

# MBS Real Studio WinDragDrop Plugin Documentation

Christian Schmitz

May 15, 2012

## 0.1 Introduction

This is the PDF version of the documentation for the Real Studio Plug-in from Monkeybread Software Germany. Plugin part: MBS Real Studio WinDragDrop Plugin

## 0.2 Content

- 1 List of all topics 3
- 2 All items in this plugin 7
- 3 List of all classes 53

# Chapter 1

## List of Topics

• 2 Drag & Drop	7
– 2.1 class WindowsDropTargetMBS	7
* 2.1.1 AttachToControl(ctl as control, showDragImage as boolean = true) as integer	8
* 2.1.1 AttachToWindow(win as window, showDragImage as boolean = true) as integer	9
* 2.1.2 Handle as Integer	10
* 2.1.2 Helper as Integer	11
* 2.1.3 DragEnter(dataObject as WinDataObjectMBS, keystate as integer, x as integer, y as integer, byref effect as integer) as integer	11
* 2.1.3 DragLeave as integer	12
* 2.1.3 DragOver(keystate as integer, x as integer, y as integer, byref effect as integer) as integer	13
* 2.1.3 Drop(dataObject as WinDataObjectMBS, keystate as integer, x as integer, y as integer, byref effect as integer) as integer	15
* 2.1.4 DROPEFFECT_COPY = 1	16
* 2.1.4 DROPEFFECT_LINK = 4	16
* 2.1.4 DROPEFFECT_MOVE = 2	17
* 2.1.4 DROPEFFECT_NONE = 0	17
* 2.1.4 DROPEFFECT_SCROLL = & h80000000	17
* 2.1.4 E_INVALIDARG = & h80070057	17
* 2.1.4 E_OUTOFMEMORY = & h80000002	17
* 2.1.4 E_UNEXPECTED = & h8000FFFF	18
* 2.1.4 MK_CONTROL = 8	18
* 2.1.4 MK_LBUTTON = 1	18
* 2.1.4 MK_MBUTTON = & h10	18
* 2.1.4 MK_RBUTTON = 2	18
* 2.1.4 MK_SHIFT = 4	19

* 2.1.4 MK_XBUTTON1 = & h20	19
* 2.1.4 MK_XBUTTON2 = & h40	19
* 2.1.4 S_FALSE = 1	19
* 2.1.4 S_OK = 0	19
– 2.2 class WindowsDragSourceMBS	20
* 2.2.1 DoDragDrop(dataObject as WinDataObjectMBS, OKEffect as integer, byref Effect as integer) as integer	21
* 2.2.2 Handle as Integer	23
* 2.2.3 GiveFeedback(Effect as integer) as integer	23
* 2.2.3 QueryContinueDrag(EscapePressed as boolean, KeyState as integer) as integer	24
* 2.2.4 DRAGDROP_S_CANCEL = & H00040101	25
* 2.2.4 DRAGDROP_S_DROP = & h00040100	25
* 2.2.4 DRAGDROP_S_USEDEFAULTCURSORS = & h00040102	26
* 2.2.4 DROPEFFECT_COPY = 1	26
* 2.2.4 DROPEFFECT_LINK = 4	26
* 2.2.4 DROPEFFECT_MOVE = 2	27
* 2.2.4 DROPEFFECT_NONE = 0	27
* 2.2.4 DROPEFFECT_SCROLL = & h80000000	27
* 2.2.4 MK_CONTROL = 8	27
* 2.2.4 MK_LBUTTON = 1	27
* 2.2.4 MK_MBUTTON = & h10	28
* 2.2.4 MK_RBUTTON = 2	28
* 2.2.4 MK_SHIFT = 4	28
* 2.2.4 MK_XBUTTON1 = & h20	28
* 2.2.4 MK_XBUTTON2 = & h40	28
* 2.2.4 S_FALSE = 1	29
* 2.2.4 S_OK = 0	29
– 2.4 class WinDataObjectMBS	41
* 2.4.1 AddDragImage(pic as picture, width as integer, height as integer, x as integer, y as integer)	41
* 2.4.1 AddDragImage(pic as picture, width as integer, height as integer, x as integer, y as integer, ImageBackgroundColor as color)	42
* 2.4.1 AddFiles(files() as folderitem)	43
* 2.4.1 AddFiles(pathes() as string)	43
* 2.4.1 AddPicture(pic as picture)	43
* 2.4.1 AddRaw(format as integer, data as string)	43
* 2.4.1 AddText(text as string)	43
* 2.4.1 Constructor	44
* 2.4.1 Constructor(files() as folderitem)	44
* 2.4.1 Constructor(pic as picture)	44
* 2.4.1 Constructor(text as string)	45

* 2.4.1 GetFileContents(index as integer) as string	45
* 2.4.1 GetFileDescriptors as WindowsFileDescriptorMBS()	45
* 2.4.1 GetPaths as folderitem()	46
* 2.4.1 GetPathStrings as string()	46
* 2.4.1 GetPicture as picture	47
* 2.4.1 GetRaw(format as integer) as string	47
* 2.4.1 GetText as string	47
* 2.4.1 HasFileDescriptors as boolean	47
* 2.4.1 HasPaths as boolean	47
* 2.4.1 HasPicture as boolean	48
* 2.4.1 HasRaw(format as integer) as boolean	48
* 2.4.1 HasText as boolean	48
* 2.4.2 Handle as Integer	48
* 2.4.2 HelperHandle as Integer	49
* 2.4.3 CF_ BITMAP = 2	49
* 2.4.3 CF_ DIB = 8	49
* 2.4.3 CF_ DIBV5 = 17	49
* 2.4.3 CF_ DIF = 5	49
* 2.4.3 CF_ ENHMETAFILE = 14	50
* 2.4.3 CF_ HDROP = 15	50
* 2.4.3 CF_ LOCALE = 16	50
* 2.4.3 CF_ METAFILEPICT = 3	50
* 2.4.3 CF_ OEMTEXT = 7	50
* 2.4.3 CF_ PALETTE = 9	50
* 2.4.3 CF_ PENDATA = 10	51
* 2.4.3 CF_ RIFF = 11	51
* 2.4.3 CF_ SYLK = 4	51
* 2.4.3 CF_ TEXT = 1	51
* 2.4.3 CF_ TIFF = 6	51
* 2.4.3 CF_ UNICODETEXT = 13	51
* 2.4.3 CF_ WAVE = 12	52
– 2.3 class WindowsFileDescriptorMBS	29
* 2.3.1 ClassID as String	29
* 2.3.1 CreationTime as Double	30
* 2.3.1 FileAttributes as Integer	30
* 2.3.1 FileName as String	31
* 2.3.1 FileSize as Int64	31
* 2.3.1 Flags as Integer	32
* 2.3.1 IconHeight as Integer	32
* 2.3.1 IconWidth as Integer	32
* 2.3.1 Index as Integer	33

