

1. 準備

まず、何よりもはじめにマクロを使うためには Visual Basic を使う為の準備が必要である。そこで、「オフィス」ボタンから「Excel のオプション」を選び、「開発」タブをメニューバー(リボンメニュー)に登録、表示させる。



□「開発」タブをリボンに表示する、をチェックして「OK」をクリック。

メニューに「開発」タブが出ることを確認してみよう。

注意：エクセル 2007 以降からマクロを使用しているエクセルのファイル名は、保存するときその拡張子を.xlsxm にしなければならない。

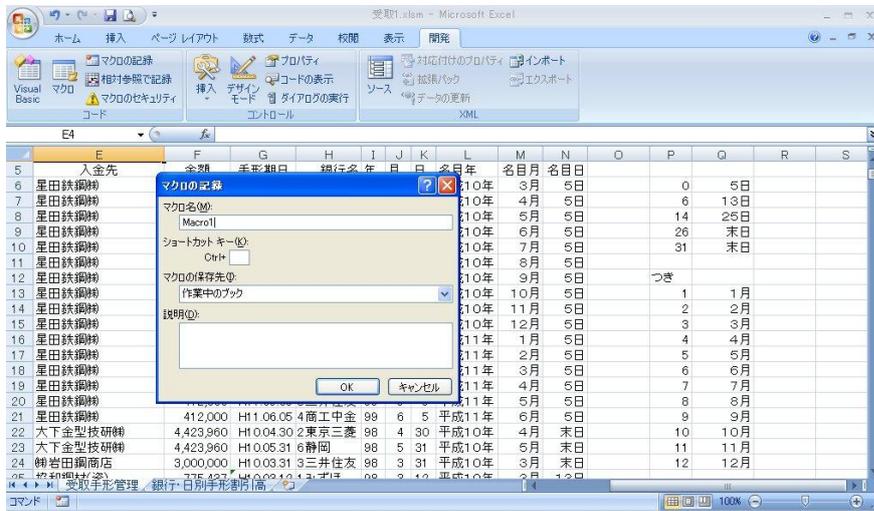
2. キーマクロの作り方

マクロプログラムは本来 Visual Basic 言語を勉強しなければ書けない言語であるが、エクセルにはロータス 1-2-3 と同様にキーマクロ、すなわち押して行ったキーの記録をとりそれを覚えさせ、後でそれを覚えさせた順に自動的に実行させるという機能を備えている。エクセルもロータス 1-2-3 も以前のバージョンでは押されたキーの記録をそのまま記憶させたが、最近のエクセルではその記録が Visual Basic で記述されるようになってきているので、エクセルのユーザーはキーを打ち込むだけで知らず知らずのうちに Visual Basic でマクロプログラムを書くことができるのである。

そこでまず、「開発」タブにおけるツールボタン群の中で左からはじめのいくつかを見てみよう。

- Visual Basic ... Visual Basic などの編集を行うウィンドウ(Visual Basic エディタとも言う)を開く。このウィンドウは画面上から消しても、点けても良く、プログラム上から無くなることはない、いつでも見たり、編集したりして再び消して、また点けることも出来る。
- マクロ ... これまで登録されているマクロを全て表示し、それらを見て、修正したり、実行したりすることが出来る。

