画像アクセス
ライブラリマニュアル

Ver.4.05

April 2010

株式会社マイクロビジョン

［目次］

［１］動作環境 1

［２］開発環境 1

［３］画像アクセスライブラリ 2

３．１．関数一覧表 2

３．２．関数の説明 4

1) MvcDib_Create 4
2) MvcDib_CreateEx 5
3) MvcDib_Destroy 6
4) MvcDib_Open 7
5) MvcDib_OpenToDib 8
6) MvcDib_Save 9
7) MvcDib_SaveRect 10
8) MvcDib_Resize 11
9) MvcDib_SeekBmpAddress 12
10) MvcDib_GetXsize 13
11) MvcDib_GetYsize 14
12) MvcDib_GetXsizeAlign 15
13) MvcDib_GetAlign 16
14) MvcDib_BitCount 17
15) MvcDib_IsColor 18
16) MvcDib_IsRle 19
17) MvcDib_IsDibEx 20
18) MvcDib_GetDibHeaderAddress 21
19) MvcDib_GetLutAddress 22
20) MvcDib_Paint 23
21) MvcDib_Paint2 24
22) MvcDib_Paint3 25
23) MvcDib_SetLut 26
24) MvcDib_GetLut 27
25) MvcDib_FillColor 28
26) MvcDib_FillColor2 29
27) MvcDib_Reverse 30
28) MvcDib_Crop 31
29) MvcDib_CropEx 32
30) MvcDib_Compose 33
31) MvcDib_SetCropInfo 34
32) MvcDib_GetCropInfo 35
33) MvcDib_GrayFromColor 36
34) MvcDib_Replice 37
35) MvcDib_Shrink 38

36) MvcDib_ShrinkEx 39
37) MvcDib_LineWrite 40
38) MvcDib_CircleWrite 41
39) MvcDib_RectangleWrite 42
40) MvcDib_DotWrite 43

［１］動作環境

PC本体	：PC/AT互換機
PCIチップセット	：TritonⅡ以上
CPU	：PentiumⅢ８50MHz以上
	(推奨　PentiumⅣ1.4GHz以上)
PCI-BIOS	：PCI-BIOS 2.1以上
メモリ	：128MBytes以上
	(推奨　256MBytes以上)
対応 OS	：Microsoft Windows 2000 (SP4 以降)
	：Microsoft Windows XP
	：Microsoft Windows Vista (32bit)
	：Microsoft Windows 7 (32bit/ 64bit)
	※いずれも日本語版

［２］開発環境

●コンパイラ

Microsoft Visual Studio 2005 以降

●ライブラリファイル

画像アクセスライブラリ	・・・MvcDib.lib
上記ライブラリのヘッダファイル	・・・MvcDib.h
上記ライブラリのDLLファイル	・・・MvcDib.dll

［３］画像アクセスライブラリ

３．１．関数一覧表

画像アクセスライブラリで使用できる画像処理関数の一覧表です。

1) MvcDib_Create	ＤＩＢ生成
2) MvcDib_CreateEx	ＤＩＢ生成
3) MvcDib_Destroy	ＤＩＢ解放
4) MvcDib_Open	画像ファイル読込処理
5) MvcDib_OpenToDib	画像ファイル読込処理
6) MvcDib_Save	画像ファイル保存処理
7) MvcDib_SaveRect	画像ファイル保存処理（範囲指定付）
8) MvcDib_Resize	ＤＩＢのサイズ変更処理
9) MvcDib_SeekBmpAddress	ＤＩＢの先頭アドレス取得
10) MvcDib_GetXsize	ＤＩＢのＸサイズ取得
11) MvcDib_GetYsize	ＤＩＢのＹサイズ取得
12) MvcDib_GetXsizeAlign	画像１行分のバイト数取得
13) MvcDib_GetAlign	画像のＸ方向のアライメント単位取得
14) MvcDib_BitCount	１画素のＢＩＴ数取得
15) MvcDib_IsColor	カラー画像判定処理
16) MvcDib_IsRle	圧縮画像判定処理
17) MvcDib_IsDibEx	拡張ＤＩＢ画像判定処理
18) MvcDib_GetDibHeaderAddress	BITMAPINFOHEADERのアドレス取得
19) MvcDib_GetLutAddress	LUTのアドレス取得
20) MvcDib_Paint	画像描画処理（案１）
21) MvcDib_Paint2	画像描画処理（案２）
22) MvcDib_Paint3	画像描画処理（案３）
23) MvcDib_SetLut	画像表示時のLUT設定
24) MvcDib_GetLut	画像表示時のLUT取得
25) MvcDib_FillColor	画像塗りつぶし処理
26) MvcDib_FillColor2	画像塗りつぶし処理（カラー対応）
27) MvcDib_Reverse	画像上下反転処理
28) MvcDib_Crop	画像切り抜き処理
29) MvcDib_Crop2	画像切り抜き処理（４点指定）
30) MvcDib_Compose	画像コピー処理
31) MvcDib_SetCropInfo	画像切り抜きモード設定処理
32) MvcDib_GetCropInfo	画像切り抜きモード取得処理
33) MvcDib_GrayFromColor	画像変換処理（カラー→白黒）
34) MvcDib_Replice	画像複製処理
35) MvcDib_Shrink	画像シュリンク処理

