C.Reiter のグラフィックスのアドオン fvj4 のテスト

SHIMURA Masato

2016 年 5 月 20 日

目次

C.Reiter の *Fractal Visualization and J* はフラクタルグラフィックスの名著として永く版を重ねている。この度 *J8(QT* 版*)* に対応した第 *4* 版が出版された。オンデマンド出版の *Lulu.com* で著書名を入力すると第 *4* 版も出てきて電子版なら *6* ドル程度で購入できる。[*1]

1 dwin のチュートリアルでテスト

J8 はグラフィックスや画面周りに QT を採用した。J6 のウインドウズとは差異が大きく、かつ QT の解説が少ないことから少し混乱が生じていた。

この度 (2016/04) に C.Reiter が *Fractal Visualization and J* 第 *4* 版に合わせて、*QT* のグラフィックスに対応した *fvj4* を公表した。

1.1 dwin の画面

映画を思い浮かべてみよう。大きなスクリーンに映写機で投影する。

一方フィルム自体は手の平サイズである。

fvj4 の *2* 次元グラフィックス *dwin.ijs* ではスクリーンは次で指定する。*(*どこかに書き込んでおけばよい）

```
setWIN_WH 500 500  NB. 1000 1000 OK
```

これに対しフィルムのサイズは *dwin* の左引数で指定する

```
min(x,y),max(x,y)  dwin  ' any name'
```

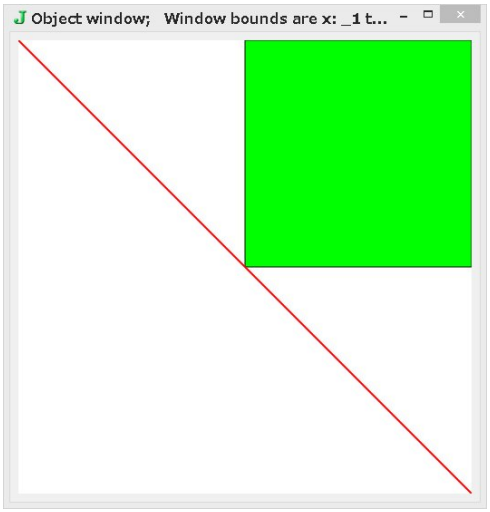
[*1] 日本の Amazon からも購入できるがオンデマンドのペーパーブックで割り高である。

```
_1 _1 1 1 dwin ''
```

dwin の左引数は任意なので、目的に合わせてフィルムのサイズを決めればよい

フィルムがスクリーンに投影される。このとき縦横の比率が双方同じならばそのまま、比率が異なる場合は、スクリーンに合わせて、上下左右の引き延しや圧縮が行われる。

1.2 2×2 の画面で先ず描く



1. *dwin* のロード

```
require '~addons/graphics/fvj4/dwin.ijs'
```

2. グラフィック画面を開く。 左下 (_1 _1), 右上 (1 1) のフィルム画面を定義。画面中央が (0 0) になる。

```
_1 _1 1 1 dwin ''
```

3. 500×500 のスクリーンの左下が（*-1,-1*）、右上が（*1, 1*）となる。
4. 四角の定義。 画面の右上 $\frac{1}{4}$ を塗りつぶす

```
] sq=: 0 0 ,0 1,1 1,: 1 0


0 0
0 1
1 1
1 0
```

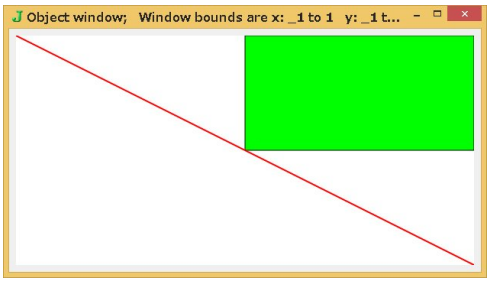
C.Reiter は `sq=: #: 0 1 3 2` と *2* 進法を用いて *4* 点を生成している

5. ポリゴンで $\frac{1}{4}$ を塗りつぶし、左上隅から右下隅へラインを弾く

```
0 255 0 dpoly sq
255 0 0 dline _1 1,:1 _1
```

6. 左引数は色指定（*RGB*）である

1.3　600 × 300 の画面を生成



1. 600 × 300 の画面を生成

```
    setWIN_WH 600 300
600 300
```

2. グラフィックコマンドはそのまま用いる。縦が $\frac{1}{2}$ に押しつぶされる

```
_1 _1 1 1 dwin ''
0 255 0 dpoly sq
255 0 0 dline _1 1,:1 _1
```

2　Ｊのグラフィックス

J8 にはおよそ次のようなグラフィックスが入っている。

1. 簡易グラフィックス *graph*
2. ２ *D* の基礎のグラフィックス *gl2*
3. 3D グラフィックス *gles*
4. ビットマップデータのビューアー *viewmat*
5. addon のパッケージ

- *fvj4*　　*C.Reiter* の *Fractal Visualization and J(4rh Ed.)* のツール
- *Turtle Graphics*

