

# J802/803 のグラフィックス (1)

SHIMURA Masato  
<http://japla.sakura.ne.jp>

2014 年 10 月 16 日

## 目次

1	キャンバスのみの場合	1
2	2面のキャンバス	3
3	モンドリアン	5
4	ボタンを付ける	8

## 1 キャンバスのみの場合

*gl2* を用いた本格的な 2 *D* グラフィックスを描く

*J802/803* の *QT* 版ではフォームエディタが (まだ) ないので、手書きする。

最初の 2 行 次の 2 行は必須の定型文である

- `coinsert 'jgl2'` と書くと *g* オブジェクト (*gl2* 関数群) を *jgl2* というバスケットに入れて、*gl2* の関数の衝突を防ぐための煩瑣なロケール記号を一切省略できる。

```
require'gl2'
```

```
coinsert'jgl2'
```



鉄壁の3本のスクリプト キャンバス (`GL2TEST`), `run(gl2test_g_run)`, `paint(gl2test_g_paint)`  
 のトリオの連携は鉄壁でる。

### 1. キャンバス

- キャンバスの名前を決める。タイトルは大文字で、文中は小文字で記入する
- キャンバスのサイズは `minwh` で  $(x,y)$  を指定する。最大は  $1000 \times 1000$
- `J602` までは `xywh` で始点  $(x,y)$  も指定した。ここを `xywh 0 0 400 200` と書き換えるだけで `J602` で動く。
- `cc g` の `g` はチャイルドの名前 (任意)。この名前以降の画面と関連付ける

```
GL2TEST=: 0 : 0
pc gl2test closeok;
minwh 400 200;cc g isigraph;
pas 0 0;
rem form end;
)
```

### 2. `run`

- `gl2test_g_run` という風に `g_run` を書き込む
- 次は最小の構成でこの2行は必須

```
run=: gl2test_g_run=: 3 : 0
wd GL2TEST
wd'pshow;'
)
```

- ここに `gl2test_g_paint ''` を書き込むとエラーになる
- `x,y` など、左右の引数は受け付けない

### 3. `paint` グラフィックス本体] 本体のスクリプトはここに定義する

NB. -----

```
gl2test_g_paint=: 3 : 0
glrgb 255 0 255
glbrush ''
glrect 10 30 180 240
glines 0 0 400 200
)
```

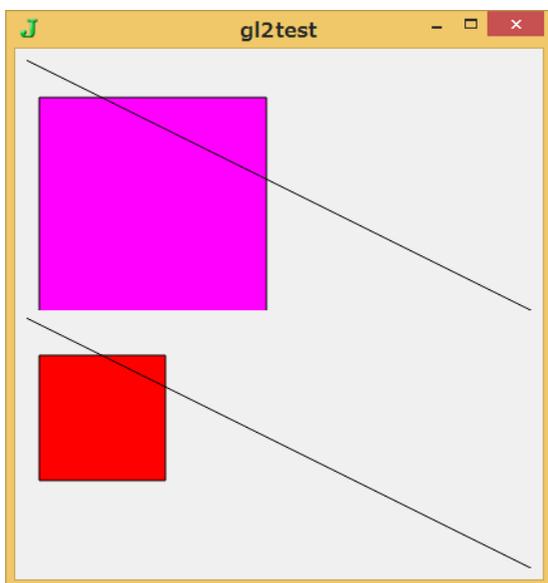
#### 4. 鉄壁の3本の関係

- `gl2test_g_paint` を単独でタイプして動かすとハングして *J* が終了する。
- `gl2test_g_paint` を `gl2test_run` の中に書き込むとエラーになる。(run の中で動かす方法が不明)

## 2 2面のキャンバス

チャイルドの名前 *run,paint* と関連付けしないと *gl2* での描画はうまくいかない。2画面 *g0,g1* で確認してみよう。

2画面の同時表示は次の方法で問題なくできる。



```

GL2TEST=: 0 : 0
pc gl2test closeok;
minwh 400 200;cc g0 isigraph;
bin s;
minwh 400 200;cc g1 isigraph;
pas 0 0;
rem form end;
)

```

```

run=: gl2test_run=: 3 : 0
wd GL2TEST
wd'pshow;'
)

