Paper 009-2008

Picture Perfect: Centering Frames in SAS/AF®

Greg McLean, Statistics Canada, Ottawa, Ontario, Canada

ABSTRACT

The ability to create professional looking and easy to use GUI (Graphical User Interface) applications is both a skill as well as an art form that is usually appreciated by the end user. Of course the primary goal of the GUI is to allow the end user to perform the task the application was intended for. However, the way the application is presented to the user will have an effect on productivity, acceptability and ease of use.

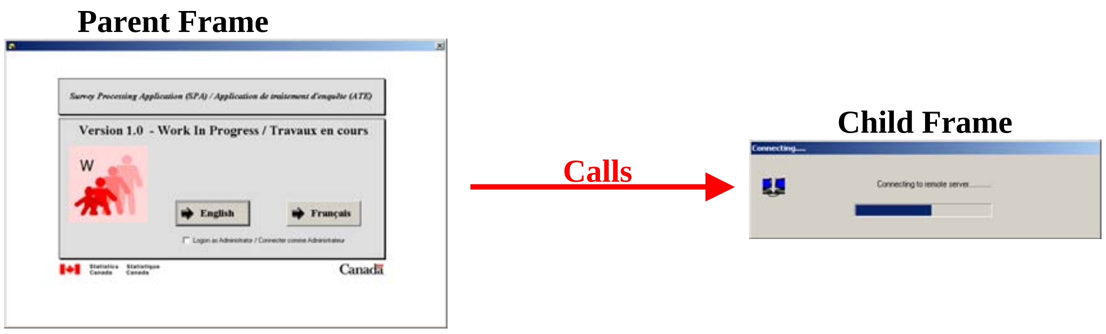
With different monitor sizes, screen resolutions and SAS/AF Frame sizes, it is very important that all Frames be viewed in a common or standardized manner. So why not have everything centered!!

This paper will illustrate a simple method that can be used to center Frames within your GUI applications. The intended audience for this paper is SAS developers with a medium to advanced knowledge of SAS/AF and SCL (Source Component Language) software.

INTRODUCTION

Just as we would not hang a picture on the wall crooked, we should also take care in how we build graphical user interfaces; in particular, how we place our Frames. In fact, building graphical user interfaces has really become a specialty or niche for some people. Various techniques, templates and even organization of visual objects will have a direct impact on productivity, acceptability and ease of use.

This paper will illustrate a simple way to have one window appear centered on another window. Throughout, reference will be made to the "Parent" Frame as well as the "Child" Frame. The "Parent" Frame is the Frame that will display first, thus calling the "Child" Frame to appear on top.

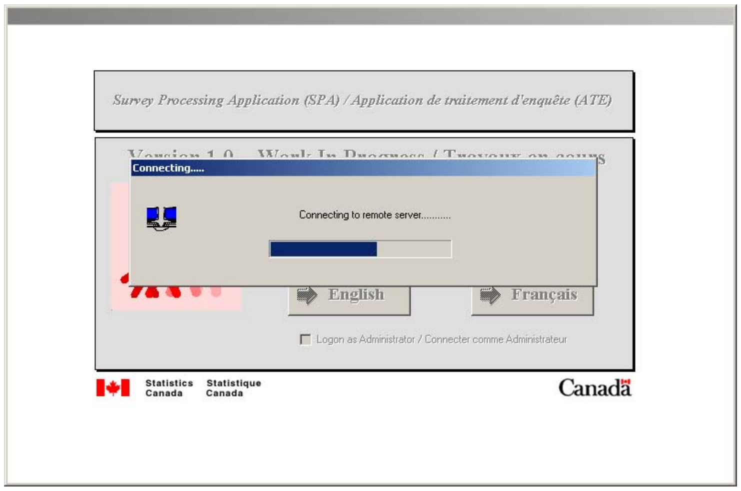


Survey Processing Application (SPA) / Application de traitement d'enquête (ATE)

Version 1.0 - Work In Progress / Travaux en cours

Logon as Administrator / Connecter comme Administrateur

Connecting to remote server..........



Survey Processing Application (SPA) / Application de traitement d'enquête (ATE)

Connecting to remote server..........

