BECKHOFF New Automation Technology

Manual | EN

User Manager

TwinCAT 2 | System

2022-07-19 | Version: 1.0

Table of contents

1 Foreword 5
1.1 Notes on the documentation 5
1.2 Safety instructions 6
1.3 Notes on information security 7
2 Common 8
3 Local Security Policies 9
4 Reference 10
4.1 TcUserManager (CoClass) 10
4.2 ITcUserManager 10
4.2.1 ImpersonateUser 11
4.2.2 ImpersonateUserDlg 11
4.2.3 RemoveUserAccount 12
4.2.4 RevertToSelf 12
4.2.5 CreateProcessAsUser 13
4.2.6 CreateProcessAsUserDlg 13
4.2.7 CreateUserAccount 14
4.2.8 CreateUserAccountDlg 15
4.2.9 EnumLocalUsers 15
4.2.10 EnumLocalGroups 15
4.2.11 UserEnumLocalGroups 16
4.2.12 UserIsAdmin 17
4.2.13 UserIsMemberOf 17
4.2.14 UserName 18

1 Foreword

1.1 Notes on the documentation

This description is only intended for the use of trained specialists in control and automation engineering who are familiar with applicable national standards.
It is essential that the documentation and the following notes and explanations are followed when installing and commissioning the components.
It is the duty of the technical personnel to use the documentation published at the respective time of each installation and commissioning.

The responsible staff must ensure that the application or use of the products described satisfy all the requirements for safety, including all the relevant laws, regulations, guidelines and standards.

Disclaimer

The documentation has been prepared with care. The products described are, however, constantly under development.
We reserve the right to revise and change the documentation at any time and without prior announcement.
No claims for the modification of products that have already been supplied may be made on the basis of the data, diagrams and descriptions in this documentation.

Trademarks

Beckhoff®, TwinCAT®, TwinCAT/BSD®, TC/BSD®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, Safety over EtherCAT®, TwinSAFE®, XFC®, XTS® and XPlanar® are registered trademarks of and licensed by Beckhoff Automation GmbH.
Other designations used in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owners.

Patent Pending

The EtherCAT Technology is covered, including but not limited to the following patent applications and patents:
EP1590927, EP1789857, EP1456722, EP2137893, DE102015105702
with corresponding applications or registrations in various other countries.

EtherCAT® is a registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany

Copyright

1.2 Safety instructions

Safety regulations

Please note the following safety instructions and explanations!
Product-specific safety instructions can be found on following pages or in the areas mounting, wiring, commissioning etc.

Exclusion of liability

All the components are supplied in particular hardware and software configurations appropriate for the application. Modifications to hardware or software configurations other than those described in the documentation are not permitted, and nullify the liability of Beckhoff Automation GmbH & Co. KG.

Personnel qualification

This description is only intended for trained specialists in control, automation and drive engineering who are familiar with the applicable national standards.

Description of symbols

In this documentation the following symbols are used with an accompanying safety instruction or note. The safety instructions must be read carefully and followed without fail!

⚠ DANGER

Serious risk of injury!

Failure to follow the safety instructions associated with this symbol directly endangers the life and health of persons.

⚠ WARNING

Risk of injury!

Failure to follow the safety instructions associated with this symbol endangers the life and health of persons.

⚠ CAUTION

Personal injuries!

Failure to follow the safety instructions associated with this symbol can lead to injuries to persons.

NOTE

Damage to the environment or devices

Failure to follow the instructions associated with this symbol can lead to damage to the environment or equipment.

Tip or pointer

This symbol indicates information that contributes to better understanding.

1.3 Notes on information security

The products of Beckhoff Automation GmbH & Co. KG (Beckhoff), insofar as they can be accessed online, are equipped with security functions that support the secure operation of plants, systems, machines and networks. Despite the security functions, the creation, implementation and constant updating of a holistic security concept for the operation are necessary to protect the respective plant, system, machine and networks against cyber threats. The products sold by Beckhoff are only part of the overall security concept. The customer is responsible for preventing unauthorized access by third parties to its equipment, systems, machines and networks. The latter should be connected to the corporate network or the Internet only if appropriate protective measures have been set up.

