

# Harbour Guide

## TFileRead()

Read a file one line at a time

### Syntax

```
oFile := TFileRead():New( <cFileName> [, <nReadSize> ] )
```

### Arguments

<cFileName> is the required name of the file to be read.

<nReadSize> is the optional size to use when reading from the file. The default value is 4096 and the allowed range is 1 through 65535. Any value outside of this range causes the default value to be used.

### Returns

### Description

TFileRead() is used to access a file one line at a time. You must specify the name of the file when an instance of the class is created. The class data should be considered private to the class.

The class methods are as follows:

New() Creates a new instance of the TFileRead class.

Open([<nFlags>]) Opens the file for reading. The optional nFlags parameter can use any of the FOPEN() flags from fileio.ch. The default is FO\_READ + FO\_SHARED. Calling this method when the file is already open causes the next ReadLine() to start over from the beginning of the file.

Close() Closes the file.

ReadLine() Returns one line from the file, stripping the newline characters. The following sequences are treated as one newline: 1) CR CR LF; 2) CR LF; 3) LF; and 4) CR. Note: LF CR is 2 newlines.

Name() Returns the name of the file.

IsOpen() Returns .T. if the file is open.

MoreToRead() Returns .T. if there are more lines to be read (think of it as an inverse EOF function).

Error() Returns .T. if an error has occurred.

ErrorNo() Returns the current error code.

ErrorMsg([<cPre>]) Returns a formatted error message.

### Examples

```
#ifdef __HARBOUR__
#define NEW_LINE CHR( 10 )
#else
#define NEW_LINE CHR( 13 ) + CHR( 10 )
#endif
#include "fileio.ch"

PROCEDURE Main( cFile )
LOCAL oFile := TFileRead():New( cFile )

oFile:Open()
IF oFile:Error()
    QOUT( oFile:ErrorMsg( "FileRead: " ) )
ELSE
    WHILE oFile:MoreToRead()
        OUTSTD( oFile:ReadLine() )
        OUTSTD( NEW_LINE )
    END WHILE
oFile:Close()
END IF
QUIT
```

## Tests

See Examples

## Status

Ready

## Compliance

This is a new Harbour Tools class

## Files

Library is libmisc

## See Also:

[TFileRead\(\)](#)

## **ISBIN()**

Check if the value is a Binary Number

### **Syntax**

```
ISBIN(<cN>) -><cNr>
```

### **Arguments**

<cN> STRING TO BE CHECKED

### **Returns**

<cNr> .T. IF THE STRING IS BYNARY,otherwise .F.

### **Description**

check if the passed string is a bynary number or not

### **Files**

Library is libmisc

### **See Also:**

[ISOCTAL\(\)](#)

[ISDEC\(\)](#)

[ISHEXA\(\)](#)

## **ISOCTAL()**

Check if the value is a Octal Number

### **Syntax**

```
ISOCTAL(<cN>) -><cNr>
```

### **Arguments**

<cN> STRING TO BE CHECKED

### **Returns**

<cNr> .T. IF THE STRING IS OCTAL;otherwise .F.

### **Description**

check if the passed string is a octal number or not

### **Files**

Library is libmisc

### **See Also:**

[ISBIN\(\)](#)

[ISDEC\(\)](#)

[ISHEXA\(\)](#)

## **ISDEC()**

Check if the value is a Decimal Number

### **Syntax**

```
ISDEC(<cN>) -><cNr>
```

### **Arguments**

<cN> STRING TO BE CHECKED

### **Returns**

<cNr> .T. IF THE STRING IS DECIMAL;otherwise .F.

### **Description**

check if the passed string is a decimal number or not

### **Files**

Library is libmisc

### **See Also:**

[ISOCTAL\(\)](#)

[ISBIN\(\)](#)

[ISHEXA\(\)](#)

## **ISHEXA()**

Check if the value is a Hexal Number

### **Syntax**

```
ISHEXA(<cN>) -><cNr>
```

### **Arguments**

<cN> STRING TO BE CHECKED

### **Returns**

<cNr> .T. IF THE STRING IS HEXA;otherwise .F.

