Serenade 3
プログラマーズマニュアル

目次

1. 概要

Serenade（セレナーデ）は Windows 上でベクトル描画を行なうための DLL（ダイナミック・リンク・ライブラリー）である．Serenade をリンクしたホストアプリケーションは出力デバイスの解像度に依存しない描画を行なうことができる．したがって実際の解像度はユーザーが使用しているディスプレイやプリンターによって決まる．

Version 3 では複数のスレッドにおいてスレッド間の衝突が起こらないようにするために，引用方法を変更した関数群を用意した．これらの関数群では文字列の符号化を従来の ANSI コードから UNICODE に変更したので注意が必要である．マルチスレッドアプリケーションについては「マルチスレッド化の指針（4 ページ）」で説明する．なお，従来の関数はそのまま残してあるので Version 2 をそのまま Version 3 に置き換えてもこれまでのホストアプリケーションは問題なく動作する．

ルーチン名	動作内容	①	②	備考	参照頁
plsize	描画ページの開始を指示する	×			19
plexit	描画ページの終了を指示する	×			14
plot	ペンを移動する	×			16
newpen	ペンの種類を換える	×			10
plfont	フォントを換える	×			15
symbol	文字列を表示する（左詰め）	×			25
mdsymb	文字列を表示する（中揃え）	×			9
rtsymb	文字列を表示する（右詰め）	×			24
number	数値を表示する	×			11
genten	原点を変更する	×			8
factor	拡大率を指定する	×		非推奨	7
circle	円を描く	×			5
dash	破線を描く	×			6
pladdp	パスに座標を追加する	×			12
plclpt	パスを閉じてリージョンを作成する	×			13
plrgbp	リージョン内に色を塗る	×			18
plhalt	ユーザーの操作を待つ				16
pl_SetRGB	番号０のペンの色を指定する	×			21
pl_SetWidth	番号０のペンの線幅を指定する	×			22
plmsg	メッセージを表示する				16
plread	文字列を入力する				17
plsizeEx	描画ページの開始を指示する		3		19
plexitEx	描画ページの終了を指示する		3		14
plotEx	ペンを移動する		3		17
newpenEx	ペンの種類を換える		3		10
symbolEx	文字列を表示する（左詰め）		3		26
mdsymbEx	文字列を表示する（中揃え）		3		9
rtsymbEx	文字列を表示する（右詰め）		3		24
numberEx	数値を表示する		3		12
gentenEx	原点を変更する		3		8
factorEx	拡大率を変更する		3	非推奨	7
circleEx	円を描く		3		5
dashEx	破線を描く		3		6
plfontEx	フォントを換える		3		15
pladdpEx	パスに座標を追加する		3		13
plclptEx	パスを閉じてリージョンを作成する		3		14
plrgbpEx	リージョン内に色を塗る		3		18
pl_SetRGBEx	番号０のペンの色を指定する		3		22
pl_SetWidthEx	番号０のペンの線幅を指定する		3		22
pl_TextWidthEx	文字列の表示上の幅を取得する		3		23
pl_PageExists	ページが表示されているかを取得する	×	3		20
pl_PageExistsEx	指定したページが表示されているかを取得する		3		21
pl_Initialize	Serenade の初期設定を行なう		3		20
pl_Terminate	Serenade ウィンドウを閉じる		3		23

①：スレッド分割が行なえないルーチンには×印

②：Version 3 で追加されたルーチンには３印

参考のためバージョンごとの主な相違を次の表に示す.

	Version 1	Version 2	Version 3
対応 OS	Windows 98/2000	Windows 2000/XP/Vista/7	Windows Vista/7
描画サイズ	面積が A4 相当以下	面積が A0 相当以下	←
最大ページ数	1000 ページ（ただし 20 ページ程度でパフォーマンス低下）	1000 ページ	制限なし
拡大縮小	50%，100%，200%	制限なし	←
ファイル出力	PostScript, BMP（180dpi 固定）	PostScript，BMP，JPEG，AVI（BMP/JPEG/AVI は解像度の指定可）	←（BMP/JPEG の全ページ一括出力も可）
フォント	MS P 明朝	複数フォンと使用可能（フォントの割り当てはユーザーが指定可能）	←
ペンの種類	８色（割り当ては内部固定，太さは変更不可）	色と太さの組み合わせで 10 種類（割り当てはユーザーが変更可能） プログラムでのみ色と太さが指定できるものが一種類	←
ペイント機能	なし	あり	←
マルチスレッド対応	なし	←	対応ルーチンあり
文字符号	ANSI	←	新規追加ルーチンは UNICODE
引数	アドレス渡し	←	新規追加ルーチンは値渡し
作業ファイル	ページごとのファイル	←	なし

●システム要求

OS	Windows Vista / 7（Windows 7 64bit 上では 32bit モードで動作）
CPU	Intel Pentium の上位互換 CPU
メモリー	512MB 以上（表示ページの複雑さとページ数に応じてより多くのメモリーが必要）
ハードディスク	10MB 以上（表示ページの複雑さとページ数に応じてより多くのハードディスクが必要）

2. 必要ファイル

(1) 作成時に使用するファイル

Serenade.lib　　インポート LIB ファイル（Borland 形式※）

Serenade_import.h　C++ 用関数宣言ファイル

※　Version 2 までは Microsoft 形式のファイルを提供していたが Version 3 より Borland 形式のファイル提供とした．Microsoft Visual Studio でプログラムを開発する場合は DEF ファイルを作成して，LIB.exe で Microsoft 形式のインポート LIB ファイルを作成してやる必要がある．