In addition, the recommendations from Beckhoff regarding appropriate protective measures should be observed. Further information regarding information security and industrial security can be found in our https://www.beckhoff.com/secguide.

Beckhoff products and solutions undergo continuous further development. This also applies to security functions. In light of this continuous further development, Beckhoff expressly recommends that the products are kept up to date at all times and that updates are installed for the products once they have been made available. Using outdated or unsupported product versions can increase the risk of cyber threats.

To stay informed about information security for Beckhoff products, subscribe to the RSS feed at https://www.beckhoff.com/secinfo.

2 Common

TwinCAT is a general automation system (or suite of automation modules) whose components can be added or removed in various configurations. Since TwinCAT is constructed in a very modular fashion and the modules are independent of each other, the User Manager can be implemented independently of the other modules. The PC operating system (Windows NT/2000, Windows CE) is also a part of the TwinCAT automation system and must therefore be included in the user administration.

For the deployment of TwinCAT under Windows NT*/2000 the user administration is based on the security system of Windows NT, i.e. an NT administrator is also a TwinCAT administrator and users of an NT domain are also recognized in TwinCAT. Thus, the total system can (if necessary) be administered using Windows NT tools. The security system of Windows NT can revoke or award access rights for each object of the operating system (partition, directory, file, network, …). Access to the NT security system is unfortunately relatively opaque, since the API is distributed over several parts of the operating system. Thus, some functions are to be found in the "Lan Manager" API whereas others are in the actual security API. The TwinCAT User Manager collects the necessary functions in a COM object "TcUserManager [▶ 10]". TcUserManager does not of course implement all functions of the NT security system, but only those functions that appear useful in a machine control system. If additional Windows NT functionality should be required in special circumstances, the appropriate Windows NT APIs can also be accessed directly, since Windows NT utilizes the same user data bank as the TwinCAT User Manager.

Deployment

Ideally the machine control system starts up without a user having to log in. As a rule, a standard user is automatically logged in and the operating interface is started up through TwinCAT. The standard user should only have restricted user privileges (those of a simple Window NT user) since he or she can otherwise execute alterations to the system (e.g. installation of drivers) that can lead to impairments of machine functions. If the execution of privileged functions (deletion of files, alteration of production sequences, alteration of the CNC program) should be necessary when the machine is operating under production conditions, a complete log out and relog into Windows NT must of course be avoided. Instead the operating interface process (in fact this is only one of the threads) can assume the status of a user with higher privileges (ImpersonateUser [▶ 11]), or an additional process such as e.g. TwinCAT System Manager can be started from the operating interface using CreateProcessAsUser [▶ 13] with the user account of an administrator. Each user in TwinCAT can belong to several user groups. In order to protect particular functions of the operating interface, access can be checked using either UserIsMemberOf [▶ 17] or UserIsAdmin [▶ 17]. At the present time the method UserIsMemberOf checks the user group on the basis of name, which means that problems associated with the language of the country can occur.In order to avoid these problems and moreover to have uniform user groups in the TwinCAT system, standard user groups are defined for TwinCAT.

- **TcUsers**
- **TcPowerUsers**
- **TcAdministrators**

These user groups are added as global Windows NT user groups through the TwinCAT installation to the Windows NT data base (this is yet to be implemented).

* Only TwinCAT 2.9 runs under Windows NT.

3 Local Security Policies

The local security plicies contain security related parameters of the local system.

Particular security settings are required in order to use the process within other applications [▶ 13] e.g. the TwinCAT User Manager.

Open the "Local Securty Settings" with the command **secpol.msc /s** to configure the required user rights.

