## Contents

**Chapter 1 • Overview of the MDDB Report Viewer** .................................................. 1  
What Is the MDDB Report Viewer? ........................................................................... 1  
Support for Access Control Features .................................................................... 1  
Requirements for Running the MDDB Report Viewer ........................................ 2

**Chapter 2 • Setting Up the MDDB Report Viewer** ............................................... 3  
Methods for Setting Up the MDDB Report Viewer .............................................. 3  
Working with Repositories .................................................................................... 6

**Chapter 3 • Using the MDDB Report Viewer** .................................................... 9  
Tips for Using the MDDB Report Viewer ............................................................. 9

**Chapter 4 • Making Advanced Customizations to the MDDB Report Viewer** ........ 15  
MDDB Report Viewer Class .................................................................................. 18  
MDDB Report Viewer Instance Variables .......................................................... 18  
Flow of Control in the MDDB Report Viewer Class ........................................... 21  
MDDB Report Viewer Variables .......................................................................... 30  
MDDB Report Viewer Cascading Style Sheets .................................................... 36  
Dictionary ............................................................................................................ 38

**Index** .................................................................................................................. 159
Chapter 1
Overview of the MDDB Report Viewer

What Is the MDDB Report Viewer?

The MDDB Report Viewer enables users to generate and view reports and graphs of data that are stored in a multidimensional database (MDDB) without running a SAS session.

An MDDB is a specialized data storage facility that stores summarized data for fast and easy access. Users can quickly view large amounts of data as a value at any cross-section of business dimensions. A business dimension can be any vision of the data that makes sense, such as time, geography, or product. Users create and update multidimensional databases using SAS/EIS software or the MDDB procedure when the SAS OLAP Server has been licensed.

The MDDB Report Viewer enables users who do not have access to SAS software (or who do not want to invoke SAS software) to view the data in an MDDB. This capability eliminates the need to have SAS software running on all users' machines and provides access to the MDDB reports and graphs in a Web environment.

Note: The SAS OLAP Server, available in SAS 9.1 and later, enables users to develop advanced SAS Business Intelligence applications using all of our new software, including SAS Information Delivery Portal, SAS Web Report Studio, and SAS Enterprise Guide. If you are developing new OLAP applications, then consider using this new technology rather than the MDDB Report Viewer.

Support for Access Control Features

The MDDB Report Viewer enables you to perform the following tasks that are associated with the Access Control features of SAS/EIS software:

• deny access to the entire table
• drop or keep hierarchies
The MDDB Report Viewer supports the following Applications Access features:

- Report Layout
- Show Detail Data.

Requirements for Running the MDDB Report Viewer

Before you begin setting up the MDDB Report Viewer, you must meet the following requirements:

- Version 9 or later of the following SAS software products must be licensed at your site:
  - Base SAS software.
  - SAS/IntrNet software. The Application Dispatcher component (consisting of the Application Broker and Application Server components) must be installed and configured.
  - SAS/GRAPH software (optional but recommended).
  - SAS/EIS software or the SAS OLAP Server must be licensed at your site.

Note: MDDB Report Viewer 9.3 works only with the V8 SAS OLAP Server, which is available with both SAS 8 (as a separate product) and SAS 9 (as part of the SAS OLAP Server).

- The MDDB that you use to generate reports must be created, registered in a repository, and stored in a location to which you have access. You can create an MDDB by using SAS/EIS software or PROC MDDB when the SAS OLAP Server has been licensed. SAS/EIS software automatically registers the MDDB in the repository. If you use PROC MDDB to create the MDDB file, you must register the MDDB in a SAS/EIS repository. See the online Help for these products for complete instructions on how to create an MDDB. The MDDB Report Viewer can use only MDDB files to create reports. It cannot use SAS data sets.

- Your Web browser must support HTML pages with frames.
Chapter 2
Setting Up the MDDB Report Viewer

Methods for Setting Up the MDDB Report Viewer

Overview of Methods for Setting Up the MDDB Report Viewer

Method 1
Method 2
Method 3

Working with Repositories

Overview of Working with Repositories
Specifying the System Repository Manager Location
Setting Up the System Repository Manager Files
Defining the Repository to Application Dispatcher
Setting Up the SASHELP Repository

Methods for Setting Up the MDDB Report Viewer

Overview of Methods for Setting Up the MDDB Report Viewer

The MDDB Report Viewer consists of three HTML pages in which users can enter information to generate reports and graphs from an MDDB. Some features of the MDDB Report Viewer pages might appear slightly different on different Web browsers. If you use more than one Web browser to access the MDDB Report Viewer, consider these differences when you set up and customize the tool.

You can use SAS/EIS software access control features with the MDDB Report Viewer. See “Support for Access Control Features” on page 1 to learn more about using access control.

Note: To run this release of the MDDB Report Viewer, your system administrator must have previously set up a Repository Manager for accessing metadata. For more information about this setup procedure, see “Working with Repositories” on page 6. You can also refer to the online SAS Help and Documentation for Base SAS software and SAS/EIS software for details about setting up a repository.

You can use any of three methods to set up the MDDB Report Viewer.

Method 1
Copy the sample webeis.html page for the MDDB Report Viewer. The sample webeis.html page is included in the SAS/IntrNet CGI Tools for the Web server installation package and can be found in the sasweb/IntrNet9/MRV directory under
your Web server root document directory. Modify the webeis.html file to specify your site's repositories, services, background colors, and so on. You can specify a subclass of the WEBEIS class to customize viewer behavior. See Method 3, Step 2 for a description of the CLASS parameter.

**Method 2**

Use the dynamic entry into the application by entering a URL that is similar to the following in your Web browser:

http://web-server-name/broker-URI?_program=sashelp.webeis.rptseld.scl &_service=myservice&metabase=sashelp.mbeis&bgtype=color&bg=red &class=sashelp.override.myweb.class

Here, broker-URI, BGTYPE, BG, and CLASS are as described in Method 3, Step 2. With this method, no HTML pages are created or stored.

**Method 3**

Run the SAS AF command to create HTML pages for your repositories and to set up the MDDB Report Viewer at your site. Follow these steps:

1. Start a SAS session.
2. To create the MDDB Report Viewer HTML file, enter the following command in the Program Editor window and submit the command to SAS for processing:

```
dm "af c=sashelp.webeis.rptseld.scl metabase=my-metabase
pathname='HTML-file' <CGI='broker-URI'>
<title='1996 Sales Report'> <bgtype='color'> <bg=blue>
<class='sashelp.override.myweb.class">"
```

Here

METABASE
is the name of the SAS/EIS repository in which the MDDB has been registered. A METABASE value is required. The name can contain up to 60 characters and blank spaces. If you use blank spaces or special characters in the name, you must delimit the name with single quotation marks ('). SAS recommends that you use the same or similar filenames for the METABASE and PATHNAME options so that you can easily determine the metabase with which a particular instance of the MDDB Report Viewer is associated.

*Note:* The term metabase is retained for backward compatibility.

PATHNAME
is the path and filename of the MDDB Report Viewer HTML file that is created by the AF command. The directory is typically located under the Web server document root or in another directory served by the Web server. A PATHNAME value is required. SAS recommends that you use the same or similar filenames for the METABASE and PATHNAME options so that you can easily determine the metabase with which a particular instance of the MDDB Report Viewer is associated.

CGI
is the optional URI for the Application Broker component of Application Dispatcher (for example, /cgi-bin/broker or scripts/broker.exe). If you do not specify a value for this option, you must supply a value in the HTML file after it is created.
TITLE

is the title that appears at the top of the report. A TITLE value is optional. If you do not specify a title, the title "Multidimensional Reports" is used.

Note: Avoid using a percent sign (%) in the title because this symbol might be misinterpreted.

BGTYPE

is the type of background that appears in the application reports. Specify bgtype='color' to control the color of the background or bgtype='image' to control the background pattern displayed in the application reports. Use this option with the BG option, described below. A BGTYPE value is optional. If you specify bgtype='color', the BG option expects one of the named colors or a hexadecimal value for one of the colors that is supported by your Web browser. If you specify bgtype='image', the BG option expects the URL of a background image file. You can specify only GIF and JPG image files for the background. If you specify BGTYPE and omit BG, or if you do not use either option, the background is the default color, silver.

Note: When you control the background color of the MDDB Report Viewer HTML pages, you might also want to control the background color of graphs that are displayed on the HTML pages. To do this, you can use a transparent GIF image, which is an image with a transparent background in which the HTML background color is visible. In effect, you create a graph in a clear frame so that the background color of the HTML page displays through the frame. A device driver to create the transparent GIF is not supplied with SAS/GRAPH software. However, you can use the TRANSPARENCY option of the SAS/GRAPH GOPTIONS statement to create a graph with a transparent background. For more information about the TRANSPARENCY option, see the documentation for the GOPTIONS statement in the SAS/GRAPH Help and Documentation.

BG

specifies the color or image to display in the background. A BG value is optional. If you specify bgtype='color', then specify a color value for BG. If you specify bgtype='image', then specify an image value for BG. You can specify a color name or a hexadecimal value for the color value. You can specify a URL for the image file value. See the documentation for your Web browser for valid color values. If you specify BG and omit BGTYPE, or if you do not use either option, the background is the default color, silver.

Note: If you specify an invalid color value, your Web browser maps the specification to a valid value.

CLASS

is the name of a subclass of the WEBEIS class. A CLASS value is optional. Add this parameter if the user has overridden any WEBEIS methods to change the viewer behavior. You can specify either a three- or four-level name. For example, the following are both valid:

sashelp.override.myweb

sashelp.override.myweb.class

3. In a text editor, open the HTML file that you created, and supply your own values in the HTML code that is preceded by a comment. These values include the following:
broker-URI
In the tag `<FORM ACTION="broker-URI">`, you must supply a value if you did not specify the CGI= option in the AF command that creates the HTML pages.

service-name, service-label
In the HTML lines

```html
<BR>Select service: <SELECT NAME="_service">
  <OPTION VALUE="service-name" SELECTED>service-label
```

specify the list of services that are available at your site. Provide an `<OPTION>` tag for each of your services. For more information about services, see _SERVICE_.

debug selection list
You can modify the list of debug options for your site in the following HTML line:

```html
Debugging level: <SELECT NAME="_debug">
```

4. Start the Application Server and point your Web browser to the HTML file that is generated in Method 3, Step 2.

You can specify the METABASE, PATHNAME, CGI, TITLE, BGTYPE, BG, and CLASS options in any order. Run the Application Server for each repository that contains MDDBs that users access when they run their reports.

---

**Working with Repositories**

**Overview of Working with Repositories**

The Common Metadata Repository is a general-purpose metadata management facility that provides common metadata services to different SAS/EIS applications. The Common Metadata Repository enables SAS/EIS software to share metadata with other SAS products.

Complete all of the following tasks to set up the Common Metadata Repository:

1. **Specify the system repository manager location.** (See page 6.)
2. **Set up the system repository manager files.** (See page 7.)
3. **Define the repository to the Application Server.** (See page 7.)
4. **Set up the SASHELP repository.** (See page 8.)

*Note:* You must have write access to the SASHELP directory to complete these tasks.

**Specifying the System Repository Manager Location**

Follow these steps to specify the location of the system repository manager:

1. Create a directory that is dedicated exclusively to the storage of repository manager files, for example:
   - Windows users: `!SASROOT\RPOSMSG`  
   - UNIX users: `!SASROOT/RPOSMSG`
This directory should not be used to store other SAS files.

Note: This system repository manager path is used later in this task.

2. Enter `REGEDIT` at a SAS command line. From the menu bar, select `Tools` \ `Options` \ `Registry Editor` to open the Registry Editor Options window. In the Select Registry View region, select the `View All` check box and then click OK. From the menu bar, select `File` \ `Close` to close the Registry Editor Options window.

3. Enter `REGEDIT` again at a SAS command line. Under the `HKEY_SYSTEM_ROOT` tree, expand `CORE` and `REPOSITORY`. Select the `REPOSITORY_MGR` node. From the menu bar, select `Tools` \ `Options` \ `Registry Editor`. Select `Open HKEY_SYSTEM_ROOT for write access`. Then click OK.

4. Select the Path item in the right pane. From the pop-up menu, select Modify. Enter the path from Step 1. For example, enter `!SASROOT\RPOSMGR`. Click OK to close the Edit String Value window. From the menu bar, select `File` \ `Close` to close the Registry Editor Options window and save the changes.

### Setting Up the System Repository Manager Files

Complete the following steps to set up the necessary system repository manager files. You must have write access to SASHELP to specify the system repository manager.

1. Create a directory that is dedicated exclusively to the storage of repository manager files, for example:
   - Windows users: `!SASROOT\RPOSMGR`
   - UNIX users: `!SASROOT/RPOSMGR`

   Do not store other SAS files in this directory.

2. At a SAS command line, enter `REPSMGR` and then select Setup Repository Manager.

3. In the Repository Manager Setup window, Library defaults to RPOSMGR. For Path, specify the path from Step 1 and then select the Write values to system registry check box. Then click OK.

4. In the resulting dialog box, click Yes to generate the necessary repository manager files.

This completes the set up for the System Repository Manager. You can create additional repository managers (a user repository manager, for example) by repeating these steps and by using a different path.

Note: This step sets the default location for the repository manager for your site. Individual users can override this location by executing the previous steps.

### Defining the Repository to Application Dispatcher

After you set up the Repository Manager files, you must include the following statements after the PROC APPSRV statement:

```
ALLOCATE LIBRARY RPOSMGR 'rposmgr-path' ;
DATALIBS RPOSMGR;
```
Setting Up the SASHELP Repository

Complete the following steps to set up the SASHELP repository:

1. At a SAS command line, enter `REPOSMGR` and then select `Repository Registration`.

2. In the Repository Registration window, select `New`.

3. In the Register Repository (New) window, enter `SASHELP` (in uppercase) in the `Repository` field. In the `Path` field, enter the full directory path where the CORE catalog is located. For example:
   - Windows users: `!SASROOT\CORE\SASHELP`
   - UNIX users: `!SASROOT/sashelp`

4. In the `Description` field, you can enter any character string (for example, `SASHELP Repository`). Click `OK` to close the Register Repository (New) window. Click `Close` to exit the Repository Registration window.

Note: Repositories cannot span multiple directories because the path cannot contain concatenated directories. If you have existing metabases in concatenated directories, copy the metabases to a single path that is referenced as a repository.
Chapter 3
Using the MDDB Report Viewer

Tips for Using the MDDB Report Viewer

Using the Interface

How do I use the MDDB Report Viewer?
The MDDB Report Viewer contains four Web pages in which you enter information or manipulate your report data:

Report Layout page
This page contains drop-down lists from which you select the MDDB and the style sheet to use.

Dimensions page
This page enables you to select the items that you want to include in the report.

• Click Options at the top of the page to go to the Optional Settings page, where you can specify a variety of options that control the layout of the report. In addition, you can specify whether to display a graph in the report.

• In the Columns section, define the report layout by selecting items to include from the Down and the Across list boxes.

• In the Analysis section, select one or more analysis variables from the Columns list box.

• In the Statistics section, select the variables that you want to specify statistics for from the Select Column list box (the items in the list box are the variables that you selected in the Analysis section). Then, from the Available list box, select one or more statistics by highlighting the desired statistics and then clicking the right-arrow button. To select all of the available statistics, click the double-right-arrow button. To deselect statistics, select the statistics in the Selected list box and then click the appropriate left-arrow button to remove them from the list box.
• Click **View Report** to display the report.

Optional Settings page
This page enables you to set the report options and to specify whether to display a graph in the report.

• Click **Dimensions** at the top of the page to go to the Dimensions page, where you select the items to include in your report.

• In the **Filter Columns** list box, select category variables for subsetting your report data.

• In the **Filter Listbox Options** section, customize the size and location of the **List By** list box on the Report page.

• In the **Report** section, specify a title for the report and whether to display a table in the report.

• In the **Graph** section, specify whether to display a graph and then customize its appearance and location in the report.

• When you click **View Report**, the report is displayed.

Report page
This page displays the table and graph that are produced from selections made in the previous pages. You can specify new variables and select subset values to change the report.

• Click **Download to spreadsheet** to download the data in the HTML table, including the titles, as it appears on the page.

• Click **Rotate** to rotate the down and across dimensions of a report.

• Click **Dimensions** to go to the Dimensions page, where you can select the items to include in your report.

• Click **Options** to go to the Optional Settings page, where you can specify a variety of options that control the layout of the report. In addition, you can specify whether to display a graph in the report.

• Click **? Help** to view the MDDB Report Viewer documentation or a Help page that you created.

• Change report dimensions by selecting different variables from the **Down** and **Across** list boxes on the Report page. After you select the new dimensions, click **View Report** to display the new report.

• In the **Filter By** list box, select the values of the category variables by which to subset your data and then click **Apply Filter**. The report is redisplayed with the subset applied. If a graph was previously displayed, it is redisplayed with the subset applied.

**How do I select items from a selection list?**
Web browsers have different selection methods. For example, some browsers use a SHIFT-click combination and others use a mouse click only. Use the selection method that is appropriate for your browser.

**How do I know whether the items that I select for a report are valid?**
Because selection list items cannot be disabled, you receive a message when an item is invalid. For example, you cannot select the same item (or hierarchy containing the same item) for the **Down** and **Across** values in a report. Simply reselect the items and run the report again.
What does the Rotate button do?
Use the **Rotate** button to rotate the down and across dimensions of a report.

How does Download to spreadsheet work?
The **Download to spreadsheet** button appears on the Report page and on the detail data page (after a reach-through to detail data). On the Report page, the **Download to spreadsheet** button downloads the data in the HTML table, including the titles, as it appears on the page. On the detail data page, the **Download to spreadsheet** button downloads the detailed data that is displayed on the page. The data is written in comma-delimited format, and you can open the file in your spreadsheet program or save the file to disk for later use.

You can use the `_MRVSEP` global variable to specify a delimiter other than a comma. For more information, see Table 2, MDDB Report Viewer Global Variables on page 34.

Printing Reports

How can I print reports?
You can print reports using the browser. Follow the instructions for printing that are appropriate for your browser.

Can I print extremely large tables?
If you print a table that is extremely wide, you might not get the results that you want. Tables cannot be resized, so when you print a large table, some columns might be truncated.

Changing the Appearance of a Report

Can I change report dimensions from the Report page?
You can change report dimensions by selecting different variables from the **Down** and **Across** list boxes on the Report page. After you select the new dimensions, click **View Report** to display the new report.

To add or change analysis variables or statistics, click **Dimensions** to return to the Dimensions page and change your selections. Then click **View Report**. The report is automatically displayed with your new selections.

Can I change the colors of my report?
Colors for report values are determined by values that are set in the RANGE entry in the SAS/EIS metabase in which the MDDB is registered. To change the colors in which report values are displayed, edit the RANGE entry in the SAS/EIS metabase. To use colors that are supplied by your browser, delete the RANGE entry in the SAS/EIS metabase. The background color of the table cell is set to the color value in the RANGE entry. Make sure that the numeric text is not set to the background color so that the text is readable.

Note: Cascading style sheet (CSS) settings overwrite a RANGE setting.
Viewing Your Data

**How do I drill down to additional values in a report?**
To drill down to other values in a report, select a **Down** or **Across** value. The report title changes when you drill down to other levels of information.

**How do I subset my report data?**
On the **Optional Settings** page, select the category variables (in the **Filter Columns** list box) by which to subset, and then click **View Report**. When the report is displayed, select the values of the category variables (in the **Filter By** list box) by which to subset your data, and click **Apply Filter**. The report is redisplayed with the subset applied. If a graph was previously displayed, it is redisplayed with the subset applied.

**How do I see the detail data?**
The numbers in the table should be hyperlinked if the BASETABLE attribute is in the metadata and if the base table exists. If the numbers are not hyperlinked, reach-through is not available for the selected MDDB. Click a number, and select the variables that you want to see from the data set. Click **Next**, and the detail data is displayed in a table.

Creating Graphs

**How do I generate a 3-D graph of the report data?**
To generate a three-dimensional graph of the report data, go to the **Optional Settings** page (by clicking **Options**), and select **3D Clickable Graph** in the **GRAPH** section. Then select the graph type (block, vertical bar, and so on) from the **Type** drop-down list. Click **View Report** to display the report along with a graph of the first column of data in the table. You can right-click within the graphics display area to change the graph's properties or to save the graph to a file. The three-dimensional graph is produced with the Graph Applet.

**How do I generate a standard GIF graph of the report data?**
To generate a standard GIF graph of the report data, go to the **Optional Settings** page (by clicking **Options**), and select **Standard GIF Graph** in the **GRAPH** section. Then select the graph type (block, vertical bar, and so on) from the **Type** drop-down list. Click **View Report** to display a report along with a graph of the first column of data in the table. You can select the GRAPH icon next to any column in the report to change the statistic that is graphed.

The GIF graph works in a different manner from the three-dimensional graph. To drill down using the GIF graph, you must drill down on the table rather than the graph itself. The GIF graph is a static graph, similar to the type of graph that is produced by the GPLOT procedure.

**How do I change the font for the standard GIF graph?**
You can specify the font for the standard GIF graph from the REQUEST INIT program that is used by your application server. In the REQUEST INIT program, set the `_GRFONT` macro variable by specifying the following:

```plaintext
%let _grfont=myfont;
```
By default, the MDDB Report Viewer uses the SWISSB font if a value is not specified for _GRFONT. For a complete list of available fonts, refer to *SAS/GRAPH: Reference*. For more information about the REQUEST INIT program, see the REQUEST statement syntax.

**Modifying the Default MDDB Report Viewer Settings**

**How do I specify the repository manager for the Application Dispatcher Server?**
After you set up the repository manager files, you must include the following statements after the PROC APPSRV statement:

```
ALLOCATE LIBRARY RPSGRM 'rpsmgr-path';
DATALIBS RPSGRM;
```

**How do I specify a different delimiter for Download to spreadsheet?**
To use a different delimiter for Download to spreadsheet, set the _MRVSEP macro variable in the REQUEST INIT program that is used by your application server. For example, to use a semicolon (;) instead of the default comma (,) delimiter, insert the following into your REQUEST INIT program:

```
%let _mrvsep=%str(;;);
```

**Can I create my own Help page?**
By default, the Help button points to the following URL, which is located on the SAS Web site:

http://support.sas.com/rnd/web/intrnet/mddbapp/hinttips.html

You can create your own Help page with information that is specific to your site. To do this, create the Help Web page and specify the URL in the _MRVHELP macro variable in the REQUEST INIT program that is used by your application server. For example, you could insert a line similar to the following in your REQUEST INIT program:

```
%let _mrvhelp=http://myserver/myhelp.html;
```

**Can I use cascading style sheets to modify the appearance of my report?**
The MDDB Report Viewer, Version 8 and later, supports cascading style sheets. Style sheets provide you with an easy way to customize the viewer for your site. For more information about how to use style sheets with the MDDB Report Viewer, see “MDDB Report Viewer Cascading Style Sheets” on page 36.

**Can I change the toolbar location?**
You can change the toolbar location by setting a macro variable in the REQUEST INIT program. Set the _MRTBLOC variable to

```
%let _mrtbloc=toolbar-location-value;
```

In this setting, the toolbar-location-value can be one of the following values: 1=top, 2=bottom, 3=left, 4=right, and 5=no toolbar.

The default toolbar location is 1=top.
Can I display reports without the Down and Across list boxes?
You can disable the display of the Down and Across list boxes by specifying the following in your service definition in the Application Broker configuration file:

ServiceSet _MRNODIMBOXES "X"

Can I disable the sorting feature?
You can disable the sorting feature by specifying the following in your service definition in the Application Broker configuration file:

ServiceSet _MRNOSORT "X"

Can I disable the row paging feature?
You can disable the row paging feature by specifying the following in your service definition in the Application Broker configuration file:

ServiceSet _MRNOPGOP "X"

Can I modify the settings for the number of rows to display?
By default, the options page lists ALL, 25, 50, and 100 as the number of rows to display. To modify these, specify a ServiceSet directive in the Application Broker configuration file for your service for the _MRVRNDX1, MRVRNDX2, MRVRNDX3, and MRVRNDX4 macro variables. For example, if you want the number of rows options to be ALL, 100, 200, and 500, use the ServiceSet directives in the Application Broker configuration file as follows:

ServiceSet _MRVRNDX1 "ALL"
ServiceSet _MRVRNDX2 "100"
ServiceSet _MRVRNDX3 "200"
ServiceSet _MRVRNDX4 "500"

Can I change the number of paging links that are displayed beneath the report table?
By default, five page links are displayed beneath the report. To modify this setting, use a ServiceSet directive for the _MRVNRLKS macro variable. For example, to display 10 paging links, specify

ServiceSet _MRVNRLKS "10"

How do I specify to the viewer not to use HTML frames?
To modify this setting, use a ServiceSet directive for the _MRNOFRAMES macro variable. For example, specify

ServiceSet _MRNOFRAMES "X"

The toolbar buttons on both the Layout and the Report pages are displayed at the top.

Can I change the appearance of the report table?
Use the _MRTBLPRM macro variable in a ServiceSet directive to change the appearance of the report table. For example, specify

ServiceSet _MRTBLPRM "CELLPADDING=4 CELSPACING=2 BORDER=3"

These attributes are inserted into the <TABLE> tag for the report.
Chapter 4
Making Advanced Customizations to the MDDB Report Viewer

MDDB Report Viewer Class .................................................. 18
MDDB Report Viewer Instance Variables ................................. 18
Flow of Control in the MDDB Report Viewer Class .................. 21
MDDB Report Viewer Variables ............................................. 30
MDDB Report Viewer Cascading Style Sheets ......................... 36
Dictionary ........................................................................... 38
_BUILD_ACROSS_LIST_Method ............................................ 38
_BUILD_ANALYSIS_LIST_Method ......................................... 38
_BUILD_ANLSORTORDER_Method ......................................... 39
_BUILD_APPLICATION_LIST_Method ....................................... 40
_BUILD_CURRENT_SUBSETS_Method ....................................... 41
_BUILD_DOWNL_LIST_Method ............................................. 42
_BUILD_STATSL_LIST_Method ............................................. 42
_BUILD_TOTAL_Method .................................................... 43
_BUILD_URL_ONSUBMIT_Method ......................................... 44
_BUILD_WHERE_FORMAT_STRING_Method ............................... 46
_CHECK_HIER_MEMBER_Method ........................................... 46
_CLOSE_FORM_Method ..................................................... 47
_CLOSE_PAGE_Method ...................................................... 48
_CLOSE_STATIC_FORM_Method ............................................. 49
_CREATE_STAT.Arrays_Method .......................................... 49
_DISPLAY_ACRROSS.Cells_Method ....................................... 53
_DISPLAY_ANALYSIS_VARS_Method ..................................... 55
_DISPLAY_DEFAULT_TITLE_Method ....................................... 56
_DISPLAY_DOWNVAR_CELL_Method ...................................... 57
_DISPLAY_ERROR_Method ................................................ 59
_DISPLAY_ONEWAY_Method ............................................... 60
_DISPLAY.ONEWAY BLOCK_Method ...................................... 61
_DISPLAY.ONEWAY.HBAR_Method ....................................... 61
_DISPLAY.ONEWAY.PIE_Method ......................................... 62
_DISPLAY.ONEWAY.VBAR_Method ....................................... 62
_DISPLAY_STATISTIC_VARS_Method .................................... 63
_DISPLAY_SUBSET_TITLE_Method ....................................... 65
_DISPLAY_TITLE_Method ................................................ 66
_DISPLAY_TWOWAY_Method ............................................. 67
_DISPLAY_TWOWAY BLOCK_Method ..................................... 68
_DISPLAY_TWOWAY.HBAR_Method ....................................... 68
_DISPLAY_TWOWAY.VBAR_Method ....................................... 69
_DISPLAY_VALUES_Method ................................................ 70
Chapter 4 • Making Advanced Customizations to the MDDB Report Viewer

_OUTPUT_DP_TITLE_OPTION_ Method .................................................... 74
_GET_ANALYSIS_VAR_NAME_ Method .................................................... 74
_GET_ANALYSIS_VARS_ Method ............................................................ 74
_GETAVAILABLE_STATS_ Method ........................................................ 75
_GETDATA_MODEL_NAME_ Method ......................................................... 75
_GET_DOWNVAR_LIST_ Method ............................................................. 76
_GET_EMDBMID_ Method .................................................................. 76
_GETGRAPH_VALUES_ Method ............................................................... 76
_MDB_NAME_ Method .................................................................... 79
_GETMESSAGE_ID_ Method ................................................................. 79
_GETMETABASE_NAME_ Method ........................................................... 79
_GETOUTPUTFILE_ID_ Method .............................................................. 80
_GETRANGE_COLOR_ Method ................................................................. 80
_GETSTATDESC_ Method .................................................................. 80
_GETSUBSET_FLAG_ Method ................................................................. 81
_OPENONEWAY_ Method ................................................................... 81
_OPEN_DYNAMICFILE_ Method ............................................................... 81
_OPENFORM_ Method ...................................................................... 81
_OPENONEWAY_ Method ................................................................... 82
_OPENSTATICFILE_ Method ................................................................. 83
_OPENTABLE_ Method ...................................................................... 83
_OPENTWOWAY_ Method ................................................................... 84
_OPENWEBOUTFORSPDSHT_ Method ....................................................... 86
_OUTPUTACROSSLIST_ Method ............................................................. 86
_OUTPUTADDTL_CLSVALPARMS_ Method .................................................. 87
_OUTPUTADDTL_RT_PARMS_ Method ...................................................... 87
_OUTPUTADDTOFAV_FUNCTION_ Method ................................................... 87
_OUTPUTALLURLITEMS_ Method ............................................................ 88
_OUTPUTANALLIST_ Method ................................................................. 88
_OUTPUTANALSELECT_ Method ............................................................. 89
_OUTPUT_ARROWFUNCTIONS_ Method ..................................................... 89
_OUTPUTBAR_SHAPE_LIST_ Method ....................................................... 91
_OUTPUTBOOKMARK_BUTTON_ Method .................................................. 91
_OUTPUTBOOKMARK_URL_ Method ....................................................... 92
_OUTPUTCLASSVAL_URL_FN_ Method ..................................................... 93
_OUTPUTCLICKABLEGRAPH_ Method .................................................... 94
_OUTPUTCONTENTHEADER_ Method ......................................................... 96
_OUTPUTCSVCCONTENTHEADER_ Method ................................................. 96
_OUTPUTDEBUGLIST_ Method .............................................................. 96
_OUTPUTDEFLTITLEOPTION_ Method ..................................................... 97
_OUTPUTDIMBTN_URL_FN_ Method ....................................................... 97
_OUTPUTDIMENSIONS_BUTTON_ Method ............................................... 98
_OUTPUTDOWNLIST_ Method ............................................................... 98
_OUTPUTDPTITLEOPTION_ Method ....................................................... 99
_OUTPUTDS2HTMHTML_ Method ........................................................... 100
_OUTPUTDS2HTMST_ Method ............................................................... 101
_OUTPUTDYNAMICHIDDENFLDS_ Method ............................................... 102
_OUTPUTEMPTYCELL_ Method ............................................................. 103
_OUTPUTEMPTYSERVICE_LIST_ Method ................................................ 103
_OUTPUTGRAPHDIMSOPTION_ Method ................................................... 104
_OUTPUTGRAPHINSTR_ Method ............................................................. 104
_OUTPUTGRAPHLIST_ Method .............................................................. 104
_OUTPUTGRAPHLOCOPTION_ Method .................................................... 105
_OUTPUTGRAPHOPTION_ Method ........................................................ 105
_OUTPUTGRAPHSOURCEOPTION_ Method ............................................. 106
_OUTPUTGRAPHTABLE_DISP_ Method ................................................... 106
MDDB Report Viewer Class

The MDDB Report Viewer class is a viewer that displays MDDB data. The class is a component of the MDDB Report Viewer, which is an application used by SAS/EIS software, SAS/IntrNet Application Dispatcher software, and SAS OLAP Server software.

