JAPLA 研究会資料 2013/10/18

Grid をしくみから学ぶプログラミング
パズル Number Link のグラフィックスを例として

西川　利男

はじめに－Number Link パズルとは

Number Link パズルは、ペンシルパズル、つまり「鉛筆と消しゴムのパズル」として、ニコリ社より考案され、数独と並んで世界中に多くの愛好者がいる。

ルールは簡単である。つぎのように「盤面上の同じ数字を互いに交わらないように、かつ隙間のないように、どうやって結ぶか」というパズルである。

```
+-+-+-+-+-+-+-+-+-+
| | | | |1|5| | |
+-+-+-+-+-+-+-+-+-+
| | | | | | | | | |
+-+-+-+-+-+-+-+-+-+
| |3| | | | |3| |
+-+-+-+-+-+-+-+-+-+
| | | | |2| | | |
+-+-+-+-+-+-+-+-+-+
|4| | | | | | |4|
+-+-+-+-+-+-+-+-+-+
| | | | | |5| | |
+-+-+-+-+-+-+-+-+-+
| |1| | | | | | |
+-+-+-+-+-+-+-+-+-+
| | | | | | | |2|
+-+-+-+-+-+-+-+-+-+
```

最初は、コンピュータでアルゴリズムに従って自動的に解が得られるのでは、というつもりであったが、やがてこれがとんでもない深いテーマであることがわかった。Mellow Melon による攻略法の入門[1]をみつけたが、当面は頭をひねってやるしかない。

かえって、「鉛筆と消しゴム」で行う代わりに、コンピュータ上でユーザインターフェースときれいなグラフィックスを J の Grid の適用例として、やってみることにした。

さらに J の Grid のしくみを基本的なところから、私自身納得いくように理解するというテーマとした。

[1] MellowMelon, "A NumberLink Solving Primer",
http://mellowmelon.wordpress.com/2010/07/24/numberlink-primer/

1.Grid の基本のしくみ

J には Grid という EXCEL に相当する便利なプログラミングツールがある。

Grid（EXCEL でも）の最大の特徴は、

「画面上のセルに値を格納し、その位置を指定して値を自由に処理できる」

ことにある。

まずは、これで作ったカレンダーの表示をあたまに描いてほしい。

しかし、J のプログラムでちょっとやれば、次のように簡単に行える。

```
   DAT =: (5 7)$((4#_), >:i.31)
   DAT
_ _ _ _  1  2  3
 4  5  6  7  8  9 10
11 12 13 14 15 16 17
18 19 20 21 22 23 24
25 26 27 28 29 30 31
```

日付の値を見るには、例えばつぎのようにする。

```
   (<1 2) { DAT
6
```

この値を、たとえば 99 に替えたければ、つぎのようにすればよい。

```
   99 (<1 2) } DAT
_ _ _ _  1  2  3
 4  5 99  7  8  9 10
11 12 13 14 15 16 17
18 19 20 21 22 23 24
25 26 27 28 29 30 31
```

そもそも Grid とは画面上のグラフィックスであり、よく考えてみると非常にふしぎである。つまり、ディスプレー画面の上にあるのは、実際には黒と白とのピクセルの集まりでしかない。しかし、マウスでクリックしたセルからはたとえば 6 という値が引出せる。これはどうしてだろうか。……まさか、最新の人工知能、パターン認識技術の成果であろうか。

実は、楽屋裏では、次のようなことを行うことで、これは可能となる。

① まず、あらかじめ配列のデータ DAT を作っておく。

② マウスクリックなどで画面上のピクセルとしての位置情報を得る。

③ グリッドの幅と高さを元にセルの行(R)と列(C)を算出する。

4 セルの値は R と C との値を元に (<R, C) { DAT として得られる。

5 また、変更するのであれば、DAT =: Value (<R, C) } DAT　とする。

6　この値を Grid の R, C の位置のセルにピクセルとして書き込み表示する。
このようなからくりが Grid のシステムに他ならないのである。

上のしくみを理解した上で、Grid 構築のプログラミングを説明していこう。

2.Grid の構築と OOP（オブジェクト指向）ー 私のアプローチの方法

ところで、簡単に Grid を利用するのであれば、J に以下のようなクラスツールが備わっていて、これを利用するのがよい。

jzgrid, jwgrid, jwatch, ...

