



FastPDFGen for .NET

Ondemand & Realtime

PDF Generator for Office Report

ユーザーズマニュアル

2007/03/31 版

株式会社 PM9

ThePM9.com

*Copyright 2007 PM9, Inc.
All rights reserved.*

1. 御利用方法

FastPDFGen は、入力したデータに応じて高速に PDF ファイルを生成するプログラムです。帳票出力の為に利用されることを想定しています。複数種類のページテンプレート(雛形)に画像やテキストデータを流し込み、ページを結合することにより PDF ファイルを生成します。

FastPDFGen for .NET は、テンプレート作成ツール(mkPDFtpl.exe)、PDF 生成ライブラリ(.NET アセンブリファイル形式:FastPDFGen.dll、FastPDFGenGDI.dll)から構成されます。通常の利用方法は、.NET 言語で帳票出力をコントロールするホストプログラムを作成し、そのホストプログラムから FastPDFGen をコントロールして PDF ファイルを生成します。

実行例

(1) テンプレートファイルの作成

MS-Word、MS-Excel、PageMaker 等によりテンプレートの元となるファイルを作成し、Acrobat と mkPDFGen.exe によりテンプレートファイルを作成します。

詳しい作成手順は、3 章「帳票テンプレート作成方法」を御参照下さい。

(2) 帳票生成プログラムの作成

プログラムソースをエディタにて作成します。(C#による例、ファイル名:pdfsample.cs)

```
using System;
using PM9;

class HelloClass {
    static void Main() {
        FastPDFGen pdfgen =
            new FastPDFGen("guest::2003/02/16::50BA70AA5D114ABF2D254A9393C3D42A");
        pdfgen.compressField();
        pdfgen.setCryptMode("", "", true, false, false, false);
        pdfgen.start("/tmp/pdfgensample_out.pdf");
        pdfgen.startPage("/tmp/pdfgensample_index.pdf.tpl");
        pdfgen.setPageFieldData("f1", "漢字 ABC");
        pdfgen.setPageFieldData("f2", "abcdefghi");
        pdfgen.endPage();
        pdfgen.startPage("/tmp/pdfgensample.pdf.tpl");
        pdfgen.setPageFieldData("f1", "漢字 ABC");
        pdfgen.setPageFieldData("f2", "abcdefghi");
        pdfgen.setPageFieldData("f3", "12,345,678");
        pdfgen.endPage();
        pdfgen.finish();
    }
}
```

上記プログラムをコンパイルします。

```
csc /target:exe /o+ pdfsample.cs /r:FastPDFGen.dll /r:FastPDFGenGDI.dll
```

(3) 実行プログラムによる帳票データの生成

コンパイルにて生成された帳票出力プログラムを実行して下さい。

```
pdfsample.exe
```

上記プログラムの実行結果として¥tmp¥pdfgensample_r.pdf が作成されます。

2. インストール

インストールする必要のあるファイルは、mkPDFtpl.exe、FastPDFGen.dll、FastPDFGenGDI.dll の3つです。これらのファイルは、パッケージ CD-ROM に同梱されています。

mkPDFtpl.exe は、WindowsPC 上のユーザデスクトップもしくは、適当なフォルダにコピーして下さい。

テンプレートを作成する際、テンプレートの元ファイルを mkPDFtpl.exe コマンドのアイコン上にドラッグ&ドロップすることにより使用しますので、アクセスし易い場所にコピーして下さい。

FastPDFGen.dll 及び FastPDFGenGDI.dll は、帳票生成プログラムから参照可能な場所にコピーしてください。

.NET 環境は、dll(.NET アセンブリ)の検索を下記のいずれかの場所から行います。

- ・DEVPATH 環境変数に列挙されているディレクトリ
- ・グローバル・アセンブリ・キャッシュ
- ・アセンブリのコードベース

3. 帳票テンプレート作成方法

(1) 帳票デザインの用意

- MS-Word、Excel、Illustrator 等を利用して帳票テンプレートの元となる帳票デザインを作成してください。

PDF 化できるものであればどのような形式でもかまいません。

Adobe Acrobat は、ほとんどの形式のファイルを PDF 化できます。

- 帳票デザインは、表紙、明細、裏表紙等、テンプレート結合処理の単位で作成してください。帳票用紙サイズに制約はありません。

(2) 帳票デザインの PDF ファイル化

- Acrobat の PDF 作成ツールを使用して帳票デザインファイルを PDF に変換してください。帳票デザインファイルを Acrobat に直接読み込ませるか、デザインの作成に使用したツールの印刷メニューからプリンタ「Adobe PDF」を選んで印刷を行い PDF ファイルを生成してください。(Acrobat Distiller による PDF の生成)