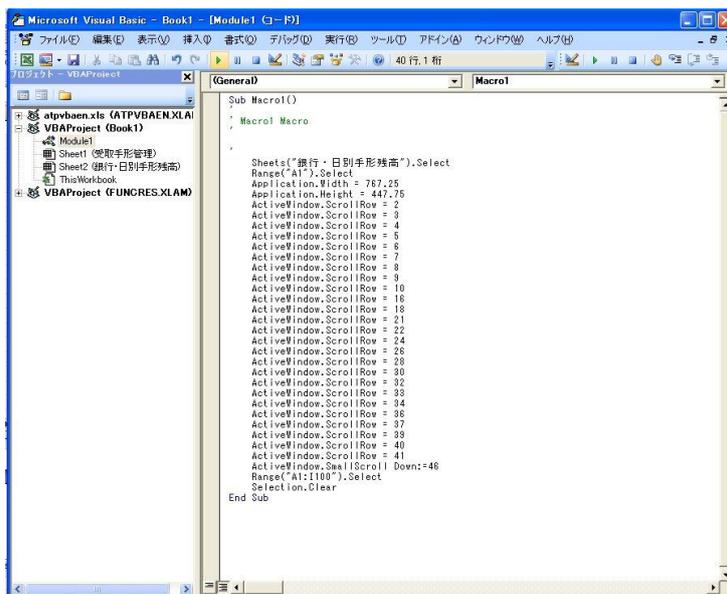
めである。ここでは「受取手形管理」というシートからマクロの記録をスタートさせる。ツールボタン「マクロの記録」をクリックするとダイアログボックスが現れて、マクロ名を Macro1 にするかどうか聞いてくるので、気に入らなければ自由にマクロ名を変えても良い。



オプションとして、出来あがったマクロの実行時にコントロールキー+アルファベットでマクロをスタートさせたいときはこの口にアルファベット1文字を入れることが出来る。マクロの保存先は作業中のブックで良いからそのままにする。説明もマクロの中でコメントが使えるのであえて書く必要が無いので、OK をクリックして記録をスタートさせる。記録は「記録終了」ボタンをクリックすると終わる。

マクロの記録をやり直すときは、マクロ名を同じにすると以前のキマクロを書きかえることができる。ただし、新しいマクロ用のシート(Module)がどんどん増えて行く可能性があるのも後で適当に編集するといふ。その方法などは次節で解説する。

3. マクロの確認と実行



キマクロが思ったように出来ているかどうか、出来あがった Visual Basic を読んだり、試しに実行

してみることが出来る。まず、マクロの内容を見るには「開発」タブで **Visual Basic** というツールボタンをクリックする。すると前ページの図のような **Visual Basic Editor** のウィンドウが開く。

図の左側がプロジェクトウィンドウでありこれを見ると、**Book1.xls** が **Sheet1**、**Sheet2** および **Module1**(標準 **Module**)からできていることが分かる。図の右側は **Module1** の内容を示している。右側に目的とするものが何も現れていないときは、**Module1** をダブルクリックすると、このような図が得られる。この内容がキーマクロの記録である。その内容を読むと、まずシート「銀行・日別手形残高」を選び、カーソルを **A1** へ持って行き、集計表の幅(**Width**)と長さ(**Height**)を指定、次々にスクロールして行って更に **46** 行下へ行って **A1:I100** という範囲をドラッグして、その領域を消して(クリア)いる。実はクリア作業は、スクロールなど無しにして、範囲を選んで消す作業だけで十分なのである。このケースでは、範囲をドラッグ(**Shift**+クリックを使用)するためにスクロールしたので、それが記録されてしまったのである。従ってもっとも少ない行数で書かれたキーマクロは

```
Sub Macro1()
```

```
    Sheets("銀行・日別手形残高").Select
```

```
    Range("A1:I100").Select
```

```
    Selection.Clear
```

```
End Sub
```

の **5** 行だけで十分なのである。

Module1 のシート上では勿論コピー&ペーストが可能で、上述したスクロール関係の命令を全て消して実行してみることが出来る。上の **5** 行だけにしてこの **5** 行の中にカーソルを置いて、**Microsoft Visual Basic** ウィンドウのツールボタンの実行「▶」(右三角)をクリックする。ちなみに、エラーの修正後のリセットには「□」、実行中の一時停止は「⏸」のボタンをクリックする。

マクロを実行する方法は幾通りものやり方がある。

- (1) 「開発」タブのツールボタンから「マクロ」のボタンをクリックしてマクロプログラムを選んで実行させる。
- (2) あらかじめ定義しておいた **CNTL**+アルファベットでマクロを起動する。
- (3) **Visual Basic** ウィンドウでカーソルを実行すべき **Module** の上に置いて実行キー▶をクリックする。
- (4) エクセルシート上に新たにボタンを作成して、そのボタンをクリックすることによりマクロを起動する。

