METALIB V5.0

mailto:info@trading-tools.com

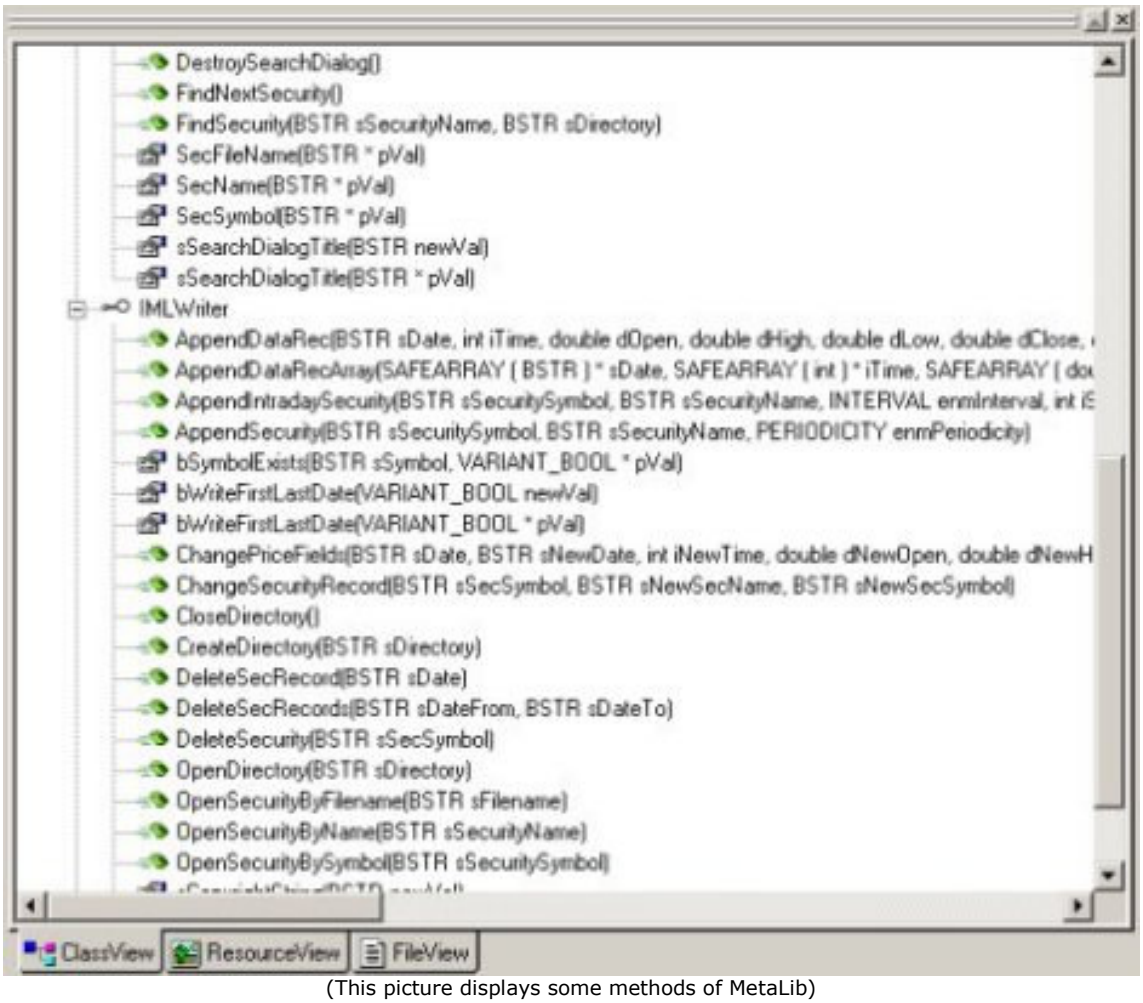
User manual

Table of Contents

Getting Started

Welcome

The MetaLib SDK provides functions for reading/writing/sorting MetaStock™ security and price data. The MetaLib library removes many of the complexities of accessing MetaStock™ data files from your program. As a result, your development time will be shorter.



(This picture displays some methods of MetaLib)

How to add MetaLib to your Visual C++ Project

1. Open your VC++ project
2. Add the MLErrorConstants.h and MLUtil.h file to your project.
3. Add the following lines to the Stdafx.h file:

```
// TODO: Enter the correct path to the MetaLib.tlb file.
#import "MetaLib.tlb" no_namespace raw_interfaces_only

#include "MLErrorConstants.h"      // MetaLib error constants
#include "MLUtil.h"
#include <afxdisp.h>

extern      IMLWriter               *g_IMLWriter;
extern      IMLReader               *g_IMLReader;
extern      IMLProgressBar          *g_IMLProgBar;
extern      IMLConverter            *g_IMLConverter;
extern      IMLRegistration         *g_IMLRegistration;
extern const IID IID_IMLRegistration;
extern const IID IID_IMLWriter;
extern const IID IID_IMLReader;
extern const IID IID_IMLConverter;
extern const IID IID_IMLProgressBar;
extern const IID LIBID_METALIBLib;
extern const CLSID CLSID_MLRegistration;
extern const CLSID CLSID_MLWriter;
extern const CLSID CLSID_MLReader;
extern const CLSID CLSID_MLConverter;
extern const CLSID CLSID_MLProgressBar;
```

4. Add the following lines to your Stdafx.cpp file:

```
IMLWriter               *g_IMLWriter = NULL;
IMLReader               *g_IMLReader = NULL;
IMLProgressBar          *g_IMLProgBar = NULL;
IMLConverter            *g_IMLConverter = NULL;
IMLSecurityFinder       *g_IMLSecFinder = NULL;
IMLRegistration         *g_IMLRegistration = NULL;

const IID IID_IMLWriter =
{0xEDC0C57F,0x8E8B,0x4417,{0xA0,0xCB,0x0D,0x71,0x25,0x28,0x02,0xE2}};

const IID IID_IMLReader =
{0x326DE151,0xC4C0,0x4692,{0x9E,0xB1,0x00,0xA7,0xDC,0x64,0x7C,0xF4}};

const IID IID_IMLProgressBar =
{0x37277B21,0x8271,0x4704,{0xB5,0x8D,0x7F,0x4F,0xD6,0x0F,0xD9,0x13}};

const IID IID_IMLSecurityFinder =
{0xE879A539,0x246F,0x4BC3,{0x84,0xC5,0x6D,0x2F,0x5D,0x8E,0x6B,0x97}};

const IID IID_IMLConverter =
{0x955D5686,0xF038,0x4BE8,{0xAF,0x7E,0xE3,0x59,0x69,0xD4,0xE6,0xC5}};

const IID IID_IMLRegistration =
{0xAF42E4DF,0xACA3,0x44FF,{0xAC,0xA4,0x2E,0x96,0xE6,0xC1,0x2F,0x74}};
```

```
const IID LIBID_METALIBLib =
{0x90338DB9,0x6AF0,0x4982,{0xA6,0x33,0xDC,0x18,0x8B,0x90,0x03,0xEF}};

const CLSID CLSID_MLWriter =
{0x20315AFA,0x7C65,0x41BF,{0x98,0xF0,0xA1,0x6D,0xAB,0x30,0x7D,0x7E}};

const CLSID CLSID_MLReader =
{0x14AE4119,0xAB4D,0x449D,{0xB4,0x1A,0x62,0x78,0xED,0xEA,0xFC,0x84}};

const CLSID CLSID_MLProgressBar =
{0x8BF4931C,0xD699,0x42DA,{0xB2,0x47,0x31,0x67,0x35,0x6D,0xAE,0x6A}};

const CLSID CLSID_MLSecurityFinder =
{0xBBB8A68B,0x1EF0,0x4C6E,{0x81,0x46,0x51,0x64,0x21,0x7E,0xA6,0xC4}};

const CLSID CLSID_MLConverter =
{0x5DC0B71F,0xAF75,0x4BBB,{0x9A,0x3F,0x64,0xB8,0xF4,0x3D,0xF5,0xA5}};

const CLSID CLSID_MLRegistration =
{0x9CCFD6AB,0x4913,0x48CF,{0xBA,0x8F,0xA8,0x2A,0xFD,0x87,0x01,0x22}};
```

5. To initialize the Metalib interfaces add the following code e.g. to your InitInstance function. This function is located in the class CYourApplicationNameApp.

```
::CoInitialize (NULL);

if(FAILED(::CoCreateInstance
(CLSID_MLReader,NULL,CLSCTX_INPROC_SERVER,IID_IMLReader,(void**)&g_IM
LReader)))
{
            TRACE("Failed to create the IID_IMLReader object!\n");
}

if(FAILED(::CoCreateInstance
(CLSID_MLWriter,NULL,CLSCTX_INPROC_SERVER,IID_IMLWriter,(void**)&g_IM
LWriter)))
{
            TRACE("Failed to create the IID_IMLWriter object!\n");
}

if(FAILED(::CoCreateInstance
(CLSID_MLProgressBar,NULL,CLSCTX_INPROC_SERVER,IID_IMLProgressBar,(vo
id**)&g_IMLProgBar)))
{
            TRACE("Failed to create the IID_MLProgressBar
object!\n");
}

if(FAILED(::CoCreateInstance
(CLSID_MLConverter,NULL,CLSCTX_INPROC_SERVER,IID_IMLConverter,(void**
)&g_IMLConverter)))
{
            TRACE("Failed to create the IID_MLConverter object!\n");
}

if(FAILED(::CoCreateInstance
(CLSID_MLSecurityFinder,NULL,CLSCTX_INPROC_SERVER,IID_IMLSecurityFind
er,(void**)&g_IMLSecFinder)))
{
            TRACE("Failed to create the IID_IMLSecurityFinder
object!\n");
}
```

```
if(FAILED(::CoCreateInstance
(CLSID_MLRegistration,NULL,CLSCTX_INPROC_SERVER,IID_IMLRegistration,(
void**)&g_IMLRegistration)))
{
      TRACE("Failed to create the IID_IMLRegistration object!\n");
}
// g_IMLRegistration->SetRegistrationInfo (L"YOUR NAME",L"YOUR KEY");
```

6. It is necessary to release all interface pointers you have used. To release all interface pointers add the following code e.g. to the destructor of the CyourApplicationNameApp class.

```
// Release all interface pointers
if(g_IMLWriter != NULL)
{
      g_IMLWriter->Release ();
      g_IMLWriter = NULL;
}

if(g_IMLReader != NULL)
{
      g_IMLReader->Release ();
      g_IMLReader = NULL;
}

if(g_IMLProgBar != NULL)
{
      g_IMLProgBar->Release ();
      g_IMLProgBar = NULL;
}

if(g_IMLConverter != NULL)
{
      g_IMLConverter->Release ();
      g_IMLConverter = NULL;
}

if(g_IMLRegistration != NULL)
{
      g_IMLRegistration->Release();
      g_IMLRegistration = NULL;
}
```

How to add MetaLib to your JAVA Project

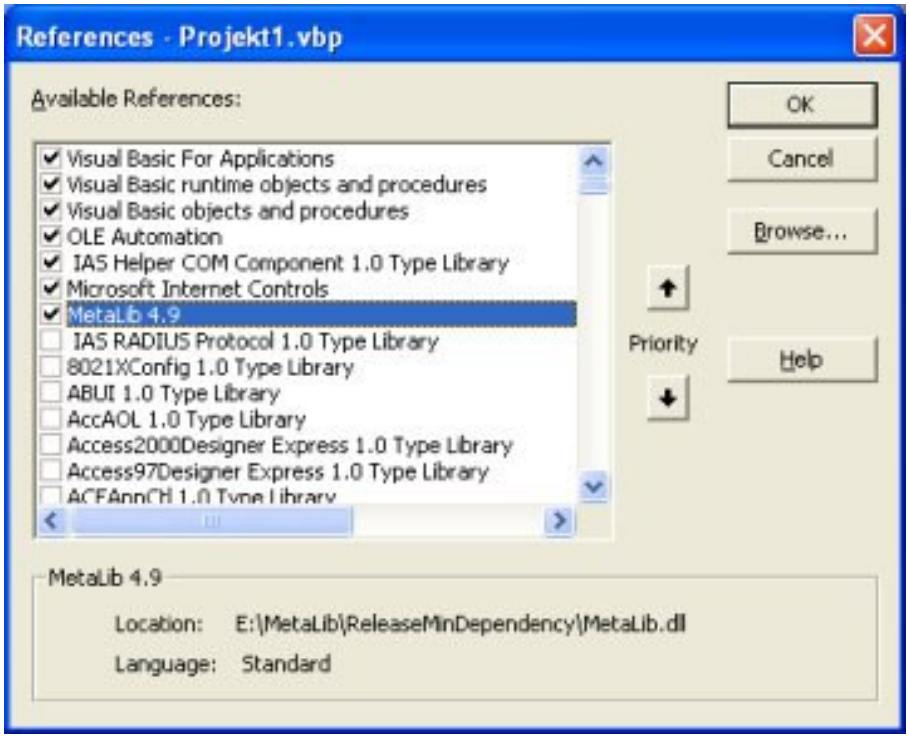
You need a COM bridge, like the JACOB project (http://danadler.com/jacob) if you want to use MetaLib with JAVA. Download the COM bridge from this site and follow the instructions.

