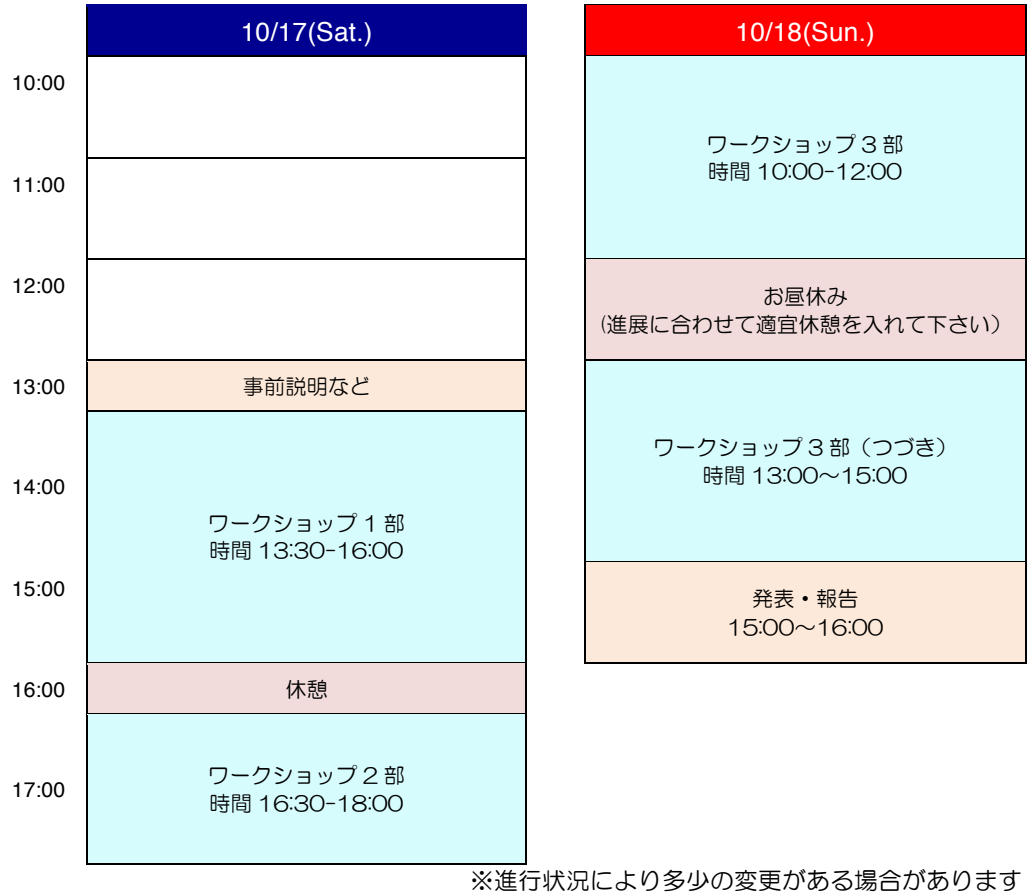
VSJ Programming Workshop 2015

@The University of Electro-Communications
2015.10.17(Sat.) – 18(Sun.)

スケジュール



※進行状況により多少の変更がある場合があります（特に10/17）

連絡事項

- 建物内にはゴミ箱がありません．建物玄関より外にでて右手（喫煙所の近く）にあるゴミ箱に分別して捨てて下さい．
- 二日目のお昼ご飯は各自で用意して下さい．（コンビニの場所等は地図を参照してください）

本ワークショップのねらい

・実験プログラミングを自力で作成できるよう，その特徴を学ぶ

・実験プログラミングの学習≠プログラミング言語の学習
　プログラミング言語の中身に関する内容の学習がメインテーマではない
　目的指向的な学習（やりたいことを実現するにはどうするか）
　グループ形式によるワークショップ（１人ではなく，みんなで学習する）

