The Fox Hunt

Keeping track of your users

By Whil Hentzen

Users. Our lives as developers would be a lot easier if we didn't have to deal with users, wouldn't it? But until we reach that state of nirvana, we'll have to do something about them.

Last month, we introduced the idea of selectively providing menu choices based on the permission of a user. We kept those permission levels in a "user table." This month, we're going to expand on this idea of a user table, and introduce a few ancillary functions that you might find useful.

Beyond the user's name and permission level, what might we keep in a user table? Most obvious, probably, is their password. The issues that arise when dealing with passwords, such as encryption techniques, display masking, and validation, could constitute an entire series of article by themselves. For now, let's make note that we should only store the encrypted password in the user table.

Another field often used is an activity flag - set to true for active users and set to false for inactive users. The logon program would then only allow users with a True flag to access the system. A second similar field would be a "is currently logged in" flag, but there's a better way to handle this - we'll cover this later in this article. And then, depending on your needs, you could store various user preferences with each user. For example, color sets, default printers, and "confirm on delete" and "exit without confirmation" flags are choices each user might want to make themselves. I've even seen systems where the time interval between a single and double mouse-click is user-configurable.

If you're going to go this far, consider making your systems a bit more flexible. In each situation where the user can choose their own preference, provide them the ability to use that preference just for the current session, or to save the choice as their new default. Naturally, you'd want to give them the ability to view their settings, and to give the system administrator the ability to set the defaults for new users, and to change the default for all users at one time.

Now back to the problem of recording users who are currently logged in. Initially, the idea of setting a "Is Logged In" flag to true when the user has logged in, and then changing that flag to false when they logout, is appealing. However, it doesn't take long for reality to set in. As soon as the program terminates abnormally - whether through a network crash, an impatient user reaching for the Ctrl-Alt-Del combination, or (gasp!) a programming error - the user's record now contains an invalid "true" flag. And if you're restricting users to a single logon by examining that flag before allowing them access to the system, you'll quickly lock out every user.

A better way is to lock the user's record when they logon, and release the lock when they log off. In order to determine who is on the system, scan through the user table, attempting to lock each record. If the lock fails, you'll know that the record was already locked - which means the user is currently logged on. The following code shows how this might be done.

```
*
* m.gcNameUser is the name of the currently logged on user - the
* person running this function
*
declare aCurUsers[1]
aCurUsers[1] = m.gcNameUser
select SYS_USER
go top
scan
 *
 * try to lock file for each user
 * we're going to skip the record for the current user
 *
 if !(cNameUser = m.gcNameUser)
  if rlock()
   *
   * since we were able to lock this user, the record wasn't already locked
   * this means they weren't already on - go on to next one
   * let's make sure to unlock this one!
   *
   unlock
  else
   *
   * we weren't able to lock, which means there was already a lock on this user
```

```
   * thus, they are already logged in!
   *
   declare aCurUsers[alen(aCurUsers)+1]
   aCurUsers[alen(aCurUsers)] = SYS_USER.cNameUser
  endif rlock()
 endif !(cNameUser = m.gcNameUser)
endscan
```

Once this routine is complete, use the array aCurUsers to populate a listbox in a screen in order to show who is logged on. In the event of an abnormal termination, the operating system will release the lock for you. Your users will thank you and you'll receive fewer "3 AM" phone calls as a result.