.NET Framework 用 バーコード作成ツール

Barcode.net

説明書

ver 4.2

2022 年 10 月

Pao@Office

はじめに

Barcode.net は、.NET Framework / .NET 5 / 6 / 7 上で動作する、バーコード作成ツール(クラス群)の総称です。

Barcode.net は、次のことを念頭において開発いたしました。

1. 高い精度

RJS のバーコード検査機、Model L2000 を使用してバーコードの精度を細かく検査しています。このため、他社のバーコード作成ツールに比べても高精度なバーコードを作成できます。また、プリンタによる線のにじみや読み取り精度の低下にも対応し、バーコードの線幅微調整も自在に行えます。

2. 使いやすさ

直感的なインターフェースを提供しています。マニュアルの使用例に示されている通り、2～3 つのステップでバーコードの出力が可能です。

3. 軽量

「Barcode.net」はシステムへの負荷が極めて小さく、わずか数 MB のメモリで動作します。

4. 汎用性

バーコードの出力は、Graphics オブジェクトや画像ファイルとして提供されます。そのため、バーコードを作成するアプリケーションからさまざまな用途で利用することができます。PDF への出力にも容易かつ高精度なバーコードを生成できます。PDF へのバーコード描画には、高速かつ軽量な Graphics2D を使用する方法と、画像ファイルを PDF に読み込む方法の 2 種類があります。また、プリンタによるバーコード読み取り精度の微調整も柔軟に行えます。PDF 出力やブラウザ出力に関しては、すぐに試用できるサンプルプログラムも提供しています。

「Barcode.net」を使用することで、.NET を活用したシステム開発において、簡単かつ便利にバーコードの出力を行うことができます。

2023 年 5 月　作者

目次

1. Barcode.net の動作環境・インストール方法

1-1. 動作環境

OS	Windows Xp / Vista / 7 / 8 / 8.1 / 10 Server 2003 / Server 2008 (R2) / Server 2012 R2
.NET Framework or .NET	2.0 / 3.0 / 3.5 / 4.0 / 4.5 / 4.5.1 / 4.5.2 / 4.6 / 4.6.1 /4.6.2 4.7 / 4.7.1 / 4.7.2 / 4.8 / 4.8.1 .NET Core 3 /.NET Core 3.1 / .NET 5 / .NET 6 / .NET 7
動作に必要なメモリ	Microsoft .NET Framework が正常に動作するために 必要な容量
画面解像度	特に制限なし
開発環境	Microsoft Visual Studio .NET 2005 /2008 / 2010 / 2012 / 2013 / 2015 / 2017 / 2019 /2022 いずれかがインストールされている事

1-2. インストール方法

Windows インストーラでのインストールとなります。
デフォルトで、C:¥Program Files (x86)¥Pao@Office¥Barcode.net
(x64 版の場合 C:¥Program Files¥Pao@Office¥Barcode.net) にインストールされます。
インストールフォルダの Pao.Barcode.dll を参照設定(追加)して、お使いください。

1-3. ご購入・ライセンス登録方法

インストールフォルダの License.bat を起動して、必要事項を入力後、
「ご購入申請」・「ライセンス登録ボタン」をクリックしてください。
スタートメニューより、「ご購入・ライセンス登録」を選択していただいても結構です。

2. Barcode.net の機能

2-1. 機能概要

Barcode.net は、以下のバーコードの作成が可能です。

(1) JAN-13(EAN-13)
(2) JAN-8(EAN-8)
(3) UPC-A
(4) UPC-E
(5) ITF(インターリーブド 2 of 5)
(6) Matrix 2 of 5
(7) NEC 2 of 5 (Coop 2 of 5)
(8) NW-7(CODA-BAR)
(9) CODE39
(10) CODE93
(11) CODE128
(12) GS1-128 (UCC/EAN-128)
- コンビニ向け標準料金代理収納用バーコード
- 医療用 医薬品等のバーコード
- 医療用 医療材料等のバーコード
- 食肉標準物流バーコード「基本バーコード」

(13) 郵便カスタマバーコード
(14) GS1 Databar 標準型 (RSS-14) … **ver 3.0 において追加**
(15) GS1 Databar 限定型 (RSS Limited) … **ver 3.0 において追加**
(16) GS1 Databar 拡張型 (RSS Expanded) … **ver 3.0 において追加**
(17) QR コード
(18) 標準料金代理収納用バーコード(コンビニバーコード)
(19) DataMatrix (GS1 DataMatrix)
(20) PDF417

※郵便カスタマバーコード・GS1 Databar・二次元バーコード(QR / DataMatrix / PDF417)以外は、以降総称して「一次元バーコード」と呼びます。

Barcode.net では、上記の各バーコードを作成するために、バーコードの種類ごとに全て別々のクラスとして利用することが可能となっております。
Barcode.net の各バーコード作成クラスは２つのコンストラクタを用意しております。一つ目は、クラスのコンストラクタで.NET の System.Drawing.Graphics オブジェクトを受け取り、Graphics オブジェクトに対してバーコードを描画します。二つ目は、クラスのコンストラクタで保存する画像ファイルパス、画像ファイルの種類(Jpeg/Png)を受け取り画像ファイルにバーコードを出力します。
また、SVG 形式のファイルへの出力が可能です。

2-1-1. コンビニ向け標準料金代理収納用バーコードについて

コンビニ向け標準料金代理収納用バーコード(コンビニバーコード)の出力も可能です。

コンビニバーコードは、「財団法人流通システム開発センター」が発行した「UCC/EAN128 による 標準料金代理収納ガイドライン」に準拠した、GS1-128(UCC/EAN128)のバーコードを生成する事が可能です。

コンビニバーコードの印字位置、バーコードの高さについては、mm(ミリ)単位で指定する事ができます。

コンビニバーコードの横幅決定の基本的な方法は、ガイドラインに準拠し、プリンタの解像度(dpi)に合わせて描画を行います。

ただし、ver 2.4.0 以降で、コンビニバーコードの幅を指定できるようにいたしました。

ガイドラインはあくまでガイドですので、お客様が自由にバーコードの幅をご指定頂いて問題ございません。

2-1-2. 一次元バーコードの種類

次の種類のバーコードを出力できます。

1．JAN-13(EAN-13)	2．JAN-8(EAN-8)
数字 13 桁。12 桁指定時、13 桁目のチェックディジットを自動計算付与。 	数字 8 桁。7 桁指定時、8 桁目のチェックディジットを自動計算付与。 
3．UPC-A	**4．UPC-E**
数字 12 桁。11 桁指定時、12 桁目のチェックディジットを自動計算付与。 	数字 7 桁。6 桁指定時、7 桁目のチェックディジットを自動計算付与。 
5．ITF(インターリーブド 2 of 5)	**6．Matrix 2 of 5**
使用可能文字：数字 	使用可能文字：数字 
7．NEC 2 of 5 (Coop 2 of 5)	**8．NW-7(CODA-BAR)**
使用可能文字：数字 	使用可能文字： ABCD.+:/$-0123456789  1 文字目 ABC いずれか入力した文字がスタート・ストップキャラクタとなる。入力がない場合 C が既定値。 **パラメータ：** スタート・ストップキャラクタ表示／非表示

９．CODE39

使用可能文字：

1234567890ABCDEFGHIJKLMNOPQRSTUVWXYZ-. *$/+%

パラメータ：
スタート・ストップキャラクタ表示／非表示

１０．CODE93

使用可能文字：

1234567890ABCDEFGHIJKLMNOPQRSTUVWXYZ-. *$/+%

１１．CODE128

多くの半角英数文字記号・エスケープシーケンスが使用可能。
CODE A/B/C により出力文字・パターンを切り替えることが可能。

パラメータ：
どのコードパターンを使用するか。
AUTO / CODE_A / CODE_B / CODE_C
既定値は、AUTO。AUTO は、数時 4 文字続けば CODE A にコードチェンジするなど、バーコードが一番小さくなるように自動調整する。

１２．GS1-128 (UCC/EAN-128)

- コンビニ向け標準料金代理収納用バーコード
- 医療用 医薬品等のバーコード
- 医療用 医療材料等のバーコード
- 食肉標準物流バーコード「基本バーコード」

コンビニバーコード 入力：
{FNC1}91912345000000000000045208750040131003295006

１３．郵便カスタマバーコード

郵便番号＋住所の英数字部分のみ入力
例：27500263-29-2-401
パラメータ：8～11.5 ポイント(大きさ)

2-1-3. 二次元バーコードの種類

次の種類のバーコードを出力できます。

1．QR コード

パラメータ：(既定値はアンダーバー)

(1) エラー訂正レベル：L / M / Q / H

(2) 文字種：数字 / 大文字英数字 / 8 ビットバイトデータ(漢字含む)

(3) バージョン： 1～40 *5*

(4) 全角文字エンコード： “shift-jis” / “utf8” /etc..

2．DataMatrix (GS1 DataMatrix)

パラメータ：(既定値はアンダーバー)

(1) シンボルコードサイズ：自動 / 10x10～144x144 / 8x18～16x48 等

(2) 全角文字エンコード： “utf8” / “shift-jis” / etc..

3．PDF417

パラメータ：(既定値はアンダーバー)

(1) エラー訂正レベル自動決定：する / しない

(2) エラー訂正レベル：1～8 2

(3) アスペクト比：0.5 / 1.0 / 2.0 等

(4) 列・行数決定方法：
アスペクト比より自動決定 / 列数指定 / 行数指定 / 列数行数指定

(5) 指定列数：1～30 5

(6) 指定行数：3～90 5

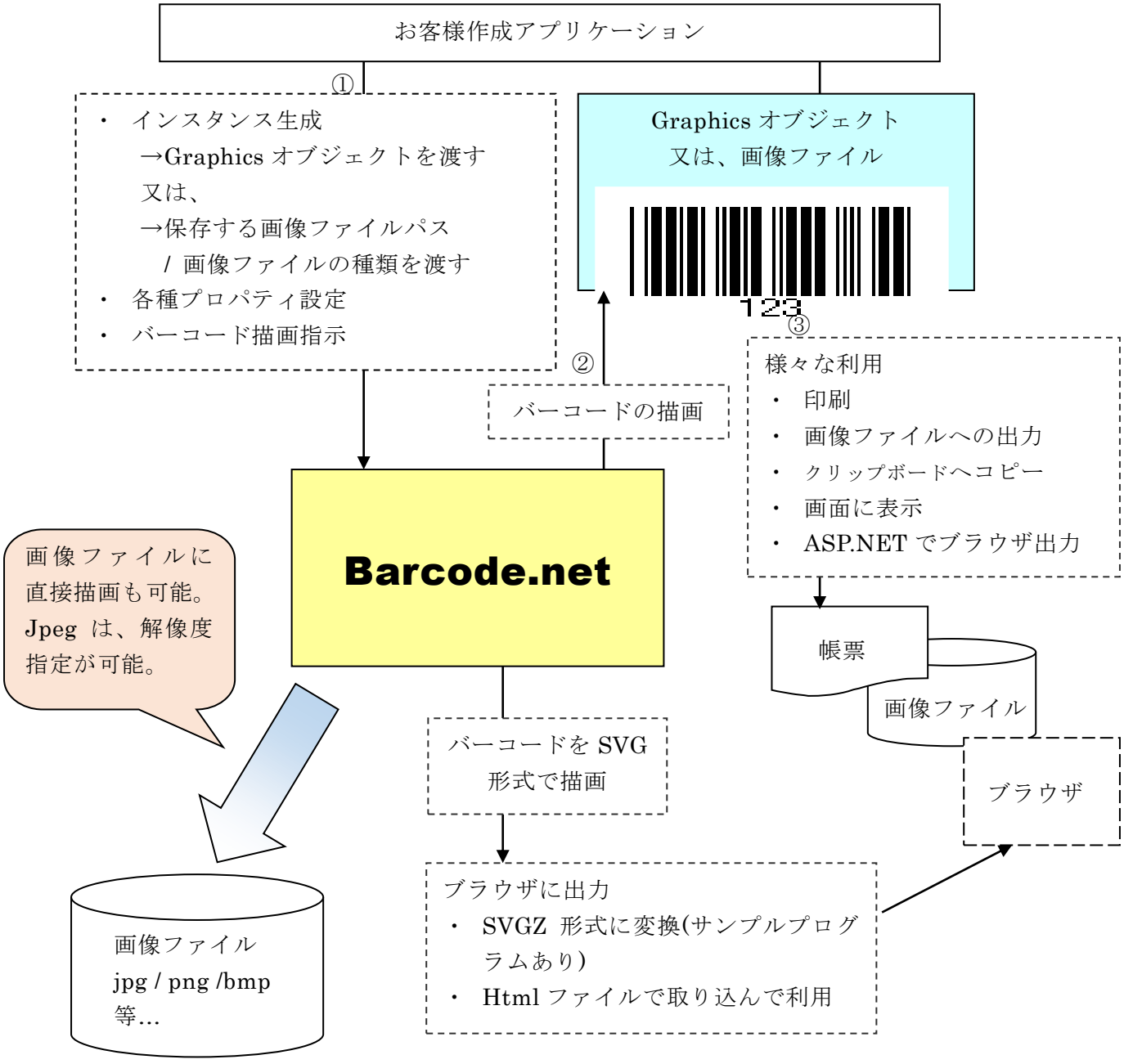
(7) 全角文字エンコード： “shift-jis” / “utf8” /etc..

