ディスクメディエーター呉葉

外部接続仕様書

Disk Mediator KUREHA COM API Reference Manual

平成１９年１１月１９日

目次

改定履歴

Revision	改定内容	発行日
1.0	新規発行	H19.11.19

1　はじめに

呉葉はOLEオートメーションサーバーとして動作し、OLEクライアントとなる外部アプリケーションから制御することが可能である。本仕様書は、呉葉の外部接続インタフェースを解説する、外部接続仕様書である。普通に呉葉を利用する方は読む必要はない。

1.1　配布条件について

呉葉を利用するOLEクライアントアプリケーションは、次の条件に該当するものであれば、本ソフトウェアを添付、若しくは組み込んだ状態で配布、販売する事ができる。

- 商用システム（法人）
- 商用パッケージソフト（法人）
- 同人ソフト（個人／団体）
- フリーソフト（個人／団体／法人）

これ以外の形態をとるアプリケーション、即ち個人／団体が作成したシェアウェア、またはPDSには添付を許可しない。

操作説明書の免責事項等の項目も参照されたい。

1.2　表記について

本書での型表現は下表の通りである。利用の際は、各言語の対応する型に読み替えた上で対応されたい。

	本書の記述	C・C++	C#	VB5・6 / VBS	VB.NET
変数型	Boolean	bool	bool	Boolean	Boolean
	Byte	unsigned char	byte	Byte	Byte
	Integer	short int	short	Integer	Short
	Long	long int	int	Long	Integer
	String	wchar_t*	string	String	String
値型	(宣言なし)	(宣言なし)	(宣言なし)	ByVal	(ByVal)
参照型	ByRef	& (C++)	ref	(ByRef)	ByRef
既定引数	[～=x]	～=x (C++)	(機能なし)	Optional～=x	Optional～=x

2　簡易制御用

マスタリングクラスは、呉葉のマスタリング機能を簡易的に使用し、CD/DVD のマスタリング作業を行うためのクラスである。

2.1　簡易マスタリングクラス Mastering

2.1.1 VolumeLabel (Property)

■概要

データトラックのボリュームラベルを指定する。

■構文

[String =] *object.***VolumeLabel** [= String]

■設定値

データトラックのボリュームラベル

■用法

設定及び参照

■備考

空の文字列を指定した場合、ボリュームラベルは作成されない。

2.1.2 InitISOFS (Method)

■概要

データトラックのファイルシステムを指定する。

■構文

[Boolean =] *object.***InitISOFS**(FileSystemType As Long)

■引数

FileSystemType

値	意味
00h	Level1
01h	DOS 互換
02h	Level2
03h	Romeo 拡張
04h	Level4
10h	Joliet 拡張
11h	Joliet+拡張

■戻り値

True　　：成功

False　　：失敗（不正な FileSystemType）

■備考

このメソッドを実行すると、データトラックの内容はクリアされる。

2.1.3 ClearISO (Method)

■概要

データトラックに追加された全てのファイルを削除する。

■構文

*object.***ClearISO**()

■引数

なし

■戻り値

なし

■備考

2.1.4 IsISOEmpty (Method)

■概要

データトラックが空かどうかを判定する。

■構文

[Boolean =] *object.***InitISOEmpty**()

■引数

なし

■戻り値

True : データトラックは空
False : データトラックにはファイル／ディレクトリが存在する

■備考

2.1.5 AddFile (Method)

■概要

データトラックにファイル／ディレクトリを追加する。

■構文

[Boolean =] *object.* **AddFile**(FileName As String,
[DirectoryName As String])

■引数

FileName : 追加対象のファイル／ディレクトリをフルパスで指定(ワイルドカード使用可能)
DirectoryName : データトラック内の追加されるパスを指定

■戻り値

True : 成功
False : 失敗

■備考

- DirectoryName を省略すると、データトラック内のルートに追加される。
- 例えば C:¥Windows ディレクトリの全ての INI ファイルを、データトラックの TEXT¥INI ディレクトリに追加するには、次のように記述する。
  Call object.AddFile("C:¥Windows¥*.ini", "TEXT¥INI")
- 追加できないファイルや、作成できないディレクトリが存在する場合、外部に例外を投げる。このためエラーの捕捉が必要となる。

2.1.6 ClearTrack (Method)

■概要

追加したトラック全てを削除する。

■構文

object. **ClearTrack**()

■引数

なし

■戻り値

なし

■備考

2.1.7 AddTrack (Method)

■概要

書き込み対象のトラックファイルを追加する。

■構文

[Boolean =] *object.***AddTrack**(TrackFileName As String,
PregapLength As Long,
PostgapLength As Long,
[TrackControlFlag As Long])

■引数

TrackFileName : 追加する wave ファイル名
PregapLength : プリギャップをフレーム単位で指定
PostgapLength : ポストギャップをフレーム単位で指定
TrackControlFlag : トラック属性を指定

値	内容
01h	高域補正トラック
02h	複製禁止トラック

論理和で複数指定

■戻り値

True : 成功
False : 失敗（非対応のファイル）

■備考

- 現状、44.1kHz でサンプリングされた、Wave フォーマットのファイルのみ指定可能。
- 1秒は 75 フレーム

2.1.8 RemoveTrack (Method)

■概要

指定されたトラックを削除する。

■構文

[Boolean =] *object.***RemoveTrack**(TrackNo As Long)

■引数

TrackNo : 削除するトラック (1～TrackCount)

■戻り値

True : 成功
False : 失敗（指定されたトラックは存在しない）

■備考

- データトラックが空ではないとき、1 トラック目はデータトラックとなる。それを削除する

と、データトラックの内容が全て削除される。

- 音楽トラックは、AddTrack メソッドで追加された順にトラック番号が採番される。

2.1.9 GetTrackCount (Method)

■概要

全トラック数を返却する。

■構文

[Long =] *object.***GetTrackCount**()

■引数

なし

■戻り値

トラック数

■備考

- データトラックが空ではないとき、1 トラック目はデータトラックとなる。この場合、AddTrack メソッドにて追加されたトラック数+1 が返却される。

2.1.10 IsReady (Method)

■概要

ドライヴの準備が出来ているかどうかを確認する。

■構文

[Boolean =] *object.***IsReady**(DriveName As String)

■引数

DriveName　　：判定対象のドライヴ名(例 "D:")

■戻り値

True　：準備が出来ている(メディアがセットされている)
False　：準備が出来ていない

■備考

2.1.11 IsDiscWritable (Method)

■概要

ドライヴにセットされているメディアが書き込み可能かどうかを判定する。

■構文

[Boolean =] *object.***IsDiscWritable**(DriveName As String)

■引数

DriveName　　：判定対象のドライヴ名(例 "D:")

■戻り値

True　：書き込み可能なメディアがセットされている
False　：書き込み不能なメディアか、または準備が出来ていない

■備考

- CD-R/DVD-R などのライトワンスメディアは、書き込み可能であっても一度でも書き込まれていれば、書き込み不能と判定される。
- CD-RW/DVD-RW などの書き換え可能メディアは、書き込みが行われファイナライズされていても、書き込み可能と判定される。

2.1.12 ShowBurnDialog (Method)

■概要

書き込み開始ダイアログを表示する。

■構文

[Long =] *object*.**ShowBurnDialog**(Wnd As Long,
[Title As String],
[DriveName As String])

■引数

hWnd	: 親ウィンドウのハンドル（無い場合は 0）
Title	: ダイアログに表示する処理名（例："バックアップ作成"）
DriveName	: 書き込み対象のデフォルトドライヴ名(例 "D:")

■戻り値

ダイアログが閉じられるまでに、実際にディスクに正常に書き込まれた枚数。

■備考

- 書き込み処理や設定は、ダイアログ上でユーザによって行われる。
- ダイアログを使用しないで制御する場合は、StartBurn/StopBurn メソッドを利用する。

2.1.13 StartBurn (Method)

■概要

書き込みを開始する。

■構文

[Boolean =] *object*.**StartBurn**(DriveName As String,
[EjectAfterBurn As Boolean])

■引数

DriveName	: 書き込み対象のドライヴ名(例 "D:")
EjectAfterBurn	: 書き込み後ディスクを排出する場合は True を指定。

■戻り値

True	: エラーが発生することなく、書き込みを完了した。
False	: 書き込みに失敗した。

■備考

- EjectAfterBurn を省略すると、True を指定したと見なされる。
- 書き込み中は、定期的に Progress イベントが外部に発行される。
- 書き込み中にエラーが発生した場合（ドライヴの準備が出来ていない場合を含む）、外部に例外を投げるため、エラーの捕捉が必要となる。
- 書き込み時の詳細設定内容は、既にそのドライヴに対して呉葉で行われている設定を参照する。

2.1.14 StopBurn (Method)

■概要

書き込みを中止する。

■構文

[Boolean =] *object*.**StopBurn**()

■引数

なし

■戻り値

True　　: 成功
False　　: 失敗

■備考

- StartBurn メソッドは、書き込みが完了若しくは失敗するまで終了しない。故に、このメソッドは StartBurn メソッドによる書き込み中に非同期で呼び出す必要がある。

