PDF Structure

Version 1.10.2

ＰＤＦメタデータ

C# C/C++ 開発環境編

株式会社トラスト・ソフトウェア・システム

2024 年 7 月

目次

1.0 はじめに

PDF Metadata は、PDF（Portable Document Format）文書のメタデータを読み書きおよび PDF 文書の各ページを画像に変換するライブラリです。メタデータの読み書きができるのは PDF データだけです。

PDF Metadata は、PostScript タイプの XObject を画像に変換しません。

2.0 利用環境および PDF のバージョンと画像フォーマット

PDF Metadata は、以下の環境で利用できます。

利用環境	Windows 8、8.1、10、11
開発環境	C#、C/C++、VB. NET
画像フォーマット	PNG（Portable Network Graphics）、JPEG（Joint Photographic Experts Group）、TIFF（Tagged Image File Format）形式、BMP(Device Independent Bitmap)形式 TIFF形式は、非圧縮、Deflate圧縮、JPEG 圧縮、LZW 圧縮を生成します。 BMP形式は非圧縮（WindowsまたはOS/2）を生成します。

PDFのバージョン PDF 1.4 から PDF 1.7 および PDF2.0（全てではありません）を変換します。

PDF Metadata は、パスワードで暗号化されたPDF文書のメタデータを読み書きできます（パスワードが必要です）が、出力される PDF ファイルを暗号化できません。

2.1 開発環境と使用説明書

PDF Metadata 使用説明書は、開発環境ごとに用意してあります。

本書は、PDF 文書のページを画像に変換する関数（メソッド）を C#および C/C++開発環境で利用するための説明書です。他の機能は以下の説明書を参照してください。

・PDF 文書のページを画像に変換する C#および C/C++開発環境編
・PDF 文書のメタデータを解析 C#および C/C++開発環境編 （本書）
・PDF 文書のプリミティブなオブジェクトを抽出 C#および C/C++開発環境編

3.0 インストール

PDF Metadata には、以下のフォルダーおよびファイルが含まれます。

doc	「PDF Metadata 説明書」および「使用許諾契約書」
include	C/C++で使用するためのヘッダファイル、他
lib	ライブラリ群
sample	C#/VB.NET、 C/C++（Visual Studio プロジェクト）サンプル コード

開発環境に応じて適切なフォルダーでご利用ください。

なお、PDF Metadata を使用するためには、適切なライセンスキーが必要です。

3.1 ファイルの概要

PDF Metadata に含まれるファイルの概要です。

libs/x64/PdfStructure.dll	PDF画像変換のためのネイティブDLL(x64専用)です。
libs/win32/PdfStructure.dll	PDF画像変換のためのネイティブDLL(win32専用)です。
libs/x64/PdfStructure.lib	C/C++(x64)開発環境の場合にリンクして使用します。
libs/win32/PdfStructure.lib	C/C++(win32)開発環境の場合にリンクして使用します。
libs/StructureNet.dll	PDF画像変換機能を C#（または VB.NET）で利用するためのラッパーDLLです。
libs/ConstantsNet.dll	メタデータのための定数群
include/Structure.h	C/C++用のヘッダファイルです。C/C++での開発時に使用します。
sample/init.xml	代替フォントを指定する初期化ファイルの例です。

3.2 .NET インターフェース

PDF画像変換ライブラリ（PdfStructure.dll）は、.NETアセンブリではありません。C#またはVB.NETから利用するための.NETアセンブリDLL（StructureNet.dll）を参照して画像変換します。開発時においてはStructureNet.dll を「参照設定」に追加する必要があります。

これらのDLLは、コンパイル・実行時において適切に参照できるようにしてください。

ネイティブDLL（PdfStructure.dll）は32ビット環境用および64ビット環境用に最適化されています。必ず開発・利用環境に沿った DLL を使用してください。

ネイティブDLL（PdfStructure.dll）の32ビット用と64ビット用をサブフォルダ「Win32」と「x64」に配置できます。これらをサブフォルダに配置すると.NETアセンブリDLL（StructureNet.dll）は利用環境に合った適切なネイティブDLLを動的に使用します。

名前空間：

```
PDFTools.PdfStructure
```

クラス名：

```
Structure
```

以下の手順で Structure をインスタンス化してください。

```
Structure stc = new Structure();
```

さらに、Primitive インターフェースを取得します。

```
PrimitiveInterface prm = stc.GetPrimitiveInterface
```