### **Description**

check if the passed string is a hexa number or not

### **Files**

Library is libmisc

### **See Also:**

[ISOCTAL\(\)](#)

[ISDEC\(\)](#)

[ISBIN\(\)](#)

## DECTOBIN()

Converts a Decimal Value to Binary

### Syntax

```
DECTOBIN(<cN>) -><cNr>
```

### Arguments

<cN> NUMBER TO BE CONVERTED

### Returns

<cNr> NUMBER CONVERTED

### Description

This function converts a string <cN> from an decimal value to an binary value.

### Files

Library is libmisc

### See Also:

[DECTOHEXA\(\)](#)

[DECTOOCTAL\(\)](#)

## **DECTOOCTAL( )**

Converts a Decimal Value to Octal

### **Syntax**

```
DECTOOCTAL(<cN>) -><cNr>
```

### **Arguments**

<cN> NUMBER TO BE CONVERTED

### **Returns**

<cNr> NUMBER CONVERTED

### **Description**

This function converts a string <cN> from an decimal value to an octal value.

### **Files**

Library is libmisc

### **See Also:**

[DECTOHEXA\(\)](#)

[DECTOBIN\(\)](#)

## **DECTOHEXA( )**

Converts a Decimal Value to Hexa

### **Syntax**

```
DECTOHEXA(<cN>) -><cNr>
```

### **Arguments**

<cN> NUMBER TO BE CONVERTED

### **Returns**

<cNr> NUMBER CONVERTED

### **Description**

This function converts a string <cN> from an decimal value to an hexadecimal value.

### **Files**

Library is libmisc

### **See Also:**

[DECTOBIN\(\)](#)

[DECTOOCTAL\(\)](#)

## **BINTODEC()**

Converts a Binary Value to Decimal

### **Syntax**

```
BIntODEC(<cN>) -><cNr>
```

### **Arguments**

<cN> NUMBER TO BE CONVERTED

### **Returns**

<cNr> NUMBER CONVERTED

### **Description**

This function converts a string <cN> from an binary value to a numeric decimal value.

### **Files**

Library is libmisc

### **See Also:**

[OCTALTODEC\(\)](#)

[HEXATODEC\(\)](#)

## OCTALTODEC()

Converts a Octal Value to Decimal

### Syntax

```
OCTALTODEC(<cN>) -><cNr>
```

### Arguments

<cN> NUMBER TO BE CONVERTED

### Returns

<cNr> NUMBER CONVERTED

### Description

This function converts a string <cN> from an octal value to a numeric decimal value.

### Files

Library is libmisc

### See Also:

[BINTODEC\(\)](#)

[HEXATODEC\(\)](#)

## HEXATODEC()

Converts a Hexa Value to Decimal

### Syntax

```
HEXATODEC(<cN>) -><cNr>
```

### Arguments

<cN> NUMBER TO BE CONVERTED

### Returns

<cNr> NUMBER CONVERTED

### Description

This function converts a string <cN> from an hexadecimal value to a numeric decimal value.

### Files

Library is libmisc

### See Also:

[OCTALTODEC\(\)](#)

[BINTODEC\(\)](#)

## FIELDTYPE( )

Determines the type of a given field.

### Syntax

```
FIELDTYPE(<nFieldNum>) --> cFieldType
```

### Arguments

<nFieldNum> Data field , which type need to be determined.

### Returns

FIELDTYPE() returns the character that designates the type of a given field:

'C'	character string;
'N'	numeric;
'L'	logical;
'D'	date;
'M'	memo.

### Description

This function determines the type of a field, designated by its number.