36) MvcDib_ShrinkEx　画像シュリンク処理（拡張版）
37) MvcDib_LineWrite　画像への線データ書き込み処理
38) MvcDib_CircleWrite　画像への円形データ書き込み処理
39) MvcDib_RectangleWrite　画像への矩形データ書き込み処理
40) MvcDib_DotWrite　画像への点データ書き込み処理

３．２．関数の説明

本節では、画像アクセスライブラリ関数の使用方法を説明します。使用方法のサンプルプログラムは、MFCコードです。

****

1) MvcDib_Create

HANDLE MvcDib_Create(int x, int y, int mode, int Align)

戻り値

0 以外: 画像データのハンドル　0:異常終了

引数

x	画像サイズ（横）
y	画像サイズ（縦）
mode	１画素のサイズ指定（バイト単位で指定）
Align	アライメント単位(バイト単位で指定)　※現在は 4 のみ対応

説明

画像メモリの生成を行います。

例）この関数で作成した画像メモリは、必ずアプリケーション終了時までには MvcDib_Destroy 関数で解放して下さい。

```
BOOL CMvc_DibApp::CreateDIB()
{
    inr ImgAcqX = 300;
    int ImgAcqY = 300;

    // 画像サイズ 640x480, 1 画素=1BYTE
    // 画像 1 ライン 4BYTE 単位
    hDIB = MvcDib_Create( ImgAcqX, ImgAcqY, 1, 4 );
            :
        (省略)
            :

    // 画像メモリ解放
    MvcDib_Destroy (hDIB);

    return TRUE;
}
```

****

2) MvcDib_CreateEx

HANDLE MvcDib_CreateEx(LPVOID pAdr, int x, int y, int mode, int Align)

戻り値

0 以外: 画像データのハンドル　0:異常終了

引数

pAdr	画像アドレス(仮想アドレス)
x	画像サイズ（横）
y	画像サイズ（縦）
mode	１画素のサイズ指定（バイト単位で指定)
Align	アライメント単位(バイト単位で指定)　※現在は 4 のみ対応

説明

画像ハンドルの生成を行います。

生成した画像ハンドルは、引数で指定した画像先頭アドレス pAdr と関連付けられます。

この関数で作成した画像メモリは、必ずアプリケーション終了時までには MvcDib_Destroy 関数で解放して下さい。

****

3) MvcDib_Destroy

void MvcDib_Destroy(HANDLE hDib)

戻り値

なし

引数

hDib　　画像データのハンドル

説明

画像メモリの解放を行います。

例)

```
BOOL CMvc_DibApp::OnCreateDIB()
{
    inr ImgAcqX = 300;
    int ImgAcqY = 300;

    // 画像サイズ 640x480, 1 画素=1BYTE
    // 画像 1 ライン 4BYTE 単位
    hDIB = MvcDib_Create( ImgAcqX, ImgAcqY, 1, 4);
            :
        (省略)
            :

    // 画像メモリ解放
    MvcDib_Destroy (hDIB);

return TRUE;
}
```

****

4) MvcDib_Open

HANDLE MvcDib_Open(char *Fname)

戻り値

0 以外: 画像データのハンドル　0:異常終了

引数

*Fname　　画像ファイルのパス（絶対パス）

説明

画像をＢＭＰ形式のファイルから読み込みます。

例）この関数で作成した画像メモリは、必ずアプリケーション終了時までには MvcDib_Destroy 関数で解放して下さい。

```
BOOL CMVcDoc::OnOpenDocument( LPCTSTR lpszPathName )
{
	char str[256];
	strcpy(str,lpszPathName);

	BeginWaitCursor();

	HANDLE hDIB = MvcDib_Open( str );
	if (!hDIB){
		AfxMessageBox(“画像読み込みエラー”);
		return FALSE;
	}

	EndWaitCursor();

	return TRUE;
}
```

