noCPROTECT&Functions~JI(`.HLP>main',`CPROT.FR_001')&Aboutyes
CPROTECTCPROTECTTRUECPROTECTCPROTECTCPROTECTTRUE
CPROTECTCPROTECTCPROTECTCPROTECTCPROTECTCPROTECTCPROTECT
CPROTECTCPROTECTCPROTECTCPROTECTCPROTECTCPROTECTCProtect
Help ¸ Thomas Lichtneckert 1995CProtectCPROT.I__001CProtect
Help ¸ Thomas Lichtneckert 1995CProtect HelpCProtect
HelpCProtectnonoE&xit&Printyesyesyes 3CProtectyes12/07/95

Table of Contents

Contents

Cprotect is shareware! You are therefore required to register after a limited time. If you want to use it commercially then registering is pure selfpreservation, since this version is available to anybody.
Your will not be required to pay any form of license fee for the programs you use it with (and make money I hope!) - I would not be able to check that anyway.

Features

Why register?

Register!

General outline of a CProtected program

Function reference

The skeleton of a CProtected program - C/C++

How to use Cprotect with Visual Basic / Access??

selfpreservation

instinctive, impulse to avoid injury or death

- *Collins National Dictionary*

Function reference

(The function declarations are given for **C/C++** and for **Visual Basic/Access Basic**.)
It is generally very important to match the function parameters!

Help file produced by **HELLLP!** v2.4b , a product of Guy Software, on 1995-07-12 for Thomas Lichtneckert.
The above table of contents will be automatically completed and will also provide an excellent cross-reference for context strings and topic titles. You may leave it as your main table of contents for your help file, or you may create your own and cause it to be displayed instead by using the I button on the toolbar. This page will not be displayed as a topic. It is given a context string of _:_ and a HelpContextID property of 32517, but these are not presented for jump selection.
HINT: If you do not wish some of your topics to appear in the table of contents as displayed to your users (you may want them ONLY as PopUps), move the lines with their titles and contexts to below this point. If you do this remember to move the whole line, not part. As an alternative, you may wish to set up your own table of contents, see Help under The Structure of a Help File.
Do not delete any codes in the area above the Table of Contents title, they are used internally by HELLLP!

Introduction

CProtect - a versatile registering and software protection utility for smart developers.

Rationale

The shareware concept is very sensible and the philosophy to try before buy is a very good and strong argument in favor of good software. However I feel that sometimes (always?!) a little extra "help" to register a shareware product, and thus ***honoring*** the author, might do a whole lot of good.

In order to offer prospective customers a "hands on" experience of my programs but still retaining the possibility to be able to sell the program to them, I constructed a scheme that allows you to lock parts of your program (like saving data, printing results etc) and also to limit the use of the "trial version" for a determined amount of time and/or number of runs. Naturally this scheme allows you to "unlock" those parts of the program that your customer pays for. This way you can sell your program by increments, you only need to maintain one version, and the user does not have to pay for features that he does not need. The unlocking is done with a **registration code** that is unique to the user´s copy of the program. (His neighbor will not be able to use the same code.) This way you will not even have to ship him another copy, just send him the code when you have received the payment and the **unique serial number** that was created when he installed his program. Feeding the user´s serial number and the access code for the parts that have been paid for to **CProtect** will generate the registration code - it is quite simple!

CProtect - for protecting your program - is implemented as a DLL, this way it is accessible by all types of Windows programs no matter how they were created (c, c++, VB, Access, Pascal etc). As a matter of fact there are **two DLLs**: one that can **generate registration codes**, that you should keep, and one that can´t, that you should ship with your program.

If there is enough interest then I will make **CProtect** as DOS-libraries as well, please contact me.

See also:
What can CProtect do for you?
Features
General outline of a CProtected program
The skeleton of a CProtected program - C/C++
How to use CProtect with Visual Basic / Access??

What can CProtect do for you?

Those who use your program (not just test it) must pay, your income will increase, you will have more time for programming (not having to worry about your economy), you can make even better programs thus making even more money....

By clever design you can enable practically all of your programs features except a very few but crucial ones in the "trial version", thus making more people realize that they need your program. It is usually very seriously limiting not to be able to save the data that you want to feed to the program and not to be able to save and print the results.

You can sell your program in increments - basic level, level 1, and so on up to level 15.

You can sell "runs" of your program - $ XX / 50 runs.

You don´t need to worry when you send out updates - only those with registered security files will be able to use it, so you can just post your update on a BBS. You save a lot of postage, diskettes, labor etc.

Registering is done by mail, phone, fax, email - even a public note on a BBS will only be of use to the right customer!

See also:
Features
General outline of a CProtected program
The skeleton of a CProtected program - C/C++
How to use CProtect with Visual Basic / Access??

`cprot20d.dll` - the **d**eveloper´s version - can create registration codes.
`cprot20u.dll` - the **u**ser´s version - can **NOT** create registration codes.

The serial numbers are created by randomly massaging the system time. They are always 12 alphanumeric characters wide.