### Examples

```
FUNCTION Main()  
LOCAL i  
USE Tests NEW  
FOR i = 1 TO FCOUNT()  
    ? FieldType( i )  
NEXT  
USE  
RETURN NIL
```

### Status

Ready

### Compliance

This function is CA-CLIPPER tools compatible

### Files

Library is libmisc

### See Also:

[FIELD SIZE\(\)](#)

[FIELD DECI\(\)](#)

## **FIELD SIZE( )**

Determines the size of a given field.

### **Syntax**

```
FIELD SIZE(<nFieldNum>) --> nFieldSize
```

### **Arguments**

<nFieldNum> Data field , which size need to be determined.

### **Returns**

**FIELD SIZE( )** returns the number that designates the size of a given field.

### **Description**

This function determines the size of a field, designated by its number.

### **Examples**

```
FUNCTION Main()  
LOCAL i  
USE Tests NEW  
FOR i = 1 TO FCOUNT()  
    ? FieldSize( i )  
NEXT  
USE  
RETURN NIL
```

### **Status**

Ready

### **Compliance**

This function is CA-CLIPPER tools compatible

### **Files**

Library is libmisc

### **See Also:**

[FIELDTYPE\( \)](#)

[FIELDDECI\( \)](#)

## **FIELDDECI( )**

Determines the number of decimal places of a given numeric field.

### **Syntax**

```
FIELDDECI(<nFieldNum>) --> nFieldDeci
```

### **Arguments**

**<nFieldNum>** Numeric data field , for which number of decimal places need to be determined.

### **Returns**

**FIELDDECI()** returns the numeric value that designates the number of decimal places of a given field.

### **Description**

This function determines the number of decimal places of a given numeric field.

### **Examples**

```
FUNCTION Main()  
LOCAL i  
USE Tests NEW  
FOR i = 1 TO FCOUNT()  
    ? FieldDeci( i )  
NEXT  
USE  
RETURN NIL
```

### **Status**

Ready

### **Compliance**

This function is CA-CLIPPER tools compatible

### **Files**

Library is libmisc

### **See Also:**

[FIELDTYPE\(\)](#)

[FIELD SIZE\(\)](#)

**THtml()**  
Html Class

## Syntax

```
oHtml:=THtml():New(<cFile>) --> oHtml
```

## Arguments

<cFile> Name of the Html file to create

## Returns

<oHtml> An instance of the THtml Class

## Description

THtml() is a class that creates an .html file of the same name you pass to the constructor.

The class methods are as follows:

New(<cFile>) Create a new instance of the THtml class

Close() Close the created file

WriteTitle(<cTitle>) Write the file title

WritePar(<cPar>) Writes a paragraph

WriteParBold(<cPar>) Same as WritePar(), but the text is bold

WriteLink(<cLink>,<cName>) Write a link to another topic

WriteText(<cText>) Write any text

## Examples

```
FUNCTION MAIN()  
  
LOCAL oHtml  
  
oHtml := THTML():New( "www\harbour.html" )  
oHtml:WriteTitle( "Harbour Reference Guide" )  
oHtml:WritePar( "HARBOUR" )  
oHtml:WriteLink( "OverView" )  
oHtml:WriteLink( "License" )  
oHtml:WriteLink( "http://www.gnu.org/copyleft/gpl" )  
oHtml:WritePar( "See the Links Above" )  
oHtml:Close()  
RETURN Nil  
</par>
```

## Status

Ready

## Compliance

This is a new Harbour Tools class

## Platforms

ALL

## See Also:

[TFileRead\(\)](#)

# T0s2()

OS/2 Documentation Class

## Syntax

```
oNg:=T0s2():New(<cFile>) --> oOs2
```

## Arguments

<cFile> Name of the IPF Source file to create

## Returns

<oOs2> An instance of the T0s2 Class

## Description

T0s2() is a class that creates the OS/2 IPF Source of the same name you pass to the constructor.

