String Studio

String studio is a MFC CString class extension and a set of routines to simplify the MFC programming. Now it includes about 150 (useful) functions for Win16 and Win32.

HEXANET

String Studio 3.0

COPYRIGHT HEXANET 1993-1995 - All rights reserved. All trades Marks are deposed by their owners. This file must not be distributed without the full String Studio 3.0 package.

THIS SOFTWARE AND DOCUMENTATION ARE SOLD "AS IS" AND WITHOUT WARRANTIES AS TO PERFORMANCE OF MERCHANTABILITY OR ANY OTHER WARRANTIES WHETHER EXPRESSED OR IMPLIED. BECAUSE OF THE VARIOUS HARDWARE AND SOFTWARE ENVIRONMENTS INTO WHICH THIS PROGRAM MAY BE PUT, NO WARRANTY OF FITNESS FOR A PARTICULAR PURPOSE IS OFFERED.
GOOD DATA PROCESSING PROCEDURE DICTATES THAT ANY PROGRAM BE THOROUGHLY TESTED WITH NON-CRITICAL DATA BEFORE RELYING ON IT. THE USER MUST ASSUME THE ENTIRE RISK OF USING THE PROGRAM.

Registration...

Versions...

The unregistered version will display the shareware principles reminders. The registered version is the version you will build with the String Studio MFC 2.5 (or 3.x for the Win32 version) source code. Of course this version removes the nag screens and let you build customised version (a _DEBUG version for instance). To get the String Studio source code (i.e. CSTR30A.CPP and CSTR30B.CPP), you must register then we will send you the required password to unzip the encrypted ZIP file (STR_SRC3.ZIP). All our 'Studio' sharewares are royalty-free.

Registration...

String Studio is part of our 'Full programmer' s pack' which includes:

- **Zip Studio API (>= 2.x),**
- **Zip Studio Shell**
- **Zip Studio Shell MFC Source Code**
- **VBX Studio**
- **Setup Studio**
- **And this 'String Studio' toolkit.**

This bundle price is 729- FF (or 135- US$, 215- DM or 95- GBP). All these toolkits are available on many public networks like Compuserve (go WINSHARE and search for 'STUDIO' sharewares, CICA ftp server, that is winftp.cica.indiana.edu, the new Microsoft Network and so on..). You may register directly to us (by Credit Card or US$, FRF or DEM check) or via PsL. With your registration you will receive the passwords via email (if you have an email address) and snail mail. We always send out a printed invoice. In most cases, orders are processed in 1 or 2 days.

If you wish to register directly to us (which is the fastest way), see the Order Form of this help file.

Ä If you prefer to order the full pack thanks to PsL, these are infos from PsL to register:
You can order with MC, Visa, or American Express from the Public (software) Library (PsL) by calling 1-800-2424-PSL or FAXing 1-713-524-6398. These numbers are for orders only. For questions about credit card orders, call PsL at 713-524-6394. You can also mail credit card orders to PsL at P.O.Box 35705, Houston, TX 77235-5705.

How to build the registered version?

When you register we will send you the required password to unzip the STR_SRC3.ZIP encrypted ZIP file. This file doesn' t include the registered LIB files: If choose to use String Studio as you did with the unregistered LIB versions you must rebuild the Registered LIB file(s). However, we found some problems with the static LIB version with VC++ 1.5X. The QuickWatch window doesn' t display the string values as it should do. Also, if you use the source files, you are able to include/remove some components of String Studio which is not available with LIB versions. So, we think it' s better to include the sources files into your projects rather than using the LIB versions. See Tips for details.

Ä To compile String Studiio, load MSVC.EXE then build a new project or open an existing MFC MAK file Then define or undefine _WINDOWS and/or _WIN32, and don' t forget to define _REGISTERED before you compile a String Studio file. If you plan to use a LIB version, remember to define _REGISTERED in all your MFC works. Of course you can delete the unregistered constructors to avoid to define _REGISTERED in your new projects.With this version, you can choose which components (FILE, TIME, INI, WINCTL...) you want to use in order to reduce the code size.

Upgrade...

Upgrade from String Studio 2.x is absolutely free. If you already own a 'Studio' shareware, there is a 300- FF bonus when you register directly to us. In this case, please don' t forget to send us your user code with the Order Form. Also, we don' t plan to ask for upgrade fees for the future versions.

Technical support...

Your comments or questions are welcome. You can join us on the Internet (hnet@dialup.francenet.fr) or on COMPUSERVE (100333,27): We check the mail every day. If you can' t access these networks, you may send us a request via snail mail, see at the bottom of this page.

About the String Studio license...

The license is an using authorization but not a product cession. Programs made with the unregistered version musn' t be distributed. To distribute your programs using String Studio, you must register to HEXANET before. Then you will build (and customize) your own String Studio Library without the shareware dialog box displayed when you create your first CStr object. You can use this product on several PC but no more than one unique copy can be used at the same time. As a conclusion you are not allowed to distribute a String library based on the String Studio toolkit...

Others Studio components...

- **Zip Studio API and Shell**

  Zip Studio is our famous Zip/Unzip API and Shell specifically designed for Windows. Current version is 2.56, see ZS25A.ZIP, ZS25B.ZIP (or ZSAPI25A.ZIP and ZSAPI25B.ZIP) and Z256UP.ZIP. The Win32/Win95 version will be available in some weeks.

- **Setup Studio**

  A powerful toolkit to create professional quality Setup programs for Windows 3.10/Win95. The 2.5 version includes templates for VB and VC++. Current version is 2.5 (see SSETUP25.ZIP).

- **VBX Studio**

  VBX Studio is a set of VBXs to build nice programs in a very short time. Also, they may be useful to port your Win32 programs to Win31. Current version is 1.2b (see VBXSTD12.ZIP).

HEXANET

HEXANET SW
BP 385.16
75768 PARIS CEDEX 16
FRANCE

EMail: hnet@dialup.francenet.fr
CIS: 100333,27
Fax(orders): (33-1) 40.72.80.77

Package

<Deleted>

Shareware?

String Studio is ShareWare:

You can try it as you want before you decide or not to register. The only one way we have to sell enough licenses to support this software, is to distribute it all over the world. So, if you think this product will help one of your friends, don' t hesitate to give him a (complete) copy of this product: This is your interest!

Ä When you distribute String Studio, don' t change any file and give the complete package. If you upload String Studio on a BBS, please name it CSTR30.ZIP or CSTR30.EXE.

Obviously, if you register, you can' t distribute your registered CPP files! Distributing programs made with String Studio unregistered version is not allowed.

Notes for vendors...

Feel free to distribute one or more 'Studio' sharewares on your next book/CD-ROM. We can' t send authorizations to all people who want to do this.

HEXANET

HEXANET SW
BP 385.16
75768 PARIS CEDEX 16
FRANCE

EMail: hnet@dialup.francenet.fr
CIS: 100333,27
Fax(orders): (33-1) 40.72.80.77

Terms

Used terms are C and SDK language type. If you do not develop with these languages, perhaps you will have some difficulties to understand them. These are some explanations:

int	Integer value.
long	Long integer value.
UINT	Signed long integer value.
BOOL	Integer value, 0 for FALSE, anything else for TRUE.
char	Character.
char far*	String.
LPSTR	String.
LPTSTR	
LPCSTR	Constant String.
LPCTSTR	
HWND	Window' s Handle(Integer value).
HINSTANCE	Application' s Handle(Integer value).
void	Null type (empty).
WINAPI	FAR PASCAL.
COLORREF	Long integer value (Colour).

Overview

String Studio advantages...

With String Studio you will find that handling strings with Microsoft Visual C++ 1.x/2.x is as easy as (even more) you can do with VB. You don' t have to re-invent the wheel each time you build a new program and you can safely port your Win31 apps to Win32. Also, with this version you have more than a single string class: You will find many useful global functions, a CTime and a CFFind class extension. String Studio 3.0 is part of our 'Full programmer' s pack' (US$ 135-). Free upgrade from 2.x and there is no royalties.

Ä To use the CStr class (i.e. the main String Studio class), you will need the MFC CString implementation (defined in <afx.h>). To know all the required include files, see STDAFX.H. Now, to uses CStr (and others String Studio classes or functions), add '#include "cstr30.h"' at the end of your project STDAFX.H file, then link with CSTR30W.LIB (for Win16) or, CSTR30NT.LIB (for Win32). Now, replace all your CString objects with CStr. Then compile. To have a look at all functions, just point on a CStr object and press the [F11] key or have a look at this HLP file. See Tips for details.

Features?

CStr class..

Trim:	*Trim, TrimBegin, TrimEnd*
Find:	*Count, FindWord, FindWordInMultilineBuffer*
Input:	*From, FromCurrentTime, FromFileTime, FromRes*
Output:	*ToInt, ToUINT, ToLong, ToULong, ToFloat, ToDouble, ToTime*
Modifications:	*Delete, Replace, Insert, Extract, ExtractToBegin, ExtractToEnd*
Improvments:	*Operator(), GetBuffer*
Numbers:	*Round, IsDigit, IsInRange, NumFormat, NumUnformat*
Format:	*FirstUpper, AddLF, Align*
Misc.:	*Blank, Replicate, CheckSize, CheckDirSize, sprintf*
Ini files:	*FromIni, ToIni*
Text files:	*FromFile, ToFile*
File names:	*IsFileName, IsPathName, IsFullFileName, GetPathValue, GetFileNameValue, CheckFullFileName, CheckFileExtension, DoesFileExist, DoesDirExist, GetTempFileName, IsUNC, FilenameToUNC, UNCToFilename*
Files:	*GetFileAttributes, SetFileAttributes, GetFileSize, DeleteFile, IsWriteable, GetDiskFreeSpace, CreateFile, CopyTo, RemoveDir, IsRDOnly*
Directories:	*ChDir, MkDir, GetAppDir, GetWinDir, GetWinSysDir, GetThisDir, GetTempDir*
Characters infos.:	*IsCharUpper, IsCharLower, IsCharAlpha, IsCharAlphaNum, IsNumeric, IsSpace*
Characters modif.:	*MakeCharUpper, MakeCharLower*
GDI:	*GetTextWidth, GetTextHeight, TabbedTextOut*
Windows specific:	*GetWindowTitle, GetClassName, GetWinfileAssociation*
Windows interface:	*ToCtl, FromCtl, MB, GetDlgItemText*

CTim class..

Build:	*Build, FromFile*
Modifications:	*SetDay, SetMonth, SetYear, SetHour, SetMin, SetHour, IncDay, IncSec, IncMin, IncHour, ToFile*
Infos:	*IsLeapYear, IsLastDay, IsValid*

CMem class..

Actions:	*CMem, Alloc, ReAlloc*
Infos:	*GetPtr, GetHandle, GetSize, IsValid*

CFFind class..

Actions:	*FindFirst, FindNext, Find, SetPath, Reset*
Infos:	*GetCount, GetAttributes, GetCreationTime, GetLastAccessTime, GetLastWriteTime, GetSize, GetShortName, GetLongName, GetFullShortName, GetFullLongName*

Global routines..

Infos:	*AfxGetFreeResRate, AfxGetFreeResValue, AfxHasWin4Look*
User' s config:	*AfxGetPrecision, AfxGetDecimalPoint, AfxGetThousandsSeparator, AfxGetShortDateFormat, AfxGetLongDateFormat*
CTL3D wrappers:	*AfxCtl3dLoad, AfxCtl3dUnload, AfxCtl3dNewColors, AfxCtl3dSubclassCtl*
Windows coord.:	*AfxWriteWindowPos, AfxGetWindowPos, AfxCenterWindow*
Dialog controls management:	*AfxEnableDlgItem, AfxEnableOK, AfxEnableCANCEL, AfxSetFocus, AfxSetDefButton, AfxActivateEdit*
Misc:	*AfxSleep, AfxGiveTheHand, AfxSetFileTime, AfxTouchFile, AfxSetFileAttributes, AfxFindWordInFile, AfxCheckFileName, AfxIsDateValid, AfxRoundToInt, AfxRound*

Of course, all CString functions are available for CStr objects! Also, the CStr class is very easy to use because it' s quite similar to the CString class but there is one main difference with CStr: Generaly when you call a CStr function, the current CStr object (this) is changed.

Warning

Ä The LIB files were compiled with VC++ 1.52 (2.52 for the 32bits version) large model without debugging informations. If you use _AFXDLL or _USRDLL, some CTime functions won' t be available any more. If you' ve registered String Studio, define _REGISTERED before you include CSTR30.H (in this case you must use the source code instead of the LIB file). The Win16 version is for LARGE model and it uses FAR strings. String Studio should not be used as a DLL. See Tips for details.

String Studio components

To reduce the size of the toolkit, we included some DEFINES at the top of CSTR30.H. This let you choose which parts of String Studio you want to use for your project. To do this, add a comment or uncomment a DEFINE. You need the source code to do it. Also you may find some incompatibilities when a necessary group is required but missing. The groups are the followings:

```
#define _CSTR30_HASMOREBASIC     -> additional functions
#define _CSTR30_HASFILE          -> all files/dirs function
#define _CSTR30_HASINI           -> All functions which use INI files
#define _CSTR30_HASWINCTL-> Dialog box controls functions
#define _CSTR30_HASNUM           -> Math functions
#define _CSTR30_HASCTIM          -> CTim class and Time functions
#define _CSTR30_HASCFFIND        -> CFFind
#define _CSTR30_HASUNC           -> UNC file names
#define _CSTR30_HASCTL3D         -> AfxCtl3d.. functions
#define _CSTR30_HASFINDWORD      -> FindWord functions
#define _CSTR30_HASCHAR          -> Character oriented functions
```

All other functions (group <none>) are standard functions (like Extract, Count..). They are required for most of the functions of String Studio.

CStr values

One nice feature of String Studio is that you can use functions that are specifically designed for a kind of data (specifically file names, times and numbers functions). In order to simplify the code, this kind of method will use the current CStr string to perform the required action. So, you must make sure your CStr object contains the required type of data. For instance, use:

```
CStr strMyFile( "C:\MYFILE.TXT")
if ( strMyFile.DoesFileExist() )...
```

but

```
CStr strMyFile( "C:\MYFILE.TXT")
CTime ctMyTime();
if ( (ctMyTime = strMyFile.ToTime()) )...
```

has no sense...

Registration

Have a look at the registration chapter.

See also...

license

HEXANET

HEXANET SW
BP 385.16
75768 PARIS CEDEX 16
FRANCE

EMail: hnet@dialup.francenet.fr
CIS: 100333,27
Fax(orders): (33-1) 40.72.80.77

CStr functions

How to use the CStr?

Before you can use a CStr function, you must create a CStr object (like you do with CString), then you apply the function to this object. CStr functions are CStr members (or methods)...

Example

```
CStr szMyString;
szMyString.FromCtl( this, IDC_STATICTITLE );
szMyString.MB();
```

ü Trim...

Trim, TrimEnd, TrimBegin	Trim the current CStr. [3.0]

ü Find...

Count	Count substrings in the current CStr.
FindWord	Find a word in the current string.
FindWordInMultilineBuffer	Find a word in a multiline string.
Count	Count substrings in the current CStr.
FindWord	Find a word in the current string.
FindWordInMultilineBuffer	Find a word in a multiline string.

ü Input...

From	Copy data into the current CStr.
FromCurrentTime	Copy the current date/Time into the current CStr.
FromFileTime	Fill the current CStr with a file date/time.

ü Output...

ToInt, ToUINT, ToLong, ToULong, ToFloat, ToDouble	Convert the current CStr string into a number.
ToTime	Retrieve a CTime object from the current CStr.

ü Modifications...

Delete	Delete a current CStr part
Replace	Replace all substrings in the current CStr.
Insert	Insert a substring in the current CStr.
Extract	Extract an item of the current CStr.
ExtractToBegin	Extract the items at the beginning of the string.
ExtractToEnd	Extract the items at the end of the string.

ü Improvments...

operator()	Modify a character in the current CStr.
GetBuffer	Return a LPTSTR buffer for the current CStr.
CStr	New CStr contructor using a resource ID.

ü Numbers...

Round	Round a numerical value in the current CStr.
IsDigit	Test for a numerical value in the current

ü Format...