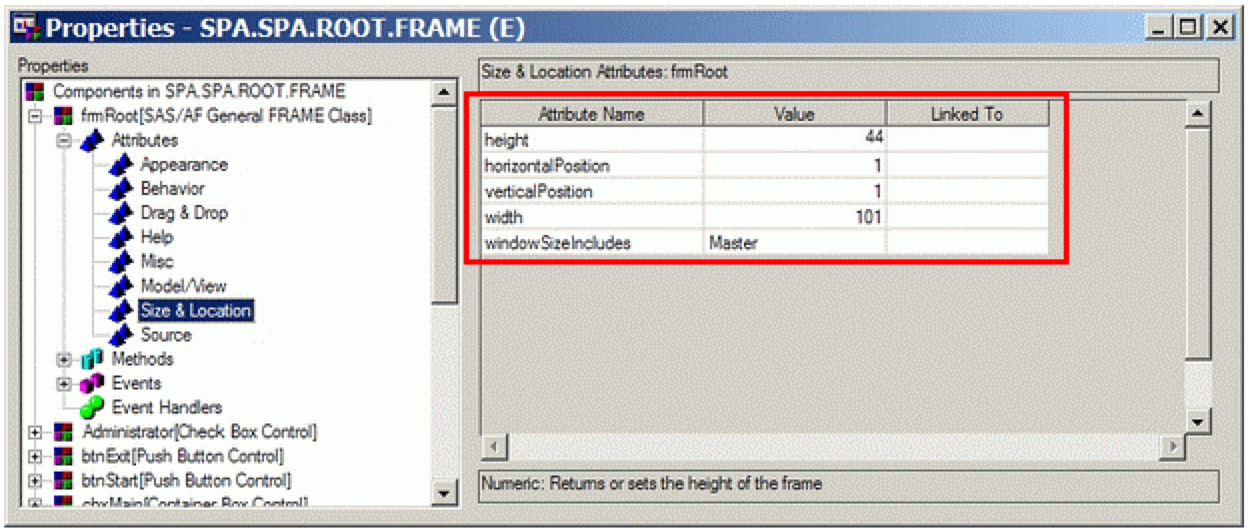
Logon as Administrator / Connecter comme Administrateur

Resulting Appearance

BACKGROUND

Before discussing how one centers a Frame we first need to talk about how a Frame gets displayed and which Frame Properties have an impact on the appearance. When a Frame is called there are five Frame properties that are referenced to decide where the Frame should be positioned and how big it should appear. In particular these Frame properties are:

- ***HorizontalPosition***
- ***VerticalPosition***
- ***Height***
- ***Width***
- ***WindowSizeIncludes***



Properties - SPA.SPA.ROOT.FRAME (E)

Properties

- Components in SPA.SPA.ROOT.FRAME
  - frmRoot[SAS/AF General FRAME Class]
    - Attributes
      - Appearance
      - Behavior
      - Drag & Drop
      - Help
      - Misc
      - Model/View
      - Size & Location
      - Source
    - Methods
    - Events
    - Event Handlers
  - Administrator[Check Box Control]
  - btnExit[Push Button Control]
  - btnStart[Push Button Control]
  - cbxMain[Container Box Control]

Size & Location Attributes: frmRoot

Attribute Name	Value	Linked To
height	44	
horizontalPosition	1	
verticalPosition	1	
width	101	
windowSizeIncludes	Master	

Numeric: Returns or sets the height of the frame

Typically these properties are set by the developer at compile time. The unit of measure used is rows and columns. Of course it is true that the developer could set these 5 property values in advance in such a way that the Child Frame would appear centered on its Parent Frame. However, as one knows, windows can be moved and resized at run-time. As the **WindowSizeIncludes** property does not usually change, it is typically set at compile time. Therefore, a more dynamic approach to setting the other 4 Frame properties will be discussed.

CENTERING THE FRAME

As the child Frame gets displayed, programmatically we want to recalculate some of the Frame properties based on the position of the parent Frame. Of course the first set of information needed would be information about the Parent Frame. Thus, this information must be passed in to the Child Frame before display. A simple solution (one of many) is to pass the list identifier of the Frame properties of the Parent Frame to the Child Frame as a Global Macro Variable.

Example

The following SCL code would be in the parent Frame SCL. Note that the Macro Variable called "PARENT_FRAME" will contain the list identifier of the Parent Frame properties (_FRAME_).

```
CALL SYMPUTN(“PARENT_FRAME”,_FRAME_);
CALL DISPLAY(“CHILD.FRAME”);
```

The following SCL code would be placed in the INIT section of the Child Frame SCL.