以前、報告した Grid による数独グラフィックスはこのようにして構築した[2]。

[2] 西川利男「J のオブジェクト指向プログラミングーその4
ーJ-Grid による数独パズルをもっと使いやすくー」JAPLA 研究会資料　2006/3/25

ここでは、多くの機能が OOP（オブジェクト指向）を利用することで行われるので、それについての知識が必要である。しかし、初めから自分でプログラミングしようとするとき、問題となるのは、クラスの階層が深くなり、また複雑なコードで理解が難しく、ブラックボックスとなってしまっていることである。

たとえば、デフォルトで決まっているセルの大きさを変えたり、マウスでの入出力など、やや高度の処理を行うには Grid を基本から理解する必要がある。

J の Lab に Grid の Low Level Programming というチュートリアルがあるが、これを読み解くには少々忍耐がいる。

今回のプログラムでは、grid 操作のプリミティブ gl2 のみを用いて行うようにした。

3.J で Grid を表示するまでー基本の初期設定

3.1　Jのクラス・スクリプトとフォーム作成

J の grid プログラムは必ずしも OOP つまりクラスとして作ることは必須ではないが、grid は J のグラフィックスであるので、フォームは作らなくてはならない。

ふつうは、[File]の[New Class]をクリックして、そのウィザードに従って行う。

① p から始まる名前でクラス・ファイルを作る。例えば pnumberlink.ijs とする。
② Form Class にチェックを入れる。
③ Select Template では empty.ijs を選ぶ。
④ ここまで来て、next とすれば、自動的にクラスファイルとそれをチェックする実行スクリプトが現れる。これで最初のスタートは OK である。

なお、この一連の操作はウィザードによって自動的に行われるが、手動では COPATH の設定がなされないので、うまくいかない。

このようにして、作られたクラスファイルと実行スクリプトは以下のとおりである。

クラスファイル

```
coclass 'pnumberlink'

PNUMBERLINK=: 0 : 0
pc pnumberlink;
rem form end;
)
create=: 3 : 0
wd PNUMBERLINK
formhwnd=: wd'qhwndp'
NB. initialize form here
wd'pshow'
)
destroy=: 3 : 0
wd 'pclose'
codestroy''
)
pnumberlink_close=:destroy

formselect=: 3 : 'wd''psel '',formhwnd'
```

実行スクリプト

```
  object =: '' conew 'pnumberlink'
```

すなわち、実行スクリプトを上のとおりに打ちこむと、conew によりクラスファイル pnumberlink のインスタンス object が作られ、これがクラスの create により実行されて、画面が表示される。

クラスファイルの内容を見てみる。

第1行の coclass の宣言が大切。これによりクラスとして参照される。

名詞 PNUMBERLINK の定義はフォームの指定、今は何もない。

動詞 create はインスタンス実行時に自動的に実行される。つまり、フォーム PNUMBERLINK が画面として表示される。

動詞 formselect の formhwnd にはフォームの名前が入れられる。動詞 pnumberlink_close は Parent pnumberlink の Child_close であり、実際は、画面の右上隅の x ボタンを押すと閉じられる機能である。

3.2　Grid のためのフォームの追加

これからは、手作業になる。クラスファイル pnumberlink の画面上で、[Edit]-[Form Editor]により必要なフォームの作成などを行う。

①　いくつかのボタンの作成（例えば、Run, Exit, Init など）を行う。必要ならばボタンの上でクリックして code の作成も行う。

②　必ずなくてはならないのが、grid 表示のためのグラフィックスで、これを isigraph として作成する。名前は任意でよいが、ふつう grid としたほうがよいだろう。

3.3　Grid への基本の設定

標準的な使用には、jwgrid ツールなどにある init_jwgrid で体裁の整ったフォームが作られるが、これでは、grid の基本のしくみや使い方は分からない。

ここでは、Init ボタンの実行プログラムとして、gl2 プリミティブを使って定義した。

```
NB. select the object for command execution == modified by TN
sel =: 3 : 0
 wd 'psel ', 'pnumberlink'  NB. select parent form id
 glsel 'grid'               NB. select child 'grid' id
)
NB.       size       of       grid       height       and       grid       width
=========================================
gridhs =: 40  NB. height of cell
gridws =: 40  NB. width of cell
NB.            convert            J            value            to            celldata
================================================
celldata =: 3 : ';(": each y.), each 0{a.'
NB.                grid                initial                start                program
================================================
pnumberlink_Init_button=: 3 : 0
sel ''
glclear ''
glmapraw ''         NB. initiate graphics map mode
glmap MM_TEXT       NB. change pixel (x, y) left_up down to right_bottom
glnoerasebkgnd 1    NB. white screen
glgrid ''           NB. draw default grid line
glgridfill gridws, gridhs, 0, 0  NB.  NB. draw grid line with gridws & gridhs
glgridrc 6 4        NB. make row(6) x column(4)
glgridh  6$gridhs   NB. make 6 cells with grid_height
```

