Visual Basic

東海大学総合情報センター

第４版

目　次

BASIC と Visual Basic 1
１.Visual Basic の操作 2
1-1.Visual Basic の起動 2
1-2.デザインウィンドウの構成 2
1-3.Visual Basic の終了 3
２.フォームの作成 4
2-1.プログラム作成の流れ 4
2-2．標準コントロール 5
2-3．オブジェクトの作成 5
2-4．オブジェクトのプロパティ 6
2-5．イベントプロシージャ 6
2-6．プログラムの保存 8
2-7．プログラムの読込 8
2-8．VB の拡張子一覧 9
３．コントロールとイベント 10
3-1 チェックボックス・オプションボタン 10
3-2.データの入力 11
3-3.タイマーイベント 12
４.メソッド 14
4-1 リストボックス・コンボボックス 14
５．TWIPS と座標系 15
5-1.PRINT メソッドへの応用 16
5-2.グラフィックメソッドへの応用 17
5-3.プリンタ出力 19
６．コンポーネントの追加 19
6-1.Common Dialog 19
７．メニュー 21
８．付録 22
8-1.Visul Basic の文法 22
8-2.ステートメント 23
8-3.主な関数 26
8-4.ヘルプ 27

BASIC と Visual Basic

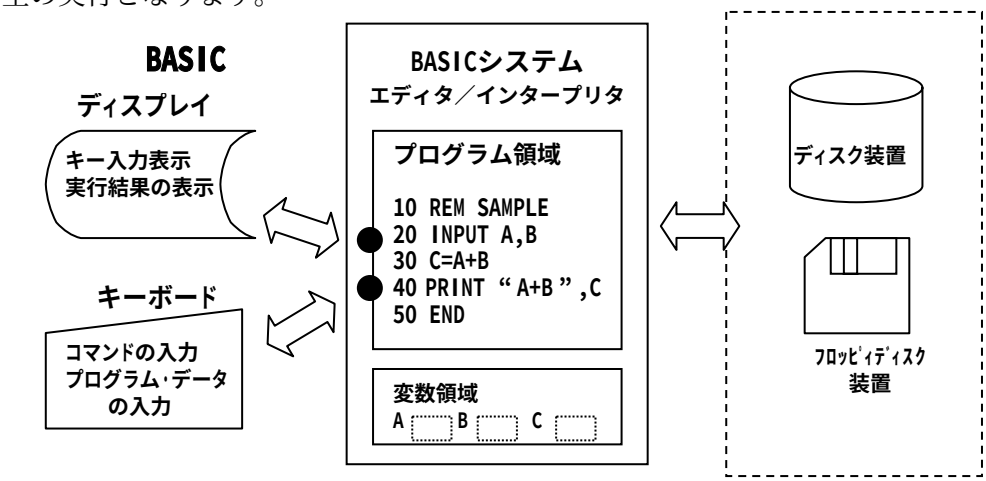
BASIC（Beginner's All purpose Symbolic Instruction Code）は名前の通り古くから親しまれてきた初心者向け、インタープリターシステムのプログラミング言語です。歴史のある言語だけに、プログラムを作成するに当たって構造化やモジュール化が困難という言語仕様上の問題点を抱えています。

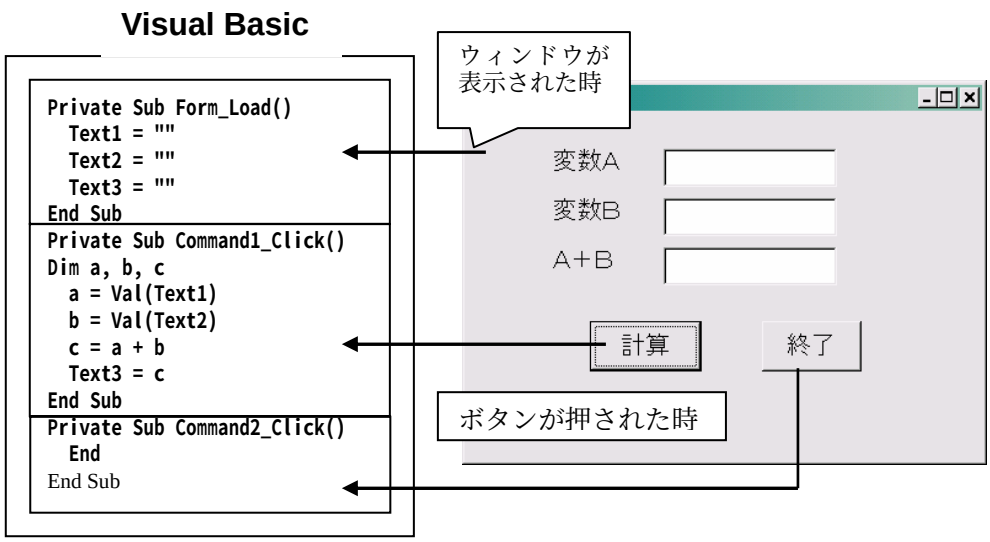
Visual Basic ではこれらの問題点を解消し、Windows スタイルのプログラムの作成が容易にできるよう言語仕様が変わっています。

内蔵されているエディタも変更され、行番号で管理するものから通常のエディタになっています。このため、プログラム中の行番号も無くなっています。

プログラミングの考え方においても従来の BASIC と同じ所もあり、また従来とは全く違い、考え方を改めなくてはならない所もあります。

大きく違う点は、従来の BASIC ではプログラムの実行順序はプログラム作成時に予め決めたとおりに実行する定義型であるの対し、Visual Basic では利用者の操作（ボタンやメニューをクリックする）によって任意の順序でプログラムの実行が行われるイベントドリブン型の実行となります。