・実験プログラミングに必要な要素（パーツ）を学習する.
　パーツの種類，役割，組み立て方

目次

0．プログラムの「読み方」

・実験プログラムとはどのようなものだろう？
・各パーツの色分けを一緒にしていきましょう.

```
1   #include <psychlops.h>
2   using namespace Psychlops;
3
4   void psychlops_main()
5   {
6       /*画面設定**********************/
7       int window_width;//提示するwindowの幅
8       int window_height;//提示するwindowの高さ
9       Canvas display;//プログラム用に画面を確保
10
11      window_width = 1024;//提示するwindowの幅
12      window_height = 768;//提示するwindowの高さ
13      display.set(window_width, window_height,Canvas::window);//提示画面を準備する
14
15
16      /*実験条件**********************/
17      const int independent_var1_step_number = 2;//形の種類(正方形、円)
18      const int    repetetion_number = 10;//一試行あたりの繰り返し数
19
20      const int max_trialnumber = independent_var1_step_number*repetetion_number;//全試行数2x10=20
21
22      int var1_condition_number[max_trialnumber];//全試行分の条件の順序(各試行でどの形を提示するか)
23      int shuffled_trialnumber[max_trialnumber];//試行の順序をシャッフルしたもの
24
25      for(int i=0; i<max_trialnumber; i++)
26      {
27          shuffled_trialnumber[i] = i;//全試行に番号をつける
28      }
29      Math::shuffle(shuffled_trialnumber, max_trialnumber);//全試行につけた番号を並び替える
30
31      for(int trialnumber=0; trialnumber<max_trialnumber; trialnumber++)
32      {
33          var1_condition_number[trialnumber] = shuffled_trialnumber[trialnumber] %
                        independent_var1_step_number; //シャッフルされた番号を条件で使う番号だけにする
34      }
35
36
37      /*刺激描画**********************/
38      Psychlops::Rectangle fixation[2];//注視点
39      Psychlops::Rectangle stim1;//刺激１(正方形)
40      Psychlops::Ellipse stim2;//刺激２(円)
41      double r = 0.0, g = 0.5, b = 1.0;//刺激の色を指定する変数
42      int stim_size = 200;//刺激のサイズ
43
44      stim1.set(stim_size,stim_size);//刺激の大きさの指定
45      stim1.centering();//中心に寄せる
46      stim1.fill = Color(r, g, b);//色の指定
47
48      stim2.set(stim_size,stim_size);//刺激の大きさの指定
49      stim2.centering();//中心に寄せる
50      stim2.fill = Color(r, g, b);//色の指定
51
52      fixation[0].set(5,1).centering();//注視点(横棒)
53      fixation[0].fill = Color(0.0,0.0,0.0);//色の指定
54
```

```
55      fixation[1].set(1,5).centering();//注視点(縦棒)
56      fixation[1].fill = Color(0.0,0.0,0.0);//色の指定
57
58
59      /*刺激提示**********************/
60      Clock timer_rt;//Clock タイマー型
61      double start_t, end_t;//タイマーの値を入れる変数
62
63      double bg_lum ;//背景輝度の指定
64
65      bg_lum = 0.5;
66
67      int blank_duration;//ブランクの長さのための変数
68      int frame;//フレーム数を数えるための変数
69
70      int result[max_trialnumber];//反応を入れるための配列
71      double result_rt[max_trialnumber];//反応時間を入れるための配列
72
73      while(!Keyboard::spc.pushed())
74      {
75          display.clear(Color(bg_lum,bg_lum,bg_lum));//画面は一回ごとにきれいにして・・・
76          display.msg("Push space key to start.",window_width/2,380,Color::white);//文字を提示
77          display.flip();//フリップする(表示する)
78      }
79
80      for(int trialNow=0 ; trialNow<max_trialnumber; trialNow++)//試行回数分繰り返す
81      {
82
83          Input::refresh();//キー入力をリセットする
84
85          blank_duration = Psychlops::random(60)+30;//予測をさせないためにブランクの時間をランダムにする
86
87          frame = 0;//フレームを数えるための数値の初期値(ゼロから数える)
88          while( frame < blank_duration)//ブランクの長さ(大きさ)になるまでframeを足し続ける
89          {
90              display.clear(Color(bg_lum, bg_lum, bg_lum));//画面は一回ごとにきれいにして・・・
91              for(int i = 0; i<2; i++)
92              {
93                  fixation[ i ].draw();//画面に描くもの(注視点)
94              }
95              display.flip();//フリップする(表示する)
96              frame++;//frameを一ずつ足す
97          }
98
99          Input::refresh();//キー入力をリセットする
100
101         display.clear(Color(bg_lum, bg_lum, bg_lum));//画面は一回ごとにきれいにして・・・
102         switch (var1_condition_number[trialNow])//カッコ内の数値によって行なう処理を変える
103         {
104         case 0:
105             stim1.draw();//画面に描くもの(正方形)
106             break;
107         case 1:
108             stim2.draw();//画面に描くもの(円)
109             break;
110         }
111         display.flip();//フリップする(表示する)
112
113
114         /*反応取得**********************/
115         timer_rt.update();//タイマーの初期化(「現在」を取得する)
```

```
116         start_t = timer_rt.at_msec();//「現在」の値(=キーを押す前の時間)
117
118         while(true)//ずっと繰り返す(breakがきたら繰り返しを終わる)
119         {
120             if (Keyboard::left.pushed())//左カーソルキーが押されたら
121             {
122                 result[trialNow] = 0;//反応はゼロ
123                 timer_rt.update();//タイマーの初期化(「現在」を取得する)
124                 break;//while文から脱出
125             }
126             if (Keyboard::right.pushed())//右カーソルキーが押されたら
127             {
128                 result[trialNow] = 1;//反応は１
129                 timer_rt.update();//タイマーの初期化(「現在」を取得する)
130                 break;//while文から脱出
131             }
132         }
133         end_t = timer_rt.at_msec();//「現在」の値(=キーを押したときの時間)
134         result_rt[trialNow] = end_t-start_t;//反応時間　=　押した後の「現在」－押す前の「現在」
135
136     }
137
138
139     /*データ保存**********************/
140     int number[max_trialnumber];
141     Input::refresh();
142     for (int i =0; i<max_trialnumber; i++)
143     {
144         number[ i ] = i+1;
145     }
146     Data::savearray("EXP_choiceRT_%TIME_.txt", "Num¥tSHAPE¥tJudgement¥tA", max_trialnumber,
                                        number, var1_condition_number, result, result_rt);
147
148     Input::refresh();
149     while(!Keyboard::spc.pushed())
150     {
151         display.clear(Color(bg_lum,bg_lum,bg_lum));//画面は一回ごとにきれいにして・・・
152         display.msg("Thanks. Push spc key to quit.",500,380,Color::white);//文字を提示
153         display.flip();//フリップする(表示する)
154
155     }
156
157
158 }
```

ポイント

実験プログラムの基本パーツ

画面設定，条件設定，刺激準備（描画），刺激提示，反応取得，データ保存，というパーツで成り立っています．

・どんなに大きなプログラムでもパーツにわけることができます．
・触る必要のないパーツもあります
・各パーツでは，設定（何を使って）→命令（どうするか），設定→命令…の繰り返しになっています．
・全体的にみても設定（画面設定，条件設定，刺激準備）→命令（刺激提示，反応取得，データ保存）となっています．

1．色々いじってみよう

・どんなパーツで構成されているか確認してみよう．
・Psychlopsの基本を知ろう．

```
1  #include <psychlops.h>
2  using namespace Psychlops;
3
4  void psychlops_main()
5  {
6      /*画面設定**********************/
7      Canvas display;//プログラム用に画面を確保
8
9      display.set(1024, 768,Canvas::window);//使う画面の大きさ、種類
10
11
12     /*刺激描画**********************/
13     Psychlops::Rectangle stim1;//使う形
14     stim1.set(50,50);//使う形のサイズ(x,y)
15     stim1.centering().shift(0,0);//使う形の位置(中心にそろえて、移動する.)
16     stim1.fill = Color(0.0, 1.0, 0.0);//使う形の色(明るさ)
17
18
19     /*刺激提示**********************/
20     while(!Keyboard::esc.pushed())//while文(中の条件になるまで繰り返す)
21     {
22         display.clear(Color(0.5, 0.5, 0.5));//画面は一回ごとにきれいにして・・・
23         stim1.draw();//画面に描くもの
24         display.flip();//フリップする(表示する)
25     }
26
27
28 }
```

ポイント

・include <psychlops.h> using namespace Psychlops; が一番上に書いてあるかを確認しましょう．
・void psychlops_main(){何をするのか} のように書いていきます．

・Canvas と書くことで，「Psychlopsで画面を使う」ことをPCに伝えます．
・Psychlops::Rectangleは，どのような形を使いたい（描画したいか）を伝えるものです．
・centering().shift()で中央に揃えて移動させます．
　この時，水平方向は（x方向）プラスで右、マイナスで左，垂直方向（y方向）はプラスで下、マイナスで上です．
・色はRGB（赤，緑，青）の三つをそれぞれ割合で伝えます
・画像の表示では，画面は一回ごとに消して，描きたいものを伝え，表示する，の繰り返しをしています．
・while(条件){何をするか}で，ある条件になるまで繰り返すことを示しています（後で説明します）

・行の最後には必ず「　；　」（セミコロン）を入れましょう．
・プログラムを書くときには半角英数字を使います．//（スラッシュを二つ）の後ろはメモです（プログラムとは判断されません）．全角文字（特にスペース）が入らないように気をつけましょう．

