

JのExcel_OLEを解析する

－オブジェクト指向からの理解－

西川 利男

はじめに

先月の例会で筆者は、Excel_VBAでWordのデータを取り出し、さらにJにより最終的にHTMLファイルを作り出すプログラムについて報告した。

筆者の究極の目標はJのOLE, DDE機能により、Windowsのいろいろなアプリケーションを自由に扱いたいということにある。そのための最初のテーマとしてExcelに対するOLEが最も適当なものと考えた。

幸い、JからExcelのデータを読み書きする処理については、志村正人氏、竹内寿一郎氏によるていねいな解説がある。[1, 2] しかしながら、OLEの操作の詳細については、ブラックボックスのようにして使えばよいとして、そのコーディングの意味づけについては明らかにされてはいない。今回、筆者はこれを元に少し修正し、コーディングの内部に入り、OLEとしてどのような処理を行っているのかを、理解したいと考えた。

Excel_VBAはいまでもなくVisual Basicのプログラミングであるが、データ構造、処理の理解の基本にはオブジェクト指向の考え方が必要であり、この面からも筆者なりのいささかのコメントをした。

オブジェクト指向とはデータと処理とを別々のものとしてではなく、両方を（これをプロパティ、メソッドと呼ぶが）備えたオブジェクトとして扱うことにある。[3]

オブジェクト	┌	プロパティ (データ)
		名前、構成要素、表示の方法など
		メソッド (処理、関数)

そして、このような機能をクラスという設計図と見て、これからインスタンスという製品をもらって、これについて処理を行うやり方である、と一口に言えよう。

以上のような理解と実験の過程を、Labシステムにより対話的にJを実行し、体験できるJのExcel_OLEの解析システム 'Analyse_Excel_OLE' につき紹介する。

[1] 志村正人「J for Win9x/NT 入門、EXCEL とのリンク他」J研究会資料1998/9/24

[2] 竹内寿一郎「Jの中でエクセル、エクセルの中でJを使う」JAPLA シンポジウム2000/12/16

[3] 西川利男「Jのオブジェクト指向プログラミングーその1」JAPLA シンポジウム2005/12/10

J の Excel_OLE の解析 (以下は Lab による対話出力を印刷したものである)

Lab: Analyse J to Excel OLE

Author: Toshio Nishikawa

Press <Ctrl+A> to advance.

-- (1 of 21) introduction -----

J の Excel_OLE とは

- J から Excel へのやり取りは次のような Client&Server 方式で行う。•

Client: J

Server: Excel

一般には Client&Server 方式とはあらかじめ Server を起動し、受けられる状態にした上で、Client から Server に処理を依頼する。

しかし、J では Client に特別な機能を持たせて、Server の起動がなくても Excel が動くようにした。これが J_Excel Automation である。•

その上で、J から Excel 上で動く VBA マクロ・プログラムを送り込み Excel 上でのいろいろな処理を可能にした。

- Excel の構造 Excel は次のような構造を持っている。これはオブジェクト指向から見れば、プロパティとメソッドから成るクラス構造と考えられる。

Application

Workbooks

Worksheets

Charts

DocumentsProperties

Names

--

- J の Excel_OLE 操作と xlutil 関数

基本的には J の OLE は以下のように

```
wd 'ole***'
```

ole で始まる命令の文字列を wd(Windows Driver)に渡すことでなされる。

しかしながら実際にはいろいろな操作を行うには次のJプログラム

```
examples¥ole¥excel¥xlutil.ijs
```

で定義されている xl で始まる各種の関数を用いて行う。

詳細はやがてあきらかになるが、これらの命令は常に parent の xlauto に対して行われるので psel (parent select) xlauto が先立つことになる。

xlutil のいろいろな定義はこのカヴァーリング (覆いかぶせ) を行うものである。たとえば xlget

```
xlget ARG は wd 'psel xlauto;oleget xl',ARG
```

