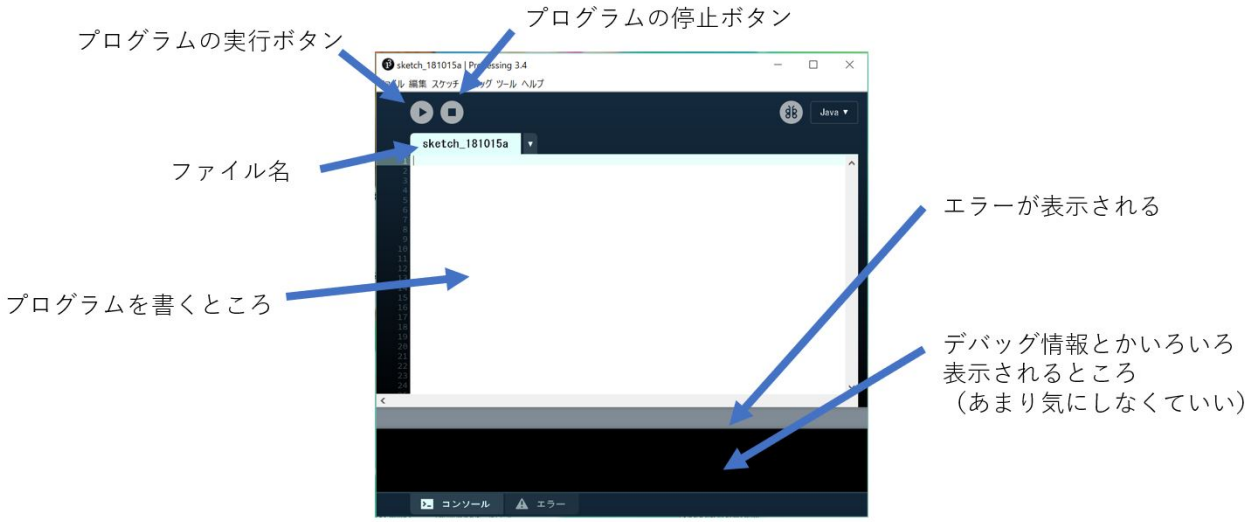
Processingを触ってみよう！

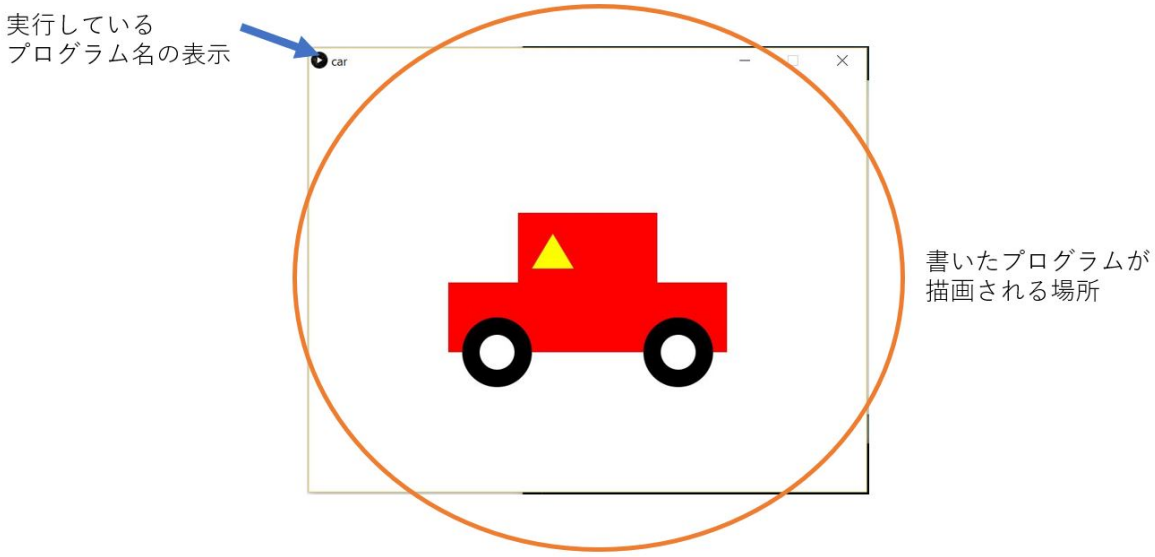
立命館大学情報理工学部情報理工学科知能情報コース2回生
伊藤聡子

2018年10月19日

1　Processing とプログラミング

1.1　Processing の画面説明





※プログラミングする際の注意
プログラミングしているとき，エラーが表示されるところが赤色であれば，エラーがある．
そうでなければエラーがない状態．
エラーがない状態になってから，実行ボタンを押す．