下記の Acrobat の各バージョンにての動作を確認しております。

- ・ Acrobat4
- ・ Acrobat5
- ・ Acrobat6 Professional
- ・ Acrobat7 Professional
- ・ Acrobat8 Professional

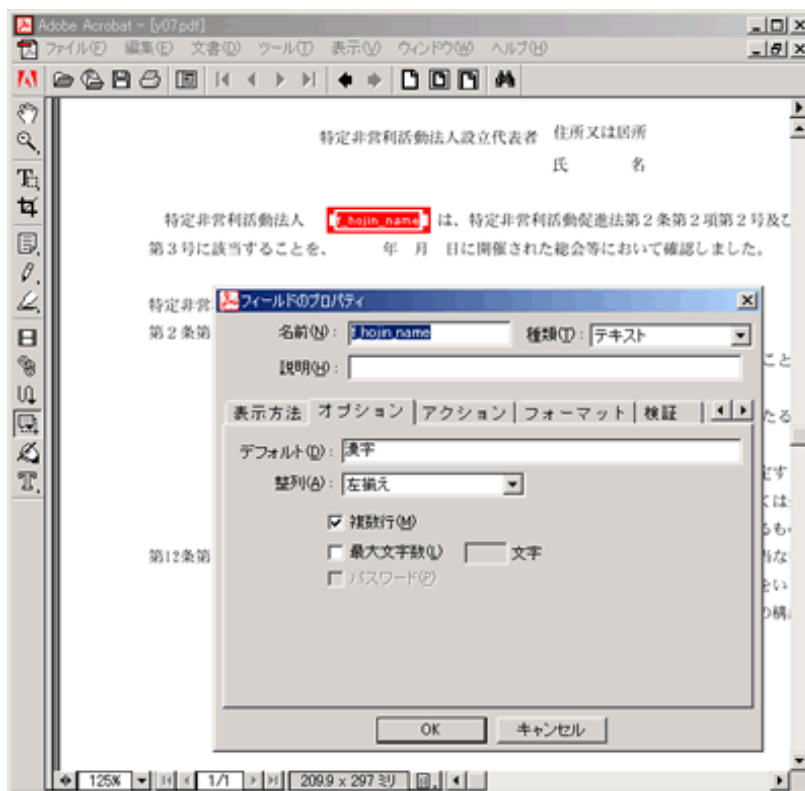
(3) アプリケーションデータ流し込みの為にフォームフィールドの作成

- 先程生成した PDF ファイルを Acrobat で開き、フォームフィールドツールを使ってフォームフィールドを作成してください。

テキストフィールドツールへのアクセスは、下記のようにになります。

[プルダウンメニュー] [ツール] [高度な編集] [フォーム] [テキストフィールドツール]
[プルダウンメニュー] [ツール] [フォーム] [テキストフィールドツール]

Adobe Designer で作成されたフォームには対応しておりません。



テキストフィールドの各項目の設定を下記のように行ってください。

- [一般] [名前]
データを流し込む際に指定するフィールド名を設定。
英数字および”_”のみ使用可能。 長さ 26 文字以内。
- [オプション] [デフォルト]
必ず 1 文字以上の文字列を設定してください。文字列の内容は任意です。