The Author

e-mail: **Thomas.Lichtneckert@abc.se**
phone: +46 - 31 145 131
fax: +46 - 31 121 621
snaimail:

Thomas Lichtneckert
Nordenskiöldsgatan 24
S-413 09 Göteborg
SWEDEN

The registration codes are based on

1. The user´s unique serial number.
2. The access code.
3. Two 64-bit keys.

They are always 12 alphanumeric characters long.

Features

CProtect has the following features:

1. Assigns a **unique serial number** to each installation of a program, 12 characters long.
2. Unlocks any one of **15 possible accesses** in a program with a registration code - different for every serial number! For each serial number there are 9,999,999,985 combinations that are wrong and 15 that are correct - those corresponding to the 15 access codes.
3. Creates a **unique registration code** for a given serial number and a given access code.
4. Checks if the program is registered.
5. Tells you the **unique serial number** of the installed program.
6. Tells you which parts of the program should be enabled by returning an **accesscode**.
7. Keeps a **record of program usage** - in actual time(hours, minutes and seconds) and number of times the program was started. This gives you the possibility to limit usage at your wish.
8. Can store and retrieve **additional configuration** data.
9. Can store and retrieve the registered users **name and company**.
10. Checks from which diskette drive the program is installed and also if the diskette is write-protected, in case you would want to put some information on the installation diskette during installation.
12. Allows you to **set a file´s timestamp** - in order to conceal that you write to a certain file.
11. Can **delete an entire directory** with files (I had to make this routine because Microsoft´s doesn´t work).
12. You can **use it with any windows program** capable of using a DLL - language independent.
13. Can be supplied as a library for DOS developers on special request.
14. **NEW!** For increased security the security file is stored in encrypted form.

See also:
General outline of a CProtected program
The skeleton of a CProtected program - C/C++
How to use CProtect with Visual Basic / Access??

Why register?

Why should you register CProtect then? This version is complete! OK, that´s fine! You realize, of course, that ***anybody who downloads this shareware version will also be able to create registration codes***...

The matching of the registration code and the serial number is done by an algorithm requiring two different 64-bit keys, which are built into the DLL-pair. This makes it possible to have *more than 10 E38 different keys* - and thus different DLL-pairs! (*I promise to extend the algorithm, when 10 E35 users have registered CProtect, in order to eliminate the remotest chances that two CProtects will have the same keys...*)
Upon registering CProtect ***you will get a unique version of the Cprotect DLLs***, meaning that **only you will be able to make the proper registration codes for your program.**
NEW! Every registered user will get a version with ***unique encryption*** of the security file - **only you will be able to read it´s contents**.

*And after all... we **are** talking about software protection, are we not?!*

Registration form

General outline of a CProtected program

1. Upon programstart check that your security file exists and that it really is your file. Use **cpGetFileinfo**(path, filevers).
   If you don't have an installation procedure then you would have to create the security file the first time the program is run, use **cpCFGMake**(path, magicnumber, filevers). if you want to put the user's name and company in the security file, then this is the right time - use **cpSetRegisteredNames**(path, username, company). In order to "conceal" the fact that the security file is newly created (or at least trying to be discrete about it) you can fix the security files timestamp with **cpSetFTime**(path, date, time).
2. Check if the registration code with **cpCheckRegCode**(path, regcode), supplying an empty string for *regcode*.
   If you wish to limit the use of the program to a certain amount of runs or time then now you check theese with **cpGetUStatistics**(path, times_used, totalhous, totalminutes, totalseconds, hours_thistime, minutes_thistime, seconds_thistime).
3. If this is the time when the user wants to register the program, then you first check that the code he supplies is valid with **cpCheckRegCode**(path, regcode), this time *regcode* should contain the code the user inputs. If **cpCheckRegCode** returns a valid access code then write the regcode to the security file with **cpRegisterProgram**(path, regcode).
4. If you want to keep track of program usage then you start the counter/timer with **cpStartProgram**(path):
5. Your program now has the necessary information about which parts/functions should be enabled.

.....

You should have some means of showing the serial number in the security file (so the user can register!).
To get it you use **cpGetSernum**(path, sernum) - (you supply an empty string in sernum).

You may want the user's name to appear on the title bar, or on printouts - find our with **cpGetRegisteredName**(path, username).

.....

6. Time to finish the program. If you started the timer then stop it now with **cpEndProgram**(path).
   If you wish to show usage statistics to the user then you can get the figures with **cpGetUStatistics**(path, times_used, totalhous, totalminutes, totalseconds, hours_thistime, minutes_thistime, seconds_thistime).
7. Again you may want to be secretive about fiddeling with the security file - use **cpSetFTime**(path, date, time).

That's really all there is to it.

The skeleton of a CProtected program - C/C++

How to use CProtect with Visual Basic / Access??

Registration Form

Please print this form and mail or fax it when completed.

