

# バーコード（1次元）認識ライブラリ

Version 2.0

EXE版／DLL版

## 説明書





## はじめに

本書では、バーコード（1次元）認識ライブラリ（以降、認識ライブラリと記載）の取扱方法について解説します。

- ※ Microsoft、Windows、Visual Studio は、米国その他の国における米国 Microsoft Corporation の登録商標です。
- ※ その他、記載されている社名、製品名は、一般に各社の商標または登録商標です。本書内では、®、™ 等は明記していません。

Copyright (C) 2015 Media Drive Corporation.

Copyright (C) 1991-1998, Thomas G. Lane.

This software is based in part on the work of the Independent JPEG Group.

Open Source Computer Vision Library

Copyright (C) 2000-2006, Intel Corporation, all rights reserved.

## 目次

1. 仕様	5
2. 動作環境	7
3. 認識対象文字	9
4. フォルダ構成	10
5. インストール	11
6. アンインストール	13
7. 関数一覧 (DLL 版)	14
8. 関数概要 (DLL 版)	15
9. 関数呼び出しの流れ (DLL 版)	16
10. プログラム説明 (EXE 版)	17
11. 関数説明 (DLL 版)	25
12. エラーコード一覧 (DLL 版)	54
付録A. 認識サイズと解像度	57

## 1. 仕様

### バーコードの種類

種類	桁数	チェックデジット
JAN	8 または 13	必須： ・モジュラス 10/ウェイト 3
ITF	2～30（偶数桁のみ）	任意： ・なし（規定値） ・モジュラス 10/ウェイト 3
CODE39	3～30	任意： ・なし（規定値） ・モジュラス 43
CODE128	4～44	必須： ・モジュラス 103
NW-7	3～30	任意： ・なし（規定値） ・モジュラス 16 ・モジュラス 10/ウェイト 3 ・モジュラス 10/ウェイト 2 ・ルーンズ （モジュラス 10/ウェイト 2） ・モジュラス 11 ・加重モジュラス 11 ・7 チェック DR
カスタマバーコード	7～20	必須
GS1 DataBar Omnidirectional (RSS-14) *1	16	必須

\*1 スタック（多層型）、および、合成シンボルの読み取りには対応していません。

### バーコードの大きさ（カスタマバーコード以外）

コード幅	150mm 以下
コード高さ	8.0mm 以上
細バー（細スペース）*2 サイズ	2 画素以上
余白	3mm×[倍率] 以上

\*2 バーコードを構成する一番細い線またはスペース

### カスタマバーコードの寸法（大きさ）、印字位置（余白）、傾き

日本郵便・カスタマバーコードの仕様に従います。

日本郵便 > 郵便番号・バーコードマニュアル > バーコード

#### カスタマバーコードの寸法


<http://www.post.japanpost.jp/zipcode/zipmanual/p12.html>

※ 拡大・縮小したサイズの異なる複数のカスタマバーコードが混在する場合は、読み取りに失敗する場合があります。

#### 印字位置、傾き

<http://www.post.japanpost.jp/zipcode/zipmanual/p13.html>

## 入力／出力

入力画像形式	BMP、TIFF（非圧縮、G3/G4 圧縮、LZW 圧縮）、JPEG、PNG、PDF 白黒 2 値、グレー、カラーに対応 ※ TIFF の 1bit パレット画像には非対応 ※ マルチページ TIFF、PDF は指定ページのみ処理 ※ PDF ファイルを読み込む場合、 <ul style="list-style-type: none"> <li>➤ Adobe Acrobat 8 ～11 が別途必要となります。</li> <li>➤ ログイン状態が必須となります。非ログイン状態（サーバー運用）での PDF 読み込みには対応していません。</li> <li>➤ PDF の作成ツール、作成手順、内部のデータ構造によっては、PDF の読み込みに失敗する場合があります*。</li> </ul>
画像サイズ	スキャン画像：最大 A 3 以下（PDF 入力時は A 4 以下） 最大解像度 600dpi（300dpi 以上を推奨） カメラ画像：500 万画素以上で撮影した画像
画像の回転	90, 180, 270 度
画像の傾き	水平方向に走査したときにコードの開始から終了までを通過することが可能な角度。 
出力方式	DLL 版はメモリ出力、EXE 版は標準出力 ※ JAN、ITF、CODE39、NW-7、GS1 DataBar Omnidirectional (RSS-14) については、チェックデジットをデータの一部として常に出力します。 ※ CODE39 については、スタート／ストップキャラクタ (*) をデータの一部として常に出力します。 ※ NW-7 については、スタート／ストップキャラクタ (A/B/C/D) をデータの一部として常に出力します。
その他	バーコードの色は黒色（背景は白色）を推奨

## 誤読や読取失敗を減らすためのヒント：

種別、桁数、読取方向を限定することで誤読や不要なコードの読み取りを減らすことができます。

また、チェックデジットを含むバーコードに対しては、チェック式を有効にすることで誤読や不要なコードの読み取りを減らすことができます。

\* ヒント：PDF の読み込みに失敗する場合

作成ツール、手順 等を変えてお試しください。

Acrobat の場合、「印刷」メニューから「互換性のある形式」で Acrobat バージョンを個別指定して作成することにより、読取可能となることがあります。

## 2. 動作環境

対応 OS	<p>Windows 8.1 / Windows 8.1 Enterprise / Pro  Windows 8 / Windows 8.1 Enterprise / Pro  Windows 7 Ultimate / Enterprise / Professional /  Home Premium / Starter  Windows Vista Ultimate / Enterprise / Business /  Home Premium / Home Basic  Windows Server 2012 R2 Standard  Windows Server 2012 Standard  Windows Server 2008 R2 Standard / Enterprise  Windows Server 2008 Standard / Enterprise  各日本語版に対応</p> <p>※ 64bit 版 Windows Vista / Windows Server 2008、Windows 7 の XP モード、Macintosh は動作保証外。  ※ 64bit 版 Windows 7 / 8 / 8.1 Windows Server 2008 R2 / Windows Server 2012 / 2012 R2 では 32bit 互換モード(WOW64)で動作します。その他の 64bit 版 OS、Windows 7 の XP モード、Macintosh は動作保証外となります。また、Windows 8 / 8.1 / Server 2012 / 2012 R2 ではデスクトップアプリケーションにのみ対応しています（Windows RT は動作保証外）。</p>
対応機種	上記 OS が正常に動作する機種
開発環境	<p>Visual C++ 2008 / 2010 / 2012 / 2013  Visual Basic 2008 / 2010 / 2012 / 2013</p>
その他	<p>本ライブラリの使用にはコピープロテクタ（USB タイプ）またはオンライン認証（アクティベーション）の何れかが必要。  アクティベーションを行うためには、インターネット接続環境が必要。  ※ インターネットに接続できない場合、インターネットに接続した別のパソコンを使って代理認証が可能。</p>

## 対応仮想化環境について

対応商品	VMware vSphere Hypervisor(ESXi) 5.1,5.5 [ゲスト OS] Windows Server 2008 Standard/Enterprise (32bit) Windows Server 2008 R2 Standard/Enterprise Windows Server 2012 Standard Windows Server 2012 R2 Standard  Hyper-V 3.0 [ホスト OS] Windows Server 2012 Standard [ゲスト OS] Windows Server 2008 Standard/Enterprise (32bit) Windows Server 2008 R2 Standard/Enterprise Windows Server 2012 Standard  Hyper-V 3.1 [ホスト OS] Windows Server 2012 R2 Standard [ゲスト OS] Windows Server 2008 Standard/Enterprise (32bit) Windows Server 2008 R2 Standard/Enterprise Windows Server 2012 Standard Windows Server 2012 R2 Standard
------	---

## 注意事項：

1. Microsoft 社からダウンロード提供されている「Microsoft Hyper-V Server 2008 R2」、「Microsoft Hyper-V Server 2012」、「Microsoft Hyper-V Server 2012 R2」はサポート対象外になります。
2. 仮想化ソフトウェアはセットアップ時の初期状態で動作確認を行っています。仮想化ソフトウェアの操作や設定についてのお問い合わせは、各メーカーのサービスセンターにお問い合わせください。
3. サポート範囲は、実環境上において発生する問題のみの対応とします。仮想化環境において個別に発生する問題、たとえば、ライブマイグレーションやバックアップなど仮想化商品固有の機能を利用した場合の動作についてはサポート対象外となります。
4. 仮想化ソフトウェアのゲスト OS 単位で、アクティベーション、または、コピープロテクタ (HASP キー) の接続 (割り当て) がそれぞれ必要になります。※Hyper-V 3.0/3.1 については、アクティベーションのみ対応 (HASP 非対応)。
5. スキャナの利用は、サポート対象外になります。



### 3. 認識対象文字

認識ライブラリの認識対象文字は次の通りです。

JAN	数字（0～9）のみ
ITF	数字（0～9）のみ
CODE39	数字（0～9） アルファベット大文字 記号（-, ., スペース, \$, /, +, %） スタート／ストップキャラクタ（*：アスタリスク）
NW-7	数字（0～9） 記号（-, \$, :, /, ., +） スタート／ストップキャラクタ（a～d）
CODE128	アスキーコード全文字
カスタマバーコード	数字（0～9） アルファベット大文字 記号（-） ※先頭から7桁目までは数字のみ
GS1 DataBar Omnidirectional (RSS-14)	数字（0～9）のみ

#### 注意事項：

EXE 版の認識プログラム、および、DLL 版のサンプルプログラムでは、読取結果をそのまま出力しています。制御コード等、表示不可能な文字を含む場合、画面表示が乱れる場合があります。読取結果の取得後、必要に応じて文字コードのチェックを別途行なってください。

## 4. フォルダ構成

本ライブラリの CD 構成は次の通りです。

ファイル／フォルダ	説明
Activator	
Setup.exe	ライセンス認証プログラムのインストーラ
bin	
BcDecode.exe	バーコード（1次元）認識プログラム（EXE 版）
QrDecode.exe	QRコード認識プログラム（EXE 版）
*.dll	ダイナミックリンクライブラリ（EXE/DLL/OCX 版）
*.ocx	ActiveX コントロール（OCX 版）
doc	ドキュメント
driver	
HASP	HASP ドライバ
MDCPrinter2	PDF 読み取り用仮想プリンタドライバ（32bit）
MDCPrinter2_x64	PDF 読み取り用仮想プリンタドライバ（64bit）
include	
*.h	共有ライブラリヘッダ（DLL 版）
*.lic	デザインタイムライセンスファイル（OCX 版）
lib	
*.lib	オブジェクトファイルライブラリ（DLL 版）
*.tlb	タイプライブラリ（OCX 版）
runtime	
msxml6.msi	MSXML 6.0
sample	
Image	サンプル画像
Project	サンプルプログラム（DLL/OCX 版）
SettingSample.txt	サンプル設定（EXE 版）

## 5. インストール

認識ライブラリ本体には、インストーラはありません。

ライブラリを構成するファイルを環境変数 **PATH** で参照可能なフォルダ、または使用するアプリケーションプログラムから参照可能なフォルダへ配置してください。