1.Visual Basic の操作

Visual Basic（以降 VB と省略）の基本的な操作方法について解説します。

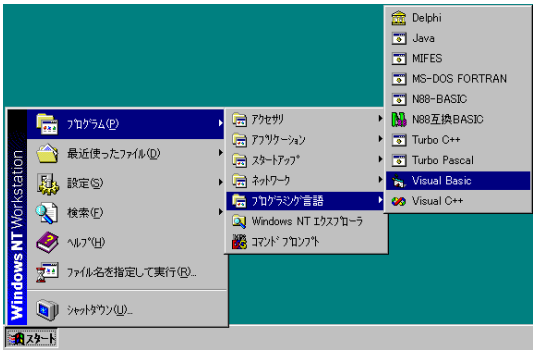
1-1.Visual Basic の起動

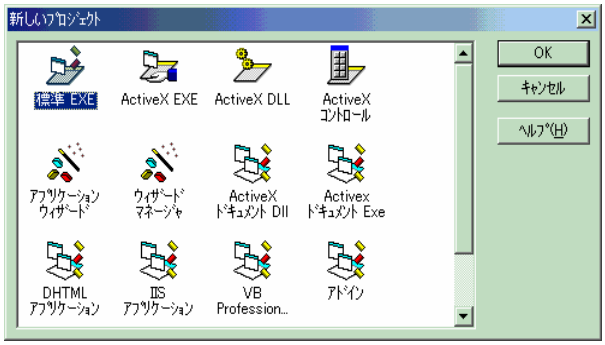
VB の起動は以下の方法で行います。

[スタート] → [プログラム] → [プログラミング言語] → [VisualBasic] を起動します。

起動すると右下の様な「新しいプロジェクト」のウィンドウが表示されます。

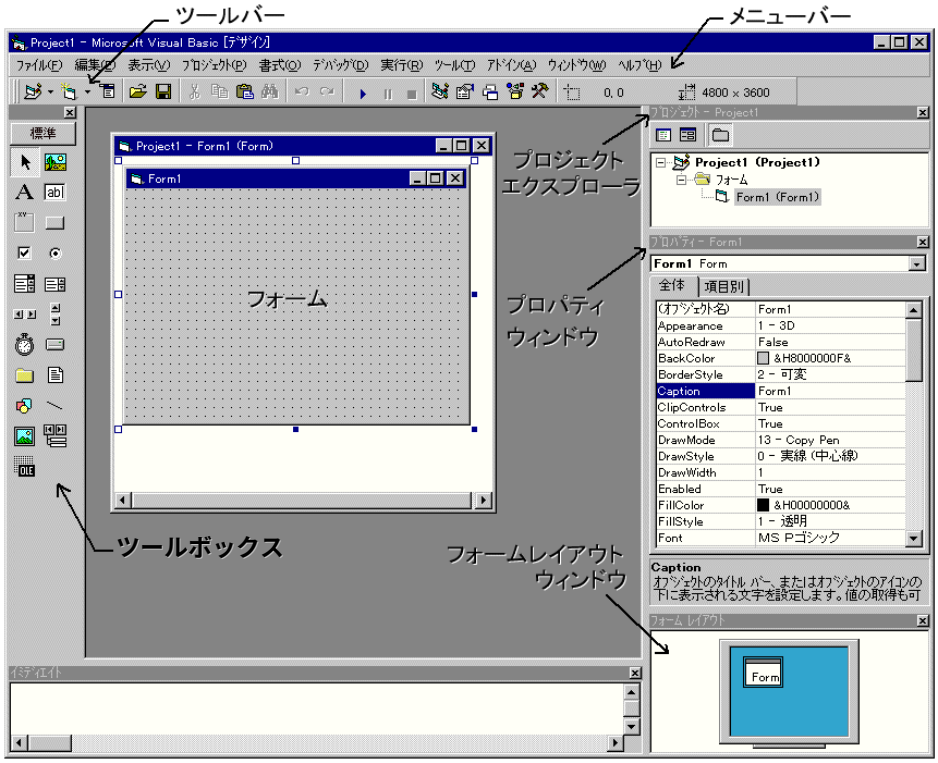
ここでは標準 EXE を選択します。





1-2.デザインウィンドウの構成

起動すると、下のような画面が表示されます。



■ **メニューバー**

メニューバーには、プログラムファイルなどのオープンやクローズ、プログラムの実行、ヘルプの表示など、さまざまな作業コマンドがメニューとして並んでいます。

■ **ツールバー**

メニューバーの中でよく使うコマンドをボタン化したものです。

■ **ツールボックス**

フォーム上に配置するコントロールの一覧

■ **フォーム**

ツールボックスのコントロールを配置する画面であり、Visual Basic でプログラムを作成するベースになります。

■ **プロジェクトエクスプローラ**

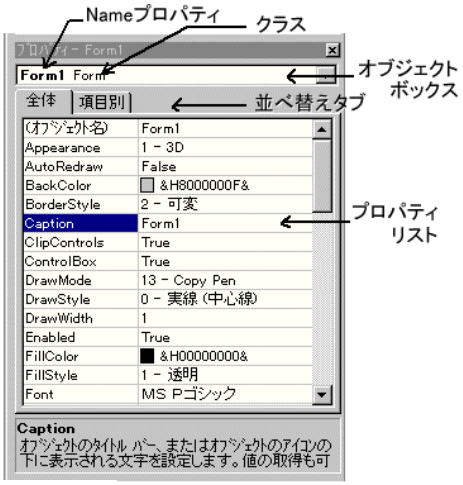
複数のフォームはここで管理しています。

■ **フォームレイアウト**

複数のフォームを使用する場合、他のフォームの位置関係を示す。

■ **プロパティウィンドウ**

各種の属性を設定します。



・オブジェクトボックス

プロパティの設定を行えるオブジェクトの名前が表示される。

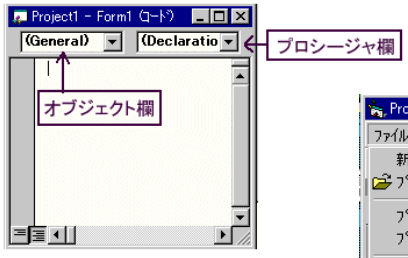
・プロパティリスト

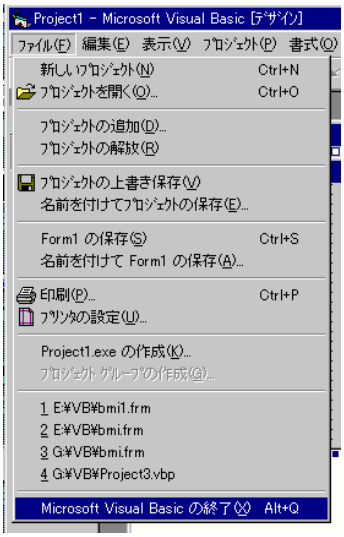
オブジェクトが持っているプロパティの一覧と、設定値が表示されています。

左側がプロパティの名称、右側が設定値になります。

■ **コードウィンドウ**

VB のプログラム（コード）を入力します。





1-3.Visual Basic の終了

Visual Basic の終了方法は、メニューバーの [ファイル(F)] → [Microsoft Visual Basic の終了(X)] をクリックします。または、ウィンドウの右上にある ☒ をクリックします。

２.フォームの作成

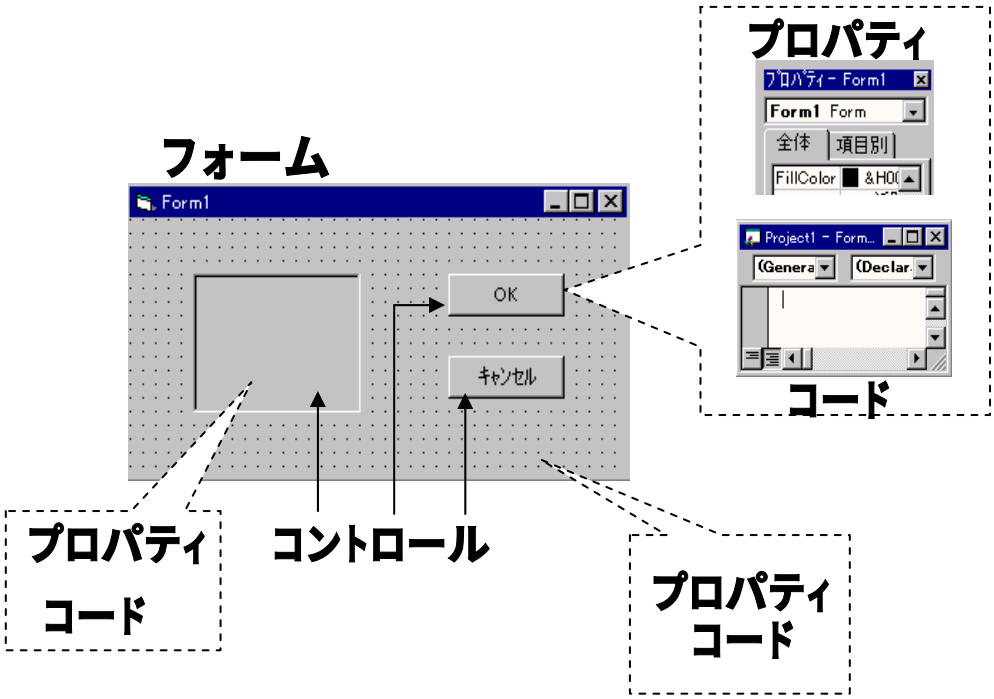
Windows 環境下ではプログラムは画面上に定義したウィンドウ（フォーム）に配置されたメニューやボタン、入力欄をマウスやキーボードを使用して操作することでプログラムの実行を行います。実行結果の表示もフォームを通して行れます。

言い換えれば、プログラムミングの第一歩はまずフォームを作成する事になります。

作成したフォームに配置されたボタンやテキストボックス（これらをコントロールと呼びます。）から発生するイベントに対応する処理を Visual Basic の文法に沿ったコード（プログラム）で記述します。

2-1. プログラム作成の流れ

VB を起動し、標準 EXE　を選択すると以下のように Form1 と名前の付けられたウィンドウが表示されます。



プログラムの作成は次の手順で行います。

① フォームに、データの入力や表示、あるいはプログラムの実行の制御を行うオブジェクト（コントロールオブジェクト）を左側に在る**標準コントロール**の一覧からマウスで選択し、フォーム上に貼り付けて配置します。

② 次に、配置したコントロールオブジェクトの属性（プロパティ）を設定します。

③ コントロールをダブルクリックするとにコードと呼ばれるステートメント（命令）を記述する画面（コードウィンドウ）が表示されます。ここに、イベント（コントロールに対する操作）が発生した場合の処理を記述します。

2-2. **標準コントロール**

1. ポインタ … コントロールではない。ここが凹んでいればマウスポインタが使用できる状態を示す。
2. ピクチャーボックス … イメージやテキストを表示する。
3. ラベル … テキストの表示領域
4. テキストボックス … テキストのキーボード入力、又は表示。
5. フレーム … 関連するオブジェクトをグループとして扱う領域。
6. コマンドボタン … マウスでクリックするボタン。
7. チェックボックス … ON/OFF の設定を行う。
8. オプションボタン … 複数の項目から 1 つを選択するボタン
9. コンボボックス … ドロップダウンリストを表示し、その中の項目を選択する。
10. リストボックス … リストを表示し、その中から 1 つを選択する。
11. 水平スクロールバー … ピクチャーボックス等で画面をはみ出したときに水平方向にスクロールできるバー
12. 垂直スクロールバー … ピクチャーボックス等で画面をはみ出したときに垂直方向にスクロールできるバー
13. タイマー … 一定時間ごとに指定したイベントを発生する。
14. ドライブリストボックス … ドライブリストの表示と選択
15. ディレクトリリストボックス … ディレクトリーパスの表示と選択。
16. ファイルリストボックス … ファイル名の表示と選択
17. シェイプ … 長方形、楕円、円の描画
18. ライン … 直線の描画
19. イメージ … イメージデータを表示する領域
20. データ … データベースのデータにアクセスする。
21. OLE コンテナ … オブジェクトの挿入
（図、表、音、その他そのパソコンで利用できるもの）

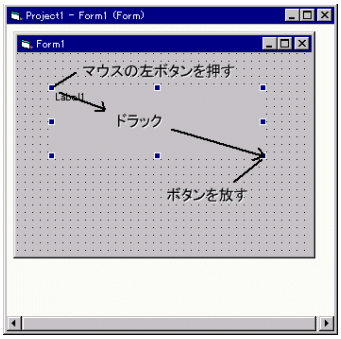
2-3. **オブジェクトの作成**

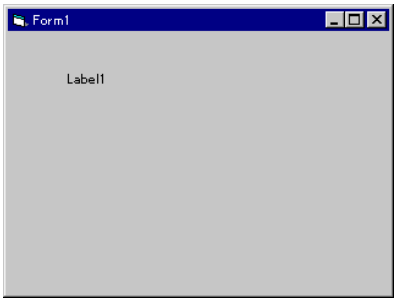
フォーム上に文字列を表示する場合は、標準コントロールのラベルオブジェクトを Form1 に作成します。手順は次のようになります。

① ツールボックスのラベルを選択する。
② マウスを Form 上に移動し、領域をマウスドラックにより指定する。
③ メニューバーの [実行] → [開始] をクリックすると文字列の入ったフォームが表示される。

プログラム（コードの部分）は 1 行も書いていませんが VB ではフォームを作成すことでプ

ログラムのかなりの部分が自動的に策されます。実行すると図のようなウィンドウが表示されます。

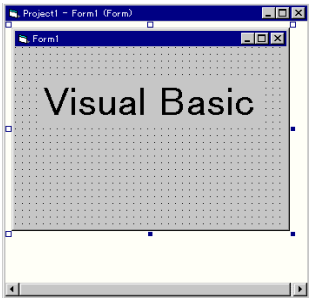




実行の停止をする場合は、メニューバーの [実行] → [終了] をクリックして下さい。

2-4. オブジェクトのプロパティ

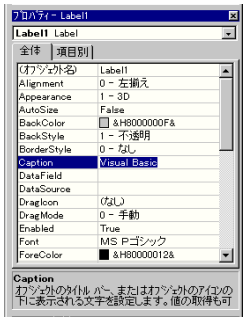
文字列の文字、サイズやフォント等は文字列オブジェクトのプロパティとして定義されます。これらについて説明します。

