明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

プログラミング演習2
クラスと継承の復習

中村，小松，小林，鹿喰

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

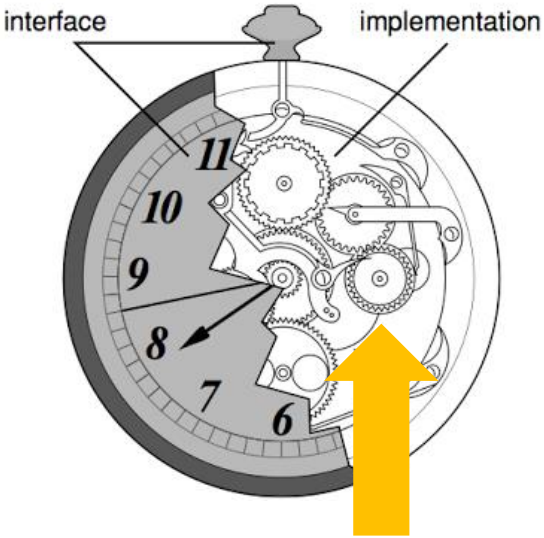
クラスも継承もマスターできた？

クラスって便利？

他の例

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 時計というクラスを作る
  - 時計のインスンタンス変数
    - 現在時間(時分秒)の情報
    - 目覚ましのON/OFF状態管理変数
    - 目覚ましの設定時間
  - 時計のインスタンスメソッド
    - 分変更メソッド
    - 時計停止メソッド(秒針停止)
    - 目覚まし設定時間変更メソッド
    - 目覚ましのON/OFF切り替えメソッド



中の仕組みが
わからなくても使える！

←第2回のp.18の意味

http://yoppa.org/proga10/1190.html

クラス、継承の説明の流れ

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

跳ね回るBallを50個表示するプログラム

Ballクラスを定義して位置や速度の変数をまとめる

移動や初期化のインスタンスメソッドを定義
コンストラクタを定義

描画もインスタンスメソッドで定義

配列にしてみる

スッキリ・
分かりやすく！

クラス、継承の説明の流れ

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

Ballを真似してSquareを定義

重複が多くて無駄！

共通部分をまとめてスーパークラスを定義

これを継承してBallやSquareクラスを定義！

変更したところは“オーバーライド”

継承元メソッドの呼出しは super. で

スッキリ・
分かりやすく！

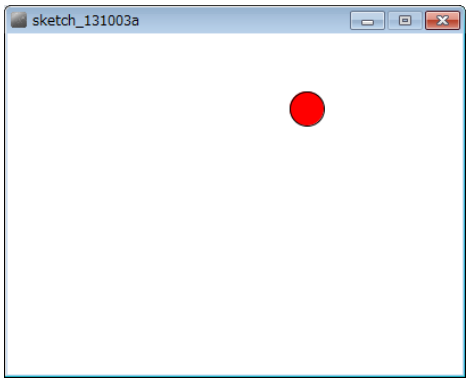
単純にオーバーライドすると、継承元でやっていた処理が
抜けて、init等が思ったように機能しなかったりするので

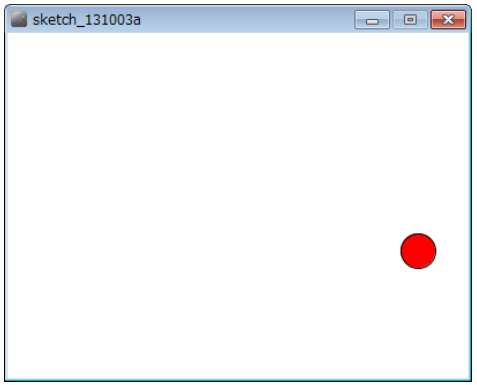
端で跳ね返るオブジェクト

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

(Q1) 400x300のウインドウ内で，画面中央から毎フレームx方向に2ピクセル，y方向に3ピクセルずつ移動する直径が30の赤い円が右端・左端・上端・下端に来ると跳ね返るようにするには？

