GENSCRNX 1.8

INTRODUCTION

GENSCRNX.PRG extends the control of code generated from FoxPro's screen builder. After Generate... is selected when using the Screen Builder, GENSCRNX first copies the .SCX database and then updates it based on comments in the snippets and setup code. Also, GENSCRNX places the .SPR into a memo field after its created to make possible code changes and/or replacements after GENSCRN. The ability to define each object into a global database called FOXSCX.DBF is performed when a define object directive is placed in an object's comment code. The FOXSCX.DBF contains the same structure as FoxPro's .SCX files except has added fields for object name, field, library, and other objects that it bases from. New screens can be created without snippet code by simply placing a base object directive in the comment snippet with the appropriate name. GENSCRNX updates the .SCX before passing it to GENSCRN. Drivers can be defined in the CONFIG.FP and screen Setup snippet code. Every driver is called once for each record in the .SCX before GENSCRN generates code. The driver may update the .SCX database with no limitations. Objects can be manipulated or replaced by pure FoxPro code using a driver procedure. GENSCRNX handles the code replacements to the .SPR. A driver may use pre-made functions contained in GENSCRNX which handle the .SCX record update for code replacement, template insertion, and other .SCX update functions. A DEFINE WINDOW command can be inserted in the .SPR between the GET/SAY fields in the Screen Layout section.. Multiple drivers may also be selected for functions such as 3D effects or auto insertion of push buttons (Next, Previous, Append, Delete, etc.). GENSCRNX is entirely written in FoxPro and fully compatible with FoxPro 2.0 and FoxPro 2.5 (all platforms). The FOXSCX library database can be updated when referenced by FoxPro 2.0 and/or FoxPro 2.5 for MS-DOS without any conversion. FoxPro 2.5 for MS-DOS MS-DOS applications can be built by referencing objects created with FoxPro 2.0 and vice versa. The FOXSCX.DBF database can contain records for FoxPro 2.5 (all platform) while GENSCRNX automatically handles the record relation between platforms.

Notes:
CONFIG.FP relates to FoxPro 2.0 and FoxPro 2.5 DOS.
CONFIG.FPW relates to FoxPro 2.5 Windows.

FEATURES

- Extended control over FoxPro's Screen Builder without changing GENSCRN. GENSCRNX can be though of as a WHEN and VALID for GENSCRN.

- Option for compiling the output file when generating from the Screen

Builder.

- Option for displaying the .SPR and .ERR files if an .ERR file is generated after compiling the output file when generating from the Screen Builder.

- Ability to store screen objects into a database library.

- Ability to retrieve screen objects from a database library with support of multiple inheritance (expressions are separated by .AND. while procedures are appended).

- Option to set Read level settings (OpenFiles, CloseFiles, Modal, OutFile, etc.) from with the screens setup that override the Generate dialog checkboxes. This allows settings to be saved with the screen without using a project.

- Ability to insert records contained in a separate .SCX file at compile time. All records row and column information is automatically adjusted. This allows subforms to be inserted without copy and paste. If the inserted screen is updated, the screen importing it can be re-generated without change.

- Ability to insert FoxPro code in place of any screen object. This allows a line or multiple lines of FoxPro code to be generated between GET commands in the Screen Layout section.

- Ability to block a GET command with any IF/ENDIF statement.

- Ability to specify any SIZE clause and override the SIZE setting defaulted by the Screen Builder.

- Ability to remove the SIZE clause from any GET command.

- Ability to create .PRG drivers that update the .SCX database at compile time before GENSCRN is called. This allows external programs to be created that automatically add, update, or remove code of any screen snippet. Drivers can make function calls to many of GENSCRNX's built in function library for parsing or insertion of .SCX information.

- Support for any expression to be evaluated at compile time using {{<expC>}} in any snippet or field. GENSCRNX will evaluate <expC> at compile time and replace {{<expC>}} with its result. If <expC> was an external function and the command was placed in the Setup snippet, the external function could act like a #INCLUDE function by returning multiple lines of code.
Example:
If the following command was in the Setup snippet and assuming the current date was 06/01/93:

```
WAIT '{{DATE()}}' WINDOW NOWAIT
```

the following code would be placed in the .SPR:

```
WAIT '06/01/93' WINDOW NOWAIT
```

INSTALLATION

After unzipping GENSCRNX.ZIP, copy GENSCRNX.PRG to all existing FoxPro 2.x directories

Change all CONFIG.FP and CONFIG.FPW files to:

_GENSCRN="<path>GENSCRNX.PRG"

MVCOUNT=512

Notes:
If MVCOUNT is already set to a number greater than 512, then do not change it. If MVCOUNT is set to a number less than 512, then change the number to 512. If a line containing MVCOUNT does not exist, then create one as above.