2.1.15 Progress (Event)

■概要

StartBurn メソッドによる書き込み進捗を通知する。

■構文

*object*_**Progress**(Value As Long, Total As Long)

■引数

Value　　: 進捗値
Total　　: 進捗値の最大値

■備考

なし

2.2　簡易制御のサンプル

言語は VB5/VB6 を想定している。VBS の場合は、型宣言は不要であり、VB7 以降の場合は、Set ステートメントは不要であるため、読み替えて対応していただきたい。

2.2.1　バックアップディスク作成例

C:¥Windows ディレクトリの全ファイルをバックアップする例を次に示す。

ただし、ディスク容量の問題でこのバックアップは失敗すると思われるため、実験の際は他の適当なパスに変更して頂きたい。

```
Sub Main()

    Dim objMastering As Object
    Dim lngRet As Long

    Set objMastering = CreateObject("KUREHA.Mastering")
    With objMastering
        Call .ClearTrack()
        Call .InitISOFS(&H11) 'Joliet+
        On Error Resume Next '注:AddFile メソッドは例外を投げることがある
        Call .AddFile("C:¥Windows¥*.*")
        On Error Goto 0
        lngRet = .ShowBurnDialog(0, "データバックアップ")
        If (lngRet > 0) Then
            Call MsgBox("成功しました")
        End If
```

```
    End With
    Set objMastering = Nothing

End Sub
```

2.2.2 音楽ディスク作成例

C:¥AudioTrack¥Hoge ディレクトリに保存されている、"Hoge_01.WAV"～"Hoge_05.WAV" ファイルをCDのオーディオトラックに書き込む例を次に示す。

```
Sub Main()

    Dim objMastering As Object
    Dim lngRet As Long

    Set objMastering = CreateObject("KUREHA.Mastering")
    With objMastering
        Call .ClearTrack()
        Call .AddTrack("C:¥AudioTrack¥Hoge¥Hoge_01.WAV", 150, 0) 'Pregap=2 秒
        Call .AddTrack("C:¥AudioTrack¥Hoge¥Hoge_02.WAV", 150, 0)
        Call .AddTrack("C:¥AudioTrack¥Hoge¥Hoge_03.WAV", 150, 0)
        Call .AddTrack("C:¥AudioTrack¥Hoge¥Hoge_04.WAV", 150, 0)
        Call .AddTrack("C:¥AudioTrack¥Hoge¥Hoge_05.WAV", 150, 0)
        lngRet = .ShowBurnDialog(0, "音楽ＣＤ生成")
        If (lngRet > 0) Then
            Call MsgBox("成功しました")
        End If
    End With
    Set objMastering = Nothing

End Sub
```

2.2.3 ダイアログを使用しない例

書き込みダイアログを使用せずに、C:¥Windows ディレクトリの全ファイルを、書き込みドライヴ D: に対してバックアップする例を次に示す。

以下のコードを実行する際は、事前に "Blacknoise KUREHA Object Library" を参照設定し、コマンドボタンとプログレスバーをフォームに張り付ける。

```
Private WithEvents mobjMastering As KUREHA.Mastering

Private Sub Command1_Click()
    Set mobjMastering = New KUREHA.Mastering

    With objMastering
        Call .ClearTrack()
        Call .InitISOFS(KUREHA_JISX0606_JOLIET2)
        On Error Resume Next '注:AddFile メソッドは例外を投げることがある
        Call .AddFile("C:¥Windows¥*.*")
        On Error Goto 0

        On Error Resume Next
        Call .StartBurn("D:")
        If (Err.Number <> 0) Then
            Call MsgBox(Err.Description)
        End If
        On Error GoTo 0
    End With
```

```
    Set mobjMastering = Nothing
End Sub

Private Sub mobjMastering_Progress(ByVal Value As Long, ByVal Total As Long)
    ProgressBar1.Max = Total
    ProgressBar1.Value = Value
End Sub
```

また、VB7 以降での記述は次の通りとなる。

```
Private WithEvents mobjMastering As KUREHA.Mastering

Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Button1.Click
    mobjMastering = New KUREHA.Mastering

    With mobjMastering
        Call .ClearTrack()
        Call .InitISOFS(KUREHA.MasteringFileSystemConstants.KUREHA_JISX0606_JOLIET2)
        On Error Resume Next
        Call .AddFile("C:¥Windows¥*.*")
        On Error GoTo 0

        On Error Resume Next
        Call .StartBurn("D:")
        If (Err.Number <> 0) Then
            Call MsgBox(Err.Description)
        End If
        On Error GoTo 0
    End With

    mobjMastering = Nothing
End Sub
```

3　低レベルエンジン

ディスク媒介エンジンなどのインターフェースを直接操作することにより、低レベルの入出力を実現するためのクラス群である。

3.1　ディスク媒介エンジン Momiji

ディスクのセクタ単位での読み書き制御を提供するクラスである。

3.1.1 GetEngineVersion (Method)

■概要

エンジンのバージョンを取得する。

■構文

[Long =] *object.***GetEngineVersion**()

■引数

なし

■戻り値

バージョン情報
下2桁　リビジョン
中2桁　マイナーバージョン
上2桁　メジャーバージョン

3.1.2 GetRemoteHosts (Method)

■概要

遠隔 SPTI サーバを検索し、そのホストアドレスを取得する。

■構文

[String =] *object.***GetRemoteHosts**(BroadCastIP As String,
PortNo As Long,
[Timeout As Long = 200])

■引数

BroadCastIP	: ブロードキャストアドレス
PortNo	: ポート番号
Timeout	: 検索タイムアウト時間(ms)

■戻り値

ホストアドレス

■備考

- ブロードキャストアドレスは、デリミタ '|' 区切りで複数指定可能。
  例:"192.168.0.255|192.168.2.255"
- 返却されるホストアドレスは、デリミタ '|' 区切りで複数返却される。
  例:"192.168.0.4|192.168.0.100|192.168.2.66"

3.1.3 OpenDevice (Method)

■概要

デバイスを開く。

■構文

[Boolean =] *object.***OpenDevice**(Drive As Integer)

■引数

Drive　　：論理デバイス番号(A=0, B=1, C=2 …)

■戻り値

True　：成功
False　：失敗

■備考

- MMC 規格の光学デバイスドライヴ以外は開けない。
- 他のアプリケーションで利用しているデバイスは開けない。

3.1.4 OpenDeviceEx (Method)

■概要

デバイスを開く。

■構文

[Boolean =] *object.***OpenDeviceEx**(DeviceName As String)

■引数

DeviceName　：論理デバイス名
("¥¥.¥D:"、"¥¥192.168.0.5:8801¥D:" など)

■戻り値

True　：成功
False　：失敗

■備考

- "¥¥.¥D:"の形式で指定すると、ローカル PC に接続されている D:ドライヴを開く。
- "¥¥192.168.0.5:8801¥D:"の形式では、IP アドレス 192.168.0.5 の 8801 番ポートに接続し、その PC の D:ドライヴを開く。予め 192.168.0.5 の PC に、遠隔 SPTI デーモンがインストールされていることが前提である。

3.1.5 CloseDevice (Method)

■概要

デバイスを閉じる。

■構文

[Boolean =] *object.***CloseDevice**()

■引数

なし

■戻り値

3.1.6 GetDeviceName (Method)

■概要

現在開いているデバイス名を取得する。

■構文

[String =] *object.***GetDeviceName**([NameType As Long = DNC_ALL])

■引数

NameType

値	内容
DNC_VENDOR(1)	ベンダ名
DNC_NAME(2)	製品名
DNC_REVISION(4)	リビジョン番号
DNC_VENDORNAME(3)	ベンダ名+製品名
DNC_ALL(7)	ベンダ名+製品名+リビジョン番号

■戻り値

現在開いているデバイスの指定された項目名

■備考

- ベンダ名は最大 8 バイト、製品名は 16 バイト、リビジョンは 4 バイトです。

3.1.7 ClearTOCStructure (Method)

■概要

エンジンにキャッシュされている TOC をクリアする。

■構文

*object.***ClearTOCStructure**()

■引数

なし

■戻り値

なし

■備考

- メディアへの書き込み開始前、SetTOCStructure を行う前に実行する。

3.1.8 GetTOCStrucure (Method)

■概要

エンジンから TOC 情報を取得する。

■構文

[TOCInformation =] *object.***GetTOCStrucure**()

■引数

なし

■戻り値

取得した TOCInformation オブジェクトを返す。

■備考

- 現在挿入されているディスクの TOC 情報は、ReadStart 実行後から、ReadEnd 実行前までの間に取得可能。
- ReadStart 実行直後は MCN/ISRC 以外の全ての情報を取得可能。MCN/ISRC は、全てのセクタに対して ReadLBA を実施した後に取得可能となる。

3.1.9 SetTOCStructure (Method)

■概要

エンジンに TOC 情報を設定する。

■構文

object.**SetTOCStrucure**(Info As TOCInformation)

