KAIT OPEN CAMPUS 2018
PROCESSING を用いた簡単な画像処理体験講座

担当：情報メディア学科　山内

はじめに

Processing はビジュアライジング(アニメーション・インタラクションコンテンツ)に特化したフリーの統合開発環境です。

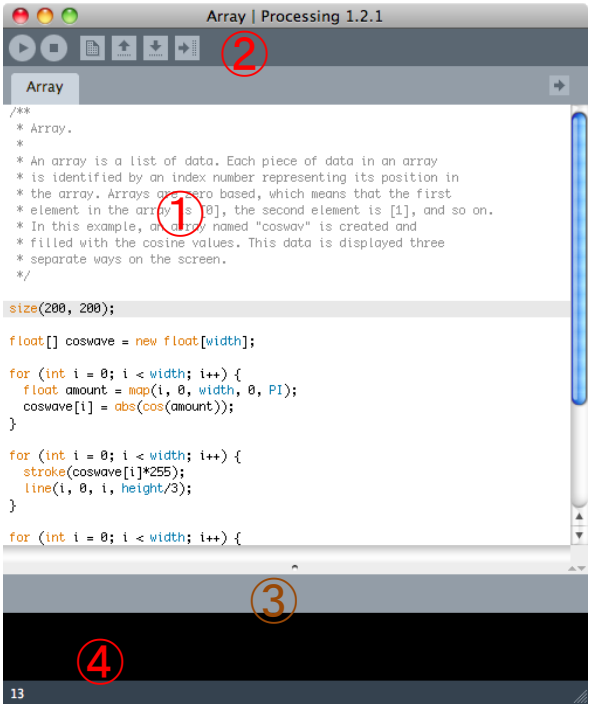
Processingは、メディアアートをはじめとしたビジュアライズに重点を置いた作品を制作する開発ツールとして有名です。初心者にも簡単でわかりやすく、さらに、ビジュアライズな表現をすることができるので、近年では、プログラミング初心者の学習ツールとして多くの大学で導入されはじめています。本講座では、簡単な画像処理を行うプログラムの作成を通じてProcessing言語の基礎を学びます。

本講座のサポートページは「http://www.tosiyama.jp/mini2018/」です。

Processingのダウンロード

Processingはhttp://www.processing.org/download/ からダウンロード出来ます。2018 年 6 月 4 日現在、最新バージョンは 3.3.7 ですが、本講座では 2.2.1 を使います。最新バージョンを試したい方は、家に帰ってから自分(自宅)のパソコンにいれてみてください。

Processinng でやってほしいこと



１．　とにかく実行してみる。
２．　サンプルコードをいじる
３．　なんども１と２を繰り返す。

読むだけでは、プログラミングを学ぶことができません。プログラミングを体験する必要があります。

■Processing の基本画面

Processing の開発環境（PDE）を実行してみると以下のような画面が表示されます。覚えることはそう多くありません。

１、　広いエリアはテキストエディタです。（プログラムを書くところ）
２、　上部にはボタンの列があります。これは、ツールバーです。（いろんなボタン）
３、　エディタの下部はメッセージエリアです。（一般メッセージ）
４、　さらにその下部にコンソールがあります。（技術者用メッセージ）

担当：情報メディア学科　山内

プログラムその１

エディタに以下を入力してください。

```
ellipse(50,50,80,80);
```

この一行のコードの意味は、”円を描きます、中心は、左から 50 ピクセル進んだところで、50 ピクセル上から下に下がったところ、幅と高さは、80 ピクセルで！！”です。

タイピングが正確であれば、円のイメージが表示されます。間違っていた場合は、メッセージエリアが赤く表示され、エラーについて説明されます。これが起きたら、サンプルコードを完全にコピーしたかどうか以下の点に注意して確かめてください。

- **英語の綴りはあっていますか？**
- **数字は括弧の中に描かれていますか？**
- **それぞれの数字はコンマで区切られていますか？**
- **行の最後にセミコロンがありますか？**

■Processing での描画について

コンピュータのスクリーンは、ピクセルと呼ばれる光の要素のグリッドで表現されています。それぞれのピクセルは、軸によって定義されたグリッドの中に位置があります。Processing では、X 軸はディスプレイウインドウの左端からの距離で、Y 軸は、上からの距離です。もしスクリーンが 200×200 ピクセルであれば、左上が、(0,0)，中央部が(100,100)，そして右下が(199,199)になります。

