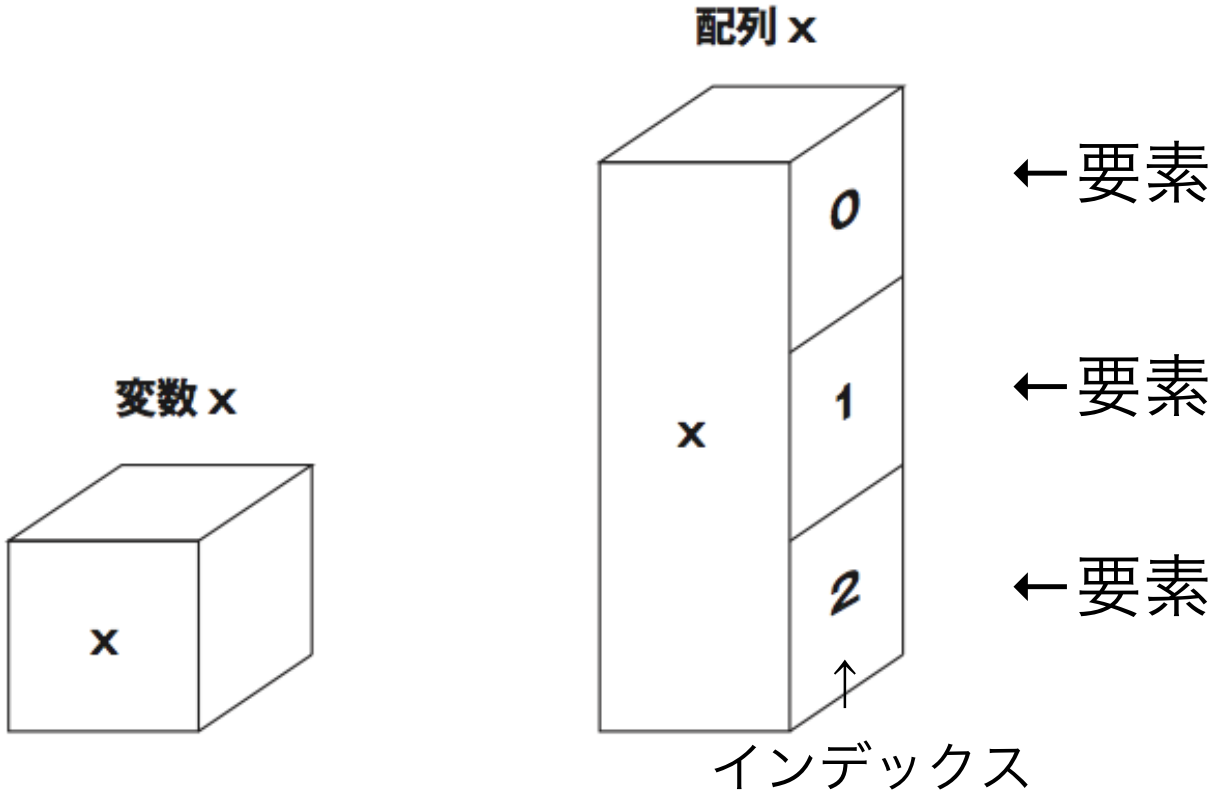
配列と変数の違い

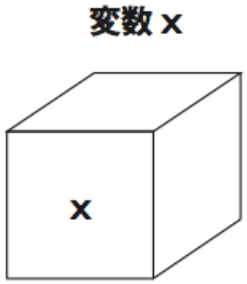
変数と配列のイメージ



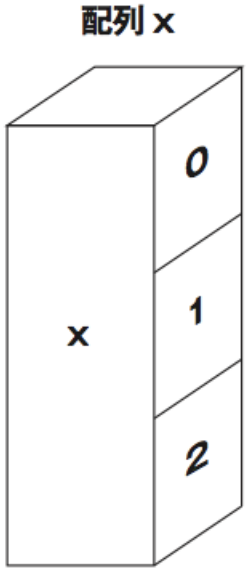
要素数は3

配列と変数の違い

宣言・作成・代入・読み出し




```
int x=0;            //宣言
x=0;                //代入
background(x);      //読み出し
```