CONFIG.FP/CONFIG.FPW OPTION SETTINGS

_GENSCRNX

Specifies program to generate .SPR file from .SCX database.

Default:
_GENSCRNX="<path>GENSCRN.PRG" in FoxPro start directory

Example:
_GENSCRNX="C:\MYDIR\MYGENSCN.PRG"

Notes:
When _GENSCRN="<pathname>\GENSCRNX.PRG", then _GENSCRNX is used to specify which program is used to generate screen code. If _GENSCRNX is not specified in the CONFIG.FP/CONFIG.FPW, the default setting is GENSCRN.PRG located in FoxPro's start-up directory.

_FOXSCX

Specifies database used for object library records.

Default:
_FOXSCX="FOXSCX.DBF" in FoxPro start directory

Example: _FOXSCX="C:\MYDIR\FOXSCX.DBF"

Notes:
It is recommended that all FoxPro 2.x CONFIG.FP/CONFIG.FPW contain the same _FOXSCX setting.

_SCXDRV1

Specifies global driver program.

Default:
_SCXDRV1=""

_SCXDRV1 to _SCXDRV8

Specifies global driver program. The numbers 1-8 represent various driver hooks throughout GENSCRNX while the .SCX databases is being generated.

Example:
_SCXDRV5="C:\3DFOX\3D"

_SPRDRV1

Specifies global driver program.

Default:
_SPRDRV1=""

_SPRDRV1 to _SPRDRV6

Specifies global driver program. The numbers 1-6 represent various driver hooks throughout GENSCRNX while the .SPR file is being updated.

Example:
_SPRDRV1="C:\MYDIR\SPRUPD1"

GENSCRNX

Specifies GENSCRNX functions enabled (ON) or disabled (OFF).

Default:
GENSCRNX=ON

COMPSPR

Specifies auto-compilation of .SPR file. A public variable called _COMPSPR to override the COMPSPR setting.

Default:
COMPSPR=OFF

Important:
This setting is ignored during screen building from projects.

DISPSPR

Specifies auto-display of .SPR and .ERR files if an .ERR file is exists. A public variable called _DISPSPR to override the DISPSPR setting.

Default:
DISPSPR=OFF

Important:
DISPSPR=ON may cause a file sharing error when SHARE.EXE is installed.

SETUP SNIPPET DIRECTIVE REFERENCE

#:SECTION 3

Used in Setup snippet (like #SECTION 1 | 2) to insert code after GETs and before READ in the Screen Layout.

*:AUTORUN

Automatically releases screen after generation and executes generated file. *:AUTORUN is automatically disabled if either a compiled file is not properly generated or a compile error was detected via the COMPSPR=ON.

*:COMPSPR

Overrides COMPSPR=OFF in CONFIG.FP/CONFIG.FPW.

New: *:DISPSPR

Overrides DISPSPR=OFF in CONFIG.FP/CONFIG.FPW.

*:SET OPENFILES ON | OFF

Open files.

Example:
*:SET OPENFILES ON

*:SET CLOSEFILES ON | OFF

Close files.

*:SET DEFWINDS ON | OFF

Define windows.

*:SET RELWINDS ON | OFF

Release windows.

*:SET READCYCLE ON | OFF

Read cycle.

*:SET MULTREADS ON | OFF

Multiple READs.

*:SET NOLOCK ON | OFF

READ nolock.

*:SET MODAL ON | OFF

Modal.

*:SET PLATONLY ON | OFF

Current platform objects only. If this setting is ON, GENSCRN will not generate code for other platform code but GENSCRNX will still process all platform records. Setting PLATONLY='ON' in the CONFIG.FP/CONFIG.FPW files will cause GENSCRNX to not pre or post process other platform records. See ADDITIONAL INFORMATION section below for controlling this setting using a public variable.

*:SET BORDERGETS ON | OFF

Border for GETs.

*:SET ASSOCWINDS TO <window title list>

Assoc. windows list. The <window title list> is appended to Assoc. windows list from screen or project.

Example:
*:SET ASSOCWINDS TO Calculator,Calendar

*:OUTFILE <file>

Output file name. *:OUTFILE is disabled when building screen from a project.