2-1-4. GS1 Databar (RSS) の種類

次の種類のバーコードを出力できます。

<table>
<tr><th colspan="2">１．GS1 データバーOmni-directional / 表示桁数：数字 14 桁(GTIN)</th></tr>
<tr><td>Omni-directional(標準型)
 
高さが狭い Truncated(カット型)という種類もございますが、この標準型の高さを調整して出力してください。</td><td>Stacked（2 層型）
  </td></tr>
<tr><td>Stacked Omni-directional
(標準 2 層型)
  </td><td></td></tr>
<tr><th colspan="2">２．GS1 データバーLimited / 表示桁数：数字 14 桁(GTIN)</th></tr>
<tr><td colspan="2">Limited(限定型)
  </td></tr>
<tr><th colspan="2">３．GS1 データバーExpanded / 表示桁数：最大数字 74 桁または英字 41 文字</th></tr>
<tr><td>Expanded(拡張一層型)
  </td><td>Stacked Expanded(拡張多層型)
  </td></tr>
</table>

2-1-5.概要図


お客様作成アプリケーション
①
・ インスタンス生成
→Graphics オブジェクトを渡す
又は、
→保存する画像ファイルパス
/ 画像ファイルの種類を渡す
・ 各種プロパティ設定
・ バーコード描画指示
Graphics オブジェクト
又は、画像ファイル
123
③
②
バーコードの描画
様々な利用
・ 印刷
・ 画像ファイルへの出力
・ クリップボードへコピー
・ 画面に表示
・ ASP.NET でブラウザ出力
Barcode.net
画像ファイルに直接描画も可能。Jpeg は、解像度指定が可能。
帳票
画像ファイル
ブラウザ
バーコードを SVG 形式で描画
画像ファイル
jpg / png /bmp
等…
ブラウザに出力
・ SVGZ 形式に変換(サンプルプログラムあり)
・ Html ファイルで取り込んで利用

2-2. 一次元バーコード作成クラスの機能

Barcode.net の各一次元バーコード作成クラスは、以下の機能を有します。

(1) バーコードの描画

コード、始点(左上の X/Y 座標)とバーコードの高さ・幅を指定してバーコードを描画します。幅の代わりに、バーコードの線幅の最小値を指定して描画することも可能です。その場合、より高い精度のバーコードが作成できますが、最終的に描画される幅の調整が必要になります。

もう一つの描画方法として、幅を指定していただきその中に納まる一番広い幅で、かつ、ドットにのった精度の高いバーコードを描画することも可能です。

それぞれ順番に、Draw / DrawDelicate / DrawDirect メソッドをご用意しております。

座標・高さ・幅の単位は、ミリやインチ、ピクセルなど、.NET の Graphics オブジェクトが持つ全ての座標単位での指定が可能です。

少し特殊な GS1-128(UCC/EAN128)において、AI(アプリケーション識別子)挿入方法は 2 通りございます。

(1) 可変長項目(データブロック)の後の AI には、FNC1 を挿入
⇒"{FNC1}"を入力してください。例："{FNC1}21"のようにコードを指定

(2) 固定長項目(データブロック)の後の AI には、固定長のため目印の FNC1 は不要
⇒"{AI}"を入力してください。例："{AI}21" のようにコードを指定

"{AI}"を指定して FNC1 を挿入しない場合も、カッコ()付コード文字は出力されます。例えば入力コードに"{AI}21"を指定した場合、添え字には(21)と出力されます。

例) (01)04512345670016(21)1 ⇒(01)の前には FNC1 を挿入し(21)の前には挿入しない。
コード指定方法 → "{FNC1}0104512345670016{AI}211"

(2) 添字の描画

バーコードの下にコードの文字列自体を描画します(既定値)。プロパティの設定で描画をしないようにすることも可能です。

添字を、コードを意味するバーの位置に描画するか(JAN 既定値)、バーコード全体の幅に均等割付するか(通常既定値)を指定することが可能です。

※ JAN(EAN)コードの場合、既定値の状態では商品コードのバーコードのような描画を行い、均等割付にすると書籍コードのバーコードのような描画を行います。

添字のフォントをプロパティで指定することも可能です。

CODE39／NW-7(CODABAR) のみスタート・ストップキャラクタを印字するかどうかをプロパティで指定することが可能です。既定値は印字しません。

(3) 回転描画

プロパティの設定により、バーコードの左上始点座標を中心に、
90 度／180 度／270 度　回転して描画することが可能です。既定値は 0 度です。

(4) 黒バー・白バーの幅調整

プロパティの設定により、描画する黒バーと白バーの幅をドット単位で微細調整できます。

既定値は、0 ドットです。

例えば、このプロパティに-1 を指定すると、バーコード内全ての黒バーの幅が 1 ドットずつ細くなります。

プリンタにより、調整が必要な場合にこの機能を使用してください。

※この機能は。DrawDirect / DrawDelicate メソッドには有効ですが、Draw メソッドには無効ですのでご注意ください。

→インクジェットプリンタで黒バーがにじんで太くなる時などに有効

例：コンビニバーコード、EPSON PX-502A(360DPI)のプリンタの場合、
白バーを＋１すると丁度良い。

2-3. コンビニ向け標準料金代理収納用バーコード(コンビニバーコード)

GS1-128(UCC/EAN128)バーコード作成クラスにコンビニバーコード描画メソッドを用意しました。コンビニバーコードメソッドは、以下の機能を有します。

コンビニのバーコード作成クラス GS1-128(EAN128)に含まれます。
Barcode.net のコンビニバーコードは、以下の機能を有します。

(1) バーコードの描画

コードと、始点(左上の X/Y 座標)及び、バーコードの高さを mm(ミリ)単位で指定し、バーコードを描画します。
バーコードの幅は、プリンタの解像度(dpi)により、以下の表の通り、
自動的に決まります。
ver 2.4.0 以降で、コンビニバーコードの幅を指定できるようにいたしました。ガイドラインはあくまでガイドですので、お客様が自由にバーコードの幅をご指定頂いて問題ございません。(単位:mm)

解像度	モジュール幅		バーコード部の幅
	ドット	Mm	
300dpi	2	0.169	48.67mm
400dpi	3	0.190	54.72mm
480dpi	3	0.158	45.50mm
600dpi	4	0.169	48.67mm
300dpi の倍数	2 の倍数	0.169	48.67mm

アプリケーション識別子(AI)の前に"{FNC1}"という文字列を指定することで GS1-128(UCC/EAN128)のバーコード作成を可能としています。
例) (91)912345 ・・・() 付きがアプリケーション識別子(AI)
コード指定方法→「{FNC1}91912345」

(2) 添字の描画

バーコードの下にコードの文字列自体を描画します。この添字(コード文字列)は、ガイドラインに従い、左詰で以下のように描画します。

(91)912345-12345678901234567892 11
020331-0-123456-2

なお、これは、コードで以下のように指定された場合です。
「{FNC1}9191234512345678901234567892110203310123456 2」

2-4. 郵便カスタマバーコード作成クラスの機能

Barcode.net の郵便カスタマバーコード作成クラスは、以下の機能を有します。

(1) バーコードの描画

コード、始点(左上の X/Y 座標)と大きさとしてポイント(8～11.5)を指定してバーコードを描画します。

コードの表記は・・・

[郵便番号の数字部分 7 桁]+[郵便番号では不明部分の住所の英数字を「－」区切り]で、指定してください。

例）〒116-0013 東京都荒川区西日暮里五丁目 37 番 5 号スタートアップオフィスＡ－207 号室

コード指定方法→「11600135-37-5-A-207」

※詳しくは、旧郵政省の web ページにマニュアルがございますのでご覧になってください。

座標の単位は、.net の Graphics オブジェクトが持つ全ての座標単位で指定が可能です。

(2) 回転描画

プロパティの設定により、バーコードの左上始点座標を中心に、
90 度／180 度／270 度　回転して描画することが可能です。
既定値は、0 度です。

2-5. QR コード作成クラスの機能

Barcode.net の QR コード作成クラスは、以下の機能を有します。

QR コードの描画
コード、始点(左上の X/Y 座標)、幅・高さを指定します。

QR コードの描画方法は一次元バーコード同様、3 種類あります。

- **幅・高さ指定ぴったり描画(Draw)**
  指定幅・高さにぴったり描画する。
  幅・高さに合わせるためバーコード画像の最低限の拡大縮小が発生
  特徴）指定サイズぴったりに描画できる。精度は他より劣る。
- **最小値指定直接描画(DrawDelicate)**
  描画する最小値を指定して精度の高いバーコードを描画。
  特徴）描画しないとサイズがわからない。ドットにのった高い精度。
- **幅指定直接描画(DrawDirect)**
  指定幅以内で一番大きく、かつ、ドットにのった正方形を描画。(高さ＝幅)
  特徴）指定サイズ(以内)で描画できる。ドットにのった高い精度。

バーコードを描画する座標・幅・高さ・最小値の単位は、ミリ・インチ・ピクセル等、.NET の Graphics オブジェクトが持つ全ての座標単位で指定することができます。

プロパティで以下の項目を指定することができます。

- バージョン(1～40)　既定値：5
- エラー訂正レベル(L,M,Q,H)　既定値：M
- エンコードモード　既定値：Z
  (N：数字モード A：英数字モード　Z：その他(漢字等、8bit byte モード)
- 全角エンコーディング（“shift-jis”,”utf-8”,etc..）既定値：shift-jis

エンコードモードに漢字モードがありませんが、漢字の入力も「Z:その他(8bit byte モード)」を指定してください。
※決めかねるときは、”Z”を使用してください。

2-6. DataMatrix 作成クラスの機能

Barcode.net の DataMatrix 作成クラスは、以下の機能を有します。

DataMatrix の描画

コード、始点(左上の X/Y 座標)、幅・高さを指定します。

DataMatrix の描画方法は一次元バーコード同様、3 種類あります。

- **幅・高さ指定ぴったり描画(Draw)**
  指定幅・高さにぴったり描画する。
  幅・高さに合わせるためバーコード画像の最低限の拡大縮小が発生
  特徴) 指定サイズぴったりに描画できる。精度は他より劣る。
- **最小値指定直接描画(DrawDelicate)**
  描画する最小値を指定して精度の高いバーコードを描画。
  特徴) 描画しないとサイズがわからない。ドットにのった高い精度。
- **幅指定直接描画(DrawDirect)**
  指定幅以内で一番大きく、かつ、ドットにのった正方形、または、シンボルコードサイズ通りの比率の長方形を描画。
  特徴) 指定サイズ(以内)で描画できる。ドットにのった高い精度。

バーコードを描画する座標・幅・高さ・最小値の単位は、ミリ・インチ・ピクセル等、.NET の Graphics オブジェクトが持つ全ての座標単位で指定することができます。

プロパティで以下の項目を指定することができます。

・ シンボルコードサイズ　　既定値：DxCodeSize.DxSzAuto

```
(enum) DxCodeSize.
  DxSzRectAuto,        DxSz24x24,        DxSz88x88,
  DxSzAuto,            DxSz26x26,        DxSz96x96,
  DxSzShapeAuto,       DxSz32x32,        DxSz104x104,
                       DxSz36x36,        DxSz120x120,
  DxSz10x10,           DxSz40x40,        DxSz132x132,
  DxSz12x12,           DxSz44x44,        DxSz144x144,
  DxSz14x14,           DxSz48x48,
  DxSz16x16,           DxSz52x52,        DxSz12x36,
  DxSz18x18,           DxSz64x64,        DxSz16x36,
  DxSz20x20,           DxSz72x72,        DxSz16x48
  DxSz22x22,           DxSz80x80,
```

・ 全角エンコーディング（"utf-8", "shift-jis", etc..) 既定値：utf-8

2-7. PDF417 作成クラスの機能

Barcode.net の PDF417 作成クラスは、以下の機能を有します。

PDF417 の描画

コード、始点(左上の X/Y 座標)、幅・高さを指定します。

PDF417 の描画方法は一次元バーコード同様、3 種類あります。

- 幅・高さ指定ぴったり描画(Draw)

