CHAPTER 26

The GREPLAY Procedure

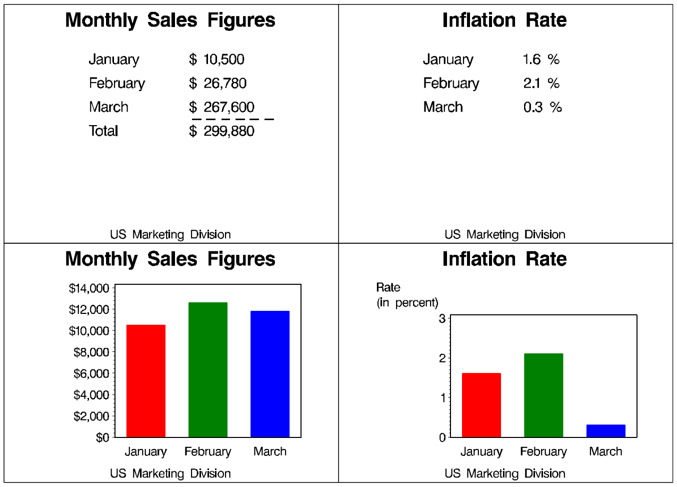
Overview

The GREPLAY procedure displays and manages graphics output that is stored in SAS catalogs. The GREPLAY procedure also creates templates and color maps that you can use when you replay your graphics output. The GREPLAY procedure operates in both windowing and line-mode environments. With the GREPLAY procedure, you can

- select one or more catalog entries from the same catalog for replay and route them to your display or other devices, such as plotters and printers.
- use, create, or modify templates. You can use templates to describe positioning on a single display for the graphics output that is stored in one or more catalog entries.
- use, create, or modify color maps. Color maps enable you to change the colors in graphics output by mapping existing colors to new colors.
- manage entries in SAS catalogs by
  - creating logical groupings of catalog entries that contain graphics output
  - renaming, deleting, or copying catalog entries that contain graphics output, templates, and color maps
  - rearranging catalog entries that contain graphics output.
- create new graphics output by replaying one or more catalog entries into panels within a template.

Figure 26.1 on page 921 shows four catalog entries that were replayed into a template and displayed as a single graph.

Figure 26.1 Graphics Output in a Template



For an example of replaying graphics in a template, see Example 2 on page 954.

Concepts

About Catalog Entries

The GREPLAY procedure uses three kinds of catalog entries:

graphics output
: catalog entries of type GRSEG. The catalog in which these entries are stored is referred to as the input-catalog and output-catalog. The *input-catalog* is the catalog that contains the graphics output, stored in catalog entries, that you want to replay. You can change the input catalog during a catalog management session. The *output-catalog* is the catalog in which graphics output that is produced by the template facility is stored. The output catalog is also the destination of copied catalog entries.

templates
: catalog entries of type TEMPLATE. This catalog is referred to as the template-catalog. The *template-catalog* is the catalog that stores templates that are created by the GREPLAY procedure. The template catalog also may contain previously created templates that you want to modify or templates to use for replaying your graphics output. SASHELP.TEMPLT is the Institute-supplied template catalog.

color maps
: catalog entries of type CMAP. This catalog is referred to as color-map-catalog. The *color-map-catalog* is the catalog that stores color maps that are created by the GREPLAY procedure. The color map catalog also may contain previously created color maps that you want to modify or color maps to use when you replay your graphics output. Note that image entries may exist in this catalog but are not recognized by the GREPLAY procedure.

Note: Image entries may exist in the catalog but are not recognized by PROC GREPLAY. △

You can store all of the previous entry types in a single SAS catalog, or you can store them in separate catalogs and use a different catalog for each type of entry. A single SAS catalog may contain graphics output, color maps, and templates.

Because the GREPLAY procedure operates on catalog entries, you must assign at least one catalog before you can perform any tasks. The GREPLAY procedure has several ways to assign the catalogs, as shown in Table 26.1 on page 922.

Table 26.1 Assigning Catalogs

Catalog	Ways to Assign
input	IGOUT= option in the PROC GREPLAY statement
	IGOUT statement
	IGOUT field in the PROC GREPLAY window
output	GOUT= option in the PROC GREPLAY statement
	GOUT statement
	GOUT field in the PROC GREPLAY window
template	TC= option in the PROC GREPLAY statement
	TC statement
	TC field in the PROC GREPLAY window
color map	CC= option in the PROC GREPLAY statement
	CC statement
	CC field in the PROC GREPLAY window

In addition, you can assign a current template, which you can use when you replay graphics output, and a current color map, which you can use to remap colors when you replay graphics output. To assign the current template, use one of the following:

- □ the TEMPLATE= option in the PROC GREPLAY statement
- □ the TEMPLATE statement
- □ the Template field in the PROC GREPLAY window.

To assign the current color map, use one of the following:

- □ the CMAP= option in the PROC GREPLAY statement
- □ the CMAP statement
- □ the Cmap field in the PROC GREPLAY window.

Duplicate Entry Names

If you try to create a catalog entry with the same name as an existing entry, the GREPLAY procedure uses the following naming conventions to prevent duplication of the name.

- □ For names that are fewer than eight characters, the procedure adds a number to the end of the name. For example, if you copy an entry that is named PLOT to a catalog that already contains an entry with that name, the procedure assigns the name PLOT1 to the new copy.
- □ For names that are eight characters long, the procedure drops the last character from the name before it adds the suffix. For example, if you copy an entry

TITLEONE to a catalog that already contains an entry with that name, the procedure assigns the name TITLEON1 to the copied entry.

□ Template entries that contain individual entries will reserve the individual entry names as well as the template entry name.

The GREPLAY procedure uses the same technique for the names of entries that contain graphics output that is produced by the template facility.

Ways to Use the GREPLAY Procedure

You can use the GREPLAY procedure to replay or manage catalog entries in two different ways:

□ by browsing or editing the fields in the GREPLAY procedure windows (see Figure 26.2 on page 946)

□ by submitting code-based GREPLAY procedure statements (see "Code-based Statements" on page 923).

If you are running SAS software in a nonwindowing environment (such as line mode or batch), you can only submit code-based GREPLAY statements. However, if you are running SAS software in a windowing environment (the SAS Display Manager System, for example), you can use either the GREPLAY windows or the GREPLAY statements.

If your device supports a windowing environment, the GREPLAY procedure automatically opens the GREPLAY procedure windows. Otherwise, the GREPLAY procedure expects you to submit GREPLAY procedure statements.

Windowing Environment

To invoke the GREPLAY windows, submit the PROC GREPLAY statement without the NOFS option, as follows:

```
proc greplay;
run;
```

SAS/GRAPH then opens the PROC GREPLAY window. For more information, see "Using the GREPLAY Procedure" on page 945.

If you are in a windowing environment, you can switch between the windows and code-based statements while you run the procedure. See the "FS Statement" on page 934 and the NOFS window command in the SAS Help facility.

Code-based Statements

If you do not use the GREPLAY windows, you can use code-based statements to replay or manage the catalog entries. The GREPLAY procedure automatically uses code-based statements if you do not have a windowing device or if you are running the GREPLAY procedure in a batch environment. To use the GREPLAY procedure with code-based statements on a windowing device, submit the PROC GREPLAY statement with the NOFS option as follows:

```
proc greplay nofs;
```

Once you submit the PROC GREPLAY statement, you can enter and submit statements and run them without re-entering the PROC GREPLAY statement.

You can exit the GREPLAY procedure with code-based statements in two ways:

□ submit the END, QUIT, or STOP statement

□ submit another PROC statement or DATA step.

Procedure Syntax

Requirements: Use statements other than the PROC GREPLAY statement only in a nonwindowing or batch environment, or with the NOFS option. In these environments at least one additional statement is required.