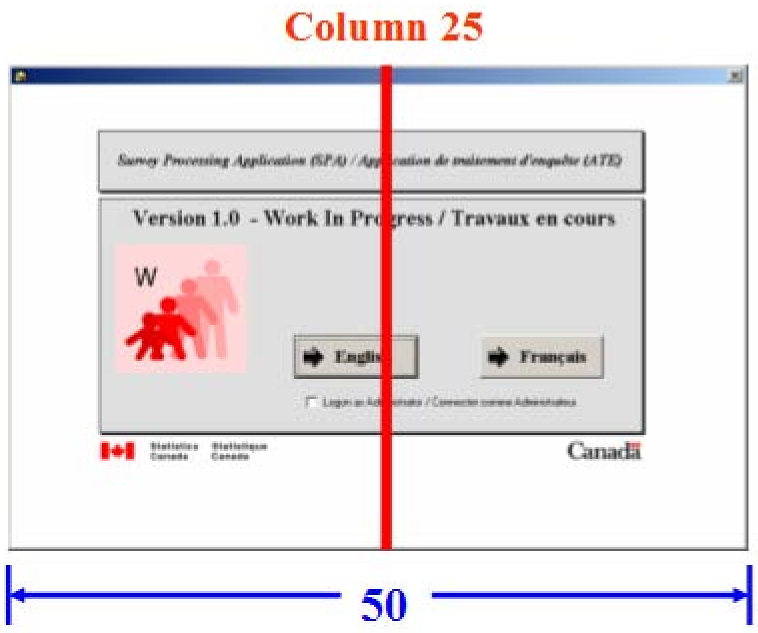
```
parent = SYMGETN(“PARENT_FRAME”);
```

Once the Child Frame has access to the Parent Frame properties (via the passed list identifier), one can then recalculate dynamically the following Child properties to ensure that the Child Frame appears centered:

- ***HorizontalPosition***
- ***VerticalPosition***

CALCULATING "HORIZONTALPOSITION"

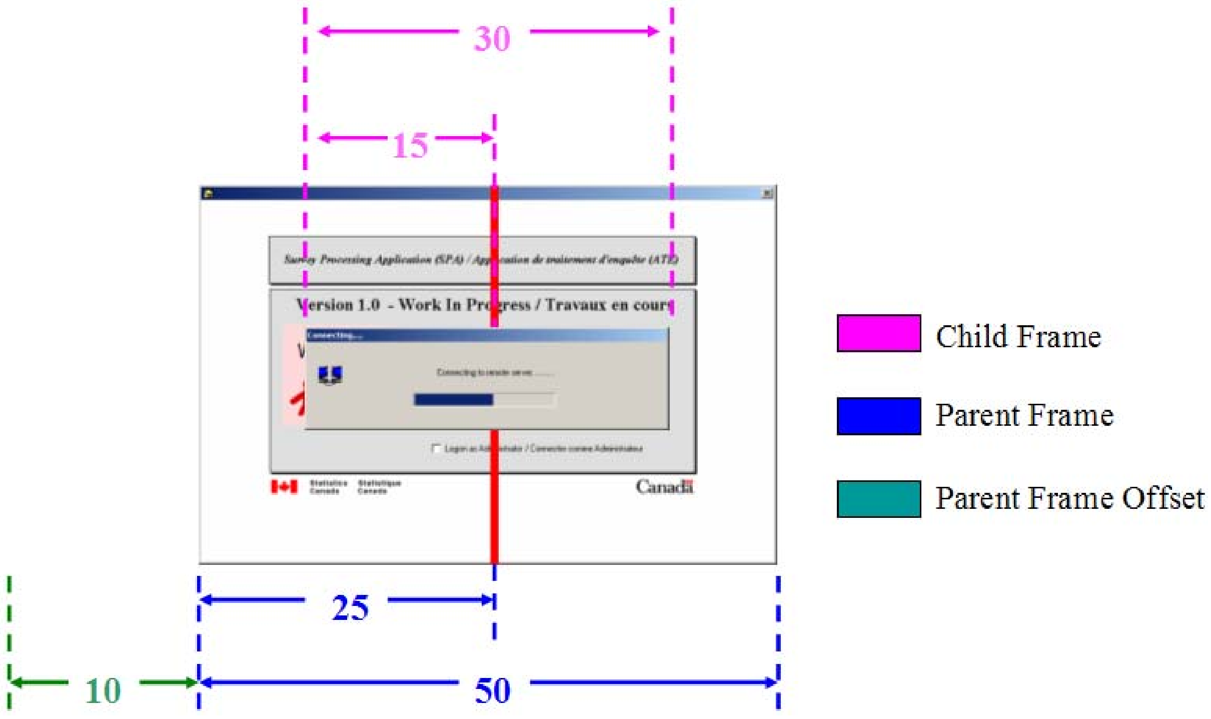
Before calculating ***HorizontalPosition*** of the Child Frame, the horizontal center column of the Parent Frame must be determined. To do this simply take the width of the Parent Frame and divide by 2 (use the **INT** function to ensure an Integer result).


Column 25
Survey Processing Application (SPA) / Application de traitement d'enquête (ATE)
Version 1.0 - Work In Progress / Travaux en cours
English
Français
Logon as Administrator / Connecter comme Administrateur
Statistics Canada
Statistique Canada
Canada
50

```
ParentColumnCenter   = INT(GETNITEMN(parent,"NCOL") / 2);
                     = INT(50 / 2)
                     = 25
```

Note: NCOL refers to the number of columns (width of Parent Frame).