3.3 C/C++ インターフェース

ネイティブC/C++開発環境では、ヘッダファイル（`Structure.h`）を利用できるようにし、ライブラリ（`PdfStructure.lib`）をリンクしてください。実行環境では `PdfStructure.dll` を適切なフォルダーに配置してください。

メソッドやプロパティと同じ機能の関数をC/C++開発環境で使用するために接頭辞「`Mlp`」が追加されて用意されています。

4.0 関数 – .NET、C/C++開発環境

C#/VB.NET、およびC/C++で利用する関数です。C/C++で利用する場合は、接頭辞「Mlp」の付いた関数名に読み替えてください。

4.1 インスタンス化および初期化

PDF Metadata ライブラリはインスタンス化およびライセンスキーを使った初期化が必要です。まず PDFTool.PdfStructure.Structure クラスをインスタンス化します。続いて、以下のメソッドで初期化します。ライブラリはその使用後に Uninitialize メソッドを使って開放します。

メソッド

```
int Initialize(string license)
void InitializeWException(string license)
```

引数

license	PDF Metadata を使用するためのライセンス文字列

戻り値

ライブラリ初期化に成功すると0（ゼロ）が戻ります。それ以外は、エラーコードです。
InitializeWException メソッドは失敗すると Exception をスローします。

使用例

```
using PDFTools.PdfStructure;

using (var stc = new Structure())
{
  try
  {
      stc.InitializeWException("ライセンスキー文字列");
  }
  catch (Exception e)
  {
      Console.WriteLine(e.Message);
  }
  ... // ここで変換などを実施します。
  stc.Uninitialize();
}
```

4.2 ライセンス情報の表示または取得

PDF Metadata の初期化の後はライセンスキー内容を文字列で表示または取得ができます。
ShowLicenseInfo は結果をコンソールに出力し、LicenseInfoStr は結果を文字列で戻します。

メソッド

```
void ShowLicenseInfo()
```

引数

ありません

プロパティ (get)

```
string LicenseInfoString
```

戻り値

ShowLicenseInfo は、戻り値がありません。
LicenseInfoStr は、成功するとライセンスを説明する文字列が戻ります。

4.3 入力PDFファイルを開く

PDF文書を開く際は、そのPDF文書が印刷する場合の解像度を低解像度に指定されている場合や、印刷そのものが禁止されている場合があります。そのような場合PDF文書はパスワードなどで暗号化されています。Metadataは、パスワードで暗号化されたPDF文書を復号して画像に変換できます。また、開くモードを「表示」や「印刷」に指定して適切に開くことができます。

PDF Metadata は、PDF文書以外に画像データを入力として開くことができます。入力画像はそのファイルの拡張子やデータの内容によって判別され適切に解釈されます。が、PDF 文書以外ではメタデータを取り扱えません。

4.3.1 PDFファイルの許可フラグを無視して開く

画像に変換するPDFファイルを開きます。

この関数はPDFファイルが暗号化されている場合でも、PDF文書に指定された許可フラグを無視します。なお、パスワードは、PDFファイルがパスワードで暗号化されている場合のみ使用し、暗号化されていない場合は無視します。

メソッド

```
int OpenDoc(string filename, string ownerPassword, string userPassword)
```

引数

filename	PDF のファイルパス
ownerPassword	所有者パスワード [1]
userPassword	ユーザーパスワード [2]

戻り値

成功すると0（ゼロ）が戻り、失敗した場合はエラーコードが戻ります。

4.3.2 PDFファイルを指定モードで開く

画像に変換するPDFファイルを開くモード（「表示」または「印刷」）に従って開きます。

この関数はPDFファイルが暗号化されている場合、PDF文書に指定された許可フラグに従って開きます。そのため、印刷が許可されないPDFファイルを「印刷」モードで開くと失敗します。また、低解像度印刷指定の場合は失敗しませんので、適切に画像化解像度を指定するため許可フラグを確認する必要があります。（許可フラグは、「4.5.2 PDF文書のパーミッション（許可）フラグ」を参照してください。）

なお、パスワードは、PDFファイルがパスワードで暗号化されている場合のみ使用し、暗号化されていない場合は無視します。

メソッド

```
int OpenDocUsage(string filename, string password, int mode)
```

引数

filename	PDF文書のファイルパス名
password	パスワード オーナーパスワード、ユーザーパスワードのいずれかを指定します。
mode	1（表示）または、2（印刷）を指定します。