The class methods are as follows:

New(<cFile>)	Create a new instance of the T0s2 class
Close()	Close the created file
WriteTitle(<cTopic>,<cTitle>)	Write the file title
WritePar(<cPar>)	Write a paragraph
WriteParBold(<cPar>)	Same as WritePar(), but the text is bold
WriteLink(<cLink>)	Write a link to another topic
ScanLink(<clink>)	Scan the aLinkRef array for a valid topic
DosToOs2Text(<cText>)	Convert a Dos string to a OS/2 String

## Examples

```
FUNCTION MAIN()  
  
LOCAL oNg  
  
oNg := T0s2():New( "ngi\harbour.ngi" )  
oNg:WriteTitle( "Harbour Reference Guide" )  
oNg:WritePar( "HARBOUR" )  
oNg:WriteLink( "OverView" )  
oNg:WriteLink( "License" )  
  
oNg:WritePar( "See the Links Above" )  
oNg:Close()  
RETURN Nil
```

## Status

Ready

## Compliance

This is a new Harbour Tools class

## Platforms

ALL

## See Also:

[TNortonGuide\(\)](#)

## TNortonGuide()

Norton Guide Class

### Syntax

```
oNg:=TNortonGuide():New(<cFile>) --> oNg
```

### Arguments

<cFile> Name of the Ng Source file to create

### Returns

<oNg> An instance of the TNortonGuide Class

### Description

TNortonGuide() is a class that creates the Norton Guide Source Code of the same name you pass to the constructor.

The class methods are as follows:

New(<cFile>) Create an instance of the TNortonGuide class

Close() Close the created file

WriteTitle(<cTopic>,<cTitle>) Write the file title

WritePar(<cPar>) Write a paragraph

WriteParBold(<cPar>) Same as WritePar(), but the text is bold

WriteLink(<cLink>) Write a link to another topic

### Examples

```
FUNCTION MAIN()  
  
LOCAL oNg  
  
oNg := TNortonGuide():New( "ngi\harbour.ngi" )  
oNg:WriteTitle( "Harbour Reference Guide" )  
oNg:WritePar( "HARBOUR" )  
oNg:WriteLink( "OverView" )  
oNg:WriteLink( "License" )  
  
oNg:WritePar( "See the Links Above" )  
oNg:Close()  
RETURN Nil
```

### Status

Ready

### Compliance

This is a new Harbour Tools class

### Platforms

ALL

### See Also:

[TTroff\(\)](#)

[TRtf\(\)](#)

[THtml\(\)](#)

[TOs2\(\)](#)

## TRtf()

Rtf Class

### Syntax

```
oNg:=TRtf():New(<cFile>) --> oRtf
```

### Arguments

<cFile> Name of the RTF file to create

### Returns

<oRtf> An instance of the TRtf Class

### Description

TRtf() is a class that creates the RTF Documentation Source Code of the same name you pass to the constructor.

The class methods are as follows:

New(<cFile>)	Create a new instance of the TRtf class
Close()	Close the create file
WriteTitle(<cTopic>,<cTitle>)	Write the file title
WritePar(<cPar>)	Write a paragraph
WriteParBold(<cPar>)	Same as WritePar(), but the text is bold
WriteLink(<cLink>)	Write a link to another topic
WriteHeader()	Write the RTF header
EndPar()	Write the end paragraph delimiter

### Examples

```
FUNCTION MAIN()  
  
LOCAL oRtf  
  
oRtf := TRtf():New( "rtf\harbour.rtf" )  
oRtf:WriteHeader()  
oRtf:WriteTitle( "Harbour Reference Guide" )  
oRtf:WritePar( "HARBOUR" ):Endpar()  
oRtf:WriteLink( "OverView" )  
oRtf:WriteLink( "License" )  
  
oRtf:WritePar( "See the Links Above" ):EndPar()  
oRtf:Close()  
RETURN Nil
```

### Status

Ready

### Compliance

This is a new Harbour Tools class

### Platforms

ALL

### See Also:

[TNortonGuide\(\)](#)

## **TTroff()**

Troff Class

### **Syntax**

```
oTroff:=TTroff():New(<cFile>) --> oTrf
```

### **Arguments**

<cFile> Name of the Troff file to create

### **Returns**

<oTrf> instance of the TTroff Class

### **Description**

TTroff() is a class that creates the TROFF Documentation Source Code of the same name you pass to the constructor.