* 2.3.1 LastAccessTime as Double	33
* 2.3.1 LastWriteTime as Double	34
* 2.3.1 PointX as Integer	34
* 2.3.1 PointY as Integer	35
* 2.3.2 FD_ ACESSTIME = & h0010	35
* 2.3.2 FD_ ATTRIBUTES = 4	36
* 2.3.2 FD_ CLSID = 1	36
* 2.3.2 FD_ CREATETIME = 8	36
* 2.3.2 FD_ FILESIZE = & h0040	37
* 2.3.2 FD_ LINKUI = & h8000	37
* 2.3.2 FD_ PROGRESSUI = & h4000	38
* 2.3.2 FD_ SIZEPOINT = 2	38
* 2.3.2 FD_ WRITESTIME = & h0020	38
* 2.3.2 FILE_ ATTRIBUTE_ ARCHIVE = & h00000020	39
* 2.3.2 FILE_ ATTRIBUTE_ ATOMIC_ WRITE = & h00000200	39
* 2.3.2 FILE_ ATTRIBUTE_ COMPRESSED = & h00000800	39
* 2.3.2 FILE_ ATTRIBUTE_ DIRECTORY = & h00000010	39
* 2.3.2 FILE_ ATTRIBUTE_ HIDDEN = & h00000002	39
* 2.3.2 FILE_ ATTRIBUTE_ NORMAL = & h00000080	40
* 2.3.2 FILE_ ATTRIBUTE_ OFFLINE = & h00001000	40
* 2.3.2 FILE_ ATTRIBUTE_ READONLY = & h00000001	40
* 2.3.2 FILE_ ATTRIBUTE_ SYSTEM = & h00000004	40
* 2.3.2 FILE_ ATTRIBUTE_ TEMPORARY = & h00000100	40
* 2.3.2 FILE_ ATTRIBUTE_ XACTION_ WRITE = & h00000400	41

## Chapter 2

# Drag & Drop

### 2.1 class WindowsDropTargetMBS

`class WindowsDropTargetMBS`

Plugin Version: 10.5 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** The WindowsDropTargetMBS interface is one of the interfaces you implement to provide drag-and-drop operations in your application.

**Notes:**

It contains methods used in any application that can be a target for data during a drag-and-drop operation. A drop-target application is responsible for:

\* Determining the effect of the drop on the target application.

Incorporating any valid dropped data when the drop occurs.

Communicating target feedback to the source so the source application can provide appropriate visual feedback such as setting the cursor.

Implementing drag scrolling.

Registering and revoking its application windows as drop targets.

The WindowsDropTargetMBS class contains methods that handle all these responsibilities except registering and revoking the application window as a drop target, for which you must call the AttachToWindow functions.

### When To Implement

Implement the `WindowsDropTargetMBS` interface if you are developing an application that can act as a target for a drag-and-drop operation. The `WindowsDropTargetMBS` interface is associated with your application windows and is implemented on your window objects. Call the `AttachToWindow` function to register your window objects as drop targets.

### When To Use

You do not call the methods of `WindowsDropTargetMBS` directly. The `DoDragDrop` function calls the `WindowsDropTargetMBS` methods during the drag-and-drop operation.

For example, `DoDragDrop` calls `WindowsDropTargetMBS.DragEnter` when it detects the mouse has moved over a window that is registered as a drag target. After the mouse has entered a drag-target window, `DoDragDrop` calls `WindowsDropTargetMBS.DragOver` as the mouse moves through the window and calls `WindowsDropTargetMBS.DragLeave` if the mouse leaves the target window or if the user cancels or completes the drag-and-drop operation. `DoDragDrop` calls `WindowsDropTargetMBS.Drop` if the drop finally occurs.

To see the ghost picture of the drag, please register a `WindowsDragSourceMBS` for the same window.

While the drag classes compile for Web Edition, they run on the server, so they have no effect on the client browser!

## 2.1.1 Methods

**AttachToControl(ctl as control, showDragImage as boolean = true) as integer**

Plugin Version: 11.2 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** Attached the drop target to the control.

#### Notes:

This method unregisters any existing drop target on the control (including the one from Real Studio).

Registers the specified control as one that can be the target of an OLE drag-and-drop operation and specifies

the `WindowsDropTargetMBS` instance to use for drop operations.

`ctl`: The control that can be a target for an OLE drag-and-drop operation.

`showDragImage`: Whether we should support the drag image methods in the newer Windows versions.

This function returns `S_OK` on success. Other possible values include the following.

Return code	Description
<code>DRAGDROP_E_INVALIDHWND</code>	Invalid handle returned in the <code>hwnd</code> parameter.
<code>DRAGDROP_E_ALREADYREGISTERED</code>	The specified window has already been registered as a drop target.
<code>E_OUTOFMEMORY</code>	Insufficient memory for the operation.

If your application can accept dropped objects during OLE drag-and-drop operations, you must call the `AttachToControl` function. Do this whenever one of your application windows is available as a potential drop target, i.e., when the window appears unobscured on the screen.

`AttachToControl` must be called on the main thread of your application.

The `AttachToControl` function only registers one control at a time, so you must call it for each application control capable of accepting dropped objects. For each control, you need your own instance of the `WindowsDropTargetMBS` class.

As the mouse passes over unobscured portions of the target control during an OLE drag-and-drop operation, the `DoDragDrop` function calls the specified `WindowsDropTargetMBS.DragOver` method for the current control. When a drop operation actually occurs in a given control, the `DoDragDrop` function calls `WindowsDropTargetMBS.Drop`.

### **`AttachToWindow(win as window, showDragImage as boolean = true) as integer`**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** Attached the drop target to the window.

#### **Notes:**

This method unregisters any existing drop target on the window (including the one from Real Studio).

Registers the specified window as one that can be the target of an OLE drag-and-drop operation and specifies the `WindowsDropTargetMBS` instance to use for drop operations.

Win: The window that can be a target for an OLE drag-and-drop operation.

showDragImage: Whether we should support the drag image methods in the newer Windows versions.

This function returns S\_OK on success. Other possible values include the following.

Return code	Description
DRAGDROP_E_INVALIDHWND	Invalid handle returned in the hwnd parameter.
DRAGDROP_E_ALREADYREGISTERED	The specified window has already been registered as a drop target.
E_OUTOFMEMORY	Insufficient memory for the operation.

If your application can accept dropped objects during OLE drag-and-drop operations, you must call the AttachToWindow function. Do this whenever one of your application windows is available as a potential drop target, i.e., when the window appears unobscured on the screen.

AttachToWindow must be called on the main thread of your application.

The AttachToWindow function only registers one window at a time, so you must call it for each application window capable of accepting dropped objects. For each window, you need your own instance of the WindowsDropTargetMBS class.

As the mouse passes over unobscured portions of the target window during an OLE drag-and-drop operation, the DoDragDrop function calls the specified WindowsDropTargetMBS.DragOver method for the current window. When a drop operation actually occurs in a given window, the DoDragDrop function calls WindowsDropTargetMBS.Drop.

## 2.1.2 Properties

### Handle as Integer

Plugin Version: 10.5 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** The internal handle.  
**Notes:** (Read and Write property)

## Helper as Integer

Plugin Version: 10.5 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** The internal reference to the helper object.

**Notes:** (Read and Write property)

### 2.1.3 Events

**DragEnter**(dataObject as WinDataObjectMBS, keystate as integer, x as integer, y as integer, byref effect as integer) as integer

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** Indicates whether a drop can be accepted, and, if so, the effect of the drop.

**Notes:**

dataObject: This data object contains the data being transferred in the drag-and-drop operation. If the drop occurs, this data object will be incorporated into the target.

