The Data Set Data Vector class is used for communication between the Data Set Data Model class and the Data Set Model class. It is also used by the Data Table and Data Form classes.

Most users of the Data Table and Data Form classes need not be concerned with the Data Set Data Vector class. In these classes, the Data Set Data Vector class is used internally. The Data Set Data Vector class is useful to users who want to do either of the following:

- call methods of their own instance of the Data Set Data Model class or the Data Set Model class directly
- make their own subclass of either the Data Set Data Model class or the Data Set Model class.

The following output shows the relationship between the elements of a table (or SAS data set) and the methods of the Data Set Data Vector class.
Output 60.1  CONTENTS Procedure Output

```
                 The SAS System
                 CONTENTS PROCEDURE

Data Set Name:  SASUSER.CLASS               Observations:   19
Member Type:    DATA                         Variables:      5
Engine:         V611                          Indexes:        0
Created:        16:32 Friday, February 28, 1992  Observation Length: 33
Last Modified:  11:05 Thursday, May 25, 1995     Deleted Observations:    0
Protection:     Compressed: NO
Data Set Type:  Sorted: NO
Label:          Student information

-----Engine/Host Dependent Information-----

        Data Set Page Size:  4096
    Number of Data Set Pages:  1
        File Format:  607
    First Data Page:  1
  Max Obs per Page:  123
  Obs in First Data Page:  19
        FILETYPE:  REGULAR

                 CONTENTS PROCEDURE

-----Alphabetic List of Variables and Attributes-----

#  Variable  Type  Len  Pos  Format  Informat  Label
---------------------------------------------------------------------
  3   AGE   Num   8   9  BEST8.  F8.  Age in years
  4  HEIGHT  Num   8  17  BEST8.  F8.  Height in inches
  1   NAME  Char   8   0   $8.   $8.  First name
  2    SEX  Char   1   8    $1.   $1.  Gender
  5  WEIGHT  Num   8  25  BEST8.  F8.  Weight in pounds

#

the number of columns (variables) in the data set. this value is passed to the
_setNumberOfColumns method and is returned by the _getNumberOfColumns
method.

Variable
the column name. Use the _setName method to set the name and _getName to
retrieve the name of a column.

Type
the column type. Use the _getType method to retrieve the type of a column.

Pos
index of the column in the data set. Pass this value as a column number.

Format
the column format. Use the _setFormat method to set the format and _getFormat
to retrieve the format of a column.

Informat
the column informat. Use the _setInformat method to set the informat and
_getInformat to retrieve the informat of a column.

Label
the column label. Use the _setLabel method to set the label and _getLabel to
retrieve the label of a column.
```
When you use the Data Set Data Model class, the class automatically creates an instance of the Data Set Data Vector class, which it uses to communicate with the Data Set Model class.

The Data Set Data Vector class enables you to read data from a data set (using the _getRow method of the Data Set Model class) and write data to a data set (using the _setRow method of the Data Set Model class). The Data Set Data Vector class also enables you to access the column information passed between the Data Set Data Model and Data Set Model classes.

In order to use an instance of the Data Set Data Vector class, you must first call the _setNumberOfColumns method. This method, in turn, calls the _describeColumns method on the specified instance of the Data Set Data Model class or the Data Set Model class. The _describeColumns method initializes the Data Set Data Vector class instance with the format, informat, label, name, and type for the columns in a row.

If you are using the Data Set Data Vector class to read from a data set, pass the Data Set Data Vector class instance to the _getRow method of the Data Set Model class. Upon returning from this method call, you can use the _getText and _getValue methods (depending on the data type of the column) to retrieve the values from the row that was read.

If you are using the Data Set Data Vector class to write to a data set, use the _setText and _setValue methods to set the values you want to write to the data set in the Data Set Data Vector class instance. Once you have set the column values to be written to the data set, pass the Data Set Data Vector class instance to the _setRow method of the Data Set Model class.

If you are creating your own subclass of the Data Set Model class, you may use the _setFormat, _setInformat, _setLabel, _setName, _setText, and _setValue methods to override the column attributes set by the superclass. If you are creating your own subclass of the Data Set Data Model class, you may use the _getFormat, _getInformat, _getLabel, _getName, _getText, _getType, and _getValue methods to retrieve column attributes set by the Data Set Model class.