Note: You must have write access to a catalog in order to modify, add, or delete device entries. Only GRSEG entry types may be replayed with the GREPLAY procedure.

Supports: RUN-group processing Output Delivery System (ODS)

PROC GREPLAY <BYLINE>
  <CC=*color-map-catalog*>
  <CMAP=*color-map-entry*>
  <FS>
  <GOUT=<*libref.*>*output-catalog*>
  <IGOUT=<*libref.*>*input-catalog*>
  <IMAGEMAP=*output-data-set*>
  <NOBYLINE>
  <NOFS>
  <PRESENTATION>
  <TC=*template-catalog*>
  <TEMPLATE=*template-entry*>;

? *required –argument*;

BYLINE;

CC *color-map-catalog*;

CCOPY <*color-map-catalog.*>*color-map-entry*<.CMAP>;

CDEF *color-map-entry*
  <*color-definition(s)*>
  <DES='*entry-description*'>;

CDELETE *color-map-entry(s)* | _ALL_;

CMAP *color-map-entry*;

COPY *entry-id(s)* | _ALL_;

DELETE *entry-id(s)* | _ALL_;

DEVICE *device-name*;

FS;

GOUT <*libref.*>*output-catalog*;

GROUP *entry-id(s)*;

IGOUT <*libref.*>*input-catalog* ;

LIST *required-argument*;

MODIFY *modify-pair(s)*;

MOVE *entry-id-1* AFTER | BEFORE *entry-id-2*;

NOBYLINE;

PREVIEW *template-entry(s)* | _ALL_ ;

QUIT | END | STOP;

REPLAY *entry-id(s)* | _FIRST_ | _LAST_ | _ALL_ ;

TC *template-catalog*;

TCOPY <*template-catalog.*>*template-entry*<.TEMPLATE>;

TDEF *template-entry*

<*panel definition(s)*>
<DES='*entry-description*'>;
TDELETE *template-entry(s)* | _ALL_ ;
TEMPLATE *template-entry*;
TREPLAY *select-pair(s)*;

PROC GREPLAY Statement

Determines whether the procedure starts in a windowing or nonwindowing environment, and whether the session is used for catalog management or output presentation.

Syntax

PROC GREPLAY <BYLINE>
<CC=*color-map-catalog*>
<CMAP=*color-map-entry*>
<FS>
<GOUT=<*libref.*>*output-catalog*>
<IGOUT=<*libref.*>*input-catalog*>
<IMAGEMAP=*output-data-set*>
<NOBYLINE>
<NOFS>
<PRESENTATION>
<TC=*template-catalog*>
<TEMPLATE=*template-entry*>;

Options

Each PROC GREPLAY statement option has an equivalent statement that you can use instead.

BYLINE

specifies that the BY statement information for the SAS catalog entries should be displayed. The BY statement information appears directly beneath the primary description of the entry. By default, the BY statement information is displayed.

CC=*color-map-catalog*

identifies the color map catalog to be used with the GREPLAY procedure. Use the CMAP= option to assign a current color map that is contained in *color-map-catalog*.

To assign a current color map or create new color maps, you must assign a color map catalog with the CC= option.

To replay graphics output using a color map, you must assign a color map catalog and a current color map with the CC= and CMAP= options.

Featured in: Example 3 on page 956

CMAP=*color-map-entry*

assigns a current color map to use when replaying graphics output, where *color-map-entry* names an existing color map in the catalog specified in the CC= option. If *color-map-entry* is not in the catalog, an error message is written to the SAS log. *Color-map-entry* must have a catalog entry type of CMAP.

If you do not specify a color map catalog using the CC= option when using the CMAP= option, a warning message is written to the SAS log.

To replay graphics output using a color map, you must assign a color map catalog and a current color map with the CC= and CMAP= options.

FS

specifies that the GREPLAY procedure should use windows. By default, if your device supports windows, the GREPLAY procedure uses windows. If your device does not support windows, the procedure begins execution in line-mode and the FS option has no effect.

GOUT=<*libref.*>*output-catalog*

specifies the SAS catalog in which to save the graphics output that is produced by the GREPLAY procedure. In addition, catalog entries that contain graphics output can be copied to *output-catalog*. If you omit the libref, SAS/GRAPH looks for the catalog in the temporary library called WORK and creates the catalog if it does not exist. *Output-catalog* can be the same catalog that is specified in the IGOUT= option.

To copy catalog entries, you must assign an input and, optionally, an output catalog with the IGOUT= and GOUT= options.

See also: "Storing Graphics Output in SAS Catalogs" on page 49

Featured in: Example 2 on page 954

IGOUT=<*libref.*>*input-catalog*

specifies the input catalog to use with the GREPLAY procedure. The input catalog that you specify with the IGOUT= option should be a catalog that contains the graphics output that will be replayed. If you omit the libref, SAS/GRAPH looks for the catalog in the temporary library called WORK. *Input-catalog* can be the same catalog that you specified in the GOUT= option.

To move, group, or delete catalog entries or to replay graphics output, you must assign an input catalog with the IGOUT= option.

To copy catalog entries, you must assign an input and, optionally, an output catalog with the IGOUT= and GOUT= options.

Featured in: Example 2 on page 954

IMAGEMAP=*output-data-set*

must be used in conjuntion with the REPLAY statement (see "REPLAY Statement" on page 939). Creates a SAS data set that contains information about the graph that is replayed from the graphics catalog and about areas in that graph. This information includes the shape and coordinates of the areas, and is used to build an HTML file that links the graph areas to other files or images. This linking provides drill-down functionality on the graph. The Imagemap data set also contains the information that is stored in the HTML variable in the data set that generated the original graph. Therefore, in order to use IMAGEMAP= to create an HTML file, the procedure that originally stored the graph in the graphics catalog must have used an HTML variable in the data set that was used to generate the graph, and it must also have specified HTML=, or HTML_LEGEND=, or both.

See also: "Customizing Web Pages for Drill-down Graphs" on page 100

NOBYLINE

suppresses the BY statement information for the SAS catalog entries. The BY statement information appears directly beneath the primary description of the entry. By default, the BY statement information is displayed.

NOFS

specifies that the GREPLAY procedure should use line mode. By default, if your device supports windows, the GREPLAY procedure uses windows. If your device does not support windows, the procedure uses line mode, regardless of whether you used the FS option or the NOFS option.

Featured in: Example 1 on page 952

PRESENTATION

specifies that the GREPLAY procedure should open the PRESENTATION window and use the catalog specified by the IGOUT= option as the input catalog. The

PRESENTATION option is often used in applications to prevent the application users from deleting or reordering the catalog entries. You can only replay graphics output from the PRESENTATION window; you cannot manage catalogs or create templates and color maps from this window.

You must use the IGOUT= option when you use the PRESENTATION option. The PRESENTATION option overrides the NOFS option on full-screen devices.

TC=*template-catalog*

identifies the template catalog to use with the GREPLAY procedure. Use the TEMPLATE= option to assign a current template from *template-catalog*.

To assign a current template or create new templates, you must assign a template catalog with the TC= option.

To replay graphics output in a template, you must assign a template catalog and a current template with the TC= and TEMPLATE= options.

Featured in: Example 1 on page 952

TEMPLATE=*template-entry*

assigns a current template to use when replaying graphics output where *template-entry* names an existing template in the template catalog that is specified in the TC= option. If *template-entry* is not in the catalog, an error message is written to the SAS log. *Template-entry* must have a catalog entry type of TEMPLATE.

When you use the TEMPLATE= option, you must also specify the name of a template catalog with the TC= option. Otherwise, a warning message is written to the SAS log.