1.2 プログラミングの基礎

1.2.1 処理とその流れ

重要な，覚えておかなければならないことは

1. プログラムは上から順に処理される
2. 一つ一つの処理の最後には「;」（セミコロン）を付ける
3. エラーがあるときは，実行できない

1.3 色と形

1.3.1 サンプルプログラムを見てみよう

itosato_ c2_ 20181019 ＞ Program ＞ shape_ and_ color ＞ shape_ and_ color.pde というファイルを開く．
ファイルには，以下のプログラムが書かれている．

Listing 1: shape_ and_ color

```
1  void setup(){
2    size(800, 600);
3  }
4
5  void draw(){
6    background(255, 255, 255);
7    noStroke();
8
9
10   fill(0, 0, 0);
11
12   ellipse(150, 200, 200, 200);
13
14   rect(0, 0, 200, 200);
15
16   triangle(650, 100, 550, 300, 750, 300);
17
18
19   fill(255, 0, 0);
20   ellipse(150, 450, 200, 200);
21
22   fill(0, 255, 0);
23   rect(300, 350, 200, 200);
24
25   fill(0, 0, 255);
```

```
26    triangle(650, 350, 550, 550, 750, 550);
27 }
```

どんなことが書かれているのか，予想してみよう．

1.4 プログラムを実行してみる

shape_ and_ color を実行する．
図形が 6 つ出てきたらよい．
自分の予想と，合っていたところ，間違っていたところを見てみよう．

1.5 色を変えてみよう

プログラムの中の，　fill(num, num, num);　の値を自由に変えてみよう．
プログラムを書き換えたら実行しよう．
プログラムのどこを変えたらどのように変化したのか，見てみよう．

次に，ブラウザを開き,「検索または Web アドレスを入力」というところに，
https://www.colordic.org/　と入力して，検索する．
上の URL を開いたら，色見本が出てくる．好きな色を選んで，RGB 値を変えてみよう．

