TUsersCS
for Delphi 3/4

User's Manual

by Tools&Comps

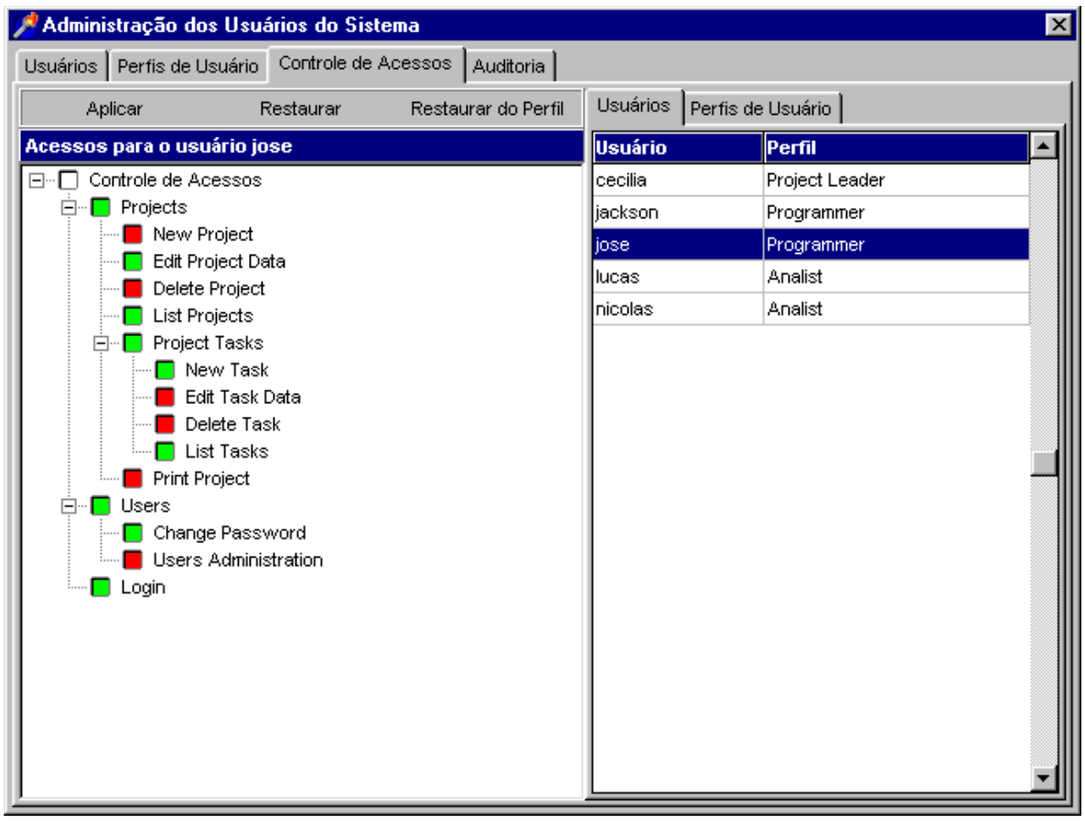
Tools&Comps Home Page: www.gtc.com.br/~jgkrs/tandco.htm
Tools&Comps e-mail: tandco@gtc.com.br

OVERVIEW

TUsersCS (**C**lient/**S**erver version) is a component that make simple implement the control of security in programs written in Delphi. With a little configuration at component level in desing time, your application is ready to control the access of its users. With **TUsersCS** you do not need to write code for implementing the user's control. You just need to connect it to the **TMenu** component of your main form and the component take cares of everything for you.

Through the connection with the menu of your application's main form, the **TUsersCS** allows that the security administrator of your application grants access to the user for each item in the menu. All the user's accesses are shown in a tree that imitate the menu hierarchy of your application. The administrator just need to click in the desired item to grant or not the user's access to this item. When the user logs in the application, the component **TUsersCS** will disable or make invisible (accordingly with the profile the users belongs to) the menu items the user do not have access.

The component **TUsersCS** has a audit module, that allows to verify the utilization of the programs by the user.