戻り値

成功すると0（ゼロ）が戻り、失敗した場合はエラーコードが戻ります。

---

[1] 所有者パスワードは「権限パスワード」と呼ばれることがあります。

[2] ユーザーパスワードは「開くパスワード」と呼ばれることがあります。

4.4 PDF文書の情報

PDF Metadata は、開いたPDF文書の情報を取得できます。

4.4.1 PDF文書のページ数取得

現在開いているPDF文書の総ページ数を取得します。

メソッド

```
int PageCount()
```

引数

ありません

戻り値

成功すると、PDF文書の総ページ数（0（ゼロ）の場合もあります）が戻り、失敗した場合はエラーコードが戻ります。なお、エラーコードは負数です。

4.4.2 PDF文書のパーミッション（許可）フラグ

PDF文書には、その文書を開いたユーザーに変更や印刷の可否を指定するフラグがあります。PDF Metadata はこの値を現在開いているPDF文書からパーミッション（許可）フラグ値として取得します。このフラグを評価するには、このPDF文書の暗号化レビジョンも必要です。暗号化レビジョンを取得するのは「4.5.3PDF文書の暗号化レビジョン」を参照してください

メソッド

```
int GetDocumentInfo(DocumentOpt.PERMISSION_FLAG, out int flag)
```

引数

flag	パーミッション（許可）フラグ値

戻り値

成功すると、0（ゼロ）が戻り、失敗した場合はエラーコードが戻ります。

パーミッションフラグの詳細は、「PDF Reference」 3.5.2 Standard Security Handler を参照してください。

パーミッションフラグ値は、PDF文書の「Standard Security Handler」ディクショナリ内のPキーに指定された整数値です。

4.4.3 PDF文書の暗号化レビジョン

PDF文書が暗号化されている場合は、印刷の許可などのフラグ（パーミッションフラグ）を確認しなければならない場合があります。以下ではこのフラグの評価に必要な暗号化レビジョンを取得します。（パーミッションフラグの取得は「4.5.2 PDF文書のパーミッションフラグ」を参照）

```
メソッド
  int GetDocumentInfo(DocumentOpt.ENCRYPT_REVISION, out int rev)
```

引数

rev	暗号化のレビジョン番号

戻り値

成功すると、0（ゼロ）が戻り、失敗した場合はエラーコードが戻ります。

暗号化レビジョンの詳細は、「PDF Reference」 3.5.2 Standard Security Handler を参照してください。
暗号化レビジョンは、PDF文書の「Standard Security Handler」ディクショナリ内のRキーに指定された値です。

4.4.4 PDF文書のメタデータ

PDF文書に記載されたメタデータをバイトデータまたは文字列で取得します。メタデータはUTF−8文字コードで記載されています。

メタデータの詳細は、「PDF Reference」 10.2 Metadata を参照してください。

```
メソッド
  int GetMetadata(out byte[] metadata)
  int GetMetadataString(out string metadataString)
```

引数

metadata	バイト列のメタデータ
metadataString	Unicode に変換されたメタデータ

戻り値

成功するとバイトサイズまたは文字数が戻り、失敗した場合はエラーコードが戻ります。

4.5 メタデータ

PDF文書のメタデータをXMP（Extensible Metadata Platform）として解析および変更します。

メタデータに記載されたプロパティの取得および変更は XmpInterface クラス（C#の場合）または XMP_TK_HANDLE を介して実行します。

```
C#開発環境の場合：
    Imr = new PdfImager();
    XmpInterface xmp = imr.GetXmpInterface();

C/C++開発環境の場合：
    XMP_TK_HANDLE handle = imr.GetXmpInterface();
```

戻り値

0 はエラーです。それ以外は値が取得できていますので、取得や編集の処理を行えます。

4.6 PDF文書処理の終了

PDF文書の画像変換処理を終了します。

メソッド

```
void CloseDoc()
```

引数

ありません。

戻り値

ありません。

4.7 ライブラリの使用終了

ライブラリの使用を終了します。

メソッド

```
void Uninitialize()
```

引数

ありません。

戻り値

ありません。

5.0 XmpInterface インターフェース

PDF文書のメタデータはXML（Extensible Markup Language）で記載され、その内容はXMP（Extensible Metadata Platform）に従って記載されることが推奨されています。