- [オプション] [複数行]
必ずチェックを ON にしてください。
 - [オプション] [整列]
データを流し込んだときに指定の整列形式で表示されます。
 - [表示方法]
データを流し込んだときに指定の形式で表示されます。
 - ・フォント
(例えば市販のバーコードフォントを指定することでバーコードの表示ができます)
 - ・サイズ
「自動」は選択しないでください。かならず、いずれかのポイントを選択してください。
 - [フォーマット]、[検証]、[計算]
FastPDFGen は、流し込んだテキストをそのまま表示します。
ここで設定された属性は、無視されます。
-
- ◆ FastPDFGen は、PDF ファイル生成時にフォームフィールドの ReadOnly 属性を ON にし、フィールド名称を FastPDFGen の内部名称に変更します。
 - ◆ アプリケーションデータの流し込み以外の目的ではなく、ボタンや入力エリアとそれに連動する JavaScript での処理の自動化の為にフォームフィールドを定義する場合は、フォームフィールド名称を、先頭が '_' アンダースコアで始まる文字列にして下さい。
この場合、フォームフィールドの属性を変更しません。

(4) 帳票デザインファイル(PDF 形式)の保存

編集作業を終えた後は、必ず「名前を付けて保存」によりファイルを保存して下さい。

「名前を付けて保存」を行うことにより、PDF の最適化が行われ、ファイルサイズを最小化することができます。結果的に FastPDFGen により生成される PDF ファイルも最適化されます。

(5) PDF 形式帳票デザインの帳票テンプレート(.tpl 形式)への変換

mkPDFtpl コマンドを使用して、帳票テンプレート(.tpl 形式)を作成します。

Windows デスクトップ上で mkPDFtpl.exe のアイコン上に PDF ファイルをドラッグして下さい。
帳票デザインファイルと同一のディレクトリに .tpl 形式の帳票テンプレートファイルが作成されます。

コマンドラインから実行する場合は、下記のように指定して下さい。

mkPDFtpl 帳票デザインファイルの PATH(PDF 形式)

4. FastPDFGen 帳票生成クラス API リファレンス

使用宣言: ソースの先頭に記述

```
using PM9;
```

初期化(コンストラクタ)

```
public FastPDFGen FastPDFGen(string licenseKey)
```

| | |
|-------------|------------|
| licenseKey: | ライセンスキー文字列 |
|-------------|------------|

FastPDFGen 御購入の際、PM9 よりお知らせするライセンスキーを設定して下さい。

評価用ライセンスキーは、WEB 上(<http://www.pm9.com/newpm9/itbiz/pdf/index.php>)に公開しています。

セキュリティ設定

```
public void setCryptMode(string user_password, string master_password,  
    bool print_permission, bool mod_permission,  
    bool copy_permission, bool annot_edit_permisson);
```

| | |
|-----------------------|---------------------------------------|
| user_password: | ユーザパスワードの指定 文字列が長さ 0 の場合パスワード設定なし |
| master_password: | マスターパスワードの指定 文字列が長さ 0 の場合パスワード設定なし |
| print_permission: | 印刷許可 |
| mod_permission: | 文書の変更許可 |
| copy_permission: | 内容のコピーまたは抽出許可 |
| annot_edit_permisson: | 注釈とフォームフィールドの作成許可 |

注意事項: かならず「初期化」コマンドの直後で使用する。

PDF データ圧縮(フィールドデータの圧縮)

public void compressField();

フィールド定義部分のデータを圧縮し生成するPDFファイルのサイズを小さくします。
効率良く圧縮を行う為には、次のようにPDF生成コマンドを作成して下さい。

1. できるだけ**makeArrayPageField**コマンドを使用する
2. **setPageImageData**、**drawLine**、**drawBox**コマンドは、**makeArrayPageField**コマンドの前に記述する

1つのテンプレートが2ページ以上で構成されている場合、そのテンプレートから生成されたページに対しては圧縮処理を行ないません。

注意事項: かならず「初期化」コマンドの直後で使用する。

PDF 生成処理開始

public void start(string generate_pdf_path);

generate_pdf_path: 生成する PDF ファイル PATH

ページ生成・開始

public void startPage(string template_pdf_path)

template_file_path: 帳票テンプレートファイル PATH

フィールドの配列化・縦方向等間隔配置表示

public void makeArrayPageField(string fieldname, int count, float interval)

fieldName: フィールド名
count: 配列要素数
interval: 縦方向配置間隔(配置間隔を POINT 数で指定)
 小数点付き数値で指定可能
 1 POINT = 1/72 inch