The class methods are as follows:

New(<cFile>) Create a new instance of the THtml class Close()

Close the created file

WriteTitle(<cTopic>,<cTitle>) Write the file title

WritePar(<cPar>) Write a paragraph

WriteParBold(<cPar>) Same as WritePar(), but the text is bold

WriteLink(<cLink>) Write a link to another topic

WriteText() Writes text without formatting

### **Examples**

```
FUNCTION MAIN()  
  
LOCAL oTroff  
oTroff := TTroff():New( "tr\harbour.ngi" )  
oTroff:WriteTitle( "Harbour Reference Guide" )  
oTroff:WritePar( "HARBOUR" )  
oTroff:WriteLink( "OverView" )  
oTroff:WriteLink( "License" )  
  
oTroff:WritePar( "See the Links Above" )  
oTroff:Close()  
  
RETURN Nil
```

### **Status**

Ready

### **Compliance**

This is a new Harbour Tools class

### **Platforms**

ALL

### **See Also:**

[TNortonGuide\(\)](#)

## **CD()**

Change the Current Directory

### **Syntax**

```
CD(<cDir>) --> lSuccess
```

### **Arguments**

```
<cDir> DIR TO BE CHANGED
```

### **Returns**

```
<lSuccess> .T. IF SUCESFUL; otherwise .F.
```

### **Description**

```
CHANGE THE CURRENT DIRECTORY
```

### **Examples**

```
IF CD("OLA")
  RETURN(.T.)
ELSE
  RETURN(.F.)
ENDIF
```

### **Files**

```
Header is Fileio.ch
```

### **See Also:**

[MD\(\)](#)

[RD\(\)](#)

## **MD()**

Creates a Directory

### **Syntax**

```
MD(<cDir>) -> <lSucess>
```

### **Arguments**

```
<cDir> DIRECTORY TO BE CREATED
```

### **Returns**

```
<lSucess> .T. IF SUCESSFUL; otherwise .F.
```

### **Description**

```
CREATE A DIRECTORY
```

### **Examples**

```
IF MD("OLA")
  RETURN(.T.)
ELSE
  RETURN(.F.)
ENDIF
```

### **Files**

```
Header is Fileio.ch
```

### **See Also:**

[CD\(\)](#)

[MD\(\)](#)

**RD()**  
Remove a Directory

## Syntax

```
RD(<cDir>) --> <lSuccess>
```

## Arguments

```
<cDir> DIR TO BE DELETED
```

## Returns

```
<lSuccess> .T. IF SUCESFUL; otherwise .F.
```

## Description

```
REMOVE A DIRECTORY
```

## Examples

```
IF RD("OLA")  
  RETURN(.T.)  
ELSE  
  RETURN(.F.)  
ENDIF
```

## Files

```
Header is Fileio.ch
```

## See Also:

[CD\(\)](#)

[MD\(\)](#)

## StrFormat()

Format a string

### Syntax

```
StrFormat(<cMask>[, <cPar1>[, <cParn>[, ...]]) --> cString
```

### Arguments

**<cMask>** Holds the mask for the resulting string  
**<cParn>** Holds the strings to be inserted in the mask maximum 9 of them can be specified.

### Returns

**<cString>** Return the mask with all the parameters inserted.

### Description

String replacment, can be useful when writing international apps. You can separate the constant strings from the variable ones. Each %1 - %9 marks will be replaced with the appropriate parameter from the parameter list. Marks can be in any order, and can be duplicated. You can print "%" character with "%%".

