PDF Structure

Version 1.10.2

ＰＤＦオブジェクト抽出

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 Primitive は、PDF（Portable Document Format）文書からそれを構成するオブジェクト（文字列、画像、図形など）を抽出するライブラリです。

PDF文書のページを画像に変換することができます。ただし、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 Primitive 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 Primitive は、パスワードで暗号化されたPDF文書からのオブジェクト抽出や各ページを画像に変換できます。（暗号化されたPDF文書をオープンする際にパスワードを必要とします。）

PDF Primitive は入力データとしてPDF形式以外に種々の画像データを指定できますが、オブジェクトを抽出できるのはPDFデータの場合だけです。

2.1 開発環境と使用説明書

PDF Primitive 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
・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 （本書）

3.0 製品パッケージ

PDF Primitive パッケージには、以下のフォルダーおよびファイルが含まれます。

doc	「PDF Primitive 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 Primitive 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 Primitive に含まれるファイルの概要です。

lib/x64/PdfStructure.dll	PDF画像変換のためのネイティブDLL(x64専用)です。
lib/win32/PdfStructure.dll	PDF画像変換のためのネイティブDLL(win32専用)です。
lib/x64/PdfStructure.lib	C/C++(x64)開発環境の場合にリンクして使用します。
lib/win32/PdfStructure.lib	C/C++(win32)開発環境の場合にリンクして使用します。
lib/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。
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 PdfImager 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てください。

```
PrimitiveInterface xmp = stc.GetPrimitivInterface();
```

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 Primitive 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 Primitive 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 Primitive 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 初期化ファイル（または初期化データ）を指定する

PDF Primitive は、PDF文書で指定されたフォントの代替などを初期化ファイルまたは初期化データで指定できます。初期化ファイル（初期化データ）はXML形式です。

初期化ファイル（初期化データ）でのフォントの代替などの詳細は、「8.0 初期化ファイル（初期化データ）について」を参照してください。

メソッド

```
int LoadInitFile(string filename)
int LoadInitData(string data, int length)
int LoadInitData(string data)
```

引数

filename	初期化ファイルのパス名
data	初期化データ（XML 形式データ）
length	初期化データのバイト数

戻り値

初期化ファイルを読み取ると0（ゼロ）が戻ります。それ以外の場合は、エラーコードです。

4.4 入力ファイル（PDFまたは画像データ）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。Primitive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 Primitive 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 オブジェクト抽出は PDF 文書以外で機能しません。

4.4.1 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。

4.4.2 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。

4.4.3 画像ファイルを開く

画像ファイルを開く場合は、OpenDoc または OpenDocUsage メソッドを利用します。ただし、ファイル名以外の引数は無視されます。また、画像形式はファイルの拡張子およびデータ内容から適切に解釈され内部データ(ピクセルデータ)に変換されます。

読み込まれた画像は、白色不透明な背景画像に描画されます。そのため、画像データにアルファチャンネル(不透明を指定したデータ)が含まれていても出力画像には現れません。

4.5 PDF文書の情報

PDF Primitive は、開いたPDF文書の情報を取得できます。

4.5.1 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。

4.5.2 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 Primitive 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。

4.5.3 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。

4.5.4 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。

4.5.5 PDF文書内のフォント名

PDF文書に記載されたフォント名を取得します。記載されたとおりに取得しますので、フォントを代替する場合の名称として利用できます。(「4.15 代替フォント指定」参照)

```
メソッド(C#)
  int SimpleFontListLength()
  int SimpleFontListGetName(int number, out fontName [,out int flag])
  string[] SimpleFontListGetName()
  FontNameInfo SimpleFontListGetNameC()

関数(C/C++)
  int MilSimpleFontListGetName(int number, TCHAR *fontName, int *flag)
```

引数

number	フォント名に Primitive が検索順に付加した番号
fontName	検索されたフォントの名称
flag	2:フォントのサブセット埋め込まれている、1:フォント全体が埋め込まれている場合、4:フォントが標準14フォントの場合、0:それ以外の場合

戻り値(C#)

SimpleFontListLength および SimpleFontListGetName で引数を指定した場合は失敗すると負数のエラーコードをもどします。それ以外は検索されたフォント名の総数を戻します。SimpleFontListGetName で引数なしの場合は検索されたすべてのフォント名が格納された配列を戻します。SimpleFontListGetNameC 検索されたフォント名と共に埋め込み・非埋め込みなどを示すフラグが格納されたクラスの配列インスタンスを戻します。

戻り値(C/C++)

MilSimpleFontListGetName は失敗すると負数のエラーコードを戻します。それ以外の場合は検索されたフォント名の総数を戻します。

4.6 オブジェクト抽出

PDF Primitive にはPDF文書からデータを直接抽出できる関数が試験的に追加されています。
ここに記載された関数群は将来その動作が変更される場合があります。利用には注意をお願いします。

各関数には、コンソールに出力、文字列として抽出、オブジェクトハンドルとして戻すものがあります。なお、C/C++開発環境で利用する場合は接頭辞 `Mlp` を加えた名称に読み替えてください。

・コンソールに出力する関数
`Show...(...)`

・文字列を戻す関数
`String...(...)`
`string` を戻します。（C/C++では `char**`を戻します。）

・`ObjectHandle` を戻す関数
`Object...(...)`
`ObjectHandle` クラスのメソッドが利用ます。（C/C++ではハンドルとしての意味を持ちます。）

4.6.1 現在文書の `xref`

現在オープンしているPDF文書から `xref` 情報を抽出します。
`xref` の詳細は PDF Reference, version1.7 の「3.4.3 Cross-Reference Table」を参照してください。

メソッド

```
void ShowXref()                    //コンソールに出力
int StringXref(out string data)    //文字列を戻す
int GetXrefOffset()                //xref のオフセット値
```

戻り値

`int` 型の0（ゼロ）または正数は成功です。それ以外はエラーコードです。

4.6.2 現在文書の `trailer`

現在オープンしているPDF文書から `trailer` 情報を抽出します。
`trailer` の詳細は PDF Reference, version1.7 の「3.4.4 File Trailer」を参照してください。

メソッド

```
int ShowTrailer(bool pretty)                      //コンソールに出力
int StringTrailer(bool pretty, out string data)   //文字列を戻す
ObjectHandle StringTrailer()
```

引数

`pretty`　　PDF オブジェクトを表示する際に改行などで見やすくします。

戻り値

`int` 型の0（ゼロ）は成功です。それ以外はエラーコードです。
`ObjectHandle` を戻す場合では、失敗すると不明な型（`UNNOWN_OBJ` 型）が戻ります。

4.6.3 現在文書の catalog

現在オープンしているPDF文書から catalog 情報を抽出します。
catalog の詳細は PDF Reference, version1.7 の「3.6.1 Document Catalog」を参照してください。

メソッド

```
int ShowCatalog(bool pretty);                          //コンソールに出力
int StringCatalog(bool pretty, out string data);  //文字列を戻す
```

4.6.4 現在文書の document information

現在オープンしているPDF文書から document information 情報を抽出します。
document information の詳細は PDF Reference, version1.7 の「10.2.1 Document Information Dictionary」を参照してください。

```
int ShowDocumentInformation(bool pretty);
int StringDocumentInformation (bool pretty, out string data);
```

4.7 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。

4.8 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 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