PDF Metadata このようなメタデータを解析または変更します。なお、解析・変更はPDF文書に記載されたメタデータをシリアル化した後に実施します。

XMPメタデータは `XmpInterface` クラスを介して解析変更します。

XMP規格は、ISO 16684-1 を参照してください。

5.1 `XmpInterface` の取得

メソッド

```
XmpInterface GetXmpInterface()          // C#の場合
XMP_TK_HANDLE MlpGetXmpInterface()      // C/C++の場合
```

引数

ありません。

戻り値

0	エラー
0 以外	成功

エラーの多くは `XMPCode.dll` が参照できない場合に起こります。このDLLを参照可能なフォルダーに格納してください。

5.2 `XmpInterface` の終了

`XmpInterface` は現在文書をクローズすると終了しますが、以下の手順でも終了させることができます。

メソッド

```
void CloseInterface()
```

引数

ありません。

戻り値

ありません。

5.3 メタデータを更新

プロパティが変更されたメタデータでPDF文書を更新します。

更新されたPDFファイルを作成するためにはこの処理が必要です。

メソッド

```
int UpdateDocument()
```

引数

ありません

戻り値

負数	エラーコード（指定された名前空間URIが不正な場合など）
0	成功

5.4 PDF文書をファイルに格納

現在のPDF文書をファイルに格納します。

格納にはPDFファイルを作成できるライセンスが必要です。このライセンスがない場合には、格納されたPDF文書に透かしが挿入され、格納の後にページを画像化した場合にも透かしが挿入されます。

メソッド

```
int DocumentToFile(string fileName)
```

引数

fileName	作成するファイルのパス名

戻り値

負数	エラーコード（指定された名前空間URIが不正な場合など）
0	成功

5.5 プロパティの有無および削除

プロパティがメタデータに存在するかを確認または削除します。確認の場合はプロパティ値の有無は無関係です。削除の場合、プロパティは、その種類に関係なく、必ず削除されます。

Simple（単純）プロパティ、Array（配列）プロパティ、Structure（構造）プロパティに限定する場合はそれぞれ、「5.3Simple プロパティ」、「5. 4Array プロパティ」、「5. 5Structure プロパティ」を参照して下さい。

メソッド

```
int DoesPropertyExist(string uri, string property)
int DeleteProperty(string uri, string property)
```

引数

uri	名前空間 URI
property	プロパティ名

戻り値

負数	エラーコード（指定された名前空間URIが不正な場合など）
0	正しく削除された、または property が存在しない
1 以上	property が存在する

5.6 Simple（単純）プロパティ

5.6.1 プロパティの有無

指定されたプロパティが「Simple（単純）プロパティ」として存在するかを確認します。プロパティの値の有無は無関係です。

メソッド
```
int DoesSimplePropertyExist(string uri, string property)
```

引数

uri	名前空間 URI
property	プロパティ名

戻り値

負数	エラーコード（指定された名前空間URIが不正な場合など）
0	名前空間URIがSimpleプロパティではない、またはSimpleプロパティに property が存在しない。
1 以上	property がSimleプロパティに存在する

5.6.2 プロパティ値の取得および設定

Simpleプロパティの値を取得または設定します。設定する場合にそのプロパティが存在しない場合は新たに追加されますが、独自プロパティの場合は名前空間URIをXmpInterfaceへ登録しなければなりません、「5.7 独自名前空間」を参照してください。

メソッド
```
string GetSimplePropertyExist(string uri, string property)
int GetSimplePropertyExist(string uri, string property, out string value)
int SetSimplePropertyExist(string uri, string property, string value)
```

引数

uri	名前空間 URI
property	プロパティ名
value	プロパティ値

戻り値

負の整数値	エラーコード
正の整数値	property が存在し、値を取得できた場合に文字列のバイト数
0	property が存在し、値を設定した
文字列	property がSimpleプロパティの場合のプロパティ値

5.6.3 プロパティの削除

Simple（単純）プロパティを削除します。

メソッド
```
int DeleteSimpleItem(string uri, string property)
```

引数

uri	名前空間 URI
property	プロパティ名
index	アイテム番号（non-zero ベース）

戻り値

負数	エラーコード
0	削除に成功、プロパティが存在しない

5.7 Array（配列）プロパティ

5.7.1 プロパティの有無

指定されたプロパティが「Array（配列）プロパティ」として存在するか、またはアイテムが存在するかを確認します。値の有無は無関係です。

メソッド
```
int DoesArrayPropertyExist(string uri, string property)
int DoesArrayItemExist(string uri, string property, int index)
```