■引数

Info : TOC 情報（TOCInformation オブジェクト）

■戻り値

なし

■備考

- ディスクへの書き込みの際は、WriteStart 実行前に、予め SetTOCStructure で必要な TOC 情報を全て転送しておく必要がある。
- SetTOCStructure で転送を開始する前に、予め ClearTOCStructure で不要な TOC 情報をクリアしておく必要がある。
- ディスクへ書き込む際、最低限標準 TOC 情報は必要である。例え拡張 TOC 情報があったとしても、標準 TOC 情報は必要。

3.1.10 GetFirstLBA (Method)

■概要

処理対象の最初の LBA を取得する。

■構文

[Long =] *object*.**GetFirstLBA**()

■引数

なし

■戻り値

LBA

■備考

- 読み込み中（ReadStart 実行後から ReadEnd 実行までの間）、または書き込み中（WriteStart 実行後から WriteEnd 実行までの間）にのみ正当な値を取得可能。
- 読み込み中に返される値は、CD の場合は-150～0、DVD の場合は 0 である。この値が物理的に読み込み可能なセクタ番号の下限、すなわち ReadLBA にて指定できるセクタ番号の下限となる。ただし、CD の場合は-150～-1 の範囲のセクタは常に指定可能で、この場合にはソフトウェア的に自動生成されたセクタイメージが取得できる。
- 書き込み中に返される値は、CD の場合は DAO モード書き込み時-150 未満の値、SAO モード書き込み時-150 で、DVD の場合は 0 である。この値が物理的に書き込み可能なセクタ番号の下限、即ち WriteLBA にて指定できるセクタ番号の下限となる。ただし CD の DAO モード時に返される、-150 未満のセクタ番号はリードイン領域となるため、通常は WriteLBA ではデータの転送を行わない。-150 以上のデータを転送する。

3.1.11 GetLastLBA (Method)

■概要

処理対象の最後の LBA を取得する。

■構文

[Long =] *object.***GetLastLBA**()

■引数

なし

■戻り値

LBA

■備考

- 読み込み中（ReadStart 実行後から ReadEnd 実行までの間）、または書き込み中（WriteStart 実行後から WriteEnd 実行までの間）にのみ正当な値を取得可能。
- 読み込み中に返される値は、物理的に読み込み可能なセクタ番号の上限、最終セッション最終トラックのセクタ番号、すなわち ReadLBA にて指定できるセクタ番号の上限となる。
- 書き込み中に返される値は、物理的に書き込み可能なセクタの上限、すなわち WriteLBA にて指定できるセクタ番号の上限となる。ただし CD の DAO モード時に返される値はリードアウト領域を含んだ値となるため、通常は最終セッション最終トラックの最終セクタ番号までで転送をやめる。

3.1.12 SetBufferBlockLength

■概要

デバイスに一度に転送するデータフレーム数を指定する。

■構文

*object.***SetBufferBlockLength**(BlockLength As Integer)

■引数

BlockLength　　：データフレーム数（1~26）

■戻り値

なし

■備考

- 指定可能な範囲は、1 以上 26 以下である。
- デフォルトでは 26 が指定されている。

3.1.13 ReadStart (Method)

■概要

ディスクの読み取りを開始する。

■構文

[Long =] *object.***ReadStart**(ReadCommand As Long,
ReadFlag As Long)

■引数

ReadCommand　読み込みコマンド

値	内容	対象
CCC_DATA(0)	セクタデータのみ取得	DVD
CCC_RAW(1)	RAW セクタ取得	CD
CCC_RAW16(2)	RAW セクタ+SUB16 取得	CD
CCC_RAW96(3)	RAW セクタ+SUB96 取得	CD

ReadFlag　読み込みデータフラグ

値	内容	対象
CDD_MAINDATA(0h)	メインデータを読み取る。	CD/DVD
CDD_FIRSTPREGAP(4h)	読み取れる場合は、ファーストプリギャップを読み取る。	CD
CDD_FIRSTSESSIONONLY(8000h)	ファーストセッションのみ読み取る。	CD
CDD_NOPREGAP(4000h)	各トラックのプリギャップの解析を行わない。	CD
CDD_COLLECTSUBPQ(800000h)	読み取り時に Sub P チャンネル、Q チャンネルデータの修正を行う。	CD

※複数内容は論理和で指定。

■戻り値

エラーステータス

値	内容
CBR_OK(0)	成功
CBR_ALREADYRUNNING(1)	既に開始されている
CBR_UNKNOWNMEDIA(2)	非対応のメディア
CBR_IGNOREFLAG(3)	非対応の読み込みデータフラグ
CBR_IGNOREREADCOMMAND(4)	非対応の読み込みコマンド
CBR_TOCREADERROR(6)	TOC の読み込みに失敗した

■備考

- CD で Sub チャンネルデータを読み込むには、CCC_RAW16 または CCC_RAW96 を指定する必要があるが、ドライヴによっては対応していない可能性がある。
- 実行後、MCN/ISRC 以外の TOC 情報がエンジンの TOC 格納領域に読み込まれ、GetTOCStructure メソッドで TOC 情報を取得できるようになる。

3.1.14 ReadLBA (Method)

■概要

指定セクタを読み込む。

■構文

[Boolean =] *object.***ReadLBA**(ByRef Data() As Byte, LBA As Long)

■引数

Data　　読み込んだセクタデータを格納するバッファ

読み込みコマンド	格納される内容	対象	有効データ長
CCC_DATA(0)	セクタデータのみ	DVD	2048
CCC_RAW(1)	RAW セクタ	CD	2352
CCC_RAW16(2)	RAW セクタ+SUB16（内部的にSUB96に変換される）	CD	2448
CCC_RAW96(3)	RAW セクタ+SUB96	CD	2448

LBA　　セクタ番号

■戻り値

True　　：成功
False　　：失敗

■備考

- ReadStart 実行後から、ReadEnd 実行前までの間に実行可能。
- 指定できるセクタ番号は、GetFirstLBA で取得できる値から、GetLastLBA で取得できる値の範囲である。
- data に用意すべきバッファサイズは、CD の場合 2448 バイト、DVD の場合は 2048 バイトである。このうち有効なデータサイズは、「有効データ長」にて示される値となる。
- エラーセクタがあった場合など、メソッドは失敗し False が返却される。特に CD などではプロテクトなどの影響でごく普通に失敗するため、アプリケーションで適当な処理を行なりして処理を実行可能である。しかし、False 返却直後に GetSCSIErrorStatus メソッドを実行し、その結果が FFFFFFh の場合はコマンド実行エラー（主にハードウェアの障害）を示すため、読み込みを終了するしかない。

3.1.15 ReadEnd

■概要

ディスクの読み取りを終了する。

■構文

[Boolean =] *object.***ReadEnd**()

■引数

なし

■戻り値

True　　：成功
False　　：失敗

3.1.16 WriteStart (Method)

■概要

ディスクの書き込みを開始する。

■構文

[Long =] *object.***WriteStart**(WriteCommand As Long,
WriteFlag As Long)

■引数

WriteCommand　書き込みコマンド

値	内容	対象
CCC_SAO(0)	SAO モード書き込み	CD/DVD
CCC_SAO_RAW(1)	SAO モード RAW セクタ書き込み	CD
CCC_DAO_RAW16(3)	RAW モード RAW+SUB16 セクタ書き込み	CD
CCC_DAO_RAW96(4)	RAW モード RAW+SUB96 セクタ書き込み	CD

WriteFlag　書き込みデータフラグ

値	内容	対象
CDD_MAINDATA(0h)	メインデータを書き込みむ。	CD/DVD
CDD_OVERBURN(80000h)	メディアの容量を無視して書き込む。	CD
CDD_OPCOFF(40000h)	レーザー出力調整を行わない。	CD/DVD
CDD_COLLECTSUBPQ(800000h)	書き込み時に Sub P チャンネル、Q チャンネルデータの修正を行う。	CD

※複数内容は論理和で指定。

■戻り値

エラーステータス

値	内容
CBR_OK(0)	成功
CBR_ALREADYRUNNING(1)	既に開始されている
CBR_UNKNOWNMEDIA(2)	非対応のメディア
CBR_IGNOREFLAG(3)	非対応の書き込みデータフラグ
CBR_IGNOREWRITECOMMAND(5)	非対応の書き込みコマンド
CBR_NOTEMPTY(7)	既に書き込まれたメディア
CBR_RWFORMATERROR(8)	RW の初期化に失敗
CBR_OVERCAPACITY(9)	書き込みサイズがメディアより大きい

■備考

- 実行前に、TOC 情報をエンジンに設定しておく必要がある。

■概要

指定セクタを書き込む。

■構文

[Boolean =] *object.***WriteLBA**(DataType As Long,
ByRef Data() As Byte, LBA As Long)

■引数

DataType データタイプ

値	格納する内容	対象	データ長
CSD_DATA(0)	データのみ	CD/DVD	2048 ～2352
CSD_MAINCHANNEL(1)	CDRAW セクタ	CD/DVD	2352
CSD_MAINSUBCHANNEL(2)	CDRAW セクタ+SUB96	CD/DVD	2448
CSD_GENERATE(4)	なし(自動生成)	CD/DVD	0
CSD_ERRORDATA(8000h)	なし(エラーセクタ)	CD	0