How to add MetaLib to your Visual Basic Project

Steps to add MetaLib to your VB/VBA project:

1. Start the Visual Basic-Editor (ALT+F11)

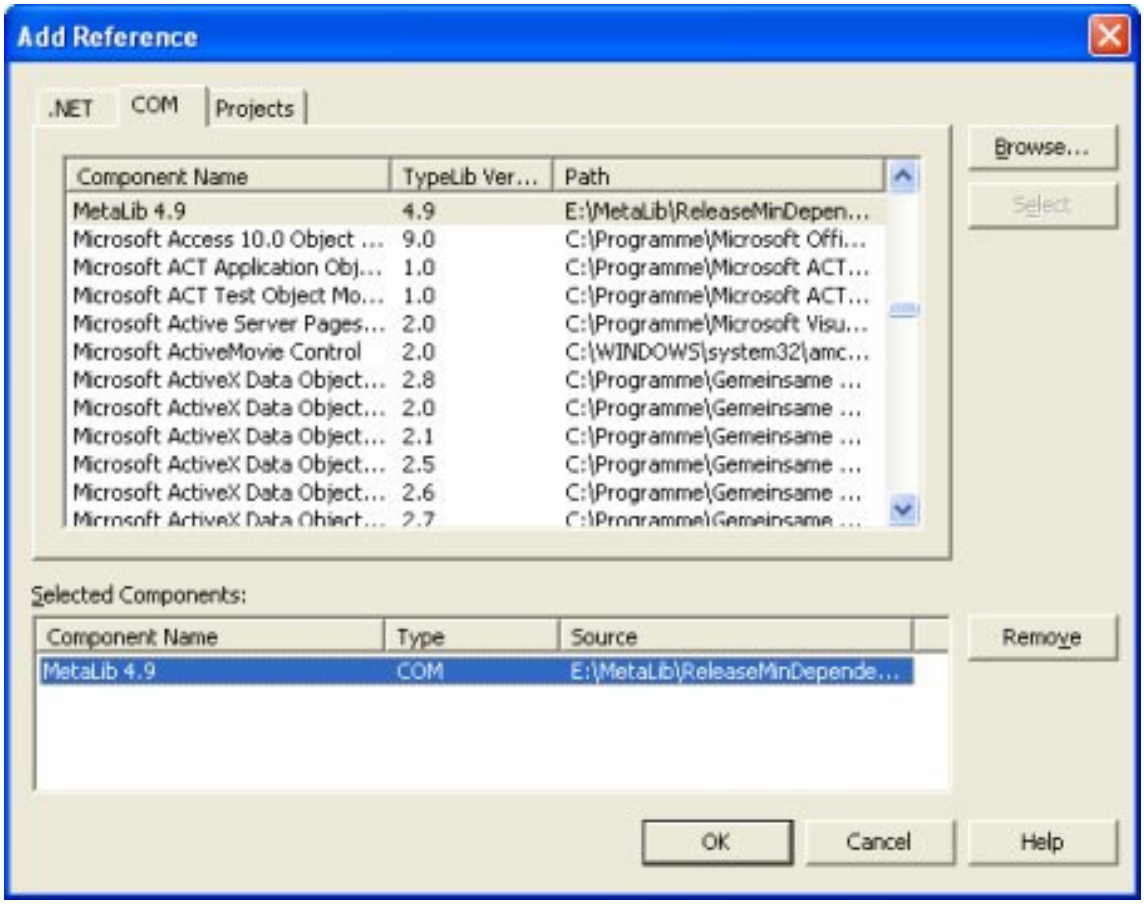
2. Open the References dialog



3. Select the MetaLib 5.0 entry.
4. Close the dialog.
5. [Optional] Add the **MetaLibUtil.Bas** (C:\…\MetaLib\Include) file to your project.

How to add MetaLib to your VB.NET project

1. Click on "Project->Add Reference"
2. The "Add Reference" dialog will appear:



3. Double click on the MetaLib 5.0 entry.
4. The MetaLib entry will appear in the "Selected Components" list box.
5. Click on "OK".

VB.NET example code:

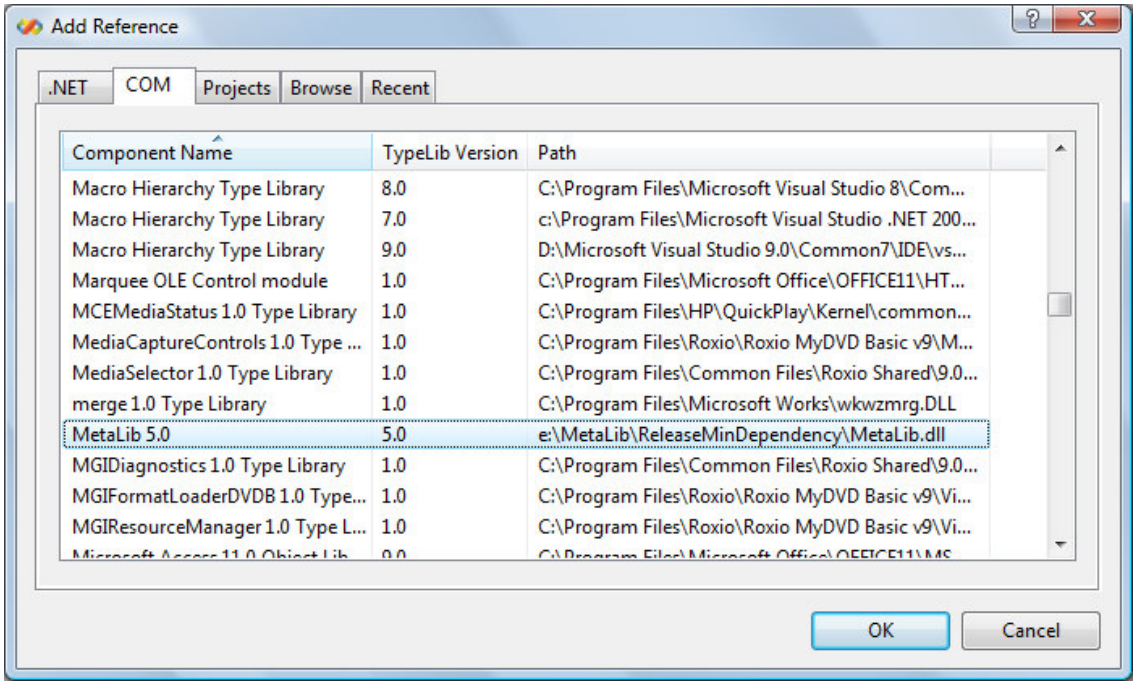
```
Dim Reader As New METALIBLib.MLReader

Reader.OpenDirectory("C:\MetaStock")
Reader.ReadMaster()
' Display all securities that are stored in the MS directory
While (Reader.iMaRecordsLeft > 0)
  Debug.WriteLine(Reader.sMaSecName + "(" + Reader.sMaSecSymbol + ")")
  Reader.ReadMaster()
End While

Reader.CloseDirectory()
```

How to add MetaLib to your C++, C# VS.NET 2003 or VS.NET 2005 Project

1. Click on "Project->Add Reference"
2. The "Add Reference" dialog will appear:



3. Double click on the MetaLib 5.0 entry.
4. The MetaLib entry will appear in the "Selected Components" list box.
5. Click on "OK".

For easier use it is recommended that you write the statement "`using METALIBLib;`" at the top of each source code file where you want to use the DLL.

Use the "`new`" statement to create a new instance of an interface:

```
MLReaderClass      lReader = new MLReaderClass ();
MLWriter           lWriter = new MLWriterClass ();
```

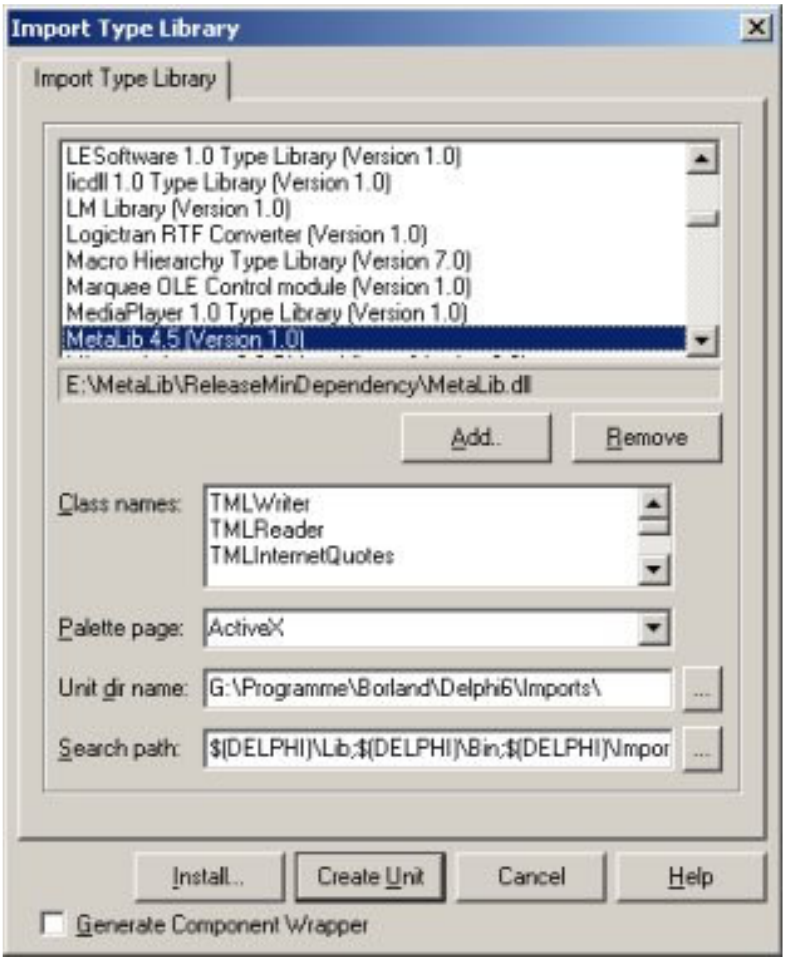
If you do not use the "`using METALIBLib;`" statement you have to declare the interfaces as shown below:

```
METALIBLib.MLReaderClass      lReader = new METALIBLib.MLReaderClass ();
METALIBLib.MLWriter           lWriter = new METALIBLib.MLWriterClass ();
```

Take a look at the C# example that are located in the "C:\Program Files\Metalib\Examples\" folder for more details.

How to add MetaLib to your Delphi Project

1. Click on the Project->Import Type Library menu.
2. The "Import Type Library" dialog will appear.



3. Select the MetaLib 5.0 entry.
4. Make sure that the "Generate Component Wrapper" check box is unchecked.
5. Click on the "Create Unit" button.