指定幅・高さにぴったり描画する。

幅・高さに合わせるためバーコード画像の最低限の拡大縮小が発生

特徴) 指定サイズぴったりに描画できる。精度は他より劣る。

- 最小値指定直接描画(DrawDelicate)

描画する最小値を指定して精度の高いバーコードを描画。

特徴) 描画しないとサイズがわからない。ドットにのった高い精度。

- 幅指定直接描画(DrawDirect)

指定幅以内で一番大きく、かつ、ドットにのった長方形を描画。

長方形の 2 編の長さの比率は、プロパティの縦横アクセプト比とサイズ種別により決定される。

特徴) 指定サイズ(以内)で描画できる。ドットにのった高い精度。

バーコードを描画する座標・幅・高さ・最小値の単位は、ミリ・インチ・ピクセル等、.NET の Graphics オブジェクトが持つ全ての座標単位で指定することができます。

プロパティで以下の項目を指定することができます。

・ サイズ種別　…データ列数・行数決定方法

自動	Pdf417.SIZE_KIND.AUTO　(既定値)
データ列数指定	Pdf417.SIZE_KIND.COLUMNS
データ行数指定	Pdf417.SIZE_KIND.ROWS
データ列数・行数指定	Pdf417.SIZE_KIND.COLUMNS_AND_ROWS

・ 行数　…出力データ行数指定

サイズ種別が、

出力行数指定の場合＝(自動サイズでない・列数指定でない場合)有効

3～90　　既定値：5

・ 列数　…出力データカラム数指定

サイズ種別が、

出力列数指定の場合＝(自動サイズでない・行数指定でない場合)有効

1～30

・ エラーレベル
0～8　　既定値：2

・ エラーレベル自動決定
自動でエラー訂正レベルを決定(する・しない) 既定値：true(する)

・ 縦横アクセプト比
シンボルの縦横比、シンボル アスペクト レシオ（比）
既定値：0.5

・ 全角エンコーディング（”utf-8”, “shift-jis”, etc..）
既定値：utf-8

2-8. GS1 データバー(RSS)作成クラスの機能

Barcode.net の各 GS1 データバー(RSS)作成クラスは、以下の機能を有します。

(1) バーコードの描画

コード、始点(左上の X/Y 座標)とバーコードの高さ・幅を指定してバーコードを描画します。幅の代わりに、バーコードの線幅の最小値を指定して描画することも可能です。その場合、より高い精度のバーコードが作成できますが、最終的に描画される幅の調整が必要になります。

もう一つの描画方法として、幅を指定していただきその中に納まる一番広い幅で、かつ、ドットにのった精度の高いバーコードを描画することも可能です。それぞれ順番に、Draw / DrawDelicate / DrawDirect メソッドをご用意しております。

標・高さ・幅の単位は、ミリやインチ、ピクセルなど、.net の Graphics オブジェクトが持つ全ての座標単位での指定が可能です。

(2) バーコードの高さ指定とその出力結果

高さ指定とその出力結果について Barcode.net 独自の仕様がございます。
次の表にまとめました。

１．GS1 データバーOmni-directional / 表示桁数：数字 14 桁(GTIN)

Omni-directional （標準型）

一次元バーコードなので、幅高さは通常の一次元バーコードと同様に自在です。

Stacked（2 層型）

バーコード上段・センター部分・下段の高さ割合が決まっているため、高さの指定があっても割合をキープして出力することが基本です。

Barcode.net では、DrawDirect / DrawDelicate メソッドを使用した時に、センター部分を含む 3 段の割合を確実にキープします。

Draw メソッドの場合は、センターの小さな正方形群の各正方形割合はキープしながら、ユーザより引数で指定された高さにより上段・下段の割合に従って上段下段の高さを決定します。従って Draw メソッドでは、本来決まっているセンター部分を含む 3 段の高さ割合は保たれません。指定された高さにより、上・下段が決められている割合を保ちながら伸び縮みします。

Stacked Omni-directional （標準 2 層型）

バーコードの上段・下段の割合は同一で、センター部分(小さな正方形 3 段)がある仕様のバーコードです。従ってセンター部分の高さを固定し、ユーザより引数で指定された高さにより、上・下段がその割合によって同じ高さで伸び縮みします。

２．GS1 データバーLimited / 表示桁数：数字 14 桁(GTIN)

Limited （限定型）

一次元バーコードなので、幅高さは通常の一次元バーコードと同様に自在です。

３．GS1 データバーExpanded / 表示桁数：最大数字 74 桁または英字 41 文字	
Expanded(拡張一層型) 一次元バーコードなので、幅高さは通常の一次元バーコードと同様に自在です。  	Stacked Expanded(拡張多層型) このバーコードは、各段(層)の割合は同一な仕様です。段と段(層と層)間の小さな 3 段の正方形の高さを固定し、ユーザより引数で指定された高さにより、各段(層)がその割合によって同じ高さで伸び縮みします。 

(3) 拡張型の AI 識別子(ファンクションコード)について

決められている AI 識別子(ファンクションコード)は、何も指定しなくても、Barcode.net がコード体系から判断し自動的に挿入します。

ただし、任意で AI 識別子を挿入する場合は、{AI}を入力してください。

例：(01)商品識別コード＋任意の AI

0100012345678905{AI}10ABC123 ⇒ (01)00012345678905(10)ABC12

(4) 添字の描画

バーコードの下にコードの文字列自体を描画します(既定値)。プロパティの設定で描画をしないようにすることも可能です。

添字を、コードを意味するバーの位置に描画するか(JAN 既定値)、バーコード全体の幅に均等割付するか(通常既定値)を指定することが可能です。

※ JAN(EAN)コードの場合、既定値の状態では商品コードのバーコードのような描画を行い、均等割付にすると書籍コードのバーコードのような描画を行います。

添字のフォントをプロパティで指定することも可能です。

(5) 回転描画

プロパティの設定により、バーコードの左上始点座標を中心に、

90 度／180 度／270 度　回転して描画することが可能です。既定値は 0 度です。

3. アプリケーションプログラムから Barcode.net の使用方法

3-1. クラス仕様

3-1-1. 概要

Barcode.net は、以下のそれぞれバーコードごとに独立したクラスで構成されております。

Pao.BarCode
- **Pao.BarCode.Jan13**
- **Pao.BarCode.Jan8**
- **Pao.BarCode.UPC_A**
- **Pao.BarCode.UPC_E**
- **Pao.BarCode.ITF**
- **Pao.BarCode.Matrix2of5**
- **Pao.BarCode.NW7**
- **Pao.BarCode.Code39**
- **Pao.BarCode.Code93**
- **Pao.BarCode.Code128**
- **Pao.BarCode.GS1_128**
- **Pao.BarCode.EAN128**
- **Pao.BarCode.YubinCustomer**
- **Pao.BarCode.QRCode**
- **Pao.BarCode.DataMatrix**
- **Pao.BarCode.Pdf417**
- **Pao.BarCode. GS1_DataBar_14**
- **Pao.BarCode. GS1_DataBar_Limited**
- **Pao.BarCode. GS1_DataBar_Expanded**

コンビニバーコード以外の一次元バーコードのクラスは、基本的に同一名のプロパティやメソッドといったメンバを所有し、それらの機能も同一です。
そこで以降の各メンバ(メソッドやプロパティ)の説明では、
- 一次元バーコードのクラス
- コンビニバーコードのメンバ(GS1_128 クラス)
- 郵便カスタマバーコードクラス
- 二次元バーコードのクラス（QR コード / DataMatrix / PDF417)
- GS1 データバー(RSS)のクラス(標準型 / 限定型 / 拡張型)

の５つに分けてご説明いたします。

3-1-2. 一次元バーコードクラスメンバ

3-1-2-1. コンストラクタ

初期処理を行う。
バーコードの種類別に以下の２インタフェイスが存在します。
(a) System.Drawing.Graphics オブジェクトへの描画

(1) `public JAN13(System.Drawing.Graphics g)`
(2) `public JAN8(System.Drawing.Graphics g)`
(3) `public UPC_A(System.Drawing.Graphics g)`
(4) `public UPC_E(System.Drawing.Graphics g)`
(5) `public ITF(System.Drawing.Graphics g)`
(6) `public Matrix2of5(System.Drawing.Graphics g)`
(7) `public NEC2of5(System.Drawing.Graphics g)`
(8) `public NW7(System.Drawing.Graphics g)`
(9) `public Code39(System.Drawing.Graphics g)`
(10) `public Code93(System.Drawing.Graphics g)`
(11) `public Code128(System.Drawing.Graphics g)`
(12) `public GS1_128(System.Drawing.Graphics g)`
(13) `public EAN128(System.Drawing.Graphics g)`

・引数
System.Drawing.Graphics g
バーコードの描画を行う **Graphics** を指定します。

(b) 画像ファイルへの描画

(1) public JAN13 (String imgFilePath, ImageFormat imgFormat)
(2) public JAN8(String imgFilePath, ImageFormat imgFormat)
(3) public UPC_A (String imgFilePath, ImageFormat imgFormat)
(4) public UPC_B (String imgFilePath, ImageFormat imgFormat)
(5) public ITF(String imgFilePath, ImageFormat imgFormat)
(6) public Matrix2of5(String imgFilePath, ImageFormat imgFormat)
(7) public NEC2of5(String imgFilePath, ImageFormat imgFormat)
(8) public NW7 (String imgFilePath, ImageFormat imgFormat)
(9) public Code39 (String imgFilePath, ImageFormat imgFormat)
(10) public Code93 (String imgFilePath, ImageFormat imgFormat)
(11) public Code128 (String imgFilePath, ImageFormat imgFormat)
(12) public GS1_128 (String imgFilePath, ImageFormat imgFormat)
(13) public EAN128 (String imgFilePath, ImageFormat imgFormat)

・引数

① String imgFilePath
バーコードを描画するデフォルトファイルパスを指定します。

② ImageFormat imgFormat
出力をする ImageFormat を指定します。プロパティの DPI を有効化する為には、png/jpeg フォーマットを指定してください。

3-1-2-2. メソッド

(1) public void Draw(string code, float x, float y, float width, float height)

バーコードの描画を行います。指定幅を正確に合わせるためにいったん描画したバーコードを縮小して描画しなおします。そのため、多少、精度が劣化します。ドット単位で正確な精度を期待する場合は、指定した幅と正確に一緒にはなりませんが、DrawDirect / DrawDelicate メソッドを使用してください。

・引数

① string code

描画を行うバーコードのコードを文字列で指定します。

Code128 等、改行や TAB といったバイナリコードを入力する場合…

```
string code = "¥n";
これに加えて、
code += ((char)10).ToString();
code += ((char)0x0a).ToString();
```

のような指定が可能です。

少し特殊な GS1-128(UCC/EAN128)において、AI(アプリケーション識別子)挿入方法は 2 通りございます。

a) 可変長項目(データブロック)の後の AI には、FNC1 を挿入
⇒これまで通り"{FNC1}"を付ける。例："{FNC1}21"のようにコードを指定

b) 固定長項目(データブロック)の後の AI には、固定長のため目印の FNC1 は不要
⇒新しく追加した"{AI}"を付ける。例："{AI}21" のようにコードを指定

"{AI}"を指定して FNC1 を挿入しない場合も、カッコ()付コード文字は出力されます。
例えば入力コードに"{AI}21"を指定した場合、添え字には(21)と出力されます。

例) (01)04512345670016(21)1
⇒(01)の前には FNC1 を挿入し(21)の前には挿入しない。
コード指定方法 → "{FNC1}0104512345670016{AI}211"

② float x

描画位置の始点(左上)の X 座標を指定します。
単位は、本メソッド呼び出し時の Grahics.PageUnit の値に依存します。

③ float y

描画位置の始点(左上)の Y 座標を指定します。
単位は、本メソッド呼び出し時の Grahics.PageUnit の値に依存します。

④ float width

バーコードの全体の幅を指定します。
単位は、本メソッド呼び出し時の Grahics.PageUnit の値に依存します。

⑤ float height

バーコードのバーの高さを指定します。
単位は、本メソッド呼び出し時の Grahics.PageUnit の値に依存します。

・戻り値

なし

・例外の種類

予測可能割込発生エラーを以下クラス別に記述します。

○ JAN13

① public errJAN13BadChar()

数字以外の文字が使用されました。
使用できる文字は数字のみです。

② public errJAN13BadLen()

コードの桁数は、13 桁か、12 桁を指定してください。
12 桁の場合チェックキャラクタを自動付与します。

③ public errJAN13CheckDigit()