Yes, I see your point and would like to register and receive a ***unique version*** of **CProtect**

DATE: _____

NAME: _____

COMPANY NAME: _____

ADDRESS _____

_____

_____

FAX NUMBER: _____

PHONE NUMBER: _____

E-MAIL ADDRESS: _____

Choose currency please and check one:

I pay :	**US$**( )	**£**( )	**DM**( )	**SEK**( )
Registration fee	50	32	70	375
Postage Sweden	N/A	N/A	N/A	10
Outside Sweden	8	5	11	N/A
Total	**58**	**37**	**81**	**385**

Method of payment, please check your choice

- International check. ( )
- Cash (registered mail). ( )
- Bank transfer to ( ) **S-E Bank Sweden, Swift code ESSESESG, Account No. 5013 00 003 02.**
- Europe: Giro transfer to ( ) **Sweden: Account No. 448 06 36-2.**

No credit cards, please.

Thomas Lichtneckert
Nordenskiöldsgatan 24
S-413 09 Göteborg
SWEDEN

Your order will be shipped by **Euroletter** (quicker then 1:st class) within 24 hours on reception of your payment.

If you have any questions, comments, suggestions or if you discover any bugs please contact the author at the above address or by

e-mail: Thomas.Lichtneckert@abc.se.
Fax: +46-31 121 621
Phone: +46-31 145 131

No comments

The skeleton of a CProtected program - C/C++

Here is an outline of how to use Cprotect in the program that you ship.

Of course if you have an installation procedure for your software, and I believe that you should, you have further possibilities to make your protection even better with Cprotect.
See also General outline of a CProtected program

When compiling and linking your program you have the usual alternatives:

1. Explicitly with **IMPORTS** statements.
2. Implicitly with an import library, which is the simplest method. You just link in the cprot20d.lib / cprot20u.lib which are included in the **CProtect** package. This is the method used in the following examples.
3 Dynamically with **LoadLibrary( )**

```
#include "cprot20.h"
...
static int f_code;
...
int PASCAL WinMain(HANDLE hInstance, HANDLE hPrevInstance, LPSTR lpCmdLine,
                   int nCmdShow)
{
    char * mysecfile = "myfile.ext";       /* this is your security file */

    /* first check if the program is registered */
    if( !cpIsRegistered( mysecfile ) )
    {
        /* then check which access your program is registered for... */
        char rcode[15];
        rcode[0] = \0;          /* make rcode empty */
        f_code = cpCheckRegCode( mysecfile, rcode );
        if( f_code < 1 || f_code > 15 )
        {
            /* here you tell the user that something has gone wrong, the
               regcode in the security file is not matching his serial number*/
            /* and close the program...*/
            return someerror;
        }
    }
    else       /* mysecfile is not found!!! */
    {
        /* if you have an installation procedure then this is an error and you
           should tell the user that somehow something went wrong...
           in this case the program ends here returning some errorcode */

        /* if you don´t have an installation procedure then this may be the
           first time the program is used, so the security file needs to be
           created. You should tell the user that obviously this is the first
           time and a serial number will be created which he should supply when
           registering */
        long magic = 123456;
```

```
    long fileversion = 1;
    if( cpCFGMake( mysecfile, magic, filevers ) )
    {
        /* oops! could not create the file!!! */
        return someerror;
    }
    /* now let´s get the serial number */
    char sernum[15];
    if( cpGetSernum( mysecfile, sernum ) )
    {
        /* problems accessing the file...*/
        return someerror;
    }
}
/* OK if we have come this far then either the program is registered and
   f_code has a value between 1 and 15 or the program is not registered and
   f_code = 0 */

/* maybe you want to check the usage and take som action?? */
if( !fcode )
{
    int times, t_hour, t_min, t_sec, l_hour, l_min, l_sec;
    if( cpGetUStatistics( mysecfile, &t_hour, &t_min, &t_sec, &l_hour,
                          &l_min, &l_sec ) )
    {
        /* what happened to the security file??? */
        /* maybe tell the user??? */
        return someerror;
    }
}
/* here you perform what you think fit with the returned staistics, maybe
    tell the user that his trial period is up!*/


/* if you want to keep record of usage then start the timer */
if( cpStartProgram( mysecfile ) )
{
    /* what happened to the security file??? */
    /* maybe tell the user??? */
    return someerror;
}

...
    /* here is your original program code, you now know if and for which
       access the program is registered and can take proper actions
       depending on the value of f_code */
...

/* time to finish the program... */
/* if you started the timer then now is the proper time to stop it */
if( cpEndProgram( mysecfile ) )
{
    /* what happened to the security file??? */
    /* maybe tell the user??? */
    return someerror;
}
/* finally you may want to set the timestamp of your security file to a
```

```
        standard value in order not to awake the user´s curiosity...*/
    long yymmdd = 950301;           /* the date */
    int hhmm = 301;                 /* the time */
    if( cpSetFTime( mysecfile, yymmdd, hhmm ) )
    {
        /* what??? is the secfile missing?? */
        /* tell the user?? */
        return someerror;
    }
    return 0;
}
```

```
#include "cprot20.h"
...
static int f_code;
...
int PASCAL WinMain(HANDLE hInstance, HANDLE hPrevInstance, LPSTR lpCmdLine,
                   int nCmdShow)
{
    char * mysecfile = "myfile.ext";

    if( !cpIsRegistered ( mysecfile ) )
    {
        char rcode[15];
        rcode[0] = \0;
        f_code = cpCheckRegCode ( mysecfile, rcode );
        if( f_code < 1 || f_code > 15 ) return someerror;
    }
    else
    {
        long magic = 123456;
        long fileversion = 1;
        if( cpCFGMake( mysecfile, magic, filevers ) ) return someerror;
        char sernum[15];
        if( cpGetSernum( mysecfile, sernum ) ) return someerror;
    }
    if( !fcode )
    {
        int times, t_hour, t_min, t_sec, l_hour, l_min, l_sec;
        if( cpGetUStatistics ( mysecfile, &t_hour, &t_min, &t_sec, &l_hour,
                               &l_min, &l_sec ) ) return someerror;
    }

    if( cpStartProgram ( mysecfile ) ) return someerror;

    ...
    ...
    if( cpEndProgram ( mysecfile ) ) return someerror;
    long yymmdd = 950301;
    int hhmm = 301;
    if( cpSetFTime ( mysecfile, yymmdd, hhmm ) ) return someerror;
    return 0;
}
```