Listed below is the main Unit1 which shows proper connection has been established with the MetaLib.dll by displaying a simple listing of both Stock symbols and their corresponding company name in a standard Listbox component.

```
Unit Unit1

 interface

 uses
     Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms,
Dialogs,
     ComObj, ActiveX, OleCtrls, StdVCL, Grids, MetaLIBLib.TLB, ExtCtrls;

 type
     TPtest1 = class(TForm)
     Panel1: TPanel;
     Listbox1: TListBox;

     procedure FormCreate(Sender: TObject);
     procedure MReader(Sender: TObject);
     procedure ShowMSG(Sender: TObject);

 private
```

```
    {Private declarations}
 public
    {Public declarations}

    end;

 var
    Reg : IMLRegistration;
    Reader : MLReader;
    Ptest1: TPtest1;
    procedure MReader; external 'MetaLib.dll';

 implementation
    {$R *.DFM}

 var
    {Global variables}

    DRT: extended;

    procedure TPtest1.FormCreate(Sender: TObject);
    var
       Bstring: String;

       begin
       {ShowMSG;}
       {MReader;}
       end;

       procedure TPtest1.MReader(Sender: TObject);
       var
       X : integer;
       Path : WideString;
       Secsym : WideString;
       Fname : WideString;
       RecsLeft : integer;

    begin
    X := 0;
    Path := ('C:\Metastock Data\nasdaq');
    Reader.OpenDirectory('C:\Metastock Data\nasdaq');    {put in the
location of your system}
    Reader.Get_iMaRecordsLeft(RecsLeft);

    For X := 1 to RecsLeft Do
    begin
    Reader.ReadMaster;
    Reader.Get_sMaSecSymbol(Secsym);
    Reader.Get_sMaSecName(Fname);
    Listbox1.Items.Add ((Secsym)+(' - ')+(Fname));

    end;

    Reader.CloseDirectory;
    end;


    procedure TPtest1.ShowMSG(Sender: TObject);
    var
    Bstring : string

    begin
```

```
    {just another routine you want}
    end;


initialization

    CoInitialize(nil);
    Reg:= CreateComObject(Class_MLRegistration) as IMLRegistration;
    Reg.SetRegistrationInfo('Your Name','Your key number');
    Reader:= CreateComObject(Class_MLReader) as IMLReader;
    Reg.DisplayAboutDlg;

finalization

    CoUninitialize;
    Reg := nil;
    Reader := nil;

end.

  {Note: Set Listbox Events 'OnClick' and 'OnEnter' to MReader}
```

Technical Specifications

- MetaLib supports the latest Y2K compatible MetaStock™ version 8.0 data format.
- Data for up to 6000 securities can be stored in each directory. However, the fewer securities you have in a folder, the better the performance of programs that access the data (e.g., MetaStock™) will be. Some programs still won't recognize more than 255 securities in a directory, the limit for previous versions of the MetaStock™ data format.
- A maximum of 64000 records can be saved for each security.
- Distributed Form: 32-bit Windows DLL
- Supported Date Range: January 1, 1910 to December 31, 2199

System Requirements

Development Environments: Visual Basic 5.0 or higher, Visual C++ 4.1 or higher, VBA (Excel, etc.), Delphi 3.0 or higher, JAVA, C# or any other environment supporting Windows DLLs.

32 bit version: Windows 95/98/ME/2000/XP

32 MB RAM (64 MB recommended).
Pentium 75 or higher recommended.
About 2 megabyte of hard drives space.

MLReader Interface

The *MLReader* interface provides functions for reading the MetaStock files.

Master File Properties

The properties contain valid data after you have called the *ReadMaster* or *ReadDay* function. All master file properties are declared as BSTR. The Security file properties are declared as floating point numbers. Floating-point numbers use the IEEE (Institute of Electrical and Electronics Engineers) format. Single-precision values with float type have 4 bytes, consisting of a sign bit, an 8-bit excess-127 binary exponent, and a 23-bit mantissa.

bSymbolExists

Returns "True" if the symbol exists in the currently opened directory.

sMaEndTime

Returns the end time of the current master record.
The time has the following format: hhmmss

sMaStartTime

Returns the start time of the current master record.
The time has the following format: hhmmss

sMaSecName

Returns the name of the security. (EX: Coca Cola)

sMaSecSymbol

Returns the ticker symbol for the security (EX: MSFT)

sMaSecFileName

Returns the filename of the security. (EX: E:\Trading\Vienna\f1.dat)

iMaFirstDate

Returns the first date that is stored in a Fxxx.xxx file. (EX: 19991214)
The date is stored in the following format: yyyymmdd.

iMaLastDate

Returns the last date that is stored in a Fxxx.xxx file. (EX: 19991214)
The date is stored in the following format: yyyymmdd.

iMaNrFields

Returns the field size of the security. The value can be between 4 (Tick) and 8.

MaInterval

Returns the interval of the security.
The MaInterval can have the following values:
None

Tick
1 Minute
5 Minute
6 Minute
10 Minute
15 Minute
20 Minute
30 Minute
60 Minute

MaPeriodicity

Returns the periodicity of the security.
The MaPeriodicity can have the following values:
Intraday
Daily
Weekly
Monthly

Master File Functions

The master file functions allow you to access a master file of directory specified in the *OpenDirectory* function. You should use the functions in the following sequence:

1. OpenDirectory
2. ReadMaster
3. Access Master file properties (sMaSecName, sMaSecSymbol, etc)
4. CloseDirectory

CloseDirectory

The *CloseDirectory* function closes the directory that was previously opened with the *OpenDirectory* function. Call this function before you want to open another master file.

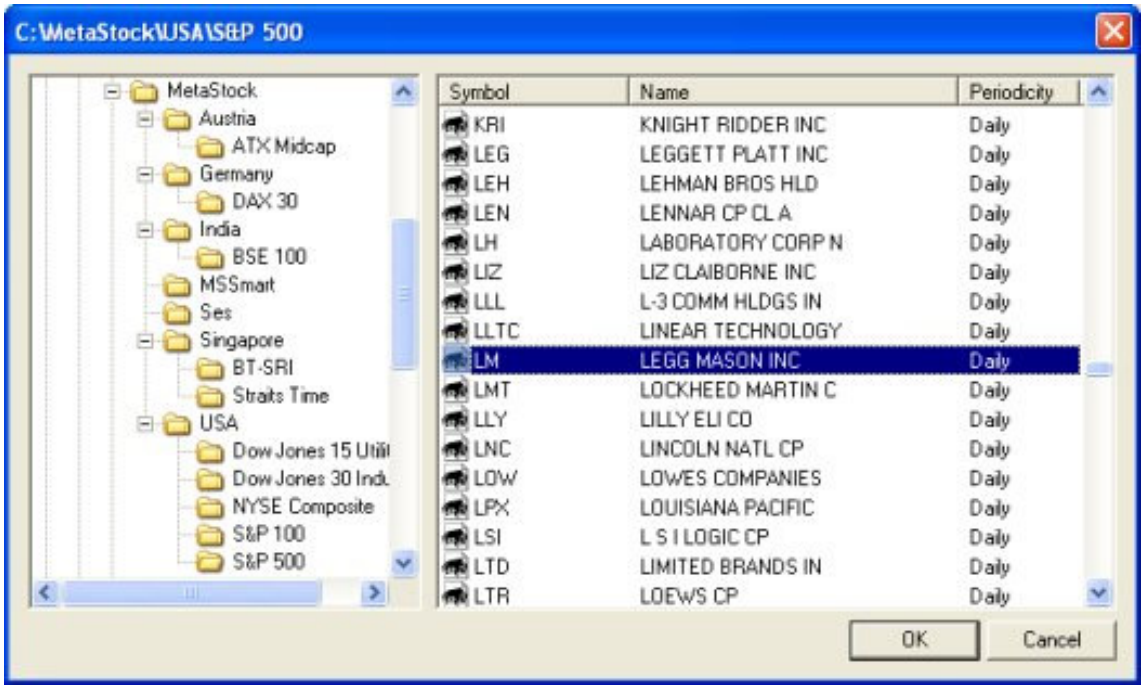
HRESULT CloseDirectory (void);

Parameters

This function has no parameters.

DisplayMSDirBrowser

The DisplayMSDirBrowser function displays a dialog that allows you to browse for MetaStock directories.



HRESULT DisplayMSDirBrowser(
[out] BSTR *sSelectedSymbolFileName,
[optional] BSTR sDirectoryToDisplay,
[optional] BSTR sReserved
);

Parameters

sSelectedSymbolFileName
Returns the file name of the selected security. (e.g. "C:\MetaStock\F1.") You can pass this value to the "OpenSecurityByFilename" function. If no security was selected the *sSelectedSymbolFileName* variable is empty.

sDirectoryToDisplay
You can define a directory that will be displayed when the dialog opens.

sReserved
Reserved for future use. Should be NULL.

Example

```
Dim r As New MLReader
Dim sSeldir As String
r.DisplayMSDirBrowser sSeldir, "C:\MetaStock\"
MsgBox "Selected security: " + sSeldir
r.OpenSecurityByFilename sSeldir
MsgBox "Number of data records: " + r.iSeRecords
```

OpenDirectory

The *OpenDirectory* function opens the Master file of the specified MetaStock directory.

HRESULT OpenDirectory (
BSTR sDirectory
);

Parameters

sDirectory

The name of the directory containing the MetaStock files.

Return Values

The *OpenDirectory* function can raise the following errors:

- File system errors
- ML_DIRECTORY_NOT_EXIST (0x0649, 1609)

For extended error descriptions go to the MetaLib error codes section.

ReadMaster

The *ReadMaster* function reads a security record from the current file position. You can access the security record if you use the master file properties. Before you can use the *ReadMaster* function, you must call the *OpenDirectory* function.

HRESULT ReadMaster (void);

Parameters

The function has no parameters.

Return Values

The *ReadMaster* function can raise the following errors:

- File system errors

For extended error descriptions go to the MetaLib error codes section.

Example

This example reads all records of a master file and displays the security names:

```
Dim Reader As New MLReader
Reader.OpenDirectory "C:\MetaStock"
Do While (Reader.iMaRecordsLeft > 0)
      Reader.ReadMaster
      MsgBox(Reader.sMaSecName)
Loop
Reader.CloseDirectory
```

Security File Properties

iSeADate

Returns the date array of a security price record file.
The date is stored in the following format: YYYYMMDD.

dSeAOpen

Returns the open quotation array of a security price record file.

dSeAHigh

Returns the high quotation array of security price record file.

dSeALow

Returns the low quotation array of security price record file.

dSeAClose

Returns the close quotation array of a security price record file.

dSeAVolume

Returns the volume array of a security price record file.

dSeOpen

Returns the open quotation of the current security record.

dSeHigh

Returns the high quotation of the current security record.

dSeLow

Returns the low quotation of the current security record.

dSeClose

Returns the close quotation of the current security record.

dSeVolume

Returns the volume of the current security record.

iSeDate

Returns the date of the current security record. (EX: 20011214)
The date is stored in the following format: YYYYMMDD.

iSeRecordsLeft

Returns the number of records that are stored in the specified security price record file deducting the number of price records that were loaded from the security file. You can use this property for an exit condition in a loop.

iSeRecords

Returns the number of records that are stored in the specified security price record file. (Fxxx.dat or Fxxx.mwd)

iSeTime

Returns the time of the current price record.

iRound

Returns all price records values rounded to a specified number of decimal places.

IsMetastockDirectory

Checks if a directory contains MetaStock™ files. Returns "TRUE" if the directory contains MetaStock™ files.

Example

```
Dim Reader As New MLReader
If Reader.IsMetastockDirectory("C:\MetaStock") Then
 ' Directory contains files -> Open the directory
 Reader.OpenDirectory "C:\MetaStock"
 ' Do something
```

```
End If
```

iStartIndex

Sets the lower bound for the dSeAOpen, dSeAClose etc. arrays. By default, this value is 0.