引数

uri	名前空間 URI
property	プロパティ名
index	アイテム番号（先頭の番号は 1 ）

戻り値

負数	エラーコード （指定された名前空間URIが不正な場合など）
0	Arrayプロパティではない、property がArrayプロパティに存在しない、もしくは index 番号のアイテムがない。
1 以上	指定された property または index のアイテムが存在する。

5.7.2 プロパティ、アイテムの追加

新たな Array(配列)プロパティを追加し、アイテム値を設定します。
既にプロパティが存在している場合は、値が最後のアイテムとして追加されます。

```
メソッド
  int AppendArrayItem(string uri, string property, long option, string
  value)
```

引数

uri	名前空間 URI
property	プロパティ名
option	新たにプロパティを追加する場合のタイプ、既存の場合は無視されます。 以下のいずれかをしています。 kXMP_PropArrayIsOrdered kXMP_PropArrayIsAlternate kXMP_PropArrayIsAltText
value	作成された Array プロパティのアイテム値

戻り値

負数	エラーコード
0	成功

5.7.3 アイテム数

Array(配列)プロパティのアイテム数を取得します。

```
メソッド
  int CountArrayItems(string uri, string property)
```

引数

uri	名前空間 URI
property	プロパティ名

戻り値

負数	エラーコード
正数	Array(配列)プロパティのアイテム数

5.7.4 アイテム値の取得および設定

Array(配列)プロパティのアイテム値を取得または設定します。

```
メソッド
  string GetArrayItems(string uri, string property)
  int GetArrayItems(string uri, string property, int index, out string
  value)
  int SetArrayItems(string uri, string property, int index, string value)
```

引数

uri	名前空間 URI
property	プロパティ名
index	Array プロパティのアイテムのインデックス番号 GetArrayItems ではアイテム数＋１とすると、最後のアイテムとして追加されます。
value	アイテム値

戻り値

負の整数値	エラーコード
正の整数値	property が存在し、値が取得できた場合に文字列のバイト数
0	property が存在し、値を設定できた
文字列	property がArrayプロパティの場合のプロパティ値

5.7.5 アイテムの削除

Array(配列)プロパティのアイテムを削除します。

```
メソッド
  int DeleteArrayItem(string uri, string property, int index)
```

引数

uri	名前空間 URI
property	プロパティ名
index	アイテム番号(non-zero ベース)

戻り値

負数	エラーコード
0	成功、プロパティまたはアイテムが存在しない

5.8 Structure（構造）プロパティ

5.8.1 プロパティ、フィールドの有無

指定されたプロパティやフィールドが「Structure（構造）プロパティ」に存在するかを確認します。プロパティの値の有無は無関係です。

```
メソッド
  int DoesStructPropertyExist(string uri, string property)
  int DoesStructFiedExist(string uri, string property,
                                    string fieldUri, string fieldName)
```

引数

uri	名前空間 URI
property	プロパティ名
fieldUri	Structure プロパティのフィールド名前空間 URI
fieldname	Structure プロパティのフィールド名

戻り値

負数	エラーコード
0	property または fieldNam が存在しない
1 以上	property または fieldName が存在する

5.8.2 フィールド値の取得および設定

Structure プロパティのフィールド値を取得または設定します。
プロパティが無い場合やフィールドが無い場合は新たに作成されます。

```
メソッド
  string GetStructField(string uri, string property, string fieldUri,
                                string fieldName)
  int GetStructField(string uri, string property, string fieldUri,
                                string fieldName, out string fieldValue)
  int SetStructField(string uri, string property, string fieldUri,
                                string fieldName, string fieldValue)
```

引数

uri	名前空間 URI
property	プロパティ名
fieldUri	フィールド名前空間 URI
fieldName	フィールド名
fieldValue	フィールド値

戻り値

負数	エラーコード
0	フィールド値の設定成功
1 以上	取得したフィールド値のバイト数
文字列	取得したフィールド値

5.8.3 フィールドの削除

Structure プロパティのフィールドを削除します。

メソッド

```
int DeleteStructField(string uri, string property,
                      string fieldUri, string fieldName)
```