Featured in: Example 2 on page 954

Details

When you submit the PROC GREPLAY statement, the mode of operation depends on both the environment in which the statement is submitted and whether the NOFS option is included, as shown in Table 26.2 on page 927.

Table 26.2 Ways of Invoking the GREPLAY Procedure

Environment	Statement	Result
windowing	PROC GREPLAY;	GREPLAY procedure windows
windowing	PROC GREPLAY NOFS;	line mode
nonwindowing	PROC GREPLAY;	line mode

You can switch back and forth between windows and line-mode within a session.

? Statement

Prints the current value of certain PROC GREPLAY options or of the current device driver.

Procedure output: Output is sent to the SAS log.

Syntax

? *required–argument*;

required-argument must be one of the following:

CC

CMAP

DEVICE

GOUT

IGOUT

TC

TEMPLATE

Required Arguments

CC
prints the name of the current color map catalog. If no color map catalog has been assigned, the GREPLAY procedure issues a message.

CMAP
prints the name of the current color map. If no color map has been assigned, the GREPLAY procedure issues a message.

DEVICE
DEV
prints the name of the current device driver.

GOUT
prints the name of the current output catalog. If you did not assign an output catalog, the GREPLAY procedure issues a message.

IGOUT
prints the name of the current input catalog. If you did not assign an input catalog, the GREPLAY procedure issues a message.

TC
prints the name of the current template catalog. If you did not assign a template catalog, the GREPLAY procedure issues a message.

TEMPLATE
prints the name of the current template. If you did not assign a template, the GREPLAY procedure issues a message.

BYLINE Statement

Displays BY statement information directly beneath the primary description of the catalog entries when you list the contents of the input catalog.

Note: BY statement information is displayed by default.

See also: NOBYLINE statement

Syntax

BYLINE;

CC Statement

Specifies a color map catalog and allows you to change the color map catalog without exiting the procedure.

Syntax

CC *color-map-catalog*;

Required Arguments

color-map-catalog
identifies the SAS catalog where color maps should be stored or the name of a SAS catalog containing color maps.

CCOPY Statement

Copies a color map from another catalog to the color map catalog or creates a duplicate copy of a color map within the color map catalog.

Requirements: Assign a color map catalog before using the CCOPY statement.

See also: CC statement

Syntax

CCOPY <*color-map-catalog.*>*color-map-entry*<.CMAP>;

Required Arguments

***<color-map-catalog.>color-map-entry* <.CMAP>**
identifies the color map entry to be copied.

color-map-catalog
is the SAS catalog that contains the color map to be copied.

color-map-entry
is the name of the entry color map.

CMAP
is the catalog entry type.

If a color map of the same name already exists in the color map catalog, the GREPLAY procedure creates a new name.

See also: “Duplicate Entry Names” on page 922

Details

To copy a color map from another catalog to the color map catalog, use the CC statement to specify *color-map-catalog* as the catalog from which the color map should be copied. For example, the following statements copy HP.CMAP from the catalog named ONE.CCAT to the catalog named TARGET.CLRMAP:

```
libname target 'SAS-data-library';
libname one 'SAS-data-library';

proc greplay nofs;
   cc target.clrmap;
   ccopy one.ccat.hp.cmap;
quit;
```

To create a duplicate copy of a color map, simply omit *color-map-catalog* from your CCOPY statement. For example, to create a duplicate copy of the color map named HP.CMAP in the color map catalog, use the following statement:

```
ccopy hp.cmap;
```

CDEF Statement

Defines or modifies a color map in the color map catalog.

Requirements: Assign a color map catalog before using the CDEF statement.

See also: CC statement

Featured in: Example 3 on page 956

Syntax

CDEF *color-map-entry*
<*color-definition(s)*>
<DES=’*entry-description*’>;

color-definition has the following form:

color-number / *from-color:to-color*

color-definition has the following form: *color-number* / *from-color:to-color*

Required Arguments

color-map-entry
identifies an existing or new color map. *Color-map-entry* is the name of a catalog entry.

If the color map name is not in the color map catalog, then the procedure creates a new color map. If the color map name is already in the color map catalog, then the procedure modifies or adds to that color map.

Options

color-number / from-color:to-color
specifies a color pair and how it is defined.

color-number
specifies the number of a color pair.

from-color:to-color
defines the colors that are being mapped:

from-color
is the color to be mapped.

to-color
is the new color that replaces *from-color* in the replayed graphics output.

DES='*entry-description*'
specifies a description of the catalog entry for the color map. The maximum length for the *entry-description* is 40 characters. By default, the GREPLAY procedure assigns a description of **** NEW COLOR MAP **** to the color map.

CDELETE Statement

Deletes one or more color maps from the current color map catalog.

Caution: The GREPLAY procedure does not prompt you to confirm your request to delete color maps.

Alias: CDEL

Syntax

CDELETE *color-map-entry(s)* | _ALL_ ;

Required Arguments

color-map-entry(s)
identifies one or more color maps that you want to delete from the color map catalog. You can submit a single entry or a list of entries in one CDELETE statement.

ALL
deletes all of the color maps from the color map catalog.

CMAP Statement

Assigns the current color map to be used when replaying graphics output.

Requirements: Assign a color map catalog before using the CMAP statement.

See also: CC statement

Featured in: Example 3 on page 956

Syntax

CMAP *color-map-entry*;

Required Arguments

color-map-entry
identifies an existing color map, contained in the color map catalog, to use when replaying your graphics output. If the color map is not in the current color map catalog, the GREPLAY procedure issues an error message in the SAS log.

COPY Statement

Copies one or more catalog entries containing graphics output from the input catalog to the output catalog.

Requirements: Assign an input catalog and an output catalog before using the COPY statement.

Note: You cannot use the COPY statement to create a duplicate of an entry containing graphics output in the same catalog. You can have only one copy of an entry containing graphics output in a catalog.

See also: GOUT and IGOUT statements

Syntax

COPY *entry-id(s)* | _ALL_ ;

Required Arguments

One of the following is required:

entry-id(s)
is the number or name of a catalog entry, or the number or name of a group of entries to be copied from the input catalog to the output catalog. Entries must contain graphics output. Multiple *entry-id(s)* can contain both numbers and names.

ALL
copies all of the graphics output entries in the input catalog to the output catalog.

DELETE Statement

Deletes SAS catalog entries containing graphics output from the current input catalog.

Caution: The GREPLAY procedure does not prompt you to confirm your request to delete an entry containing graphics output.

Alias: DEL

Syntax

DELETE *entry-id(s)* | _ALL_ ;

Required Arguments

One of the following is required:

entry-id(s)
is the number or name of a catalog entry, or the number or name of a group of entries to be deleted from the input catalog. Entries must contain graphics output. Multiple *entry-id(s)* can contain both numbers and names.

ALL
deletes all of the graphics output entries in the input catalog.

DEVICE Statement

Specifies the device driver.

Requirements: You must specify a device driver that your graphics device can support and that is available in your SAS session.

Alias: DEV

Syntax

DEVICE *device-name*;

Required Arguments

device-name
specifies the device driver to use when you replay graphics output. The device driver that you specify becomes the current device and is used for subsequent replays until you submit another DEVICE statement or change the device driver in another way.

FS Statement

Switches from line mode to the GREPLAY procedure windows.

Requirements: Your device must support windows.

See also: NOFS on page 926

Syntax

FS;

GOUT Statement

Assigns the current output catalog used by the GREPLAY procedure.

Note: You may change the output catalog without exiting the procedure by using the GOUT statement.

Syntax

GOUT *<libref.>output-catalog*;

Required Arguments

<libref.>output-catalog
identifies the SAS catalog that you want to use as an output catalog. By default, the output catalog is WORK.GSEG.

GROUP Statement

Creates groups of entries in the current input catalog.