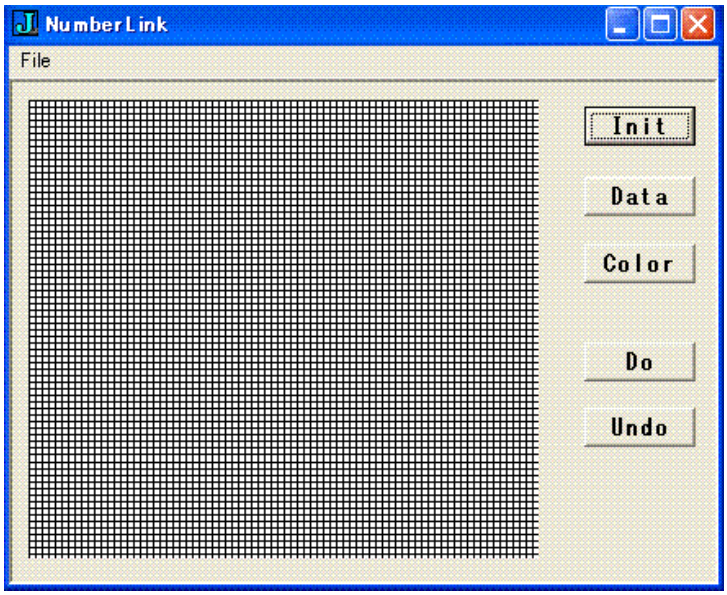
```
glgridw  4$gridws   NB. make 4 cells with grid_width
NB. glgridfill 15 10 0 0
glgridtext celldata i.6 4      NB. write string converted J value
DR =: 11 0 0 0                 NB. border along side values
border =: 0 0 6 4, 16$DR
glgridborder border
glshow ''
)
```

4.Grid の基本画面の実行

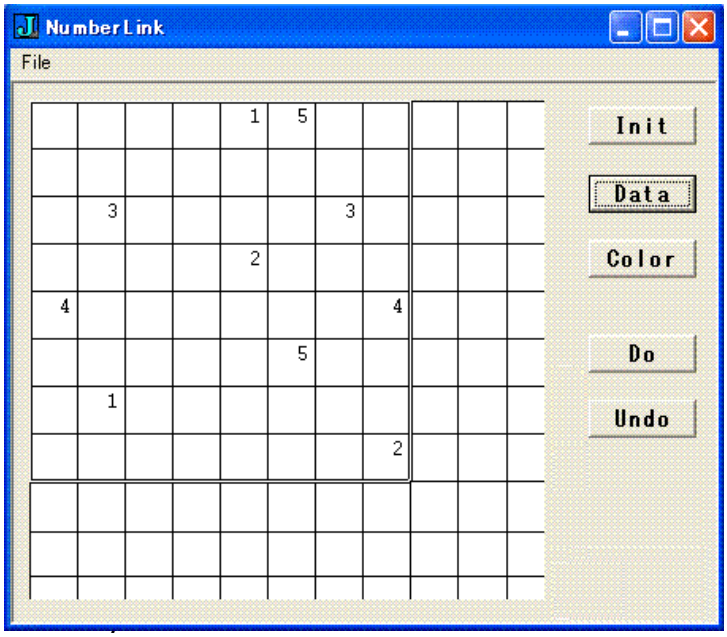
これまでの J Grid のプログラムを Number Link のデータ DB を指定して

```
run 'DB'
```

として実行してみる。



まず、白い画面が現れるので、右上の[Init]ボタンを押すと、つぎのような基本のピクセルサイズで Grid のデフォルト画面が表示される。



続いて、[Data]ボタンを押すとプログラムで指定した

grid の高さ、gridhs

grid の幅、gridws

の値に従った grid 画面に変わり、

各セル上には、Number Link のデータの値 DB が表示される。

これで、一応の grid 画面ができた。

5.Grid による Number Link の実行

以上のようにしてで出来た Grid のセルのそれぞれの値に対応した色を付けて、さらにマウスの操作でこれを連結して、Number Link をコンピュータ画面上で楽しむようにしよう。

5.1　セルへの色づけ＝属性の指定

セルの色づけは、セルの属性を指定することによってなされる。
セルの属性は、基本的には次の項目で行う。

①　色名の指示

```
    WHITE =: 255 255 255
  BLACK =:   0   0   0
  RED   =: 255   0   0
  GREEN =:   0 255   0
  BLUE  =:   0   0 255
   ------
```

② atts (属性値)の指定

```
  atts =: 3 : 0
  j =.    <WHITE,  WHITE,  BLACK, 1  2  _1
  j =.  j, <WHITE,  RED,   BLACK, 0 _1   1
  j =.  j, <WHITE,  GREEN, BLACK, 0 _1   1
  j =.  j, <WHITE,  BLUE,  BLACK, 0 _1   1
  20&{. &> j
  )
```

セルの枠、セル面、文字についての色を指定する。 それに続く値は文字のフォント属性（0(glgridfont0）, 1(glgridfont1)として選べる)、align（2　中央、_1　右寄せ）、edit、unused(8)である。セル1つ当たり20バイトで構成する。

③ 属性（色指定）の実行

```
     setcolor =: 3 : 0
  glgridatt ,atts''                NB. 属性指定値を読み込む
  glgridrchw 1, 1, Row, Col    NB. Row(行) Col(列)に1セル指定する
  glgridtype TXT               NB. データを文字として書く
```

実際の Number Link の Grid 表示では、Number Link の値 DB に応じて、色名を対応させ、それにより、セルの色づけを行った。

NB. Coloring ==========

```
pnumberlink_Color_button=: 3 : 0
sel ''
glgridfont0 'arial 18 bold'
setinitcolor ''
setborder ''
)

setinitcolor =: 3 : 0
glgridatt ,atts''
glgridrchw 0, 0, $DAT
glgridtype 3 + , DAT
glpaintx ''
)
```