(2) 実行時に使用するファイル

Serenade.dll　　Serenade ダイナミックリンクライブラリー

3. コーディング上の注意点

3.1 分類

描画は仮想ペンによるベクトル描画である．したがって解像度は出力デバイスによって決まる．ルーチンは大きく分けて次の３グループに分類できる．

(1) 引用順に一ページずつ描画を行なうルーチン群

plsize，plexit，plot，newpen，plfont，symbol，mdsymb，rtsymb，number，genten，factor，circle，dash，pladdp，plclpt，plrgbp，pl_SetRGB，pl_SetWidth，pl_PageExists

(2) 任意のページを指定して描画を行なう（ページ内は引用順描画）ルーチン群

plsizeEx，plexitEx，plotEx，newpenEx，plfontEx，symbolEx，mdsymbEx，rtsymbEx，numberEx，gentenEx，factorEx，circleEx，dashEx，pladdpEx，plclptEx，plrgbpEx，pl_SetRGBEx，pl_SetWidthEx，pl_PageExistsEx，pl_TextWidthEx

(3) 補助的なルーチンで描画とは関係のないルーチン群

plhalt，plmsg，plread

理論的には (1) と (2) の混在は可能であるが，混乱しやすいので (1) か (2) のどちらか一方の使用を推奨する．(3) は適宜使用してよい．蛇足ながら述べれば，(1) と (3) はもともと順次実行型（ユーザーイベント駆動型でないという意味）のプログラムでの使用を念頭に用意したルーチン群なので，Windows アプリケーションにおいては使用する必要がないかあるいは使用すべきでないと考える[1]．つまり (1) と (3) は旧アプリケーションの動作のためだけに残したルーチン群であり，これから Serenade を使ったアプリケーションを作成する場合は (2) だけを使用することを強く推奨する．したがって関数リファレンスは (1) と (2) の混用をしないことを前提として記述している[2]．

ページの指定には「ページ識別子」を使用する．ページ識別子は plsizeEx を引用することで得られる．以後画面上にこのページが表示されている間は識別子として有効である．この識別子が無効になるのは Serenade 画面が閉じられた時である．

3.2 文字コード

引数に文字列を指定する場合，Version2 以前から存在するルーチン（前節の (1)，(3) のルーチン群）では符号化は ANSI コードであるが, Version3 で新設されたルーチン（前節の (2) のルーチン群）では符号化は UTF8 コードである．これに伴って文字列長の指定も従来のバイト数ではなく文字数になっているので注意が必要．

3.3 引数の引き渡し方法

Version2 以前のルーチンは Fortran からの引用も考慮して「アドレス渡し」が基本となっている．Version3 で新設されたルーチンは「値渡し」を基本としている．これは今後 Fortran で Serenade を引用するアプリケーションの新規開発の需要がほとんどないと考えたため，C++ からの引用の利便性を考えて決めた方針である．この方針によって文字列引数の直後に必要だった文字バイト数の引数は不要とした[3]．

1　ただし後述（引数の引き渡し方法参照）するように引数をアドレス渡しにせざるを得ない場合は (1) を使う必要がある．

2　たとえば plfont は実際には symbol だけでなく symbolEx にも影響を及ぼすが，そのことは記述していない．

3　MS FORTRAN，DEC FORTRAN，Compaq Fortran では文字型引数の直後にバイト数が隠れた引数として自動挿入されていた．

3.4 座標系

水平軸は左から右に，垂直軸は下から上に向かう右手系座標である．座標値はユーザーが指定する単位の値である．

3.5 マルチスレッド化の指針

Intel Core 2 Duo 以降の CPU はマルチコア化が進んでおり，CPU 処理を並列化することで処理時間が短縮できる．マルチコア CPU はいわば CPU が複数存在するようなもので，スーパーコンピューターのベクトルプロセッサとは違い各々のコアは概ねスレッド単位で動作する．したがって使用する CPU のコア数程度のスレッド分割を行なうと最も処理時間が短縮できることになる．ところで Serenade による描画は各ルーチンの引用順が重要な意味を持っているので『ルーチン単位』のマルチスレッド化は不可能である．ではどのようなマルチスレッド化が可能かというと，『ページ単位』のマルチスレッド化である．つまりページ単位の描画処理をスレッド単位にする[1]．そのために Version 3 からページ識別子によるページ指定描画ルーチンが追加されたのである．注意が必要なのはスレッド数が CPU コア数をあまり超えないようにすることである．たとえば 100 ページを超えるような描画を行なう場合，一度にページごとのスレッドを生成すると同時に 100 スレッド以上が動作することになる．これはオーバーヘッドが大きくなって逆効果（逆に時間がかかってしまう）である．キューなどを使って常に CPU コア数程度のスレッドが動作するようなアプリケーションを作成するべきである．

1　インタラクティブなルーチン（plmsg，plread）以外はプログラムの流れを変えるものはないので，ページ間に依存関係はないはずでありスレッド分割が可能である．ただし CPU 以外の資源（ファイルなど）の取り合いは起こりうるのでそのオーバーヘッドを最小限にするような設計が必要である．

4. 関数リファレンス

全ルーチンの詳細をアルファベット順に記述する.

circle

```
void circle(double *x, double *y, double *tho, double *thf, double *ro, double *rf, double *dl)
```

【互換性】Version 2，Version 3

【引数】

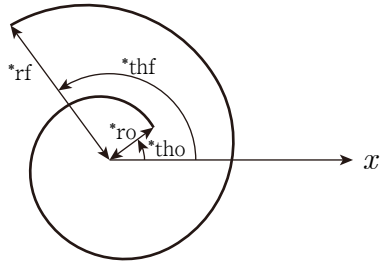
double *x	円弧始点座標
double *y	円弧始点座標
double *tho	円弧開始角度（x 軸から反時計回りの角度．単位は度）
double *thf	円弧終了角度（x 軸から反時計回りの角度．単位は度）
double *ro	円弧開始の半径
double *rf	円弧終了の半径
double *dl	未使用

【機能】

・座標（*x，*y）を始点とする円弧を描く.