The administration module is very easy of use, mainly for the end user of your application. In this module, the administrator can:

- Create users;
- Create users profiles;
- associate users with a profile;
- Grant permissions of accesses for users (individually) and for profiles, visually;
- Update permissions of access for all users that belongs to a profile;
- Change user's data and password;
- Enforces the change of password for all users of a profile at each interval of days especified;
- Put the user (or all users of a profile) in a audit mode, to verify the use of the application by the user, among other actions.

The component **TUsersCS** can be used with any DBMS avaiable on the market that supports the ANSI-SQL standard and that can be accessed by Delphi using **BDE** or **ODBC.**

Another important **TUsersCS** feature is that that tables for users's control can be shared among any number of applications at the same time, and the users do not nedd to be inserted again for all applications.

TUsersCS is implemented in pure Object Pascal, inclusive the module of administration of users, that is implemented together with the component. External librarys are not made necessary neither additional programs that need to be sent together with your Delphi program so that the component works correctly. You just need to send the tables of users with your application.

INSTALLING TUSERSCS

In order to install the **TUsersCS:**

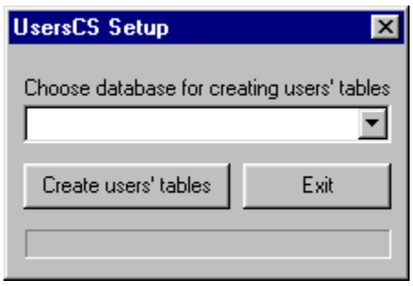
- open the file ToolsAndComps\UsersCS\Delphi3Trial\uCSD3.dpk, through the menu File|Open of your Delphi and clic in the **Install** button.
- At the Delphi IDE, open Tools|Enviroment Options|Library|Library Path, and add c:\ToolsAndComps\UsersCS\Delphi3Trial, and then clicking in the buttom **OK**.

Note: The procedures are the same for Delphi 4.

HOW THE COMPONENTE TUSERSCS WORKS

1. Creating the user's tables

The first step in order to use the **TUsersCS** is create the tables that will go store the information of the users of your system. The component **TUsersCS** comes accompanied by an utilitary that aid in the creation of that table, the **UCSSetup**.**EXE**.



In order to create the user's tables, you should enter the alias (database name) where yua want to store this tables (you must create previously an alias at the BDE or a DSN in the ODBC administrator of your computer. After entering the necessary data, just clic in the button " **Create users's tables** " so that the utilitary will create the tables in accordance with the specified information. The tables that will be created are:

1. USERSCS
2. USERINFO
3. PROFILES
4. PROGS
5. UACCESS
6. PACCESS
7. HISTORY
8. AUDIT

Tip: You can create the users's tables in the same database that your application will work or you can create a específic database for the users's control.

Note: The security of the users's tables must be the same of the data your application will work.

When the utilitary UCSSetup.EXE creates the tables of users, it automatically inserts a record with name of **'master'** user, with password also **'master'** and total access to the system. That is the user that will go to manage the system of security of your programs, not being possible delete such user, but its possible to change its password (what is recommended).

Note: For an better security of the system, the password of all users are encrypted.

2. Using the TUsersCS inside the Delphi's IDE

In order to use the TUsersCS in your programs, in the components' palette, select the **ToolsAndComps** page, then add the component to the main form of your program.

So that the component works correctly, you must insert initially the value of the following properties in the Object Inspector:

AppKey: String[20]

Key for the application. This property is mandatory and allows that the users's tables can be shared among your applications. It must be unique among the applications that will share the same users's database. Note that the component cannot verify that this key is unique, so, you must take care about it.

DatabaseName: String

Indicate the alias that that component TUsersCS will connect to open the users's tables.

IMPORTANT: Throught the alias specified as parameter, the component will try to open the users's database (Created with the utility **UCSSetup**) using the alias' properties, including the user and the password that your application uses for open your database. Note that the component must be "authorized" to access the users's database. This authorization can be made in the follow way:

1. Ask to your DBA to create a user in your database and that such users, and just it, will have permissions of access to the users's tables.
2. Add a TDatabase component to your main form and connect it to the database where are stored the users's tables.
3. In the property Params of the TDatabase, add the follow lines:
   USER NAME=" User Name Created by the DBA"
   PASSWORD="User's Password"
4. Change the value of the property LoginPrompt to FALSE
5. Connect the component TUsersCS of your main form to the TDatabase component created.