```

```

NB. -----
gl2test_g0_paint=: 3 : 0
NB. tg0=: 3 : 0
glrgb 255 0 255
glbrush ''
glrect 10 30 180 240
gllines 0 0 400 200
)

```

```

gl2test_g1_paint=: 3 : 0
NB. tg1=: 3 : 0
glrgb 255 0 0      NB. RGB red
glbrush ''        NB. red brush
glrect 10 30 100 100 NB. red rectangle
gllines 0 0 400 200
)

```

### 3 モンドリアン

チューリッヒ美術館展に出品されている、モンドリアンの「赤、青、黄色のコンポジション (1936)」を *gl2* で描く。

- 最初の 2 行

```
NB. *****
require'gl2' NB. load gl2 definitions in jgl2 locale
coinert'jgl2' NB. allow use of gl2... without _jgl2_
```

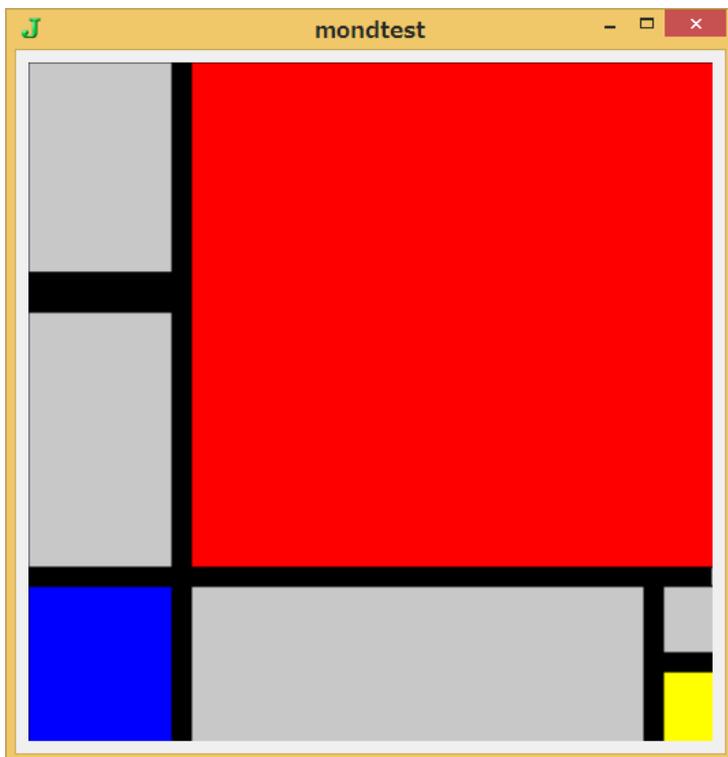
- キャンバスの設定。最初は (1000 × 1000) で座標を取ったが、ノート PC に合わせて (500 × 500) に縮小した

NB. Mondrian 1930 Composition with Red Blue and Yellow

```
MONDTEST=: 0 : 0
pc mondtest closeok;
minwh 500 500 ;cc gx0 isigraph;
pas 0 0;
rem form end;
)
```

```
mondtest_gx0_run=: 3 : 0
wd MONDTEST
wd'pshow;'
)
```

- 最近の *J* は左上が (0, 0)、右下が最大で (1000, 1000) となっている。座標を上から下へ、覗き込むように取らないと天地逆となる。
- 13 個の 4 角形で描き、直線は用いていない。
- *glbrush* ” は一個ずつ絵筆を変える。



- `mondtest_gx0_paint` など *paint* 系のスクリプトは引数 (*y*) を受け付けない。 ’’のみ
- `<.` は *Floor* (小数点以下切り捨て)
- `-:` は  $\frac{1}{2}$