The MDDB Report Viewer class enables you to specify dimensions that can be hierarchies or category variables, in addition to analysis variables. This class enables you to drill down on the hierarchy and other navigation, as well as to specify various types of graphic charts. The class writes output from the application to HTML in a Web browser.

PARENT: SASHELP.FSP.OBJECT.CLASS
CLASS: SASHELP.WEBEIS.WEBEIS.CLASS

MDDB Report Viewer Instance Variables

The following instance variables are used in many of the MDDB Report Viewer methods:

ACRDRL_ specifies the list of drill-down values for the across variables.
ACRVARS_ specifies the list of selected variables for the across dimension.
ALEVELS_ specifies the list of drill-down levels for the across variables.
ANALLBLS_ specifies the list of analysis variable long labels.
ANALLIST_ specifies the list of analysis variables and computed columns.
ANALVARS_ specifies the list of selected analysis variables.
ATOTAL_ specifies a flag that indicates whether the across totals are turned on.
CLASS_ contains the three- or four-level name of the WEBEIS subclass.
CSSTURL_  
contains the URL for the toolbar frame style sheet.

CSSURL_  
contains the URL for the style sheet.

DEBUG_  
contains the application server debug level.

DEFTITLE_  
contains the value of the default title that is specified by the user.

DIMLBLs_  
specifies the list of labels for the down and across dimensions.

DLEVELs_  
specifies the list of drill-down levels for the down variables.

DLSEP_  
contains the download-to-spreadsheet delimiter. The default value is a comma.

DMODEL_  
specifies the four-level name of the data model class.

DOWNDRL_  
specifies the list of drill-down values for the down variables.

DOWNL_  
specifies the down variables list from the application list.

DOWNVARS_  
specifies the list of selected variables for the down dimension.

DPTITLE_  
specifies a flag that indicates whether the drill-path title is displayed.

DTOTAL_  
specifies a flag that indicates whether the down totals are turned on.

EMMDDBMID_  
specifies the identifier of the data model class instance.

EXPFLAG_  
specifies a flag that indicates whether the expands are displayed.

EXPLIST_  
specifies a list that contains sublists for each expand. The sublists are of the form

VAR='VALUE'.

EXPVALS_  
specifies a list that contains the values of the expanded rows only.

EXPVAR_  
specifies the name of the expanded variable.

GRFHT_  
contains the value of the graph height option.

GRFSRC_  
specifies the graph source that is selected by the user, where 1 is a 3-D clickable graph and 2 is a standard GIF graph.

GRFWID_  
contains the value of the graph width option.
GRLOC specifies the graph location that is selected by the user, where 1=bottom, 2=top, 3=left, and 4=right.

GRPHTYPE specifies the graph type selected by the user. Valid types include: BLOCK=block chart, HBAR=horizontal bar chart, PIE=pie chart, PLOT=plot, and VBAR=vertical bar chart.

GRPHVALS specifies a list that contains the data points for the 3-D graph.

HIERL specifies the list of metabase hierarchies.

HMODEL specifies the four-level name of the HOLAP data model class. The default value is SASTOOL_.DMDB.HOLAP_M.CLASS.

HTMLFILE specifies the identifier of the output file for writing HTML.

IMGURL contains the URL for the images.

MDDB specifies the name of the selected MDDB.

METABASE specifies the name of the selected metabase.

ROTFLAG specifies a flag that indicates whether the user selected the Rotate button, where 1=Rotate button was selected and 2=Rotate button was not selected.

SESSIONID specifies the value for the _SESSIONID variable for the application server session.

SHOWTAB specifies a flag that indicates whether to display the table, where 1=yes, 2=no.

STATDESC specifies a list of all possible statistics labels.

STATLIST specifies a list of the available statistics from the metabase.

STATVARS specifies a list of the selected statistics.

SUBHT indicates the number of rows to display in the filter list boxes.

SUBLOC specifies the location of the filter list boxes, where 1=right, 2=left, 3=top, and 4=bottom.

SUBSET_BY specifies the list of selected filter values.

SUBSET_FLAG indicates whether filter values have been selected, where 1=filters have been selected and 0=filters have not been selected.
SUBVARS
specifies the list of selected filter variables.

SUBWID
contains the maximum width (in characters) of the filter list boxes.

TBLOC
specifies the location of the filter toolbar, where 1=top, 2=bottom, 3=left, 4=right, and 5=do not display a toolbar.

THISSESSION
specifies the value for the _THISSESSION variable for the application server session.

USEHOLAP
indicates whether a HOLAP metabase registration is being used, where 1=HOLAP metabase registration is being used and 0=HOLAP metabase registration is not being used.

VMDOFF
specifies a flag that indicates whether metadata verification checking is done on the data model, where any nonblank character=do not perform metadata checking and a blank=perform metadata checking.

Flow of Control in the MDDB Report Viewer Class

The following figures illustrate the flow of control in the MDDB Report Viewer WEBEIS class. For more information about the methods listed in these figures, refer to the individual method descriptions.
Figure 1. Flow of Control for the Layout Page

Initialization and error checking

Create an instance of WEBEIS class

Open _webout for writing HTML

Open repository

Call _OUTPUT_FRAME_HDR2_ method to generate the Dimensions and Options frames

Close repository and _webout, and terminate WEBEIS class

Return

This generates the <FRAMESET> tag for the Dimensions and Options pages, as well as the <FRAME> tags for the toolbar and layout frames.
Flow of Control in the MDDB Report Viewer Class

Figure 2. Flow of Control for the Dimensions Page

Initialization and error checking

Instance WBEIS class

Open repository

Call_GET_DOWNVAR_LIST_

Call_GET_ANALYSIS_VARS_

Call_GETAVAILABLE_STATS_

Open_webout for writing HTML

Call_GETOUTPUT_FILE_ID_

Call_OUTPUT_HDR_

Call_OUTPUT_VARIABLE_SEL/Form_

Output all HTML hidden fields for Dimensions form

Close repository and _webout, and terminate WBEIS class

Return
Figure 4. Flow of Control for the Report Page

- Initialization and error checking
- Create an instance of WEBEIS class
- Open_webout for writing HTML
  - Call_GET_METABASE_NAME_
  - Call_GET_OUTPUT_FILE_ID_
    - Open repository
    - Call_OUTPUT_FRAME_HDR_
    - Close repository
    - Close_webout
    - Terminate WEBEIS class

This generates the <FRAMESET> for the Report page, as well as
Figure 5. Flow of Control for the the Report Page (part 1)

1. Initialization and error checking
   - Create an instance of WEBIBS class
   - Cell_GET_BMDOBMID_
   - Open_webout for writing HTML
   - Cell_GET_METADATA_NAME_
   - Cell_GET_MODE_NAME_
   - Cell_GET_MESSAGE_ID_
   - Cell_GET_USEHELP_
   - Cell_GET_ACROSS_DRILLS_
   - Cell_GET_DOWN_DRILLS_

2. SPDHT macro not blank?
   - Yes: Call_OUTPUT_CSV_CONTENT_HDR_
   - No:
     - Cell_GET_OUTPUT_FILE_ID_
     - Open repository
     - Cell_repository_GET_DATA_ATTRIB_

3. SPDHT macro not blank?
   - Yes: Call_DISPLAY_DEFAULT_TITLE_
   - No:
     - Cell_DISPLAY_TITLE_for Down variable or Down title
     - Cell_DISPLAY_TITLE_for Across variable or Down title

4. MPBODYONLY macro variable blank?
   - No: Call_OUTPUT_SETURL_FUNCTION_
   - Yes:
     - Output dSCRIPTs, and dHEAD's tag

5. Output dHEAD's tag

(Continued)
Figure 5. Flow of Control for the Report Page (part 2)

(Continued)

```
(Continued)

MINOFRAMES not blank
  Yes
  Output <TABLE> tag
  No
  Cell _OUTPUT_TOOLBAR_

Cell _OUTPUT_HTML_AFTER_BODY_
  Yes
  Cell _OPEN_FORM_
  Output <CENTER> tag
  Cell _OPEN_TABLE_
  Filter variable displayed on top or left?
  Yes
  Cell repository _GET_HIERARCHY_
  No
  Cell _SET_HIERL_LIST_

Cell _OUTPUT_SUBSET_SELECTIONS_
  Yes
  SSL macro = 2 or 3
  No
  Cell _DISPLAY_DEFAULT_TITLE_
  Cell _DISPLAY_TITLE_for Down variable dropdown title
  Cell _DISPLAY_TITLE_for A cross variable dropdown title
  Cell _DISPLAY_SUBSET_TITLE_

Cell _SHOW_GRAPH_
  Yes
  GL macro = 2 or 3
  No
  Cell _OUTPUT_DOWN_LIST_

Cell _OUTPUT_AROSS_LIST_
  Yes
  A cross variable selected?
  No
  Cell _OUTPUT_WRAPRT_BUTTON_
```

Note: The elements in the section between (1) and (2) are wrapped in the HTML tags <TR> and <TD> as appropriate for the specified layout structure. For simplification, these tags have been omitted from this diagram.
Figure 5. Flow of Control for the Report Page (part 3)
Figure 6. Flow of Control for the Report Page Toolbar (part 1)

1. Initialization and error checking
2. Create an instance of WEBSITE class
3. Open webout for writing HTML
4. Call _GET_MESSAGE_ID_
5. Call _GET_OUTPUT_FILE_ID_
6. Output text HTML content type header
7. Output <HTML>, <HEAD>, and <TITLE> tags
8. Output file information
9. Output <TABLE> tag
10. Stylesheet parameter specified
    - Yes: Output <LINK> tag with stylesheet information
    - No: Output <SCRIPT> tag
11. Call _OUTPUT_VAR_FUNCTIONS_
12. Call _OUTPUT_ROTATE_FUNCTION_
13. Call _OUTPUT_CLASSVAL_URL_FN_
14. Call _OUTPUT_SETURL_FUNCTION_
15. Call _OUTPUT_ADDTOFAV_FUNCTION_
The MDDB Report Viewer uses macro variables that are set by users and passed into the viewer when the application executes. Table 4.1 on page 31 lists and describes the macro variables.

The MDDB Report Viewer also uses global variables that you can set in the REQUEST INIT program that is used by your application server. Table 4.2 on page 34 lists and describes these variables. For more information about the REQUEST INIT program, see PROC APPSRV, REQUEST Statement syntax.
<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Description</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>MDDB</td>
<td>Selected MDDB</td>
<td>Selected MDDB (for example, SASHELP.PRDMDDB)</td>
</tr>
<tr>
<td>METABASE</td>
<td>Selected metabase</td>
<td>Selected metabase (for example, SASHELP.MBEIS)</td>
</tr>
<tr>
<td>SR</td>
<td>First row to display in the table</td>
<td></td>
</tr>
<tr>
<td>NR</td>
<td>Number of rows to display in the table</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>Down</td>
<td>Hierarchies and category variables. Do not create variable names that contain an embedded percent sign (%) if the percent sign precedes the following characters: 0 through 9, a through f, and A through F. Names that contain these character combinations could be misinterpreted due to encoding and decoding issues. For example, a variable name of Product%20Line could be incorrectly interpreted as Product Line because %20 is the encoding sequence for a blank space.</td>
</tr>
<tr>
<td>Variable Name</td>
<td>Description</td>
<td>Details</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>AC</td>
<td>Across</td>
<td>Hierarchies and category variables. Do not create variable names that contain an embedded percent sign (%) if the percent sign precedes the following characters: 0 through 9, a through f, and A through F. Names that contain these character combinations could be misinterpreted due to encoding and decoding issues. For example, a variable name of Product%20Line could be incorrectly interpreted as Product Line because %20 is the encoding sequence for a blank space.</td>
</tr>
<tr>
<td>A</td>
<td>Analysis variable</td>
<td>Analysis variable. This macro variable is deprecated.</td>
</tr>
<tr>
<td>S</td>
<td>Statistic</td>
<td>Globally applied statistic. This macro variable is deprecated.</td>
</tr>
<tr>
<td>Am</td>
<td>Analysis variable</td>
<td>Analysis variable, where ( m ) designates the particular variable.</td>
</tr>
<tr>
<td>AmSn</td>
<td>Statistic</td>
<td>Statistic that is applied only to the analysis variable specified by ( Am ). ( n ) designates the particular statistic. For example, A1S1 and A1S2 designate statistics that are applied only to the A1 analysis variable.</td>
</tr>
<tr>
<td>SV</td>
<td>Filter variables</td>
<td>Category variables to filter by</td>
</tr>
<tr>
<td>SL</td>
<td>Filter variable values</td>
<td>Values to filter by (for example, SL=COUNTRY:CANADA)</td>
</tr>
<tr>
<td>EX</td>
<td>Expand values</td>
<td>For example, EX=COUNTRY=CANADA</td>
</tr>
<tr>
<td>V</td>
<td>Down dimension drill-down values</td>
<td>For example, V=YEAR=1995</td>
</tr>
<tr>
<td>Variable Name</td>
<td>Description</td>
<td>Details</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------</td>
<td>---------</td>
</tr>
<tr>
<td>VA</td>
<td>Across dimension drill-down values</td>
<td>For example, VA=PRODTYPE= FURNITURE</td>
</tr>
<tr>
<td>ST</td>
<td>Display table</td>
<td>1=yes, 2=no</td>
</tr>
<tr>
<td>DT</td>
<td>Default title</td>
<td>Max length=200</td>
</tr>
<tr>
<td>DP</td>
<td>Show drill-path in title</td>
<td>1=yes, 2=no</td>
</tr>
<tr>
<td>DC</td>
<td>Show down totals</td>
<td>1=yes</td>
</tr>
<tr>
<td>ACB</td>
<td>Show across totals</td>
<td>1=yes</td>
</tr>
<tr>
<td>GSC</td>
<td>Graph source</td>
<td>1=3-D clickable graph, 2=standard GIF graph (SAS/GRAPH software)</td>
</tr>
<tr>
<td>GL</td>
<td>Graph location</td>
<td>1=bottom, 2=top, 3=left, 4=right</td>
</tr>
<tr>
<td>GRT</td>
<td>Graph type</td>
<td>BLOCK, HBAR, PIE, PLOT, VBAR</td>
</tr>
<tr>
<td>BS</td>
<td>Graph bar shapes</td>
<td>STAR, HEXAGON, PRISM, CYLINDER</td>
</tr>
<tr>
<td>SPDSHT</td>
<td>Download to Spreadsheet flag</td>
<td></td>
</tr>
<tr>
<td>GW</td>
<td>Graph width</td>
<td>Default=600, max length=4</td>
</tr>
<tr>
<td>GH</td>
<td>Graph height</td>
<td>Default=450, max length=4</td>
</tr>
<tr>
<td>SSL</td>
<td>Filter list box location</td>
<td>1=right, 2=left, 3=top, 4=bottom</td>
</tr>
<tr>
<td>SW</td>
<td>Filter list box width</td>
<td>Default=15</td>
</tr>
<tr>
<td>SH</td>
<td>Filter list box height</td>
<td>Default=3</td>
</tr>
<tr>
<td>VIEW</td>
<td>View Report button</td>
<td>Value=view report</td>
</tr>
<tr>
<td>GD</td>
<td>Graph down variable</td>
<td>Category variable for graphing</td>
</tr>
<tr>
<td>GA</td>
<td>Graph across variable</td>
<td>Across variable for filtering graph values</td>
</tr>
<tr>
<td>GG</td>
<td>Graph group variable</td>
<td>Graph group-by variable, second innermost down variable</td>
</tr>
<tr>
<td>Variable Name</td>
<td>Description</td>
<td>Details</td>
</tr>
<tr>
<td>---------------</td>
<td>------------------------------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>GSG</td>
<td>Graph subgroup variable</td>
<td>Graph subgroup-by variable, third innermost down variable</td>
</tr>
<tr>
<td>SD</td>
<td>Down variable</td>
<td>Same as D, needed for Filter FORM</td>
</tr>
<tr>
<td>SAC</td>
<td>Across variable</td>
<td>Same as AC, needed for Filter FORM</td>
</tr>
<tr>
<td>CLASS</td>
<td>WEBEIS class name</td>
<td>For subclassing, default is SASHELP.WEBEIS.WEBEI.SCLASS</td>
</tr>
<tr>
<td>CSS</td>
<td>Style sheet URL</td>
<td>Applies to Variable Selection and Report pages</td>
</tr>
<tr>
<td>CSST</td>
<td>Toolbar style sheet URL</td>
<td>Applies to toolbar frame; if not specified, uses CSS value</td>
</tr>
<tr>
<td>BG</td>
<td>Background color or image</td>
<td>Color name, hexadecimal value, or image URL</td>
</tr>
<tr>
<td>BGTYPE</td>
<td>Background type</td>
<td>COLOR, IMAGE</td>
</tr>
</tbody>
</table>

**Table 4.2  MDDB Report Viewer Global Variables**

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Description</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>VMDOFF</td>
<td>Turn VERIFYMD checking off</td>
<td>Any nonblank character turns it off; the default is on</td>
</tr>
<tr>
<td>_GRFONT</td>
<td>SWISSB is the default; use SAS font names</td>
<td></td>
</tr>
<tr>
<td>_MRVHELP</td>
<td>URL of Help file</td>
<td>The default is the Hints and Tips page</td>
</tr>
<tr>
<td>_MRTBLOC</td>
<td>Toolbar location</td>
<td>1=top, 2=bottom, 3=left, 4=right, and 5=none; the default is top</td>
</tr>
<tr>
<td>_MRVSEP</td>
<td>Download to spreadsheet delimiter</td>
<td>A comma is the default</td>
</tr>
<tr>
<td>_MRVTBSC</td>
<td>Toolbar frame scrolling</td>
<td>NO or blank indicates no scrolling; YES adds a scroll bar to frame</td>
</tr>
<tr>
<td>_MRVTBSZ</td>
<td>Toolbar size in pixels</td>
<td>A character string of the form (horizontal, vertical); the default is (50, 125)</td>
</tr>
<tr>
<td>Variable Name</td>
<td>Description</td>
<td>Details</td>
</tr>
<tr>
<td>----------------</td>
<td>------------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>_MRNODIMBOXES</td>
<td>Turns off the down and across list boxes and the View Report button on display</td>
<td>A nonblank value turns this off; the default is on</td>
</tr>
<tr>
<td>_MRNOFRAMES</td>
<td>Indicates whether to use HTML frames in the output</td>
<td>A nonblank value turns this off; the default is on</td>
</tr>
<tr>
<td>_MRNOVARCHECK</td>
<td>Turns off the down and across variable selection error checking</td>
<td>A nonblank value turns this off; the default is on</td>
</tr>
<tr>
<td>_MRBODYONLY</td>
<td>Nonblank</td>
<td>Outputs the HTML between the &lt;BODY&gt; and &lt;/BODY&gt; tags on the Layout and the Report pages</td>
</tr>
<tr>
<td>_MRVFRAMESET</td>
<td>Enables you to specify a custom &lt;FRAMESET&gt; tag on the Report page</td>
<td></td>
</tr>
<tr>
<td>_MRVNOPGOP</td>
<td>Nonblank</td>
<td>Turns off the paging feature</td>
</tr>
<tr>
<td>_MRVRNDX1</td>
<td>Value for the radio button that represents the first number of rows</td>
<td>The default is All</td>
</tr>
<tr>
<td>_MRVRNDX2</td>
<td>Value for the radio button that represents the second number of rows</td>
<td>The default is 25</td>
</tr>
<tr>
<td>_MRVRNDX3</td>
<td>Value for the radio button that represents the third number of rows</td>
<td>The default is 50</td>
</tr>
<tr>
<td>_MRVRNDX4</td>
<td>Value for the radio button that represents the fourth number of rows</td>
<td>The default is 100</td>
</tr>
<tr>
<td>_MRVNRLKS</td>
<td>Minimum number of paging lines to display beneath the report table</td>
<td>The default is 5</td>
</tr>
<tr>
<td>_MRNOSORT</td>
<td>Turns off the sorting feature</td>
<td>A nonblank value turns this off; the default is on</td>
</tr>
<tr>
<td>_MRTBLPRM</td>
<td>Sets report &lt;TABLE&gt; tag parameters</td>
<td>For example, &quot;CELLPADDING=4 CELLS PACING=2 BORDER=3&quot;</td>
</tr>
</tbody>
</table>
MDDB Report Viewer Cascading Style Sheets

The MDDB Report Viewer’s cascading style sheet (CSS) properties enable you to customize the viewer output. You can use cascading style sheets to modify background colors, fonts, and the size and location of the HTML elements and to indicate whether the HTML elements are displayed. For more information about style sheet capabilities, consult your favorite HTML reference guide.

HTML elements use the CLASS parameter to surface style sheet properties. Table 4.3 on page 36 lists the CLASS definitions that are used by the MDDB Report Viewer. An example style sheet is shipped with the viewer, and you can create your own to use as well. To apply a style sheet to the viewer output, specify the CSS parameter as a hidden field on your initial HTML page. For example,

```html
<INPUT TYPE="hidden" NAME="CSS" VALUE="http://myserver/mystyle.css">
```

You can also add the CSS parameter to the URL of bookmarked reports, as in the following (note the URL encoding):

```
&CSS=http%3A//myserver/mystyle.css
```

An additional CSST parameter is provided so that you can apply a separate style sheet to the toolbar frame. If you do not specify the CSST parameter, the toolbar frame uses the value that is specified by the CSS parameter.

<table>
<thead>
<tr>
<th>Class Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAINTAB</td>
<td>Main report table</td>
</tr>
<tr>
<td>ROWLAB</td>
<td>Row label cells</td>
</tr>
<tr>
<td>TROWLAB</td>
<td>Total row label cell</td>
</tr>
<tr>
<td>STROWLAB</td>
<td>Total row label cell for expanded row (for example, “subtotals”)</td>
</tr>
<tr>
<td>TROWCELL</td>
<td>Total row data cells</td>
</tr>
<tr>
<td>TDCELL</td>
<td>All other data cells</td>
</tr>
<tr>
<td>TCOLLAB</td>
<td>Total column label cell</td>
</tr>
<tr>
<td>STCOLLAB</td>
<td>Total column label cell for nested totals</td>
</tr>
<tr>
<td>TCOLCELL</td>
<td>Total column data cells</td>
</tr>
<tr>
<td>COLLAB</td>
<td>Column label cells</td>
</tr>
<tr>
<td>EMPTY</td>
<td>Empty cell in upper left corner</td>
</tr>
<tr>
<td>FILTERBOX</td>
<td>Table containing filter list boxes</td>
</tr>
<tr>
<td>Class Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>DIMBOX</td>
<td>Table containing dimension selector list boxes</td>
</tr>
<tr>
<td>DIMSELBOX</td>
<td>Table containing dimension selector list boxes (Report Layout page)</td>
</tr>
<tr>
<td>ANALYBOX</td>
<td>List box for selecting analysis variable (Report Layout page)</td>
</tr>
<tr>
<td>ANALYSIS</td>
<td>Class for the &lt;DIV&gt; tag for the analysis variable list box</td>
</tr>
<tr>
<td>STATSBOX</td>
<td>List box for selecting statistic (Report Layout page)</td>
</tr>
<tr>
<td>STATS</td>
<td>Class for the &lt;DIV&gt; tag for the statistics list box</td>
</tr>
<tr>
<td>ANALYCOL</td>
<td>Analysis variable column</td>
</tr>
<tr>
<td>STATSCOL</td>
<td>Statistics column</td>
</tr>
<tr>
<td>GRAPH</td>
<td>Class for the &lt;IMG&gt; tag for standard GIF graph</td>
</tr>
<tr>
<td>GRAPHP</td>
<td>Class for the graph application tag</td>
</tr>
<tr>
<td>TOOLTAB</td>
<td>Class for the toolbar</td>
</tr>
<tr>
<td>IMGBKMRK</td>
<td>Class for the &lt;IMG&gt; tag for Bookmark</td>
</tr>
<tr>
<td>IMGDIM</td>
<td>Class for the &lt;IMG&gt; tag for Dimensions</td>
</tr>
<tr>
<td>IMGOPT</td>
<td>Class for the &lt;IMG&gt; tag for Options</td>
</tr>
<tr>
<td>IMGHELP</td>
<td>Class for the &lt;IMG&gt; tag for Help</td>
</tr>
<tr>
<td>IMGLAY</td>
<td>Class for the &lt;IMG&gt; tag for Layout</td>
</tr>
<tr>
<td>IMGLOGOUT</td>
<td>Class for the &lt;IMG&gt; tag for Logout</td>
</tr>
<tr>
<td>IMGROTATE</td>
<td>Class for the &lt;IMG&gt; tag for Rotate</td>
</tr>
<tr>
<td>HEADER</td>
<td>Report Layout HTML headers</td>
</tr>
<tr>
<td>LABEL</td>
<td>Report Layout HTML labels</td>
</tr>
<tr>
<td>SELECT</td>
<td>Report Layout HTML for &lt;SELECT&gt; and &lt;INPUT&gt; tags</td>
</tr>
<tr>
<td>SSELECT</td>
<td>Class for statistics selection list boxes</td>
</tr>
<tr>
<td>SUBMIT</td>
<td><strong>Submit (View Report)</strong> button class</td>
</tr>
</tbody>
</table>
Dictionary

_BUILD_ACROSSLIST_LIST_ Method

Builds the across list (variables in the across dimension) on the application list

Syntax

CALL SEND(OBJID,’_BUILD_ACROSSLIST_LIST_’.application-list,across-variable);

Required Arguments

application-list
the list ID of the application list. For more information about application lists, see the online Help for SAS/EIS software.
Type: Numeric

across-variable
the variable that is selected for the across dimension (optional and no longer used).
Type: Character

Details

This method
• clears the across sublist on the application list
• adds the selected across variables to the across sublist.

Example

acrosvar='Product Line';
rc=insertc(acrvars_,acrosvar,-1);
applist=makelist();
rc=fillist('CATALOG','SASHELP.EISRG.ONEWAY.EIS',applist);
call send(webid,’_BUILD_ACROSSLIST_LIST_’,applist);

The following sublist is added to the application list:

Across:( PRODUCT LINE= ( HIERARCH= 'Product Line' ) )

_BUILD_ANALYSIS_LIST_ Method

Builds the analysis sublist on the application list
Syntax
CALL SEND(OBJID,'_BUILD_ANALYSIS_LIST_','application-list');

Required Argument
application-list
the list ID of the application list. For more information about application lists, see the online Help for SAS/EIS software.
Type: Numeric

Details
This method
• clears the analysis sublist on the application list
• adds the selected analysis variables to the analysis sublist.