KeyState: The current state of the keyboard modifier keys on the keyboard. Possible values can be a combination of any of the flags MK\_ CONTROL, MK\_ SHIFT, MK\_ ALT, MK\_ BUTTON, MK\_ LBUTTON, MK\_ MBUTTON, and MK\_ RBUTTON.

x and y: A point containing the current cursor coordinates in screen coordinates.

effect: The value of the Effect parameter of the DoDragDrop function. On return, must contain one of the DROPEFFECT flags, which indicates what the result of the drop operation would be.

Return S\_ OK on success. Other possible values include the following.

Return code	Description
E_ UNEXPECTED	An unexpected error has occurred.
E_ INVALIDARG	The Effect parameter is NULL on input.
E_ OUTFOMEMORY	There was insufficient memory available for this operation.

You do not call DragEnter directly; instead the DoDragDrop function calls it to determine the effect of a drop the first time the user drags the mouse into the registered window of a drop target.

To implement `DragEnter`, you must determine whether the target can use the data in the source data object by checking three things:

- \* The format and medium specified by the data object
- The input value of `Effect`
- The state of the modifier keys

To check the format and medium, use the `WinDataObjectMBS` object.

On entry to `WindowsDropTargetMBS.DragEnter`, the `Effect` parameter is set to the effects given to the `OkEffect` parameter of the `DoDragDrop` function. The `WindowsDropTargetMBS.DragEnter` method must choose one of these effects or disable the drop.

The following modifier keys affect the result of the drop.

Key Combination	User-Visible Feedback	Drop Effect
CTRL + SHIFT	=	DROPEFFECT_LINK
CTRL	+	DROPEFFECT_COPY
No keys or SHIFT	None	DROPEFFECT_MOVE

On return, the method must write the effect, one of the `DROPEFFECT` flags, to the `Effect` parameter. `DoDragDrop` then takes this parameter and writes it to its `Effect` parameter. You communicate the effect of the drop back to the source through `DoDragDrop` in the `Effect` parameter. The `DoDragDrop` function then calls `WindowsDragSourceMBS.GiveFeedback` so that the source application can display the appropriate visual feedback to the user through the target window.

### DragLeave as integer

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** Removes target feedback and releases the data object.

#### Notes:

Return `S_OK` on success. Other possible values include the following.

Return code	Description
<code>E_OUTOFMEMORY</code>	There is insufficient memory available for this operation.

You do not call this method directly. The `DoDragDrop` function calls this method in either of the following cases:

\* When the user drags the cursor out of a given target window.  
When the user cancels the current drag-and-drop operation.

To implement `WindowsDropTargetMBS.DragLeave`, you must remove any target feedback that is currently displayed. You must also release any references you hold to the data transfer object.

**DragOver(keystate as integer, x as integer, y as integer, byref effect as integer) as integer**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** Provides target feedback to the user and communicates the drop's effect to the `DoDragDrop` function so it can communicate the effect of the drop back to the source.

**Notes:**

**KeyState:** The current state of the keyboard modifier keys on the keyboard. Valid values can be a combination of any of the flags `MK_CONTROL`, `MK_SHIFT`, `MK_ALT`, `MK_BUTTON`, `MK_LBUTTON`, `MK_MBUTTON`, and `MK_RBUTTON`.

**x and y:** The point containing the current cursor coordinates in screen coordinates.

**Effect:** On input, pointer to the value of the `Effect` parameter of the `DoDragDrop` function. On return, must contain one of the `DROPEFFECT` flags, which indicates what the result of the drop operation would be.

Return `S_OK` on success. Other possible values include the following.

Return code	Description
<code>E_UNEXPECTED</code>	An unexpected error has occurred.
<code>E_INVALIDARG</code>	The <code>Effect</code> value is not valid.
<code>E_OUTOFMEMORY</code>	There was insufficient memory available for this operation.

You do not call `DragOver` directly. The `DoDragDrop` function calls this method each time the user moves the mouse across a given target window. `DoDragDrop` exits the loop if the drag-and-drop operation is canceled, if the user drags the mouse out of the target window, or if the drop is completed.

In implementing `WindowsDropTargetMBS.DragOver`, you must provide features similar to those in `WindowsDropTargetMBS.DragEnter`. You must determine the effect of dropping the data on the target by examining the `FORMATETC` defining the data object's formats and medium, along with the state of the modifier keys. The mouse position may also play a role in determining the effect of a drop. The following modifier keys affect the result of the drop.

Key Combination	User-Visible Feedback	Drop Effect
CTRL + SHIFT	=	DROPEFFECT_LINK
CTRL	+	DROPEFFECT_COPY
No keys or SHIFT	None	DROPEFFECT_MOVE

You communicate the effect of the drop back to the source through `DoDragDrop` in `Effect`. The `DoDragDrop` function then calls `WindowsDragSourceMBS.GiveFeedback` so the source application can display the appropriate visual feedback to the user.

On entry to `WindowsDropTargetMBS.DragOver`, the `Effect` parameter must be set to the allowed effects passed to the `OkEffect` parameter of the `DoDragDrop` function. The `WindowsDropTargetMBS.DragOver` method must be able to choose one of these effects or disable the drop.

Upon return, `Effect` is set to one of the `DROPEFFECT` flags. This value is then passed to the `Effect` parameter of `DoDragDrop`. Reasonable values are `DROPEFFECT_COPY` to copy the dragged data to the target, `DROPEFFECT_LINK` to create a link to the source data, or `DROPEFFECT_MOVE` to allow the dragged data to be permanently moved from the source application to the target.

You may also wish to provide appropriate visual feedback in the target window. There may be some target feedback already displayed from a previous call to `WindowsDropTargetMBS.DragOver` or from the initial `WindowsDropTargetMBS.DragEnter`. If this feedback is no longer appropriate, you should remove it.

For efficiency reasons, a data object is not passed in `WindowsDropTargetMBS.DragOver`. The data object passed in the most recent call to `WindowsDropTargetMBS.DragEnter` is available and can be used.

When `WindowsDropTargetMBS.DragOver` has completed its operation, the `DoDragDrop` function calls `WindowsDragSourceMBS.GiveFeedback` so the source application can display the appropriate visual feedback to the user.

### Notes to Implementers

This function is called frequently during the `DoDragDrop` loop so it makes sense to optimize your implementation of the `DragOver` method as much as possible.

**Drop**(dataObject as WinDataObjectMBS, keystate as integer, x as integer, y as integer, byref effect as integer) as integer

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** Incorporates the source data into the target window, removes target feedback, and releases the data object.

**Notes:**

dataObject: The data object being transferred in the drag-and-drop operation.

KeyState: The current state of the keyboard modifier keys on the keyboard. Possible values can be a combination of any of the flags MK\_ CONTROL, MK\_ SHIFT, MK\_ ALT, MK\_ BUTTON, MK\_ LBUTTON, MK\_ MBUTTON, and MK\_ RBUTTON.

x and y: The point containing the current cursor coordinates in screen coordinates.

Effect: On input, the value of the Effect parameter of the DoDragDrop function. On return, must contain one of the DROPEFFECT flags, which indicates what the result of the drop operation would be.

Return S\_ OK on success. Other possible values include the following.