****

5) MvcDib_OpenToDib

int MvcDib_OpenToDib(char *Fname, HANDLE hDib)

戻り値

0:正常終了　1:異常終了

引数

*Fname　　画像ファイルのパス（絶対パス）

hDib　　画像データのハンドル

説明

画像をファイルから読み込み、画像メモリにコピーします。

```
BOOL CMVcDoc::OnOpenDocument( LPCTSTR lpszPathName )
{
	char str[256];
	strcpy(str,lpszPathName);

	BeginWaitCursor();

	HANDLE hDIB = MvcDib_Create( ImgAcqX, ImgAcqY, 1, 4);

	int iRet = MvcDib_OpenToDib( str, hDIB );
	if (iRet){
		AfxMessageBox(“画像読み込みエラー”);
		return FALSE;
	}

	EndWaitCursor();

	return TRUE;
}
```

****

6) MvcDib_Save

BOOL MvcDib_Save(char *Fname, HANDLE hDib)

戻り値

TRUE:正常終了　FALSE:異常終了

引数

*Fname　　画像ファイルのパス（絶対パス）

hDib　　保存したい画像ハンドル

説明

画像を BMP 形式に変換して保存します。

例）

```
BOOL CMVcDoc::OnSaveDocument( LPCTSTR lpszPathName )
{
	BOOL bRet = FALSE;
	char str[256];

	strcpy(str, lpszPathName );

	bRet = MvcDib_Save( str, hDIB );
	if (bRet == FALSE){
		AfxMessageBox(“画像保存エラー”);
		return FALSE;
	}

	return TRUE;
}
```

****

7) MvcDib_SaveRect

BOOL MvcDib_SaveRect(char *Fname, HANDLE hDib, RECT *lprc)

戻り値

TRUE:正常終了　FALSE:異常終了

引数

*Fname	画像ファイルのパス（絶対パス）
hDib	保存したい画像ハンドル
*lprc	保存範囲

説明

指定した範囲内の画像をファイルに保存します。

例）

```
BOOL CMVcDoc::OnSaveDocument( LPCTSTR lpszPathName )
{
	BOOL bRet = FALSE;
	RECT rc = {0,0,100,100}
	char str[256];

	strcpy(str, lpszPathName );

	bRet = MvcDib_SaveRect( str, hDIB, &rc );
	if (bRet == FALSE){
		AfxMessageBox(“画像保存エラー”);
		return FALSE;
	}

	return TRUE;
}
```

****

8) MvcDib_Resize

HANDLE MvcDib_Resize(HANDLE hDib, int x, int y)

戻り値

0 以外:サイズ変更後の画像データのハンドル　0:異常終了

引数

hDib　画像データのハンドル
x　変更後の画像サイズ（横）
y　変更後の画像サイズ（縦）

説明

画像サイズの変更を行います。

例）

```
BOOL CMVcDoc::OnResizeDIB()
{
	HANDLE hNewDIB;
	inr ImgAcqX = 200;
	int ImgAcqY = 200;

	hNewDIB = MvcDib_Resize ( hDIB, ImgAcqX, ImgAcqY );
	if (!hNewDIB){
		AfxMessageBox(“画像サイズの変更エラー”);
		return FALSE;
	}

	return TRUE;
}
```

****

9) MvcDib_SeekBmpAddress

LPVOID MvcDib_SeekBmpAddress(HANDLE hDib)

戻り値

0 以外:画像データの先頭アドレス　0:異常終了

引数

hDib　画像データのハンドル

説明

画像データの先頭アドレスを取得します。

例)

```
void GetBmpAdress( HANDLE hDIB )
{
	LPVOID pAdr;

	pAdr = MvcDib_SeekBmpAddress ( hDIB );
	if ( !pAdr ) {
		AfxMessageBox(“画像データの先頭アドレス取得エラー”);
		return;
	}
}
```

****

10) MvcDib_GetXsize

int MvcDib_GetXsize(HANDLE hDib)

戻り値

0 以外:ＤＩＢサイズ（横）　0:異常終了

引数

hDib　　画像データのハンドル

説明

画像データのＸサイズを取得します。

例）

```
BOOL CMvc_DibApp::GetDIBLength ( HANDLE hDIB, int *x, int *y )
{
	if (!hDIB)
		return FALSE;

	*x = MvcDib_GetXsize( hDIB );		// X サイズ
	*y = MvcDib_GetYsize( hDIB );		//  Ｙサイズ

	return TRUE;
}
```