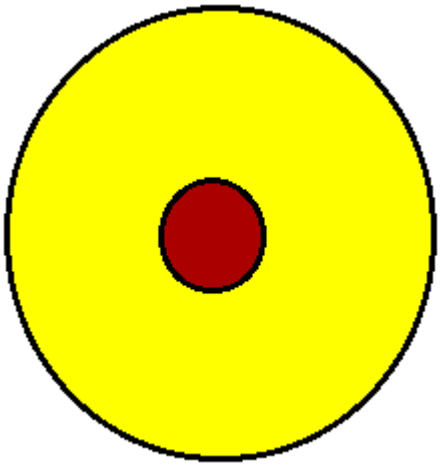
How to use CProtect with Visual Basic / Access??

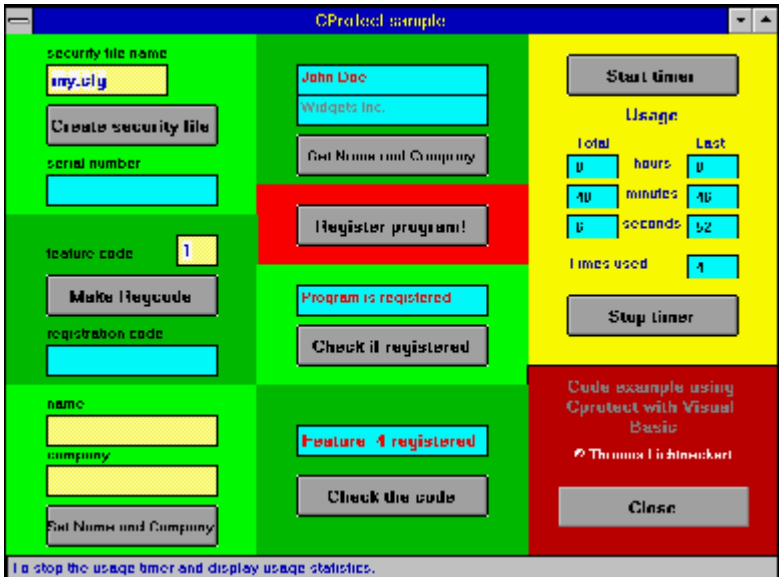
A sample program is supplied both in code (`cptest.mak`, `cptest.frm`, `global.bas` and `cptest.bas`) and in compiled form (you need vbrun100.dll). Please study the source code and you will find all the necessary information in this testprogram.

NB Most of the functions that the sample program performs should be totally transparent to the user and thus not be assigned to pushbuttons. They should be carried out during program start and program ending. Please study also General outline of a CProtected program

You can also look at the event functions by clicking the buttons on the picture of the **CPtest** screen below.

If you have `vbrun100.dll` then press the button to run the program!





You can see the event functions by clicking the buttons in this picture!

You find the global constants and the function declarations for your CProtected applications in

`global.bas`
See also General outline of a CProtected program

Global Constants

```
Global Const FILE_NOT_FOUND              = -10
Global Const PATH_NOT_VALID              = -11
Global Const NO_PATH                     = -12
Global Const NOT_REMOVABLE               = -13
Global Const DRIVE_OK                    = 0
Global Const NO_ARGUMENTS                = 1
Global Const DRIVE_NOT_FLOPPY            = 2
Global Const DRIVE_NOT_READY             = 3
Global Const DRIVE_NOT_VALID             = 4
Global Const DRIVE_ERROR                 = 5
Global Const WRONG_DISKETTE              = 6
Global Const DRIVE_WRPROT                = 7
Global Const DRIVE_WRPROT_A              = 8
Global Const DRIVE_WRPROT_B              = 9
Global Const DISK_ISIN_A                 = 10
Global Const DISK_ISIN_B                 = 11
Global Const NOT_REGISTERED              = 20
Global Const NO_REGINFO                  = 21
Global Const NO_SERNUMBER                = 22
Global Const FEATURES_NOT_AVAILABLE      = 23
Global Const WRONG_CODE                  = 24
Global Const IS_REGISTERED               = 748
```