Example:
*:OUTFILE TEST.PRG

*:PRG

A Setup snippet directive called *:PRG that is used to automatically change the .SPR extension to a .PRG extension and also add the #NOREAD PLAIN directive to the Setup snippet. The *:PRG directive used in conjunction with screen objects with the *:INSTXT directive will allow a .PRG file to be created that has no GETs, SAYs, or READ, while the screen builder Object Order controls the order of the FoxPro source code generated in the .PRG file. When building a screen from a project, the *:PRG directive is ignored since the project expects that file specified in the project to be generated and will abort project generation if the project specified file is not created..

*:PJXSET

Place in the Setup snippet before the *:PRG directive or any *:SET directive to force the project information to have priority settings when building the screen from a project.

Example:
If the following was in the Setup snippet of a screen:

*:SET MODAL ON
*:PJXSET
*:SET READCYCLE OFF

Then if the screen was generated from the Screen Builder, the READ would contain the clauses MODAL and CYCLE no matter what the check box settings were set to before selecting <Generate>. If the screen was generated from a project, the READ would contain MODAL no matter what the settings were set to in the project but the CYCLE setting would be set to whatever the project setting was set to.

*:BRACES

Overrides a CONFIG.FP/FPW setting of BRACES=OFF.

*:NOBRACES

Turns off the auto braces detection for GENSCRNX. By default, GENSCRNX automatically searches all snippets for any {{<expC>}} expressions to be evaluated. For screens with many objects, this could result in a few seconds overhead depending on the speed of the computer being used. Using *:NOBRACES will force GENSCRNX to only evaluate snippets if *:EVLTXT is in the Comment snippet (or Setup snippet for the header record). Overrides a CONFIG.FP/FPW setting of BRACES=ON.

*:IGNOREBRACES

Ignores all {{<expC>}} expressions. *:IGNOREBRACES cam only be declared in the Setup snippet and overrides *:BRACES and *:NOBRACES.

*:SCNOBJ

Enables the invisible button m.scnobjn above to be generated. Although it is generated by default, *:SCNOBJ can be used to override a SCNOBJ=OFF setting in the CONFIG.FP/FPW files.

*:NOSCNOBJ

Disables the invisible button m.scnobjn above to be generated. *:NOSCNOBJ can be used to override a SCNOBJ=ON setting in the CONFIG.FP/FPW files. *:NOSCNOBJ is automatic when either the #NOREAD directive exists in the Setup snippet or no GET objects exist for the screen.

*:DEFLIB <library name>

Defines library name. *:DEFLIB can be used with stand alone screens only and cannot be used with screens in a screen set.

*:INCLIB <library name>

Includes library in base object search path.

*:BASLIB <library name>

Base library objects for field name match.

*:SAVESIZE

Used with *:DEFOBJ in a library object to force the SIZE information to be retrieved fromt he library when the object is based in a screen.

*:SAVEPICT

Used with *:DEFOBJs to force the PICTURE information to be retrieved from the library when the object is based in a screen.

*:BASBEFORE

Used with *:DEFOBJ in a library object to force any inherited expressions or procedures to be inserted before rather than appended after to any

screen surface code.

*:SCXDRV1 <file>

Specifies screen driver program.

*:SCXDRV1 to *:SCXDRV8

Specifies screen driver program. The numbers 1-8 represent various driver hooks throughout GENSCRNX while the .SCX databases is being generated.

*:SPRDRV1 <file>

Specifies screen driver program.

*:SPRDRV1 to *:SPRDRV6

Specifies screen driver program. The numbers 1-6 represent various driver hooks throughout GENSCRNX while the .SPR file is being updated.

*:MEMVAR

Replaces all aliases in GET name from alias.variable to m.variable. All alias.variable names referenced in the WHEN, VALID, ERROR, MESSAGE, RANGE LO, and RANGE HIGH snippets will be replaced with m.variable.

*:NAME

The following example demonstrates how *:NAME affects FoxPro 2.5's #NAME directive.

#NAME v_show

is changed to

```
#NAME v_showd  && _DOS=.T.
#NAME v_showw  && _WINDOWS=.T
#NAME v_showm  && _MAC=.T.
#NAME v_showu  &&_UNIX=.T.
```

The above changes will occur before GENSCRN is called.
Then, a function is appended to the Cleanup snippet as follows:

```
FUNCTION V_SHOW

DO CASE
 CASE _DOS
```

```
   RETURN V_SHOWD()
 CASE _WINDOWS
   RETURN V_SHOWW()
 CASE _MAC
   RETURN V_SHOWM()
 CASE _UNIX
   RETURN V_SHOWU()
ENDCASE
RETURN .F.

FUNCTION V_SHOWM
RETURN .F.

FUNCTION V_SHOWU
RETURN .F.
```

This will result in the exact same code execution as if a CASE _DOS, CASE _WINDOWS, etc. was generated in the snippet. The only rule is that *:NAME uses only the first 9 characters of the snippet name specified. The 10th character is used for the platform character. Also, any PARAMETER statement that follows the #NAME in the snippet will be properly handled in the cross-platform function that is generated. The only rule here is that the PARAMETER statements must be identical for all platforms having the same #NAME definition.