5.2　マウスによる連結操作

　連結には、まずマウス右ボタンで開始セルを指定し、左ボタンで終了セルを決める。
以上の設定をした後、[Do]ボタンを押すと、連結したセルにすべて値を書き込み、かつ色を塗る。、やり直しには[Undo]ボタンで直前の状態に復帰できる。

　J Grid のコーディングの詳細は稿末を見ていただくが、基本的には、マウスボタンにより、画面の位置から、セル位置を算出し、それに従って配列データより修正データを求め、グリッド画面への変更、つまり値の表示と色づけをおこない表示を更新する。

```
NB. Mouse Right Down - Set Start Cell ========================
pnumberlink_grid_mbrdown =: 3 : 0
'R C Value' =: nmouse ''
R0 =: R
C0 =: C
NB. border double line
DR =. 11 0 0 0
Border =. R0, C0, 1, 1, 16$DR
glgridborder Border
glpaintx ''
)

NB. Mouse Left Down - Set End Cell ============================
```

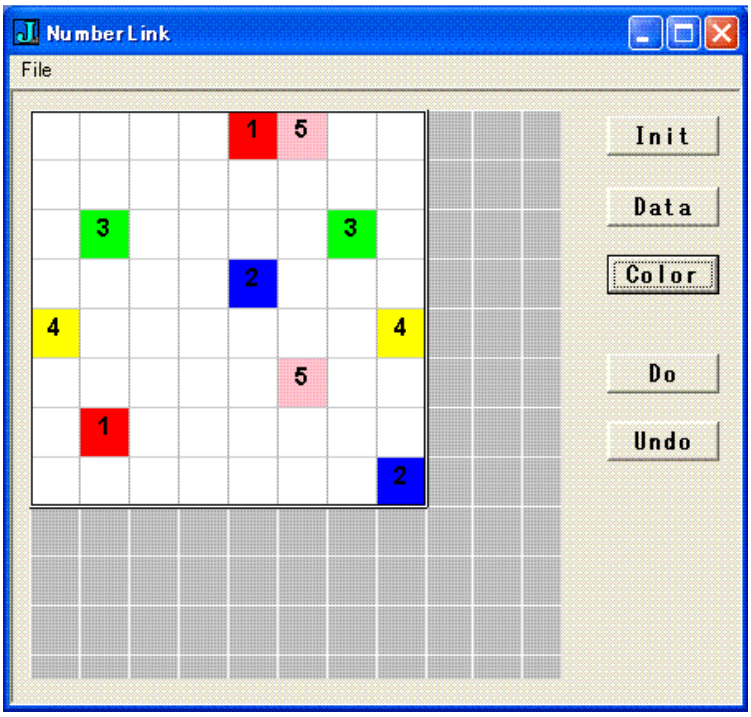
```
pnumberlink_grid_mbldown =: 3 : 0
'R C Value' =. nmouse ''
RR =: R
CC =: C
NB. border double line
DR =. 11 0 0 0
Border =. RR, CC, 1, 1, 16$DR
glgridborder Border
glpaintx ''
)
```

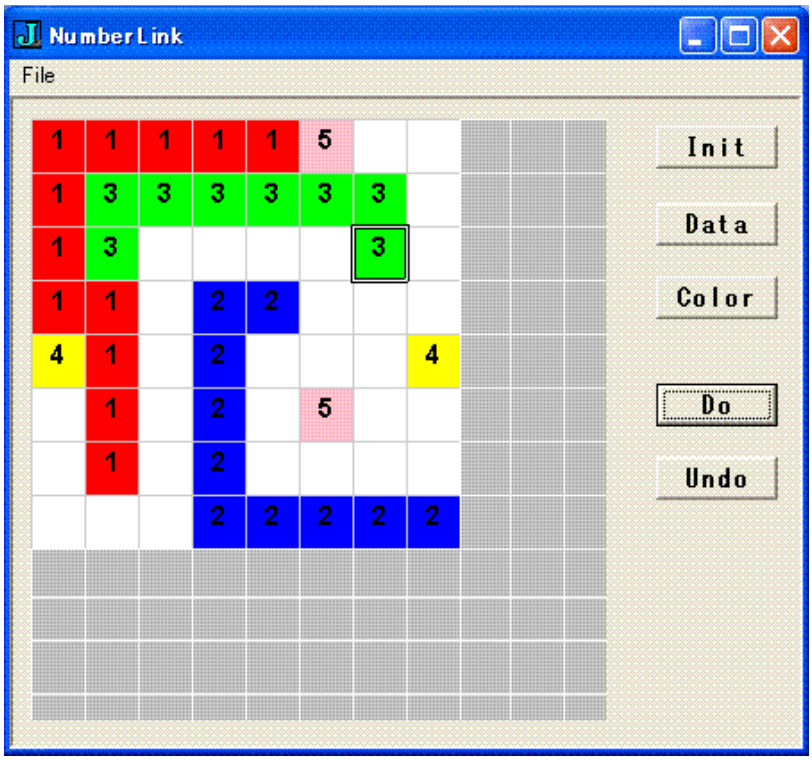
5.3　Number Link を楽しむ―実際の操作

```
NB.「四角の迷宮」p.79, 80
DB0 =:      0 0 0 0 1 5 0 0
DB0 =: DB0, 0 0 0 0 0 0 0 0
DB0 =: DB0, 0 3 0 0 0 0 3 0
DB0 =: DB0, 0 0 0 0 2 0 0 0
DB0 =: DB0, 4 0 0 0 0 0 0 4
DB0 =: DB0, 0 0 0 0 0 5 0 0
DB0 =: DB0, 0 1 0 0 0 0 0 0
DB0 =: DB0, 0 0 0 0 0 0 0 2
DB0 =: 8 8$DB0
DB  =: DB0

NB. 朝日新聞 2013/9/23 土曜日版　問題　難度3つ星
DD0 =:      1 0 0 0 6 0 0 0 0
DD0 =: DD0, 0 0 2 0 0 0 5 4 0
DD0 =: DD0, 0 0 3 0 0 0 0 0 0
DD0 =: DD0, 0 0 0 0 0 0 0 0 0
DD0 =: DD0, 0 0 0 0 0 3 0 0 0
DD0 =: DD0, 0 0 0 0 0 4 0 0 1
DD0 =: DD0, 0 0 0 0 0 5 0 0 0
DD0 =: DD0, 0 6 0 0 0 0 0 2 0
DD0 =: DD0, 0 0 0 0 0 0 0 0 0
DD0 =: 9 9$DD0
```