のように、内部では oleget の処理を行う。

なお、以下の3つの命令が重要である。

```
xlget 値、オブジェクトなどを得る
```

```
xlset プロパティに値をセットする
```

```
xlcmd メソッドを実行する
```

)

```
-- (2 of 21) prepare -----
```

まず準備として excel utitily を読み込む。

)

```
require 'examples¥ole¥excel¥xlutil.js'
```

```
-- (3 of 21) prepare (continued) -----
```

最初に Excel OLE Automation のオブジェクトの作成を行う。

このコーディングの意味は次の通りである。

pc (parent create) により、parent window として xlauto を定義し、

cc (child create) により、child control として xl を定義する。

オブジェクト指向で考えれば、クラス Excel.Application のインスタンス base がデフォルトとして作られる。

その名前をチェックしておく。

)

```
wd 'pc xlauto'
```

```
wd 'cc xl oleautomation:excel.application'
```

```
xlget 'base name'
```

Microsoft Excel

```
-- (4 of 21) prepare (continued) -----
```

この状態を Excel 画面として見てみる。

Alt-F11 により VBA-Editor をチェックせよ。まだ何もない。

Excel から戻るときは、最小化中断とすること。

なお、このコードは次の意味である。

オブジェクト base のプロパティ visible に値 1 をセットする。

これによって Excel のすべての画面が表示可能になる。

ふつうは

```
xlshow ''
```

として行う。

)

```
wd 'psel xlauto;oleset xl base visible 1'
```

```
-- (5 of 21) prepare (continued) -----
```

クラス workbooks のインスタンス wb を作成する。

)

```
xlget 'base workbooks'
```

```
xlid 'wb'
```

```
-- (6 of 21) prepare (continued) -----
```

ここで、JMACROS.xls なるマクロだけの Excel をオープンする。

つまり、J のシステムにある Excel に VBA のマクロ・プログラムを送りこむ。

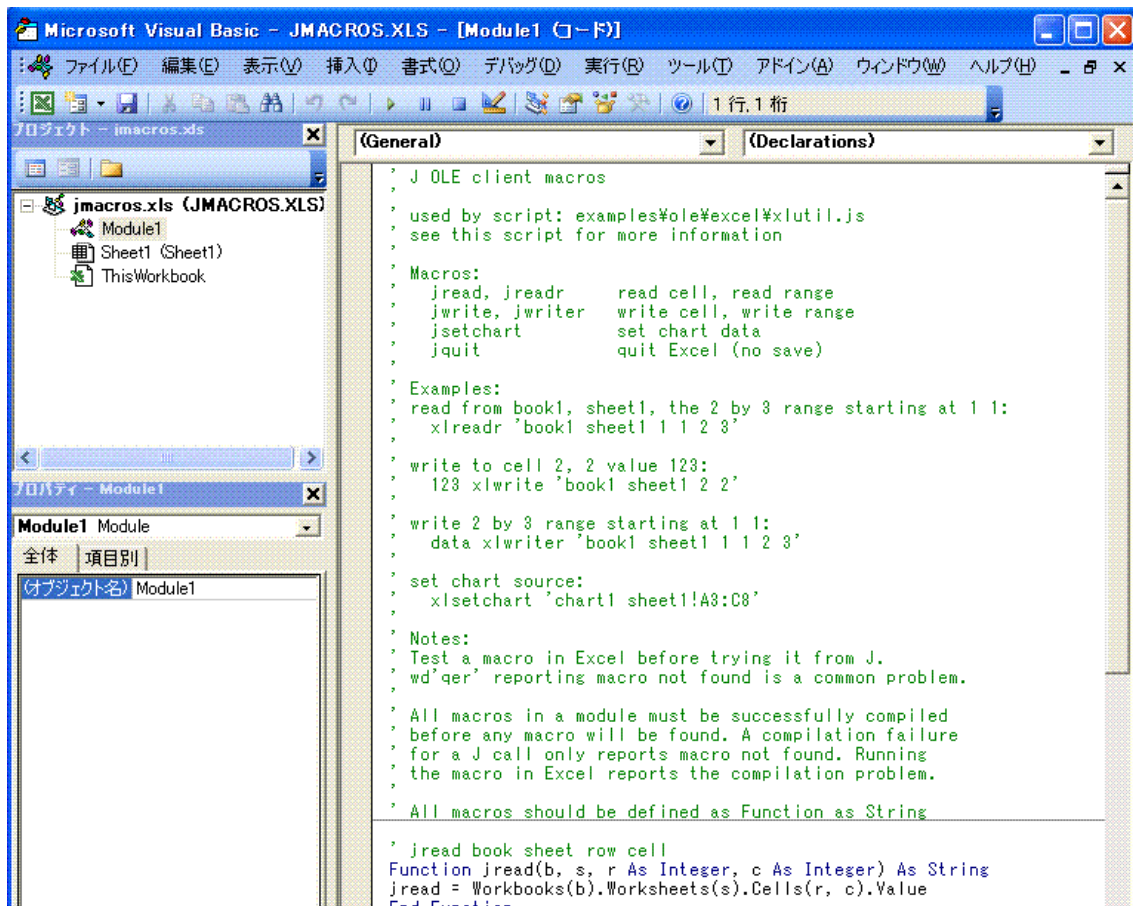
)

```
xlcmd 'wb open ', JMACROS
```

```
-- (7 of 21) prepare (continued) -----
```