Data　書き込むセクタデータを格納したバッファ
LBA　セクタ番号

■戻り値

True　: 成功
False　: 失敗

■備考

- WriteStart 実行後から、WriteEnd 実行前までの間に実行可能。
- 指定できるセクタ番号は、GetFirstLBA で取得できる値から、GetLastLBA で取得できる値の範囲である。
- セクタ番号は基本的に、GetFirstLBA で示される値から GetLastLBA で示される値まで連続して指定して行くが、途中の番号を飛ばすことも可能である。この場合、飛ばしたセクタは CSD_GENERATE が指定されたものとして自動的に書き込まれる。具体的には、リードイン、リードアウト、ファーストプリギャップ領域などをこの手法を利用して省略することで、ISO ファイルなどの内容をリニアに指定して書き込むことが可能である。
- 既に書き込んだセクタ番号を再度指定した場合は基本的にエラーとなるが、書き込みコマンドが CCC_DAO_RAW16 または CCC_DAO_RAW96 の場合に限り、多重セクタとして容量の許す限りは正常に書き込める。
- data に用意すべきバッファサイズは、CD の場合 2448 バイト、DVD の場合は 2048 バイトである。このうち有効なデータを格納する領域は、表の「データ長」となる。
- CSD_ DATA(0)を指定した場合、Data にはセクタ本体のデータのみを格納する。このデータは DVD ならば 2048 バイトとなるが、CD の場合は物理フォーマットにより、2048 バイト(Mode1/Mode2XAForm1)、2334 バイト(Mode2XAForm2)、2336 バイト(Mode2)、2352 バイト(CDDA)となる。
- CSD_MAINCHANNEL(1)を指定した場合、Data には CD 形式の RAW セクタデータ 2352 バイト分を格納する。(書き込み先が DVD の場合は、Mode1/Mode2XAForm1 形式の RAW セクタデータであれば自動的に変換されて書き込まれる)
- CSD_MAINSUBCHANNEL(2)を指定した場合、Data には CD 形式の RAW セクタデ

ータ 2352 バイト分と、P-W Sub チャンネルデータ 96 バイト分の、合計 2448 バイトを格納する。

- CSD_GENERATE(4)を指定した場合、Data には何も格納する必要がない。エンジンがセクタ番号より自動的に生成したデータを書き込む。リードイン、リードアウトに相当するセクタ番号を書き込む際には必ず指定する。そのほか、ギャップ等を書き込む際にも利用可能である。
- CSD_ERRORDATA(8000h)を指定した場合、Data には何も格納する必要はない。そのセクタ番号はエラーセクタとして書き込まれる。

3.1.18 WriteEnd (Method)

■概要

ディスクの書き込みを終了する。

■構文

[Boolean =] *object.***WriteEnd**([Abort As Boolean] = False) As Boolean

■引数

Abort　強制終了フラグ(True=強制終了, False=通常終了)

■戻り値

True　: 成功
False　: 失敗

■備考

- このメソッドを呼び出さない限り、ドライヴは書き込みを終了しない。このため、一度 WriteStart を実行したら、例え強制終了する場合であってもこのメソッドを実行する必要がある。
- 強制終了フラグを True にすると、本来必要な後処理を全て省略して、可能な限り高速に終了する。この際、書き込み中であったディスクは使用不能になる可能性が高い。ただし、書き込み対象が DVD の場合には、強制終了であってもそれなりの時間が掛かる。

3.1.19 LoadTray (Method)

■概要

トレーを開閉する。

■構文

[Boolean =] *object.***LoadTray**(LoadUnload As Boolean)

■引数

LoadUnload　開閉状態(True=閉じる, False=開く)

■戻り値

True　: 成功
False　: 失敗

3.1.20 IsReady (Method)

■概要

メディアがセットされているかを確認する。

■構文

[Boolean =] *object.***IsReady**()

■引数

なし

■戻り値

True　　: セットされている
False　　: セットされていない

3.1.21 GetMediaType (Method)

■概要

セットされているメディアの種別を取得する。

■構文

[Long =] *object.***GetMediaType**()

■引数

なし

■戻り値

メディア種別

値	内容
MTC_CD_ROM(08h)	CD-ROM
MTC_CD_R(09h)	CD-R
MTC_CD_RW(0Ah)	CD-RW
MTC_DVD_ROM(10h)	DVD-ROM
MTC_DVD_R(11h)	DVD-R
MTC_DVD_RAM(12h)	DVD-RAM
MTC_DVD_RW_OW(13h)	DVD-RW(OverwriteMode)
MTC_DVD_RW(14h)	DVD-RW
MTC_DVD_R_DL(15h)	DVD-R DL
MTC_DVD_R_DL_JR(16h)	DVD-R DL(JumpRecording)
MTC_DVD _RW_DL(17h)	DVD-RW DL
MTC_DVD_DDR(18h)	DVD-Download
MTC_PC_RW(1Ah)	DVD+RW
MTC_PC_R(1Bh)	DVD+R
MTC_PC_RW_DL(2Ah)	DVD+RW DL
MTC_PC_R_DL(2Bh)	DVD+R DL
MTC_BD_ROM(40h)	BD-ROM
MTC_BD_R(41h)	BD-R
MTC_BD_R_RR(42h)	BD-R(RandomRecordingMode)
MTC_BD_RE(43h)	BD-RE
MTC_HDDVD_ROM(50h)	HDDVD-ROM
MTC_HDDVD_R(51h)	HDDVD-R
MTC_HDDVD_RAM(52h)	HDDVD-RAM
MTC_HDDVD_RW(53h)	HDDVD-RW
MTC_HDDVD_R_DL(58h)	HDDVD-R DL
MTC_HDDVD_RW_DL(5Ah)	HDDVD-RW DL
MTC_ERROR(FFFFh)	不明（非対応メディア）

3.1.22 GetMediaFamilyType (Method)

■概要

セットされているメディアのファミリを取得する。

■構文

[Long =] *object.***GetMediaFamilyType**()

■引数

なし

■戻り値

メディアファミリ種別

値	内容
MFT_CDFAMILY(01h)	CD 系媒体
MFT_DVDFAMILY(02h)	DVD、PC 系媒体
MFT_BDFAMILY(03h)	BD 系媒体
MFT_HDDVDFAMILY(04h)	HDDVD 系媒体
MFT_UNKNOWN (FFh)	不明（非対応メディア）

3.1.23 LockUnlock (Method)

■概要

トレー開閉機構をロック／解除する。

■構文

[Boolean =] *object.***LockUnlock**(Lock As Boolean)

■引数

Lock　　ロック状態(True=ロック, False=アンロック)

■戻り値

True　　: 成功
False　　: 失敗

■備考

- ロックすると、ドライヴ前面のイジェクトボタンや LoadTray メソッドでトレーの排出が出来なくなる。

3.1.24 EraseMedia (Method)

■概要

書き換え可能メディアを消去する。

■構文

[Boolean =] *object.***EraseMedia**([Quickly As Boolean = True])

■引数

Quickly　消去方法(True=高速消去, False=通常消去)

■戻り値

True　　: 成功
False　　: 失敗

3.1.25 IsSupportMedia (Method)

■概要

現在開いているデバイスが指定メディア種別に対応しているかを確認する。

■構文

[Boolean =] *object.***IsSupportMedia**(MediaType As Long)

■引数

MediaType　　メディア種別（内訳は GetMediaType の戻り値を参照）

■戻り値

True　：対応している
False　：対応していない

3.1.26 IsReadSupport (Method)

■概要

現在開いているデバイスが指定メディアファミリ種別の読み込みに対応しているかを確認する。

■構文

[Boolean =] *object.***IsReadSupport**(MediaFamilyType As Long)

■引数

MediaFamilyType　メディアファミリ種別
（内訳は GetMediaFamilyType の戻り値を参照）

■戻り値

True　：対応している
False　：対応していない（※注：備考欄参照）

■備考

- このメソッドは、IsSupportMedia メソッドに対応していない、初期のドライヴのために用意されている。DVD 対応ドライヴ以降では、IsSupportMedia メソッドを利用されたい。
- このメソッドは、メディアファミリ種別が MFT_CDFAMILY、MFT_DVDFAMILY 以外では機能しない（常に「対応していない」となる）。それ以降のメディアの対応状況の検出には IsSupportMedia メソッドを利用されたい。

3.1.27 IsWriteSupport (Method)

■概要

現在開いているデバイスが指定メディアファミリ種別の書き込みに対応しているかを確認する。

■構文

[Boolean =] *object.***IsWriteSupport**(MediaFamilyType As Long)

■引数

MediaFamilyType　メディアファミリ種別
（内訳は GetMediaFamilyType の戻り値を参照）

■戻り値

True　：対応している
False　：対応していない（※注：備考欄参照）

■備考

- このメソッドは、IsSupportMedia メソッドに対応していない、初期のドライヴのために用意されている。DVD 対応ドライヴ以降では、IsSupportMedia メソッドを利用されたい。
- このメソッドは、メディアファミリ種別が MFT_CDFAMILY、MFT_DVDFAMILY 以外では機能しない（常に「対応していない」となる）。それ以降のメディアの対応状況の検