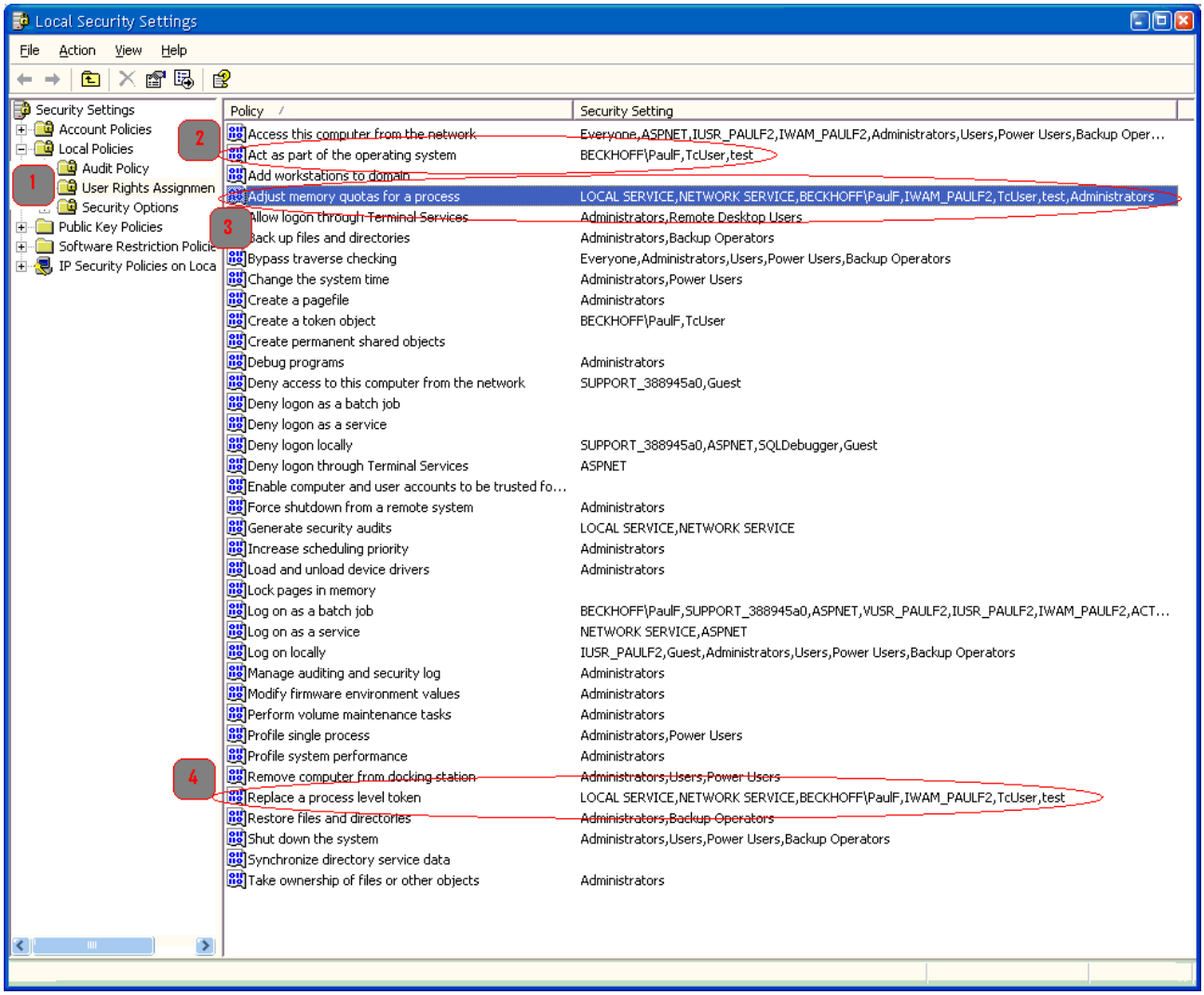

Fig. 1: "Local Security Settings" of a Windows XP Professional operating system

The "Policies" categories are listed on the left side. Use "User Rights Assignment" (assigning user rights, step 1) to the configre the required user rights to run a process with different user credentials.

To specify the user on the rights double click on them:

- **2** act as part of the operating system
- **3** adjust memory quotas for a process
- **4** Replace a process level token

These policies should contain the users who should be impersonated and the user who called the application

4 Reference

4.1 TcUserManager (CoClass)

The TwinCAT User Manager is implemented in the class TcUserManager. TcUserManager stacks directly onto the Windows NT user administration and can therefore offer all the security features of Windows NT. An important feature of the Windows NT security system is the option of either providing or withholding privileges for each operating system object (e.g. hard disk, file, network, …). Furthermore the users, user groups and their privileges can be propagated across the network.