Now that the center column of the Parent Frame is known, we then can calculate the horizontal position of the Child Frame. Basically, half of the Child Frame must appear to the left of the centre column of the Parent Frame and the other half to the right.


30
15
Survey Processing Application (SPA) / Application de traitement d'enquête (ATE)
Version 1.0 - Work In Progress / Travaux en cours
Connecting...
Statistics Canada Statistique Canada
Canadä
25
10
50
Child Frame
Parent Frame
Parent Frame Offset

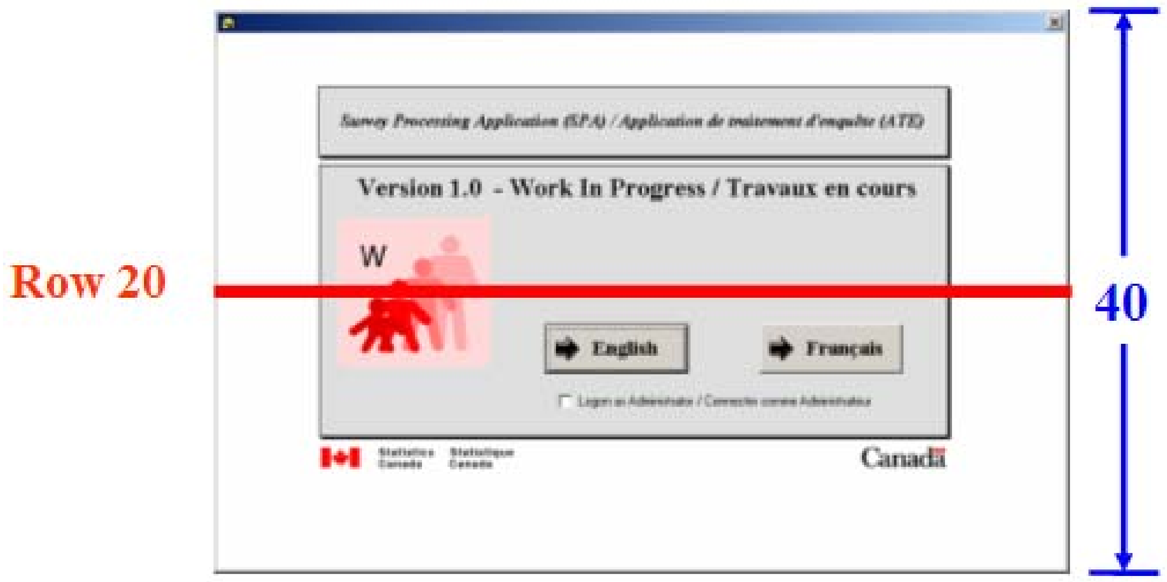
Therefore, we determine half the width of the Child Frame and subtract that amount from the center column of the Parent Frame. This would be all that is needed assuming that the Parent Frame was positioned in the first column. However, if the Parent Frame is offset from column one, the Child Frame must also be offset by the same number of columns.

```
ChildHorizintalPosition =GETNITEMN(parent,"SCOL") + ParentColumnCentre – INT(_SELF_.width/2);
                        = 10 + 25 – (30 / 2)
                        = 35 – 15
                        = 20
```

Notes: Because **"_SELF_**" is referenced in the Child Frame, it would refer to the list of Child Frame properties.

SCOL refers to the starting column (of Parent Frame).

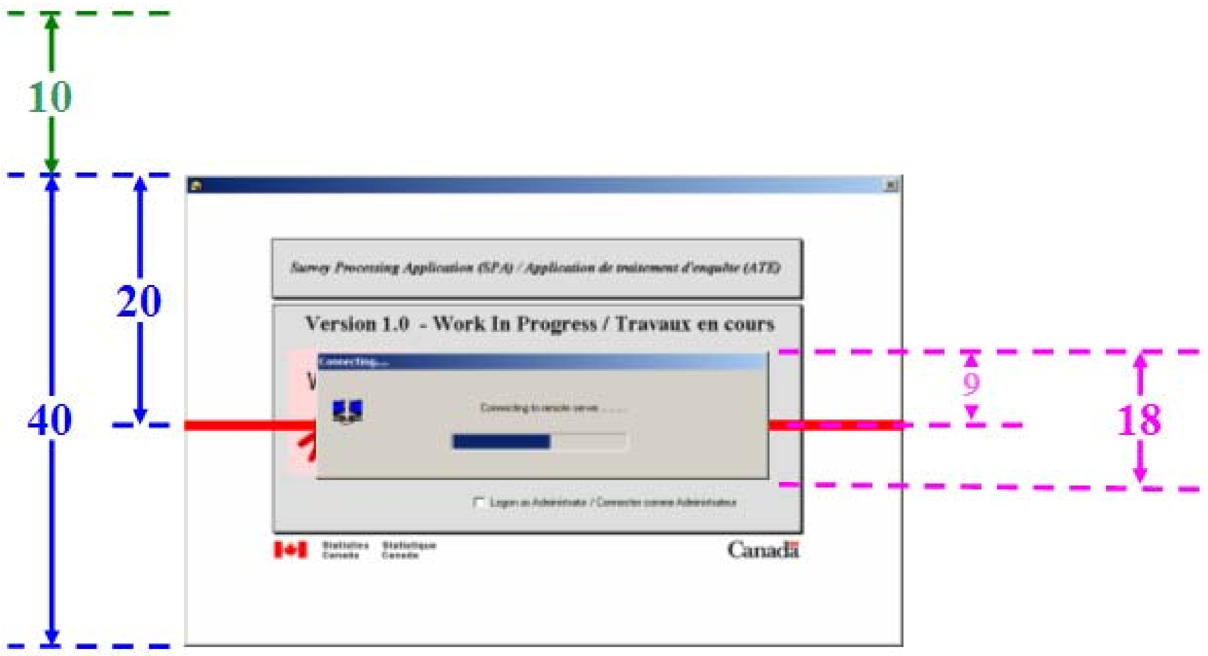
CALCULATING "VERTICALPOSITION"
The next value that needs to be calculated is the vertical position of the Child Frame. Again, before we can make this calculation, we need to determine information about the Parent Frame; we need to determine the vertical center row of the Parent Frame. To do this simply take the height of the Parent Frame and divide by 2 (use the **INT** function to ensure an Integer result).