Function declarations

NB! the .DLL names should match! For your program that creates the *registration codes* you should use **cprot20d.dll** since **cpMakeRegCode** is not present in cprot20u.dll**!**

Declare Function **cpMakeRegCode** Lib "cprot20u.dll" (ByVal access%, ByVal serialnum$, ByVal regcode$) As Integer
Declare Function **cpCFGMake** Lib "cprot20u.dll" (ByVal path$, ByVal magic&, ByVal filever&) As Integer
Declare Function **cpGetSernum** Lib "cprot20u.dll" (ByVal path$, ByVal serialnum$) As Integer
Declare Function **cpRegisterProgram** Lib "cprot20u.dll" (ByVal path$, ByVal regcode$) As Integer
Declare Function **cpCheckRegCode** Lib "cprot20u.dll" (ByVal path$, ByVal regcode$) As Integer
Declare Function **cpGetRegisteredName** Lib "cprot20u.dll" (ByVal path$, ByVal uname$) As Integer
Declare Function **cpGetRegisteredCompany** Lib "cprot20u.dll" (ByVal path$, ByVal company$) As Integer
Declare Function **cpSetRegisteredNames** Lib "cprot20u.dll" (ByVal path$, ByVal uname$, ByVal company$) As Integer
Declare Function **cpIsRegistered** Lib "cprot20u.dll" (ByVal path$) As Integer
Declare Function **cpGetUParams** Lib "cprot20u.dll" (ByVal path$, lparam&, iparam1%, iparam2%, iparam3%, iparam4%, iparam5%) As Integer
Declare Function **cpSetUParams** Lib "cprot20u.dll" (ByVal path$, ByVal lparam&, ByVal iparam1%, ByVal iparam2%, ByVal iparam3%, ByVal iparam4%, ByVal iparam5%) As Integer
Declare Function **cpSetFTime** Lib "cprot20u.dll" (ByVal path$, ByVal fdate&, ByVal ftime%) As Integer
Declare Function **cpWhichDrive** Lib "cprot20u.dll" (ByVal file$) As Integer
Declare Function **cpDirRemove** Lib "cprot20u.dll" (ByVal path$) As Integer
Declare Function **cpStartProgram** Lib "cprot20u.dll" (ByVal path$) As Integer
Declare Function **cpEndProgram** Lib "cprot20u.dll" (ByVal path$) As Integer
Declare Function **cpGetUStatistics** Lib "cprot20u.dll" (ByVal path$, utimes%, totalhour%, totalmin%, totalsec%, lasthour%, lastMin%, lastsec%) As Integer
Declare Function **cpGetFileinfo** Lib "cprot20u.dll" (ByVal path$, version&) As Long

```
Sub Cfgname_LostFocus ()
    fil$ = Cfgname.Text
    If magicnumber = cpGetFileinfo(fil$, vernum&) Then
        fileisok = 1
        sernum$ = String$(15, 0)
        a = cpGetSernum(fil$, sernum$)
        Snumber.Text = sernum$
    Else
        fileisok = 0
        Snumber.Text = ""
    End If

    regcode.Text = ""

    Uname1.Text = ""
    Ucompany1.Text = ""

    Thour.Text = ""
    Tmin.Text = ""
    Tsec.Text = ""
    Chour.Text = ""
    CMin.Text = ""
    Csec.Text = ""
    Ttimes.Text = ""

    Checkmessage.Text = ""
    Codecheck.Text = ""
End Sub
```

```
Sub Cfgmake_Click ()
    file$ = Cfgname.Text
    a = cpCFGMake(file$, magicnumber, filevesion)
    fileisok = 1
    sernum$ = String$(15, 0)
    a = cpGetSernum(file$, sernum$)
    Snumber.Text = sernum$
    regcode.Text = ""

    Uname1.Text = ""
    Ucompany1.Text = ""

    Thour.Text = ""
    Tmin.Text = ""
    Tsec.Text = ""
    Chour.Text = ""
    CMin.Text = ""
    Csec.Text = ""
    Ttimes.Text = ""

    Checkmessage.Text = ""
    Codecheck.Text = ""End Sub
```

```
Sub Regmake_Click ()
    fcode = Val(Feature.Text)
    rcode$ = String$(15, 0)
    ser$ = Snumber.Text
    a = cpMakeRegCode(fcode, ser$, rcode$)
    regcode.Text = rcode$
End Sub
```

```
Sub Feature_Change ()
    regcode.Text = ""
End Sub
```

```
Sub Uname_Change ()
    Uname1.Text = ""
End Sub
```

```
Sub Ucompany_Change ()
    Ucompany1.Text = ""
End Sub
```

```
Sub Setnames_Click ()
    If fileisok Then
        u_name$ = Uname.Text
        u_comp$ = Ucompany.Text
        cfg_name$ = Cfgname.Text
        a = cpSetRegisteredNames(cfg_name$, u_name$, u_comp$)
    End If

    Uname1.Text = ""
    Ucompany1.Text = ""
End Sub
```

```
Sub Getnames_Click ()
    If fileisok Then
        u_name$ = String$(28, 0)
        u_comp$ = String$(28, 0)
        cfg_name$ = Cfgname.Text
        a = cpGetRegisteredName(cfg_name$, u_name$)
        Uname1.Text = u_name$
        a = cpGetRegisteredCompany(cfg_name$, u_comp$)
        Ucompany1.Text = u_comp$
    End If
End Sub
```

```
Sub Checkreg_Click ()
    If fileisok Then
       a1$ = "Program is "
       a2$ = "registered"
       cfg_name$ = Cfgname.Text
       a = cpIsRegistered(cfg_name$)
       If a = 0 Then
          yes$ = ""
       Else
          yes$ = "not "
       End If
       Checkmessage.Text = a1$ + yes$ + a2$
    Else
       Checkmessage.Text = " No file! "
    End If
End Sub
```

```
Sub Register_Click ()
    If fileisok Then
        reg_code$ = regcode.Text
        cfg_name$ = Cfgname.Text
        If reg_code$ > "" Then
            a = cpRegisterProgram(cfg_name$, reg_code$)
        End If
    End If
    Checkmessage.Text = ""
    Codecheck.Text = ""
End Sub
```

```
Sub CheckCode_Click ()
    If fileisok Then
        code$ = ""
        cfg_name$ = Cfgname.Text
        a = cpCheckRegCode(cfg_name$, code$)
        If a = 0 Or a = WRONG_CODE Then
            t$ = "Regcode incorrect!"
        ElseIf a < 0 Then
            t$ = "Fileproblems!"
        Else
            t$ = "Access " + Str$(a) + " registered"
        End If
    Else
        t$ = " No file! "
    End If
    Codecheck.Text = t$
End Sub
```

```
Sub Start_Click ()
    Thour.Text = ""
    Tmin.Text = ""
    Tsec.Text = ""
    Chour.Text = ""
    CMin.Text = ""
    Csec.Text = ""
    Ttimes.Text = ""
    If fileisok Then
        cfg_name$ = Cfgname.Text
        a = cpStartProgram(cfg_name$)
    End If
End Sub
```

```
Sub Stopnow_Click ()
Dim times As Integer, t_hour As Integer, t_min As Integer, t_sec As Integer,
    c_hour As Integer, c_min As Integer, c_sec As Integer
    If Ttimes.Text = "" And fileisok Then
        cfg_name$ = Cfgname.Text
        a = cpEndProgram(cfg_name$)
        a = cpGetUStatistics(cfg_name$, times, t_hour, t_min, t_sec, c_hour,
            c_min, c_sec)
        Thour.Text = Str$(t_hour)
        Tmin.Text = Str$(t_min)
        Tsec.Text = Str$(t_sec)
        Chour.Text = Str$(c_hour)
        CMin.Text = Str$(c_min)
        Csec.Text = Str$(c_sec)
        Ttimes.Text = Str$(times)
    End If
End Sub
```

```
Sub Goodbye_Click ()
    Unload Form1
End Sub
```

