Processing INTRODUCTION

2014コンピューター・アート

Processingのインストール

http://processing.org/download/index.html
このサイトから最新のバージョンがダウンロード可能

記号の読み方

/ スラッシュ	' シングルクオート	\ バックスラッシュ （日本語環境では¥で表示）	; セミコロン
// ダブルスラッシュ	" ダブルクオート	~ チルダ	{ 左中カッコ
* アスタリスク	& アンド	@ アットマーク	} 右中カッコ
, カンマ	. ピリオド	: コロン	< 大なり
; セミコロン	! エクスクラメーション	| バー	> 小なり

プログラム実行方法

3通りのプログラム実行方法があります

○テストで実行する

画面左上の「RUN」ボタンを押す。

○インターネット上で動作するアプリケーションを制作する

「Export」ボタンを押すと保存フォルダ内に「applet」フォルダが作られる。その中のファイルをアップロードすることにより、インターネット上で閲覧することができる。

○単体で動作するアプリケーションを制作する

メニュー「file」から「Export Application」を選択すると、ファイルの保存フォルダ内に「application.linux」　「application.macosx」「application.windows」というフォルダが制作される。このフォルダ内のファイルはmac os ,windows, linuxのOSに最適化されており、Processingのソフトがなくても単体で動作させることが可能。

基本命令

1. 画面のサイズを指定する

size(横のピクセル数, 縦のピクセル数) ;

例　/ 縦に480ピクセル、横に480ピクセルの画面サイズを設定する

```
size(640, 480);
```

2. 点を描く

point(x座標, y座標);

例 / 座標(50, 10)に点を描く

```
point(50, 10);
```

3.コメントをつける

// 以下任意の文章

例

```
//テスト
```

ダブルスラッシュ以下はプログラムに影響しない。制作者のメモや一時的にプログラムを無効にする際に使用

4. 直線を描く

line(始点の x 座標, 始点の y 座標, 終点の x 座標, 終点の y 座標);

例 / 座標(200,0) から座標(0,200)まで直線を引く

```
line( 200, 0, 0, 200);
```

5. 長方形を描く

rect(x座標, y座標, 幅, 高さ);

例 / 座標(100,10)に幅20高さ30の長方形を描く

```
rect( 100, 10, 20, 30);
```

6. 円を描く

ellipse(x座標, y座標, 横の直径, 縦の直径);

例 / 座標(100,10)に横20縦30の楕円を描く

```
ellipse( 100, 10, 20, 30);
```

7. 三角形を描く

triangle(x1, y1, x2, y2, x3, y3);

例 / 座標(100,10)と(50,100)と(100,50)を頂点とする三角形を描く

```
triangle(10, 10, 50, 100, 100, 50);
```

8. 四角形を描く

quad (x1, y1, x2, y2, x3, y3, x4, y4);

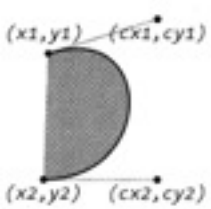
例 / 座標(10,10)と(70,5)と(90,50) と(10,100) を頂点とする四角形を描く

```
quad(10, 10, 70, 5, 90, 50,10,100);
```

9. ベジェ曲線を描く

bezier (x1, y1, cx1, cy1, cx2, cy2, x2, y2);

例 / 座標(32,20)と(80,5)のコントロールポイント、
座標(80,75)と(30,75)のコントロールポイントを結ぶベジェ曲線を描く



```
bezier (32, 20, 80, 5, 80, 75, 30, 75);
```

---

10. 背景に色をつける

background(明度);
または
background(red, green, blue);
数値は0から255の値の範囲内で指定する

例

```
background(0); //黒色背景
```

```
background(255,165,0); //オレンジ色
```

```
background(255); //白色背景
```

色の指定方法について

processingの標準設定では色を0から255までの数値で指定します。
メニューバーの「Tools→ColorSelector」を使用し、色の選択を行って下さい。