FirstUpper	Convert the first character into uppercase.
AddLF	Add a CRLF to the current CStr.

	CStr.		
IsInRange	Check if the numerical value is in the range.	Align	Change the string datas alignment.
NumFormat	Format a number using a pattern.		
NumUnformat	Remove unwanted characters from a number.		
ü Miscellaneous...		**ü INI files...**	
Blank	Replace all characters by a space.	FromIni	Fill the current CStr with an INI file value.
Replicate	Create a string for the current CStr.	ToIni	Write the current CStr value to an INI file.
CheckSize	Check the length of the current CStr.		
CheckDirSize	Check the length of the dir. in the current CStr.	**ü Text files...**	
sprintf	Similar to wvsprintf/vsprintf. [3.0]	FromFile	Loads a text file into the current CStr.
		ToFile	Puts the current CStr value into a text file.
ü File names...		**ü File attributes...**	
GetPathValue	Return a file name path.	GetFileAttributes	Retrieve the attributes of the file. [3.0]
GetFileNameValue	Return a file name value (without its path).	SetFileAttributes	Change the file attributes. [3.0]
CheckFullFileName	Check (and change) all the file name components.	IsRDOnly	TRUE if the file is read-only. [3.0]
CheckFileExtension	Check the file extension.		
GetTempFileName	Returns a temporary file name.	**ü File operations...**	
IsFileName	TRUE if the current CStr value is a file name.	DoesFileExist	Returns TRUE if the specified file exists.
IsFullFileName	Check all the file specification components.	GetFileSize	Retrieve the size of the file. [3.0]
IsPathName	TRUE if the currrent CStr value is a dir. name.	DeleteFile	Delete the file. [3.0]
IsUNC	Returns TRUE if the filename is an UNC name. [3.0]	IsWriteable	Returns TRUE if the file can be written. [3.0]
		GetDiskFreeSpace	Retrieve the hard disk free space. [3.0]
		CopyTo	Copy the current file. {3.0]
		CreateFile	Creates a new file. [3.0]
ü Directories...		**ü Characters...**	
ChDir	Go to the specified directory.	IsCharUpper	Return TRUE if specified character is

			in uppercase.
MkDir	Build the specified directory.	IsCharLower	Return TRUE if specified character is in lowercase.
DoesDirExist	Return TRUE if the specified dir. exists.	IsCharAlpha	Return TRUE if specified character is alphabetic.
GetAppDir	Return the Windows application directory.	IsCharAlphaNum	Return TRUE if specified character is alpha or num.
GetWinDir	Return the WINDOWS directory.	IsCharNumeric	Return TRUE if specified character is a digit.
GetWinSysDir	Return the SYSTEM directory.	IsSpace	Return TRUE if specified character is ithe space c..
GetThisDir	Return the current directory.	MakeCharUpper	Convert a character in uppercase.
GetTempDir	Return the temporary directory name.	MakeCharLower	Convert a character in lowercase.
RemoveDir	Delete one or more directories. [3.0]		

ü Windows...

FromRes	Load a string from resources.
GetWindowTitle	Return a window title. [3.0]
GetClassName	Retrieve the class name of a window.
GetWinfileAssociation	Return the associated program name for a document extension.
FromCtl	Retrieve the text of an Windows control.
ToCtl	Put the current CStr value into a Dialog control.
MB	Display the current CStr value into an MessageBox.
GetDlgItemText	Retrieve the caption of a control.

ü GDI...

GetTextWidth	Return the string width in logical units
GetTextHeight	Return the string height in logical units.
TabbedTextOut	Display the current string

See also...

- CTim functions
- CFFind functions
- CMem functions
- Afx... functions

CStr::Trim

CString CStr::Trim(void); [3.0]

Description
Trims a string (delete spaces, tabs and CR-LF characters).CHANGES THE CSTR OBJECT

Compatibility
WIN16, WIN32, WINDLL

Group
<none>

Example
```
CStr szMyString("  ABC   ");
szMyString.Trim();
szMyString.MB();
```

See also...
CStr::TrimEnd
CStr::TrimBegin
CStr::Replace
CStr::FirstUpper
CStr::Blank

CStr::TrimEnd

CString CStr::TrimBegin(void); [3.0]

Description
Trims the begining of a string (delete spaces, tabs and CR-LF characters).CHANGES THE CSTR OBJECT

Compatibility
WIN16, WIN32, WINDLL

Group
<none>

Example
```
CStr szMyString("  ABC   ");
szMyString.TrimBegin();
szMyString.MB();
```

See also...
CStr::Trim
CStr::TrimBegin
CStr::Replace
CStr::FirstUpper
CStr::Blank

CStr::TrimBegin

CString CStr::TrimEnd(void); [3.0]

Description
Trims the end of a string (delete spaces, tabs and CR-LF characters).CHANGES THE CSTR OBJECT

Compatibility
```
WIN16, WIN32, WINDLL
```

Group
```
<none>
```

Example
```
CStr szMyString("  ABC   ");
szMyString.TrimEnd();
szMyString.MB();
```

See also...
CStr::Trim
CStr::TrimEnd
CStr::Replace
CStr::FirstUpper
CStr::Blank

CStr::Count

unsigned int CStr::Count(LPCTSTR szSubString) const;
unsigned int CStr::Count(char szSubString) const;

Description

Returns the number of a char or a string in the current CStr object .

Compatibility

WIN16, WIN32, WINDLL

Group

<none>

Example

```
CStr szMyString("ABA"), szMsg;
int a = 0;
a = szMyString.Count("A");
szMsg.From( (int)a );
szMsg.MB();
```

CStr::From

CString CStr::From(long data, int base = 10, LPCTSTR szSeparator = ",");
CString CStr::From(unsigned long data, int base = 10, LPCTSTR szSeparator = ",");
CString CStr::From(int data, int base = 10, LPCTSTR szSeparator = ",");
CString CStr::From(unsigned int data, int base = 10, LPCTSTR szSeparator = ",");
CString CStr::From(double data, int precision = 2, LPCTSTR szSeparator = ",", LPCTSTR szDecimalPoint = ".");
CString CStr::From(CTime& TimeValue, LPCTSTR szFilter = "%d/%m/%y"); [3.0]

Description

- **From long, unsigned long, int and unsigned int:**
Translates a numerical value specified with <data> to a string, <base> is 10 or 16 and <szSeparator> indicates the character to use as a thousands separator ("," is highly recommended for translations).
- **From double**:
Translates a double value specified with <data> to a string, <precison> is the number of decimal digits and <szSeparator> indicates the character to use as a thousands separator ("," is highly recommended for translations) and <szDecimalPoint> is the character to use as decimal point.
- **From CTime:**
Translates a CTime object specified with <TimeValue> to a string using <szFilter> mask. see CTime.Format for possible values for <szFilter>.CHANGES THE CURRENT CSTR OBJECT.

Compatibility

```
WIN16, WIN32, WINDLL (but From(CTime))
```

Group

```
_CSTR30_HASNUM
_CSTR30_HASMOREBASIC ( From(CTime))
```

Example

```
CStr szMyString("ABA"), szMsg;
int a = 0;
a = szMyString.Count("A");
szMsg.From( (int)a );
szMsg.MB();
```

See also...

CStr::FromCurrentTime
CStr::FromFileTime
CStr::ToInt
CStr::ToUINT
CStr::ToLong
CStr::ToULong
CStr::ToFloat
CStr::ToDouble
CStr::ToTime
CStr::IsInRange
CStr::FromIni
CStr::FromFile
CStr::FromCtl
CTim::ToFile
CTim::FromFile

CStr::FromCurrentTime

CString CStr::FromCurrentTime(LPCTSTR szFilter = "%d/%m/%y");

Description

Retrieves the current date/time and puts it into the current CStr object using <szFilter>.
See CTime::Format for possible values for <szFilter>.CHANGES THE CURRENT CSTR OBJECT.

Compatibility

WIN16, WIN32

Group

_CSTR30_HASMOREBASIC

Example

```
CStr szMsg;
szMsg.FromCurrentTime();
szMsg.MB();
```

See also...

CStr::From
CStr::FromFileTime
CStr::ToTime
CStr::FromIni
CTim::ToFile
CTim::FromFile

CStr::FromFileTime

CString CStr::FromFileTime(LPCTSTR szFile, LPCTSTR szFilter = "%d/%m/%y");

Description

Retrieves the file date/time specified with <szFile> and puts it into the current CStr object using <szFilter>. See CTime::Format for possible values for <szFilter>.CHANGES THE CURRENT CSTR OBJECT.

Compatibility

WIN16, WIN32

Group

_CSTR30_HASMOREBASIC
_CSTR30_HASCTIM

Example

```
CStr szMsg;
szMsg.FromFileTime("C:\\WINDOWS\\WIN.INI");
szMsg.MB();
```

See also...

CStr::From
CStr::FromCurrentTime
CStr::ToTime
CStr::FromFile
CStr::GetFileAttributes
CStr::SetFileAttributes
CStr::GetFileSize
CStr::IsRDOnly
CTim::ToFile
CTim::FromFile
AfxSetFileTime
AfxTouchFile
AfxSetFileAttributes

CStr::ToInt

signed int CStr::ToInt(LPCTSTR szSeparator = ",") const;

Description
Returns the value of the current CStr object as a signed integer or 0 if CStr doesn' t correspond to a number.

Compatibility
WIN16, WIN32

Group
_CSTR30_HASNUM

Example
```
int iValue = 125, iTest = 0;
CStr szMsg;
szMsg.From( (int)iValue );
iTest = szMsg.ToInt();
szMsg.From( (int)iTest );
szMsg.MB();
```

See also...
CStr::From
CStr::ToUINT
CStr::ToLong
CStr::ToULong
CStr::ToFloat
CStr::ToDouble
CStr::ToTime
CStr::Round
CStr::IsDigit
CStr::IsInRange

CStr::ToUINT

UINT CStr::ToUINT(LPCTSTR szSeparator = ",") const;

Description

Returns the value of the current CStr object as an unsigned integer or 0 if CStr doesn' t correspond to a number.

Compatibility

WIN16, WIN32

Group

_CSTR30_HASNUM

Example

```
int iValue = 125, iTest = 0;
CStr szMsg;
szMsg.From( (int)iValue );
iTest = (int)szMsg.ToUINT();
szMsg.From( (int)iTest );
szMsg.MB();
```

See also...

CStr::From
CStr::ToInt
CStr::ToLong
CStr::ToULong
CStr::ToFloat
CStr::ToDouble
CStr::ToTime
CStr::Round
CStr::IsDigit
CStr::IsInRange

CStr::ToLong

signed long CStr::ToLong(LPCTSTR szSeparator = ",") const;

Description
Returns the value of the current CStr object as a signed long or 0L if CStr doesn' t correspond to a number.

Compatibility
WIN16, WIN32

Group
_CSTR30_HASNUM

Example
```
long LValue = 125000L, LTest = 0;
CStr szMsg;
szMsg.From( (long)LValue );
LTest = szMsg.ToLong();
szMsg.From( (long)LTest );
szMsg.MB();
```

See also...
CStr::From
CStr::ToInt
CStr::ToUINT
CStr::ToULong
CStr::ToFloat
CStr::ToDouble
CStr::ToTime
CStr::Round
CStr::IsDigit
CStr::IsInRange

CStr::ToULong

unsigned long CStr::ToULong(LPCTSTR szSeparator = ",") const;

Description
Returns the value of the current CStr object as an unsigned long or 0L if CStr doesn' t correspond to a number.

Compatibility
WIN16, WIN32

Group
_CSTR30_HASNUM

Example
```
unsigned long LValue = 125000L, LTest = 0;
CStr szMsg;
szMsg.From( (unsigned long)LValue );
LTest = szMsg.ToULong();
szMsg.From( (unsigned long)LTest );
szMsg.MB();
```

See also...
CStr::From
CStr::ToInt
CStr::ToUINT
CStr::ToLong
CStr::ToFloat
CStr::ToDouble
CStr::ToTime
CStr::Round
CStr::IsDigit
CStr::IsInRange

CStr::ToFloat

float CStr::ToFloat(LPCTSTR szSeparator = ",", LPCTSTR szDecimalPoint = ".") const;

Description

Returns the value of the current CStr object as a float or 0 if CStr doesn' t correspond to a number.

Compatibility

WIN16, WIN32

Group

_CSTR30_HASNUM

Example

```
float FValue = 125.382, LTest = 0;
CStr szMsg;
szMsg.From( (double)FValue );
FTest = szMsg.ToFloat();
szMsg.From( (double)FTest );
szMsg.MB();
```

See also...

CStr::From
CStr::ToInt
CStr::ToUINT
CStr::ToLong
CStr::ToULong
CStr::ToDouble
CStr::ToTime
CStr::Round
CStr::IsDigit
CStr::IsInRange

CStr::ToDouble

double CStr::ToDouble(LPCTSTR szSeparator = ",", LPCTSTR szDecimalPoint = ".") const;

Description
Returns the value of the current CStr object as a double or 0 if CStr doesn' t correspond to a number.

Compatibility
WIN16, WIN32

Group
_CSTR30_HASNUM

Example
```
double DValue = 125.382, DTest = 0;
CStr szMsg;
szMsg.From( (double)DValue );
DTest = szMsg.ToDouble();
szMsg.From( (double)DTest );
szMsg.MB();
```

See also...
CStr::From
CStr::ToInt
CStr::ToUINT
CStr::ToLong
CStr::ToULong
CStr::ToFloat
CStr::ToTime
CStr::Round
CStr::IsDigit
CStr::IsInRange

CStr::ToTime

CTime CStr::ToTime(LPCTSTR szFilter = "D/M/Y") const;

Description

Returns the value of the current CStr object as a CTime object using <szFilter>. <szFilter> uses C++ runtime _strftime format specification but **'m' must be replaced with 'M'**. Also, you can use:

- ü D For the day (01-31),
- ü M For the month (01-12),
- ü Y For the year with century (1900-1999),
- ü y For the year without any century (00-99)
- ü H For the Hour (00-23),
- ü m For the minute (00-59),
- ü S For the second (00-59).

The other strftime specifiers won' t be used. Value must be filled with 0 ('01' instead of '1').

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASMOREBASIC

Example

```
CTime TheTime;
CStr szTest("12/01/1994");
TheTime = szTest.ToTime();
```

See also...

CStr::From
CStr::FromCurrentTime
CStr::FromFileTime

CStr::Delete

CStr CStr::Delete(unsigned PFirst = 0, unsigned PLength = 0);

Description

Deletes a piece of a string starting with <PFirst> and with the <PLength> length. <PFirst begin> is 0 indexed. CHANGES THE CURRENT CSTR OBJECT.

Compatibility

WIN16, WIN32, WINDLL

Group

<none>

Example

```
CStr szMsg("0123456789");
szMsg.Delete(2,3);
szMsg.MB();
```

See also...

CStr::Replace
CStr::Insert
CStr::Extract
CStr::Blank

CStr::Replace

CStr CStr::Replace(LPCTSTR szOld, LPCTSTR szNew = "");
CStr CStr::Replace(char szOld, char szNew);

Description

Replaces all substrings <szOld> of the current CStr object with <szNew>. If you use LPCTSTR, <szOld> and <szNew> can have different lengths, otherwise they must correspond to a character. CHANGES THE CURRENT CSTR OBJECT.

Compatibility

WIN16, WIN32, WINDLL

Group

<none>

Example

```
CStr szMsg("ABABAB");
szMsg.Replace( "AB", "C");
szMsg.MB();
```

See also...

CStr::Count
CStr::Delete
CStr::Insert
CStr::Extract
CStr::operator()

CStr::Insert

CStr CStr::Insert(LPCTSTR szSub, int Pos = 0);
CStr CStr::Insert(char szSub, int Pos = 0);

Description

Inserts the string <szSub> or the char <szSub> in the cuurent CStr object at position <Pos> (0 indexed).
CHANGES THE CURRENT CSTR OBJECT.

Compatibility

WIN16, WIN32, WINDLL

Group

<none>

Example

```
CStr szMsg("ABABAB");
szMsg.Insert( "00", 3);
szMsg.MB();
```

See also...