### Examples

```
StrFormat("Please insert disk %1 to drive %2", LTrim(Str(2)), "A:")  
StrFormat("This is %1 from %2", "Victor", "Hungary")  
StrFormat("%2 %1 %2", "Param1", "Param2")
```

### Tests

```
? StrFormat("Please insert disk %1 to drive %2", LTrim(Str(2)), "A:")  
? StrFormat("This is %1 from %2", "Victor", "Hungary")  
? StrFormat("%2 %1 %2", "Param1", "Param2")  
? StrFormat("Hello")  
? StrFormat("%1 - %2", "one")  
? StrFormat("%1 - %2", "one", "two")  
? StrFormat("%2 - %1", "one", "two")  
? StrFormat("%2 - %", "one", "two")  
? StrFormat("%% - %", "one", "two")  
? StrFormat("%9 - %", "one", "two")
```

### Status

Done

### Compliance

All platforms

### Files

Library is libmisc

## **AMONTHS ( )**

Returns an array with the months names.

### **Syntax**

```
AMONTHS() --> aMonths
```

### **Arguments**

### **Returns**

`<aMonths>` The array which holds the months names.

### **Description**

This function returns an array with all the months names in the selected current language.

### **Examples**

```
aMonths := AMonths()  
? aMonths[1] -> January  
? aMonths[1] -> Enero (if the selected language is Spanish)
```

### **Status**

Ready

### **Compliance**

This function is new in Harbour.

### **Platforms**

All

### **Files**

Library is libmisc

### **See Also:**

[ADAYS\(\)](#)

## **ADAYS()**

Returns an array with the days names.

### **Syntax**

```
ADAYS() --> aDays
```

### **Arguments**

### **Returns**

<aDays> The array which holds the days names.

### **Description**

This function returns an array with all the days names in the selected current language.

### **Examples**

```
aDays := ADays()  
? aDays[1] -> Sunday  
? aDays[1] -> Domingo (if the selected language is Spanish)
```

### **Status**

Ready

### **Compliance**

This function is new in Harbour.

### **Platforms**

All

### **Files**

Library is libmisc

### **See Also:**

[ADAYS\(\)](#)

## **ISLEAPYEAR( )**

Checks if the given date is a leap year.

### **Syntax**

```
ISLEAPYEAR( <dDate> ) --> lTrueOrFalse
```

### **Arguments**

<dDate>     A valid date.

### **Returns**

<lTrueOrFalse>     A logical that indicates if the date year is leap

### **Description**

This function returns true if the given date is a leap year and false if isn't.

### **Examples**

```
? IsLeapYear( DToC( "01/01/2000" ) ) -> .t.  
? IsLeapYear( DToC( "01/01/2001" ) ) -> .f.
```

### **Status**

Ready

### **Compliance**

This function is new in Harbour.

### **Platforms**

All

### **Files**

Library is libmisc

### **See Also:**

[DAYSINMONTH\( \)](#)

## **DAYSINMONTH( )**

Gets the days in a month.

### **Syntax**

```
DAYSINMONTH( <dDate> ) --> nDays
```

### **Arguments**

<dDate> A valid date.

### **Returns**

<nDays> The number of days of the month.

### **Description**

This function returns the number of days of the given date month.

### **Examples**

```
? DaysInMonth( DToC( "01/01/2000" ) ) -> 31  
? DaysInMonth( DToC( "02/01/2000" ) ) -> 29
```

### **Status**

Ready

### **Compliance**

This function is new in Harbour.

### **Platforms**

All

### **Files**

Library is libmisc

### **See Also:**

[ISLEAPYEAR\(\)](#)

## **EOM( )**

Gets the last day in a month.

### **Syntax**

```
EOM( <dDate> ) --> dEOM
```

### **Arguments**

<dDate>     A valid date.

