JAPLA 研究会資料 2007/2/24

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

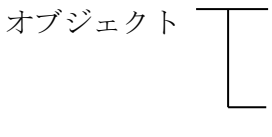
先月の例会で筆者は、Excel_VBA で Word のデータを取り出し、さらに J により最終的に HTML ファイルを作り出すプログラムについて報告した。

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 はいうまでもなく 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]



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みれば、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xlutily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'
```

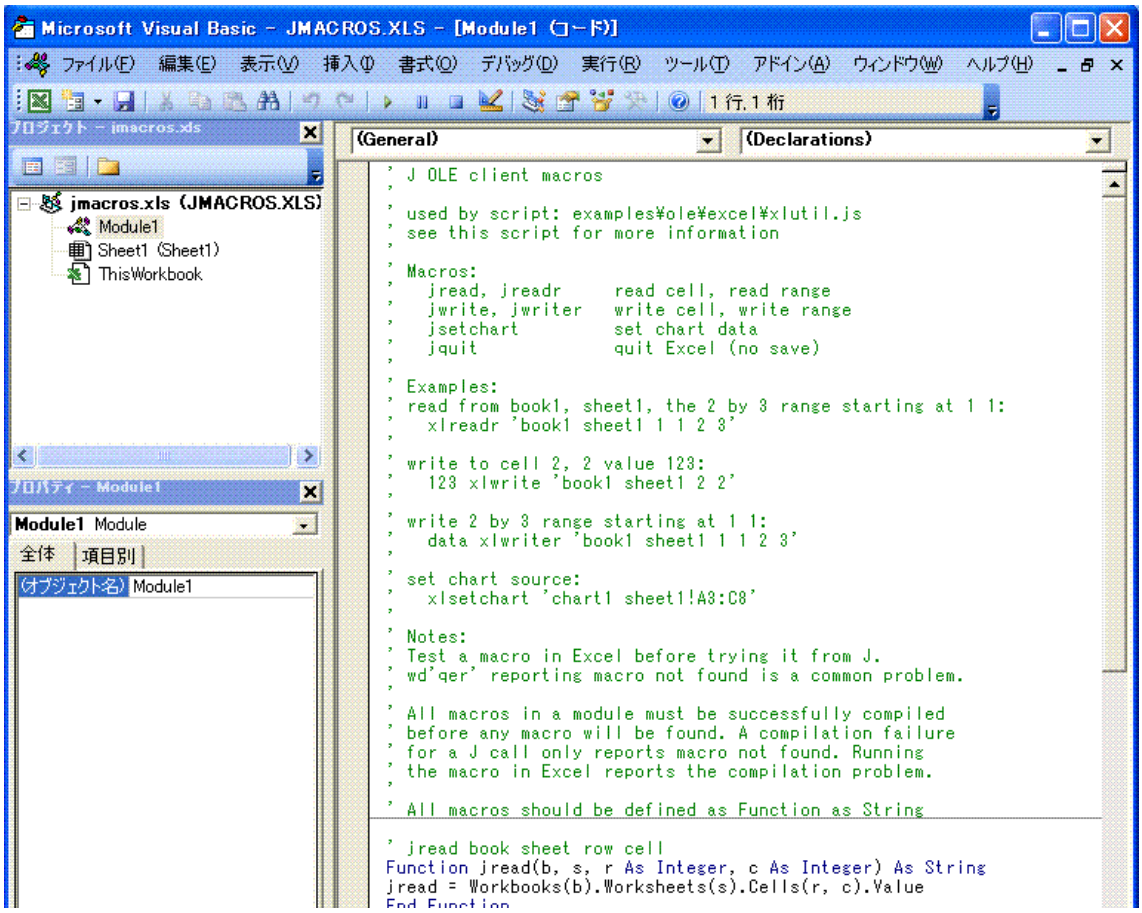
```
   xlget 'base name'
Microsoft Excel

-- (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画面を見てみる。
)
   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画面を見てみる。
)
   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
```

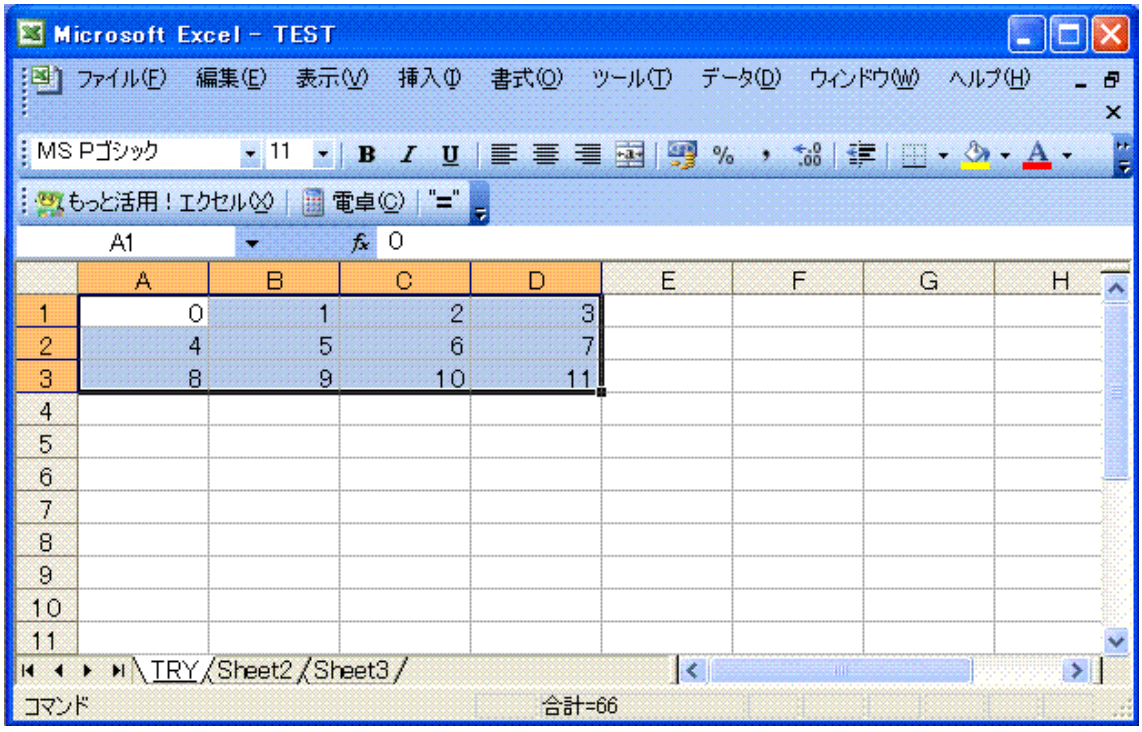
```
-- (14 of 21) prepare (continued) ----------------------------
```
オブジェクト sh1 の名前を Sheet1 から TRY に替える。
そして、オブジェクト sh1 をアクティブにする。

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 画面を見てみる。
)
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