Graphics with Processing

2012-05 複雑な図形の描画

http://vilab.org

塩澤秀和

5.1 頂点列による図形描画

複雑な図形描画

- beginShape(図形)
  - 頂点列モードの開始
  - 図形が空欄なら頂点を線で結ぶ（折れ線か多角形になる）
  - その他，下記図形を指定できる POINTS, LINES, TRIANGLES, TRIANGLE_FAN, TRIANGLE_STRIP, QUADS, QUAD_STRIP
- endShape()
  - 頂点列モードの終了
  - endShape(CLOSE): 最初の点と最後の点を結ぶ線を描く
- 塗りつぶし
  - fill()とnoFill()で指定できる

頂点の追加

- vertex(x, y)
  - 図形に次の頂点を加える
  - 色は頂点の前にstrokeで指定
- curveVertex(x, y)
  - 曲線でつなぐ頂点を追加する
- bezierVertex(x1, y1, x2, y2, x3, y3)
  - ベジエ曲線をつなげる

例（塗りつぶさない多角形）

```
noFill();
beginShape();
vertex(30, 20); vertex(30, 75);
vertex(50, 75); vertex(50, 20);
endShape(CLOSE);
```

5.2 図形の回転・拡大（予習）

基本的な書きかた

```
pushMatrix();
translate(x, y);
rotate(a);
/* (x,y)からの相対位置で描画 */
popMatrix();
```

簡単な意味

- pushMatrix()～popMatrix()
  - 座標の変更部分を囲む
- translate(x, y)
  - 座標原点（回転・拡大の中心）を(x, y)に移動する
- rotate(a)
  - 原点を中心に，aラジアン回転
- scale(s), scale(sx, sy)
  - 原点を中心に，拡大または縮小

図形の回転の例

```
int angle = 0;

void setup() {
  size(400, 400);
  rectMode(CENTER);
}

void draw() {
  background(255);
  fill(#ffa0a0);
  pushMatrix();
    translate(width/2, height/2);
    rotate(radians(angle));
    ellipse(0, 0, 200, 100);
  popMatrix();
  angle++;
}
```

新しい原点

(0, 0)は新しい原点の位置

5.3 画像データの表示

画像データ

- 画像の利用
  - サンプル Basics → Image → LoadDisplayImage など
  - 対応形式: jpg gif png tga
- PImage型
  - 画像を扱うには，PImage型のグローバル変数を用意しておく

    PImage img;
- loadImage("ファイル名")
  - 画像データの読み込み
  - 通常，setup()で1回だけ行う

    img = loadImage("a.jpg")
  - ファイルは，事前にメニューのSketch → Add File…でデータフォルダにコピーしておく

画像表示

- image(画像, x, y)
- image(画像, x, y, 幅，高さ)
  - 画像の描画（拡大・縮小）
- imageMode(モード)
  - rectMode/elipseModeと同様

図形データ

- ベクター（ベクトル）表現
  - 座標とパラメータによる図形表現
  - 対応形式: SVG
- PShape, loadShape, shape, shapeMode ⇒ ほぼ画像と同様
  - サンプル Basics → Shape → LoadDisplayShape など

5.4 オブジェクト指向基礎

オブジェクト指向

- オブジェクトとは
  - データとその操作をセットにして，使いやすくしたもの
  - 例）PImage img

オブジェクト指向用語

- 「クラス」:オブジェクトの型
  - 例）PImage
- 「インスタンス」:オブジェクト変数
  - 例）img
- 「フィールド」:オブジェクトの属性
  - 例）img.height
- 「メソッド」:オブジェクトの操作
  - 例）img.save()

PImage型の例

- フィールド
  - img.width, img.height
    - 画像のサイズ（横・縦の幅）
  - img.pixels[]
    - 画像データのピクセル配列
- メソッド（一部）
  - img.save("ファイル名")
    - 画像にファイル名をつけて保存
  - img.get(x, y, 幅, 高さ)
    - 画像の一部を画像として取り出す
  - img.resize(幅, 高さ)
    - 画像のサイズを変更する
  - img.loadPixels(), img.updatePixels()
    - ピクセル処理のためのメソッド