Example
applist= makelist();
rc= fillist('CATALOG','SASHELP.EISRG.ONEWAY.EIS',applist);
call send(webid,'_BUILD_ANALYSIS_LIST_',applist);
The following sublist is added to the application list:
Analysis:{ ACTUAL= ()[1083] )[985]

_BUILD_ANLSORTORDER_ Method
Updates the ANLSORTORDER sublist on the application list that is used to specify an analysis/statistic column sort

Syntax
CALL SEND(OBJID,'_BUILD_ANLSORTORDER_','application-list');

Required Argument
application-list
the list ID of the application list. For more information about application lists, see the online Help for SAS/EIS software.
Type: Numeric

Example
applist= makelist();
rc=fillist('CATALOG','SASHELP.EISRG.ONEWAY.EIS',applist);
call send(webid,'_BUILD_ANLSORTORDER_',applist);
BUILD_APPLICATION_LIST Method

Builds the application list for the data model

Syntax

CALL SEND(OBJID,'BUILD_APPLICATION_LIST', application-list, metabase-id, catalog-entry, down-variable, across-variable);

Required Arguments

application-list
the list ID of the application list.
Type: Numeric

metabase-id
the ID number of the metabase.
Type: Numeric

catalog-entry
the catalog entry of the Report Gallery template.
Type: Character

down-variable
the variable that is selected for the down dimension (optional). (This parameter is included for compatibility with previous releases of this application.)
Type: Character

across-variable
the variable that is selected for the across dimension (optional). (This parameter is included for compatibility with previous releases of this application.)
Type: Character

Details

This method
• copies the Report Gallery Template application list
• changes the table name on the application list to the selected MDDB
• replaces the metabase name on the application list with the selected metabase
• calls _BUILD_DOWNL_LIST to add the selected down variables to the application list
• calls _BUILD_ACROSS_LIST to add the selected across variables to the application list (if necessary)
• calls _BUILD_ANALYSIS_LIST to add the selected analysis variables to the application list
• calls _BUILD_STATS_LIST to add the selected statistics to the application list
• calls _CLEAR_POPUP_ to clear the unneeded popup_l sublist on the application list
• calls _BUILD_TOTAL_ to turn report totals on for the down variables
• calls _BUILD_TOTAL_ to turn report totals on for the across variables (if necessary).

For more information about the structure of application lists, see the online Help for SAS/EIS software.

Example

```plaintext
applist= makelist();
mbid= instance(loadclass('SASHELP.MB.METABASE.CLASS'));
centry= 'SASHELP.EISRG.ONEWAY.EIS';
downvar= 'Geographic';
rc=insertc(downvars_, downvar, -1);
acrosvar= 'Year';
rc=insertc(acrvars_, acrosvar, -1);
call send(webid,'_BUILD_APPLICATION_LIST_',applist,mbid,centry);
```

_BUILD_CURRENT_SUBSETS_ Method

Updates the saved_l sublist on the application list to define the specified filters

Syntax

CALL SEND(OBJID,'_BUILD_CURRENT_SUBSETS_','application-list,metabase-id);

Required Arguments

(application-list)
the list ID of the application list. For more information about application lists, see the online Help for SAS/EIS software.

Type: Numeric

(metabase-id)
the ID number of the metabase.

Type: Numeric

Details

This method

• builds the HIERARCHIES_L sublist on the SAVED_L list if it is empty
• builds the CURRENT_SUBSETS and CURRENT_DRILLS lists on the HIERARCHIES_L sublist if it is empty
• updates the CURRENT_SUBSETS lists for each hierarchy and class variable with the current filter information.

Example

```plaintext
applist= makelist();
rc=fillist('CATALOG','SASHELP.EISRG.ONEWAY.EIS',applist);
mbid=instance(loadclass('SASHELP.MB.METABASE.CLASS'));
call send(webid,'_BUILD_CURRENT_SUBSETS_','applist,mbid,centry);
```
BUILD DOWNL_LIST Method

Builds the DOWNL sublist on the application list

Syntax

CALL SEND(OBJID,'_BUILD_DOWNL_LIST_',application-list,down-variable);

Required Arguments

application-list
the list ID of the application list. For more information about application lists, see the online Help for SAS/EIS software.

Type: Numeric

down-variable
the selected down variable. (This optional parameter is included for compatibility with previous releases of the MDDB Report Viewer.)

Type: Character

Details

This method
• clears the down sublist on the application list
• adds the selected down variable to the down sublist.

Example

applist= makelist();
rc= fillist('CATALOG','SASHelp.EISRG.ONEWAY.EIS',applist);
downvar='Geographic';
rc=insertc(downvars_,downvar,-1);
call send(webid,'_BUILD_DOWNL_LIST_',applist);

The following sublist is added to the application list:

downl: ( GEOGRAPHIC= 'Geographic')

BUILD_STATSL_LIST Method

Builds the STATSL sublist on the application list

Syntax

CALL SEND(OBJID,'_BUILD_STATSL_LIST_',application-list);
**Required Argument**

*application-list*

the list ID of the application list. For more information about application lists, see the online Help for SAS/EIS software.

*Type:* Numeric

**Details**

This method

- clears the statistics sublist on the application list
- adds the selected statistics to the statistics sublist.

**Example**

```plaintext
applist= makelist();
rc= fillist('CATALOG','SASHELP.EISRG.ONEWAY.EIS',applist);
call send(webid,'_BUILD_STATSLIST_',applist);

The following sublist is added to the application list:

```statsl: ( SUM= 'SUM' )```

---

**BUILD_TOTAL_Method**

Builds the TOTALS sublist on the application list to turn report totals on

**Syntax**

CALL SEND(OBJID,'BUILD_TOTAL',application-list,metabase-id,total-variable);

**Required Arguments**

*application-list*

the list ID of the application list. For more information about application lists, see the online Help for SAS/EIS software.

*Type:* Numeric

*metabase-id*

the ID number of the metabase.

*Type:* Numeric

*total-variable*

the variable that is selected from the down or across dimension.

*Type:* Character

**Example**

```plaintext
applist= makelist();
rc= fillist('CATALOG','SASHELP.EISRG.ONEWAY.EIS',applist);
mbid=instance(loadclass('SASHELP.MB.METABASE.CLASS'));
```
downvar='COUNTRY';
call send(webid,'_BUILD_TOTAL_',applist,mbid,downvar);

The following sublist is added to the application list:

```plaintext
TOTALS: {   DSNAME= 'SASHELP.PRDMDDB'
            MBNAME= 'SASHELP.MBEIS'
            SEL_EXCL= 'CATEGORY'
            MB_AVAIL= 1
            CUSTOM= ( COUNTRY= ( TOTALON= 1
                           LABEL= 'TOTAL'
                           FONT= ( ) [1095]
                      ) [1093]
                ) [1063]
          ) [1061]
```

**_BUILD_URL_ONSUBMIT_ Method**

Outputs the geturl JavaScript function on the Dimensions page

**Syntax**

```plaintext
CALL SEND(OBJID,'_BUILD_URL_ONSUBMIT_','url');
```

**Required Argument**

`url`

the Application Broker component of the URL.

**Type:** Character

**Details**

This function runs when the View Report button is pressed. It builds the URL for the report request.

**Example**

Sample output:

```plaintext
function geturl(down,across,analysis) {
  D0=0; A0=0; AC0=0; var href="/mddbapp.hlp/"; var stats="";
  param=new Object;
  param._SERVICE = "default";
  param._PROGRAM = "sashelp.webeis.showrpt.scl";
  param._DEBUG = "2";
  param.MDDB = "SASHELP.PRDMDDB";
  param.METABASE = "SASHELP";
  param.CSS="http%3A%2F%2Flocalhost%2Fcss%2Fmrv.css";
  param.GRT="NONE";
  param.DC="1";
  param.ACB="1";
  param.ST="1";

  href = "*/cgi-bin/broker.exe?";
```
for (name in param) { href += name + "=" + param[name] + "&" }

href2="";

for (i=0; i<down.options.length; i++) {
    if (down.options[i].selected) {
        D0=eval(D0+1);
        href2+="&D" +D0 +"=" +down.options[i].value;
        if (eval(D0)==1) {
            href2+="&D" +"=" +down.options[i].value;
        }
    }
    href+="D0" +D0 +href2;
}

href2="";

for (i=0; i<across.options.length; i++) {
    if (across.options[i].selected & across.options[i].value!="") {
        AC0=eval(AC0+1);
        href2+="&AC" +AC0 +"=" +across.options[i].value;
        if (eval(AC0)==1) {
            href2+="&AC" +"=" +across.options[i].value;
        }
    }
    href+="&AC0=" +AC0 +href2;
}

href2="";

for (i=0; i<analysis.options.length; i++) {
    if (analysis.options[i].selected) {
        A0=eval(A0+1);
        href2+="&A" +A0 +"=" +analysis.options[i].value;
        if (eval(A0)==1) {
            href2+="&A" +"=" +analysis.options[i].value;
        }
        stats=analysis.options[i].value+"STATS";
        statsarray=eval(stats);
        if (statsarray.length==1 & statsarray[0]=="nunique") {
            href2+="&A" +A0 +"S" +"=" +"NUNIQUE";
        }
        else if (statsarray.length==1 & statsarray[0]!="nunique") {
            href2+="&A" +A0 +"S" +"=" +"SUM";
        }
        else {
            if (statsarray.length == 2) {
                href2+="&A" + A0 + "S=" + statsarray[1];
            }
            else {
                for (j=1; j<statsarray.length; j++)
                    href2+="&A" + A0 + "S" + j +"=" +statsarray[j];
            }
        }
    }
    href+="&A0=" +A0 +href2;
BUILD_WHERE_FORMAT_STRING_ Method
Builds a portion of the WHERE clause that provides the reach-through to detail data, including the variable format

Syntax
CALL SEND(OBJID,'BUILD_WHERE_FORMAT_STRING_',metabase-id,variable-name,in-data-value,out-data-value);

Required Arguments

metabase-id
the ID number of the metabase.
Type: Numeric

variable-name
the name of the variable in the metabase.
Type: Character

in-data-value
the unformatted data value.
Type: Character

out-data-value
the string to add to the reach-through WHERE clause.
Type: Character

Example
mbid=instance (loadclass('SASHELP.MB.METABASE.CLASS'));
myvar='MONTH';
myvalue='Jan';
fmtval=' ';
call send (webid,'BUILD_WHERE_FORMAT_STRING_ ', mbid,myvar,myvalue,fmtval);
The following output is produced:
fmtval=put(MONTH,$MONTH.)='Jan'

CHECK_HIER_MEMBER_ Method
Checks to make sure that one dimension variable (member-var) is not a member of the hierarchy chosen for the other dimension variable (hierarchy-var)
Syntax

CALL SEND(OBJID,'_CHECK_HIER_MEMBER_',metabase-id,error-flag, hierarchy-var, member-var,message);

Required Arguments

metabase-id
the ID number of the metabase.
Type: Numeric

error-flag
an error flag, where 0=no error, and 1=error.
Type: Numeric

hierarchy-var
the hierarchy variable.
Type: Character

member-var
the member variable.
Type: Character

message
the error message to display.
Type: Character

Details

This method ensures that the variables users select to create a report are valid. For example, specifying `DOWN=COUNTRY, ACROSS=GEOGRAPHIC` produces an error if country is a member of the geographic hierarchy.

Example

mbid=instance(loadclass('SASHELP.MB.METABASE.CLASS'));
downvar='Geographic';
acrosvar='COUNTRY';
call send(webid,'_CHECK_HIER_MEMBER_',mbid,varerr,downvar,acrosvar,msg);

_CLOSE_FORM_ Method

Outputs the closing variable selection form tags

Syntax

CALL SEND(OBJID,'_CLOSE_FORM_ ',initial-url,service-name, metabase-name, background-type,background-value, title,webeis-class);

Required Arguments

initial-url
the URL of the initial HTML page.
Type: Character

service-name
the Application Broker service value.

Type: Character

metabase-name
the metabase name.

Type: Character

background-type
an optional background type (IMAGE or COLOR).

Type: Character

background-value
an optional background value.

Type: Character

title
the HTML page title.

Type: Character

webcls-class
the WEBEIS class name.

Type: Character

Details
This method outputs
• the </FORM> tag
• the link back to the initial HTML page.

Example

mddblink= "http://www.test.com/mddbpage.html';
service= 'default';
metabase= 'SASHELP.MBEIS';
bgtype= 'COLOR';
bg= 'YELLOW';
title= 'Third Quarter Sales Reports';
webcls= 'SASHELP.WEBCAT.MYWEB.CLASS';
call send(webid,'_CLOSE_FORM_',mddblink,service,metabase,
        bgtype,bg,title,webcls);

The following output is produced:

</TD></TR>
</FORM>
</TD></TR>

_CLOSE_PAGE_ Method

Outputs the </TABLE>, </BODY>, and </HTML> tags
Syntax
CALL SEND(OBJID,'_CLOSE_PAGE_');

Example
The following output is produced:

```html
</TABLE>
</BODY>
</HTML>
```

__CLOSE_STATIC_FORM__ Method

Outputs the Next button and the closing </TABLE>, </FORM>, </BODY>, and </HTML> tags for the initial HTML page

Syntax
CALL SEND(OBJID,'_CLOSE_STATIC_FORM_');

Example
The following output is produced:

```html
<TR><TD>&nbsp;</TR></TD>
<TR><TD colspan=2 align=center>
  <INPUT TYPE= "submit" VALUE= "Next">
</TABLE>
</FORM>
</BODY>
</HTML>
```

__CREATE_STAT ARRAYS__ Method

Outputs the stats JavaScript function and the associated statistics JavaScript arrays on the Dimensions page

Syntax
CALL SEND(OBJID,'_CREATE_STAT ARRAYS_');

Details
This function updates the list of displayed available and selected statistics based on the selected analysis variable.
Example

The following output is produced:

```javascript
var ACTUALSTATS= new Array(
"analysis",
"NMISS",
"N",
"SUM",
"MIN",
"MAX",
"USS",
"RANGE",
"AVG",
"CSS",
"VAR",
"STD",
"STDERR",
"CV",
"T",
"PRT",
"LCLM",
"UCLM",
"PCTSUM",
"PCTN",
);

var DIFFSTATS= new Array(
"computed",
"MAX",
"MIN",
"PCTN",
"PCTSUM",
"SUM",
"N",
);

var PREDICTSTATS= new Array(
"analysis",
"NMISS",
"N",
"SUM",
"MIN",
"MAX",
"USS",
"RANGE",
"AVG",
"CSS",
"VAR",
"STD",
"STDERR",
"CV",
"T",
"PRT",
"LCLM",
"UCLM",
"PCTSUM",
"PCTN",
);```

Chapter 4 • Making Advanced Customizations to the MDDB Report Viewer
var SALESRATSTATS= new Array(
"computed",
"MAX",
"MIN",
"PCTN",
"PCTSUM",
"SUM",
"N"
);

var statslabellist = new Array();
statslabellist["SUM"]="Sum";
statslabellist["PCTSUM"]="Percent of Sum";
statslabellist["AVG"]="Average";
statslabellist["N"]="Total Number of Nonmissing Values";
statslabellist["PCTN"]="Percent of Total Number";
statslabellist["MIN"]="Minimum";
statslabellist["MAX"]="Maximum";
statslabellist["RANGE"]="Range";
statslabellist["NMISS"]="Total Number of Missing Values";
statslabellist["STD"]="Standard Deviation";
statslabellist["STDERR"]="Standard Error of Mean";
statslabellist["LCLM"]="Lower Confidence Limit";
statslabellist["UCLM"]="Upper Confidence Limit";
statslabellist["USS"]="Uncorrected Sum of Squares";
statslabellist["CSS"]="Corrected Sum of Squares";
statslabellist["VAR"]="Variance";
statslabellist["CV"]="Coefficient of Variation";
statslabellist["T"]="T Value";
statslabellist["PRT"]="Probability of Greater Absolute Value";
statslabellist["SUMWGT"]="Sum of Weights";
statslabellist["UWSUM"]="Unweighted Sum";
statslabellist["NUNIQUE"]="Nunique";
statslabellist["MIXED"]="*MIXED SELECTIONS";

analysisdesclist = new Array(
"SUM",
"PCTSUM",
"AVG",
"N",
"PCTN",
"MIN",
"MAX",
"RANGE",
"NMISS",
"STD",
"STDERR",
"LCLM",
"UCLM",
"USS",
"CSS",
"VAR",
"CV"
computeddesclist = new Array("MAX", "MIN", "PCTN", "PCTSUM", "SUM", "N");

cmunquedesclist = new Array("SUM");
nunquedesclist = new Array("NUNIQUE");

var vararrayname = new Array();
num = 0;

//STATS
function stats(select, statbox) {
    var vararrayname="";
    var varstatsstring="";
    var allstatsstring="";
    var allstatsarray="";
    for (i=0; i < select.options.length; i++) {
        if (select.options[i].selected) {
            vararrayname=select.options[i].value+"STATS";
            varstatsstring=eval(vararrayname).toString();
            if (num==1) {
                varstatsstring=eval(vararrayname)[0];
                for (j=0; j < statbox.length; j++) {
                    if (statbox.options[j].text!="" && statbox.options[j].value!="ACTUALSTATS") {
                        varstatsstring+="," +statbox.options[j].value;
                    }
                }
            } else {
                if (num>1) {
                    allstatsarray=eval(vararrayname[0]+"desclist");
                    allstatsstring=allstatsarray.toString();
                    if (""!=statbox.options[j].text!="" && ""!=statbox.options[j].value) {
                        varstatsstring+="," +statbox.options[j].value;
                    }
                }
            }
            temparray=varstatsstring.split",";
            if (""!=vararrayname) {
                ACTUALSTATS.length=temparray.length;
            }
        }
    }else {
        if (num>1) {
            allstatsarray=eval(vararrayname[0]+"desclist");
            allstatsstring=allstatsarray.toString();
            if (""!=statbox.options[j].text!="" && ""!=statbox.options[j].value) {
                varstatsstring+="," +statbox.options[j].value;
            }
        }
    }
}

"T", "PRT", "SUMWGT", "UNSUM";

computeddesclist = new Array("MAX", "MIN", "PCTN", "PCTSUM", "SUM", "N");

cmunquedesclist = new Array("SUM");
nunquedesclist = new Array("NUNIQUE");

var vararrayname = new Array();
num = 0;

//STATS
function stats(select, statbox) {
    var vararrayname="";
    var varstatsstring="";
    var allstatsstring="";
    var allstatsarray="";
    for (i=0; i < select.options.length; i++) {
        if (select.options[i].selected) {
            vararrayname=select.options[i].value+"STATS";
            varstatsstring=eval(vararrayname).toString();
            if (num==1) {
                varstatsstring=eval(vararrayname)[0];
                for (j=0; j < statbox.length; j++) {
                    if (statbox.options[j].text!="" && statbox.options[j].value!="ACTUALSTATS") {
                        varstatsstring+="," +statbox.options[j].value;
                    }
                }
            } else {
                if (num>1) {
                    allstatsarray=eval(vararrayname[0]+"desclist");
                    allstatsstring=allstatsarray.toString();
                    if (""!=statbox.options[j].text!="" && ""!=statbox.options[j].value) {
                        varstatsstring+="," +statbox.options[j].value;
                    }
                }
            }
            temparray=varstatsstring.split",";
            if (""!=vararrayname) {
                ACTUALSTATS.length=temparray.length;
            }
        }
    }else {
        if (num>1) {
            allstatsarray=eval(vararrayname[0]+"desclist");
            allstatsstring=allstatsarray.toString();
            if (""!=statbox.options[j].text!="" && ""!=statbox.options[j].value) {
                varstatsstring+="," +statbox.options[j].value;
            }
        }
    }
}
for (k=0; k < temparray.length; k++)
    ACTUALSTATS[k]=temparray[k];
}
else if (*DIFFSTATS*==vararrayname) {
    DIFFSTATS.length=temparray.length;
    for (k=0; k < temparray.length; k++)
        DIFFSTATS[k]=temparray[k];
}
else if (*PREDICTSTATS*==vararrayname) {
    PREDICTSTATS.length=temparray.length;
    for (k=0; k < temparray.length; k++)
        PREDICTSTATS[k]=temparray[k];
}
else if (*SALESRATSTATS*==vararrayname) {
    SALESRATSTATS.length=temparray.length;
    for (k=0; k < temparray.length; k++)
        SALESRATSTATS[k]=temparray[k];
}
} //STATS

_DISPLAY_ACROSS_CELLS_ Method
Displays the values for the across dimension

Syntax
CALL SEND(OBJID,'_DISPLAY_ACROSS_CELLS_','column-list,actions-list,view-report-flag,
    analysis-variable,statistic-variable,across-variable,url,argument-string,argument-string2,
    initial-url,background-type,background-value,title,webeis-class,dlflag,service);

Required Arguments

column-list
the column list from the EMDB_M class.
Type: Numeric

actions-list
the actions sublist for drill down.
Type: Numeric

view-report-flag
the flag for the View Report button.
Type: Numeric

analysis-variable
the analysis variable to graph.
Type: Character

statistic
the statistic to graph.
Type: Character
**across-variable**
the analysis variable for graphing.
**Type:** Character

**url**
the Application Broker component of the URL.
**Type:** Character

**argument-string**
the argument string for the next query.
**Type:** Character

**argument-string2**
the argument string for the next query.
**Type:** Character

**initial-url**
the URL of the initial HTML page. (This parameter is obsolete. It is included in the METHOD statement so that overrides are not broken.)
**Type:** Character

**background-type**
the optional background type (IMAGE or COLOR).
**Type:** Character

**background-value**
the optional background value.
**Type:** Character

**title**
the HTML page title.
**Type:** Character

**webeis-class**
the WEBEIS class name.
**Type:** Character

**dlflag**
a flag that indicates whether to download the table to a spreadsheet, where 0=output HTML tags with data values, and 1=output data values with spreadsheet delimiters.
**Type:** Numeric

**service**
the service name.
**Type:** Character

### Details
This method
- calls the 
  _SET_ACTIVE_VALUE_ method of the EMDDB_M class
- calls the 
  _SET_ACTION_STATUS_ method of the EMDDB_M class
- outputs the class values for the across dimension with <A> tags for drill down, if drill down is valid.
Example

emddbmid= instance(loadclass('SASHELP.WEBEIS.EMDB_M.CLASS'));
collist= makelist();
call send(emddbmid,'_GET_CLASS_COMBINATIONS_','COL',collist);
actionsl= makelist();
rc= insertc(actionsl,'','-1','CL_DRILL');
vrflag= 1;
grphvar= 'Actual Sales';
grphstat= 'Sum';
grphacr= 'Month';
url= 'cgi-bin/broker?_PROGRAM=SASHELP.WEBEIS.MDDBRPTS.SCL&_SERVICE=
default&Debug=&0GRT=BLOCK';
args= '&MDDB=SASUSER.SALES&METABASE=SASUSER.NEWMB&COUNTRY&C=ACTUAL&D=MONTH&
A0=PREDICT';
args2= '&S0=SUM&S1=AVG';
bgtype= 'COLOR';
bg= 'YELLOW';
title= '1996 Sales Reports';
webcls= 'SASHELP.WEBEIS.WEBEIS.CLASS';
dlflag=0;
services='DEFAULT';
call send(webid,'_DISPLAY_ACROSS_CELLS_','COL',collist,actionsl,vrflag,gaphvar,
grphstat,gaphacr,url,args,args2,' ',bgtype,bg,title,webcls,dlflag,service);
The following output is produced:

<table>
<thead>
<tr>
<th>Month</th>
<th>Jan</th>
<th>Feb</th>
<th>Mar</th>
<th>Apr</th>
<th>May</th>
<th>Jun</th>
<th>Jul</th>
<th>Aug</th>
<th>Sep</th>
<th>Oct</th>
<th>Nov</th>
<th>Dec</th>
<th>Total</th>
</tr>
</thead>
</table>

_DISPLAY_ANALYSIS_VARS_ Method

Outputs the chosen analysis variables to the report table

Syntax

CALL SEND(OBJID,'_DISPLAY_ANALYSIS_VARS_','column-list,dlflag);
**Required Arguments**

**column-list**

the column list from the _EMDB_M_ class.

*Type:* Numeric

**dlflag**

a flag that indicates whether to download the table to a spreadsheet.

*Type:* Numeric

**Example**

The following output is produced:

```html
<TR>
<TH COLSPAN=2 CLASS="analycol">
<DIV CLASS="analysis">
<Select NAME="A" CLASS="ANALYBOX" onChange="submit();">
<OPTION SELECTED VALUE=ACTUAL> Actual Sales
<OPTION VALUE=DIFF> Sales Lag
<OPTION VALUE=PREDICT> Predicted Sales
<OPTION VALUE=SALESRAT> Sales Ratio
</SELECT>
</DIV>
</TH>
<TH COLSPAN=2 CLASS="analycol">
<DIV CLASS="analysis">
<Select NAME="A" CLASS="ANALYBOX" onChange="submit();">
<OPTION VALUE=ACTUAL> Actual Sales
<OPTION VALUE=DIFF> Sales Lag
<OPTION SELECTED VALUE=PREDICT> Predicted Sales
<OPTION VALUE=SALESRAT> Sales Ratio
</SELECT>
</DIV>
</TH>
<TH COLSPAN=2 CLASS="analycol">
ACTUAL SALES
</TH>
<TH COLSPAN=2 CLASS="analycol">
PREDICTED SALES
</TH>
</TR>
```

**_DISPLAY_DEFAULT_TITLE_ Method**

Displays the user-specified title

**Syntax**