コード末尾のチェックデジットが誤っています。

○ JAN8

④ public errJAN8BadChar()

数字以外の文字が使用されました。
使用できる文字は数字のみです。

⑤ public errJAN8BadLen()

コードの桁数は、8 桁か、7 桁を指定してください。
12 桁の場合チェックキャラクタを自動付与します。

⑥ public errJAN8CheckDigit()

コード末尾のチェックデジットが誤っています。

○ UPC-A

⑦ public errUPC_ABadChar()

数字以外の文字が使用されました。
使用できる文字は数字のみです。

⑧ public errUPC_ABadLen()

コードの桁数は、12 桁か、11 桁を指定してください。
11 桁の場合チェックキャラクタを自動付与します。

⑨ public errUPC_ACheckDigit()

コード末尾のチェックデジットが誤っています。

○ UPC-E

⑩ public errUPC_EBadChar()

数字以外の文字が使用されました。
使用できる文字は数字のみです。

⑪ public errUPC_EBadLen()

コードの桁数は、6 桁か、7 桁か、8 桁を指定してください。
6 桁の場合、頭に 0 と末尾にチェックキャラクタを自動付与します。
7 桁の場合、末尾にチェックキャラクタを自動付与します。

⑫ public errUPC_ECheckDigit()
コード末尾のチェックデジットが誤っています。
(８桁の数値が指定されている時のエラーです。)

⑬ public errUPC_E_BadCodeForCheckDigit ()
チェックディジットを計算できるコード体系ではありません。
１桁目=0 / 7 桁目=0～9 でなければいけません。
(７桁以上の数値が指定されている時のエラーです。)

○ ITF

⑭ public errITFBadChar()
数字以外の文字が使用されました。
使用できる文字は数字のみです。

○ Matrix2of5

⑮ public errMatrix2of5BadChar()
数字以外の文字が使用されました。
使用できる文字は数字のみです。

○ NEC2of5

⑯ public errNEC2of5BadChar()
数字以外の文字が使用されました。
使用できる文字は数字のみです。

○ NW7

⑰ public errNW7BadChar()
利用できない文字 ='? ' が使用されました。
使用できる文字は"ABCD.+:/$-0123456789"です。

○ Code39

⑱ public errCode39BadChar()
利用できない文字 ='? ' が使用されました。
使用できる文字は
"1234567890ABCDEFGHIJKLMNOPQRSTUVWXYZ-. *$/+%"です。

○ Code128 / GS1_128 / EAN128
予測可能割り込み発生エラーなし。

(2) public void Draw(string code, float x, float y, float width, float height, string imgFilePath)

(1)の Draw メソッドのオーバーロードメソッドでバーコードを画像ファイルに出力する。

imgFilePath で指定したファイルへバーコードの描画を行います。
ver 2.4 で追加されたオーバーロードメソッドです。
他の機能について Draw メソッドと同様となります。

・引数

① string code

描画を行うバーコードのコードを文字列で指定します。
Code128 等、改行や TAB といったバイナリコードを入力する場合…

```
string code = "¥n";
```

これに加えて、

```
code += ((char)10).ToString();
code += ((char)0x0a).ToString();
```

のような指定が可能です。

少し特殊な GS1-128(UCC/EAN128)において、AI(アプリケーション識別子)挿入方法は 2 通りございます。

a) 可変長項目(データブロック)の後の AI には、FNC1 を挿入
⇒これまで通り"{FNC1}"を付ける。例："{FNC1}21"のようにコードを指定

b) 固定長項目(データブロック)の後の AI には、固定長のため目印の FNC1 は不要
⇒新しく追加した"{AI}"を付ける。例："{AI}21" のようにコードを指定

"{AI}"を指定して FNC1 を挿入しない場合も、カッコ()付コード文字は出力されます。
例えば入力コードに"{AI}21"を指定した場合、添え字には(21)と出力されます。

例) (01)04512345670016(21)1
⇒(01)の前には FNC1 を挿入し(21)の前には挿入しない。
コード指定方法 → "{FNC1}0104512345670016{AI}211"

② float x

0 のみが指定可能です。

③ float y

0 のみが指定可能です。

④ float width

バーコードの全体の幅を指定します。
単位は、本メソッド呼び出し時の ImgDrawUnit プロパティ値に依存します。

⑤ float height

バーコードのバーの高さを指定します。
単位は、本メソッド呼び出し時の ImgDrawUnit プロパティ値に依存します。

⑥ string imgFilePath

出力を行うファイルパスを指定します。

イメージファイルの DPI は本メソッド呼び出し時の ImgDpi プロパティ値に依存します。

・戻り値
なし

・例外の種類
Draw メソッドと同様。

public void DrawDirect(string code, float x, float y, float width, float height)

バーコードの描画を行います。
指定幅以内で最も広い幅でバーコードを直接描画します。
ドット単位での描画精度を実現します。

・引数

① **string code**

描画を行うバーコードのコードを文字列で指定します。
Code128 等、改行や TAB といったバイナリコードを入力する場合…

```
string code = "¥n";
これに加えて、
code += ((char)10).ToString();
code += ((char)0x0a).ToString();
```

のような指定が可能です。

少し特殊な GS1-128(UCC/EAN128)において、AI(アプリケーション識別子)挿入方法は 2 通りございます。

a) 可変長項目(データブロック)の後の AI には、FNC1 を挿入
⇒これまで通り"{FNC1}"を付ける。例："{FNC1}21"のようにコードを指定

b) 固定長項目(データブロック)の後の AI には、固定長のため目印の FNC1 は不要
⇒新しく追加した"{AI}"を付ける。例："{AI}21" のようにコードを指定

"{AI}"を指定して FNC1 を挿入しない場合も、カッコ()付コード文字は出力されます。
例えば入力コードに"{AI}21"を指定した場合、添え字には(21)と出力されます。

例) (01)04512345670016(21)1
⇒(01)の前には FNC1 を挿入し(21)の前には挿入しない。
コード指定方法 → "{FNC1}0104512345670016{AI}211"

② **float x**

描画位置の始点(左上)の X 座標を指定します。
単位は、本メソッド呼び出し時の Grahics.PageUnit の値に依存します。

③ **float y**

描画位置の始点(左上)の Y 座標を指定します。
単位は、本メソッド呼び出し時の Grahics.PageUnit の値に依存します。

④ **float width**

バーコードの全体の幅を指定します。
指定した幅以内で最も広い幅のバーコードを描画します。
単位は、本メソッド呼び出し時の Grahics.PageUnit の値に依存します。

⑤ **float height**

バーコードのバーの高さを指定します。
単位は、本メソッド呼び出し時の Grahics.PageUnit の値に依存します。

・戻り値

なし

・例外の種類

Draw メソッドと同様。

(3) public void **DrawDirect**(string code, float x, float y, float width, float height, string imgFilePath)

(3)の DrawDirect メソッドのオーバーロードメソッドでバーコードを画像ファイルに出力する。

imgFilePath で指定したファイルへバーコードの描画を行います。
ver 2.4 で追加されたオーバーロードメソッドです。
他の機能について DrawDirect メソッドと同様となります。

・引数

① **string code**

描画を行うバーコードのコードを文字列で指定します。
Code128 等、改行や TAB といったバイナリコードを入力する場合…

```
string code = "¥n";
```

これに加えて、

```
code += ((char)10).ToString();
code += ((char)0x0a).ToString();
```

のような指定が可能です。

少し特殊な GS1-128(UCC/EAN128)において、AI(アプリケーション識別子)挿入方法は 2 通りございます。

a) 可変長項目(データブロック)の後の AI には、FNC1 を挿入
⇒これまで通り"{FNC1}"を付ける。例："{FNC1}21"のようにコードを指定

b) 固定長項目(データブロック)の後の AI には、固定長のため目印の FNC1 は不要
⇒新しく追加した"{AI}"を付ける。例："{AI}21" のようにコードを指定

"{AI}"を指定して FNC1 を挿入しない場合も、カッコ()付コード文字は出力されます。
例えば入力コードに"{AI}21"を指定した場合、添え字には(21)と出力されます。

例) (01)04512345670016(21)1
⇒(01)の前には FNC1 を挿入し(21)の前には挿入しない。
コード指定方法 → "{FNC1}0104512345670016{AI}211"

② **float x**

0 のみが指定可能です。

③ **float y**

0 のみが指定可能です。

④ **float width**

バーコードの全体の幅を指定します。
単位は、本メソッド呼び出し時の **ImgDrawUnit プロパティ**値に依存します。

⑤ **float height**

バーコードのバーの高さを指定します。
単位は、本メソッド呼び出し時の **ImgDrawUnit プロパティ**値に依存します。

⑥ **string imgFilePath**

出力を行うファイルパスを指定します。

イメージファイルの DPI は本メソッド呼び出し時の ImgDpi プロパティ値に依存します。

・戻り値
なし

・例外の種類
Draw メソッドと同様。

(4) **public void DrawDelicate(string code, float x, float y, float minLineWidth, float height)**

バーコードの描画を行います。

Draw メソッドとの違いは、バーコード全体の幅を指定するのではなく、バーを描画する一番細い線の幅を指定します。**Draw** メソッドに比べて、**DrawDirect** メソッドと同様に精度の高いバーコードを描画することが可能です。ただし、バーコード全体の幅の調整が必要になります。

※この **DrawDelicate** メソッドを使用してバーの最小幅を指定した場合に精度が高くなる理由は、直接、Graphics オブジェクトに線を描画するためです。

Draw メソッドを使用して全体の幅を指定した場合、一旦、仮想空間に描画したバーコードを指定された全体の幅に対して当てはまるように Graphics オブジェクトに縮小描画しております。

・引数

① **string code**

描画を行うバーコードのコードを文字列で指定します。

Code128 等、改行や TAB といったバイナリコードを入力する場合…

```
string code = "¥n";
これに加えて、
code += ((char)10).ToString();
code += ((char)0x0a).ToString();
```

のような指定が可能です。

少し特殊な GS1-128(UCC/EAN128)において、AI(アプリケーション識別子)挿入方法は 2 通りございます。

a) 可変長項目(データブロック)の後の AI には、FNC1 を挿入
⇒これまで通り"{FNC1}"を付ける。例："{FNC1}21"のようにコードを指定

b) 固定長項目(データブロック)の後の AI には、固定長のため目印の FNC1 は不要
⇒新しく追加した"{AI}"を付ける。例："{AI}21" のようにコードを指定

"{AI}"を指定して FNC1 を挿入しない場合も、カッコ()付コード文字は出力されます。
例えば入力コードに"{AI}21"を指定した場合、添え字には(21)と出力されます。

例) (01)04512345670016(21)1
⇒(01)の前には FNC1 を挿入し(21)の前には挿入しない。
コード指定方法 → "{FNC1}0104512345670016{AI}211"

② **float x**

描画位置の始点(左上)の X 座標を指定します。
単位は、本メソッド呼び出し時の Grahics.PageUnit の値に依存します。

③ **float y**

描画位置の始点(左上)の Y 座標を指定します。
単位は、本メソッド呼び出し時の Grahics.PageUnit の値に依存します。

④ **float minLineWidth**

バーコードを描画するバーの最小幅の値を指定します。
単位は、本メソッド呼び出し時の Grahics.PageUnit の値に依存します。

⑤ float height

バーコードのバーの高さを指定します。

単位は、本メソッド呼び出し時の Grahics.PageUnit の値に依存します。

・戻り値

なし

・例外の種類

Draw メソッドと同様。

(5) public void DrawDelicate(string code, float x, float y, float width, float height, string imgFilePath)

(5)の DrawDelicate メソッドのオーバーロードメソッドでバーコードを画像ファイルに出力する。

imgFilePath で指定したファイルへバーコードの描画を行います。他の機能について DrawDelicate メソッドと同様となります。

・引数

① string code

描画を行うバーコードのコードを文字列で指定します。

Code128 等、改行や TAB といったバイナリコードを入力する場合…

string code = "¥n";

これに加えて、

code += ((char)10).ToString();

code += ((char)0x0a).ToString();

のような指定が可能です。

少し特殊な GS1-128(UCC/EAN128)において、AI(アプリケーション識別子)挿入方法は 2 通りございます。

a) 可変長項目(データブロック)の後の AI には、FNC1 を挿入

⇒これまで通り"{FNC1}"を付ける。例："{FNC1}21"のようにコードを指定

b) 固定長項目(データブロック)の後の AI には、固定長のため目印の FNC1 は不要

⇒新しく追加した"{AI}"を付ける。例："{AI}21" のようにコードを指定

"{AI}"を指定して FNC1 を挿入しない場合も、カッコ()付コード文字は出力されます。

例えば入力コードに"{AI}21"を指定した場合、添え字には(21)と出力されます。

例) (01)04512345670016(21)1

⇒(01)の前には FNC1 を挿入し(21)の前には挿入しない。

コード指定方法 → "{FNC1}0104512345670016{AI}211"