****

11) MvcDib_GetYsize

int MvcDib_GetYsize(HANDLE hDib)

戻り値

0 以外:ＤＩＢサイズ（縦）　0:異常終了

引数

hDib　　画像データのハンドル

説明

画像データのＹサイズを取得します。

例）

```
BOOL CMvc_DibApp::GetDIBLength ( HANDLE hDIB, int *x, int *y )
{
	if (!hDIB)
		return FALSE;

	*x = MvcDib_GetXsize( hDIB );		// X サイズ
	*y = MvcDib_GetYsize( hDIB );		//  Ｙサイズ

	return TRUE;
}
```

****

12) MvcDib_GetXsizeAlign

int MvcDib_GetXsizeAlign(HANDLE hDib)

戻り値

0 以外:バイト数　0:異常終了

引数

hDib　　画像データのハンドル

説明

画像 1 行分のバイト数を取得します。(通常 4 バイトアライメント考慮)

例)

```
int ImgToDib(HANDLE hDib)
{
        :
    (省略)
        :
    int iAlign = MvcDib_GetXsizeAlign( hDib );
    return iAlign;
}
```

****

13) MvcDib_GetAlign

int MvcDib_GetAlign(HANDLE hDib)

戻り値

アライメント単位

引数

hDib　　画像データのハンドル

説明

画像のＸ方向のアライメント単位を取得します。

例）

```
HANDLE  CreateCompatibleDIB( HANDLE hDib, int x, int y )
{
    // アライメント取得
    int Align = MvcDib_GetAlign(hDib);

    //  １画素のＢＩＴ数取得
    int ByteCount = MvcDib_BitCount(hDib) / 8;

    // ＤＩＢの複製作成
    HANDLE hNew = MvcDib_Create(x, y, ByteCount, Align);

    return hNew;
}
```

****

14) MvcDib_BitCount

int MvcDib_BitCount(HANDLE hDib)

戻り値

0 以外:正常終了　0:異常終了

引数

hDib　　画像データのハンドル

説明

１画素のＢＩＴ数を取得します。

例)

```
HANDLE  CreateCompatibleDIB( HANDLE hDib, int x, int y )
{
	// アライメント取得
	int Align = MvcDib_GetAlign(hDib);

	//  １画素のＢＩＴ数取得
	int ByteCount = MvcDib_BitCount(hDib) / 8;

	// ＤＩＢの複製作成
	HANDLE hNew = MvcDib_Create(x, y, ByteCount, Align);

	return hNew;
}
```

****

15) MvcDib_IsColor

BOOL MvcDib_IsColor(HANDLE hDib)

戻り値

TRUE:カラー画像　FALSE:白黒画像

引数

hDib　　画像データのハンドル

説明

カラー画像かどうかの判定を行ないます。

例)

```
BOOL CMVcDoc::DIBCheck( HANDLE hDib )
{
	// カラー画像判定
	if (MvcDib_IsColor(hDib)){
		AfxMessageBox("カラー画像です。");
		return FALSE;
	}

	return TRUE;
}
```

****

16) MvcDib_IsRle

BOOL MvcDib_IsRle(HANDLE hDib)

戻り値

TRUE: RLE 形式の画像　FALSE: RLE 形式以外の画像

引数

hDib　　画像データのハンドル

説明

圧縮画像かどうかの判定を行ないます。

例)

```
BOOL CMVcDoc::DIBCheck( HANDLE hDib )
{
	// RLE 形式の画像判定
	if (MvcDib_IsRle(hDib)){
		AfxMessageBox("RLE 形式の画像です。");
		return FALSE;
	}

	return TRUE;
}
```

****

17) MvcDib_IsDibEx

BOOL MvcDib_IsDibEx(HANDLE hDib)

戻り値

TRUE:DIB 画像　FALSE:DIB 画像以外

引数

hDib　　画像データのハンドル

説明

拡張ＤＩＢ画像かどうかの判定を行ないます。

例）

```
BOOL CMVcDoc::DIBCheck(HANDLE hDib)
{
	// 拡張ＤＩＢ画像判定
	if (MvcDib_IsDibEx(hDib)){
		AfxMessageBox("拡張ＤＩＢ画像です。");
		return FALSE;
	}

	return TRUE;
}
```