Return code	Description
E_ UNEXPECTED	An unexpected error has occurred.
E_ INVALIDARG	The pdwEffect parameter is not valid.
E_ OUTOFMEMORY	There is insufficient memory available for this operation.

You do not call this method directly. The DoDragDrop function calls this method when the user completes the drag-and-drop operation.

In implementing Drop, you must incorporate the data object into the target. Use the formats available in WinDataObjectMBS, available through dataObject, along with the current state of the modifier keys to determine how the data is to be incorporated, such as linking or embedding.

In addition to incorporating the data, you must also clean up as you do in the WindowsDropTargetMBS.DragLeave method:

Remove any target feedback that is currently displayed.

Release any references to the data object.

You also pass the effect of this operation back to the source application through DoDragDrop, so the source application can clean up after the drag-and-drop operation is complete:

Remove any source feedback that is being displayed.

Make any necessary changes to the data, such as removing the data if the operation was a move.

### 2.1.4 Constants

#### **DROPEFFECT\_COPY = 1**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the drop effect constants.

**Notes:**

Drop results in a copy. The original data is untouched by the drag source.

Your application should always mask drop effect constants to ensure compatibility with future implementations. Presently, only some of the bit positions in a drop effect value have meaning. In the future, more interpretations for the bits will be added. Drag sources and drop targets should carefully mask these values appropriately before comparing. They should never compare a drop effect value against, say, DROPEFFECT\_COPY by doing the following:

```
if DropEffect = DROPEFFECT_COPY then
```

Instead, the application should always mask for the value or values being sought as using one of the following techniques:

```
if bitwiseAnd(DropEffect, DROPEFFECT_COPY) = DROPEFFECT_COPY then
```

This allows for the definition of new drop effects, while preserving backwards compatibility with existing code.

#### **DROPEFFECT\_LINK = 4**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the drop effect constants.

**Notes:** Drag source should create a link to the original data.

**DROPEFFECT\_MOVE = 2**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the drop effect constants.

**Notes:** Drag source should remove the data.

**DROPEFFECT\_NONE = 0**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the drop effect constants.

**Notes:** Drop target cannot accept the data.

**DROPEFFECT\_SCROLL = & h80000000**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the drop effect constants.

**Notes:** Scrolling is about to start or is currently occurring in the target. This value is used in addition to the other values.

**E\_INVALIDARG = & h80070057**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the OLE error codes.

**Notes:** An invalid argument was passed.

**E\_OUTOFMEMORY = & h80000002**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the OLE error codes.

**Notes:** There was insufficient memory available for this operation.

**E\_ UNEXPECTED = & h800FFFF**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the OLE error codes.

**Notes:** An unexpected error has occurred.

**MK\_ CONTROL = 8**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the key state constants.

**Notes:** The CTRL key is down.

**MK\_ LBUTTON = 1**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the key state constants.

**Notes:** The left mouse button is down.

**MK\_ MBUTTON = & h10**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the key state constants.

**Notes:** The middle mouse button is down.

**MK\_ RBUTTON = 2**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the key state constants.

**Notes:** The right mouse button is down.

**MK\_ SHIFT = 4**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the key state constants.

**Notes:** The SHIFT key is down.

**MK\_ XBUTTON1 = & h20**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the key state constants.

**Notes:** The first X button is down.

**MK\_ XBUTTON2 = & h40**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the key state constants.

**Notes:** The second X button is down.

**S\_ FALSE = 1**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the OLE error codes.

**S\_ OK = 0**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the OLE error codes.

## 2.2 class WindowsDragSourceMBS

### class WindowsDragSourceMBS

Plugin Version: 10.5 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** The class for Drag and Drop on Windows to create a drag source.

#### Notes:

The WindowsDragSourceMBS class is one of the class you implement to provide drag-and-drop operations in your application. It contains methods used in any application used as a data source in a drag-and-drop operation. The data source application in a drag-and-drop operation is responsible for:

- \* Determining the data being dragged based on the user's selection.
- Initiating the drag-and-drop operation based on the user's mouse actions.
- Generating some of the visual feedback during the drag-and-drop operation, such as setting the cursor and highlighting the data selected for the drag-and-drop operation.
- Canceling or completing the drag-and-drop operation based on the user's mouse actions.
- Performing any action on the original data caused by the drop operation, such as deleting the data on a drag move.

WindowsDragSourceMBS contains the events for generating visual feedback to the end user and for canceling or completing the drag-and-drop operation.

#### When To Implement

Implement WindowsDragSourceMBS if you are developing a container or server application that can act as a data source for a drag-and-drop operation. The WindowsDragSourceMBS interface is only required during the drag-and-drop operation.

If you implement the WindowsDragSourceMBS class, you must also implement the DataObjectMBS class on the same object to represent the data being transferred.

You can use the same implementation of DataObjectMBS for drag-and-drop data as for the data object offered to the clipboard. After you have implemented clipboard operations in your application, you can add drag-and-drop operations with only a little extra work.

#### When To Use

You do not usually call the `WindowsDragSourceMBS` methods directly. Instead, your data source calls the `DoDragDrop` function when it detects that the user has initiated a drag-and-drop operation. Then, `DoDragDrop` calls the `WindowsDragSourceMBS` methods during the drag-and-drop operation.

For example, `DoDragDrop` calls `WindowsDragSourceMBS.GiveFeedback` when you need to change the cursor shape or when you need to provide some other visual feedback. `DoDragDrop` calls `WindowsDragSourceMBS.QueryContinueDrag` when there is a change in the mouse button state to determine if the drag-and-drop operation was canceled or completed.

While the drag classes compile for Web Edition, they run on the server, so they have no effect on the client browser!

### 2.2.1 Methods

**DoDragDrop**(dataObject as WinDataObjectMBS, OKEffect as integer, byref Effect as integer) as integer

Plugin Version: 10.5 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** Carries out an OLE drag and drop operation.

**Notes:**

Parameters:

**dataObject:** The `WinDataObjectMBS` data object that contains the data being dragged.

**OKEffects:** Effects the source allows in the OLE drag-and-drop operation. Most significant is whether it permits a move. The `OKEffect` and `Effect` parameters obtain values from the `DROPEFFECT*` constants.

**Effect:** Pointer to a value that indicates how the OLE drag-and-drop operation affected the source data. The `pdwEffect` parameter is set only if the operation is not canceled.

This function returns `S_OK` on success. Other possible values include the following.

Return code	Description
<code>DRAGDROP_S_DROP</code>	The OLE drag-and-drop operation was successful.
<code>DRAGDROP_S_CANCEL</code>	The OLE drag-and-drop operation was canceled.
<code>E_UNSPEC</code>	Unexpected error occurred.

## Remarks

If you are developing an application that can act as a data source for an OLE drag-and-drop operation, you must call `DoDragDrop` when you detect that the user has started an OLE drag-and-drop operation.

The `DoDragDrop` function enters a loop in which it calls various methods in the `WindowsDragSourceMBS` and `WindowsDropTargetMBS` interfaces. (For a successful drag-and-drop operation, the application acting as the data source must also implement `WindowsDragSourceMBS`, while the target application must implement `WindowsDropTargetMBS`.)

The `DoDragDrop` function determines the window under the current cursor location. It then checks to see if this window is a valid drop target.