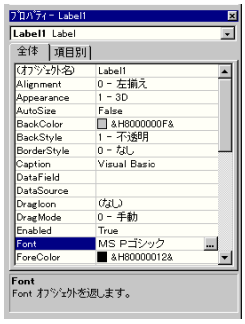


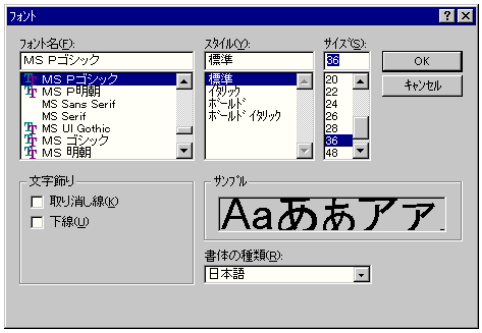
まずは、Label を Form1 に配置してください。次に、文字・フォントサイズの変更の方法について説明します。

まず、Form1 内の Label1 の領域をクリックします。このとき画面の右側に表示されているプロパティウィンドウが Label 用に変わり、Caption 欄が反転されます。この Caption 欄が実際に表示されている文字になります。Caption 覧に「Visual Basic」と入力します。

次に、フォントサイズを変更します。プロパティウィンドウの Font 欄をクリックしてください。すると、フォントの種類の右にボタンがあります。そのボタンをクリックすると、フォント指定のダイアログボックスが表示されます。ここでは、「MS P ゴシック」の「36」を指定します。







この時、文字列が Label の領域に表示できない場合、Label の大きさをドラッグにより調節してください。実行して、フォームの表示を見てください。

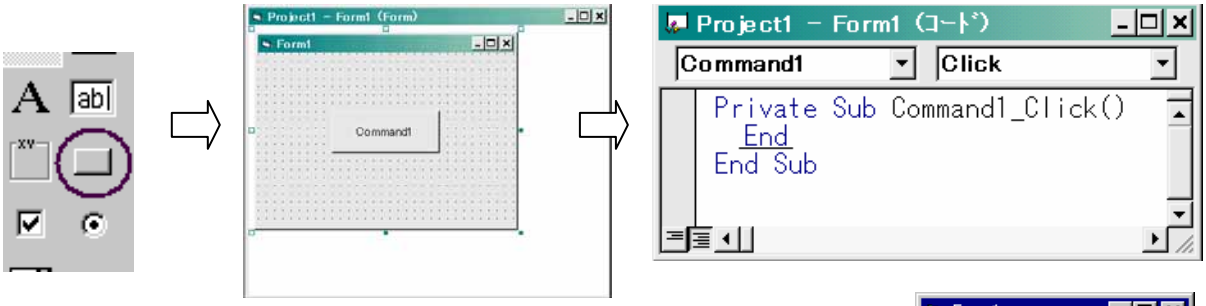
2-5．イベントプロシージャ

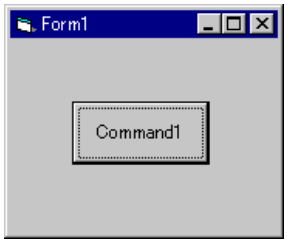
フォーム内に実行の停止を指示するためのコマンドボタンを配置し、コマンドボタンを押す（イベントを発生させる）事によりプログラムの終了を行う機能を追加します。これは以下の手順で行います。

① ツールボックスにある Commond Button を選択し Form1 上に配置します。

② 配置したコマンドボタンをクリックし、コードウィンドウを表示させる。

③ Command1 のオブジェクトの Private Sub で始まる行から End Sub の行の間に End ステートメント下線部）を入力します。



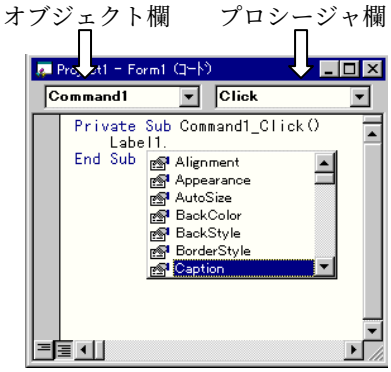


プログラムを入力した後、実行します。実行するとコマンドボタンを持つフォームが表示されます。Command1 ボタンをクリックすると、Click イベントが発生してイベントプロシージャ Command1_Click が実行され、プロシージャ内の End ステートメントが実行されることによりプログラムが終了します。

Private Sub から End Sub の間に記述されたコードを Sub プロシージャと呼び、コマンドボタンのクリックなどのイベントによって起動されるものをイベントプロシージャと呼びます。

コマンドボタンの表面の文字は Caption プロパティによって指定できます。

次に、コマンドボタンを押して、form1 に表示されている文字列を変更するようにプログラム（イベントプロシージャ）を作成する例を示します。まず、別のコマンドボタンをフォーム上に配置し、作成したコマンドボタンをダブルクリックしてコードウィンドウを開きます。



表示されている文字列の変更は Label1 オブジェクトの Caption プロパティを変更することで行われます。プロパティの変更も通常の変数へ値を代入するのと同様に代入文を用いて行います。

コードは Private Sub の次の行から記入します。

```
Label1.
```

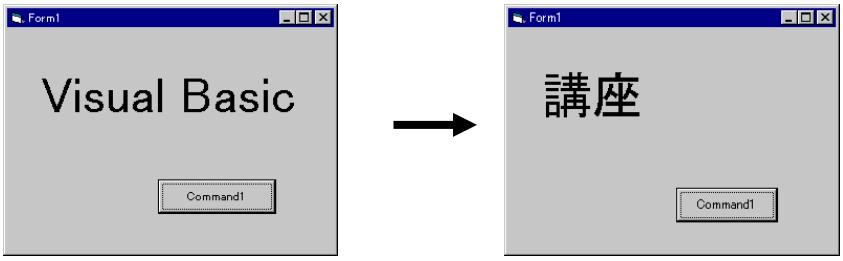
. を記入後、自動的に入力する候補が表示されます。Caption をダブルクリックするか、↑ ↓キーで Caption を選択して

TAB キーをクリックして入力を確定します。続けて、

="講座"　　と入力し代入文を完成させます。

ステートメントや関数およびフォームに配置されているオブジェクト、プロパティ等の名称など、あらかじめ登録されているものは入力中に候補や書式が表示されます。また入力後は自動的に登録されている名称に修正されます。

プログラムを作成した後、実行をすると左のようなフォームが表示されます。ここで、新たに作成したボタンをクリックします。すると、右のフォームのように文字列が「講座」に変更されます。



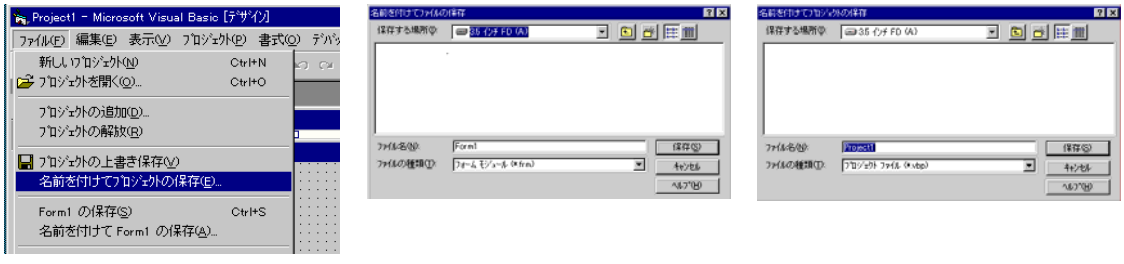
演習

2-5 で作成した Form1 に Command Button を 1 つ追加して、「Visual Basic」の文字に戻す命令を与えてみましょう。

2-6. プログラムの保存

保存するには、メニューバーから [ファイル(F)] → [名前を付けてプロジェクトの保存(E)] をクリックします。通常、VB のプログラムは複数のファイルから構成されます。プロジェクトとは複数のフォーム等のファイルの情報を一括管理するためのファイルです。

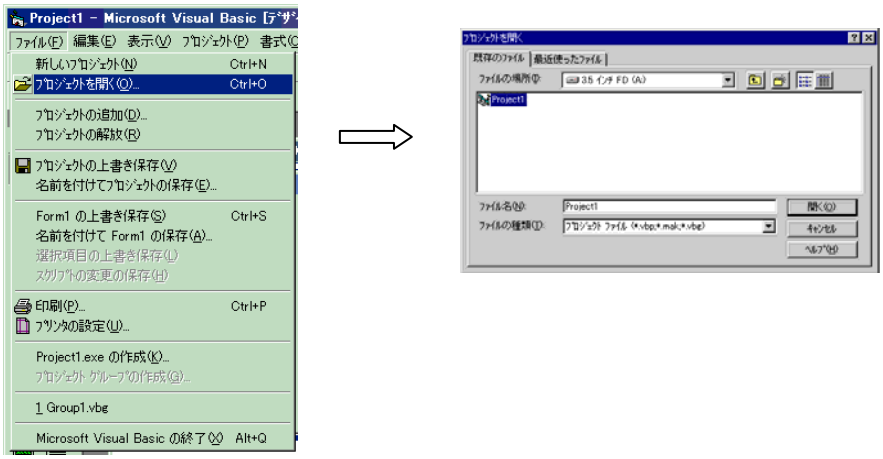
- フォームのファイル名を指定するダイアログボックスが表示されます。名前を付け保存ボタンをクリックします。この時、自動的に拡張子として「.frm」が付きます。
- フォームが複数個ある場合はフォームの個数だけ、保存のダイヤログが表されます。
- 次に、[名前を付けてプロジェクトの保存]ダイアログボックスが表示されます。ここも、名前を付けて保存します。ここでは「.vbp」の拡張子が付きます。



2-7. **プログラムの読込**

保存したプログラムを読み込むには、メニューバーの [ファイル(F)] → [プロジェクトを開く(0)] をクリックします。

クリック後、プロジェクトを開くダイアログボックスが表示されます。そこで、読込みたいファイルを選択し、開く(O) ボタンをクリックしてください。



2-8. VB **の拡張子一覧**

Visual Basic のプログラムを構成するファイルは用途別に多数あります。これらは種類別に拡張子が違っています。現在使われているファイル拡張子は次のとおりです。

拡張子	用途
.bas	標準モジュール
.cls	クラス モジュール
.ctl	ユーザー コントロール ファイル
.ctx	ユーザー コントロール バイナリ ファイル
.dca	アクティブ デザイナ キャッシュ
.dep	セットアップ ウィザード依存ファイル
.dob	ユーザー ドキュメント フォーム ファイル
.dox	ユーザー ドキュメント バイナリ フォーム ファイル
.dsr	アクティブ デザイナ ファイル
.dsx	アクティブ デザイナ バイナリ ファイル
.frm	フォーム ファイル
.frx	バイナリ フォーム ファイル
.log	ロード エラー用ログ ファイル
.oca	コントロール タイプ ライブラリ キャッシュ
.pag	プロパティ ページ ファイル
.pgx	バイナリ プロパティ ページ ファ