****

18) MvcDib_GetDibHeaderAddress

LPVOID MvcDib_GetDibHeaderAddress(HANDLE hDib)

戻り値

NULL 以外:正常終了　NULL:異常終了

引数

hDib　画像データのハンドル

説明

BITMAPINFOHEADER 構成体へのアドレスを取得します。

例)

```
void GetDibHeaderAdress(HANDLE hDIB)
{
	LPVOID pAdr;

	pAdr = MvcDib_GetDibHeaderAddress ( hDIB );
	if ( !pAdr ) {
		AfxMessageBox( “BITMAPINFOHEADER 取得エラー” );
		return;
	}
}
```

****

19) MvcDib_GetLutAddress

LPVOID MvcDib_GetLutAddress(HANDLE hDib)

戻り値

NULL 以外:正常終了　NULL:異常終了

引数

hDib　画像データのハンドル

説明

LUT のアドレスを取得します。

例)

```
void GetLutAdress(HANDLE hDIB)
{
	LPVOID pAdr;

	pAdr = MvcDib_GetLutAddress( hDIB );
	if( !pAdr ) {
		AfxMessageBox(“画像 LUT 取得エラー”);
		return;
	}
}
```

****

20) MvcDib_Paint

```
BOOL MvcDib_Paint( HDC hDC, HANDLE hDib,
                   int DstLeft, int DstTop, int DstRight, int DstBottom,
                   int SrcLeft, int SrcTop, int SrcRight, int SrcBottom )
```

戻り値

0 以外:正常終了　0:異常終了

引数

hDC	ウィンドウのハンドル
hDib	画像データのハンドル
DstLeft	表示先開始位置（左）
DstTop	表示先開始位置（上）
DstRight	表示先終了位置（右）
DstBottom	表示先終了位置（下）
SrcLeft	表示元開始位置（左）
SrcTop	表示元開始位置（上）
SrcRight	表示元終了位置（右）
SrcBottom	表示元終了位置（下）

説明

画像データを指定されたウィンドウに拡大（縮小）して描画します。

例）

```
void CAppDibView::DibPaint(CDC* pDC, HANDLE hDib)
{
    // 画像サイズ
    int x = MvcDib_GetXsize(hDib);
    int y = MvcDib_GetYsize(hDib);

    // 画像表示範囲
    RECT    rcDest, rcSrc;
    rcDest.left = rcDest.top = 0;
    rcSrc.left = rcSrc.top = 0;
    rcDest.right = (long)(DispRatioX * x) - 1;
    rcDest.bottom = (long)(DispRatioY * y) - 1;
    rcSrc.right = x - 1;
    rcSrc.bottom = y - 1;
    if ( !MvcDib_Paint( pDC->m_hDC, hDib,
                        rcDest.left, rcDest.top, rcDest.right, rcDest.bottom,
                        rcSrc.left, rcSrc.top, rcSrc.right, rcSrc.bottom == FALSE ){
        AfxMessageBox("描画エラー");
        break;
    }
}
```

****

21) MvcDib_Paint2

BOOL MvcDib_Paint2(HDC hDC, HANDLE hDib, RECT *rcDest, RECT *rcSrc)

戻り値

0 以外:正常終了　0:異常終了

引数

hDC	ウィンドウのハンドル
hDib	画像データのハンドル
*rcDest	表示先位置
*rcSrc	表示元位置

説明

画像データを指定されたウィンドウに拡大（縮小）して描画します。

例)

```
void CAppDibView::DibPaint(CDC* pDC, HANDLE hDib)
{
	// 画像サイズ
	int x = MvcDib_GetXsize(hDib);
	int y = MvcDib_GetYsize(hDib);

	// 画像表示範囲
	RECT	rcDest, rcSrc;
	rcDest.left = rcDest.top = 0;
	rcSrc.left = rcSrc.top = 0;
	rcDest.right = (long)(DispRatioX * x) - 1;
	rcDest.bottom = (long)(DispRatioY * y) - 1;
	rcSrc.right = x - 1;
	rcSrc.bottom = y - 1;
	if ( !MvcDib_Paint2(pDC->m_hDC, hDib, &rcDest, &rcSrc) == FALSE ){
		AfxMessageBox("描画エラー");
		break;
	}
}
```

****

22) MvcDib_Paint3

BOOL MvcDib_Paint3(MVCDIB_VIEWPAR *mdv)