If the window is a valid drop target, `DoDragDrop` calls `WindowsDropTargetMBS.DragEnter`. This method supplies an effect code indicating what would happen if the drop actually occurred. For a list of valid drop effects, see the `DROPEFFECT*` constants.

`DoDragDrop` calls `WindowsDragSourceMBS.GiveFeedback` with the effect code so that the drop source interface can provide appropriate visual feedback to the user.

`DoDragDrop` tracks mouse cursor movements and changes in the keyboard or mouse button state.

If the user moves out of a window, `DoDragDrop` calls `WindowsDropTargetMBS.DragLeave`.

If the mouse enters another window, `DoDragDrop` determines if that window is a valid drop target and then calls `WindowsDropTargetMBS.DragEnter` for that window.

If the mouse moves but stays within the same window, `DoDragDrop` calls `WindowsDropTargetMBS.DragOver`.

If there is a change in the keyboard or mouse button state, `DoDragDrop` calls `WindowsDragSourceMBS.QueryContinueDrag` and determines whether to continue the drag, to drop the data, or to cancel the operation based on the return value.

If the return value is `S_OK`, `DoDragDrop` first calls `WindowsDropTargetMBS.DragOver` to continue the operation. This method returns a new effect value and `DoDragDrop` then calls `WindowsDragSourceMBS.Give-`

Feedback with the new effect so appropriate visual feedback can be set. For a list of valid drop effects, see the DROPEFFECT constants. `WindowsDropTargetMBS.DragOver` and `WindowsDragSourceMBS.GiveFeedback` are paired so that as the mouse moves across the drop target, the user is given the most up-to-date feedback on the mouse's position.

If the return value is `DRAGDROP_S_DROP`, `DoDragDrop` calls `WindowsDropTargetMBS.Drop`. The `DoDragDrop` function returns the last effect code to the source, so the source application can perform the appropriate operation on the source data, for example, cut the data if the operation was a move.

If the return value is `DRAGDROP_S_CANCEL`, the `DoDragDrop` function calls `WindowsDropTargetMBS.DragLeave`.

## 2.2.2 Properties

### Handle as Integer

Plugin Version: 10.5 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** The internal handle.  
**Notes:** (Read and Write property)

## 2.2.3 Events

### GiveFeedback(Effect as integer) as integer

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** Enables a source application to give visual feedback to the end user during a drag-and-drop operation by providing the `DoDragDrop` function with an enumeration value specifying the visual effect.

**Notes:**

**Effect:** The drop effect value returned by the most recent call to `WindowsDropTargetMBS.DragEnter`, `WindowsDropTargetMBS.DragOver`, or `WindowsDropTargetMBS.DragLeave`.

Return `S_OK` on success. Return `DRAGDROP_S_USEDEFAULTCURSORS` to indicate successful completion of the method, and requests OLE to update the cursor using the OLE-provided default cursors.

When your application detects that the user has started a drag-and-drop operation, it should call the `Do`

DragDrop function. DoDragDrop enters a loop, calling WindowsDropTargetMBS.DragEnter when the mouse first enters a drop target window, WindowsDropTargetMBS.DragOver when the mouse changes its position within the target window, and WindowsDropTargetMBS.DragLeave when the mouse leaves the target window.

For every call to either WindowsDropTargetMBS.DragEnter or WindowsDropTargetMBS.DragOver, DoDragDrop calls WindowsDropTargetMBS.GiveFeedback, passing it the drop effect value returned from the drop target call.

DoDragDrop calls WindowsDropTargetMBS.DragLeave when the mouse has left the target window. Then, DoDragDrop calls WindowsDropTargetMBS.GiveFeedback and passes the DROPEFFECT\_NONE value in the dwEffect parameter.

The Effect parameter can include DROPEFFECT\_SCROLL, indicating that the source should put up the drag-scrolling variation of the appropriate pointer.

#### Notes to Implementers

This function is called frequently during the DoDragDrop loop, so you can gain performance advantages if you optimize your implementation as much as possible.

GiveFeedback is responsible for changing the cursor shape or for changing the highlighted source based on the value of the dwEffect parameter. If you are using default cursors, you can return DRAGDROP\_USEDEFAULTCURSORS, which causes OLE to update the cursor for you, using its defaults.

#### **QueryContinueDrag(EscapePressed as boolean, KeyState as integer) as integer**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** Determines whether a drag-and-drop operation should be continued, canceled, or completed. You do not call this method directly. **Notes:**

The OLE DoDragDrop function calls this method during a drag-and-drop operation.

EscapePressed: Indicates whether the Esc key has been pressed since the previous call to QueryContinueDrag or to DoDragDrop if this is the first call to QueryContinueDrag. A true value indicates the end user has pressed the escape key; a false value indicates it has not been pressed.

KeyState: The current state of the keyboard modifier keys on the keyboard. Possible values can be a combination of any of the flags MK\_ CONTROL, MK\_ SHIFT, MK\_ ALT, MK\_ BUTTON, MK\_ LBUTTON, MK\_ MBUTTON, and MK\_ RBUTTON.

This event can return the following values.

Return code	Description
S_ OK	The drag operation should continue. This result occurs if no errors are detected, the mouse button starting the drag-and-drop operation has not been released, and the Esc key has not been detected.
DRAGDROP_ S_ DROP	The drop operation should occur completing the drag operation. This result occurs if KeyState indicates that the key that started the drag-and-drop operation has been released.
DRAGDROP_ S_ CANCEL	The drag operation should be canceled with no drop operation occurring. This result occurs if EscapePressed is true, indicating the Esc key has been pressed.

The DoDragDrop function calls QueryContinueDrag whenever it detects a change in the keyboard or mouse button state during a drag-and-drop operation. QueryContinueDrag must determine whether the drag-and-drop operation should be continued, canceled, or completed based on the contents of the parameters KeyState and EscapePressed.

#### 2.2.4 Constants

**DRAGDROP\_ S\_ CANCEL = & H00040101**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the OLE error codes.

**Notes:** Drag and Drop was cancelled.

**DRAGDROP\_ S\_ DROP = & h00040100**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the OLE error codes.

**Notes:** Do the drop operation.

**DRAGDROP\_S\_USEDEFAULTCURSORS = & h00040102**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the OLE error codes.

**DROPEFFECT\_COPY = 1**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the drop effect constants.

**Notes:**

Drop results in a copy. The original data is untouched by the drag source.

Your application should always mask drop effect constants to ensure compatibility with future implementations. Presently, only some of the bit positions in a drop effect value have meaning. In the future, more interpretations for the bits will be added. Drag sources and drop targets should carefully mask these values appropriately before comparing. They should never compare a drop effect value against, say, `DROPEFFECT_COPY` by doing the following:

```
if DropEffect = DROPEFFECT_COPY then
```

Instead, the application should always mask for the value or values being sought as using one of the following techniques:

```
if bitwiseAnd(DropEffect, DROPEFFECT_COPY) = DROPEFFECT_COPY then
```

This allows for the definition of new drop effects, while preserving backwards compatibility with existing code.

**DROPEFFECT\_LINK = 4**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the drop effect constants.

**Notes:** Drag source should create a link to the original data.

**DROPEFFECT\_MOVE = 2**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the drop effect constants.

**Notes:** Drag source should remove the data.

**DROPEFFECT\_NONE = 0**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the drop effect constants.

**Notes:** Drop target cannot accept the data.

**DROPEFFECT\_SCROLL = & h80000000**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the drop effect constants.

**Notes:** Scrolling is about to start or is currently occurring in the target. This value is used in addition to the other values.