配列化を行なったフィールドを **setPageFieldData** コマンド等で指定する場合のフィールド名称は、配列化前のフィールド名称の後ろに「__(1 から始まる要素要素番号)」(アンダースコアを2文字と数字)を付加した文字列となる。

フィールドへのテキストデータ流し込み

public void setPageFieldData(string fieldName, string fieldData)

fieldName: フィールド名
fieldData: 流し込みデータ
 流し込みデータに改行(¥n)コードを入れることにより、
 フィールド中で改行することができます。

フィールドへのテキストデータ流し込み(修飾指定)

```
public void setPageFieldDataWithAttribute(string fieldName, string fieldData,  
float linefeed, float font_size, int color)
```

| | |
|------------|---|
| fieldName: | フィールド名 |
| fieldData: | 流し込みデータ |
| linefeed: | 改行幅(POINT 数指定)を小数点付き数値で指定 1 POINT = 1/72 inch -1 を指定するとフォームフィールド定義時のデフォルト値を使用 |
| font_size: | フォントサイズ(POINT)を小数点付き数値で指定 -1 を指定するとフィールド定義時のフォントサイズを使用 |
| color: | フォントカラー 各 RGB 値を 16 進値で指定 例) 0xFF0000 赤、0x00FF00 緑、0x0000FF 青 -1 を指定するとフィールド定義時のフォントカラーを使用 |

フィールドへのテキストデータ流し込み(修飾指定・フィールドフォントの指定)

```
public void setPageFieldDataWithAttribute(string fieldName, string fieldData,  
float linefeed, float font_size, int color, string font_name)
```

| | |
|------------|--|
| font_name: | フォント名称を指定 指定する Font 名称は、テンプレート作成コマンド mkPDFGen の実行時に生成された.log ファイルの末尾に表示される Font 一覧のうちいずれか(font short_name の部分)を指定する |
|------------|--|

(mkPDFGen のフィールド定義 Font 表示機能)
全フィールドを対象として、フィールド定義 Font を一覧として.log
ファイルの末尾に表示する。setPageFieldDataWithAttribute
にて指定したい Font は、いずれかのフィールドで定義されてい
る必要がある。
もし、通常のフィールドに指定したい Font が無い場合は、ダミー
のフィールドを作成し、その Font を指定する必要がある。

フィールドへのテキストデータ流し込み(修飾指定・フィールドフォント・改行方式の指定)

```
public void setPageFieldDataWithAttribute(string fieldName, string fieldData,  
float linefeed, float font_size, int color, string font_name, int linefeed_mode)
```

font_name: フォント名称を指定
 ""(長さ 0 の文字列)を指定すると、フィールドのデフォルトフォントで表示される

linefeed_mode: 複数行表示時の改行方式を指定
 「0」を指定した場合、日本語ワープロ風に自動改行を行う
 「1」を指定した場合、欧文ワープロ風に自動改行を行う
 「2」を指定した場合、自動改行を行わない

フィールドへのテキストデータ流し込み(修飾指定・フィールド中の文字列表示開始位置指定)

```
public void setPageFieldDataWithAttribute(string fieldName, string fieldData,  
float linefeed, float font_size, int color, string font_name, int linefeed_mode,  
start_ypos)
```

start_ypos: フィールド中の文字列表示開始位置を指定(省略可)
 フィールド枠上段からの相対位置(POINT)を指定

フィールドへのテキストデータ流し込み(表示開始位置、文字間スペース、太字の指定)

```
public void setPageFieldDataWithAttribute(string fieldName, string fieldData,  
float linefeed, float font_size, int color, string font_name, int linefeed_mode,  
float start_ypos, float start_xpos, float char_spacing, float char_thickness)
```

start_xpos: フィールド中の文字列表示開始位置を指定(省略可)
 フィールド枠左端からの相対位置(POINT)を指定

char_spacing: 文字と文字の間のスペースを指定(省略可)
 小数点付き数値(POINT 数指定)にて指定
 マイナス値を設定すると文字間が狭くなる

char_thickness: 文字を指定した数量分太く表示する(省略可)
 小数点付き数値(POINT 数指定)にて指定

円記号表示選択

public void setYenSign(bool yenSignMode)

yenSignMode: フィールドへのテキストデータ流し込み処理時の円記号の表示方式
 を選択する
 false のとき、流し込みデータ中の"¥"文字を"\"と表示
 true のとき、流し込みデータ中の"¥"文字を"¥"と表示