等である。次節にエクセルのシート上にボタンを作成してマクロを起動する方法について述べる。

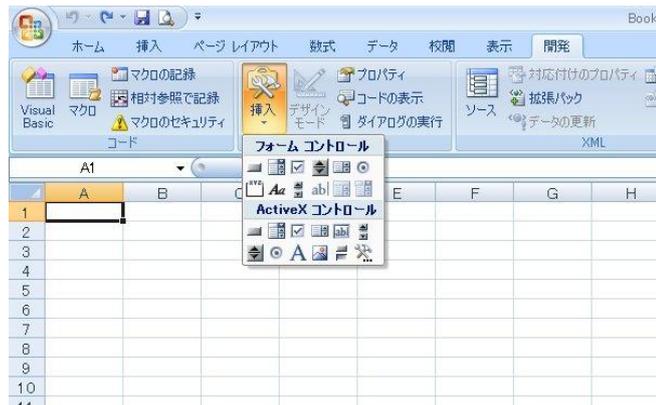
なお、マクロは標準 **Module** を挿入してそこに **Visual Basic** のプログラムを書けば、自動的にマクロプログラムとして登録され実行することが出来る。このことは、ここでマクロプログラムを修正すれば、マクロの内容やマクロ名を変更することが出来ることをも意味している。

マクロプログラムを記述する **Module**(標準 **Module**)を新たに追加するには、**Visual Basic** ウィンドウでのメニュー「挿入」から標準モジュールを選ぶか、新たにキーマクロを作成するとよい。そうすれば標準モジュールはどんどん増加する。見たい標準モジュールをダブルクリックすることによっていつでも好きなモジュールがウィンドウに表示できるし、それらをコピー&ペーストすることにより、適当に編集することが出来る。

そして標準モジュールを消すには標準 Module 上で右クリックをして、ショートカットメニューから「Module の解放」を選ぶ。消す前にエクスポートしておけば Module を保存することが出来る。Module をエクスポートしなければその Module は完全に消去されることになる。

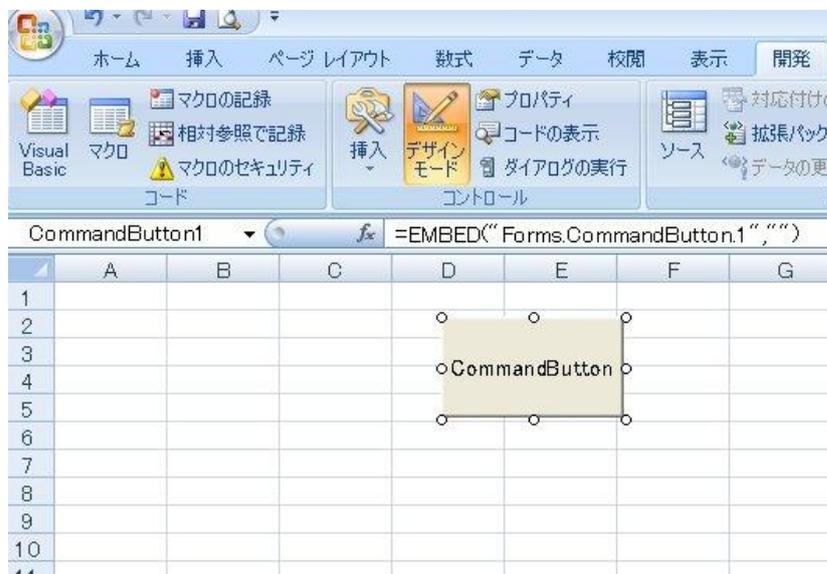
4. マクロボタンの作り方

「開発」タブで、挿入から「ActiveX コントロール」の中の、最上左上のボタンをクリックして、＋マークをドラッグすることにより適当な大きさのボタンボックスを描く。