戻り値

0 以外:正常終了　0:異常終了

引数

*mdv　　表示パラメータ

説明

画像データを指定されたウィンドウに拡大（縮小）して描画します。

例)

※あらかじめ定義をしておく。

```
typedef struct {
	HDC hDC;                    // 表示対象のウィンドウハンドル
	HANDLE hDib;                // 表示対象の DIB 画像
	int DstLeft,DstTop;         // 表示先開始位置
	int DstRight,DstBottom;     // 表示先終了位置
	int SrcLeft,SrcTop;         // 表示元開始位置
	int SrcRight,SrcBottom;         // 表示元終了位置
} MVCDIB_VIEWPAR;

void CAppDibView::DibPaint(CDC* pDC, HANDLE hDib)
{
	MVCDIB_VIEWPAR	mdv;

	// 画像サイズ
	int x = MvcDib_GetXsize(hDib);
	int y = MvcDib_GetYsize(hDib);

	// 画像表示範囲
	mdv.hDC = pDC->m_hDC;
	mdv.hDib = hDib;
	mdv.DstLeft = mdv.DstTop = 0;
	mdv.DstRight = (long)(DispRatioX * x) - 1;
	mdv.DstBottom= (long)(DispRatioY * y) - 1;
	mdv.SrcLeft = mdv.SrcTop = 0;
	mdv.SrcRight = x-1;
	mdv.SrcBottom = y-1;
	if ( !MvcDib_Paint3( &mdv ) == FALSE ){
		AfxMessageBox("描画エラー");
		break;
	}
}
```

****

23) MvcDib_SetLut

BOOL MvcDib_SetLut(HANDLE hDib, DWORD *Lut)

戻り値

0 以外:正常終了　0:異常終了

引数

hDib　　画像データのハンドル

* Lut　　表示時の LUT

説明

画像表示時の２値ＬＵＴを設定します。

例）256 階調 の白黒画像の時のみ有効です。

```
void CMainFrame::OnDibSetLut()
{
	BOOL	bRet;
	int	dm;
	CString	ss;
	char	str[256];
	DWORD	dwLut[256];

	for (int i = 0; i < 256; i++) {
		ss.Format("SetLut %d",i);
		strcpy(str,ss);
		Read_IniFile_Int("DIB", str, &dm);
		dwLut[i] = dm;
	}

	bRet = MvcDib_SetLut(hDib, dwLut);
	if (bRet == FALSE)
		AfxMessageBox("MvcDib_SetLut Error",MB_OK);
}
```

****

24) MvcDib_GetLut

BOOL MvcDib_GetLut(HANDLE hDib, DWORD *Lut)

戻り値

0 以外:正常終了　0:異常終了

引数

hDib　　画像データのハンドル

* Lut　　表示時の LUT

説明

画像表示時のＬＵＴを取得します。

例）256 階調 の白黒画像の時のみ有効です。

```
void CMainFrame::OnDibGetLut()
{
	BOOL	bRet;
	int	dm;
	CString	ss;
	char	str[256];
	DWORD	dwLut[256];

	bRet = MvcDib_GetLut(hDib, dwLut);
	if (bRet == FALSE){
		AfxMessageBox("MvcDib_GetLut Error",MB_OK);
		return;
	}

	for (int i = 0; i < 256; i++) {
		ss.Format("GetLut %d",i);
		strcpy(str,ss);
		dm = (int)dwLut[i];
		Write_IniFile_Int("DIB", str, dm);
	}
}
```

****

25) MvcDib_FillColor

BOOL MvcDib_FillColor(HANDLE hDib, DWORD color)

戻り値

TRUE:正常終了　FALSE:異常終了

引数

hDib　　画像データのハンドル

color　　塗りつぶし色

説明

画像を指定された色で塗りつぶします。

例)

```
BOOL CMainFrame::OnDibFillColor( HANDLE hDIB ,DWORD color )
{
	BOOL bRet;

	bRet = MvcDib_FillColor ( hDIB, color );
	if( bRet == FALSE ){
		AfxMessageBox(“画像塗りつぶしエラー”);
		return FALSE;
	}

	return TRUE;
}
```

****

26) MvcDib_FillColor2

BOOL MvcDib_FillColor2(HANDLE hDib, DWORD color)

戻り値

TRUE:正常終了　FALSE:異常終了

引数

hDib　　画像データのハンドル

color　　塗りつぶし色(RGB 指定)