What you need

cprotxxd.dll	
cprotxxu.dll	
cprotxx.h	if you program in C/C++
cprotxx.inc	if you program with Visual Basic / Access

cpCFGMake

Prototype

INT16 cpCFGMake(char * path, unsigned long magic, unsigned long filever)

Function cpCFGMake (ByVal path$, ByVal magic&, ByVal filever&) As Integer

Description

This function creates your security file, assigns a unique serial number and writes it into the file. You can also define a "magic number" for verification and a file version for information purpose. This is the file that will contain all the candid information for verification, registration etc.

Parameters

path a string containing a valid filename (the name of the security file) with or without full path.

magic a 32-bit integer which you can use to verify that this actually is the correct file (if there happens to be another file with the name of your file...) If you set it to 0 then it will default to 0x3539504C.

filever a 32-bit integer denoting fileversion in case a later version of your program might need that information. If you set it to 0 then it will be dafaulted to 0x 00000200.

Return value

0 if successful

PATH_NOT_VALID if ***path*** points to a not exsisting directory or the filename portion is invalid.

C/C++ example

Visual Basic / Access Example

cpMakeRegCode

Prototype

INT16 cpMakeRegCode(INT16 access, char * serialnumber, char * regcode)

Function cpMakeRegCode (ByVal access%, ByVal serialnum$, ByVal regcode$) As Integer

Description

THIS FUNCTION ONLY in cprotXXd.dll

This is where the registration code is created. Because your software protection will become worthless if your user gets hold of this routine, of course you will have to ship a version of this DLL that does not contain this routin, cprotXXu.dll.

The apropriate registration code is written to ***regcode*** on return from the function.

Parameters

access a 16-bit integer in the range 1 - 15.

serialnumber a string containing the users serialnumber.

regcode an initialized string, minimum 14 characters long.

Return value

0 if successful

a **positiv integer** if ***access*** is outside limits 1 - 15 or if no ***seralnumber*** is supplied.

Visual Basic / Access Example

cpGetFileinfo

Prototype

INT32 cpGetFileinfo (char * path, INT32 * version)

Function cpGetFileinfo (ByVal path$, version&) As Long

Description

Gets the magic filenumber and fileversion.