フィールド非表示

public void setPageFieldInvisible(string fieldName)

fieldName: フィールド名

JavaScript からアクセス可能なフィールド値の設定

public void setPageFieldScriptValue(string fieldname, string fieldData)

fieldName: フィールド名
fieldData: 流し込みデータ

画像データ挿入(Path 指定/座標指定)

public void setPageImageData(string imageFile, float xpos, float ypos, float zoom)

imageFile: 元画像ファイルのパス
xpos: 表示位置-X 軸(ページ左上を基点として POINT 数で指定)
 1 POINT = 1/72 inch
ypos: 表示位置-Y 軸(ページ左上を基点として POINT 数で指定)
zoom: 倍率

画像ファイルとして、BMP、GIF、JPEG、PNG 形式のファイルを指定することができます。
IIS 環境で実行する場合、ルートディレクトリからの絶対パスで指定して下さい。

画像データ挿入(Bitmap オブジェクト指定/座標指定)

```
public void setPageImageData(Bitmap bitmap, float xpos, float ypos, float zoom)
```

| | |
|------------|---|
| imageFile: | 元画像ファイルのパス |
| xpos: | 表示位置-X 軸(ページ左上を基点として POINT 数で指定) 1 POINT = 1/72 inch |
| ypos: | 表示位置-Y 軸(ページ左上を基点として POINT 数で指定) |
| zoom: | 倍率 |

画像データ挿入(Path 指定/フォームフィールド指定)

```
public void setPageFieldImageData(string fieldname, string imageFile, float zoom, int align, int valign)
```

| | |
|------------|--|
| fieldName: | 画像の表示位置を指定するためのテキストフォームフィールドの名称 |
| imageFile: | 元画像ファイルのパス |
| zoom: | 倍率を小数点付き数値で指定 0 を設定すると、自動フィットとなります。 |
| align: | 左右寄せルール(-1: 左寄せ、0: 中央、1: 右寄せ) |
| valign: | 上下寄せルール(-1: 上寄せ、0: 中央、1: 下寄せ) |

画像ファイルとして、BMP、GIF、JPEG、PNG 形式のファイルを指定できます。

IIS 環境で実行する場合、PATH の指定をルートディレクトリからの絶対パスで指定することをお勧めいたします。

画像データ挿入(Path 指定/フォームフィールド指定/画像背景の透明化)

```
public void setPageFieldImageData(string fieldname, string imageFile, float zoom, int align, int valign, int mask_mode)
```

| | |
|------------|--|
| mask_mode: | 画像背景の透明化 (0: 透明化無し、1: 白地を透明化、2: 黒地を透明化) |
|------------|--|

画像を他のフィールドの上に上被せする場合は、画像を流し込むフィールドを後に作成・定義してください。

画像データ挿入(Bitmap オブジェクト指定/フォームフィールド指定)

public void setPageFieldImageData(string fieldname, Bitmap bitmap, float zoom, int align, int valign)

| | |
|------------|---------------------------------------|
| fieldName: | 画像の表示位置を指定するためのテキストフォームフィールドの名称 |
| imageFile: | 画像 Bitmap オブジェクト |
| zoom: | 倍率を小数点付き数値で指定 0を設定すると、自動フィットとなります。 |
| align: | 左右寄せルール(-1: 左寄せ、0: 中央、1: 右寄せ) |
| valign: | 上下寄せルール(-1: 上寄せ、0: 中央、1: 下寄せ) |

画像データ挿入(Bitmap オブジェクト指定/フォームフィールド指定/画像背景の透明化)

public void setPageFieldImageData(string fieldname, Bitmap bitmap, float zoom, int align, int valign, int mask_mode)

| | |
|------------|--|
| mask_mode: | 画像背景の透明化 (0: 透明化無し、1: 白地を透明化、2: 黒地を透明化) |
|------------|--|