Syntax

GROUP *entry-id(s)*;

Required Arguments

entry-id(s)
is the number or name of a catalog entry that contains graphics output. All of the entries that are specified in the GROUP statement are included in a single group

with a group header. You can submit a single entry or a list of entries with a single GROUP statement. A list of entries can contain both entry numbers and entry names.

Details

You can manage and display groups of entries with the DELETE, COPY, and REPLAY statements in the same way that you manage single entries.

Only one group can be created per group statement. The default name for a group header is GROUP. The default description for the group header is *** new group ***. The GREPLAY procedure uses a naming convention to avoid duplicate names. See "Duplicate Entry Names" on page 922 for more information on the naming convention.

To change the name (and description) of a group, use the MODIFY statement.

IGOUT Statement

Assigns the current input catalog used by the GREPLAY procedure.

Note: You may change the input catalog without exiting the procedure by using the IGOUT statement.

Syntax

IGOUT *<libref.>input-catalog*;

Required Arguments

<libref.>input-catalog
identifies the SAS catalog with entries that contain graphics output that you want to replay.

LIST Statement

Prints entries in the input, template, and color map catalogs, as well as the contents of templates and color maps.

Procedure output: The output from the LIST statement is sent to the SAS log.

Note: Entries are listed in the order of their creation date.

Featured in: Example 3 on page 956

Syntax

LIST *required-argument*;

required-argument must be one of the following:

CC

CMAP

IGOUT
TC
TEMPLATE

Required Arguments

One of the following is required:

CC
prints the color maps that are in the current color map catalog. If the catalog contains both templates and color maps, only color maps are listed.

CMAP
prints the *From* and *To* color values in the current color map.

IGOUT
prints the number, names, and descriptions of the entries in the input catalog that contains graphics output. In addition, the type of graphics output (dependent or independent) is shown.

TC
prints the templates in the current template catalog. If the catalog contains both templates and color maps, only the templates are listed.

TEMPLATE
prints the panel definition values of the current template.

MODIFY Statement

Changes the name, description, and BY statement information of entries or group headers in the input catalog.

Syntax

MODIFY *modify-pair(s)*;

modify-pair(s) has the following form:

entry-id / *entry-description(s)*

Required Arguments

entry-id / entry-description(s)
specifies the entry to modify.

entry-id
is the number or name of a catalog entry, or the number or name of a group of entries in the input catalog. Entries must contain graphics output. Multiple *entry-id(s)* can contain both numbers and names.

entry-description(s)
must be at least one of the following:

BYLINE='*character-string*'
specifies a character string that can be used for additional information or for BY statement information. *Character-string* can be up to 40 characters long and

must be enclosed in quotation marks. BY statement information appears directly beneath the primary description of the catalog entry.

NAME=’*entry-name*’
specifies the new name of the catalog entry for the graph. The maximum length for *entry-name* is eight characters. If the specified name duplicates the name of an existing entry, SAS/GRAPH software adds a number to the duplicate name to create a unique entry.

Note: The value for *entry-name* can be either with or without quotation marks. △

DES=’*entry-description*’
specifies the description of the catalog entry for the graph. The maximum length for *entry-description* is 40 characters. The description does not appear on the graph.

MOVE Statement

Rearranges entries in the input catalog by moving entries either before or after other entries.

Syntax

MOVE *entry-id-1* AFTER | BEFORE *entry-id-2*;

Required Arguments

entry-id-1
is the name or number of an existing catalog entry or a group header that is to be moved.

entry-id-2
is the name or number of an existing catalog entry or a group header. *Entry-id-1* can be placed before or after *entry-id-2*.

AFTER | BEFORE
specifies whether *entry-id-1* should be moved before or after *entry-id-2*.

Details

To move an entire group, use the name of the group for *entry-id-1*. To move an entry into a group, move the entry after a group header or before or after an entry in the group. For example, this statement moves the entry CHART3 into the group that is named NEW_SALES:

```
move chart3 after new_sales;
```

NOBYLINE Statement

Suppresses BY statement information.

Note: By default, the BY statement information is displayed.

See also: BYLINE statement

Syntax

NOBYLINE;

PREVIEW Statement

Displays the panel outlines for one or more templates using the current device. Use the TC statement to specify the template catalog before using the PREVIEW statement.

Tip: When you preview a list of templates, press END or ENTER to preview the next template in the list.

Note: The graphics output produced when you preview a template is stored in a catalog named WORK.GTEM, which is deleted at the end of your session.

Syntax

PREVIEW *template-entry(s)* | _ALL_ ;

Required Arguments

One of the following is required:

template-entry(s)
identifies one or more template entries that are contained in the current template catalog. You can preview one entry or a list of entries with one PREVIEW statement.

ALL
previews all of the templates in the current template catalog.

QUIT Statement

Exits the GREPLAY procedure.

Aliases: END, STOP

Syntax

QUIT;

REPLAY Statement

Selects one or more entries for replay from the current input catalog.

Note: If any entries specified in a REPLAY statement are not found in the input catalog, PROC GREPLAY issues a message in the SAS log and continues to replay valid entries.

Alias: PLAY

Syntax

REPLAY *entry-id(s)* | _FIRST_ | _LAST_ | _ALL_ ;

Required Arguments

One of the following is required:

entry-id(s)
is the number or name of a catalog entry, or the number or name of a group of entries in the input catalog. Entries must contain graphics output. Multiple *entry-id(s)* can contain both numbers and names. For example, this statement specifies both the entry named GRAPH and the third entry in the catalog:

```
replay graph 3;
```

ALL
replays all of the entries in the input catalog.

FIRST
replays the first entry in the input catalog.

LAST
replays the last entry in the input catalog.

TC Statement

Specifies the template catalog for the GREPLAY procedure.

Note: SASHELP.TEMPLT is the Institute-supplied template catalog.

Tip: Use the TC statement to change the template catalog without exiting the procedure.

Syntax

TC *template-catalog*;

Required Arguments

template-catalog
identifies the SAS catalog where templates are to be stored or identifies the name of a SAS catalog that contains templates.

TCOPY Statement

Copies templates from another catalog to the template catalog or creates a duplicate copy of a template within the template catalog.

Requirements: Assign a template catalog before using the TCOPY statement.

See also: TC statement

Syntax

TCOPY <*template-catalog.*>*template-entry*<.TEMPLATE>;

Required Arguments

<*template-catalog.*>*template-entry*<.TEMPLATE>
identifies the template entry that is to be copied.

template-catalog
is the SAS catalog that contains the template that is to be copied.

template-entry
is the template entry name.

TEMPLATE
is the catalog entry type. If a template of the same name already exists in the template catalog, the GREPLAY procedure creates a new name.

See also: "Duplicate Entry Names" on page 922

Details

To copy a template from another catalog to the template catalog, specify *template-catalog* as the catalog from which the template should be copied. For example, if you want to copy NEWTEMP.TEMPLATE from the catalog named ONE.TEMPLT to the catalog named TARGET.TEMPLT, use the following statements:

```
libname target 'SAS-data-library';
libname one 'SAS-data-library';

proc greplay nofs;
   tc target.templt;
   tcopy one.templt.newtemp.template;
quit;
```

To create a duplicate copy of a template, simply omit *template-catalog* from your TCOPY statement. For example, to create a duplicate copy of a template named NEWTEMP within the template catalog, you could use the following statement:

```
tcopy newtemp.template;
```

TDEF Statement

Defines or modifies templates in the current template catalog.

Requirements: Assign a template catalog before using the TDEF statement.