Row 20
40
Survey Processing Application (SPA) / Application de traitement d'enquête (ATE)
Version 1.0 - Work In Progress / Travaux en cours
English
Français
Logon as Administrator / Connexion comme Administrateur
Statistics Canada Statistique Canada
Canada

```
ParentRowCenter = INT(GETNITEMN(parent,"NROW") / 2);
                = INT(40 / 2)
                = 20
```

Note: NROW refers to the number of rows (height of Parent Frame).

Now that the center row of the Parent Frame is known, we then can calculate the vertical position of the Child Frame. Basically, half of the Child Frame must appear above the centre row of the Parent Frame and the other half below.


10
20
40
9
18
Survey Processing Application (SPA) / Application de traitement d'enquête (ATE)
Version 1.0 - Work In Progress / Travaux en cours
Connecting...
Logon as Administrator / Connecter comme Administrateur
Statistics Canada Statistique Canada
Canada

🟪 Child Frame

🟦 Parent Frame

🟩 Parent Frame Offset

Therefore, we determine half the height of the Child Frame and subtract that amount from the center row of the Parent Frame. This would be all that is needed assuming that the Parent Frame was positioned in the first row. However, if the Parent Frame is offset from row one, the Child Frame must also be offset by the same number of rows.

```
ChildVerticalPosition = GETNITEMN(parent,"SROW") + ParentRowCentre – INT(_SELF_.height/2);
                      = 10 + 20 – (18 / 2)
                      = 30 – 9
                      = 21
```

Notes: Because "**_SELF_**" is referenced in the Child Frame, it would refer to the list of Child Frame properties.

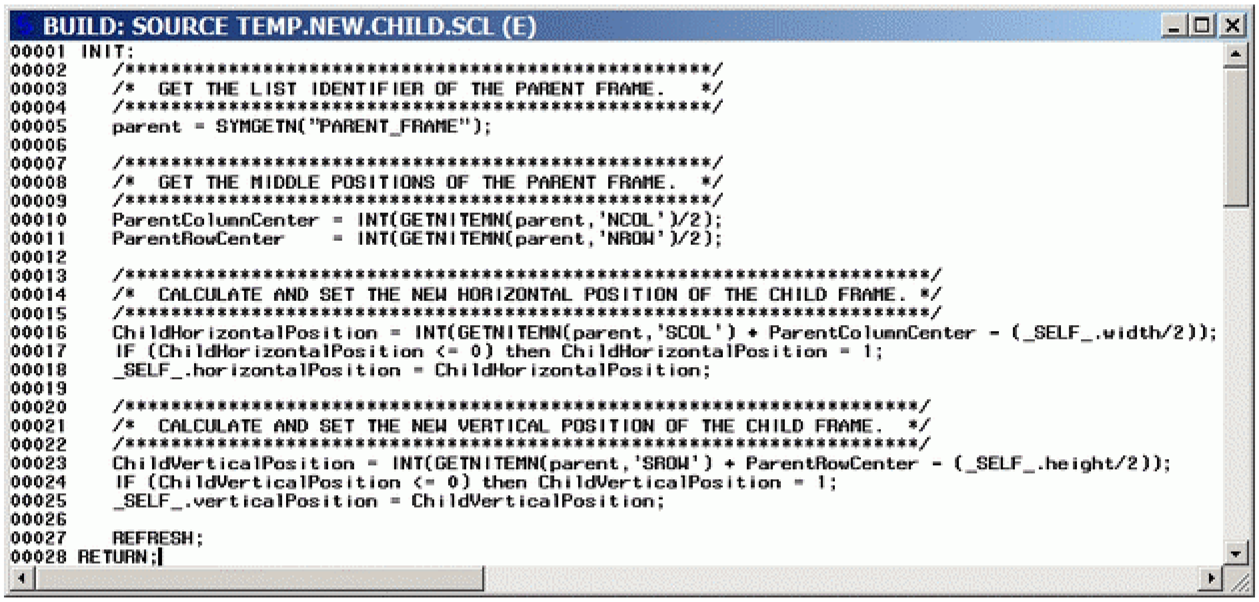
SROW refers to the starting row (of the Parent Frame).