- 考え方
  - 右端・左端・上端・下端で衝突する時の条件を整理
  - 衝突した時の速度を反転させる
    - speedX = -speedX;
    - speedY = -speedY;





```
int posX;
int posY;
int speedX;
int speedY;

void setup() {
  size( 400, 300 );
  posX = (int)random( 0, width );
  posY = (int)random( 0, height );
  speedX = (int)random( 1,5 );
  speedY = (int)random( 1,5 );
  fill( 255, 0, 0 );
}
```

```
void draw() {
  background( 255 );
  posX += speedX;
  posY += speedY;
  if ( posX > width-15 ) {
    posX = width-15;
    speedX = -speedX;
  }
  if ( posX < 15 ) {
    posX = 15;
    speedX = -speedX;
  }
  if ( posY > height-15 ) {
    posY = height-15;
    speedY = -speedY;
  }
  if ( posY < 15 ) {
    posY = 15;
    speedY = -speedY;
  }
  ellipse( posX, posY, 30, 30 );
}
```

今までの知識で
プログラムを組むと

クラスで考える

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 座標，スピードを持ったオブジェクトを作る
  - ここではx, y座標(posX, posY)とspeedX，speedYを持つBallクラスを考え，その変数を定義する．

定義する

```
class Ball {
    int posX;
    int posY;
    int speedX;
    int speedY;
}
```

変数を定義

インスタンス化する（使えるようにする）

```
Ball  ball;
ball = new Ball();
ball.posX = 200;
ball.posY = 150;
ball.speedX = 5;
ball.speedY = 7;
```

変数の定義（箱を作る）

newで具体化

値を代入

描画してみる

```
ellipse( ball.posX, ball.posY, 30, 30 );
```

オブジェクト変数名 . インスタンス変数名

クラスを作るときは別タブで

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

クラスを使ってみる

```
class Ball {
  int posX;
  int posY;
  int speedX;
  int speedY;
}

Ball  ball;

void setup() {
  size( 400, 300 );
  ball = new Ball();
  ball.posX = (int)random( 0, width );
  ball.posY = (int)random( 0, height );
  ball.speedX = (int)random( 1 ,5 );
  ball.speedY = (int)random( 1, 5 );
  fill( 255, 0, 0 );
}
```

```
void draw() {
  background( 255 );
  ball.posX += ball.speedX;
  ball.posY += ball.speedY;
  if ( ball.posX > width-15 ) {
    ball.posX = width-15;
    ball.speedX = -ball.speedX;
  }
  if ( ball.posX < 15 ) {
    ball.posX = 15;
    ball.speedX = -ball.speedX;
  }
  if ( ball.posY > height-15 ) {
    ball.posY = height-15;
    ball.speedY = -ball.speedY;
  }
  if ( ball.posY < 15 ) {
    ball.posY = 15;
    ball.speedY = -ball.speedY;
  }
  ellipse( ball.posX, ball.posY, 30, 30 );
}
```

ボールを3つにすると？

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 3つの変数を定義
  - new で具体化
    - miyashita
    - komatsu
    - fukuchi