説明

画像を指定された色で塗りつぶします。

例)

```
BOOL CMainFrame::OnDibFillColor( HANDLE hDIB ,DWORD color )
{
	BOOL bRet;

	bRet = MvcDib_FillColor2 ( hDIB, RGB(255,0,0));
	if( bRet == FALSE ){
		AfxMessageBox(“画像塗りつぶしエラー”);
		return FALSE;
	}

	return TRUE;
}
```

****

27) MvcDib_Reverse

BOOL MvcDib_Reverse(HANDLE hDib)

戻り値

TRUE:正常終了　FALSE:異常終了

引数

hDib　　画像データのハンドル

説明

画像データを上下反転させます。

例)

```
BOOL CMainFrame::OnDibReverse( HANDLE hDIB )
{
	BOOL bRet;

	bRet = MvcDib_Reverse( hDIB );
	if( bRet == FALSE ){
		AfxMessageBox( “画像反転エラー” );
		return FALSE;
	}

	return TRUE;
}
```

****

28) MvcDib_Crop

HANDLE MvcDib_Crop(HANDLE hDib, RECT *rc)

戻り値

0 以外:切りぬいた画像データのハンドル　0:異常終了

引数

hDib　　画像データのハンドル

*rc　　切り抜き範囲

説明

画像データの指定された範囲の部分を切り抜き、新たに画像を生成します。

例）この関数で作成した画像メモリは、必ずアプリケーション終了時までには MvcDib_Destroy 関数で解放して下さい。

```
void CMainFrame::OnDibCrop( HANDLE hDIB, RECT *rc )
{
	HANDLE hNewDIB;

	hNewDIB = MvcDib_Crop( hDIB, &rc );
	if( !hNewDIB ){
		AfxMessageBox(“画像切り抜きエラー”);
		return;
	}
		:
	  (省略)
		:

	// 画像メモリ解放
	MvcDib_Destroy(hNewDIB);

	return;
}
```

****

29) MvcDib_CropEx

HANDLE MvcDib_CropEx(HANDLE hDib, POINT *p1, POINT *p2, POINT *p3, POINT *p4)

戻り値

0 以外:切りぬいた画像データのハンドル　0:異常終了

引数

hDib　　画像データのハンドル

p1- p4　　切り抜き範囲（p1→p2→p3→p4 と続いている事）

説明

画像データの指定された４点で囲まれたの部分を切り抜き、新たに画像を生成します。

４点で囲まれた領域以外は０データ（黒）になります。

****

30) MvcDib_Compose

BOOL MvcDib_Compose(HANDLE hDst, HANDLE hSrc)

戻り値

TRUE:正常終了　FALSE:異常終了

引数

hDst　　表示先画像データのハンドル

hSrc　　表示元画像データのハンドル

説明

画像データのコピーを行ないます。

例)

```
BOOL CMainFrame::OnDibCopy( HANDLE hDst, HANDLE hSrc )
{
	BOOL bRet;

	bRet = MvcDib_Compose ( hDst, hSrc );
	if( bRet == FALSE ){
		AfxMessageBox(“画像データのコピーエラー”);
		return FALSE;
	}

	return TRUE;
}
```

****

31) MvcDib_SetCropInfo

int MvcDib_SetCropInfo(int mode)

戻り値

現在の画像の切り抜きモード

引数

mode　　画像の切り抜きモード

0:指定ハンドルの画像をそのまま切り抜く(カラーならカラーにする)

1:指定ハンドルの画像の R 成分だけ切り抜く

2:指定ハンドルの画像の G 成分だけ切り抜く

3:指定ハンドルの画像の B 成分だけ切り抜く

説明

画像の切り抜きモードの設定をします。(対象画像が、カラー画像の場合のみ有効)

****

32) MvcDib_GetCropInfo

int MvcDib_GetCropInfo(void)

戻り値

現在の画像の切り抜きモード

0:指定ハンドルの画像をそのまま切り抜く(カラーならカラーにする)

1:指定ハンドルの画像の R 成分だけ切り抜く

2:指定ハンドルの画像の G 成分だけ切り抜く

3:指定ハンドルの画像の B 成分だけ切り抜く

引数

なし

説明

画像の切り抜きモードの取得をします。

****

33) MvcDib_GrayFromColor

int MvcDib_GrayFromColor(HANDLE hGray, HANDLE hColor, int iRGB)