WHERE TO PLACE THE SCL CODE

The two calculations mentioned above (**HorizontalPosition** and **VerticalPosition**) must be calculated somewhere in the Child Frame. Two likely places for these calculations will be discussed further.

INIT SECTION

A simple solution to recalculate the Child Frame **VerticalPosition** and the **HorizontalPosition** is to place the SCL code in the **INIT** section of the Child Frame. However, there is one drawback from doing this. That is, the user will see a momentary flash on the screen. The Child Frame will get displayed for a fraction of a second in the position defined by the properties specified at compile time. Then, as the **INIT** section runs, the recalculation of **VerticalPosition** and **HorizontalPosition** is performed and the Frame then refreshes to its new centered position. Although this flash will be minimal it is in fact noticeable and may not be the best solution.

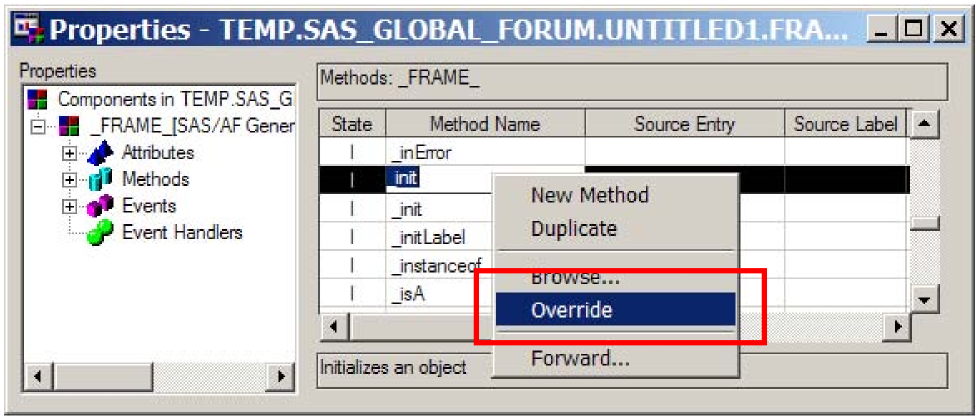


```
INIT:
   /****************************************************/
   /*  GET THE LIST IDENTIFIER OF THE PARENT FRAME.   */
   /****************************************************/
   parent = SYMGETN("PARENT_FRAME");

   /****************************************************/
   /*  GET THE MIDDLE POSITIONS OF THE PARENT FRAME.  */
   /****************************************************/
   ParentColumnCenter = INT(GETNITEMN(parent,'NCOL')/2);
   ParentRowCenter    = INT(GETNITEMN(parent,'NROW')/2);

   /*******************************************************************/
   /*  CALCULATE AND SET THE NEW HORIZONTAL POSITION OF THE CHILD FRAME. */
   /*******************************************************************/
   ChildHorizontalPosition = INT(GETNITEMN(parent,'SCOL') + ParentColumnCenter - (_SELF_.width/2));
    IF (ChildHorizontalPosition <= 0) then ChildHorizontalPosition = 1;
    _SELF_.horizontalPosition = ChildHorizontalPosition;

   /******************************************************************/
   /*  CALCULATE AND SET THE NEW VERTICAL POSITION OF THE CHILD FRAME.  */
   /******************************************************************/
   ChildVerticalPosition = INT(GETNITEMN(parent,'SROW') + ParentRowCenter - (_SELF_.height/2));
    IF (ChildVerticalPosition <= 0) then ChildVerticalPosition = 1;
    _SELF_.verticalPosition = ChildVerticalPosition;

   REFRESH;
RETURN;
```

OVERRIDE METHOD

The second solution will render a more professional result. And in fact is the method used by many SAS Graphical Applications at Statistics Canada. The idea or concept here is to recalculate the Child Frame **VerticalPosition** and the **HorizontalPosition** before the Child Frame is initially displayed. To do this we must "*Override*" the "**_INIT**" method for the Child Frame. This will allow us to perform the calculation before the "**INIT**" section is even run. It can be thought of as "pre-INIT" SCL code.