出には IsSupportMedia メソッドを利用されたい。

3.1.28 IsDiscEmpty (Method)

■概要

ディスクに何も書き込まれていないかどうかを確認する。

■構文

[Boolean =] *object*.**IsDiscEmpty**()

■引数

なし

■戻り値

True : ディスクが空
False : 書き込まれている

3.1.29 SetECCMode (Method)

■概要

ディスク読み込み時のエラー訂正機能の有効状態を設定する。

■構文

[Boolean =] *object*.**SetECCMode**(Enabled As Boolean,
[RetryCount As Long = 0])

■引数

Enabled 訂正モード(True=有効, False=無効)
RetryCount エラー検出時の再試行回数

■戻り値

True : 成功
False : 失敗

■備考

- このメソッドは CD の読み込み時のみに意味を持ち、それ以外では設定内容は無効となる。

3.1.30 SetReadSpeed (Method)

■概要

ディスクの読み込み速度を指定する。

■構文

[Boolean =] *object*.**SetReadSpeed**([Speed As Integer = &hFFFF],
[CAV As Boolean = False])

■引数

Speed 転送速度[KiB/s](FFFFh でデフォルトの最高速)
CAV CAV/CLV(True=CAV , False=CLV)

■戻り値

True : 成功
False : 失敗

■備考

- ドライヴ、メディアの組み合わせによって指定可能な速度が限られる。設定内容に合致する速度が選択できない場合には、最も近い速度が選択される。

3.1.31 SetWriteSpeed (Method)

■概要

ディスクの書き込み速度を指定する。

■構文

[Boolean =] *object.***SetWriteSpeed**([Speed As Integer = &hFFFF],
[CAV As Boolean = False])

■引数

Speed : 転送速度[KiB/s]（FFFFh でデフォルトの最高速）
CAV : CAV/CLV(True=CAV , False=CLV)

■戻り値

True : 成功
False : 失敗

■備考

- ドライヴ、メディアの組み合わせによって指定可能な速度が限られる。設定内容に合致する速度が選択できない場合には、最も近い速度が選択される。

3.1.32 GetReadSpeed (Method)

■概要

対応する読み込み速度を取得する。

■構文

[DMESpeeds =] *object.***GetReadSpeed**()

■引数

なし

■戻り値

取得した速度情報が格納された MMC 速度情報コレクションを返す。

■備考

- 返却される読み込み速度は、ドライヴから取得したものではなくエンジン内部で生成した物であり、信頼性はない。GetWriteSpeed メソッドと対応づけるための仮想機能である。

3.1.33 GetWriteSpeed (Method)

■概要

対応する書き込み速度を取得する。

■構文

[DMESpeeds =] *object.***GetWriteSpeed**()

■引数

なし

■戻り値

取得した速度情報が格納された DMESpeeds コレクションを返す。

■備考

- ドライヴ、メディアの組み合わせによって指定可能な速度が限られる。設定内容に合致する速度が選択できない場合には、最も近い速度が選択される。

3.1.34 CheckWriteMode (Method)

■概要

指定された書き込みコマンドに対応しているかどうかを調査する。

■構文

[Boolean =] *object.***CheckWriteMode**(WriteCommand As Long,
TestMode As Boolean,
MediaFamilyType As Long)

■引数

WriteCommand	書き込みコマンド（WriteStart メソッド参照）
TestMode	テスト書き込みモード(True=テスト, False=本番)
MediaFamilyType	メディアファミリ種別（GetMediaFamilyType メソッド参照）

■戻り値

True	: 対応
False	: 非対応

3.1.35 SetWriteCacheBufferSize (Method)

■概要

非同期書き込み用緩衝域の容量を指定する。

■構文

*object.***SetWriteCacheBufferSize**(Size As Long)

■引数

Size	緩衝域の容量

■戻り値

なし

■備考

- デフォルトでは 10,485,760 バイト(10MiB)が指定されたものとして書き込み処理が実施される。
- 容量に 0 を指定すると、同期書き込みとなる。

3.1.36 GetUsedWriteCacheBufferSize (Method)

■概要

非同期書き込み用緩衝域の使用容量を取得する。

■構文

[Long =] *object.***GetUsedWriteCacheBufferSize**()

■引数

なし

■戻り値

使用しているの緩衝域の容量

■備考

- このメソッドは、書き込み中に緩衝域の状態を確認するために使用する。

3.1.37 GetSCSIErrorStatus (Method)

■概要

SCSI エラー情報を取得する。

■構文

[Long =] *object.***GetSCSIErrorStatus**()

■引数

なし

■戻り値

センスステータス
(SenseKey<<16 | ASC<<8 | ASCQ)
（但し FFFFFFh はコマンド実行エラーを表す）

■備考

- 読み書き等に失敗した場合（主に ReadLBA、WriteLBA メソッド実行後に False が返った場合）、SCSI コマンドのセンスステータスが取得できる。
- 戻り値は SCSI コマンドのセンスステータスであるが、FFFFFFh のみはコマンド実行エラーを示す致命的コードとなる。この状態になるのは、時間内にデバイスが応答を返さなかった場合で、主にハードウェアの障害が原因である。通常は発生することはないが、遠隔制御中にLANケーブルを引っこ抜くことにより、比較的容易に現象を発生させることが出来る。

3.1.38 GetSCSIErrorDescription (Method)

■概要

SCSI エラー情報明細を取得する。

■構文

[String =] *object.***GetSCSIErrorDescription**()

■引数

なし

■戻り値

センスステータスの明細内容

3.2 MMC 速度情報コレクション DMESpeeds

媒介エンジンの GetReadSpeed、GetWriteSpeed メソッド等で取得可能な、速度情報を格納するコレクションである。

3.2.1 Add (Method)

■概要

速度アイテムを追加する。

■構文

*object.***Add**(CAV As Boolean, Speed As Integer)

■引数

CAV : CAV/CLV(True=CAV, False=CLV)
Speed : 転送速度[KiB/s]（FFFFh でデフォルトの最高速）

■戻り値

なし

3.2.2 Clear (Method)

■概要

速度アイテムを全て削除する。

■構文

*object.***Clear**()

■引数

なし

■戻り値

なし

3.2.3 Count (Property)

■概要

コレクションに格納されているアイテム数を参照する。

■構文

[Integer =] *object.***Count**

■参照値

アイテム数

■用法

参照のみ

3.2.4 Delete (Method)

■概要

速度アイテムを削除する。

■構文

[Boolean =] *object.***Delete**(Index As Integer)

■引数

Index : 削除するアイテムのインデックス番号

■戻り値

True : 成功
True : 失敗

3.2.5 Items (Property)

■概要

速度アイテムを参照する。

■構文

[DMESpeed =] *object*.**Items**(Index As Integer)

■引数

Index　：アイテムのインデックス番号

■参照値

MMC 速度情報

■用法

参照のみ

3.3 MMC 速度情報 DMESpeed

速度情報コレクションにて格納される速度情報の実体を格納するクラスである。

3.3.1 Clear (Method)

■概要

速度情報を全て削除する。

■構文

*object.***Clear**()

■引数

なし

■戻り値

なし

3.3.2 Speed (Property)

■概要

速度を指定または参照する。

■構文

[Integer =] *object.***Speed** [= Integer]

■参照値

速度

■用法

設定及び参照

3.4　総合 TOC 情報 TOCInformation

媒介エンジンの GetTOCStructure、SetTOCStructure で利用する、総合 TOC 情報の枠組みを提供するクラスである。

TOC 情報には基本 TOC 情報、拡張 TOC 情報、MCN 情報、ISRC 情報、CDText 情報があり、この内必須の情報は基本 TOC 情報のみである。CD 系媒体の場合のみ、オプションでその他の情報を持つことが可能であり、ローモードでの読み書き時に有用な情報を取得、設定出来る。

3.4.1 Clear (Method)

■概要

基本 TOC アイテムを全て削除する。

■構文

*object.***Clear**()

■引数

なし

■戻り値

なし

3.4.2 TOCAdd (Method)

■概要

基本 TOC アイテムを追加する。

■構文

*object.***TOCAdd**(SessionNo As Byte,
TrackNo As Byte,
IndexNo As Byte,
Flags As Byte,
SectorMode As Byte,
StartLBA As Long,
EndLBA As Long)

■引数

SessionNo : セッション番号(1~)
TrackNo : トラック番号(1~)
IndexNo : インデックス番号(0~)
Flags : トラック属性

値	内容
TCF_PREEMPHASIS(1h)	高域補正
TCF_COPYPERMITTED(2h)	複製禁止
TCF_DATATRACK(4h)	データトラック

※論理和で複数指定

SectorMode : セクタモード

値	内容
SMC_AUDIO (0h)	音楽トラック
SMC_MODE1 (1h)	Mode1 データトラック
SMC_MODE2 (2h)	Mode2 データトラック

※CD 以外のデータトラックは SMC_MODE1 を指定

StartLBA : 開始 LBA
EndLBA : 終了 LBA

■戻り値

なし

3.4.3 TOCDelete (Method)

■概要

基本 TOC アイテムを削除する。

■構文

[Boolean =] *object.***TOCDelete**(Index As Integer)