DD =: DD0


NumberLink
File
Init
Data
Color
Do
Undo


NumberLink
File
Init
Data
Color
Do
Undo

```
NB.  Number Link
NB.    using pnumberlink.ijs as class program
NB.  Usage: e.g. run 'DB'

load 'system\Packages\color\colortab.ijs'
Path =: 1!:40 ''
load Path, 'user\classes\pnumberlink.ijs'

NB. e.g. run 'DA', run 'DB', run 'DC',etc.
NB.  returned modified value
run =: 3 : 0
obj =: y. conew 'pnumberlink'
)

NB. 朝日新聞 2013/7/27 土曜日版　サンプル
DA0 =:      1 0 2 3
DA0 =: DA0, 0 0 1 0
DA0 =: DA0, 0 2 0 0
DA0 =: DA0, 0 0 0 3
DA0 =: 4 4$DA0

NB. 朝日新聞 2013/7/27 土曜日版　問題　難度4つ星
DC0 =:      1 0 0 0 0 0 0 3 2
DC0 =: DC0, 0 0 0 0 0 0 0 0 0
DC0 =: DC0, 0 0 4 3 0 0 0 0 0
DC0 =: DC0, 0 0 0 0 5 6 0 0 0
DC0 =: DC0, 0 0 0 0 0 0 5 0 0
DC0 =: DC0, 0 0 0 0 0 0 1 0 0
DC0 =: DC0, 0 0 0 0 0 0 0 0 0
DC0 =: DC0, 0 6 0 0 2 0 0 0 0
DC0 =: DC0, 0 0 0 0 0 0 0 0 4
DC0 =: 9 9$DC0
```

NB. 朝日新聞 2013/9/23 土曜日版　問題　難度3つ星
DD0 =:　　1 0 0 0 6 0 0 0 0
DD0 =: DD0, 0 0 2 0 0 0 5 4 0
DD0 =: DD0, 0 0 3 0 0 0 0 0 0
DD0 =: DD0, 0 0 0 0 0 0 0 0 0
DD0 =: DD0, 0 0 0 0 0 3 0 0 0
DD0 =: DD0, 0 0 0 0 0 4 0 0 1
DD0 =: DD0, 0 0 0 0 0 5 0 0 0
DD0 =: DD0, 0 6 0 0 0 0 0 2 0
DD0 =: DD0, 0 0 0 0 0 0 0 0 0
DD0 =: 9 9$DD0

NB.「四角の迷宮」p.79, 80
DB0 =:　　0 0 0 0 1 5 0 0
DB0 =: DB0, 0 0 0 0 0 0 0 0
DB0 =: DB0, 0 3 0 0 0 0 3 0
DB0 =: DB0, 0 0 0 0 2 0 0 0
DB0 =: DB0, 4 0 0 0 0 0 0 4
DB0 =: DB0, 0 0 0 0 0 5 0 0
DB0 =: DB0, 0 1 0 0 0 0 0 0
DB0 =: DB0, 0 0 0 0 0 0 0 2
DB0 =: 8 8$DB0