```sql
CALL SEND(OBJID,'_DISPLAY_TITLE_','dlflag');
```
Required Argument

dflag
a flag that indicates whether to download the table to a spreadsheet, where 0=output HTML tags with data values and 1=output data values with spreadsheet delimiters.
Type: Numeric

Details
This method
• gets the default title value from the DT macro variable
• outputs the title in HTML format or in comma-separated format, depending on the value of dflag.

Example
dflag=0;
call send(webid,'_DISPLAY_DEFAULT_TITLE_',dflag);

The following output is produced:
<H2>1998 Sales Reports </H2>

_DISPLAY_DOWNVAR_CELL_ Method
Displays the down dimension

Syntax
CALL SEND(OBJID,'_DISPLAY_DOWNVAR_CELL_',row-list,vrflag,analysis-variable,statistic-variable,down-variable,across-variable,_url,_argument-string,_argument-string2,initial-url,service-name,url,background-type,background-value,title,webeis-class,dflag);

Required Arguments
row-list
the row list from the EMDDB_M class.
Type: Character

vrflag
a flag that indicates that the View Report button was pressed.
Type: Character

analysis-variable
the analysis variable to graph.
Type: Numeric

statistic-variable
the statistic to graph.
Type: Character

down-variable
the down dimension variable to graph.
across-variable  
the across dimension variable to graph.
  Type: Character

_url    
the Web browser component of the URL.
  Type: Character

_argument-string  
the argument string for the next query.
  Type: Character

_argument-string2  
the argument string for the next query.
  Type: Character

initial-url  
the URL of the initial HTML page.
  Type: Character

service-name  
the Application Broker service.
  Type: Character

url  
the Application Broker component of the URL.
  Type: Character

background-type  
the background type (IMAGE or COLOR). This value is optional.
  Type: Character

background-value  
the background value. This value is optional.
  Type: Character

title  
the title. This value is optional.
  Type: Character

web-class  
the WEBEIS class name (for subclassing).
  Type: Character

dlflag  
a flag that indicates whether to download the table to a spreadsheet.
  Type: Numeric

Details

If the user has drilled down, this method displays the down dimension cell with an up arrow. This method
• calls _GET_CLASS_LABEL_ of the data model to get the cell label
• outputs the labeled cell with an arrow (if necessary) for drilling up.
Example

call send(emddbmid_, '_GET_CLASS_COMBINATIONS_', 'ROW', rowlist);
vrflag=1;
grphvar='Actual+Sales';
grphstat='Sum';
grphdown='';
grphacr='PRODTYPE';
_url='/cgi-bin/broker?_PROGRAM=sashelp.webeis.mddbrpts.scl&SERVICE=default
&D_DEBUG=0&RPTTYPE=2&GRTYPE=BLOCK
_args='/&MDDB=PERMDATA.MAPINFO&METABASE=PERMDATA.MB612&DOWN=Geographic&ACROSS
=Product+Line&A=ACTUAL'
_args2='&S=SUM&V1=COUNTRY=U.S.A.'
mddblink='http://myserver.com/test.html';
service='default';
url='/cgi-bin/broker';
bgtype='';
bg='';
title='';
webcls='SASHELP.WEBEIS.WEBEIS';
dlflag=0;
call send(webid,'_DISPLAY_DOWNVAR_CELL_ ', rowlist, vrflag, grphvar,
grphstat, grphdown, grphacr, _url, _args, _args2, mddblink, service, url,
bgtype, bg, title, webcls, dlflag);

This example produces the following output:

Example

call send(emddbmid_, '_GET_CLASS_COMBINATIONS_', 'ROW', rowlist);
vrflag=1;
grphvar='Actual+Sales';
grphstat='Sum';
grphdown='';
grphacr='PRODTYPE';
_url='/cgi-bin/broker?_PROGRAM=sashelp.webeis.mddbrpts.scl&SERVICE=default
&D_DEBUG=0&RPTTYPE=2&GRTYPE=BLOCK
_args='/&MDDB=PERMDATA.MAPINFO&METABASE=PERMDATA.MB612&DOWN=Geographic&ACROSS
=Product+Line&A=ACTUAL'
_args2='&S=SUM&V1=COUNTRY=U.S.A.'
mddblink='http://myserver.com/test.html';
service='default';
url='/cgi-bin/broker';
bgtype='';
bg='';
title='';
webcls='SASHELP.WEBEIS.WEBEIS';
dlflag=0;
call send(webid,'_DISPLAY_DOWNVAR_CELL_ ', rowlist, vrflag, grphvar,
grphstat, grphdown, grphacr, _url, _args, _args2, mddblink, service, url,
bgtype, bg, title, webcls, dlflag);

This example produces the following output:

_DISPLAY_ERROR_ Method
Displays an error message on dynamic pages

Syntax

CALL SEND(OBJID,'_DISPLAY_ERROR_','error-message');

Required Argument

error-message
the error message to display.

Type: Character

Example

The following output is produced:
_DISPLAY_ONEWAY_ Method

Calls methods to produce one-way tabular reports

Syntax

CALL SEND(OBJID,'_DISPLAY_ONEWAY_','dlflag');

Required Argument

dlflag

a flag that indicates whether to download the table to a spreadsheet.

Type: Numeric

Details

This method

• checks for selected down and analysis variables and statistics
• calls the '_OPEN_' method of the metabase
• calls the '_GET_HIERARCHY_' method of the metabase to get the list of hierarchies
• calls the '_BUILD_APPLICATION_LIST_' method
• calls the '_BUILD_ARGS_STRING_' method
• calls the '_BUILD_ARGS2_STRING_' method
• calls the '_GET_VARIABLES_' method of the metabase class to get the list of analysis variables
• calls the '_SET_DRILL_LEVELS_' method, if necessary, to drill down to the current level
• calls the '_SET_APPLICATION_' method of the data model
• calls the '_EXPAND_VALUE_' method of the data model for all expanded variables to request the expanded data values
• calls the '_GET_CLASS_COMBINATIONS_' method of the data model to get the row list
• calls the '_GET_CLASS_COMBINATIONS_' method of the data model to get the column list
• calls the '_OUTPUT_DOWN_LIST_' method to output the list of down variables and outputs the HTML tags to format the selection list
• calls the '_OPEN_ONEWAY_' method
• calls the '_DISPLAY_ANALYSIS_VARS_' method
• calls the \_DISPLAY\_DOWNVAR\_CELL\_ method
• calls the \_DISPLAY\_STATISTIC\_VARS\_ method
• calls the \_DISPLAY\_VALUES\_ method.

\_DISPLAY\_ONEWAY\_BLOCK\_ Method
Submits the PROC GCHART statements to produce the one-way block chart

Syntax
CALL SEND(OBJID,'\_DISPLAY\_ONEWAY\_BLOCK\_','statistic,analysis-variable,
          down-variable,dsname,gif-device);

Required Arguments

statistic
    the statistic to graph.
    Type: Character

analysis-variable
    the analysis variable to graph.
    Type: Character

down-variable
    the down dimension variable to graph.
    Type: Character

dlname
    the data set name from the \_WRITE\_ method.
    Type: Character

gif-device
    the device driver name.
    Type: Character

\_DISPLAY\_ONEWAY\_HBAR\_ Method
Submits the PROC GCHART statement to produce the one-way horizontal bar chart

Syntax
CALL SEND(OBJID,'\_DISPLAY\_ONEWAY\_HBAR\_','statistic,analysis-variable,
          down-variable,dsname,gif-device);

Required Arguments

statistic
    the statistic to graph.
    Type: Character
**analysis-variable**
the analysis variable to graph.
**Type:** Character

**down-variable**
the down dimension variable to graph.
**Type:** Character

**dsname**
the data set name from the _WRITE_ method.
**Type:** Character

**gif-device**
the device driver name.
**Type:** Character

---

**_DISPLAY_ONEWAY PIE_ Method**
Submits the PROC GCHART statement to produce the one-way pie chart

---

**Syntax**

CALL SEND(OBJID,'_DISPLAY_ONEWAY_PIE_','statistic,analysis-variable, down-variable,dsname,gif-device);

**Required Arguments**

**statistic**
the statistic to graph.
**Type:** Character

**analysis-variable**
the analysis variable to graph.
**Type:** Character

**down-variable**
the down dimension variable to graph.
**Type:** Character

**dsname**
the data set name from the _WRITE_ method.
**Type:** Character

**gif-device**
the device driver name.
**Type:** Character

---

**_DISPLAY_ONEWAY_VBAR_ Method**
Submits the PROC GCHART statement to produce the one-way vertical bar chart
Syntax
CALL SEND(OBJID,'_DISPLAY_ONEWAY_VBAR_','statistic',analysis-variable,
  down-variable,dsname, gif-device);

**Required Arguments**

*statistic*
  the statistic to graph.
  **Type:** Character

*analysis-variable*
  the analysis variable to graph.
  **Type:** Character

*down-variable*
  the down dimension variable to graph.
  **Type:** Character

*dsname*
  the data set name from the _WRITE_ method.
  **Type:** Character

*gif-device*
  the device driver name.
  **Type:** Character

---

**_DISPLAY_STATISTIC_VARS_ Method**
Outputs the selected statistics to the report table

Syntax
CALL SEND(OBJID,'_DISPLAY_STATISTIC_VARS_','column-list',analysis-variable, _url,
  _argument-string,_argument-string2, initial-url, URL, service, background-type,
  background-value, title, webcls, dfflag, rowlist);

**Required Arguments**

*column-list*
  the column list from the EMDDB_M class.
  **Type:** Numeric

*analysis-variable*
  the analysis variable to graph.
  **Type:** Numeric

*_url*
  the URL of the next query.
  **Type:** Character

*_argument-string*
  the argument string for the next query.
  **Type:** Character
_argument-string2
  the argument string for the next query.
  Type: Character

initial-url
  the URL of the initial HTML page.
  Type: Character

URL
  the Application Broker component of the URL.
  Type: Character

service
  the Application Broker service.
  Type: Character

background-type
  the background type (IMAGE or COLOR). This value is optional.
  Type: Character

background-value
  the background value. This value is optional.
  Type: Character

title
  the HTML page title.
  Type: Character

webcls
  the WEBEIS class name.
  Type: Character

dlflag
  a flag that indicates whether to download the table to a spreadsheet, where 0=output HTML tags with data values and 1=output data values with spreadsheet delimiters. This parameter is optional.
  Type: Numeric

rowlist
  the rowlist from the _GET_CLASS_COMBINATIONS_ method. This parameter is optional.
  Type: Numeric

Details

This method outputs

- a <TH> tag for each statistic in the column list
- a selection list of statistics on the first occurrence of each selected statistic
- an <A> tag followed by an <IMAGE> tag for each statistic if the standard GIF graph is displayed.

Example

call send(emddbid_,'_GET_CLASS_COMBINATIONS_','COL',collist);
call send(emddbid_,'_GET_CLASS_COMBINATIONS_','ROW',rowlist);
_url='_/cgi-bin/scripts?_PROGRAM=SASHHELP.WEBEIS.MDDBRPTS.SCL&SERVICE=default
&_DEBUG=0&RPTTYPE=1&GRTYPE=BLOCK';
_args='&MDDB=PERMDATA.MAPINFO&METABASE=PERMDATA.MB612&DOWN=Geographic&A=ACTUAL';
_args2='&S0=2&S1=SUM&S2=PCTSUM';
mddblink='DYNAMIC';
url='/cgi-bin/broker';
read='default';
bgtype='color';
bg='yellow';
title='';
webcls='SASHELP.WEBEIS.WEBEIS';
call send(webid,'_DISPLAY_STATISTIC_VARS_','collist','',url,_args,_args2,mddblink,url,service,bgtype,bg,title,webcls,dlflag,rowlist);

The example produces the following output:

```html
<br>
<TH CLASS="statscol" VALIGN=BOTTOM><DIV CLASS="stats">
<select NAME="s" CLASS="statsbox" onChange="submit();">
<option VALUE="SUM" SELECTED>Sum
<option VALUE="PCTSUM">% of Sum
<option VALUE="AVG">Average
<option VALUE="N">Total Count
<option VALUE="PCTN">% of Total #
<option VALUE="MIN">Minimum
<option VALUE="MAX">Maximum
<option VALUE="RANGE">Range
</select>
</DIV>
</TH>

<TH CLASS="statscol" VALIGN=BOTTOM><DIV CLASS="stats">
<select NAME="s" CLASS="statsbox" onChange="submit();">
<option VALUE="SUM" SELECTED>Sum
<option VALUE="PCTSUM">% of Sum
<option VALUE="AVG">Average
<option VALUE="N">Total Count
<option VALUE="PCTN">% of Total #
<option VALUE="MIN">Minimum
<option VALUE="MAX">Maximum
<option VALUE="RANGE">Range
</select>
</DIV>
</TH>
```

---

**_DISPLAY_SUBSET_TITLE_ Method**

Displays the applied subsets in a title

---

**Syntax**

```
CALL SEND(OBJID,'_DISPLAY_SUBSET_TITLE_','dlflag');
```

**Required Argument**

`dlflag`

A flag that indicates whether to download the data to a spreadsheet. This parameter is optional.
Type: Numeric

Example
The following output is produced:

```xml
<TABLE><TR><TD><STRONG>Filter by: Country=Canada, Germany Month=Jan,Apr,May</STRONG></TD></TR></TABLE>
```

(DISPLAY_TITLE) Method
Displays the drill titles above the tabular report

Syntax
CALL SEND(OBJID,'DISPLAY_TITLE','srchchar,titlemsg,varname,dlflag');

Required Arguments

`srchchar`
the drill string for the down variable (V) or the across variable (VA).
Type: Character

`titlemsg`
the name of the title message, where the name can be CL_DOWN (for down) or IN_ACROSS (for across).
Type: Character

`varname`
the down or across variable.
Type: Character

`dlflag`
a flag that indicates whether to download the table to a spreadsheet, where 0=output HTML tags with data values and 1=output data values with spreadsheet delimiters. This parameter is optional.
Type: Numeric

Example

dflag=0;
downvar='Geographic';
call send(webid,'DISPLAY_TITLE','V','CL_DOWN',downvar,dlflag);

The following output is produced:

```xml
<TABLE>
<TR><TD><STRONG>Down: Country=CANADA</STRONG></TD></TR></TABLE>
```
_DISPLAY_TWOWAY_ Method

Calls the methods to display the two-way report

Syntax

CALL SEND(OBJID,'_DISPLAY_TWOWAY_','dlflag');

Required Argument

dlflag  
a flag that indicates whether to download the table to a spreadsheet where 0=output HTML tags with data values and 1=output data values with spreadsheet delimiters.

Type: Numeric

Details

This method

• checks for the required variables for a two-way report.
• calls the metabase _GET_HIERARCHY_ method to get a list of hierarchies.
• calls the _BUILD_APPLICATION_LIST_ method.
• calls the _CHECK_HIER_MEMBER_ method.
• calls the _SET_DRILL_LEVELS_ method to drill to the current level.
• calls the EMDDB_M class _SET_APPLICATION_ method.
• calls the _BUILD_ARGS_STRING_ method.
• calls the _BUILD_ARGS2_STRING_ method.
• calls the metabase _GET_VARIABLES_ method to get a list of analysis variables.
• calls the _EXPAND_VALUE_ data model method for all expanded variables.
• calls the EMDDB_M class _GET_CLASS_COMBINATIONS_ method to get the row list.
• calls the EMDDB_M class _GET_CLASS_COMBINATIONS_ method to get the column list.
• calls the _OUTPUT_DOWN_LIST_, _OUTPUT_ACROSS_LIST_, and _OUTPUT_VIEWRPT_BUTTON_ methods to place down and across selection lists and the View Report button above the report. This method also outputs the HTML tags to format these elements on the page.
• calls the _OPEN_TABLE_ method.
• calls the _OPEN_TWOWAY_ method.
• calls the _DISPLAY_ACROSS CELLS_ method.
• calls the _OUTPUT_EMPTY_CELL_ method.
• calls the _DISPLAY_ANALYSIS_VARS_ method.
• calls the _DISPLAY_DOWNVAR_CELL_ method.
• calls the _DISPLAY_STATISTIC_VARS_ method.
- calls the _DISPLAY_VALUES_ method.

---

**_DISPLAY_TWOWAY_BLOCK_ Method**

Submits the SAS/GRAPH PROC GCHART statements to produce the two-way block chart

**Syntax**

```plaintext
CALL SEND(OBJID,'_DISPLAY_TWOWAY_BLOCK_',statistic,analysis-variable,
          down-variable,across-variable,dsname,gif-device,subset-list);
```

**Required Arguments**

- **statistic**
  - the statistic to graph.
  - **Type:** Character

- **analysis-variable**
  - the analysis variable to graph.
  - **Type:** Character

- **down-variable**
  - the down variable to graph.
  - **Type:** Character

- **across-variable**
  - the across variable to graph.
  - **Type:** Character

- **dsname**
  - the data set name from the _WRITE_ method.
  - **Type:** Character

- **gif-device**
  - the device driver name.
  - **Type:** Character

- **subset-list**
  - the initial subset list. This parameter is optional.
  - **Type:** Numeric

---

**_DISPLAY_TWOWAY_HBAR_ Method**

Submits the SAS/GRAPH PROC GCHART statements to produce the two-way horizontal bar chart

**Syntax**

```plaintext
CALL SEND(OBJID,'_DISPLAY_TWOWAY_HBAR_',statistic,analysis-variable,
          down-variable,across-variable,dsname,gif-device,subset-list);
```
Required Arguments

*statistic*
  the statistic to graph.
  **Type:** Character

*analysis-variable*
  the analysis variable to graph.
  **Type:** Character

*down-variable*
  the down variable to graph.
  **Type:** Character

*across-variable*
  the across variable to graph.
  **Type:** Character

*dsname*
  the data set name from the _WRITE_ method.
  **Type:** Character

*gif-device*
  the device driver name.
  **Type:** Character

*subset-list*
  the subset list for the initial graph. This parameter is optional.
  **Type:** Numeric

(DISPLAY_TWOWAY_VBAR_ Method)

Submits the SAS/GRAPH PROC GCHART statements to produce the two-way vertical bar chart

Syntax

CALL SEND(OBJID,'_DISPLAY_TWOWAY_VBAR_','stat,var,down,across, dsname,gifdev, subset-list);

Required Arguments

*stat*
  the statistic to graph.
  **Type:** Character

*var*
  the analysis variable to graph.
  **Type:** Character

*down*
  the down variable to graph.
  **Type:** Character

*across*
  the across variable to graph.
Type: Character
dname
the data set name from the _WRITE_ method.
Type: Character
gifdev
the device driver name.
Type: Character
subset-list
the subset list for the initial graph. This parameter is optional.
Type: Numeric

.DISPLAY_VALUES_ Method
Outputs the numerical values to the report table

Syntax
CALL SEND(OBJID,'_DISPLAY_VALUES_','row-list,column-list,actions-list,metabase-id,'viewreport-flag,'_url','_argument-string','_argument-string2,initial-url,analysis-variable,'statistic-variable,across-variable,background-type,background-value,title,webcls,dlflag);

Required Arguments
row-list
the row list from EMDDB_M.
Type: Numeric
column-list
the column list from EMDDB_M.
Type: Numeric
actions-list
the actions sublist that determines drill-down.
Type: Numeric
metabase-id
the metabase ID number.
Type: Numeric
viewreport-flag
the View Report button flag.
Type: Numeric
_url
the URL of the next query.
Type: Character
_argument-string
the argument string for the next query.
Type: Character
_argument-string2_
the argument string for the next query.
**Type:** Character

_init-url_
the URL of the initial HTML page.
**Type:** Character

_analysis-variable_
the analysis variable to graph.
**Type:** Character

_statistic-variable_
the statistic to graph.
**Type:** Character

_across-variable_
the analysis variable to graph.
**Type:** Numeric

_background-type_
the background type (IMAGE or COLOR). This value is optional.
**Type:** Character

_background-value_
the background value. This value is optional.
**Type:** Character

_titles_
the HTML page title.
**Type:** Character

_webcls_
the WEBEIS class name.
**Type:** Character

_dlflag_
a flag that indicates whether to download the table to a spreadsheet, where 0=output HTML tags with data values and 1=output data values with spreadsheet delimiters.
**Type:** Numeric

_**Details**_

This method
- calls the _GET_DATA_ATTR_ method of the METABASE class to get the base
table name for reach-through
- calls the _GET_EXPANDABLE_CLASS_ method of the data model to get the
expand variable
- calls the EMDDB_M_SET_ACTIVE_VALUE_ method
- calls the EMDDB_M_SET_ACTION_STATUS_ method to validate drill-down
- outputs the class value for the current row
- outputs an <A> tag if drill-down is valid
- outputs the expand link if the expand is valid
- outputs the collapse link if the collapse is valid
• calls the EMDDB_M_GET_VALUES_ method to get the numerical value of the current statistic/analysis pair
• calls the _GET_ANALYSIS_VAR_NAME_ method
• calls the metabase _GET_VAR_ATTR_ method to get the variable attributes
• calls the _GET_RANGE_COLOR_ method if a range is applied
• calls the EMDDB_M_GET_CLASS_FORMAT_ method
• outputs the numerical value to a table cell
• calls the _OUTPUT_REACHTHRU_LINK_ method if the reach-through to detail is valid
• outputs the closing HTML table tag.

**Example**

rowlist=makelist();
call send(emddbmid_, '_GET_CLASS_COMBINATIONS_', 'ROW', rowlist);
collist=makelist();
call send(emddbmid_, '_GET_CLASS_COMBINATIONS_', 'COL', collist);
actionsl=makelist();
rc=insertc(actionsl, '', -1, 'CL_DRILL');
mbid=instance(loadclass('SASHELP.MB.METABASE'));
vrflag=1;
_url='/cgi-bin/broker?_PROGRAM=sashelp.webeis.mddbrpts.scl&_SERVICE=default &DEBUG=0&RPTTYPE=2&GRTYPE=BLOCK';
_args='&MDDB=SASHELP.PRDMDDB&METABASE=SASHELP.MBEIS&D=Geographic&ACROSS =Product+Line&A=ACTUAL';
_args2='&S=SUM';
grphvar='';
grphstat='';
grphacr='PRODTYPE';
bgtype='color';
bg='yellow';
title='';
webcls='SASHELP.WEBEIS.WEBEIS';
dlflag=0;
call send(_self_, '_DISPLAY_VALUES_', rowlist, collist, actionsl, mbid, vrflag,
_url, _args, _args2, mddblink, grphvar, grphstat, grphacr,
bgtype, bg, title, webcls, dlflag);

The following output is produced:

```html
<TR><TH CLASS="rowlab" NOWRAP ROWSPAN=1 COLSPAN=1>
<A href="/mddbapp.hlp/" onClick="this.href=clsurl('V11=COUNTRY=CANADA\&V10=1 &PROGRAM=SASHELP.WEBEIS.OPFRAME.SCL')" TARGET="_top">CANADA</A></TH>
<TD CLASS="tdcell" BGCOLOR=#008000><A href="/mddbapp.hlp/" onClick="this.href=rturl
```
<table>
<thead>
<tr>
<th>Region</th>
<th>Furniture</th>
<th>Office</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Canada</td>
<td>$97,864</td>
<td>$149,126</td>
<td>$246,990</td>
</tr>
<tr>
<td>Germany</td>
<td>$101,194</td>
<td>$144,804</td>
<td>$245,998</td>
</tr>
<tr>
<td>U.S.A.</td>
<td>$91,567</td>
<td>$145,782</td>
<td>$237,349</td>
</tr>
</tbody>
</table>

**Total**: $731,494
_DRILL_TO_LEVEL_ Method

Sets the drill-down values

Details

This method has been replaced by the _SET_DRILL_LEVELS_ method. See the _SET_DRILL_LEVELS_ on page 151 method description for more information.

_GET_ANALYSIS_VAR_NAME_ Method

Returns the name of the analysis variable that is identified by the label

Syntax

CALL SEND(OBJID,'_GET_ANALYSIS_VAR_NAME_','label,varlist,name');

Required Arguments

label
  the long label for an analysis variable.
  Type: Character

varlist
  the list of analysis variables.
  Type: Numeric

name
  the analysis variable name.
  Type: Character

_GET_ANALYSIS_VARS_ Method

Returns the available analysis variables from the metabase and builds the labels list

Syntax

CALL SEND(OBJID,'_GET_ANALYSIS_VARS_','metabase-id');
**_GET_DATA_MODEL_NAME_ Method**

Returns the data model name from the DMODEL_ instance variable

**Syntax**

```plaintext
CALL SEND(OBJID,'_GET_DATA_MODEL_NAME_','model-name');
```

**Required Argument**

`model-name`

the name of the data model to use.

Type: Character

---

**_GET_AVAILABLE_STATS_ Method**

Gets the available statistics from the metabase

**Syntax**

```plaintext
CALL SEND(OBJID,'_GET_AVAILABLE_STATS_','metabase-id');
```

**Required Argument**

`metabase-id`

the metabase ID number.

Type: Numeric

---

**Required Argument**

*metabase-id*

the metabase ID number.

**Details**

This method

- calls the Metabase _GET_VARIABLES_ method
- builds the list of analysis variable labels that is identified by the ANALLBLS_ instance variable.

**Example**

The following output is produced:

```plaintext
anallbls_=['Predicted Sales'
          'Actual Sales']
```

---

**_GET_VARIABLES_ Method**

**Syntax**

```plaintext
CALL SEND(OBJID,'_GET_VARIABLES_','metabase-id');
```

**Required Argument**

*metabase-id*

the metabase ID number.

**Type:** Numeric
__GET_DOWNVAR_LIST__ Method
Builds the down variable list and the dimensions label list

Syntax
CALL SEND(OBJID,'__GET_DOWNVAR_LIST__','metabase-id');

Required Argument
metabase-id
  the metabase ID number.
  Type: Numeric

Details
This method
• calls the metabase __GET_HIERARCHY__ method
• calls the metabase __GET_VARIABLES__ method
• builds the down variable list and the dimensions label list.

__GET_EMDDBMID__ Method
Returns the ID of the data model from the EMDDBMID_ instance variable

Syntax
CALL SEND(OBJID,'__GET_EMDDBMID__','id');

Required Argument
id
  the ID of the data model.
  Type: Numeric

__GET_GRAPH_VALUES__ Method
Gets the numeric values for the 3-D clickable graph

Syntax
CALL SEND(OBJID,'__GET_GRAPH_VALUES__');
Details

The values are stored in the GRPHVALS_ instance variable. Thus, the graph can be displayed with or without the report. This method:

• calls _BUILD_APPLICATION_LIST_ to build the application list
• calls _SET_DRILL_LEVELS_ to set the drill-down subsets
• calls _SET_APPLICATION_ of the data model to get the initial data table
• calls _SET_ACTIVE_VALUE_ and _EXPAND_VALUE_ of the data model for each of the expanded variables (if necessary)
• calls _GET_CLASS_COMBINATIONS_ of the data model to get the row class values
• calls _GET_CLASS_COMBINATIONS_ of the data model to get the column class values
• calls _GET_VALUES_ of the data model for each crossing from the row and column lists
• calls _GET_CLASS_FORMAT_ for the analysis variable to get its format
• adds the class values, the numerical data, and the format to the GRPHVALS_ list.

Example

The GRPHVALS_ instance variable contains the following:

```
( ( COUNTRY='CANADA'
   _ANLSYS_='Actual Sales'
   _STATS_='Sum'
   PRODTYPE='FURNITURE'
   '97864'
   'DOLLAR12.'
 ))[1073]
( COUNTRY='CANADA'
   _ANLSYS_='Actual Sales'
   _STATS_='Sum'
   PRODTYPE='OFFICE'
   '149126'
   'DOLLAR12.'
 ))[227]
( COUNTRY='CANADA'
   _ANLSYS_='Actual Sales'
   _STATS_='Sum'
   PRODTYPE='TOTAL'
   '246990'
   'DOLLAR12.'
 ))[1411]
( COUNTRY='GERMANY'
   _ANLSYS_='Actual Sales'
   _STATS_='Sum'
   PRODTYPE='FURNITURE'
   '101194'
   'DOLLAR12.'
 ))[1631]
( COUNTRY='GERMANY'
   _ANLSYS_='Actual Sales'
   _STATS_='Sum'
   PRODTYPE='OFFICE'
   '257680'
   'DOLLAR12.'
 ))[411]
```
<table>
<thead>
<tr>
<th>Country</th>
<th>Analysis Type</th>
<th>Stats</th>
<th>Prod Type</th>
<th>Value</th>
<th>Currency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Germany</td>
<td>Actual Sales</td>
<td>Sum</td>
<td>Office</td>
<td>144804</td>
<td>Dollar12</td>
</tr>
<tr>
<td>U.S.A.</td>
<td>Actual Sales</td>
<td>Sum</td>
<td>Total</td>
<td>245998</td>
<td>Dollar12</td>
</tr>
<tr>
<td>U.S.A.</td>
<td>Actual Sales</td>
<td>Sum</td>
<td>Furniture</td>
<td>91567</td>
<td>Dollar12</td>
</tr>
<tr>
<td>U.S.A.</td>
<td>Actual Sales</td>
<td>Sum</td>
<td>Office</td>
<td>145782</td>
<td>Dollar12</td>
</tr>
<tr>
<td>Total</td>
<td>Actual Sales</td>
<td>Sum</td>
<td>Total</td>
<td>237349</td>
<td>Dollar12</td>
</tr>
<tr>
<td>Total</td>
<td>Actual Sales</td>
<td>Sum</td>
<td>Furniture</td>
<td>290625</td>
<td>Dollar12</td>
</tr>
<tr>
<td>Total</td>
<td>Actual Sales</td>
<td>Sum</td>
<td>Office</td>
<td>439712</td>
<td>Dollar12</td>
</tr>
<tr>
<td>Total</td>
<td>Actual Sales</td>
<td>Sum</td>
<td>Total</td>
<td>730337</td>
<td>Dollar12</td>
</tr>
</tbody>
</table>
**_GET_MDDDB_NAME_ Method**

Returns the MDDB name from the MDDB_ instance variable

**Syntax**

```
CALL SEND(OBJID,'_GET_MDDDB_NAME_','mddb');
```

**Required Argument**

- `mddb`
  - the MDDB name.
  - **Type:** Character

---

**_GET_MESSAGE_ID_ Method**

Returns the ID of the message class from the DMODEL_ instance variable

**Syntax**

```
CALL SEND(OBJID,'_GET_MESSAGE_ID_','msgid');
```

**Required Argument**

- `msgid`
  - the ID of the message object.
  - **Type:** Numeric

---

**_GET_METABASE_NAME_ Method**

Returns the metabase name from the METABASE_ instance variable

**Syntax**

```
CALL SEND(OBJID,'_GET_METABASE_NAME_','metabase');
```

**Required Argument**

- `metabase`
  - the metabase name.
  - **Type:** Character
_GET_OUTPUT_FILE_ID_ Method
Returns the output file ID from the HTMLFILE_ instance variable

Syntax
CALL SEND(OBJID,'_GET_OUTPUT_FILE_ID_','fileid');

Required Argument
fileid
the ID of the output file.
Type: Numeric

_GET_RANGE_COLOR_ Method
Returns the display color that is defined in the RANGE entry for a numeric value

Syntax
CALL SEND(OBJID,'_GET_RANGE_COLOR_','color,range-list,num');

Required Arguments
color
the display color.
Type: Character
range-list
the RANGE list.
Type: Character
num
the numerical value to search for.
Type: Numeric

_GET_STATDESC_ Method
Returns the ID of the statistics description list from the STATDESC_ instance variable

Syntax
CALL SEND(OBJID,'_GET_STATDESC_','statdesc');
**Required Argument**

*statdesc*

the ID of the list that contains statistics descriptions.

_Type:_ Numeric

---

**_GET_SUBSET_FLAG_ Method**

Returns the value of the SUBSET_FLAG_ instance variable

**Syntax**

CALL SEND(OBJID,'_GET_SUBSET_FLAG_','flagval');

**Required Argument**

*flagval*

the value of the subset flag.

_Type:_ Character

---

**_GET_USEHOLAP_ Method**

Returns the value of the HOLAP flag from the USEHOLAP_ instance variable

**Syntax**

CALL SEND(OBJID,'_GET_USEHOLAP_','useholap');

**Required Argument**

*id*

the ID of the data model.

_Type:_ Numeric

---

**_OPEN_DYNAMIC_FILE_ Method**

Opens the _WEBOUT file for dynamic writing

**Syntax**

CALL SEND(OBJID,'_OPEN_DYNAMIC_FILE_');

---

**_OPEN_FORM_ Method**

Outputs the <FORM> tag for the dynamic HTML pages
Syntax
CALL SEND(OBJID,'_OPEN_FORM_','url,form-name,form-target');

**Required Arguments**

*url*
  - the URL of the next query.
  - **Type:** Character

*form-name*
  - the name of the form. This parameter is optional.
  - **Type:** Character

*form-target*
  - the target window name. This parameter is optional.
  - **Type:** Character

**Details**
For further explanation of the `<FORM>` tag, refer to your favorite HTML reference documentation.

**Example**
CALL SEND (WEBID, 'OPEN_FORM_','/SCRIPTS/BROKER','MYFORM','MENUFORM');

The following output is produced:

```xml
<Form Action="/SCRIPTS/BROKER" Name="MYFORM" Target="MENUFORM">_OPEN_ONEWAY_Method
Opens the one-way report table

**Syntax**
CALL SEND(OBJID,'_OPEN_ONEWAY_','dlflag');

**Required Argument**

*dlflag*
  - a flag that indicates whether to download the table to a spreadsheet.
  - **Type:** Numeric

**Details**
This method
- outputs the `<TABLE>` tag for the report
- outputs the empty cell in the upper left corner of the report.
Example
The following output is produced:

```
<TABLE CLASS="MAINTAB" BORDER=1>
<TR> <TH COLSPAN=2 CLASS="COLLAB" > </TH>
```

__OPEN_STATIC_FILE__ Method
Opens a file in which static HTML is written

**Syntax**

```
CALL SEND(OBJID,'OPEN_STATIC_FILE_\PUREINDEXFILEmsgdest,rc);
```

**Required Arguments**

- `indxfile`  
  the fileref of the file to open.  
  Type: Character

- `msgdest`  
  the destination for error messages. Valid values are LOG or DIALOG.  
  Type: Character

- `rc`  
  the return code for errors (1=error).  
  Type: Numeric

__OPEN_TABLE__ Method
Outputs the `<TABLE>` tag for the dynamic HTML pages

**Syntax**

```
CALL SEND(OBJID,'OPEN_TABLE_',brdrvalue,table-width, border-color-dark, 
border-color-light,background-color, cell-padding, cell-spacing css-class);
```

**Required Arguments**

- `brdrvalue`  
  an optional parameter that specifies the table border thickness.  
  Type: Character

- `table-width`  
  an optional parameter that specifies the width of the table cells (as a percentage of the document width).  
  Type: Character

- `border-color-dark`  
  an optional parameter that specifies a table cell border color attribute.
**Type**: Character

**border-color-light**
an optional parameter that specifies a table cell border color attribute.

**Type**: Character

**background-color**
an optional parameter that specifies the background color of the table.

**Type**: Character

**cell-padding**
an optional parameter that specifies the spacing inside the table cells.

**Type**: Character

**cell-spacing**
an optional parameter that specifies the spacing between the table cells.

**Type**: Character

**css-class**
an optional parameter that specifies the label for a cascading style sheet tag.

**Type**: Character

**Details**
For more information about the `<TABLE>` tag, refer to your favorite HTML reference documentation.

**Example**

CALL SEND (webid, '_OPEN_TABLE_', '3', '50', 'RED', 'YELLOW', 'GRAY', '2', 'mytable');

The following output is produced:

```html
<TABLE BORDER=3 WIDTH=50% BORDERCOLOR=RED BORDERCOLORLIGHT=YELLOW BGCOLOR=GRAY CELLPADDING=2 CELLSPACING=2 CLASS="mytable">
```

---

**_OPEN_TWOWAY_ Method**

Opens the two-way report table

---

**Syntax**

CALL SEND(OBJID,'_OPEN_TWOWAY_','column-list,viewreport-flag,_url,argument-string, _argument-string2,_argument-string3,_url,service,analysis-variable, statistic-variable,across-variable,background-type,background-value,webeis-class,dlflag);

**Required Arguments**

**column-list**
the column list from the _EMDB_M class.

**Type**: Numeric

**viewreport-flag**
the View Report button flag.

**Type**: Numeric
_url_
the Application Broker component of the URL.
_Type_: Character

_argument-string_
the argument string for the next query.
_Type_: Character

_argument-string2_
the argument string for the next query.
_Type_: Character

_argument-string3_
the argument string for the next query.
_Type_: Character

_initial-url_
the URL of the initial HTML page.
_Type_: Character

_url_
the URL for the next query.
_Type_: Character

_service_
the Application Broker service being used.
_Type_: Character

_analysis-variable_
the analysis variable to graph.
_Type_: Character

_statistic-variable_
the statistic to graph.
_Type_: Character

_analysis-variable_
the analysis variable to graph.
_Type_: Numeric

_background-type_
the background type (IMAGE or COLOR). This parameter is optional.
_Type_: Character

_background-value_
the background value. This parameter is optional.
_Type_: Character

_webets-class_
the WEBEIS class name.
_Type_: Character

_dllflag_
a flag that indicates whether to download the table to a spreadsheet, where 0=output HTML tags with data values and 1=output data values with spreadsheet delimiters.
_Type_: Numeric
Details

This method

• outputs the <TABLE> tag
• calls the EMDB_M class _GET_CLASS_LABEL_ method to get the label of the across variable
• outputs the across variable label cell
• outputs the arrow <IMAGE> tag if drill-down has occurred.

_Open_WEBOUT_FOR_SPDSHT_Method

Opens the _WEBOUT file in output mode for the spreadsheet

Syntax

CALL SEND(OBJID,'_OPEN_WEBOUT_FOR_SPDSHT_');

_Output_ACROSS_LIST_Method

Outputs a label and HTML tags for a selection list

Syntax

CALL SEND(OBJID,'_OUTPUT_ACROSS_LIST_','across-variable');

Required Argument

across-variable

the previously selected across variable. This parameter is optional.

Type: Character

Details

This method outputs

• the across label for the selection list
• a <SELECT> tag for the variable list
• an <OPTION> tag for each available variable
• the closing </SELECT> tag.

Example

The following output is produced:

Across:<BR>
<Select NAME="ac" SIZE=3 MULTIPLE onChange="change(document.mF.ac)">
<option VALUE=""></option>
<option SELECTED VALUE=Product+Line>Product Line (hier)
<OPTION VALUE=Geographic>Geographic (hier)
<OPTION VALUE=Time>Time (hier)
<OPTION VALUE=COUNTRY>Country
<OPTION VALUE=COUNTY>County
<OPTION VALUE=MONTH>Month
<OPTION VALUE=PRODTYPE>Product Type
<OPTION VALUE=PRODUCT>Product
<OPTION VALUE=QUARTER>Quarter
<OPTION VALUE=STATE>State/Province
<OPTION VALUE=YEAR>Year
</SELECT>

_OUTPUT_ADDTLCLSVAL_PARMS_Method

Adds additional URL parameters to the JavaScript function

Syntax

CALL SEND(OBJID,"_OUTPUT_ADDTLCLSVAL_PARMS_");

Details

This stub method is called from the _OUTPUT_CLASSVAL_URL_FN_ method.

_OUTPUT_ADDTLRTPARMS_Method

Adds additional URL parameters to the reach-through links

Syntax

CALL SEND(OBJID,'_OUTPUT_ADDTLRTPARMS_');

Details

This stub method is called from the _OUTPUT_REACHTHRU_URL_FN_ method.

_OUTPUT_ADDTOFAV_FUNCTION_Method

Outputs the addtofav JavaScript function on the Toolbar page

Syntax

CALL SEND(OBJID,'_OUTPUT_ADDTOFAV_FUNCTION_');

Details

When a user clicks the Bookmark button, the addtofav function saves the URL in the browser’s bookmark list.
Example

The following output is produced:

```javascript
function addtofav(varName) {
    LinkName=window.document.title;
    with (window.parent.table_window) {
        linkUrl=eval(varName);
    }
    window.external.AddFavorite(linkUrl,LinkName);
}
```

_OUTPUT_ALL_URL_ITEMS_ Method

Outputs the parameters for the getUrl JavaScript function that builds the URL for the report request.

Syntax

```javascript
CALL SEND(OBJID,'_OUTPUT_ALL_URL_ITEMS_','service-name,next-program');
```

Required Arguments

- **service-name**
  - the Application Broker service value.
  - **Type:** Character

- **next-program**
  - the next SCL program to execute.
  - **Type:** Character

_OUTPUT_ANAL_LIST_ Method

Outputs a label and HTML tags for a selection list.

Syntax

```javascript
CALL SEND(OBJID,'_OUTPUT_ANAL_LIST_');
```

Details

This method outputs

- the analysis label for the selection list
- a `<SELECT>` tag for the variable list
- an `<OPTION>` tag for each available variable
- the closing `</SELECT>` tag.

Example

The following output is produced:
**OUTPUT_ANAL_SELECT_ Method**

Outputs the `<SELECT>` tag and OPTIONS for the Analysis variable list box

**Syntax**

CALL SEND(OBJID,'_OUTPUT_ANAL_SELECT_','.tblflag,selvar');

**Required Arguments**

*b *a flag that indicates whether the list is in a table, where 1=the output is in the table and 0=the output is not in the table.

**Type:** Character

*selvar*  
the analysis variable to mark SELECTED.

**Type:** Character

**Example**

The following output is produced:

```html  
<DIV CLASS="analysis">  
<select name="A" multiple size=3>  
<option selected value=ACTUAL>Actual Sales  
<option value=DIFF>Sales Lag  
<option value=PREDICT>Predicted Sales  
<option value=SALESRAT>Sales Ratio  
</select>  
</DIV>
```

**OUTPUT_ARROW_FUNCTIONS_ Method**

Outputs the moveall and movesel JavaScript functions on the Dimensions page

**Syntax**

CALL SEND(OBJID,'_OUTPUT_ARROW_FUNCTIONS_');
Details

The moveall and movesel functions update the available and selected statistics list boxes as the user makes statistic selections for the report display.

Example

The following output is produced:

```javascript
function moveall(fromlistbox, tolistbox) {
  pos=0;
  if (fromlistbox.options.length!=0) {
    pos=tolistbox.options.length;
    for (i=0; i<fromlistbox.options.length; i++) {
      if (fromlistbox.options[i].value!="" && fromlistbox.options[i].value!="MIXED") {
        tolistbox.options[pos]=new Option(statslabellist[fromlistbox.options[i].value],
        fromlistbox.options[i].value);
        pos++;
      }
    }
  }
  fromlistbox.options.length=0;
  stats(document.mf.sa,document.mf.s);
}

function movesel(fromlistbox, tolistbox) {
  pos=0; index=0; newlength=0;
  if (fromlistbox.options.length!=0) {
    pos = tolistbox.options.length;
    var listofstats = new Array();
    j = 0;
    for (i=0; i < fromlistbox.options.length; i++) {
      if (fromlistbox.options[i].selected==false && fromlistbox.options[i].value!="MIXED" && fromlistbox.options[i].text!="") {
        listofstats[j]=fromlistbox.options[i].value;
        j++;
      }
    }
  }
  for (j=0; j < fromlistbox.length; j++) {
    if (fromlistbox.options[j].selected && fromlistbox.options[j].text!="" && fromlistbox.options[j].value!="MIXED") {
      tolistbox.options[pos]=new Option(statslabellist[fromlistbox.options[j].value],
      fromlistbox.options[j].value);
      pos++;
    }
  }
  remstatanal(fromlistbox);
  if (num > 1) {
    j=0;
    fromlistbox.options[j]=new Option(statslabellist["MIXED"],"MIXED");
  } else
    j=-1;
  for (i=0; i < listofstats.length; i++) {
```
j++; if ( j==listofstats.length ) break; else fromlistbox.options[j]=new Option(statslabellist[listofstats[i]], listofstats[i]); }
}
stats(document.mf.sa,document.mf.s);

__OUTPUT_BAR_SHAPE_LIST__ Method
Outputs the graph bar shape option on the Options page

Syntax
CALL SEND(OBJID,'__OUTPUT_BAR_SHAPE_LIST__','bar-shape,vier-report-flag');

Required Arguments

bar-shape
the currently selected graph bar shape.
Type: Character

view-report-flag
the View Report flag.
Type: Numeric

Example
barshape='HEXAGON';
vrfflag=1;
call send(webid,'__OUTPUT_BAR_SHAPE_LIST__',barshape,vrflag);

The following output is produced:
<TD CLASS="label">Bar Shape:
<select NAME="BS" CLASS="select">
<option VALUE=Block>Block
<option VALUE=Cylinder>Cylinder
<option SELECTED VALUE=Hexagon>Hexagon
<option VALUE=Prism>Prism
<option VALUE=Star>Star

__OUTPUT_BOOKMARK_BUTTON__ Method
Outputs the Bookmark button on the toolbar when Access Control is enabled
Syntax
CALL SEND(OBJID,'_OUTPUT_BOOKMARK_BUTTON_');

_OUTPUT_BOOKMARK_URL_ Method
Outputs the bookmarkURL JavaScript string on the Report page for the Bookmark button URL.

Syntax
CALL SEND(OBJID,'_OUTPUT_BOOKMARK_URL_','vrflag',url, service-name, analysis-variable, statistic, down-variable, graph-type, background-type, background-value, title, webeis-class);

Required Arguments

*vrflag*  
the View Report button flag.  
*Type:* Numeric

*url*  
the Application Broker component of the URL.  
*Type:* Character

*service-name*  
the Application Broker service value.  
*Type:* Character

*analysis-variable*  
the analysis variable to graph.  
*Type:* Character

*statistic*  
the statistic to graph.  
*Type:* Character

*down-variable*  
the down variable to graph.  
*Type:* Character

*graph-type*  
the graph type (BLOCK, HBAR, PIE, PLOT, or VBAR).  
*Type:* Character

*background-type*  
the background type (IMAGE or COLOR). This value is optional.  
*Type:* Character

*background-value*  
the background value. This value is optional.  
*Type:* Character

*title*  
the HTML page title.  
*Type:* Character
**webcis-class**
the WEBEIS class name.
**Type:** Character

**Example**

```plaintext
vflag=1;
url='/cgi-bin/broker';
service='default';
grphvar='ACTUAL';
grphstat='SUM';
grphdown='COUNTRY';
grptype='VBAR';
bctype='COLOR';
bg='yellow';
title='1995 Sales Report';
webcls='SASHELP.WEBEIS.WEBEIS';
call send(_self_,'_OUTPUT_BOOKMARK_URL_',vflag,url,service,grphvar,grphstat,
grphdown,grptype,bctype,bg,title,webcls);
```

The following output is produced:

```plaintext
bookmarkURL=
"http://mywebserver/cgi-bin/broker/.csv?_PROGRAM=SASHELP.WEBEIS.SHOWRPT.SCL
&_SERVICE=default&_DEBUG=0&MDB=SASHELP&PRMDDB&METABASE=SASHELP&D=COUNTRY
&AC=YEAR&A=ACTUAL&A1S1=SUM&BCT=COLOR&BG=YELLOW&GRT=VBAR&DC=1&ACB=1
&ST=1&GL=1&GSC=1&SSL=1&SH=3&S=15&GH=450&GW=600&DP=1"
```

---

**_OUTPUT_CLASSVAL_URL_FN_ Method**

Outputs the CLSVAL JavaScript function on the Report page

**Syntax**

```plaintext
CALL SEND(OBJID,'_OUTPUT_CLASSVAL_URL_FN_', service-name, analysis-variable,
statistic, across-variable, by-type, webcls, by-value, URL, title, vflag);
```

**Required Arguments**

**service-name**
the Application Broker service value.
**Type:** Character

**analysis-variable**
the analysis variable to graph.
**Type:** Character

**statistic**
the statistic to graph.
**Type:** Character

**across-variable**
the across variable to graph.
**Type:** Character
background-type
the background type (IMAGE or COLOR). This value is optional.
Type: Character

webcls-class
the WEBEIS class name.
Type: Character

background-value
the background value. This value is optional.
Type: Character

title
the HTML page title.
Type: Character

url
the Application Broker component of the URL.
Type: Character

vrflag
the View Report bottom flag.
Type: Character

Details
This is a stub method.

Example

service= 'default';
grophvar='ACTUAL';
grophstat='SUM'
across='TEAR';
bgtype= 'COLOR';
bg= 'YELLOW';
title= '1995 Sales Report';
webcls= 'SASHELP.WEBCAT.MYWEB.CLASS';
url='/cgi-bin/broker';
vrflag=1;
call send(webid,'_OUTPUT_CLASSVAL_URL_FN_',service,grphvar, grphstat,across,bytype,
  webcls,by,url,title,vrflag)'

The following output is produced:

/_OUTPUTCLICKABLEGRAPH_/ Method
Outputs the <APPLET> tag for the 3-D clickable graph
Syntax

CALL SEND(OBJID,'_OUTPUT_CLICKABLE_GRAPH_','url, service-name, graph-type,
  analysis-variable, statistic, down-variable, across-variable, webcls, bg-type, bg-value,
  bar-shape);

Required Arguments

url
  the Application Broker component of the URL.
  Type: Character

service-name
  the Application Broker service value.
  Type: Character

graph-type
  the graph type (BLOCK, HBAR, PIE, PLOT, or VBAR).
  Type: Character

analysis-variable
  the analysis variable to graph.
  Type: Character

statistic
  the statistic to graph.
  Type: Character

donw-variable
  the down variable to graph.
  Type: Character

across-variable
  the across variable to graph.
  Type: Character

webcls-class
  the WEBEIS class name.
  Type: Character

background-type
  the background type (IMAGE or COLOR). This value is optional.
  Type: Character

background-value
  the background value. This value is optional.
  Type: Character

bar-shape
  the graph bar shape (Block, Cylinder, Hexagon, Prism, or Star).
  Type: Character

Details

In addition, the method outputs the Drive Applet Javascript function that initializes this graph.
Example

```r
url='/cgi-bin/broker';
graphtype=' ';
service= 'default';
grphvar='ACTUAL';
grphstat='SUM';
down='COUNTRY';
avcross='YEAR';
bgtype= 'COLOR';
bgs= 'YELLOW';
title= '1995 Sales Report';
webcls= 'SASHELP.WEBCAT.MYWEB.CLASS';
barshape='Star';
call send(webid, barshape='Star', '_OUTPUT_CLICKABLE_GRAPH_','url, service, 
graphtype, grphvar, grphstat, down, across, webcls, bgtype, 
by, barshape);
```

The following output is produced:

```r
<TD></TR>
</FORM>
</TD></TR>
<TD><HR><A HREF="http://www.test.com/mddbpage.html">Select New
File</A></TD></TR>
```

---

**_OUTPUT_CONTENT_HEADER_ Method**

Outputs the "text/html" content-type header

---

**Syntax**

```
CALL SEND(OBJID,'_OUTPUT_CONTENT_HEADER_');
```

---

**_OUTPUT_CSV_CONTENT_HEADER_ Method**

Outputs the content-type header for the CSV form

---

**Syntax**

```
CALL SEND(OBJID,'_OUTPUT_CSV_CONTENT_HEADER_');
```

---

**_OUTPUT_DEBUG_LIST_ Method**

Outputs a default debug value selection list

---

**Syntax**

```
CALL SEND(OBJID,'_OUTPUT_DEBUG_LIST_');
```
__OUTPUT_DEFRT_TITL_OPTIO_METHOD__

Outputs a text input field to specify a default title

---

**Syntax**

CALL SEND(OBJID,'_OUTPUT_DEFRT_TITL_OPTIO_');

**Example**

The following output is produced:

```
<TD CLASS="label">1998 Sales Report</TD>:
<TD><INPUT NAME="DT" CLASS="SELECT" TYPE=TEXT
    SIZE=30 MAXLENGTH=200></TD>
```

---

__OUTPUT_DIMBTN_URL_FN_METHOD__

Outputs the dimbtnurl JavaScript function in the Dimensions and Options toolbar page

---

**Syntax**

CALL SEND(OBJID,'_OUTPUT_DIMBTN_URL_FN_','url');

**Details**

The dimbtnurl function is called when the **Dimensions** button is pressed.

**Example**

The following output is produced:

```javascript
function dimbtnurl() {
    with (window.parent.main.document.options) {
        var limit = elements.length;
        var href = "'/cgi-bin/broker?PROGRAM=SASHELP.WEBEIS.LAYOUT.SCL";
        for (i=0; i<limit; i++) {
            if (elements[i].value != "") {
                if (elements[i].name == "PROGRAM" || elements[i].name == "VIEW")
                    continue;
                var thisvar=elements[i].name.toUpperCase();
                if (thisvar == "SV") {
                    var sellength = elements[i].options.length;
                    var numselected = 0;
                    for (j=0; j<sellength; j++) {
                        if (elements[i].options[j].selected) {
                            numselected++;
                            if (numselected == 1) {
```
href += "&" + elements[i].name + "=" +
    elements[i].options[j].value;
}
href += "&" + elements[i].name + eval(numselected) + "=" +
    elements[i].options[j].value;
}
}
if (numselected > 0) {
    href += "&" + elements[i].name + "0=" + eval(numselected);
}
else {
    href += "&" + elements[i].name + "=" + elements[i].value;
}
}
return href;

_OUTPUT_DIMENSIONS_BUTTON_ Method
Outputs the <A> and <IMAGE> tags for the Dimensions button on the Layout toolbar page

Syntax
CALL SEND(OBJID,'_OUTPUT_DIMENSIONS_BUTTON_');

Example
The following output is produced:
<A href="../mddbapp.hlp/* onClick="this.href=dimbtnurl();" TARGET="main">
<IMG CLASS="imglay" SRC="http://mywebserver/images/btn_dim.gif" ALT="Dimensions"
    BORDER=0></A>

_OUTPUT_DOWN_LIST_ Method
Outputs a label and HTML tags for a selection list

Syntax
CALL SEND(OBJID,'_OUTPUT_DOWN_LIST_',down-variable,url);

Required Arguments

down-variable
the previously selected down variable. This parameter is optional.

Type: Character
The Application Broker component of the URL. This parameter is optional.

**Type:** Character

**Details**

This method outputs

- the Down label for the selection list
- a `<SELECT>` tag for the variable list
- an `<OPTION>` tag for each available variable
- the closing `</SELECT>` tag.

**Example**

The following output is produced:

```html
Down: <BR>
<select name="d" size=3 multiple onchange="change(document.mF.d)"
<option selected value="Geographic">Geographic (hier)
<option value="Product+Line">Product Line (hier)
<option value="Time">Time (hier)
<option value="Country">Country
<option value="County">County
<option value="Month">Month
<option value="PRODTYPE">Product Type
<option value="PRODUCT">Product
<option value="QUARTER">Quarter
<option value="STATE">State/Province
<option value="YEAR">Year
</select>
```

**_OUTPUT_DP_TITLE_OPTION_ Method**

Outputs radio buttons for the Show Drillpath option in the Table list box

**Syntax**

```javascript
CALL SEND(OBJID,'_OUTPUT_DP_TITLE_OPTION_');
```

**Example**

The following output is produced:

```html
<tr>
    <td class='Label'>Show Drillpath</td>
    <td>
        <input name="DP" class="select" type="radio" value="1" checked>Yes
        <input name="DP" class="select" type="radio" value="2" checked>No
    </td>
</tr>
```
__OUTPUT_DS2HTM_HTML__ Method

Outputs the HTML for the reach-through to the detail data page

Syntax

CALL SEND('OUTPUT_DS2HTM_HTML_',dataset-name,background,url,
    service-name,dataset-member,next-program-library,next-program-catalog,next-program,
    debug-value,where-clause);

Required Arguments

dataset-name
    the base table data set name.
    Type: Character

background
    the HTML background value.
    Type: Character

url
    the Application Broker component of the URL.
    Type: Character

service-name
    the Application Broker service value.
    Type: Character

dataset-member
    the data set name (for example, PRDSALE).
    Type: Character

next-program-library
    the library for the download to spreadsheet program.
    Type: Character

next-program-catalog
    the catalog for the download to spreadsheet program.
    Type: Character

next-program
    the next SCL program to execute to display additional rows of data.
    Type: Character

depug-value
    the Application Broker debug value.
    Type: Character

where-clause
    the WHERE clause to apply to the data.
    Type: Character
Example

dataset='SASHELP.PRDSALE';
bgcolor='BGCOLOR=YELLOW';
url='/cgi-bin/broker';
services='default';
member='PRDSALE';
pgmlib='SASHELP';
pgmcat='WEBEIS';
program='SASHELP.WEBEIS.DS2HTM.SCL';
debug='0';
where='COUNTRY=CANADA';
call send(webid,'_OUTPUT_DS2HTM_HTML_',dataset,bgchar,url,service,member,pgmlib,pdmcat,
program,debug,where);

_OUTPUT_DS2HTM_ST_ Method

Outputs the DS2HTM statement to generate the detail data table

Syntax

CALL SEND (OBJID,'_OUTPUT_DS2HTM_ST_','dataset-name, variable-string,startat,
number-of-rows, total-rows);

Required Arguments

dataset-name
  the base table data set name.
  Type: Character

variable-string
  the selected variables to display, separated by spaces.
  Type: Character

startat
  the starting row to display.
  Type: Numeric

number-of-rows
  the number of rows to display.
  Type: Numeric

total-rows
  the total number of rows of detail data.
  Type: Numeric

Example

dataset='SASHELP.PRDSALE';
varchar='COUNTRY ACTUAL PREDICT';
startat=1;
atatime=50;
numrows=480;
call send(webid,'_OUTPUT_DS2HTM_ST_',dataset,varchar,startat,atime,numrows);

__OUTPUT_DYNAMIC_HIDDEN_FLDSS__ Method

Outputs the necessary hidden fields for the initial dynamic HTML page

Syntax
CALL SEND(OBJID,'_OUTPUT_DYNAMIC_HIDDEN_FLDSS_\',metabase, background-value, background-type,service, debug,title,webeis-class);

Required Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>metabase</td>
<td>the metabase name.</td>
<td>Character</td>
</tr>
<tr>
<td>background-value</td>
<td>the background image URL or color value. This value is optional.</td>
<td>Character</td>
</tr>
<tr>
<td>background-type</td>
<td>the background type (COLOR or IMAGE). This value is optional.</td>
<td>Character</td>
</tr>
<tr>
<td>service</td>
<td>the application server service.</td>
<td>Character</td>
</tr>
<tr>
<td>debug</td>
<td>the debug level.</td>
<td>Character</td>
</tr>
<tr>
<td>title</td>
<td>the HTML page title.</td>
<td>Character</td>
</tr>
<tr>
<td>webeis-class</td>
<td>the WEBEIS class name.</td>
<td>Character</td>
</tr>
</tbody>
</table>

Example

metabase='SASHELP.MBEIS';
bgtype='color';
bgs='yellow';
services='default';
debug=0;
title='1997+Sales+Reports';
webcls='SASHELP.WEBEIS.WEBEIS';
call send(webid,'_OUTPUT_DYNAMIC_HIDDEN_FLDSS_',metabase,bgtype,bg, service,debug,title,webcls);
The following output is produced:

```html
<INPUT TYPE="hidden" NAME="metabase" VALUE="SASHELP.MBEIS">
<INPUT TYPE="hidden" NAME="_program" VALUE="sashelp.webeis.mddbrpts.scl">
<INPUT TYPE="hidden" NAME="bgtype" VALUE="color">
<INPUT TYPE="hidden" NAME="bg" VALUE="yellow">
<INPUT TYPE="hidden" NAME="_service" VALUE="default">
<INPUT TYPE="hidden" NAME="debug" VALUE="0">
<INPUT TYPE="hidden" NAME="title" VALUE="1997+Sales+Reports">
<INPUT TYPE="hidden" NAME="class" VALUE="SASHELP.WEBEIS.WEBEIS">
```

_**OUTPUT_EMPTY_CELL_** Method

Outputs an empty cell in the HTML table

**Syntax**

```sql
CALL SEND(OBJID,'_OUTPUT_EMPTY_CELL_','spannum,dlflag,cssclass');
```

**Required Arguments**

`spannum`
the number of columns to span.

**Type:** Numeric

`dlflag`
a flag that indicates whether to download the table to a spreadsheet where 0output HTML tags with data values and 1output data values with spreadsheet delimiters.

**Type:** Numeric

`cssclass`
the class name for the cascading style sheet class tag. This parameter is optional.

**Type:** Character

_**OUTPUT_EMPTY_SERVICE_LIST_** Method

Outputs an empty service list

**Syntax**

```sql
CALL SEND(OBJID,'_OUTPUT_EMPTY_SERVICE_LIST_');
```

**Details**

This method outputs

- the `<SELECT>` tag
- an example `<OPTION>` tag with comments that instruct users to edit or add `<OPTION>` tags for their services.
**_OUTPUT_GRAPH_DIMS_OPTION_ Method**

Outputs text fields for specifying the graph's width and height

**Syntax**

```
CALL SEND(OBJID,'_OUTPUT_GRAPH_DIMS_OPTION_');
```

**Example**

The following output is produced:

```
<TR><TD CLASS="label">Width</TD><TD><INPUT TYPE=text
NAME="gw" CLASS="select" SIZE=4 MAXLENGTH=4 VALUE="600"></TD></TR>
<TR><TD CLASS="label">Height</TD><TD><INPUT TYPE=text
NAME="gh" CLASS="select" SIZE=4 MAXLENGTH=4 VALUE="450"></TD></TR>
```

**_OUTPUT_GRAPH_INSTR_ Method**

Outputs the Change Graph Type instructions and the Apply button

**Syntax**

```
CALL SEND(OBJID,'_OUTPUT_GRAPH_INSTR_');
```

**Details**

This method outputs
- the Change Graph Type instructions to the HTML
- the Apply button to the HTML.