The default interface of **TcUserManager** is called ITcUserManager [▶ 10], **TcUserManager** is implemented as "In Process Server" in the DLL "TCATUserMan.dll". For reasons of compatibility part of the functionality can also be attained through exported functions of the DLL. New applications should however use the COM interface ITcUserManager [▶ 10].

4.2 ITcUserManager

The interface ITcUserManager allows access to the TwinCAT User Manager. The TwinCAT User Manager is based on the Windows NT security system. An important feature of the Windows NT security system is the option of either providing or withholding privileges for each operating system object (e.g. hard disk, file, network, …). Furthermore the users, user groups and their privileges can be propagated across the network.

Table 1: Methods in Vtable sequence

IUnknown Methods	Description
QueryInterface	Returns a pointer to the interface requested.
AddRef	Increments the reference counter.
Release	Decrements the reference counter.
IDispatch Methods	Description
GetTypeInfoCount	Supplies the number of "type information" interfaces, that an object offers. (0 or 1).
GetTypeInfo	Gets the type information for an object.
GetIDsOfNames	Connects names of methods with optional arguments with a corresponding set of DISPIDs.
Invoke	Offers access to properties and methods of an object.

ITcUserManager Methods	Description
CreateUserAccount [▶ 14]	Creates a new user account.
CreateUserAccountDlg [▶ 15]	Dialogue for generation of a new user account.
ImpersonateUser [▶ 11]	The invoking thread impersonates the user denoted by name and password.
ImpersonateUserDlg [▶ 11]	Dialogue with password query for impersonation of a user.
RevertToSelf [▶ 12]	A previous impersonation is cancelled.
UserIsMemberOf [▶ 17]	Checks whether the invoking thread is a member of the desired user group.
CreateProcessAsUser [▶ 13]	Starts a process with the desired user account.
CreateProcessAsUserDlg [▶ 13]	Dialogue for the start of a process with specified user account.
UserIsAdmin [▶ 17]	Checks whether the invoking thread belongs to the administrators' group.

ITcUserManager Methods	Description
UserName [▶ 18]	Property for display of the current user name.
UserEnumLocalGroups [▶ 16]	Returns the user groups of a user.
RemoveUserAccount [▶ 12]	Removes a user account.
EnumLocalGroups [▶ 15]	Returns the locally defined user groups.
EnumLocalUsers [▶ 15]	Returns the locally defined users.

4.2.1 ImpersonateUser

ITcUserManager [▶ 10]::ImpersonateUser

The invoking thread impersonates the user denoted by name and password.

```
HRESULT ImpersonateUser(
  BSTR bstrUserName,
  BSTR bstrPassword,
  VARIANT varDomain
);
```

Parameters

bstrUserName	[in] Name of the user who is to be impersonated.
bstrPassword	[in] Password of the user.
varDomain	[in, optional] If the user belongs to a domain, there is an option here to provide the name of the appropriate domain. The parameter provided must be of BSTR type to be accepted.

Return Values

HRESULT == S_OK	User account has successfully been created.
HRESULT != S_OK	If an error occurs an error code generated by Windows NT appears in HRESULT. In order to instigate a COM/OLE error handling sequence, an HRESULT is created from the error code from HRESULT_FROM_NT(nErrror) or HRESULT_FROM_WIN32(nErrror). Here it should be noted that the Windows NT error code is displayed in low-order 16 bit format.

Comments

The **ImpersonateUser** method assigns the ”impersonation“ token of the desired user to the invoking thread. For this purpose, the user is interactively logged in with username and password and the created token is used for impersonation. After a successful execution of ImpersonateUser the thread has adopted the identity of the user. This can be reversed by calling the method RevertToSelf [▶ 12] .

4.2.2 ImpersonateUserDlg

ITcUserManager [▶ 10]::ImpersonateUserDlg

Dialogue to change the user while process is running.

```
HRESULT ImpersonateUserDlg()
```

Return Values

HRESULT == S_OK	User account has successfully been created.
HRESULT != S_OK	If an error occurs an error code generated by Windows NT appears in HRESULT. In order to instigate a COM/OLE error handling sequence, an HRESULT is created from the error code from HRESULT_FROM_NT(nErrror). Here it should be noted that the Windows NT error code is displayed in low-order 16 bit format.