*:NOGEN

Prevents GENSCRN from being called so that no .SPR file is generated. *:NOGEN should be used with templates since templates do not need code to be generated.

*:NOXGEN

Prevents GENSCRNX from updating .SCX database and .SPR file.

*:GENSCRNX <file>

Used to specify which program is used to generate screen code. *:GENSCRNX overrides any _GENSCRNX in the CONFIG.FP and CONFIG.FPW files. If both *:GENSCRNX and _GENSCRNX are notspecified, the default setting is GENSCRN.PRG located in FoxPro's start-up directory. *:GENSCRNX can be used to specify a modified GENSCRN needed for a particular screen rather than changing _GENSCRN before generating a screen.

*:NOCOMPSPR

Overrides COMPSPR=ON in CONFIG.FP/CONFIG.FPW.

*:NODISPSPR

Overrides DISPSPR=ON in CONFIG.FP/CONFIG.FPW.

*:NOWCLAUSES <clause list>

Removes a list of clauses from the DEFINE WINDOW command of a screen. Any list of clauses can be removed (except COLOR) by listing the name of each clause separated by a space delimiter.

Example:
Add following line in the Setup snippet to remove all FROM, TO, AT, SIZE, FONT, and STYLE clauses will be removed from the DEFINE WINDOW command:

*:NOWCLAUSES FROM TO AT SIZE FONT STYLE

To directly add any of the removed clauses, use GENSCRN's #WCLAUSES directive.

Example:
Add following lines in the Setup snippet to add a custom AT <row,col> SIZE <height,width>:

*:NOWCLAUSES AT SIZE
#WCLAUSES AT 1,1 SIZE 10,30

or

*:NOWCLAUSES AT SIZE
#WCLAUSES AT {{VPOS}},{{HPOS}} SIZE {{HEIGHT}},{{WIDTH}}

Note:
The {{<expC>}} evaluates any expression and replace its result as source code. In the above example, the field names are referencing the .SCX header record which contains the screen layout window data.

Example:
Add following lines in the Setup snippet to add a custom FONT <fontface> STYLE <fontstyle>:

*:NOWCLAUSES FONT STYLE
#WCLAUSES FONT m.myfontface STYLE m.myfontstyl

*:DRVOFF <file>

Specified in the Setup snippet to disable any driver setting that is specified in the CONFIG.FP/CONFIG.FPW. The number of *:DRVOFF directives specified in the Setup snippet is unlimited and the files included are retained for all screens in a screen set. If *:DRVOFF is specifed in the Setup snippet in a screen set, then all screens following in that screen set will inherit the *:DRVOFF for the specified driver.

Example:
If the 3D.PRG is specified in the CONFIG.FPW as _SCXDRV5="3D.PRG", for the 3D driver to be executed globally for

every screen, then specifying *:DRVOFF 3D in the Setup snippet would disable the 3D driver for that screen.

GENSCRNX creates comments in the Setup snippet as that include the the following information at compile time.

Example:
```
*          This program was preprocessed by GENSCRNX.
*--GENSCRNX 1.7
*--Screen    C:\SAMPLE\CUST2.SCX
*--Project   C:\SAMPLE\SAMPLE.PJX
*--FOXSCX    C:\FOXPRO25\FOXSCX.DBF
*--Platform  DOS
*--Time      08/25/93 20:29:46
```

COMMENT SNIPPET DIRECTIVE REFERENCE

*:DEFOBJ <object name>

Defines object name.

*:BASOBJ [<library name.]<object name>

Specify base object.

*:INSOBJ [<library name.]<object name>

Insert object from FOXSCX.DBF in place of screen object.

*:INSSCX <file>

Insert screen from template in place of screen object. Using *:INSSCX with FoxPro 2.0 and FoxPro 2.5 for DOS screens work are fully compatible for both directions.

*:FUNCTION <function name>

Automatically insert a function into the Cleanup snippet. Function needs to be written just like a typical FoxPro UDF except that *:FUNCTION is used instead of FUNCTION. GENSCRNX will automatically remove the *: from *:FUNCTION. Multiple *:FUNCTION/*:ENDFNCT text blocks can exist per Comment snippet. Also, muliptle FUNCTIONs can be definfed between *:FUNCTION and *:ENDFNCT.