**MK\_CONTROL = 8**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the key state constants.

**Notes:** The CTRL key is down.

**MK\_LBUTTON = 1**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the key state constants.

**Notes:** The left mouse button is down.

**MK\_ MBUTTON = & h10**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the key state constants.

**Notes:** The middle mouse button is down.

**MK\_ RBUTTON = 2**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the key state constants.

**Notes:** The right mouse button is down.

**MK\_ SHIFT = 4**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the key state constants.

**Notes:** The SHIFT key is down.

**MK\_ XBUTTON1 = & h20**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the key state constants.

**Notes:** The first X button is down.

**MK\_ XBUTTON2 = & h40**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the key state constants.

**Notes:** The second X button is down.

**S\_ FALSE = 1**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the OLE error codes.

**S\_ OK = 0**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the OLE error codes.

## 2.3 class WindowsFileDescriptorMBS

**class WindowsFileDescriptorMBS**

Plugin Version: 11.2 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** The class for a file description.s

### 2.3.1 Properties

**ClassID as String**

Plugin Version: 11.2 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** The file type identifier.

**Example:**

```
dim d as WindowsFileDescriptorMBS // your file description
```

```
if BitwiseAnd(d.Flags, d.FD_ CLSID) <>0 then  
MsgBox d.ClassID  
end if
```

**Notes:**

Only valid if FD\_ CLSID is set in the flags.  
(Read and Write property)

### CreationTime as Double

Plugin Version: 11.2 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** The time of file creation.

#### Example:

```
dim d as WindowsFileDescriptorMBS // your file description

if BitwiseAnd(d.Flags, d.FD_ CREATETIME) <>0 then
dim da as new date
da.TotalSeconds = d.CreationTime
MsgBox da.LongDate
end if
```

#### Notes:

Only valid if FD\_ CREATETIME is set in the flags.  
(Read and Write property)

### FileAttributes as Integer

Plugin Version: 11.2 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** File attribute flags.

#### Example:

```
dim d as WindowsFileDescriptorMBS // your file description

if BitwiseAnd(d.Flags, d.FD_ ATTRIBUTES) <>0 then
if BitwiseAnd(d.FileAttributes, d.FILE_ ATTRIBUTE_ TEMPORARY) = d.FILE_ ATTRIBUTE_ TEM-
PORARY then
MsgBox "temp file"
else
MsgBox "no temp file"
end if
end if
```

**Notes:**

This will be a combination of the FILE\_ ATTRIBUTE\_ \* constants.  
Only valid if FD\_ ATTRIBUTES is set in the flags.  
(Read and Write property)

**FileName as String**

Plugin Version: 11.2 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** The string that contains the name of the file.

**Example:**

```
dim d as WindowsFileDescriptorMBS // your file description
```

```
MsgBox d.FileName
```

**Notes:** (Read and Write property)

**FileSize as Int64**

Plugin Version: 11.2 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** The file size, in bytes.

**Example:**

```
dim d as WindowsFileDescriptorMBS // your file description
```

```
if BitwiseAnd(d.Flags, d.FD_ FILESIZE) <>0 then  
MsgBox str(d.FileSize)  
end if
```

**Notes:**

Only valid if FD\_ FILESIZE is set in flags.  
(Read and Write property)

**Flags as Integer**

Plugin Version: 11.2 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** An array of flags that indicate which of the other structure members contain valid data.

**Notes:**

A combination of the FD\_ \* constants.  
(Read and Write property)

**IconHeight as Integer**

Plugin Version: 11.2 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** The height of the file icon.

**Example:**

```
dim d as WindowsFileDescriptorMBS // your file description

if BitwiseAnd(d.Flags, d.FD_ SIZEPOINT) <>0 then
  MsgBox "file object at "+strR(d.Pointx)+" / "+str(d.Pointy)+" with size "+str(d.IconWidth)+" / "+str(d.IconHeight)
end if
```

**Notes:**

Only valid if FD\_ SIZEPOINT is set in the flags.  
(Read and Write property)

**IconWidth as Integer**

Plugin Version: 11.2 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** The width of the file icon.

**Example:**

```
dim d as WindowsFileDescriptorMBS // your file description

if BitwiseAnd(d.Flags, d.FD_ SIZEPOINT) <>0 then
  MsgBox "file object at "+strR(d.Pointx)+" / "+str(d.Pointy)+" with size "+str(d.IconWidth)+" / "+str(d.IconHeight)
end if
```

**Notes:**

Only valid if FD\_ SIZEPOINT is set in the flags.  
(Read and Write property)

**Index as Integer**

Plugin Version: 11.2 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** The index of the file descriptor.

**Notes:**

Use this entry for GetFileContent call.  
(Read and Write property)

**LastAccessTime as Double**

Plugin Version: 11.2 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** The time that the file was last accessed.

**Example:**

```
dim d as WindowsFileDescriptorMBS // your file description

if BitwiseAnd(d.Flags, d.FD_ ACESSTIME) <>0 then
  dim da as new date
  da.TotalSeconds = d.LastAccessTime
  MsgBox da.LongDate
end if
```

**Notes:**

Only valid if FD\_ ACESSTIME is set in the flags.  
(Read and Write property)

**LastWriteTime as Double**

Plugin Version: 11.2 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** The time of the last write operation.

**Example:**

```
dim d as WindowsFileDescriptorMBS // your file description
```

```
if BitwiseAnd(d.Flags, d.FD_ WRITESTIME) <>0 then
dim da as new date
da.TotalSeconds = d.LastWriteTime
MsgBox da.LongDate
end if
```

**Notes:**

Only valid if FD\_ WRITESTIME is set in the flags.  
(Read and Write property)

**PointX as Integer**

Plugin Version: 11.2 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** The x screen coordinate of the file object.

**Example:**

```
dim d as WindowsFileDescriptorMBS // your file description
```

```
if BitwiseAnd(d.Flags, d.FD_ SIZEPOINT) <>0 then
MsgBox "file object at "+str(d.Pointx)+" / "+str(d.Pointy)+" with size "+str(d.IconWidth)+" / "+str(d.IconHeight)
end if
```

**Notes:**

Only valid if FD\_ SIZEPOINT is set in the flags.  
(Read and Write property)

**PointY as Integer**

Plugin Version: 11.2 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** The y screen coordinate of the file object.

**Example:**

```
dim d as WindowsFileDescriptorMBS // your file description

if BitwiseAnd(d.Flags, d.FD_ SIZEPOINT) <>0 then
MsgBox "file object at "+str(d.Pointx)+" / "+str(d.Pointy)+" with size "+str(d.IconWidth)+" / "+str(d.IconHeight)
end if
```

**Notes:**

Only valid if FD\_ SIZEPOINT is set in the flags.  
(Read and Write property)

**2.3.2 Constants**

**FD\_ ACESSTIME = & h0010**

Plugin Version: 11.2 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the flag constants.

**Example:**

```
dim d as WindowsFileDescriptorMBS // your file description

if BitwiseAnd(d.Flags, d.FD_ ACESSTIME) <>0 then
dim da as new date
da.TotalSeconds = d.LastAccessTime
MsgBox da.LongDate
end if
```

**Notes:** The LastAccessTime member is valid.

**FD\_ ATTRIBUTES = 4**

Plugin Version: 11.2 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the flag constants.