### 5.1. コピープロテクタ

コピープロテクタ（ハードウェアプロテクタ、またはソフトウェアプロテクタのいずれか）  
ハードウェアプロテクタ・・・USB プロテクタ  
ソフトウェアプロテクタ・・・認証キー（ライセンス取得のための文字列）

#### (1) ハードウェアプロテクタ版の場合

本ライブラリに **USB** タイプのコピープロテクタが付属する場合、コピープロテクタ用の **HASP** デバイスドライバのインストールが必要となります。次のファイルを参照し、**HASP** デバイスドライバをインストールしてください。

CD 内： [driver¥HASP¥Readme \(HowToInstall\).txt](#)

#### 注意事項：

**HASP** デバイスドライバのインストールには、管理者権限が必要となります。

#### (2) ソフトウェアプロテクタ版の場合

認証プログラムと認証キーを用いてオンライン認証を行うことで本ライブラリが利用可能となります。オンライン認証については、本商品に付属の「**バーコード(1次元)認識ライブラリ／QR コード認識ライブラリ セットアップガイド**」または次のオンラインドキュメントを参照してください。

CD 内： [doc¥ソフトウェアプロテクトの利用方法.pdf](#)

### 5.2. 仮想プリンタドライバ

認識ライブラリで **PDF** 読み込みを行う場合は、専用の仮想プリンタ（**MDC Printer2**）が必要となります。次の仮想プリンタドライバを必要に応じてインストールしてください。

CD 内： [driver¥MDCPrinter2¥MDCPrinter2Setup.exe](#)

**注意事項：**

仮想プリンタドライバのインストール時において、

- **MDCPrinter2Setup.exe** のパスに 2 バイト文字列を含む場合、インストールに失敗する場合があります。

PDF 読み込みを行う場合、

- **Adobe Acrobat 8 ～11** が別途必要となります。
- 処理速度が通常より低下します。
- ログイン状態が必須となります。非ログイン状態（サーバー運用）での PDF 読み込みには対応していません。

## 6. アンインストール

配置したファイルを削除すれば、認識ライブラリ本体のアンインストールは完了です。

### 6.1. ドライバのアンインストール

CD 内の次のプログラムを用いてアンインストールを行ってください。

HASP ドライバ: `driver¥HASP¥remove.bat`

仮想プリンタドライバ: `driver¥MDCPrinter2¥MDCPrinterUnsetup.exe`

## 7. 関数一覧 (DLL 版)

認識ライブラリには、11 個の認識関数と 8 個の一般関数が存在します。

## 7.1. 認識関数一覧

## 【認識関数一覧】

動作	関数名
(1) バーコード認識を初期化する。	MdBcDecode_Initialize
(2) バーコード認識の終了処理をする。	MdBcDecode_Terminate
(3) 指定領域内のバーコードを認識する (スキャナ)	MdBcDecode_DecodeRect
(4) 指定領域内のバーコードを認識する (スキャナ／カメラ)	MdBcDecode_DecodeRectEx
(5) 画像内のバーコードを認識する (スキャナ)	MdBcDecode_Decode
(6) 画像内のバーコードを認識する (スキャナ／カメラ)	MdBcDecode_DecodeEx
(7) バーコード認識のオプションを設定する。	MdBcDecode_SetOption
(8) 認識データを取得する。	MdBcDecode_GetData
(9) モードを取得する。	MdBcDecode_GetMode
(10) バーコード認識領域を取得する。	MdBcDecode_GetRect
(11) 種別を取得する。	MdBcDecode_GetVersion

## 7.2. 一般関数一覧

## 【一般関数一覧】

動作	関数名
(1) 画像ファイル内頁数を取得する。	MdBcCommon_CountPage
(2) 画像ファイルを読み込む (スキャナ)	MdBcCommon_ReadFile
(3) 画像ファイルを読み込む (スキャナ／カメラ)	MdBcCommon_LoadImage
(4) DIB イメージを読み込む (スキャナ／カメラ)	MdBcCommon_ConvertDIBToImageHandle
(5) 画像ハンドルを解放する。	MdBcCommon_ReleaseImage
(6) 画像サイズを取得する。	MdBcCommon_GetImageSize
(7) 画像解像度を取得する。	MdBcCommon_GetImageDPI
(8) 画像を配列にコピーする。	MdBcCommon_GetImage

## 8. 関数概要 (DLL 版)

### 8.1. 認識関数概要

認識ライブラリでは、4 種類の認識関数を提供します。

- ① 認識エンジン呼び出すためのインスタンスを管理する関数  
`MdBcDecode_Initialize`、`MdBcDecode_Terminate`
- ② 認識の属性を設定する関数  
`MdBcDecode_SetOption`
- ③ 認識を実行する関数  
`MdBcDecode_DecodeRect`、`MdBcDecode_Decode`  
`MdBcDecode_DecodeRectEx`、`MdBcDecode_DecodeEx`
- ④ 認識結果を取得する関数  
`MdBcDecode_GetData`、`MdBcDecode_GetMode`、`MdBcDecode_GetRect`、  
`MdBcDecode_GetVersion`

### 8.2. 一般関数概要

認識ライブラリでは、3 種類の一般関数を提供します。

- ① 画像ファイルの情報を取得する関数  
`MdBcCommon_CountPage`
- ② 画像を扱うためのインスタンスを管理する関数  
`MdBcCommon_ReadFile`、  
`MdBcCommon_LoadImage`、`MdBcCommon_ConvertDIBToImageHandle`、  
`MdBcCommon_ReleaseImage`
- ③ 画像の情報を取得する関数  
`MdBcCommon_GetImageSize`、`MdBcCommon_GetImageDPI`、`MdBcCommon_GetImage`

## 9. 関数呼び出しの流れ (DLL 版)

関数の具体的な用途は次の通りです。

### 9.1. 初期化・終了

認識エンジンを読み出すにはまず `MdBcDecode_Initialize` で初期化 (インスタンス生成) を行います。

不要になったインスタンスは、`MdBcDecode_Terminate` で終了 (破棄) します。

### 9.2. 認識

認識には、`MdBcDecode_Decode`、または、`MdBcDecode_DecodeEx` を呼び出します。

### 9.3. 結果取得

認識結果は、`MdBcDecode_GetData` で取得します。

### 9.4. 結果捕捉情報取得

結果補足情報は、`MdBcDecode_GetMode`、`MdBcDecode_GetVersion`、  
で必要に応じて取得します。



## 10. プログラム説明 (EXE 版)

プログラム名	BcDecode.exe
--------	--------------

概要
バーコード（1次元）を認識する。

実行形式
<p>BcDecode [画像パス名] [/I=入力フォルダ* [/E=画像種類]]</p> <p>[/BIN=2 値化種別] [/NOISE=ノイズサイズ] [/HOLE=孔サイズ]</p> <p>[/U=単位] [/X=解像度 /Y=解像度]</p> <p>[/P=処理頁] [/Max=最大数] [/A=処理矩形]</p> <p>[/D=読取方向] [/B=種別] [/L=桁数] [/C=チェック式]</p> <p>[/F=出力書式] [/O=出力ファイル [/W=上書き]]</p> <p>[/S=設定ファイル]</p>

引数の説明			
コマンドライン引数：			
画像パス名			
認識する画像ファイルのパス名（*.bmp, *.tif, *.jpg, *.png, *.pdf）を指定。			
/I="フォルダ"			
フォルダ内ファイル一括処理を行う場合に指定。			
/E=画像種類			
/I 指定時のファイル拡張子フィルタ。指定した拡張子のファイルのみ処理を行う。			
tif	TIFF		
bmp	Bitmap		
jpg	JPEG		
png	PNG		
pdf	PDF		
複数指定可：例） /E="tif bmp jpg"			
初期値： /E="tif bmp jpg png pdf"			
/BIN=2 値化種別			
WHOLE	全体2値化	—	画像全体から2値化閾値を決定します。 背景色が均一な画像に対して有効です。
LOCAL	分割2値化	—	分割した局所領域から2値化閾値を決定します。 背景色にムラのある画像に対して有効です。
初期値： WHOLE			
/NOISE=ノイズサイズ			
1/100mm 単位で指定。			
縦+横が指定サイズ以下のノイズ（黒画素）を無視する。			
ノイズの混入による読取性能の低下を軽減します。			

指定範囲：0～200（0：ノイズ除去なし）  
初期値：30

#### /HOLE=孔サイズ

1/100mm 単位で指定。  
縦＋横が指定サイズ以下の孔（白画素）を無視する。  
かすれ等による読取性能の低下を軽減します。  
指定範囲：0～200（0：穴埋めなし）  
初期値：30

#### /U=単位

処理矩形の単位 0=ピクセル(画素数)、1=ミリ(0.1mm 単位)  
未指定時：0  
※処理矩形を使わない場合は無視  
PDF から領域指定で読み取る場合、または、PDF から読み取り座標を取得する場合は、  
ミリ単位(1)を指定ください。

#### /X=解像度

#### /Y=解像度

X, Y 方向の解像度を指定。  
0 を指定、または指定しなければ画像ファイル内の解像度情報に従う。

#### /P=頁番号

認識するページ番号 0=all, 1-n  
存在しないページ番号の場合は処理しない。  
未指定時：1  
マイナスや文字列しかない場合は未指定。

#### /Max=最大数

1 頁内のバーコード検出最大数を指定（既定値：10）。

#### /A=処理矩形

認識エリア指定 “x|y|w|h”（値は4つ必要）  
未指定時：“0|0|0|0”（自動検出）  
解析失敗時：“0|0|0|0”（自動検出）  
PDF から領域指定で読み取る場合は、  
処理矩形の単位をミリ単位としてください。

#### /D =“読取方向”

バーコード認識の読取方向を指定。  
RIGHT →方向  
LEFT ←方向  
UP ↑方向  
DOWN ↓方向  
ANY 全方向（→／←／↑／↓）  
複数指定可：例）/D=“LEFT|RIGHT”  
既定値：“ANY”

#### /B=種別

読み取るバーコードの種別を指定（指定外の種別は無視）  
JAN13 JAN 13  
JAN8 JAN 8  
ITF Interleaved 2 of 5  
CODE39 CODE39  
CODE128 CODE128