---

11. 線に色をつける

stroke(明度);
または
stroke(red, green, blue);
数値は0から255の値の範囲内で指定する

例1 / オレンジ色の線を描く

```
stroke(255,165,0); //オレンジ色
line(10, 10, 60, 50);
```

例2 / オレンジ色と緑色の線を描く

```
stroke(255,165,0); //オレンジ色
line(10, 10, 60, 50);
stroke(0,165,100); //緑色色
line(50, 50, 10, 50);
```

---

12. 線の太さを変える

strokeWeight(線の太さ);
線の太さはピクセル数で指定

例/ 5ピクセルの太さで直線を描く

```
strokeWeight(5);
line(10, 10, 60, 50);
```

---

13. 線を描かない

noStroke();

例/ 輪郭線を描かずに円を描く

```
noStroke();
ellipse(40, 40, 50, 50);
```

---

14. 色を塗りつぶす

fill(明度);
または
fill(red, green, blue);

例/ オレンジ色で塗りつぶした円を描く

```
fill(255,165,0);
ellipse(40, 40, 50, 50);
```

---

15. 色を塗りつぶさない（輪郭線のみ描く）

noFill();

例/ 輪郭線のみの円を描く

```
noFill();
ellipse(40, 40, 50, 50);
```

---

16. 半透明にする

fill(red, green, blue, 透明度);

透明度の項目に0から255までの透明度を設定する。数値が小さいほど透明度が高い。
透明度の指定はfillの他、strokeなど他の命令でも指定可能。

例：一方の円を半透明にして重ねる

```
fill( 0, 88, 255);
ellipse(40, 40, 50,50);
fill( 230, 188, 0, 70);
ellipse(80, 80, 100,100);
```

---

17. 図形のエッジをスムーズにする

smooth();

例：スムーズの有無を二種類の円で比較する

```
smooth();
ellipse(40, 40, 50,50);
noSmooth();
ellipse(60, 60, 50,50);
```

基本命令その2（複数の行で表現する命令）

18. 多角形を描く（連続した線を描く）

beginShape();
vertex(x座標, y座標);
endShape();

曲線を描くためにはvertex命令をbeginShapeとendShape命令の間に記述する必要があります。

```
size(200, 200);
beginShape();
vertex( 20, 20);
vertex( 60, 60);
vertex( 180, 20);
vertex( 20, 180);
vertex( 20, 20);
endShape();
```

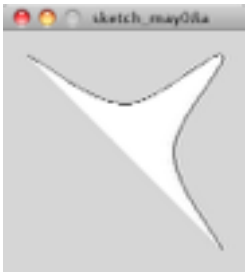
19. 連続した曲線を描く

beginShape();
curveVertex(x座標, y座標);
endShape();

曲線を描くためにはcurveVertex命令をbeginShapeとendShape命令の間に記述する必要があります。
Vertex内の引数を通る滑らかな曲線を描きます。
また、始点と終点の座標は二回ずつ指定します。

例　/ キャンバスのサイズを横200、縦200に設定し、各座標を通過する曲線を作成する

```
size(200, 200);
beginShape();
curveVertex( 20, 20);
curveVertex( 20, 20);
curveVertex( 100, 60);
curveVertex( 180, 20);
curveVertex( 140, 100);
curveVertex( 180, 180);
curveVertex( 180, 180);
endShape();
```



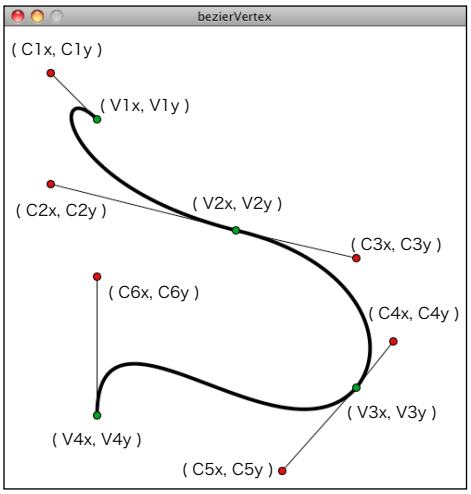
20. 連続したベジェ曲線を描く

beginShape();
bezierVertex(x座標, y座標);
endShape();