拡張子	用途
.res	リソース ファイル
.swt	Visual Basic セットアップ ウィザード テンプレート ファイル
.tlb	リモート オートメーション タイプ ライブラリ ファイル
.vbg	Visual Basic グループ プロジェクト
.vbl	ユーザー コントロール ライセンシング ファイル
.vbp	Visual Basic プロジェクト
.vbr	リモート オートメーション登録ファイル
.vbw	Visual Basic プロジェクト ワークスペース
.vbz	ウィザード起動ファイル

３．コントロールとイベント

VB のプログラムの多くはイベントプロシージャとして記述されます。

２章ではコマンドボタンを押したときに発生する Click イベントに対して処理を行うプログラムを記述しました。

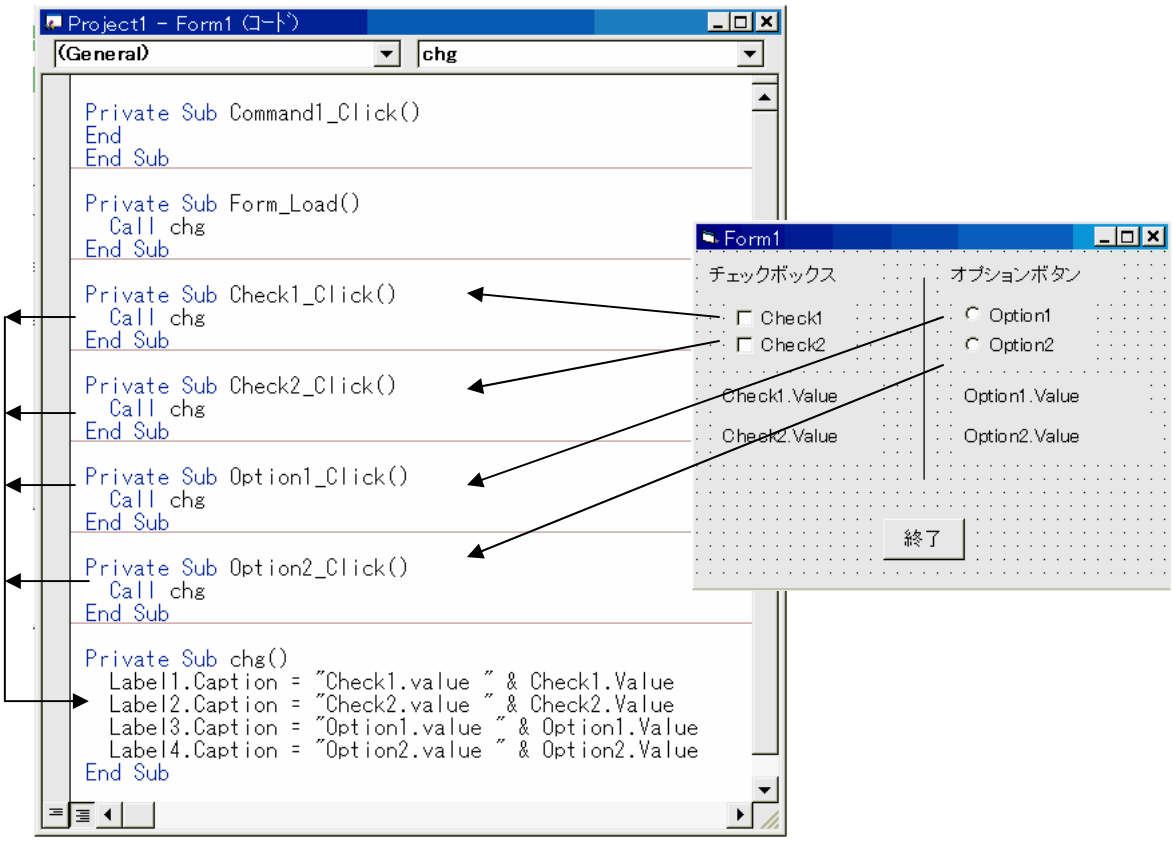
この章では、コマンドボタン以外のコントロールのイベント処理について解説します。

3-1 チェックボックス・オプションボタン

チェックボックスとオプションボタンはどちらも Value プロパティに On/Off の状態が反映されます。違う点はチェックボックスが個別に On/Off の設定ができるのに対し、オプションボタンは同一フォーム上にあるオプションボタンの内 On になるのは１つだけです。

チェックボックスまたはオプションボタンをクリックすると Click イベントが発生し各オブジェクトの状態は Value プロパティに反映されます。

Sub プロシージャはイベントプロシージャとして使用されるだけでなく Call ステートメントから呼び出し利用する Sub プロシジャーとして定義することもできます。



演習

フロッピィディスクの Misc フォルダの中に納められているサンプルプログラム chk_opt を実行してチェックボックスとオプションボタンの機能を確認してください。

3-2.データの入力

従来の BASIC ではキーボードからデータの入力は INPUT ステートメントによって行われていましたが VB では Input ステートメントはファイルからの入力のみとなっています。

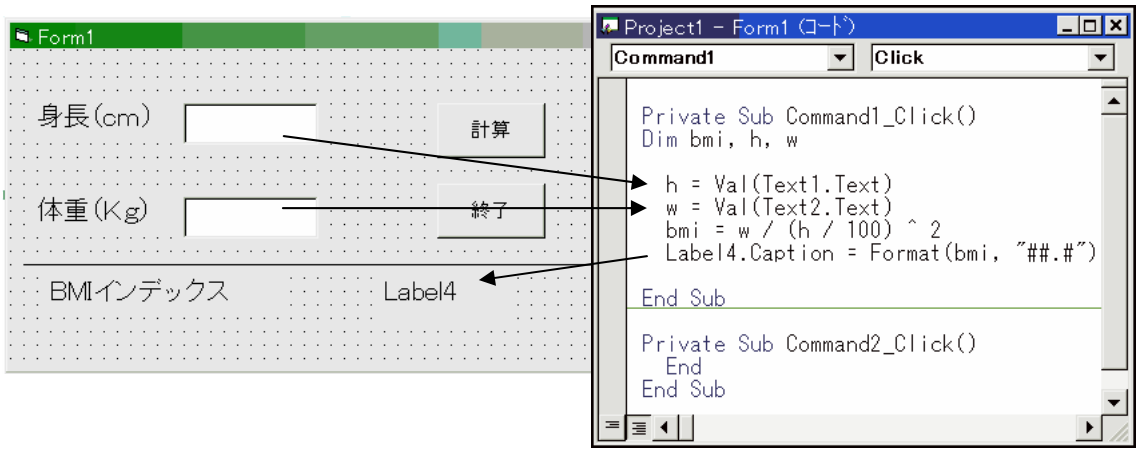
キーボードからの入力を行うためにはテキストボックス、表示を行うためにはラベルのコントロールを使用する必要があります。

テキストボックスへの入力はヴァリアント型の文字列となります。これを数値として扱うためには Val 関数を使用して文字列を数値に変換する必要があります。また、ラベルは文字列を表示するものですから、数値を表示するためには Str 関数または Format 関数を使用して数値を文字列に変化して Caption プロパティに代入します。

以下のプログラムは身長と体重から BMI インデックスを計算するプログラムです。

ＢＭＩは次の式で求められます。

$$bmi = \frac{体重(Kg)}{身長(m)^2}$$



Command1_Click の Sub プロシージャ内の Dim ステートメントで定義されている変数 h,w.bmi はヴァリアント型となります。これの型は最初に代入される値の型を持ちます。

Sub プロシージャ内で定義された変数は定義されたプロシージャ内だけで参照が可能です。

演習

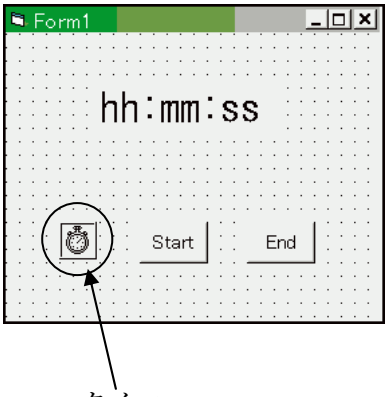
フロッピィディスクの bmi フォルダの中に納められているサンプルプログラム bmi を実行してテキストボックス、計算式等の代入文の機能を確認してください。

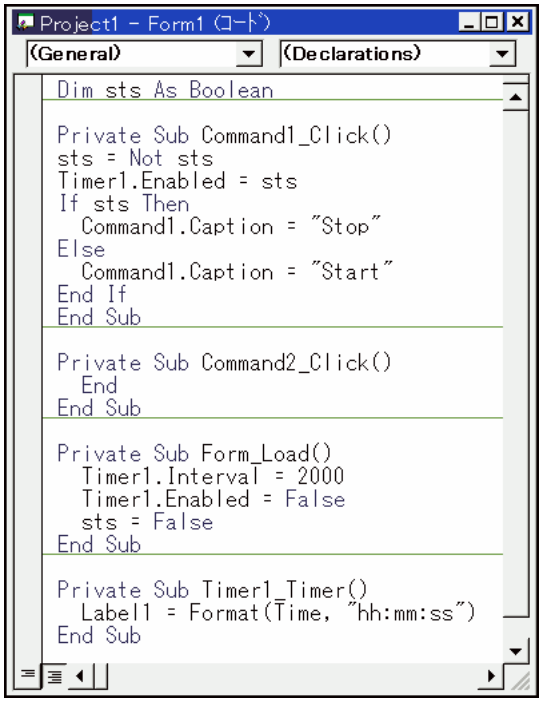
また、このフォルダにある Bmi0 のプログラムはテキストボックスやラベル等のコントーロールオブジェクトを使用せずに Inputbox、Msgbox の関数を使用して同じ計算を行うプログラムです。

3-3.タイマーイベント

コマンドボタンのようにマウスをクリックする事によって発生するイベントとは違って指定した時間間隔で定期的にイベントを発生させるのがタイマーイベントです。

タイマーオブジェクトはフォームに埋めこみますが実行時に表示はされません。





タイマーイベントが発生する間隔は Interval プロパティでミリ秒(ms)単位で指定します。イベントの発生は開始／停止は Enabled プロパティで制御します。