Example:
```
x='This text does not get placed in the Cleanup snippet;
*:FUNCTION beep1
```

?? CHR(7)
*:ENDFNCT
x='This text does not get placed in the Cleanup snippet'

Example:
x='This text does not get placed in the Cleanup snippet'
*:FUNCTION beep1
?? CHR(7)
FUNCTION beep2
?? CHR(7)+CHR(7)
*:ENDFNCT
x='This text does not get placed in the Cleanup snippet'

Example:
x='This text does not get placed in the Cleanup snippet'
*:FUNCTION beep1
?? CHR(7)
*:ENDFNCT
x='This text does not get placed in the Cleanup snippet'
*:FUNCTION beep2
?? CHR(7)+CHR(7)
*:ENDFNCT
x='This text does not get placed in the Cleanup snippet'

*:ENDFNCT

Place at end of code that follows *:FUNCTION to mark ending of text. *:ENDTXT is now used with *:INSTXT and is not used with *:FUNCTION.

*:EVLTXT

By default, this directive is not needed. *:EVLTXT is used to force evaluation of any {{<expC>}} found in any of the snippets. This directive is only used when either *:NOBRACES is specified in the Setup snippet or BRACES=OFF is specified in the CONFIG.FP/FPW.

*:INSTXT

Insert all preceding text in place of screen object.

*:ENDTXT

Place at end of code that follows *:INSTXT to mark ending of text. *:ENDTXT is not required and is only used as a separator if non-*:INSTXT text follows the code to be inserted.

*:TRNTXT <expC1> || <expC2> [|| <expN1>] [|| <expN2>]]

Transform text of *all* memo fields. The search is *not* case-sensitive.

<expC1>

The character expression that's searched for.

<expC2>
The search character expression <expC1> is replaced by the character expression <expC2>. If <expC2> is omitted, <expC1> is replaced with
the null string.

<expN1>
The optional numeric expression <expN1> specifies which occurrence of <expC1> is the first to be replaced. For example, if <expN1> is 4, replacement begins with the fourth occurrence, counting from the left, and the first three occurrences remain unchanged. The occurrence where replacement begins defaults to 1 if <expN1> is omitted.

<expN2>
<expN2> specifies the number of occurrences of <expC1> to replace. If <expN2> is omitted, all occurrences of <expC1>, starting with the occurrence specified in <expN1>, are replaced.

Note:
*:TRNTXT is mainly used with the *:BASOBJ command for data translation of code being referenced from a library object.

*:IF <expL>

Blocks object with IF ... ENDIF statements.

*:SIZE <expC>

Replaces object SIZE clause with <expC>. <expC> can be any character expression, including variable names or FoxPro functions.

*:NOSIZE

Removes SIZE clause from object. *:NOSIZE is ignored for EDIT objects.

*:DEFAULT <expC>

Replaces object DEFAULT clause with <expC>. <expC> can be any character expression, including variable names or FoxPro functions. Push buttons, Radio buttons, and Check boxes use the value of <expC>. Lists, invisible buttons, and spinners cannot use the *:DEFAULT directive. All other objects use <expC> with a direct replacement. If a character default is desired, be sure to include the quotes in the expression. If the current object's color is set to default, then a COLOR SCHEME <expN> or COLOR <color pair list> may be included in <expC>.

*:PICTURE <expC>

Replaces object PICTURE clause with <expC>. <expC> can be any character expression, including variable names or FoxPro functions. *:PICTURE can also be used to create multi-state .BMP/.ICO pictures for check boxes, radio buttons, and push buttons (Windows platform only).

Example:
To force a data driven PICTURE clause for a GET object using a variable called m.mypict, place the following in the Comment snippet:

*:PICTURE m.mypict

Example:
To force a tri-state picture check box:
*:PICTURE erase01.ico,erase02.ico,clear.ico

Example:
To force a dual-state picture check box without respecifying the currently set off mode picture:
*:PICTURE ,fax2.ico

Note:
If the first .BMP/.ICO file name in the comma separated list is left out as in the above example, the picture current set by the screen builder will be used as the off mode picture. This way changing the off mode picture can be done in the screen builder without having to also change in the Comment snippet.

*:REFRESH

Replaces object REFRESH clause with .T.. *:REFRESH will override the refresh setting for a SAY object and can also be used to allow a picture to be refreshed in the Read Level Show using either SHOW GETS or SHOW GETS OFF.