ベジェ曲線を連続して描くためにはbezierVertex命令をbeginShapeとendShape命令の間に記述します。

↓

```
size(500,500);
background(255);
noFill();
beginShape();
vertex(100, 100);
bezierVertex(50, 50, 50, 170, 250, 220);
bezierVertex(380, 250, 420, 340, 380, 390);
bezierVertex(300, 480, 100, 270, 100, 420);
endShape();
```



↓

```
beginShape();
vertex(V1x, V1y);
bezierVertex(C1x, C1y, C2x, C2y, V2x, V2y);
bezierVertex(C3x, C3y, C4x, C4y, V3x, V3y);
bezierVertex(C5x, C5y, C6x, C6y, V4x, V4y);
endShape();
```

RGB以外の色彩指定方法

色指定の際、標準ではRGB（Red,Green,Blue)で指定を行うが、HSB(色相Hue・彩度Saturation・輝度Brightness）での指定も可能

○HSBでの色指定

colorMode(HSB);
この命令を宣言後、色指定は（色相Hue、彩度Saturation、輝度Brightness）の数値で指定を行う

```
colorMode(HSB);
fill( 0, 188, 100);
ellipse(80, 80, 100,100);
```

○RGBでの色指定

colorMode(RGB);
この命令を宣言後、色指定は（ Red,Green,Blue）の数値の指定を行う

```
colorMode(RGB);
fill( 0, 188, 100);
ellipse(80, 80, 100,100);
```

○色指定の範囲の変更

colorMode(RGB, 色の範囲);
標準では色の範囲を0から255までの値で指定するが、任意の色の範囲に変更することも可能。
たとえば、 colorMode(RGB, 100);とすると0から100までの色指定に変更することが可能である。

変数・計算

1. 計算する

演算子を使用して計算する

例　/ 引数内での計算

```
rect(200-100, 20*5, 120-20, 120/2);
```

計算記号について

+	**足す (和)**
-	**引く (差)**
*	**掛ける (積)**
/	**割る (商)**
%	**割り算の余り (剰余)**

2. 変数

変数とは数値（データ）を納める入れ物である

①変数を宣言する　(代入も同時に行う)

例

int abc = 10;	変数abcを作り10を代入する
float b = 10.4;	変数bを作り、10.4を代入する
color c1 = color(30,60,99);	変数c1を作り、色の値(30,60,90)を代入する
strings s = “tsukuba”;	変数sを作り文字を代入する

変数を使用する前に、必ず定義すること
変数の名称は英数字であれば任意に設定出来る。大文字小文字は区別される。

②変数へ代入する

例

b = 10;	変数bに10を代入する
b = 10*2;	変数bに10×2を計算した値を代入
b = b + 1;	変数bの今の中身に1を足して代入し直す

備考　特殊な計算式

x++; (x = x + 1; と同義) x--; (x = x - 1; と同義) x += 10; (x = x +10; と同義) x -= 10; (x = x -10; と同義)

③変数を利用する

例

rect(50, 50, a, a);	座標(50,50)に一辺がaの正方形を描く
fill(c1);	変数c1のカラーで塗りつぶす

③小数と整数の相互変換

Processingでは小数と整数は明確に区別されているので扱いに注意すること。小数と整数の変数は相互に代入出来ないので、下記の手順が必要となります

たとえば、

```
int a = 4;
float def = 10.4;
a = def +1;
```

ではエラーになります。そこで

```
int a = 4;
float def = 10.4;
a = (int)(def +1);
```

このようにすれば、aには整数に変換された値が代入されます。

---

3. システム変数

width
　→画面の横幅
height
__→画面の縦幅

上記の変数に数値を代入することはできません。数値を利用することだけが可能です。

繰り返し

for文 指定した回数繰り返す

for(繰り返し変数をつくる ; ループを行う条件 ; 変数の変化){
繰り返しの処理の内容
}

例 / {}の間のline文が10回繰り返される

```
for (int i=0; i<10 ; i=i+1){
line ( i*4 , 10, i*4 + 40, 50); // ここの部分が繰り返される
}
```