NW7	NW-7 (CODABAR)
CUSTOMER	カスタマバーコード ※単独指定の時のみ有効
RSS14	GS1 DataBar Omnidirectional (RSS-14)
ANY	以上全種別 (カスタマバーコード以外)
複数指定可 : 例)	/B="JAN13 JAN8 CODE128"
既定値 :	"ANY"
/L=桁数	
読み取るバーコードの桁数を指定 (指定外の桁数は無視)	
指定範囲 : 0~44 (0 : 任意桁数)	
既定値 : 0	
/LENGTH_MIN_ITF=ITF 最小桁数	
ITF の最小桁数を指定	
指定範囲 : 2~30 (偶数桁のみ)	
既定値 : 4	
/C=チェック式	
チェックデジットチェック式を指定 (指定外のチェック式は無視)	
NONE	なし
MOD10W3	モジュラス 10 ウェイト 3
MOD10W2	モジュラス 10 ウェイト 2
LUHN	ルーンズチェック (モジュラス 10 ウェイト 2)
MOD11_2-7	モジュラス 11
WMOD11	加重モジュラス 11
MOD16	モジュラス 16
MOD43	モジュラス 43
MOD103	モジュラス 103
7DR	7 チェック DR
既定値 :	NONE
/F=書式	
出力テキストの内容 (「出力書式」の項を参照)	
※書式指定文字以外の文字列はそのまま出力	
既定値 : /F="%Data%CRLF" (認識結果+改行)	
/O=ファイル	
出力テキストのファイル名 フルパス	
未指定時 : 標準出力のみ	
/W=上書き	
上書きか追記の指定 0=追記, 1=上書き	
未指定時 : 追記	
範囲外や文字列しかない場合は未指定	
/S="ファイル"	
設定ファイル名 (「動作設定」の項を参照)	
優先度 : コマンドライン引数 > INI ファイル > プログラム既定値	

戻り値

終了コード :

=0 : 正常終了  
 <0 : エラー番号

## 出力書式

## 書式指定子 :

%InputPath	入力フォルダのパス	
%Name	入力ファイル名	
%Ext	入力ファイル拡張子	
%Page	頁番号	
%RetVal	戻り値	
%Length	認識結果文字数	
%Data	認識結果	
%DateTime	処理日時	
%Kind	バーコード種別	
%X, %Width	横 開始位置、幅	PDF から読み取り座標を取得する場合は、
%Y, %Height	縦 開始位置、高さ	処理矩形の単位をミリ単位としてください。
%Code	頁内バーコード連番	
%Serial	ファイル内バーコード連番	
%CRLF	復帰改行	

## 動作設定

INI ファイル形式でプログラムの動作設定が可能。

例 :

> BcDecode.exe /S=sample¥SettingSample.txt

sample¥SettingSample.txt

[Settings]

' 入力フォルダ (絶対パス)

InputPath= D:¥sample¥image

- ' 特定フォルダ内のファイルを一括処理する場合に用いる
- ' 未指定時は一括処理を行わない

' 画像ファイル形式

InputExt = tif|bmp|jpg|png|pdf

- ' 特定フォルダ内のファイルを一括処理する際のファイル拡張子フィルタ
- ' 区切り文字「|」で複数指定可
- ' tif     TIFF
- ' bmp     Bitmap
- ' jpg     JPEG
- ' png     PNG
- ' pdf     PDF
- ' 既定値 : "tif|bmp|jpg|png|pdf"

' 2 値化種別

Bin     = WHOLE

- ' WHOLE   全体 2 値化   —   画像全体から 2 値化閾値を決定します。
- '           背景色が均一な画像に対して有効です。
- ' LOCAL   分割 2 値化   —   分割した局所領域から 2 値化閾値を決定します。
- '           背景色にムラのある画像に対して有効です。

```

' 初期値 : WHOLE

' ノイズサイズ
Noise    = 30
' 1/100mm 単位で指定。
' 縦+横が指定サイズ以下のノイズ（黒画素）を無視する。
' ノイズの混入による読取性能の低下を軽減します。
' 指定範囲 : 0~200 (0 : ノイズ除去なし)
' 初期値 : 30

' 孔サイズ
Hole     = 30
' 1/100mm 単位で指定。
' 縦+横が指定サイズ以下の孔（白画素）を無視する。
' かすれ等による読取性能の低下を軽減します。
' 指定範囲 : 0~200 (0 : 穴埋めなし)
' 初期値 : 30

' 単位
Unit     = 1
' 処理矩形の単位 0=ピクセル(画素数)、1=ミリ (0.1mm 単位)
' 未指定時 : 0
' ※処理矩形を使わない場合は無視
' PDF から領域指定で読み取る場合、または、PDF から読み取り座標を取得する場合は、
' ミリ単位 (1) を指定ください。

' 解像度
DpiX     = 0
DpiY     = 0
' 0 を指定、または指定しなければ画像ファイル内の解像度情報に従う。

' 頁指定
RecogPage= 0
' 認識する頁番号 : =0:全頁、>0:指定頁
' 存在しないページ番号の場合処理しない
' 未指定時 : 1
' マイナスや文字列しかない場合は未指定

' 最大数
Max       = 10
' 1 頁内のバーコード検出最大数を指定 (既定値 : 10)。

' 処理矩形
' Area    =
' 認識エリア指定 x|y|w|h (値は 4 つ必要)
' 未指定時 : 0|0|0|0 (自動検出)
' 解析失敗時 : 0|0|0|0 (自動検出)
' PDF から読み取り座標を取得する場合は、
' 処理矩形の単位をミリ単位としてください。

' 読取方向
Direction= ANY
' バーコード認識の読取方向を指定。
' RIGHT  →方向
' LEFT   ←方向
' UP     ↑方向

```

```

' DOWN    ↓方向
' ANY     全方向 (→/←/↑/↓)
' 複数指定可: 例) Direction = left|right
' 未指定時: "ANY"

' 種別
Kind      = "JAN13|JAN8|ITF|CODE39|CODE128|NW7|RSS14"
読み取るバーコードの種別を指定 (指定外の場合は無視)
JAN13     JAN 13
JAN8      JAN 8
ITF       Interleaved 2 of 5
CODE39    CODE39
CODE128   CODE128
NW7       NW-7 (CODABAR)
CUSTOMER  カスタマバーコード ※単独指定の時のみ有効
RSS14     GS1 DataBar Omnidirectional (RSS-14)
複数指定可: 例) /B="JAN13|JAN8|CODE128"
既定値:   "JAN13|JAN8|ITF|CODE39|CODE128|NW7|RSS14"

' 桁数
Length    = 0
読み取るバーコードの桁数を指定 (指定外の桁数の場合は無視)
指定範囲: 0~44 (0: 任意桁数)
既定値:   0

' ITF 最小桁数
LengthMinITF = 4
ITF の最小桁数を指定
指定範囲: 2~30 (偶数桁のみ)
既定値:   4

' ITF チェック式
CheckITF = NONE
ITF のチェックデジットチェックの有無を指定
NONE     なし
MODE10W3 あり (モジュラス 10 ウェイト 3)
既定値:   NONE

' CODE39 チェック式
CheckCODE39 = NONE
CODE39 のチェックデジットチェックの有効/無効を指定
NONE     なし
MOD43    あり (モジュラス 43)
既定値:   NONE

' NW-7 チェック式
CheckNW7 = NONE
チェックデジットチェックを指定 (指定外の桁数の場合は無視)
NONE     なし
MOD10W3  モジュラス 10 ウェイト 3
MOD10W2  モジュラス 10 ウェイト 2
LUHN     ルーンズチェック (モジュラス 10 ウェイト 2)
MOD11_2-7 モジュラス 11

```

```

WMOD11    加重モジュール 11
MOD16     モジュール 16
7DR       7 チェック DR
既定値：  NONE

' 出力書式
OutputFormat= %RetVal, %Data, %InputPath
' 出力テキストの内容（「出力書式」の項を参照）
' ※書式指定文字以外の文字列はそのまま出力
' 未指定時：%Data のみ

' 出力ファイル
OutputFile= .¥csvout.txt
' 出力テキストのファイルパス
' 未指定時：標準出力のみ

' 上書き
OverWrite= 1
' 上書きか追記の指定 0=追記, 1=上書き
' 未指定時：追記
' 範囲外や文字列しかない場合は未指定

' [EOF]

```

#### 備考

- 入力は Bitmap／TIFF／JPEG／PNG／PDF ファイル  
カラー画像の場合は内部で 2 値化処理を行います。
- 出力は標準出力またはファイル
  - ※ JAN、ITF、CODE39、NW-7、GS1 DataBar Omnidirectional (RSS-14) については、チェックデジットをデータの一部として常に出力します。
  - ※ CODE39 については、スタート／ストップキャラクタ (\*) をデータの一部として常に出力します。
  - ※ NW-7 については、スタート／ストップキャラクタ (A/B/C/D) をデータの一部として常に出力します。

#### 実行例

##### 実行例 1

```
> BcDecode.exe sample¥image¥BC_JAN13.tif
```

```

画像ファイル      : BC_JAN13.tif
出力              : 標準出力に認識結果を表示
その他の設定      : 画像全領域を認識する

```

##### 実行例 2

```
> BcDecode.exe /S=sample¥SettingSample.txt
```

```

設定              : SettingSample.txt (サンプル設定)
入力ファイル      : sample¥image フォルダ内のファイル (*.tif, *.jpg, *.bmp, *.pdf)
出力              : .¥csvout.txt
その他の設定      : 画像全領域を認識する、他 SettingSample.txt 参照

```

##### 実行例 3

```
> BcDecode.exe sample¥image¥BC_JAN13.tif /U=1
/A="0, 0, 180, 200"
```

画像ファイル	: BC_JAN13.tif
出力	: 標準出力に認識結果を表示
その他の設定	: 長さ単位 [mm]、 処理矩形"X 座標, Y 座標, 幅, 高さ"="0.0, 0.0, 18.0, 20.0" [mm]

エラー番号
-------

プログラム終了コード	
0	正常終了
-1000	多重起動時
-1001	引数エラー
-1003	/?
-1004	存在しない設定ファイル
-1005	取り込む画像がない
-1101	CSV の削除に失敗
-1102	CSV のオープンに失敗
// エラーコード	
「12. エラーコード一覧 (DLL 版)」を参照	



## 11. 関数説明 (DLL 版)

### 11.1. 認識関数説明

関数名	1. MdBcDecode_Initialize		
概要	バーコード認識を初期化する。		
宣言形式	<pre>mdRESULT MdBcDecode_Initialize( ( OUT mdHANDLE* outHandle )</pre>		
パラメータ	<table> <tr> <td>outHandle</td><td>バーコード認識ハンドル</td></tr> </table>	outHandle	バーコード認識ハンドル
outHandle	バーコード認識ハンドル		
戻り値	<table> <tr> <td>mdRESULT</td><td>エラー番号</td></tr> </table>	mdRESULT	エラー番号
mdRESULT	エラー番号		
解説	<p>バーコード認識を初期化して、outHandle へ認識ハンドルを返す。</p> <p>未認証で 14 日試用期間終了後、または、コピープロテクタ未挿入の場合、認識ハンドル取得に失敗してエラー:MDRC_ERR_LICENSE_PRODUCT (-32700) を返す。</p>		
使用例	<pre>mdRESULT ret; mdHANDLE bc; // バーコード認識を初期化する ret = MdBcDecode_Initialize( &amp;bc ); if ( ret != MDRC_OK ) {     std::cerr &lt;&lt; "エラー発生";     ... }</pre>		