Note:
Using both *:REFRESH and *:PICTURE <variable name> with a picture from file object can allow picture fields to be refreshed at runtime. If a transparent picture is used, any updated picture will overwrite (not erase) the previous picture. Using an opaque picture will overwrite and erase the previous picture but will usually have a white background when using a gray window background. Another option if a transparent picture is desired and when using the 3D driver for GENSCRNX (version 1.7 or later) is to draw a box around the picture and use a *:3D <bevel width> BOX REFRESH directive to allow the 3D box to be refreshed in the Read Level Show using either SHOW GETS or SHOW GETS OFF.

*:CLICK <function>

Adds invisible button with a WHEN snippet that calls the mouse click function specified. The () after the function name are only required if parameters are passed. In the Windows platform, *:CLICK supports text, box, and picture objects while in the MS-DOS platform, *:CLICK supports

text and box objects.

Example:
To have a function called myfnct() exectuted from a mouse click on the object, place the following in the Comment snippet:

*:CLICK myfnct

*:DELETE

Delete screen object at compile time. Use *:DELETE for objects that need to appear while using the Screen Builder but not in the .SPR file at run-time.

*:DELOBJ

Delete screen object at compile time after preprocessing is complete. Use *:DELOBJ for objects that need to reside in the .SCX database during preprocessing but not in the .SPR file at run-time.

PROCEDURE SNIPPET DIRECTIVE REFERENCE

#:INSERT <file>

Screen generator directive inserts the contents of <file> into generated screen code. Not only does GENSCRNX support the #INSERT directive for FoxPro 2.0, but the #:INSERT directive performs the same operation as FoxPro 2.5's #INSERT except it is much faster when inserting large files.

FILE SIZE	GENSCRN #INSERT	GENSCRNX #:INSERT
2K	3.215	2.938
135K	178.717	3.475
330K	970.478	6.630

Time is in seconds using 486-50DX

When using GENSCRNX, use #:INSERT instead of #INSERT for better performance.

#:INSERTTOP <file>

Inserts file at top of .SPR code before DO CASE of cross-platform block. If #:INSERTTOP <file> appears more than once due to cross-platform snippets containing the same code, the <file> will only be inserted into the .SPR once. This allows header files containing #DEFINE directives

to be inserted once per .SPR file instead of one per platform inside the DO CASE block.

SNIPPET COMMAND REFERENCE

{{<expC>}}

Text surrounded by double braces performs the EVALUATION of <expC> at compile time and returns the value in string form. {{<expC>}} is replaced with the string of EVALUATE(<expC>). <expC> can be any type (character, numeric, date, logical, etc.) and {{<expC>}} will always return the result in character form.

Example:
If the following command was in the Setup snippet and assuming the current date was 06/01/93:

```
WAIT '{{DATE()}}' WINDOW NOWAIT
```

the following code would be placed in the .SPR:

```
WAIT '06/01/93' WINDOW NOWAIT
```

If the following command was in the Valid snippet:

```
DEFINE POPUP pop_test FROM {{VPOS+HEIGHT}},{{HPOS-1}};
 TO {{VPOS+HEIGHT+7}},{{HPOS+WIDTH}};
 PROMPT FIELD items.item
```

and VPOS=5, HPOS=10, WIDTH=8, HEIGHT=1 in the .SCX database, then the following code would result in the Valid snippet of that object in the .SPR:

```
DEFINE POPUP pop_test FROM 6,9;
 TO 13,18;
 PROMPT FIELD items.item
```

{{&.<expC>}}

Text surrounded by double braces with a &. immediately after the open braces performs the macro substitution of <expC> at compile time and returns a null value in string form. {{<expC>}} is replaced with a null string. <expC> can be any FoxPro command that can be executed within a macro substitution string.

Example:
If the following command was in the Setup snippet and assuming the current date was 06/01/93:

```
{{& WAIT '{{DATE()}}' WINDOW NOWAIT}}
```

the following WAIT window would appear at compile time of the screen:

06/01/93

If the following command was in the Setup snippet:

{{&.DO MYPROG}}

then a program called MYPROG would be executed as a subroutine at compile time of the screen. If the program was to return a character string for code insertion, then {{MYPROG()}} would have been used.

{{< <file> }}
Insert a file at compile time. The < that follows the open braces is the command that evaluates the contents of a file and inserts the file at that location. {{< <file> }} can be included in the Comment snippet and the file inserted can contain other GENSCRNX directives and also may contain any {{<expC>}} expressions to be evaluated.