**Example:**

```
dim d as WindowsFileDescriptorMBS // your file description

if BitwiseAnd(d.Flags, d.FD_ ATTRIBUTES) <>0 then
if BitwiseAnd(d.FileAttributes, d.FILE_ ATTRIBUTE_ TEMPORARY)<>0 then
MsgBox "temp file"
else
MsgBox "no temp file"
end if
end if
```

**Notes:** The FileAttributes member is valid.

**FD\_ CLSID = 1**

Plugin Version: 11.2 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the flag constants.

**Example:**

```
dim d as WindowsFileDescriptorMBS // your file description

if BitwiseAnd(d.Flags, d.FD_ CLSID) <>0 then
MsgBox d.ClassID
end if
```

**Notes:** The ClassID member is valid.

**FD\_ CREATETIME = 8**

Plugin Version: 11.2 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the flag constants.

**Example:**

```

dim d as WindowsFileDescriptorMBS // your file description

if BitwiseAnd(d.Flags, d.FD_ CREATETIME) <>0 then
dim da as new date
da.TotalSeconds = d.CreationTime
MsgBox da.LongDate
end if

```

**Notes:** The CreationTime member is valid.

**FD\_ FILESIZE = & h0040**

Plugin Version: 11.2 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the flag constants.

**Example:**

```

dim d as WindowsFileDescriptorMBS // your file description

if BitwiseAnd(d.Flags, d.FD_ FILESIZE) <>0 then
MsgBox str(d.FileSize)
end if

```

**Notes:** Whether the FileSize member is valid.

**FD\_ LINKUI = & h8000**

Plugin Version: 11.2 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the flag constants.

**Notes:** Treat the operation as a shortcut.

**FD\_PROGRESSUI = & h4000**

Plugin Version: 11.2 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the flag constants.

**Notes:** A progress indicator is shown with drag-and-drop operations.

**FD\_SIZEPOINT = 2**

Plugin Version: 11.2 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the flag constants.

**Example:**

```
dim d as WindowsFileDescriptorMBS // your file description

if BitwiseAnd(d.Flags, d.FD_SIZEPOINT) <>0 then
  MsgBox "file object at "+str(d.Pointx)+" / "+str(d.Pointy)+" with size "+str(d.IconWidth)+" / "+str(d.IconHeight)
end if
```

**Notes:** The Icon\* and point\* members are valid.

**FD\_WRITEESTIME = & h0020**

Plugin Version: 11.2 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the flag constants.

**Example:**

```
dim d as WindowsFileDescriptorMBS // your file description

if BitwiseAnd(d.Flags, d.FD_WRITEESTIME) <>0 then
  dim da as new date
  da.TotalSeconds = d.LastWriteTime
  MsgBox da.LongDate
end if
```

**Notes:** Whether the LastWriteTime property is valid.

**FILE\_ATTRIBUTE\_ARCHIVE = & h00000020**

Plugin Version: 11.2 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the file attribute constants.

**Notes:** A file or directory that is an archive file or directory. Applications typically use this attribute to mark files for backup or removal .

**FILE\_ATTRIBUTE\_ATOMIC\_WRITE = & h00000200**

Plugin Version: 11.2 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the file attribute constants.

**FILE\_ATTRIBUTE\_COMPRESSED = & h00000800**

Plugin Version: 11.2 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the file attribute constants.

**Notes:** A file or directory that is compressed. For a file, all of the data in the file is compressed. For a directory, compression is the default for newly created files and subdirectories.

**FILE\_ATTRIBUTE\_DIRECTORY = & h00000010**

Plugin Version: 11.2 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the file attribute constants.

**Notes:** The handle that identifies a directory.

**FILE\_ATTRIBUTE\_HIDDEN = & h00000002**

Plugin Version: 11.2 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the file attribute constants.

**Notes:** The file or directory is hidden. It is not included in an ordinary directory listing.

**FILE\_ATTRIBUTE\_NORMAL = & h00000080**

Plugin Version: 11.2 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the file attribute constants.

**Notes:** A file that does not have other attributes set. This attribute is valid only when used alone.

**FILE\_ATTRIBUTE\_OFFLINE = & h00001000**

Plugin Version: 11.2 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the file attribute constants.

**Notes:** The data of a file is not available immediately. This attribute indicates that the file data is physically moved to offline storage. This attribute is used by Remote Storage, which is the hierarchical storage management software. Applications should not arbitrarily change this attribute.

**FILE\_ATTRIBUTE\_READONLY = & h00000001**

Plugin Version: 11.2 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the file attribute constants.

**Notes:** A file that is read-only. Applications can read the file, but cannot write to it or delete it.

**FILE\_ATTRIBUTE\_SYSTEM = & h00000004**

Plugin Version: 11.2 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the file attribute constants.

**Notes:** A file or directory that the operating system uses a part of, or uses exclusively.

**FILE\_ATTRIBUTE\_TEMPORARY = & h00000100**

Plugin Version: 11.2 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the file attribute constants.

**Notes:** A file that is being used for temporary storage. File systems avoid writing data back to mass storage if sufficient cache memory is available, because typically, an application deletes a temporary file after the handle is closed. In that scenario, the system can entirely avoid writing the data. Otherwise, the data is written after the handle is closed.

**FILE\_ATTRIBUTE\_XACTION\_WRITE = & h00000400**

Plugin Version: 11.2 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the file attribute constants.

## 2.4 class WinDataObjectMBS

**class WinDataObjectMBS**

Plugin Version: 10.5 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** The class for a data object for Drag and Drop.

**Example:**

```
// take some picture
dim p as Picture = LogoMBS(500)

// create data object
dim w as new WinDataObjectMBS(p)
```

**Notes:** While the drag classes compile for Web Edition, they run on the server, so they have no effect on the client browser!

### 2.4.1 Methods

**AddDragImage(pic as picture, width as integer, height as integer, x as integer, y as integer)**

Plugin Version: 11.0 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** Adds a drag image to the data object.

**Notes:**

pic: the picture to use.

width and height: The size of the picture.

x and y: The location of the cursor within the drag image. The point should contain the offset from the upper-left corner of the drag image to the location of the cursor.

Requires Windows 2000 Professional with SP3, Windows XP.  
On success the HelperHandle property is not zero.

Turn off antialiasing when drawing text. Otherwise, artifacts could occur at the edges, between the text color and the color key.

This function takes the picture (and it's mask) and turns it in a nice drag picture. This includes applying the mask and passing black for the background color. Dark colors which should be transparent will be made lighter.

See also:

- 2.4.1 AddDragImage(pic as picture, width as integer, height as integer, x as integer, y as integer, ImageBackgroundColor as color) 42

**AddDragImage(pic as picture, width as integer, height as integer, x as integer, y as integer, ImageBackgroundColor as color)**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** Adds a drag image to the data object.

**Notes:**

pic: the picture to use.

width and height: The size of the picture.

x and y: The location of the cursor within the drag image. The point should contain the offset from the upper-left corner of the drag image to the location of the cursor.

ImageBackgroundColor: The color used by the control to fill the background of the drag image.