DA =: DA0
DB =: DB0
DC =: DC0
DD =: DD0

NB. subprograms called from pnumberlink.ijs
==========

NB. 行ごと、列ごとまとめて修正のための amend ツール ==========
NB.
NB.　6 to 2 => 6 5 4 3 2
NB.　2 to 6 => 2 3 4 5 6

```
to =: 3 : 0
:
if. x. <: y.
  do.     x. + i. >: y. - x.
  else.   x. - i. >: x. - y.
end.
)

NB. i行 j-k列の amend 用のアドレスを作る
makerow =: 3 : 0
:
<"(1) x. ,. y.
)



NB.   1 makerow 2 to 6
NB. +---+---+---+---+---+
NB. |1 2|1 3|1 4|1 5|1 6|
NB. +---+---+---+---+---+



NB. i列 j-k行の amend 用のアドレスを作る
makecol =: 3 : 0
:
<"(1) y. ,. x.
)

NB. display on the screen, ================
NB. e.g. display DB

NB. zero to space
zero2sp =: (":)`(' '"_)@.(0&=)
NB. e.g. zero2sp"(0) 10 2 0 12

NB. display DB revised 2013/9/6
display =: 3 : 0
```

zero2sp L:0 <"(0) y.
)

```
coclass 'pnumberlink'

require 'gl2'
corequire 'jzgrid'

PNUMBERLINK=: 0 : 0
pc pnumberlink;pn "NumberLink";
menupop "File";
menu new "&New" "" "" "";
menu open "&Open" "" "" "";
menusep ;
menu exit "&Exit" "" "" "";
menupopz;
xywh 330 174 34 12;cc cancel button;cn "Exit";
xywh 5 5 312 247;cc grid isigraph;
xywh 331 7 34 11;cc Init button;
xywh 329 148 34 11;cc Test0 button;
xywh 331 26 34 11;cc Data button;
xywh 331 44 34 11;cc Color button;
xywh 331 71 34 11;cc Do button;
xywh 329 89 34 11;cc Undo button;
pas 6 6;pcenter;
rem form end;
)

create0=: 3 : 0
wd PNUMBERLINK
formhwnd=: wd'qhwndp'
NB. initialize form here
wd 'pshow;'
)

create =: 3 : 0
NB. import data from argument y. of 'conew' function in the base script
n=: y.,('_'~:{:y.)#'_base_'
if. 0~:4!:0 <n do. wdinfo 'Invalid name: ',y. return. end.
```

```
smoutput n
smoutput 'Data:'
DAT =: ". n
smoutput DAT  NB. data for display, n = 'DA_base_' etc.
Col =: {: $DAT
Row =: {. $DAT
smoutput 'Col: ', (": {. $DAT), ' Row: ', (": {: $DAT)

wd PNUMBERLINK
wdfit ''
formhwnd=: wd'qhwndp'
wd 'pshow;'
)


destroy=: 3 : 0
wd'pclose'
codestroy''
)

pnumberlink_cancel=:pnumberlink_cancel_button=:pnumberlink_close=:de
stroy

formselect=: 3 : 'wd''psel '',formhwnd'


NB. select the object for command execution ================
NB.   modified by TN

sel =: 3 : 0
 wd 'psel ', 'pnumberlink'  NB. select parent form id
 glsel 'grid'               NB. select child 'grid' id
)

NB. ========================================
```

```
gridhs =: 30
gridws =: 30

NB. revised 0 => space
celldata =: 3 : 0
; (zero2sp_base_ each y.), each 0{a.
)


NB. celldata =: 3 : ';(": each y.), each 0{a.'

pnumberlink_Init_button=: 3 : 0
sel ''
glclear ''
glmapraw ''
glmap MM_TEXT
glnoerasebkgnd 1
glgrid ''
glshow ''
)


pnumberlink_Test0_button=: 3 : 0
sel ''
glgridfill gridws, gridhs, 0, 0
glgridrc 6 4        NB. make row(6) x column(4)
glgridh  6$gridhs
glgridw  4$gridws
NB. glgridfill 15 10 0 0
glgridtext celldata i.6 4
DR =: 11 0 0 0
border =: 0 0 6 4, 16$DR
glgridborder border
glpaintx ''
)
```

```
pnumberlink_Data_button=: 3 : 0
sel ''
glgridfill gridws, gridhs, 0, 0
glgridrc Row, Col        NB. make row(6) x column(4)
glgridh  Row$gridhs
glgridw  Col$gridws
NB. glgridfill 15 10 0 0
glgridtext celldata DAT
DR =: 11 0 0 0
border =: 0, 0, Row, Col, 16$DR
glgridborder border
glpaintx ''
)


NB. Coloring ================================
pnumberlink_Color_button=: 3 : 0
sel ''
glgridfont0 'arial 18 bold'
setinitcolor ''
setborder ''
)

WHITE  =: 255 255 255
SILVER =: 192 192 192