*2

*2 OpenGL と TurtleGraphics の QT への対応は遅れ気味である

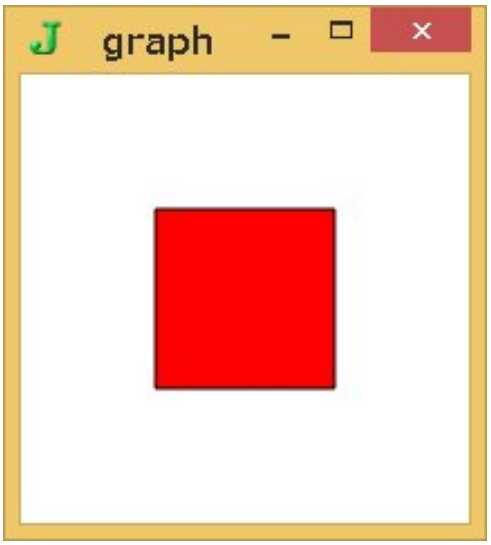
2.1 graph

- 簡易グラフィックス *graph* のチュートリアルは次に入っており、デモが見られる。
  Help/Studio/Labs/Graphics/Graph Utilities
- *Lab* のドキュメント *(ijt)* は次にある。
  addons/labs/labs/graphics/graphut.ijt
  ここに多くの *ijt* が入っている。また、*lab* に *ijt* を入れておくと *Studio* に反映される。
- *graph* の定義ファイルは次にある。プリントしてみるのもよい。
  *addons/graphics/graph/*にある *graph.ijs* と *jzgraph.ijs*

graph では次のように簡単に描くことが出来る

```
reruire 'graph'

RED gdrect gddraw _0.4 _0.4 0.8 0.8
```



2.2 gl2

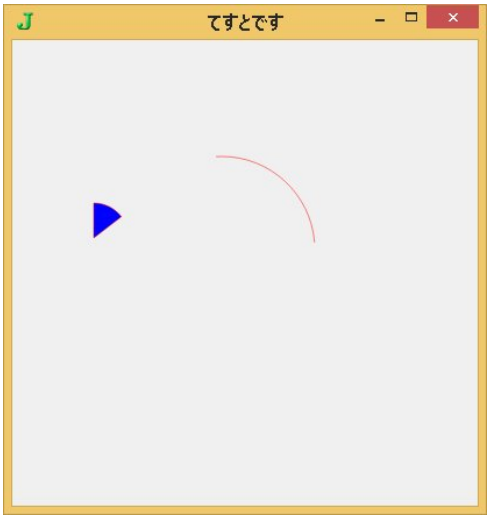
基礎グラフィックスの定義法は厳密である。

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

GL2TEST=: 0 : 0
pc gl2test closeok;pn "てすとです";
minwh 400 400;cc g0 isigraph flush;
pas 0 0;
rem form end;
)

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

gl2test_g0_paint=: 3 : 0
NB. arc pie
glrgb 255 0 0
glpen 0 1
glrgb 0 0 255
glbrush''
glpie 40 140 60 60 200 70 70 0
glarc 100 100 160 160 300 170 170 0
)
```



2.3 dwin と gl2

C.Reiter の *dwin* は *gl2* 上で動くグラフィックツール。*J6* を継承し、*QT* に対応させたうえで、スマートフォンやタブレット上でも動くようにしたもの

基本のグラフィックコマンドは次の *4* 個。

	gl2	*dwin*

スクリーン	*minwh*	`setWIN_WH`
フィルム		*dwin*
ポリゴン	*glpolygon*	*dpoly*
ライン	*glline*	*dline*
ピクセル	*glpixel*	*dpixel*

3 dwin の解説

3.1 キャンバス

1. 原点は左下を踏襲
2. *setWIN_WH*
   窓、絵画のキャンバス、映画のスクリーン。大きさは好みで指定できます。
3. *dwin*
   上村松園や黒田清輝の展覧会に下絵も展示されていた。原寸大もあれば小さいものもあった。画家は小さな下絵で構想を確認することもあるようだ。
   PC、スマートフォン、タブレットとサイズや形状が異なるときは下絵や映画のフィルム方式が便利なようだ。

   ```
   setWIN_WH 500 500
   _1 _1 1 1 dwin 'testdwin'
   ```

   このように小さい任意のサイズで原画を描いておき、スクリーンに投影する方式を取っている。
4. *fvj4* では *dwin* 本体は *QT* に対応するように大幅に書き直されているが他の描画関数は小幅な修正で済ませている
5. データは縦型

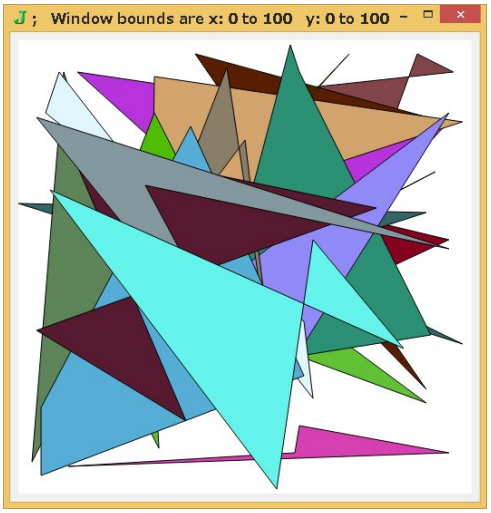
3.2 500×500 の画面で乱数で出したポリゴンを描く

1. 画面のリセット

```
wd'reset;'
```

2. 500×500 の画面を生成

```
   setWIN_WH 500 500
500 500
```



3. ポリゴンの *4* ポイントを乱数で出す

```
   ,<"2 ? 5 4 2 $100
+-----+-----+-----+-----+-----+
|36 91|71 54|33 33|71 75|67 18|
|74 47|72 31|27 31|67  4|51 46|
|96 66|35 20|93 80|25 33|71 46|
|24 51|72 58|23  4|61 79|78 95|
+-----+-----+-----+-----+-----+
```

4. ポリゴンを *20* 枚描く。*RGB* も乱数で打ち出す。

```
   0 0 100 100 dwin ''
(?20 3$256) dpoly ?20 4 2$100
```