Comments

ImpersonateUserDlg calls up the ImpersonateUser [▶ 11] method internally.

4.2.3 RemoveUserAccount

ITcUserManager [▶ 10]::RemoveUserAccount

Removes a user account.

```
HRESULT RemoveUserAccount(
BSTR bstrUserName,
  VARIANT serverName
);
```

Parameters

bstrUserName	[in] Name of the user account that is to be removed.
varComputerName	[in, optional] There is an option to provide the name of the computer as a BSTR. If no name is provided, the user account is removed from the local computer.

Return Values

HRESULT == S_OK	Removal executed successfully.
HRESULT != S_OK	If an error occurs an error code generated by Windows NT appears in HRESULT. In order to instigate a COM/OLE error handling sequence, an HRESULT is created from the error code from HRESULT_FROM_NT(nErrror). Here it should be noted that the Windows NT error code is displayed in low-order 16 bit format.

4.2.4 RevertToSelf

ITcUserManager [▶ 10]::RevertToSelf

A previous impersonation is cancelled.

```
HRESULT RevertToSelf();
```

Return Values

HRESULT == S_OK	User account has successfully been created.
HRESULT != S_OK	If an error occurs an error code generated by Windows NT appears in HRESULT. In order to instigate a COM/OLE error handling sequence, an HRESULT is created from the error code from HRESULT_FROM_NT(nErrror) or HRESULT_FROM_WIN32(nErrror). Here it should be noted that the Windows NT error code is displayed in low-order 16 bit format.

Comments

The RevertToSelf method reassigns the „primary token“ of the process to the invoking thread. Thus RevertToSelf does not only cancel the last impersonation, but also after successive impersonations re-establishes the first user of the process.

4.2.5 CreateProcessAsUser

ITcUserManager [▶ 10]::CreateProcessAsUser

ITcUserManager [▶ 10]

Starts a process with the desired user account.

```
HRESULT CreateProcessAsUser(
  BSTR bstrUserName,
  BSTR bstrPassword,
  BSTR bstrProcessPath,
  VARIANT varDomain
);
```

Parameters

bstrUserName	[in] Name of the user account with which the process should be started.
bstrPassword	[in] Password of the user account.
varDomain	[in, optional] If the user belongs to a domain, there is an option here to provide the name of the appropriate domain. The parameter provided must be of BSTR type to be accepted.

Return Values

HRESULT == S_OK	Process has successfully been started.
HRESULT != S_OK	If an error occurs an error code generated by Windows NT appears in HRESULT. In order to instigate a COM/OLE error handling sequence, an HRESULT is created from the error code from HRESULT_FROM_NT(nErrror) or HRESULT_FROM_WIN32(nErrror). Here it should be noted that the Windows NT error code is displayed in low-order 16 bit format.

Comments

The **CreateProcessAsUser** method starts a process with the privileges of the specified use. The process therefore has access to all objects (partitions, directories, files, networks, etc) that are released for the user in question. A particular feature appears in the case of Windows Explorer. Standard practice is that all Explorer windows belong to the same process. If a new instance of Explorer is started using **CreateProcessAsUser**, a conflict arises because the Explorer process that is already active belongs to another user account. The instance is shut down again without having been observed by the user. If it is therefore desired to start an instance of Explorer with another user account, the Explorer process that is running must firstly be terminated.

4.2.6 CreateProcessAsUserDlg

ITcUserManager [▶ 10]::CreateProcessAsUserDlg

Dialogue to start a process with the specified user account.

```
HRESULT CreateProcessAsUserDlg()
```

Return Values

HRESULT == S_OK	The process has successfully been started.
HRESULT != S_OK	If an error occurs an error code generated by Windows NT appears in HRESULT. In order to instigate a COM/OLE error handling sequence, an HRESULT is created from the error code from HRESULT_FROM_NT(nErrror). Here it should be noted that the Windows NT error code is displayed in low-order 16 bit format.