Example:
If the following command was in the Comment snippet of a GET object:

{{<PSWDCHK.PRG}}

and PSWDCHK.PRG contained the following lines:

*:IF m.password>=5

then the resulting code in the .SPR would be:

```
IF m.password>=5
   @ row,col GET expr
ENDIF
```

{{@ <expC> }}

Retrieve a directive at compile time. The @ that follows the open braces is the command that performs a wordsearch() operation in the Comment or Setup snippet searching for the directive specified by <expC>. {{@ <expC> }} can be included in the Comment snippet and the file inserted can contain other GENSCRNX directives and also may contain any {{<expC>}} expressions to be evaluated.

Example:
If the following command was in the Valid snippet of a GET object that had a *:IF m.p>5 in the Comment snippet:

WAIT 'IF: {{@*:IF}}' WINDOW NOWAIT

then the resulting code in the .SPR would be:

WAIT 'IF: m.p>5' WINDOW NOWAIT

Example:

If the following command was in the Comment snippet of an object:

{{Button1::@*:IF}}

then the object would use the *:IF directive specifed in an object containing *:DEFOBJ Button1 in the Comment snippet.

Example:
If the following command was in the Setup snippet of an object:

*:SCXDRV5 3D
:ALL3D {{MAIN.All3D_Setting::@:ALL3D}}

while a library MAIN contains an object called All3D_Setting that contained *:ALL3D 4 in the Setup snippet then the 3D driver would use a shadow of 4 pixels as the default for all 3D objects. The Setup snippet would result in the following:

*:SCXDRV5 3D
*:ALL3D 4

The above technique can be used to control default settings in a globally to have muliptle screens use the same directive settings.

*:METHOD

Place at start of code to mark beginning of method code.

*:ENDMTHD

Place at end of the code that follows *:METHOD to mark ending of text.

{{ <expC1> :: [<expC2>] [:: <expC3>] }}

Insert code from a screen or library object. <expC1> is the library.object name just as in *:DEFOBJ, *:BASOBJ, etc. Note if the library name is not included, the object is searched for specified by the *:INCLIB and *:BASLIB directives in the Setup snippet. Also, if a matching object is defined via the *:DEFOBJ directive on the surface screen, that object will have proirity over any matching library objects. <expC2> is the string to be evaluated. After the .SCX record is matched, any string can be evaluated (ex. 'VALID' to return the VALID snippet). If <expC2> is null, the COMMENT contents will be returned. <expC3> is the option method name. If <expC3> is included, the text block specified by the matching method defined by *:METHOD <name> ... *:ENDMTHD is returned.

Example:
Suppose the following code is placed in the WHEN snippet of an object used for entering a Phone number and *:DEFOBJ Get_Phone was placed in the Comment snippet to label the object:

*:METHOD Check_MDOWN
IF .NOT.MDOWN()

```
 RETURN .F.
ENDIF
*:ENDMTHD
*:METHOD Check_EditMode
IF .NOT.m.editmode
 RETURN .F.
ENDIF
*:ENDMTHD
```

Suppose you wanted the WHEN snippet of another object to contain the code used to check for the mouse being pressed but not for the edit mode status. Instead of using the copy and paste method, place the following line of code in the WHEN snippet:

{{Get_Phone::WHEN::Check_MDOWN}}

then the resulting code in the WHEN snippet would be:

```
IF .NOT.MDOWN()
 RETURN .F.
ENDIF
```

Suppose you wanted the WHEN snippet of another object to contain all code used in the Phone object's WHEN snippet but wanted a beep to occur before the check. Instead of using the copy and paste method, place the following line of code in the WHEN snippet:

```
?? CHR(7)
{{Get_Phone::WHEN}}
```

then the resulting code in the WHEN snippet would be:

```
?? CHR(7)
IF .NOT.MDOWN()
 RETURN .F.
ENDIF
IF .NOT.m.editmode
 RETURN .F.
ENDIF
```

Note:
The expression {{Get_Phone::}} is identical to {{Get_Phone::COMMENT}} since the Comment snippet is the default.

Note:
Complex expressions can be used like the following:
{{Get_Phone::WHEN+VALID}} which would insert both the WHEN and VALID snippets of the Get_Phone object.

DRIVER INFORMATION

Driver programs are specified either in the CONFIG.FP/CONFIG.FPW

files by defining:

_SCXDRV3="<pathname>\[<file>]".

Driver programs can also be specified a screen Setup snippet by defining:

*:SCXDRV3 <pathname>\[<file>].