・とにかくいじって慣れていきましょう．

2．変数とは？

・変数の中身を変化させると…　変数の名前を変更すると…

```
1   #include <psychlops.h>
2   using namespace Psychlops;
3
4   void psychlops_main()
5   {
6       /*画面設定**********************/
7       int window_width;//提示するwindowの幅
8       int window_height;//提示するwindowの高さ
9       Canvas display;//プログラム用に画面を確保
10
11      window_width = 1024;//提示するwindowの幅
12      window_height = 768;//提示するwindowの高さ
13      display.set(window_width, window_height,Canvas::window);//提示画面を準備する
14
15
16      /*刺激描画**********************/
17      Psychlops::Rectangle stim1;//使う形
18
19      double r1, g1, b1;//使う形の色
20      int stim1_sizeW, stim1_sizeH;//使う形のサイズ
21      int stim1_posiX = 0, stim1_posiY = 20;//使う形の位置
22
23      r1 = 0.0;//r
24      g1 = 1.0;//g
25      b1 = 0.0;//b
26
27      stim1_sizeW = 50;//形の幅
28      stim1_sizeH = 50;//形の高さ
29
30      stim1.set(stim1_sizeW,stim1_sizeH);//使う形のサイズ(x,y)
31      stim1.centering().shift(stim1_posiX,stim1_posiY);//中央寄せして位置を移動する
32      stim1.fill = Color(r1, g1, b1);//使う形の色(明るさ)
33
34
35      /*刺激提示**********************/
36      double bg_lum;//背景輝度
37      bg_lum = 0.5;//背景輝度の値
38
39      while(!Keyboard::esc.pushed())//while文(中の条件になるまで繰り返す)
40      {
41          display.clear(Color(bg_lum, bg_lum, bg_lum));//画面は一回ごとにきれいにして・・・
42          stim1.draw();//画面に描くもの
43          display.flip();//フリップする(表示する)
44      }
45
46
47  }
```

ポイント

・変数の名前は自由に決められます．
・基本的な使い方は，何を使うかを伝え（宣言），それにどんな数値を入れるのかを伝えます（初期化）．宣言と初期化を同時に行なうこともできます（この後しばらくはわかりやすさのために，宣言と初期化を分けて書いています）
・変数は値を変更しない限り，ずっと同じ値が入ったままです．
・整数ならばint 小数ならばdoubleを使いましょう．
・異なる型同士の計算の時には注意が必要です．小数点精度が必要な時にはdoubleを使いましょう．

3．たくさん描いてみよう１

・同じものを複数表示したい場合にはどのように書いたらよいでしょう？
・別の形を描きたい時にはどうしたらよいでしょう？

```
#include <psychlops.h>
using namespace Psychlops;

void psychlops_main()
{
    /*画面設定**********************/
    int window_width;//提示するwindowの幅
    int window_height;//提示するwindowの高さ
    Canvas display;//プログラム用に画面を確保

    window_width = 1024;//提示するwindowの幅
    window_height = 768;//提示するwindowの高さ
    display.set(window_width, window_height,Canvas::window);//提示画面を準備する


    /*刺激描画**********************/
    Psychlops::Rectangle stim1, stim2;//使う形(Psychlops::Ellipse stim1;に変えると・・・)

    double r1, g1, b1;//使う形の色(stim1)
    int stim1_sizeW, stim1_sizeH;//使う形のサイズ(stim1)
    int stim1_posiX, stim1_posiY;//使う形の位置(stim1)

    r1 = 0.0;
    g1 = 1.0;
    b1 = 0.0;//使う形の色(stim1)

    stim1_sizeW = 50;
    stim1_sizeH = 50;//使う形のサイズ(stim1)

    stim1_posiX = 0;
    stim1_posiY = 20;//使う形の位置(stim1)

    stim1.set(stim1_sizeW,stim1_sizeH);//使う形のサイズ(x,y)
    stim1.centering().shift(stim1_posiX,stim1_posiY);//中央寄せして位置を移動する
    stim1.fill = Color(r1, g1, b1);//使う形の色(明るさ)

    double r2, g2, b2;//使う形の色(stim2)
    int stim2_sizeW, stim2_sizeH;//使う形のサイズ(stim2)
    int stim2_posiX, stim2_posiY;//使う形の位置(stim1)

    r2 = 0.0;
    g2 = 1.0;
    b2 = 0.0;//使う形の色(stim2)

    stim2_sizeW= 50;
    stim2_sizeH= 50;//使う形のサイズ(stim2)

    stim2_posiX = -100;
    stim2_posiY = 20;//使う形の位置(stim1)

    stim2.set(stim2_sizeW,stim2_sizeH);//使う形のサイズ(x,y)
    stim2.centering().shift(stim2_posiX,stim2_posiY);//中央寄せして位置を移動する
    stim2.fill = Color(r2, g2, b2);//使う形の色(明るさ)


```

```
56      /*刺激提示**********************/
57      double bg_lum;
58
59      bg_lum = 0.5;//背景輝度
60
61      while(!Keyboard::esc.pushed())//while文(中の条件になるまで繰り返す)
62      {
63          display.clear(Color(bg_lum, bg_lum, bg_lum));//画面は一回ごとにきれいにして・・・
64          stim1.draw();//画面に描くもの
65          stim2.draw();//画面に描くもの
66          display.flip();//フリップする(表示する)
67      }
68
69
70  }
```