First, at compile time, we must tell the Child Frame that we wish to override the "**_INIT**" method and tell it where to find the corresponding SCL code to run. To do this we go to the Child **_FRAME_** properties window and click on "**Methods**". Scroll through the list of "**Methods**" and find the first "**_INIT**" method. There are two "**_INIT**" methods with different signatures. Select the one that has no parameters (Signature = ()V). With the mouse, click on this entry using the secondary mouse button. From the displayed list select "*Override*".



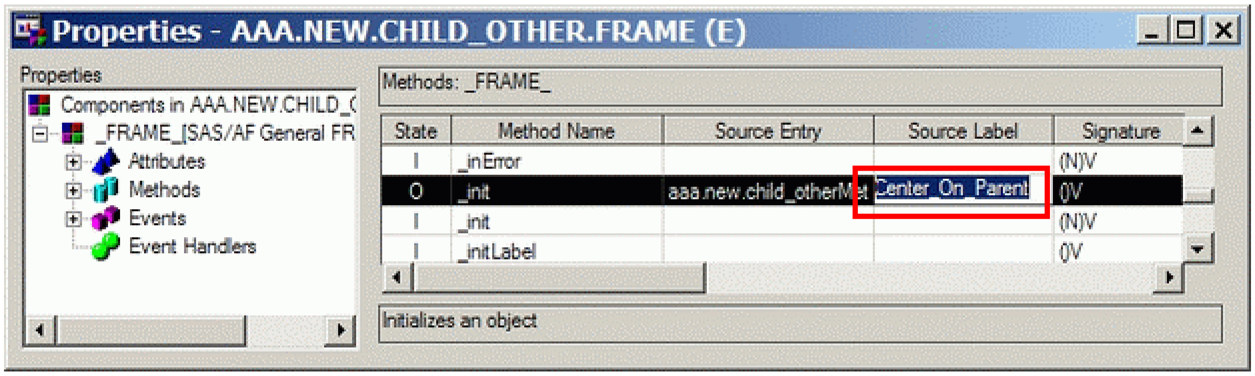
State	Method Name	Source Entry	Source Label
I	_inError		
I	_init		
I	_init		
I	_initLabel		
I	_instanceof		
I	_isA		

We then must specify a 4 level name of the entry which contains the SCL code to be run when the Child Frame is called. Thus a new SCL entry must be created first.

<SAS LIBREF>. <Catalog Name>. <Entry Name> .SCL

Place the cursor in the "Source Entry" field for the "_**INIT**" method or use … beside the field to make or type your selection.

We then must specify the "Label" within the new SCL entry that is to be used or run. Because many methods or "Labels" can exist in a given SCL entry, we must be specific about the one to be used. Place the cursor in the "Source Label" field for the "_**INIT**" method and type your "Label" name. In our example we will call it "Center_On_Parent".