② float x

0 のみが指定可能です。

③ float y

0 のみが指定可能です。

④ float width

バーコードの全体の幅を指定します。

単位は、本メソッド呼び出し時の ImgDrawUnit プロパティ値に依存します。

⑤ float height

バーコードのバーの高さを指定します。

単位は、本メソッド呼び出し時の ImgDrawUnit プロパティ値に依存します。

⑥ string imgFilePath

出力を行うファイルパスを指定します。

イメージファイルの DPI は本メソッド呼び出し時の ImgDpi プロパティ値に依存します。

・戻り値
なし

・例外の種類
Draw メソッドと同様。

(6) public void WriteSVG
(string code, float x, float y, float width, float height , string filePath)
SVG ファイルへのバーコードの出力を行います。

・引数

① string code
描画を行うバーコードのコードを文字列で指定します。少し特殊な Code128 等、改行や TAB といったバイナリコードを入力する場合…

```
string code = "¥n";
これに加えて、
code += ((char)10).ToString();
code += ((char)0x0a).ToString();
```

のような指定が可能です。

GS1-128(UCC/EAN128)において、AI(アプリケーション識別子)挿入方法は 2 通りございます。

c) 可変長項目(データブロック)の後の AI には、FNC1 を挿入
⇒これまで通り"{FNC1}"を付ける。例："{FNC1}21"のようにコードを指定

d) 固定長項目(データブロック)の後の AI には、固定長のため目印の FNC1 は不要
⇒新しく追加した"{AI}"を付ける。例："{AI}21" のようにコードを指定

"{AI}"を指定して FNC1 を挿入しない場合も、カッコ()付コード文字は出力されます。
例えば入力コードに"{AI}21"を指定した場合、添え字には(21)と出力されます。

例) (01)04512345670016(21)1
⇒(01)の前には FNC1 を挿入し(21)の前には挿入しない。
コード指定方法 → "{FNC1}0104512345670016{AI}211"

② float x
描画位置の始点(左上)の X 座標を指定します。
単位は、本メソッド呼び出し時の Grahics.PageUnit の値に依存します。

③ float y
描画位置の始点(左上)の Y 座標を指定します。
単位は、本メソッド呼び出し時の Grahics.PageUnit の値に依存します。

④ float width
バーコードの全体の幅を指定します。
単位は、本メソッド呼び出し時の Grahics.PageUnit の値に依存します。
ただし、SVG ファイルは、ブラウザ表示用のため、画面の pixel とバーコードの線が一致しないといけないため、指定された幅以下のサイズに最適化されます。

⑤ float height
バーコードのバーの高さを指定します。
単位は、本メソッド呼び出し時の Grahics.PageUnit の値に依存します。

⑥ string filePath

SVG ファイルのファイル名をフルパスで指定してください。

・戻り値

なし

・例外の種類

Draw メソッドと同様。

3-1-2-3. プロパティ

(1) public bool TextWrite
　　true： 添字の描画を行う。(既定値)
　　false： 添字を描画しない。

(2) public bool TextKintou
　　true： 添字の描画は、バーコード全体の幅に均等割付で行う。
　　false： 添字を描画は、コードを意味するバーの位置に行う。(既定値)

(3) public Font TextFont
　　添字のフォント。
　　GS1_128 / EAN128 について既定値は、”ＭＳ ゴシック　8 ポイント　標準”。
　　それ以外のバーコード既定値は、”ＭＳ ゴシック　9 ポイント　標準”。

(4) public float RotateAngle
　　回転角度を数値で指定。左下を軸に右回転して描画を行う。
　　既定値は、０度。

(5) public bool DispStartStopCode
　　Code39/NW7 のみ使用可能なプロパティ
　　true： スタート／ストップコードの描画を行う。
　　false： スタート／ストップコードを描画しない。(既定値)

(6) public int KuroBarChousei
　　黒バーの幅を微細調整(加減)するドット数を指定する。
　　マイナスの値で黒バーを細くすることも可能。
　　既定値は、0(pixel)。

(7) public int ShiroBarChousei
　　白バーの幅を微細調整(加減)するドット数を指定する。
　　マイナスの値で白バーを細くすることも可能。
　　既定値は、0(pixel)。

　　→インクジェットプリンタで黒バーがにじんで太くなる時などに有効
　　例：コンビニバーコード、EPSON PX-502A(360DPI)のプリンタの場合、
　　　白バーを＋１すると丁度良い。

(8) public GraphicsUnit ImgDrawUnit

イメージファイル出力時の座標単位を指定する。
バーコード画像ファイル出力時にのみ有効。
既定値は、GraphicsUnit.Pixel。
他、mm(ミリメートル) / inch(インチ) / point 等指定可能。

(9) public float ImgDpi

イメージファイル出力時の DPI を指定する。
バーコード画像ファイル出力時にのみ有効。
既定値は、600。

◇次のプロパティは、CODE128 / GS1_128(EAN128)のみで使用するプロパティです。

(1) public CodeSet128 CodeABC

CODE128 のコードセットには、次の種類があります。

CODE-A： フルアスキー
CODE-B: 1 桁の アルファ・ニューメリック
CODE-C: 2 桁の数字

本プロパティでは、次の指定が可能です。

列挙値 Pao.BarCode.CodeSet128	
AUTO	Barcode.net が自動でコードセットを組み合わせ最小幅のバーコードとする。
CODE_A	CODE-A
CODE_B	CODE-B
CODE_C	CODE-C

3-1-2-4. GS1_128 コンビニバーコードメソッド

(1) public void DrawConvenience(string code, float x, float y, float height)

コンビニバーコードの描画を行います。

「GS1-128 標準料金代理収納ガイドライン(財団法人 流通システム開発センター)」

の記述に準拠し、通常使うプリンタの DPI から、バーコードの幅を自動的に決定しています。以下の表の通りです。

プリンタ解像度	モジュール幅		バーコード部の幅
	ドット	mm	
300dpi	2	0.169	48.67mm
400dpi	3	0.19	54.72mm
480dpi	3	0.158	45.50mm
600dpi	4	0.169	48.67mm
300dpi の倍数	2 の倍数	0.169	48.67mm

・引数

① string code

描画を行うバーコードのコードを文字列で指定します。

アプリケーション識別子(AI)の前に"{FNC1}"という文字列を指定することで GS1-128(UCC/EAN128)のバーコード作成を可能としています。

例) (91)912345 ・・・() 付きがアプリケーション識別子(AI)

コード指定方法→「{FNC1}91912345」

② float x

描画位置の始点(左上)の X 座標を指定します。

単位は、mm(ミリ)です。

③ float y

描画位置の始点(左上)の Y 座標を指定します。

単位は、mm(ミリ)です。

④ float height

バーコードのバーの高さを指定します。

単位は、mm(ミリ)です。

・戻り値

なし

(2) public void DrawConvenience(string code, float x, float y, float width, float height)

(1)の DrawConvenience メソッドのオーバーロードメソッドで、バーコード幅指定可能。

コンビニバーコードの描画を行います。
(1)の DrawConvenience メソッドと異なり、バーコードの幅を指定することが可能です。
「UCC/EAN-128 標準料金代理収納ガイドライン(財団法人 流通システム開発センター)」は、あくまでガイドですので、こちらのオーバーロードしたメソッドで自在な幅を指定したバーコードを描画して頂くことができます。
ver 2.4 で追加されたオーバーロードメソッドです。

・引数

① string code
描画を行うバーコードのコードを文字列で指定します。
アプリケーション識別子(AI)の前に"{FNC1}"という文字列を指定することで GS1-128(UCC/EAN128)のバーコード作成を可能としています。
例) (91)912345 ・・・() 付きがアプリケーション識別子(AI)
コード指定方法→「{FNC1}91912345」

② float x
0 のみが指定可能です。(画像ファイルに座標は不要なため)

③ float y
0 のみが指定可能です。(画像ファイルに座標は不要なため)

④ float width
バーコードのバーの高さを指定します。
単位は、mm(ミリ)です。

⑤ float height
バーコードのバーの高さを指定します。
単位は、mm(ミリ)です。

⑥ string imgFilePath
出力を行うファイルパスを指定します。
イメージファイルの DPI は本メソッド呼び出し時の ImgDpi プロパティ値に依存します。

・戻り値
なし

(3) public void DrawConvenience(string code, float x, float y, float width, float height, string imgFilePath)

(1)の DrawConvenience メソッドのオーバーロードメソッドで、バーコードを画像ファイルに出力する。

imgFilePath で指定したファイルへバーコードの描画を行います。
ver 2.4 で追加されたオーバーロードメソッドです。
他の機能について DrawConvenience メソッドと同様となります。
「UCC/EAN-128 標準料金代理収納ガイドライン(財団法人 流通システム開発センター)」
の記述に準拠し、通常使うプリンタの DPI から、バーコードの幅を自動的に決定しています。以下の表の通りです。

プリンタ解像度	モジュール幅		バーコード部の幅
	ドット	mm	
300dpi	2	0.169	48.67mm
400dpi	3	0.19	54.72mm
480dpi	3	0.158	45.50mm
600dpi	4	0.169	48.67mm
300dpi の倍数	2 の倍数	0.169	48.67mm

・引数

⑦ string code
描画を行うバーコードのコードを文字列で指定します。
アプリケーション識別子(AI)の前に"{FNC1}"という文字列を指定することで GS1-128(UCC/EAN128)のバーコード作成を可能としています。
例) (91)912345 ・・・() 付きがアプリケーション識別子(AI)
コード指定方法→「{FNC1}91912345」

⑧ float x
0 のみが指定可能です。(画像ファイルに座標は不要なため)

⑨ float y
0 のみが指定可能です。(画像ファイルに座標は不要なため)

⑩ float height
バーコードのバーの高さを指定します。
単位は、mm(ミリ)です。

⑪ string imgFilePath
出力を行うファイルパスを指定します。
イメージファイルの DPI は本メソッド呼び出し時の ImgDpi プロパティ値に依存します。

・戻り値
なし

(4) public void DrawConvenience(string code, float x, float y, float width, float height, string imgFilePath)

(2)の DrawConvenience メソッドのオーバーロードメソッドで、バーコードを画像ファイルに出力する。

imgFilePath で指定したファイルへバーコードの描画を行います。他の機能について DrawConvenience メソッドと同様となります。
「UCC/EAN-128 標準料金代理収納ガイドライン(財団法人 流通システム開発センター)」
は、あくまでガイドですので、こちらのオーバーロードしたメソッドで自在な幅を指定したバーコードを描画して頂くことができます。
ver 2.4 で追加されたオーバーロードメソッドです。

・引数

⑫ string code
描画を行うバーコードのコードを文字列で指定します。
アプリケーション識別子(AI)の前に"{FNC1}"という文字列を指定することで GS1-128(UCC/EAN128)のバーコード作成を可能としています。
例) (91)912345 ・・・() 付きがアプリケーション識別子(AI)
コード指定方法→「{FNC1}91912345」

⑬ float x
0 のみが指定可能です。(画像ファイルに座標は不要なため)

⑭ float y
0 のみが指定可能です。(画像ファイルに座標は不要なため)

⑮ float height
バーコードのバーの高さを指定します。
単位は、mm(ミリ)です。

⑯ string imgFilePath
出力を行うファイルパスを指定します。
イメージファイルの DPI は本メソッド呼び出し時の ImgDpi プロパティ値に依存します。

・戻り値
なし

3-1-2-5. 使用プロパティ

(1) public Font TextFont

添字のフォント。

既定値は、”ＭＳ ゴシック　8 ポイント　標準”。

(2) public bool TextWrite

true： 添字の描画を行う。(既定値)

false： 添字を描画しない。

(3) public float ImgDpi

イメージファイル出力時の DPI を指定する。

既定値は、600。

3-1-3. GS1 データバー(RSS) クラスメンバ

GS1 データバーのクラスのインタフェイス(コンストラクタ・メソッド・プロパティ)は、基本的に一次元バーコードのものと変わりません。クラス自体も同じ Interface で実装し、特別な機能のプロパティのみ追加されております。
従って、メッソッドやプロパティの説明は、冗長にならない目的のためにも割愛させていただいている部分がございます。