ポイント

・複数の形を提示したい場合には，同じコードをコピペすれば描けます．

・別の形を使用したいときには，Psychlosp::Rectangleの部分を変更してみましょう

４．たくさん描いてみよう２

・コピペをするだけ，とは言っても数が多くなると大変です．
・他の書き方（配列）を使ってみましょう．

```
#include <psychlops.h>
using namespace Psychlops;

void psychlops_main()
{
    /*画面設定**********************/
    int window_width;//提示するwindowの幅
    int window_height;//提示するwindowの高さ
    Canvas display;//プログラム用に画面を確保

    window_width = 1024;//提示するwindowの幅
    window_height = 768;//提示するwindowの高さ
    display.set(window_width, window_height,Canvas::window);//提示画面を準備する


    /*刺激描画**********************/
    Psychlops::Rectangle stim1[2];//使う形（配列によって２つにしている）

    double r1[2], g1[2], b1[2];//使う形の色(stim1)
    int stim1_sizeH[2], stim1_sizeY[2];//使う形のサイズ(stim1)
    int stim1_posiX[2], stim1_posiY[2];//使う形の位置(stim1)

    r1[0] = 0.0;
    g1[0] = 1.0;
    b1[0] = 0.0;//使う形の色(stim1)

    stim1_sizeH[0] = 50;
    stim1_sizeY[0] = 50;//使う形のサイズ(stim1)

    stim1_posiX[0] = 0;
    stim1_posiY[0] = 20;//使う形の位置(stim1)

    stim1[0].set(stim1_sizeH[0],stim1_sizeY[0]);//使う形のサイズ(x,y)
    stim1[0].centering().shift(stim1_posiX[0],stim1_posiY[0]);//中央寄せして位置を移動する
    stim1[0].fill = Color(r1[0], g1[0], b1[0]);//使う形の色(明るさ)

    r1[1] = 0.0;
    g1[1] = 1.0;
    b1[1] = 0.0;//使う形の色(stim1)

    stim1_sizeH[1] = 50;
    stim1_sizeY[1] = 50;//使う形のサイズ(stim1)

    stim1_posiX[1] = -100;
    stim1_posiY[1] = 20;//使う形の位置(stim1)

    stim1[1].set(stim1_sizeH[1],stim1_sizeY[1]);//使う形のサイズ(x,y)
    stim1[1].centering().shift(stim1_posiX[1],stim1_posiY[1]);//中央寄せして位置を移動する
    stim1[1].fill = Color(r1[1], g1[1], b1[1]);//使う形の色(明るさ)


    /*刺激提示**********************/
    double bg_lum;

    bg_lum = 0.5;//背景輝度
```

```
56
57      while(!Keyboard::esc.pushed())//while文(中の条件になるまで繰り返す)
58      {
59          display.clear(Color(bg_lum, bg_lum, bg_lum));//画面は一回ごとにきれいにして・・・
60          stim1[0].draw();//画面に描くもの
61          stim1[1].draw();//画面に描くもの
62          display.flip();//フリップする(表示する)
63      }
64
65
66  }
```

ポイント

・配列とは，Excelの表の座標が数字になったものと思って下さい

(A列 = 変数名[0]，B列 =変数名[1]，C列 =変数名[2]…)

	A	B	C	D	E
1	vari1[0]	vari1[1]	vari1[2]	vari1[3]	vari1[4]

・配列は，使いたい変数の名前の後ろに[ ]をつけて，使いたい数を書きます．

・初期化の時には，使いたい番号を入れますが，一番目は[0]，二番目は[1]，三番目は[2]…という風に 1 を引いた値によって使いたい番号を入れます（配列の中身を参照する，と言ったりします）．

5．たくさん描いてみよう３

・配列を使ってもコピペとほとんど変わらない…？
・繰り返しの作業をforを使って書いてみましょう

```
#include <psychlops.h>
using namespace Psychlops;

void psychlops_main()
{
    /*画面設定**********************/
    int window_width;//提示するwindowの幅
    int window_height;//提示するwindowの高さ
    Canvas display;//プログラム用に画面を確保

    window_width = 1024;//提示するwindowの幅
    window_height = 768;//提示するwindowの高さ
    display.set(window_width, window_height,Canvas::window);//提示画面を準備する


    /*刺激描画**********************/
    Psychlops::Rectangle stim1[2];//使う形（配列によって２つにしている）

    double r1[2] = {0.0, 0.0}, g1[2] = {1.0, 1.0}, b1[2] = {0.0, 0.0};//使う形の色(宣言＋初期化)
    int stim1_sizeH[2] = {50, 50}, stim1_sizeY[2] = {50, 50};//使う形のサイズ(宣言＋初期化)
    int stim1_posiX[2] = {0, -100}, stim1_posiY[2] = {20, 20};//使う形の位置(宣言＋初期化)

    for (int I = 0; i<2; i++)
    {
        stim1[i].set(stim1_sizeH[i],stim1_sizeY[i]);//使う形のサイズ(x,y)
        stim1[i].centering().shift(stim1_posiX[i],stim1_posiY[i]);//中央寄せして位置を移動する
        stim1[i].fill = Color(r1[i], g1[i], b1[i]);//使う形の色(明るさ)
    }


    /*刺激提示**********************/
    double bg_lum;

    bg_lum = 0.5;//背景輝度

    while(!Keyboard::esc.pushed())//while文(中の条件になるまで繰り返す)
    {
        display.clear(Color(bg_lum, bg_lum, bg_lum));//画面は一回ごとにきれいにして・・・
        for (int i = 0; i<2; i++)
        {
            stim1[i].draw();//画面に描くもの
        }
        display.flip();//フリップする(表示する)
    }


}
```

ポイント

・変数が配列の時　変数名[大きさ] = {要素1，要素2，…}として，宣言と初期化をすることができます．
・for文には，１繰り返す，２複数ある配列の番号を示す，という二つの役割があります．
・書き方の決まりは　　for (宣言＆初期化，範囲，変化量){繰り返したい内容}　　です．
・「++」と書くと，その部分に進む度に＋１されます．（「--」だと－１されます）．「+=（数値）」で好きな数値の大きさを足していくことができます．

6．画像を表示してみよう１

・プログラムを使って簡単な図形が出せるようになりました．今度は写真などの画像を提示する方法を試してみましょう．

```
#include <psychlops.h>
using namespace Psychlops;

void psychlops_main()
{
    /*画面設定**********************/
    int window_width;//提示するwindowの幅
    int window_height;//提示するwindowの高さ
    Canvas display;//プログラム用に画面を確保

    window_width = 1024;//提示するwindowの幅
    window_height = 768;//提示するwindowの高さ
    display.set(window_width, window_height,Canvas::window);//提示画面を準備する


    /*刺激描画**********************/
    Psychlops::Image stim1;//使う形(Imageを使う)
    stim1.load("anzu1.png");//画像のある場所を指定する
    stim1.centering();//中央寄せ


    /*刺激提示**********************/
    double bg_lum;

    bg_lum = 0.5;//背景輝度

    while(!Keyboard::esc.pushed())//while文(中の条件になるまで繰り返す)
    {
        display.clear(Color(bg_lum, bg_lum, bg_lum));//画面は一回ごとにきれいにして・・・
        stim1.draw();//画面に描くもの
        display.flip();//フリップする(表示する)
    }


}
```

ポイント

・Psychlops::Imageと書くと，画像を使うということを伝えることができます．
・.load(“画像の場所”)で画像が置いてある場所を伝えることができます．
・提示する方法は図形と同じです．