Example

```
Dim Reader As New MLReader
Dim Dat() As Long
Dim Time() As Long
Dim Op() As Single
Dim High() As Single
Dim Low() As Single
Dim Cl() As Single
Dim Volume() As Single
Dim Openint() As Single

Reader.iStartIndex = 1
Reader.OpenSecurityByFilename "C:\Metastock\F1.dat"
Reader.ReadDayArray

' Get the arrays
Dat = Reader.iSeADate ' Date
Time = Reader.iSeATime ' Time
Op = Reader.dSeAOpen ' Open
High = Reader.dSeAHigh ' High
Low = Reader.dSeALow ' Low
Cl = Reader.dSeAClose ' Close
Volume = Reader.dSeAVolume ' Volume
Openint = Reader.dSeAOpenInterest 'OpenInterest

' Now display all values in a message box
For i = 1 To UBound(Dat)
  MsgBox Dat(i) & ", " & Time(i) & ", " & Op(i) & ", " & High(i) & ", " & _
  Low(i) & ", " & Cl(i) & ", " & Volume(i) & ", " & Openint(i)
Next I
```

If you set the *iStartIndex* property to 0 you have to access the arrays the following way:

```
For i = 0 To UBound(Dat)
  MsgBox Dat(i) & ", " & Time(i) & ", " & Op(i) & ", " & High(i) & ", " & _
  Low(i) & ", " & Cl(i) & ", " & Volume(i) & ", " & Openint(i)
Next I
```

Security File Functions

The Security file functions allow you to access security price record files (Fxxx.dat or Fxxx.mwd). You should use the functions in the following sequence:

1. OpenDirectory
2. OpenSecurity function (OpenSecurityByName, etc.)
3. Seek functions (optional)
4. ReadDay
5. Access Security file properties (iSeDate, dSeOpen, etc.)
6. CloseDirectory and CloseSecurity

CloseSecurity

The *CloseSecurity* function closes the price record file (Fxxx.dat or Fxxx.mwd) that was previously opened with the *OpenSecurityBy** functions.

HRESULT CloseSecurity (void);

Parameters

This function has no parameters.

OpenSecurityByFilename

The *OpenSecurityByFileName* function opens the price record file (Fxxx.dat or Fxxx.mwd) specified in the sFileName parameter. It is **not** required to call the *OpenDirectory* function before you call the *OpenSecurityByFileName* function.

HRESULT OpenSecurityByFilename (
    BSTR sFileName
);

Parameters

sFilename
    The name of the file containing the security records.
    ("C:\MetaStock\Dow30\F1.dat")

Return Values

The *OpenSecurityByFilename* function can raise the following errors:

- File system errors

For extended error descriptions go to the MetaLib error codes section.

OpenSecurityByName

The *OpenSecurityByName* function opens the price record file (Fxxx.dat or Fxxx.mwd) containing the specified security name. Before you can use this function you must call the *OpenDirectory* function.

HRESULT OpenSecurityByName (
    BSTR sSecurityName
);

Parameters

sSecurityName
    The name of the security. (Note: Security name is not case sensitive.)

Return Values

The *OpenSecurityByName* function can raise the following errors:

- File system errors
- ML_SECURITY_NAME_NOT_FOUND (0x0643, 1603)
- ML_NO_DIRECTORY_OPENED (0x0648, 1608)

For extended error descriptions go to the MetaLib error codes section.

OpenSecurityBySymbol

The *OpenSecurityBySymbol* function opens the price record file (Fxxx.dat or Fxxx.mwd) containing the specified security symbol. Before you can use this function you must call the *OpenDirectory* function.

HRESULT OpenSecurityBySymbol (

BSTR sSecuritySymbol

);

Parameters

sSecuritySymbol

The symbol using to identify the security. (Note: Security symbol is not case sensitive.)

Return Values

The *OpenSecurityBySymbol* function can raise the following errors:

- File system errors
- ML_SECURITY_SYMBOL_NOT_FOUND (0x0642, 1602)
- ML_NO_DIRECTORY_OPENED (0x0648, 1608)

For extended error descriptions go to the MetaLib error codes section.

ReadDay

The *ReadDay* function reads a price record (date, open, high, etc.) from the current file position. You can access the price record if you use the security file properties. Before you can use the *ReadDay* function, you must call one of the *OpenSecurityBy** functions.

HRESULT ReadDay (void);

Parameters

This function has no parameters.

Return Values

The *ReadDay* function can raise the following errors:

- File system errors

For extended error descriptions go to the MetaLib error codes section.

Example

This example reads all records of a price record file (Fxxx.dat or Fxxx.mwd) and displays them.

```
Dim Reader As New MLReader
Reader.OpenDirectory "C:\MetaStock"
Reader.OpenSecurityBySymbol "MSFT"
Do While (Reader.iSeRecordsLeft > 0)
      Reader.ReadDay
      MsgBox (Reader.iSeDate & Chr(9) & Reader.dSeHigh & Chr(9) _
      & Reader.dSeLow & Chr(9) & Reader.dSeClose)
Loop
Reader.CloseSecurity
Reader.CloseDirectory
```

ReadDayArray

The *ReadDayArray* function reads a price record array (date, open, high, etc.) from a price record file (Fxxx.dat or Fxxx.mwd). All price records of the files will be stored in arrays. (dSeAOpen, dSeAHigh, dSeAClose, etc.) Before you can use the *ReadDay* function, you must call one of the *OpenSecurityBy** functions.

HRESULT ReadDayArray (void);

Parameters

This function has no parameters.

Return Values

The *ReadDayArray* function can raise the following errors:

- File system errors

For extended error descriptions go to the MetaLib error codes section.

Example

This example reads all records of a price record file and saves them to an array and displays all values in a message box.

```
Dim Reader As New MLReader
Dim Dat() As Long
Dim Time() As Long
Dim Op() As Single
Dim High() As Single
Dim Low() As Single
Dim Cl() As Single
Dim Volume() As Single
Dim Openint() As Single

Reader.OpenSecurityByFilename "C:\MetaStock\Dow\F1.dat"
Reader.ReadDayArray

' Get the arrays
Dat = Reader.iSeADate ' Date
Time = Reader.iSeATime ' Time
Op = Reader.dSeAOpen ' Open
High = Reader.dSeAHigh ' High
Low = Reader.dSeALow ' Low
Cl = Reader.dSeAClose ' Close
Volume = Reader.dSeAVolume ' Volume
Openint = Reader.dSeAOpenInterest 'OpenInterest

' Now display all values in a message box
For i = 0 To UBound(Dat)
 MsgBox Dat(i) & ", " & Time(i) & ", " & Op(i) & ", " & High(i) & ", "& _
     Low(i) & ", " & Cl(i) & ", " & Volume(i) & ", " & Openint(i)
Next I
Reader.CloseDirectory
```

Seek

The *Seek* function repositions the pointer in a previously opened file. (To open a security file use the *OpenSecurityBy** functions.) No data is actually read during the seek. After you have called the *Seek* function you should call the *ReadDay* function to read records from the specified position.

HRESULT Seek (
    int iNrOfDays
);

Parameters

iNrOfDays
    Number of days to move the pointer. This value can be > 0 or < 0.

Return Values

The *Seek* function can raise the following errors:

- File system errors

- ML_BAD_SEEK (0x067b, 1659)

For extended error descriptions go to the MetaLib error codes section.

Example

This example reads the last record of a price record file (Fxxx.dat or Fxxx.mwd) file and displays the price record in a message box:

```
Dim Reader As New MLReader
Reader.OpenDirectory "C:\MetaStock "
Reader.OpenSecurityBySymbol "MSFT"
Reader.SeekToEnd 'Move pointer to the file end
Reader.Seek (-1) '...and move up one record
Do While (Reader.iSeRecordsLeft > 0)
      Reader.ReadDay
      MsgBox (Reader.iSeDate & Chr(9) & Reader.dSeHigh & Chr(9) _
      & Reader.dSeLow & Chr(9) & Reader.dSeClose)
Loop
Reader.CloseDirectory
```

SeekToBegin (void)

The *SeekToBegin* function sets the position of the file pointer to the beginning of the file. (To open a security file use the *OpenSecurityBy** functions.) No data is actually read during the seek.

HRESULT SeekToBegin (void);

Parameters

This function has no parameters.

Return Values

The *SeekToBegin* function can raise the following errors:

- File system errors
- ML_BAD_SEEK (0x067b, 1659)

For extended error descriptions go to the MetaLib error codes section.

SeekToDate

The *SeekToDate* function repositions the pointer in a previously opened file to the specified date. (To open a security file use the *OpenSecurityBy** functions.) No data is actually read during the seek. After you have called the *Seek* function you must use the *ReadDay* function to read records from the specified position.

HRESULT SeekToDate (
    BSTR sDate
);

Parameters

sDate
    The date where to move the pointer. The date must have the following format: yyyymmdd

Return Values

The *SeekToDate* function can raise the following errors:

- File system errors

- ML_DATE_NOT_FOUND (0x0644, 1604)
- ML_BAD_SEEK (0x067b, 1659)

For extended error descriptions go to the MetaLib error codes section.

Example

This example reads all records starting from the date " 19991214" to the end of a security file and displays the price records in a message box:

```
Dim Reader As New MLReader
Reader.OpenDirectory "C:\MetaStock"
Reader.OpenSecurityBySymbol "MSFT"
Reader.SeekToDate "19991214"
Do While (Reader.iSeRecordsLeft > 0)
      Reader.ReadDay 'read price record
      MsgBox (Reader.iSeDate & Chr(9) & Reader.iSeHigh & Chr(9) _
      & Reader.iSeLow & Chr(9) & Reader.iSeClose)
Loop
Reader.CloseDirectory
```

SeekToEnd

The *SeekToEnd* function sets the position of the file pointer to the logical end of the file. (To open a security file use the *OpenSecurityBy** functions.) No data is actually read during the seek.

HRESULT SeekToEnd (void);

Parameters

This function has no parameters.

Return Values

The *SeekToEnd* function can raise the following errors:

- File system errors
- ML_BAD_SEEK (0x067b, 1659)

For extended error descriptions go to the MetaLib error codes section.

Example

This example reads the last record of a price record file (Fxxx.dat or Fxxx.mwd) file and displays the price record in a message box:

```
Dim Reader As New MLReader
Reader.OpenDirectory "C:\MetaStock"
Reader.OpenSecurityBySymbol "MSFT"
Reader.SeekToEnd 'Move pointer to the file end
Reader.Seek (-1) '...and move up one record
Do While (Reader.iSeRecordsLeft > 0)
      Reader.ReadDay
      MsgBox (Reader.iSeDate & Chr(9) & Reader.dSeHigh & Chr(9) _
      & Reader.dSeLow & Chr(9) & Reader.dSeClose)
Loop
Reader.CloseDirectory
```

SeekToNearestDate

The *SeekToNearestDate* function repositions the pointer in a previously opened file to the specified date. (To open a security file use the *OpenSecurityBy** functions.) If the date does not exist the previous (Set the bUp parameter to "FALSE") or the next (Set the bUp parameter to "TRUE") date in the file will be used. No data is actually read during the seek. After you have called the Seek function you must use the *ReadDay* function to read records from the specified position.

HRESULT SeekToNearestDate (
    BSTR sDate,
    BOOL bUp
);

Parameters

sDate
    The date where to move the pointer. The date must have the following format: yyyymmdd

bUp
    Direction where to move the file pointer if the date does not exist.

Return Values

The *SeekToNearestDate* function can raise the following errors:

- File system errors
- ML_BAD_SEEK (0x067b, 1659)

For extended error descriptions go to the MetaLib error codes section.

Other Properties

sDirectory

Returns the name of the currently opened MetaStock directory. (Read only)

MLWriter Interface

The *MLWriter* interface provides functions to write to MetaStock files. You can add/rename/delete securities, add/rename/delete price records, sort price records and much more.

There are three possibilities to add a price record to a security file:

1. Call the *OpenDirectory* function of the MLWriter interface. Use the *OpenSecurityBySymbol* or *OpenSecurityByName* function to specify where the price record should be written.

2. Call the *OpenSecurityByFilename* function of the MLWriter interface. The price records will be written to the file you have specified in the *BSTR sFilename* parameter.

3. Call the *OpenDirectory* function of the MLWriter interface. Use the *AppendSecurity* function to add a security to the master file.

Now you can call the *AppendDataRec* of the MLWriter interface to add a price record to the selected security file.

It is recommended that you call the *CloseDirectory* afterwards.

MLWriter Properties

bDateExists

Returns "True" if the specified date exists. The date has the format "YYYYMMDD".

Example

```
If Writer.bDateExists("Date") Then
  MsgBox ("Date exists")
End If

If Not Writer. bDateExists ("Date") Then
  MsgBox ("Date does not exists")
End If
```

bIsUsed

Returns "TRUE" if the directory you want to open is used by another program.

Example

```
Dim Writer as New MLWriter
Dim bIsUsed as Boolean

bIsUsed = Writer.bIsUsed ("C:\MetaStock")
```

bSymbolExists

Returns "TRUE" if the specified symbol exists.