    というボールにする！
  - そのそれぞれの値を参照できるようにする

```
class Ball{
   int posX;
   int posY;
   int speedX;
   int speedY;
}

Ball miyashita;
Ball komatsu;
Ball fukuchi;

void setup() {
  size( 400, 300 );

  miyashita = new Ball();
  komatsu = new Ball();
  fukuchi = new Ball();
}
```

```
class Ball{
   int posX;
   int posY;
   int speedX;
   int speedY;
}
Ball miyashita;
Ball komatsu;
Ball fukuchi;
void setup() {
  size( 400, 300 );
  fill( 255, 0, 0 );
  miyashita =
  komatsu = n
  fukuchi = ne
  miyashita.p
  miyashita.p
  miyashita.s
  miyashita.s
  komatsu.posX = (int)random(width);
  komatsu.posY = (int)random(height);
  komatsu.sp
  komatsu.sp
  fukuchi.pos
  fukuchi.pos
  fukuchi.spe
  fukuchi.speedY = (int)random(1,5);
}
```

```
void draw() {
  background(255);
  miyashita.posX = miyashita.posX + miyashita.speedX;
  miyashita.posY = miyashita.posY + miyashita.speedY;
  komatsu.posX = komatsu.posX + komatsu.speedX;
  komatsu.posY = komatsu.posY + komatsu.speedY;
  fukuchi.posX = fukuchi.posX + fukuchi.speedX;
  fukuchi.posY = fukuchi.posY + fukuchi.speedY;

  if ( miyashita.posX > width-15 ) {
    miyashita.posX = width - 15;
    miyashita.speedX = -miyashita.speedX;
  }
  ellipse(  fukuchi.posX, fukuchi.posY, 30, 30 );
}
```

まったく楽になっていない！
というかむしろ大変になっている！！
クラス面倒なだけじゃん！！！

データ型としてしか使っていないため
面倒で当然

インスタンスメソッド

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 移動をインスタンスメソッドにしてしまう
  - 全ての円は場所や速度は違うけれど，同じルールで動いているのでまとめることが可能！
  - 内部で勝手に振る舞うメソッド（関数）にしてしまう

下記のように指定するだけで動くように！

```
miyashita.move();
komatsu.move();
fukuchi.move();
```

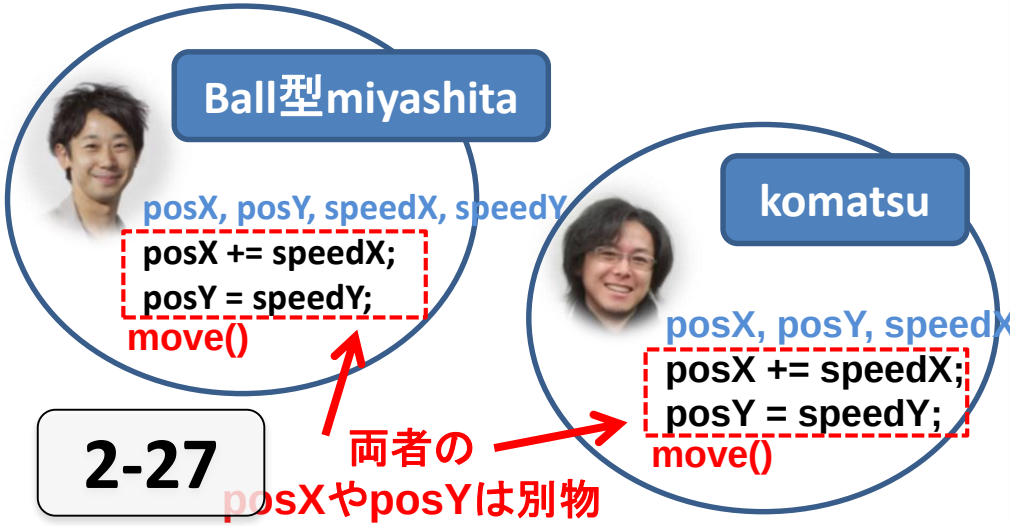
オブジェクト変数名 . インスタンスメソッド名

void move() を作る

- 移動用メソッドを追加
  - 位置を変更
  - 端に来ると跳ね返る
- インスタンスメソッドではクラス内変数を活用

```
class Ball{
   int posX;
   int posY;
   int speedX;
   int speedY;
   void move(){
      this.posX += this.speedX;
      this.posY += this.speedY;
      if (this.posX > width-15 ) {
         this.posX = width - 15;
         this.speedX = -this.speedX;
      }
      if( this.posX < 15 ){
         this.posX = 15;
         this.speedX = -this.speedX;
      }
      if( this.posY > height-15){
         this.posY = height - 15;
         this.speedY = -this.speedY;
      }
      if( this.posY - 15 < 0 ){
         this.posY = 15;
         this.speedY = -this.speedY;
      }
   }
}
```