引数

uri	名前空間 URI
property	プロパティ名
fieldUri	フィールド名前空間 URI
fieldName	フィールド名

戻り値

負数	エラーコード
0	削除成功、またはプロパティやフィールドが存在しない

5.9 日付のプロパティ値

プロパティ値をローカルの日時データとして取得または設定します。

メソッド

```
DateTime GetPropertyDate(string uri, string property)
int GetPropertyDate(string uri, string property, out DateTime datetime)
int SetSimplePropertyDate(string uri, string property, DateTime datetime)
int SetSimplePropertyDate(string uri, string property, int year, int month,
                          int day, int hour, int minute, int second)
int SetSimplePropertyCurrentDate(string uri, string property)
```

引数

uri	名前空間 URI
property	プロパティ名
datetime	取得または設定する日時データ
year	年
month	月
day	日
hour	時
minute	分
second	秒

戻り値

負数	エラーコード
0	成功
DateTime	日時データ

5.10 独自名前空間

独自の名前空間を使用する場合は、XmpInterface への登録が必要です。

メソッド

```
string RegisterNamespace (string uri, string property)
int RegisterNamespace(string uri, string suggested, out string registered)
```

引数

uri	名前空間 URI
suggested	プロパティのプリフィックス
registered	プロパティの登録されたプリフィックス

戻り値

負数	エラーコード
0	成功
文字列	名前空間 URI に関連付けられたプロパティのプリフィックス

5.11 XMP データをダンプする

現在文書のXMPデータをダンプします。

メソッド

```
int DumpObjectToConsole()
int DumpObjectToFile(string fileName)
int DumpObjectToChar(out string data)
```

引数

fileName	出力ファイル名
data	出力文字列

戻り値

負数	エラーコード
0	成功

5.12 エラーの詳細

エラーコードが戻された場合に、より詳細なエラー内容を取得できる場合があります。
このメッソドで取得できるメッセージは必ずしも直前のものではありません。

メソッド

```
string GetLastError()
int GetLastError(out string message)
```

引数

message	エラーのより詳細な内容

戻り値

負数	エラーコード
0	成功

6.0 エラーコード 一覧

エラーコードを以下に記します。

エラーコード	値	エラー内容（および対処）
MLP_ALREADY_INITIALIZED	-1	既に初期化されています。 MlpUninitialize()で終了する、もしくはそのまま処理を続けます。
MLP_NOT_INITIALIZED	-2	初期化できない、もしくは、初期化していません。 MlpInitialize()で再度初期化してください。
MLP_INIT_FILE_OPEN_ERROR	-3	初期化ファイルを読めません。
MLP_INIT_DATA_LOAD_ERROR	-3	初期化データをロードできません。
MLP_LICENSE_ERROR	-4	不正なライセンスキーもしくは、評価用ライセンスキーの期限切れです。 有効なライセンスキーを使用してください。
MLP_ALREADY_OPENED	-5	既にPDF文書をオープンしています。 MlpCloseDoc関数でクローズしてから再度オープンしてください。
MLP_FILE_OPEN_ERROR	-6	指定の入力文書をオープンできません。または、入力画像ファイルを認識できません。 入力ファイルのパスや名前を正しく指定してください。 または、正しい画像ファイルを指定してください。
MLP_FILE_IS_NOT_PDF	-7	PDF文書として解析できません。 正しいPDF文書を指定してください。
MLP_FILE_NOT_DECRYPTED	-8	PDF文書が暗号化されていますが、指定のパスワードでは復号できません。 正しいパスワードを指定してください。
MLP_FILE_NOT_OPENED	-9	PDF文書がオープンされていません。 入力の文書をオープンしてから実行してください。
MLP_INIT_FILE_OPEN_ERROR	-10	初期化ファイルを読めません。 初期化ファイルのパスや名前を正しく指定してください。
MLP_PDF_PARSE_ERROR	-11	PDF文書の解析中にエラーとなりました。 指定のPDF文書を解析できません。
MLP_PDF_HAS_NOT_PAGE	-12	指定のPDF文書にはページがありません。 ページのあるPDF文書を指定してください。
MLP_INVALID_PAGE_NUMBER	-13	指定したページの番号は無効です。 ページ番号は1以上で入力文書のページ総数を超えない値を指定してください。
MLP_INVALID_RESOLUTION	-14	指定された解像度は無効です。 解像度は、50以上2000以下を指定してください。
MLP_INVALID_QUALITY	-15	指定されたJPEG品質は無効です。 JPEG品質は、10以上100以下を指定してください。
MLP_NO_OUTPUT_FILE	-16	出力ファイルが指定されていません。または、指定の出力ファイルの形式（拡張子）が無効です。 正しい形式の出力ファイル名を指定してください。
MLP_TOO_LARGE_PIXEL	-17	作成しようとしている画像が大きすぎます。 解像度下げるか入力文書のページサイズを小さくしてください。
MLP_DRAW_ERROR	-18	画像作成用のメモリー領域を確保できません。または、画像の書き出しに失敗しました。
MLP_MEMORY_ERROR	-20	メモリー領域の確保に失敗しました。

