes of the sphere tool
#
view_transf: sphere_radius 1.500000
view_transf: sphere_axis 1.000000 0.100000 0.000000
view_transf: sphere_origin 0.500000 0.700000 1.000000
#
# Turns off the display of the sphere tool
#
tools: quadric none
```

See Also:

Associated view_transf commands

view_transf: sphere_origin

Command:

```
view_transf: sphere_origin <x_coord> <y_coord> <z_coord>
```

Purpose:

Sets the origin point for the first end of the sphere tool.

Parameters:

Type	Parameter	Description
float	x_coord	x, y, z coordinates of 1st end of the sphere tool
	y_coord	
	z_coord	

Notes:

Coordinates are in reference to the selected coordinate frame.

Example Usage:

```
#
# Turns display of the quadric sphere tool on
# (only one quadric tool at a time can be on)
#
tools: quadric sphere
#
# Sets the origin and other attributes of the sphere tool
#
view_transf: sphere_radius 1.500000
view_transf: sphere_axis 1.000000 0.100000 0.000000
view_transf: sphere_origin 0.500000 0.700000 1.000000
#
# Turns off the display of the sphere tool
#
tools: quadric none
```

See Also:

Associated view_transf commands

Command:

```
view_transf: sphere_radius <value>
```

Purpose:

Defines the radius of the sphere tool.

Parameters:

Type	Parameter	Description
float	value	radius of the sphere tool

Example Usage:

```
#
# Turns display of the quadric sphere tool on
# (only one quadric tool at a time can be on)
#
tools: quadric sphere
#
# Sets the axis and other attributes of the sphere tool
#
view_transf: sphere_radius 1.500000
view_transf: sphere_axis 1.000000 0.100000 0.000000
view_transf: sphere_origin 0.500000 0.700000 1.000000
#
# Turns off the display of the sphere tool
#
tools: quadric none
```

See Also:

Associated view_transf commands

view_transf: spin

Command:

```
view_transf: spin <num> <x_ang> <y_ang> <z_ang>
```

Purpose:

Applies a global rotation multiple times.

Parameters:

Type	Parameter	Description
int	num	number of times that the rotation is to be applied
float	x_ang	number of degrees to rotate about the x, y, z axes. (sign is according to the right-hand rule)
	y_ang	
	z_ang	

Notes:

Interactively, the model will continue to spin until the user stops it. Once stopped, the command language contains the number of times the rotation was applied.

Example Usage:

```
#
# Selecting viewport(s)
#
viewport: select_begin
2
viewport: select_end
#
# Spin only works with global rotations
#
view_transf: function global
#
# Apply the rotation specified
* (In this case, 105 times)
#
view_transf: spin 105 0.602679 -10.175333 0.000000
```

See Also:

Associated view_transf commands

Command:

```
view_transf: translate <x_dist> <y_dist> <z_dist>
```

Purpose:

Specifies a translation in each of the coordinate directions. Used for global, frame, and tool transformations.

Parameters:

Type	Parameter	Description
float	x_dist	distance to translate in the x, y, z directions
	y_dist	
	z_dist	

Notes:

Since transformations can be different per viewport, this can be set for each viewport, thus a viewport selection is needed if more than one viewport is being used.

Example Usage:

```
#
# Sets the function to cursor mode
# and translates the cursor
#
view_transf: function cursor
view_transf: translate 0.295634 -0.014792 -0.144601
#
# Return to global transformation mode
#
view_transf: function global
#
# Does some global transformations
# (one of which is a translate)
#
view_transf: zoom 1.431051
view_transf: rotate -1.071430 21.547768 0.000000
view_transf: translate 0.620558 0.000000 0.000000
#
# Sets Frame transform mode (local)
#
view_transf: function local
#
# Selects the viewport and frame to use
#
viewport: select_begin
3
viewport: select_end
frame: select_begin
1
frame: select_end
#
# And translate the parts attached to frame 1
#
view_transf: translate 0.0000E+00 -4.0000e+00 0.0000e+00
#
# Return to global transformation mode
```

```
#
view_transf: function global
```

See Also:

Associated view_transf commands

Command:

```
view_transf: undo
```

Purpose:

Undo the last transformation or tool definition.

Parameters:

none

Example Usage:

```
view_transf: function global
view_transf: rotate 2.510301e+01 7.117291e+01 0.000000e+00
view_transf: undo
view_transf: redo
```

See Also:

Associated view_transf commands

Command:

```
view_transf: view_recall <option>
```

Purpose:

View the model from a standard (or customized) view position.

Parameters:

Type	Parameter	Description	
constant	option	F5	right side view (or a custom view if so saved)
		F6	top view (or a custom view if so saved)
		F7	front view (or a custom view if so saved)
		F8	previous view (or a custom view if so saved)
		+X	from the positive x direction toward the origin
		+Y	from the positive y direction toward the origin
		+Z	from the positive z direction toward the origin
		-X	from the negative x direction toward the origin
		-Y	from the negative y direction toward the origin
		-Z	from the negative z direction toward the origin

Notes:

The `F5, F6, F7` views have standard right, top, and front views for a default. These can be altered by the user using the "`view_transf: view_store`" command. Since transformations can be different per viewport, this can be set for each viewport, thus a viewport selection is needed if more than one viewport is being used.

Example Usage:

```
#
# Select viewport for which you wish
# to modify the view
#
viewport: select_begin
1
viewport: select_end
view_transf: view_recall F5
#
# Selecting a different viewport
# to show another example
#
viewport: select_begin
2
viewport: select_end
view_transf: view_recall +Y
```

See Also:

Associated view_transf commands

Command:

```
view_transf: view_store <option>
```

Purpose:

Stores a current View state as a state attached to one of the four customizable function keys.

Parameters:

Type	Parameter	Description	
constant	option	F5	function key F5, (default of which is right side view)
		F6	function key F6, (default of which is top view
		F7	function key F7, (default of which is front view)
		F8	function key F8, (default of which is previous view)

Notes:

The F5, F6, F7 views have standard right, top, and front views for a default. These can be altered by the user using this command. The views stored here can be recalled using the "view_transf: view_recall" command.

Since transformations can be different per viewport, this can be set for each viewport, thus a viewport selection is needed if more than one viewport is being used.

Example Usage:

```
#
# Select viewport that contains the
# view that you wish to store.
#
viewport: select_begin
1
viewport: select_end
#
# Store it into the desired F key
#
view_transf: view_store F5
```

See Also:

Associated view_transf commands

view_transf: viewport_link

Command:

```
view_transf: viewport_link <toggle>
```

Purpose:

Link the interactive global transformations of multiple viewports. This causes any transformations applied to one of the viewports to be applied to all the other linked viewports as well.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	to link all currently selected viewports
		`OFF`	to unlink all currently selected viewports

Notes:

Only interactive transforms are linked.

Example Usage:

```
#
# Selecting three viewports, and linking them
#
viewport: select_begin
0 1 2
viewport: select_end
view_transf: viewport_link ON
```

See Also:

Associated view_transf commands

Command:

```
view_transf: viewport_state_copy
```

Purpose:

Save the transformation state of the currently selected viewport (for the purpose of applying it to a different viewport with the "`view_transf: viewport_state_paste`" command.

Parameters:

none

Example Usage:

```
#
# Select viewport for which you wish
# to save the transformation state
#
viewport: select_begin
1
viewport: select_end
view_transf: viewport_state_copy
#
# Select the viewport you wish to
# apply this state to
#
viewport: select_begin
2
viewport: select_end
view_transf: viewport_state_paste
```

See Also:

Associated view_transf commands

view_transf: zclip_back/front

Command:

```
view_transf: zclip_back <z_loc>
view_transf: zclip_front <z_loc>
```

Purpose:

Sets the position of the back/front z-clip plane

Parameters:

Type	Parameter	Description
float	`z_loc`	z coordinate of the back/front z-clipping plane

Notes:

This value is used only when "`view_transf: zclip_float`" is `OFF`. Since transformations can be different per viewport, this can be set for each viewport, thus a viewport selection is needed if more than one viewport is being used.

Example Usage:

```
#
# Select viewport
#
viewport: select_begin
1
viewport: select_end
#
# Set z-clip float OFF and set the location
# the front and back clipping planes
#
view_transf: zclip_float OFF
view_transf: zclip_front 7.2296e+00
view_transf: zclip_back 3.5459e+01
```

See Also:

Associated view_transf commands

Command:

```
view_transf: zclip_float <toggle>
```

Purpose:

Sets the z-clipping state. The front and back z-clipping planes can "`float`" so that they are always outside the model - causing no z-clipping to occur, or they can be set once initially and are then subject to manual modification.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	to have the z-clip planes float near, but outside the model z extents
		`OFF`	to be able to manually set front and back z-clipping plane locations

Notes:

Since the z-buffer uses the resolution between the front and back clipping planes, it is desirable to have these planes "near" the model. However, unless you specifically want to perform z-clipping, these by default "float" as needed to avoid any z-clipping. If you want z-clipping control, you can turn this "floating" behavior off and set the front and back plane locations yourself. During the float mode, near clipping will occur if it must be at the specified minimum z-clip location. A minimum is needed to avoid z reflection.Since transformations can be different per viewport, this can be set for each viewport, thus a viewport selection is needed if more than one viewport is being used.

Example Usage:

```
#
# Select viewport
#
viewport: select_begin
1
viewport: select_end
#
# Set z-clip float OFF and set the location
# of the front and back clipping planes
#
view_transf: zclip_float OFF
view_transf: zclip_front 7.2296e+00
view_transf: zclip_back 3.5459e+01
#
# To have them float instead
#
view_transf: zclip_float ON
view_transf: zclip_min 4.00000e-04
```

See Also:

Associated view_transf commands

view_transf: zclip_min

Command:

```
view_transf: zclip_min <z_min>
```

Purpose:

Sets the minimum to which the front z-clipping plane is allowed to "float".

Parameters:

Type	Parameter	Description
float	z_min	minimum allowable z location for the front z-clipping plane

Notes:

Since the z-buffer uses the resolution between the front and back clipping planes, it is desirable to have these planes "near" the model. However, unless you specifically want to perform z-clipping, these by default "float" as needed to avoid any z-clipping. If you want z-clipping control, you can turn this "floating" behavior off and set the front and back plane locations yourself. During the float mode, near clipping will occur if it must be at the specified minimum z-clip location. A minimum is needed to avoid z reflection. Since transformations can be different per viewport, this can be set for each viewport, thus a viewport selection is needed if more than one viewport is being used.

Example Usage:

```
#
# Select viewport
#
viewport: select_begin
1
viewport: select_end
#
# Set z-clip float OFF and set the location
# of the front and back clipping planes
#
view_transf: zclip_float OFF
view_transf: zclip_front 7.2296e+00
view_transf: zclip_back 3.5459e+01
#
# To have them float instead
# (note the minimum being set)
#
view_transf: zclip_float ON
view_transf: zclip_min 4.00000e-04
```

See Also:

Associated view_transf commands

Command:

```
view_transf: zoom <factor>
```

Purpose:

Move the viewer closer or farther away.

Parameters:

Type	Parameter	Description
float	`factor`	amount of zoom scaling

Notes:

Since transformations can be different per viewport, this can be set for each viewport, thus a viewport selection is needed if more than one viewport is being used.

Example Usage:

```
#
# Selecting viewport(s)
#
viewport: select_begin
2
viewport: select_end
#
# Applying a zoom factor
#
view_transf: zoom 1.431051
```

See Also:

Associated view_transf commands

view_transf: zoom_style

Command:

```
view_transf: zoom_style <style>
```

Purpose:

Select the zoom user interaction style.

Parameters:

Type	Parameter	Description	
constant	style	automatic_slide	will continue to zoom as long as mouse button is down
		manual_drag	will zoom only when the mouse is dragged

Example Usage:

```
view_transf: zoom_style automatic_slide
```

See Also:

Associated view_transf commands

Commands in this Section:

viewport

See Also:

User Manual - Section 8.4, VPort Mode

How To Define and Change Viewports

viewport: background_type

Command:

```
viewport: background_type <type>
```

Purpose:

Specify the type of background to be displayed for the selected viewport(s).

Parameters:

Type	Parameter	Description	
constant	type	the following types are applied to the selected viewports	
		blended	blend the background color between the specified colors between each specified level
		constant	color the background a constant color (default)
		inherit	inherits the background color of the main viewport, (viewport 0), i.e. the main view's background color shows through

Example Usage:

```
viewport: select_begin
 0
viewport: select_end
viewport: background_type blended
viewport; blend_levels 3
viewport: blend_editlevel 1
viewport: blend_rgb 1.0000e+00 1.0000e+00 1.0000e+00
viewport: blend_editlevel 2
viewport: blend_rgb 0.0000e+00 0.0000e+00 1.0000e+00
viewport: blend_editlevel 3
viewport: blend_position 3.2000e-01
viewport: blend_rgb 0.0000e+00 0.0000e+00 0.0000e+00
viewport: select_begin
 1
viewport: select_end
viewport: background_type inherit
viewport: select_begin
 2
viewport: select_end
viewport: background_type constant
viewport: constant_rgb 0.0000e+00 0.0000e+00 0.0000e+00
```

See Also:

Associated viewport commands

Command:

```
viewport: blend_editlevel <level_num>
```

Purpose:

Specify the level to edit of the corresponding blended background type for the selected viewport(s), i.e. used to edit either the position or color of the level.

Parameters:

Type	Parameter	Description
int	`level_num`	specified level number for the blended background type

Example Usage:

```
viewport: select_begin
 0
viewport: select_end
viewport: background_type blended
viewport; blend_levels 3
viewport: blend_editlevel 1
viewport: blend_rgb 1.0000e+00 1.0000e+00 1.0000e+00
viewport: blend_editlevel 2
viewport: blend_rgb 0.0000e+00 0.0000e+00 1.0000e+00
viewport: blend_editlevel 3
viewport: blend_position 3.2000e-01
viewport: blend_rgb 0.0000e+00 0.0000e+00 0.0000e+00
```

See Also:

Associated viewport commands

viewport: blend_levels

Command:

```
viewport: blend_levels <num>
```

Purpose:

Specify the number of (vertical) levels of the corresponding blended background type for the selected viewport(s).

Parameters:

Type	Parameter	Description
int	num	number of levels in I[1,n] for the blended background type

Example Usage:

```
viewport: select_begin
 0
viewport: select_end
viewport: blend_levels 3
```

See Also:

Associated viewport commands

Command:

```
viewport: blend_position <position>
```

Purpose:

Specify the position of the specified blend level of the corresponding blended background type for the selected viewport(s).

Parameters:

Type	Parameter	Description
float	position	normalized position in I[0.,1.] for the blended background level

Notes:

Blend level 0 must be located at 0.

Blend level N must be locted at 1.

Blend level N must be > N-1.

Example Usage:

```
viewport: select_begin
 0
viewport: select_end
viewport: background_type blended
viewport; blend_levels 3
viewport: blend_editlevel 2
viewport: blend_position 3.2000e-01
viewport: blend_rgb 0.0000e+00 0.0000e+00 1.0000e+00
```

See Also:

Associated viewport commands

viewport: blend_rgb

Command:

```
viewport: blend_rgb <red_val> <grn_val> <blu_val>
```

Purpose:

Specify the red, green and blue components of the blended background color for the specified level of the selected viewport(s).

Parameters:

Type	Parameter	Description
float	red_val	red, green, blue color component values given in I[0.,1.]
	grn_val	
	blu_val	

Example Usage:

```
viewport: select_begin
 0
viewport: select_end
viewport: background_type blended
viewport; blend_levels 3
viewport: blend_editlevel 2
viewport: blend_position 3.2000e-01
viewport: blend_rgb 0.0000e+00 0.0000e+00 1.0000e+00
```

See Also:

Associated viewport commands

Command:

```
viewport: border_rgb <red_val> <grn_val> <blu_val>
```

Purpose:

Set the color for the viewport border.

Parameters:

Type	Parameter	Description
float	red_val	red, green, blue color component values given in I[0.,1.]
	grn_val	
	blu_val	

Notes:

By default the border color is white.

Used only if `viewport: border_visible` is `ON`.

Example Usage:

```
viewport: select_begin
 1
viewport: select_end
viewport: border_rgb 1. 0. 1.
```

See Also:

Associated viewport commands

viewport: border_visible

Command:

```
viewport: border_visible <toggle>
```

Purpose:

Set visibility of viewport border.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	for visible border
		`OFF`	for invisible border

Notes:

Viewport borders are on by default.

Example Usage:

```
viewport: select_begin
 1
viewport: select_end
viewport: border_visible OFF
```

See Also:

Associated viewport commands

Command:

```
viewport: constant_rgb <red_val> <grn_val> <blu_val>
```

Purpose:

Specify the red, green and blue components of the constant background color for the selected viewport(s)

Parameters:

Type	Parameter	Description
float	red_val	red, green, blue color component values given in I[0.,1.]
	grn_val	
	blu_val	

Example Usage:

```
viewport: select_begin
  2
viewport: select_end
viewport: background_type constant
viewport: constant_rgb 0.0000e+00 0.0000e+00 0.0000e+00
```

See Also:

Associated viewport commands

viewport: delete

Command:

```
viewport: delete
```

Purpose:

Delete all selected viewports.

Parameters:

none

Notes:

After selected viewports have been deleted, the main viewport (0) will be selected.

The main viewport (0) can not be deleted.

Example Usage:

```
viewport: select_begin
 2 5
viewport: select_end
viewport: delete
viewport: select_begin
 0
viewport: select_end
```

See Also:

Associated viewport commands

Command:

```
viewport: height <height>
```

Purpose:

Specify the normalized height for the selected viewport(s).

Parameters:

Type	Parameter	Description
float	height	normalized height I[0.,1.]

Example Usage:

```
viewport: select_begin
 0
viewport: select_end
viewport: origin_x 3.3333e-01
viewport: origin_y 0.0000e+00
viewport: width 5.1712e-01
viewport: height 1.0000e+00
viewport: select_begin
 1 2
viewport: select_end
viewport: height 2.5000e-01
```

See Also:

Associated viewport commands

viewport: hidden_line

Command:

```
viewport: hidden_line <toggle>
```

Purpose:

Toggle whether hidden line meshes are to be overlaid onto parts in active viewport(s).

Parameters:

Type	Parameter	Description
constant	toggle	ON
		OFF

Notes:

A part will be displayed with hidden lines if:

(a) `view: hidden-line  ON`

(b) `viewport: hidden_line  ON`

(c) `part: hidden_line  ON`

Example Usage:

```
viewport: select_begin
 0
viewport: select_end
viewport: hidden_line OFF
viewport: select_begin
 1
viewport: select_end
viewport: hidden_line ON
```

See Also:

Associated viewport commands

Command:

```
viewport: hidden_surface <toggle>
```

Purpose:

Toggle whether hidden surfaces are to be removed from parts in active viewport(s).

Parameters:

Type	Parameter	Description
constant	toggle	ON
		OFF

Notes:

A part will be displayed with shading if:

(a) `view: hidden-surface  ON`

(b) `viewport: hidden_surface  ON`

(c) `part: hidden_surface  ON`

Example Usage:

```
viewport: select_begin
 0
viewport: select_end
viewport: hidden_surface OFF
viewport: select_begin
 1
viewport: select_end
viewport: hidden_surface ON
```

See Also:

Associated viewport commands

viewport: lower

Command:

```
viewport: lower
```

Purpose:

Lower the selected viewport(s) to be located beneath all the other viewports - except the main viewport.

Parameters:

none

Notes:

The main viewport (0) always remains beneath all viewports; and thus, is neither raised or lowered.

Example Usage:

```
viewport: select_begin
 3 4
viewport: select_end
viewport: lower
```

See Also:

Associated viewport commands

Command:

```
viewport: new
```

Purpose:

Specify a new viewport to be created.

Parameters:

none

Notes:

New viewport indexing sequentially increments in the range I[1,16].

Example Usage:

```
viewport: new
viewport: select_begin
 1
viewport: select_end
```

See Also:

Associated viewport commands

viewport: origin_x/y

Command:

```
viewport: origin_x <coord>
viewport: origin_y <coord>
```

Purpose:

Specify the normalized x/y-coordinate value of the origin for the selected viewport(s).

Parameters:

Type	Parameter	Description
float	coord	lower left/bottom of viewport with range I[0.,1.]

Example Usage:

```
viewport: select_begin
 0
viewport: select_end
viewport: origin_x 3.3333e-01
viewport: origin_y 0.0000e+00
viewport: width 5.1712e-01
viewport: height 1.0000e+00
viewport: select_begin
 1 2
viewport: select_end
viewport: origin_x 1.0000e-01
```

See Also:

Associated viewport commands

Command:

```
viewport: perspective <toggle>
```

Purpose:

Toggle whether viewing frustrum is to be perspective or orthographic in selected viewport(s).

Parameters:

Type	Parameter	Description	
constant	toggle	one of following toggle values:	
		ON	perspective viewing frustrum (default)
		OFF	orthographic viewing frustrum

Notes:

Geometry will be shown with perspective if:

(a) `view: perspective  ON`

(b) `viewport: perspective  ON`

Example Usage:

```
viewport: select_begin
 0
viewport: select_end
viewport: perspective OFF
viewport: select_begin
 1
viewport: select_end
viewport: perspective ON
```

See Also:

Associated viewport commands

viewport: raise

Command:

```
viewport: raise
```

Purpose:

Specify the selected viewport(s) to be located above all the other viewports.

Parameters:

none

Notes:

The main viewport (0) always remains beneath all viewports; and thus, is neither raised or lowered.

Example Usage:

```
viewport: select_begin
 2
viewport: select_end
viewport: raise
```

See Also:

Associated viewport commands

Command:

```
viewport: select_all
```

Purpose:

Select all created viewports for editing.

Parameters:

none

Example Usage:

```
viewport: select_all
```

See Also:

Associated viewport commands

viewport: select_begin/end

Command:

```
viewport: select_begin
  <id1 id2 ...idn>
viewport: select_end
```

Purpose:

Set active viewport(s) selection construct for subsequent viewport modification.

Parameters:

Type	Parameter	Description
int	`idn`	list of viewport number(s) to be active for editing.

Notes:

All other "`viewport:`" commands operate off this setting.

Example Usage:

```
viewport: select_begin
 0
viewport: select_end
view_trans: function global
viewport: viewport_layout two_vertical
viewport: select_begin
 1 2
viewport: select_end
```

See Also:

Associated viewport commands

Command:

```
viewport: select_default
```

Purpose:

Allows the setting of default viewport attributes prior to creating the next viewport.

Parameters:

none

Example Usage:

```
viewport: select_default
viewport: perspective OFF
viewport: background_type constant
viewport: constant_rgb 0.0000e+00 0.0000e+00 0.0000e+00
```

See Also:

Associated viewport commands

viewport: viewport_dimension

Command:

```
viewport: viewport_dimension <option>
```

Purpose:

Set the dimension of the viewport.

Parameters:

Type	Parameter	Description
constant	`option`	includes the following viewport dimensions:
		`3D` (default)
		`2D`

Notes:

Dimension 2D is handy for displaying 2D clips and/or developed surfaces.

A 2D viewport can only be transfomed in 2D.

Example Usage:

```
viewport: select_begin
 0 1
viewport: select_end
viewport: viewport_dimension 3D
viewport: select_begin
 2
viewport: select_end
viewport: viewport_dimension 2D
```

See Also:

Associated viewport commands

Command:

```
viewport: viewport_layout <option>
```

Purpose:

Specify the type of layout to be used in dividing the graphics window into viewports.

Parameters:

Type	Parameter	Description	
constant	option	includes the following types of viewport layouts:	
		one	
		two_vertical	
		two_horizontal	
		four_equal	
		four_withthreeleft	Three viewports on left + one large
		four_withthreebottom	Three viewports on bottom + one large

Notes:

When creating a layout of viewports, the active viewport defaults to the last viewport created, i.e. if creating `four_equal` viewports, the active viewport is the last of the four created.

Example Usage:

```
viewport: select_begin
 0
viewport: select_end
view_trans: function global
viewport: viewport_layout two_vertical
viewport: select_begin
 1
viewport: select_end
```

See Also:

Associated viewport commands

viewport: visible

Command:

```
viewport: visible <toggle>
```

Purpose:

Toggle whether selected viewports are visible.

Parameters:

Type	Parameter	Description	
constant	`toggle`	`ON`	viewport is visible
		`OFF`	viewport is not visible

Example Usage:

```
viewport: select_begin
 0
viewport: select_end
viewport: visible OFF
viewport: select_begin
 1
viewport: select_end
viewport: visible ON
```

See Also:

Associated viewport commands

Command:

```
viewport: width <width>
```

Purpose:

Specify the normalized width for the selected viewport(s).

Parameters:

Type	Parameter	Description
float	`width`	normalized width I[0.,1.]

Example Usage:

```
viewport: select_begin
 0
viewport: select_end
viewport: origin_x 3.3333e-01
viewport: origin_y 0.0000e+00
viewport: width 5.1712e-01
viewport: height 1.0000e+00
viewport: select_begin
 1 2
viewport: select_end
viewport: width 2.5000e-01
```

See Also:

Associated viewport commands

Commands in this Section:

viewport_bounds

See Also:

User Manual - Chapter 8, Part Bounds Attributes

Command:

```
viewport_bounds: auto_size <toggle>
```

Purpose:

When geometry changes the bounds display will resize if this toggle is set

Parameters:

Type	Parameter	Description	
constant	toggle	ON	resize the bounds when the geometry changes
		OFF	do not resize the bounds

Example Usage:

```
view: bounds ON
viewport: select_begin
 0
viewport: select_end
viewport_bounds: auto_size OFF
```

See Also:

Associated viewport_bounds commands

viewport_bounds: axis_linewidth

Command:

```
viewport_bounds: axis_linewidth <line_width>
```

Purpose:

Set the line width for the bounds axis

Parameters:

Type	Parameter	Description
int	`line_width`	the pixel width for the axis

Example Usage:

```
view: bounds ON
viewport: select_begin
 0
viewport: select_end
viewport_bounds: visible ON
viewport_bounds: axis_linewidth 2
viewport_bounds: axis_rgb 1.0000e+00 0.0000e+00 1.0000e+00
```

See Also:

Associated viewport_bounds commands

Command:

```
viewport_bounds: axis_rgb  <red_val> <grn_val> <blu_val>
```

Purpose:

Set the color for the bounds axis

Parameters:

Type	Parameter	Description
float	red_val	red, green, blue color components (must be between 0.0 and 1.0)
	grn_val	
	blu_val	

Example Usage:

```
view: bounds ON
viewport: select_begin
 0
viewport: select_end
viewport_bounds: visible ON
viewport_bounds: axis_linewidth 1
viewport_bounds: axis_rgb 1.0000e+00 0.0000e+00 1.0000e+00
```

See Also:

Associated viewport_bounds commands

viewport_bounds: axis_x/y/z_format

Command:

```
viewport_bounds: axis_x_format <format>
viewport_bounds: axis_y_format <format>
viewport_bounds: axis_z_format <format>
```

Purpose:

Specifies the format to be used for labels along the x/y/z axis

Parameters:

Type	Parameter	Description
string	`format`	may be any valid C language format string

Example Usage:

```
view: bounds ON
viewport: select_begin
 0
viewport: select_end
viewport_bounds: axis_x_format %.2f
viewport_bounds: axis_y_format %g
```

See Also:

Associated viewport_bounds commands

Command:

```
viewport_bounds: axis_x_gridextentloc <extent>
viewport_bounds: axis_y_gridextentloc <extent>
viewport_bounds: axis_z_gridextentloc <extent>
```

Purpose:

Specifies which bounds will receive grids and subgrid lines

Parameters:

Type	Parameter	Description	
constant	extent	Can be any of the following:	
		min	draw the grid/subgrid lines for the minimum axis
		max	draw the grid/subgrid lines for the maximum axis
		both	draw the grid/subgrid lines for both the min and max axis

Example Usage:

```
view: bounds ON
viewport: select_begin
 0
viewport: select_end
viewport_bounds: axis_x_gridextentloc both
viewport_bounds: axis_y_gridextentloc min
viewport_bounds: axis_z_gridextentloc max
```

See Also:

Associated viewport_bounds commands

viewport_bounds: axis_x/y/z_gridtype

Command:

```
viewport_bounds: axis_x_gridtype <grid_type>
viewport_bounds: axis_y_gridtype <grid_type>
viewport_bounds: axis_z_gridtype <grid_type>
```

Purpose:

Set the line type for the x/y/z grid bounds

Parameters:

Type	Parameter	Description	
constant	`grid_type`	Can be any of the following:	
		`none`	specifies no grid lines
		`grid`	specifies a solid grid line
		`tick`	specifies tick marks only

Notes:

If `grid`, the line type is specified via the `viewport_bounds: grid_linetype` command

Example Usage:

```
view: bounds ON
viewport: select_begin
 0
viewport: select_end
viewport_bounds: axis_x_gridtype solid
viewport_bounds: axis_y_gridtype none
viewport_bounds: axis_z_gridtype tick
viewport_bounds: grid_linetype solid
```

See Also:

Associated viewport_bounds commands

Command:

```
viewport_bounds: axis_x_labelaxisloc <label_location>
viewport_bounds: axis_y_labelaxisloc <label_location>
viewport_bounds: axis_z_labelaxisloc <label_location>
```

Purpose:

Specifies the location for the x/y/z axis labels for bounds display

Parameters:

Type	Parameter	Description	
constant	label_location	Can be any of the following:	
		none	do not display labels along this axis
		all	display all of the labels
		beg_end	display only the labels for the first and last grid

Notes:

When label_location is "all", the number of labels is controlled by the viewport_bounds: axis_x/y/z_numgrid command

Example Usage:

```
view: bounds ON
viewport: select_begin
 0
viewport: select_end
viewport_bounds: length as_specified
viewport_bounds: axis_x_min 0.
viewport_bounds: axis_x_max 1.
viewport_bounds: axis_x_numgrid 5
viewport_bounds: axis_x_labelaxisloc beg_end
```

See Also:

Associated viewport_bounds commands

viewport_bounds: axis_x/y/z_labelextentloc

Command:

```
viewport_bounds: axis_x_labelextentloc <extent>
viewport_bounds: axis_y_labelextentloc <extent>
viewport_bounds: axis_z_labelextentloc <extent>
```

Purpose:

Specifies which bounds axis will receive labels

Parameters:

Type	Parameter	Description	
constant	`extent`	Can be any of the following:	
		`min`	label the grid marks for the minimum axis
		`max`	label the grid marks for the maximum axis
		`both`	label both the minimum and maximum axis

Example Usage:

```
view: bounds ON
viewport: select_begin
 0
viewport: select_end
viewport_bounds: axis_x_labelextentloc both
viewport_bounds: axis_y_labelextentloc min
viewport_bounds: axis_z_labelextentloc max
```

See Also:

Associated viewport_bounds commands

Command:

```
viewport_bounds: axis_x_labelrgb <red_val> <grn_val> <blu_val>
viewport_bounds: axis_y_labelrgb <red_val> <grn_val> <blu_val>
viewport_bounds: axis_z_labelrgb <red_val> <grn_val> <blu_val>
```

Purpose:

Set the color for the x/y/z axis labels

Parameters:

Type	Parameter	Description
float	red_val	red, green, blue color components (must be between 0.0 and 1.0)
	grn_val	
	blu_val	

Example Usage:

```
view: bounds ON
viewport: select_begin
 0
viewport: select_end
viewport_bounds: axis_x_labelrgb 4.0000e-01 0.0000e+00 1.0000e+00
```

See Also:

Associated viewport_bounds commands

viewport_bounds: axis_x/y/z_min/max

Command:

```
viewport_bounds: axis_x_min <min>
viewport_bounds: axis_x_max <max>
viewport_bounds: axis_y_min <min>
viewport_bounds: axis_y_max <max>
viewport_bounds: axis_z_min <min>
viewport_bounds: axis_z_max <max>
```

Purpose:

Sets the minimum value for the x/y/z axis

Parameters:

Type	Parameter	Description
float	min	the minimum axis value
	max	the maximum axis value

Notes:

If the length attribute is set to rounded the resulting min/max x/y/z value will be modified from the specified value

Example Usage:

```
view: bounds ON
viewport: select_begin
 0
viewport: select_end
viewport_bounds: axis_linewidth 1
viewport_bounds: axis_rgb 1.0000e+00 0.0000e+00 1.0000e+00
viewport_bounds: length as_specified
viewport_bounds: axis_x_min 0.2
viewport_bounds: axis_x_max 0.6
viewport_bounds: axis_y_min 0.0
viewport_bounds: axis_y_max 1.0
viewport_bounds: axis_z_min -1.
viewport_bounds: axis_z_max 1.
```

See Also:

Associated viewport_bounds commands

Command:

```
viewport_bounds: axis_x_numgrid <num>
viewport_bounds: axis_y_numgrid <num>
viewport_bounds: axis_z_numgrid <num>
```

Purpose:

Set the number of grid marks along the x/y/z axis

Parameters:

Type	Parameter	Description
int	num	the number of grid marks to be used

Notes:

If the length attribute is set to `rounded` the resulting number of grid marks will be modified from the specified value.

Example Usage:

```
view: bounds ON
viewport: select_begin
 0
viewport: select_end
viewport_bounds: axis_linewidth 1
viewport_bounds: axis_rgb 1.0000e+00 0.0000e+00 1.0000e+00
viewport_bounds: length as_specified
viewport_bounds: axis_x_numgrid 5
viewport_bounds: axis_y_numgrid 3
viewport_bounds: axis_z_numgrid 7
```

See Also:

Associated viewport_bounds commands

viewport_bounds: axis_x/y/z_numsubgrid

Command:

```
viewport_bounds: axis_x_numsubgrid <num>
viewport_bounds: axis_y_numsubgrid <num>
viewport_bounds: axis_z_numsubgrid <num>
```

Purpose:

Set the number of subgrid marks along the x/y/z axis

Parameters:

Type	Parameter	Description
int	`num`	the number of subgrid marks to be used

Notes:

If the length attribute is set to `rounded` the resulting number of subgrid marks will be modified from the specified value.

Example Usage:

```
view: bounds ON
viewport: select_begin
 0
viewport: select_end
viewport_bounds: length as_specified
viewport_bounds: axis_x_numsubgrid 2
viewport_bounds: axis_y_numsubgrid 2
viewport_bounds: axis_z_numsubgrid 3
```

See Also:

Associated viewport_bounds commands

Command:

```
viewport_bounds: axis_x_sgridtype <grid_type>
viewport_bounds: axis_y_sgridtype <grid_type>
viewport_bounds: axis_z_sgridtype <grid_type>
```

Purpose:

Set the line type for the x/y/z subgrid bounds

Parameters:

Type	Parameter	Description	
constant	grid_type	Can be any of the following:	
		none	specifies no grid lines
		grid	specifies a solid grid line
		tick	specifies tick marks only

Notes:

If "grid", the line type is specified via the viewport_bounds: subgrid_linetype command

Example Usage:

```
view: bounds ON
viewport: select_begin
 0
viewport: select_end
viewport_bounds: axis_x_sgridtype solid
viewport_bounds: axis_y_sgridtype none
viewport_bounds: axis_z_sgridtype tick
viewport_bounds: subgrid_linetype solid
```

See Also:

Associated viewport_bounds commands

viewport_bounds: dimension

Command:

```
viewport_bounds: dimension <dimension>
```

Purpose:

Set the bounds display to 2D or 3D.

Parameters:

Type	Parameter	Description	
constant	dimension	2D	set the bound display to 2D
		3D	set the bound display to 3D

Notes:

The 2D option is only available for 2D viewports and 3D viewports display in orthographic mode.

Example Usage:

```
#
# turn on bounds display and make the bounds for
# viewport 0 2D
#
view: bounds ON
viewport: select_begin
 0
viewport: select_end
viewport_bounds: dimension 2D
```

See Also:

Associated viewport_bounds commands

Command:

```
viewport_bounds: grid_linetype <line_type>
```

Purpose:

Specifies the type of grid lines that will be drawn for bounds display.

Parameters:

Type	Parameter	Description
constant	line_type	Can be any of the following:
		solid
		dotted
		dash

Example Usage:

```
view: bounds ON
viewport: select_begin
 0
viewport: select_end
viewport_bounds: visible ON
viewport_bounds: grid_linewidth 2
viewport_bounds: grid_linetype dotted
```

See Also:

Associated viewport_bounds commands

viewport_bounds: grid_linewidth

Command:

```
viewport_bounds: grid_linewidth <width>
```

Purpose:

Set the bounds axis grid line width

Parameters:

Type	Parameter	Description
int	`width`	specifies the width in pixels

Example Usage:

```
view: bounds ON
viewport: select_begin
 0
viewport: select_end
viewport_bounds: visible ON
viewport_bounds: grid_linewidth 2
```

See Also:

Associated viewport_bounds commands

Command:

```
viewport_bounds: grid_rgb <red_val> <grn_val> <blu_val>
```

Purpose:

Set the color for the bounds grid lines

Parameters:

Type	Parameter	Description
float	red_val	red, green, blue color components (must be between 0.0 and 1.0)
	grn_val	
	blu_val	

Example Usage:

```
view: bounds ON
viewport: select_begin
 0
viewport: select_end
viewport_bounds: visible ON
viewport_bounds: grid_linewidth 2
viewport_bounds: grid_rgb 0.5 1 0
```

See Also:

Associated viewport_bounds commands

viewport_bounds: length

Command:

```
viewport_bounds: length <size_option>
```

Purpose:

Use min/max information as specified or use round numbers

Parameters:

Type	Parameter	Description	
constant	size_option	as_specified	use the min/max values for the axis and number of tick/subticks as specified
		rounded	use the min/max values specified to produce round numbers. Also round the number of ticks/subticks.

Example Usage:

```
view: bounds ON
viewport: select_begin
 0
viewport: select_end
viewport_bounds: visible ON
viewport_bounds: axis_linewidth 1
viewport_bounds: length as_specified
```

See Also:

Associated viewport_bounds commands

Command:

```
viewport_bounds: subgrid_linetype <line_type>
```

Purpose:

Specifies the type of subgrid lines that will be drawn for bounds display.

Parameters:

Type	Parameter	Description
constant	line_type	Can be any of the following:
		solid
		dotted
		dash

Example Usage:

```
view: bounds ON
viewport: select_begin
 0
viewport: select_end
viewport_bounds: visible ON
viewport_bounds: subgrid_linewidth 2
viewport_bounds: subgrid_linetype dotted
```

See Also:

Associated viewport_bounds commands

viewport_bounds: subgrid_linewidth

Command:

```
viewport_bounds: subgrid_linewidth <width>
```

Purpose:

Set the bounds axis subgrid line width

Parameters:

Type	Parameter	Description
int	`width`	specifies the width in pixels

Example Usage:

```
view: bounds ON
viewport: select_begin
 0
viewport: select_end
viewport_bounds: visible ON
viewport_bounds: subgrid_linewidth 2
```

See Also:

Associated viewport_bounds commands

Command:

```
viewport_bounds: subgrid_rgb <red_val> <grn_val> <blu_val>
```

Purpose:

Set the color for the bounds subgrid lines

Parameters:

Type	Parameter	Description
float	red_val	red, green, blue color components (must be between 0.0 and 1.0)
	grn_val	
	blu_val	

Example Usage:

```
view: bounds ON
viewport: select_begin
 0
viewport: select_end
viewport_bounds: visible ON
viewport_bounds: subgrid_linewidth 2
viewport_bounds: subgrid_rgb 0.5 0.5 0.5
```

See Also:

Associated viewport_bounds commands

viewport_bounds: transparency

Command:

```
viewport_bounds: transparency <value>
```

Purpose:

Sets the opaqueness for the bounds display

Parameters:

Type	Parameter	Description	
float	value	opaqueness value	0 = completely transparent
			1 = completely opaque

Example Usage:

```
view: bounds ON
viewport: select_begin
 0
viewport: select_end
viewport_bounds: transparency 0.5
```

See Also:

Associated viewport_bounds commands

Command:

```
viewport_bounds: visible <toggle>
```

Purpose:

Turn on bounds display for a viewport

Parameters:

Type	Parameter	Description	
constant	toggle	ON	turn bounds display on
		OFF	turn bounds display off

Example Usage:

```
#
# turn on bounds display but turn the bounds off for viewport 1
#
view: bounds ON
viewport: select_begin
 1
viewport: select_end
viewport_bounds: visible OFF
```

See Also:

Associated viewport_bounds commands

Commands in this Section:

vortexcore

vortexcore: begin
vortexcore: end
vortexcore: create
vortexcore: density <variable_name>
vortexcore: method <algorithm>
vortexcore: momentum <variable_name>
vortexcore: ratio_of_spec_heat <name or value>
vortexcore: select_default
vortexcore: threshold <variable_name>
vortexcore: threshold_filter <option>
vortexcore: threshold_max <threshold_value>
vortexcore: threshold_min <threshold_value>
vortexcore: threshold_value <threshold_value>
vortexcore: velocity <variable_name>

See Also:

User Manual - Section 7.19, Vortex Core Create/Update

How To Extract Vortex Cores

vortexcore: begin/end

Command:

```
vortexcore: begin
vortexcore: end
```

Purpose:

Delimit the modifications for vortexcore attributes.

Parameters:

None.

Notes:

These commands are used to set the attributes for a vortex core creation.

Example Usage:

```
#
# Specify the 3D parent part(s)
# from which to create the vortex cores
#
part: select_begin
 1
part: select_end
#
# Compute the vortex cores part
#
vortexcore: begin
vortexcore: end
vortexcore: create
#
# Now select the newly created vortex part
# for subsequent operations
#
part: select_begin
 5
part: select_end
```

See Also:

Associated vortexcore commands

Command:

```
vortexcore: create
```

Purpose:

Create a vortex core part with the currently defined attributes and the selected 3D parent parts.

Parameters:

None.

Example Usage:

```
#
# Specify the 3D parent part(s)
# from which to create the vortex cores
#
part: select_begin
 1
part: select_end
#
# Compute the vortex cores part
#
vortexcore: begin
vortexcore: end
vortexcore: create
#
# Now select the newly created vortex part
# for subsequent operations
#
part: select_begin
 5
part: select_end
```

See Also:

Associated vortexcore commands

vortexcore: density

Command:

```
vortexcore: density <variable_name>
```

Purpose:

Specify the density variable to use in creating/updating a vortex core part or the default attributes.

Parameters:

Type	Parameter	Description
string	`variable_name`	the name of the density variable (the default `<none>` means to use nothing)

Notes:

This variable is assigned automatically if the variable name `<Density>` exists.

Since the computation of separation and attachment lines depend on velocity, either the velocity variable must be defined or both the density and momentum variables must be defined to obtain the velocity variable.

Example Usage:

```
variables: activate Density
vortexcore: select_default
part: modify_begin
vortexcore: density Density
part: modify_end
```

See Also:

Associated vortexcore commands

Command:

```
vortexcore: method <algorithm>
```

Purpose:

Specify the algorithm to use to compute the vortex core parts.

Parameters:

Type	Parameter	Description	
constant	algorithm	vorticity	vortex cores computed when the vorticity and velocity vectors are coincident
		eigen_analysis	vortex cores computed when the appropriate eigen-analysis indicates the element is intersected by swirling flow (default)

Example Usage:

```
part: select_begin
 5
part: select_end
part: modify_begin
vortexcore: method vorticity
part: modify_end
```

See Also:

Associated vortexcore commands

vortexcore: momentum

Command:

```
vortexcore: momentum <variable_name>
```

Purpose:

Specify the momentum variable to use in creating/updating a vortex core part or the default attributes.

Parameters:

Type	Parameter	Description
string	`variable_name`	name of the momentum variable (the default name `<none>` means to use nothing)

Notes:

This variable is assigned automatically if the variable name `<Momentum>` exists.

Since the computation of separation and attachment lines depend on velocity, either the velocity variable must be defined or both the density and momentum variables must be defined to obtain the velocity variable.

Example Usage:

```
variables: activate Momentum
vortexcore: select_default
part: modify_begin
vortexcore: momentum Momentum
part: modify_end
```

See Also:

Associated vortexcore commands

Command:

```
vortexcore: ratio_of_spec_heat <name or value>
```

Purpose:

Specify the ratio of specific heat constant value or variable name to use in creating/updating a vortex core part or the default attributes.

Parameters:

Type	Parameter	Description
string	`name`	scalar field variable name for the ratio of specific heats
float	`value`	constant value (default = 1.4) for the ratio of specific heats

Example Usage:

```
vortexcore: select_default
part: modify_begin
vortexcore: ratio_of_spec_heat 1.4
part: modify_end
```

See Also:

Associated vortexcore commands

vortexcore: select_default

Command:

```
vortexcore: select_default
```

Purpose:

Specify that the default attributes of the vortex core part are to be set for all subsequent vortex core creates and updates.

Parameters:

None.

Example Usage:

```
vortexcore: select_default
part: modify_begin
vortexcore: momentum none
part: modify_end
```

See Also:

Associated vortexcore commands

Command:

```
vortexcore: threshold <variable_name>
```

Purpose:

Specify the threshold variable description by which to filter the segments of the selected vortex cores part.

Parameters:

Type	Parameter	Description
string	`variable_name`	name of the scalar variable by which to filter the vortex core segments

Notes:

The default threshold variable for vortex cores is "`fx_vortcore_streng`".

Example Usage:

```
part: select_begin
 5
part: select_end
part: modify_begin
vortexcore: threshold Density
part: modify_end
```

See Also:

Associated vortexcore commands

vortexcore: threshold_filter

Command:

```
vortexcore: threshold_filter <option>
```

Purpose:

Specify how to filter out segments of the specified vortex cores part, according to the specified threshold variable and value.

Parameters:

Type	Parameter	Description	
constant	option	<	(default) filter out all values of the specified threshold variable less than the specified threshold value
		>	filter out all values of the specified threshold variable greater than the specified threshold value

Example Usage:

```
part: select_begin
 5
part: select_end
part: modify_begin
vortexcore: threshold_filter >
part: modify_end
```

See Also:

Associated vortexcore commands

Command:

```
vortexcore: threshold_max <threshold_value>
```

Purpose:

Specify the maximum threshold limit under which to filter the threshold value of the selected vortex cores part(s).

Parameters:

Type	Parameter	Description
float	`threshold_value`	maximum threshold variable range limit below which to threshold the vortex cores part

Notes:

This value must be greater than the `threshold_min` value, and be a valid threshold variable value. This value is initially automatically adjusted usually to an order of magnitude above the threshold value.

Example Usage:

```
part: select_begin
 5
part: select_end
part: modify_begin
vortexcore: threshold_max 1.0000e+01
part: modify_end
```

See Also:

Associated vortexcore commands

vortexcore: threshold_min

Command:

```
vortexcore: threshold_min <threshold_value>
```

Purpose:

Specify the minimum threshold limit above which to filter the threshold value of the selected vortex cores part(s).

Parameters:

Type	Parameter	Description
float	`threshold_value`	minimum threshold variable range limit above which to threshold the vortex cores part

Notes:

This value must be less than the `threshold_max` value, and be a valid threshold variable value. This value is initially adjusted to the threshold value.

Example Usage:

```
part: select_begin
 5
part: select_end
part: modify_begin
vortexcore: threshold_min 1.0000e-01
part: modify_end
```

See Also:

Associated vortexcore commands

Command:

```
vortexcore: threshold_value <threshold_value>
```

Purpose:

Specify the threshold value by which to filter the segments from the selected vortex cores part(s).

Parameters:

Type	Parameter	Description
float	`threshold_value`	variable value by which to apply the threshold filter to the vortex core segments

Notes:

This value is automatically set to the minimum vortex core strength variable value of the vortex cores part. This value must lie within the threshold variable interval [`threshold_min,theshold_max`].

Example Usage:

```
part: select_begin
 5
part: select_end
part: modify_begin
vortexcore: threshold_value 5.0000e-01
part: modify_end
```

See Also:

Associated vortexcore commands

vortexcore: velocity

Command:

```
vortexcore: velocity <variable_name>
```

Purpose:

Specify the velocity variable to use in creating/updating a vortex core part or the default attributes.

Parameters:

Type	Parameter	Description
string	`variable_name`	name of the velocity variable (the default `<none>` means to use nothing)

Notes:

This variable is automatically assigned if the variable name `<velocity>` exists.

Since the computation of separation and attachment lines depend on velocity, either the velocity variable must be defined or both the density and momentum variables must be defined to obtain the velocity variable.

Example Usage:

```
variables: activate Velocity
vortexcore: select_default
part: modify_begin
vortexcore: velocity Velocity
part: modify_end
```

See Also:

Associated vortexcore commands

A

Index

Index

Index

Index