On return **version** will contain the value of your fileversion.

Parameters

path a string containing a valid filename (the name of the security file) with or without full path

version an initialized 32-bit signed integer (usaually long).

Return value

the **magic number** to be able to ascertain that it really is your file.

FILE_NOT_FOUND if ***path*** is not valid.

Visual Basic / Access Example

cpGetSernum

Prototype

INT16 cpGetSernum(char * path, char * serialnumber)

Function cpGetSernum (ByVal path$, ByVal serialnum$) As Integer

Description

Returns the serialnumber in **serialnumber.**

Parameters

path a string containing a valid filename (the name of the security file) with or without full path

serialnumber an initialized string, minimum 14 characters long.

Return value

0 if successful

FILE_NOT_FOUND if ***path*** is not valid

C/C++ Example

Visual Basic / Access Example

cpCheckRegCode

Prototype

INT16 cpCheckRegCode(char * path, char * regcode)

Function cpCheckRegCode (ByVal path$, ByVal regcode$) As Integer

Description

Checks if ***regcode*** is valid.

1. If you supply an empty string (**NB** NOT a NULLpointer in c/c++) in ***regcode*** then check is made that the registration code in the security file is correct. This is the way to use it when the user starts your program - the returned integer will tell the access registered and your program will know what actions to take.
2. If you do supply a regcode in ***regcode*** then this code is checked against the serial number in the security file. This is the way to use it when the user is inputting the regcode you supply, i.e. when he is registering the program.

Parameters

path a string containing a valid filename (the name of the security file) with or without full path

regcode a string containing the registration code or an empty string.

Return value

access code if the regcode, supplied or in the security file, is correct

WRONG_CODE if the regcode is incorrect, either the one you supplied or, if **regcode** was empty, the regcode in the security file.

FILE_NOT_FOUND if ***path*** is not valid

C/C++ Example

Visual Basic / Access Example

cpRegisterProgram

Prototype

INT16 cpRegisterProgram (char * path, char * regcode)

Function cpRegisterProgram (ByVal path$, ByVal regcode$) As Integer

Description

Writes the registration code to the security file.

NB It is your responsibility to check that ***regcode*** is valid, use cpCheckRegCode( ).

Parameters

path a string containing a valid filename (the name of the security file) with or without full path

regcode a string containing the registration code

Return value

0 if successful

FILE_NOT_FOUND if ***path*** is not valid

Visual Basic / Access Example

cpSetRegisteredNames

Prototype

INT16 cpSetRegisteredNames (char * path, char * name, char * company)

Function cpSetRegisteredNames (ByVal path$, ByVal name$, ByVal company$) As Integer

Description

Writes name and company to the security file.

Parameters

path a string containing a valid filename (the name of the security file) with or without full path

name a string containing the user´s name, maximum 27 characters will be written.

company a string containing the users company or an empty string, maximum 27 characters will be written.

Return value

0 if successful

FILE_NOT_FOUND if ***path*** is not valid

Visual Basic / Access Example

cpGetRegisteredName

Prototype

INT16 cpGetRegisteredName (char* path, char * name)

Function cpGetRegisteredName (ByVal path$, ByVal name$) As Integer

Description

Returns the user´s name in ***name*** on return.

Parameters

path a string containing a valid filename (the name of the security file) with or without full path

name an initialised 28 characters long string.

Return value

0 if successful.

FILE_NOT_FOUND if ***path*** is not valid.

Visual Basic / Access Example

cpGetRegisteredCompany

Prototype

INT16 cpGetRegisteredCompany (char* path, char * company)

Function cpGetRegisteredCompany (ByVal path$, ByVal company$) As Integer

Description

Returns the user´s company in ***company*** on return.

Parameters

path a string containing a valid filename (the name of the security file) with or without full path.

company an initialised 28 characters long string.

Return value

0 if successful.

FILE_NOT_FOUND if ***path*** is not valid.

Visual Basic / Access Example

cpIsRegistered

Prototype

INT16 cpIsRegistered(char * path)

Function cpIsRegistered (ByVal path$) As Integer

Description

Checks if a regcode exists in the security file. Useful for checking if a registered version already exists upon installation. Naturally you will have to find all (if any) occurrences of your security file on the users hard disks.

Parameters

path a string containing a valid filename (the name of the security file) with or without full path.

Return value

0 if a regcode is found.

FILE_NOT_FOUND if ***path*** is not valid.

a **positive integer** if no regcode is found in the security file.

C/C++ Example

Visual Basic / Access Example

cpStartProgram

Prototype

INT16 cpStartProgram (char * path)

Function cpStartProgram (ByVal path$) As Integer

Description

Starts the program timer and increments the times_used counter. Should be used immediately after program start.

Parameters

path a string containing a valid filename (the name of the security file) with or without full path.

Return value

0 if successful.

FILE_NOT_FOUND if ***path*** is not valid.

C/C++ Example

Visual Basic / Access Example

cpEndProgram

Prototype

INT16 cpEndProgram (char * path)

Function cpEndProgram (ByVal path$) As Integer

Description

Stops the program timer, calculates total usage and usage this time and writes theese to the security file.