Example

```
If Writer.bSymbolExists("Symbol") Then
  MsgBox ("Symbol exists")
End If

If Not Writer.bSymbolExists("Symbol") Then
  MsgBox ("Symbol does not exists")
End If
```

iFields

Sets the field size of the security. This value can be between 4 (Tick) and 8.
Note: By default MetaLib sets the field size of the price record file.

iLastDate

Returns the last date that is stored in the security file. (fx.dat file)

iLastTime

Returns the last time that is stored in the seucirty file.

sCopyrightString

Sets your own copyright string in the master, EMaster, XMaster and the price record files. This string can be 46 characters long.

sDirectory

Returns the name of the currently opened MetaStock directory. (Read only)

sPeriodicity

The period of the security is returned. (Daily, Weekly, Monthly or Intraday)

MLWriter Functions

AppendIntradaySecurity

The *AppendIntradaySecurity* function adds an intraday security record to a directory. Before you can use the *AppendIntradaySecurity* function, you must call the *CreateDirectory* or *OpenDirectory* function. If you want to add price records to this security record use the *AppendDataRec* function.

HRESULT AppendIntradaySecurity(
BSTR sSecuritySymbol,
BSTR sSecurityName,
INTERVAL enmInterval,
int iStartTime,
int iEndTime
);

Parameters

sSecuritySymbol
The ticker symbol for the security. (max. 14 chars)
sSecurityName
The name of the security. (max. 45 chars)
enmInterval
The enmInterval can have the following values:
None
Tick
Minute1
Minute5
Minute6
Minute10
Minute15
Minute20
Minute30
Minute60
iStartTime
The start time. The time must have the following format: HHMMSS (EX: 183000)
iEndTime
The end time. The time must have the following format: HHMMSS (EX: 183000)

Return Values

The *AppendIntradaySecurity* function can raise the following errors:

- File system errors
- ML_DUPLICATE_RECORD (0x0647, 1607)
- ML_NO_DIRECTORY_OPENED (0x0648, 1608)
- ML_INVALID_SECURITY_NAME (0x064a, 1610)
- ML_INVALID_SYMBOL (0x064b, 1611)
- ML_TOO_MUCH_SEC_FILES (0x064c, 1612)
- ML_INVALID_TIME (0x064f, 1615)

For extended error descriptions go to the MetaLib error codes section.

Example

The following example adds an intraday security to the directory.

```
Dim Writer As New MLWriter
Writer.CreateDirectory "C:\MetaStock"
Writer.AppendIntradaySecurity "MSFT","Microsoft", Minute1, 90000, 153000
Writer.AppendDataRec 20000101, 141500, 10, 20, 9, 19, 2323, 8333
Writer.CloseDirectory
```

AppendSecurity

The *AppendSecurity* function adds a security record to a directory. Before you can use the *AppendSecurity* function, you must call the *CreateDirectory* or *OpenDirectory* function. If you want to add price records to this security record use the *AppendDataRec* function.

HRESULT AppendSecurity(
BSTR sSecuritySymbol,
BSTR sSecurityName,
PERIODICITY sPeriodicity
);

Parameters

sSecuritySymbol
The ticker symbol for the security. (max. 14 chars)
sSecurityName
The name of the security. (max. 45 chars)
sPeriodicity
sPeriodicity can have following values: Intraday, Daily, Weekly, Monthly

Return Values

The *AppendSecurity* function can raise the following errors:

- File system errors
- ML_DUPLICATE_RECORD (0x0647, 1607)
- ML_NO_DIRECTORY_OPENED (0x0648, 1608)
- ML_INVALID_SECURITY_NAME (0x064a, 1610)
- ML_INVALID_SYMBOL (0x064b, 1611)
- ML_TOO_MUCH_SEC_FILES (0x064c, 1612)

For extended error descriptions go to the MetaLib error codes section.

Example

The following example only adds a security to a directory

```
Dim Writer As New MLWriter
Writer.CreateDirectory "C:\MetaStock"
Writer.AppendSecurity "MSFT", "Microsoft", Daily
Writer.AppendDataRec 20010721, 0, 123, 200, 100, 123, 3221, 0
Writer.CloseDirectory
```

AppendDataRec

The *AppendDataRec* function adds a price record to a security price record file. (Fxxx.dat or Fxxx.mwd) Before you can use the *AppendDataRec* function, you must call one of the *OpenSecurityBy** functions or the *AppendSecurity* function. Call the *Sort* function if you want to sort the price records after you have used the *AppendDataRec* function.

HRESULT AppendDataRec(
int iDate,
int iTime,
float dOpen,
float dHigh,
float dLow,
float dClose,
float dVolume,
float dOpenInterest
);

Parameters

iDate
The date must have the following format: YYYYMMDD.
iTime
The time must have the following format: HHMMSS.
dOpen
Open quotation
dHigh
High quotation
dLow
Low quotation
dClose
Close quotation
dVolume
Volume
dOpenInterest
Open interest

Return Values

The *AppendDataRec* function can raise the following errors:

- File system errors
- ML_INVALID_DATE (0x0645, 1605)
- ML_NO_DIRECTORY_OPENED (0x0648, 1608)
- ML_INVALID_TIME (0x064f, 1615)
- ML_PRICE_RECORD_ALREADY_EXISTS (0x0657, 1623)

For extended error descriptions go to the MetaLib error codes section.

Example

The following example calculates random values and saves these values to MetaStock™ files.

```
Dim Writer As New MLWriter
Dim iDate As Long
Writer.CreateDirectory "C:\MetaStock"

Randomize Timer
For i = 1 To 270
  Writer.AppendSecurity "SecSymbol" + CStr(i), "SecName" + CStr(i), Daily
  iDate = 20000104
  For j = 1 To 20
    fOpen = Int(Rnd * 500)
    fHigh = Int(Rnd * 500)
    fLow = Int(Rnd * 500)
    fClose = Int(Rnd * 500)
    fVolume = Int(Rnd * 10000)
    iOpenInt = Int(Rnd * 10000)
    Writer.AppendDataRec iDate, 0, fOpen, fHigh, fLow, fClose, fVolume,
iOpenInt
```

```
    iDate = iDate + 1
  Next j
Next i
```

AppendDataRecArray

The *AppendDataRecArray* function adds a price record array to a security price record file. (Fxxx.dat or Fxxx.mwd) All arrays must have the same size. Before you can use the *AppendDataRecArray* function, you must call one of the *OpenSecurityBy** functions or the *AppendSecurity* function.

HRESULT AppendDataRecArray(
SAFEARRAY(int) *iDate,
SAFEARRAY(int) *iTime,
SAFEARRAY(float) *dOpen,
SAFEARRAY(float) *dHigh,
SAFEARRAY(float) *dLow,
SAFEARRAY(float) *dClose,
SAFEARRAY(float) *dVolume,
SAFEARRAY(float) *dOpenInterest
);

Parameters

iDate
The date must have the following format: YYYYMMDD.
iTime
The time must have the following format: HHMMSS.
dOpen
Open quotation
dHigh
High quotation
dLow
Low quotation
dClose
Close quotation
dVolume
Volume
dOpenInterest
Open interest

Return Values

The *AppendDataRecArray* function can raise the following errors:

- File system errors
- ML_INVALID_DATE (0x0645, 1605)
- ML_NO_DIRECTORY_OPENED (0x0648, 1608)
- ML_INVALID_TIME (0x064f, 1615)
- ML_ARRAYS_MUST_HAVE_SAME_SIZE (0x0654, 1620)

For extended error descriptions go to the MetaLib error codes section.

Example

The following example creates price records arrays and saves these values to MetaStock™ files.

```
Private Declare Function GetTickCount Lib "kernel32" () As Long
Private Sub AppendArray
 Dim Writer As New MLWriter
```

```
 Dim Dat(1) As Long
 Dim Time(1) As Long
 Dim Op(1) As Single
 Dim High(1) As Single
 Dim Low(1) As Single
 Dim Cl(1) As Single
 Dim Volume(1) As Single
 Dim Openint(1) As Single

 Dat(0) = 20001016
 Time(0) = 0
 Op(0) = 23.12  'Open
 High(0) = 26.22 'High
 Low(0) = 22.2 'Low
 Cl(0) = 24.15 'Close
 Volume(0) = 4234332
 Openint(0) = 0

 Dat(1) = 20001017
 Time(1) = 0
 Op(1) = 24.15  'Open
 High(1) = 29.12 'High
 Low(1) = 21.87 'Low
 Cl(1) = 27.45 'Close
 Volume(1) = 123232
 Openint(1) = 0

 aNow = GetTickCount 'Timer start
 Writer.CreateDirectory "C:\test"
 Writer.AppendSecurity "Symbol", "Name", Daily
 Writer.AppendDataRecArray Dat, Time, Op, High, Low, Cl, Volume, Openint
 Writer.CloseDirectory

 aLater = GetTickCount 'Timer stop
 MsgBox "Added in " & CStr((aLater - aNow) / 1000) & " sec."
End Sub
```

ChangeLastPriceRecord

The *ChangeLastPriceRecord* overwrites the last record of a security file. No new record is added to the security file!

HRESULT ChangePriceFields(
int iDate,
int iTime,
float fOpen,
float fHigh,
float fLow,
float fClose,
float fVolume,
float fNewOpenInt
);

Parameters

iDate
The date must have the following format: YYYYMMDD.

iTime
The time must have the following format: HHMMSS.

dOpen
Open quotation

dHigh
High quotation
dLow
Low quotation
dClose
Close quotation
dVolume
Volume
dOpenInterest
Open interest

Return Values

The *ChangePriceFields* function can raise the following errors:

- File system errors
- ML_DATE_NOT_FOUND (0x0644, 1604)
- ML_INVALID_DATE (0x0645, 1605)
- ML_NO_DIRECTORY_OPENED (0x0648, 1608)
- ML_INVALID_TIME (0x064f, 1615)

ChangePriceFields

The *ChangePriceFields* function allows you to change a price record of a security price record file. (Fxxx.dat or Fxxx.mwd). Before you can use the *ChangePriceFields* function, you must call one of the *OpenSecurityBy** functions.

HRESULT ChangePriceFields(
int iDate,
int iNewDate,
int iNewTime,
float dNewOpen,
float dNewHigh,
float dNewLow,
float dNewClose,
float dNewVolume,
float dNewOpenInt
);

Parameters

iDate
The date of the record you want to change.
iNewDate
The new date value. If you do not want to change the value use the value "-1".
iNewTime
The new time value. If you do not want to change the value use the value "-1".
dNewOpen
The new open value. If you do not want to change the value use the value "-1".
dNewHigh
The new high value. If you do not want to change the value use the value "-1".
dNewLow
The new low value. If you do not want to change the value use the value "-1".
dNewClose
The new close value. If you do not want to change the value use the value "-1".
dNewVolume
The new volume value. If you do not want to change the value use the value "-1".
dNewOpenInt
The new open interest value. If you don't want to change the value use the value "-1".

Return Values

The *ChangePriceFields* function can raise the following errors:

- File system errors
- ML_DATE_NOT_FOUND (0x0644, 1604)
- ML_INVALID_DATE (0x0645, 1605)
- ML_NO_DIRECTORY_OPENED (0x0648, 1608)
- ML_INVALID_TIME (0x064f, 1615)

For extended error descriptions go to the MetaLib error codes section.

Example

This example changes a security record of the "C:\Vienna\f1.dat" file.

```
Dim Writer As New MLWriter
Writer.OpenSecurityByFilename "C:\MetaStock\f1.dat"
'The date of the record won't be changed. Some price values will be changed
Writer.ChangePriceFields 20001021,-1,-1,23.1,24.11,21.02,24.2,32321,202
'The date of the record will be changed. Some price values will be chaged
Writer.ChangePriceFields 20001022,200001023,-1,-1,-1,-1,20,32321,0
Writer.CloseDirectory
```

ChangePriceFieldsIntraday

The *ChangePriceFieldsIntraday* function allows you to change an intraday price record of a security price record file. (Fxxx.dat or Fxxx.mwd) Before you can use the *ChangePriceFieldsIntraday* function, you must call one of the *OpenSecurityBy** functions.