戻り値

0:正常終了　-1:異常終了

引数

hGray	白黒画像データのハンドル
hColor	カラー画像データのハンドル
iRGB	対象色（0:赤　1:緑　2:青　3:RGB 成分から輝度作成）

説明

カラー画像を白黒画像に変換します。

****

34) MvcDib_Replice

HANDLE MvcDib_Replice(HANDLE hDib)

戻り値

0 以外:複製した画像データのハンドル　0:異常終了

引数

hDib　　元となる画像データのハンドル

説明

画像の複製を行います。

この関数で作成した画像メモリは、必ずアプリケーション終了時までには MvcDib_Destroy 関数で解放して下さい。

****

35) MvcDib_Shrink

BOOL MvcDib_Shrink(HANDLE hDib, int iShrink)

戻り値

TRUE:正常終了　FALSE:異常終了

引数

hDib　　表示対象の画像データのハンドル

iShrink　　シュリンクレベル（0:1/1　1:1/2　2:1/4　3:1/8　4:1/16）

説明

画像の圧縮変換を行います。

シュリンクレベル＝１の場合、縦サイズ、横サイズがそれぞれ１/2 のサイズに圧縮されます。

したがって、全体の画像サイズは 1/４になります。

****

36) MvcDib_ShrinkEx

BOOL MvcDib_ShrinkEx(HANDLE hDst, HANDLE hSrc, int iShrink)

戻り値

TRUE:正常終了　FALSE:異常終了

引数

hDst	コピー先の画像データのハンドル
hSrc	コピー元の画像データのハンドル
iShrink	シュリンクレベル（0:1/1　1:1/2　2:1/4　3:1/8　4:1/16）

説明

画像の圧縮変換を行い、変換した画像を別の画像メモリにコピーします。
シュリンクレベル＝１の場合、縦サイズ、横サイズがそれぞれ１/2 のサイズに圧縮されます。
したがって、全体の画像サイズは 1/4 になります。

****

37) MvcDib_LineWrite

int MvcDib_LineWrite(HANDLE hDib, int sx, int sy, int ex, int ey, int iValue)

戻り値

正常終了の時は、0 を返します。それ以外は、以下の値を返します。

1：画像ハンドルが NULL

2：8bit か 24bit 以外の画像データ

3：座標値エラー

引数

hDib	画像データのハンドル
sx	開始Ｘ座標
sy	開始Ｙ座標
ex	終了Ｘ座標
ey	終了Ｙ座標
iValue	データの階調値　※24bit の場合は、RGB(R, G, B)で指定

説明

画像メモリ上の指定された開始座標から終了座標まで線データを書き込みます。

****

38) MvcDib_CircleWrite

int MvcDib_CircleWrite(HANDLE hDib, int iCX, int iCY, int iR, int iValue)

戻り値

正常終了の時は、0 を返します。それ以外は、以下の値を返します。

1：画像ハンドルが NULL

2：8bit か 24bit 以外の画像データ

3：座標値エラー

引数

hDib	画像データのハンドル	
iCX	円の中心Ｘ座標	
iCY	円の中心Ｙ座標	
iR	円の半径	
iValue	データの階調値	※24bit の場合は、RGB(R, G, B)で指定

説明

画像メモリ上の指定された座標に円形データ(塗りつぶし)を書き込みます。

****

39) MvcDib_RectangleWrite

int MvcDib_RectangleWrite(HANDLE hDib, RECT *rcArea, int iValue)

戻り値

正常終了の時は、0 を返します。それ以外は、以下の値を返します。

1：画像ハンドルが NULL

2：8bit か 24bit 以外の画像データ

3：座標値エラー

引数

hDib	画像データのハンドル	
rcArea	矩形座標	
iValue	データの階調値	※24bit の場合は、RGB(R, G, B)で指定

説明

画像メモリ上の指定された座標に矩形データ(塗りつぶし)を書き込みます。

****

40) MvcDib_DotWrite

int MvcDib_DotWrite(HANDLE hDib, int iPosX, int iPosY, int iValue)

戻り値

正常終了の時は、0 を返します。それ以外は、以下の値を返します。

1：画像ハンドルが NULL

2：8bit か 24bit 以外の画像データ

3：座標値エラー

引数

hDib	画像データのハンドル	
iPosX	Ｘ座標	
iPosY	Ｙ座標	
iValue	データの階調値	※24bit の場合は、RGB(R, G, B)で指定

説明

画像メモリ上の指定された座標に点データを書き込みます。

株式会社マイクロビジョン

〒950-0986　新潟市中央区神道寺南1－9－15
TEL　025－247－8312
FAX　025－247－8313