なぜ、0 から 199 なのか、1 から 200 じゃないのか?と思うかもしれませんが、コンピュータの世界では、普通０から数えるからです。

ディスプレイウインドウやイメージは、関数とよばれるコード要素を通じて作成されたり、描かれます。関数は、Processing プログラムの基礎づくりのまとまりです。関数のふるまいは、その引数によって定義されます。例えば、ほとんどすべてのProcessing プログラムは、ディスプレウインドウの幅と高さを設定する Size()関数を持ちます。（もし、プログラムに size()ファンクションがない場合、100×100 ピクセルの設定で定義されます。）

担当：情報メディア学科　山内

プログラムその 2

座標(20,50)から(420,110)の間の線を描いてみます。

```
size(480,120);
line(20,50,420,110);
```

■ 色の設定

すべてのシェイプは黒の罫線で白く塗りつぶされており、ディスプレイのウインドウの背景は、明るい灰色です。それらを変えるには、background()関数と、fill()そしてstroke()関数を使います。引数の値は、0 から 255 の幅で、255 が白、128 が中ぐらいの灰色、0 が黒です。図 3-3 は、0 から 255 までの違う灰色マップの値を示しています。

グレイスケールの値をだけでなく、色の構成要素、赤、緑、青の３つの引数を利用して色の設定をすることも出来ます。

```
size(480,120);
background(120,15,203);
fill(120);
ellipse(278,-100,400,400);
smooth();
strokeWeight(3);
fill(255,0,0);

ellipse(120,100,150,150);
strokeWeight(10);
fill(0,255,255);
ellipse(412,60,40,40);
```

担当：情報メディア学科　山内

■変数

プログラムでは、数値を扱いやすくするために**変数**を使います。変数とは数値を記録するための箱のようなものです。

```
size(480, 120);
smooth();
int x=100;
int y = 60;
int d = 80;
ellipse(x, y, d, d);   // Left
ellipse(x+x, y, d, d);  // Middle
ellipse(x*3, y, d, d);  // Right
```

上記のプログラムでは、変数を使ってそれぞれの円を描くプログラムです。
int とは integer(整数)の略で、**整数型の変数**を作るときに宣言します。

■draw()の使い方

マウスやキーボードといったデバイスからの入力に反応させるには、何が起こっているのかを知るために、コンピュータを連続して動かさなければいけません。Processing では、draw と呼ばれる関数でこれを行います。

```
void draw() {
  println("I' m drawing:"+frameCount);
}
```

この draw 関数は、1 秒間に 30 回(デフォルトの設定で)、draw の中に記された内容を繰り返します。frameCount は、draw によって繰り返されるフレームの回数を数えるものであり、println()は()内の引数をコンソールへと表示する関数です。このプログラムを実行するとコンソールにフレームの回った回数が次々と表示されます。

■setup()の使い方

完全な draw 関数にするためには、setup 関数が必要です。setup()は、プログラムを実行したときに一度だけ処理される関数です。

```
void setup() {
  println("I' m starting");
}

void draw() {
  println("I' m drawing:"+frameCount);
}
```

典型的なプログラムでは、setup()の中には、size()が記され、つぎに stroke や図形の色が書かれます。

これで、setup()と draw()の使い方がわかりましたが、setup()の中に変数を作ると、draw からはその変数が使えません。変数を作るときは、setup()や、draw()の外に作らなければいけません。関数の外で作られた変数はグローバル変数と呼ばれ、プログラムのどこからでもアクセスできます。

■mousePressd と if 文の使い方

Processing では、mousePressed を利用することで、簡単にマウスを押したかどうかを知ることができます。mousePressed では、変数の boolean と呼ばれる型と同じで真(true)か偽(false)の値が使われます。マウスのボタンが押されると真、それ以外が偽になります。以下のプログラムを入力してみてください。

```
void setup() {
  size(240, 120);
  smooth();
  strokeWeight(30);
}

void draw() {
  background(204);
  stroke(102);
  line(40, 0, 70, height);
```

```
  if (mousePressed == true) {
    stroke(0);
  }
  line(0, 70, width, 50);
}
```

このプログラムには if 文がマウスを押されたかどうか知る時に使われています。もし、マウスが押されていない場合は、if 以下のプログラムは無視されます。
If 文は主に以下のような使い方をします。

```
If(条件式){
  条件式が真(正しいとき)の時の動作
}
```

先ほどのプログラムでは、条件式の中に(mousePressed==true)となっています。この条件式の”==”は**式の左と右が同じですか？**という意味です。
条件式には以下の種類があります。