Requires Windows 2000 Professional with SP3, Windows XP.  
On success the HelperHandle property is not zero.

Turn off antialiasing when drawing text. Otherwise, artifacts could occur at the edges, between the text color and the color key.

See also:

- 2.4.1 AddDragImage(pic as picture, width as integer, height as integer, x as integer, y as integer) 41

**AddFiles(files() as folderitem)**

Plugin Version: 10.5 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** Adds file references in the format the explorer can understand them.

See also:

- 2.4.1 AddFiles(pathes() as string) 43

**AddFiles(pathes() as string)**

Plugin Version: 11.0 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** Adds file pathes in the format the explorer can understand them.

**Notes:** Folder pathes should have no backslash on the end.

See also:

- 2.4.1 AddFiles(files() as folderitem) 43

**AddPicture(pic as picture)**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** Adds a picture to the data object.

**AddRaw(format as integer, data as string)**

Plugin Version: 10.5 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** Adds raw data to the data object.

**Notes:** Depending of the format you may need to add chr(0) on the end.

**AddText(text as string)**

Plugin Version: 10.5 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** Adds the text to the data object.

## Constructor

Plugin Version: 10.5 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** Creates an empty data object.

See also:

- 2.4.1 Constructor(files() as folderitem) 44
- 2.4.1 Constructor(pic as picture) 44
- 2.4.1 Constructor(text as string) 45

## Constructor(files() as folderitem)

Plugin Version: 10.5 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** Creates a data object and adds the files.

See also:

- 2.4.1 Constructor 44
- 2.4.1 Constructor(pic as picture) 44
- 2.4.1 Constructor(text as string) 45

## Constructor(pic as picture)

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** Creates a new data object and adds the given picture.

**Example:**

```
// take some picture
dim p as Picture = LogoMBS(500)

// create data object
dim w as new WinDataObjectMBS(p)
```

See also:

- 2.4.1 Constructor 44

## 2.4. CLASS WINDATAOBJECTMBS 45

- 2.4.1 Constructor(files() as folderitem) 44
- 2.4.1 Constructor(text as string) 45

### Constructor(text as string)

Plugin Version: 10.5 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** Creates a new data object and adds the given text.

#### Example:

```
// create data object with text  
dim w as new WinDataObjectMBS("Hello World")
```

See also:

- 2.4.1 Constructor 44
- 2.4.1 Constructor(files() as folderitem) 44
- 2.4.1 Constructor(pic as picture) 44

### GetFileContents(index as integer) as string

Plugin Version: 11.2 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** Queries file content for the file with the given index.

#### Notes:

Use index from WindowsFileDescriptorMBS.index property.

This works for files up to a few hundred megabytes in size. For larger files we will have to change plugin if you want to receive those.

### GetFileDescriptors as WindowsFileDescriptorMBS()

Plugin Version: 11.2 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** Queries file descriptors.

#### Example:

```

dim dataObject as WinDataObjectMBS // your data object

dim des(-1) as WindowsFileDescriptorMBS = dataObject.GetFilesDescriptors

for each d as WindowsFileDescriptorMBS in des
// we got file descriptions. Some metadata and the data. No path.
dim data as string = dataObject.GetFileContents(0)
msgbox "File """"+d.FileName+"""" with "+str(lenb(data))+ " bytes data."
next

```

**Notes:** Result array is empty on any error.

### GetPaths as folderitem()

Plugin Version: 10.5 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** Queries the paths in the data object.

#### Example:

```

dim dataObject as WinDataObjectMBS // your data object

dim files(-1) as FolderItem = dataObject.GetPaths

for each f as FolderItem in files
// we got a file you can use like any other file (e.g. copy)
listbox1.AddRow "Path """"+f.AbsolutePath+""""
next

```

**Notes:** Checks for a CF\_ HDROP type. May return one or more folderitems.

### GetPathStrings as string()

Plugin Version: 11.0 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** Queries the paths in the data object.

**Notes:** Checks for a CF\_ HDROP type. May return one or more folderitems.

**GetPicture as picture**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** Queries the picture from the data object.

**Notes:** Supports CF\_BITMAP/TYMED\_GDI.

**GetRaw(format as integer) as string**

Plugin Version: 10.5 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** Queries the raw data for the given type.

**GetText as string**

Plugin Version: 10.5 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** Queries the text from the data object.

**Notes:** Returns unicode or ANSI text depending on what is available. Unicode is preferred.

**HasFileDescriptors as boolean**

Plugin Version: 11.2 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** Checks whether this data object contains file descriptors.

**Notes:**

Returns true if a path is found.

Checks for CF\_FILEGROUPDESCRIPTOR.

Windows uses File Descriptors and FileContents for drag and drop operations where the data is not stored in a file. You get the descriptors and if you need you can get the data which is delivered just in time.

**HasPaths as boolean**

Plugin Version: 10.5 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** Checks whether this data object contains file paths.

**Notes:**

Returns true if a path is found.  
Checks for CF\_HDROP.

### **HasPicture as boolean**

Plugin Version: 10.5 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** Checks if a picture is part of this data object.

**Notes:**

Returns true if a picture is found.  
Checks for CF\_BITMAP.

### **HasRaw(format as integer) as boolean**

Plugin Version: 10.5 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** Checks whether this data object contains data in the given format.

### **HasText as boolean**

Plugin Version: 10.5 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** Checks whether this data object contains Text or UnicodeText.

**Notes:**

Returns true if text is found.  
Checks for CF\_UNICODETEXT and CF\_TEXT.

## **2.4.2 Properties**

### **Handle as Integer**

Plugin Version: 10.5 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** The internal handle.  
**Notes:** (Read and Write property)

### HelperHandle as Integer

Plugin Version: 10.5 Console & Web: Yes Mac: No, Win: Yes, Linux: No, . **Function:** The internal helper object to handle the drag image.

**Notes:**

This value is not zero if the AddDragImage call was successful.  
(Read and Write property)

### 2.4.3 Constants

#### CF\_BITMAP = 2

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the constants for the data types.

#### CF\_DIB = 8

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the constants for the data types.

#### CF\_DIBV5 = 17

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the constants for the data types.

#### CF\_DIF = 5

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the constants for the data types.

**CF\_ENHMETAFILE = 14**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the constants for the data types.

**CF\_HDROP = 15**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the constants for the data types.

**CF\_LOCALE = 16**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the constants for the data types.

**CF\_METAFILEPICT = 3**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the constants for the data types.

**CF\_OEMTEXT = 7**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the constants for the data types.

**CF\_PALETTE = 9**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the constants for the data types.

**CF\_PENDATA = 10**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the constants for the data types.

**CF\_RIFF = 11**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the constants for the data types.

**CF\_SYLK = 4**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the constants for the data types.

**CF\_TEXT = 1**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the constants for the data types.

**CF\_TIFF = 6**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the constants for the data types.

**CF\_UNICODETEXT = 13**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the constants for the data types.

**CF\_WAVE = 12**

Plugin Version: 10.5 Console & Web: No Mac: No, Win: Yes, Linux: No, . **Function:** One of the constants for the data types.

## Chapter 3

### List of all classes

- WinDataObjectMBS 41
- WindowsDragSourceMBS 20
- WindowsDropTargetMBS 7
- WindowsFileDescriptorMBS 29