Parent:

SASHELP.FSP.OBJECT.CLASS
Class:

SASHELP.FSP.DSDVEC.CLASS

Using the Data Set Data Vector Class

The following example shows how to use several of the methods for this class. In this example, the first row of the table is read. Information about each column, such as column name, type, format, and so on, is retrieved. If the column type is character, then the character text is retrieved for that row; otherwise, the numeric value is retrieved.

```
length colname $ 8 format informat $ 14
label $ 40 text $ 200 type $ 1;
```

Load the data class.

```
INIT:
datcl=loadclass('sashelp.fsp.
```
Using the Data Set Data Vector Class

```
datid=instance(datcl);
veccl=loadclass('sashelp.
    fsp.dsdvec.class');
vecid=instance(veccl);

// Set the table

call send (datid, '_setDataset',
    'sasuser.class');

// Get the number of columns in the table

call send (datid, '_getNumberOfColumns',
    ncols);
call send (vecid, '_setNumberOfColumns',
    ncols, datid);

// Verify the number of columns.

call send (vecid, '_getNumberOfColumns',
    ncols);

// Get a row of data.

call send (datid, '_getRow', vecid, 1);

// For each column, get the name, type, format, informat, and label.

do column=1 to ncols;
call send (vecid, '_getName',
    column, colname);
call send (vecid, '_getType', column,
    type);
call send (vecid, '_getFormat',
    column, format);
call send (vecid, '_getInformat',
    column, informat);
call send (vecid, '_getLabel',
    column, label);

// Put out the column name, type, format, informat, and label.
```

Set the table

```
call send (datid, '_setDataset',
    'sasuser.class');
```

Get the number of columns in the table

```
call send (datid, '_getNumberOfColumns',
    ncols);
call send (vecid, '_setNumberOfColumns',
    ncols, datid);
```

Verify the number of columns.

```
call send (vecid, '_getNumberOfColumns',
    ncols);
```

Get a row of data.

```
call send (datid, '_getRow', vecid, 1);
```

For each column, get the name, type, format, informat, and label.

```
do column=1 to ncols;
call send (vecid, '_getName',
    column, colname);
call send (vecid, '_getType', column,
    type);
call send (vecid, '_getFormat',
    column, format);
call send (vecid, '_getInformat',
    column, informat);
call send (vecid, '_getLabel',
    column, label);
```

Put out the column name, type, format, informat, and label.
put column= colname= type=;
put format= informat= label=;

If the column is numeric, get the value.

if (type='N') then do;
   call send (vecid, '_getValue',
                column, value);
   put value=;
end;

If the column is character, get the text.

if (type='C') then do;
   call send (vecid, '_getText',
                column, text);
   put text=;
end;
end;

term the objects.

call send (datid, '_term');
call send (vecid, '_term');
return;

---

Methods

Methods specific to the Data Set Data Vector class are described here. Inherited methods are described in the Object class.

Note: The col-num argument used in many of the following methods must be >= 1 and <= cols, where cols is the value passed to the _setNumberOfColumns method.

---

Dictionary

__getFormat

Returns the current format for a column in the data vector
**Syntax**

CALL SEND (data-vector-id, 'getFormat', col-num, format);

<table>
<thead>
<tr>
<th>Argument</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>col-num</td>
<td>N</td>
<td>specifies the column number</td>
</tr>
<tr>
<td>format</td>
<td>C</td>
<td>returns the format for the column</td>
</tr>
</tbody>
</table>

---

**getInformat**

Returns the current informat for a column in the data vector

**Syntax**

CALL SEND (data-vector-id, 'getInformat', col-num, informat);

<table>
<thead>
<tr>
<th>Argument</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>col-num</td>
<td>N</td>
<td>specifies the column number</td>
</tr>
<tr>
<td>informat</td>
<td>C</td>
<td>returns the informat for the column</td>
</tr>
</tbody>
</table>

---

**getLabel**

Returns the current label for a column in the data vector

**Syntax**

CALL SEND (data-vector-id, 'getLabel', col-num, label);
<table>
<thead>
<tr>
<th>Argument</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>col-num</td>
<td>N</td>
<td>specifies the column number</td>
</tr>
<tr>
<td>label</td>
<td>C</td>
<td>returns the label for the column</td>
</tr>
</tbody>
</table>

**_getName**

Returns the current name for a column in the data vector

**Syntax**

CALL SEND (data-vector-id, '_getName', col-num, name);

<table>
<thead>
<tr>
<th>Argument</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>col-num</td>
<td>N</td>
<td>specifies the column number</td>
</tr>
<tr>
<td>name</td>
<td>C</td>
<td>returns the name for the column</td>
</tr>
</tbody>
</table>

**_getNumberOfColumns**

Returns the number of columns in the data vector

**Syntax**

CALL SEND (data-vector-id, '_getNumberOfColumns', cols);