HRESULT ChangePriceFields(
int iDate,
int iNewDate,
int iTime,
int iNewTime,
float dNewOpen,
float dNewHigh,
float dNewLow,
float dNewClose,
float dNewVolume,
float dNewOpenInt
);

Parameters

iDate
The date of the record you want to change.
iNewDate
The new date value. If you do not want to change the value use the value "-1".
iTime
The time of the record you want to change.
iNewTime
The new time value. If you do not want to change the value use the value "-1".
dNewOpen
The new open value. If you do not want to change the value use the value "-1".
dNewHigh
The new high value. If you do not want to change the value use the value "-1".
dNewLow
The new low value. If you do not want to change the value use the value "-1".
dNewClose
The new close value. If you do not want to change the value use the value "-1".

dNewVolume
The new volume value. If you do not want to change the value use the value "-1".
dNewOpenInt
The new open interest value. If you don't want to change the value use the value "-1".

Return Values

The *ChangePriceFields Intraday* function can raise the following errors:

- File system errors
- ML_DATE_NOT_FOUND (0x0644, 1604)
- ML_INVALID_DATE (0x0645, 1605)
- ML_NO_DIRECTORY_OPENED (0x0648, 1608)
- ML_INVALID_TIME (0x064f, 1615)

For extended error descriptions go to the MetaLib error codes section.

ChangeSecurityRecord

The *ChangeSecurityRecord* allows you to change the name, symbol or the periodicity of a security record of a Master/EMaster or XMaster file. Before you can use the *ChangeSecurityRecord* function, you must call the *OpenDirectory* function.

HRESULT ChangeSecurityRecord(
BSTR sSecSymbol,
BSTR sNewSecName,
BSTR sNewSecSymbol
);

Parameters

sSecSymbol
The symbol of the security you want to change.
sNewSecName
The new name of the security. If you don't want to change the value pass the value "-1".
sNewSecSymbol
The new security symbol. If you don't want to change the value pass the value "-1".

Return Values

The *ChangeSecurityRecord* function can raise the following errors:

- File system errors
- ML_SECURITY_NAME_NOT_FOUND (0x0643, 1603)
- ML_DUPLICATE_RECORD (0x0647, 1607)

For extended error descriptions go to the MetaLib error codes section.

Example

This example changes the symbol of the security record "DAX" to the symbol "DAX 30". The security name will not be changed.

```
Dim Writer As New MLWriter
Writer.OpenDirectory "C:\MetaStock"
Writer.ChangeSecurityRecord "DAX", "-1", "DAX 30"
Writer.CloseDirectory
```

CloseDirectory

Call the *CloseDirectory* function if you have all data written to the master and security files. The first and last date of the price records will be written to the master files.

HRESULT CloseDirectory(void);

Return Values

The *CreateDirectory* function can raise the following errors:

- File system errors

For extended error descriptions go to the MetaLib error codes section.

CreateDirectory

The *CreateDirectory* function creates a new directory where Master/EMaster/XMaster and security price record files (Fxxx.dat or Fxxx.mwd) can be stored.
Note: If the directory already exists all files and subfolders of the directory will be deleted.
Use the *CreateDirectoryEx* function if you do not want to delete all the subfolders of a directory. (Writer.CreateDirectoryEx "C:\MetaStock", false)

HRESULT CreateDirectory(
    BSTR sDirectory
);

Parameters

sDirectory
    The name of the directory you want to create. (EX: "C:\MetaStock\Dow")

Return Values

The *CreateDirectory* function can raise the following errors:

- File system errors
- ML_CREATION_OF_DIRECTORY_FAILED (0x0646, 1606)

For extended error descriptions go to the MetaLib error codes section.

DeleteIntradaySecRecord

The *DeleteIntradaySecRecord* function removes an intraday price record of a security price record file (Fxxx.dat or Fxxx.mwd). Before you can use the *DeleteIntradaySecRecord* function, you must call one of the *OpenSecurityBy** functions.

HRESULT DeleteIntradaySecRecord(
    int iDate,
    int iTime
);

Parameters

iDate
    The date of the intraday price record you want to delete.
iTime
    The time of the intraday price record you want to delete.

Return Values

The *DeleteIntradaySecRecord* function can raise the following errors:

- File system errors

- ML_DATE_NOT_FOUND (0x0644, 1604)
- ML_FAILED_TO_DEL_PRICE_RECORD (0x0652, 1618)

For extended error descriptions go to the MetaLib error codes section.

DeleteIntradaySecRecordEx

The *DeleteIntradaySecRecordEx* function removes one ore more intraday price records of a security price record file (Fxxx.dat or Fxxx.mwd) between a certain time range. Before you can use the *DeleteIntradaySecRecordEx* function, you must call one of the *OpenSecurityBy** functions.

HRESULT DeleteIntradaySecRecordEx(
int iDate,
int iTimeFrom,
int iTimeTo
);

Parameters

iDate
The date of the intraday price record you want to delete.
iTimeFrom
The start time of the time range.
iTimeTo
The end time of the time range.

Return Values

The *DeleteIntradaySecRecordEx* function can raise the following errors:
- File system errors
- ML_DATE_NOT_FOUND (0x0644, 1604)
- ML_FAILED_TO_DEL_PRICE_RECORD (0x0652, 1618)

For extended error descriptions go to the MetaLib error codes section.

DeleteSecurity

The *DeleteSecurity* function allows you to remove a security record from a Master/EMaster or XMaster file. Before you can use the *DeleteSecurity* function, you must call the *OpenDirectory* function.

HRESULT DeleteSecurity(
BSTR sSecSymbol
);

Parameters

sSecSymbol
The symbol of the security you want to delete.

Return Values

The *DeleteSecurity* function can raise the following errors:
- *File system errors*
- ML_SECURITY_SYMBOL_NOT_FOUND (0x0642, 1602)
- ML_INVALID_SYMBOL (0x064b, 1611)

For extended error descriptions go to the MetaLib error codes section.

Example

This example removes the security "MSFT" from the directory.

```
Dim Writer As New MLWriter
Writer.OpenDirectory "C:\MetaStock"
Writer.DeleteSecurity "MSFT"
Writer.CloseDirectory
```

DeleteSecurities

The *DeleteSecurities* function allows you to remove one or more security records from a Master/EMaster or XMaster file. Before you can use the *DeleteSecurities* function, you must call the *OpenDirectory* function.
Note: This function cannot be called from Visual Basic!

HRESULT DeleteSecurities(
    SAFEARRAY(BSTR) sSecuritySymbols
);

Parameters

sSecuritySymbols
    The symbol of the securities you want to delete.

Return Values

The *DeleteSecurities* function can raise the following errors:

- File system errors
- ML_SECURITY_SYMBOL_NOT_FOUND (0x0642, 1602)
- ML_INVALID_SYMBOL (0x064b, 1611)

For extended error descriptions go to the MetaLib error codes section.

DeleteSecRecord

The *DeleteSecRecord* function removes a price record from a security price record file (Fxxx.dat or Fxxx.mwd). Before you can use the *DeleteSecRecord* function, you must call one of the *OpenSecurityBy** functions.

HRESULT DeleteSecRecord(
    int iDate
);

Parameters

iDate
    The date of the price record you want to delete.

Return Values

The *DeleteSecRecord* function can raise the following errors:

- File system errors
- ML_DATE_NOT_FOUND (0x0644, 1604)
- ML_FAILED_TO_DEL_PRICE_RECORD (0x0652, 1618)

For extended error descriptions go to the MetaLib error codes section.

OpenDirectory

The *OpenDirectory* function opens a directory that contains Master/EMaster/XMaster and security price record files. (Fxxx.dat or Fxxx.mwd) The function returns the "ML_ACCESS_DENIED (0x0674, 1652)" error if the directory is currently opened by MetaStock.

OpenDirectory(
    BSTR sDirectory
);

Parameters

sDirectory
    The name of the directory that you want to open.

Return Values

The *OpenDirectory* function can raise the following errors:

- File system errors
- ML_DIRECTORY_NOT_EXIST (0x0649, 1608)
- ML_ACCESS_DENIED (0x0674, 1652)

For extended error descriptions go to the MetaLib error codes section.

OpenSecurityByFilename

The *OpenSecurityByFileName* function opens the Fxxx.xxx file specified in the sFileName parameter. Use the *AppendDataRec* function to add price records to the specified file.

OpenSecurityByFilename(
    BSTR sFileName
);

Parameters

sFileName
    The name of the file containing the security records. (EX: "C:\MetaStock\Dow\F1.dat"

Return Values

The *OpenSecurityByFilename* function can raise the following errors:

- File system errors

For extended error descriptions go to the MetaLib error codes section.

Example

This example appends a security record at the end of the "C:\MetaStock\Dow\F1.dat" file.

```
Dim Writer As New MLWriter
Writer.OpenSecurityByFilename "C:\MetaStock\F1.dat"
Writer.AppendDataRec 20001021,0,23.12,24.11,21.02,24.2,0,0
Writer.CloseDirectory
```

OpenSecurityByName

The *OpenSecurityByName*function opens the Fx.xxx file that contains the specified security name. Use the *AppendDataRec* function to add price records to the specified file. Before you can call this function you must call the *OpenDirectory* function.

OpenSecurityByName(
BSTR sSecurityName
);

Parameters

sSecurityName
The name of the security. (Note: Security name is not case sensitive.)

Return Values

The *OpenSecurityByName* function can raise the following errors:

- File system errors
- ML_SECURITY_NAME_NOT_FOUND (0x0643, 1603)
- ML_NO_DIRECTORY_OPENED (0x0648, 1608)

For extended error descriptions go to the MetaLib error codes section.

Example

This example adds a security record at the end of a fxxx.xxx the file that was opened with the OpenSecurityByName function.

```
Dim Writer As New MLWriter
Writer.OpenDirectory "C:\MetaStock"
Writer.OpenSecurityByName "Microsoft"
Writer.AppendDataRec 20001021,0,23.12,24.11,21.02,24.2,0,0
Writer.CloseDirectory
```

OpenSecurityBySymbol

The *OpenSecurityBySymbol* function opens the Fx.xxx file that contains the specified security symbol. Use the *AppendDataRec* function to add price records to the specified file. Before you can call this function you must call the *OpenDirectory* function.

HRESULT OpenSecurityBySymbol (
BSTR sSecuritySymbol
);

Parameters

sSecuritySymbol
The ticker symbol for the security. (Note: Security symbol is not case sensitive.)

Return Values

The *OpenSecurityBySymbol* function can raise the following errors:

- File system errors
- ML_SECURITY_SYMBOL_NOT_FOUND (0x0642, 1602)
- ML_NO_DIRECTORY_OPENED (0x0648, 1608)

For extended error descriptions go to the MetaLib error codes section.

Example

This example adds a security record at the end of a security price record file opened by the OpenSecurityBySymbol function.

```
Dim Writer As New MLWriter
Writer.OpenDirectory "C:\MetaStock"
Writer.OpenSecurityBySymbol "MSFT"
Writer.AppendDataRec 20001021,0,23.12,24.11,21.02,24.2,0,0
Writer.CloseDirectory
```

Sort

The *Sort* function sorts the price records of the opened security.

HRESULT Sort (void);

Return Values

The *Sort* function can raise the following errors:

- File system errors

For extended error descriptions go to the MetaLib error codes section.

Example

This example appends some price records to a security file and sorts them afterwards.

```
Dim Writer As New MLWriter
Writer.CreateDirectory "C:\MetaStock"
Writer.AppendSecurity "MSFT", "Microsoft", Daily
Writer.AppendDataRec 20001021,0,23.12,24.11,21.02,24.2,0,0
Writer.AppendDataRec 20001020,0,23.12,24.11,21.02,24.2,0,0
Writer.AppendDataRec 20001024,0,23.12,24.11,21.02,24.2,0,0
Writer.AppendDataRec 19990920,0,23.12,24.11,21.02,24.2,0,0
Writer.AppendDataRec 20100621,0,23.12,24.11,21.02,24.2,0,0
Writer.AppendDataRec 19980101,0,23.12,24.11,21.02,24.2,0,0
Writer.Sort
Writer.CloseDirectory
```

After you have called the "Sort" function, the dates will be stored in the following sequence: 19980101, 19990920, 20001020, 20001021, 20001024, 20100621

SortEx

The *SortEx* function sorts the price records of the opened security. It also removes duplicate records from the security price record file.

HRESULT SortEx (void);

Return Values

The *SortEx* function can raise the following errors:

- File system errors

For extended error descriptions go to the MetaLib error codes section.

