

[print](#) | [close](#)

APIs by Example: User Function Registration APIs, Part 3

[System iNetwork Programming Tips Newsletter](#)

[Carsten Flensburg](#)

Carsten Flensburg

Thu, 09/15/2005 (All day)

Here's the third and final part of the User Function Registration APIs by Example. This week, I put the User Function commands from parts one and two into action. I use the new CHGUSRSTS (Change User Status) command as the basis for this presentation.

As I discussed in part two, the User Function APIs offer a unique method of implementing application-level access control. The User Function facility is primarily targeted at providing and controlling access to resources and functions otherwise not accessible to users that legitimately require this access as part of their job functions.

To demonstrate such a scenario, I'm using the CHGUSRSTS command, which lets authorized help desk personnel enable a disabled user profile and also reset the profile's password. I have created two access levels by means of the User Function facility. One gives access to reset and enable all user profiles. The other gives access only to user profiles that have a user class of *USER and no inheritance of authority from related and more powerful user profiles.

The user profile functions that the CHGUSRSTS command performs usually require both *SECADM and *ALLOBJ special authority, so for the command to work properly, I have made the CPP adopt its owner's authority and changed the program object owner to QSECOFR. I then have the CPP initially check the User Function register to ensure that only registered user profiles can run the CHGUSRSTS command.

If the user profile is registered against the most privileged user function, it can change all user profiles; otherwise, only user class *USER profiles are available to change. Unauthorized access of any kind is logged to the system audit journal QAUDJRN.

The example setup program CBX943S runs the following CL commands:

```
AddFcnReg  FcnId( CBX_HELPDESK_ADMIN )
            FcnType( *PRODUCT )
            FcnName( 'Help desk administration' )
            FcnDesc( 'Help desk function access' )
AddFcnReg  FcnId( CBX_HELPDESK_TOOLS )
            FcnPrdId( CBX_HELPDESK_ADMIN )
            FcnType( *GROUP )
            FcnName( 'Help desk tools' )
            FcnDesc( 'Utilities for help desk personnel' )
```

The preceding commands register a product and a group type User Function to use when registering the two administration type User Functions:

```

AddFcnReg  FcnId( CBX_USRPRF_ADM_USER )
           FcnGrpId( CBX_HELPDESK_TOOLS )
           FcnPrdId( CBX_HELPDESK_ADMIN )
           AllObjAut( *NOTUSED )
           Default( *DENIED )
           FcnType( *ADMIN )
           FcnName( 'User profile administration *USER' )
           FcnDesc( 'User class *USER administration' )

AddFcnReg  FcnId( CBX_USRPRF_ADM_SECOFR )
           FcnGrpId( CBX_HELPDESK_TOOLS )
           FcnPrdId( CBX_HELPDESK_ADMIN )
           AllObjAut( *NOTUSED )
           Default( *DENIED )
           FcnType( *ADMIN )
           FcnName( 'User profile administration *SECOFR' )
           FcnDesc( 'User class *SECOFR administration' )

```

The preceding commands register the two levels of access that I want to implement for the CHGUSRSTS command. One gives access to all user profiles, and the other gives access only to user class *USER user profiles.

The User Function default access of *DENIED ensures that only user profiles registered to the preceding User Functions are granted access. The ALLOBJAUT setting of *NOTUSED indicates that user profiles having *ALLOBJ special authority cannot use this authority to gain access and need to be explicitly registered to the User Function to be authorized.

```

RtvJobA    CurUser( &CurUsr )
ChgUsrFcnU FcnId( CBX_USRPRF_ADM_SECOFR )
           UsrPrf( &CurUsr )
           Usage( *ALLOWED )

```

The user profile running the setup program is eventually allowed CHGUSRSTS command access to all user profiles. If you prefer, you could register the user profile to the alternate user function, only allowing access to user class *USER user profiles. To do this, change the function ID appropriately.

Remember that you can also register a group profile to a User Function to allow all group member user profiles access to that function. This can greatly simplify the usage administration.

When you have finished testing the foregoing example, you can run the CBX143R CL program to remove the setup User Function registrations, if you wish.

```

                                Change User Status (CHGUSRSTS)
Type choices, press Enter.
User profile . . . . . Name
Reset password . . . . . *NO *NO, *YES
Enable user profile . . . . . *NO *NO, *YES

```

The help panel group of the CHGUSRSTS command documents the command and its parameters in more detail.

The CHGUSRSTS CPP performs the User Function check. Please note that someone could bypass this check by placing his own RTVFCNUSG (Retrieve Function Usage) command higher in his library list. If this is a concern on your system, you should qualify the RTVFCNUSG command, specifying the library that contains the correct RTVFCNUSG command:

```
library-name/RtvFcnUsg  FcnId( CBX_USRPRF_ADM_USER )      +
                        UsrPrf( &CurUsr )                +
                        UsgInd( &UsgInd )
```

Likewise, you should change the RTVFCNUSG command to point directly to the correct CPP:

```
ChgCmd Cmd( RTVFCNUSG )
      Pgm( library-name/CBX142 )
```

The RTVFCNUSG command first appeared in part two of this series. You can read that article at the following link:

<http://www2.systeminetwork.com/article.cfm?id=51418>

Run the following commands to create the CHGUSRSTS CPP with the required ownership and attributes:

```
CrtClPgm  Pgm( CBX143 )
          SrcFile( QCLSRC )
          SrcMbr( *PGM )
          Option( *NOSRCDBG )
          UsrPrf( *OWNER )
          Log( *NO )
          AlwRtvSrc( *NO )
ChgPgm    Pgm( CBX143 )
          RmvObs( *ALL )
ChgObjOwn Obj( CBX143 )
          ObjType( *PGM )
          NewOwn( QSECOFR )
```

The CHGUSRSTS command includes the following sources:

CBX143 -- CPP

CBX143X -- Command definition source member

CBX143H -- Command help text panel group

The example setup and removal programs:

CBX143S -- Program to set up example

CBX143R -- Program to remove example setup

Compilation instructions are in the source headers.

Important Note: I have discovered an error in the validity-checking program of the ADDFCNREG (Add Function Registration) command provided with part one of this article. I apologize for any inconvenience experienced due to this error. A revised version of the CBX1401V program source is included with this article. Before running the example setup program, please replace your version of the CBX1401V program with the current one.

This article demonstrates the following APIs:

Check Function Usage (QsyCheckUserFunctionUsage) API:

<http://as400bks.rochester.ibm.com/iserics/v5r2/ic2924/info/apis/QSYCKUFU.htm>

Retrieve User Information (QSYRUSRI) API:

<http://as400bks.rochester.ibm.com/iserics/v5r2/ic2924/info/apis/qsyrusri.htm>

Move Program Message (QMhMOVPM) API:

<http://as400bks.rochester.ibm.com/iserics/v5r2/ic2924/info/apis/qmhmovpm.htm>

Resend Escape Message (QMhRSNEM) API:

<http://as400bks.rochester.ibm.com/iserics/v5r2/ic2924/info/apis/QMhRSNEM.htm>

You can retrieve the source code for this API example from the following link:

http://www.pentontech.com/IBMContent/Documents/article/51525_36_UserFuncReg3.zip

Source URL: <http://iprodeveloper.com/rpg-programming/apis-example-user-function-registration-apis-part-3>