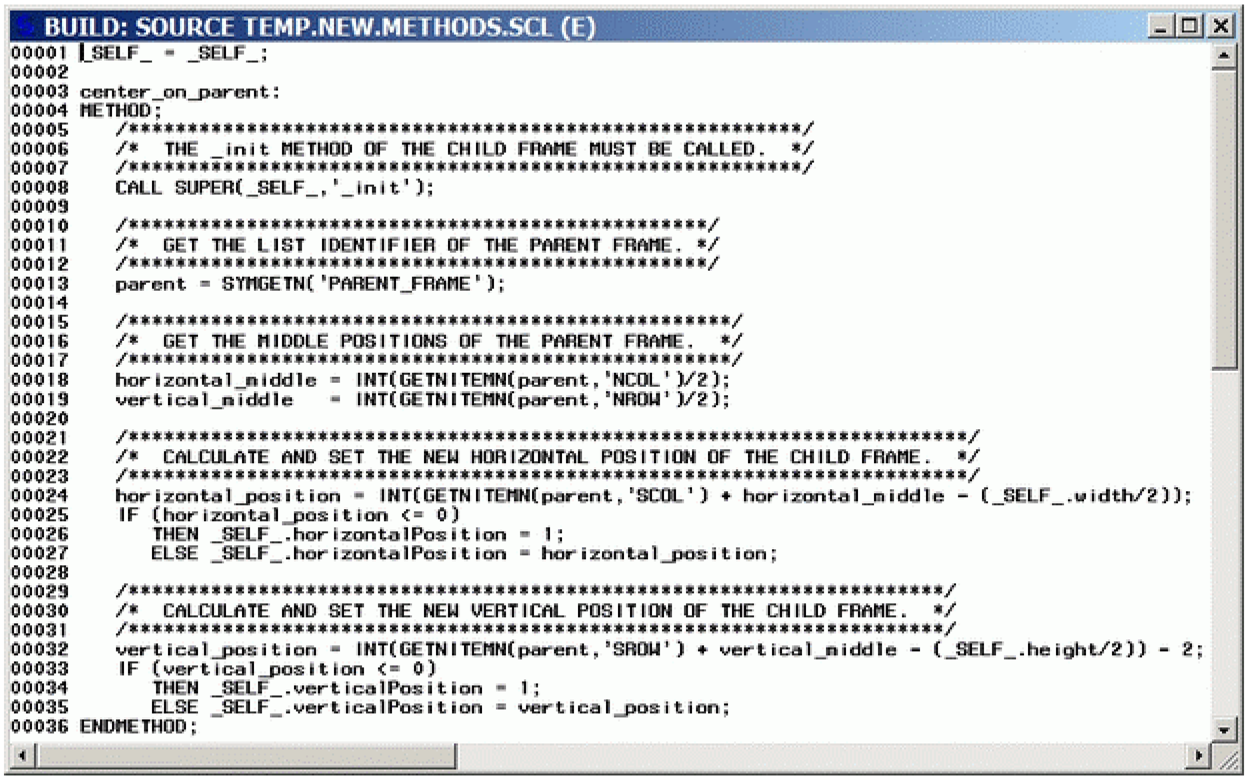

Properties - AAA.NEW.CHILD_OTHER.FRAME (E)
Properties
Components in AAA.NEW.CHILD_(
FRAME[SAS/AF General FR
Attributes
Methods
Events
Event Handlers

Methods: _FRAME_

State	Method Name	Source Entry	Source Label	Signature
I	_inError			(N)V
O	_init	aaa.new.child_otherMet	Center_On_Parent	()V
I	_init			(N)V
I	_initLabel			()V

Initializes an object

Next we must create the SCL code that will get run when the _**INIT** method is called. Typically a SAS Catalog entry (of type SCL) would be created in the same catalog as the Frames and SCL entries (GUI system). In fact, other overridden methods could also be placed in this same SCL entry. So let's assume that we create a new SCL entry within the same SAS catalog and call this new SCL entry: "METHODS". The following code would be placed in this entry:



```
_SELF_ = _SELF_;

center_on_parent:
METHOD;
   /****************************************************************/
   /*  THE _init METHOD OF THE CHILD FRAME MUST BE CALLED.  */
   /****************************************************************/
   CALL SUPER(_SELF_,'_init');

   /****************************************************/
   /*  GET THE LIST IDENTIFIER OF THE PARENT FRAME. */
   /****************************************************/
   parent = SYMGETN('PARENT_FRAME');

   /*****************************************************/
   /*  GET THE MIDDLE POSITIONS OF THE PARENT FRAME.  */
   /*****************************************************/
   horizontal_middle = INT(GETNITEMN(parent,'NCOL')/2);
   vertical_middle   = INT(GETNITEMN(parent,'NROW')/2);

   /**********************************************************************/
   /*  CALCULATE AND SET THE NEW HORIZONTAL POSITION OF THE CHILD FRAME.  */
   /**********************************************************************/
   horizontal_position = INT(GETNITEMN(parent,'SCOL') + horizontal_middle - (_SELF_.width/2));
   IF (horizontal_position <= 0)
      THEN _SELF_.horizontalPosition = 1;
      ELSE _SELF_.horizontalPosition = horizontal_position;

   /********************************************************************/
   /*  CALCULATE AND SET THE NEW VERTICAL POSITION OF THE CHILD FRAME.  */
   /********************************************************************/
   vertical_position = INT(GETNITEMN(parent,'SROW') + vertical_middle - (_SELF_.height/2)) - 2;
   IF (vertical_position <= 0)
      THEN _SELF_.verticalPosition = 1;
      ELSE _SELF_.verticalPosition = vertical_position;
ENDMETHOD;
```

Finally, this new SCL entry must be compiled and saved.

Now, when the Child Frame is called by the Parent Frame, the calculation of **VerticalPosition** and **HorizontalPosition** will be calculated and set before the Child Frame is initially displayed. Thus, the Child Frame will display as being centered on its Parent Frame.

SUMMARY

Although the concept of centering Frames may seem to be trivial and insignificant, it is one of many other enhancements that can be made to ensure the Graphical User Interface is more readable, more useable and hopefully more accepted by its end users.

So next time you build your GUI with SAS/AF, why not make it:

“Picture Perfect”

CONTACT INFORMATION

For information on topics covered in this paper please contact:

Statistics Canada Statistique Canada

Greg McLean
Project Leader – SAS Technology Centre
System Development Division
R.H. Coats Building, 14th Floor, Section Q

Ottawa, Ontario, Canada K1A 0T6
(613) 951-2396 Fax (613) 951-0607
greg.mclean@statcan.ca

Canadä