関数名	2. MdBcDecode_Terminate		
概要	バーコード認識の終了処理をする。		
宣言形式	<pre>mdRESULT MdBcDecode_Terminate (     IN const mdHANDLE inHandle )</pre>		
パラメータ	<table> <tr> <td>inHandle</td><td>バーコード認識ハンドル</td></tr> </table>	inHandle	バーコード認識ハンドル
inHandle	バーコード認識ハンドル		
戻り値	<table> <tr> <td>mdRESULT</td><td>エラー番号</td></tr> </table>	mdRESULT	エラー番号
mdRESULT	エラー番号		
解説	バーコード認識で使用したメモリを開放する。		
使用例	<pre>mdRESULT ret; mdHANDLE bc; // バーコード認識を初期化する ret = MdBcDecode_Initialize( &amp;bc ); ... // バーコード認識の終了処理をする ret = MdBcDecode_Terminate( bc ); if ( ret != MDRC_OK ) {     std::cerr &lt;&lt; "エラー発生";     ... }</pre>		

関数名	3. MdBcDecode_DecodeRect
-----	--------------------------

概要
指定領域内のバーコードを認識する。(スキャナ)

宣言形式
<pre>mdINT32 MdBcDecode_DecodeRect (     IN  const mdHANDLE    inHandle,     IN  const mdBYTE*     inImage,     IN  const mdINT32     inImageWidth,     IN  const mdINT32     inImageHeight,     IN  const mdINT32     inImageWidthByte,     IN  const mdINT32     inLeft,     IN  const mdINT32     inTop,     IN  const mdINT32     inWidth,     IN  const mdINT32     inHeight )</pre>

パラメータ	
inHandle	バーコード認識ハンドル
inImage	画像の先頭アドレス
inImageWidth	画像の幅
inImageHeight	画像の高さ
inImageWidthByte	画像のバイト幅
inLeft	処理矩形左端
inTop	” 上端
inWidth	” 幅
inHeight	” 高さ

戻り値
mdINT32            >=0 の場合は認識バイト数、<0 の場合はエラー番号

解説
<p>バーコードを認識して、戻り値として認識バイト数を返す。          認識結果は、MdBcDecode_GetData() で取得する。          認識結果のコードの種類は、MdBcDecode_GetMode() で取得する。          画像は1ビット/画素の2値画像とする。          バーコード画像は、3mm×[倍率]以上の余白を必要とする。          認識領域内で最初に見つかったコードのみを返す。          inLeft, inTop, inWidth, inHeight で指定する領域内にバーコード以外を含む場合は、MdBcDecode_Decode 関数を使用してください。          PDF から領域指定で読み取る場合は、処理矩形の単位をミリ単位としてください。</p>

## 使用例

```
mdRESULT ret;
mdHANDLE bc;
// バーコード認識を初期化する
ret = MdBcDecode_Initialize( &bc );
...
// 2 値画像を取得する
mdBYTE *bin;           // 2 値画像
mdINT32 binWidth;       // 2 値画像の幅
mdINT32 binHeight;      // 2 値画像の高さ
mdINT32 binWidthByte;   // 2 値画像のバイト幅
mdINT32 left;           // 処理矩形左端
mdINT32 top;            //      "      上端
mdINT32 width;          //      "      幅
mdINT32 height;         //      "      高さ
...
// バーコードを認識する
mdINT32 leng = MdBcDecode_DecodeRect( bc, bin,
                                     binWidth, binHeight, binWidthByte,
                                     left, top, width, height );

if ( leng < 0 )
{
    MdBcDecode_Terminate( bc );
    std::cerr << "エラー発生";
    ...
}
```

関数名	4. MdBcDecode_DecodeRectEx
-----	----------------------------

概要	指定領域内のバーコードを認識する。(スキャナ／カメラ)
----	-----------------------------

宣言形式	<pre>mdINT32 MdBcDecode_DecodeRectEx (     IN const mdHANDLE    inHandle,     IN const mdHANDLE    inImage,     IN const mdINT32     inLeft,     IN const mdINT32     inTop,     IN const mdINT32     inWidth,     IN const mdINT32     inHeight )</pre>
------	--

パラメータ	<table> <tr> <td>inHandle</td><td>バーコード認識ハンドル</td></tr> <tr> <td>inImage</td><td>イメージハンドル</td></tr> <tr> <td>inLeft</td><td>処理矩形左端</td></tr> <tr> <td>inTop</td><td>" 上端</td></tr> <tr> <td>inWidth</td><td>" 幅</td></tr> <tr> <td>inHeight</td><td>" 高さ</td></tr> </table>	inHandle	バーコード認識ハンドル	inImage	イメージハンドル	inLeft	処理矩形左端	inTop	" 上端	inWidth	" 幅	inHeight	" 高さ
inHandle	バーコード認識ハンドル												
inImage	イメージハンドル												
inLeft	処理矩形左端												
inTop	" 上端												
inWidth	" 幅												
inHeight	" 高さ												

戻り値	mdINT32 >=0 の場合は認識バイト数、<0 の場合はエラー番号
-----	-------------------------------------

解説	<p>バーコードを認識して、戻り値として認識バイト数を返す。</p> <p>イメージハンドル(inImage)は、MdBcCommon_LoadImage 関数、または、MdBcCommon_ConvertDIBToImageHandle 関数で取得する。</p> <p>認識結果は、MdBcDecode_GetData() で取得する。</p> <p>認識結果のコードの種類は、MdBcDecode_GetMode() で取得する。</p> <p>画像は 1bit 白黒 2 値、8bit グレyscale、24bit カラー画像に対応。</p> <p>1bit 白黒 2 値以外の場合、内部的に 2 値化処理を行う。</p> <p>バーコード画像は、3mm×[倍率] 以上の余白を必要とする。</p> <p>認識領域内で最初に見つかったコードのみを返す。</p> <p>inLeft, inTop, inWidth, inHeight で指定する領域内にバーコード以外を含む場合は、MdBcDecode_DecodeEx 関数を使用してください。</p> <p>PDF から領域指定で読み取る場合は、処理矩形の単位をミリ単位としてください。</p>
----	--

## 使用例

```
mdRESULT ret;
mdHANDLE bc;
// バーコード認識を初期化する
ret = MdBcDecode_Initialize( &bc );

mdHANDLE hImage;
// 画像の読み込み
ret = MdBcCommon_LoadImage( path, page, &hImage, 300 );

mdINT32 left;           // 処理矩形左端
mdINT32 top;            //  "   上端
mdINT32 width;          //  "   幅
mdINT32 height;         //  "   高さ
...
// バーコードを認識する
mdINT32 leng = MdBcDecode_DecodeRectEx( bc, hImage,
                                         left, top, width, height );

if ( leng < 0 )
{
    MdBcDecode_Terminate( bc );
    std::cerr << "エラー発生";
    ...
}
```

関数名	5. MdBcDecode_Decode
-----	----------------------

概要
画像内のバーコードを認識する。(スキャナ)

宣言形式
<pre>mdINT32 MdBcDecode_Decode (     IN  const mdHANDLE    inHandle,     IN  const mdBYTE*     inImage,     IN  const mdINT32     inImageWidth,     IN  const mdINT32     inImageHeight,     IN  const mdINT32     inImageWidthByte,     IN  const mdINT16     inMaxCount,     OUT mdINT16*          outLength )</pre>

パラメータ	
inHandle	バーコード認識ハンドル
inImage	画像の先頭アドレス
inImageWidth	画像の幅
inImageHeight	画像の高さ
inImageWidthByte	画像のバイト幅
inMaxCount	認識バイト数配列長（バーコード最大数）
outLength	認識バイト数配列

戻り値
mdINT32            >=0 の場合は認識バーコード数、<0 の場合はエラー番号

解説
<p>バーコードを認識して、戻り値として認識バーコード数を返す。          認識結果は、MdBcDecode_GetData() で取得する。          認識結果のコードの種類は、MdBcDecode_GetMode() で取得する。          画像は 1 ビット/画素の 2 値画像とする。          バーコード画像は、3mm×[倍率] 以上の余白を必要とする。          画像内の複数コードを読み取り結果を返す。          画像ページ内での読取順序は不定。</p>

## 使用例

```
mdRESULT ret;
mdHANDLE bc;
// バーコード認識を初期化する
ret = MdBcDecode_Initialize( &bc );
...
// 2 値画像を取得する
mdBYTE *bin;           // 2 値画像
mdINT32 binWidth;       // 2 値画像の幅
mdINT32 binHeight;      // 2 値画像の高さ
mdINT32 binWidthByte;   // 2 値画像のバイト幅
mdINT16 leng[10];       // 認識バイト数配列
...
// バーコードを認識する
mdINT32 count = MdBcDecode_Decode( bc, bin,
                                   binWidth, binHeight, binWidthByte,
                                   sizeof(leng)/sizeof(leng[0]), leng );

if ( count < 0 )
{
    MdBcDecode_Terminate( bc );
    std::cerr << "エラー発生";
    ...
}
```



関数名	6. MdBcDecode_DecodeEx								
概要	画像内のバーコードを認識する。(スキャナ／カメラ)								
宣言形式	<pre>mdINT32 MdBcDecode_DecodeEx (     IN  const mdHANDLE    inHandle,     IN  const mdHANDLE    inImage,     IN  const mdINT16     inMaxCount,     OUT mdINT16*          outLength )</pre>								
パラメータ	<table> <tr> <td>inHandle</td><td>バーコード認識ハンドル</td></tr> <tr> <td>inImage</td><td>画像の先頭アドレス</td></tr> <tr> <td>inMaxCount</td><td>認識バイト数配列長 (バーコード最大数)</td></tr> <tr> <td>outLength</td><td>認識バイト数配列</td></tr> </table>	inHandle	バーコード認識ハンドル	inImage	画像の先頭アドレス	inMaxCount	認識バイト数配列長 (バーコード最大数)	outLength	認識バイト数配列
inHandle	バーコード認識ハンドル								
inImage	画像の先頭アドレス								
inMaxCount	認識バイト数配列長 (バーコード最大数)								
outLength	認識バイト数配列								
戻り値	mdINT32      >=0 の場合は認識バーコード数、<0 の場合はエラー番号								
解説	<p>バーコードを認識して、戻り値として認識バーコード数を返す。              イメージハンドル(inImage)は、MdBcCommon_LoadImage 関数、または、              MdBcCommon_ConvertDIBToImageHandle 関数で取得する。              認識結果は、MdBcDecode_GetData() で取得する。              認識結果のコードの種類は、MdBcDecode_GetMode() で取得する。              画像は 1bit 白黒 2 値、8bit グレyscale、24bit カラー画像に対応。              1bit 白黒 2 値以外の場合、内部的に 2 値化処理を行う。              バーコード画像は、3mm×[倍率] 以上の余白を必要とする。              画像内の複数コードを読み取り結果を返す。              画像ページ内での読取順序は不定。</p>								

## 使用例

```
mdRESULT ret;
mdHANDLE bc;
// バーコード認識を初期化する
ret = MdBcDecode_Initialize( &bc );
...
// 画像ファイルを読み込む
mdCSTRING file = "C:\\MDC_BCR_LIB10\\Sample\\Image\\TestImage01.tif";
mdHANDLE img = 0;
mdRESULT ret = MdBcCommon_LoadImage( file, 1, &img );

mdINT16 leng[10];          // 認識バイト数配列
...
// バーコードを認識する
mdINT32 count = MdBcDecode_DecodeEx( bc, img,
                                     sizeof(leng)/sizeof(leng[0]), leng );

if ( count < 0 )
{
    MdBcDecode_Terminate( bc );
    std::cerr << "エラー発生";
    ...
}
```