<table>
<thead>
<tr>
<th>Argument</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cols</td>
<td>N</td>
<td>returns the total number of columns in the data vector, as specified on the _setNumberOfColumns method</td>
</tr>
</tbody>
</table>

**_getText**

Returns the current text value for a character column in the data vector

**Syntax**

CALL SEND (data-vector-id, '_getText', col-num, text);
### _getType

*Returns the current type for a column in the data vector*

**Syntax**

```plaintext
CALL SEND (data-vector-id, '_getType', col-num, type);
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>col-num</td>
<td>N</td>
<td>specifies the column number</td>
</tr>
<tr>
<td>type</td>
<td>C</td>
<td>returns the value of the column type:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>'N' the column is numeric</td>
</tr>
<tr>
<td></td>
<td></td>
<td>'C' the column is character</td>
</tr>
</tbody>
</table>

### _getValue

*Returns the current value of a numeric column in the data vector*

**Syntax**

```plaintext
CALL SEND (data-vector-id, '_getValue', col-num, value);
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>col-num</td>
<td>N</td>
<td>specifies the column number</td>
</tr>
<tr>
<td>value</td>
<td>N</td>
<td>returns the value for the column</td>
</tr>
</tbody>
</table>

**Details**

The _getValue method should only be called for columns of type 'N' (numeric).
_setFormat

Specifies the format for a column in the data vector

Syntax
CALL SEND (data-vector-id, '_setFormat', col-num, format);

<table>
<thead>
<tr>
<th>Argument</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>col-num</td>
<td>N</td>
<td>specifies the column number</td>
</tr>
<tr>
<td>format</td>
<td>C</td>
<td>specifies the format for the column</td>
</tr>
</tbody>
</table>

Details
The format must be a valid format for the column type.

_setInformat

Specifies the informat for a column in the data vector

Syntax
CALL SEND (data-vector-id, '_setInformat', col-num, informat);

<table>
<thead>
<tr>
<th>Argument</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>col-num</td>
<td>N</td>
<td>specifies the column number</td>
</tr>
<tr>
<td>informat</td>
<td>C</td>
<td>specifies the informat for the column</td>
</tr>
</tbody>
</table>

Details
The informat must be a valid informat for the column type.

_setLabel

Specifies the label for a column in the data vector

Syntax
CALL SEND (data-vector-id, '_setLabel', col-num, label);
### _setName

Specifies the name for a column in the data vector

---

**Syntax**

CALL SEND (data-vector-id, '_setName', col-num, name);

<table>
<thead>
<tr>
<th>Argument</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>col-num</td>
<td>N</td>
<td>specifies the column number</td>
</tr>
<tr>
<td>name</td>
<td>C</td>
<td>specifies a valid SAS name for the column</td>
</tr>
</tbody>
</table>

### _setNumberOfColumns

Specifies the number of columns to set up in the data vector

---

**Syntax**

CALL SEND (data-vector-id, '_setNumberOfColumns', cols, ds-object);

<table>
<thead>
<tr>
<th>Argument</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cols</td>
<td>N</td>
<td>specifies the new number of columns for the data vector</td>
</tr>
<tr>
<td>ds-object</td>
<td>N</td>
<td>specifies the object identifier for the instance of the Data Set Model class with which this instance of the Data Set Data Vector class is used.</td>
</tr>
</tbody>
</table>

**Details**

You must call the _setNumberOfColumns method before passing the identifier of this Data Set Data Vector instance to any methods of the Data Set Model class. This method initializes the column information. Passing an instance of the Data Set Data
Vector class to the Data Set Model class before calling this method results in a program halt. The _setNumberOfColumns method should be called after a call to _setDataset on the instance of the Data Set Model class.

### _setText

**Specifies the text for a character column in the data vector**

#### Syntax

CALL SEND (data-vector-id, '_setText', col-num, text);

<table>
<thead>
<tr>
<th>Argument</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>col-num</td>
<td>N</td>
<td>specifies the column number</td>
</tr>
<tr>
<td>text</td>
<td>C</td>
<td>specifies the text for the column. The length of text should be no longer than the data length of the column.</td>
</tr>
</tbody>
</table>

#### Details

The _setText method should only be called for columns of type 'C' (character).

### _setValue

**Specifies the value for a numeric column in the data vector**

#### Syntax

CALL SEND (data-vector-id, '_setValue', col-num, value);

<table>
<thead>
<tr>
<th>Argument</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>col-num</td>
<td>N</td>
<td>specifies the column number</td>
</tr>
<tr>
<td>value</td>
<td>N</td>
<td>specifies the value for the column</td>
</tr>
</tbody>
</table>

#### Details

The _setValue method should only be called for columns of type 'N' (numeric).