Enabled = True	タイマーイベント発生
Enabled = False	タイマーイベント停止

General の場所で定義されたブール型の変数 sts はフォームの中に含まれるすべてのプロシージャから共通に参照できる変数となります。プロシージャ内で Dim ステートメントによって定義された変数はプロシージャ内のみ参照可能となります。

プログラムは起動時に Form_Load のイベントが発生します。これを利用して変数の初期化やタイマーの初期化を行います。

コマンドボタン Command1 はクリックする度に sts 変数の値によって Caption の文字列が Start/Stop と交互に表示するように If ステートメントを記述しています。

変数 sts はブール型（論理型）なので条件式の変わりに sts 変数のみを記述しています。タイマーの状態の制御は sts 変数の値をそのまま利用しています。

```
sts = Not sts        ← 論理式
Timer1.Enabled = sts
If sts Then
   Command1.Caption = "Stop"
Else                 ← 関係式
   Command1.Caption = "Start"
End If
```

演習

フロッピィディスクの clock フォルダの中に納められているサンプルプログラム clock を実行して If ステートメントの働きを確認してください。

Interval プロパティを変更しイベントの発生間隔を変更してみましょう。

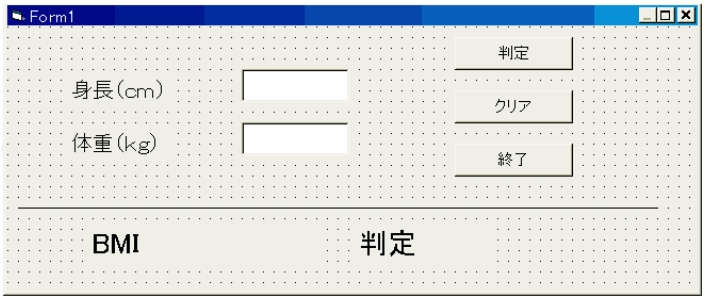
総合演習

１．下図のようなフォームをデザインし、テキストボックスに身長（ｃｍ）と体重（ｋｇ）を入力して BMI インデックスを計算し痩せ型・肥満型の判定を表示するプログラムを作成しなさい。BMI の判定はコマンドボタンによって行う。

判　定のボタンをクリックすると BMI インデックスを計算し判定を表示する。

クリアのボタンを押した場合は入力用のテキストボックスの内容を消去する。

終　了のボタンを押すとプログラムを終了する。



Bmi の値と体型は以下のようになります

Bmi ＜ 19	痩せ型
19 ≦ bmi ＜25	標準型
bmi ≧ 25	肥満型

⇨

```
If bmi < 19 Then
   痩せ型の場合
ElseIf bmi >= 19 And bmi < 25 Then
   標準型の場合
Else
   肥満型の場合
endif
```

２．BMI による体型の判定プログラムが作成できたなら、フロッピィディスク内の bmi_img フォルダには痩せ型、標準型、肥満型に対応した gif 形式の画像が在ります。これを利用して体型の判定に対応する画像を表示する様にプログラムを変更しなさい。画像は次の手順でイメージボックスに貼り付けフォーム上に配置します。

① フォーム上にイメージボックスを作成する。

② Picture プロパティに画像ファイルを指定する

③ Strech プロパティを True、Visible を False に、BorderStyle を「0 - なし」に設定する。

判定に対応するイメージボックスの Visible プロパティを True に変えることにより画像の表示，非表示のコントロールが行えます。

４.メソッド

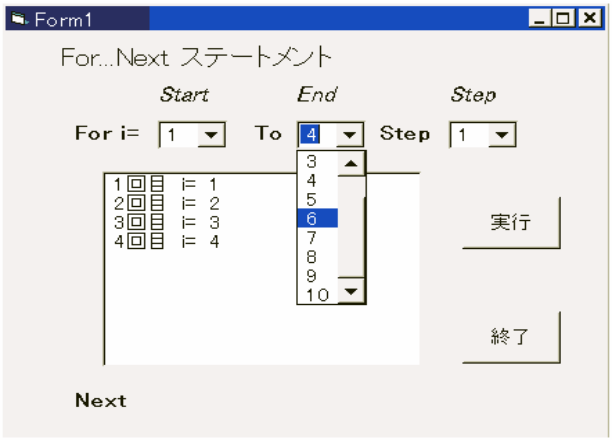
メソッドとはオブジェクトに対してなんだかの操作を要求するものです。従来の BASIC で使用できたグラフィックス関連のステートメントや PRINT ステートメントは**メソッド**として実行されます。Show や Hide（オブジェクトの表示・非表示）などのメソッドのようにプロパティの値を変更することによっても同様の処理が行えるものもあります。

4-1 リストボックス・コンボボックス

リストボックスはあらかじめ定義されたリストを表示し、リストから文字列を選択するための入力用のコントロールオブジェクトです。コンボボックスはテキストボックスとリストボックスを一緒にしたもので直接文字列の入力も可能なようになっています。

リストボックスあるいはコンボボックスにリストを作成するためには Additem メソッドを使用します。リストを削除するためには Clear メソッドを使用します。

コンボボックス、リストボックスを使用して For ステートメントの動作を説明するためのプログラムの例を示します。For ステートメントの繰り返しの初期値、終値、ステップ値をコンボボックスによって設定します。実行の様子はリスボックスのリストとして表示します。



コンボボックスとリストボックスの値は Text プロパティに反映されます。

```
Dim p1, p2, p3
_____________________________________
Private Sub Combo1_click()
p1 = Val(Combo1.Text)
End Sub
_____________________________________
Private Sub Combo2_click()
p2 = Val(Combo2.Text)
End Sub
_____________________________________
Private Sub Combo3_click()
p3 = Val(Combo3.Text)
End Sub
_____________________________________
Private Sub Command1_Click()
Dim i, n
List1.Clear    ←  リストの消去
For i = p1 To p2 Step p3
  n = n + 1
  List1.AddItem Str(n) & "回目  i= " & Str(i)
Next
End Sub
_____________________________________
Private Sub Command2_Click()
End
End Sub
_____________________________________
Private Sub Form_Load()
Dim i
p1 = 1: p2 = 1: p3 = 1
For i = 1 To 10
  Combo1.AddItem Str(i)     ←リストの登録
  Combo2.AddItem Str(i)
Next
For i = 1 To 3
  Combo3.AddItem Str(i)    ←リストの登録
  Combo3.AddItem Str(-i)
  Next
End Sub
```

演習

フロッピィディスクの misc のフォルダ内のプログラム for を実行して For…Next ステートメントによる繰り返しの動作を確認してください。

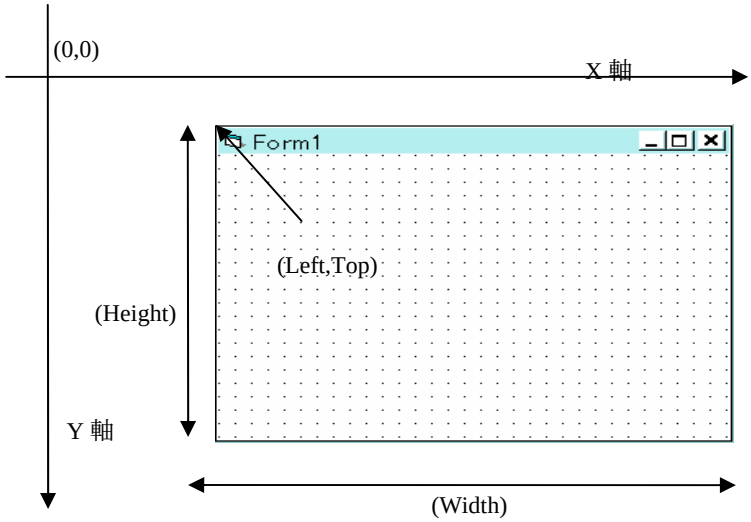
注意）step の値が 0 の場合はエラーで実行が終了します。
Start と End が等しい場合も一回繰り返しがおきます。

５．TWIPS と座標系

従来 BASIC で利用してきた PRINT ステートメントは Print メソッドとなります。Print メソッドを使用する場合はフォーム（ピクチャーボックス）の座標系を理解することが必要となります。

フォームの単位系は TWIP で表されます。これはフォームをプリンタに出力した場合、以下の倍率で正規化されます。

567twip=1cm
1440twip=1 インチ



フォームの位置は Left,Top のプロパティで決定されます。大きさは Hight,Width の各プロパティによって決まります。座標系は画面左上が原点となります。

Width，Height プロパティはフォームの物理的な大きさを指定します。単位は twip です。

フォームに対する書き出し位置は CurrentX、CurrentY プロパティに保持されています。このプロパティの値を変更することにより PRINT メソッドやグラフィックメソッドの書き出し位置を設定することができます。

twip 以外の座標系を利用したい場合は ScaleMode プロパティの値を 0,1 以外の値に設定します。これによって以下のような単位系に設定できます。

0	ユーザ定義座標系（VbUser）	
1	twip 座標系（VbTwips）	既定値
2	ポイント（VbPoints）	論理インチあたり 72 ポイント
3	ピクセル （VbPixels）	
4	文字数（VbCharacters ）	１文字 120 twip(横)×240 twip(縦)
5	インチ（VbInches）	
6	ミリメートル（VbMillimeters）	
7	センチメートル VbCentimeters）	

ScaleMode を 0,1 以外に設定した場合でも、ScaleHeight、ScaleWidth、ScaleTop、ScaleLeft 等のスケールプロパティの値を変更すると自動的に ScaleMode は 0（ユーザ定義座標系）に変更されます。