**_OUTPUT_GRAPH_LIST_ Method**

Outputs the list of graph types

**Syntax**

```
CALL SEND(OBJID,'_OUTPUT_GRAPH_LIST_','grphtype,vrflag');
```

**Required Arguments**

*grphtype*
- the previously selected graph type.

*Type:* Character
vrflag
the View Report button flag, which takes the following values:

1 View Report button click on previous action.
0 No View Report button click on previous action.

Type: Numeric

Details
This method outputs
• the <SELECT> tag
• an <OPTION> tag for each graph type.

Example
The following output is produced:

```
<TR><TD CLASS="label">Type</TD>
<TD><SELECT NAME="grt" CLASS="select">
<Option SELECTED VALUE=NONE>None
<Option VALUE=VBAR>Vertical bar
<Option VALUE=BLOCK>Block
<Option VALUE=HBAR>Horizontal bar
<Option VALUE=PIE>Pie
<Option VALUE=PLOT>Plot
```

_OUTPUT_GRAPH_LOC_OPTION_ Method
Outputs a selection list for the Graph Location option

Syntax
CALL SEND(OBJID,'_OUTPUT_GRAPH_LOC_OPTION_');

Example
The following output is produced:

```
<TR><TD CLASS="label">Location</TD>
<TD><SELECT NAME="gl" CLASS="select"> <OPTION VALUE="1" SELECTED>Bottom
 <OPTION VALUE="2">Top
 <OPTION VALUE="3">Left
 <OPTION VALUE="4">Right
 </SELECT></TD></TR>
```

_OUTPUT_GRAPH_OPTION_ Method
Outputs an OPTION tag for the Graph Type selection list
Syntax
CALL SEND(OBJID,'_OUTPUT_GRAPH_OPTION_','grtype,gormsg,groption');

Required Arguments

grtype
the previously selected graph type.
Type: Character

gmsg
the mnemonic of the graph type message.
Type: Character

groption
the value for the <OPTION> tag.
Type: Character

_OUTPUT_GRAPH_SOURCE_OPTION_ Method
Outputs radio buttons for the Graph Source option

Syntax
CALL SEND(OBJID,'_OUTPUT_GRAPH_SOURCE_OPTION_');

Example
The following output is produced:

    <TR><TD CLASS="label">Graph Source</TD><TD>
    <INPUT NAME="GSC" CLASS="select" TYPE=RADIO VALUE="1" CHECKED>3D Clickable Graph
    <INPUT NAME="GSC" CLASS="select" TYPE=RADIO VALUE="2">Standard GIF Graph
    </TD></TR>

_OUTPUT_GRAPH_TABLE_DISP_ Method
Outputs the check boxes on the Options page for the Display Table and Display Graph options

Syntax
CALL SEND(OBJID,'_OUTPUT_GRAPH_TABLE_DISP_');

Example
The following output is produced:
_OUTPUT_HDR_ Method

Outputs the opening tags for the Report Layout page

Syntax

CALL SEND(OBJID,'_OUTPUT_HDR_','url,background-type,background-value');

Required Arguments

url
the Application Broker component of the URL.
  Type: Character

background-type
the background type (COLOR or IMAGE). This parameter is optional.
  Type: Character

background-value
the background value. This parameter is optional.
  Type: Character

Example

The following output is produced:

<HTML><HEAD><TITLE>MDDB Report Viewer Layout</TITLE></HEAD><script language="javascript">

function List(list) {
    for (key in list)
        if (list[key] != null) this[key]= list[key];
}

selected= new List;
selected2= new List;
function change(select) {
    if ((navigator.appName == "Netscape" &&
        navigator.appVersion.indexOf("3.0") != -1) ||
    (navigator.appName == "Microsoft Internet Explorer" &&
        navigator.appVersion.indexOf("4.0") != -1)) {
        options= new Object;
        for (i= 0; i < select.options.length; i++) {
            options[select.options[i].text]=select.options[i].value;
            selected[select.options[i].text]=
                select.options[i].selected ? select.options[i].value : null;
        }
    }
}

selected= new List(selected);
select.options.length= 0;
for (key in selected)
  select.options[select.options.length]=
    new Option(key, selected[key], false, true);
for (key in options)
  if (selected[key] == null)
    select.options[select.options.length]=
      new Option(key, options[key]);
}
}

function update() {
  str= "";
  for (key in selected)
    str= str + key + ",";
  if (str.length)
    document.form.order.value= str.substring(0, str.length - 1);
}

</SCRIPT>
</HEAD>
<BODY BGCOLOR=white>
<CENTER>
<TABLE CELSPACING=1 BORDER=1>

_OUTPUT_HELP_BUTTON_ Method

Outputs the Help button on the toolbar

Syntax

CALL SEND(OBJID,'_OUTPUT_HELP_BUTTON_:');

Details

This method outputs the HTML tags for the Help button hypertext link and the Help button image.

Example

The following output is produced:

<A HREF="http://support.sas.com/rnd/web/intrnet/mddbapp/hinttips.html" TARGET="_blank">
  <IMG CLASS="imghelp" SRC="/my_images/btn_hlp.gif" ALT="Help" BORDER=0></A>

_OUTPUT_HIDDEN_FIELDS_ Method

Outputs the HTML hidden fields on the tabular report that are necessary for processing the next user action
Syntax

CALL SEND(_OBJID,'_OUTPUT_HIDDEN_FIELDS', across-variable, statistic-variable, analysis-variable, initial-url, service, bgtype, bg, title, webcls);

Required Arguments

*across-variable*
the across value to graph.
*Type:* Character

*statistic-variable*
the statistic to graph.
*Type:* Character

*analysis-variable*
the variable to graph.
*Type:* Character

*initial-url*
the URL of the initial HTML page.
*Type:* Character

*service*
the Application Broker service.
*Type:* Character

*background-type*
the background type (IMAGE or COLOR). This parameter is optional.
*Type:* Character

*background-value*
the background value. This parameter is optional.
*Type:* Character

*title*
the title for the HTML page. This parameter is optional.
*Type:* Character

*webcls*
the WEBEIS class name (for subclassing).
*Type:* Character

Example

The following output is produced:

```html
<INPUT TYPE="hidden" NAME="_SERVICE" value="default">
<INPUT TYPE="hidden" NAME="_DEBUG" value="2">
<INPUT TYPE="hidden" NAME="MDDB" value="SASHELP.PRDMDDB">
<INPUT TYPE="hidden" NAME="METABASE" value="SASHELP.MBEIS">
<INPUT TYPE="hidden" NAME="BGTYPE" value="color">
<INPUT TYPE="hidden" NAME="BG" value="%23FFFFE7">
<INPUT TYPE="hidden" NAME="GRT" value="NONE">
<INPUT TYPE="hidden" NAME="GL" value="1">
<INPUT TYPE="hidden" NAME="GSC" value="1">
<INPUT TYPE="hidden" NAME="SSL" value="1">
<INPUT TYPE="hidden" NAME="ST" value="1">
```
<INPUT TYPE="hidden" NAME="SH" value="3">
<INPUT TYPE="hidden" NAME="SW" value="15">
<INPUT TYPE="hidden" NAME="GH" value="450">
<INPUT TYPE="hidden" NAME="GW" value="600">
<INPUT TYPE="hidden" NAME="ACB" value="1">
<INPUT TYPE="hidden" NAME="DP" value="1">

__OUTPUT_HIDDEN_VARS__ Method
Outs the filter variables, analysis variables, and statistics as HTML hidden fields for the filter form

**Syntax**

CALL SEND(OBJID,'__OUTPUT_HIDDEN_VARS__');

__OUTPUT_HTML_AFTER_BODY__ Method
Enables users to add HTML tags to the Report page

**Syntax**

CALL SEND(OBJID,'__OUTPUT_HTML_AFTER_BODY__');

**Details**

This stub method is called after the <BODY> tag is output for the Report page.

__OUTPUT_HTML_BEF_CLOSE_BODY__ Method
Enables users to add HTML tags to the end of the Report page

**Syntax**

CALL SEND(OBJID,'__OUTPUT_HTML_BEF_CLOSE_BODY__');

**Details**

This stub method is called before the </BODY> tag is output for the Report page.

__OUTPUT_HTML_FORM_HEADER__ Method
Outputs the opening tags for the static HTML page
Syntax

CALL SEND(OBJID,'_OUTPUT_HTML_FORM_HEADER_','title,cgi,background-value,background-type);

Required Arguments

title
an optional title for the page.
Type: Character
cgi
the Application Broker component for the <ACTION> tag.
Type: Character
background-value
the background image URL or color value. This parameter is optional.
Type: Character
background-type
the background type (COLOR or IMAGE). This parameter is optional.
Type: Character

Details

This method
• outputs the opening HTML page tags
• outputs the <BODY> tag with the appropriate background parameters
• outputs a title
• outputs the <FORM> tag.

_Output_LAYOUT_BUTTON_ Method

Outputs the Layout button on the toolbar to enable users to return to the Variable Selection page

Syntax

CALL SEND(OBJID,'_OUTPUT_LAYOUT_BUTTON_');

Details

This method outputs the HTML tags for the Layout button hypertext link and the Layout button image.

Example

The following output is produced:

<A href="../mddbapp.hlp/"
onClick="this.href=clsurl('_PROGRAM=SASHELP.WEBEIS.MDDBRPTS.SCL')" TARGET="_parent">
<IMG CLASS="imglay" SRC="/my_images/btn_lay.gif" ALT="Layout" BORDER=0"></A>
_OUTPUT_LAYOUT_FRAME_ Method

Outputs the <FRAME> tag for the Dimensions page

Syntax

CALL SEND(OBJID,'_OUTPUT_LAYOUT_FRAME_','url','service-name',
  background-type,
  graph-type,background-value, analysis-variable,statistic,down-variable, across-variable);

Required Arguments

url
  the Application Broker component of the URL.
  Type: Character

service-name
  the Application Broker service value.
  Type: Character

background-type
  the background type (IMAGE or COLOR). This parameter is optional.
  Type: Character

graph-type
  the graph type (BLOCK, HBAR, PIE, PLOT, or VBAR).
  Type: Character

background-value
  the background value. This parameter is optional.
  Type: Character

analysis-variable
  the analysis variable to graph.
  Type: Character

statistic
  the statistic to graph.
  Type: Character

down-variable
  the down variable to graph.
  Type: Character

across-variable
  the across variable to graph.
  Type: Character

Example

url='/cgi-bin/broker';
service='default';
graphvar='ACTUAL';
graphstat='SUM';
graphdown='COUNTRY';
grphacr='YEAR';
grptype='VBAR';
bgtype='COLOR';
bgs='YELLOW';
call send(_self_, '_OUTPUT_LAYOUT_FRAME_', url, service,bgtype,grphtype,bg,grphvar,grphstat,
grphdown,grphacr);

The following output is produced:

```html
<FRAME NAME="main"
SRC="/cgi-bin/broker?_program=sashelp.webis.layout.scl&_service=default&_debug=0&mrvdebug=2&mddb=SASHELP.PRDMDDB&metabase=SASHELP&D=COUNTRY&AC=YEAR&A=ACTUAL&A1S1=SUM&GRT=VBAR&BGTYPE=COLOR&BG=YELLOW&GV=ACTUAL&GS=SUM&GD=COUNTRY&GA=YEAR&DC=1&ACB=1">

```

__OUTPUT_LAYOUT_TOOLBAR__ Method

Outputs the Dimensions and Options buttons on the Layout toolbar page

Syntax

CALL SEND(OBJID,'__OUTPUT_LAYOUT_TOOLBAR__');

Example

The following output is produced:

```html
<TR>
<TD>
<A href="../mddbapp.hlp/" onClick="this.href=dimbtnurl();" TARGET="main">
<IMG CLASS="imglay" SRC="http://mywebserver/images/btn_lay.gif" ALT="Dimensions" BORDER=0></A>
</TD>
<TD>
<A href="../mddbapp.hlp/" onClick="this.href=optbtnurl();" TARGET="main">
<IMG CLASS="imglay" SRC="http://mywebserver/images/btn_lay.gif" ALT="Options" BORDER=0></A>
</TD>
</TR>
```

__OUTPUT_LOGOUT_BUTTON__ Method

Outputs the Logout button on the toolbar when access control is enabled

Syntax

CALL SEND(OBJID,'__OUTPUT_LOGOUT_BUTTON__');
_OUTPUT_MAIN_TOOLBAR_FRAME_ Method

Outputs the <FRAME> tag for the toolbar on the Dimensions and Options page

Syntax

CALL SEND(OBJID,'_OUTPUT_MAIN_TOOLBAR_FRAME_','url',service-name, background-type, graph-type, background-value, analysis-variable, statistic, down-variable, across-variable);

Required Arguments

url
the Application Broker component of the URL.
Type: Character

service-name
the Application Broker service value.
Type: Character

background-type
the background type (IMAGE or COLOR). This parameter is optional.
Type: Character

graph-type
the graph type (BLOCK, HBAR, PIE, PLOT, or VBAR).
Type: Character

background-value
the background value. This parameter is optional.
Type: Character

analysis-variable
the analysis variable to graph.
Type: Character

statistic
the statistic to graph.
Type: Character

down-variable
the down variable to graph.
Type: Character

across-variable
the across variable to graph.
Type: Character

Example

url='/cgi-bin/broker';
services='default';
grphvar='ACTUAL';
grphstat='SUM';
grphdown='COUNTRY';
grphacr='YEAR';
grptype='VBAR';
bgtype='COLOR';
bg='YELLOW';
call send(_self_,'_OUTPUT_MAIN_TOOLBAR_FRAME_',url,service,bgtype,grphtype,bg,grphvar,
grphstat,grphdown,grphacr);
The following output is produced:

<FRAME NAME="header" SRC="/cgi-bin/broker?_program=sashelp.webeis.header.scl
&_service=default&_debug=0&mrvdebug=2&mddb=SASHELP.PRMDDDB
&metabase=SASHELP&D=COUNTRY&AC=YEAR&A=ACTUAL
&A1S1=SUM&GRT=VBAR&BGTYPE=COLOR&BG=YELLOW
&GV=ACTUAL&GS=SUM&GD=COUNTRY&GA=YEAR&DC=1
&ACB=1" SCROLLING="NO">

---

_OUTPUT_MDB_LIST_Method

Outputs the list of MDDBs

**Syntax**

CALL SEND(OBJID,'_OUTPUT_MDB_LIST_',mddblist,mddb);

**Required Arguments**

*mddblist*

the list of MDDBs.

Type: Numeric

*mddb*

the currently selected MDDB. This parameter is optional.

Type: Character

**Details**

This method outputs the <SELECT> and <OPTION> tags for selecting an MDDB.

---

_OUTPUT_NUMROWS_LINKS_Method

Outputs the hypertext links beneath a report that enable paging through selected rows in the report

**Syntax**

CALL SEND(OBJID,'_OUTPUT_NUMROWS_LINKS_');

**Example**

The following output is produced:
_OUTPUT_NUMROWS_OPTION_ Method

Outputs the radio buttons to select the number of rows in the report table to display

Syntax

CALL SEND(OBJID,'_OUTPUT_NUMROWS_OPTION_');

Example

The following output is produced:

<table>
<thead>
<tr>
<th>Number of Rows</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
</tbody>
</table>

_OUTPUT_OPTBTN_URL_FN_ Method

Outputs the optbtnurl JavaScript function in the Dimensions and Options toolbar page

Syntax

CALL SEND(OBJID,'_OUTPUT_OPTBTN_URL_FN_',url);

Details

The optbtnurl function is called when the Options button is pressed.

Example

The following output is produced:
function optbtnurl() {
    with (window.parent.main.document.mf) {
        var limit = elements.length;
        href = "/cgi-bin/broker?PROGRAM=SASHELP.WEBEIS.OPTIONS.SCL";
        for (i=0; i<limit; i++) {
            if (elements[i].value != "") {
                if (elements[i].name == "_PROGRAM")
                    continue;
                var thisvar=elements[i].name.toUpperCase();
                if (thisvar == "D" || thisvar == "AC" || thisvar == "A") {
                    var selength = elements[i].options.length;
                    var numselected = 0;
                    for (j=0; j<selength; j++) {
                        if (elements[i].options[j].selected) {
                            numselected++;
                            if (numselected == 1) {
                                href += ";" + elements[i].name + ";" + elements[i].options[j].value;
                            }
                        }
                        if (thisvar == "A") {
                            var href2=""
                            var stats=elements[i].options[j].value+"STATS";
                            var statsstr="window.parent.main." + stats;
                            statsarray=eval(statsstr);
                            if (statsarray.length==1 && statsarray[0] == "nunique") {
                                href2+="&A" + j + ";" + + "NUNIQUE";
                            }
                            else if (statsarray.length==1 && statsarray[0] != "nunique") {
                                href2+="&A" + j + ";" + "SUM";
                            }
                            else {
                                var anum=0;
                                for (k=1; k<statsarray.length; k++) {
                                    anum++;j+1;
                                    href2+="&A" + anum + "S" + k + "=" + statsarray[k];
                                }
                                var numstats = statsarray.length - 1;
                                if (numstats > 1) {
                                    href2+="&A" + anum + "S0" = numstats;
                                }
                            }
                        }
                        href += href2;
                    }
                }
                if (numselected > 0) {
                    href += ";" + elements[i].name + "0" = eval(numselected);
                }
            }
            else {
                href += ";" + elements[i].name + "=" + elements[i].value;
            }
        }
    }
}
 return href;
}

_OUTPUT_OPTIONS_BUTTON_ Method
Outs the <A> and <IMAGE> tags for the Options button on the Layout toolbar page

Syntax
CALL SEND(OBJID,'_OUTPUT_OPTIONS_BUTTON_');

Example
The following output is produced:

<A href="../mddbapp.hlp/" onClick="this.href=optbtnurl();" TARGET="main">
<IMG CLASS="imglay" SRC="http://mywebserver/images/btn_opt.gif" ALT="Options" BORDER=0>
</A>

_OUTOUTPUT_OPTIONS_FORM_ Method
Outs the HTML <FORM> tag for the Options page

Syntax
CALL SEND(OBJID,'_OUTPUT_OPTIONS_FORM_','_url,message-id,graph-type,bar-shape');

Required Arguments

_url
the Application Broker component of the URL.
Type: Character

message-id
the ID of the message system.
Type: Numeric

graph-type
the graph type.
Type: Character

bar-shape
the graph bar shape.
Type: Character

Example
The following output is produced:
_url='cgi-bin/broker';
msgid=instance(loadclass('sashelp.fsp.astmsg.class'),1);
grptype='VBAR';
barshape='HEXAGON';
call send(webid,'_OUTPUT_ OPTIONS_ FORM_',url,msgid,grphtype,barshape);

_OUTPUT_REACHTHRU_LINK_ Method
Outputs the hypertext link for the numeric data in the report to enable reach-through to the detail data

Syntax
CALL SEND(OBJID,'_OUTPUT_REACHTHRU_LINK_ mbid,rowlist,rowndx, collist,coln, curlist);

Required Arguments

mbid
the ID number of the metabase.
_Type: Numeric

rowlist
the row list from the EMDDB_M class.
_Type: Numeric

rowndx
the index of the current row in the rowlist.
_Type: Numeric

collist
the column list from the EMDDB_M class.
_Type: Numeric

coln
the index of the current column in the collist.
_Type: Numeric

curlist
the list of classes and their associated values.
_Type: Numeric

Example
The following output is produced:

<A href='../mddbapp.hlp/* onClick="this.href=rturl(''_WHERE=COUNTRY%3D%22CANADA%22 &_WHERE=PRODTYPE%3D%22FURNITURE%22')"TARGET="_blank">
Syntax

CALL SEND(OBJID,'_OUTPUT_REACHTHRU_URL_FN',service,nextpgm,dataset,bgtype,bg,url);

Required Arguments

service
the Application Broker service.
Type: Character

nextpgm
the four-level name of the program to run to display the detail data. The default is SASHELP.WEBEIS.DS2HTM.SCL.
Type: Character

dataset
the name of the data set that contains the detail data.
Type: Character

bgtype
the background type (IMAGE, COLOR, or blank).
Type: Character

bg
the background value.
Type: Character

url
the Application Broker component of the URL.
Type: Character

Example

The following output is produced:

```javascript
function rturl(str) {
    param=new Object;
    param._PROGRAM = "SASHELP.WEBEIS.VARLIST.SCL";
    param._SERVICE = "default";
    param._DEBUG = "2";
    param.MDDB = "SASHELP.PRDMDDB";
    param.METABASE = "SASHELP.MBEIS";
    param.D = "Geographic";
    param.AC = "Product%20Line";
    param.V10="0";
    param.VA10="0";
    param.A = "ACTUAL";
    param.S = "SUM";
    param.NEXTPGM = "SASHELP.WEBEIS.DS2HTM.SCL";
    param.DATASET = "SASHELP.PRDSALE";
    param.BGTYPE = "color";
    param.BG = "%23FFFFE7";
    href = "*/cgi-bin/broker?";
    for (name in param) { href += name + "=" + param[name] + ";" }
    if (str) {href += str}
    return href;
}
```
_OUTPUT_REPORT_FRAME_ Method

Outputs the <FRAME> tag to create the frame in which the report is displayed

Syntax

CALL SEND(OBJID,'_OUTPUT_REPORT_FRAME_','url,service,bgtype,grphtype,bg,grphvar, grphstat,grphdown,grphacr,debug);

Required Arguments

url
  the Application Broker component of the URL.
  Type: Character

service
  the Application Broker service.
  Type: Character

bgtype
  the background type (IMAGE, COLOR, or blank).
  Type: Character

grphtype
  the selected graph type.
  Type: Character

bg
  the background value.
  Type: Character

grphvar
  the analysis variable to graph.
  Type: Character

grphstat
  the statistic to graph.
  Type: Character

grphdown
  the down dimension variable to graph.
  Type: Character

grphacr
  the across dimension variable to graph.
  Type: Character

debug
  the Application Broker debug value.
  Type: Character

Example

The following output is produced:
Chapter 4 • Making Advanced Customizations to the MDDB Report Viewer

_OUTPUT_REPORT_RADIO_BTNS_ Method

Outs the Report Selection radio buttons

Syntax

CALL SEND(OBJID,'_OUTPUT_REPORT_RADIO_BTNS_');

_OUTPUT_REPORT_TYPE_SELECT_ Method

Outs the Report type selection list

Syntax

CALL SEND(OBJID,'_OUTPUT_REPORT_TYPE_SELECT_','rpttype');

Required Argument

rpttype
  a previously selected report type.
  Type: Character

_OUTPUT_ROTATE_BUTTON_ Method

Outs the Rotate button for the two-dimensional report

Syntax

CALL SEND(OBJID,'_OUTPUT_ROTATE_BUTTON_','viewreport-flag,url,service,initial-url,
  across-variable,down-variable,analysis-variable,statistic-variable,down-variable,
  graph-type,background-type,background-value,title,webgis-class,hideflag');

Required Arguments

viewreport-flag
  the View Report button flag.
  Type: Numeric

url
  the Application Broker component of the URL.
  Type: Character
service
   the Application Broker service.
   **Type:** Character

initial-url
   the URL of the initial HTML page.
   **Type:** Character

across-variable
   the across variable that is selected.
   **Type:** Character

down-variable
   the down variable that is selected.
   **Type:** Character

analysis-variable
   the analysis variable to graph.
   **Type:** Character

statistic-variable
   the statistic to graph.
   **Type:** Character

down-variable
   the down variable to graph.
   **Type:** Numeric

graph-type
   the selected graph type.
   **Type:** Numeric

background-type
   the background type (IMAGE or COLOR). This parameter is optional.
   **Type:** Character

background-value
   the background value. This parameter is optional.
   **Type:** Character

title
   the HTML title page.
   **Type:** Character

webeis-class
   the WEBEIS class name.
   **Type:** Character

hideflag
   a hidden variables flag. If hideflag = 1, variables are not output. This parameter is optional.
   **Type:** Character

Details

This method outputs an HTML form that contains hidden fields that are necessary to process the rotate request and output the **Rotate** submit button.
Example

The following example illustrates the use of this method:

```plaintext
vrflag=1;
_url='/cgi-bin/broker?_PROGRAM=sashelp.webeis.mddbrpts.scl&_SERVICE=default &_DEBUG=0&RPTTYPE=2&GRTYPE=BLOCK';

service='default';
mddblink='DYNAMIC';
across='Geographic';
down='Product+Line';
avar='ACTUAL';
stat='SUM';
grhdown='';
grtype='BLOCK';
bgtype='color';
bg='yellow';
title='';
webcls='SASHELP.WEBEIS.WEBEIS';
hideflag='1';
call send(webid,'_OUTPUT_ROTATE_BUTTON','vrflag,'url,service,mddblink,across,down,avar,stat,grhdown,grtype,bgtype,
bg,title,webcls,hideflag);
```

The following output is produced:

```html
<A href="../mddbapp.hlp/" onClick="this.href=clsurl('ROTATE=1 &_PROGRAM=SASHELP.WEBEIS.SHOWRPT.SCL')"TARGET="_parent">
  <IMG CLASS="imgrotate" SRC="/my_images/btn_rot.gif" ALT="Rotate"> </A>
</A>
```

_OUTPUT_ROTATE_URL_ Method

Outputs the rotateURL JavaScript string on the Report page for the **Rotate** button URL

Syntax

```
CALL SEND(OBJID,'_OUTPUT_ROTATE_URL_','vrflag,url,service-name,analysis-variable,
statistic,down-variable,graph-type,background-type,background-value,title,webeis-class);
```

Required Arguments

- **vrflag**
  - the **View Report** button flag.
  - **Type:** Numeric
- **url**
  - the Application Broker component of the URL.
  - **Type:** Character
- **service-name**
  - the Application Broker service value.
  - **Type:** Character
**analysis-variable**
the analysis variable to graph.
_Type:_ Character

**statistic**
the statistic to graph.
_Type:_ Character

**down-variable**
the down variable to graph.
_Type:_ Character

**graph-type**
the graph type (BLOCK, HBAR, PIE, PLOT, or VBAR).
_Type:_ Character

**background-type**
the background type (IMAGE or COLOR). This parameter is optional.
_Type:_ Character

**background-value**
the background value. This parameter is optional.
_Type:_ Character

**title**
the HTML page title.
_Type:_ Character

**webcls-class**
the WEBEIS class name.
_Type:_ Character

**Example**
This example illustrates the use of the method:

```plaintext
vrf lag=1;
url='/cgi-bin/broker';
service='default';
grphvar='ACTUAL';
grphstat='SUM';
grphdown='COUNTRY';
grptype='VBAR';
bgtype='COLOR';
bg='yellow';
title='1995 Sales Report';
webcls='SASHELP.WEBEIS.WEBEIS';
call send(_self_,'_OUTPUT_ROTATE_URL_',vrf lag,url,service,grphvar,grphstat,
grphdown,grptype,bgtype,bg,title,webcls);
```

The following output is produced:

```
rotateURL="http://mywebserver/cgi-bin/broker/.csv?_PROGRAM=SASHELP.WEBEIS.OPRPT.SCL
&ROTATE=1&_SERVICE=default&_DEBUG=0&MDB=SASHELP.PRDMMDB&METABASE=SASHELP&D
=COUNTRY&AC=YEAR&A=ACTUAL&AI=S1=SUM&GRT=VBAR&DC=1&ACB=1&ST=1&GL=1&GSC=1
&SSL=1&SH=3&SW=15&GH=450&GW=600&DP=1"'
```
**_OUTPUT_SETURL_FUNCTION_ Method**

Outputs the seturl JavaScript function in the toolbar page

**Syntax**

CALL SEND(OBJID,'_OUTPUT_SETURL_FUNCTION_');

**Details**

This function is called when either the **Rotate** button or the **Download to Spreadsheet** button is pressed.

**Example**

The following output is produced:

```javascript
function setURL(varName) {
    newURL='';
    with (window.parent.frames[1]) {
        newURL=eval(varName);
    }
    if (varName == 'downloadURL')
        document.location=newURL;
    else if (varName == 'rotateURL')
        window.parent.frames[1].document.location=newURL;
}
function addtofav(varName){
    LinkName=window.document.title;
    with (window.parent.table_window) {
        linkUrl=eval(varName);
    }
    window.external.AddFavorite(linkUrl,LinkName);
}
```

**_OUTPUT_SPREADSHEET_BUTTON_ Method**

Outputs the **Download to Spreadsheet** button as an image

**Syntax**

CALL SEND(OBJID,'_OUTPUT_SPREADSHEET_BUTTON_','vrflag.url, service.grphvar, grphstat.grphdown.grptype, bgtype, bg.title, webcls');

**Required Arguments**

- **vrflag**
  
  a flag indicating that the **View Report** button was pressed.
  