Comments

CreateProcessAsUserDlg calls up the CreateProcessAsUser [▶ 13] method internally.

4.2.7 CreateUserAccount

ITcUserManager [▶ 10]::CreateUserAccount

ITcUserManager [▶ 10]

Sets up a new local user account.

```
HRESULT CreateUserAccount(
  BSTR bstrUserName,
  BSTR bstrPassword,
  VARIANT varUserGroup,
  VARIANT varDomain
);
```

Parameters

bstrUserName	[in] Name of the new user account.
bstrPassword	[in] Character string for the desired password.
varUserGroup	[in, optional] There is an option to specify a user group to which the new user account is to belong. The parameter provided must be of BSTR type to be accepted.
varDomain	[in, optional] Similarly, there is an option to provide the domain in which the new user account is to be created. The parameter provided must be of BSTR type to be accepted. At the present time this option is not yet supported.

Return Values

HRESULT = S_OK	User account has successfully been created.
HRESULT <> S_OK	If an error occurs an error code generated by Windows NT appears in HRESULT. In order to instigate a COM/OLE error handling sequence, an HRESULT is created from the error code from HRESULT_FROM_NT(nErrror). Here it should be noted that the Windows NT error code is displayed in low-order 16 bit format.

Comments

Along with the new user account **CreateUserAccount** creates additional user privileges, that are required for further functions of the TwinCAT User Manager (e.g. CreateProcessAsUser). The additional user privileges are:	
SE_TCB_NAME	"Act as part of the operating system". This privilege is required internally to log in a user.

SE_ASSIGNPRIMARYTOKEN_NAME	"Replace process level token", is required for CreateProcessAsUser.
SE_INCREASE_QUOTA_NAME	"Increase quotas", is also required for CreateProcessAsUser.

4.2.8 CreateUserAccountDlg

ITcUserManager [▶ 10]::CreateUserAccountDlg

Dialogue for generation of a new user account.

```
HRESULT CreateUserAccountDlg()
```

Return Values

HRESULT == S_OK	User account has successfully been created.
HRESULT != S_OK	If an error occurs an error code generated by Windows NT appears in HRESULT. In order to instigate a COM/OLE error handling sequence, an HRESULT is created from the error code from HRESULT_FROM_NT(nErrror). Here it should be noted that the Windows NT error code is displayed in low-order 16 bit format.

Comments

CreateUserAccountDlg calls up the CreateUserAccount [▶ 14] method internally.

4.2.9 EnumLocalUsers

ITcUserManager [▶ 10]::EnumLocalUsers

Returns the locally defined users.

```
HRESULT EnumLocalUsers(
  VARIANT varComputerName,
  SAFEARRAY(BSTR)* ppUsers
);
```

Parameters

varComputerName	[in, optional] There is an option to provide the name of the target system as a BSTR. If no name is provided, the users are listed on the local computer.
ppUsers	[out, retval] Pointer to SAFEARRAY of BSTRs, in which the names of the user groups are stored.

Return Values

HRESULT == S_OK	Query executed successfully.
HRESULT != S_OK	If an error occurs an error code generated by Windows NT appears in HRESULT. In order to instigate a COM/OLE error handling sequence, an HRESULT is created from the error code from HRESULT_FROM_NT(nErrror). Here it should be noted that the Windows NT error code is displayed in low-order 16 bit format.

4.2.10 EnumLocalGroups

ITcUserManager::EnumLocalGroups

ITcUserManager

Returns the locally defined user groups.

```
HRESULT EnumLocalGroups(
  VARIANT varComputerName,
  SAFEARRAY(BSTR)* ppGroups
);
```

Parameters

varComputerName	[in, optional] There is an option to provide the name of the target system as a BSTR. If no name is provided, the users are listed on the local computer.
ppGroups	[out, retval] Pointer to SAFEARRAY of BSTRs, in which the names of the user groups are stored.

Return Values

HRESULT == S_OK	Query executed successfully.
HRESULT != S_OK	If an error occurs an error code generated by Windows NT appears in HRESULT. In order to instigate a COM/OLE error handling sequence, an HRESULT is created from the error code from HRESULT_FROM_NT(nErrror). Here it should be noted that the Windows NT error code is displayed in low-order 16 bit format.