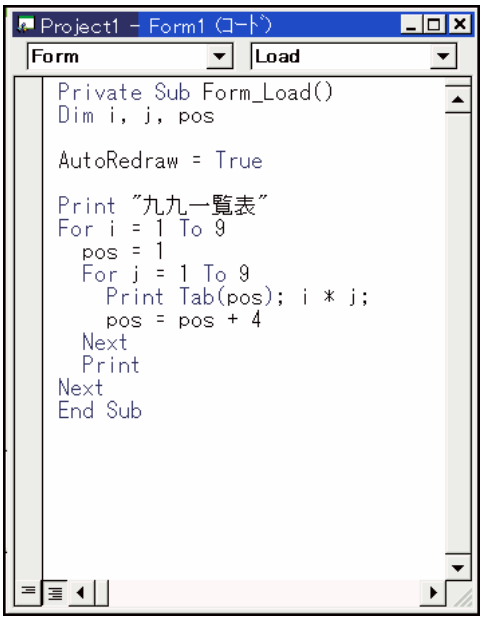
ScaleMode を 0,1 以外に設定し、ScaleHeight、ScaleWidth の各値を希望する数値に設定する場合は Hight,Width の値を変化させて調整します。ScaleMode を 4 に設定するとフォームの大きさを 1 行の文字数と行数で設定することができます。実際に出力する場合の目安となります。

5-1.PRINT メソッドへの応用

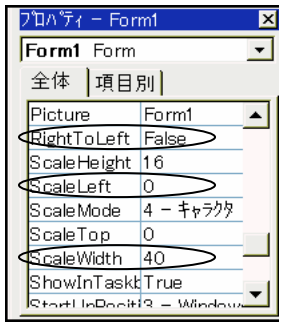
print メソッドの使用して九九の一覧表を作成する例を示します。Print メソッドは従来の Print ステートメントと同様に利用できます。但し、フォームは自動的にスクロールしない点に注意が必要です。ピクチャーボックスにも同様に出力できます。但し、フォームの範囲外に書かれたデータは表示されません。

ピクチャーボックスで Print メソッドを使用する際にも AutoRedraw プロパティが True になっている必要があります。通常は False になっています。False のままだと表示されません。

AutoRedraw プロパティはフォームまたはピクチャーボックスに書き換えが発生した場合の再描画の方法を指示するものです。頻繁に画面の書き換えが発生するプログラムにおいてこのプロパティを True にしておくと実行に時間がかかる場合があります。



Scale プロパティは「プロパティ－Form1」のように設定します。



また、フォームに直接表示するのではなく、リストボックスに Additem メソッドを使用して表示することもできます。詳細はフロッピィディスクの print フォルダの中の kuku2 のプログラムを参照してください。

5-2.グラフィックメソッドへの応用

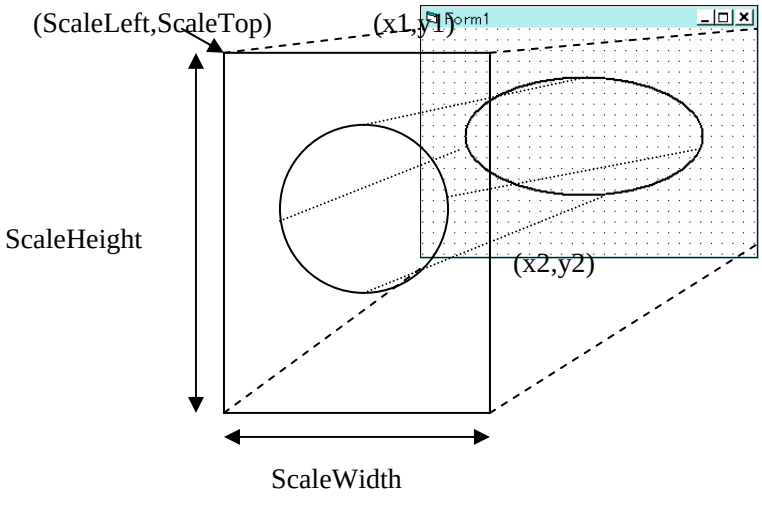
グラフィック関連(LINE や CIRCLE 等)のステートメントはグラフィックメソッドとして利用できます。グラフィックメソッドで使用する座標系は ScaleHeight、ScaleWidth、ScaleTop、ScaleLeft のスケールプロパティで規定されます。

ScaleHeight、ScaleWidth はフォームの論理的な大きさを、ScaleTop、ScaleLeft は論理的な原点の位置を定義します。

スケールプロパティで指定される座標系とフォームとの対応関係は右の図のようになります。

これらスケールプロパティの設定はスケールメソッドを使用することによって設定することもできます。

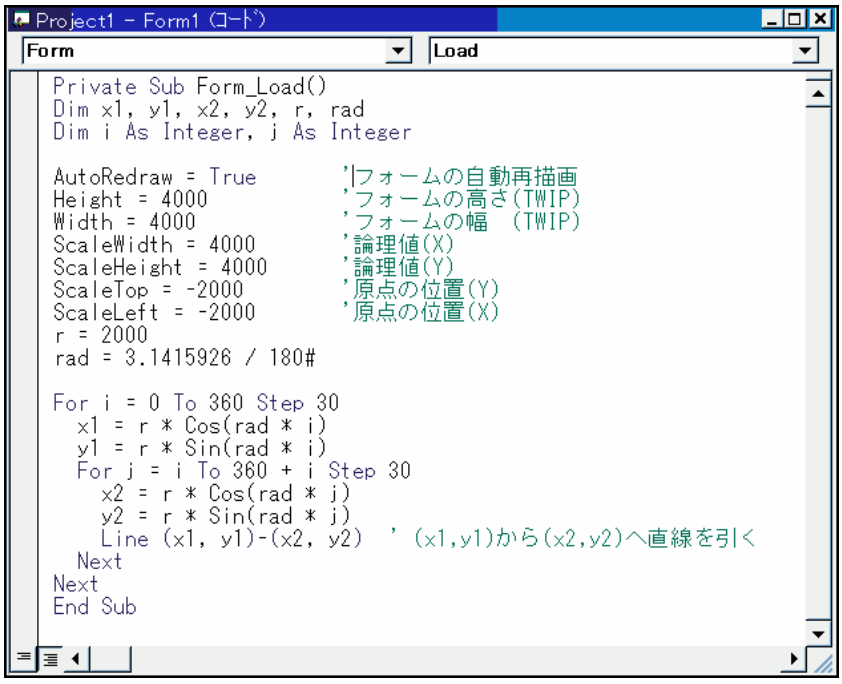
スケールメソッドでは論理的な座標系の左上隅と右下隅の座標を設定することで行います。



Object.Scale (x1,y1) – (x2,y2)

スケールメソッドあるいはスケールプロパティで指定する矩形とフォームが相似でない場合は描画される図形は変形を起こします。

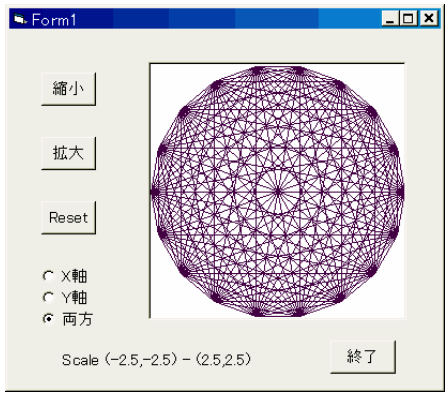
フォームに Line メソッドを使用して図形を描画するプログラムの例を示します。



この例ではフォームの大きさや Scale プロパティはプログラム実行時に設定しています。デザイン画面での Form1 の各プロパティをあらかじめ設定することも可能です。
このプログラムは Form_Load イベント（フォームがメモリにロードされるときに発生するイベント）が発生したときに実行されます。

フロッピィディスクの line フォルダの中に line1,line2 の２つのプログラムがあります。Line1 は上に示したプログラムでフォームに描画します。Line2 はピクチャーボックス内に描画します。

描画方法に違いはありません。但し、line2 プログラムでは描画を行うオブジェクトが Form1 から Picture1 に変わっているため Line メソッドが次のように書き換えられています。



Picture1.Line (x1,y1) – (x2,y2)

演習
フロッピィディスクの scale.exe を実行し、拡大、縮小のボタンを押したときのスケールメソッドのパラメータの変化とピクチャーチャーボックス内に描画される図形の変化の関係を考えてください。

5-3.プリンタ出力

作成したフォームをプリンタに出力する場合は PrintForm メソッドを使用します。

PrintForm メソッドは印刷する対象のフォームを作成した後に実行します。出力先のプリンタは「通常使うプリンタ」に設定されているプリンタに出力されます。

Object.PrintForm

印刷時にプリンタの設定、選択等を行う必要がある場合は「６．コンポーネントの追加」の章で解説してある Common Dialog のオブジェクトを使用して実現します。

また、フォームを印刷する場合は、印刷領域はフォームのサイズできまってきます。フォームの最大値が A５のサイズより少し大きい領域までとなります。これ以上の用紙に印刷を行う場合は Printer メソッドを使用します。