関数名	7. MdBcDecode_SetOption
-----	-------------------------

概要
バーコード認識のオプションを設定する。

宣言形式
<pre>mdRESULT MdBcDecode_SetOption (     IN const mdHANDLE inHandle,     IN const mdINT16 inOptionType,     IN const mdINT32 inOptionValue )</pre>

パラメータ	
inHandle	バーコード認識ハンドル
inOptionType	オプション種別
inOptionValue	オプション設定値

戻り値	
mdRESULT	エラー番号

解説

MdBCDecode\_DecodeRect、MdBCDecode\_DecodeRectEx または、MdBCDecode\_Decode、MdBCDecode\_DecodeEx の認識オプションを設定する。

オプション種別

仮想解像度	BC_OPTION_VDPI
画像解像度 X	BC_OPTION_XDPI
画像解像度 Y	BC_OPTION_YDPI
単位長	BC_OPTION_UNIT
読取方向	BC_OPTION_DIRECTION
種別	BC_OPTION_BARTYPE
桁数	BC_OPTION_LENGTH
ITF チェック式	BC_OPTION_CHECK_ITF
CODE39 チェック式	BC_OPTION_CHECK_CODE39
NW-7 チェック式	BC_OPTION_CHECK_NW7
2値化種別	BC_OPTION_BINTYPE
ノイズ除去	BC_OPTION_NOISE_SIZE
穴埋め	BC_OPTION_HOLE_SIZE
ITF 最小桁数	BC_OPTION_LENGTH_MIN_ITF

オプション設定値

仮想解像度	200～600 (未指定時画像解像度)
画像解像度 X,Y	画像の解像度を指定

省略時(既定値)は、400dpi BC_OPTION_UNIT 指定時は必須		
単位長	1 pixel	BC_UNIT_PIXEL (既定値)
	0.1mm	BC_UNIT_MM
PDF から領域指定で読み取る場合、または、PDF から読み取り座標を取得する場合は、ミリ単位 (BC_UNIT_MM) を指定ください。		
読取方向	→	BC_DIRECTION_RIGHT
	←	BC_DIRECTION_LEFT
	↓	BC_DIRECTION_DOWN
	↑	BC_DIRECTION_UP
	自動判定	BC_DIRECTION_ANY
種別	JAN8	BC_BARTYPE_JAN8
	JAN13	BC_BARTYPE_JAN13
	CODE39	BC_BARTYPE_CODE39
	CODE128	BC_BARTYPE_CODE128
	ITF	BC_BARTYPE_ITF
	NW-7	BC_BARTYPE_NW7
	カスタムバーコード	BC_BARTYPE_CUSTOM
	RSS-14	BC_BARTYPE_RSS14
自動判定		BC_BARTYPE_ANY
単独指定のみ有効 GS1 DataBar Omni カスタムバーコード 以外		
桁数	0～44	(0 の場合、桁数自動判定)
チェック式	なし	BC_CHECK_NONE
	モジュラス 10 ウェイト 3	BC_CHECK_MOD10W3
	モジュラス 10 ウェイト 2	BC_CHECK_MOD10W2
	ルーンズ	BC_CHECK_LUHN
	モジュラス 11	BC_CHECK_MOD11
	加重モジュラス 11	BC_CHECK_WMOD11
	モジュラス 16	BC_CHECK_MOD16
	モジュラス 43	BC_CHECK_MOD43
	モジュラス 103	BC_CHECK_MOD103
	7 チェック DR	BC_CHECK_7DR
2値化種別	全体2値化	BC_BINTYPE_WHOLE 画像全体から 2 値化閾値を決定します。 背景色が均一な画像に対して有効です。
	分割2値化	BC_BINTYPE_LOCAL 分割した局所領域から 2 値化閾値を決定します。 背景色にムラのある画像に対して有効です。
ノイズサイズ	0～200	1/100mm 単位で指定。 縦+横が指定サイズ以下のノイズ (黒画素) を無視する。

		ノイズの混入による読取性能の低下を軽減します。 指定範囲：0～200（0：ノイズ除去なし） 初期値：30
穴サイズ	0～200	1/100mm 単位で指定。 縦＋横が指定サイズ以下の孔（白画素）を無視する。 かすれ等による読取性能の低下を軽減します。 指定範囲：0～200（0：穴埋めなし） 初期値：30
ITF 最小桁数	2～30（偶数桁のみ）	指定の桁数に満たないコードは返しません。 初期値：4

#### 使用例

```
mdRESULT ret;
mdHANDLE bc;
// バーコード認識を初期化する
ret = MdBcDecode_Initialize( &bc );
...
// 2 値画像を取得する
mdBYTE *bin;           // 2 値画像
mdINT32 binWidth;       // 2 値画像の幅
mdINT32 binHeight;      // 2 値画像の高さ
mdINT32 binWidthByte;   // 2 値画像のバイト幅
...
// バーコードを認識する
MdBcDecode_SetOption( BC_OPTION_DIRECTION, BC_DIRECTION_ANY ); // 方向自動判定
mdINT32 leng = MdBcDecode_DecodeRect( bc, bin,
                                     binWidth, binHeight, binWidthByte,
                                     1000, 1000, 500, 500 ); // 左上 10cm から縦横 5cm の範囲

if ( leng < 0 )
{
    MdBcDecode_Terminate( bc );
    std::cerr << "エラー発生";
    ...
}
```

関数名	8. MdBcDecode_GetData								
概要	認識データを取得する。								
宣言形式	<pre>mdRESULT MdBcDecode_GetData (     IN  const mdHANDLE    inHandle,     IN  const mdINT16     inIndex,     IN  const mdINT32     inSize,     OUT mdBYTE*           outData )</pre>								
パラメータ	<table> <tr> <td>inHandle</td><td>バーコード認識ハンドル</td></tr> <tr> <td>inIndex</td><td>バーコードインデックス (1～)</td></tr> <tr> <td>inSize</td><td>データ配列長</td></tr> <tr> <td>outData</td><td>データ配列</td></tr> </table>	inHandle	バーコード認識ハンドル	inIndex	バーコードインデックス (1～)	inSize	データ配列長	outData	データ配列
inHandle	バーコード認識ハンドル								
inIndex	バーコードインデックス (1～)								
inSize	データ配列長								
outData	データ配列								
戻り値	mdRESULT エラー番号								
解説	<p>MdBcDecode_DecodeRect、MdBcDecode_DecodeRectEx または、MdBcDecode_Decode、MdBcDecode_DecodeEx で認識した結果を、データ配列として取得する。  データ配列長は、MdBcDecode_DecodeRect、MdBcDecode_DecodeRectEx の戻り値または、MdBcDecode_Decode、MdBcDecode_DecodeEx の outLength[] を使用する。  データ配列は呼び出し側でメモリ確保する。  データ配列のコードの種類は、MdBcDecode_GetMode で取得できるモード(配列)によって判断する。  データ配列長は、MdBcDecode_GetMode() で取得できる文字数(配列)から求めることもできる。</p> <p>※ JAN、ITF、CODE39、NW-7、GS1 DataBar Omnidirectional(RSS-14)については、チェックデジットをデータの一部として常に出力します。  ※ CODE39 については、スタート/ストップキャラクタ(*)をデータの一部として常に出力します。  ※ NW-7 については、スタート/ストップキャラクタ(A/B/C/D)をデータの一部として常に出力します。</p>								

### 使用例 1

```
// バーコードを認識する
mdINT32 leng = MdBcDecode_DecodeRect( bc, bin,
                                     binWidth, binHeight, binWidthByte,
                                     left, top, width, height );

if ( leng < 0 )
{
    MdBcDecode_Terminate( bc );
    std::cerr << "エラー発生";
    ...
}

// 認識データを取得する
mdBYTE *data = new unsigned char[leng+1]; // データ配列
mdRESULT ret = MdBcDecode_GetData( bc, 1, leng, data );
if ( ret != MDRC_OK )
{
    delete [] data;
    MdBcDecode_Terminate( bc );
    std::cerr << "エラー発生";
    ...
}
```

### 使用例 2

```
// バーコードを認識する
mdINT16 leng[10];
mdINT32 count = MdBcDecode_Decode( bc, bin,
                                   binWidth, binHeight, binWidthByte,
                                   sizeof(leng)/sizeof(leng[0]), leng );

if ( count < 0 )
{
    MdBcDecode_Terminate( bc );
    std::cerr << "エラー発生";
    ...
}

// 認識データを取得する
for ( int i = 1; i < count; i++ )
{
    mdBYTE *data = new unsigned char[leng[i-1]+1]; // データ配列
    mdRESULT ret = MdBcDecode_GetData( bc, i, leng[i-1], data );
    if ( ret != MDRC_OK )
    {
        delete [] data;
        MdBcDecode_Terminate( bc );
        std::cerr << "エラー発生";
        ...
    }
    ...
    delete [] data
}
}
```

関数名	9. MdBcDecode_GetMode										
概要	モードを取得する。										
宣言形式	<pre>mdRESULT MdBcDecode_GetMode (     IN  const mdHANDLE    inHandle,     IN  const mdINT16     inIndex,     INOUT mdINT32*        ioSize,     OUT  mdINT32*         outMode,     OUT  mdINT32*         outLength )</pre>										
パラメータ	<table> <tr> <td>inHandle</td><td>バーコード認識ハンドル</td></tr> <tr> <td>inIndex</td><td>バーコードインデックス (1～)</td></tr> <tr> <td>ioSize</td><td>配列数</td></tr> <tr> <td>outmode</td><td>モード配列 (NULL 指定可)</td></tr> <tr> <td>outLength</td><td>文字数配列 (NULL 指定可)</td></tr> </table>	inHandle	バーコード認識ハンドル	inIndex	バーコードインデックス (1～)	ioSize	配列数	outmode	モード配列 (NULL 指定可)	outLength	文字数配列 (NULL 指定可)
inHandle	バーコード認識ハンドル										
inIndex	バーコードインデックス (1～)										
ioSize	配列数										
outmode	モード配列 (NULL 指定可)										
outLength	文字数配列 (NULL 指定可)										
戻り値	mdRESULT エラー番号										
解説	<p>MdBcDecode_DeCodeRect、MdBcDecode_DeCodeRectEx、または、MdBcDecode_DeCode、MdBcDecode_DeCodeEx で認識した結果のモード配列を取得する。</p> <p>モード配列または文字数配列が NULL の場合、配列数を返す。</p> <p>モード配列と文字数配列ともに NULL でない場合、配列数に従ってモード配列と文字数配列に値を返す。</p> <p>モード配列と文字数配列は呼び出し側でメモリ確保する。</p> <p>モードは、認識結果のコードの種類を表し、「1/2/4/8 = 数字/英数字/8ビットバイト/漢字」* となる。バイト数/文字数は、モードにより異なり、「数字/英数字/8ビットバイト/漢字 = 1/1/1/2[バイト]」† となる。</p> <p>したがって、MdBcDecode_DeCodeRect、MdBcDecode_DeCodeRectEx の戻り値である認識バイト数は、          認識バイト数 = 数字数 + 英数字数 + 8ビットバイト数 + 漢字数 * 2          となる。(MdBcDecode_DeCode、MdBcDecode_DeCodeEx の引数に返す認識バイト数配列の値も同様)</p> <p>本関数の使用手順を次に示す。</p> <ol style="list-style-type: none"> <li>1. モード配列または文字数配列が NULL とし、本関数を使用して配列数を取得する。</li> <li>2. 1の配列数でモード配列と文字数配列のメモリを確保する。</li> <li>3. 2のモード配列と文字数配列を設定し、本関数を使用してモード配列と文字数配列を取得する。</li> </ol>										