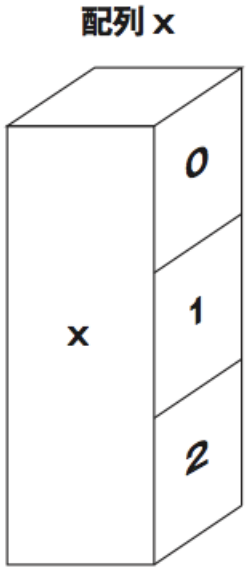



```
int[] x;              //宣言（int型）
x = new int[3];       //作成 3は要素の数
x[0]=0;               //代入
background(x[0]); //読み出し
```

↑
インデックス

配列と変数の違い

宣言・作成・代入・読み出し




```
int[] x;              //宣言（int型）
x = new int[3];       //作成 3は要素の数
```

宣言と作成は同時に1行で書いてもよい。

```
int[] x = new int[3];
```

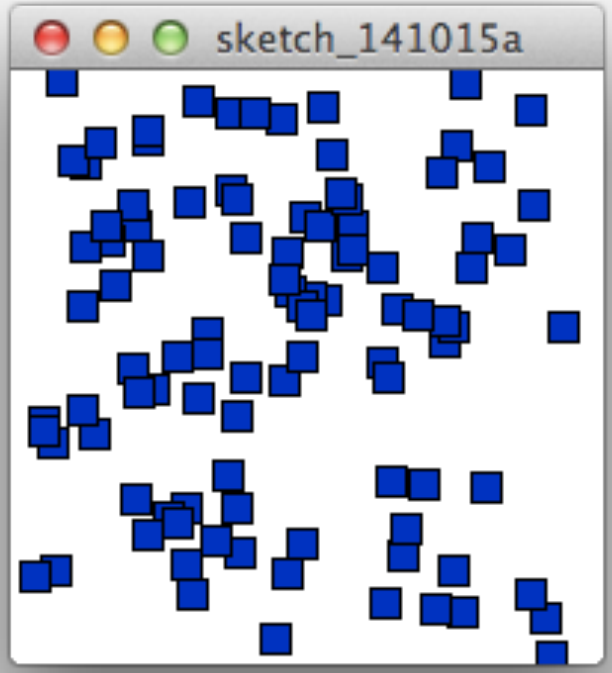
使い方の基本

```
int[] x = new int[3];

size(400,400);

x[0] = 10;
x[1] = 120;
x[2] = 240;

stroke(255,0,0);

line(0,0,x[0],100);
line(0,0,x[1],100);
line(0,0,x[2],100);
```

```
int[] x = new int[3];

size(400,400);

x[0] = 10;
x[1] = 120;
x[2] = 240;

stroke(255,0,0);

for(int i=0; i<3; i++){
  line(0,0,x[i],100);
}
```