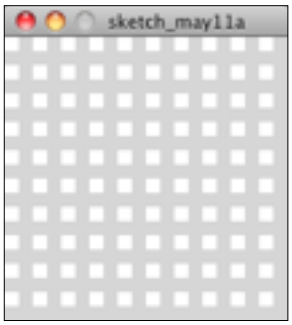
「変数iを0にして、毎回1を足しながら繰り返して、iが10になったら次に進む」

繰り返しを繰り返す

for文の中に、さらにfor文を入れることでさらに複雑な繰り返し処理が可能である

例 / {}の間のrect文が100回繰り返される

```
size(200,200);
noStroke();
for (int y=0; y<10 ; y++){
  for (int x=0; x<10 ; x++){
    rect(x*20, y*20, 10,10);
  }
}
```



一定の条件になるまで繰り返す-while文

while(ループする為の条件){
繰り返しの処理の内容
}

例 / {}の間のrect文が15回繰り返される

```
int a = 0;
while( a < 15){
a ++;
rect( a*5, a*5, 10, 10 );
}
```

条件式

条件によって処理の内容を変える- if ~ else 文

```
if( 条件式 ){
条件式がtrueの場合の処理
}else{
条件式がfalseの場合の処理
}
```

備考: elseは省略可能

例 / for文によって10本の線を引くが、前半の5本と後半の5本で線の太さを変える

```
for (int i=0; i<10 ; i=i+1){

 if ( i < 5){
   strokeWeight(1); //iが5以下の時
   }else{
   strokeWeight(1.3); //iが5以上の時
   }
 line ( i*8 , 10, i*8 + 40, 50);

}
```

条件の記述方法

```
if( i > 5) {処理} //iが 5 より大きい場合
if( i < 5) {処理}  //iが 5 より小さい場合

if( i >= 5) {処理} //iが 5 以上の場合
if( i <=5) {処理} //iが 5 以上場合

if( i == 5) {処理} //iが 5 場合
```

条件分岐にはif文の他にswitch case文がある

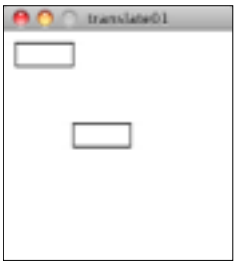
座標命令

1.translate 座標を移動させる

translate(Xの移動量、Yの移動量)；

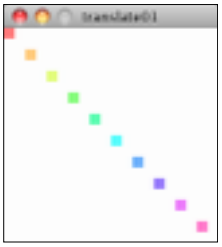
例) 同じ座標にrect命令で矩形を描くが、間にtranslate命令を入れることで原点を移動することが可能

```
size(200,200);
rect(10,10,50,20);
translate(50,70);
rect(10,10,50,20);
```



例)　forを使用し複数回translateを実行する

```
size(200,200);
colorMode(HSB, 100);
background(99);
noStroke();

for(int i = 0; i < 10 ; i++){
  fill(i*10, 60, 99);
  rect(0,0,10,10);
  translate(20, 20);
}
```



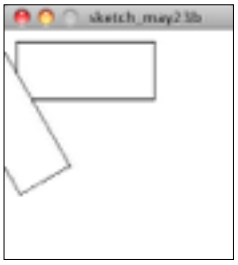
2. rotate 座標を回転させる

rotate(角度のラジアン値)；　もしくは　rotate(radians(角度)；

ラジアンと角度の関係:　2π ラジアン＝360°

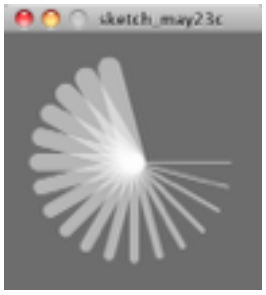
例)　rotateを使用し、矩形を回転させる

```
size(200,200);
rect(10,10,120,50);
rotate(radians(60));
rect(10,10,120,50);
```