See the samples that come with the component.

Menu: TMenu

Indicates the menu component that the **TUsersCS** will control. If your form has a TMenu Component, the **TUsersCS** will automatically be connected to it.

Important: Notice that these should be the values that you entered when created the tables of users using the utilitary **UCSSetup**.**EXE**. Without the correct values, the component won't works properly.

3. Other properties of the TUsersCS

AutoLogin: Boolean

Specifies if the component is going to call the function of Login in the system automatically. If *AutoLogin* is **True**, the component calls the function of Login when the application is begun. If the user doesn't get logged in the application after 3 tentatives, the component finishes the application. If *AutoLogin* is **False** the function of Login should be called manually.

DatabaseName: String

Specifies the name of the internal alias created by the component. That alias is also available for the application.

UseAppDir: Boolean

If *UseAppDir* is **True**, the component is going to use the directory where is store the executable file of your application together with the *TableDir* property in order to have access to the table of users.

ActualUser: TUser

That property contains data about the user that is logged in the system.

4. Properties of the TUser Class

UserName: String

Name of the user that is logged

RealName: String

Real Name of the user that is logged

UserId: Integer

Number of identification of the user that is logged

Password: String

Password of the user that is logged.

Note: The component keeps a password history of all users, to enforce then to use different passwords everytime it changes your password. In order to user the same password, the users's administrator must remove the password from the password history of the users, in the administration module.

LastPwdChange: TDateTime

Last date the user changed its password

ExpirationDate: TDateTime

The date that the user will expire in the application. Starting to this date, the user will can not login in the application. I order to use the application again, the seciruty administrator must set the user active again and change his expiration date. This property just make sense if the property *UserExpire* is TRUE.

Important: Note that this value belongs to the application. It means that the user has a different expiration date for each application.

UserExpire: Boolean

Indicates if the user will be valid in the application after the expiration date.

Important: Note that this value belongs to the application. It means that the user can expire or not, depending of the application.

UserActive: Boolean

Indicates if the user is active.

Important: Note that this value belongs to the application. It means that the user can be active or not, depending of the application.

Profile: String

Returns the profile name the user belongs to.

ProfileId: Integer

Returns the profile identifier the user belongs to.

AuditMode: Boolean

Indicates if the user is in audit mode or not.

5. Methods of the component TUsersCS

Login: boolean

That method should be used in order to call the screen of Login of the system. If after 3 tentatives, the user doesn't get logged in the system, that function is going to return **False**. For a login well happened, that function is going to return **True**. The call to that function is made necessary when the *AutoLogin* property is **False**. In this case, the developer should write the necessary code so that the application could not be used for an user not authorized. See the examples.

ChangeUserPassword

That method permits the logged user to change his password. Must be called by the developer.

UsersAdm

Calls the module of administration of users of the system. Must be called by the developer.

SUPPORT TO OTHER LANGUAGES

The component TUsersCS was designed so that the developer could modify the language used by the component easily. The current version has translations of all their strings of interface with the user in Portuguese, Spanish and English. For each one of the mentioned languages, the package of the component TUsersCS comes with the resources file already compiled(UsersCSRes.RES), the source file of the STRING TABLE (UsersCSRes.RC) and the file with the constants (UsersCSRes.PAS). In order to compile your program for one of the mentioned languages, include in your project, the UsersCSRes.Pas file that is in the directory of the wanted language.

If you want to use the component in German, for example, follow these steps:

1. Inside the directory ToolsAndComps\UsersCS\Resources, create a directory for the wanted language
2. Copy all the files of the directory of some of the languages for the new directory.

3. Translate the UsersCSRes.RC file for the wanted language.
4. Compile the new UsersCSRes.RC file using the utilitary BRCC32.EXE, that is in the BIN directory of your Delphi installation. Consult your Delphi documentation to regarding the use of the BRCC32.EXE.
5. Add to your project, the UsersCSRes.Pas file that is in the directory of the new language. The new generated UsersCSRes.RES file should also be in that directory.
6. Rebuild your project.

REGISTERING THE COMPONET TUSERSCS

Visit our home page to obtain upadted information about registering the component TUsersCS v1.0.