See also: TC statement

Featured in: Example 1 on page 952

Syntax

TDEF*template-entry*
    <*panel-definition(s)*>
    <DES='*entry-description*'>;

panel-definition has the following form:

panel-number / <*panel-option(s)*>

panel-option(s) can be one or more of the following:
CLIP
COLOR=*border-color*
COPY=*panel-number*
DEF
DELETE
LLX=*x*
LLY=*y*
LRX=*x*
LRY=*y*
ROTATE=*degrees*
SCALEX=*factor*
SCALEY=*factor*
ULX=*x*
ULY=*y*
URX=*x*
URY=*y*
XLATEX=*distance*
XLATEY=*distance*

Required Arguments

template-entry
identifies an existing or a new template. If the template is not in the template catalog, the procedure creates it. If the template is already in the template catalog, the procedure modifies or makes additions to that template.

Only *template-entry* is required, but if you specify only the template name without any options, no changes are made to an existing template and no new template is created.

Options

CLIP
specifies that any panels behind this panel should be clipped. If clipping is in effect for a panel, only the graphics output that is to be placed in that panel can appear in the space that the panel occupies, unless a previous panel occupies all or part of that space.

COLOR=*border-color*
specifies the color of the panel border. If you omit *border-color*, then no border is displayed around the panel when you replay graphics output in the panel. If you preview a template that contains a panel without a border color, the GREPLAY procedure uses the first color in the colors list as the outline for the border.

COPY=*panel-number*
specifies the number of the panel definition that is to be copied to this panel.

DEF
specifies a default panel. A default panel has the following characteristics:

Panel Corner	Coordinates
lower left	(0,0)
upper left	(0,100)
upper right	(100,100)
lower right	(100,0)

DELETE
DEL
deletes the panel.

DES='*entry-description*'
specifies the description of the catalog entry for the template. The maximum length for *entry-description* is 40 characters. By default, the procedure uses *** new template *** for the description.

LLX=*x*
specifies the X coordinate of the lower-left corner of the panel. Units for *x* are percentage of the graphics output area.

LLY=*y*
specifies the Y coordinate of the lower-left corner of the panel. Units for *y* are percentage of the graphics output area.

LRX=*x*
specifies the X coordinate of the lower-right corner of the panel. Units for *x* are percentage of the graphics output area.

LRY=*y*
specifies the Y coordinate of the lower-right corner of the panel. Units for *y* are percentage of the graphics output area.

panel-number
identifies the number of the panel that is being defined or modified.

ROTATE=*degrees*
specifies the rotation angle for the panel. The coordinates of the panel corners are automatically adjusted.

SCALEX=*factor*
specifies the scale factor for the X coordinates in the panel. You can use this scale factor to increase or decrease the size of the panel in the X direction or to reverse the X coordinates for the panel.

SCALEY=*factor*

specifies the scale factor for Y coordinates in the panel. You can use this scale factor to increase or decrease the size of the panel in the Y direction or to reverse the Y coordinates for the panel.

ULX=*x*

specifies the X coordinate of the upper-left corner of the panel. Units for *x* are percentage of the graphics output area.

ULY=*y*

specifies the Y coordinate of the upper-left corner of the panel. Units for *y* are percentage of the graphics output area.

URX=*x*

specifies the X coordinate of the upper-right corner of the panel. Units for *x* are percentage of the graphics output area.

URY=*y*

specifies the Y coordinate of the upper-right corner of the panel. Units for *y* are percentage of the graphics output area.

XLATEX=*distance*

specifies the distance to move the X coordinates of the panel. Units for *distance* are percentage of the graphics output area.

XLATEY=*distance*

specifies the distance to move the Y coordinates of the panel. Units for *distance* are percentage of the graphics output area.

Details

Use coordinate values that are less than 0 and greater than 100 in the LLX=, LLY=, LRX=, LRY=, ULX=, ULY=, URX=, and URY= options to zoom in on the graphics output. That is, you can see only that part of the replayed graphics output in the range from 0 to 100 percent of the graphics output area.

The values that you supply for the SCALEX= and SCALEY= options are used to change the size and orientation of the panel. The scale factors are used for the corresponding X and Y coordinates of the panel. For example, if you specify

```
scalex=.5
scaley=2
```

the X coordinates are scaled to half the original size, and the Y coordinates are scaled to twice the original size.

If you supply a scale factor of 0, all of the coordinates are set to the same value. If you use a scale factor of 1, nothing happens. If you use a scale factor greater than 1, the values of the coordinates are increased and hence the size of the panel increases. If you use a scale factor less than 1 but greater than 0, the values of the coordinates are decreased and hence the size of the panel decreases. If you use a negative scale factor, the coordinates are reversed and hence the panel (and any graphics output replayed in the panel) is reversed.

TDELETE Statement

Deletes templates from the template catalog.

Caution: The GREPLAY procedure does not prompt you to confirm your request to delete templates.

Alias: TDEL

Syntax

TDELETE *template-entry(s)* | _ALL_ ;

Required Arguments

One of the following is required:

template-entry(s)
identifies a template that is to be deleted from the template catalog. You can submit a single entry or a list of entries in a single TDELETE statement.

ALL
deletes all of the templates in the template catalog.

TEMPLATE Statement

Assigns a current template to use when replaying graphics output.

Requirements: Assign a template catalog before using the TEMPLATE statement.

Note: If you specify a template that is not in the current template catalog or if you specify a template before you have assigned a template catalog, the GREPLAY procedure issues an error message.

Featured in: Example 1 on page 952

Syntax

TEMPLATE *template-entry*;

Required Arguments

template-entry
identifies an existing template to use when replaying graphics output. Use the TREPLAY statement to replay graphics output in the template.

TREPLAY Statement

Selects one or more catalog entries from the same catalog for replay in template panels.

Requirements: Assign a template catalog and a current template before using the TREPLAY statement.

Alias: TPLAY

See also: TC statement

Featured in: Example 2 on page 954

Syntax

TREPLAY *select-pair(s)*;

select-pair has the following form:
panel-number:entry-id

select-pair has the following form: *panel-number:entry-id*

Required Arguments

panel-number:entry-id
specifies the panel number and the name of a catalog entry.

panel-number
is the number of the panel in the current template in which to replay the graphics output.

entry-id
is the name or number of a catalog entry that contains the graphics output to be replayed in *panel-number*.

Details

When you replay existing GRSEG entries in a template, the GREPLAY procedure creates new graphics output that is stored in the output catalog.

You can replay as many entries as you want in a single TREPLAY statement as shown here:

```
treplay 1:plot1 2:plot2 3:chart1;
```

The order in which you list the entries in the TREPLAY statement determines the order in which they are played into the template. PLOT1 will be placed in panel 1 of the current template, PLOT2 will be placed in panel 2, and CHART1 will be placed in panel 3. You can use entry numbers in the place of entry names.

Using the GREPLAY Procedure

Using the GREPLAY Windows

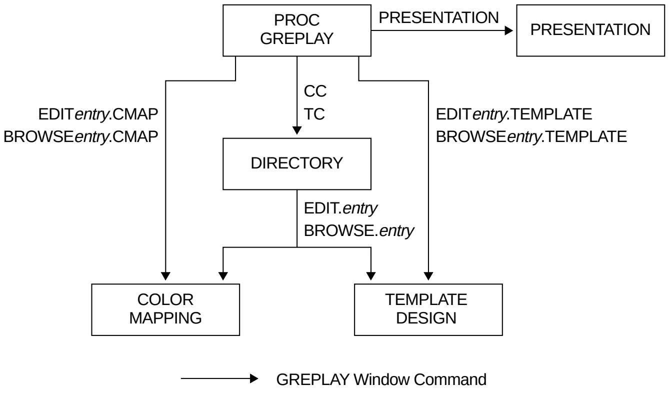
You can use the GREPLAY windows instead of code-based statements to replay and manage catalog entries. You perform tasks that use the GREPLAY procedure windows by entering values in the fields that are displayed in the windows and by issuing commands from the command line.