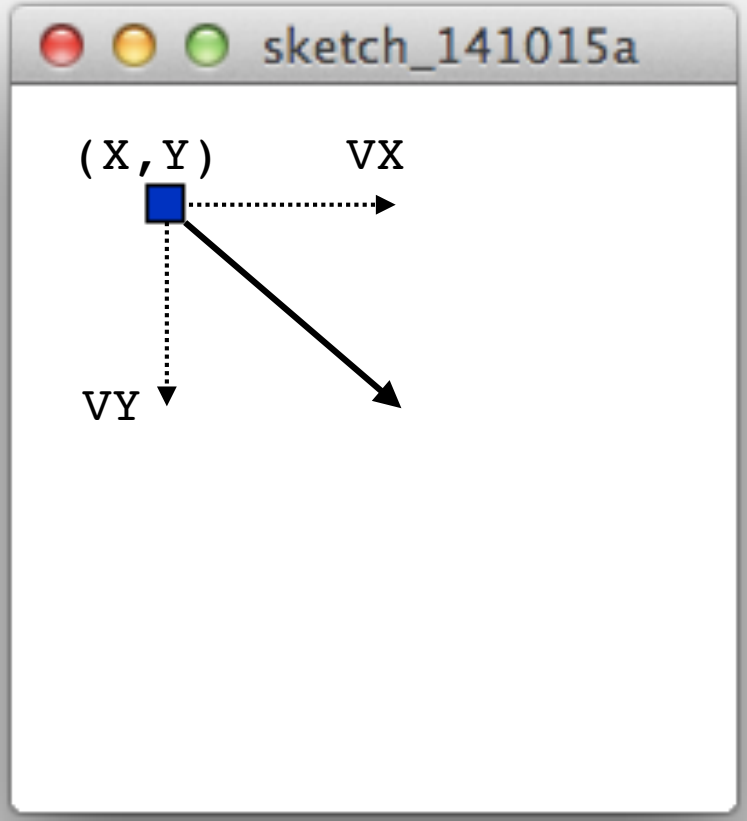
今日のゴール

多量の物体を独立してアニメーションさせる



前回の復習

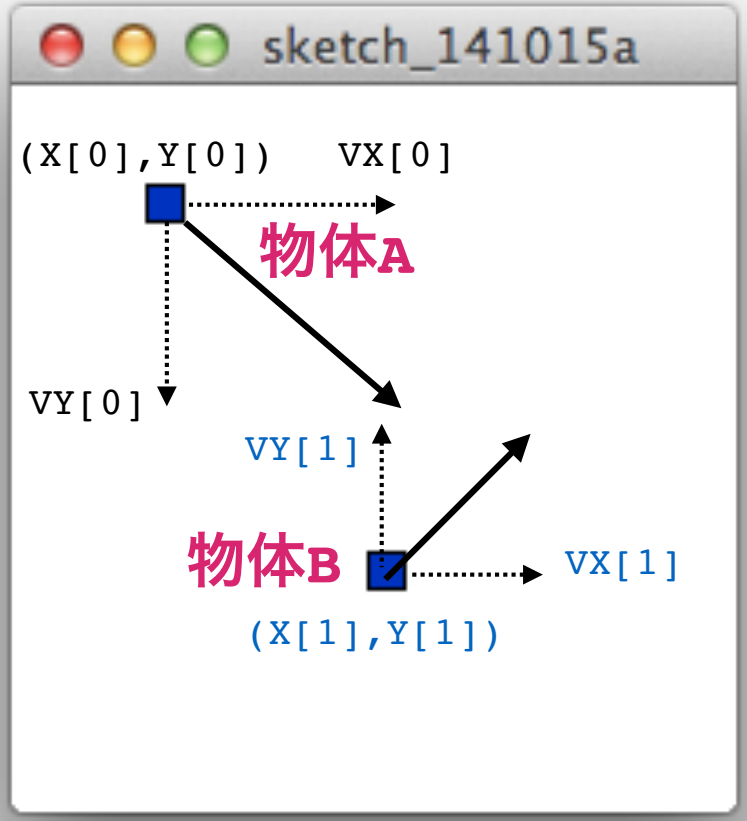
ひとつの物体を動かす時




```
int X;                          //物体の座標X
int Y;                          //物体の座標Y
int VX;                         //物体の速度VX
int VY;                         //物体の速度VY

void setup(){
 size(200,200);
 fill(0);
 X = 10;
 Y = 20;
 VX = 1;                        //データの初期化
 VY = 1;
}

void draw(){


 X = X + VX;                    //座標の更新
 Y = Y + VY;

 rect(X,Y,10,10);               //物体の描画
}
```