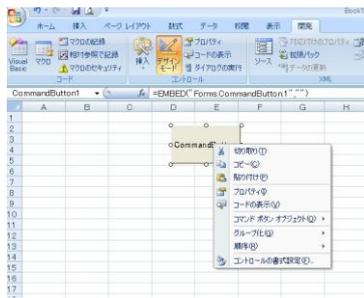
ボックスの大きさ、名前(CommandButton1 などの表記)、ボタンの色や文字の色と大きさ、フォントなどは後で自由に変更されるから、初めは本当に適当で良い。

出来あがったボックスの辺と四隅にある小さな 8 個の○印をドラッグすると、大きさが自由に変更出来る、＋印が出ているときにドラッグするとボタンの位置が変更出来る、ボタンを移動させることが出来る。



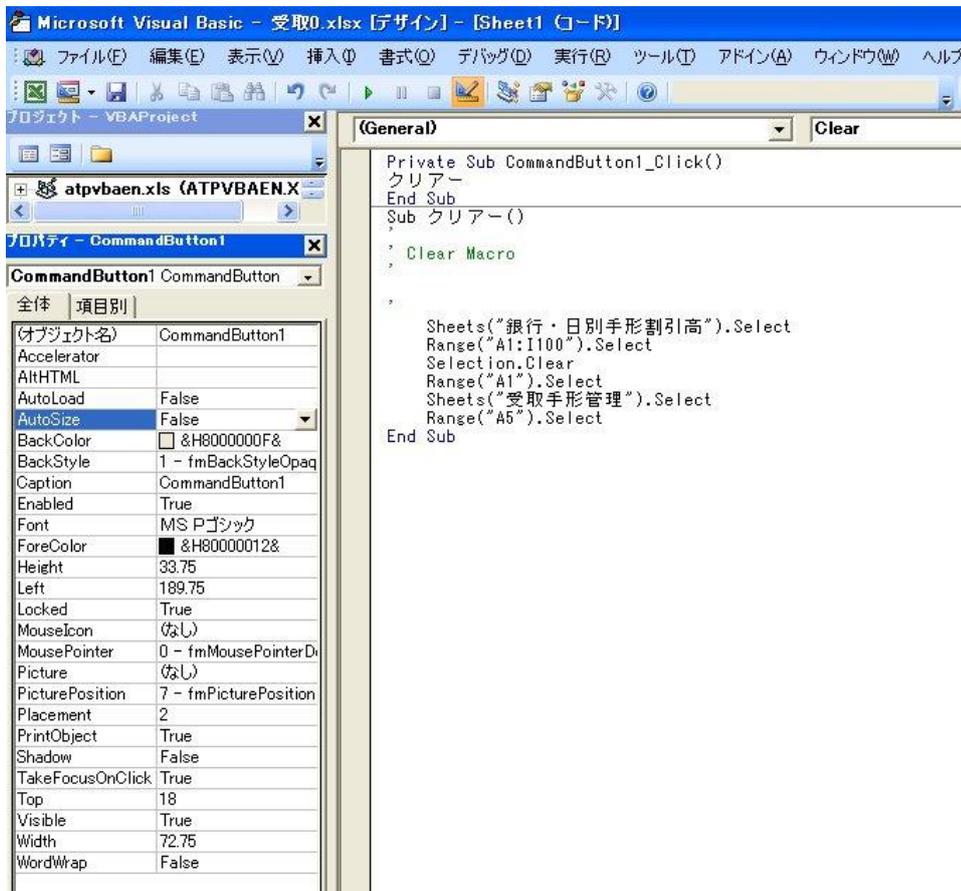
5. ボタンのプロパティ

ボタンの上にポインタを置いて右クリック(ショートカットメニュー)もしくはツールボタンからプロパティを選んでボタンのいろいろな内容を指定することが出来る。この節ではここで設定できるボタンのいくつかの設定項目についてのべることにする。



左の図はボタンのプロパティを設定するために、ショートカットメニューを表示させたところである。ここのメニューで「プロパティ」を選択すると下の図の左側ウィンドウが表示される。右ウィンドウは同じく「コードの表示」を選択して「クリアー」と **Caption** を指定し、かつクリアー作業をキーマクロで作成した結果を表示したものである。