Also see about this

- ITcUserManager [▶ 10]

4.2.11 UserEnumLocalGroups

ITcUserManager [▶ 10]::UserEnumLocalGroups

ITcUserManager [▶ 10]

Returns the user groups to which a user belongs.

```
HRESULT UserEnumLocalGroups(
  VARIANT varUserName,
  SAFEARRAY(BSTR)* ppGroups
);
```

Parameters

varUserName	[in, optional] There is an option to provide the name of the user as a BSTR. If no name is provided, the user of the invoking thread is queried.
ppGroups	[out, retval] Pointer to SAFEARRAY of BSTRs, in which the names of the user groups are stored.

Return Values

HRESULT == S_OK	Query executed successfully.
HRESULT != S_OK	If an error occurs an error code generated by Windows NT appears in HRESULT. In order to instigate a COM/OLE error handling sequence, an HRESULT is created from the error code from HRESULT_FROM_NT(nErrror). Here it should be noted that the Windows NT error code is displayed in low-order 16 bit format.

4.2.12 UserIsAdmin

ITcUserManager [▶ 10]::UserIsAdmin

ITcUserManager [▶ 10]

Checks whether the invoking thread belongs to the administrators' group.

```
HRESULT UserIsAdmin(
VARIANT varUserName,
  VARIANT_BOOL* pbResult
);
```

Parameters

varUserName	[in] Optional: Name of a user, provided as a BSTR. At the present time this option is not supported. The user status of the invoking thread is checked.
pbResult	[out, retval] Pointer to a VARIANT_BOOL, in which the result of the query is to be stored.

Return Values

HRESULT == S_OK	Query executed successfully.
HRESULT != S_OK	If an error occurs an error code generated by Windows NT appears in HRESULT. In order to instigate a COM/OLE error handling sequence, an HRESULT is created from the error code from HRESULT_FROM_NT(nErrror). Here it should be noted that the Windows NT error code is displayed in low-order 16 bit format.

4.2.13 UserIsMemberOf

ITcUserManager [▶ 10]::UserIsMemberOf

ITcUserManager [▶ 10]

Checks whether the invoking thread is a member of the desired user group.

```
HRESULT UserIsMemberOf(
  BSTR bstrUserGroup,
  VARIANT varUserName,
  VARIANT_BOOL* pbResult
);
```

Parameters

bstrUserGroup	[in] Via the pointer pbIsMemberOf.
varUserName	[in] Optional name of a user provided as a BSTR.
pbResultc	[out, retval] Pointer to a VARIANT_BOOL, in which the result of the query is to be stored.

Return Values

HRESULT == S_OK	Query executed successfully.
HRESULT != S_OK	If an error occurs an error code generated by Windows NT appears in HRESULT. In order to instigate a COM/OLE error handling sequence, an HRESULT is created from the error code from HRESULT_FROM_NT(nErrror). Here it should be noted that the Windows NT error code is displayed in low-order 16 bit format.

4.2.14 UserName

ITcUserManager [▶ 10]::UserName

Property for display of the current username.

```
HRESULT UserName(
BSTR* bstrUserName,
);
```

Parameters

bstrUserName	[out, retval] Name of the user account to which the invoking thread belongs, provided as a pointer to a BSTR object.

Return Values

HRESULT == S_OK	Query executed successfully.
HRESULT != S_OK	If an error occurs an error code generated by Windows NT appears in HRESULT. In order to instigate a COM/OLE error handling sequence, an HRESULT is created from the error code from HRESULT_FROM_NT(nErrror). Here it should be noted that the Windows NT error code is displayed in low-order 16 bit format.

Comments

UserName displays the name of the user account to which the invoking thread belongs. This name can differ from the name of the user who is logged in.

More Information:
www.beckhoff.com/automation

Beckhoff Automation GmbH & Co. KG
Hülshorstweg 20
33415 Verl
Germany
Phone: +49 5246 9630
info@beckhoff.com
www.beckhoff.com