7．画像を表示してみよう２

・複数の画像を表示する時も配列を使うことができます．さらに文字も変数として使えます．

```
#include <psychlops.h>
using namespace Psychlops;

void psychlops_main()
{
    /*画面設定**********************/
    int window_width;//提示するwindowの幅
    int window_height;//提示するwindowの高さ
    Canvas display;//プログラム用に画面を確保

    window_width = 1024;//提示するwindowの幅
    window_height = 768;//提示するwindowの高さ
    display.set(window_width, window_height,Canvas::window);//提示画面を準備する


    /*刺激描画**********************/
    Psychlops::Image stim1[2];//使う形
    char load_file_name[2][255]; //文字に関する配列　XXX[使う数][文字のバイト数(255)]
    int image_position[2];//画像を提示する位置を配列で作っておく

    image_position[0] = -100;//画像を提示する位置を配列で作っておく
    image_position[1] = 100;//画像を提示する位置を配列で作っておく

    for (int i = 0; i<2; i++)
    {
        sprintf(load_file_name[i],"anzu%d.png",i+1);//ファイル名をload_file_nameに登録する
        stim1[i].load(load_file_name[i]);//画像のある場所を指定する
        stim1[i].centering().shift(image_position[i],0);//中央寄せして位置を移動する
    }


    /*刺激提示**********************/
    double bg_lum;

    bg_lum = 0.5;//背景輝度

    while(!Keyboard::esc.pushed())//while文(中の条件になるまで繰り返す)
    {
        display.clear(Color(bg_lum, bg_lum, bg_lum));//画面は一回ごとにきれいにして・・・
        for (int i = 0; i<2; i++)
        {
            stim1[i].draw();//画面に描くもの
        }
        display.flip();//フリップする(表示する)
    }


}
```

ポイント

・文字を変数として使う時にはcharを使います．変数名[使用する数][文字(のバイト)数]　と書きます．
・sprintfを使うと連続する番号のついた画像を読み込み時に便利です．

8．条件を設定してみよう

・ある特定の条件にあてはまった時だけ，書かれていることを実行するための方法です．

```
#include <psychlops.h>
using namespace Psychlops;

void psychlops_main()
{
    /*画面設定**********************/
    int window_width = 1024;//提示するwindowの幅(宣言＋初期化)
    int window_height = 768;;//提示するwindowの高さ(宣言＋初期化)
    Canvas display;//プログラム用に画面を確保

    display.set(window_width, window_height,Canvas::window);//提示画面を準備する


    /*刺激描画**********************/
    Psychlops::Image stim1[2];//使う形
    char load_file_name[2][255]; //文字に関する配列　XXX[使う数][文字のバイト数(255)]
    int image_position[2] = {-100, 100};//画像を提示する位置を配列で作っておく(宣言＋初期化)

    for (int i = 0; i<2; i++)
    {
        sprintf(load_file_name[i],"anzu%d.png",i+1);//ファイル名をload_file_nameに登録する
        stim1[i].load(load_file_name[i]);//画像のある場所を指定する
        stim1[i].centering().shift(image_position[i],0);//中央寄せして位置を移動する
    }


    /*刺激提示**********************/
    double bg_lum;//背景輝度
    int stim_flag;//条件を分けるための変数(フラグ)

    bg_lum = 0.5;//背景輝度
    stim_flag = 0;//条件を分けるための変数(フラグ)

    while(!Keyboard::esc.pushed())//while文(中の条件になるまで繰り返す)
    {
        display.clear(Color(bg_lum, bg_lum, bg_lum));//画面は一回ごとにきれいにして・・・

        if (stim_flag == 0)
        {
            stim1[0].draw();//画面に描くもの
        }
        if (stim_flag == 1)
        {
            stim1[1].draw();//画面に描くもの
        }
        if (stim_flag == 2)
        {
            stim1[0].draw();//画面に描くもの
            stim1[1].draw();//画面に描くもの
        }
        display.flip();//フリップする(表示する)
    }


}
```

56

ポイント

・If文は「もしも〇〇だったらこっちに進んでほしい」という書き方です.

```
If(条件)
{
何をするのか
}
else if(条件)    //それ以外の別の条件だったら
{
何をするのか
}
else    //上記以外だったら
{
何をするのか
}
```

という風にもかけます

・条件には同じだったら（==）,異なったら（!=）といった書き方をします（論理演算子と言います）
・実はwhile(条件){}も同じような使い方です.

・switch文は「〇〇と一致するところに進んでほしい」という書き方です.

```
switch(変数)
{
case 0:
  何をするのか
  break;
case 1:
  何をするのか
break;
……（以下同様）
}
```

・case の後ろにくる数値は，変数がとり得る値を書きます.

9．表示時間を設定してみよう

・実験では限られた時間だけ刺激を提示することが多いです.
・時間を設定するいくつかの方法を学びます.

```
#include <psychlops.h>
using namespace Psychlops;

void psychlops_main()
{
    /*画面設定**********************/
    int window_width = 1024;//提示するwindowの幅(宣言＋初期化)
    int window_height = 768;;//提示するwindowの高さ(宣言＋初期化)
    Canvas display;//プログラム用に画面を確保

    display.set(window_width, window_height,Canvas::window);//提示画面を準備する


    /*刺激描画**********************/
    Psychlops::Image stim1[2];//使う形
    char load_file_name[2][255]; //文字に関する配列　XXX[使う数][文字のバイト数(255)]
    int image_position[2] = {-100, 100};//画像を提示する位置を配列で作っておく(宣言＋初期化)

    for (int i = 0; i<2; i++)
    {
        sprintf(load_file_name[i]," anzu%d.png",i+1);//ファイル名をload_file_nameに登録する
        stim1[i].load(load_file_name[i]);//画像のある場所を指定する
        stim1[i].centering().shift(image_position[i],0);//中央寄せして位置を移動する
    }


    /*刺激提示**********************/
    double bg_lum;

    bg_lum = 0.5;//背景輝度

    Clock timer;//Clock タイマー型
    double start_t, end_t;//タイマーの値を入れる変数

    int frame;//フレーム数を数えるための変数

    //制御方法１
    timer.update();//タイマーの初期化(現在の時刻を取得する)
    start_t = timer.at_msec();//現在の時刻を変数とする
    while (end_t-start_t<1000)//while文(中の条件になるまで繰り返す)
    {
        display.clear(Color(bg_lum, bg_lum, bg_lum));//画面は一回ごとにきれいにして・・・
        stim1[0].draw();//画面に描くもの
        display.flip();//フリップする(表示する)

        timer.update();//タイマーの初期化(現在の時刻を取得する)
        end_t = timer.at_msec();//現在の時刻を変数とする
    }

```

```
50      //制御方法2
51      frame = 0;//フレームを数えるための数値の初期値(ゼロから数える)
52      while(frame<60)//while文(中の条件になるまで繰り返す)
53      {
54          display.clear(Color(bg_lum, bg_lum, bg_lum));//画面は一回ごとにきれいにして・・・
55
56          stim1[1].draw();//画面に描くもの
57          display.flip();//フリップする(表示する)
58          frame++;//frameを一ずつ足す
59      }
60
61      //制御方法3
62      display.clear(Color(bg_lum, bg_lum, bg_lum));//画面は一回ごとにきれいにして・・・
63      stim1[0].draw();//画面に描くもの
64      stim1[1].draw();//画面に描くもの
65      display.flip();//フリップする(表示する)
66
67      timer.update();//タイマーの初期化(「現在」を取得する)
68      start_t = timer.at_msec();//「現在」の値
69      while (end_t-start_t<500)//while文(中の条件になるまで繰り返す)
70      {
71          timer.update();//タイマーの初期化(「現在」を取得する)
72          end_t = timer.at_msec();//「現在」の値
73      }
74
75      while(!Keyboard::esc.pushed())//while文(中の条件になるまで繰り返す)
76      {
77          display.clear(Color(bg_lum, bg_lum, bg_lum));//画面は一回ごとにきれいにして・・・
78          display.flip();//フリップする(表示する)
79      }
80
81
82  }
83
```