・開始角度は *tho，終了角度は *thf で指定する．したがって円弧の中心座標は（*x-cos(*tho)，*y-sin(*tho)）となる.

・開始半径と終了半径を異なる値にすると，角度に比例して半径が変化する「アルキメデスらせん」となる.



circleEx

```
void circleEx
(RPageDescripter *pd, double x, double y, double tho, double thf, double ro, double rf, double dl)
```

【互換性】Version 3

【引数】

RPageDescripter *pd	ページ識別子
double x	円弧始点座標
double y	円弧始点座標
double tho	円弧開始角度（x 軸から反時計回りの角度．単位は度）
double thf	円弧終了角度（x 軸から反時計回りの角度．単位は度）
double ro	円弧開始の半径

double rf	円弧終了の半径
double dl	未使用

【機能】

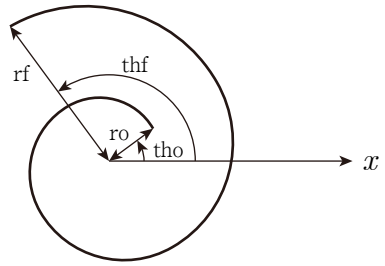
・ページ識別子で指定されたページに円弧を描く.

・座標（x，y）を始点とする円弧を描く.

・開始角度は tho，終了角度は thf で指定する．したがって円弧の中心座標は（x-cos(tho)，y-sin(tho)）となる.

・開始半径と終了半径を異なる値にすると，角度に比例して半径が変化する「アルキメデスらせん」となる.

・ページを指定することができる他は circle と同機能.



【制限】

・ページ識別子は引用前に plsizeEx で取得されていなければならない.

dash

```
void dash(double *xs, double *ys, double *xe, double *ye, double *dl)
```

【互換性】Version 2，Version 3

【引数】

double *xs	始点座標
double *ys	始点座標
double *xe	終点座標
double *ye	終点座標
double *dl	破線間隔

【機能】

・座標（*xs，*ys）から座標（*xe，*ye）までの破線を描く.

・線のある部分の長さと線のない部分の長さはどちらも *dl となる.

dashEx

```
void dashEx(RPageDescripter *pd, double xs, double ys, double xe, double ye, double dl)
```

【互換性】Version 3

【引数】

RPageDescripter *pd	ページ識別子
double xs	始点座標
double ys	始点座標
double xe	終点座標
double ye	終点座標
double dl	破線間隔

【機能】

・ページ識別子で指定されたページに破線を描く.
・座標（xs，ys）から座標（xe，ye）までの破線を描く.
・線のある部分の長さと線のない部分の長さはどちらも dl となる.
・ページを指定することができる他は dash と同機能.

factor

```
void factor(double *r)
```

【互換性】Version 2，Version 3

【引数】

double *r	スケールファクター（＞0）

【機能】

・この後すべての関数の引用で指定される座標値，および文字の高さに *r をかけて描画する.

【制限】

・指定するスケールファクターは正の値でなければならない.

【注意】

スケールファクターの乗じ方に様々な考え方があり，仕様上の誤解などからホストプログラムが要求通りの描画にならないことが予想される．また，スケールファクターの操作は本来ホストアプリケーション側で行なうべきものであろう．したがって本ルーチンはできるかぎり使用しないことを推奨する.

factorEx

```
void factorEx(RPageDescripter *pd, double r)
```

【互換性】Version 3

【引数】

RPageDescripter *pd	ページ識別子
double r	スケールファクター（＞0）

【機能】
・この後すべての関数の引用で指定される座標値，および文字の高さに r をかけて描画する．

【制限】
・ページ識別子で指定されたページのスケールファクターを設定する．
・指定するスケールファクターは正の値でなければならない．
・ページを指定することができる他は factor と同機能．

【注意】
スケールファクターの乗じ方に様々な考え方があり，仕様上の誤解などからホストプログラムが要求通りの描画にならないことが予想される．また，スケールファクターの操作は本来ホストアプリケーション側で行なうべきものであろう．したがって本ルーチンはできるかぎり使用しないことを推奨する．

genten

```
void genten(double *x, double *y)
```

【互換性】Version 2，Version 3

【引数】
double *x	新しい原点座標
double *y	新しい原点座標

【機能】
・現在の座標系での座標（*x，*y）を新しい原点とする．

gentenEx

```
void gentenEx(RPageDescripter *pd, double x, double y)
```

【互換性】Version 3

【引数】
RPageDescripter *pd	ページ識別子
double x	新しい原点座標
double y	新しい原点座標

【機能】
・ページ識別子で指定されたページの原点を設定する．
・現在の座標系での座標（*x，*y）を新しい原点とする．
・ページを指定することができる他は genten と同機能．

mdsymb

```
void mdsymb(double *x, double *y, double *h, char *s, int ns1, double *deg, int *ns2)
```

【互換性】Version 2，Version 3

【引数】

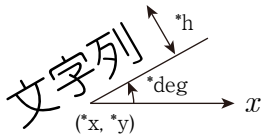
double *x	文字列描画原点（中央下）座標
double *y	文字列描画原点（中央下）座標
double *h	文字列高さ
char *s	文字列（ANSI）
int ns1	文字列長さ（バイト数）
double *deg	角度（x 軸から反時計回りの角度．単位は度で -180 ～ 180）
int *ns2	文字列長さ（通常は ns1 と同じ値）