\* 拡張予定の機能：現行版では、常に「1」を返す。

† 同上



### 使用例

```
// バーコードを認識する
mdINT32 leng = MbBcDecode_DecodeRect( bc, bin,
                                     binWidth, binHeight, binWidthByte,
                                     left, top, width, height );

if ( leng < 0 )
{
    MbBcDecode_Terminate( bc );
    std::cerr << "エラー発生";
    ...
}

// モードを取得する
// 配列数を取得する
mdINT32 size; //配列数
mdRESULT ret = MbBcDecode_GetMode( bc, 1, &size, (mdINT32*)NULL, (mdINT32*)NULL );
if ( ret != MDRC_OK )
{
    MbBcDecode_Terminate( bc );
    std::cerr << "エラー発生";
    ...
}

// モード配列と文字数配列を取得する
mdINT32 *mode = new mdINT32[size]; // モード配列
mdINT32 *length = new mdINT32[size]; // 文字数配列
ret = MbBcDecode_GetMode( bc, 1, &size, mode, length );
if ( ret != MDRC_OK )
{
    delete [] mode;
    delete [] length;
    MbBcDecode_Terminate( bc );
    std::cerr << "エラー発生";
    ...
}
```

関数名	10. MdBcDecode_GetRect												
概要	バーコード認識領域を取得する。												
宣言形式	<pre>mdRESULT MdBcDecode_GetRect (     IN  const mdHANDLE    inHandle,     IN  const mdINT16     inIndex,     OUT mdINT32*          outLeft,     OUT mdINT32*          outTop,     OUT mdINT32*          outWidth,     OUT mdINT32*          outHeight )</pre>												
パラメーター	<table> <tr> <td>inHandle</td><td>バーコード認識ハンドル</td></tr> <tr> <td>inIndex</td><td>バーコードインデックス (1～)</td></tr> <tr> <td>outLeft</td><td>認識矩形左端 (NULL 指定可)</td></tr> <tr> <td>outTop</td><td>” 上端 (NULL 指定可)</td></tr> <tr> <td>outWidth</td><td>” 幅 (NULL 指定可)</td></tr> <tr> <td>outHeight</td><td>” 高さ (NULL 指定可)</td></tr> </table>	inHandle	バーコード認識ハンドル	inIndex	バーコードインデックス (1～)	outLeft	認識矩形左端 (NULL 指定可)	outTop	” 上端 (NULL 指定可)	outWidth	” 幅 (NULL 指定可)	outHeight	” 高さ (NULL 指定可)
inHandle	バーコード認識ハンドル												
inIndex	バーコードインデックス (1～)												
outLeft	認識矩形左端 (NULL 指定可)												
outTop	” 上端 (NULL 指定可)												
outWidth	” 幅 (NULL 指定可)												
outHeight	” 高さ (NULL 指定可)												
戻り値	mdRESULT エラー番号												
解説	<p>認識した結果の矩形情報を取得する。</p> <p>未認識および認識失敗時は、mdRESULT へエラーを返し、outLeft, outTop, outWidth, outHeight は、-1 で初期化する。</p> <p>PDF から読み取り座標を取得する場合は、処理矩形の単位をミリ単位としてください。</p>												

### 使用例

```
// バーコードを認識する
mdINT32 leng = MdBcDecode_DecodeRect( bc, bin,
                                     binWidth, binHeight, binWidthByte,
                                     search_left, search_top, search_width, search_height );

if ( leng < 0 )
{
    MdBcDecode_Terminate( bc );
    std::cerr << "エラー発生";
    ...
}

// バーコード領域を取得する
mdINT32 found_left, found_top, found_width, found_height;
mdRESULT ret = MdBcDecode_GetRect( bc, 1,
                                   &found_left, &found_top, &found_width, &found_height );

if ( ret != MDRC_OK)
{
    MdBcDecode_Terminate( bc );
    std::cerr << "エラー発生";
    ...
}
```

関数名	11. MdBcDecode_GetVersion																
概要	<p>種別を取得する。</p> <p>種別は、JAN8／JAN13／CODE39／CODE128／ITF／NW-7 ／カスタマバーコード／GS1 DataBar Omnidirectional (RSS-14)</p>																
宣言形式	<pre>mdRESULT MdBcDecode_GetVersion (     IN const mdHANDLE    inHandle,     IN const mdINT16     inIndex )</pre>																
パラメーター	<table><tr><td>inHandle</td><td>バーコード認識ハンドル</td></tr><tr><td>inIndex</td><td>バーコードインデックス (1～)</td></tr></table>	inHandle	バーコード認識ハンドル	inIndex	バーコードインデックス (1～)												
inHandle	バーコード認識ハンドル																
inIndex	バーコードインデックス (1～)																
戻り値	<table><tr><td>mdRESULT</td><td>&gt;0 の場合は種別、&lt;=0 の場合はエラー番号</td></tr></table>	mdRESULT	>0 の場合は種別、<=0 の場合はエラー番号														
mdRESULT	>0 の場合は種別、<=0 の場合はエラー番号																
解説	<p>MdBcDecode_DecodeRect、MdBcDecode_DecodeRectEx、または、MdBcDecode_Decode、 MdBcDecode_DecodeEx で認識した結果の種別を取得する。</p> <p>認識失敗時(inIndex 範囲外)の場合は、エラーコードを返す。</p> <p>種別</p> <table><tr><td>BC_BARTYPE_JAN8</td><td>JAN (8 桁)</td></tr><tr><td>BC_BARTYPE_JAN13</td><td>JAN (13 桁)</td></tr><tr><td>BC_BARTYPE_CODE39</td><td>CODE39</td></tr><tr><td>BC_BARTYPE_CODE128</td><td>CODE128</td></tr><tr><td>BC_BARTYPE_ITF</td><td>ITF</td></tr><tr><td>BC_BARTYPE_NW7</td><td>NW-7</td></tr><tr><td>BC_BARTYPE_CUSTOM</td><td>カスタマバーコード</td></tr><tr><td>BC_BARTYPE_RSS14</td><td>GS1 DataBar Omnidirectional (RSS-14)</td></tr></table>	BC_BARTYPE_JAN8	JAN (8 桁)	BC_BARTYPE_JAN13	JAN (13 桁)	BC_BARTYPE_CODE39	CODE39	BC_BARTYPE_CODE128	CODE128	BC_BARTYPE_ITF	ITF	BC_BARTYPE_NW7	NW-7	BC_BARTYPE_CUSTOM	カスタマバーコード	BC_BARTYPE_RSS14	GS1 DataBar Omnidirectional (RSS-14)
BC_BARTYPE_JAN8	JAN (8 桁)																
BC_BARTYPE_JAN13	JAN (13 桁)																
BC_BARTYPE_CODE39	CODE39																
BC_BARTYPE_CODE128	CODE128																
BC_BARTYPE_ITF	ITF																
BC_BARTYPE_NW7	NW-7																
BC_BARTYPE_CUSTOM	カスタマバーコード																
BC_BARTYPE_RSS14	GS1 DataBar Omnidirectional (RSS-14)																

### 使用例

```
// バーコードを認識する
mdINT32 leng = MdBcDecode_DecodeRect( bc, bin,
                                     binWidth, binHeight, binWidthByte,
                                     left, top, width, height );

if ( leng < 0 )
{
    MdBcDecode_Terminate( bc );
    std::cerr << "エラー発生";
    ...
}

// 種別を取得する
mdINT32 version = MdBcDecode_GetVersion( bc, 1 );
if (version <= 0 )
{
    MdBcDecode_Terminate( bc );
    std::cerr << "エラー発生";
    ...
}
```

## 11.2. 一般関数説明

関数名	1. MdBcCommon_CountPage		
概要	画像ファイル内頁数を取得する。		
宣言形式	<pre>mdRESULT MdBcCommon_CountPage (     IN mdCSTRING inFilePath )</pre>		
パラメータ	<table> <tr> <td>inFilePath</td><td>ファイル名 (パス)</td></tr> </table>	inFilePath	ファイル名 (パス)
inFilePath	ファイル名 (パス)		
戻り値	mdRESULT      >=0 の場合は頁数、<0 の場合はエラー番号		
解説	画像ファイルに含まれる頁数を返す。		
使用例	<pre>// 画像の頁数を取得する mdCSTRING file = "C:¥¥MDC_BCR_LIB¥¥Sample¥¥Image¥¥TestImage01.tif"; mdRESULT pages = MdBcCommon_CountPage( file ); if ( pages &lt; 0 ) {     std::cerr &lt;&lt; "エラー発生";     ... }</pre>		

関数名	2. MdBcCommon_ReadFile
-----	------------------------

概要
画像ファイルを読み込む。(スキャナ)

宣言形式
<pre>mdRESULT MdBcCommon_ReadFile (     IN  mdCSTRING      inFilePath,     IN  const mdINT16  inPage,     OUT mdHANDLE*      outHandle,     IN  mdINT16        inDpi = 400 )</pre>

パラメータ	
inFilePath	ファイル名（パス）
inPage	頁番号
outHandle	画像ハンドル
inDpi	規定解像度（解像度情報なし時）

戻り値
mdRESULT      エラー番号

解説
<p>画像ファイルを読み込んで、outHandle に画像ハンドルを取得する。</p> <p>複数頁 TIFF の場合、inPage に読み込む頁番号を指定する。</p> <p>画像データのインスタンス解放は、MdBcCommon_ReleaseImage() で行う。</p> <p>コピープロテクタ未挿入の場合、画像ハンドル取得に失敗してエラーを返す。</p> <p>ファイルが2値画像でない場合は、内部で2値化処理を行う。</p> <p>PDF ファイルの場合、画像解像度が取得不能な場合は、解像度情報に inDpi をセットする。</p> <p>PDF ファイルの場合、画像解像度が 200dpi 未満、または 600dpi より大きい場合は、200dpi、600dpi へそれぞれ変換して画像を取得する。</p>

使用例
<pre>// 画像ファイルを読み込む mdCSTRING file = "C:\\MDC_BCR_LIB10\\Sample\\Image\\TestImage01.tif"; mdHANDLE img = 0; mdRESULT ret = MdBcCommon_ReadFile( file, 1, &amp;img ); if (ret != MDRC_OK ) {     std::cerr &lt;&lt; "エラー発生";     ... } ... MdBcCommon_ReleaseImage( img ); // 画像ハンドルの解放</pre>