CStr::Delete
CStr::Replace
CStr::Extract
CStr::operator()

CStr::Extract

CStr CStr::Extract(int Pos = 1, LPCTSTR szDelimit = "\t");
CStr CStr::Extract(int Pos, char szDelimit);

Description

Extracts an item of the current CStr object using the <szDelimit> string as an item separator. <szDelimit> must be 1 character length if you use the LPCTSTR version. <Pos> is the item number you wish to extract (1 indexed). CHANGES THE CURRENT CSTR OBJECT.

Compatibility

WIN16, WIN32, WINDLL

Group

<none>

Example

```
CStr szMsg("ITEM1\tITEM2");
if ( szMsg.Count('\t') )
  {
  szMsg.Extract(2);
  szMsg.MB();
  }
```

See also...

CStr::Count
CStr::Delete
CStr::Replace
CStr::Insert
CStr::operator()

CStr::operator()

char& CStr::operator()(int Index);

Description
You can use this operator like you do with the CString SetAt function. Of course, the character specified with <index> (0 indexed) must be available (ie. in the string)!CHANGES THE CURRENT CSTR OBJECT.

Compatibility
WIN16, WIN32, WINDLL

Group
<none>

Example
```
CStr szMsg("0123456789");
szMsg(0) = 'A';
szMsg.MB();
```

See also...
CStr::Replace
CStr::Insert
CStr::Extract
CStr::FirstUpper

CStr::GetBuffer

virtual LPTSTR CStr::GetBuffer(int size = 0);

Description
Same as the CString function, but, if you don' t specify a <size>, GetBuffer will use the current object size or 255 if this one is 0. CHANGES THE CURRENT CSTR OBJECT.

Compatibility
WIN16, WIN32, WINDLL

Group
<none>

Example
```
CStr szMsg;
lstrcpy( szMsg.GetBuffer(), "TEST" );
szMsg.ReleaseBuffer();
szMsg.MB();
```

See also...
CStr::operator()
CStr::sprintf

CStr::Round

CStr CStr::Round(LPCTSTR szDecimalPoint = "."); [3.0]

Description

Rounds the current CStr object (which indicates a number) to the next integer. CHANGES THE CURRENT CSTR OBJECT.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASNUM

Example

```
CStr szMsg;
szMsg.From((float)123.236);
szMsg.Round();
szMsg.MB();
```

See also...

CStr::From
CStr::ToInt
CStr::ToUINT
CStr::ToLong
CStr::ToULong
CStr::ToFloat
CStr::ToDouble
CStr::IsDigit
CStr::IsInRange
CStr::NumFormat
AfxRound
AfxRoundToInt

CStr::FirstUpper

CStr CStr::FirstUpper(void);

Description

Put the first character into uppercase. CHANGES THE CURRENT CSTR OBJECT.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASMOREBASIC

Example

```
CStr szMsg("abcde AAAA");
szMsg.FirstUpper();
szMsg.MB();
```

See also...

CStr::operator()
CStr::MakeCharUpper
CStr::IsCharUpper

CStr::AddLF

CStr CStr::AddLF(void);

Description

Adds a Carriage return and a line feed to the current CStr object. CHANGES THE CURRENT CSTR OBJECT.

Compatibility

WIN16, WIN32, WINDLL

Group

<none>

Example

```
CStr szMsg("abcde AAAA");
szMsg.AddLF();
szMsg.ToFile("C:\\TRACE.LOG");
```

CStr::Blank

CStr CStr::Blank(void);

Description
Fills the current string with blanks characters (spaces) using the original length of the string (so that the string length stay the same). CHANGES THE CURRENT CSTR OBJECT.

Compatibility
```
WIN16, WIN32, WINDLL
```

Group
```
_CSTR30_HASMOREBASIC
```

Example
```
CStr szMsg("abcde AAAA");
szMsg.Blank();
```

See also...
CStr::operator()
CStr::AddLF
CStr::Replicate

CStr::Replicate

CStr CStr::Replicate(char cPattern = ' ', int iLen = 1);

Description

Creates a string with the character <cPattern> and the length <iLen> (but it' s not a constructor!). The string is filled with <cPattern>. CHANGES THE CURRENT CSTR OBJECT.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASMOREBASIC

Example

```
CStr szMsg;
szMsg.Replicate('.',25);
szMsg.MB();
```

See also...

CStr::Insert
CStr::operator()
CStr::Blank

CStr::CheckSize

CStr CStr::CheckSize(int iLength = 255, BOOL bAddPoints = TRUE);

Description

Checks the size of the current CStr object, and, if the length of the string is higher than <iLength>, this function will truncate the string and add '...' if <bAddPoints> is set to TRUE. This function is quite useful to use before you put values in Static controls.CHANGES THE CURRENT CSTR OBJECT.

Compatibility

WIN16, WIN32, WINDLL

Group

<none>

Example

```
Cstr szMsg;
szMsg.FromIni("General","FirstName","C:\\MYAPP\\MYAPP.INI");
szMsg.CheckSize( 25 );
szMsg.ToCtl( this, IDC_STATICFNAME );
```

See also...

CStr::Delete
CStr::Extract
CStr::FirstUpper
CStr::CheckDirSize
CStr::FromCtl
CStr::ToCtl

CStr::CheckDirSize

CStr CStr::CheckDirSize(int iLength = 60, BOOL bShift = TRUE);

Description

Checks the size of the current CStr object, and, if the length of the string is higher than <iLength>, this function will truncate the string and will try to keep the drive and the last dir or file name. So, in case some dirs are too long to be conserved, they will be replaced by '...'. If <bShift> is TRUE, the string will be converted to uppercase. This function is quite useful to use before you put values in Static controls. CHANGES THE CURRENT CSTR OBJECT.

Compatibility

WIN16, WIN32, WINDLL

Group

<none>

Example

```
Cstr szMsg;
szMsg.FromIni("General","AppPath","C:\\MYAPP\\MYAPP.INI");
szMsg.CheckDirSize( 80 );
szMsg.ToCtl( this, IDC_APPPATH );
```

See also...

CStr::Delete
CStr::Extract
CStr::FirstUpper
CStr::CheckSize
CStr::FromCtl
CStr::ToCtl

CStr::IsDigit

BOOL CStr::IsDigit(LPCTSTR szSeparator = ",", LPCTSTR szDecimalPoint = ".") const; [3.0]

Description

Returns TRUE is the current CStr object corresponds to a number, ie. all characters are numbers or '-', a space or <szSeparator> or <szDecimalPoint>.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASNUM

Example

```
Cstr szMsg("123,024.23");
if ( szMsg.IsDigit() )
  szMsg.MB();
```

See also...

CStr::From
CStr::ToInt
CStr::ToUINT
CStr::ToLong
CStr::ToULong
CStr::ToFloat
CStr::ToDouble
CStr::Round
CStr::IsInRange
CStr::FromCtl

CStr::IsInRange

BOOL CStr::IsInRange(double dMini = 0, double dMaxi = 100) const;

Description

Returns TRUE if the current number specified by the current CStr object is higher or equal to <dMini> and lower or equal to <dMaxi>.In many cases you must use IsDigit before this function.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASNUM

Example

```
Cstr szMsg("99.23");
if ( szMsg.IsInrange(50,100) )
  szMsg.MB();
```

See also...

CStr::From
CStr::ToInt
CStr::ToUINT
CStr::ToLong
CStr::ToULong
CStr::ToFloat
CStr::ToDouble
CStr::Round
CStr::IsDigit
CStr::CheckSize
CStr::FromCtl

CStr::FromIni

CStr CStr::FromIni(LPCTSTR szSection, LPCTSTR szKey, LPCTSTR szFile = NULL, int iSize = 512);

Description

Reads a value in the <szFile> INI File, or if this one is set to NULL, in WIN.INI. Also looks at the <szSection> section ("[...]"), at the <szKey> Key ("XXX=") and reads <iSize> characters. If this key is not defined, the current CStr will contain "". See GetProfileString documentation for more informations. CHANGES THE CURRENT CSTR OBJECT.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASINI

Example

```
CStr szIniValue;
szIniValue.FromIni("MyApp","UserName");
szIniValue.MB();
```

See also...

CStr::From
CStr::ToIni
CStr::FromFile

CStr::ToIni

void CStr::ToIni(LPCTSTR szSection, LPCTSTR szKey, LPCTSTR szFile = NULL) const;

Description

Write the current CStr object into an INI file. This INI file is <szFile>, or, if this value is NULL, the WIN.INI file will be used. Also this function will use the <szSection> section and the <szKey> key. See WriteProfileString documentation for more informations.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASINI

Example

```
CStr szIniValue("MyName");
szIniValue.ToIni("MyApp","UserName");
```

See also...

CStr::FromIni
CStr::ToFile

CStr::FromFile

CStr CStr::FromFile(LPCTSTR szFile, BOOL bKeepString = FALSE, BOOL bAddPoints = TRUE);

Description

Reads the file <szFile> and puts its content into the current CStr object. If <bKeepString> is set to TRUE, the file content will be appended to the current CStr. This function can read up to 32700 characters and if there are more than 32700 characters in the string or file, if <bAddPoints> is TRUE, the function will add "..." to the string. CHANGES THE CURRENT CSTR OBJECT.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASFILE

Example

```
CStr szFileList;
szFileList.FromFile("C:\\MYAPP\\MYLIST.DAT");
szFileList.MB();
```

See also...

CStr::From
CStr::FromFileTime
CStr::AddLF
CStr::FromIni
CStr::ToFile
CTim::ToFile
CTim::FromFile
AfxSetFileTime
AfxTouchFile

CStr::ToFile

BOOL CStr::ToFile(LPCTSTR szFile, BOOL bAppend = TRUE) const;

Description

Writes the current CStr object into the file <szFile>. If <bAppend> is TRUE, the string will be appended to the file, otherwise, the file will be deleted. Returns TRUE if successful, FALSE if not.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASFILE

Example

```
CStr szFileList;
szFileList = "Line1";
szFileList.AddLF();
szFileList += "Line2";
szFileList.AddLF();
szFileList.ToFile("C:\\MYAPP\\MYLIST.DAT", FALSE);
```

See also...

CStr::AddLF
CStr::ToIni
CStr::FromFile
CTim::ToFile
CTim::FromFile
AfxSetFileTime
AfxTouchFile

CStr::FromCtl

CStr CStr::FromCtl(CWnd* pWnd, int iCtl);

Description

Reads the text value, or the current selected value, of control <iCtl> of the window (or dialog box) <pWnd>. In many cases, <pWnd> is the local pointer 'this'. This function doesn' t support multiple selection listboxes.
CHANGES THE CURRENT CSTR OBJECT.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASWINCTL

Example

```
CStr szMsg;
szMsg.FromCtl( this, IDC_VALUE1 );
szMsg.MB();
```

See also...

CStr::From
CStr::FirstUpper
CStr::ToCtl
CStr::GetDlgItemText

CStr::ToCtl

void CStr::ToCtl(CWnd* pWnd, int iCtl, BOOL bAltMode = FALSE) const;

Description

Puts the current CStr string into the Windows control <iCtl> of the Window (or Dialog box) <pWnd>.
In many cases, <pWnd> is the local pointer 'this'. This function supports all Windows controls. In case of a ComboBox or a ListBox, the string will be appended to the end of the list without any sorting or, if you set <bAltMode> to TRUE, this value will be appended with a sorting process if this one is available.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASWINCTL

Example

```
CStr szMsg("Value...");
szMsg.ToCtl( this, IDC_VALUE1 );
```

See also...

CStr::FirstUpper
CStr::CheckDirSize
CStr::CheckSize
CStr::FromCtl
CStr::GetDlgItemText

CStr::MB

int CStr::MB(CWnd* pWnd = NULL, LPCTSTR szTitle = NULL, UINT mbType = MB_OK | MB_ICONINFORMATION) const;

Description

Display a message box using the current CStr value. <pWnd> is the parent Window to use ("this") and it can be NULL. <szTitle> is the title of the message box and <mbType> the type of this message box (see "MessageBox" SDK documentation for possible values). This function is useful because you can call it without any parameter.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASWINCTL

Example

```
CStr szMsg("Test");
szMsg.MB();
```

See also...

CStr::FirstUpper

Order form

To register you can print this help page and manually fill it, or use ORDER.TXT. Unreadable orders will be ignored. This order form applies to check or Credit Card orders you send to us. If you prefer to register thanks to the PsL registration service (in the USA), see Registration.

Ä If you are an European Community Company outside of France, don' t forget to give your EC VAT number (if you are not in this case, there is no additional fees).

ü You are:

```
Company name.........: _____________________________________________
Last name............: _____________________________________________
First name...........: _____________________________________________
Delivery address.....: _____________________________________________
                       _____________________________________________
                       _____________________________________________
City.................: _____________________________________________
Zip code/State.......: _____________________________________________
Country..............: _____________________________________________
Phone................: _____________________________________________
Fax..................: _____________________________________________
EMail................: _____________________________________________
CIS Mail.............: _____________________________________________
EC VAT Number........: _____________________________________________

Where did you find our softwares?
_____________________________________________________________________

Which development system are you using?
_____________________________________________________________________

Any Comments/Questions?
_____________________________________________________________________
_____________________________________________________________________
_____________________________________________________________________
_____________________________________________________________________
_____________________________________________________________________
```

ü You wish to register with your Credit Card:

Ä We accept VISA and EuroCard/MasterCard orders. You can send the form at our CIS or Internet address.In this case you will receive the passwords, in most cases, the same day. Your credit card will be billed in French Francs (1 FF = 0.19 US$ = 0.28 DM), while your bank account will display the corresponding amount in your local currency.

	QUANTITY	PRODUCTS	U. PRICE	TOTAL
q	_____	Zip Studio Shell.....	149 FF	_____
q	_____	Full programmer's pack..... (Zip + Zip Shell Src. + VBX + String + Setup Studio)	729 FF	_____
q	_____	Zip Studio Shell X 10.....	449 FF	_____
q	_____	Zip Studio Shell UNLIMITED.....	2099 FF	_____

q I already have a license for a studio
toolkit (my user code is: _____)
There is a 300 FF reduction for the
full programmer' s pack..... -300 FF _____

ORDER AMOUNT..... _____

Credit Card Type.....: q VISA
q EuroCard/MasterCard
Credit Card name.....: q Company name
q My name
Credit Card Number...: _ _ _ _ / _ _ _ _ / _ _ _ _ / _ _ _ _
Expiration date.....: _ _ / _ _

Stamp/Signature.....:
(Printed orders only) _____

ü You prefer a check payment:

Choose one currency..: q FRF
q US$
q DM
q GBP

Quantity	Product	FRF price	US$ price(*)	DM price	GBP price	Total
_____	Full pack	729-	135-	215-	95-	_____
_____	Zip Shell	149-	25-	44-	18-	_____
_____	Zip Shell (x10)	449-	90-	135-	59-	_____
_____	Zip Shell (unlimited)	2099-	420-	599-	269-	_____

q I already have a license for a studio toolkit (my user code is: _____) There is a 300 FF reduction for the full programmer' s pack.(FRF orders only)

Order amount.....**_____**

Ä (*) If you pay in US$, please make out your check to 'Denis CHEVRON' instead of 'HEXANET'

ü Send this form with your payment to:

HEXANET

HEXANET SW
BP 385.16
75768 PARIS CEDEX 16
FRANCE

EMail: hnet@dialup.francenet.fr
CIS: 100333,27
Fax(orders): (33-1) 40.72.80.77

CTim functions

How to use the CTim?

Before you can use a CTim function, you must create a CTim object (thanks to a classic CTime constructor), then you apply the function to this object. CTim functions are CTim members!

Example

```
CTim MyTime;
myTime.SetDay( 10 );
```

ü CTim()	CTim new constructor.	ü SetDay	Change the CTim day value.
ü Build	Fill the CTim fields.	ü SetHour	Change the CTim hour value.
ü FromFile	Retrieve a file last modification date.	ü SetMonth	Change the CTim month value.
ü IncDay	Increment the current date with a number of days.	ü SetYear	Change the CTim minute value.
ü IncSec	Increment the current date with a number of seconds.	ü SetSec	Change the CTim second value.
ü IncMin	Increment the current date with a number of minutes.	ü SetMin	Change the CTim year value.
ü IncHour	Increment the current date with a number of hours.	ü ToFile	Change the file last modification date. [3.0]
ü IsLeapYear	Returns TRUE if the current date concerns a Leap Year.		
ü IsLastDay	Returns TRUE if the current date is the last day of the month.		
ü IsValid	Returns TRUE if the current date is valid. [3.0]		

See also...