【機能】

・座標（*x，*y）を文字列中央下として左右に均等に振り分けられた文字列 s（ANSI）を描画する．

・文字高さは *h，文字列角度は *deg で指定する．

・文字列の符号化は ANSI コードで行ない，文字列長さはバイト数で指定する．



【制限】

・文字数 ns1 は実際に渡される文字列 s のバイト数でなければならない．一方，文字数 *ns2 は実際に描画する文字数でよいので *ns2 ≦ ns1 の範囲で指定することができる．

mdsymbEx

```
void mdsymbEx(RPageDescripter *pd, double x, double y, double h, wchar_t *s, double deg, int ns)
```

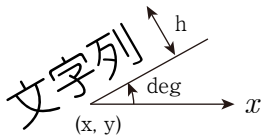
【互換性】Version 3

【引数】

RPageDescripter *pd	ページ識別子
double x	文字列描画原点（中央下）座標
double y	文字列描画原点（中央下）座標
double h	文字列高さ
wchar_t *s	文字列（UTF8）
double deg	角度（x 軸から反時計回りの角度．単位は度で -180 ～ 180）
int ns	文字列長さ（文字数）

【機能】

・ページ識別子で指定されたページに文字列を中央揃えで描画する.
・座標（x, y）を文字列中央下として左右に均等に振り分けられた文字列 s（UTF8）を描画する.
・文字高さは h, 文字列角度は deg で指定する.
・文字列の符号化は UTF8 コードで行ない, 文字列長さは実際の文字数（半角文字, 全角文字の区別なし）で指定する.
・ページを指定することができる他は mdsymb と同機能.



【制限】
・文字数 ns は実際に渡される文字列 s の文字数以下でなければならない.

newpen

```
void newpen(int *n)
```

【互換性】Version 2, Version 3

【引数】
int *n　　ペン番号（0 ～ 10）

【機能】
・指定したペン番号にペンを変更する.
・この後に引用される線描画ルーチンすべてに有効となる.
・ペンには色と線幅の属性がある.
・ペン番号 1 ～ 10 の属性はアプリケーション実行時にユーザーが変更可能.
・ペン番号 0 の属性はアプリケーション内で指定しなければならない. ペンの色は pl_SetRGB で指定する. ペンの線幅は pl_SetWidth で指定する [1].

【制限】
ペン番号は 0 ～ 10 でなければならない.

newpenEx

```
void newpenEx(RPageDescripter *pd, int n)
```

【互換性】Version 3

【引数】
RPageDescripter *pd ページ識別子
int n　　ペン番号（0 ～ 10）

1　pl_SetRGBEx や pl_SetWidthEx でも指定可能であるができるかぎり避けるべきである.

【機能】

・ページ識別子で指定されたページのペンを変更する.

・指定したペン番号にペンを変更する.

・この後に引用される線描画ルーチンすべてに有効となる.

・ペンには色と線幅の属性がある.

・ペン番号 1 ～ 10 の属性はアプリケーション実行時にユーザーが変更可能.

・ペン番号 0 の属性はアプリケーション内で指定しなければならない. ペンの色は pl_SetRGBEx で指定する. ペンの線幅は pl_SetWidthEx で指定する [1].

・ページを指定することができる他は newpen と同機能.

【制限】

ペン番号は 0 ～ 10 でなければならない.

number

```
void number(double *x, double *y, double *h, double *v, double *deg, int *n)
```

【互換性】Version 2, Version 3

【引数】

double *x	数字列描画原点座標
double *y	数字列描画原点座標
double *h	数字高さ
double *v	数値
double *deg	角度（x 軸から反時計回りの角度. 単位は度で -180 ～ 180）
int *n	表示形式

【機能】

・座標（*x, *y）に数値 *v を文字表示する.

・座標（*x, *y）は数字列の左下を指定する.

・文字高さは *h, 文字角度は *deg で指定する.

・数値から文字列への変換は *n によって次のように行なわれる.

〔*n ＜ 0 の場合〕

・数値 *v の整数部だけを左詰めで文字列に変換する.

・数値 *v の整数部は 10 の（*n の絶対値－ 1 ）乗の桁まで丸める.

〔0 ≦ *n ≦ 9 の場合〕

・小数点以下第（*n）位までを左詰めで文字列に変換する.

・*n ＝ 0 の場合は小数点までが文字列に加えられる.

〔*n ＞ 9 の場合〕

・数値を右詰めで文字列に変換する.

・小数点以下の桁数は *n の一の位の数値で指定する.

・全体の文字数は *n の十の位以上の数値で指定する.

