ODS Report Writing Interface Tip Sheet

Table Example

```sas
data _null_; set sashelp.class(obs=5) end=done;
/* do header only at beginning */
if N = 1 then do;
/* Create odsout object */
declare odsout o();
/* start table */
o.table_start();
o.head_start();
o.row_start();
o.format_cell(data: "Name, Gender, Age & BMI", column_span: 4);
o.format_cell(data: age);
o.format_cell(data: "Male");
o.format_cell(data: "Female");
o.row_end();
o.head_end();
o.body_start();
if sex = "F" then
  o.format_cell(data: name);
else
  o.format_cell(data: "Male");
o.format_cell(data: age);
o.format_cell(data: bmi = (weight/(height*height))*703);
o.row_end();
if done then do;
o.body_end();
o.table_end();
end;
run;
```

Layout Example

```sas
data _null_; set sashelp.cars(obs=80) end=done;
by make;
if _N_ = 1 then do;
/* start absolute layout */
o.layout_absolute(style_attr: "background=lightgray");
  o.region(x: "0.5in", style_attr: "background=gray");
o.format_text(data: Model);
o.format_text(data: make);
o.format_text(data: "Region 2", just: "c", style_attr: "background=blue", color="white");
o.layout_end();
o.page();
/* start gridded layout */
o.layout_gridded(columns: 3);
if first.make then do;
o.region(style_attr: "background=lighgray");
o.format_text(data: "System Title");
o.format_text(data: model);
end;
if done then do;
o.layout_end();
end;
run;
```

This tip sheet places frequently used information in one place, on one sheet of paper, so you don’t have to search through the online documentation. It also gives you something to take home, type in, and try.

The ODS Report Writing Interface allows you to create highly customized reports from within the DATA step. As it is fully integrated with the ODS system, it lets you combine the powerful programming features from the DATA step with ODS features giving you flexibility and control for every piece of output in your report. The ODS Report Writing Interface is currently supported for the HTML and PDF output destinations.
Common Arguments

These arguments are common to all methods.

style_elem : ‘style-element-name’
specifies the style element to apply. The value can also be a character variable.

style_attr : ‘style-attribute(s)’
specifies style overrides to apply. The value can also be a character variable.

Table Methods

Methods that come in _start / _end pairs, must both be invoked for the item to be well formed.

obj.table_start( arguments )
start or end a table, respectively.

obj.head_start( arguments )
start or end a table header section, respectively.

obj.body_start( arguments )
start or end a table body section, respectively.

obj.row_start( arguments )
start or end a row, respectively.

type : ‘H’ | ‘B’ | ‘F’
specifies the type (header, body or footer) of the table row. Default is body. This is used as an alternative to the head, body, and foot methods.

Table Methods (continued)

obj.format_cell( arguments )
or

obj.cell_start( arguments )

obj.cell_end( )
create a table cell.

data : ‘string’ | number | variable-name
specifies the value to display in the table cell. This is a required argument for this method.

format : ‘sas-format-name’
specifies the format used to display the cell value.

just : ‘L’ | ‘C’ | ‘R’
specifies the horizontal alignment of the cell.

vjust : ‘T’ | ‘M’ | ‘B’
specifies the vertical alignment of the cell.

specifies the cell borders that should not be displayed. (Printer destinations only)

height : ‘dimension’
specifies the vertical height of the cell.

width : ‘dimension’
specifies the horizontal width of the cell.

column_span : integer
specifies the number of columns that the cell will occupy. The default value is 1.

row_span : integer
specifies the number of rows that the cell will occupy. The default value is 1.

inline_attr : ‘style-attribute(s)’
overrides specific style attributes for the preceding data argument only.

inline_elem : ‘style-element-name’
changes the style element for the preceding data argument only.

Text Methods

obj.format_text( arguments )

obj.note( arguments )
create text output or note, respectively.

data : ‘string’ | number | variable-name
specifies the value to display. This is a required argument for this method.

format : ‘sas-format-name’
specifies the format used to display the text.

just : ‘L’ | ‘C’ | ‘R’
specifies the horizontal alignment of the text.

vjust : ‘T’ | ‘M’ | ‘B’
specifies the vertical alignment of the text.

inline_attr : ‘style-attribute(s)’
overrides specific style attributes for the preceding data argument only.

inline_elem : ‘style-element-name’
changes the style element for the preceding data argument only.

Page Methods

obj.page( )
force a page break.

obj.title( arguments )
obj.footnote( arguments )
add a title or footnote to the system.

text : ‘string’ | variable-name
specifies the text value of the title / footnote. This is a required argument for this method.

start : integer
specifies the title / footnote index to start with. Valid numeric range is 1–10.

Absolute Layout Methods

(obj.layout_absolute( arguments )
start and end an absolute layout, respectively.

x : ‘dimension’
specifies the x and y starting point.

y : ‘dimension’
specifies the height and width.

height : ‘dimension’
specifies the height and width.

width : ‘dimension’
specifies the height and width.

obj.region( arguments )
start a new region within the layout. Arguments are same as for layout_absolute( ).

Gridded Layout Methods

(obj.layout_gridded( arguments )
obj.layout_end( )
start and end a gridded layout, respectively.

columns : integer
rows : integer
specifies the number of columns and rows.

height : ‘dimension’
specifies the height and width.

width : ‘dimension’
specifies the height and width.

obj.region( arguments )
start a new region within the gridded layout.

column_span : integer
rows_span : integer
specifies the number of columns that the region will occupy. The default value is 1.

specifies the number of rows that the region will occupy. The default value is 1.