Split

The *Split* function is used to adjust the price data and volume for a stock split. Please use caution when applying this command. Once an adjustment is made, it may be difficult to bring the data back to its original state. Use the *SplitEx* function if you want to set a start and end date of the price records that should be splitted.

HRESULT Split (
    float fRatio,
    float fRatio2,
    [optional] VARIANT_BOOL bSplitVolume
);

Parameters

fRatio, fRatio2
e.g. 2 for 1, 3 for 2, etc.
bSplitVolume
Set this value to "TRUE" if you also want to split the volume. (default)

Return Values

The *Split* function can raise the following errors:

- File system errors
- ML_DATE_NOT_FOUND (0x0644, 1604)
- ML_FAILED_TO_SPLIT_PRICE_RECORD (0x0659, 1626)

More info

When a stock splits, the historical prices must be adjusted to reflect the split. For example, if XYZ company has a 2 for 1 stock split, shareholders will receive two shares for each share that they own. To keep the total market value of the stock equal, the price per share is reduced by one-half. Therefore shareholders will have twice as many shares at half the price. Stock splits are usually done in order to increase the liquidity of the stock.

When price data is adjusted for a stock split, the volume is also affected. For example, in a 2 for 1 split where prices are cut in half, the volume is doubled so that the market value remains unchanged.

Example

Price records before using Split function (Open, High, Low, Close, Volume):

```
33.1800     33.2800     32.1300     32.9500     1351801
```

Using the split function:

```
Dim w As New MLWriter
w.OpenDirectory "C:\Metastock"
w.OpenSecurityBySymbol "MSFT"
w.Split 2,1, true
```

Price records after using the Split function (Open, High, Low, Close, Volume):

```
16.5900     16.6400     16.0650     16.4750     2703602
```

UpdateChart

The *UpdateChart* function updates the values of opened charts in MetaStock in real-time. The chart will be updated automatically and you do not have to press the "refresh" button in MetaStock manually anymore.

Note: This function is only available in the "Real-time" version of MetaLib!

```
Dim Writer As New MLWriter
Writer.OpenDirectory "C:\MetaStock"
Writer.OpenSecurityBySymbol "MSFT"
Writer.AppendDataRec 20001021,0,23.12,24.11,21.02,24.2,0,0
Writer.UpdateChart
```

MLSecurityFinder Interface

The *MLSecurityFinder* interface provides functions to search for securities in directories. A search progress dialog can be displayed during this operation.

MLSecurityFinder Functions

DestroySearchDialog

If the b*DisplaySearchDialog* parameter is set to "TRUE" MetaLib displays a progress dialog during the search. After the search has finished you must call the *DestroySearchDialog* function that the dialog will disappear.

HRESULT DestroySearchDialog (void);

Return Values

The *DestroySearchDialog* function can raise the following errors:

- ML_SEARCH_DIALOG_FAILED (0x0664, 1636)

For extended error descriptions go to the MetaLib error codes section.

FindNextSecurity

The *FindNextSecurity* function continues a security search from a previous call to the *FindSecurity* function.

HRESULT FindNextSecurity (void);

Return Values

The *FindNextSecurity* function can raise the following errors:

- File system errors
- ML_SECURITY_NAME_NOT_FOUND (0x0643, 1603)

For extended error descriptions go to the MetaLib error codes section.

FindSecurity

The *FindSecurity* function allows you to search for a specified security name/symbol in all MetaStock™ files that are stored in a directory. Use the *FindNextSecurity* function to continue the security search.

HRESULT FindSecurity (
    BSTR sSecurityName,
    BSTR sDirectory
);

Parameters

sSecurityName
    The name of the security name or security symbol. (Note: Security name/symbol is not case sensitive.) The parameter can contain wildchars like "*" or "?".

sDirectory
    The name of the directory where to start the search. If the *fScanSubDirectories* property is set to "TRUE" MetaLib scans through all subdirectories of the directory specified in the *sDirectory* parameter.

Return Values

The *FindSecurity* function can raise the following errors:

- File system errors
- ML_SECURITY_NAME_NOT_FOUND (0x0643, 1603)

For extended error descriptions go to the MetaLib error codes section.

Example

This example scans trough all directories of "C:" and checks if there are some MetaStock™ files containing price records of Microsoft. If a file was found the file name will be displayed in a message box. You can find a C++ sample in the C:\…\MetaLib\Examples\VC\SecurityFinder directory.

```
On Error GoTo errorhandler
Dim Finder As New MLSecurityFinder

Finder.bDisplaySearchDialog = True
Finder.bScanSubDirectories = True
Finder.FindSecurity "Micros*", "c:" ' You can use wildchars: * or ?
'Finder.FindSecurity "MSFT", "c:" ' You can also use a ticker symbol

MsgBox (Finder.SecFileName)
Do
   Finder.FindNextSecurity
   MsgBox (Finder.SecFileName)
Loop

Finder.DestroySearchDialog
Exit Sub

errorhandler:
If (Err.Number = 1603) Then
   Finder.DestroySearchDialog
   Exit Sub
End If
```

MLSecurityFinder Properties

bDisplaySearchDialog

Set the *fDisplaySearchDialog* property to "TRUE" if you want that a progress dialog will appear during the search. Don't forget to call the *DestroySearchDialog* function after the search.

bScanSubDirectories

Set the *fScanSubDirectories* property to "TRUE" if you want that MetaLib scan trough all subdirectories of the directory specified in the *sDirectory* parameter of the *FindSecurity* function.

hSearchDialog

Returns the window handle of the search dialog. This property only contains a valid value if the *bDisplaySearchDialog* parameter was set to "TRUE".

SecFileName

The *SecFileName* property contains the filename of the specified security after you have called the *FindSecurity* or *FindNextSecurity* function and these functions didn't raise any error.

SecName

The *SecName* property contains the name of the specified security after you have called the *FindSecurity* or *FindNextSecurity* function and these functions didn't raise any error.

SecSymbol

The *SecSymbol* property contains the symbol of the specified security after you have called the *FindSecurity* or *FindNextSecurity* function and these functions didn't raise any error.

sSearchDialogTitle

Returns/sets the window title of the search dialog. This property only contains a valid value if the *bDisplaySearchDialog* parameter was set to "TRUE".

MLConverter Interface

AIQ2MetaStock

The *AIQ2MetaStock* function converts all AIQ format files of a directory to the MetaStock™ format.

An **EOD** record in your AIQ text file should look like this:

```
NT,d,11/21/97,11.726,11.726,11.398,11.468,40472,0
```

Symbol, Periodicity (d,m,w), Date (DD/MM/YY), Open, High, Low, Close, Volume, Open interest

An **intraday** record in your AIQ text file should look like this:

```
AKAM,I,09/01/06,0849,39.820,39.820,39.650,39.740,260,0
```

Symbol, Periodicity (i), *Date (DD/MM/YY), Time(HHMM), Open, High, Low, Close, Volume, Open interest*

HRESULT AIQ2MetaStock (
BSTR sSourceDirectory**,**
BSTR sOutputDirectory**,**
BSTR sFileExtension
);

Parameters

sSourceDirectory
The directory that contains the AIQ files.
sOutputDirectory
The directory where to write the files in MetaStock™ format. If the directory does not exist a new directory will be created automatically.
sFileExtension
The directory where to write the files in MetaStock™ format.

Return Values

The *AIQ2MetaStock* function can raise the following exceptions:

- File system errors

For extended error descriptions go to the MetaLib error codes section.

Example

```
Dim Converter As new MLConverter
Converter.AIQ2MetaStock "C:\AIQ", "C:\MetaStock", "TXT"
```

MetaStock2CSV

The *MetaStock2CSV* function converts all MetaStock™ security files of the specified directory to the CSV format. The *MetaStock2CSVEx* function works the same way as the *MetaStock2CSV* function except that the "Open Interest" value will also be stored in the CSV file. Therefore you should use the *MetaStock2CSV* function for converting stocks and the *MetaStock2CSVEx* function to convert Futures to CSV format.

HRESULT MetaStock2CSV (
BSTR sInputDirectory,
BSTR sOutputDirectory
BOOL bIncludeTimeField
BOOL bUseSecNamesAsFileName
);

Parameters

sSourceDirectory
The directory that contains the Metastock™ files. (Master, Fx.dat)
sOutputDirectory
The directory where to write the CSV files.
bIncludeTimeField
Set this value to true to store the time value in the CSV file.
bUseSecNamesAsFileName
Set this value to true to use the security name as CSV file name instead of the symbol code.

Return Values

The *MetaStock2CSV* function can raise the following exceptions:

- File system errors

For extended error descriptions go to the MetaLib error codes section.

Example

```
Dim Converter As new MLConverter
Converter.MetaStock2CSV "C:\Metastock","C:\CSV", True, False
```

MLProgBar Interface

MLProgBar Properties

bCancel

Check this property periodically to determine if the user has cancelled the operation.

Return Value

TRUE if the user has selected the Cancel button, otherwise false.

bEnableCancelButton

Set this option to TRUE if you want that a cancel button is displays on the progress bar. Otherwise set this option to FALSE. The default value of this property is TRUE.

MLProgBar Functions

The MLProgBar functions allow you to use a progress dialog in your application. A progress dialog is what you commonly see during setup operations. It contains a progress indicator for updating the user on the completion status of a lengthy operation. The progress dialog includes the following features:

- A Cancel button, along with a message handler for the button. (optional)
- A status text window.
- A percentage display of the current status.

Create

Call this function to create the progress dialog.

HRESULT Create (void);

Parameters

This function has no parameters.

Return Value

The Create function can raise the following errors:

- ML_PROGRESSBAR_FAILED (0x0663, 1635)

For extended error descriptions go to the MetaLib error codes section.

Destroy

Call this function to destroy the progress dialog.

HRESULT Destroy (void);

Parameters

This function has no parameters.

Return Value

The *Destroy* function can raise the following exceptions:

- ML_PROGRESSBAR_FAILED (0x0663, 1635)

For extended error descriptions go to the MetaLib error codes section.

OffsetPos

Advances the progress bar's current position by the increment specified by pos and redraws the bar to reflect the new position.

HRESULT OffsetPos(
    int iPos
);

Parameters

iPos

Amount to advance the position.

SetPos

Sets the progress bar's current position as specified by pos and redraws the bar to reflect the new position. The position of the progress bar is not the physical location on the screen, but rather is between the upper and lower range indicated in *SetRange*.

HRESULT SetPos (
    int iPos
);

Parameters

iPos
    The new position of the progress bar.

SetRange

Sets the upper and lower limits of the progress bar control's range and redraws the bar to reflect the new ranges.

HRESULT SetRange (
    int iLower**,**
    int iUpper
);

Parameters

iLower
    Specifies the lower limit of the range (default is zero).
iUpper
    Specifies the upper limit of the range (default is 100).

SetStatus

Call this function when you wish to update the status text.

HRESULT SetStatus (
    BSTR sMessage
);

Parameters

sMessage
    A string that contains the status text to be displayed.

SetStep

Specifies the step increment for a progress bar. The step increment is the amount by which a call to *StepIt* increases the progress bar's current position.

HRESULT SetStep (
    int iStep
);

Parameters

iStep
    New step increment.

StepIt

Advances the current position for a progress bar control by the step increment and redraws the bar to reflect the new position. The step increment is set by the *SetStep* member function.

HRESULT StepIt (void);

Parameters

This function has no parameters.

MetaLib Error Codes

To catch the MetaLib errors, use the following mechanism:

VB/VBA:

Use the "On Error Goto" statement:

```
On Error Goto MLError

Dim reader As New MLReader

reader.OpenSecurityByFilename "C:\Vienna\F1.dat"

Do While (reader.iSeRecordsLeft > 0)
      reader.ReadDay
      Debug.Print reader.iSeDate & ", " & reader.dSeOpen & ", " & _
      reader.dSeClose
Loop

Exit Sub

MLError:
      // Display the error message
      MsgBox(Err.Number & CHR(13) & Err.Description)
```

C(++)

Use the "SUCCEEDED" and "FAILED" macros to determine if the function succeeded/failed.

```
HRESULT     hr = S_OK;
if(SUCCEEDED(hr = g_IMLReader->OpenDirectory (L"E:\\Vienna")))
{
      g_IMLReader->get_iMaRecordsLeft (&iMaRecLeft);
}
else
{
      // Display the error message
      AfxMessageBox(GetErrorDescription(hr));
}
```

To display "HRESULT" values its' recommended that you display the error codes in "Hexadecimal Display". For example the ML_SECURITY_NAME_NOT_FOUND error has the following error value: 0x800a0643. Take the last three digits and compare this number with the error constants in the MLErrorConstatns.h file. Now you know the name of the error that was raised by one of the MetaLib functions.