配列を使う

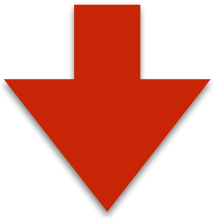
複数の物体を動かす時




```
int num = 2;

int[]X = new int[num];          //物体の座標X
int[]Y = new int[num];          //物体の座標Y
int[]VX = new int[num];         //物体の速度VX
int[]VY = new int[num];         //物体の速度VY

void setup(){
 size(200,200);
 fill(0);
 X[0] = 10;
 Y[0] = 20;
 VX[0] = 1;                     //物体Aのデータの初期化
 VY[0] = 1;

 X[1] = 50;
 Y[1] = 80;
 VX[1] = -1;                    //物体Bのデータの初期化
 VY[1] = -1;
}

void draw(){

  X[0] = X[0]+VX[0];
  Y[0] = Y[0]+VY[0];            //物体A座標の更新

  X[1] = X[1]+VX[1];            //物体Bの座標の更新
  Y[1] = Y[1]+VY[1];

  rect(X[0],Y[0],10,10);        //物体Aの描画
  rect(X[1],Y[1],10,10);        //物体Bの描画
}
```

複数の物体に対して同じ処理を行っているところをfor文でまとめよう

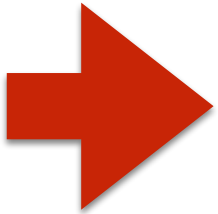
```
X[0] = X[0]+VX[0];
Y[0] = Y[0]+VY[0];          //物体A座標の更新

X[1] = X[1]+VX[1];          //物体Bの座標の更新
Y[1] = Y[1]+VY[1];

rect(X[0],Y[0],10,10);      //物体Aの描画
rect(X[1],Y[1],10,10);      //物体Bの描画
```



```
for(int i=0; i<num; i++){
  X[i] = X[i]+VX[i];        //すべての物体の座標の更新
  Y[i] = Y[i]+VY[i];
  rect(X[i],Y[i],10,10);    //すべての物体を描画
}
```

ウィンドウの端での跳ね返りをコードする

```
for(int i=0; i<num; i++){
  X[i] = X[i]+VX[i];
  Y[i] = Y[i]+VY[i];
  rect(X[i],Y[i],10,10);
}
```



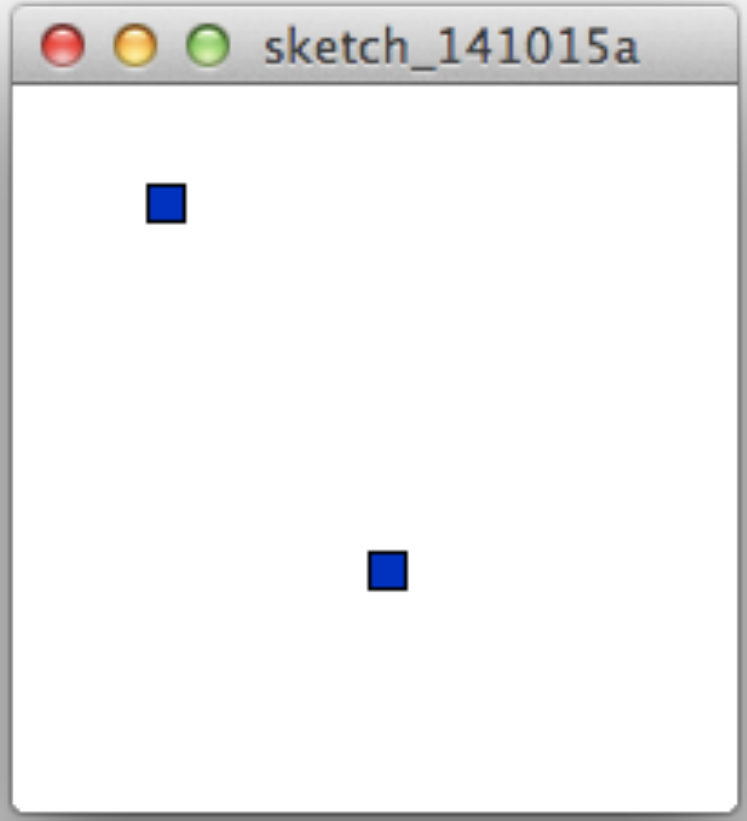
```
for(int i=0; i<num; i++){
  if(X[i] > width-10){
    VX[i] = VX[i] *-1;
    X[i] = width -10;
  }
  if(X[i] < 0){
    VX[i] = VX[i] *-1;
    X[i] = 0;
  }
  if(Y[i] > height-10){
    VY[i] = VY[i] *-1;
    Y[i] = height -10;
  }
  if(Y[i] < 0){
    VY[i] = VY[i] *-1;
    Y[i] = 0;
  }
  X[i] = X[i]+VX[i];
  Y[i] = Y[i]+VY[i];
  rect(X[i],Y[i],10,10);
}
```