3-1-3-1. コンストラクタ

初期処理を行う。
バーコードの種類別に以下の２インタフェイスが存在します。

(a) System.Drawing.Graphics オブジェクトへの描画

(1) `public GS1_DataBar_14 (System.Drawing.Graphics g)`

(2) `public GS1_DataBar_Limited (System.Drawing.Graphics g)`

(3) `public GS1_DataBar_Expanded (System.Drawing.Graphics g)`

・引数

`System.Drawing.Graphics g`
バーコードの描画を行う `Graphics` を指定します。

(b) 画像ファイルへの描画

(1) `public GS1_DataBar_14 (String imgFilePath, ImageFormat imgFormat)`

(2) `public GS1_DataBar_Limited (String imgFilePath, ImageFormat imgFormat)`

(3) `public GS1_DataBar_Expanded(String imgFilePath, ImageFormat imgFormat)`

・引数

③ String imgFilePath
バーコードを描画するデフォルトファイルパスを指定します。

④ ImageFormat imgFormat
出力をする ImageFormat を指定します。プロパティの DPI を有効化する為には、png/jpeg フォーマットを指定してください。

3-1-3-2. メソッド

(1) public void Draw(string code, float x, float y, float width, float height)

バーコードの描画を行います。指定幅を正確に合わせるためにいったん描画したバーコードを縮小して描画しなおします。そのため、多少、精度が劣化します。ドット単位で正確な精度を期待する場合は、指定した幅と正確に一緒にはなりませんが、DrawDirect / DrawDelicate メソッドを使用してください。

・引数

① string code

描画を行うバーコードのコードを文字列で指定します。

《拡張型の AI 識別子(ファンクションコード)について》
決められている AI 識別子(ファンクションコード)は、何も指定しなくとも、コード体系から自動的に挿入されます。
ただし、任意で AI 識別子を挿入する場合は、{AI}を入力してください。
例：(01)商品識別コード＋任意の AI
0100012345678905{AI}10ABC123 ⇒ (01)00012345678905(10)ABC12

② float x

描画位置の始点(左上)の X 座標を指定します。
単位は、本メソッド呼び出し時の Grahics.PageUnit の値に依存します。

② float y

描画位置の始点(左上)の Y 座標を指定します。
単位は、本メソッド呼び出し時の Grahics.PageUnit の値に依存します。

③ float width

バーコードの全体の幅を指定します。
単位は、本メソッド呼び出し時の Grahics.PageUnit の値に依存します。

④ float height

バーコードのバーの高さを指定します。
単位は、本メソッド呼び出し時の Grahics.PageUnit の値に依存します。

GS1 RSS の高さについては、Barcoe.net 独自に次の仕様がございます。

◇ 通常の一次元バーコードと同じ

種類：標準一層型 / 限定型 / 拡張一層型

引数で指定された高さは、そのままバーコードの高さとなります。

◇ 各層が均一の伸縮

種類：標準二層型 / 拡張多層型

各段(層)の割合は同一な仕様です。段と段(層と層)間の小さな正方形部分の高さ

を固定し、引数で指定された高さを全体の高さ都市、各段(層)がその割合によって同じ高さで伸び縮みします。

◊ 2 層とセンター部の割合が固定

種類：二層型(標準)

バーコード上段・センター部分・下段の高さ割合が決まっているため、高さの指定があっても割合をキープして出力することが基本です。

Barcode.net では、DrawDirect / DrawDelicate メソッドを使用した時に、センター部分を含む 3 段の割合を確実にキープします。

本 Draw メソッドの場合は、センターの小さな正方形群の各正方形割合はキープしながら、ユーザより引数で指定された高さにより上段・下段の割合に従って上段下段の高さを決定します。従って Draw メソッドでは、本来決まっているセンター部分を含む 3 段の高さ割合は保たれません。指定された高さにより、上・下段が決められている割合を保ちながら伸び縮みします。

・戻り値

なし

(2) `public void Draw`
`(string code, float x, float y, float width, float height, string imgFilePath)`

imgFilePath で指定したファイルへバーコードの描画を行います。
他の引数については、(1)の Draw メソッドと同様ですので割愛します。

(3) `public void DrawDirect`
`(string code, float x, float y, float width, float height)`

バーコードの描画を行います。
指定幅以内で最も広い幅でバーコードを直接描画します。
ドット単位での描画精度を実現します。
引数については、(1)の Draw メソッドと同様ですので割愛します。

(4) `public void DrawDirect`
`(string code, float x, float y, float width, float height, string imgFilePath)`

imgFilePath で指定したファイルへバーコードの描画を行います。
バーコードの描画を行います。
指定幅以内で最も広い幅でバーコードを直接描画します。
ドット単位での描画精度を実現します。
引数については、(1)の Draw メソッドと同様ですので割愛します。

(5) **public void DrawDelicate**
(string code, float x, float y, float minLineWidth, float height)

バーコードの描画を行います。
Draw メソッドとの違いは、バーコード全体の幅を指定するのではなく、バーを描画する一番細い線の幅を指定します。**Draw** メソッドに比べて、**DrawDirect** メソッドと同様に精度の高いバーコードを描画することが可能です。ただし、バーコード全体の幅の調整が必要になります。

※この **DrawDelicate** メソッドを使用してバーの最小幅を指定した場合に精度が高くなる理由は、直接、Graphics オブジェクトに線を描画するためです。

Draw メソッドを使用して全体の幅を指定した場合、一旦、仮想空間に描画したバーコードを指定された全体の幅に対して当てはまるように Graphics オブジェクトに縮小描画しております。

minLineWidth 以外の引数は、(1) Draw メソッドと同様ですので割愛します。

・引数
float minLineWidth

・戻り値
なし

(6) **public** void **DrawDelicate**(string code, float x, float y, float width, float height, string imgFilePath)

imgFilePath で指定したファイルへバーコードの描画を行います。他の機能については、(5) **public void DrawDelicate** と同様ですので割愛します。

3-1-3-3. プロパティ

◇次のプロパティは、一次元バーコードと同様です。
クラスとしても同じ Interface を実装しています。

(1) `public bool TextWrite`
添字の描画を行う。(既定値: true)

(2) `public bool TextKintou`
添字を描画は、コードを意味するバーの位置に行う。(既定値: false)

(3) `public Font TextFont`
添字のフォント。(既定値: MS ゴシック 9pt)

(4) `public float RotateAngle`
回転角度を数値で指定。左下を軸に右回転して描画を行う。(既定値は: 0)

(5) `public int KuroBarChousei`
黒バーの幅を微細調整(加減)するドット数を指定する。(既定値: 0)

(6) `public int ShiroBarChousei`
白バーの幅を微細調整(加減)するドット数を指定する。(既定値: 0)

(7) `public GraphicsUnit ImgDrawUnit`
イメージファイル出力時の座標単位を指定する。
バーコード画像ファイル出力時にのみ有効。

(8) `public float ImgDpi`
イメージファイル出力時の DPI を指定する。
バーコード画像ファイル出力時にのみ有効。既定値は、600。

◇次のプロパティは、GS1 データバー(RSS)のみで使用するものです。

(1) `public DatabaType SymbolType`
GS1 データバーのタイプです。
限定型では使用しません。標準型・拡張型で次のように定義されています。
出力するデータバーのタイプを標準型・拡張型、各々で指定します。

GS1 Databa RSS 14 のタイプ	
`enum Pao.BarCode.GS1_DataBar_14.DatabarType`	
OMNIDIRECTIONAL	標準型
STACKED	二層型
STACKED_OMNIDIRECTIONAL	標準二層型

GS1 Databa RSS Expanded のタイプ	
`enum Pao.BarCode.GS1_DataBar_Expanded.DatabarType`	
UNSTACKED	一層型
STACKED	多層型

3-1-4. 郵便カスタマバーコードクラスメンバ

3-1-4-1. コンストラクタ

初期処理を行う。

public YubinCustomer(System.Drawing.Graphics g)
通常のコンストラクタ：System.Drawing.Graphics オブジェクトにバーコードを描画する際に使用してください。(印刷・画面へのバーコード出力)

・引数
　System.Drawing.Graphics g
　バーコードの描画を行う **Graphics** を指定します。
・戻り値
　なし。
・割込発生エラー
　なし。

public YubinCustomer (String imgFilePath, ImageFormat imgFormat)
画像ファイルにバーコードを出力する際に使用してください。
ver 2.4 で追加されたコンストラクタです。

・引数
① String imgFilePath
　バーコードを描画するデフォルトファイルパスを指定します。
② ImageFormat imgFormat
　出力をする ImageFormat を指定します。プロパティの DPI を有効化する為には、png/jpeg フォーマットを指定してください。
・戻り値
　なし。
・割込発生エラー
　なし。

3-1-4-2. メソッド

(1) public void Draw(string code, float x, float y, float point)
バーコードの描画を行います。

・引数
① string code
描画を行うバーコードのコードを文字列で指定します。
コードは・・・
[郵便番号の数字部分 7 桁]+[郵便番号では不明部分の住所の英数字を「－」区切り]
で、指定してください。
例）〒116-0013 東京都荒川区西日暮里五丁目 37 番 5 号スタートアップオフィスＡ－207 号室
コード指定方法→「11600135-37-5-A-207」
② float x
描画位置の始点(左上)の X 座標を指定します。
単位は、本メソッド呼び出し時の Grahics.PageUnit の値に依存します。
③ float y
描画位置の始点(左上)の Y 座標を指定します。
単位は、本メソッド呼び出し時の Grahics.PageUnit の値に依存します。
④ float point
バーコード大きさを表すポイントを指定します。
ポイントは、8～11.5 の範囲内の数値で指定してください。

・戻り値
なし

・例外の種類
public errYubinBadChar ()
半角英数字－(ハイフォン)以外の文字が使用されました。

(2) public void Draw(string code, float x, float y, float point , string imgFilePath)

(1)の Draw メソッドのオーバーロードメソッドで、バーコードを画像ファイルに出力する。

imgFilePath で指定したファイルへバーコードの描画を行います。
ver 2.4 で追加されたオーバーロードメソッドです。
他の機能について Draw メソッドと同様となります。

・引数

① string code
描画を行うバーコードのコードを文字列で指定します。
コードは・・・
[郵便番号の数字部分 7 桁]+[郵便番号では不明部分の住所の英数字を「－」区切り]
で、指定してください。
例）〒116-0013 東京都荒川区西日暮里五丁目 37 番 5 号スタートアップオフィスＡ－207 号室
コード指定方法→「11600135-37-5-A-207」

② float x
0 のみが指定可能です。

③ float y
0 のみが指定可能です。

④ float point
バーコード大きさを表すポイントを指定します。
ポイントは、8～11.5 の範囲内の数値で指定してください。

⑤ string imgFilePath
出力を行うファイルパスを指定します。
イメージファイルの DPI は本メソッド呼び出し時の ImgDpi プロパティ値に依存します。

・戻り値
なし

・例外の種類
public errYubinBadChar ()
半角英数字－(ハイフォン)以外の文字が使用されました。

```
public void WriteSVG
    (string code, float x, float y, float point , string filePath)
```

SVG ファイルへのバーコードの出力を行います。

・引数

① string code

描画を行うバーコードのコードを文字列で指定します。

コードは・・・

[郵便番号の数字部分 7 桁]+[郵便番号では不明部分の住所の英数字を「－」区切り]

で、指定してください。

例）〒116-0013 東京都荒川区西日暮里五丁目 37 番 5 号スタートアップオフィスＡ－207 号室

コード指定方法→「11600135-37-5-A-207」

② float x

描画位置の始点(左上)の X 座標を指定します。

単位は、本メソッド呼び出し時の Grahics.PageUnit の値に依存します。

③ float y

描画位置の始点(左上)の Y 座標を指定します。

単位は、本メソッド呼び出し時の Grahics.PageUnit の値に依存します。

④ float point

バーコード大きさを表すポイントを指定します。

ポイントは、8～11.5 の範囲内の数値で指定してください。

⑤ string filePath

SVG ファイルのファイル名をフルパスで指定してください。

・戻り値

なし

・例外の種類

public errYubinBadChar ()

半角英数字－(ハイフォン)以外の文字が使用されました。

3-1-4-3. プロパティ

(1) `public float RotateAngle`

回転角度を数値で指定。左下を軸に右回転して描画を行う。
既定値は、0 度。

(2) `public float ImgDpi`

イメージファイル出力時の DPI を指定する。
既定値は、600。

3-1-5. QR コード / DataMatrix / PDF417 クラスメンバ

3-1-5-1. QR コードコンストラクタ

初期処理を行う。

public QRCode(System.Drawing.Graphics g)

・引数

System.Drawing.Graphics g

QR コードの描画を行う **Graphics** を指定します。

・戻り値

なし。

・割込発生エラー

なし。

public QRCode (String imgFilePath, ImageFormat imgFormat)