AfxIsDateValid

Afx global routines

ü Miscellaneous...

AfxIsDateValid	Returns TRUE if sepecified date is correct.
AfxFindWordInFile	Find a word in a text file.
AfxGiveTheHand	Let the user do something... [3.0]
AfxRound	Round a number with specified decimals.
AfxRoundToInt	Convert a number to an integer.
AfxSleep	Wait for n seconds [3.0]

ü Files...

AfxCheckFileName	Check (and change) a DOS filename.
AfxSetFileTime	Change the file last modification date. [3.0]
AfxSetFileAttributes	Change the attributes of a file. [3.0]
AfxTouchFile	Change the file last modification date. [3.0]

ü Configuration

AfxGetDecimalPoint	Returns the decimal character for this PC.
AfxGetFreeResRate	Return some free resources rates.
AfxGetFreeResValue	Return some free resources values.
AfxGetPrecision	Returns the number of decimals for this PC.
AfxGetLongDateFormat	Returns the full date format for this PC.
AfxGetShortDateFormat	Returns the short date format for this PC.
AfxGetThousandsSeparator	Returns the thousands separator character for this PC.
AfxHasWin4Look	Returns TRUE if the Win95 interface is here.

ü CTL3D...

AfxCtl3dLoad	Load CTL3DV2.DLL or CTL3D32.DLL. [3.0]
AfxCtl3dNewColors	Change CTL3D colors.
AfxCtl3dUnload	Unload CTL3DV2.DLL or CTL3D32.DLL.
AfxCtl3dSubclassCtl	Add 3D style to a control. [3.0]

ü Dialog boxes...

AfxCenterWindow	Center a Window (dialog box).
AfxGetWindowPos	Recover a Window position.
AfxWriteWindowPos	Keep a Window position.

ü Controls...

AfxActivateEdit	Activate an edit control text.
AfxEnableCANCEL	Enable/disable the dialog box CANCEL button.
AfxEnableDlgItem	Enable/disable a dialog box control.
AfxEnableOK	Enable/disable the dialog box OK button.
AfxSetDefButton	Change the dialog box default button.
AfxSetFocus	Set the focus to a dialog control.

CTim::CTim constructors

CTim::CTim(void)
CTim::CTim(const CTime& timeSrc)
CTim::CTim(time_t time)
CTim::CTim(int nYear, int nMonth, int nDay, int nHour = 0, int nMin = 0, int nSec = 0)
CTim::CTim(WORD wDosDate, WORD wDosTime)
CTim::CTim(const FILETIME& ft) [3.0] (Win32 only)
CTim::CTim(const SYSTEMTIME& ft) [3.0] (Win32 only)

Description

CTim constructors are the same than MFC 2.x/3.x CTime contructors. CTim add the CTim::CTim() constructor. This constructor doesn' t require any parameter and it will initialize the CTim object with the date 1/1/1980. This constructor is helpful if you plan to use AfxIsDateValid and/or Build.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASCTIM

Example

```
CTim myTime();
if ( AfxIsDateValid(iYear, iMonth, iDay ))
 myTime.Build( iYear, iMonth, iDay );
```

See also...

CTim::Build
AfxIsDateValid

CTim::Build

int CTim::Build(int iYear, int iMonth, int iDay, int iHour = 0, int iMin = 0, int iSec = 0);

Description

This method fills the CTim fields. Parameters must be the same as CTime constructor parameters (i.e. <iYear> must be in the 1970-2038 range and <iMonth> and <iDay> must be higher than 0). The function returns:

- CTIME_OK 0
- CTIME_OVERFLOW 1
- CTIME_NOTVALID 2

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASCTIM

Example

```
CTim myTime();
if ( AfxIsDateValid(iYear, iMonth, iDay ))
 myTime.Build( iYear, iMonth, iDay );
```

See also...

CTim()
AfxIsDateValid

CTim::FromFile

BOOL CTim::FromFile(LPCTSTR szFile);

Description

This method fills the CTim fields using the last modification date of the <szFile> file. The function returns FALSE if this date is not valid.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASCTIM

Example

```
CTim myTime();
if ( !myTime.FromFile("C:\\WINDOWS\\WIN.INI") )
  MessageBox( "Can' t retrieve WIN.INI date!" );
```

See also...

CTim::ToFile
CTim::CTim()AfxIsDateValid
AfxSetFileTime
AfxTouchFile

CTim::ToFile

BOOL ToFile(LPCTSTR szFile) const; [3.0]

Description

Change the last modification date/time of <szFile> using the current CTim value. The file must exist. The function returns FALSE if this date isn' t set.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASCTIM

Example

```
CTim myTime();
if ( !myTime.ToFile("C:\\WINDOWS\\WIN.INI") )
  MessageBox( "Can' t change WIN.INI date!" );
```

See also...

CTim::FromFile
CTim::CTim()
AfxIsDateValid
AfxSetFileTime
AfxTouchFile

CTim::SetMonth

BOOL CTim::SetMonth(int iMonth = 1);
BOOL CTim::SetMonth(LPCTSTR szMonth);

Description

Change the current CTim MONTH value. Valid range is 1-12. Returns TRUE if the new MONTH value is set.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASCTIM

Example

```
CTim myTime();
myTime.SetMonth(1);
myTime.SetDay(10);
myTime.SetYear(1995);
myTime.SetHour(12);
myTime.SetMin(30);
```

See also...

CTim::SetSec
CTim::SetMin
CTim::SetHour
CTim::SetDay
CTim::SetYear
AfxIsDateValid

CTim::SetDay

BOOL CTim::SetDay(int iDay = 1);
BOOL CTim::SetDay(LPCTSTR szDay);

Description
Change the current CTim DAY value. Valid range is 1-31. Returns TRUE if the new DAY value is set.

Compatibility
WIN16, WIN32, WINDLL

Group
_CSTR30_HASCTIM

Example
```
CTim myTime();
myTime.SetMonth(1);
myTime.SetDay(10);
myTime.SetYear(1995);
myTime.SetHour(12);
myTime.SetMin(30);
```

See also...
CTim::SetSec
CTim::SetMin
CTim::SetHour
CTim::SetMonth
CTim::SetYear
AfxIsDateValid

CTim::SetYear

BOOL CTim::SetYear(int iYear);
BOOL CTim::SetYear(LPCTSTR szYear);

Description
Change the current CTim YEAR value. Valid range is 1970-2038. Returns TRUE if the new YEAR value is set.

Compatibility
WIN16, WIN32, WINDLL

Group
_CSTR30_HASCTIM

Example
```
CTim myTime();
myTime.SetMonth(1);
myTime.SetDay(10);
myTime.SetYear(1995);
myTime.SetHour(12);
myTime.SetMin(30);
```

See also...
CTim::SetSec
CTim::SetMin
CTim::SetHour
CTim::SetDay
CTim::SetMonth
AfxIsDateValid

CTim::SetHour

BOOL CTim::SetHour(int iHour = 0);
BOOL CTim::SetHour(LPCTSTR szHour);

Description

Change the current CTim HOUR value. Valid range is 0-23. Returns TRUE if the new HOUR value is set.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASCTIM

Example

```
CTim myTime();
myTime.SetMonth(1);
myTime.SetDay(10);
myTime.SetYear(1995);
myTime.SetHour(12);
myTime.SetMin(30);
```

See also...

CTim::SetSec
CTim::SetMin
CTim::SetDay
CTim::SetMonth
CTim::SetYear
AfxIsDateValid

CTim::SetMin

BOOL CTim::SetMin(int iMin = 0);
BOOL CTim::SetMin(LPCTSTR szMin);

Description
Change the current CTim MINUTE value. Valid range is 0-59. Returns TRUE if the new MINUTE value is set.

Compatibility
WIN16, WIN32, WINDLL

Group
_CSTR30_HASCTIM

Example
```
CTim myTime();
myTime.SetMonth(1);
myTime.SetDay(10);
myTime.SetYear(1995);
myTime.SetHour(12);
myTime.SetMin(30);
```

See also...
CTim::SetSec
CTim::SetHour
CTim::SetDay
CTim::SetMonth
CTim::SetYear
AfxIsDateValid

CTim::SetSec

BOOL CTim::SetSec(int iSec = 0);
BOOL CTim::SetSec(LPCTSTR szSec);

Description

Change the current CTim SECOND value. Valid range is 0-59. Returns TRUE if the new SECOND value is set.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASCTIM

Example

```
CTim myTime();
myTime.SetMonth(1);
myTime.SetDay(10);
myTime.SetYear(1995);
myTime.SetHour(12);
myTime.SetMin(30);
```

See also...

CTim::SetMin
CTim::SetHour
CTim::SetDay
CTim::SetMonth
CTim::SetYear
AfxIsDateValid

CTim::IncDay

BOOL CTim::IncDay(int iDay = 1);

Description

Increment/Decrement the current CTim object with <iDay> days. You can add (or substract) up to 25000 days...The function returns TRUE if <iDay> is in this range.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASCTIM

Example

```
CTim myTime();
myTime.Build( 1995, 1, 1, 23, 0);
myTime.IncDay(30);
myTime.IncSec(-30);
myTime.IncMin(-1);
myTime.IncHour();
```

See also...

CTim::IncMin
CTim::IncSec
CTim::IncHour
AfxIsDateValid

CTim::IncSec

BOOL CTim::IncSec(int iSec = 1);

Description

Increment/Decrement the current CTim object with <iSec> seconds. <iSec> valid range is -59/+59. The function returns TRUE if <iSec> is in this range.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASCTIM

Example

```
CTim myTime();
myTime.Build( 1995, 1, 1, 23, 0);
myTime.IncDay(30);
myTime.IncSec(-30);
myTime.IncMin(-1);
myTime.IncHour();
```

See also...

CTim::IncMin
CTim::IncHour
CTim::IncDay
AfxIsDateValid

CTim::IncMin

BOOL CTim::IncMin(int iMin = 1);

Description

Increment/Decrement the current CTim object with <iMin> minutes. <iMin> valid range is -59/+59. The function returns TRUE if <iMin> is in this range.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASCTIM

Example

```
CTim myTime();
myTime.Build( 1995, 1, 1, 23, 0);
myTime.IncDay(30);
myTime.IncSec(-30);
myTime.IncMin(-1);
myTime.IncHour();
```

See also...

CTim::IncDay
CTim::IncSec
CTim::IncHour
AfxIsDateValid

CTim::IncHour

BOOL CTim::IncHour(int iHour = 1);

Description

Increment/Decrement the current CTim object with <iHour> hours. <iHour> valid range is -23/+23. The function returns TRUE if <iHour> is in this range.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASCTIM

Example

```
CTim myTime();
myTime.Build( 1995, 1, 1, 23, 0);
myTime.IncDay(30);
myTime.IncSec(-30);
myTime.IncMin(-1);
myTime.IncHour();
```

See also...

CTim::IncMin
CTim::IncDay
CTim::IncSec
AfxIsDateValid

CTim::IsLeapYear

BOOL CTim::IsLeapYear( ) const;

Description

As you think this function returns TRUE if the current CTim value concerns a Leap year.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASCTIM

Example

```
CTim myTime();
myTime.Build( 1995, 1, 1, 23, 0);
if ( myTime.IsLeapYear() )
MessageBox( "It is a Leap year...");
```

See also...

CTim::IsValid
CTim::IsLastDay
AfxIsDateValid

CTim::IsLastDay

BOOL CTim::IsLastDay() const;

Description
This function returns TRUE if the current CTim value correspond to the last day of the month.

Compatibility
WIN16, WIN32, WINDLL

Group
_CSTR30_HASCTIM

Example
```
CTim myTime();
myTime.Build( 1994, 12, 31, 23, 0);
if ( myTime.IsLastDay() )
  MessageBox( "12/31 is really the last day of december...");
```

See also...
CTim::IsLeapYear
CTim::IsValid
AfxIsDateValid

CTim::IsValid

BOOL CTim::IsValid() const;

Description

This function returns TRUE if the current CTim value corresponds to a valid date/time.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASCTIM

Example

```
CTim myTime();
myTime.Build( 1994, 12, 31, 23, 0);
if ( !myTime.IsValid() )
  MessageBox( "31 december 1994 should be valid...");
```

See also...

CTim::ToFile
CTim::FromFile
AfxIsDateValid
AfxSetFileTime
AfxTouchFile

AfxIsDateValid

BOOL AfxIsDateValid(int iYear, int iMonth, int iDay);

Description

Returns TRUE if the specified components correspond to a correct date. <iYear> is the year value (with its century), <iMonth> the month value and <iDay> the day value.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASCTIM

Example

```
CTim myTime();
if ( AfxIsDateValid(iYear, iMonth, iDay ))
 myTime.Build( iYear, iMonth, iDay );
```

See also...

CTim
CTim::IsLeapYear
CTm::IsLastDay
CTm::IsValid
CTim::ToFile
CTim::FromFile
AfxSetFileTime
AfxTouchFile

AfxGetFreeResRate

UINT AfxGetFreeResRate(int iResCode);

Description

Return a free resources rate for the GLOBAL, USER, GDI or DISK parameter. <iResCode> must be one of the following constants:

- FREERES_GLOBAL 0
- FREERES_USER 1
- FREERES_GDI 2
- FREERES_DISK 4

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASMOREBASIC

Example

```
CStr sMsg1, sMsg2;
sMsg1.From( AfxGetFreeResRate( FREERES_GDI ));
sMsg2.From( AfxGetFreeResValue( FREERES_MEM ));
sMsg1.MB(NULL, "Free GDI rate...");
sMsg2.MB(NULL, "Free Memory...");
```

See also...

AfxGetFreeResValue
AfxHasWin4Look

AfxGetFreeResValue

unsigned long AfxGetFreeResValue(int iResCode);

Description

Return a free resources value for the current DISK or MEMORY parameter. <iResCode> must be one of the following constants and the returned value is in KBytes.

- ü FREERES_MEM 3
- ü FREERES_DISK 4

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASMOREBASIC

Example

```
CStr sMsg1, sMsg2;
sMsg1.From( AfxGetFreeResRate( FREERES_GDI ));
sMsg2.From( AfxGetFreeResValue( FREERES_MEM ));
sMsg1.MB(NULL, "Free GDI rate...");
sMsg2.MB(NULL, "Free Memory...");
```

See also...

AfxGetFreeResRate
AfxHasWin4Look

AfxGetPrecision

int AfxGetPrecision();

Description
This function returns the number of decimal digits the user specified thanks to the control panel.

Compatibility
WIN16, WIN32, WINDLL

Group
_CSTR30_HASINI

See also...
AfxGetDecimalPoint
AfxGetLongDateFormat
AfxGetShortDateFormat
AfxGetThousandsSeparator

AfxGetDecimalPoint

CStr AfxGetDecimalPoint();

Description
This function returns the character to use display as the decimal point the user specified thanks to the control panel.

Compatibility
WIN16, WIN32, WINDLL

Group
_CSTR30_HASINI

See also...
AfxGetPrecision
AfxGetLongDateFormat
AfxGetShortDateFormat
AfxGetThousandsSeparator

AfxGetThousandsSeparator

CStr AfxGetThousandsSeparator();

Description

This function returns the character to use display as the thousands separator the user specified thanks to the control panel.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASINI

See also...

AfxGetDecimalPoint
AfxGetPrecision
AfxGetLongDateFormat
AfxGetShortDateFormat

AfxGetShortDateFormat

CString AfxGetShortDateFormat();

Description

This function returns the short date format string the user specified thanks to the control panel. This string can be used with the CStr::From(CTime) function. If you use this string with the CStr::ToTime function, you must replace 'M' with 'm' and 'm' with 'M'.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASINI

Example

```
CStr szTestString;
szTestString.FromCurrentTime( AfxGetShortDateFormat() );
szTestString.MB();
```

See also...