1 pl_SetRGB や pl_SetWidth でも指定可能であるができるかぎり避けるべきである.

numberEx

```
void numberEx(RPageDescripter *pd, double x, double y, double h, double v, double deg, int n)
```

【互換性】Version 3

【引数】

RPageDescripter *pd	ページ識別子
double x	数字列描画原点座標
double y	数字列描画原点座標
double h	数字高さ
double v	数値
double deg	角度（x 軸から反時計回りの角度．単位は度で -180 ～ 180）
int n	表示形式

【機能】

・ページ識別子で指定されたページに数値を表示する．
・座標（x，y）に数値 v を文字表示する．
・座標（x，y）は数字列の左下を指定する．
・文字高さは h，文字角度は deg で指定する．
・数値から文字列への変換は n によって次のように行なわれる．
〔n ＜ 0 の場合〕
・数値 v の整数部だけを左詰めで文字列に変換する．
・数値 v の整数部は 10 の（n の絶対値 − 1 ）乗の桁まで丸める．
〔0 ≦ n ≦ 9 の場合〕
・小数点以下第（n）位までを左詰めで文字列に変換する．
・n ＝ 0 の場合は小数点までが文字列に加えられる．
〔n ＞ 9 の場合〕
・数値を右詰めで文字列に変換する．
・小数点以下の桁数は n の一の位の数値で指定する．
・全体の文字数は n の十の位以上の数値で指定する．
・ページを指定することができる他は number と同機能．

pladdp

```
void pladdp(double *x, double *y)
```

【互換性】Version 2，Version 3

【引数】

double *x	パスに追加する座標
double *y	パスに追加する座標

【機能】
・カレントパスに指定した座標（*x，*y）を追加する.
・カレントパスが存在しない場合は指定した座標（*x，*y）を始点とする新しいパスを作成しカレントパスとする.

pladdpEx

```
void pladdpEx(RPageDescripter *pd, double x, double y)
```

【互換性】Version 3

【引数】
RPageDescripter *pd ページ識別子
double x　　パスに追加する座標
double y　　パスに追加する座標

【機能】
・ページ識別子で指定されたページのカレントパスに座標を追加する.
・カレントパスに指定した座標（x，y）を追加する.
・カレントパスが存在しない場合は指定した座標（x，y）を始点とする新しいパスを作成しカレントパスとする.
・ページを指定することができる他は pladdp と同機能.

plclpt

```
void plclpt(int *n)
```

【互換性】Version 2，Version 3

【引数】
int *n　　リージョン作成モード（０または１）

【機能】
・カレントパスを閉じて閉じたパスをリージョンに追加する.
・*n が０の場合は指定された順でリージョンを作成するが，１の場合は指定された逆順でリージョンを作成する.

【制限】
・パスを構成する座標は三点以上なければならない.

plclptEx

```
void plclptEx(RPageDescripter *pd, int n)
```

【互換性】Version 3

【引数】
RPageDescripter *pd ページ識別子
int n　　リージョン作成モード（０または１）

【機能】
・ページ識別子で指定されたページにリージョンを追加する.
・カレントパスを閉じて閉じたパスをリージョンに追加する.
・*n が０の場合は指定された順でリージョンを作成するが，１の場合は指定された逆順でリージョンを作成する.
・ページを指定することができる他は plclpt と同機能.

【制限】
・パスを構成する座標は三点以上なければならない.

plexit

```
void plexit()
```

【互換性】Version 2，Version 3

【機能】
・描画処理を終了してページを画面上に表示する.
・描画処理は plsize の引用から plexit の引用までで可能である.

【制限】
plsize の引用から plexit の引用までの間以外で描画ルーチンを引用してはならない.

plexitEx

```
void plexitEx(RPageDescripter *pd)
```

【互換性】Version 3

【引数】
RPageDescripter *pd ページ識別子

【機能】
・ページ識別子で指定されたページの描画処理を終了する.

・描画処理を終了してページを画面上に表示する.
・描画処理は plsize の引用から plexit の引用までで可能である.
・ページを指定することができる他は plexit と同機能.

【制限】
plsizeEx の引用から plexitEx の引用までの間以外で描画ルーチンを引用してはならない.

plfont

```
void plfont(int *n)
```

【互換性】Version 2，Version 3

【引数】
int *n　　フォント番号（1 ～ 10）

【機能】
・指定したフォント番号にフォントを変更する.
・この後に引用される symbol，number，mdsymb，rtsymb に影響を及ぼす.
・フォント番号に対するフォントの割り当てはユーザーが変更可能.

【制限】
・フォント番号は 1 ～ 10 以外を指定してはならない.

plfontEx

```
void plfontEx(RPageDescripter *pd, int n)
```

【互換性】Version 3

【引数】
RPageDescripter *pd ページ識別子
int n　　フォント番号（1 ～ 10）

【機能】
・ページ識別子で指定されたページのフォントを変更する.
・指定したフォント番号にフォントを変更する.
・この後に引用される symbolEx，numberEx，mdsymbEx，rtsymbEx に影響を及ぼす.
・フォント番号に対するフォントの割り当てはユーザーが変更可能.
・ページを指定することができる他は plfont と同機能.

【制限】
・フォント番号は 1 ～ 10 以外を指定してはならない.

plhalt

```
void plhalt()
```

【互換性】Version 2，Version 3

【機能】
・プログラムを一時停止状態にしてユーザーの操作を待つ.

【注意】
・このルーチンはバッチ型のプログラムで Serenade を使う場合に有効である．つまりバッチ型のプログラムが Serenade を使って描画を行なった後，そのままプログラムが終了すると描画結果はあっという間に閉じられて消えてしまう．そのためにわざとプログラムの終了を延期して描画結果を画面上に表示する時間を与えるものである．一時停止状態は Serenade 画面を閉じることで解消されプログラムの実行に戻る．したがってユーザーイベント駆動型の Windows アプリケーションではこれを使う必要はない.

plmsg

```
void plmsg(char *s, int ns)
```

【互換性】Version 2，Version 3

【引数】

char *s	表示する文字列（ANSI）
int ns	表示する文字列の長さ（バイト数）

【機能】
・指定した文字列 s を画面に表示する.
・ユーザーが OK ボタンをクリックするまでプログラムは次に進まない.
・描画を行なうものではないので描画ページが有効になっていない状態でも引用できる.