NB Please remember to start the program timer with cpStartProgram before you use this function - or cpGetUStatistics will return incorrect time-values.

Parameters

path a string containing a valid filename (the name of the security file) with or without full path.

Return value

0 if successful.

FILE_NOT_FOUND if ***path*** is not valid.

C/C++ Example

Visual Basic / Access Example

cpGetUStatistics

Prototype

INT16 cpGetUStatistics (char * path, UINT16 * utimes, UINT16 *totalhour, UINT16 *totalmin, UINT16 *totalsec, UINT16 *lasthour, UINT16 *lastmin, UINT16 *lastsec **)**

Function cpGetUStatistics (ByVal path$, ByVal utimes%, ByVal totalhour%, ByVal totalmin%, ByVal totalsec%, ByVal lasthour%, ByVal lastmin%, ByVal lastsec%**) As Integer**

Description

Returns usage statistics in the supplied parameters. Useful only if also cpStartProgram( ) and cpEndProgram( ) are used.

Parameters

path a string containing a valid filename (the name of the security file) with or without full path.

utimes a 16-bit initialised integer that will contain number of times the program was started.

totalhour, totalmin, totalsec 16-bit initialised integers that will contain the hours, minutes and seconds of usage totally.

lasthour, lastmin, lastsec 16-bit initialised integers that will contain the hours, minutes and seconds of last time used.

Return value

0 if successful.

FILE_NOT_FOUND if ***path*** is not valid.

C/C++ Example

Visual Basic / Access Example

cpSetUParams

Prototype

INT16 cpSetUParams (char* path, UINT32 paraml, INT16 param1, INT16 param2, INT16 param3, INT16 param4, UINT16 param5)

Function cpSetUParams (ByVal path$, lparam&, iparam1%, iparam2%, iparam3%, iparam4%, iparam5%) As Integer

Description

Writes additional parameters to the security file.

Parameters

path a string containing a valid filename (the name of the security file) with or without full path.

lparam a 32-bit integer.

iparam1, iparam2, iparam3, iparam4, iparam5 16-bit integers.

Return value

0 if successful.

FILE_NOT_FOUND if ***path*** is not valid.

cpGetUParams

Prototype

INT16 cpGetUParams(char* path, UINT32 *paraml, INT16 *param1, INT16 *param2, INT16 *param3, INT16 *param4, UINT16 *param5)

Function cpGetUParams (ByVal path$, ByVal lparam&, ByVal iparam1%, ByVal iparam2%, ByVal iparam3%, ByVal iparam4%, ByVal iparam5%) As Integer

Description

Retreives additional parameters stored in the security file.

Parameters

path a string containing a valid filename (the name of the security file) with or without full path.

lparam a 32-bit initialised integer.

iparam1, iparam2, iparam3, iparam4, iparam5 16-bit initialised integers.

Return value

0 if successful.

FILE_NOT_FOUND if ***path*** is not valid.

cpSetFTime

Prototype

INT16 cpSetFTime (char * path, UINT32 date, UINT16 time);

Function cpSetFTime (ByVal path$, ByVal date&, ByVal time%) As Integer

Description

Sets the time stamp of your security file. Sometimes it may be desirable not to be very obvious about which file you are writing to.... Since all the above functions actually open and close the security file this must be the last function you use on the security file!

Parameters

path a string containing a valid filename with or without full path.

date a 32-bit integer containing the desired date in the form YYMMDD

time a 16-bit integer containing the desired time in the form HHMM (the seconds will always be set to 0).

Return value

0 if successful.

FILE_NOT_FOUND if ***path*** is not valid.

C/C++ Example

cpWhichDrive

Prototype

INT16 cpWhichDrive (char * file);

Function cpWhichDrive (ByVal file$) As Integer

Description

Checks if ***file*** is on a disk in drive A. or drive B: and if the diskette is writeprotected. Useful if you want to write some information to the installation diskette during installation, maybe something that could be useful if the user wants to reinstall...

Parameters

file a string containing the name of the file you are looking for, **NB** no path should be specified!.

Return value

NO_ARGUMENTS if you supply an empty string

DRIVE_NOT_FLOPPY ??? can A: or B: be anything else ??? in that case this will result

DRIVE_NOT_READY forgot to close the door?

WRONG_DISKETTE *file* not on the diskette(s)

DRIVE_WRPROT_A *file* found on disk in drive A: disk is write-protected

DRIVE_WRPROT_B *file* found on disk in drive B: disk is write-protected

DISK_ISIN_A *file* found on disk in drive A: disk is not write-protected

DISK_ISIN_B *file* found on disk in drive B: disk is not write-protected

cpDirRemove

Prototype

INT16 cpDirRemove (char * path)

Function cpDirRemove (ByVal path$) As Integer

Description

Removes an entire directory with contents, as long as no file is marked SYSTEM or READ_ONLY. This might be useful during installation.

Parameters

path a string containing a valid full pathname.

Return value

0 if successful.

NO_PATH if you forgot to supply a path.

PATH_NOT_VALID if ***path*** is not valid.