例)　rotateとforを使用し、線を回転と同時に線を太くしてゆく（原点は中心で固定）

```
void setup(){
  size(200,200);
  smooth();
  noStroke();
  noLoop();
}

void draw(){
  background(90);
  stroke(255, 120);
  translate(100,100);
  for( int i = 1; i < 19; i++){
    strokeWeight(i);
    line(0 ,0 ,75, 0);
    rotate(radians(15));
  }
}
```



例) 原点を移動させながら回転させる

```
size(200,200);
colorMode(HSB, 120);
background(120);
noStroke();

int angle = 30;
int margin = 40;

translate(120, 30);

for(int i = 0; i< 12; i++){
  fill(i*10, 100, 119, 60);
  rect(0, 0, 30, 30);

  rotate(radians(angle));
  translate(margin, 0);
}
```

3. pushMatrix, popMatrix 座標を戻す

pushMatrix(); 座標をセーブする
popMatrix(); セーブした座標に戻る

例) pushMatrixで現在の座標をセーブし、popMatrixでセーブ時の座標に戻す

```
size(200,200);
background(255);
noFill();
pushMatrix();

for(int i = 0; i<5; i++){
  rect(0,0,15,15);
  translate(40,40);
}

popMatrix();
fill(255,0,0);
ellipse(0,0,20,20);
```

例) pushMatrix、popMatrixの応用例

```
size(200,200);
colorMode(HSB, 300);
background(300);
noFill();
smooth();

int angle = 24;
int x = 3;

translate(width/2, height/2);

for(int i = 0; i < 30 ; i++){
  stroke(i*10,299,299);

  pushMatrix();
  rotate(radians(i*angle));
  translate(i*x, 0);
  rect(0,0,10,10);
  popMatrix();
}
```

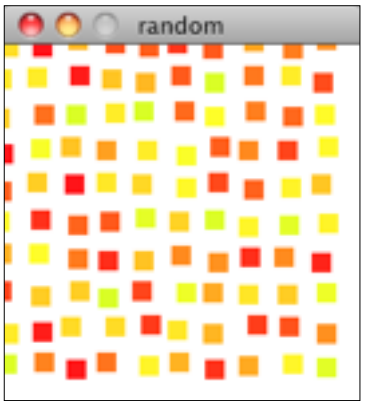
ランダム

random(最大値); 0から最大値までの範囲で乱数をつくる

random(最小値,最大値); 最小値から最大値までの範囲で乱数をつくる

例) randomの応用例

```
size(200, 200);
colorMode(HSB, 255);
background(255);
noStroke();
rectMode(CENTER);

int range = 3;

for(int y=0 ; y<10 ; y++){
  for(int x=0; x<10 ; x++){
    fill(random(50), 255, 255);
    rect(x*20 + random(-range, range),
    y*20 + random(-range, range),10,10);
  }
}
```

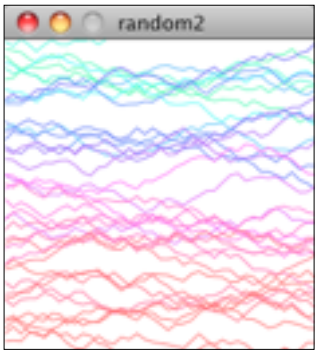


例) randomの応用例

```
background(0);
stroke(255, 60);
float r;
float offset;

for (int i = 0; i< 100; i++){
  r = random(10);
  strokeWeight(r);
  offset = r * 5.0;
  line(i-20, 100, i+offset, 0);
}
```

例) randomの応用例

```
size(200, 200);
colorMode(HSB, 100);
background(100);
noFill();
smooth();

int range_color = 10;
int range_y = 5;
float fluc_color = 50;
float fluc_y;

for(int y =0; y < height ; y+=5){
  fluc_color +=random(-range_color,
   range_color);
  stroke(fluc_color, 60, 99);
  fluc_y = 0;

  beginShape();

  for(int x = 0; x<=width; x +=5){
    fluc_y +=random(-range_y, range_y);
    vertex(x, y+ fluc_y);
  }

  endShape();
}
```