		致命的なエラーです。
MLP_INVALID_CANVAS_SIZE	-22	カンバスサイズが不正です。 正しい値を指定してください。
MLP_INVALID_ARG_VALUE	-22	関数の引数が不正です。 正しい引数値を指定してください。
MLP_INVALID_PICT_TYPE	-23	変換される画像の形式が不正です。 正しい画像形式を指定してください。
MLP_INVALID_CMD	-24	指定されたコマンドが不正です。 正しいコマンドを指定してください。
MLP_NO_DATA	-25	データがありません。
MLP_FAIL_TO_GET_PAGE	-27	ページの取得に失敗しました。
MLP_INVALID_DATA	-30	無効なデータ
MLP_NO_LICENSE_KEY	-31	ライセンスキーが指定されていません。
MLP_UNUSABLE_LICENSE	-32	このバージョンでは使えないライセンスです。
MLP_EXPIRED_LICENSE	-33	失効したライセンスです。
MLP_ILLIGUL_OS_LICENSE	-34	このOSでは使えないライセンスです。
MLP_ILLIGUL_OS_LICENSE	-35	このOSでは使えないライセンスです。
MLP_COLOR_PROFILE_NOT_FOUND	-36	カラープロファイルを読み込めません。
MLP_INVALID_VER_COLOR_PROFILE	-37	Veresio.2でないカラープロファイルです。
MLP_INVALID_COLOR_PROFILE	-38	不正なカラープロファイルです。
MLP_FAIL_TO_COUNT_PAGES	-39	総ページ数の取得に失敗しました。
MLP_INVALID_PDF_VERSION	-40	不正なPDF文書のバージョンです。
MLP_FONT_NOT_FOUND	-41	フォントが見つかりません。
MLP_ALT_FONT_NOT_FOUND	-42	代替フォントが見つかりません。
MLP_FONT_FILE_NOT_OPENED	-43	フォントファイルを開けません。
MLP_FONT_NOT_LOADED	-44	フォントをロードできません。フォントデータが不正です。
MLP_GLYPH_NOT_FOUND	-51	グリフを検索できません。
MLP_UNAVAILABLE_CMD	-61	入力文書に対して利用できないコマンドです。
MLP_NO_METADATA	-71	メタデータがありません。
MLP_COULDNT_PARSE_XML	-72	XMLを解析できません。
MLP_INVALID_METADATA	-73	不正なメタデータ
MLP_COULDNT_PARSE_METADATA	-74	メタデータを解析できません。
MLP_XMP_INVALID_NS_URI	-75	不正な名前空間URI
MLP_XMP_PROPERTY_NOT_EXIST	-76	このプロパティ名はありません。
MLP_XMP_NOT_SIMPLE_PROPERTY	-81	プロパティはSimpleプロパティではありません。
MLP_XMP_NOT_ARRAY_PROPERTY	-82	プロパティはArrayプロパティではありません。
MLP_XMP_ARRAY_HAS_NO_ITEMS	-83	プロパティはArrayプロパティですが、アイテムがありません。
MLP_XMP_TOO_BIG_ARRAY_INDEX	-84	プロパティはArrayプロパティですが、アイテム番号が大きすぎます。
MLP_XMP_INVALID_ARRAY_INDEX	-85	アイテム番号が不正
MLP_XMP_NOT_STRUCT_PROPERTY	-86	プロパティはStructプロパティではありません。
MLP_XMP_FEILD_NOT_EXITS	-87	プロパティはStructプロパティですが、このフィールドはありません。
MLP_XMP_ERROR	-91	XMPエラー