ポイント

・提示される画像は，実はパラパラ漫画のように提示されています．
画像の表示のときにflip（めくる）と書くのはそのためです
・flipのタイミングは，ディスプレイのリフレッシュレートによって決まっています．
リフレッシュレートとは１秒間に画面が何回切り替わるかを示す数値です（垂直同期周波数とも言います）
多くの液晶ディスプレイでは60 Hzになっています．
・**while**(frame<60) によってflipが60回未満であれば{ }内に書いたことを繰り返すことを示しています．
60 Hzのリフレッシュレートの液晶ディスプレイでは１秒間提示しなさい，ということを意味します．

・Clockと書いて宣言される変数がtimerです．
・timerはストップウォッチのように使います．.updateでボタンを押すことで，押した時間がわかります．
ただし，そのままでは見ることができないので，別に用意していた変数にミリ秒単位の数字で代入します．

・**while**文の中では，常にボタンを押し続けて，押した時間を記録しています．そして，**while**文の外で押されたボタンの時間と常に比較しています．
・ずっと比較をしていてある条件になったら（ここでは500ミリ秒よりも大きくなったら），繰り返しをやめます．

10. 文字を表示してみよう

サイズや太さなども変更して文字を表示してみましょう

```
#include <psychlops.h>
using namespace Psychlops;

void psychlops_main()
{
    /*画面設定**********************/
    int window_width = 1024;//提示するwindowの幅(宣言＋初期化)
    int window_height = 768;;//提示するwindowの高さ(宣言＋初期化)
    Canvas display;//プログラム用に画面を確保

    display.set(window_width, window_height,Canvas::window);//提示画面を準備する


    /*刺激描画**********************/
    Font font1(40, Font::bold);//フォントの大きさ，太さ
    Font font2(25, Font::bold, Font::italic);//フォントの大きさ，太さ，スタイル

    Letters let1(L"位置も色も変わります。");
    Letters let2(L"大きさも太さ変わります。", font1);
    Letters let3(L"A Latin alphabet can be used.", font2);


    /*刺激提示**********************/
    double bg_lum;//背景輝度

    bg_lum = 0.5;

    while(!Keyboard::esc.pushed())//while文(中の条件になるまで繰り返す)
    {
        display.clear(Color(bg_lum, bg_lum, bg_lum));//画面は一回ごとにきれいにして・・・
        let1.centering().shift(-200,-100).draw(Color(1.0, 0.0, 1.0));//位置と色の指定
        let2.centering().shift(0,-50).draw(Color(0.0, 1.0, 1.0));//位置と色の指定
        let3.centering().shift(0,200).draw(Color(0.0, 0.0, 0.0));//位置と色の指定
        display.msg(let1,0,0);//文字を提示
        display.msg(let2,0,0);//文字を提示
        display.msg(let3,0,0);//文字を提示
        display.flip();//フリップする(表示する)
    }


}
```

ポイント

・Fontで宣言をした変数を，Lettersで宣言をした変数の中で使うことができます．
・Lettersで宣言した変数は，centering()，shift()など図形と同じように位置を指定することができます．

11. マウス，キー入力

マウスやキーボードを使って入力する方法です．

```
#include <psychlops.h>
using namespace Psychlops;

void psychlops_main()
{
    /*画面設定**********************/
    int window_width = 1024;//提示するwindowの幅(宣言＋初期化)
    int window_height = 768;;//提示するwindowの高さ(宣言＋初期化)
    Canvas display;//プログラム用に画面を確保

    display.set(window_width, window_height,Canvas::window);//提示画面を準備する


    /*刺激描画**********************/
    Psychlops::Rectangle stim1[2];//使う形（配列によって２つにしている）

    double r1[2] = {0.0, 0.0}, g1[2] = {1.0, 1.0}, b1[2] = {0.0, 0.0};//使う形の色(stim1)
    int stim1_sizeH[2] = {50, 50}, stim1_sizeY[2] = {50, 50};//使う形のサイズ(stim1)
    int stim1_posiX[2] = {0, -100}, stim1_posiY[2] = {20, 20};//使う形の位置(stim1)

    for (int i = 0; i<2; i++)
    {
        stim1[i].set(stim1_sizeH[i],stim1_sizeY[i]);//使う形のサイズ(x,y)
        stim1[i].centering().shift(stim1_posiX[i],stim1_posiY[i]);//中央寄せして位置を移動する
        stim1[i].fill = Color(r1[i], g1[i], b1[i]);//使う形の色(明るさ)
    }


    /*刺激提示**********************/
    double bg_lum;//背景輝度

    bg_lum = 0.5;

    Input::refresh();//キー入力をリセットする
    while(!Keyboard::esc.pushed())
    {
        display.clear(Color(bg_lum, bg_lum, bg_lum));//画面は一回ごとにきれいにして・・・

        stim1[0].centering(Mouse::x, Mouse::y);//使う形の位置(マウスの位置を中心にする)

        if (Mouse::left.pushed())//マウスの左ボタンを「押した」とき
        {
            stim1[0].fill = Color(1.0,0.0,0.0);//色の指定
        }
        if (Mouse::left.released())//マウスの左ボタンを「放した」とき
        {
            stim1[0].fill = Color(r1[0], g1[0], b1[0]);//色の指定
        }

        for (int i =0; i<2; i++)
        {
            stim1[i].draw();//画面に描くもの
        }
        display.flip();//フリップする(表示する)
    }

```

```
57
58  }
59
```

ポイント

・Mouse::x，Mouse::yでマウスの水平位置，垂直位置を知ることができます．
・Mouse::left，Mouse::rightはマウスの左ボタン，右ボタンに対応します．
・Keyboard::（キーボード名前）の名前と書くことで，使うキーボードを指定します．
・どちらの場合もpushed()は押したとき， pressed()は押されているとき，released ()は離したときを表しています．