この状態の Excel 画面を見てみる。

Alt-F11 により VBA-Editor をチェックせよ。



jmacro.xls がロードされ、module1 の VBA プログラムを確認せよ。

Excel から戻るときは、最小中断とすること。

ここまでの操作はふつうは

```
xlopen ''
```

としてまとめて行われる。

)

```
xlshow ''
```

-- (8 of 21) prepare (continued) -----

空の Excel に追加により、新規にブック wa を作る。

オブジェクト wa の名前をチェックする。

)

```
xlcmd 'wb add'
```

```
xlid 'wa'
```

```

    xlget 'wa name'
Book1

-- (9 of 21) prepare (continued) -----
そしてExcel 画面を試みる。
)
    xlshow ''

-- (10 of 21) prepare (continued) -----
wa をファイル"TEST.xls"に書き出す。
)
    xlcmd 'wa saveas "TEST.xls"'
-1

-- (11 of 21) prepare (continued) -----
この状態のExcel 画面を試みる。
)
    xlshow ''

-- (12 of 21) prepare (continued) -----
wa のプロパティ worksheets を取り出し、オブジェクト ws とする。
)
    xlget 'wa worksheets'
    xlid 'ws'

-- (13 of 21) prepare (continued) -----
ws のプロパティ item, sheet1 をとり、オブジェクト sh1 とする。

この名前をチェックする。
)
    xlget 'ws item sheet1'
    xlid 'sh1'

    xlget 'sh1 name'
Sheet1

```

-- (14 of 21) prepare (continued) -----

オブジェクト sh1 の名前を Sheet1 から TRY に替える。

そして、オブジェクト sh1 をアクティブにする。

また、Excel 画面でチェックする。

)

```
xlset 'sh1 name TRY'
```

```
xlget 'sh1 name'
```

TRY

```
xlcmd 'sh1 activate'
```

-1

```
xlshow ''
```

-- (15 of 21) prepare (continued) -----

ここで、現在アクティブのブック、シートの名前を確認する。

)

```
xlwbws ''
```

```
+-----+-----+
```

```
|TEST. xls|TRY|
```

```
+-----+-----+
```

-- (16 of 21) calc -----

さてここで、J と Excel とでやり取りするデータをつくる。

)

```
Dat0 =: i. 3 4
```

```
Dat0
```

```
0 1 2 3
```

```
4 5 6 7
```

```
8 9 10 11
```