plot

```
void plot(double *x, double *y, int *n)
```

【互換性】Version 2，Version 3

【引数】

double *x	ペンの移動先座標
double *y	ペンの移動先座標
int *n	移動モード

【機能】
・座標（*x，*y）にペンを移動する.
・*n の絶対値が２なら現在の座標から移動先までを直線で描画する.
・*n の絶対値が３なら移動先までペンを移動するだけで描画は行なわない.
・*n が負の場合，移動先を新しく原点とする.
・*n の絶対値が２でも３でもない場合は何もしない.

plotEx

```
void plotEx(RPageDescripter *pd, double x, double y, int n)
```

【互換性】Version 3

【引数】

RPageDescripter *pd	ページ識別子
double x	ペンの移動先座標
double y	ペンの移動先座標
int n	移動モード

【機能】
・ページ識別子で指定されたページのペンを移動する.
・座標（x，y）にペンを移動する.
・n の絶対値が２なら現在の座標から移動先までを直線で描画する.
・n の絶対値が３なら移動先までペンを移動するだけで描画は行なわない.
・n が負の場合，移動先を新しく原点とする.
・n の絶対値が２でも３でもない場合は何もしない.
・ページを指定することができる他は plot と同機能.

plread

```
void plread(char *s1, int ns1, char *s2, int ns2, int *ms2)
```

【互換性】Version 2，Version 3

【引数】

char *s1	プロンプト文字列（ANSI）
int ns1	プロンプト文字列長さ（バイト数）
char *s2	《出力》ユーザーが入力した文字列（ANSI）
int ns2	ユーザー入力文字列用変数（char *s2）サイズ（バイト数）
int *ms2	《出力》実際に入力された文字列の長さ（バイト数）

【機能】

・プロンプト文字列を持つ文字列入力用が面を表示する.
・入力テキストボックスにはデフォールトとして s2 が入っている.
・OK ボタンかキャンセルボタンのどちらかがクリックされるまでプログラムは次に進まない.
・OK ボタンがクリックされたら入力テキストボックスの文字列を s2 にセットし，*ms2 にはその長さをセットして戻る.
・キャンセルボタンがクリックされたら何もセットしないで戻る.
・描画を行なうものではないので描画ページが有効になっていない状態でも引用できる.

plrgbp

```
void plrgbp(double *r, double *g, double *b, int *n)
```

【互換性】Version 2，Version 3

【引数】

double *r	カラーパラメーター赤（０～１）
double *g	カラーパラメーター緑（０～１）
double *b	カラーパラメーター青（０～１）
int *n	描画モード（現在未使用）

【機能】
・リージョン内部を指定した色で塗りつぶす.
・各カラーパラメーターは０が輝度最小で１が輝度最大である．たとえばすべて０なら黒，すべて１なら白となる.
・塗りつぶし後にリージョンは削除される.

【制限】
・リージョンが存在しない状態で塗りつぶしを行なってはならない.
・各カラーパラメーターに負の値や１を超える値を指定してはならない.

plrgbpEx

```
void plrgbpEx(RPageDescripter *pd, double r, double g, double b, int n)
```

【互換性】Version 3

【引数】

RPageDescripter *pd	ページ識別子
double r	カラーパラメーター赤（０～１）
double g	カラーパラメーター緑（０～１）
double b	カラーパラメーター青（０～１）
int n	描画モード（現在未使用）

【機能】

・ページ識別子で指定されたページのリージョンを塗りつぶす.
・リージョン内部を指定した色で塗りつぶす.
・各カラーパラメーターは０が輝度最小で１が輝度最大である．たとえばすべて０なら黒，すべて１なら白となる.
・塗りつぶし後にリージョンは削除される.
・ページを指定することができる他は plrgbp と同機能.

【制限】
・リージョンが存在しない状態で塗りつぶしを行なってはならない.
・各カラーパラメーターに負の値や１を超える値を指定してはならない.

plsize

```
void plsize(double *x, double y, char *unit, int n)
```

【互換性】Version 2，Version 3

【引数】

double *x	描画ページ幅
double *y	描画ページ高さ
char *unit	単位 "mm"，"cm"，"m"，"in"，"ft"（ANSI）
int n	単位の文字数（バイト数）

【機能】
・指定したサイズの描画ページを新規作成し有効にする.
・有効にした描画ページの座標指定の単位は unit になる.
・ペン番号とフォント番号はどちらも１に初期化される.
・原点は（０，０）に初期化される.
・拡大率は１に初期化される.
・単位指定では大文字と小文字の区別はしない.

【制限】
・有効な描画ページがある状態（plexit で完成されていない描画ページがある状態）で plsize を引用してはならない.
・対応していない単位を使用してはならない.

plsizeEx

```
RPageDescripter plsizeEx(double x, double y, wchar_t *unit)
```

【互換性】Version 3

【戻り値】新規作成された描画ページのページ識別子

【引数】

double x	描画ページ幅
double y	描画ページ高さ
wchar_t *unit	単位 "mm", "cm", "m", "in", "ft"（UTF8）

【機能】
・指定したサイズの描画ページを新規作成し有効にする.
・有効にした描画ページの座標指定の単位は unit になる.
・ペン番号とフォント番号はどちらも１に初期化される.
・原点は（０, ０）に初期化される.
・拡大率は１に初期化される.
・単位指定では大文字と小文字の区別はしない.
・作成した描画ページのページ識別子を関数値として返す.
・ページ指定を行なう描画ルーチンを引用する際にはこの戻り値をページ識別子として指定する.