画像ファイルに QR コードを出力する際に使用してください。

・引数

① String imgFilePath

バーコードを描画するデフォルトファイルパスを指定します。

② ImageFormat imgFormat

出力をする ImageFormat を指定します。プロパティの DPI を有効化する為には、png/jpeg フォーマットを指定してください。

・戻り値

なし。

・割込発生エラー

なし。

3-1-5-2. DataMatrix コンストラクタ

初期処理を行う。

public DataMatrix(System.Drawing.Graphics g)

・引数

System.Drawing.Graphics g

DataMatrix の描画を行う **Graphics** を指定します。

・戻り値

なし。

・割込発生エラー

なし。

public DataMatrix (String imgFilePath, ImageFormat imgFormat)

画像ファイルに DataMatrix を出力する際に使用してください。

・引数

① String imgFilePath

バーコードを描画するデフォルトファイルパスを指定します。

② ImageFormat imgFormat

出力をする ImageFormat を指定します。プロパティの DPI を有効化する為には、png/jpeg フォーマットを指定してください。

・戻り値

なし。

・割込発生エラー

なし。

3-1-5-3. PDF417 コンストラクタ

初期処理を行う。

public Pdf417(System.Drawing.Graphics g)

・引数

System.Drawing.Graphics g

PDF417 の描画を行う Graphics を指定します。

・戻り値

なし。

・割込発生エラー

なし。

public Pdf417 (String imgFilePath, ImageFormat imgFormat)

画像ファイルに PDF417 を出力する際に使用してください。

・引数

① String imgFilePath

バーコードを描画するデフォルトファイルパスを指定します。

② ImageFormat imgFormat

出力をする ImageFormat を指定します。プロパティの DPI を有効化する為には、png/jpeg フォーマットを指定してください。

・戻り値

なし。

・割込発生エラー

なし。

3-1-5-4. QR コード / DataMatrix / PDF417 メソッド (Draw 系同一)

(1) public void Draw (string code, float x, float y, float width, float height)

バーコードの描画を行います。指定幅を正確に合わせるためにいったん描画したバーコードを縮小して描画しなおします。そのため、多少、精度が劣化します。ドット単位で正確な精度を期待する場合は、指定した幅と正確に一緒にはなりませんが、DrawDirect / DrawDelicate メソッドを使用してください。

・ 引数

① string code

描画を行うバーコードのコードを文字列で指定します。

② float x

描画位置の始点(左上)の X 座標を指定します。

単位は、本メソッド呼び出し時の Grahics.PageUnit の値に依存します。

③ float y

描画位置の始点(左上)の Y 座標を指定します。

単位は、本メソッド呼び出し時の Grahics.PageUnit の値に依存します。

③ float width

バーコード全体の幅を指定します。

単位は、本メソッド呼び出し時の Grahics.PageUnit の値に依存します。

通常、height と、同じ値を指定します。

④ float height

バーコードの全体の高さを指定します。

単位は、本メソッド呼び出し時の Grahics.PageUnit の値に依存します。

通常、width と、同じ値を指定します。

・戻り値

なし

・例外の種類

指定されたコードの文字数が、バーコードに格納できる文字数をオーバーした。

(2) public void Draw(string code, float x, float y, float width, float height , string imgFilePath)

(1)の Draw メソッドのオーバーロードメソッドで QR コードを画像ファイルに出力する。

imgFilePath で指定したファイルへ QR コードの描画を行います。
描画するたびに出力ファイルを変更するときなどに利用してください。
他の機能について Draw メソッドと同様となります。

・ 引数

① string code
描画を行う QR コードのコードを文字列で指定します。

② float x
0 のみが指定可能です。

③ float y
0 のみが指定可能です。

④ float width
QR コード全体の幅を指定します。
単位は、本メソッド呼び出し時の ImgDrawUnit プロパティ値に依存します。
通常、height と、同じ値を指定します。

⑤ float height
QR コード全体の高さを指定します。
単位は、本メソッド呼び出し時の ImgDrawUnit プロパティ値に依存します。
通常、width と、同じ値を指定します。

⑥ string imgFilePath
出力を行うファイルパスを指定します。
イメージファイルの DPI は本メソッド呼び出し時の ImgDpi プロパティ値に依存します。

・戻り値
なし

・例外の種類
指定されたコードの文字数が、バーコードに格納できる文字数をオーバーした。

(3) public void DrawDirect (string code, float x, float y, float width, float height)

バーコードの描画を行います。
指定幅以内で最も広い幅でバーコードを直接描画します。
ドット単位での描画精度を実現します。

引数

① string code
描画を行うバーコードのコードを文字列で指定します。

② float x
描画位置の始点(左上)の X 座標を指定します。
単位は、本メソッド呼び出し時の Grahics.PageUnit の値に依存します。

③ float y
描画位置の始点(左上)の Y 座標を指定します。
単位は、本メソッド呼び出し時の Grahics.PageUnit の値に依存します。

④ float width
バーコードの全体の幅を指定します。
指定した幅以内で最も広い幅のバーコードを描画します。
単位は、本メソッド呼び出し時の Grahics.PageUnit の値に依存します。
通常、height と、同じ値を指定します。

⑤ float height
バーコードのバーの高さを指定します。
単位は、本メソッド呼び出し時の Grahics.PageUnit の値に依存します。
通常、width と、同じ値を指定します。

・戻り値
なし

・例外の種類
指定されたコードの文字数が、バーコードに格納できる文字数をオーバーした。

(4) public void DrawDirect(string code, float x, float y, float width, float height , string imgFilePath)

(3)の DrawDirect メソッドのオーバーロードメソッドで、バーコードを画像ファイルに出力する。

imgFilePath で指定したファイルへバーコードの描画を行います。
描画するたびに出力ファイルを変更するときなどに利用してください。
他の機能について DrawDirect メソッドと同様となります。

・ 引数

⑦ string code
描画を行うバーコードのコードを文字列で指定します。

⑧ float x
0 のみが指定可能です。

⑨ float y
0 のみが指定可能です。

⑩ float width
バーコードの全体の幅を指定します。
単位は、本メソッド呼び出し時の ImgDrawUnit プロパティ値に依存します。
通常、height と、同じ値を指定します。

⑪ float height
バーコードのバーの高さを指定します。
単位は、本メソッド呼び出し時の ImgDrawUnit プロパティ値に依存します。
通常、width と、同じ値を指定します。

⑫ string imgFilePath
出力を行うファイルパスを指定します。
イメージファイルの DPI は本メソッド呼び出し時の ImgDpi プロパティ値に依存します。

・戻り値
なし

・例外の種類
指定されたコードの文字数が、バーコードに格納できる文字数をオーバーした。

(3) public void DrawDelicate (string code, float x, float y, float minLineWidth)

バーコードの描画を行います。

Draw メソッドとの違いは、バーコード全体の幅を指定するのではなく、バーを描画する一番細かい単位を指定します。

Draw メソッドに比べて、**DrawDirect** メソッドと同様に精度の高いバーコードを描画することが可能です。ただし、バーコード全体の幅の調整が必要になります。

※この **DrawDelicate** メソッドを使用してバーの最小幅を指定した場合に精度が高くなる理由は、直接、Graphics オブジェクトに描画するためです。

Draw メソッドを使用して全体の幅を指定した場合、一旦、仮想空間に描画したバーコードを指定された全体の幅に対して当てはまるように Graphics オブジェクトに縮小描画しております。

・ 引数

① string code

描画を行うバーコードのコードを文字列で指定します。

② float x

描画位置の始点(左上)の X 座標を指定します。

単位は、本メソッド呼び出し時の Grahics.PageUnit の値に依存します。

③ float y

描画位置の始点(左上)の Y 座標を指定します。

単位は、本メソッド呼び出し時の Grahics.PageUnit の値に依存します。

④ float minLineWidth

バーコードを描画するの最小値を指定します。

単位は、本メソッド呼び出し時の Grahics.PageUnit の値に依存します。

・戻り値

なし

・例外の種類

指定されたコードの文字数が、バーコードに格納できる文字数をオーバーした。

(3) public void DrawDelicate(string code, float x, float y, float width, float height, string imgFilePath)

imgFilePath で指定したファイルへバーコードの描画を行います。
描画するたびに出力ファイルを変更するときなどに利用してください。
他の機能について DrawDelicate メソッドと同様となります。

・ 引数

① string code
描画を行うバーコードのコードを文字列で指定します。

② float x
0 のみが指定可能です。

③ float y
0 のみが指定可能です。

④ float width
バーコードの全体の幅を指定します。
単位は、本メソッド呼び出し時の ImgDrawUnit プロパティ値に依存します。
通常、height と、同じ値を指定します。

⑤ float height
バーコードのバーの高さを指定します。
単位は、本メソッド呼び出し時の ImgDrawUnit プロパティ値に依存します。
通常、width と、同じ値を指定します。

⑥ string imgFilePath
出力を行うファイルパスを指定します。
イメージファイルの DPI は本メソッド呼び出し時の ImgDpi プロパティ値に依存します。

・戻り値
なし

・例外の種類
指定されたコードの文字数が、バーコードに格納できる文字数をオーバーした。

(3) public void WriteSVG
(string code, float x, float y, float width, float height, string filePath)
SVG ファイルへのバーコードの出力を行います。

・ 引数

① string code
描画を行うバーコードのコードを文字列で指定します。

② float x
描画位置の始点(左上)の X 座標を指定します。
単位は、本メソッド呼び出し時の Grahics.PageUnit の値に依存します。

③ float y
描画位置の始点(左上)の Y 座標を指定します。
単位は、本メソッド呼び出し時の Grahics.PageUnit の値に依存します。

④ float width
バーコードの全体の幅を指定します。
単位は、本メソッド呼び出し時の Grahics.PageUnit の値に依存します。
ただし、SVG ファイルは、ブラウザ表示用のため、画面の pixel とバーコードの線が一致しないといけないため、指定された幅以下のサイズに最適化されます。
通常、height と、同じ値を指定します。

⑤ float height
バーコードのバーの高さを指定します。
単位は、本メソッド呼び出し時の Grahics.PageUnit の値に依存します。
ただし、SVG ファイルは、ブラウザ表示用のため、画面の pixel とバーコードの線が一致しないといけないため、指定された幅以下のサイズに最適化されます。
通常、width と、同じ値を指定します。

⑥ string filePath
SVG ファイルのファイル名をフルパスで指定してください。

・戻り値
なし

・例外の種類
指定されたコードの文字数が、バーコードに格納できる文字数をオーバーした。

3-1-5-5. QR コードプロパティ

(1) public int Version
バージョン 1～40 を指定・取得。

(2) public string ErrorCorrect
エラー訂正レベル：”L”/”M”/”Q”/”H” のいずれかを指定・取得。

(3) public string EncodeMode
エンコードモードを指定・取得します。
“N”:数字モード
“A”:英数字モード
その他:8bit byte モード
エンコードモードに漢字モードがありませんが、漢字の入力も「その他:8bit byte モード」を指定してください。”N”/”A”以外の文字であればなんでも OK です。

(4) public GraphicsUnit ImgDrawUnit
イメージファイル出力時の座標単位を指定する。
既定値は、GraphicsUnit.Pixel。

(5) public float ImgDpi
イメージファイル出力時の DPI を指定する。
既定値は、６００。

3-1-5-6. DataMatrix プロパティ

(1) public DxCodeSize CodeSize
シンボルコードサイズ　　既定値：DxCodeSize.DxSzAuto(自動)

(public enum) DxCodeSize.