■引数

Index : アイテムのインデックス番号

■戻り値

True : 成功
False : 失敗

3.4.4 TOC (Property)

■概要

基本 TOC アイテムを参照する。

■構文

[TOCTrack =] *object.***TOC**(Index As Integer)

■引数

Index　：アイテムのインデックス番号

■参照値

基本 TOC 情報

■用法

参照のみ

3.4.5 TOCCount (Property)

■概要

基本 TOC アイテム数を参照する。

■構文

[Integer =] *object.***TOCCount**

■参照値

基本 TOC アイテム数

■用法

参照のみ

3.4.6 RawTOCAdd (Method)

■概要

拡張 TOC アイテムを追加する。

■構文

*object.***RawTOCAdd**(SessionNo As Byte,
AdrControl As Byte,
Track As Byte,
Flags As Byte,
Point As Byte,
Minute As Byte,
Second As Byte,
Frame As Byte,
Zero As Byte,
PMinute As Byte,
PSecond As Byte,
PFrame As Byte)

■引数

各項の詳細については割愛。但し、以下のパラメータにのみ注意。

AdrControl　：上 4 ビット=ADR、下 4 ビット=CONTROL

詳細は、MMC 仕様書 READ TOC/PMA/ATIP コマンドの項に記載されている、Multi-session Information の応答書式(READ TOC/PMA/ATIP response data (Format = 0010b))を参照されたい。

■戻り値

なし

3.4.7 RawTOCDelete (Method)

■概要

拡張 TOC アイテムを削除する。

■構文

[Boolean =] *object.***RawTOCDelete**(Index As Integer)

■引数

Index　：アイテムのインデックス番号

■戻り値

True　：成功
False　：失敗

3.4.8 RawTOC (Property)

■概要

拡張 TOC アイテムを参照する。

■構文

[TOCRawTrack =] *object.***RawTOC**(Index As Integer)

■引数

Index　：アイテムのインデックス番号

■参照値

拡張 TOC 情報

■用法

参照のみ

3.4.9 RawTOCCount (Property)

■概要

拡張 TOC アイテム数を参照する。

■構文

[Integer =] *object.***RawTOCCount**

■参照値

拡張 TOC アイテム数

■用法

参照のみ

3.4.10 MCN (Property)

■概要

メディアカタログ番号を参照、または設定する。

■構文

[String =] *object.***MCN** [= String]

■参照値

メディアカタログ番号

■用法

設定及び参照

■備考

メディアカタログ番号は、必ず 13 桁の数字でなければならない。

3.4.11 ISRCAdd (Method)

■概要

ISRC アイテムを追加する。

■構文

*object.***ISRCAdd**(ISRC As String)

■引数

ISRC　　: 国際標準レコーディング番号

■戻り値

なし

■備考

- アイテムはトラック数分だけ追加し、各アイテムが各トラックの ISRC に相当する。ISRC が存在しないトラックは、空の文字列を追加しなければならない。
- 国際標準レコーディング番号は、12 桁の英数字でなければならない。

3.4.12 ISRCDelete (Method)

■概要

ISRC アイテムを削除する。

■構文

[Boolean =] *object.***ISRCDelete**(Index As Integer)

■引数

Index　　: アイテムのインデックス番号

■戻り値

True　　: 成功
False　　: 失敗

3.4.13 ISRC (Property)

■概要

ISRC アイテムを参照する。

■構文

[String =] *object.***ISRC**(Index As Integer)

■引数

Index　　: アイテムのインデックス番号

■参照値

国際標準レコーディング番号

■用法

参照のみ

■備考

- アイテムはトラック数分だけ追加存在しなければならない。即ち、インデックス番号はトラック番号と一致する筈である。

3.4.14 ISRCCount (Property)

■概要

ISRC アイテム数を参照する。

■構文

[Integer =] *object.***ISRCCount**

■参照値

ISRC アイテム数

■用法

参照のみ

3.4.15 CDTextAdd (Method)

■概要

CDText アイテムを追加する。

■構文

*object.***CDTextAdd**(PacketType As Byte,
TrackNo As Byte,
PacketNo As Byte,
Flags As Byte,
Data(12) As Byte,
Crc(2) As Byte)

■引数

各項の詳細については割愛。

■戻り値

なし

3.4.16 CDTextDelete (Method)

■概要

CDText アイテムを削除する。

■構文

[Boolean =] *object.***CDTextDelete**(Index As Integer)

■引数

Index　: アイテムのインデックス番号

■戻り値

True　: 成功
False　: 失敗

3.4.17 CDText (Property)

■概要

CDText アイテムを参照する。

■構文

[TOCCDText =] *object.***CDText**(Index As Integer)

■引数

Index　：アイテムのインデックス番号

■参照値

CDText 情報

■用法

参照のみ

3.4.18 CDTextCount (Property)

■概要

CDText アイテム数を参照する。

■構文

[Integer =] *object.***CDTextCount**

■参照値

CDText アイテム数

■用法

参照のみ

3.4.19 SaveCUEFile (Method)

■概要

総合 TOC 情報を CUE 形式ファイルに保存する。

■構文

[Boolean =] *object.***SaveCUEFile**(FileName As String,
ImageFileName As String,
CDTextFileName As String,
[SectorSize As Long = 2352])

■引数

FileName　：保存する CUE ファイル名
ImageFileName　：イメージ本体を保存したファイル名
CDTextFileName　：コンパイル済み CD-Text 情報を保存したファイル名
SectorSize　：イメージ本体のセクタサイズ

■戻り値

True　：成功
False　：失敗

■備考

- ファイルに保存されるのは、基本 TOC 情報、MCN 情報、ISRC 情報、CDText 情報のみである。拡張 TOC 情報は保存されないため、特に CD の TOC を保存する場合、情報の一部が失われる。
- マルチセッション形式の CD は基本 TOC 情報だけでは表現できないため、保存に失敗する。

3.4.20 SaveCCDFile (Method)

■概要

総合 TOC 情報を CCD 形式ファイルに保存する。

■構文

[Boolean =] *object.***SaveCCDFile**(FileName As String)

■引数

FileName　　: 保存する CCD ファイル名

■戻り値

True　: 成功
False　: 失敗

■備考

- ファイルに保存されるのは、基本 TOC 情報、拡張 TOC 情報、MCN 情報、ISRC 情報、CDText 情報と、全ての情報である。
- ファイル形式の仕様上拡張 TOC 情報が必須であるため、拡張 TOC 情報が存在しない CD 以外のメディアでは保存に失敗する。

3.4.21 LoadISOFile (Method)

■概要

ISO 形式ファイルから総合 TOC 情報を生成する。

■構文

[Boolean =] *object.***LoadISOFile**(FileName As String,
ByRef SectorSize As Long)

■引数

FileName　　: 読み込む ISO ファイル名
SectorSize　　: [出力] イメージのセクタサイズ

■戻り値

True　: 成功
False　: 失敗

■備考

- ISO 形式ファイルから読み込まれるのは、基本 TOC 情報のみである。従って、ISO 形式で読み込み後、CCD 形式で保存するなどということは不可能である。
- 正常に読み込みが行えると、引数 SectorSize に参照指定された変数にイメージのセクタサイズが格納される。このサイズが、書き込み時の基本セクタサイズとなる。

3.4.22 LoadCUEFile (Method)

■概要

CUE 形式ファイルを読み込み、総合 TOC 情報を設定する。

■構文

[Boolean =] *object*.**LoadCUEFile**(FileName As String,
ByRef ImageFileName As String,
ByRef SectorSize As Long)

■引数

FileName : 読み込む CUE ファイル名
ImageFileName : [出力] イメージ本体のファイル名
SectorSize : [出力] イメージのセクタサイズ

■戻り値

True : 成功
False : 失敗

■備考

- CUE 形式ファイルから読み込まれるのは、基本 TOC 情報、MCN 情報、ISRC 情報、CDText 情報のみである。従って、CUE 形式で読み込み後、CCD 形式で保存するなどということは不可能である。
- 正常に読み込みが行えると、引数 ImageFileName、SectorSize に参照指定された変数にイメージ本体のファイル名、セクタサイズが格納される。メディアへの書き込みは、このイメージを開き、このセクタサイズを利用して実施する。

3.4.23 LoadCCDFile (Method)

■概要

CCD 形式ファイルを読み込み、総合 TOC 情報を設定する。

■構文

[Boolean =] *object*.**LoadCCDFile**(FileName As String)

■引数

FileName : 読み込む CCD ファイル名

■戻り値

True : 成功
False : 失敗

3.5　低レベルエンジンの使用サンプル

このサンプルはVB5/6のコードであるが、VarPtr関数を利用するため、.NET環境への移植は注意を要する。該当個所はファイルの読み書き込みに Win32API を利用することによるものであるため、標準的な別の処理に置き換えることで解決が可能である。