-- (17 of 21) calc (continued) -----

この値を Excel のデータとして、ファイル

TEST.xls

のシート

TRY

のセル(A1)、つまり1行、1列から

1 1

として書き込む。

xlwriterは範囲(range)を指定しての書き込みである。

)

```
Dat0 xlwriter 'TEST.xls TRY 1 1'
```

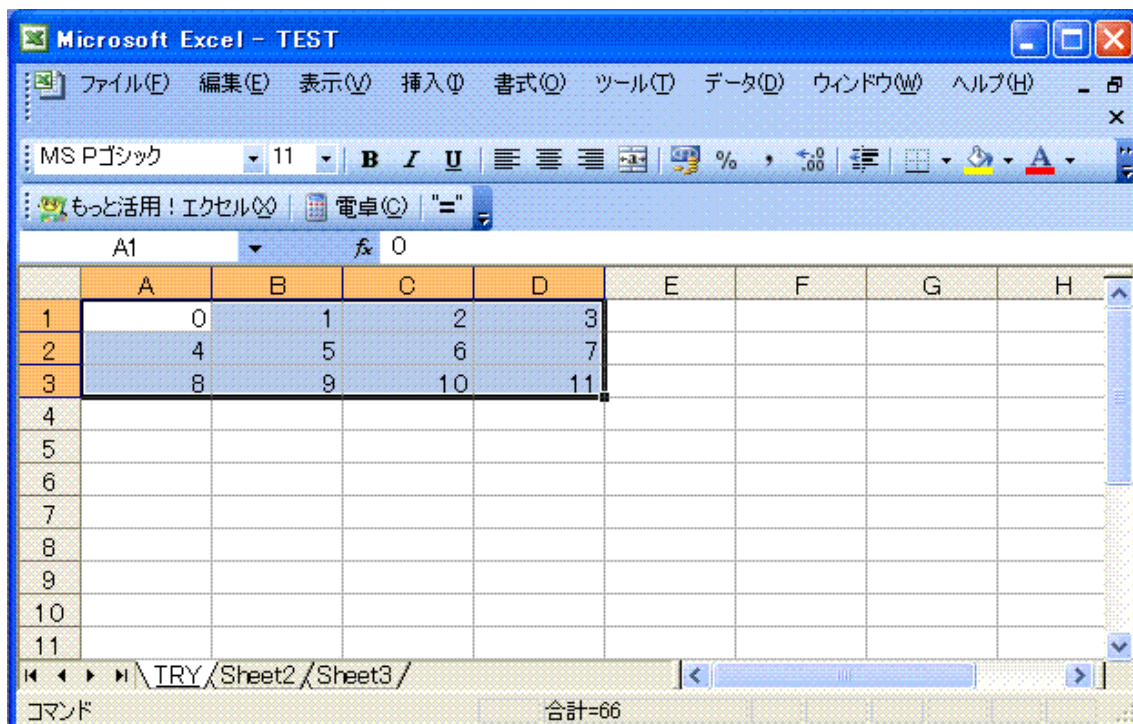
1

-- (18 of 21) calc (continued) -----

この状態のExcel画面をしてみる。

)

```
xlshow ''
```



-- (19 of 21) calc (continued) -----

次にこのExcelデータの

セルA2 つまり行2, 列1

の位置から

2行、3列

の値を読み出し、Jの値にする。

xlreadr は範囲(range)を指定しての読み出しである。

)

```
Dat1 =: xlreadr 'TEST.xls TRY 2 1 2 3'
```

```
Dat1
```

```
++++++
```

```
|4|5|6 |
```

```
++++++
```

```
|8|9|10|
```

```
++++++
```

```
-- (20 of 21) calc (continued) -----
```

最後に Excel 画面をチェックする。

今度は Excel を終了して、閉じる。

)

```
xlshow ''
```

```
-- (21 of 21) fin -----
```

最後に Excel_OLE を終了する。

)

```
xlquit ''
```

End of lab