完成したコード

```
int num = 2;

int[]X = new int[num];
int[]Y = new int[num];
int[]VX = new int[num];
int[]VY = new int[num];

void setup(){

 size(200,200);

 X[0] = 10;
 Y[0] = 20;
 VX[0] = 1;
 VY[0] = 1;

 X[1] = 50;
 Y[1] = 80;
 VX[1] = -1;
 VY[1] = -1;

}
```

```
void draw(){
  background(255);
  fill(0,0,200);
  for(int i=0; i<num; i++){

    if(X[i] > width-10){
      VX[i] = VX[i] *-1;
      X[i] = width -10;
    }
    if(X[i] < 0){
      VX[i] = VX[i] *-1;
      X[i] = 0;
    }
    if(Y[i] > height-10){
      VY[i] = VY[i] *-1;
      Y[i] = height -10;
    }
    if(Y[i] < 0){
      VY[i] = VY[i] *-1;
      Y[i] = 0;
    }

    X[i] = X[i]+VX[i];
    Y[i] = Y[i]+VY[i];
    rect(X[i],Y[i],10,10);
  }
}
```

完成したコード



多量の物体を表示するには？

```
int num = 2;

int[]X = new int[num];
int[]Y = new int[num];
int[]VX = new int[num];
int[]VY = new int[num];

void setup(){

 size(200,200);

 X[0] = 10;
 Y[0] = 20;
 VX[0] = 1;
 VY[0] = 1;

 X[1] = 50;
 Y[1] = 80;
 VX[1] = -1;
 VY[1] = -1;

}
```

nを増やす

データの初期化も
for文でまとめる

```
void draw(){
  background(255);
  fill(0,0,200);
  for(int i=0; i<num; i++){

    if(X[i] > width-10){
      VX[i] = VX[i] *-1;
      X[i] = width -10;
    }
    if(X[i] < 0){
      VX[i] = VX[i] *-1;
      X[i] = 0;
    }
    if(Y[i] > height-10){
      VY[i] = VY[i] *-1;
      Y[i] = height -10;
    }
    if(Y[i] < 0){
      VY[i] = VY[i] *-1;
      Y[i] = 0;
    }

    X[i] = X[i]+VX[i];
    Y[i] = Y[i]+VY[i];
    rect(X[i],Y[i],10,10);
  }
}
```

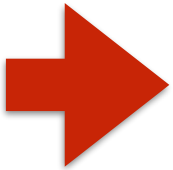
randomメソッドを使ってデータの初期化もfor文でおこなう。

```
int num = 2;

int[]X = new int[num];
int[]Y = new int[num];
int[]VX = new int[num];
int[]VY = new int[num];

void setup(){

 size(200,200);

 X[0] = 10;
 Y[0] = 20;
 VX[0] = 1;
 VY[0] = 1;

 X[1] = 50;
 Y[1] = 80;
 VX[1] = -1;
 VY[1] = -1;

}
```