事前に "Blacknoise KUREHA Object Library" を参照設定しておくこと。

3.5.1 ディスク読み込み例

セットされたディスクの全セクタを読み込み、CD は CCD+IMG+SUB ファイル、DVD は CUE+IMG ファイルに保存する例を次に示す。

```
Private Declare Function WriteFile Lib "kernel32" ( _
    ByVal hFile As Long, _
    ByVal lpBuffer As Long, _
    ByVal nNumberOfBytesToWrite As Long, _
    ByVal lpNumberOfBytesWritten As Long, _
    ByVal lpOverlapped As Long) As Long
Private Declare Function CreateFile Lib "kernel32" Alias "CreateFileA" ( _
    ByVal lpFileName As String, _
    ByVal dwDesiredAccess As Long, _
    ByVal dwShareMode As Long, _
    ByVal lpSecurityAttributes As Long, _
    ByVal dwCreationDisposition As Long, _
    ByVal dwFlagsAndAttributes As Long, _
    ByVal hTemplateFile As Long) As Long
Private Declare Function CloseHandle Lib "kernel32" (ByVal hObject As Long) As Long

'**** メイン
Sub Main()
    'D:ドライヴにセットされたCD/DVDのデータをc:¥sample.imgに保存
    Call ReadDisc("¥¥.¥D:", "c:¥sample.img")
End Sub

'*******************************************************************************
'*　名　前：ReadDisc
'*　機　能：ディスクを読み込み、ファイルに保存する
'*　引　数：strDeviceName ..... 読み込みドライヴ名
'*　　　　：　　　　　　　　　　("¥¥.¥Q:"、"¥¥192.168.0.2:8801¥Q:"など)
'*　　　　：strImageFileName ... イメージ本体の書き込みファイル名
'*　備　考：イメージ本体の他に、CDはCCD、DVDはCUEファイルを出力する
'*******************************************************************************
Private Function ReadDisc( _
    ByVal strDeviceName, _
    ByVal strImageFileName As String) As Boolean

    Dim objDevice      As Momiji
    Dim objTOC         As TOCInformation

    Dim strTOCFileName As String
    Dim strSUBFileName As String

    Dim lngMediaFamily As MediaFamilyTypeConstants
    Dim blnDone        As Boolean
    Dim ixTrack        As Integer
    Dim lngLBA         As Long
    Dim bytData(2352 + 96) As Byte
    Dim lngSectorSize  As Long
```

```
Dim hMainData As Long
Dim hSubData  As Long
Dim lngLength As Long

Set objDevice = New Momiji

'デバイスを開く
If (objDevice.OpenDeviceEx(strDeviceName)) Then
    'ディスクが挿入されているかを確認
    blnDone = objDevice.IsReady()
    If (Not blnDone) Then
        Call MsgBox("デバイスの準備が出来ていません。")
    End If

    'ディスク取り出し禁止
    Call objDevice.LockUnlock(True)

    'メディアファミリ取得
    If (blnDone) Then
        '拡張子を入れ替えてファイル名を作っているだけ
        lngMediaFamily = objDevice.GetMediaFamilyType()
        If (lngMediaFamily = MFT_CDFAMILY) Then
            strTOCFileName = Left(strImageFileName, _
                             (InStrRev(strImageFileName, ".") _
                              + Len(strImageFileName)) _
                              Mod (Len(strImageFileName) + 1)) & ".ccd"
            strSUBFileName = Left(strImageFileName, _
                             (InStrRev(strImageFileName, ".") _
                              + Len(strImageFileName)) _
                              Mod (Len(strImageFileName) + 1)) & ".sub"
        ElseIf (lngMediaFamily = MFT_DVDFAMILY) Then
            strTOCFileName = Left(strImageFileName, _
                             (InStrRev(strImageFileName, ".") _
                              + Len(strImageFileName)) _
                              Mod (Len(strImageFileName) + 1)) & ".cue"
            strSUBFileName = ""
        Else
            blnDone = False
            Call MsgBox("非対応のメディアファミリ")
        End If
    End If

    '※速度指定やエラー訂正モード等の変更を行うのであれば、この辺で実施する。

    'ファイルを開く
    hMainData = -1
    hSubData = -1
    If (blnDone) Then
        'CreateFile(FileName, GENERIC_READ, FILE_SHARE_NO, 0, CREATE_ALWAYS, 0, 0)
        hMainData = CreateFile(strImageFileName, &H40000000, &H0&, 0, 2, 0, 0)
        blnDone = (hMainData <> -1)
        If (strSUBFileName <> "") Then
            hSubData = CreateFile(strSUBFileName, &H40000000, &H0&, 0, 2, 0, 0)
            blnDone = blnDone And (hSubData <> -1)
        End If
        If (Not blnDone) Then
            Call MsgBox("ファイルが開けません。")
        End If
```

```
    End If

    '読み込み開始
    If (blnDone) Then
        If (lngMediaFamily = MFT_CDFAMILY) Then
            'CD では ReadCommand は[CCC_RAW/CCC_RAW16/CCC_RAW96]の何れか
            blnDone = (objDevice.ReadStart(CCC_RAW96, _
                        CDD_MAINDATA Or CDD_COLLECTSUBPQ) = CBR_OK)
            lngSectorSize = 2352 + 96 'CCC_RAW96 のときのブロック長
        Else
            'DVD では ReadCommand は[CCC_DATA]のみ
            blnDone = (objDevice.ReadStart(CCC_DATA, CDD_MAINDATA) = CBR_OK)
            lngSectorSize = 2048
        End If
        If (Not blnDone) Then
            Call MsgBox("ディスクの読み込みを開始できません。")
        End If
    End If

    '読み込み
    If (blnDone) Then
        'TOC 取得
        Set objTOC = objDevice.GetTOCStructure()

        'トラックループ
        For ixTrack = 1 To objTOC.TOCCount
            'トラック内の LBA ループ
            For lngLBA = objTOC.TOC(ixTrack).StartLBA _
            To objTOC.TOC(ixTrack).EndLBA
                '１フレーム読み込み
                If (Not objDevice.ReadLBA(bytData, lngLBA)) Then
                    'ReadLBA の戻り値でエラーセクタを判定可能。
                    '戻り値が False の場合、bytData に意味のあるデータは格納され
                    'ない。プロテクト等で態とエラーが入っている場合が多いため、
                    '無視するなり適当な処理を行えば良いが、SCSIErrorStatus が
                    'FFFFFFh の場合はコマンド実行エラーのため、強制終了しなけ
                    'ればならない。
                    If (objDevice.GetSCSIErrorStatus() = &HFFFFFF) Then
                        Call MsgBox(objDevice.GetSCSIErrorDescription())
                        GoTo LoopExit '◎ 多重ループ抜け
                    End If
                End If

                'ファイル書き込み
                If (lngSectorSize <= 2352) Then
                    lngLength = 0
                    Call WriteFile(hMainData, _
                                   VarPtr(bytData(0)), _
                                   lngSectorSize, _
                                   VarPtr(lngLength), 0)
                    blnDone = blnDone And (lngLength = lngSectorSize)
                Else
                    lngLength = 0
                    Call WriteFile(hMainData, _
                                   VarPtr(bytData(0)), _
                                   2352, _
                                   VarPtr(lngLength), 0)
                    blnDone = blnDone And (lngLength = 2352)
                    lngLength = 0
```

```
                        Call WriteFile(hSubData, _
                                       VarPtr(bytData(2352)), _
                                       96, _
                                       VarPtr(lngLength), 0)
                        blnDone = blnDone And (lngLength = 96)
                    End If
                    If (Not blnDone) Then
                        Call MsgBox("ファイル書き込みエラー")
                        GoTo LoopExit '◎ 多重ループ抜け
                    End If
                Next lngLBA
            Next ixTrack
LoopExit:

            'TOC 再取得・保存
            '※MCN/ISRC 取得のため、最新のデータを取得し直す必要がある
            Set objTOC = objDevice.GetTOCStructure()
            If (lngMediaFamily = MFT_CDFAMILY) Then
                Call objTOC.SaveCCDFile(strTOCFileName)
            Else
                Call objTOC.SaveCUEFile(strTOCFileName, strImageFileName, _
                                        "", lngSectorSize)
            End If

            '読み込み終了
            Call objDevice.ReadEnd
        End If

        'ファイルを閉じる
        If (hMainData <> -1) Then
            Call CloseHandle(hMainData)
        End If
        If (hSubData <> -1) Then
            Call CloseHandle(hSubData)
        End If

        '※エラー訂正モード等の変更を行ったのであれば、この辺で元に戻す

        'ディスク取り出し許可
        Call objDevice.LockUnlock(False)

        'デバイスを閉じる
        Call objDevice.CloseDevice
    Else
        Call MsgBox("デバイスが開けません。")
    End If

    Set objDevice = Nothing

End Function
```