１２．反応時間を取ってみよう

これまでのパーツを組み合わせた簡単な実験プログラムです．
まずは，どのような構成になっているのかプログラムを読んでみましょう

```
#include <psychlops.h>
using namespace Psychlops;

void psychlops_main()

{
    /*画面設定**********************/
    int window_width = 1024;//提示するwindowの幅(宣言＋初期化)
    int window_height = 768;;//提示するwindowの高さ(宣言＋初期化)
    Canvas display;//プログラム用に画面を確保

    display.set(window_width, window_height,Canvas::window);//提示画面を準備する

    /*実験条件**********************/
    const int max_trialnumber = 10;//全試行数


    /*刺激描画**********************/
    Psychlops::Rectangle fixation[2];//注視点
    Psychlops::Rectangle stim1;//刺激１(正方形)
    double r = 0.0, g = 0.5, b = 1.0;//刺激の色を指定する変数
    int stim_size = 100;//刺激のサイズ

    stim1.set(stim_size,stim_size);//刺激の大きさの指定
    stim1.centering();//中心に寄せる
    stim1.fill = Color(r, g, b);//色の指定

    fixation[0].set(5,1).centering();//注視点(横棒)
    fixation[0].fill = Color(0.0,0.0,0.0);//色の指定

    fixation[1].set(1,5).centering();//注視点(縦棒)
    fixation[1].fill = Color(0.0,0.0,0.0);//色の指定


    /*刺激提示**********************/
    Clock timer_rt;//Clock タイマー型
    double start_t, end_t;//タイマーの値を入れる変数

    double bg_lum = 0.5;//背景輝度の指定

    int blank_duration;//ブランクの長さのための変数
    int frames;//フレーム数を数えるための変数

    int result[max_trialnumber];//反応を入れるための配列
    double result_rt[max_trialnumber];//反応時間を入れるための配列

    while(!Keyboard::spc.pushed())
    {
        display.clear(Color(bg_lum,bg_lum,bg_lum));//画面は一回ごとにきれいにして・・・
        display.msg("Push space key to start.",window_width/2,380,Color::white);//文字を提示
        display.flip();//フリップする(表示する)
    }

    for(int trialNow=0 ; trialNow<max_trialnumber; trialNow++)//試行回数分繰り返す
    {
```

```
56          Input::refresh();//キー入力をリセットする
57
58          blank_duration = Psychlops::random(60)+30;//予測をさせないためにブランクの時間をランダムにする
59
60          frames = 0;//フレームを数えるための数値の初期値(ゼロから数える)
61          while( frame < blank_duration)//ブランクの長さ(大きさ)になるまでframeを足し続ける
62          {
63              display.clear(Color(bg_lum, bg_lum, bg_lum));//画面は一回ごとにきれいにして・・・
64              for(int i = 0; i<2; i++)
65              {
66                  fixation[ i ].draw();//画面に描くもの(注視点)
67              }
68              display.flip();//フリップする(表示する)
69              frame++;//frameを一ずつ足す
70          }
71          Input::refresh();//キー入力をリセットする
72
73          display.clear(Color(bg_lum, bg_lum, bg_lum));//画面は一回ごとにきれいにして・・・
74          stim1.draw();//画面に描くもの(正方形)
75          display.flip();//フリップする(表示する)
76
77
78          /*反応取得**********************/
79          timer_rt.update();//タイマーの初期化(「現在」を取得する)
80          start_t = timer_rt.at_msec();//「現在」の値(=キーを押す前の時間)
81          while(true)//ずっと繰り返す(breakがきたら繰り返しを終わる)
82          {
83              if (Keyboard::up.pushed())//左カーソルキーが押されたら
84              {
85                  result[trialNow] = 1;//反応は１
86                  timer_rt.update();//タイマーの初期化(「現在」を取得する)
87                  break;//while文から脱出
88              }
89          }
90          end_t = timer_rt.at_msec();//「現在」の値(=キーを押したときの時間)
91          result_rt[trialNow] = end_t-start_t;//反応時間　=　押した後の「現在」ー押す前の「現在」
92
93      }
94
95
96      /*データ保存**********************/
97      Input::refresh();
98      int number[max_trialnumber];
99      for (int i = 0; i<max_trialnumber; i++)
100     {
101         number[ i ] = i+1;
102     }
103     Data::savearray("EXP_simpleRT_%TIME_.txt", "Num¥tJudgement¥tRT¥tA", max_trialnumber, number,
                                                                                result, result_rt);
104
105     Input::refresh();
106     while(!Keyboard::spc.pushed())
107     {
108         display.clear(Color(bg_lum,bg_lum,bg_lum));//画面は一回ごとにきれいにして・・・
109         display.msg("Thanks. Push spc key to quit.",500,380,Color::white);//文字を提示
110         display.flip();//フリップする(表示する)
111
112     }
113
114 }
```

ポイント

・実験プログラムの基本パーツ（復習）
　　画面設定，条件設定，刺激準備（描画），刺激提示，反応取得，データ保存，という構成です

・条件設定では，試行回数や条件の割り振り方などを設定します．

・刺激提示では全試行を繰り返し提示するためのfor文でできています．

・あらかじめデータを入れておきたい配列を，全試行数の大きさで宣言しておきます

```
int result[max_trialnumber];
double result_rt[max_trialnumber];
```

・反応取得では，キー入力をif文で判定し，キーが押された時にresult[]という配列に数字を入れています．
　　・while(true){}はbreak;が現れない限り，ずっと繰り返しつづけます．
　　・while(true){}のすぐ上と中でtimerが使われています．while文に入る前の時間と，キー入力があった時点での時間を引き算することで，反応時間を計算しています．（result_rt[]という配列にその結果を入れています）

・データの保存

```
Data::savearray("ファイル名", "ファイルの一行目のラベル", データ数, 配列1, 配列2, 配列3, …);
```