A>B	より大きい
A<B	より小さい
A>=B	より大きいか等しい
A<=B	より小さいか等しい
A==B	等しい
A!=B	等しくない

担当：情報メディア学科　山内

演習１　画像の読み込みと表示

それでは、画像処理の最初の演習として画像の読み込みと表示を行ってみましょう。以下のプログラムを入力して保存した後、サポートページから「view.bmp」をダウンロードしてからそのアイコンをテキストエディタにドラッグ＆ドロップしてください。

```
String target = "view.bmp"; // 画像ファイル名の設定
PImage  targetImg;

void setup() {
  targetImg  = loadImage(target); // 画像の読み込み
}

void draw() {
  image(targetImg, 0, 0); // 画像を表示
}
```

実行すると、以下のようなウィンドウが表示されます。

図１ 実行画面（その１）

これは、デフォルトで縦・横それぞれ 100 ピクセルのウィンドウが表示されたために、画像の一部しか表示できていないのです。画像全体を表示するために、プログラムを以下のように書き換えてみましょう。

```
String target = "view.bmp"; // 画像ファイル名の設定
PImage  targetImg;
```

担当：情報メディア学科　山内

```
void setup() {
  targetImg  = loadImage(target); // 画像の読み込み

  int w = targetImg.width; // 横幅の取得
  int h = targetImg.height; // 高さの取得

  size(w, h); // Window の大きさを指定
}

void draw() {
  image(targetImg, 0, 0); // 画像を表示
}
```

実行すると、以下のようなウィンドウが表示されます。

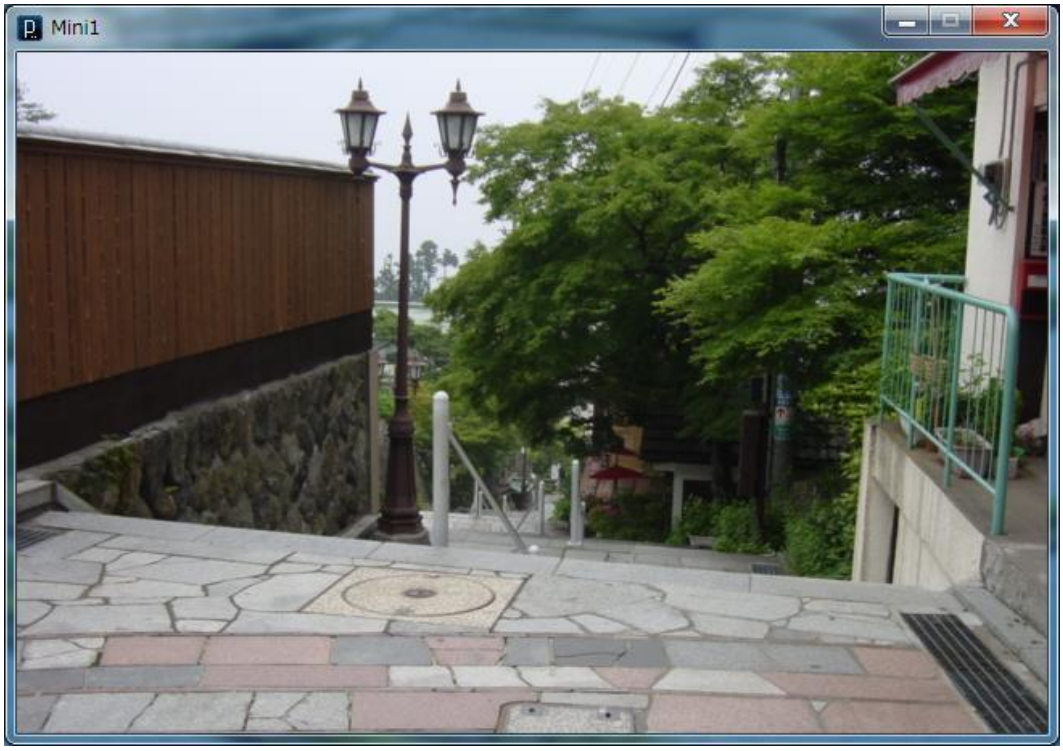


図２ 実行画面（その２）

画像全体が表示されるように、画像の横幅と高さをそれぞれ変数 w,h に格納してそれを size()関数で利用しています。

担当：情報メディア学科　山内

演習 2　画像形式の変換

画像には色々な形式があります。代表的なものを表 1 にまとめてみました。

表 1　代表的な画像形式

名称	拡張子	備考
JPEG	jpg, jpeg	圧縮して画像を保存する画像形式の一つで、相反する圧縮度と画質の度合いを選択して保存できることが特徴である。
GIF	gif	256 色までの画像を圧縮して保存する画像形式の一つ。使用している圧縮アルゴリズムの特許を UNISYS 社が保有しているため、PNG が策定された。
PNG	png	フルカラーの画像を劣化なしで保存できるのが大きな特徴である。
Windows ビットマップ	bmp	Windows で標準使用されている画像形式で、基本的に無圧縮であるため解像度の高い画像のサイズは非常に大きくなる。

