

PHONE_FORMAT Conversion Source Code

The source for the **PHONE_FORMAT** conversion is found in the **SYSPROCS** table, in the **SYSPROG** account. In addition to illustrating customized input and output conversions the code shows how to create a customized error message if the conversion fails. See the **DisplayError** internal subroutine, which sets **Status() = 3** and uses the **msg()** function to display a customized error message.. The source is below:

```
compile SUBROUTINE PHONE_FORMAT( charstr CONV, charstr ANS, charstr BRANCH, charstr RETURN_DATA)
*
*   PHONE_FORMAT is an example of a developer's custom prompt formatting
*   routine using the square brackets call.
*
*   It should be placed in square brackets, like this:
*
*           [PHONE_FORMAT]
*
* This subroutine should be used as the first and only "Validation Pattern" in
* an OpenInsight control. Placed in "Conversion Pattern", it properly formats any
* reasonable string of numbers into a consistent US telephone number format.
*
* mtr 5-29-01  Changed @upper.case to @lower.case conversion
* mtr 3-18-02  Added '.' as a valid delimiter.
!
begin condition
pre:
post:
end condition
* Subroutine declarations
$insert msg_equates
declare function msg
* Local Equates
* The STATUS() variable is used to indicate the error condition of the
* pattern. They are:
EQU VALID$          TO 0      ;* Successful
EQU INVALID_MSG$    TO 1      ;* Bad Data      -   Print error message window
EQU INVALID_CONV$   TO 2      ;* Bad Conversion -   "           "
EQU INVALID_NOMSG$  TO 3      ;* Bad but do not print the error message window
EQU THREEDGRAY$     TO 192
* Begin Conversion
*
RETURN_DATA = ""
IF ANS NE "" THEN
    TEL = ANS
    ANS = ""
    STATUS() = VALID$

    *DFLT_AREA_CODE = ""
    * PHONE_FORMAT can support a default area code. To assign a default
    * simply set the variable DFLT_AREA_CODE. In this example it is set to
    * null.
    *CONVERT " -()" TO "" IN DFLT_AREA_CODE
    *IF NUM( DFLT_AREA_CODE ) ELSE DFLT_AREA_CODE = ""

    CONVERT " -()." TO "" IN TEL
    * mtr
    CONVERT @LOWER.CASE TO @UPPER.CASE IN TEL
    CONVERT "ABCDEFGHJKLMNOPQRSTUVWXYZ" TO "222333444555666777888999Z" IN TEL

    IF NUM( TEL ) THEN
        LENGTH = LEN( TEL )
        * Case statement to validate all possible types of phone numbers. If
        * a new format is required simply add another case.
        * The fall-through (CASE 1) traps invalid conversions.
        BEGIN CASE
            CASE LENGTH = 10
                IF CONV EQ "OCONV" THEN
                    RETURN_DATA = FMT( TEL, "L(###) ###-####")
                END ELSE
                    RETURN_DATA = TEL
                END
            CASE LENGTH EQ 7
```

```

        IF CONV EQ "OCONV" THEN
            RETURN_DATA = FMT( TEL, "L###-####")
        END ELSE
            RETURN_DATA = TEL
        END
    CASE 1
        IF CONV = "ICONV" THEN
            gosub DisplayError
        END
        STATUS() = INVALID_NOMSG$
    END CASE
END ELSE
    IF CONV = "ICONV" THEN
        gosub DisplayError
    END
    STATUS() = INVALID_NOMSG$
END
END
RETURN
*
DisplayError:
    msgrec          = ""
    msgrec<MCAPTION$> = "Data Validation Error"
    msgrec<MTEXT$>   = TEL : " is not a valid phone number. Please enter a seven or ten digit number in any
format."
    msgrec<MBKCOLOR$> = THREEEDGRAY$:@VM:THREEEDGRAY$:@VM:THREEEDGRAY$
    msgrec<MJUST$>    = 'L'
    result = msg( "", msgrec)
return

```