詳しくはヘルプで「Printer オブジェクト」をキーワードにして検索してください。

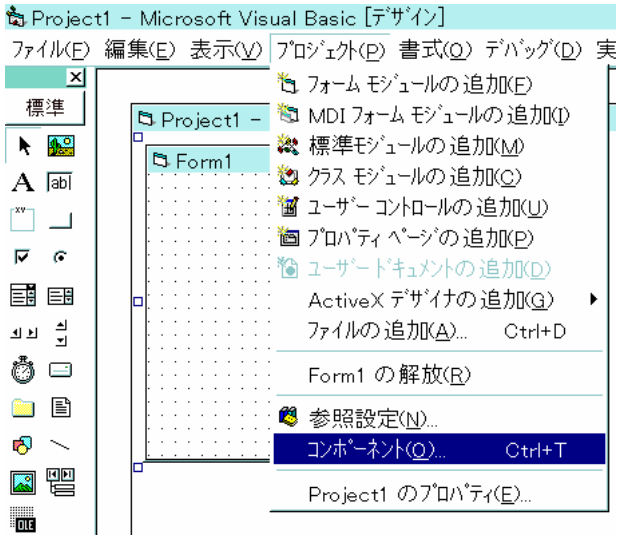
６．コンポーネントの追加

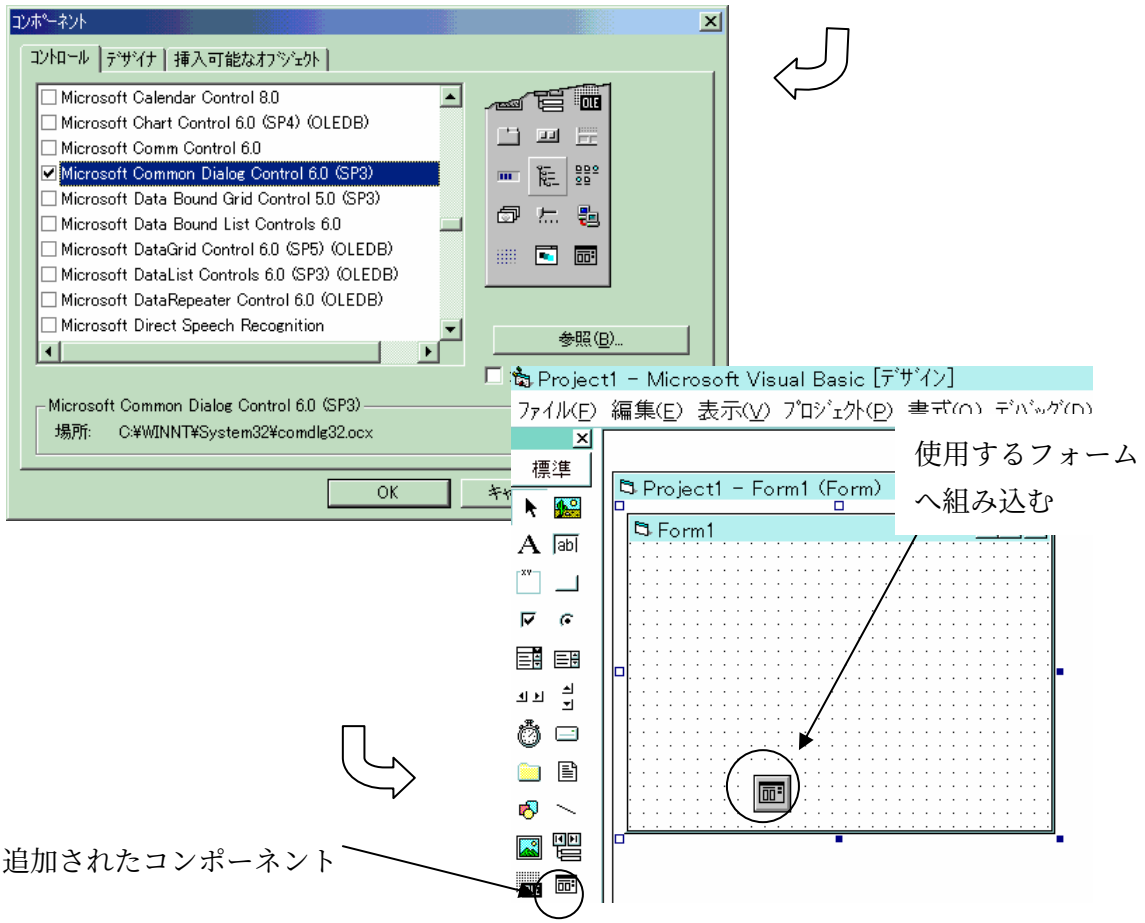
VB を使用すれば手軽にプログラムの作成ができるとは言え、すべての機能を自分で記述するのは大変です。ここでは予め作成されたコンポーネントの使用方法を紹介します。

6-1.Common Dialog

標準のコントロール以外の ActiveX コントロールを追加する場合はメニューバーより次の手順で行います。

[プロジェクト(P)]→[コンポーネント(O)]の順に選択しコンポーネントのウィンドウを表示します。組み込みたいコンポーネントのチェックボックスをチェックし OK のボタンを押します。ここでは Microsoft Common Dialog Control 6.0 を選択します。





Common Dialog ボックスを組み込み、印刷ダイヤログボックスを使用する例を示します。

印刷ダイヤログボックスは印刷時にプリンタの設定、選択を行うダイヤログボックスです。

印刷ダイヤログは ShowPrinter メソッドを呼び出し実行します。

ShowPrinter メソッドを実行すると、出力を行うプリンタの選択と設定ができます。
さらに次のプロパティに値が設定されて戻されます。

FromPage	開始ページ
ToPage	終了ページ
Copies	部数

右の例では FromPage、ToPage のプロパティは意味を持たないので無視します。

1 部だけ印刷する場合は、ShowPrinter メソッドを呼び出した後、PrintForm メソッドを呼び出し印刷します。印刷ダイヤログボックスで指定した部数印刷するためには右の例のように必要回数 PrinForm メソッドを実行します。

```
Private Sub Form_Load()
Dim x1, y1, x2, y2, r, rad
Dim i As Integer, j As Integer

AutoRedraw = True
Height = 4000
Width = 4000
Scale (-2000, -2000)-(2000, 2000)
r = 2000
rad = 3.1415926 / 180#

For i = 0 To 360 Step 30
   x1 = r * Cos(rad * i)
   y1 = r * Sin(rad * i)
   For j = i To 360 + i Step 30
      x2 = r * Cos(rad * j)
      y2 = r * Sin(rad * j)
      Line (x1, y1)-(x2, y2)
   Next
Next
End Sub
```

図形の描画

```
Private Sub print_Click()
Dim p_st, p_end, p_cp, i
   CommonDialog1.CancelError = True
   On Error GoTo ErrHandler

   CommonDialog1.ShowPrinter
   p_st = CommonDialog1.FromPage
   p_end = CommonDialog1.ToPage
   p_cp = CommonDialog1.Copies

   For i = 1 To p_cp
      PrintForm
   Next i
   Exit Sub

ErrHandler:
      Exit Sub
End Sub
```

フォームの印刷

```
Private Sub end_Click()
   End
End Sub
```

Copies プロパティに反映される

Common Dialog には ShowPrinter 以外に次のメソッドがあります。

ShowOpen	[ファイルを開く] ダイアログ ボックスが表示されます。
ShowSave	[ファイル名を付けて保存] ダイアログ ボックスが表示されます。
ShowColor	[色の設定] ダイアログ ボックスが表示されます。

ShowFont　　[フォントの指定] ダイアログ ボックスが表示されます。

ShowHelp　　Windows のヘルプ エンジンが呼び出されます。

詳細はヘルプを参照してください。

7．メニュー

メニューバー（プルダウンメニュー）は標準のコントロールに含まれていません。

フォームにメニューバーを作成し、メニューを設定する方法を示します。

PrintForm メソッドを使用してフォームを印刷する場合、メニューバーは印刷の対象となりません。

メニューの作成手順は次のようになります。

・ メニューバーの[ツール(T)]→[メニューエディタ(M)]の順に選択します。

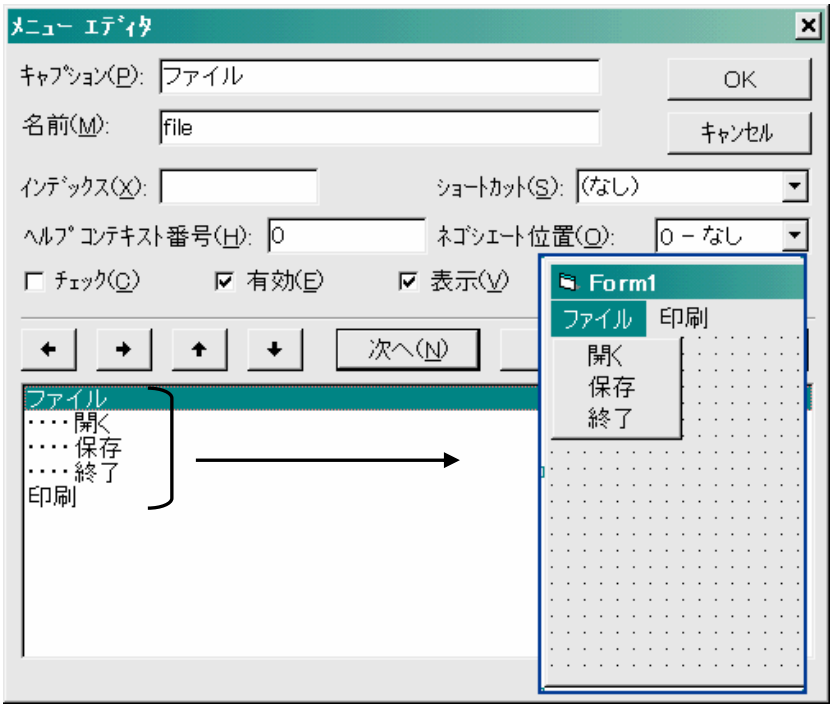
・ 「メニューエディタ」のウィンドウが表示されます。

キャプション(P)：作成するメニューの名称を指定します。

名前(M)　　：メニューオブジェクトの名前を指定します。

←　→　↑　↓　：メニューの階層構造を指定します。

イベントプロシージャを作成する場合は、フォーム上に作成したメニューを開き、該当項目をダブルクリックしてコードウィンドウを開きます。そして一般のコントロールと同様にコードを記述します。



８．付録

8-1.Visul Basic の文法

Visual Basic のプログラムの記述するに際に必要な最小限の項目について記載します。詳しくはマニュアル、ヘルプ等を参照してください。

記述方法

プログラムの記述は行単位に行います。基本的には１行に１つのステートメント（命令）を記述します。複数のステートメントをコロン":"で区切り 1 行に記述することもできます。

長いステートメントは、行継続文字 " _"(空白と下線の組み合わせ) を使用することにより複数行に分割できます。

コメント記号" ' "からその行の行末までの記述はコメントとして扱われ、プログラムの実行には影響を与えません。文字定数は文字列の最初と最後に ” を付けて表します。

例

```
Text1.Text = "Hello" : Red = 255                ←  複数ステートメント
Form1.Capture = _                               ←  行の分割
        "変数A,B,Cに対するデータの入力"
'これは画面の左端から始まるコメントです          ←  コメント
Text1.Text = "Hi!"        'メッセージの表示      ←  コメント
```

名前の命名規則

プロシージャ名、変数名、定数名などの命名規則は次のようになります。

・先頭は英文字、漢字、ひらがな、カタカナで始まり最大255文字の英数字、漢字、ひらがな、カタカナとアンダースコア _ で構成される文字列。

・ピリオド　.　や型宣言文字　％＆！＃＄ の記号は使用できない。

・予約キーワード(If、Loop 、Len 、Absなど)と同じ名前は使用できない。

演算子

算術演算子は右の表のものが使用できます。

例

$a \cdot b$　→　ａ＊ｂ

x^2　→　ｘ＾２

$\frac{(1-x)}{(1-y)}$ **→（１－ｘ）／（１－ｙ）**

演算子	機　能
^	数値のべき乗
＊	掛け算
／	除算
¥	除算を行い商の整数部分を返す
Mod	除算を行い剰余を返す
＋	和
－	差
&	文字列の連結

データの型

変数や定数、関数の戻り値はデータ型に従って保持できる値の種類、範囲が決まります。表にデータ型と保持できる値の範囲を示します。ヴァリアント型は最初に代入した値のデータ型を持ちます。

データ型			値の範囲
ヴァリアント		Variant	最初に代入した値の型の範囲
バイト	数値	Byte	0 ～ 255
整数		Integer	-32,768 ～ 32,767
長整数		Long	-2,147,483,648 ～ 2,147,483,647
単精度浮動小数点数		Single	-3.402823E38 ～ 3.402823E38
倍精度浮動小数点数		Double	-1.79769313486232E308 ～ 1.79769313486232E308
ブール		Boolean	True (真) または False (偽)
文字列		String	文字列の長さ 0 ～ 2GByte

変数の宣言

変数は Dim ステートメントによって宣言され、記憶領域が確保されます。Dim に続き変数名とデータ型を対にして指定します。データ型を省略した場合はヴァリアント型になります。

Dim 変数名 1 [As データ型] [, 変数名 2 [As データ型].....]