右側に表示されているマクロプログラムは2つ登録されている。上のサブプログラムは出来あがったボタン(コモンボタン)をクリックすると「クリアー」と名付けられたプログラムがスタートするようになっている。「クリアー」という名のプログラムの内容は下のプログラムで、それを読んでみると、まずシート「銀行・日別手形割引残高」が選択され、次にセル範囲 A1~I100 が選択され、その範囲の内容を全てクリアーするというマクロが実行され、シート「受取手形管理」の A5 へカーソルが移動して終了するようにプログラムされている。このようにプログラムリスト等を画面に出すようにするのが、メニューの最左端にある **VisualBasic** というツールボタンである。



また先ほどのショートカットメニューのところで、「コードの表示」の代わりに「プロパティ」クリッ

クしてもプロパティ画面を表示することもできる。Visual Basic Editor のプロパティのツールボタン  をクリックした後、CommonButton を選択することによりプロパティ画面を表示することもできる。

ここで、プロパティ全ては分からないが、少なくとも必要と思われる重要な項目だけでも分かる範囲で述べておきたい（アルファベット順に並べている）。

BackColor ... ボタンの背景の色、&H800000F&

Caption ... ボタンの名前、クリアー

Font ... ボタンのフォント名とスタイルと大きさ、MS P ゴシック・標準・11ポイント

ForeColor ... ボタンの文字の色、&H80000012&

Height ... ボタンの縦の長さ、33.75

Left ... ボタンの左上のエクセルでの横座標の位置、189.75

Picture ... ボタンの中に絵を入れるか?、無し

PicturePosition ... 絵をボタン中のどの位置に入れるか、無し

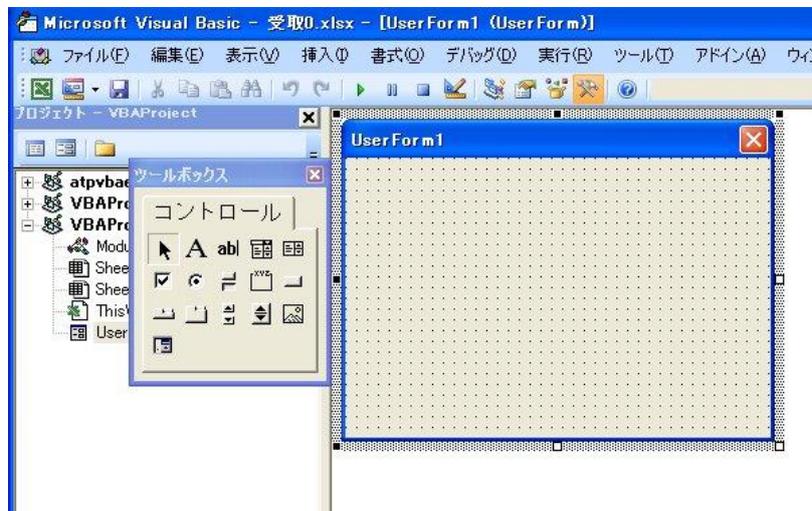
Shadow ... ボタンに影を付けるか?、無し(False)

Top ... ボタンの左上のエクセルでの縦座標の位置、18

Width ... ボタンの幅の長さ、72.25

6. その他

マクロを起動するだけではなく、より一般的にはボタンを更に拡張した、いろいろな種類のボタンやリスト、テーブルを加えたりすることが出来る「フォーム機能」を使った方法もある。



Visual Basic Editor 画面でのリボンメニューから「挿入」→「ユーザーフォーム」を選択すると、上のような図が得られ、左の「コントロール」から任意のボタンを選んだ後、右のフォームの任意の位置にドラッグすると、適当な大きさのコントロールボタンを作成することが出来る。このボタンにプロパティを設定し、コード(マクロすなわち Visual Basic)を定義することにより、上述してきたようなマクロプログラムを走らせることも出来る。これはJ言語にも用意されているフォームと全く同じで、これについては後日機会があったらチュートリアルセッションで述べてみたい。

2011年6月24日 竹内ハガネ商行 竹内寿一郎