関数名	3. MdBcCommon_LoadImage
-----	-------------------------

概要
画像ファイルを読み込む。(スキャナ／カメラ)

宣言形式
<pre>mdRESULT MdBcCommon_LoadImage (     IN  mdCSTRING      inFilePath,     IN  const mdINT16  inPage,     OUT mdHANDLE*      outHandle,     IN  mdINT16        inDpi = 400 )</pre>

パラメータ	
inFilePath	ファイル名（パス）
inPage	頁番号
outHandle	画像ハンドル
inDpi	規定解像度（解像度情報なし時）

戻り値
mdRESULT      エラー番号

解説
<p>画像ファイルを読み込んで、outHandle に画像ハンドルを取得する。</p> <p>複数頁 TIFF の場合、inPage に読み込む頁番号を指定する。</p> <p>画像データのインスタンス解放は、MdBcCommon_ReleaseImage () で行う。</p> <p>コピープロテクタ未挿入の場合、画像ハンドル取得に失敗してエラーを返す。</p> <p>ファイルが2値画像でない場合、本関数内部で2値化処理は行わず、認識時に2値化処理を行う。</p> <p>PDF ファイルの場合、画像解像度が取得不能な場合は、解像度情報に inDpi をセットする。</p> <p>PDF ファイルの場合、画像解像度が 200dpi 未満、または 600dpi より大きい場合は、200dpi、600dpi へそれぞれ変換して画像を取得する。</p>

使用例
<pre>// 画像ファイルを読み込む mdCSTRING file = "C:\\MDC_BCR_LIB10\\Sample\\Image\\TestImage01.tif"; mdHANDLE img = 0; mdRESULT ret = MdBcCommon_LoadImage( file, 1, &amp;img ); if (ret != MDRC_OK ) {     std::cerr &lt;&lt; "エラー発生";     ... } ... MdBcCommon_ReleaseImage( img ); // 画像ハンドルの解放</pre>



関数名	4. MdBcCommon_ConvertDIBToImageHandle						
概要	DIB イメージを読み込む。(スキャナ／カメラ)						
宣言形式	<pre>mdRESULT MdBcCommon_ConvertDIBToImageHandle (     IN BITMAPINFO*    bih,     IN unsigned char* image,     OUT mdHANDLE*     outHandle )</pre>						
パラメータ	<table> <tr> <td>bih</td><td>BITMAPINFO ヘッダ</td></tr> <tr> <td>image</td><td>画像データ</td></tr> <tr> <td>outHandle</td><td>画像ハンドル (=NULL:取得しない)</td></tr> </table>	bih	BITMAPINFO ヘッダ	image	画像データ	outHandle	画像ハンドル (=NULL:取得しない)
bih	BITMAPINFO ヘッダ						
image	画像データ						
outHandle	画像ハンドル (=NULL:取得しない)						
戻り値	mdRESULT エラー番号						
解説	<p>DIB イメージを読み込んで、outHandle に画像ハンドルを取得する。          画像データのインスタンス解放は、MdBcCommon_ReleaseImage() で行う。          コピープロテクタ未挿入の場合、画像ハンドル取得に失敗してエラーを返す。          ファイルが2値画像でない場合、本関数内部で2値化処理は行わず、認識時に2値化処理を行う。</p>						
使用例	<pre>// DIB イメージを読み込む BITMAPINFO *src_bih = NULL; unsigned char* src_image = NULL;  ... TODO: src_bih, src_image に画像をセット (お客様実装)  mdHANDLE img = 0; ret = MdBcCommon_ConvertDIBToImageHandle( src_bih, src_image, &amp;img ); if (ret != MDRC_OK ) {     std::cerr &lt;&lt; "エラー発生";     ... } ... MdBcCommon_ReleaseImage( img ); // 画像ハンドルの解放</pre>						

関数名	5. MdBcCommon_ReleaseImage		
概要	画像ハンドルを解放する。		
宣言形式	<pre>mdRESULT MdBcCommon_ReleaseImage (     IN mdHANDLE inHandle )</pre>		
パラメータ	<table> <tr> <td>inHandle</td><td>画像ハンドル</td></tr> </table>	inHandle	画像ハンドル
inHandle	画像ハンドル		
戻り値	<table> <tr> <td>mdRESULT</td><td>エラー番号</td></tr> </table>	mdRESULT	エラー番号
mdRESULT	エラー番号		
解説	MdBcCommon_ReadFile() で生成した画像データのインスタンスを解放する。		
使用例	<pre>// 画像ファイルを読み込む mdCSTRING file = "C:¥¥MDC_BCR_LIB¥¥Sample¥¥Image¥¥TestImage01.tif"; mdHANDLE img = 0; mdRESULT ret = MdBcCommon_ReadFile( file, 1, &amp;img ); if (ret != MDRC_OK ) {     std::cerr &lt;&lt; "エラー発生";     ... } ... MdBcCommon_ReleaseImage( img ); // 画像ハンドルの解放</pre>		

関数名	6. MdBcCommon_GetImageSize										
概要	画像サイズを取得する。										
宣言形式	<pre>mdRESULT MdBcCommon_GetImageSize (     IN  const mdHANDLE  inHandle,     OUT mdINT32*         outImageWidth,     OUT mdINT32*         outImageHeight,     OUT mdINT32*         outImageWidthByte,     OUT mdINT16*         outImageDepth )</pre>										
パラメータ	<table> <tr> <td>inHandle</td><td>画像ハンドル</td></tr> <tr> <td>outImageWidth</td><td>画像横ピクセル数 (=NULL: 取得しない)</td></tr> <tr> <td>outImageHeight</td><td>画像縦ピクセル数 (=NULL: 取得しない)</td></tr> <tr> <td>outImageWidthByte</td><td>画像横バイト数 (=NULL: 取得しない)</td></tr> <tr> <td>outImageDepth</td><td>画像ビット数 (=NULL: 取得しない)</td></tr> </table>	inHandle	画像ハンドル	outImageWidth	画像横ピクセル数 (=NULL: 取得しない)	outImageHeight	画像縦ピクセル数 (=NULL: 取得しない)	outImageWidthByte	画像横バイト数 (=NULL: 取得しない)	outImageDepth	画像ビット数 (=NULL: 取得しない)
inHandle	画像ハンドル										
outImageWidth	画像横ピクセル数 (=NULL: 取得しない)										
outImageHeight	画像縦ピクセル数 (=NULL: 取得しない)										
outImageWidthByte	画像横バイト数 (=NULL: 取得しない)										
outImageDepth	画像ビット数 (=NULL: 取得しない)										
戻り値	mdRESULT エラー番号										
解説	画像の横幅、高さ、バイト幅およびビット数を outImageWidth, outImageHeight, outImageWidthByte、outImageDepth にそれぞれ取得する。										
使用例	<pre>// 画像ファイルを読み込む mdCSTRING file = "C:\\¥MDC_BCR_LIB10¥¥Sample¥¥Image¥¥TestImage01.tif"; mdHANDLE img = 0; mdRESULT ret = MdBcCommon_ReadFile( file, 1, &amp;img ); ... // 画像サイズを取得する mdINT32 binWidthByte; // 2 値画像の幅 mdINT32 binHeight;    // 2 値画像の高さ mdINT32 binWidthByte; // 2 値画像のバイト幅 ret = MdBcCommon_GetImageSize( img, &amp;binWidth, &amp;binHeight, &amp;binWidthByte, NULL ); if (ret != MDRC_OK ) {     MdBcCommon_ReleaseImage( img );     std::cerr &lt;&lt; "エラー発生";     ... }</pre>										

関数名	7. MdBcCommon_GetImageDPI						
概要	画像解像度を取得する。						
宣言形式	<pre>mdRESULT MdBcCommon_GetImageDPI (     IN  const mdHANDLE  inHandle,     OUT mdINT16*        outDpiX,     OUT mdINT16*        outDpiY )</pre>						
パラメータ	<table> <tr> <td>inHandle</td><td>画像ハンドル</td></tr> <tr> <td>outDpiX</td><td>横解像度(=NULL:取得しない)</td></tr> <tr> <td>outDpiY</td><td>縦解像度(=NULL:取得しない)</td></tr> </table>	inHandle	画像ハンドル	outDpiX	横解像度(=NULL:取得しない)	outDpiY	縦解像度(=NULL:取得しない)
inHandle	画像ハンドル						
outDpiX	横解像度(=NULL:取得しない)						
outDpiY	縦解像度(=NULL:取得しない)						
戻り値	mdRESULT エラー番号						
解説	画像の横解像度および縦解像度を outDpiX, outDpiY にそれぞれ取得する。						
使用例	<pre>// 画像ファイルを読み込む mdCSTRING file = "C:¥¥MDC_BCR_LIB¥¥Sample¥¥Image¥¥TestImage01.tif"; mdHANDLE img = 0; mdRESULT ret = MdBcCommon_ReadFile( file, 1, &amp;img ); ... // 画像解像度を取得する mdINT16 binDpiX; // 2 値画像の横解像度 mdINT16 binDpiY; // 2 値画像の縦解像度 ret = MdBcCommon_GetImageDPI( img, &amp;binDpiX, &amp;binDpiY ); if (ret != MDRC_OK ) {     MdBcCommon_ReleaseImage( img );     std::cerr &lt;&lt; "エラー発生";     ... }</pre>						

関数名	8. MdBcCommon_GetImage				
概要	画像を配列にコピーする。				
宣言形式	<pre>mdRESULT MdBcCommon_GetImage (     IN  const mdHANDLE  inHandle,     OUT      mdBYTE*    outImage )</pre>				
パラメータ	<table> <tr> <td>inHandle</td><td>画像ハンドル</td></tr> <tr> <td>outImage</td><td>画像配列</td></tr> </table>	inHandle	画像ハンドル	outImage	画像配列
inHandle	画像ハンドル				
outImage	画像配列				
戻り値	mdRESULT エラー番号				
解説	<p>画像ハンドルで指定した画像データを outImage で指定する配列に取得(コピー)する。          画像配列は呼び出し側でメモリを確保する。          画像配列に必要な配列サイズは、MdBcCommon_GetImageSize() 関数で取得する。</p>				
使用例	<pre>// 画像ファイルを読み込む mdCSTRING file = "C:\\MDC_BCR_LIB10\\Sample\\Image\\TestImage01.tif"; mdHANDLE img = 0; mdRESULT ret = MdBcCommon_ReadFile( file, 1, &amp;img ); ... // 画像サイズを取得する mdINT32 binWidth;      // 2 値画像の幅 mdINT32 binHeight;     // 2 値画像の高さ mdINT32 binWidthByte;  // 2 値画像のバイト幅 ret = MdBcCommon_GetImageSize( img, &amp;binWidth, &amp;binHeight, &amp;binWidthByte, NULL ); ... // 画像を配列にコピーする mdBYTE* bw = new mdBYTE[binWidthByte * binHeight]; ret = MdBcCommon_GetImage( img, bw ); if (ret != MDRC_OK ) {     delete [] bw;     MdBcCommon_ReleaseImage( img );     std::cerr &lt;&lt; "エラー発生";     ... }</pre>				