### **Returns**

<dEOM>     The last day in the month.

### **Description**

This function returns the last day of a given month date.

### **Examples**

```
? EOM( DToC( "01/01/2000" ) ) -> "01/31/2000"  
? EOM( DToC( "02/01/2000" ) ) -> "01/29/2000"
```

### **Status**

Ready

### **Compliance**

This function is new in Harbour.

### **Platforms**

All

### **Files**

Library is libmisc

### **See Also:**

[BOM\(\)](#)

[TFileRead\(\)](#)

## **BOM()**

Gets the first day in a month.

### **Syntax**

```
BOM( <dDate> ) --> dBOM
```

### **Arguments**

<dDate>     A valid date.

### **Returns**

<dBOM>     The first day in the month.

### **Description**

This function returns the first day of a given month date.

### **Examples**

```
? BOM( DToC( "01/25/2000" ) ) -> "01/01/2000"  
? BOM( DToC( "02/24/2000" ) ) -> "02/01/2000"
```

### **Status**

Ready

### **Compliance**

This function is new in Harbour.

### **Platforms**

All

### **Files**

Library is libmisc

### **See Also:**

[EOM\(\)](#)

[TFileRead\(\)](#)

## DOY()

Gets the day number of the year.

### Syntax

```
DOY( <dDate> ) --> nDay
```

### Arguments

<dDate> A valid date.

### Returns

<nDay> The day number

### Description

This function returns the day number of the year for a given date.

### Examples

```
? DOY( DToC( "01/31/2000" ) ) -> 31
? DOY( DToC( "02/20/2000" ) ) -> 51
```

### Status

Ready

### Compliance

This function is new in Harbour.

### Platforms

All

### Files

Library is libmisc

### See Also:

[WOY\(\)](#)

## WOY()

Gets the week number of the year.

### Syntax

```
WOY( <dDate>, <lIso> ) --> nWeek
```

### Arguments

<dDate> A valid date.

### Returns

<nWeek> The week number <lIso> Flag that indicates if <nWeek> is in ISO format.

### Description

This function returns the week number of the year for a given date. It returns the week number in ISO format ( range 0 - 52, by default or passing TRUE as second parameter) or 1 - 52 if lIso is FALSE.

### Examples

```
? Woy( DToC( "01/31/2000" ) ) -> 3  
? Woy( DToC( "01/31/2000" ), FALSE ) -> 4
```

### Status

Ready

### Compliance

This function is new in Harbour.

### Platforms

All

### Files

Library is libmisc

### See Also:

[DOY\(\)](#)

## **EOY()**

Gets the last date of the year.

### **Syntax**

```
EOY( <dDate> ) --> dEOY
```

### **Arguments**

<dDate>     A valid date.

### **Returns**

<dEOY>     The last date of the year.

### **Description**

This function returns the last date of a given year date.

### **Examples**

```
? EOY( DToC( "01/01/2000" ) ) -> "31/12/2000"  
? EOY( DToC( "01/01/2001" ) ) -> "31/12/2001"
```

### **Status**

Ready

### **Compliance**

This function is new in Harbour.

### **Platforms**

All

### **Files**

Library is libmisc

### **See Also:**

[BOY\(\)](#)

## **BOY()**

Gets the first date of the year.

### **Syntax**

```
BOY( <dDate> ) --> dBOY
```

### **Arguments**

<dDate>     A valid date.

### **Returns**

<dBOY>     The first day in the year.

### **Description**

This function returns the first date of a given year date.

### **Examples**

```
? BOY( DToC( "01/25/2000" ) ) -> "01/01/2000"  
? BOY( DToC( "02/24/2001" ) ) -> "01/01/2001"
```

### **Status**

Ready

### **Compliance**

This function is new in Harbour.

### **Platforms**

All

### **Files**

Library is libmisc

### **See Also:**

[EOY\(\)](#)