BLACK  =:   0   0   0
AQUA   =:   0 255 255
RED    =: 255   0   0
GREEN  =:   0 255   0
BLUE   =:   0   0 255
YELLOW =: 255 255   0
PINK   =: 255 192 203
NB. PURPLE =: 128   0 128
PURPLE =: 128   0 238
VIOLET =: 238 130 238
```

```
NB. COLORTABLE is imported via base loading ===
NB.   from 'system\Packeges\color\colortab.ijs'

OLIVE  =: ". 98 { COLORTABLE_base_
NB. AZURE =: 240 255 255
TAN   =: ". 130 { COLORTABLE_base_
CORAL =: ". 16 { COLORTABLE_base_
CYAN  =: ". 20 { COLORTABLE_base_
KHAKI =: ". 59 { COLORTABLE_base_
ORANGE =: ". 100 { COLORTABLE_base_
SKYBLUE  =: ". 124 { COLORTABLE_base_
MAGENTA  =: ". 80 { COLORTABLE_base_

attinit =: 3 : 0
j =.    <SILVER, WHITE,  BLACK, 1 _1 1
j =. j, <SILVER, RED,    BLACK, 0 _1 1
j =. j, <SILVER, BLUE,   BLACK, 0 _1 1
j =. j, <SILVER, GREEN,  BLACK, 0 _1 1
20&{. &> j
)

atts =: 3 : 0
j =.    <WHITE,  SILVER, BLACK, 1 2 _1
j =. j, <WHITE,  WHITE,  BLACK, 0 2 1
j =. j, <SILVER, AQUA,   BLACK, 0 2 1
j =. j, <SILVER, WHITE,  BLACK, 0 2 1
j =. j, <SILVER, RED,    BLACK, 0 2 1
j =. j, <SILVER, BLUE,   BLACK, 0 2 1
j =. j, <SILVER, GREEN,  BLACK, 0 2 1
j =. j, <SILVER, YELLOW, BLACK, 0 2 1
j =. j, <SILVER, PINK,   BLACK, 0 2 1
j =. j, <SILVER, PURPLE, BLACK, 0 2 1
j =. j, <SILVER, OLIVE,  BLACK, 0 2 1
j =. j, <SILVER, TAN  ,  BLACK, 0 2 1
j =. j, <SILVER, CORAL,  BLACK, 0 2 1
j =. j, <SILVER, CYAN ,  BLACK, 0 2 1
```

```
j =. j, <SILVER, KHAKI,  BLACK, 0 2 1
j =. j, <SILVER, ORANGE, BLACK, 0 2 1
j =. j, <SILVER, SKYBLUE, BLACK, 0 2 1
j =. j, <SILVER, MAGENTA, BLACK, 0 2 1
20&{. &> j
)

setinitcolor =: 3 : 0
glgridatt ,atts''
glgridrchw 0, 0, $DAT
glgridtype 3 + , DAT
glpaintx ''
)

setborder =: 3 : 0 NB. bordered with double black lines
DB =: 11 0 0 0                  NB. double black line
border =: 0, 0, ($DAT) ,16$DB    NB. bordering region
glgridborder border
glpaintx ''
)

setcolor=: 3 : 0
glgridatt ,atts''
glgridrchw P, Q
glgridtype , (Q $ COLOR)
glpaintx ''
)

setcolor0=: 3 : 0
glgridatt ,atts''
glgridrchw ColStart, ColRegion
glgridtype , (ColRegion $ COLOR)
glpaintx ''
)
```

```
pnumberlink_Do_button=: 3 : 0
smoutput 'Start: ', ":(R0), (C0)
smoutput 'End:   ', ":(RR), (CC)
smoutput 'Value: ', ": Value

nmouse2param ''

setcolor0 ''

glpaintx ''
return.

if. CC = C0
 do.
  RX =. (R0) to_base_ (RR)
  AM =. (CC)  makecol_base_ RX
 end.
if. RR = R0
 do.
  CX =. (C0) to_base_ (CC)
  AM =. (RR)  makerow_base_ CX
 end.
DAT =: Value AM } DAT  NB. revised value
NB. convert DAT to TXT and writes DAT on the grid - 2013/8/25 OK!
TXT =: ; (0{a.),~ L:0 (": L:0 <"(0) DAT)
glgridrchw 0, 0, Row, Col  NB. rewrite all DAT on the grid
glgridtext celldata DAT
glpaintx ''

NB.                  Coloring                  with                  Attribute
=======================================

COLOR =: Value + 3
smoutput 'COLOR: ', (":COLOR)
```

```
NB. AMM =. >:L:0 AM
AMM =. AM
smoutput 'AMM: ', ": AMM

NB. in case of reverse mouse set, reverse AMM - 2013/8/26
if. (CC = C0) *. ( ({. > {. AMM) > ({. > {: AMM) )
  do. AMM =. |. AMM end.
if. (RR = R0) *. ( ({: > {. AMM) > ({: > {: AMM) )
  do. AMM =. |. AMM end.
AMs =. {. AMM
AMe =. {: AMM
P =: >AMs
if. RR = R0
  do.
    Q =: 1, 1 + ({: >AMe) - ({: >AMs)
  end.
if. CC = C0
  do.
    Q =: (1 + ({. >AMe) - ({. >AMs)), 1
  end.
smoutput 'P: ', (":P), ', Q: ', (":Q)
setcolor ''
)