## 12. エラーコード一覧 (DLL 版)

エラーコード		
エラー番号		
ERROR_BC1_OK	0	// 正常終了
ERROR_BC1_ERROR	-100	// その他のエラー
ERROR_BC1_MEMORY	-101	// メモリエラー
ERROR_BC1_HANDLE	-102	// 無効ハンドル
ERROR_BC1_PARAM	-103	// 引数エラー (NULL ポインタ)
ERROR_BC1_IMAGE	-104	// 画像が不正
ERROR_BC1_BARTYPE	-105	// 種別エラー (非対応)
ERROR_BC1_DIGIT	-106	// 桁数処理エラー
ERROR_BC1_DECODE	-107	// デコード処理エラー
ERROR_BC1_CHECK	-108	// チェックエラー
// PDF 読み込みを行う場合		
-1008	// 有効なバージョンの Acrobat がインストールされていない	
-1011	// MDC Printer2 利用不可 (ドライバがインストールされていない)	
// 共通エラーコード		
MDRC_OK	0	// 正常終了
MDRC_COMMON_ERR_BASE	-32000	// 共通エラーコード基準
MDRC_ERROR	-32000	// エラー
MDRC_MEMORY	-32001	// メモリ不足
MDRC_INVALIDHANDLE	-32002	// 無効ハンドル
MDRC_INVALIDPARAM	-32003	// パラメータエラー
MDRC_INVALIDPIC	-32004	// 無効画像データ
MDRC_FILEOPEN	-32005	// ファイルがオープンできない
MDRC_FILEREAD	-32006	// ファイル読み取りエラー
MDRC_FILEWRITE	-32007	// ファイル書き込みエラー
MDRC_BINPIC	-32008	// 2 値画像ではない
MDRC_CANNOT_LOAD_DLL	-32009	// DLL がロードできない
MDRC_NOTIMPL	-32010	// API が提供されていません
MDRC_ERR_LICENSE_PRODUCT	-32700	// ライセンスエラー・プロダクト
MDRC_ERR_LICENSE_DATE	-32701	// ライセンスエラー・有効期限エラー
MDRC_ERR_LICENSE_MACHINE	-32702	// ライセンスエラー・マシン ID
// カスタマバーコード エラーコード		
-32200	縦バーを検出するとき、輪郭画像を複製できません。	
-32201	縦バーを検出するとき、輪郭抽出に失敗しました。戻り値, ポインタ = %d, 0x%x	
-32202	傾いた矩形を傾かない矩形 (4 頂点) に変換するとき、左端 > 右端 です。	
-32203	傾いた矩形を傾かない矩形 (4 頂点) に変換するとき、上端 > 下端 です。	
-32204	矩形リストから DPI を求めるとき、グループ番号が不正です。グループ番号=%d	
-32205	矩形リストから DPI を求めるとき、矩形リストが空です。	
-32500	MdcCustomerBarcode_Find() のエラー規定番号	
-32501	MdcCustomerBarcode_Find() のメモリ	
-32502	MdcCustomerBarcode_Find() のその他のエラー	
-32503	MdcCustomerBarcode_Find() のハンドル	
-32504	MdcCustomerBarcode_Find() の文字列処理	
-32520	例外発生	

-32530	MdcCustomerBarcode_Open () のエラー規定番号
-32531	MdcCustomerBarcode_Open () のメモリ
-32532	MdcCustomerBarcode_Open () のその他のエラー
-32533	MdcCustomerBarcode_Open () のハンドル
-32534	MdcCustomerBarcode_Open () の文字列処理
-32600	領域を検出するとき、解像度が不正です。
-32601	領域を検出するとき、領域数が不正です。
-32602	領域を検出するとき、方向が不正です。
-32603	領域を検出するとき、処理画像を保存できません。
-32604	領域を検出するとき、領域を検出できません。
-32605	認識するとき、領域を検出していません。
-32606	認識するとき、領域インデックスが不正です
-32607	認識するとき、文字配列が空です。
-32608	認識するとき、バーコード方向が不正です。
-32609	認識するとき、スペース数が%d未満です。
-32610	認識するとき、スペース数が%dを超えました。
-32611	領域座標を取得するとき、領域を検出していません
-32612	領域座標を取得するとき、領域インデックスが不正です
-32613	領域座標を取得するとき、領域配列が空です。
-32614	処理画像の保存パス名を設定するとき、ロケールを設定できません。
-32615	処理画像の保存パス名を設定するとき、ファイル名の長さが不正です。
-32616	処理画像の保存パス名を設定するとき、パス名をシフトJISに変換できません。
-32617	2値画像を処理画像に変換するとき、画像の先頭アドレスが無効です。
-32618	2値画像を処理画像に変換するとき、画像の幅が不正です。
-32619	2値画像を処理画像に変換するとき、画像の高さが不正です。
-32620	2値画像を処理画像に変換するとき、画像のバイト幅が不正です。
-32621	2値画像を処理画像に変換するとき、解像度横が不正です。
-32622	2値画像を処理画像に変換するとき、解像度縦が不正です。
-32623	2値画像を処理画像に変換するとき、処理矩形の左端が不正です。
-32624	2値画像を処理画像に変換するとき、処理矩形の上端が不正です。
-32625	2値画像を処理画像に変換するとき、処理矩形の幅が不正です。
-32626	2値画像を処理画像に変換するとき、処理矩形の高さが不正です。
-32627	2値画像を処理画像に変換するとき、処理画像を作成できません。
-32628	2値画像を処理画像に変換するとき、その他の例外が発生しました。
-32629	画像を-90度回転するとき、回転画像を作成できません。
-32630	再領域検出するとき、点列を作成できません。
-32631	再領域検出するとき、輪郭画像を複製できません。
-32632	再領域検出するとき、輪郭抽出に失敗しました。
-32633	縦バーを検出するとき、輪郭画像を複製できません。
-32634	縦バーを検出するとき、輪郭抽出に失敗しました。
-32635	縦バーをグループ矩形に変換するとき、グループ別の点列を作成できません。
-32636	グループ矩形を結合するとき、処理画像を作成できません。
-32637	画素率を取得するとき、矩形の大きさが不正です。
-32638	画素率を取得するとき、点列を作成できません。
-32639	画素率を取得するとき、マスク画像を作成できません。
-32640	画素率を取得するとき、画素率[0.0 - 1.0]が不正です。
-32641	画素率を取得するとき、メモリ例外が発生しました。
-32642	画素率を取得するとき、その他の例外が発生しました。
-32643	一致度を取得するとき、矩形数の重みが不正です。
-32644	四角形を傾かない矩形に変換するとき、整形画像を作成できません。
-32645	四角形を傾かない矩形に変換するとき、変換行列を作成できません。
-32646	画像の余白を削除するとき、輪郭画像を複製できません。
-32647	画像の余白を削除するとき、輪郭抽出に失敗しました。

-32648	画像の余白を削除するとき、マスク画像を作成できません。
-32649	画像の余白を削除するとき、左端が見付りません。
-32650	画像の余白を削除するとき、右端が見付りません。
-32651	画像の余白を削除するとき、左端が右端以上です。
-32652	画像の余白を削除するとき、上端が見付りません。
-32653	画像の余白を削除するとき、下端が見付りません。
-32654	画像の余白を削除するとき、上端が下端以上です。
-32655	画像の余白を削除するとき、削除画像を作成できません。
-32656	画像の余白を削除するとき、メモリ例外が発生しました。
-32657	画像の余白を削除するとき、その他の例外が発生しました。
-32658	画素密度を調整するとき、下限が上限より大きいです。
-32659	画素密度を調整するとき、最大処理回数が0以下です。
-32660	スペースを検出するとき、周辺分布メモリを確保できません。
-32661	スペースを検出するとき、平滑化メモリを確保できません。
-32662	スペースを検出するとき、平滑化に失敗しました。
-32663	スペースを検出するとき、2次平滑化メモリを確保できません。
-32664	スペースを検出するとき、2次平滑化に失敗しました。
-32665	スペースを検出するとき、差メモリを確保できません。
-32666	スペースを検出するとき、差の平滑化メモリを確保できません。
-32667	スペースを検出するとき、差の平滑化に失敗しました。
-32668	スペースを検出するとき、微分メモリを確保できません。
-32669	スペースを検出するとき、微分に失敗しました。
-32670	スペース数を調整するとき、入力のスペース数が少な過ぎます。
-32671	スペース数を調整するとき、入力のスペース数が多過ぎます。
-32672	スペース数を調整するとき、調整したスペース数が[%d]に一致しません。
-32673	バー種類を取得するとき、輪郭画像を複製できません。
-32674	バー種類を取得するとき、輪郭抽出に失敗しました。
-32675	バー種類を取得するとき、高さが不正です。
-32676	バー種類を取得するとき、メモリ例外が発生しました。
-32677	バー種類を取得するとき、その他の例外が発生しました。
-32678	k-means 法で分類するとき、分類数が不正です。
-32679	バー種類を3バーコードに変換するとき、スタート・ストップコードのロングバーが不正です。
-32680	バー種類を3バーコードに変換するとき(→)、開始・終了符号のセミロングバーが不正です。
-32681	バー種類を3バーコードに変換するとき(←)、開始・終了符号のセミロングバーが不正です。
-32682	バー種類を3バーコードに変換するとき(←→)、開始・終了符号のセミロングバーが不正です。
-32683	バー種類を3バーコードに変換するとき、バーコード方向が不正です。
-32684	バー種類を3バーコードに変換するとき、変換テーブルインデックスが不正です。
-32685	バー種類を3バーコードに変換するとき、[バー種類→コード]変換できません。
-32686	バー種類を3バーコードに変換するとき、チェックデジットが不正です。
-32687	3バーコードを文字コードに変換するとき、郵便番号が不正です。
-32688	3バーコードを文字コードに変換するとき、第二制御コードが不正です。
-32689	3バーコードを文字コードに変換するとき、英字が不正です。
-32690	3バーコードを文字コードに変換するとき、連続すべき空白コード(CC4)が不正です。



## 付録A. 認識サイズと解像度

認識可能なバーコードの最小の大きさ（目安）は、次の画像を A4 用紙に等倍率で印刷した場合、左から 300/200dpi でスキャンしたものとなる。

### JAN13



### ITF



### CODE39



### NW-7



### GS1 DataBar Omnidirectional (RSS-14)



### カスタマバーコード

（バーコードの端から端までが 64mm～88mm の範囲となるよう印刷してください）





<ご注意>

- 1.本書の内容の一部または全部を無断で転載することはお断りいたします。
- 2.本書に記載した内容は、将来予告なく変更することがあります。あらかじめご了承ください。
- 3.本書の内容について万一ご不審な点や誤りがありましたら、ご連絡下さるようお願いいたします
- 4.運用した結果の影響につきましては、責任を負いかねますのでご了承ください。

※ メディアドライブ株式会社の許可なく複製、改変などを行うことはできません。

第1版            2015年 3月  
メディアドライブ株式会社