There are five GREPLAY windows:

- □ PROC GREPLAY window
- □ PRESENTATION window
- □ DIRECTORY window
- □ TEMPLATE DESIGN window
- □ COLOR MAPPING window.

Figure 26.2 on page 946 shows how these windows relate to each other. Each window can be scrolled forward or backward as needed to display additional fields and information.

Figure 26.2 GREPLAY Procedure Windows



This section briefly describes the GREPLAY windows; for a complete description of each window and its fields, refer to the SAS Help facility.

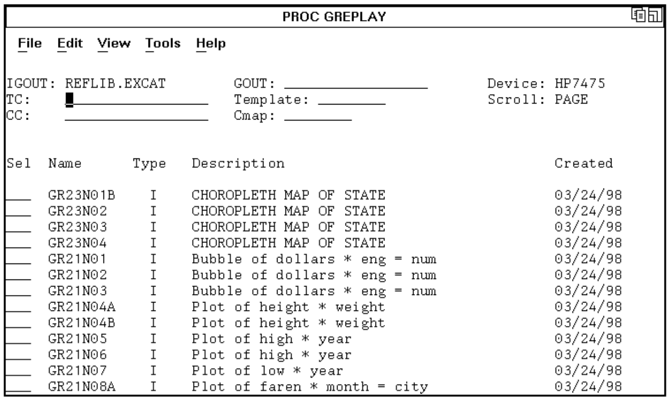
GREPLAY Window Commands

You can navigate and manipulate the GREPLAY windows by entering commands on the command line or by selecting them from the menus. For a complete description of all the GREPLAY window commands, refer to the SAS Help facility.

PROC GREPLAY Window

This window is the first to appear when you submit the PROC GREPLAY statement on a full-screen device without the PRESENTATION or NOFS option. You can use it to both replay graphics output and to manage catalog entries that contain graphics output.

Display 26.1 The PROC GREPLAY Window



```
IGOUT: REFLIB.EXCAT        GOUT: _________________        Device: HP7475
TC:    _________________   Template: ________             Scroll: PAGE
CC:    _________________   Cmap: ________


Sel  Name      Type   Description                               Created

___  GR23N01B    I    CHOROPLETH MAP OF STATE                   03/24/98
___  GR23N02     I    CHOROPLETH MAP OF STATE                   03/24/98
___  GR23N03     I    CHOROPLETH MAP OF STATE                   03/24/98
___  GR23N04     I    CHOROPLETH MAP OF STATE                   03/24/98
___  GR21N01     I    Bubble of dollars * eng = num             03/24/98
___  GR21N02     I    Bubble of dollars * eng = num             03/24/98
___  GR21N03     I    Bubble of dollars * eng = num             03/24/98
___  GR21N04A    I    Plot of height * weight                   03/24/98
___  GR21N04B    I    Plot of height * weight                   03/24/98
___  GR21N05     I    Plot of high * year                       03/24/98
___  GR21N06     I    Plot of high * year                       03/24/98
___  GR21N07     I    Plot of low * year                        03/24/98
___  GR21N08A    I    Plot of faren * month = city              03/24/98
```

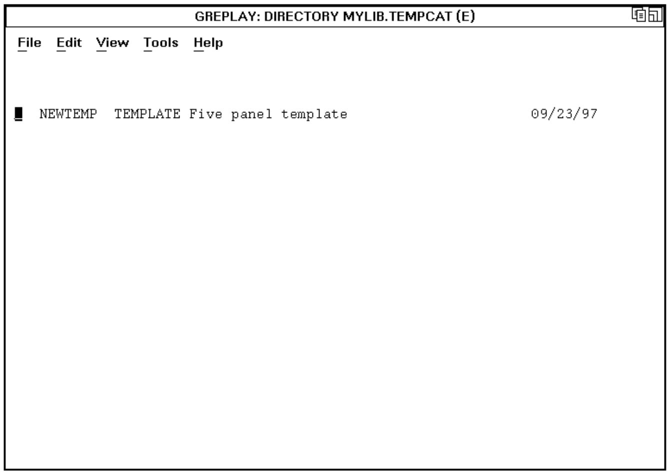
PRESENTATION Window

This window is a modified version of the PROC GREPLAY window that enables you to replay graphics output while it prevents you from deleting entries or changing templates and color maps. Once you have created and organized your catalog, you may want to use the PRESENTATION window in an application for replaying graphics output.

DIRECTORY Window

This window lists the names of the catalog entries, gives a brief description of each, and indicates the date on which each entry was created or last changed. Although all catalog entry types are displayed in the DIRECTORY window, you can manage only entries of type CMAP or TEMPLATE from this window.

Display 26.2 The DIRECTORY Window

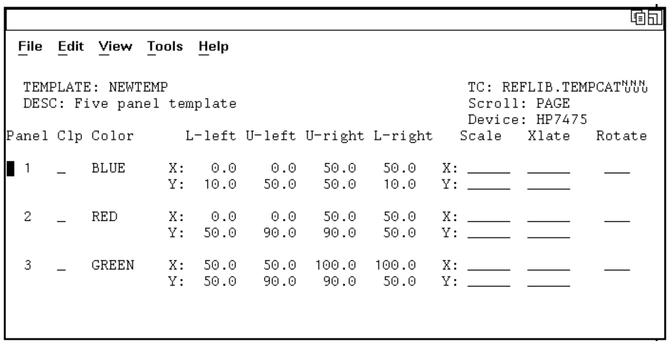

GREPLAY: DIRECTORY MYLIB.TEMPCAT (E)
File Edit View Tools Help

```
_ NEWTEMP  TEMPLATE Five panel template                09/23/97
```

TEMPLATE DESIGN Window

This window lets you design templates that you can use to present graphics output. You design a template by specifying the coordinates of its panels and determining the order in which the template panels are filled. Once you enter coordinates for a panel, you can alter them easily by using the Scale, Xlate (translate), and Rotate utility fields. These utility fields recalculate coordinate values automatically.

Display 26.3 The TEMPLATE DESIGN Window

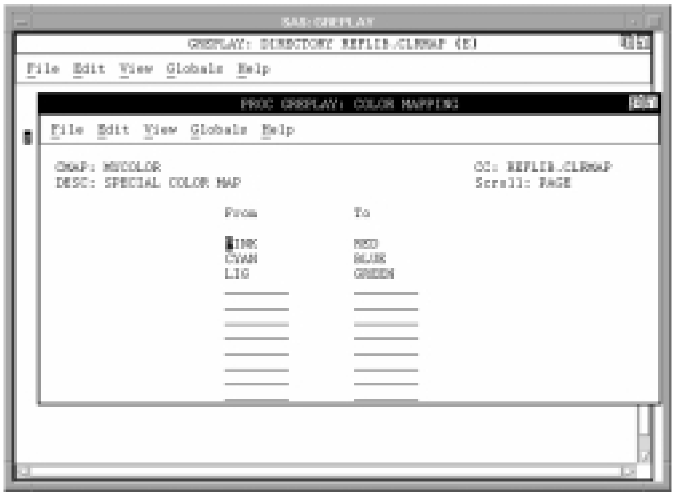


```
 TEMPLATE: NEWTEMP                                         TC: REFLIB.TEMPCAT
 DESC: Five panel template                                 Scroll: PAGE
                                                           Device: HP7475
Panel Clp Color      L-left U-left U-right L-right   Scale   Xlate   Rotate

  1   _   BLUE     X:   0.0    0.0   50.0    50.0   X: _____ _____    ___
                   Y:  10.0   50.0   50.0    10.0   Y: _____ _____

  2   _   RED      X:   0.0    0.0   50.0    50.0   X: _____ _____    ___
                   Y:  50.0   90.0   90.0    50.0   Y: _____ _____

  3   _   GREEN    X:  50.0   50.0  100.0   100.0   X: _____ _____    ___
                   Y:  50.0   90.0   90.0    50.0   Y: _____ _____
```

COLOR MAPPING Window

This window allows you to map colors in existing graphics output to new colors when you replay the graphics output. When you replay graphics output that is contained in a catalog entry and assign a current color map, any color in the graphics output that appears in the From column of the color map is mapped to the corresponding color in the To column of the color map. Using a color map does not change the contents of the replayed graphics output and does not produce new graphics output unless a template is also used.