AfxGetDecimalPoint
AfxGetPrecision
AfxGetLongDateFormat
AfxGetThousandsSeparator

AfxGetLongDateFormat

CString AfxGetLongDateFormat();

Description

This function returns the long (full) date format string the user specified thanks to the control panel. This string can be used with the CStr::From(CTime) function. If you use this string with the CStr::ToTime function, you must replace 'M' with 'm' and 'm' with 'M'.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASINI

Example

```
CStr szTestString;
szTestString.FromCurrentTime( AfxGetLongDateFormat() );
szTestString.MB();
```

See also...

AfxGetDecimalPoint
AfxGetPrecision
AfxGetShortDateFormat
AfxGetThousandsSeparator

AfxCtl3dLoad

extern "C" int AfxCtl3dLoad( ); [3.0]

Description

This function will load (and 'register') the CTL3DV2.DLL (for Win16) or CTLD3D32.DLL (for NT < 3.52). For details and terms regarding this DLL, have a look at this Microsoft product.

Ä If you choose to use CTL3D and String Studio, you must call this function in your CWinApp::InitInstance method. Also you can call AfxCtl3dNewColors on the next line to reset the used colors. As we use the automatic mode (CTL3D_ALL), the only thing you will have to do is to call the AfxCtl3dUnload in your CWinApp::ExitInstance function. The main interest of the String Studio CTL3D wrappers comes from the dynamic DLL loading: If the library don' t find the DLL, no warning will be displayed and the program will run. The new version only load CTL3D from WINDOWS/SYSTEM as required.

Ä Also, if you are using VC++ 1.5x but your (16bits) works may be used under Win95 you should do the following: First, use AfxHasWin4Look in you InitInstance method. If the function returns FALSE, call <SetDialogBkColor();>. Now use <AfxCtl3dLoad> in all cases.

The function returns one of the foilowing constants:

- ü AFXCTL3D_OK 0 // Ok to use CTL3D
- ü AFXCTL3D_NOTFOUND 1 // The user doesn' t get this DLL
- ü AFXCTL3D_BADDLL 2 // Error
- ü AFXCTL3D_ERROR 3 // Error

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASCTL3D

Example

```
CMyApp::InitInstance(...
{
...
if ( bUseCtl3d = AfxCtl3dLoad())
  AfxCtl3dNewColors();
...
}

CMyApp::ExitInstance(...
{
AfxCtl3dUnload();
...
}
```

See also...

AfxCtl3dUnload
AfxCtl3dNewColors
AfxCtl3dSubclassCtl
AfxHasWin4Look

AfxCtl3dUnload

extern "C" BOOL AfxCtl3dUnload( );

Description

Unregister and unload the CTL3DV2.DLL (for Win16) or CTL3D32.DLL (for NT < 3.52) you loaded with AfxCtl3dLoad .Returns FALSE if an error occured.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASCTL3D

Example

```
CMyApp::InitInstance(...
{
...
if ( bUseCtl3d = AfxCtl3dLoad())
  AfxCtl3dNewColors();
...
}

CMyApp::ExitInstance(...
{
AfxCtl3dUnload();
...
}
```

See also...

AfxCtl3dLoad
AfxCtl3dNewColors
AfxCtl3dSubclassCtl

AfxCtl3dNewColors

extern "C" BOOL AfxCtl3dNewColors( );

Description

Tell CTL3DV2.DLL to use new colors. You can use this function in your CWinApp::InitInstance and/or CMainFrm::OnSysColorChanged for instance. The function returns TRUE if the new colors are set.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASCTL3D

Example

```
CMyApp::InitInstance(...
{
...
if ( bUseCtl3d = AfxCtl3dLoad())
  AfxCtl3dNewColors();
...
}

CMyApp::ExitInstance(...
{
AfxCtl3dUnload();
...
}
```

See also...

AfxCtl3dLoad
AfxCtl3dUnload
AfxCtl3dSubclassCtl

AfxCtl3dSubclassCtl

extern "C" BOOL AfxCtl3dSubclassCtl(HWND hItem); [3.0]

Description

<AfxCtl3d...> functions use the automatic mode so that you don' t need to subclass each control of your dialog boxes. However, sometimes, when you dunamically create a control for instance, you need to perform a manual subclassing. In this case use this function and give the HWND of your control as <hItem>. The function returns TRUE if successful. See the CTL3D help file for details.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASCTL3D

See also...

AfxCtl3dLoad
AfxCtl3dUnload
AfxCtl3dNewColors

AfxWriteWindowPos

BOOL AfxWriteWindowPos(HWND hWnd , BOOL bUseParent = FALSE, LPCTSTR szWindowName = "Main", LPCTSTR szIni = NULL);

Description

This function write the <hWnd> coordinates to the <szIni> Inifile under the <szWindowName> key. If <szIni> is NULL, WIN.INI will be used. AfxWriteWindowPos will use the "WindowPos" section in the INI file. if <bUseParent> is set to TRUE the function will use the parent window to compute the <hWnd> window coordinates (it is specially usefull with MDI child windows). If the Window is iconized or maximized the function does not perform anything. The function returns TRUE if the coordinates were written.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASINI

Example

```
OnInitDialog()
{
...
AfxGetWindowPos( m_hWnd, "MyDialogBox", szIniFile );
...
}

OnDestroy()
{
AfxWriteWindowPos( m_hWnd, FALSE, "MyDialogBox", szIniFile );
CDialog::OnDestroy();
...
}
```

See also...

AfxGetWindowPos
AfxCenterWindow

AfxGetWindowPos

RECT AfxGetWindowPos(HWND hWnd = NULL, LPCTSTR szWindowName = "Main", LPCTSTR szIni = NULL, BOOL bMove = TRUE, BOOL bSize = FALSE);

Description

This function will retrieve a window coordinates from an INI file then it will change the specified window position if chosen. <hWnd> is the window to use to change the coordinates, <szWindowName> is the key label in <szIni> INI file under the "WindowPos" section. If <szIni> is NULL the function will use WIN.INI. <bMove> specify you wish to move the Window and set <bSize> if you wish to resize the Window. The function doesn' t move iconized or maximized windows. If you wish to perform a special process before moving the window, set <bSize> and <bMove> to FALSE, then use the returned RECT structure.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASINI

Example

```
OnInitDialog()
{
...
AfxGetWindowPos( m_hWnd, "MyDialogBox", szIniFile );
...
}

OnDestroy()
{
AfxWriteWindowPos( m_hWnd, FALSE, "MyDialogBox", szIniFile );
CDialog::OnDestroy();
...
}
```

See also...

AfxWriteWindowPos
AfxCenterWindow

AfxCenterWindow

BOOL AfxCenterWindow(CWnd* pWnd, BOOL bAbsolute = TRUE);

Description

This function center the <pWnd> window. If <bAbsolute> is FALSE the <pWnd> window will be centered on its parent window. The function returns TRUE in case of success.

Compatibility

WIN16, WIN32, WINDLL

Group

<none>

Example

```
OnInitDialog()
{
...
AfxCenterWindow( this );
...
}
```

See also...

AfxWriteWindowPos
AfxGetWindowPos

AfxEnableDlgItem

BOOL AfxEnableDlgItem(CWnd* pWnd, UINT id = 1, BOOL bEnabled = TRUE);

Description

This function enable or disable the control <id> in the dialog box <pWnd>. The control will be disabled if <bEnabled> is set to FALSE;

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASWINCTL

Example

```
AfxEnableDlgItem( this, IDOK );
```

See also...

AfxActivateEdit
AfxEnableCANCEL
AfxEnableOK
AfxSetDefButton
AfxSetFocus

AfxEnableOK

BOOL AfxEnableOK(CWnd* pWnd, BOOL bEnabled = TRUE);

Description

This function enable or disable the IDOK button in the dialog box <pWnd>. The control will be disabled if <bEnabled> is set to FALSE;

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASWINCTL

Example

```
AfxEnableOK( this, FALSE );
```

See also...

AfxActivateEdit
AfxEnableCANCEL
AfxEnableDlgItem
AfxSetDefButton
AfxSetFocus

AfxEnableCANCEL

BOOL AfxEnableCANCEL(CWnd* pWnd, BOOL bEnabled = TRUE);

Description

This function enable or disable the IDCANCEL button in the dialog box <pWnd>. The control will be disabled if <bEnabled> is set to FALSE;

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASWINCTL

Example

```
AfxEnableCANCEL( this, FALSE );
```

See also...

AfxActivateEdit
AfxEnableDlgItem
AfxEnableOK
AfxSetDefButton
AfxSetFocus

AfxSetFocus

void AfxSetFocus(CWnd* pWnd, UINT id = 1);

Description
This function set the focus to the control <id> in the dialog box <pWnd>.

Compatibility
```
WIN16, WIN32, WINDLL
```

Group
```
_CSTR30_HASWINCTL
```

Example
```
AfxSetFocus( this, IDC_LIST1 );
```

See also...
AfxActivateEdit
AfxEnableCANCEL
AfxEnableDlgItem
AfxEnableOK
AfxSetDefButton

AfxSetDefButton

void AfxSetDefButton(CWnd* pWnd, UINT id = 1);

Description

This function set the default button to the control <id> in the dialog box <pWnd>. This fonction don' t work with all the kinds of dialog controls.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASWINCTL

Example

```
AfxSetDefButton( this, IDEDIT );
```

See also...

AfxActivateEdit
AfxEnableCANCEL
AfxEnableDlgItem
AfxEnableOK
AfxSetFocus

AfxActivateEdit

void AfxActivateEdit(CWnd* pWnd, UINT id);

Description
This function set the focus to the edit control <id> in the dialog box <pWnd>, then it select all the control text.

Compatibility
WIN16, WIN32, WINDLL

Group
_CSTR30_HASWINCTL

Example
```
AfxActivateEdit( this, IDC_EDIT1 );
```

See also...
AfxEnableCANCEL
AfxEnableDlgItem
AfxEnableOK
AfxSetDefButton
AfxSetFocus

AfxRoundToInt

#define AfxRoundToInt(x)

Description
This macro rounds <x> to an integer.

Compatibility
`WIN16, WIN32, WINDLL`

Group
`<none>`

See also...
AfxRound
CStr::NumFormat

AfxRound

double AfxRound(double x, int precision = 2);

Description

This function rounds the double value <x> with <precision> number of decimals. So, the returned value is <x> with <precision> decimal digits.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASNUM

See also...

AfxRoundToInt
CStr::NumFormat

AfxFindWordInFile

BOOL AfxFindWordInFile(LPCTSTR szFile, LPCTSTR szWord, unsigned long iFromLine = 0L, int iFromCursor = 0, unsigned long* rLineValue = NULL, int* rCursorValue =NULL, BOOL bExact = TRUE, BOOL bMatchCase = FALSE);

Description

This function is the same as CStr::FindWordInMultilineBuffer but it uses a file instead of a string. <szFile> is a text file you will use as the data source. <szWord> is the word to look for. <iFromLine> is the index of the line from which you want to search <szWord>; <iFromCursor> is the position in the line from which you want to begin the search process; Set <bExact> to TRUE if you want to exclude the partial names and <bMatchCase> to TRUE if you want to check the case too. <rLineValue> and <rCursorValue> are filled on output and they specify the word position in the file. If they are set to NULL they won' t be used by the function. As usual all indexes are 0 based. The function returns TRUE if <szWord> exist, FALSE if not.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASFINDWORD

Example

```
CStr szMsg, szLine, szCursor;
unsigned long Y=0L;
int X=0;
if ( AfxFindWordInFile("C:\\TEST.TXT", "test", 0L, 0, &Y, &X ) )
 {
 szLine.From( Y );
 szCursor.From( X );
 szMsg = "test found at position ";
 szMsg += szLine;
 szMsg += ":";
 szMsg += szCursor;
 szMsg.MB();
 }
```

See also...

CStr::FindWord
CStr::FindWordInMultilineBuffer

AfxCheckFileName

BOOL AfxCheckFileName(CString& sValue, BOOL bNoWildCards = TRUE);

Description

This function checks the <sValue> DOS filename. <sValue> is a file name single component (without any drive and path specification) like 'MYDIR', 'MYFILE.TXT' or '??o.*'. If <sValue> is not a filename component it will be translated to an appropriate DOS file name. As <sValue> is passed by reference this value can be changed by the function. If <bNoWildCard> is set to TRUE, the function will remove * and ?. The function returns TRUE if <sValue> is a valid DOS filename.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASFILE

Example

```
CStr myFileName = "123+89,78*";
AfxCheckFileName( myFileName );
myFileName.MB();
```

See also...

CStr::CheckFullFileName
CStr::IsFileName
CStr::IsPathName
CStr::IsFullFileName
CStr::GetPathValue
CStr::GetFileNameValue
CStr::CheckFullFileName
CStr::CheckFileExtension

CStr::CStr

CStr::CStr(UINT iStringID, HINSTANCE hInstance = NULL):

Description

A new constructor for the CStr objects. It loads the string <iStringID> from the resources of <hInstance>. You must cast <iStringID> to an UINT and, if <hInstance> is NULL (default), the library will use the default HINSTANCE (thanks to AfxGetInstanceHandle()).

Compatibility

WIN16, WIN32, WINDLL

Group

<none>

Example

```
CStr myProgName((UINT)IDS_APPNAME );
myProgName.MB();
```

See also...

CStr::FromRes

CStr::IsFileName

BOOL CStr::IsFileName(BOOL bAllowWildCards = FALSE) const;

Description

The function checks the current CStr object and returns TRUE if it is a single component filename. This component can be a file name without the drive and the path specification (like 'MYFILE.TXT' or 'DIR1'). If <bAllowWildCard> is set to TRUE, the function will allow DOS pattern characters (* and ?). The function returns TRUE if the string is a file name, FALSE if not.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASFILE

Example

```
CStr szFile = "AUTOEXEC.BAT";
if ( szFile.IsFileName() )
 MessageBox( "AUTOEXEC.BAT is a valid file name" );
```

See also...

CStr::IsPathName
CStr::IsFullFileName
CStr::GetPathValue
CStr::GetFileNameValue
CStr::CheckFullFileName
CStr::CheckFileExtension
AfxCheckFileName

CStr::IsPathName

BOOL CStr::IsPathName( ) const;

Description

Returns TRUE if the current CStr object specifies a valid DOS directory name (with its full path). String Studio 2.0 will returns FALSE if the first right character is not '\\'. The function doesn' t test the directory exist. It returns TRUE if this is a valid DOS path name.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASFILE

Example

```
CStr szMyDir = "C:\\DOS\\DIR1\\";
if ( szMyDir.IsPathName() )
 MessageBox( "C:\\DOS\\DIR1\\ is a valid directory name" );
```

See also...

CStr::IsFileName
CStr::IsFullFileName
CStr::GetPathValue
CStr::GetFileNameValue
CStr::CheckFullFileName
CStr::CheckFileExtension
AfxCheckFileName

CStr::IsFullFileName

BOOL CStr::IsFullFileName(BOOL bNoWildCards = TRUE) const;

Description

Returns TRUE if the current CStr object specifies a valid DOS file name (with its full path). If <bNoWildCards> is set to FALSE, * and ? will be allowed. It returns TRUE if this is a valid DOS file name. As usual, this function doesn' t support UNC file names.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASFILE

Example

```
CStr szMyFile = "C:\\DOS\\MYFILE.TXT";
if ( szMyFile.IsFullFileName() )
 MessageBox( "C:\\DOS\\MYFILE.TXT is a valid file name" );
```

See also...

CStr::IsFileName
CStr::IsPathName
CStr::GetPathValue
CStr::GetFileNameValue
CStr::CheckFullFileName
CStr::CheckFileExtension
AfxCheckFileName

CStr::GetPathValue

CStr CStr::GetPathValue( );

Description

Returns the path value for a full file name (with the path). The string must include a file name (i.e. the first right character must not be '\\'). CHANGES THE CURRENT CSTR VALUE.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASFILE

Example

```
CStr szMyFile = "C:\\DOS\\COMMAND.COM";
szMyFile.GetPathValue();
szMyFile.MB();
```

See also...