*:SCXDRV1 is used before any GENSCRNX compilation. It can be used as a #INCLUDE to add any GENSCRN or GENSCRNX directives. Another method of obtaining a #INCLUDE type function is the use the braces ({{<expC>}}) when <expC> contains an external function. The character string returned from the function will replace the {{<expC>}} directly. For example, if the Setup snippet contained the following line:

{{inc_test()}}

and the external function inc_test() return a character string of #NOREAD, then the {{inc_test()}} line would be directly replaced by the #NOREAD command. Also, the returned character string may contain a carriage return and line feeds (CHR(13)+CHR(10)) to separate lines when multiple lines are needed for insertion. Refer to the {{<expC>}} definition supplied with GENSCRNX for further information.
Notes:
If the <file> parameter of a driver directive does not include a file extension, the following extensions are checked in this order:
.EXE, .APP, .PRG, .FXP

The n in SCXDRVn represents the hook number from GENSCRNX. GENSCRNX has 8 different places during the compiling loop that can call out to drivers. The most common one to use is #3. You can have infinite drivers for #3:

Example:
*:SCXDRV3 <driver1>
*:SCXDRV3 <driver2>

The order they are listed is the order they are called. Hook #1 is before compilation (like #INCLUDE), hook #2 is the first in the first compile loop, hook #3 is the first in each compile loop, #7 is after preprocessing (except *:FUNCTION and the insertion of _ScnObjn) is complete, and #8 is after all preprocessing is complete.

Important:
Only one driver can be specified in the CONFIG.FP/CONFIG.FPW files. If more than one driver is specified in the Setup snippet, the drivers are called in the order they are listed. Drivers specified in the CONFIG.FP/CONFIG.FPW are called before the drivers specified in the Setup snippet.

USING GENSCRNX AS TRANSPORTX

In the CONFIG.FP/FPW file, you can place the following:

_TRANSPRT="<path>GENSCRNX.PRG"
_TRNDRV1="<path><prg 1>"
_TRNDRV2="<path><prg 2>"

What happens here is that whenever the FoxPro calls the transporter, the following occurs:

1) GENSCRNX gets called.
2) If prg 1 is defined as above, prg 1 is called.
3) Based on the return value of prg 1, GENSCRNX will either return an open as is, cancel, or call TRANSPRT.PRG.
4) Upon return from TRANSPRT.PRG, if prg 2 is defined as above, prg 2 is called.

Notes:

<prg 1> can be used as a custom control program to determine if the transporter needs to be called or any preprocessing needs to occur to the .SCX before TRANSPRT.PRG is called.

<prg 2> can be used to updated the .SCX after TRANSPRT.PRG is complete to override any unwanted defaults such as fonts, row/column, screen color, or .SCX header information.

ADDITIONAL INFORMATION

For performance optimization, GENSCRNX only pre and post processes a screen if the *: or {{ characters exist somewhere in either the Setup snippet or at least one of the Comment snippets. The first screen a screen set of more than one screen must have either a GENSCRNX directive or at least a simple *: in the Setup snippet of the first screen for GENSCRNX to properly preprocess the screen set.

All *: directives used for GENSCRNX must be specified starting in column one of the snippet. Do not indent the *: directives with spaces or tabs.

GENSCRNX automatically creates two null invisible button at row,col 0,0 as the first and last GET in the Screen Layout. Each screen of a screen set will have two null invisible buttons with the name corresponding to the screen set. Since the invisible button's WHEN is set to .F., the objects are null objects and have no effect on the generated screen. The purpose of this is to allow generic reference to the first or last GET object in any screen of a screen set. For example in a screen set with one screen, the first GET would be m.scnobj1 and the last GET would be m.scnend1.

Example:
_CUROBJ=OBJNUM('m.scnobj1')

Example: The first GET of the first screen of a screen set would have an invisible button at 0,0 called m.scnobj1 while the second screen of a screen set would have an invisible button called m.scnobj2.

When using the Standard version of FoxPro for MS-DOS, the .SPR file size must be less than 64K.

If a public variable called _GENSCRNX is set to OFF, GENSCRNX with pass the .SCX directly to GENSCRN and all GENSCRNX directives and commands will be ignored. GENSCRNX can also be specified in the CONFIG.FP/CONFIG.FPW files and changed without re-entering FoxPro.

If a public variable called _PLATONLY is set to ON, GENSCRNX and GENSCRN will only generate code for the current running platform. PLATONLY can also be specified in the CONFIG.FP/CONFIG.FPW files and changed without re-entering FoxPro. _PLATONLY='ON' is useful during development when cross-platform code generation is not required for screens until development is complete.

COPYRIGHT NOTICE

COMMENTS/SUGGESTIONS/PROBLEMS/QUESTIONS

Please use CompuServe's FoxForum (section 3rd Party Products) directed to:

Ken Levy 76350,2610