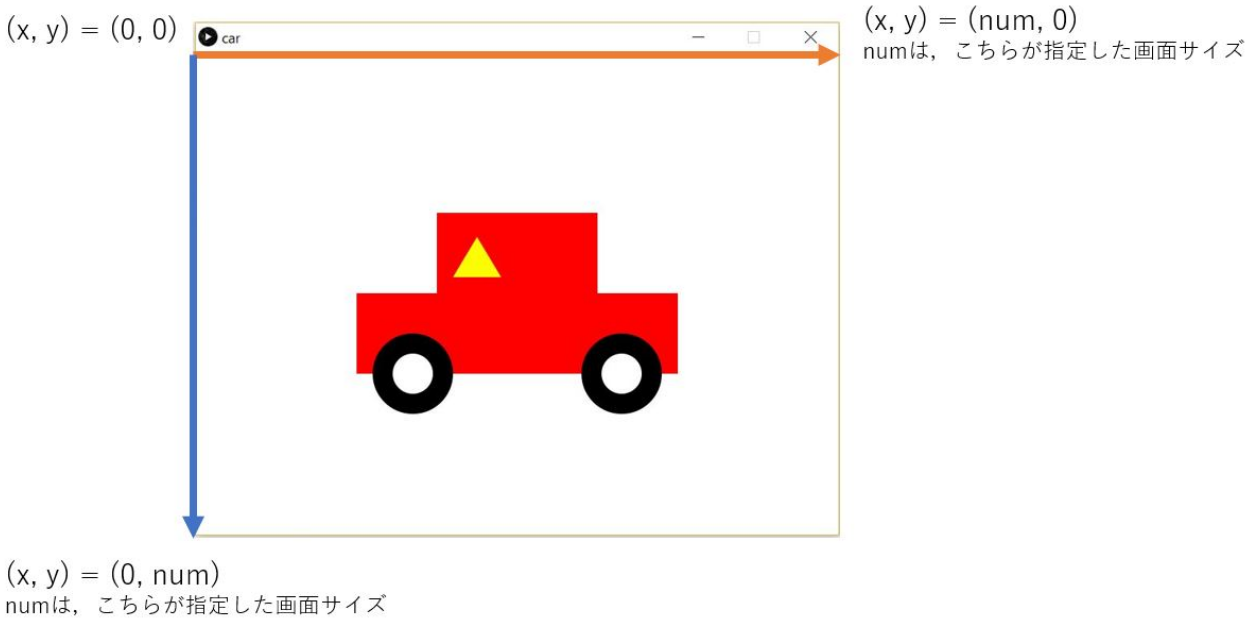
1.6 形を変えてみよう

次に，RGB 値以外のところの数値を変えてみよう．

プログラムを書き換えたら実行する．
プログラムをどのように変えたらどのように変化したのか，見てみよう．

1.7　座標

Processing は，図形を描くときに座標（位置）を決める必要がある．
これは，x 座標と y 座標を指定して決めている．



x 軸は右方向に増加する．左上端が x=0．右上端が指定した画面サイズの値となる．
y 軸は下方向に増加する．左上端が y=0．左下端が指定した画面サイズの値となる．

1.8　void と関数

プログラム中にある「void」とは，関数のこと．ここでいう関数とは，数学の関数とは違い，プログラムの中で処理をまとめたもののことである．
関数は，

```
void 関数名 (){

}
```

という風に書く．
{ と } の間に，実行したいプログラムを書いていく．今回の授業の中では，2 つの関数を使っている．

1.8.1　setup

その名の通り，プログラムを動かすためのセットアップ（準備）をするための関数．
ここは，図形を描画するためのキャンバスとなる画面のサイズ指定を行うなどする場所である．

1.8.2 draw

その名の通り，描くための関数.
この関数の中に図形を描画するためのプログラムを書いていく.

2 図形を描く

2.1 サンプルプログラムを見てみよう

itosato_ c2_ 20181019 ＞ Program ＞ car ＞ car.pde　というファイルを開き，プログラムを実行する.

Listing 2: car

```
 1 void setup(){
 2  size(800, 600);
 3 }
 4
 5 void draw(){
 6   background(255, 255, 255);
 7   noStroke();
 8
 9
10   fill(255, 0, 0);
11   rect(200, 300, 400, 100);
12   rect(300, 200, 200, 100);
13
14   fill(0, 0, 0);
15   ellipse(270, 400, 100, 100);
16   ellipse(530, 400, 100, 100);
17
18   fill(255, 255, 255);
19   ellipse(270, 400, 50, 50);
20   ellipse(530, 400, 50, 50);
21
22   fill(255, 255, 0);
23   triangle(350, 230, 320, 280, 380, 280);
24 }
```

図形を組み合わせることにより，簡単な絵を描くことが出来る.

2.2 図形の描き方

ここでは，基本的な図形である，丸，四角，三角の描き方を紹介する.

2.2.1　丸の描き方

丸を書くためには，描きたい円の中心座標と縦横の直径を指定する必要がある．
具体的には，以下のような書き方をする．

```
ellipse(中心のｘ座標，中心のｙ座標，縦の直径，横の直径);
```

横の長さと縦の長さを一緒にすると円が，別にすると楕円を描くことが出来る．

2.2.2　四角の描き方

四角を描くためには，左上頂点の座標と，縦横の長さを指定する必要がある．
具体的には，以下のような書き方をする．

```
rect(左上頂点のｘ座標，左上頂点のｙ座標，横の長さ，縦の長さ);
```