  **Type:** Numeric
url
    the Application Broker path.
    Type: Character

service
    the Application Broker service.
    Type: Character

graphvar
    the analysis variable to graph.
    Type: Character

graphstat
    the statistic to graph.
    Type: Character

graphdown
    the down dimension variable to graph.
    Type: Character

graphtype
    the graph type.
    Type: Character

background-type
    the background type (IMAGE or COLOR). This parameter is optional.
    Type: Character

background-value
    the background value. This parameter is optional.
    Type: Character

title
    the title. This parameter is optional.
    Type: Character

webcls
    the WEBEIS class name (for subclassing).
    Type: Character

Example
The following example code illustrates the use of this method:

vrflag=1;
url='/cgi-bin/broker';
service='default';
grphvar='ACTUAL';
grphstat='SUM';
grphdown='COUNTRY';
grphtype='VBAR';
btype='COLOR';
bg='YELLOW';
title=''
webcls=''
call send (webid,'_OUTPUT_SPREADSHEET_BUTTON_',vrflag,url,service,grphvar,
    grphstat,grphdown,grphtype,bgtype,bg,title,webcls);
The following output is produced:

```
<A HREF="/cgi-test-bin/broker/prdmxdb.csv?_service=default&_debug=0
&_program=sashelp.webeis.oprpt.scl&SPDSHT=X&mdb=SASHELP.PRDMDDB&metabase=
=SASHELP.MBEIS&D=GGeographic&AC=Product%20Line&A=ACTUAL&S=SUM&ST=1&GL=1
&DC=1&ACB=1&DP=1&_SAVEAS=prdmxdb.csv" TARGET="_self"><IMG CLASS="imgdown"
SRC="/my_images/btn_xls.gif"ALT="Download to Spreadsheet" BORDER=0"></A>
```

**_OUTPUT_SPREADSHEET_URL_ Method**

Outputs the URL for the Download to Spreadsheet button as a JavaScript text string on the Report page

**Syntax**

```
CALL SEND(OBJID,'_OUTPUT_SPREADSHEET_URL_','vrflag, url, service-name,
apalysis-variable, statistic, down-variable, graph-type, background-type, background-value, title,
webelis-class);
```

**Required Arguments**

- **vrflag**
  - the View Report button flag.
  - Type: Numeric

- **url**
  - the Application Broker component of the URL.
  - Type: Character

- **service-name**
  - the Application Broker service value.
  - Type: Character

- **analysis-variable**
  - the analysis variable to graph.
  - Type: Character

- **statistic**
  - the statistic to graph.
  - Type: Character

- **down-variable**
  - the down variable to graph.
  - Type: Character

- **graph-type**
  - the graph type (BLOCK, HBAR, PIE, PLOT, or VBAR).
  - Type: Character

- **background-type**
  - the background type (IMAGE or COLOR). This parameter is optional.
  - Type: Character

- **background-value**
  - the background value. This parameter is optional.
  - Type: Character
**Example**

The following example illustrates the use of this method:

```sas
vrflag=1;
url='/cgi-bin/broker';
service='default';
grphvar='ACTUAL';
grphstat='SUM';
grphdown='COUNTRY';
grptype='VBAR';
bgtpe='COLOR';
bq='yellow';
title='1995 Sales Report';
webcls='SASHELP.WEBEIS.WEBEIS';
call send(_self_,'_OUTPUT_STANDARD_GRAPH_ URL_',vrflag,url,service,grphvar,grphstat,
grphdown,grptype,bgtype,bg,title,webcls);
```

The following output is produced:

```
downloadURL="http://mywebserver/cgi-bin/broker/prdmddb.csv?_service=default&_debug=0
&_program=sashelp.webeis.cprt.scl&SPDSHT=&X=mddb=SASHELP.PRDMDDB&metabase=SASHELP
&D=COUNTRY&AC=YEAR&A=ACTUAL&A1S1=SUM&DC=1&ACB=1&ST=1&GL=1&GSC=1&SSL=1&SH=3
&SW=15&GH=450&GW=600&DP=1&NR=ALL&BS=Star&_SAVEAS=prdmddb.csv"
```

---

**_OUTPUT_STANDARD_GRAPH_ Method**

Outputs the URL that drives the standard GIF Graph request

**Syntax**

```sas
CALL SEND(OBJID,'_OUTPUT_STANDARD_GRAPH_','url,service,graph-type,

   analysis-variable,statistic-variable,down-variable,across-variable,webcls);
```

**Required Arguments**

- **url**
  - the URL for the next query.
  - **Type:** Character

- **service**
  - the Application Broker service.
  - **Type:** Character

- **graph-type**
  - the selected graph type.
**Type:** Character

**analysis-variable**
the analysis variable to graph.

**Type:** Character

**statistic-variable**
the statistic to graph.

**Type:** Character

**down-variable**
the down variable to graph.

**Type:** Character

**across-variable**
the analysis variable to graph.

**Type:** Character

**webcls**
the WEBEIS class name.

**Type:** Character

---

**Example**

The following example illustrates the use of this method:

```plaintext
url='/cgi-bin/broker';
service='default';
grptype='VBAR';
grphvar='ACTUAL';
grphstat='SUM';
grphdown='COUNTRY';
grphacr='PRODTYPE';
webcls=' ';
call send (webid,'_OUTPUT_STANDARD_GRAPH_',url,service,
grptype,grphvar,grphstat,grphdown,grphacr,
webcls);
```

The following output is produced:

```plaintext
<br><br><p><img class="graph" src="/cgi-bin/broker?program=sashelp.webeis.grf2way.scl&service=default&mddb=SASHELP.PRDMDDB&metabase=SASHELP.MBEIS&d=Geographic&ac=Product%20Line&sa=ACTUAL&S=SUM&g=VBAR&gv=Actual%20Sales&gs=Sum&gd=COUNTRY&dc=1&acb=1&gac=PRODTYPE&gsb=PRODTYPE=TOTAL&sl=%20" alt="Please wait." align=center width=600 height=450 bgcolor=silver></p>
```

---

**_OUTPUT_STAT_BOXES_ Method**

Outputs the Select Column and the Available and Selected list boxes for selecting statistics based on the analysis variable

**Syntax**

```plaintext
CALL SEND(OBJID,'_OUTPUT_STAT_BOXES_');
```
Example

The following output is produced:

```
<TH ROWSPAN=2 CLASS=laylabel>
Statistics</TH>
<TD CLASS=label>
Select Column
<DIV CLASS="stats">
<Select NAME="sa" CLASS="sselect" MULTIPLE SIZE="5"
ALIGN="left" onChange="change(document.mf.sa); updatestatslist(document.mf.sa); ">
<OPTION VALUE=ACTUAL>Actual Sales</OPTION>
</SELECT>
</DIV>
</TD>
<TD CLASS=label>
Available
<DIV CLASS="stats">
<Select NAME="as" CLASS="sselect" MULTIPLE SIZE="5"
ALIGN="left" onChanges="change(document.mf.as);"></SELECT>
</DIV>
</TD>
<TD ALIGN=CENTER CLASS=arrows>
<A href="/mddbapp.hlp/" onClick="moveall(document.mf.as,document.mf.s);
remstatanal(document.mf.as); return true"><IMG SRC="http://localhost/images/double_right_02g.gif" width="20" height="24" alt="Add all" BORDER=0><BR>
<A href="/mddbapp.hlp/" onClick="movesel(document.mf.as,document.mf.s);
return true"><IMG SRC="http://localhost/images/right_02g.gif" width="20" height="24" alt="Add selected" BORDER=0><BR>
<A href="/mddbapp.hlp/" onClick="movesel(document.mf.s,document.mf.as);
remstatanal(document.mf.s);"><IMG SRC="http://localhost/images/left_02g.gif" width="20" height="24" alt="Remove selected" BORDER=0><BR>
</Ahref="/mddbapp.hlp/" onClick="moveall(document.mf.s,document.mf.as);
remstatanal(document.mf.s);" />
</DIV>
</TD>
<TD CLASS=label>
Selected
<SELECT NAME="s" CLASS="sselect" MULTIPLE SIZE="5" align="left"
onChange="change(document.mf.s);">
</SELECT></DIV>
</TD>
```

_OUTPUT_STAT_LIST_ Method

Outputs a list of available statistics

Syntax

```
CALL SEND(OBJID,'_OUTPUT_STAT_LIST_');
```
Example

The following example illustrates the use of the method:

```
<TR><TD CLASS="label">Statistics
<DIV CLASS="stats">
<SELECT NAME="s" CLASS="select" MULTIPLE SIZE=3 onChange="change(document.mf.s)"
   <OPTION VALUE="SUM" SELECTED>Sum
   <OPTION VALUE="PCTSUM">% of Sum
   <OPTION VALUE="AVG">Average
   <OPTION VALUE="N">Total Count
   <OPTION VALUE="PCTN">% of Total #
   <OPTION VALUE="MIN">Minimum
   <OPTION VALUE="MAX">Maximum
   <OPTION VALUE="RANGE">Range
</SELECT>
</DIV>
</TD>
</TR>
```

_OUTPUT_STATIC_HIDDEN_FLDS_ Method

Outputs the necessary hidden fields for the initial static HTML page

Syntax

CALL SEND(OBJID,'_OUTPUT_STATIC_HIDDEN_FLDS_',metabase,background-type,
   background-value,webeis-class);

Required Arguments

metabase
  the metabase name.
  Type: Character

background-value
  the background image URL or color value. This parameter is optional.
  Type: Character

background-type
  the background type (COLOR or IMAGE). This parameter is optional.
  Type: Character

webeis-class
  the WEBEIS class name.
  Type: Character

_OUTPUT_SUBSET_DIMS_OPTION_ Method

Outputs text input fields for the width and height of the subset list box
Syntax
CALL SEND(OBJID,'_OUTPUT_SUBSET_DIMS_OPTION_');

Example
The following output is produced:

```html
<TR><TD CLASS="label">Width</TD><TD><INPUT TYPE=text NAME="sw" CLASS="select" SIZE=3 MAXLENGTH=3 VALUE="15"></TD></TR>
<TR><TD CLASS="label">Height</TD><TD><INPUT TYPE=text NAME="sh" CLASS="select" SIZE=3 MAXLENGTH=3 VALUE="3"></TD></TR>
```

.Output Subset Option Method
Outputs a selection list for the Location option in the Filter Listboxes list

Syntax
CALL SEND(OBJID,'_OUTPUT_SUBSET_LOC_OPTION_');

Example
The following output is produced:

```html
<TR><TD CLASS="label">Location</TD><TD><SELECT NAME="ssl" CLASS="select"><OPTION VALUE="1" SELECTED>Right
 <OPTION VALUE="2">Left
 <OPTION VALUE="3">Top
 <OPTION VALUE="4">Bottom
</SELECT></TD></TR>
```

.Output Subset Selections Method
Outputs the subset selection lists

Syntax
CALL SEND(OBJID,'_OUTPUT_SUBSET_SELECTIONS_','subloc');

Required Argument

subloc
the list box location.
Type: Character

Example
The following output is produced:
_OUTPUT_SUBSETS_ Method

Outputs the list of character variables for subsetting

Syntax

CALL SEND(OBJID,'_OUTPUT_SUBSETS_');

Example

The following output is produced:
<TR><TD CLASS="label" ALIGN=LEFT>Filter Columns:  <BR>
<SELECT NAME="SV" CLASS="select" MULTIPLE SIZE=3>
<OPTION VALUE="" SELECTED>
<OPTION VALUE="COUNTRY">Country
<OPTION VALUE="DIVISION">Division
<OPTION VALUE="MONTH">Month
<OPTION VALUE="PRODTYPE">Product type
<OPTION VALUE="PRODUCT">Product
<OPTION VALUE="QUARTER">Quarter
<OPTION VALUE="REGION">Region
<OPTION VALUE="YEAR">Year
</SELECT></TD></TR>

OUTPUTTABLE_DISP_OPTION_Method

Outputs radio buttons for the Display Table option

Syntax

CALL SEND(OBJID,'_OUTPUT_TABLE_DISP_OPTION_');

Example

The following output is produced:

<TD CLASS="label">Display Table</TD>
<TD>
<input NAME="ST" CLASS="select" TYPE=RADIO VALUE="1" CHECKED>Yes
<input NAME="ST" CLASS="select" TYPE=RADIO VALUE="2">No
</TD>
</TR>

OUTPUT_TABLE_OPTIONS_Method

Outputs the check boxes on the Options page for the Row Totals, Column Totals, and Drillpaths options

Syntax

CALL SEND(OBJID,'_OUTPUT_TABLE_OPTIONS_');

Example

The following output is produced:

<TD CLASS="label" COLSPAN="2">Row Totals<input TYPE="checkbox" NAME="dc" VALUE="1">Column Totals<input TYPE="checkbox" NAME="acb" VALUE="1">Drillpaths</TD>
_OUTPUT_TOOLBAR_ Method

Outputs the <FRAME> tag to create the frame in which the report is displayed

Syntax

CALL SEND(OBJID,'_OUTPUT_TOOLBAR_',vrflag,url, service,grphvar,grphstat, 
grphdown,grphtype,bgtype,bg, title,webcls, tbloc);

Required Arguments

vrflag
  a flag indicating that the View Report button was pressed.
  Type: Numeric
url
  the Application Broker component of the URL.
  Type: Character
service
  the Application Broker service.
  Type: Character
grphvar
  the analysis variable to graph.
  Type: Character
grphstat
  the statistic to graph.
  Type: Character
grphdown
  the down dimension variable to graph.
  Type: Character
grphtype
  the selected graph type.
  Type: Character
bgtype
  the background type (IMAGE, COLOR, or blank).
  Type: Character
bg
  the background value.
  Type: Character
title
  the title. This value is optional.
  Type: Character
webcls
  the WEBEIS class name.
  Type: Character
tbloc
the toolbar location, where 1=top, 2=bottom, 3=left, 4=right, and 5=none.

Type: Character

Example
The following output is produced:

```html
<tr>
  <td>
    <a href="/cgi-bin/broker/prdmddb.csv?_service=default&_debug=0&_program=sashelp.webeis.oprt.scl&SPDSHT=X&mddb=SASHELP.PRDMDDB&metabase=SASHELP.MBEIS&D=Geographic&AC=Product%20Line&A=ACTUAL&S=SUM&ST=1&DC=1&ACB=1&DP=1&_SAVEAS=prdmddb.csv" target="_self"><img src="/my_images/btn_xls.gif" alt="Download to Spreadsheet" border=0></a>
  </td>
  <td>
    <a href="../mddbapp.hlp/" onmouseover="this.href=clsurl('PROGRAM=SASHELP.WEBEIS.SHOWRPT.SCL')" target="_parent"><img src="/my_images/btn_rot.gif" alt="Rotate" border=0></a>
  </td>
  <td>
    <a href="../mddbapp.hlp/" onmouseover="this.href=clsurl('PROGRAM=SASHELP.WEBEIS.MDDBRPTS.SCL')" target="_parent">
      <img src="/my_images/btn_lay.gif" alt="Layout" border=0></a>
  </td>
  <td>
    <a href="http://support.sas.com/rnd/web/intrnet/mddbapp/hinttips.html" target="_blank"><img src="/my_images/btn_hlp.gif" alt="Help" border=0></a>
  </td>
</tr>
```

---

**_OUTPUT_TOOLBAR_FRAME_ Method**
Outputs the FRAME tag for the toolbar frame

---

**Syntax**

```plaintext
CALL SEND(OBJID,'_OUTPUT_TOOLBAR_FRAME_','url,service,bgtype,grptype,bg,grphvar,grphstat,grphdown,grphacr');
```

**Required Arguments**

**url**
the Application Broker component of the URL.

Type: Character

**service**
the Application Broker service.

Type: Character
**Example**

The following output is produced:

```
<FRAME NAME="toolbar_window" SRC="/cgi-bin/broker?_program=sashelp.webeis.optbar.scl
&_service=default&_debug=0&mddb=SASHELP.PRDMDDB
&metabase=SASHELP.MBEIS&D=Geographic&AC=Product%2520Line
&A=ACTUAL&S=SUM&GRT=VBAR
&GG=AC&BGTYPE=color&BG=%23FFFFE7&DC=1
&ACB=1&SL=1&GLC=2&SSL=1&SH=3
&SW=15&GH=450&GW=600&DP=1" SCROLLING="NO">
```

### _OUTPUT_TOTALS_OPTIONS_ Method

Outputs check boxes for the Show Totals option for the down and across variables

### Syntax

```
CALL SEND(OBJID,'_OUTPUT_TOTALS_OPTIONS_');
```

### Example

The following output is produced:
Show Totals

<INPUT TYPE="checkbox" NAME="dc" CLASS="select" VALUE="1" CHECKED>Down
<INPUT TYPE="checkbox" NAME="acb" CLASS="select" VALUE="1" CHECKED>Across</TD></TR>

_OUTPUT_UPDATE_CLEAR_ Method

Outputs the addstatanal and remstatanal JavaScript functions on the Dimensions page

Syntax

CALL SEND(OBJID,'_OUTPUT_UPDATE_CLEAR_');

Details

The addstatanal and remstatanal functions update the list of selected analysis variables as the user makes selections for the report.

Example

The following output is produced:

```javascript
function addstatanal(select,analysisbox) {
  select.length=0;
  for (i=0; i < analysisbox.length; i++){
    if (analysisbox.options[i].selected) {
      select.options[i] = new Option(analysisbox.options[i].text,
                                      analysisbox.options[i].value);
    }
  }
}

function remstatanal(listbox) {
  if ( listbox.options.length > 0 ){
    listbox.options.length=0;
  }
  return false;
}
```

OUTPUT_URL_OPTIONS_ Method

Outputs the viewer options, filter variables and selections, and expand information for a viewer URL

Syntax

CALL SEND(OBJID,'_OUTPUT_URL_OPTIONS_','noexp');
**Required Argument**

*noexp*

An instruction not to output or expand the information. A nonblank means do not output.

**Type:** Character

---

**_OUTPUT_VAR_FUNCTIONS_ Method**

Outputs JavaScript functions for ordering variable selections

---

**Syntax**

CALL SEND(OBJID,’_OUTPUT_VAR_FUNCTIONS ’);

---

**Example**

The following output is produced:

```javascript
function List(list) {
    for (key in list)
        if (list[key] != null) this[key] = list[key];
}

function change(select) {
    if (navigator.appName == "Netscape" &&
        navigator.appVersion.indexOf("3.0") != -1) ||
    (navigator.appName == "Microsoft Internet Explorer" &&
        navigator.appVersion.indexOf("4.0") != -1)) {
        selected = new List;
        options = new Object;
        for (i = 0; i < select.options.length; i++) {
            options[select.options[i].text] = select.options[i].value;
            selected[select.options[i].text] =
                select.options[i].selected ? select.options[i].value : null;
        }
        selected = new List(selected);
        select.options.length= 0;
        for (key in selected)
            select.options[select.options.length]=
                new Option(key, selected[key], false, true);
        for (key in options)
            if (selected[key] == null)
                select.options[select.options.length]=
                    new Option(key, options[key]);
    }
}

function update() {
    str = "";
    for (key in selected)
        str = str + key + ",";
    if (str.length)
        
}```
_OUTPUT_VARIABLE_SEL_FORM_ Method

Outputs the HTML table elements to arrange the Variable Selection page and calls the methods that output the variable and options HTML elements.

Syntax

CALL SEND(OBJID,"_OUTPUT_VARIABLE_SEL_FORM_","url",msgid,vrflag,grphtype);

Required Arguments

url
the Application Broker component of the URL.
   Type: Character

msgid
the ID number of the message system.
   Type: Numeric

vrflag
a flag indicating that the View Report button was pressed.
   Type: Numeric

grphtype
the selected graph type.
   Type: Character

Example

The following output is produced:

<FORM ACTION="/cgi-bin/broker" NAME="mf">
 <TR>
   <TD VALIGN=TOP>
     <TABLE>
     <TR>
       <TD CLASS=header>Dimensions</TD>
     </TR>
     <TR CLASS="dimselbox">
       <TD CLASS=label>Down: <BR></TD>
       <SELECT NAME="d" CLASS="select" SIZE=3 MULTIPLE onChange="change(document.mf.d)">
         <OPTION SELECTED VALUE=Geographic>Geographic (hier)
         <OPTION VALUE=Product%2520Line>Product Line (hier)
         <OPTION VALUE=Time>Time (hier)
         <OPTION VALUE=COUNTRY>Country
         <OPTION VALUE=DIVISION>Division
         <OPTION VALUE=MONTH>Month
         <OPTION VALUE=PRODTYPE>Product type
         <OPTION VALUE=PRODUCT>Product
     </SELECT>
     </TR>
   </TABLE>
 </TD>
</TR>
<OPTION VALUE="QUARTER">Quarter
<OPTION VALUE="REGION">Region
<OPTION VALUE="YEAR">Year
</SELECT>
</TD>
</TR>

<TR CLASS="dimselbox">
<TD CLASS="label">
Across:  <BR>
<SELECT NAME="ac" CLASS="select" SIZE=3 MULTIPLE onChange="change(document.mf.ac)"
<OPTION SELECTED VALUE="">
<OPTION VALUE="Geographic">Geographic (hier)
<OPTION VALUE="Product%20Line">Product Line (hier)
<OPTION VALUE="Time">Time (hier)
<OPTION VALUE="COUNTRY">Country
<OPTION VALUE="DIVISION">Division
<OPTION VALUE="MONTH">Month
<OPTION VALUE="PRODTYPE">Product type
<OPTION VALUE="PRODUCT">Product
<OPTION VALUE="QUARTER">Quarter
<OPTION VALUE="REGION">Region
<OPTION VALUE="YEAR">Year
</SELECT>
</TD>
</TR>

<TR><TD CLASS="label">Analysis
</DIV>
<SELECT NAME="a" CLASS="select" MULTIPLE SIZE=3 onChange="change(document.mf.a)"
<OPTION SELECTED VALUE="ACTUAL">Actual Sales
<OPTION VALUE="DIFF">Sales Lag
<OPTION VALUE="LPERDAY">LPERDAY
<OPTION VALUE="PREDICT">Predicted Sales
<OPTION VALUE="SALESRAT">Sales Ratio
</SELECT>
</DIV>
</TD>
</TR>

<TR><TD CLASS="label">Statistics
</DIV>
<SELECT NAME="s" CLASS="select" MULTIPLE SIZE=3 onChange="change(document.mf.s)"
<OPTION VALUE="SUM" SELECTED>Sum
<OPTION VALUE="PCTSUM">% of Sum
<OPTION VALUE="AVG">Average
<OPTION VALUE="N">Total Count
<OPTION VALUE="PCTN">% of Total #
<OPTION VALUE="MIN">Minimum
<OPTION VALUE="MAX">Maximum
<OPTION VALUE="RANGE">Range
</SELECT>
</DIV>
</TD>
</TR>

<TR><TD CLASS="label" ALIGN="LEFT">Filter Columns:  <BR>
<SELECT NAME="SV" CLASS="select" MULTIPLE SIZE=3>
<OPTION VALUE="">
<OPTION VALUE="COUNTRY">Country
</SELECT>
</DIV>
</TD>
</TR>
<OPTION VALUE="DIVISION">Division
<OPTION VALUE="MONTH">Month
<OPTION VALUE="PRODTYPE">Product type
<OPTION VALUE="PRODUCT">Product
<OPTION VALUE="QUARTER">Quarter
<OPTION VALUE="REGION">Region
<OPTION VALUE="YEAR">Year
</SELECT></TD></TR>
</TD>
<TD ALIGN=CENTER VALIGN=TOP>
<TABLE>
 TR>
<TD COLSPAN=2 CLASS=header>
Table</TD>
</TR>
<TR><TD CLASS="label">Display Table</TD>
<TD>
<INPUT NAME="ST" CLASS="select" TYPE=RADIO VALUE="1" CHECKED>Yes
<INPUT NAME="ST" CLASS="select" TYPE=RADIO VALUE="2">No
</TD>
</TR>
<TR>
<TD CLASS="label">Default Title</TD>
<TD><INPUT NAME="DT" CLASS="select" TYPE=TEXT SIZE=30 MAXLENGTH=200></TD>
</TR>
<TR>
<TD CLASS="label">Show Drillpath</TD>
<TD>
<INPUT NAME="DP" CLASS="select" TYPE=RADIO VALUE="1" CHECKED>Yes
<INPUT NAME="DP" CLASS="select" TYPE=RADIO VALUE="2">No
</TD>
</TR>
<TR><TD CLASS="label">Show Totals</TD>
<TD><INPUT TYPE="checkbox" NAME="dc" CLASS="select" VALUE="1" CHECKED>Down
<INPUT TYPE="checkbox" NAME="acb" CLASS="select" VALUE="1" CHECKED>Across</TD>
</TR>
<TR><TD COLSPAN=2 CLASS=header>
Graph</TD>
</TR>
<TR><TD CLASS="label">Graph Source</TD>
<TD>
<INPUT NAME="GSC" CLASS="select" TYPE=RADIO VALUE="1" CHECKED>3D Clickable Graph
<INPUT NAME="GSC" CLASS="select" TYPE=RADIO VALUE="2">Standard GIF Graph</TD>
</TR>
<TR><TD CLASS="label">Location</TD>
<TD><SELECT NAME="gl" CLASS="select">
<OPTION SELECTED VALUE=NONE>None
<OPTION VALUE=VBAR>Vertical bar
</SELECT></TD>
</TR>
<TR><TD CLASS="label">Type</TD>
<TD><SELECT NAME="grt" CLASS="select">
<OPTION VALUE=DIVISION">Division
<OPTION VALUE=MONTH">Month
<OPTION VALUE=PRODTYPE">Product type
<OPTION VALUE=PRODUCT">Product
<OPTION VALUE=QUARTER">Quarter
<OPTION VALUE=REGION">Region
<OPTION VALUE=YEAR">Year
</SELECT></TD>
</TR>
<TR><TD CLASS="label">Method</TD>
<TD>OUTPUT_VARIABLE_SEL_FORM_ Method 143</TD>
Chapter 4 • Making Advanced Customizations to the MDDB Report Viewer

<OPTION VALUE=BLOCK>Block
<OPTION VALUE=H BAR>Horizontal bar
<OPTION VALUE=PIE>Pie
<OPTION VALUE=PLOT>Plot
</SELECT>
</TD>
</TR>
<TR>
<TD CLASS="label">Width</TD><TD>
<input TYPE=text NAME="gw" CLASS="select"
SIZE=4 MAXLENGTH=4 VALUE="600"></TD></TR>
<TD CLASS="label">Height</TD><TD>
<input TYPE=text NAME="gh" CLASS="select"
SIZE=4 MAXLENGTH=4 VALUE="450"></TD></TR>
</TR>

Filter Listboxes</TD>
</TR>
<TD CLASS="label">Location</TD>
<TD>
<select NAME="ssl" CLASS="select"><option VALUE="1" SELECTED>Right
<option VALUE="2">Left
<option VALUE="3">Top
<option VALUE="4">Bottom
</select></TD></TR>
<TD CLASS="label">Width</TD><TD>
<input TYPE=text NAME="sw" CLASS="select"
SIZE=3 MAXLENGTH=3 VALUE="15"></TD></TR>
<TD CLASS="label">Height</TD><TD>
<input TYPE=text NAME="sh" CLASS="select"
SIZE=3 MAXLENGTH=3 VALUE="3"></TD></TR>
</TD>
</TR>
</TABLE>

</TD>
</TR>
</TD>
</TR>
</TABLE>

<TD ALIGN=CENTER COLSPAN=2>
<input TYPE="submit" NAME="view" CLASS="submit" VALUE="View Report">
</TD>
</TR>
</TD>
</TR>

_OUTPUT_VARLIST_FORM_ Method
Outputs the HTML for the reach-through to the detail variable selection page

Syntax

CALL SEND(OBJID,\_OUTPUT_VARLIST_FORM\_\_dataset-name\_url, htmlfile-id, message-id,
\_dataset-id\_service-name, debug-value, next-program, background-type, background-value);

Required Arguments

dataset-name
the base table data set name.
Type: Numeric

url
the Application Broker component of the URL.
Type: Numeric
htmlfile-id
  the ID for the _webout file.
  Type: Numeric

message-id
  the ID of the message system.
  Type: Numeric

dataset-id
  the ID for the base table data set.
  Type: Numeric

service-name
  the Application Broker service value.
  Type: Numeric

debug-value
  the application server debug level.
  Type: Numeric

next-program
  the next SCL program to execute when the form is completed.
  Type: Numeric

background-type
  the background type (IMAGE or COLOR). This parameter is optional.
  Type: Numeric

background-value
  the background value. This parameter is optional.
  Type: Numeric

Example

The following output is produced:

dataset='SASHELP.PRDSALE';
url='/cgi-bin/broker';
htmlfile=fopen('_WEBOUT','A');
msgid=instance(loadclass('sashelp.fsp.astmsg.class'),1);
dsdi=open(dataset);
service='default';
degug='0';
nexpgm='SASHELP.WEBEIS.DS2HTM.SCL';
btype='COLOR';
bg='yellow';
call send(webid,'_OUTPUT_VARLIST_',dataset,url,htmlfile,msgid,dsid,service,
  debug,nexpgm,btype,bg);

_OUTPUT_VARLIST_FUNCTIONS_ Method

Outputs the var_order, resetfields, and pickall JavaScript functions on the reach-through variable selection page
Syntax

CALL SEND(OBJID,'_OUTPUT_VARLIST_FUNCTIONS_',dataset-id,htmlfile-id);

Required Arguments

dataset-id
the base table data set identifier.
  Type: Numeric

htmlfile-id
the identifier for the _webout file.
  Type: Numeric

Example

htmlfile=fopen('_WEBOUT','A');
dsid=open('SASHELP.PRODSALE');
call send(webid,'_OUTPUT_VARLIST_FUNCTIONS_',dsid,htmlfile);

The following output is produced:

labels = new Array("placeholder","Actual Sales","Predicted Sales","Country","Region","Division","Product type","Product","Quarter","Year","Month");
varorder = new Array();
varlabel = new Array();
varorder.num = 0;
if (navigator.appName == 'Netscape') document.forms[0].reset();
function var_order(fieldnum,labeltext)
  { if (document.forms[0].elements[fieldnum].checked)
    { varorder[varorder.num] = document.forms[0].elements[fieldnum].value;
      varlabel[varorder.num] = labels[fieldnum];
      varorder.num++
    }
  else
  { for(i = 0; i < varorder.num; i++)
    { if (varorder[i] == document.forms[0].elements[fieldnum].value)
      for(j = i; j < varorder.num; j++)
      { varorder[j] = varorder[j+1];
        varlabel[j] = varlabel[j+1];
      }
      varorder.num--;
    }
  resetfields(labeltext);
}
function resetfields(labeltext)
{
    document.forms[0].elements[labeltext].value = ' ';  
    document.forms[0].elements[0].value = ' ';  
    if (varorder.num > 0)
    {
        document.forms[0].elements[labeltext].value = varlabel[0];  
        document.forms[0].elements[0].value = varorder[0];  
    }
    for(i = 1; i < varorder.num; i++)
    {
        document.forms[0].elements[labeltext].value =  
        document.forms[0].elements[labeltext].value + '\r\n' + varlabel[i];  
        document.forms[0].elements[0].value =  
        document.forms[0].elements[0].value + ' ' + varorder[i];  
    }
}

function pickall(num)
{
    for (i = 1; i <= num ; i++)
    {
        if (document.forms[0].elements[i].checked == false)
        {
            varlabel[varorder.num] = labels[i];  
            varorder[varorder.num] = document.forms[0].elements[i].value;  
            document.forms[0].elements[i].checked = true;  
            varorder.num++;  
        }
    }
    resetfields(num+1);
}

_OUTPUT_VARLIST_HTML_ Method

Outputs the HTML for the reach-through to the detail variable selection page

Syntax
CALL SEND(OBJID,'_OUTPUT_VARLIST_HTML_','dataset-id,htmlfile-id,message-id,dataset-name,url,service-name,debug-value,next-program,background-type,background-value);

Required Arguments

dataset-id
the ID for the base table data set.  
Type: Numeric

htmlfile-id
the ID for the _webout file.  
Type: Numeric

message-id
the ID of the message system.  
Type: Numeric

dataset-name
the base table data set name.  
Type: Numeric
url
  the Application Broker component of the URL.
  Type: Numeric

service-name
  the Application Broker service value.
  Type: Numeric

default-value
  the application server debug level.
  Type: Numeric

next-program
  the next SCL program to execute when the form is completed.
  Type: Numeric

background-type
  the background type (IMAGE or COLOR). This parameter is optional.
  Type: Numeric

background-value
  the background value. This parameter is optional.
  Type: Numeric

Example

dataset='SASHELP.PRDSALE';
dsid=opendataset(dataset);
htmlfile=fopen('_WEBOUT','A');
msgid=instance(loadclass('sashelp.fsp.astmsg.class'),1);
url='/cgi-bin/broker';
service='default';
debug='0';
nextpgm='SASHELP.WEBEIS.DS2HTM.SCL';
btype='COLOR';
bg='yellow';
call send(webid,'_OUTPUT_VARLIST_HTML_',dsid,htmlfile,msgid,dataset,url,service,
  debug,nextpgm,btype,bg);

_OUTPUT_VIEWRPT_BUTTON_ Method

Outputs the View Report button

Syntax

CALL SEND(OBJID,'_OUTPUT_VIEWRPT_BUTTON_');

Example

The following output is produced:

<Input TYPE="submit" NAME="view" CLASS="submit" VALUE="View Report">
_OUTPUT_VIEWRPT2_BUTTON_ Method
Outputs the View Report button on the Dimensions page

Syntax
CALL SEND(OBJID,'_OUTPUT_VIEWRPT2_BUTTON_');

Example
The following output is produced:

```html
<A href="../mddbapp.hlp/* onClick="this.href=geturl
(document.mf.d,document.mf.ac,document.mf.a)"TARGET="_parent">
<IMG SRC="view-report.gif" width="29" height="24"></A>
```

_POST_DISPLAY_OPTIONS_ Method
Specifies additional options on the Layout page

Syntax
CALL SEND(OBJID,'_POST_DISPLAY_OPTIONS_',<parmlist>);

Optional Argument

parmlist
an optional list for passing in information to the method.
Type: Numeric

Details
This stub method is called after all of the display options are called. It is useful for adding additional options to the Layout page.

_PRE_DISPLAY_OPTIONS_ Method
Specifies additional options on the Layout page

Syntax
CALL SEND(OBJID,'_PRE_DISPLAY_OPTIONS_',<parmlist>);

Optional Argument

parmlist
an optional list for passing in information to the method.
This stub method is called before any of the display options are called. It is useful for adding additional options to the Layout page.

_TRAN_A_BLANK_ Method
Prints the character code to fill an empty cell

Syntax
CALL SEND(OBJID,'_PRINT_A_BLANK_');

_SET_ACROSS_TOTAL_FLAG_ Method
Sets the atotal_ instance variable to activate across totals

Syntax
CALL SEND(OBJID,'_SET_ACROSS_TOTAL_FLAG_', ttlflag);

Required Argument

ttlflag
a value that indicates whether to set a flag for the totals in the across dimension, where X=set the flag on and blank=do not set the flag.

Type: Character

_SET_DOWN_TOTAL_FLAG_ Method
Sets the dtotal_ instance variable to activate down totals

Syntax
CALL SEND(OBJID,'_SET_DOWN_TOTAL_FLAG_', ttlflag);

Required Argument

ttlflag
a value that indicates whether to set a flag for the totals in the down dimension, where X=set the flag on and blank=do not set the flag.

Type: Character
_SET_DRILL_LEVELS_ Method

Updates the SAVED_L sublist on the application list to set the drill-down values

Syntax

CALL SEND(OBJID,'_SET_DRILL_LEVELS_','application-list');

Required Argument

*application-list*

the list ID of the application list. For more information about application lists, see the online Help for SAS/EIS software.

Type: Numeric

Details

This method

- builds the HIERARCHIES_L and SAVED_L sublists on the application list if the list is empty
- builds the CURRENT_DRILLS sublist on the HIERARCHIES_L sublist if it is empty
- updates the CURRENT_DRILLS sublist for each hierarchy with the current drill-down information
- sets the CURRENT_LEVEL value for each hierarchy on the HIERARCHIES_L sublist.

Example

applist= makelist();
rc=fillist('CATALOG','SASHELP.EISRG.ONEWAY.EIS',applist);
call send(webid,'_SET_DRILL_LEVELS_','applist');

_SET_EMDBBMID_ Method

Sets the EMDBBMID_ instance variable

Syntax

CALL SEND(OBJID,'_SET_EMDBBMID_','id');

Required Argument

*id*

the ID of the data model.

Type: Numeric
_SET_EXPAND_FLAG_ Method

Sets the expflag_ instance variable that indicates whether values can be expanded

Syntax

CALL SEND(OBJID,'_SET_EXPAND_FLAG_','rowlist,actionsl');

Required Arguments

rowlist
the rowlist from the GET_CLASS_COMBINATIONS method.
Type: Numeric

actionsl
the actionsl list from the data model.
Type: Numeric

_SET_HIERL_LIST_ Method

Sets the hierl_ instance variable

Syntax

CALL SEND(OBJID,'_SET_HIERL_LIST_','listid');

Required Argument

listid
the list ID of the target list to copy.
Type: Numeric

_SET_SUBSET_BY_LIST_ Method

Builds the subset_by_ list from the filter value selections

Syntax

CALL SEND(OBJID,'_SET_SUBSET_BY_LIST_');

Example

The following illustrates an example of a subset_by_ list:

```
subset_by_ ( COUNTRY = {'CANADA'}
```
DIVISION = {'EDUCATION'
  
MONTH = {'Jan'
    'Feb'
  }

---

**_SET_SUBSET_FLAG_ Method**

Sets the value of the SUBSET_FLAG_instance variable

---

**Syntax**

CALL SEND(OBJID,'_SET_SUBSET_FLAG_'.flagval);

**Required Argument**

*flagval*

the value of the subset flag.

**Type:** Character

---

**_SET_SUBSETS_LIST_ Method**

Defines the subsets to be used

---

**Syntax**

CALL SEND(OBJID,'_SET_SUBSETS_LIST_'.varnum);

**Required Argument**

*varnum*

the number of selected subset values.

**Type:** Numeric

**Details**

This method sets and fills the subvars_instance variable and adds the subvars_list to the _self_list data model for applying the filters.

---

**_SHOW_GRAPH_ Method**

Sets the graphing variables and calls a graphing method
Syntax

CALL SEND(OBJID,'_DISPLAY_GRAPH_','url','service','_argument-string','_argument-string2','graph-type','analysis-variable','statistic-variable','down-variable','across-variable','webcls');

Required Arguments

url
the Application Broker component of the URL.
Type: Character

service
the Application Broker service.
Type: Character

_argument-string
the argument string for the next query.
Type: Character

_argument-string2
the argument string for the next query.
Type: Character

graph-type
the selected graph type.
Type: Character

analysis-variable
the analysis variable to graph.
Type: Character

statistic-variable
the statistic to graph.
Type: Character

down-variable
the down variable to graph.
Type: Character

across-variable
the across variable to graph.
Type: Numeric

webcls
the WEBEIS class name.
Type: Character

Details

This method sets the default graphing variables if their values have not been specified and calls the appropriate graphing method (_OUTPUT_STANDARD_GRAPH_ or _OUTPUTCLICKABLE_GRAPH_) for the selected graph source.

.SUBMIT_GOPTIONS_Method

Submits the SAS/GRAPH GOPTIONS statement for the standard GIF graph
Syntax
CALL SEND(OBJID,'_SUBMIT_GOPTIONS_','gifdev');

Required Argument

gifdev
the name of the device driver to use.

Type: Character

_SUBMIT_GRAPH_PATTERN_ Method
Submits the SAS/GRAPH PATTERN statements for the standard GIF graphs

Syntax
CALL SEND(OBJID,'_SUBMIT_GRAPH_PATTERN_');

_SUBMIT_GRAPH_TITLE_ Method
Submits the SAS/GRAPH TITLE statement for the standard GIF graph

Syntax
CALL SEND(OBJID,'_SUBMIT_GRAPH_TITLE_','stat',var);

Required Arguments

stat
the statistic used in the graph.

Type: Character

var
the analysis variable used in the graph.

Type: Character

_UPDATE_STATS_LIST_ Method
Outputs the updatestatslist JavaScript function on the Dimensions page

Syntax
CALL SEND(OBJID,'_UPDATE_STATS_LIST_');

Details
The updatestatslist function modifies the list of available and selected statistics as the user makes statistic selections for the report display.
Example

The following output is produced:

```javascript
function updatestatslist(select) {
    pos = 0;
    num = 0;
    newlength = 0;
    var arrayname = "";
    var analysistype = "";
    var arrayofstats = "";
    for (i=0; i < select.options.length; i++) {
        if (select.options[i].selected) {
            num=num+1;
            arrayname = select.options[i].value+"STATS";
            analysisarray=eval(arrayname);
            if (analysistype.indexOf(analysisarray[0])==-1 ) {
                analysistype=analysisarray[0] +"," +analysistype;
            }
        }
    }
    if (analysistype.substr(eval(analysistype.lastIndexOf","+1), 1)=="" ) {
        analysistype=analysistype.slice(0,analysistype.lastIndexOf",");
    }
    arrayoftypes = analysistype.split",";
    arrayoftypes.sort();
    document.mf.as.options.length=0;
    document.mf.s.options.length=0;
    if (num > 1) {
        for (i=0; i < arrayoftypes.length; i++) {
            if ( i==0 ) {
                arrayname = eval(arrayoftypes[0] +"desclist");
                pos = arrayname.length;
                for ( j=0; j < arrayname.length; j++ ) {
                    document.mf.as.options[j] = new Option(statslabellist[arrayname[j]],
                        arrayname[j]);
                }
            } else if (arrayoftypes[i]=="nunique") {
                arrayname = eval( arrayoftypes[i] +"desclist");
                document.mf.as.options[pos] = new Option(statslabellist[arrayname[0]],
                        arrayname[0]);
            }
        }
        document.mf.s.options[0] = new Option("*MIXED SELECTIONS", "MIXED");
    } else if ( num==1 ) {
        k=0;
        arrayofstats=eval( arrayoftypes[0] +"desclist");
        for (i=0; i < select.options.length; i++) {
            if (select.options[i].selected) {
                arrayname = eval(select.options[i].value +"STATS");
                for ( j=1; j < arrayname.length; j++ ) {
                    document.mf.s.options[j-1] = new Option(statslabellist[arrayname[j]],
                        arrayname[j]);
                }
            }
        }
    }
}
```

Chapter 4 • Making Advanced Customizations to the MDDB Report Viewer
for (i=0; i < arrayofstats.length; i++) {
    var repeat="false";
    for (j=1; j < arrayname.length; j++) {
        if (arrayofstats[i]==arrayname[j]) {
            repeat="true";
            break;
        }
    }
    if (repeat=="false" && arrayofstats[i]!="") {
        document.mf.as.options[k] = new Option(statslabellist[arrayofstats[i]],
            arrayofstats[i]);
        k++;
    }
}
Index

Special Characters

\_BUILD\_ACROSS\_LIST\_ method 38
\_BUILD\_ANALYSIS\_LIST\_ method 39
\_BUILD\_ANLSORTORDER\_ method 39
\_BUILD\_APPLICATION\_LIST\_ method 40
\_BUILD\_CURRENT\_SUBSETS\_ method 41
\_BUILD\_DOWNL\_LIST\_ method 42
\_BUILD\_STATSL\_LIST\_ method 43
\_BUILD\_TOTAL\_ method 43
\_BUILD\_URL\_ONSUBMIT\_ method 44
\_BUILD\_WHERE\_FORMAT\_STRING\_ method 46
\_CHECK\_HIER\_MEMBER\_ method 47
\_CLOSE\_FORM\_ method 47
\_CLOSE\_PAGE\_ method 49
\_CLOSE\_STATIC\_FORM\_ method 49
\_CREATE\_STAT\_ARRAYS\_ method 49
\_DISPLAY\_ACROSS\_CELLS\_ method 53
\_DISPLAY\_ANALYSIS\_VARS\_ method 56
\_DISPLAY\_DEFAULT\_TITLE\_ method 57
\_DISPLAY\_DOWNVAR\_CELL\_ method 57
\_DISPLAY\_ERROR\_ method 59
\_DISPLAY\_ONEWAY\_method 60
\_DISPLAY\_ONEWAY\_BLOCK\_ method 61
\_DISPLAY\_ONEWAY\_HBAR\_ method 62
\_DISPLAY\_ONEWAY\_PIE\_ method 62
\_DISPLAY\_ONEWAY\_VBAR\_ method 63
\_DISPLAY\_STATISTICS\_VARS\_ method 63
\_DISPLAY\_SUBSET\_TITLE\_ method 65
\_DISPLAY\_TITLE\_ method 66
\_DISPLAY\_TWOWAY\_method 67
\_DISPLAY\_TWOWAY\_BLOCK\_ method 68
\_DISPLAY\_TWOWAY\_HBAR\_ method 69
\_DISPLAY\_TWOWAY\_VBAR\_ method 69
\_DISPLAY\_VALUES\_ method 70
\_DRILL\_TO\_LEVEL\_ method 74
\_GET\_ANALYSIS\_VAR\_NAME\_ method 74
\_GET\_ANALYSIS\_VARS\_ method 75
\_GET\_AVAILABLE\_STATS\_ method 75
\_GET\_DATA\_MODEL\_NAME\_ method 75
\_GET\_DOWNVAR\_LIST\_ method 76
\_GET\_EMDDBMID\_ method 76
\_GET\_GRAPH\_VALUES\_ method 77
\_GET\_MDDB\_NAME\_ method 79
\_GET\_MESSAGE\_ID\_ method 79
\_GET\_METABASE\_NAME\_ method 79
\_GET\_OUTPUT\_FILE\_ID\_ method 80
\_GET\_RANGE\_COLOR\_ method 80
\_GET\_STATDESC\_ method 81
\_GET\_SUBSET\_FLAG\_ method 81
\_GET\_USEHOLAP\_ method 81
\_GRFONT\_ macro\ variable 12, 34
\_MRBODYONLY\_ macro\ variable 35
\_MRNODIMBOXES\_macro\ variable 35
\_MRNOFRAMES\_ macro variable 14, 35
\_MRNOSORT\_ macro variable 35
\_MRNOVARCHECK\_ macro variable 35
\_MRNOFRAMESET\_ macro variable 35
\_MRTBLOC\_ macro variable 13, 34
\_MRTBLPRM\_ macro variable 14, 35
\_MRVFRAMESET\_ macro variable 35
\_MRVHELP\_ macro variable 13, 34
\_MRVNOPGOP\_ macro variable 35
_MRVNRLKS macro variable 14, 35
_MRVRNDX1 macro variable 14, 35
_MRVRNDX2 macro variable 14, 35
_MRVRNDX3 macro variable 14, 35
_MRVRNDX4 macro variable 14, 35
_MRVSEP macro variable 11, 13, 34
_MRVTBSC macro variable 34
_MRVTBSSZ macro variable 34
_OPEN_DYNAMIC_FILE_method 81
_OPEN_FORM_method 82
_OPEN_ONEWAY_method 82
_OPEN_STATIC_FILE_method 83
_OPEN_TABLE_method 83
_OPEN_TWOWAY_method 84
_OPEN_WEBOUT.For_SPDSHT_method 86
_OPEN_ACROSS_LIST_method 86
_OPEN_ADDTL_CLSVAL_PARMS_method 87
_OPEN_ADDTL_RT_PARMS_method 87
_OPEN_ADDTOFAv_FUNCTION_method 87
_OPEN_ALL_URL_ITEMS_method 88
_OPEN_ANAL_LIST_method 88
_OPEN_ANAL_SELECT_method 89
_OPEN_ARROW_FUNCTIONS_method 90
_OPEN_BAR_SHAPE_LIST_method 91
_OPEN_BOOKMARK_BUTTON_method 92
_OPEN_BOOKMARK_URL_method 92
_OPEN_CLASSVAL_URL_FN_method 93
_OPEN_CLICKABLE_GRAPH_method 95
_OPEN_CONTENT_HEADER_method 96
_OPEN_CSV_CONTENT_HEADER_method 96
_OPEN_DEBUG_LIST_method 96
_OPEN_DEFtLT TITLE_OPTION_method 97
_OPEN_DIMBTN_URL_FN_method 97
_OPEN_DIMENSIONS_BUTTON_method 98
_OPEN_DOWN_LIST_method 98
_OPEN_DP TITLE_OPTION_method 99
_OUTPUT_DS2HTM.HTML_OPTION_method 100
_OUTPUT_DS2HTM.ST_OPTION_method 101
_OUTPUT_DYNAMIC_HIDDEN_FLDs_method 102
_OUTPUT_EMPTY_CELL_method 103
_OUTPUT_EMPTY_SERVICE_LIST_method 104
_OUTPUT_GRAPH_INSTR_method 104
_OUTPUT_GRAPH_LIST_method 104
_OUTPUT_GRAPH_LOC_OPTION_method 105
_OUTPUT_GRAPH_OPTION_method 106
_OUTPUT_GRAPH_SOURCE_OPTION_method 106
_OUTPUT_GRAPH_TABLE_DISP_method 106
_OUTPUT_HDR_method 107
_OUTPUT_HELP_BUTTON_method 108
_OUTPUT_HIDDEN_Fields_method 109
_OUTPUT_HIDDEN_VARS_method 110
_OUTPUT_HTML.AFTER_BODY_method 110
_OUTPUT_HTML.BEF.CLOSE_BODY method 110
_OUTPUT_HTML_Form_HEADER BODY_method 111
_OUTPUT_LAYOUT_BUTTON_method 111
_OUTPUT_LAYOUT_FRAME_method 112
_OUTPUT_LAYOUT_TOOLBAR_method 113
_OUTPUT_LOGOUT_BUTTON_method 113
_OUTPUT_MAIN_TOOLBAR_FRAME_method 114
_OUTPUT_MDDB_LIST_method 115
_OUTPUT_NUMROWS_LINKS_method 115
_OUTPUT_NUMROWS_OPTION_method 116
_OUTPUT_OLPBTN.URL_FN_method 116
_OUTPUT_OPTIONS_BUTTON_method 118
_OUTPUT_OPTIONS_FORM_method 118
_OUTPUT_REACTHRU_LINK_method 119
_OUTPUT_REACTHRU_URL_FN_method 120
displaying reports without list boxes 14
ServiceSet directive 14
Application Dispatcher
defining repository to 7
MDDB Report Viewer requirements 2
specifying repository manager 13
APPSSRV procedure 7, 13, 30
ATOTAL_ instance variable 18

B
BG macro variable 34
BG option, AF command 5, 6
BGTYPE macro variable 34
BGTYPE option, AF command 5, 6
BS macro variable 33

C
cascading style sheets (CSS)
changing report appearance 13
MDDB Report Viewer properties 36
CGI option, AF command 4, 6
CLASS_ instance variable 18
CLASS macro variable 34
CLASS option, AF command 5, 6
CLASS parameter, HTML elements 36
COLLAB CSS class tag 36
colors, changing for reports 11
Common Metadata Repository 6
control flow in WEBEIS class 21
CSS (cascading style sheets)
changing report appearance 13
MDDB Report Viewer properties 36
CSS macro variable 34
CSST macro variable 34
CSSTURL_ instance variable 19
CSSURL_ instance variable 19

D
D macro variable 31
DC macro variable 33
DEBUG_ instance variable 19
DEFTITLE_ instance variable 19
delimiters, specifying 11, 13
DIMBOX CSS class tag 37
Dimensions page
about 9
accessing 10
Analysis section 9
changing report dimensions 11
Columns section 9
Statistics section 9
DIMLRLS_ instance variable 19
DIMSELBOX CSS class tag 37
DLEVELS_ instance variable 19
DLSEP_ instance variable 19
DMODEL_ instance variable 19
DOWNDRL_ instance variable 19
DOWNL_ instance variable 19
Download to spreadsheet button 11, 13
DOWNVARS_ instance variable 19
DP macro variable 33
DPTITLE_ instance variable 19
dT macro variable 33
dTOTAL_ instance variable 19

E
EMDDBMID_ instance variable 19
EMPTY CSS class tag 36
EX macro variable 32
EXPFLAG_ instance variable 19
EXPLIST_ instance variable 19
EXPVALS_ instance variable 19
EXPVAR_ instance variable 19
FILTERBOX CSS class tag 36
font, changing for GIF graphs 12

G
GA macro variable 33
GD macro variable 33
GG macro variable 33
GH macro variable 33
GIF graphs
changing font for 12
generating 12
GL macro variable 33
global variables 34
See also macro variables
GOPTIONS statement 5
GRAPH CSS class tag 37
GRAPHSRC CSS class tag 37
graphs, creating 12
GRFHT_ instance variable 19
GRFSRC_ instance variable 19
GRFWID_ instance variable 19
GRLOC_ instance variable 20
GRPHTYPE_ instance variable 20
GRPHVALS_ instance variable 20
GRT macro variable 33
GSC macro variable 33
GSG macro variable 34
GW macro variable 33
H
HEADER CSS class tag  37
Help button  13
Help page  13
HIERL_ instance variable  20
HMODEL_ instance variable  20
HTML elements  36
HTML frames  14
HTMLFILE_ instance variable  20

I
IMGDIM CSS class tag  37
IMGHELP CSS class tag  37
IMGLAY CSS class tag  37
IMGLOGOUT CSS class tag  37
IMGOPT CSS class tag  37
IMGROTATE CSS class tag  37
IMGURL_ instance variable  20
instance variables  18

L
LABEL CSS class tag  37
list boxes  14

M
macro variables  30, 31
MAINTAB CSS class tag  36
MDDB_ instance variable  20
MDDB (multidimensional database)  1, 2
MDDB macro variable  31
MDDB procedure  2
MDDB Report Viewer
about  1, 9
Access Control feature support  1, 3
changing appearance of reports  11
changing settings  13
creating graphs  12
CSS properties  36
instance variables  18
macro variables  30, 34
printing reports  11
requirements for running  2
Rotate button  11
setting up  3, 4
system repository manager location  6
viewing data  12
working with repositories  6
MDDB Report Viewer class
See WEBEIS class
METABASE_ instance variable  20
METABASE macro variable  31
METABASE option, AF command  4, 6
metadata management  6
MGBKMRK CSS class tag  37
multidimensional database (MDDB)  1, 2

N
NR macro variable  31

O
Optional Settings page
about  10
accessing  9, 10
Filter Columns list box  10
Filter Listbox Options section  10
generating 3-D graphs  12
generating standard GIF graphs  12
Graph section  10, 12
Report section  10
subsetting report data  12

P
page links, changing number displayed  14
PATHNAME option, AF command  4, 6
printing
large tables  11
reports  11

R
RANGE entry, SAS/EIS metabase  11
Registry Editor Options window  7
report data
generating 3-D graphs of  12
generating standard GIF graphs of  12
subsetting  12
viewing detailed  12
Report Layout page  9
Report page
about  10
changing report dimensions  11
Download to spreadsheet button  11, 13
Filter By list box  10
reports
changing appearance of  11, 13
changing appearance of tables  14
changing colors on  11
displaying without list boxes  14
printing  11
repositories
defining to Application Dispatcher  7
setting up SASHELP  8
system repository manager  6, 7
working with  6
Repository Manager  3
REQUEST INIT program
  _GRFONT macro variable 12
  _MRTBLOC macro variable 13
  _MRVHELP macro variable 13
  _MRVSEP macro variable 13
  macro variables 30

Rotate button 11
ROTFLAG_ instance variable 20
row paging feature, disabling 14
ROWLAB CSS class tag 36
rows to display, changing 14

S
  S macro variable 32
  SAC macro variable 34
  SAS OLAP Server 1, 2
  SAS/EIS software
    Access Control features 1, 3
    changing report colors 11
    Common Metadata Repository 6
    MDDB Report Writer requirements 2
  SAS/GRAPH software 2, 5
  SAS/IntrNet software 2, 7
  SASHELP repository 8
  SD macro variable 34
  SELECT CSS class tag 37
  selection lists
    determining validity of items 10
    selecting items from 10
  ServiceSet directive
    _MRNORFRAMES macro variable 14
    _MRTBLPRM macro variable 14
    _MRVNLKLS macro variable 14
    _MRVRNDX1 macro variable 14
    _MRVRNDX2 macro variable 14
    _MRVRNDX3 macro variable 14
    _MRVRNDX4 macro variable 14
  SESSIONID_ instance variable 20
  SH macro variable 33
  SHOWTAB_ instance variable 20
  SL macro variable 32
  sorting feature, disabling 14
  SPSHT macro variable 33
  SR macro variable 31
  SSELECT CSS class tag 37
  SSL macro variable 33
  ST macro variable 33
  STATDESC_ instance variable 20
  STATLIST_ instance variable 20
  STATS CSS class tag 37
  STATSBOX CSS class tag 37
  STATSCOL CSS class tag 37
  STATVARS_ instance variable 20
  STCOLLAB CSS class tag 36
  STROWLAB CSS class tag 36
  SUBHT_ instance variable 20
  SUBLOC_ instance variable 20
  SUBMIT CSS class tag 37
  SUBSET_BY_ instance variable 20
  SUBSET_FLAG_ instance variable 20
  subsetting report data 12
  SUBVARS_ instance variable 21
  SUBWID_ instance variable 21
  SV macro variable 32
  SW macro variable 33
  system repository manager
    setting up files 7, 13
    specifying location 6

T
  tables
    changing appearance of 14
    printing large 11
  TBLOC_ instance variable 21
  TCOLCELL CSS class tag 36
  TCOLLAB CSS class tag 36
  TDCELL CSS class tag 36
  THISSESSION_ instance variable 21
  TITLE option, AF command 5, 6
  toolbar, changing location 13
  TOOLTAB CSS class tag 37
  TRANSPARENCY option, GOPTIONS statement 5
  TROWCELL CSS class tag 36
  TROWLAB CSS class tag 36

U
  USEHOLAP_ instance variable 21

V
  V macro variable 32
  VA macro variable 33
  variables
    instance 18
    macro 30, 31, 34
  VIEW macro variable 33
  viewing data 12
  VMDOFF_ instance variable 21
  VMDOFF macro variable 34

W
  Web browsers
    access considerations 3
    setting up MDDB Report Viewer 4
  WEBEIS class
    CLASS subclass 3, 5
    flow of control in 21
methods for 38  webeis.html page 3