CStr::IsFileName
CStr::IsPathName
CStr::IsFullFileName
CStr::GetFileNameValue
CStr::CheckFullFileName
CStr::CheckFileExtension
AfxCheckFileName

CStr::GetFileNameValue

CStr CStr::GetFileNameValue( );

Description

Returns the file name value for a full file name (with the path). The string must include a file name (i.e. the first right character must not be '\\'). CHANGES THE CURRENT CSTR VALUE.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASFILE

Example

```
CStr szMyFile = "C:\\DOS\\COMMAND.COM";
szMyFile.GetFileNameValue();
szMyFile.MB();
```

See also...

CStr::IsFileName
CStr::IsPathName
CStr::IsFullFileName
CStr::GetPathValue
CStr::GetFileNameValue
CStr::CheckFullFileName
CStr::CheckFileExtension
AfxCheckFileName

CStr::CheckFullFileName

CStr CStr::CheckFullFileName(BOOL bNoWildCards = TRUE, BOOL bAddSlash =FALSE);

Description

This function will check a full file name or directory name value. You can use relative directories ('//' or '..//' or './/'). <bNoWildCards> must be set to FALSE if you wish to use a DOS pattern and set <bAddSlash> to TRUE if you are using a directory with a final '\\'. The function returns the new full file name or a blank string. CHANGES THE CURRENT CSTR VALUE.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASFILE

Example

```
CStr szMyFile = "C:\\BAD++\\W.R.O.N.G";
szMyFile.CheckFullFileName();
szMyFile.MB();
```

See also...

CStr::IsFileName
CStr::IsPathName
CStr::IsFullFileName
CStr::GetPathValue
CStr::GetFileNameValue
CStr::CheckFileExtension
AfxCheckFileName

CStr::CheckFileExtension

CStr CStr::CheckFileExtension(LPCTSTR szExtension, BOOL bForceExtension = FALSE);

Description

The function will search for the <szExtension> in the current CStr object. If not found <szExtension> will be added. If <bForceExtension> is set to TRUE, the new <szExtension> extension will be used in all cases. You can use single component file names or full file names. The function returns the new string. CHANGES THE CURRENT CSTR VALUE.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASFILE

Example

```
CStr szMyFile = "TEST";
szMyFile.CheckFileExtension( "ZIP");
szMyFile.MB();
```

See also...

CStr::IsFileName
CStr::IsPathName
CStr::IsFullFileName
CStr::GetPathValue
CStr::GetFileNameValue
CStr::CheckFullFileName
AfxCheckFileName

CStr::ChDir

BOOL CStr::ChDir(BOOL bIncludesFName = FALSE, BOOL bCreateDir = FALSE) const;

Description

Go to the current CStr directory specification. If <bIncludeFName> is set to TRUE, the function will temporary remove the file name component before the action. If <bCreateDir> is set to TRUE, the function will try to create the new required directories. The function can use a different drive and a relative path. It will return TRUE if the new directory is reached. If <bIncFile> is set to TRUE you must not use a relative path specification.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASFILE

Example

```
szOpenFileDir = "C:\\TEST\\BACKUP\\BACK.TXT";
if ( szOpenFile.ChDir( TRUE, TRUE ) )
 MessageBox( "ChDir to C:\\TEST\\BACKUP successful!");
```

See also...

CStr::MkDir
CStr::GetPathValue
CStr::DoesDirExist
CStr::DoesFileExist
CStr::GetAppDir
CStr::GetWinDir
CStr::GetWinSysDir
CStr::GetThisDir
CStr::GetTempDir

CStr::MkDir

BOOL CStr::MkDir(BOOL bIncFile = FALSE) const;

Description

Build a new directory as specified in the current CStr object. If <bIncFile> is set to TRUE the function will temporary removes the file name before it make the new directories. It can build more than one directory at the same time, use a relative path or a different drive. If <bIncFile> is set to TRUE you must not use a relative path specification. The function returns FALSE if an error occured.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASFILE

Example

```
CStr szNewDir = "..\\BACKUP\\";
if ( szNewDir.MkDir() )
 MessageBox( "..\\BACKUP succesfully created!" );
```

See also...

CStr::ChDir
CStr::RemoveDir
CStr::CreateFile
CStr::DeleteFile
CStr::GetPathValue
CStr::DoesDirExist
CStr::DoesFileExist
CStr::GetAppDir
CStr::GetWinDir
CStr::GetWinSysDir
CStr::GetThisDir
CStr::GetTempDir

CStr::DoesFileExist

BOOL CStr::DoesFileExist( ) const;

Description
Returns TRUE if the current CStr file specification exists. The function doesn' t allow relative path.

Compatibility
WIN16, WIN32, WINDLL

Group
_CSTR30_HASFILE

Example
```
CStr szFile = "C:\\AUTOEXEC.BAT";
if (!szFile.DoesFileExist() )
 MessageBox( "AUTOEXEC.BAT missing...");
```

See also...
CStr::ChDir
CStr::MkDir
CStr::GetPathValue
CStr::DoesDirExist
CStr::GetAppDir
CStr::GetWinDir
CStr::GetWinSysDir
CStr::GetThisDir
CStr::GetTempDir
CStr::GetFileAttributes
CStr::SetFileAttributes
CStr::IsRDOnly
CStr::GetFileSize
CStr::CreateFile
CStr::DeleteFile
AfxSetFileAttributes

CStr::DoesDirExist

BOOL CStr::DoesDirExist(BOOL bIncFile = FALSE) const;

Description

Returns TRUE if the current CStr directory specification exists. The function doesn' t allow relative path. If <bIncFile> is set to TRUE the function will temporary remove the file name component to test the dir.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASFILE

Example

```
CStr szDir = "C:\\DOS";
if (!szDir.DoesDirExist() )
 MessageBox( "DOS directory missing...");
```

See also...

CStr::ChDir
CStr::MkDir
CStr::RemoveDir
CStr::GetPathValue
CStr::DoesFileExist
CStr::GetAppDir
CStr::GetWinDir
CStr::GetWinSysDir
CStr::GetThisDir
CStr::GetTempDir

CStr::GetAppDir

CStr CStr::GetAppDir(LPCTSTR szAppName = NULL);

Description

Returns the application directory. If <szAppName> is set to NULL, this function will use the default application name thanks to a <AfxGetAppName()> call. Otherwise, <szAppName> must be a 'module' name like 'MYAPP.EXE' or 'MYMODULE.DLL'. CHANGES THE CURRENT CSTR VALUE.

Under Win16, this function uses TOOLHELP.DLL (but you don' t have to link with TOOLHELP.LIB) and it can return a bad value in case you use more than one module with the same base name (MYPROG.DLL and MYPROG.EXE for instance).

Ä Since TOOLHELP.DLL isn' t available under Win32, the function may fail or returns if <szAppName> isn' t NULL.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASFILE

Example

```
CStr szAppPath;
szAppPath.GetAppDir();
szAppPath.MB( this, "Application path is:" );
```

See also...

CStr::ChDir
CStr::MkDir
CStr::GetPathValue
CStr::DoesDirExist
CStr::DoesFileExist
CStr::GetWinDir
CStr::GetWinSysDir
CStr::GetThisDir
CStr::GetTempDir

CStr::GetWinDir

CStr CStr::GetWinDir( );

Description
Returns the WINDOWS directory (with the final '\\' character). CHANGES THE CURRENT CSTR VALUE.

Compatibility
`WIN16, WIN32, WINDLL`

Group
`_CSTR30_HASFILE`

See also...
- CStr::ChDir
- CStr::MkDir
- CStr::GetPathValue
- CStr::DoesDirExist
- CStr::DoesFileExist
- CStr::GetAppDir
- CStr::GetWinSysDir
- CStr::GetThisDir
- CStr::GetTempDir

 CStr::GetWinSysDir

CStr CStr::GetWinSysDir( );

Description
Returns the WINDOWS\SYSTEM directory (with the final '\\' character). CHANGES THE CURRENT CSTR VALUE.

Compatibility
`WIN16, WIN32, WINDLL`

Group
`_CSTR30_HASFILE`

See also...
- CStr::ChDir
- CStr::MkDir
- CStr::GetPathValue
- CStr::DoesDirExist
- CStr::DoesFileExist
- CStr::GetAppDir
- CStr::GetWinDir
- CStr::GetThisDir
- CStr::GetTempDir

CStr::GetThisDir

CStr CStr::GetThisDir( );

Description
Returns the current directory (with the final '\\' character). CHANGES THE CURRENT CSTR VALUE.

Compatibility
WIN16, WIN32, WINDLL

Group
_CSTR30_HASFILE

Example
```
CStr szLastOpenedDir;
szLastOpenedDir.GetThisDir();
szLastOpenedDir.MB( this, "Current dir is:" );
```

See also...
CStr::ChDir
CStr::MkDir
CStr::GetPathValue
CStr::DoesDirExist
CStr::DoesFileExist
CStr::GetAppDir
CStr::GetWinDir
CStr::GetWinSysDir
CStr::GetTempDir

CStr::GetTempFileName

CStr CStr::GetTempFileName( );

Description

Returns a temp file name or a blank string. GetTempFileName can use the TEMP directory as specified in AUTOEXEC.BAT. If there are no temporary file name available, the function fails and returns a blank string. CHANGES THE CURRENT CSTR VALUE.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASFILE

Example

```
CStr szTempFile;
szTempFile.GetTempFileName();
szTempFileName.MB( this, "temporary file name is:" );
```

See also...

CStr::ChDir
CStr::MkDir
CStr::GetPathValue
CStr::DoesDirExist
CStr::DoesFileExist
CStr::GetAppDir
CStr::GetWinDir
CStr::GetWinSysDir
CStr::GetThisDir
CStr::GetTempDir

CStr::GetTempDir

CStr CStr::GetTempDir();

Description

Returns the temporary directory to use or a blank string. GetTempDir can use the TEMP directory as specified in AUTOEXEC.BAT. If there are no temporary file name available, the function fails and returns a blank string. CHANGES THE CURRENT CSTR VALUE.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASFILE

Example

```
CStr szTempDir;
szTempDir.GetTempDir();
szTempDir.MB( this, "temporary directory is:" );
```

See also...

CStr::ChDir
CStr::MkDir
CStr::GetPathValue
CStr::DoesDirExist
CStr::DoesFileExist
CStr::GetAppDir
CStr::GetWinDir
CStr::GetWinSysDir
CStr::GetThisDir

CStr::IsCharUpper

BOOL CStr::IsCharUpper(int iIndex = 0) const;

Description
Returns TRUE if the 0 indexed <iIndex> character in the current CStr object is in uppercase.

Compatibility
WIN16, WIN32, WINDLL

Group
_CSTR30_HASCHAR

Example
```
CStr szCharTest = "Abcd";
if ( szCharTest.IsCharUpper(0))
 MessageBox("ok..");
```

See also...
CStr::IsCharLower
CStr::IsCharAlpha
CStr::IsCharAlphaNum
CStr::IsCharNumeric
CStr::IsSpace

CStr::IsCharLower

BOOL CStr::IsCharLower(int iIndex =0) const;

Description
Returns TRUE if the 0 indexed <iIndex> character in the current CStr object is in lowercase.

Compatibility
WIN16, WIN32, WINDLL

Group
_CSTR30_HASCHAR

Example
```
CStr szCharTest = "Abcd";
if ( szCharTest.IsCharLower(1))
 MessageBox("ok..");
```

See also...
CStr::IsCharUpper
CStr::IsCharAlpha
CStr::IsCharAlphaNum
CStr::IsCharNumeric
CStr::IsSpace

CStr::IsCharAlpha

BOOL CStr::IsCharAlpha(int iIndex =0) const;

Description
Returns TRUE if the 0 indexed <iIndex> character in the current CStr object is alphabetic.

Compatibility
WIN16, WIN32, WINDLL

Group
_CSTR30_HASCHAR

Example
```
CStr szCharTest = "Abcd";
if ( szCharTest.IsCharAlpha(0))
 MessageBox("ok..");
```

See also...
CStr::IsCharUpper
CStr::IsCharLower
CStr::IsCharAlphaNum
CStr::IsCharNumeric
CStr::IsSpace

CStr::IsCharAlphaNum

BOOL CStr::IsCharAlphaNum(int iIndex =0) const;

Description
Returns TRUE if the 0 indexed <iIndex> character in the current CStr object is alphanumeric (0-9, A-Z).

Compatibility
WIN16, WIN32, WINDLL

Group
_CSTR30_HASCHAR

Example
```
CStr szCharTest = "Abcd";
if ( szCharTest.IsCharAlphaNum(0))
 MessageBox("ok..");
```

See also...
CStr::IsCharUpper
CStr::IsCharLower
CStr::IsCharAlpha
CStr::IsCharNumeric
CStr::IsSpace

CStr::IsCharNumeric

BOOL CStr::IsCharNumeric(int iIndex =0) const;

Description
Returns TRUE if the 0 indexed <iIndex> character in the current CStr object is numeric (0-9).

Compatibility
WIN16, WIN32, WINDLL

Group
_CSTR30_HASCHAR

Example
```
CStr szCharTest = "A2cd";
if ( szCharTest.IsCharNumeric(1))
 MessageBox("ok..");
```

See also...
CStr::IsCharUpper
CStr::IsCharLower
CStr::IsCharAlpha
CStr::IsCharAlphaNum
CStr::IsSpace

CStr::IsSpace

BOOL CStr::IsSpace(int iIndex =0) const;

Description
Returns TRUE if the 0 indexed <iIndex> character in the current CStr object is the space character (0x20).

Compatibility
WIN16, WIN32, WINDLL

Group
_CSTR30_HASCHAR

Example
```
CStr szCharTest = "A bcd";
if ( szCharTest.IsSpace(1))
 MessageBox("ok..");
```

See also...
CStr::IsCharUpper
CStr::IsCharLower
CStr::IsCharAlpha
CStr::IsCharAlphaNum
CStr::IsCharNumeric

CStr::NumFormat

CStr CStr::NumFormat(double dValue, LPCTSTR szFormat = "###,###,###,###.00;-###,###,###,###.00; ", LPCTSTR szDecimalPoint = ".", LPCTSTR szThousandSeparator =",");

Description

This function formats the <dValue> number and puts it in the current CStr object. <dValue> is the number to use (don' t forget to include MATH.H in your STDAFX.H include file), <szFormat> is the format string to use (it' s quite similar to Microsoft Access): The string can contain up to 3 parts separated with ';': The first part is the pattern to use for positive numbers, the second one for negative number and the last one for the zero values. You must provide at least 1 format string. A signle space string will return a blank string. CHANGES THE CURRENT CSTR VALUE. To build the pattern you can use all the characters you wish but not the following characters since they will be replaced with the correct values:

- ü # For a digit or nothing,
- ü 0 For a digit or a '0',
- ü . (point) For the decimal point,
- ü , (comma) For the thousands separator,
- ü % For this sign and the value will be multiply by 100
- ü () Will be added if the number is less than 0.

Ä You must give all the necessary masks: For instance if your <dValue> is 1 000 000 and your <szFormat> string is ###,###.00, the returned string will be 1000,000.00. So, give ###,###,###.00 to obtain 1,000,000.00... You can also use '(' and ')' to format a negative number. <szDecimalPoint> is the character to use as the decimal point and <szThousandsSeparator> is the character to use as the thousands separator. Of course this function will round the returned string depending on the used format.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASNUM

Example

```
CStr szOut = "";
szOut.NumFormat( 123456.789,"###,###.00", (LPCTSTR)AfxGetDecimalPoint(),
(LPCTSTR)AfxGetThousandsSeparator() );
szOut.ToCtl(this, IDC_EDITAMOUNT);
```

See also...

CStr::From
CStr::NumUnformat
CStr::sprintf
CStr::From
CStr::ToInt
CStr::ToUINT
CStr::ToLong
CStr::ToULong
CStr::ToFloat
CStr::ToDouble
CStr::IsDigit
CStr::IsInRange
CStr::sprintf
AfxRound

AfxRoundToInt

CStr::NumUnformat

CStr CStr::NumUnformat(LPCTSTR szDecimalPoint = ".");

Description

The function removes the non digit characters from the current CStr value and uses <szDecimalPoint> as the (user) decimal point. If '(' or ')' are found, the function will add '-' at the begiining of the number. CHANGES THE CURRENT CSTR VALUE.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASNUM

Example

```
CStr szOut = "";
szOut.NumFormat( 123456.789,"###,###.00", (LPCTSTR)AfxGetDecimalPoint(),
(LPCTSTR)AfxGetThousandsSeparator() );
szOut.ToCtl(this, IDC_EDITAMOUNT);
szOut.NumUnformat( (LPCTSTR)AfxGetDecimalPoint());
szOut.MB();
```

See also...