【制限】
・対応していない単位を使用してはならない.

pl_Initialize

```
void pl_Initialize(TComponent *Owner)
```

【互換性】Version 3

【引数】
TComponent *Owner オーナーフォームオブジェクトへのポインター

【機能】
・セレナーデウィンドウのオーナーとなるフォームを設定する.
・オーナーフォームが削除されるときにセレナーデウィンドウも自動的に削除される.
・バッチ型アプリケーションのようにフォームを持たないアプリケーションでは引用する必要はない. この場合オーナーは NULL となる.

【制限】
・オーナーフォームは C++Builder か Delphi の TComponent でなければならない.

pl_PageExists

```
HWND pl_PageExists()
```

【互換性】Version 3

【戻り値】セレナーデのウィンドウハンドル. セレナーデウィンドウが存在しない場合は NULL を返す.

【機能】

・セレナーデウィンドウハンドルを取得する.

pl_PageExistsEx

```
int pl_PageExistsEx(RPageDescripter *pd)
```

【互換性】Version 3

【戻り値】指定されたページが存在すれば１，存在しなければ０

【引数】
RPageDescripter *pd ページ識別子

【機能】
・ページ識別子で指定されたページが存在するかどうかを取得する.

【補足】
描画ページはユーザーの操作によって閉じられる場合がある．アプリケーションはいったん閉じられた画面の再描画なのかそれともまったく新規の画面描画なのかが認識できなければ，同じ画面を Serenade 画面内に複数表示してしまうことを避けることができない．そこでこのルーチンを使って該当画面が閉じられてしまったかどうかを判断できるようにしてある.

pl_SetRGB

```
void pl_SetRGB(double *r, double *g, double *b)
```

【互換性】Version 2，Version 3

【引数】

double *r	カラーパラメーター赤（０～１）
double *g	カラーパラメーター緑（０～１）
double *b	カラーパラメーター青（０～１）

【機能】
・ペン番号０のペンの色を指定する.
・各カラーパラメーターは０が輝度最小で１が輝度最大である．たとえばすべて０なら黒，すべて１なら白となる.

【制限】
・各カラーパラメーターに負の値や１を超える値を指定してはならない.
・newpen によるペン番号０の指定は pl_SetRGB による色指定の後で行なわなければならない.

pl_SetRGBEx

```
void pl_SetRGBEx(RPageDescripter *pd, double r, double g, double b)
```

【互換性】Version 3

【引数】

RPageDescripter *pd	ページ識別子
double r	カラーパラメーター赤（0～1）
double g	カラーパラメーター緑（0～1）
double b	カラーパラメーター青（0～1）

【機能】

・ページ識別子で指定されたページのペン番号0のペンの色を指定する.

・各カラーパラメーターは0が輝度最小で1が輝度最大である．たとえばすべて0なら黒，すべて1なら白となる.

・ページを指定することができる他は pl_SetRGB と同機能.

【制限】

・各カラーパラメーターに負の値や1を超える値を指定してはならない.

・newpenEx によるペン番号0の指定は pl_SetRGBEx による色指定の後で行なわなければならない.

pl_SetWidth

```
void pl_SetWidth(double *w)
```

【互換性】Version 2，Version 3

【引数】

double *w	ペン番号0の線幅（単位は point）

【機能】

・ペン番号0のペンの線幅を指定する.

・単位は point.　1 point=1/72 インチ.

【制限】

・newpen によるペン番号0の指定は pl_SetWidth による線幅指定の後で行なわなければならない.

pl_SetWidthEx

```
void pl_SetWidthEx(RPageDescripter *pd, double w)
```

【互換性】Version 3

【引数】

RPageDescripter *pd　ページ識別子

double w　　ペン番号０の線幅（単位は point）

【機能】

・ページ識別子で指定されたページのペン番号０のペンの線幅を指定する.

・単位は point.　1 point=1/72 インチ.

・ページを指定することができる他は pl_SetWidth と同機能.

【制限】

・newpen によるペン番号０の指定は pl_SetWidth による線幅指定の後で行なわなければならない.

pl_Terminate

```
void pl_Terminate()
```

【互換性】version 3

【機能】

・セレナーデウィンドウを閉じる.

【制限】

・セレナーデウィンドウのオーナーフォームが pl_Initialize で指定されていなければならない.

【補足】

Version2 まではセレナーデウィンドウを閉じるための手段がユーザー操作以外になかったが，pl_Terminate を引用することでプログラム側からセレナーデウィンドウを閉じることができるようになる.

pl_TextWidthEx

```
double pl_TextWidthEx(RPageDescripter *pd, wchar_t *s, int n, double h)
```

【互換性】Version 3

【戻り値】指定された文字列の文字幅

【引数】

RPageDescripter *pd　ページ識別子

wchar_t *s　　文字列（UTF8）

int n　　フォント番号（１～10）

double h　　文字高さ

【機能】

・ページ識別子で指定されたページにフォント番号 n で文字列 s を高さ h で表示する場合の文字幅を返す.
・文字列の符号化は UTF8.

【制限】
・フォント番号は 1 ～ 10 でなければならない.

rtsymb

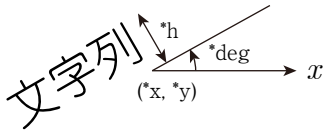
```
void rtsymb(double *x, double *y, double *h, char *s, int ns1, double *deg, int *ns2)
```

【互換性】Version 2，Version 3

【引数】

double *x	文字列描画原点（右下）座標
double *y	文字列描画原点（右下）座標
double *h	文字列高さ
char *s	文字列（ANSI）
int ns1	文字列長さ（バイト数）
double *deg	角度（x 軸から反時計回りの角度．単位は度で -180 ～ 180）
int *ns2	文字列長さ（通常は ns1 と同じ値）

【機能】
・座標（*x，*y）を文字列右下とする文字列 s（ANSI）を描画する.
・文字高さは *h，文字列角度は *deg で指定する.
・文字列の符号化は ANSI コードで行ない，文字列長さはバイト数で指定する.