Display 26.4 The COLOR MAPPING Window


SAS:GREPLAY
GREPLAY: DIRECTORY REFLIB.CLRMAP (E)
File Edit View Globals Help
PROC GREPLAY: COLOR MAPPING
File Edit View Globals Help

```
CMAP: MYCOLOR                                        CC: REFLIB.CLRMAP
DESC: SPECIAL COLOR MAP                              Scroll: PAGE

                   From            To

                   PINK            RED
                   CYAN            BLUE
                   LIG             GREEN
                   ________        ________
                   ________        ________
                   ________        ________
                   ________        ________
                   ________        ________
                   ________        ________
                   ________        ________
```

Table 26.3 GREPLAY Procedure Windows

If you are in the...	and you want to open the...	then...
PROC GREPLAY statement	PROC GREPLAY window	Submit the PROC GREPLAY statement without using the PRESENTATION or NOFS options.
	PRESENTATION window	Submit the PROC GREPLAY statement and include the PRESENTATION and IGOUT= options.
PROC GREPLAY window	PRESENTATION window	Specify a catalog and issue the PRES command.
	DIRECTORY window	Specify a template catalog and issue the TC command. OR Specify a color map catalog and issue the CC command.
	TEMPLATE DESIGN window	Specify a template catalog and issue the following command: edit *template-name*.template
	COLOR MAPPING window	Specify a color map catalog and issue the following command: edit *color-map-name*.cmap
DIRECTORY window	TEMPLATE DESIGN window	Place an S beside the name of an existing template. OR Issue the following command: edit *template-name*.template
	COLOR MAPPING window	Place an S beside the name of an existing color map. OR Issue the following command: edit *color-map-name*.cmap

Managing Catalog Entries

You can replay entries or perform a variety of catalog management tasks with GREPLAY code-based statements. Table 26.4 on page 950 lists several common tasks and the statements that you use to perform them.

Table 26.4 Ways of Performing Common GREPLAY Procedure Tasks

Task	Code-based Statement
copy graphics output (GRSEGs) from the input catalog to the output catalog*	COPY statement
arrange GRSEG entries into logical groupings	GROUP statement
reorder the entries	MOVE statement
delete unneeded GRSEG entries	DELETE statement
change entry names and descriptions of entries in the input catalog	MODIFY statement
replay an entry from the input catalog	REPLAY statement
replay an entry in a template panel	TREPLAY statement

*You must assign an output catalog before copying graphics output.

Replaying Catalog Entries

To select catalog entries for replay, first assign an input catalog that contains the graphics output that is to be replayed. Then assign the entry with the REPLAY statement.

For example, the following statements replay the GRSEG entry named GRAPH1 from the catalog MYLIB.GRAPHS, which is assigned with the IGOUT= option:

```
libname mylib 'SAS-data-library';

proc greplay igout=mylib.graphs nofs;
   replay graph1;
run;
quit;
```

If you do not specify an output catalog with the GOUT= option in the PROC statement of the SAS/GRAPH procedure that creates the graphics output, the graphics output is automatically stored in the WORK.GSEG catalog. Replay the graphics output that is stored in this catalog as follows:

```
proc greplay nofs;
   igout work.gseg;
   replay _all_;
run;
quit;
```

Creating Templates and Color Maps

You can use the GREPLAY procedure to create templates and color maps. You can use templates to replay graphics output from several catalog entries on a single display, or to change the shape or size of graphics output. You can use color maps to remap colors when replaying graphics output.

A color map is a list of up to 256 pairs of colors that enables you to change the colors in graphics output by mapping the original colors to a list of new colors. Color maps are useful for controlling how colors that are not available on the current device are remapped.

When you assign a current color map and replay graphics output that is stored in a catalog entry, any color in your graphics output that appears in the From column of the color map is mapped to the corresponding color in the To column of the color map. The new colors are not saved with the graphics output, and you do not create new graphics output when you use a color map to replay graphics output.

Table 26.5 Ways of Performing Common GREPLAY Procedure Tasks

Task	In line mode
assign a color map catalog	CC statement
copy a color map from another catalog	CCOPY statement
define or modify a color map in the current catalog	CDEF statement
assign a color map to use when you replay graphics output	CMAP statement
delete unneeded entries	DELETE statement
assign a template catalog	TC statement
copy a template from another catalog	TCOPY statement
delete a template	TDELETE statement
define a template	TDEF statement
display the panel outlines for a template	PREVIEW statement
assign a template to use when you replay graphics output	TEMPLATE statement
replay an entry in a template panel	TREPLAY statement

Before you create a template, you must assign a template catalog. If you are use the GREPLAY procedure in line mode, use the TDEF statement to define a template and the PREVIEW statement to preview a template. For example, the following statements define and preview a template named TEMPLT:

```
tdef templt 1/def;
preview templt;
```

Before you create a color map, you must assign a color map catalog. If you use the GREPLAY procedure in line mode, use the CDEF statement to define a color map. For example, the following statement defines a color map named CLRMAP:

```
cdef clrmap 1 / cyan : blue;
```

Replaying Graphics Output in a Template

You can use the GREPLAY procedure to create new graphics output by replaying existing graphics output in templates. You can use the templates that are provided with SAS/GRAPH software and stored in the catalog SASHELP.TEMPLT, or you can create your own templates. Before you can replay graphics output in a template, you must assign a template catalog and a current template, as well as an input catalog. Then assign the entries to the template with the TREPLAY statement.

For example, the following statements replay the entries GRAPH1 and GRAPH2 into the V2 template, which is stored in the catalog SASHELP.TEMPLT. The TC statement specifies the catalog that contains the template, and the TEMPLATE statement specifies the template. The TREPLAY statement assigns each entry to a panel. (The V2 template has two panels, so there is an assignment for panel 1 and panel 2.)

```
proc greplay igout=mylib.graphs nofs;
   tc sashelp.templt;
   template v2;
   treplay 1:graph1 2:graph2;
run;
```

When you replay graphics output in a template, the new graphics output that is created by the GREPLAY procedure is automatically provided a default name and is stored in the output catalog, WORK.GSEG.

Templates are often used to describe positioning for replaying graphics output from several catalog entries on a single display.

Examples

The following examples illustrate major features of the GREPLAY procedure.

Example 1: Creating a Template

Procedure features:

GREPLAY statement options:

NOFS
TC=

TDEF statement

TEMPLATE statement

Sample library member: GR26N01

This example creates a template with five panels. Four of the panels are small and equal in size. The fifth panel is a large, full-size panel that can be used later to display a common title or footnote for the entire template (see). In this example, the LIST statement displays the template contents in the log. Output 26.1 on page 954 shows the template definition that is written to the log file. The template that is defined here is also used in Example 2 on page 954.