MetaLib can raise the following errors:

General Error Codes

ML_FILE_NAME_NOT_FOUND (0x0641, 1601)

The file name that was passed with the *OpenSecurityByFilename* function doesn't exist.

ML_SECURITY_SYMBOL_NOT_FOUND (0x0642, 1602)

The symbol name of the security that was passed with the *OpenSecurityBySymbol* function doesn't exist in the directory. Call the *OpenDirectory* function to open another directory.

ML_SECURITY_NAME_NOT_FOUND (0x0643, 1603)

The name of the security that was passed with the *OpenSecurityByName* function doesn't exist in the directory. Call the *OpenDirectory* function to open another directory.

ML_DATE_NOT_FOUND (0x0644, 1604)

The date that was passed with the *SeekToDate* function doesn't exist in the specified security file. Use the *OpenSecurityBy** functions to open another security file.

ML_INVALID_DATE (0x0645, 1605)

The date that was passed with the *AppendDataRec* function is invalid. The date must have the following form: yyyymmdd

ML_CREATION_OF_DIRECTORY_FAILED (0x0646, 1606)

The creation of the directory failed.

ML_DUPLICATE_RECORD (0x0647, 1607)

The security name and the stock name that were passed with the *AppendSecurity* function already exists in the current directory. Call the *OpenDirectory* function to open another directory.

ML_NO_DIRECTORY_OPENED (0x0648, 1608)

You have used the *OpenSecurityBy** functions without specifying a directory first. Before you can call one of the *OpenSecurityBy** functions you must call the *OpenDirectory* or *CreateDirectory* function.

ML_DIRECTORY_NOT_EXIST (0x0649, 1609)

The directory that you have tried to open does not exist.

ML_INVALID_SECURITY_NAME (0x064a, 1610)

The length of the security name is too short/long. The security name can have max. 45 chars.

ML_INVALID_SYMBOL (0x064b, 1611)

The length of the security symbol is too short/long. The security symbol can have max. 14 chars.

ML_TOO_MUCH_SEC_FILES (0x064c, 1612)

The maximum number of security files in one directory has been reached. Only 2000 security files can be stored in one directory.

ML_FILE_IS_USED_BY_ANOTHER_APP (0x064d, 1613)

The file is used by another application. You must close the running application first.

ML_INVALID_TIME_FRAME (0x064e, 1614)

Supported time frames: D (daily), W (weekly), M (monthly)

ML_ONLY_FOR_US_MARKET (0x0650, 1616)

Historical quotes are only available for the US market.

ML_CONTAINS_NO_METASTOCK_FILES (0x0651, 1617)

The directory contains no MetaStock™ files. You must select a another direcotory.

ML_FAILED_TO_DEL_PRICE_RECORD (0x0652, 1618)

Failed to delete the price record.

ML_FAILED_TO_CHANGE_PRICE_RECORD (0x0653, 1619)

Failed to change the price record. Make sure that the date that identifies the price record is correct.

ML_ARRAYS_MUST_HAVE_SAME_SIZE (0x0654, 1620)

The arrays must all have the same size.
This code causes this error:

```
Dim Time(2000) As Long
Dim High(1000) As Single
Dim Close(1000) As Single
….

Writer.AppendDataRecArray Dat, Time, Op, High, Low, Cl, Volume, Openint
```

Correct:

```
Dim Time(1000) As Long
Dim High(1000) As Single
Dim Close(1000) As Single
….

Writer.AppendDataRecArray Dat, Time, Op, High, Low, Cl, Volume, Openint
```

ML_COPYRIGHT_STRING_TOO_LONG (1621, 0x0655)

The copyright string is too long.

ML_NO_QUOTES_AVAILABLE (1622, 0X0656)

No quoatations available. (Invalid symbol or wrong start and end date specified)

ML_PRICE_RECORD_ALREADY_EXISTS (0x0657, 1623)

The security record already exist in the security file. You cannot add a record with the same date, open, high, low, close, volume price again.

ML_FAILED_TO_DIAL_UP (0x065e, 1630)

A connection to the Internet could not be established. Check if the connection name is correct.

ML_SERVER_NOT_CONNECTED (0x065f, 1631)

A connection to the Yahoo.com server could not be established.

ML_FAILED_TO_HANG_UP (0x0660, 1632)

Failed to disconnect a RAS connection.

ML_CONNECTION_NAME_NOT_FOUND (0x0661, 1633)

The connection name doesn't exist. Use a name that was returned by the *Connections* property.

ML_PROGRESSBAR_FAILED (0x0663), 1635)

The creation of the progress bar failed.

ML_SEARCH_DIALOG_FAILED (0x0664), 1636)

The creation of the search dialog failed.

File system error codes:

ML_DISK_FULL (0x0672, 1650)

The disk is full.

ML_TOO_MANY_OPEN_FILES (0x0673, 1651)

The permitted number of open files was exceeded.

ML_ACCESS_DENIED (0x0674, 1652)

The file could not be accessed.

ML_LOCK_VIOLATION (0x0675, 1653)

There was an attempt to lock a region that was already locked.

ML_SHARING_VIOLATION (0x0676, 1654)

SHARE.EXE was not loaded, or a shared region was locked.

ML_HARD_IO (0x0677, 1655)

There was a hardware error.

ML_INVALID_FILE (0x0678, 1656)

There was an attempt to use an invalid file handle.

ML_FILE_NOT_FOUND (0x0679, 1657)

The file could not be located.

ML_BAD_PATH (0x067a, 1658)

All or part of the path is invalid.

ML_BAD_SEEK (0x067b, 1659)

There was an error trying to set the file pointer.

ML_GENERIC (0x067c, 1660)

An unspecified error occurred.

Metastock™ Format Description

File Structure Description

Metastock™ uses three files to store its information:
Master: This is the main file that is created and put in a directory, and it contains the information of all the files that are stored in the directory.
Every stock is saved (stored) in an Fx.dat or Fx.mwd file. (The (x) here represents a number.)

Master

The Master file contains the following information:

- Total Record Number
- Highest Record Used
- Copyright Info
- Chat values usage
- File Number
- Number of Fields
- Auto Flag
- Record Size
- Symbol
- File Name
- Time Frame (I, D, W, M)
- Time Interval, the time between trades in minute intervals, minimum 1 minute
- First Date on Record
- The beginning of the trade for the stock of the present date
- Last Date on Record
- The end of the trade for the stock of the present date
- Start of trading
- End of trading

Fx.dat and Fx.mwd file

The Fx.dat and Fx.mwd files contain the following information:

- Date
- Time
- Open
- High
- Low
- Close
- Volume

Registration and Price Information

If you decide to use the MetaLib SDK only for private purposes you have to order the private licence for **59.95 EURO**.

If you decide to use the MetaLib SDK for commercial products you have to order the commercial version of MetaLib. **(249.95 EURO)**

If you want to update opened charts in MetaStock real-time you have to order the real-time version of MetaLib. **(449.95 EURO)**

Licence type	Price	Use
Private	59.95 EUR	Only for private purposes. (You can only install MetaLib on these computers that are owned by you)
Commercial	249.95 EUR	For private and commercial purposes (You can distribute MetaLib to as many PCs as you want)
Real-time	449.95 EUR	For private and commercial purposes (You can distribute MetaLib to as many PCs as you want) Charts that are opened in MetaStock can be refreshed automatically with the MetaLib DLL in real-time. (UpdateChart function is only available in this version)

This copy of MetaLib is available for a 20 days evaluation period. If you decide to continue using the program, you need to register the program by using one of the methods described below. By registering the software, you are able to continue using it legally, and are supporting our efforts to continually develop innovative products to best serve your needs. If you order via credit card you receive the fully registered version of MetaLib within 5 minutes!

You can order MetaLib online on:
http://shareit1.element5.com/product.html?productid=199762

You can pay via **credit card**, **wire transfer**, **cash** or **check**. Please follow the instructions on the website.

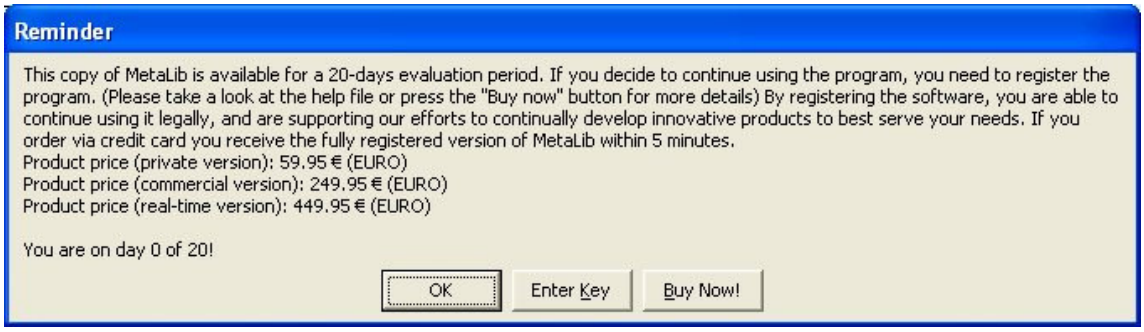
Benefits of Registering:

- No restrictions on the number of starts
- No splashing screen at start up
- Free technical support provided by email. Questions usually answered same day

How to use the Registration Key?

Private version:

When you start MetaLib following dialog will appear:



Click on the "Enter key" to enter your unlock key.

Commercial and Real-time version:

It is necessary that you call the *SetRegistrationInfo* of the *MLRegistration* interface before you call any other function of MetaLib. In the *SetRegistrationInfo* you must specify the registration name and the registration key. After you have done this MetaLib is unlocked and provides full functionality. It is recommended that you call the *SetRegistrationInfo* when you start your program.

In Visual Basic add the following code lines in the *Form_Load* procedure of your main form.

```
Private Sub Form_Load()
 Registration.SetRegistrationInfo "YOUR NAME", "YOUR KEY"
End Sub
```

Add a module to your VB project and add following line:

```
Public Registration As New MLRegistration
```

If you are using .NET add following line:

```
Public Registration As New METALIBLib.MLRegistration
```

In Visual C++ add the following code lines to the *InitInstance* function.

```
BOOL CMLYourApp::InitInstance()
{
      g_IMLRegistration->SetRegistrationInfo (L"YOUR NAME", L"YOUR KEY");
      return TRUE;
}
```

Deploying MetaLib

You only have to include the MetaLib.DLL file to your application. Make sure that the DLL will be registered on the target computer. (regsvr32 C:\MetaLibPath\Metalib.dll) No other files are needed.

Disclaimer of Warranty

Users of MetaLib must accept this disclaimer of warranty. If you do not accept this disclaimer, do not use MetaLib.

'METALIB IS SUPPLIED AS IS. THE AUTHOR DISCLAIMS ALL WARRANTIES OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES

OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DAMAGES WHATSOEVER INCLUDING DIRECT, INDIRECT, INCIDENTAL, CONSEQUENTIAL, LOSS OF BUSINESS PROFITS OR SPECIAL DAMAGES, WHICH MAY RESULT FROM THE USE OF METALIB EVEN IF THE AUTHOR HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.'

MetaStock™ is a registered trademark of EQUIS International. (http://www.equis.com) CompuTrac™ is a registered trademark of Dow Jones Telerate.

Contact

We provide support via email. You can email your questions, comments or bug reports to info@trading-tools.com. Normally all emails are answered throughout the 24-hour day.

Web: http://www.trading-tools.com

Special thanks to Elden Taylor for providing a Delphi example project and Thomas Pfluegl for the MetaLib-idea, for testing and for developing the MetaLib Tools Excel workbook ("MetaLib-Tools.xls").
This spreadsheet implements some basic features of MetaLib and is available at http://members.aon.at/tips/download.htm

Web: http://members.aon.at/tips/index.html