NB. Cambus is expected 1000\*1000  
 NB. for note Pc minimized 500 \* 500  
 NB. 1000 is 0 <--> 999

```
mondtest_gx0_paint=: 3 : 0
NB. Mondrian composition (1936)
NB. -: is half //same *&1r2
NB. --first -draew long line---
NB. --vertical line
glrgb 0 0 0
glbrush ''
glrect -: 0 0 239 999
NB. -----1 to 5----
glrgb 200 200 200 NB. Gray
glbrush ''
glrect -: 0 0 210 311
glrgb 0 0 0 NB. Black
glbrush ''
glrect -: 0 311 210 369
glrgb 200 200 200 NB. Gray
glbrush ''
glrect -: 0 369 210 744
glrgb 0 0 0
glbrush ''
glrect -: 0 744 999 772
glrgb 0 0 255 NB. Blue
glbrush ''
glrect -: 0 772 210 999
```

```
NB. -----
glrgb 255 0 0 NB. Redh
glbrush ''
glrect -: 239 0 999 744
glrgb 200 200 200
glbrush ''
glrect -: 239 772 901 999
NB. -----
glrgb 0 0 0
glbrush ''
glrect -: 901 772 929 999
NB. -----
glrgb 200 200 200
glbrush ''
glrect -: 929 772 999 870
glrgb 0 0 0
glbrush ''
glrect -: 929 870 999 898
glrgb 255 255 0 NB. Yellow
glbrush ''
glrect -: 929 898 999 999
)

run=: mondtest_gx0_run
```

### 3.1 歓迎されない gl2 の動作

- 色の滲み出し。区画から滲み出ることがある。絵具ではないのではみ出た区画を上書きすれば修復できる
- QT 版の縮小画面では各長方形は指定サイズを保っているが、拡大画面にすると指定が外れる (602 でも同様)

## 4 ボタンを付ける

グラフィックスの傍らに QT でボタンを付けてみよう。

### 4.1 ボタンを付ける

QT ではボタンの座標は指定しない。グループづけと改行でボタンを整然と並べる。

```
ISEDIT=: 0 : 0
```

```
pc isedit closeok;
```

```
bin vh;
```

```
cc minus button;cn "<<";
```

```
cc plus button;cn ">>";
```

```
bin s;
```

```
cc redisplay button;cn "&Redisplay";
```

```
cc cancel button;cn "&Cancel";
```

```
bin z;
```

```
minwh 400 200;cc graf isigraph;
```

```
bin z;
```

```
pas 0 0;
```

```
rem form end;
```

```
)
```

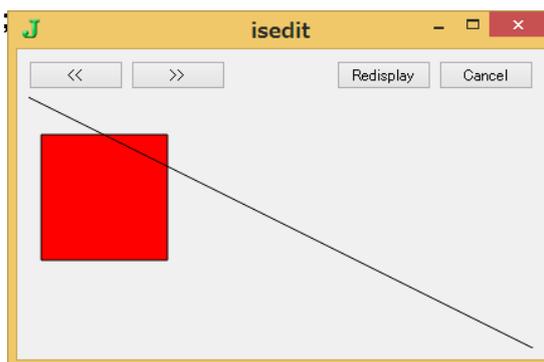
```
isedit_run=: 3 : 0
```

```
wd ISEDIT
```

```
wd 'pshow'
```

```
)
```

- *bin vh* 取り去ると 4 個のボタンが縦 1 列に並ぶ
- *bin s* 中間の区切り (グループの区分け) 取り去ると 4 個が横 1 列に並ぶ
- *bin z* 改行



```

isedit_graf_paint=: 3 : 0
glrgb 255 0 0      NB. RGB red
glbrush ''        NB. red brush
glrect 10 30 100 100 NB. red rectangle
gllines 0 0 400 200
)

```

## 4.2 run と paint の入れ替え

- *paint* の内容を一々書き換えなくて入替えたいが、*run,paint* は引数を受け付けない。
- グラフィックスの内容は *edit0,edit1* のように纏めておき、*isedit\_graph\_paint* の中で引用することはできる

```

NB. argument is definition
run2=:isedit_run=: 3 : 0
wd ISEDIT
gsel graf      NB. OK
wd 'pshow'
)

edit0=: 3 : 0
glrgb 255 0 255      NB. RGB red
glbrush ''          NB. red brush
glrect 10 30 100 100 NB. red rectangle
gllines 0 0 400 200
)

isedit_graf_paint=: 3 : 0
edit0 ''
)

edit1=: 3 : 0
glrgb 255 0 0      NB. RGB red
glbrush ''        NB. red brush
glrect 10 30 100 100 NB. red rectangle
gllines 0 0 400 200
)

```

## 4.3 ボタンへの関連付け

- ボタン毎に1のスクリプトを作る

```

isedit_cancel_button=: 3 : 0
wd 'pclose'
)

```

isedit\_close=: isedit\_cancel\_button