線分の描画

```
public void drawLine(float xpos, float ypos, float width, float height, float line_width, int dash1,
int dash2, int dash3, int dash4)
```

xpos: 線分始点位置-X 軸(ページ左上を基点として POINT 数で指定)を
小数点付き数値で指定

1 POINT = 1/72 inch

ypos: 線分始点位置-Y 軸(ページ左上を基点として POINT 数で指定)を
小数点付き数値で指定

width: 線分始点位置から終点位置までの幅(POINT 数で指定)を
小数点付き数値で指定

height: 線分始点位置から終点位置までの高さ(POINT 数で指定)を
小数点付き数値で指定

line_width: 線分の太さ(POINT 数で指定)を小数点付き数値で指定

dash1, dash2, dash3, dash4: 破線のパターン指定

dash1:破線の最初の黒線部分の長さを POINT 数で指定

dash2: dash1 に続く白線部分の長さを POINT 数で指定

dash3: dash2 に続く黒線部分の長さを POINT 数で指定

dash4: dash3 に続く白線部分の長さを POINT 数で指定

実線、破線、一点鎖線のみ表示可能。

実線の例: 1(TAB)0(TAB)0(TAB)0

破線の例: 5(TAB)2(TAB)0(TAB)0

一点鎖線の例: 10(TAB)2(TAB)4(TAB)2

矩形の描画

```
public void drawBox(float xpos, float ypos, float width, float height, float line_width, int dash1,
int dash2, int dash3, int dash4)
```

xpos: 矩形表示位置-X 軸(ページ左上を基点として POINT 数で指定)を
小数点付き数値で指定

1 POINT = 1/72 inch

ypos: 矩形表示位置-Y 軸(ページ左上を基点として POINT 数で指定)を
小数点付き数値で指定

width: 矩形の幅(POINT 数で指定)を小数点付き数値で指定

height: 矩形の高さ(POINT 数で指定)を小数点付き数値で指定

line_width: 線分の太さ(POINT 数で指定)を小数点付き数値で指定

dash1、dash2、dash3、dash4: 破線のパターン指定

dash1:破線の最初の黒線部分の長さを POINT 数で指定

dash2:dash1 に続く白線部分の長さを POINT 数で指定

dash3:dash2 に続く黒線部分の長さを POINT 数で指定

dash4:dash3 に続く白線部分の長さを POINT 数で指定

実線、破線、一点鎖線のみ表示可能。

実線の例: 1(TAB)0(TAB)0(TAB)0

破線の例: 5(TAB)2(TAB)0(TAB)0

一点鎖線の例: 10(TAB)2(TAB)4(TAB)2

ページ生成・終了

```
public void endPage()
```

PDF 生成処理終了

```
public void finish();
```

5. エラーコード

FastPDFGen は、異常な状態を検出した場合、次のような例外を発生します。

ライセンスキー不正

System.MemberAccessException **Illegal Licensekey.**

ライセンスキー期限切れ

System.MemberAccessException **License Time Expired.**

コマンド呼び出し順不正

System.MemberAccessException **Illegal method calling sequence.**

帳票テンプレート読み出しエラー

System.ArgumentException **File Read Error(帳票テンプレートファイル名)**

帳票テンプレート上に指定のフォームフィールドが存在しない

System.ArgumentException **Undefined Field Name(フォームフィールド名)**

帳票テンプレート上に指定のフォームフィールドにはデフォルト値が設定されていない

System.ArgumentException **Undefined Field Appearance(フォームフィールド名)**

フィールドへのテキストデータ流し込み時にテンプレート上に存在しないフォントを指定した

System.ArgumentException:: Undefined Font(font short_name:フォント名)

イメージファイル読み出しエラー

System.ArgumentException Imagefile Read Error(イメージファイル名)

生成された PDF ファイルの書き込みエラー

System.ArgumentException File Write Error(生成 PDF ファイル名)

FastPDFGen for .NET ユーザーズマニュアル

株式会社 PM9

〒158-0095

HomePage

e-mail

東京都世田谷区瀬田 3-8-14

<http://www.pm9.com/>

info@pm9.com

Copyright 2007 PM9, Inc. All rights reserved.