【制限】
・文字数 ns1 は実際に渡される文字列 s のバイト数でなければならない．一方，文字数 *ns2 は実際に描画する文字数でよいので *ns2 ≦ ns1 の範囲で指定することができる.

rtsymbEx

```
void rtsymbEx(RPageDescripter *pd, double x, double y, double h, wchar_t *s, double deg, int ns)
```

【互換性】Version 3

【引数】

RPageDescripter *pd	ページ識別子
double x	文字列描画原点（右下）座標

double y	文字列描画原点（右下）座標
double h	文字列高さ
wchar_t *s	文字列（UTF8）
double deg	角度（x 軸から反時計回りの角度．単位は度で -180 ～ 180）
int ns	文字列長さ（文字数）

【機能】

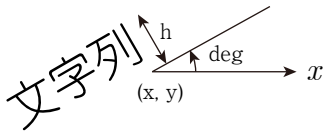
・ページ識別子で指定されたページに文字列を右詰めで描画する．

・座標（x，y）を文字列右下とする文字列 s（UTF8）を描画する．

・文字高さは h，文字列角度は deg で指定する．

・文字列の符号化は UTF8 コードで行ない，文字列長さは実際の文字数（半角文字，全角文字の区別なし）で指定する．

・ページを指定することができる他は rtsymb と同機能．



【制限】

・文字数 ns は実際に渡される文字列 s の文字数以下でなければならない．

symbol

```
void symbol(double *x, double *y, double *h, char *s, int ns1, double *deg, int *ns2)
```

【互換性】Version 2，Version 3

【引数】

double *x	文字列描画原点（左下）座標
double *y	文字列描画原点（左下）座標
double *h	文字列高さ
char *s	文字列（ANSI，センターシンボルの場合は未使用）
int ns1	文字列長さ（バイト数）
double *deg	角度（x 軸から反時計回りの角度．単位は度で -180 ～ 180）
int *ns2	文字列長さ（通常は ns1 と同じ値）または負の整数

【機能】

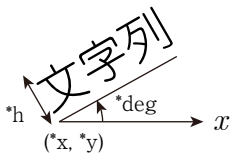
・*ns2 が正の場合は通常の文字列表示，負の場合はセンターシンボル表示となる．

〔文字列表示の場合〕

・座標（*x，*y）を文字列左下とする文字列 s（ANSI）を描画する．

・文字高さは *h，文字列角度は *deg で指定する．

・文字列の符号化は ANSI コードで行ない，文字列長さはバイト数で指定する．



〔センターシンボルの場合〕

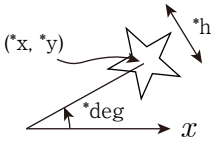
・シンボル高さは *h で指定する．シンボル幅は高さと等しい．

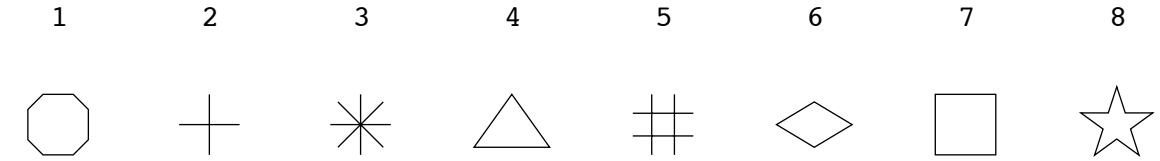
・*s と ns1 は使用しない．

・座標（*x，*y）をシンボルの中心として *deg だけ傾いたシンボルを描画する．

・シンボルパターンは *ns2 の絶対値（１～８）によって決まる．

・*ns2 の絶対値が８を超えている場合は８とみなす．





【制限】

・文字数 ns1 は実際に渡される文字列 s のバイト数でなければならない．一方，文字数 *ns2 は実際に描画する文字数でよいので *ns2 ≦ ns1 の範囲で指定することができる．

symbolEx

```
void symbolEx(RPageDescripter *pd, double x, double y, double h, wchar_t *s, double deg, int ns)
```

【互換性】Version 3

【引数】

RPageDescripter *pd	ページ識別子
double x	文字列描画原点（左下）座標
double y	文字列描画原点（左下）座標
double h	文字列高さ
wchar_t *s	文字列（UTF8，センターシンボルの場合は未使用）
double deg	角度（x 軸から反時計回りの角度．単位は度で -180 ～ 180）
int ns	文字列長さ（文字数）または負の整数

【機能】

・ページ識別子で指定されたページに文字列を左詰めで描画するかセンターシンボルを表示する．

・ページを指定することができる他は symbol と同機能．

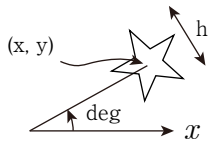
・ns が正の場合は通常の文字列表示，負の場合はセンターシンボル表示となる．

〔文字列表示の場合〕

・座標（x，y）を文字列左下とする文字列 s（UTF8）を描画する.

・文字高さは h，文字列角度は deg で指定する.

・文字列の符号化は UTF8 コードで行ない，文字列長さは文字数で指定する.



1 2 3 4 5 6 7 8

【制限】

・文字数 ns は実際に渡される文字列 s の文字数以下でなければならない.