横の長さと縦の長さを一緒にすると正方形が，別にすると長方形を描くことが出来る．

2.2.3　三角の描き方

三角を描くには，各頂点の座標を指定する必要がある．
具体的には，以下のような書き方をする．

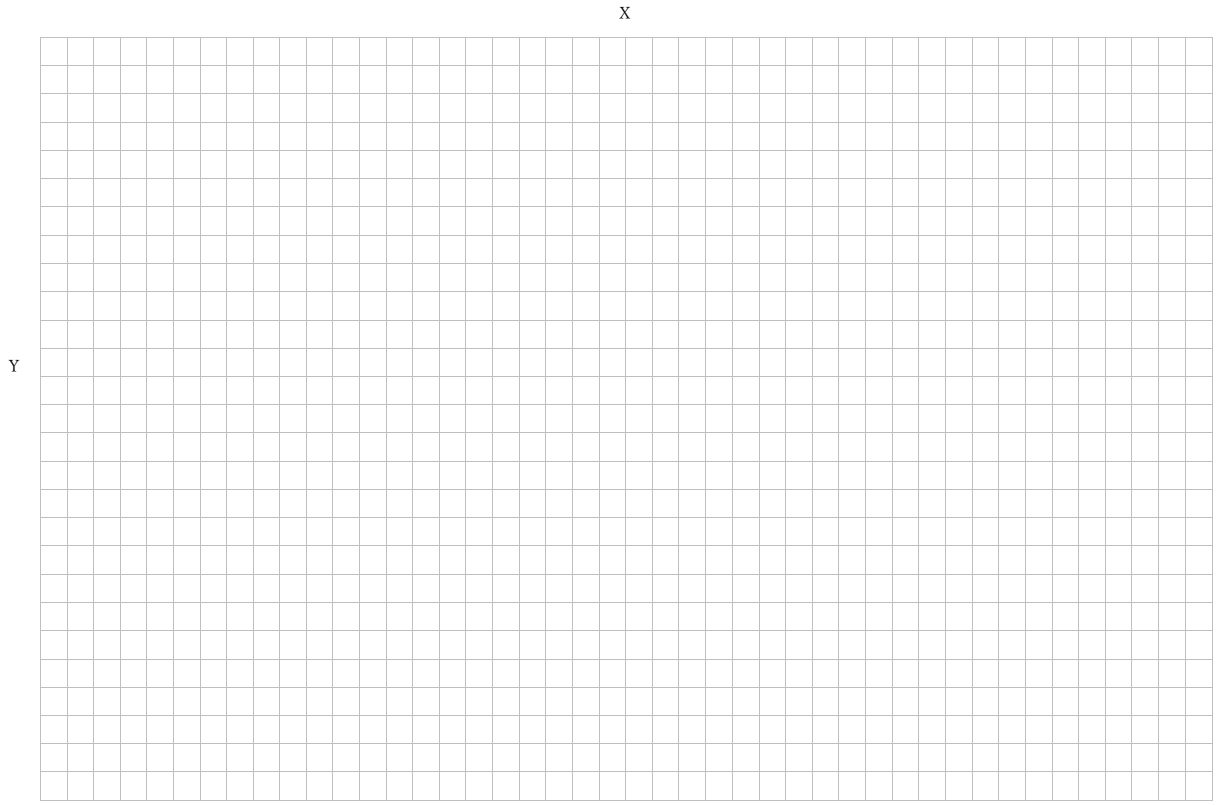
```
triangle(1つ目の頂点のｘ座標，1つ目の頂点のｙ座標，2つ目の頂点のｘ座標，2つ目の
頂点のｙ座標，3つ目の頂点のｘ座標，3つ目の頂点のｙ座標);
```

各頂点の座標次第では，いびつな形の三角形を描くことが出来る．

3　自由に絵を描く

3.1　何を描くか考える

　以下の方眼紙に，絵を下書きしよう．
1 メモリを 10 や 20 で考えるのがいいと思われる．



3.2　どのようなプログラムになるか予想する

　どんなことを書かなければならないのか，自分が分かるようにざっくりとプログラムの流れを書いてみよう．
最低限考えなければならないのは，

- どの図形を描くのか
- どこに図形を描くのか
- 何色にするのか

以上の 3 点である．

3.3 実際にプログラムを書く

今までの流れを思い出して，プログラムを書いてみよう！
わからなくなったら，手を挙げてすぐに質問しよう．
まずは，以下の必ず書かなければならない部分を書いておこう．

Listing 3: フォーマット

```
1  void setup(){
2    size(num, num);
3  }
4
5  void draw(){
6    background(255, 255, 255);
7    noStroke();
8
9
10
11 }
```

noStroke(); の下から自由にプログラムを書いていこう．
画面サイズは自由に変更しても構わない．