5.5 タイポグラフィ（文字表示）

```
// 描画用フォントの変数（PFont型）
PFont font1, font2;

void setup()  {
  size(300, 300);

  // Processing用フォント
  //（フォントファイルはあらかじめメニューの
  //   Tools→Create Font... で作っておく）
  font1 = loadFont("Impact-48.vlw");

  // 実行環境のフォント
  //（JavaまたはOSのフォント名が指定可能）
  hint(ENABLE_NATIVE_FONTS);
  font2 = createFont("メイリオ", 48);

  // 座標指定モード（通常はMODEL）
  textMode(MODEL);
}
```

```
void draw() {
  background(255);

  // xy方向の位置あわせ方法
  textAlign(CENTER, BOTTOM);

  pushMatrix();
  translate(width/2, height/2);
  rotate(radians(frameCount));

  fill(128, 0, 0);  // 文字の色
  textFont(font1, 32);  // フォントとサイズ
  text("Processing", 0, 0);  // 文字列と座標

  fill(0, 0, 128);
  textFont(font2, 48);
  text("角度 " + frameCount, 0, 100);
  popMatrix();
}
```

5.6 対話的入力処理

システム変数

- mouseX, mouseY
- mousePressed
  - 既出
- pmouseX, pmouseY
  - 前フレームでのマウス位置
- mouseButton
  - 押されたマウスボタン
  - LEFT, RIGHT, CENTER
- keyPressed
  - キーが押されていればtrue
- key
  - 押された文字
- keyCode
  - 特殊キーのキーコード

コールバック関数

- void mousePressed()
  - この関数があると，マウスボタンが押されたときに自動的に実行
- void mouseReleased()
  - 同様に，ボタンが離されたとき
- void mouseMoved()
  - マウスが動かされたとき（ただし，ボタンは押されていないとき）
- void mouseDragged()
  - マウスがドラッグされたとき
- void keyPressed()
  - キーが押されたとき
- void keyReleased()
  - キーが離されたとき

5.7 ファイル入出力

簡易ファイル入出力

- loadStrings("ファイル")
  - ファイルから1行ごとに文字列として読み出して配列に入れる
  - 画像と同様，ファイルは事前にSketch → Add File…でデータフォルダにコピーしておく

```
String lines[] =
    loadStrings("data.txt");
for (int i = 0; i < lines.length;
      i++) {
    // lines[i]の処理
}
```

- saveStrings("ファイル", 配列)
  - ファイルに文字列を保存
  - loadStringsの逆（行単位）

文字列処理

- float(文字列), int(文字列)
  - 文字列を数値に変換
- str(数値)
  - 数値を文字列に変換
- hex(整数)
  - 整数を16進文字列に変換
- unhex(文字列)
  - 16進文字列を数値に変換
- trim(文字列)
  - 文字列から前後の空白を除去
- join(文字列配列)
  - 文字列の連結
- split(文字列)
  - 文字列を空白で分割（joinの逆）

5.8 演習課題

課題

- マウスでクリックした点の座標を順に結ぶ“折れ線”を描くプログラムを作成しなさい
  - **条件**: beginShape() を使って折れ線を書くこと
  - beginShapeとendShapeは，ループの中に入れない
  - 右のプログラム（部分）を参考にして改造するとよい
  - できた人は，ファイルから頂点列を読み込めるようにしてみなさい

注意!!

- 見やすいプログラムを提出せよ
  - Tools → Auto Format でプログラムを整形できる（かも…）

```
int npos = 0; // 点の数
int x[] = new int[100];
int y[] = new int[100];

void setup() {
    // 途中省略
    noLoop(); // アニメーション停止
}

void draw() {
    background(0);
    // ここにbeginShape
    for (int i = 0; i < npos; i++) {
        // 下の行を変更
        ellipse(x[i], y[i], 10, 10);
    }
    // ここにendShape
}

void mousePressed() {
    x[npos] = mouseX;
    y[npos] = mouseY;
    npos++;
    redraw();  // 点が増えたら再描画
}
```