DxSzRectAuto,	DxSz24x24,	DxSz88x88,
DxSzAuto,	DxSz26x26,	DxSz96x96,
DxSzShapeAuto,	DxSz32x32,	DxSz104x104,
	DxSz36x36,	DxSz120x120,
DxSz10x10,	DxSz40x40,	DxSz132x132,
DxSz12x12,	DxSz44x44,	DxSz144x144,
DxSz14x14,	DxSz48x48,	
DxSz16x16,	DxSz52x52,	DxSz12x36,
DxSz18x18,	DxSz64x64,	DxSz16x36,
DxSz20x20,	DxSz72x72,	DxSz16x48
DxSz22x22,	DxSz80x80,	

(2) public string StringEncoding
全角エンコーディング（”utf-8”, “shift-jis”, etc..）　既定値：utf-8

3-1-5-6. PDF417 プロパティ

(1) public SIZE_KIND SizeKind
サイズ種別　…データ列数・行数決定方法
自動　　Pdf417.SIZE_KIND.AUTO　(既定値)
データ列数指定　　Pdf417.SIZE_KIND.COLUMNS
データ行数指定　　Pdf417.SIZE_KIND.ROWS
データ列数・行数指定　　Pdf417.SIZE_KIND.COLUMNS_AND_ROWS

(2) public int CodeRows
行数　…出力データ行数指定
サイズ種別が、
出力行数指定の場合＝(自動サイズでない・列数指定でない場合)有効
3～90　　既定値：5

(3) public int CodeColumns
列数　…出力データカラム数指定
サイズ種別が、
出力列数指定の場合＝(自動サイズでない・行数指定でない場合)有効
1～30　　既定値：5

(4) public int ErrorLevel
エラー訂正レベル　0～8　　既定値：0

(5) public bool UseAutoErrorLevel
エラー訂正レベル自動決定
自動でエラー訂正レベルを決定(する・しない)　　既定値：true(する)

(6) public float AspectRatio
縦横アクセプト比　既定値：0.5
シンボルの縦横比、シンボル アスペクト レシオ（比）

(7) public string StringEncoding
全角エンコーディング（”utf-8”, “shift-jis”, etc..)　　既定値：utf-8

(8) public GraphicsUnit ImgDrawUnit
イメージファイル出力時の座標単位を指定する。
既定値は、GraphicsUnit.Pixel。

(9) public float ImgDpi
イメージファイル出力時の DPI を指定する。
既定値は、６００。

3-2. C#での使用例(Code39 の例)

ここでは、Code39 のバーコード印刷を例にして簡単な使用方法を説明します。

まず、フォーム上に Button コントロールと PrintDocument コントロールを貼り付けてください。
次に、それぞれの各イベントに以下のようにコードを入れてください。

```
private void button1_Click(object sender, System.EventArgs e)
{
        printDocument1.Print();
}

private void printDocument1_PrintPage(object sender,
                                      System.Drawing.Printing.PrintPageEventArgs e)
{
        Pao.BarCode.Code39 cd39 = new Pao.BarCode.Code39(e.Graphics);
        cd39.Draw("12345", 50, 50, 200, 50);
}
```

ボタンをクリックすると、これだけで Code39 のバーコードは、出力されるでしょう。
しかし、指定した x=50/y=50/width=200/height=50 の単位はいったい何なんでしょう？
GDI+で使用できる単位系は複数あり、Graphics オブジェクトのプロパティである、PageUnit の値のいずれかで指定することが可能です。
そこで以下のように予めミリメートルを長さの単位に指定しますと、以下の例では x=1cm/y=1cm の位置から、幅=5cm/高さ=1.5cm のバーコードを出力することになります。

```
        e.Graphics.PageUnit = GraphicsUnit.Millimeter;
        Pao.BarCode.Code39 cd39 = new Pao.BarCode.Code39(e.Graphics);
        cd39.Draw("12345", 10, 10, 50, 15);
```

また、以下の例のように各種プロパティを指定して、添字を均等割付にし、フォントを変更し 270 度回転して印刷すること等も試してみてください。

```
        e.Graphics.PageUnit = GraphicsUnit.Millimeter;
        Pao.BarCode.Code39 cd39 = new Pao.BarCode.Code39(e.Graphics);
        cd39.TextKintou = true;
        cd39.TextFont = new Font("Times New Roman",12,FontStyle.Bold);
        cd39.RotateAngle = Pao.BarCode.RotateAngle.Angle270;
        cd39.Draw("12345", 100, 100, 50, 15);
```

詳しくは、サンプルプログラムをご用意させていただきましたので、是非、じっくり弄繰り回してください。

3-3. サンプルプログラム

C#.NET / VB.NET で作成した Barcode.net を利用したサンプルプログラムを８本ご用意いたしました。
http://www.pao.ac/barcode.net/#download
より、zip ファイルをダウンロードして解凍後、インストーラでインストールしてください。Pao.BarCode.dll と一緒に７つのフォルダが作成されると思います。それぞれのフォルダ内に更に、C#／VB という２つのフォルダがあり、それぞれに、C#.NET／VB.NET を利用したサンプルプログラムが入っております。是非お試しください。

(1) バーコードの印刷・プレビューサンプル
BarApp(通常のサンプル)¥C# 又は BarApp(通常のサンプル)¥VB フォルダ内にある BarApp.sln ファイルを起動する事により、このサンプルプログラムを開くことができます。
サンプルプログラムは、Barcode.net の機能をフルに利用した印刷・プレビュー処理を実現しています。サンプルプログラムの割には、市販のバーコード作成ソフトにも見劣りしない多くの機能を実装しております。このままでも色々な用途があると思いますが、是非、弄繰り回して改造して遊んでください。

(2) 画像ファイルに保存するサンプル
BarApp2(クリップボード・画像処理)¥C# 又は BarApp2(クリップボード・画像処理)¥VB フォルダ内にある BarApp2.sln ファイルを起動する事により、このサンプルプログラムを開くことができます。
PNGやJPEG等の画像ファイルにバーコードを出力するサンプルプログラムです。また、バーコードをクリップボードに貼り付ける処理も入っております。

(3) QR コードの描画・印刷・プレビュー・画像ファイル出力サンプル
QrApp(QR コードサンプル)¥C# 又は QrApp(QR コードサンプル)¥VB フォルダ内にある QRApp.sln ファイルを起動する事により、このサンプルプログラムを開くことができます。
このサンプルプログラムは、QR コードの印刷・プレビュー、画面出力、PNG や JPEG 等の画像ファイル出力を実現しています。

(4) SVG / SVGZ 出力・ブラウザ表示サンプル
BarSVG(SVG･SVGZ サンプル)¥C# 又は BarSVG(SVG･SVGZ サンプル)¥VB フォルダ内にある SVG.sln ファイルを起動する事により、このサンプルプログラムを開くことができます。
QR コード以外のバーコードの SVG /SVGZ 出力とブラウザ表示をを行うサンプルです。

(5) QR コード、SVG / SVGZ 出力・ブラウザ表示サンプル
QrSVG(SVG・SVGZ サンプル)¥C# 又は QrSVG(SVG・SVGZ サンプル)¥VB フォルダ内にある QrSVG.sln ファイルを起動する事により、このサンプルプログラムを開くことができます。
QR コードの SVG /SVGZ 出力とブラウザ表示をを行うサンプルです。

(6) ASP.NET を使って QR コードブラウザ出力するサンプル
QRWeb(ASP.NET サンプル)フォルダ内にある QRWeb.sln ファイルを起動する事により、このサンプルプログラムを開くことができます。
ただし、このプロジェクトは、ASP.NET アプリケーションですので、C:¥Inetpub¥wwwroot¥ 等の IIS の WEB サイトフォルダにコピーして、ASP.NET で動く設定を行ってから、ソリューションを開いてください。
今のところ、C#版のみです。VB.NET 版は、作成していません。
QR コードを ASP.NET を使用してブラウザ表示をを行うサンプルです。

(7) コンビニバーコードの印刷・プレビュー、画像ファイル出力サンプル
コンビニ EAN128 サンプル¥C# 又は コンビニ EAN128 サンプル¥VB フォルダ内にある BarApp.sln ファイルを起動する事により、このサンプルプログラムを開くことができます。
このサンプルプログラムは、コンビニバーコードの印刷・プレビュー、画面出力、PNG や JPEG 等の画像ファイル出力を実現しています。

4. 使用条件等

4-1. お試し版と製品版

Barcode.net は、いつでもどこでも誰でも試用できるように、
http://www.pao.ac/barcode.net/
にお試し版として最新バージョンを常にご用意させていただいております。

お試し版の制限は、バーコードに控えめに「SAMPLE」という文字が入ります。

QR コードのお試し版の制限は、指定したコードの先頭に半角の「9」という文字が入ります。

ライセンス登録することにより、試用制限は解除されます。

ライセンス登録方法は、
インストール先(デフォルト:C:¥Program File (x86)¥Pao@Office¥Barcode.net)の License.bat を起動するか、スタートメニューより「ご購入・ライセンス登録」をクリックしてください。

バージョンアップの際は、WEB サイトにてお知らせいたします。
http://www.pao.ac/barcode.net/
最新版をダウンロード後インストールしていただければ、いつでも無償でバージョンアップを行えます。

4-2. 使用許諾

Barcode.net の使用について、Barcode.net の使用者(以下「利用者様」と称します)と有限会社パオ・アット・オフィス(以下「弊社」と称します)は、以下の各項目についての内容に同意するものとします。

1. Barcode.net の使用に関する使用許諾書
   この使用許諾書は、利用者様がお使いのパソコンにおいて、Barcode.net を使用する場合に同意しなければならない契約書です。
2. 使用許諾書の同意
   利用者様が Barcode.net を使用する時点で、本使用許諾書に同意されたものとします。同意されない場合は、Barcode.net を使用する事はできません。
3. ライセンス(使用権)の購入
   利用者様が Barcode.net の製品版を使用して開発を行う場合には、1 台の開発用コンピュータで Barcode.net を使用するにあたり、1 ライセンスを購入する必要があります。
   お客様環境等、開発コンピュータでないマシンで Barcode.net を使用する場合ライセンスは必要ありません。ランタイムライセンスフリーでございます。
4. 著作権
   Barcode.net の著作権は、いかなる場合においても弊社に帰属いたします。
5. 免責
   Barcode.net の使用によって、直接的、あるいは、間接的に生じた、いかなる損害に対しても、弊社は補償賠償の責任を負わないものとします。
6. 禁止事項
   Barcode.net 及びその複製物を第三者に譲渡・貸与する事は出来ません。
   Barcode.net を開発ツールとして再販/再配布することを禁止します。なお、モジュールとして組み込みを行い再販/再配布する場合は、開発ツールとしての再販/再配布には含まれませんので、OK です。
7. 保証の範囲
   弊社は Barcode.net の仕様を予告無しに変更することがあります。その場合の利用者様に対する情報提供は、弊社 WEB サイトにて行う事とします。
8. 適用期間
   本使用許諾条件は利用者様が Barcode.net を使用した日より有効です。利用者様が本使用許諾条件のいずれかの条項に違反した場合、又は、本許諾条件に同意出来ない場合は、利用者様は Barcode.net を一切使用出来ないものとします。

4-3. 代金支払い方法(ライセンス登録の方法)

Barcode.net の製品版をご利用頂ける場合は、ライセンスを購入して頂く必要があります。ライセンス形態及び代金支払方法は以下のとおりです。

- 必要なライセンス数の数え方
  - Barcode.net をインストールするパソコンの台数

- 1 ライセンス当たりの価格
  - 20,000 円(税込:22,000 円)
  - ソースコード付：92,000 円(税込:99,360 円)
    - ✧ バグフィックス等のバージョンアップは原則として無償とさせて頂きます。
    - ✧ 大幅な機能追加等によるバージョンアップの場合には別ライセンスとさせていただく場合がございます。
    - ✧ 本価格は Barcode.net の使用権に対するものです。カスタマイズや保守等の費用は一切含まれておりません。
- お支払方法

(22,000 円 or 99,360 円×ライセンス数)を下記口座へ銀行振込、または、郵便振替による送金をして下さい。

銀行名	支店名 (コード)	口座番号	名義
三菱ＵＦＪ銀行	新宿支店 (341)	普通 3831891	ユ）パオアツトオフイス
PayPay 銀行	すずめ支店 (002)	普通 6461359	ユ）パオアツトオフイス

郵便口座番号	名義
００１５０－０－ ５７６８４５	有限会社 パオ・アット・オフィス

  - ✧ 振込手数料は利用者様負担でお願い致します。
- お支払いの通知と製品の送付
  - 振り込みが完了した時点で、必ず弊社 WEB サイトの「Barcode.net 入金連絡フォーム」から入金のご連絡をお願いいたします。
    http://www.pao.ac/barcode.net/buy.html#form
  - 弊社では上記連絡を受けて入金確認を行い、Barcode.net のライセンスキーを利用者様へ電子メールにてお送りさせて頂きます。
    - ✧ 利用者様へは電子メール以外での製品の提供は原則として行いません。
    - ✧ 製品の再送付は原則として行いません。製品のファイルは消去してしまわないように大切にお取り扱いください。
  - お振り込み頂いても入金の連絡がない場合、こちらか振り込み人様の情報が分からないため、製品の送付が行えません。必ず入金連絡を行って頂くようお願いいたします。
  - 入金の連絡(ご購入申請)は、Barcode.net が、インストールされているマシン上のスタートメニューより「ご購入・ライセンス申請」又は、インストールフォルダの License.bat を起動して行うこともできます。

- 見積書/納品書/請求書/領収証の発行、納品後のお支払いについて
  見積書/納品書/請求書/領収証の発行は可能でございます。本製品納品後のお支払いも可能でございます。
  http://www.pao.ac/barcode.net/buy.html
  上記サイトでの手続きにより、弊社からの見積書/納品書/請求書/領収証の発行、及び、納品後のお客様からのお支払いを行えるようになっております。