NB. simple data_from_mouse routine by T.N 2013/9/1 =========
nmouse =: 3 : 0
NB. smoutput sysdata
NB. extract x, y pixel position
'x y h' =. 0 1 3 { 0 ". sysdata
y =. h - y
NB. wd 'mb "x, y" ','*', (": x, y)
NB. calc. column, row position, and Value
C =.  <. x % gridws
R =.  <. y % gridhs
Value =. (<R, C){DAT
```

```
R, C, Value
)
NB. mouse selected to parameters
============================
NB. get DAT revised, and ColStart(color start) and ColRegion(region)
nmouse2param =: 3 : 0
if. CC = C0
 do.
  RX =. (R0) to_base_ (RR)
  AM =. (CC)  makecol_base_ RX
 end.
if. RR = R0
 do.
  CX =. (C0) to_base_ (CC)
  AM =. (RR)  makerow_base_ CX
 end.
DAT =: Value AM } DAT  NB. revised value
NB. convert DAT to TXT and writes DAT on the grid - 2013/8/25 OK!
TXT =: ; (0{a.),~ L:0 (": L:0 <"(0) DAT)
glgridrchw 0, 0, Row, Col  NB. rewrite all DAT on the grid
glgridtext celldata DAT
glpaintx ''

NB. Coloring with Attribute
===================================

COLOR =: Value + 3
smoutput 'COLOR: ', (":COLOR)

NB. AMM =. >:L:0 AM
AMM =. AM
smoutput 'AMM: ', ": AMM

NB. in case of reverse mouse set, reverse AMM - 2013/8/26
if. (CC = C0) *. ( ({. > {. AMM) > ({. > {: AMM) )
 do. AMM =. |. AMM end.
```

```
if. (RR = R0) *. ( ({: > {. AMM) > ({: > {: AMM) )
  do. AMM =. |. AMM end.
AMs =. {. AMM
AMe =. {: AMM
P =: >AMs
if. RR = R0
  do.
    Q =: 1, 1 + ({: >AMe) - ({: >AMs)
  end.
if. CC = C0
  do.
    Q =: (1 + ({. >AMe) - ({. >AMs)), 1
  end.
smoutput 'P: ', (":P), ', Q: ', (":Q)
ColStart =: P
ColRegion =: Q
NB. return DAT, ColStart, ColRegion
)

NB. Mouse Right Down - Set Start Cell ======================
pnumberlink_grid_mbrdown =: 3 : 0
'R C Value' =: nmouse ''
R0 =: R
C0 =: C
NB. border double line
DR =. 11 0 0 0
Border =. R0, C0, 1, 1, 16$DR
glgridborder Border
glpaintx ''
NB. wd 'mb "R, C, Value" ','*', '(', (": R, C),') = ', (": Value)
return.
NB. clear border line
CL =. 1 0 0 0
Clear =. R0, C0, 1, 1, 16$CL
glgridborder Clear
glpaintx ''
```

```
)

NB. Mouse Left Down - Set End Cell
============================
NB. editflag =: 0
pnumberlink_grid_mbldown =: 3 : 0
'R C Value' =. nmouse ''
RR =: R
CC =: C
NB. glpaintx ''
NB. wd 'mb "R, C, Value" ','*', '(', (": R, C),') = ', (": Value)
NB. border double line
DR =. 11 0 0 0
Border =. RR, CC, 1, 1, 16$DR
glgridborder Border
glpaintx ''
)


pnumberlink_Undo_button=: 3 : 0
smoutput '------------'
smoutput 'Undo:'
smoutput 'Start: ', ":(R0), (C0)
smoutput 'End:   ', ":(RR), (CC)
R0Start =. R0
C0Start =. C0
RREnd =. RR
CCEnd =. CC

Value =. (<R0, C0) { DAT
smoutput Value
smoutput 'Data:  '
smoutput DAT
NB.
if. (C0 = CC) *. (R0 > RR)
  do. RX =. R0
```

```
    R0 =. RR
    RR =. RX
  end.
if. (R0 = RR) *. (C0 > CC)
  do.
    CX =. C0
    C0 =. CC
    CC =. CX
  end.

NB. rewrite all DAT on the grid ==================
smoutput 'Test:'
smoutput R0, C0
smoutput RR, CC

NB. undo coloring ================================
if. C0 = CC
  do.
   smoutput 'test column'
   DAT =: 0 (< ( (>:R0)+i.(RR-R0)); C0 ) } DAT
  end.
if. R0 = RR
  do.
   smoutput 'test row'
   DAT =: 0 (< R0; (>:C0)+i.(CC-C0) ) } DAT
  end.

smoutput DAT

DAT =: Value (<R0Start, C0Start) } DAT
DAT =: 0 (<RREnd, CCEnd) } DAT
smoutput '--------------------'
smoutput DAT

glgridrchw 0, 0, Row, Col
glgridtext celldata DAT
```

setinitcolor ''
glpaintx ''
)