　　・配列１には，試行番号（1から最後まで）を入れています
　　・配列２には result（反応），配列２にはresult_rt（反応時間）を入れています．

１３．プログラムの「読み方」再び

ここまでの学習した内容を元に，４ページ目から掲載されている実験プログラムをもう一度読んでみましょう．

14. 実験プログラムを眺めよう

・ここまでの話の中で使っていないパーツは？

15. 条件の割り振り方

・ここはやや複雑です．Web教材で実際に出力をみながら学びましょう

```
1   /*実験条件**********************/
2   const int independent_var1_step_number = 2; //実験要因１
3   const int independent_var2_step_number = 3;//実験要因２
4
5   double independent_var1_step[independent_var1_step_number]= {10, 100};//要因１のパラメタ
6   double independent_var2_step[independent_var2_step_number]= {-50, 0, 50};//要因２のパラメタ
7
8   int repetition = 10;//一試行あたりの繰り返し数
9   const int max_trialnumber =
10            independent_var1_step_number*independent_var2_step_number*repetition;//全試行数
11
12  double independent_var1_trial[max_trialnumber];
13  double independent_var2_trial[max_trialnumber];
14
15  int var1_condition_number[max_trialnumber];//要因１の条件番号を格納（0, 1）
16  int var2_condition_number[max_trialnumber];//要因２の実験番号を格納（0, 1, 2）
17  int shuffled_trialnumber[max_trialnumber];
18
19  for(int i = 0; i<max_trialnumber; i++)
20  {
21      shuffled_trialnumber[i] = i; //全試行分の配列を用意
22  }
23  Math::shuffle(shuffled_trialnumber, max_trialnumber); //バラバラにする
24
25  for(int trialnumber=0; trialnumber<max_trialnumber; trialnumber++)
26  {
27      var1_condition_number[trialnumber] = shuffled_trialnumber[trialnumber] %
                                                    independent_var1_step_number;
28      independent_var1_trial[trialnumber] =
                        independent_var1_step[var1_condition_number[trialnumber]];
29
30      var2_condition_number[trialnumber] = floor(shuffled_trialnumber[trialnumber] /
                        independent_var1_step_number) % independent_var2_step_number;
31      independent_var2_trial[trialnumber] =
                        independent_var2_step[var2_condition_number[trialnumber]];
32  }
```

ポイント

・各実験要因ごとに，刺激のサイズや位置など，実際に用いるパラメタが入った配列を用意します．
・全試行回数の大きさを持った配列を用意し，各試行でどの条件を出すのかをあらかじめ決めておきます．
　　実験番号が入っている配列　var1_condition_number[], var2_condition_number[]
　　実験パラメタが入っている配列，independent_var1_trial[],independent_var2_trial[]
・配列の中身を配列によって指定することもできます．

・こうした配列が，刺激描画や刺激提示の時にされています．

１６．測定の方法

・測定する心理量は自分で定義する必要があります．
　検出閾，弁別閾，主観的等価点などがよく使われる指標です．

・心理量の測定方法には，恒常法，調整法，極限法（階段法）などがあります．

恒常法
　様々な強さの刺激を準備する．
　それらをランダムに提示して，それぞれに対する反応を繰り返し測定(20-2000回)

調整法
　基準となる刺激に対して比較する刺激の「大きさ」や「強さ」などをそろえる

極限法
　例えば，光が見える最小の強さを測定する場合，全く光が見えない強度から徐々に刺激強度をあげていき（上昇系列），「見えた」と判断できた刺激強度を記録する．次に光が確実に見える強度から徐々に刺激強度を下げていき（下降系列），「見えない」と判断された刺激強度を記録する．これらを数回繰り返して平均を取る．これらを一試行ごとに連続的に行なう方法もある（階段法）

１７．実験プログラムをもう一度眺めてみよう．

・恒常法と調整法のプログラムを比較して，どこが違うかを確認しましょう．
・刺激描画，刺激提示のパーツの中に条件の割り振りのパーツで使った変数が入っています
　それぞれどのようなプログラムとなっているのか，グループで話し合ってみましょう．
　最後にわかったこと，わからなかったことを報告しましょう．

18. 実験プログラムを作成しよう

昨日までに学んだ実験プログラムの構成，パーツの種類，役割，つなぎ方を参考にしながらグループで独自の実験プログラムを作成してみよう．

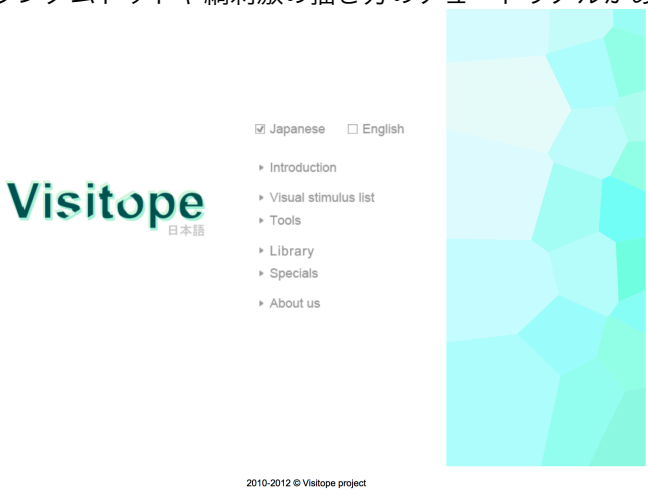
・実験計画を話し合ってみよう
　どんな条件を設定する？どんな刺激を提示する？どんな反応をとる？

・プログラムにその計画を再現してみよう
　どんな役割のパーツが必要？刺激の描き方は？反応はどうやってとる？

参考資料１

Visitope http://visitope.org/

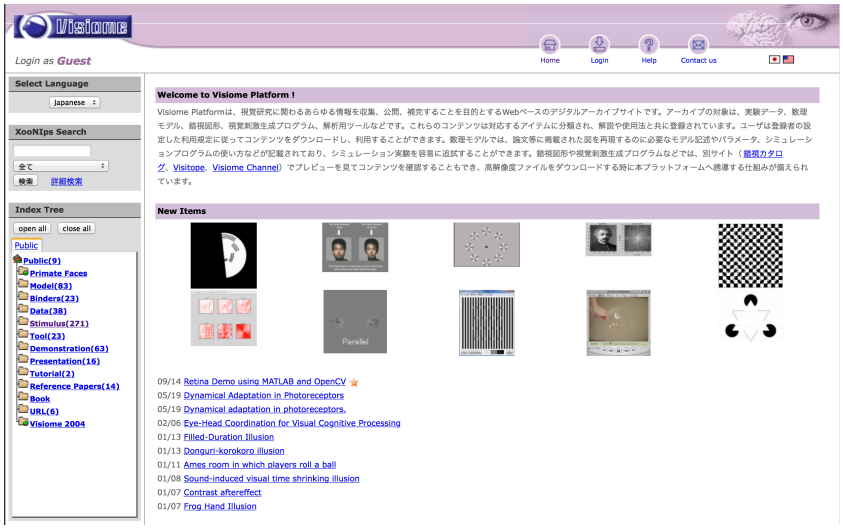
Visual stimulus listの中には様々な刺激の例があります．C++のプログラムコードをダウンロードできます．Toolsの中にはランダムドットや縞刺激の描き方のチュートリアルがあります．



参考資料２

Visiome platform https://visiome.neuroinf.jp

Psychlopsで作成した錯視や刺激が登録されています．左側の検索窓にPsychlopsと入力してプログラムコードを探すことができます．



1

VSJ　Programming Workshop　2015

2015年10月17日　初版発行

著者・発行者
　視覚心理実験プログラミングワークショップ運営委員会
　URL：　http://visitope.org/workshop/ProgrammingWorkshop/index.html
　E-mail（連絡先）：　programmingws2015open@googlegroups.com
共催
　日本視覚学会若手の会
後援
　NIJC・Visiome platform