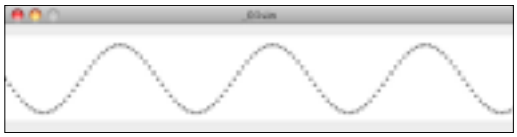


数式による描写

sin(角度のラジアン値)；　もしくは　sin(radians(角度)；

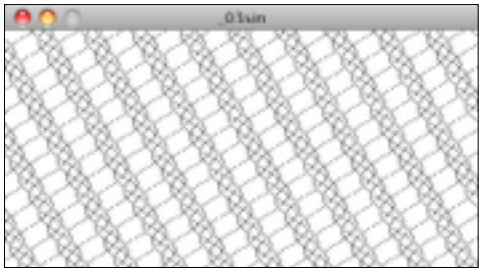
例) sinの例1

```
size(600, 100);
background(255);
smooth();
strokeWeight(3);
float angle = 0.0;

for (int x = 0; x <= width; x +=5){
  float y = 50 + (sin(radians(angle)) * 40.0);
  point(x,y);
  angle += 10;
}
```



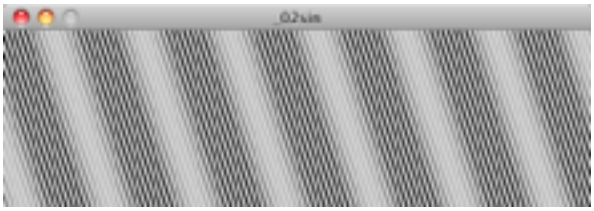
例) sinの例2

```
size(400, 200);
background(255);
fill(255, 20);
noFill();
float angle = 0.0;
float scaleVal = 18.0;
float angleInc = 10;

for (int offset = -10; offset < width+10; offset +=8){
  for(int y = 0; y <= height; y +=2){
    float x = offset + (sin(radians(angle))* scaleVal);
    noStroke();
    stroke(0);
    point(x, y);
    angle += angleInc;
  }
}
```



例) sinの例3

```
size(500, 150);
smooth();
strokeWeight(2);
float angle = 0.0;
float scaleVal = 126.0;
float angleInc = 0.42;

for (int x = -52; x <= width; x +=5){
  float y = 50 + (sin(angle) * scaleVal);
  stroke(y);
  line(x, 0, x+50, height);
  angle += angleInc;
}
```



例) sinの例4

```
size(700, 100);
smooth();
float angle = 0.0;
float amplitude = 50;
float x = 0, y = 0;
float xSpeed = 1;
float frequency = 6.0;
float damping = .994;
strokeWeight(3);

for (int i = 0; i<width; i+=xSpeed){
  x+=xSpeed;
  y = height/2 + sin(radians(angle))*amplitude;
  point(x,y);
  amplitude*=damping;
  angle+=frequency;
}
```

印刷用データの書き出し

例) PDFファイルへ静止画を描き出す (制作: 志原彩未)

```
import processing.pdf.*;

size(750,350, PDF, "kakidasi.pdf" );
colorMode(RGB,100);
background(100,100,100);

float a=0.01;
noFill();
strokeWeight(a);

smooth();
stroke(random(50,100),10,10);

for(int i=0; i<100; i=i+1){

  beginShape();

  curveVertex(random(0,400),0);
  curveVertex(random(0,400),0);
  curveVertex(500,80);
  curveVertex(random(0,80),350);
  curveVertex(random(0,80),350);

  curveVertex(random(680,750),0);
  curveVertex(random(680,750),0);
  curveVertex(100,270);
  curveVertex(750,random(400,680));
  curveVertex(750,random(400,680));

  curveVertex(0,random(280,350));
  curveVertex(0,random(280,350));
  curveVertex(random(250,310),random(250,350));
  curveVertex(random(480,680),350);
  curveVertex(random(480,680),350);

  curveVertex(750,random(280,350));
  curveVertex(750,random(280,350));
  curveVertex(random(250,310),random(250,350));
  curveVertex(random(480,680),350);
  curveVertex(random(480,680),350);

  endShape();
}

exit();
```