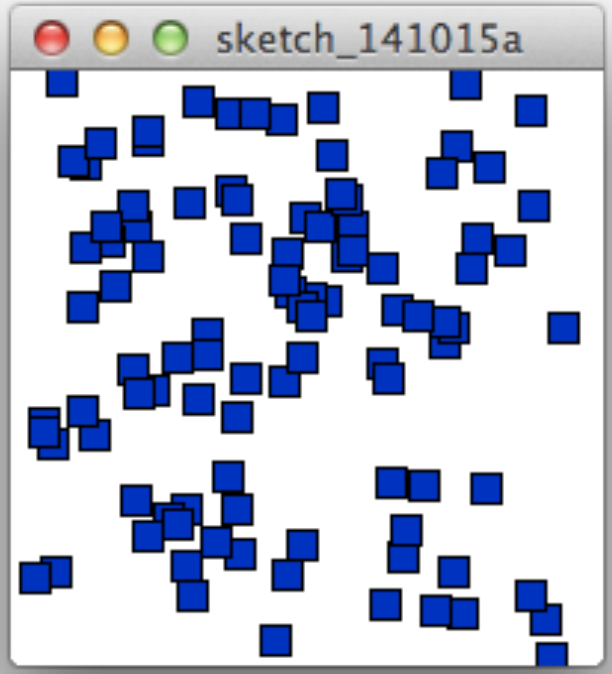
randomメソッドの結果は
少数なので、配列も少数型に変更

```
int num = 2;

float[]X = new float[num];
float[]Y = new float[num];
float[]VX = new float[num];
float[]VY = new float[num];

void setup(){

 size(200,200);

 for(int i=0; i<num; i++){
   X[i] = random(width);
   Y[i] = random(height);
   VX[i] = random(-5,5);
   VY[i] = random(-5,5);
 }
}
```

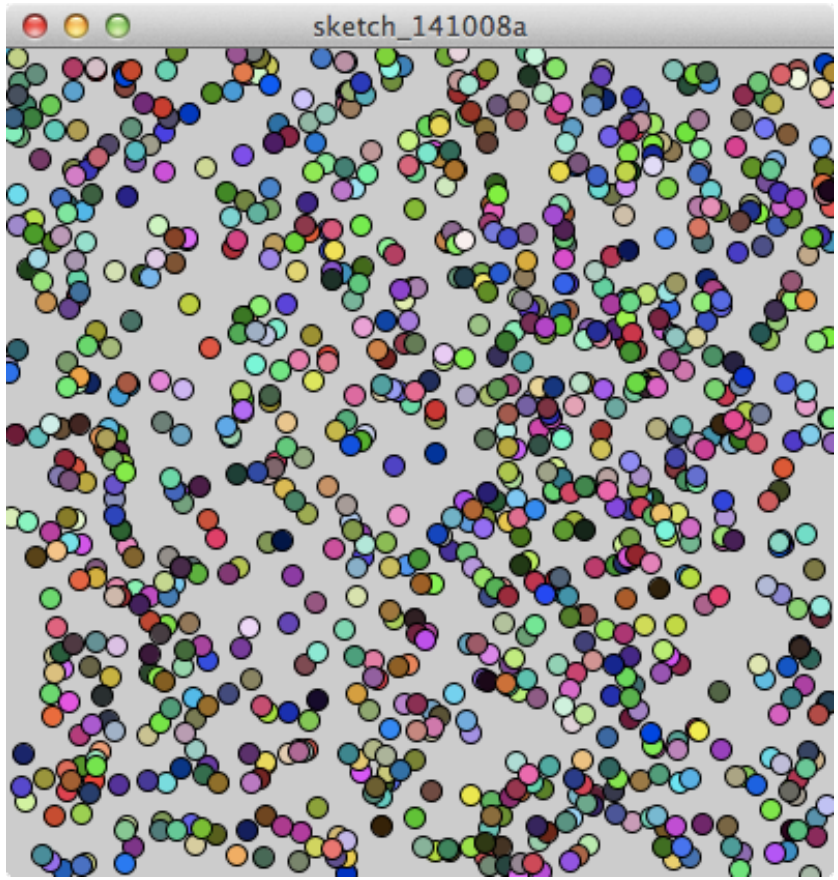
for文でまとめる



n=100



演習1　　**色の違う500個の物体を同時に動かす。**



ヒント

配列、r、g、bを追加し

float r[i] = random(255);

のように、rに0〜255の値で
データを初期化する

ヒント2

物体を描画する前に

```
fill(r[i],g[i],b[i]);
```

演習2　**色の違う500個の物体を同時に動かす。かつ、重力を与える。**

ヒント　重力はすべての物体に等しく作用する下向きの加速度なので
配列でなく変数。

すべての物体のY方向の速度を重力で加速する