Assign the libref and set the graphics environment.

```
libname reflib 'SAS-data-library';
goptions reset=global gunit=pct border cback=white
         colors=(black blue green red)
         ftext=swissb htitle=6 htext=3
```

Start the GREPLAY procedure. NOFS starts the procedure in line-mode. TC=assigns TEMPCAT as the template catalog.

```
proc greplay tc=reflib.tempcat nofs;
```

Define a template with four panels. The TDEF statement defines a template named NEWTEMP and places it in the previously defined template catalog. Each definition identifies the panel number and specifies the coordinates of the four corners. The panels are arranged within the template as follows: panel 1 is lower left; panel 2 is upper left; panel 3 is upper right; panel 4 is lower right; panel 5 is the full size so that it can contain a common title and footnote for all the template entries. COLOR= draws a border for each panel in the specified color.

```
tdef newtemp des='Five panel template'

     1/llx=0   lly=10
       ulx=0   uly=50
       urx=50  ury=50=
       lrx=50  lry=10
       color=blue

     2/llx=0   lly=50
       ulx=0   uly=90
       urx=50  ury=90=
       lrx=50  lry=50
       color=red

     3/llx=0   lly=50
       ulx=0   uly=90
       urx=100 ury=90=
       lrx=100 lry=50
       color=green

     4/llx=50   lly=10
       ulx=50  uly=50
       urx=100 ury=50=
       lrx=100 lry=10
       color=cyan
```

Assign the current template. The TEMPLATE statement assigns the newly created template NEWTEMP as the current template.

```
    template newtemp;
```

Write the contents of the current template to the log.

```
    list template;
 quit;
```

Output 26.1 Defining a Template (GR26N01)

```
.
.
.

64         /* list contents of current template */
65      list template;

 NEWTEMP       Five panel template

 Pan Clp Color      Ll-x  Ll-y  Ul-x  Ul-y  Ur-x  Ur-y  Lr-x  Lr-y

  1      BLUE        0.0  10.0   0.0  50.0  50.0  50.0  50.0  10.0
  2      RED         0.0  50.0   0.0  90.0  50.0  90.0  50.0  50.0
  3      GREEN      50.0  50.0  50.0  90.0 100.0  90.0 100.0  50.0
  4      CYAN       50.0  10.0  50.0  50.0 100.0  50.0 100.0  10.0
  5      LIPK        0.0   0.0   0.0 100.0 100.0 100.0 100.0   0.0

66   quit;
.
.
.
```

Example 2: Replaying Graphics Output in a Template

Procedure features:

GREPLAY statement options:

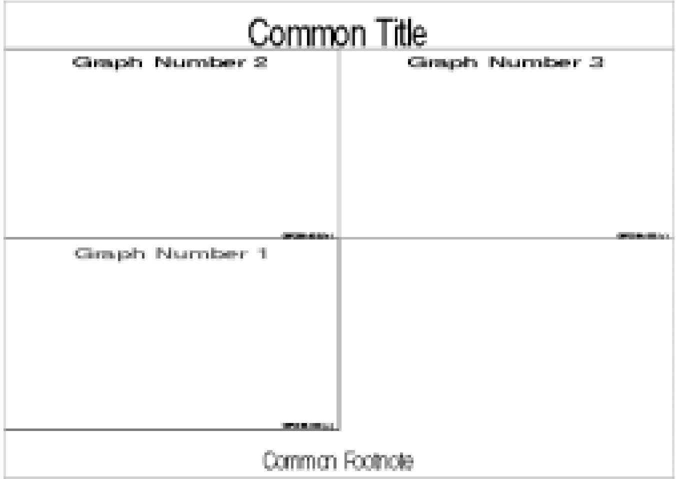
- GOUT=
- IGOUT=
- TEMPLATE=

TREPLAY statement

Other features:

PROC GSLIDE

Sample library member: GR26N02



The TREPLAY statement replays into the template NEWTEMP four catalog entries that contain graphics output. The NEWTEMP template is defined in Example 1 on page 952. It contains four equally sized panels and one large, full-size panel. Note that assignments are made to all but one of the panels. Because the fourth panel is not listed in the TREPLAY statement, it does not appear in the graphics output. The HSIZE= and VSIZE= options are adjusted and then reset to default in order to reflect the dimensions of the different-sized template panels.

Assign the libref and set the graphics environment. HSIZE= and VSIZE= are set for the dimensions of template panels 1, 2, 3, and 4.

```
libname reflib 'SAS-data-library';
goptions reset=all gunit=pct border
         cback=white colors=(black blue green red)
         ftext=swiss htitle=8 htext=5 nodisplay
         hsize=4in vsize=3.8in;
```

Generate three graphs in the permanent catalog GRAFCAT. The GSLIDE procedure creates three text slides and stores them in REFLIB.GRAFCAT as specified by the GOUT= option. By default, these are stored as GSLIDE, GSLIDE1, and GSLIDE2.

```
proc gslide gout=reflib.grafcat;
   title c=red 'Graph Number 1';
   footnote h=3 j=r 'GR26N02(a) ';
run;
   title 'Graph Number 2';
   footnote h=3 j=r 'GR26N02(b) ';
run;
   title 'Graph Number 3';
   footnote h=3 j=r 'GR26N02(c) ';
run;
```

Reset HSIZE= and VSIZE= to the default values and generate a text slide with PROC GSLIDE. Resetting the HSIZE and VSIZE values enables you to create a text slide with the proper aspect ratio for use in template panel 5.

```
goptions hsize=0in vsize=0in;
proc gslide gout=reflib.grafcat;
   title 'Common Title';
   footnote 'Common Footnote';
run;
```

Display the graphics output.

```
goptions display;
```

Start the GREPLAY procedure. IGOUT= assigns GRAFCAT as the input catalog. GOUT= assigns EXCAT as the output catalog. TEMPLATE= assigns NEWTEMP as the current template.

```
proc greplay igout=reflib.grafcat gout=reflib.excat
             tc=reflib.tempcat nofs;
   template=newtemp;
```

Replay three graphs into template. The TREPLAY statement assigns three entries to panels in the NEWTEMP template. Each assignment is a panel number and the name of a graphics output entry. Names are the default names assigned by the GSLIDE procedure.

```
   treplay 1:gslide
           2:gslide1
           3:gslide2
           5:gslide3;
quit;
```

Example 3: Creating a Color Map

Procedure features:

GREPLAY statement option:

CC=
GOUT=

CDEF statement

CMAP statement

LIST statement

Sample library member: GR26N03

This example uses the CDEF statement to define a color map. The LIST statement is used in this example to display the color map definition in the log. Output 26.2 on page 957 shows a partial listing of the log.

Assign the libref and set the graphics environment.

```
libname reflib 'SAS-data-library';
goptions reset=global gunit=pct border cback=white
```

```
         colors=(black blue green red)
         ftext=swissb htitle=6 htext=3;
```

Start the GREPLAY procedure. CC= assigns CLRMAP as the color map catalog. GOUT= specifies the permanent catalog in which to place the graphics output.

```
proc greplay cc=reflib.clrmap gout=reflib.excat nofs;
```

Define a color map. The CDEF statement defines a color map named MYCOLOR that contains three color pairs.

```
   cdef mycolor des='Special Color Map'
        1 / pink  : red
        2 / cyan  : blue
        3 / lig   : green;
```

Specify current color map and write contents to the log. The CMAP statement assigns MYCOLOR as the current color map. The contents of CMAP are listed in the log.

```
   cmap mycolor;
   list cmap;
quit;
```

Output 26.2 Defining a Color Map (GR26N03)

```
.
.
75        /* list the contents of the color map */
76     list cmap;

 MYCOLOR       Special Color Map

       FROM        TO

  1    PINK        RED
  2    CYAN        BLUE
  3    LIG         GREEN

77  quit;
.
```

The correct bibliographic citation for this manual is as follows: SAS Institute Inc., *SAS/GRAPH® Software: Reference, Version 8*, Cary, NC: SAS Institute Inc., 1999.

SAS/GRAPH® Software: Reference, Version 8

ISBN 1-58025-525-6

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513.

1st printing, October 1999

The Institute is a private company devoted to the support and further development of its software and related services.