CStr::NumFormat
CStr::From
CStr::ToInt
CStr::ToUINT
CStr::ToLong
CStr::ToULong
CStr::ToFloat
CStr::ToDouble
CStr::IsDigit
CStr::IsInRange
AfxRound
AfxRoundToInt

CStr::Align

CStr CStr::Align(int aMode, int iSize);

Description

Align the current CStr value. <aMode> is one of the following constants and <iSize> is the length (in characters) to use to build the new string:

- TA_LEFT 0x0000
- TA_RIGHT 0x0002
- TA_CENTER 0x0006

The string must not be longer than 511 characters. Returns the new string. CHANGES THE CURRENT CSTR VALUE.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASMOREBASIC

Example

```
CStr szAlignTest = "890";
szAlignTest.Align( TA_RIGHT, 10);
```

See also...

CStr::TabbedTextOut
CStr::GetTextWidth
CStr::GetTextHeight
CStr::Replicate
CStr::Blank
CStr::AddLF

CStr::MakeCharUpper

CStr CStr::MakeCharUpper(int iIndex =0);

Description

Put the specified <iIndex> character in the current CStr object in uppercase. <iIndex> is zero indexed. CHANGES THE CURRENT CSTR VALUE.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASCHAR

Example

```
CStr szMakeTest = "abcd";
szMakeTest.MakeCharUpper(3);
szMakeTest.MB();
```

See also...

CStr::MakeCharLower
CStr::IsCharUpper
CStr::IsCharLower

CStr::MakeCharLower

CStr CStr::MakeCharLower(int iIndex =0);

Description

Put the specified <iIndex> character in the current CStr object in lowercase. <iIndex> is zero indexed. CHANGES THE CURRENT CSTR VALUE.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASCHAR

Example

```
CStr szMakeTest = "abcd";
szMakeTest.MakeCharUpper(3);
szMakeTest.MB();
```

See also...

CStr::MakeCharUpper
CStr::IsCharUpper
CStr::IsCharLower

CStr::ExtractToEnd

CStr CStr::ExtractToEnd(int iFrom, char szSeparator);
CStr CStr::ExtractToEnd(int iFrom, LPCTSTR szSeparator = ";");

Description

Extracts some items of the current CStr object using the <szSeparator> string as an item separator. <szDelimit> must be 1 character length if you use the LPCTSTR version. The function returns the string part from <iFrom> : For instance, if the string is "item1;item2;item3" and <iFrom> is 2, the function returns "item2;item3". WARNING: <iFrom> is not 0 based (the first item is 1). CHANGES THE CURRENT CSTR OBJECT.

Compatibility

WIN16, WIN32, WINDLL

Group

<none>

Example

```
CStr szExtractTest = "item1;item2;item3";
szExtractTest.ExtractToEnd(2);
szExtractTest.MB();
```

See also...

CStr::Extract
CStr::ExtractToBegin

CStr::ExtractToBegin

CStr CStr::ExtractToBegin(int iFrom, char szSeparator);
CStr CStr::ExtractToBegin(int iFrom, LPCTSTR szSeparator = ";");

Description

Extracts some items of the current CStr object using the <szSeparator> string as an item separator. <szDelimit> must be 1 character length if you use the LPCTSTR version. The function returns the first string part to <iFrom> : For instance, if the string is "item1;item2;item3" and <iFrom> is 2, the function returns "item1;item2". WARNING: <iFrom> is not 0 based (the first item is 1). CHANGES THE CURRENT CSTR OBJECT.

Compatibility

WIN16, WIN32, WINDLL

Group

<none>

Example

```
CStr szExtractTest = "item1;item2;item3";
szExtractTest.ExtractToBegin(2);
szExtractTest.MB();
```

See also...

CStr::Extract
CStr::ExtractToEnd

CStr::FindWord

int CStr::FindWord(LPCTSTR szWord, int iFrom = 0, BOOL bExact = TRUE, BOOL bMatchCase =FALSE) const;

Description

The function will search for the <szWord> word in the current CStr object. If the word exists the function returns its zero based idex; if not found -1 is returned. <iFrom> is the zero based index from which the process must begin. Set <bExact> to FALSE if you wish the function also uses partial match. If you w<ish to distinguish charaters case, set <bMatchCase> to TRUE.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASFINDWORD

Example

```
CStr szFindTest = "FindWord test...";
if ( szFindTest.FindWord("FindWord") > -1 )
 MessageBox( "Found!");
```

See also...

CStr::FindWord
CStr::FindWordInMultilineBuffer
AfxFindWordInFile

CStr::FindWordInMultilineBuffer

BOOL CStr::FindWordInMultilineBuffer(LPCTSTR szWord, unsigned long iFromLine = 0L, int iFromCursor = 0, unsigned long* rLineValue = NULL, int* rCursorValue = NULL, BOOL bExact = TRUE, BOOL bMatchCase =FALSE) const;

Description

This function is the same than AfxFindWordInFile but it uses the current CStr object instead of a text file. <szWord> is the word to look for. <iFromLine> is the index of the line from which you want to search <szWord>; <iFromCursor> is the position in the line from which you want to begin the search process; Set <bExact> to TRUE if you want to exclude the partial names and <bMatchCase> to TRUE if you want to check the case too. <rLineValue> and <rCursorValue> are filled on output and they specify the word position in the file. If they are set to NULL they won' t be used by the function. As usual all indexes are 0 based. The function returns TRUE if <szWord> exist, FALSE if not.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASFINDWORD

Example

```
CStr szMsg, szLine, szCursor, szTextString;
unsigned long Y=0L;
int X=0;
szTextString = "This is the first line.\nThis is the second line!";
if ( szTextString.FindWordInMultilineBuffer("second", 0L, 0, &Y, &X ) )
 {
 szLine.From( Y );
 szCursor.From( X );
 szMsg = "'second' found at position ";
 szMsg += szLine;
 szMsg += ":";
 szMsg += szCursor;
 szMsg.MB();
 }
```

See also...

CStr::FindWord
AfxFindWordInFile

CStr::FromRes

CStr CStr::FromRes(UINT iStringID, HINSTANCE hInstance = NULL, LPCTSTR szItem1 = NULL, LPCTSTR szItem2 = NULL, LPCTSTR szItem3 = NULL, LPCTSTR szItem4 = NULL, LPCTSTR szItem5 = NULL, LPCTSTR szItem6 = NULL, LPCTSTR szItem7 = NULL, LPCTSTR szItem8 = NULL, LPCTSTR szItem9 = NULL);

Description

Loads a string resource from the resources and change the '%1' to '%9' values. <iStringID> is the string resource constant (IDS_...) to use to load the string. <hInstance> is the HINSTANCE to use to load the string: If <hInstance> is NULL, the function will call <AfxGetInstanceHandle()>. <szItem1> is the string to use to replace all the '%1' items in the resource string. <szItem2> to <szItem9> will do the same with '%2' to %9'. The function returns the new string. CHANGES THE CURRENT CSTR VALUE.

Compatibility

WIN16, WIN32, WINDLL

Group

<none>

Example

```
CStr szWarning;
szWarning.FromRes( IDS_BADFILE, NULL, szInputFile );
szWarning.MB(NULL, szAppName, MB_OK | MB_ICONSTOP);
```

See also...

CStr::CStr()

CStr::GetTextWidth

int CStr::GetTextWidth(HDC hDC, int iTabCount = 0, int far* pTabList = NULL) const;
int CStr::GetTextWidth(CDC* pDC , int iTabCount = 0, int far* pTabList = NULL) const;
int CStr::GetTextWidth(CWnd* pWnd , int iTabCount = 0, int far* pTabList = NULL) const;

Description

Returns the current CStr text width in logical units using an HDC, a CDC* or a CWnd*. If the string includes some customised tab keys you must fill <iTabCount> and <pTabList>. <iTabCount> is the number of values in the <pTabList> array. <pTabList> is a tab key array (see <CDC::GetTabbedTextExtent> for possible values).

Compatibility

WIN16, WIN32 (CWnd/CDC only), WINDLL

Group

_CSTR30_HASMOREBASIC

Example

```
CStr szMyString = "This is my string...", szMsg;
szMsg.From( szMyString.GetTextWidth(this));
szMsg.MB( this, "String width:" );
```

See also...

CStr::TabbedTextOut
CStr::GetTextHeight
CStr::Replicate
CStr::Blank
CStr::AddLF

CStr::GetTextHeight

int CStr::GetTextHeight(HDC hDC , int iTabCount = 0, int far* pTabList = NULL) const;
int CStr::GetTextHeight(CDC* pDC , int iTabCount = 0, int far* pTabList = NULL) const;
int CStr::GetTextHeight(CWnd* pWnd , int iTabCount = 0, int far* pTabList = NULL) const;

Description

Returns the current CStr text height in logical units using an HDC, a CDC* or a CWnd*. If the string includes some customised tab keys you must fill <iTabCount> and <pTabList>. <iTabCount> is the number of values in the <pTabList> array. <pTabList> is a tab key array (see <CDC::GetTabbedTextExtent> for possible values).

Compatibility

WIN16, WIN32 (CWnd/CDC only), WINDLL

Group

_CSTR30_HASMOREBASIC

Example

```
CStr szMyString = "This is my string...", szMsg;
szMsg.From( szMyString.GetTextHeight(this));
szMsg.MB( this, "String height:" );
```

See also...

CStr::GetTextWidth
CStr::TabbedTextOut
CStr::Replicate
CStr::Blank
CStr::AddLF

CStr::GetWindowTitle

CStr GetWindowTitle(CWnd* pWnd, BOOL bIncDocName = FALSE); [3.0]
CStr GetWindowTitle(HWND hWnd, BOOL bIncDocName = FALSE); [3.0]

Description

Return a window title (caption). <pWnd> or <hWnd> is the window to use. If <bIncDocName> is set to TRUE the function will return the first part of the caption if a document name is specified. To do this the document name must use '-' as deilimiter. The function supports reversed titles as Win95 does. CHANGES THE CURRENT CSTR VALUE.

Compatibility

WIN16, WIN32, WINDLL

Group

<none>

Example

```
CStr szWindowTitle;
szWindowTitle.GetWindowTitle(this);
szWindowTitle.MB(this, "Current window title is:");
```

See also...

CStr::GetClassName

CStr::TabbedTextOut

CSize CStr::TabbedTextOut(CDC* pDC, int X, int Y, int iTabCount = 0, int far* TabList = NULL) const;
CSize CStr::TabbedTextOut(CWnd* pWnd, int X, int Y, int iTabCount = 0, int far* TabList = NULL) const;

Description

Display the current CStr string using a CDC* or a CWnd*. <X> and <Y> are the coordinates to use to display the string. If the string includes some customised tab keys you can fill <iTabCount> and <pTabList>. <iTabCount> is the number of values in the <pTabList> array. <pTabList> is a tab key array (see <CDC::GetTabbedTextExtent> for possible values). Returns a CSize structure with the height and the width for the string in logical units.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASMOREBASIC

Example

```
CStr szMsg = "Direct output";
szMsg.TabbedTextOut( pDC, 10, 10);
```

See also...

CStr::GetTextWidth
CStr::GetTextHeight
CStr::Replicate
CStr::Blank
CStr::AddLF

CStr::GetClassName

int CStr::GetClassName(CWnd* pWnd);
int CStr::GetClassName(HWND hWnd);

Description

Fill the current CStr object with the classname of the specified CWnd* or HWND. The function returns the length of this name. CHANGES THE CURRENT CSTR VALUE.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASMOREBASIC

Example

```
CStr szMyClass;
szMyClass.GetClassName( this );
szMsg.MB( this, "Current window class name is:" );
```

See also...

CStr::GetWindowTitle

CStr::GetWinfileAssociation

CStr CStr::GetWinfileAssociation(LPCTSTR szExtension);

Description

Returns the associated WinFiles program for a kind of document. <szExtension> can be the extension (like 'ZIP') or a file name (like 'MYFILE.ZIP'). Do not use full file names since these names can include a directory extension. If there is no associated program for this kind of document a blank string is returned. CHANGES THE CURRENT CSTR VALUE.

Compatibility

WIN16, WIN32, WINDLL

Group

<none>

Example

```
CStr szProgram;
szProgram.GetWinfileAssociation( "TXT");
szProgram.MB( this, "Program for TXT files is:" );
```

CStr::GetDlgItemText

int CStr::GetDlgItemText(CWnd* pWnd, UINT iControl);

Description

Same as FromCtl. Retrieve a dialog control text. <pWnd> is the dialog box CWnd* ('this') and <iControl> is the control ID as specified in the RESOURCE.H file. The function returns the text length. CHANGES THE CURRENT CSTR VALUE.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASWINCTL

Example

```
CStr szLabel1Text;
szLabel1Text.GetDlgItemText( this, IDC_STATICLABEL1 );
szLabel1Text.MB( this, "IDC_STATICLABEL1 text is:" );
```

See also...

CStr::ToCtl
CStr::FromCtl
CStr::GetDlgItemText

AfxGiveTheHand

void AfxGiveTheHand(HWND hDialog = NULL); [3.0]

Description

This function flushes the Windows message loop, so that the other applications 'have the hand'. This is specially useful when you use a big loop which may freeze Windows. It' s similar to the VB <DoEvent> function. If you give a valid <hDialog> HWND, the function won' t treat the messages for this dialog box.

Compatibility

WIN16, WIN32, WINDLL

Group

<none>

Example

```
int i;
while ( i < 1000 )
    {
    AfxGiveTheHand();
    i++;
    }
```

See also...

AfxSleep

AfxSleep

void AfxSleep(int iSecs, BOOL bBackGnd = TRUE); [3.0]

Description

This function waits for <iSecs> seconds. If <bBackGnd> is set to TRUE, the user can use others programs during this delay, if not, the WAIT cursor is displayed during this delay (in this case your application must own a CWinApp class).

Compatibility

WIN16, WIN32, WINDLL

Group

<None>

Example

```
AfxSleep(3);
```

See also...

AfxGiveTheHand

AfxSetFileTime

BOOL AfxSetFileTime(LPCTSTR szSrcFile, LPCTSTR szDestFile); [3.0]

Description

Thanks to this function, the last modification date/time of the existing file <szDestFile> will be changed using the <szSrcFile> date/time value. The function returns TRUE if both files own the same date. This function doesn' t work with directories (but you can change a directory attributes with AfxSetFileAttributes or CStr::SetFileAttributes)

Ä If you plan to use AfxSetFileAttributes too, you must call AfxSetFileTime before because the function may change the file attributes in case it is read-only.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASCTIM

Example

```
AfxSetFileTime( "C:\AUTOEXEC.BAT", "C:\AUTOEXEC.BAK");
```

See also...

AfxSetFileAttributes
AfxTouchFile
CTim::ToFile
CTim::FromFile
CStr::GetFileAttributes
CStr::SetFileAttributes

AfxSetFileAttributes

BOOL AfxSetFileAttributes(LPCTSTR szSrcFile, LPCTSTR szDestFile); [3.0]

Description

Thanks to this function, the attributes of the existing file <szDestFile> will be replaced by the <szSrcFile> attributes. The function returns TRUE if both files own the same attributes.

Ä If you plan to use AfxSetFileTime too, you must call AfxSetFileTime before because the function may change the file attributes in case it is read-only.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASFILE

Example

```
AfxSetFileAttributes( "C:\AUTOEXEC.BAT", "C:\AUTOEXEC.BAK");
```

See also...

AfxSetFileTime
AfxTouchFile
CStr::IsRDOnly
CStr::GetFileAttributes
CStr::SetFileAttributes
CTim::ToFile
CTim::FromFile

AfxTouchFile

BOOL AfxTouchFile(LPCTSTR szFile); [3.0]

Description

AfxTouchFile changes the date/time of the existing <szFile> file using the current system date/time. Returns TRUE if successful. This function doesn' t work with directories (but you can change a directory attributes with AfxSetFileAttributes or CStr::SetFileAttributes)

Ä If you plan to use AfxSetFileAttributes too, you must call AfxTouchFile before because the function may change the file attributes in case it is read-only.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASFILE

Example

```
AfxTouchFile( "C:\LASTCHK.FLG");
```

See also...