3.5.2 ディスク書き込み例

CUE+IMG、CCD+IMG+SUB、または ISO ファイルを CD または DVD に書き込む例を次に示す。

```
Private Declare Function ReadFile Lib "kernel32" ( _
    ByVal hFile As Long, _
    ByVal lpBuffer As Long, _
```

```
    ByVal nNumberOfBytesToRead As Long, _
    ByVal lpNumberOfBytesRead As Long, _
    ByVal lpOverlapped As Long) As Long
Private Declare Function CreateFile Lib "kernel32" Alias "CreateFileA" ( _
    ByVal lpFileName As String, _
    ByVal dwDesiredAccess As Long, _
    ByVal dwShareMode As Long, _
    ByVal lpSecurityAttributes As Long, _
    ByVal dwCreationDisposition As Long, _
    ByVal dwFlagsAndAttributes As Long, _
    ByVal hTemplateFile As Long) As Long
Private Declare Function CloseHandle Lib "kernel32" (ByVal hObject As Long) As Long

'**** メイン
Sub Main()
    ' c:¥sample.ccd を D:ドライヴにセットされた CD/DVD に書き込む
    Call WriteDisc("c:¥sample.ccd","¥¥.¥D:")
End Sub

'*******************************************************************************
'*  名　前 : WriteDisc
'*  機　能 : ディスクを読み込み、ファイルに保存する
'*  引　数 : strTOCFileName ... 読み込む TOC/ISO ファイル名(.cue/.ccd/.iso)
'*         : strDeviceName .... 書き込みドライヴ名
'*         :  ("¥¥.¥Q:"、"¥¥192.168.0.2:8801¥Q:"など)
'*  備　考 : イメージ本体の他に、CD は CCD、DVD は CUE ファイルを出力する
'*******************************************************************************
Private Function WriteDisc( _
    ByVal strTOCFileName As String, _
    ByVal strDeviceName As String) As Boolean

    Dim objDevice      As Momiji
    Dim objTOC         As TOCInformation

    Dim strIMGFileName As String
    Dim strSUBFileName As String

    Dim lngMediaFamily As MediaFamilyTypeConstants
    Dim blnDone        As Boolean
    Dim lngReturn      As CDBIOSReturnConstants
    Dim ixTrack        As Integer
    Dim lngLBA         As Long
    Dim bytData(2352 + 96) As Byte
    Dim lngSectorSize  As Long
    Dim lngSectorData  As CDSectorDataConstants

    Dim hMainData As Long
    Dim hSubData  As Long
    Dim lngLength As Long

    Set objDevice = New Momiji
    Set objTOC = New TOCInformation

    'デバイスを開く
    If (objDevice.OpenDeviceEx(strDeviceName)) Then
        'ディスクが挿入されているかを確認
        blnDone = objDevice.IsReady()
        If (Not blnDone) Then
            Call MsgBox("デバイスの準備が出来ていません。")
```

```
End If

'ディスク取り出し禁止
Call objDevice.LockUnlock(True)

'TOC 情報をファイルから読み込む
'同時にイメージ本体、サブチャンネルファイル名、及びセクタサイズを取得
If (blnDone) Then
    Select Case Mid(strTOCFileName, lnStrRev(strTOCFileName, ".") + 1)
        Case "cue", "CUE"
            blnDone = objTOC.LoadCUEFile(strTOCFileName, _
                                         strIMGFileName, lngSectorSize)
            strSUBFileName = ""
        Case "ccd", "CCD"
            blnDone = objTOC.LoadCCDFile(strTOCFileName)
            strIMGFileName = Left(strTOCFileName, _
                                  (lnStrRev(strTOCFileName, ".") _
                                   + Len(strTOCFileName)) _
                                   Mod (Len(strTOCFileName) + 1)) & ".img"
            strSUBFileName = Left(strTOCFileName, _
                                  (lnStrRev(strTOCFileName, ".") _
                                   + Len(strTOCFileName)) _
                                   Mod (Len(strTOCFileName) + 1)) & ".sub"
            lngSectorSize = 2352 + 96
        Case "iso", "ISO"
            blnDone = objTOC.LoadISOFile(strTOCFileName, lngSectorSize)
            strIMGFileName = strTOCFileName
            strSUBFileName = ""
    End Select
    If (Not blnDone) Then
        Call MsgBox("TOC 情報の読み込みに失敗しました。")
    End If
End If

'※速度指定等の変更を行うのであれば、この辺で実施する。

'ファイルを開く
hMainData = -1
hSubData = -1
If (blnDone) Then
    'CreateFile(FileName,GENERIC_READ,FILE_SHARE_READ,0,OPEN_EXISTING,0,0)
    hMainData = CreateFile(strIMGFileName, &H80000000, &H1&, 0, 3, 0, 0)
    blnDone = (hMainData <> -1)
    If (strSUBFileName <> "") Then
        hSubData = CreateFile(strSUBFileName, &H80000000, &H1&, 0, 3, 0, 0)
        blnDone = blnDone And (hSubData <> -1)
    End If
    If (Not blnDone) Then
        Call MsgBox("ファイルが開けません。")
    End If
End If

'書き込み開始
If (blnDone) Then
    'TOC 設定
    Call objDevice.SetTOCStructure(objTOC)

    'メディアファミリ取得
    lngMediaFamily = objDevice.GetMediaFamilyType()
```

```
            If (lngMediaFamily = MFT_CDFAMILY) Then
                'CD では WriteCommand はドライヴが対応していればどれでも可
                lngReturn = objDevice.WriteStart(CCC_DAO_RAW96, _
                            CDD_MAINDATA Or CDD_COLLECTSUBPQ)
                blnDone = (lngReturn = CBR_OK)
            Else
                'DVD では WriteCommand は[CCC_SAO]のみ
                lngReturn = objDevice.WriteStart(CCC_SAO, CDD_MAINDATA)
                blnDone = (lngReturn = CBR_OK)
            End If
            If (Not blnDone) Then
                Select Case lngReturn
                    Case CBR_UNKNOWNMEDIA
                        Call MsgBox("書き込み非対応のメディアです。")
                    Case CBR_IGNOREWRITECOMMAND
                        Call MsgBox("非対応の書き込みコマンドが指定されました。")
                    Case CBR_NOTEMPTY
                        Call MsgBox("書き込み対象のメディアは空ではありません。")
                    Case CBR_RWFORMATERROR
                        Call MsgBox("RW メディアの初期化に失敗しました。")
                    Case CBR_OVERCAPACITY
                        Call MsgBox("書き込みサイズがメディアより大きい。")
                    Case Else
                        Call MsgBox("ディスクの書き込みを開始できません。")
                End Select
            End If
        End If

        '書き込み
        If (blnDone) Then
            'トラックループ
            For ixTrack = 1 To objTOC.TOCCount
                'トラック内の LBA ループ
                For lngLBA = objTOC.TOC(ixTrack).StartLBA _
                To objTOC.TOC(ixTrack).EndLBA
                    'ファイル読み込み
                    If (lngSectorSize = 2352 + 96) Then
                        lngSectorData = CSD_MAINSUBCHANNEL
                        lngLength = 0
                        Call ReadFile(hMainData, _
                                      VarPtr(bytData(0)), _
                                      2352, _
                                      VarPtr(lngLength), 0)
                        blnDone = blnDone And (lngLength = 2352)
                        lngLength = 0
                        Call ReadFile(hSubData, _
                                     VarPtr(bytData(2352)), _
                                     96, _
                                     VarPtr(lngLength), 0)
                        blnDone = blnDone And (lngLength = 96)
                    ElseIf (lngSectorSize = 2352) Then
                        lngSectorData = CSD_MAINCHANNEL
                        lngLength = 0
                        Call ReadFile(hMainData, _
                                      VarPtr(bytData(0)), _
                                      lngSectorSize, _
                                      VarPtr(lngLength), 0)
                        blnDone = blnDone And (lngLength = lngSectorSize)
                    ElseIf (lngSectorSize < 2352) Then
```

```
                        lngSectorData = CSD_SECTORONLY
                        lngLength = 0
                        Call ReadFile(hMainData, _
                                      VarPtr(bytData(0)), _
                                      lngSectorSize, _
                                      VarPtr(lngLength), 0)
                        blnDone = blnDone And (lngLength = lngSectorSize)
                    Else
                        blnDone = False
                    End If
                    If (Not blnDone) Then
                        Call MsgBox("ファイル読み込みエラー")
                        GoTo LoopExit '◎ 多重ループ抜け
                    End If

                    '１フレーム書き込み
                    blnDone = objDevice.WriteLBA(lngSectorData, bytData, lngLBA)
                    If (Not blnDone) Then
                        Call MsgBox("書き込みエラー " & _
                                    objDevice.GetSCSIErrorDescription())
                        GoTo LoopExit '◎ 多重ループ抜け
                    End If
                Next lngLBA
            Next ixTrack
LoopExit:

            '書き込み終了
            Call objDevice.WriteEnd(Not blnDone) 'エラー発生時はアボート
        End If

        'ファイルを閉じる
        If (hMainData <> -1) Then
            Call CloseHandle(hMainData)
        End If
        If (hSubData <> -1) Then
            Call CloseHandle(hSubData)
        End If

        '※速度情報等の変更を行ったのであれば、この辺で元に戻す

        'ディスク取り出し許可
        Call objDevice.LockUnlock(False)

        '成功時はディスク取り出し
        If (blnDone) Then
            Call objDevice.LoadTray(False)
        End If

        'デバイスを閉じる
        Call objDevice.CloseDevice
    Else
        Call MsgBox("デバイスが開けません。")
    End If

    Set objTOC = Nothing
    Set objDevice = Nothing

End Function
```

保護用紙