例

```
Dim x,y                 ←  x,y ヴァリアント型
Dim a,b As Interger     ←  a:ヴァリアント型,b:整数型
DIm message As String   ←  文字列型
```

8-2.ステートメント

エラー処理

On Error GoTo 行ラベル	行ラベルで指定した行から始まるエラー処理ルーチンを有効にします。
On Error Resume Next	実行時エラーが発生してもプログラムを中断せず、エラーが発生したステートメントの次のステートメントから実行を継続します。
On Error GoTo 0	現在のプロシージャに含まれる使用可能なエラー処理ルーチンを無効にします。

実行の終了・中断

End　　プログラムの実行を終了させます。

Stop　　実行を中断するフロー制御ステートメントです。

繰り返し

Do...Loop ステートメント

Do [While **条件式**]
　・
　ステートメント
　・
Loop

条件式が　真の間 Do..Loop 間で繰り返しを行う。
While 条件式　が省略された場合は無限ループとなる

For...Next ステートメント

For　*counter* = *start* To *end* [Step *step*]
　・
ステートメント
　・
Next

counter の値を *start* から *end* まで *step* で規定された増分を加え繰返す。
Step 以降を省略すると Step 1 となる。

繰り返しからの抜け出し

Exit For　　For...Next ループからの抜け出し

Exit do　　Do...Loop 構文からの抜け出し

条件分岐

If .... End IF ステートメント

形式１

If 条件式　Then
　ステートメント
End If

形式２

If 条件式　Then
　ステートメント１
Else
　ステートメント２
End If

形式３

If 条件式　Then
　ステートメント１
ElseIF 条件式　Then
　ステートメント２

論理演算子

演算子	機　能
And	2 つの式の論理積を求める。
Eqv	2 つの式の論理等価演算を行う
Imp	2 つの式の論理等価演算を行う
Not	式の論理否定を求める
Or	2 つの式の論理和を求める
Xor	2 つの式の排他的論理和を求める

比較演算子

演算子	関　係
＜	より小さい
＜＝	以下
＞	より大きい
＞＝	以上
＝	等しい
＜＞	等しくない

```
Else
        ステートメント 3
End If
```

プロシージャ

プロシージャには Sub、Function の２種類があります。この２つの違いは戻り値を持つか否かによります。Sub プロシージャは戻り値を持ちませんが、Function プロシージャは戻り値を持ちます。

Sub プロシージャ

Sub プロシージャの呼び出しは CALL ステートメントによって行われます。

Call ステートメント

```
Call プロシジャ名([引数[,引数...]])
```

Sub..Sub End ステートメント

```
[Private|Public] Sub プロシジャ名([引数[,引数...]])
        .
    ステートメント
End Sub
```

引数は、次の形式で指定します。
変数名 1 [As データ型]
[, 変数名 2 [As データ型],...]]

Function プロシージャ

Function プロシージャの呼び出しは文中にファンクション名を記述することによっておこなわれます。また

```
[Private|Public] Function ファンクション名([引数[,引数...]]) as データ型
        .
    ステートメント
        .
End Function
```

ファンクション名は戻り値を持ちます。
プロシージャ内でファンクション名に戻り値を代入する必要があります。

プロシージャ・ファンクションからの抜け出し

Exit Function	Function プロシージャの実行を止め呼び出し元へ戻る
Exit Sub	Sub プロシージャの実行を止め呼び出し元へ戻る

8-3.主な関数

以下に主な関数の一覧を表示します詳細はヘルプ内に VisualBasic のランゲージリファレンスを参照してください。

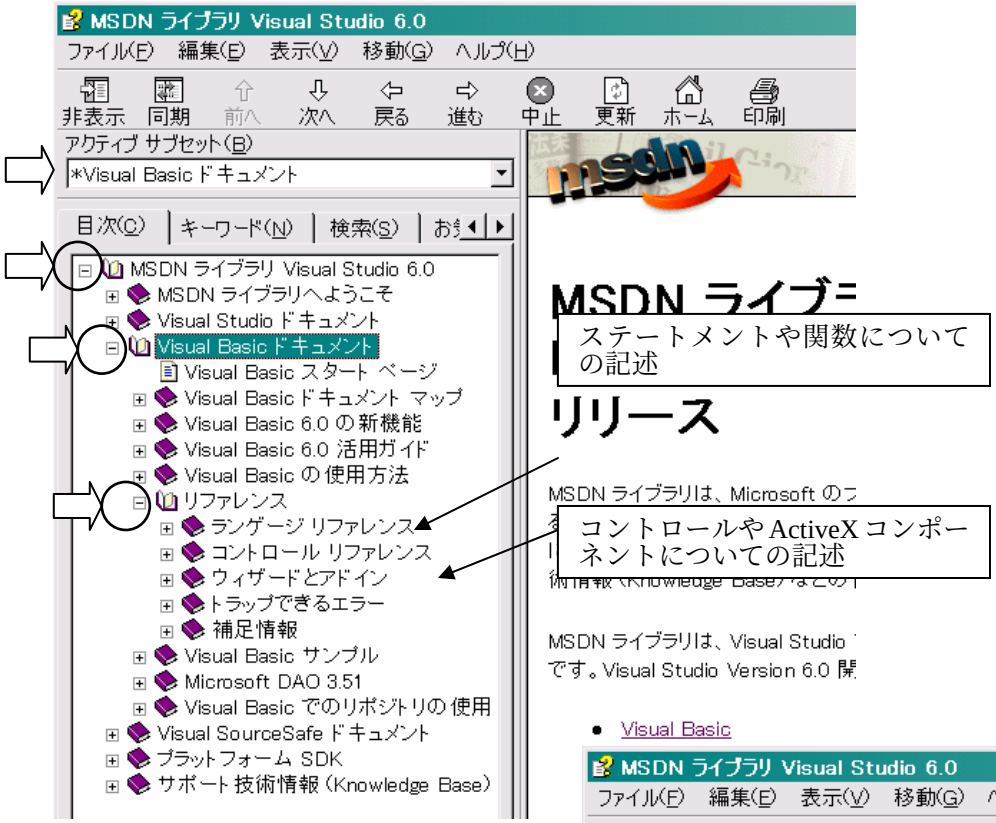
参照方法	機能
Str(*数値*)	数値を文字列に変換
Val(*文字列*)	数字からなる文字列を数値に変換
Format(*数値,書式*)	数値を書式に従って文字列に変換 書式　# 対応する桁が 0 の時は空白 0 対応する桁が 0 の時は 0 . そのまま出力
IsNumeric(*文字列*)	*文字列*が数値として評価できるかどうかを調べ、結果をブール型 (Boolean) で返す。
Abs(*数値*)	数値の絶対値をとる
Atn(*数値*)	三角関数 Atan の値を求める
Cos(*数値*)	三角関数 Cos の値を求める
Exp(*数値*)	e を底とする数式のべき乗を求める
Fix(*数値*)	*数値* の小数部分を取り除いた整数値を求める
Int(*数値*)	*数値* の小数部分を取り除いた整数値を求める
Log(*数値*)	e を底とする対数を求める
Rnd(*数値*)	0 以上、1 未満の乱数を求める。 *数値*< 0　常に、引数のシード値によって決まる同じ数値を求める。 *数値*> 0　乱数系列の次の乱数を求める。 *数値*= 0　直前に生成した乱数を求める。 省略時　乱数系列の次の乱数を求める。
Sin(*数値*)	三角関数 Sin の値を求める
Tan(*数値*)	三角関数 Tan の値を求める
InputBox(*ﾒｯｾｰｼﾞ[, ﾀｲﾄﾙ] [, 初期値] [,表示位置 X] [,表示位置 Y] [,helpfile, context]*))	ダイアログ ボックスに*ﾒｯｾｰｼﾞ*で指定された文字列とテキスト ボックスを表示し、文字列の入力を行う。
MsgBox(*ﾒｯｾｰｼﾞ[, buttons] [, ﾀｲﾄﾙ] [, helpfile, context]*)	ダイアログ ボックスに*ﾒｯｾｰｼﾞ*で指定した文字列を表示し、ボタンがクリックされるのを待って、どのボタンがクリックされたのかを示す値を返す。

8-4.ヘルプ

ステートメントや関数、コンポーネントの詳細、使用方法はヘルプで参照することができます。

・ **目次からの検索** （HELP(H) → **目次**(C)）

ヘルプを起動した後 アクティブサブセット欄に ［**Visual Basic ドキュメント**］ を選択し目次(C)のボタンを押します。 下の例のように、左側の欄の＋の部分をクリックしドキュメントを展開します。



・ **キーワードでの検索**

（HELP(H)→**キーワード**(I)）

ヘルプが起動した後、「アクティブサブセット(B)」の欄に **Visual Basic ドキュメント** を選択します。

次に、キーワード(N) のボタンを押し検索したいキーワードを「キーワードを入力してください(W)」の欄に入力します。

右の例では **グラフィックス メソッド** のキーワードで検索を実施しています。