以下のプログラムを入力して保存した後、サポートページから「view.bmp」をダウンロードしてからそのアイコンをテキストエディタにドラッグ＆ドロップしてください。

```
String target = "view.bmp"; // 画像ファイル名の設定
String savename = "view.gif"; // 保存ファイル名の設定
PImage  targetImg;

void setup() {
  targetImg  = loadImage(target); // 画像の読み込み

  int w = targetImg.width; // 横幅の取得
  int h = targetImg.height; // 高さの取得

  size(w, h); // Window の大きさを指定
}
```

```
void draw() {
  image(targetImg, 0, 0); // 画像を表示
}

void keyPressed() {
  if (key == ' ') {
    targetImg.save(savename);
  }
}
```

実行すると、演習１と同じように画像が表示されます。ウィンドウが選択されている状態でキーボードのスペース・キーを押すと、変数 savename に格納した名前の画像ファイルが作成されます。今回は、読み込んだ「Windows ビットマップ形式」の画像ファイルが「GIF」形式に変換されて保存されます。同様に、保存する際の拡張子を所望のものにかえればその形式で保存されるのです。簡単で便利ですね！

演習３　画像の明るさ調整

次に、画像の明るさ調整を行ってみましょう。以下のプログラムを入力して保存した後、サポートページから「view.bmp」をダウンロードしてからそのアイコンをテキストエディタにドラッグ＆ドロップしてください。

```
String target = "view.bmp"; // 画像ファイル名の設定
String savename = "view2.png";

PImage  targetImg;
int w, h;

float rate = 2;

void setup() {
  targetImg  = loadImage(target); // 画像の読み込み

  w = targetImg.width; // 横幅の取得
  h = targetImg.height; // 高さの取得
```

```
    size(w, h); // Window の大きさを指定
}

void draw() {
  image(targetImg, 0, 0);
}

void keyPressed() {
  int x, y;
  color c1, c2;

  if (key == ' ') {
    for (y = 0; y < h; y++) {
      for (x = 0; x < w; x++) {
        c1 = targetImg.pixels[y * w + x];
        c2 = color(red(c1) * rate, green(c1) * rate, blue(c1) * rate);
        targetImg.pixels[y * w + x] = c2;
      }
    }

    targetImg.save(savename);
    exit();
  }
}
```

実行すると、やはり演習１と同じように画像を表示します。ウィンドウが選択された状態で、キーボードのスペースキーを押すとプログラムが終了します。プログラムと同じフォルダを調べて見てください。図３に示すような全体的に明るくなった画像が保存されているはずです。

担当：情報メディア学科　山内

図３ 実行結果の画像（その１）

今度は、プログラム中の変数 rate を１より小さな値に設定してみましょう。図４は rate を 0.5 に設定した実行した時の画像です。全体的に暗くなっているのがわかります。

担当：情報メディア学科　山内

図４ 実行結果の画像（その２）

演習４　画像の拡大・縮小

最後に、画像の拡大・縮小を行ってみましょう。以下のプログラムを入力して保存した後、サポートページから「slope.jpg」をダウンロードしてからそのアイコンをテキストエディタにドラッグ＆ドロップしてください。

```
String target = "slope.jpg";
String savename = "slope2.jpg";

PImage targetImg;
int w,h;

void setup() {
  targetImg = loadImage(target);

  w = targetImg.width;
  h = targetImg.height;
```

担当：情報メディア学科　山内

```
  size(w, h);
}

void draw() {
  image(targetImg, 0, 0);
}

void keyPressed() {
  if (key == ' ') {
    targetImg.resize(100,0);
    targetImg.save(savename);
    exit();
  }
}
```

実行すると、やはり演習１と同じように画像を表示します。ウィンドウが選択された状態で、キーボードのスペースキーを押すとプログラムが終了します。プログラムと同じフォルダを調べて見てください。図５に示すように、横幅が 100 ピクセルで元画像の縦横比を維持したままの高さになっています。

図５ 実行結果の画像（その３）

resize()関数のパラメータを色々変えてその働きを確認してみましょう！