AfxSetFileTime
CTim::ToFile
CTim::FromFile

CStr::sprintf

CStr CStr::sprintf(LPCTSTR szFormat, ...); [3.0]

Description

Same as wvsprintf (for the Windows version) or vsprintf but doesn' t require you retrieve the string buffer first. Returns the new string. CHANGES THE CURRENT CSTR VALUE.

Ä Current implementation of CStr::spintf uses a 512 char buffer: Do not use more characters..

Compatibility

WIN16, WIN32, WINDLL

Group

<None>

Example

```
CStr szMyString;
szMyString.sprintf("Free memory rate is: %i",
(int)AfxGetFreeResRate( FREERES_MEM) );
szMsg.MB( this, "sprintf sample" );
```

See also...

CStr::NumFormat
CStr::GetBuffer

CStr::GetFileAttributes

DWORD CStr::GetFileAttributes( ) const; [3.0]

Description

Returns the current WIN32 files attributes for the current CStr file. Files attributes may be one or more following attributes:

- FILE_ATTRIBUTE_READONLY 0x00000001
- FILE_ATTRIBUTE_HIDDEN 0x00000002
- FILE_ATTRIBUTE_SYSTEM 0x00000004
- FILE_ATTRIBUTE_DIRECTORY 0x00000010
- FILE_ATTRIBUTE_ARCHIVE 0x00000020
- FILE_ATTRIBUTE_NORMAL 0x00000080

Notice this function is available under Win16 and Win32.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASFILE

Example

```
DWORD dwResult = 0L;
CStr strSrcFile("C:\TESTSRC.TXT"), strDestFile("C:\TESTDEST.TXT");
dwResult = strSrcFile.GetFileAttributes();
if ( !( dwResult & FILE_ATTRIBUTE_DIRECTORY ) )
    strDestFile.SetFileAttributes( dwResult );
```

See also...

CStr::SetFileAttributes
CStr::IsRDOnly
CStr::GetFileSize
AfxSetFileAttributes

CStr::SetFileAttributes

BOOL CStr::SetFileAttributes(DWORD dwNewAttrib) const; [3.0]

Description

Changes the current CStr file using the Win32 <dwNewAttrib> attributes. The file must exist. The function returns TRUE if the new attributes were set. Files attributes may be one or more following attributes:

- ü FILE_ATTRIBUTE_READONLY 0x00000001
- ü FILE_ATTRIBUTE_HIDDEN 0x00000002
- ü FILE_ATTRIBUTE_SYSTEM 0x00000004
- ü FILE_ATTRIBUTE_DIRECTORY 0x00000010
- ü FILE_ATTRIBUTE_ARCHIVE 0x00000020
- ü FILE_ATTRIBUTE_NORMAL 0x00000080

Ä While this function uses Win32 file attributes, it is available under Win16 too. Also, if you plan to change the file date too, you must call SetFileAttributes once the date/time is changed since CTim::ToFile may change the file attributes in case it is read-only.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASFILE

Example

```
DWORD dwResult = 0L;
CStr strSrcFile("C:\TESTSRC.TXT"), strDestFile("C:\TESTDEST.TXT");
dwResult = strSrcFile.GetFileAttributes();
if ( !( dwResult & FILE_ATTRIBUTE_DIRECTORY ) )
    strDestFile.SetFileAttributes( dwResult );
```

See also...

CStr::GetFileAttributes
CStr::IsRDOnly
CStr::GetFileSize
AfxSetFileAttributes

CStr::IsRDOnly

BOOL CStr::IsRDOnly( ) [3.0]

Description
Returns TRUE if the current CStr file is read-only.

Compatibility
WIN16, WIN32, WINDLL

Group
_CSTR30_HASFILE

Example
```
CStr strSrcFile("C:\TESTSRC.TXT");
if ( strSrcFile.IsRDOnly() )
    strSrcFile.MB(this, "READ-ONLY");
```

See also...
CStr::GetFileAttributes
CStr::SetFileAttributes
CStr::GetFileSize
AfxSetFileAttributes

CStr::GetFileSize

unsigned long CStr::GetFileSize(unsigned long* pulSizeInBytes = NULL) const; [3.0]

Description

Retrieve the current CStr file size. If successful, the returned value is in KBytes, but, if the pointer <pulSizeInBytes> isn' t NULL, it will contain the file size in bytes. Under Win32, this function isn' t useful if you handle files larger than ULONG_MAX.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASFILE

Example

```
unsigned long ulResult = 0L;
CStr strSrcFile("C:\TESTSRC.TXT");
dwResult = strSrcFile.GetFileSize();
```

See also...

CStr::GetFileAttributes
CStr::SetFileAttributes
CStr::IsRDOnly
AfxSetFileAttributes

CStr::DeleteFile

BOOL CStr::DeleteFile(BOOL bDeleteRO = TRUE) const; [3.0]

Description

Deletes the current CStr file. If <bDeleteRO> is set to TRUE, even read-only files can be removed. The function returns TRUE if the file doesn' t exist any longer, FALSE if not.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASFILE

Example

```
CStr strSrcFile("C:\TESTSRC.TXT");
strSrcFile.DeleteFile();
```

See also...

CStr::DoesFileExist
CStr::GetFileAttributes
CStr::SetFileAttributes
CStr::IsRDOnly
CStr::RemoveDir
CStr::GetFileSize
CStr::CreateFile
AfxSetFileAttributes

CStr::IsWriteable

BOOL CStr::IsWriteable(long lBytes = 0L) const; [3.0]

Description

This function is useful when you want to write some data on a diskette. Give the size of the data you want to write in bytes as <lBytes> and check the returned value.The current CStr value must be the destnation file or directory.

Ä This function should not be used during a copy process (i.e. when a file is opened). In this case you can use IsWriteable before the process then, the TRY/CATCH macros during the process.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASFILE

Example

```
CStr szMyDestFile("A:\");
if ( !szMyDestFile.IsWriteable() )
    szMyDestFile.MB( this, "Drive not ready" );
```

See also...

CStr::DoesFileExist
CStr::GetDiskFreeSpace

CStr::GetDiskFreeSpace

long CStr::GetDiskFreeSpace( ) const; [3.0]

Description

Returns the free space in KBytes for the current CStr file or directory.

Ä This function should not be used during a copy process (i.e. when a file is opened). In this case you can use GetDiskFreeSpace before the process then, the TRY/CATCH macros during the process.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASMOREBASIC

Example

```
CStr szMyDestFile("A:\");
if ( szMyDestFile.GetDiskFreeSpace() < 1L )
    szMyDestFile.MB( this, "Drive not ready" );
```

See also...

CStr::IsWriteable

CStr::CopyTo

int CStr::CopyTo(LPCTSTR szDestFile, BOOL bDeleteDest = TRUE, BOOL bChangeDate = TRUE, BOOL bChangeAttrib = TRUE , BOOL bDeleteSrc = FALSE, int iBufferSize = 4096 | 16384) const; [3.0]

Description

This function copies an existing file. <szDestFile> is the new file you wish to build. It can be a directory (with '\') or a file name. If one or more directories don' t exist, they will be built during the process. If <bDeleteDest> is TRUE, the function will delete the destination file in case that one already exists. Also, set <bChangeDate> and <bChangeAttrib> to TRUE if you want the new file has the same date/attributes as the source file. Once done, if <bDeleteSrc> is TRUE the source file will be removed.

You can change the internal buffer size thanks to <iBufferSize>. Default value for <iBufferSize> is 4096 bytes with _WINDLL or 16384 bytes. There is no background processing during the <CopyTo> process. The function returns one of the following code:

ü COPYTO_SUCCESS	0
ü COPYTO_NOTFOUND	1
ü COPYTO_BADSRCFNAME	2
ü COPYTO_BADDESTFNAME	3
ü COPYTO_NOMEM	4
ü COPYTO_CANTCREATEDEST	5
ü COPYTO_READERROR	6
ü COPYTO_WRITEERROR	7
ü COPYTO_DISKFULL	8
ü COPYTO_CANTDELETEDEST	9
ü COPYTO_NOATTRIB	10
ü COPYTO_NODATE	11
ü COPYTO_CANTDELETESRC12	

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASFILE

Example

```
CStr szMySrcFile("C:\SRCTST.TXT");
szMySrcFile.CopyTo( "D:\" );
```

See also...

CStr::CreateFile
CStr::DeleteFile

CStr::RemoveDir

BOOL CStr::RemoveDir(BOOL bIncFile = FALSE, BOOL bDeleteRO = TRUE) const; [3.0]

Description

Removes one or more directory even in case they contains read-only files. Set <bIncFile> to TRUE if your current CStr object is a file name instead of a diretory name. If <bDeleteRO> is FALSE, the function won' t try to delete the read-only files/directories. Returns TRUE if the specified directory doesn' t exist any longer.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASFILE

Example

```
CStr szMyDir("A:\OLD\BACKUP\";
szMyDir.RemoveDir();
```

See also...

CStr::CreateFile
CStr::DeleteFile
CStr::IsRDOnly

CStr::IsUNC

BOOL CStr::IsUNC( ) const; [3.0]

Description

Returns TRUE if the current file name doesn' t have a drive specification ("A:"), but begins with a valid server name. This server must be available. UNC names look like '\\server\shared\dirs..\file'. Returns FALSE if the file doesn' t have an UNC file name.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASUNC

Example

```
CStr szMyUNCFile("\\\\ourserver\\bin\\hnetworks\\tstfile.txt");
if ( szMyUNCFile.IsUNC())
    szMyUNCFile.MB( this, "Following UNC name recognized:" );
```

CStr::CreateFile

BOOL CStr::CreateFile(UINT uiWin16Attrib = OF_SHARE_COMPAT) const; [3.0]

Description

Build a new file or replace an old file. If the current file has subdirectories, they will be built if necessary. Once built, the file will be closed. If you are using Win16, you may give an <uiWin16Attrib> like OF_SHARE_COMPAT (see 'OpenFile' in WIN31WH.HLP). Returns TRUE if the new file exists, FALSE if not.

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASFILE

Example

```
CStr szMyNewFile("A:\TSTNEW.TXT");
szMyNewFile.CreateFile();
```

See also...

CStr::DeleteFile
CStr::ToFile
CStr::FromFile

CMem

Description

String Studio uses the CMem class for its buffers. CMem simplify HGLOBAL handles and buffers management for 2 reasons: First, GlobalAlloc/GlobalLock and GlobalUnlock/GlobalFree are grouped. And, the CMem destructor frees the memory so that it' s not necessary to manually do it. If you use global CMem objects, you may need to call <Free> during your program execution.

Functions

ü CMem()	Simple constructor
ü CMem(DWORD dwSize, UINT uFlags)	Constructor with allocation
ü ~CMem()	Destructor which frees pointers
ü Alloc(DWORD dwSize, UINT uFlags)	Allocates a buffer
ü ReAlloc(DWORD dwSize, UINT uFlags)	ReAllocates a buffer
ü GetHandle()	Returns the HGLOBAL
ü GetPtr()	Returns the pointer
ü GetSize()	Returns the actual buffer size
ü IsValid()	TRUE if the pointer is ok.

How to use CMem

In most case you will do like this:

```
LPTSTR szBuffer;            // Your buffer
CMem cm1;             // Your CMem object
/*
It' s better to use the simple constructor instead
of allocating the buffer when you declare your CMem
object
*/
szBuffer = (LPTSTR)cm1.Alloc( 1024L );
/*
Don' t forget to cast the Allocated pointer..
*/
if ( szBuffer == NULL )
/*
Make sure that <szBuffer> is here.
*/
...
if ( !cm1.IsValid() )
/*
If you ask for a big pointer, make sure that <szBuffer>
size is >= 1024..
*/
```

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASCMEM

CFFind

Description

CFFind is a simple wrapper for _dos_find.. (under Win16) functions and Find... (under Win32) functions. There are 2 advantages: First of all you can use the same class, with the same arguments for all your projects; And, you may use the <Find> method instead of <FindFirst> and <FindNext>. As usual, we use Win32 files attributes instead of Win16 (DOS) attributes.

Functions

---

q There are 2 constructors:
ü **CFFind()**
ü **CFFind(LPCTSTR szPath, UINT win16Attrib)**

When you use the simple constructor, you must call <SetPath> to give the path you want to check out. <win16Attrib> will be ignored on Win32. Under Win31 you may give one or more _A_.. attrib to perform an attributes based search.

To change the path you can use:
ü **SetPath(LPCTSTR szPath, UINT win16Attrib)**
ü **Reset()**

---

q **T**o begin the search process use either:
ü **Find()**

or
ü **FindFirst()**
ü **FindNext()**

Find will call <FindFirst> or <FindNext>. These 3 functions returns TRUE if a file is found.

---

q Now to retrieve the details about this file, use:

ü **DWORD GetAttributes();**
To retrieve the Win32 attributes (FILE_ATTRIBUTE_...) for the current file.

ü **DWORD GetSize();**
To retrieve the size (in bytes) for the current file.

ü **GetCreationTime(CTim* pTim);**
ü **GetLastAccessTime(CTim* pTim);**
ü **GetLastWriteTime(CTim* pTim);**
Which fill CTim with the file time. These 3 dates are the same under Win16.

ü **GetShortName(LPTSTR szName);**
ü **GetShortName(CString& szName);**
Retrieve the short file name (DOS formatted).

ü **GetFullShortName(LPTSTR szName);**
ü **GetFullShortName(CString& szName);**
Same as <GetShortName> but includes the full path.

ü **GetLongName(LPTSTR szName);**
ü **GetLongName(CString& szName);**

Retrieve the long file name (up to 254 chars). This is the same as GetShortName under Win16.

Ü **GetFullLongName(LPTSTR szName);**
Ü **GetFullLongName(CString& szName);**
Same as <GetLongName> but includes the full path.

Example

```
CFFind ff;
CStr strMsg;
int Y = 0;

ff.SetPath("C:\TEST\*.TXT");
while ( ff.Find() )
   {
   Y+=15;
   strMsg.sprintf("%s size is %ld bytes.\n", ff.GetShortName(),
   (long)ff.GetSize());
   strMsg.TabbedTextOut( this, 10, Y );
   }
```

Compatibility

WIN16, WIN32, WINDLL

Group

_CSTR30_HASCFFIND

Tips to avoid linker messages

VC++ 2.x messages

There are 2 common error messages you can get: with the 32bits linker:

- **Unresolved external: WNetGetConnection...**
  This message occurs because you don' t link with MPR.LIB. By default, VC++ (2.52) doesn' t include this static library. To add the library, choose [Projet][Settings] then click on 'Link' tab and choose 'General' in the 'Gategory' combobox. Now, in the 'Object/Library' field, simply add 'MPR.LIB'. If you own the String Studio, undefine _CSTR30_HASUNC if you don' t want to use it.

- **Unresolved external: _beginThreadEx...**
  This is because you want to build a 'Single Thread'' MFC application whereas the default settings required a 'Multiple threads' application. The string studio Win32 LIB file was build with the 'Multiple threads' application mode. To change it, choose [Project][Settings] and thanks to the 'C/C++' tab choose 'Code generation' in the 'Catogory' combobox. Then Set the 'Use Runtime Library' combobox value to 'Multithreaded'.

All versions tips

As you know, the String Studio evaluation LIB files are static libraries compiled without debuging informations. We do this to distribute a small ZIP file.. So, to evaluate the toolkit, please do not use the _DEBUG mode (it doesn' t work). Still with the unregistered version, you may not build a DLL because the library doesn' t use this flag.

AfxHasWin4Look

BOOL AfxHasWin4Look( ); [3.0]

Description

Returns TRUE if the current Windows version is higher than (or equal to) 3.52. This function may be use with VC++ 1.5x to disable the background color (which is false under Win95) under Win95. See the sample below.

Compatibility

WIN16, WIN32, WINDLL

Group

<None>

Example

```
CWinApp::InitInstance(..
if ( !AfxHasWin4Look() )
    SetDialogBkColor();
...
```

See also...

AfxCtl3dLoad