クラス内で定義されているposXやposY，speedX，speedYを利用したり変更したりできる

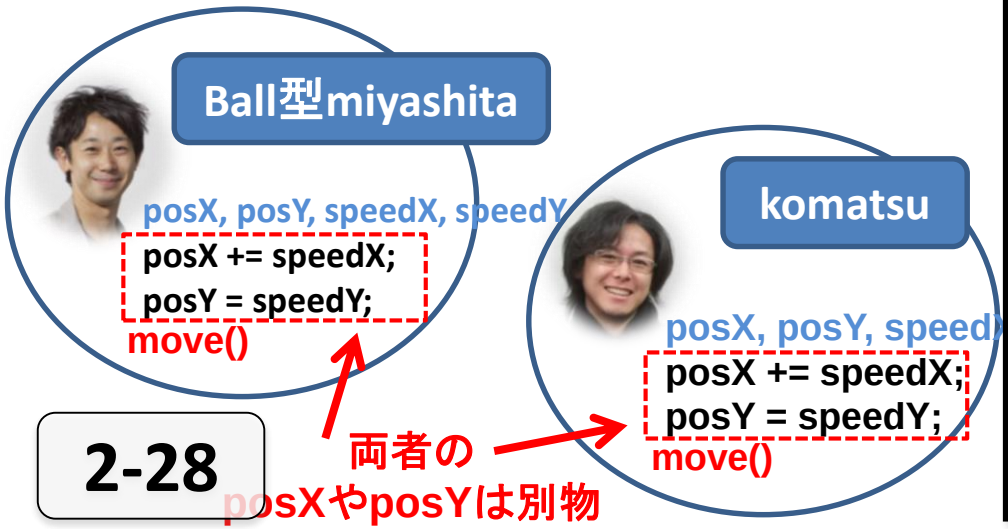


void move() を作る

- 移動用メソッドを追加
  - 位置を変更
  - 端に来ると跳ね返る
- インスタンスメソッドではクラス内変数を活用

```
class Ball{
  int posX;
  int posY;
  int speedX;
  int speedY;
  void move(){
    posX += speedX;
    posY += speedY;
    if ( posX > width-15 ) {
      posX = width - 15;
      speedX = -speedX;
    }
    if( posX < 15 ){
      posX = 15;
      speedX = -speedX;
    }
    if( posY > height-15){
      posY = height - 15;
      speedY = -speedY;
    }
    if( posY - 15 < 0 ){
      posY = 15;
      speedY = -speedY;
    }
  }
}
```

this.は省略可能



改良したBallクラスを使うと

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室


```
Ball miyashita;
Ball komatsu;
Ball fukuchi;
void setup() {
  size( 400, 300 );
  fill( 255, 0, 0 );
  miyashita = new Ball();
  komatsu = new Ball();
  fukuchi = new Ball();
  miyashita.posX = (int)random(width);
  miyashita.posY = (int)random(height);
  miyashita.speedX = (int)random(5);
  miyashita.speedY = (int)random(5);
  komatsu.posX = (int)random(width);
  komatsu.posY = (int)random(height);
  komatsu.speedX = (int)random(5);
  komatsu.speedY = (int)random(5);
  fukuchi.posX = (int)random(width);
  fukuchi.posY = (int)random(height);
  fukuchi.speedX = (int)random(5);
  fukuchi.speedY = (int)random(5);
```

```
void draw() {
  background(255);
  miyashita.move();
  komatsu.move();
  fukuchi.move();

  ellipse( miyashita.posX, miyashita.posY, 30, 30 );
  ellipse( komatsu.posX, komatsu.posY, 30, 30 );
  ellipse(  fukuchi.posX, fukuchi.posY, 30, 30 );
}
```

draw() がかなり短くなった！

インスタンスメソッド続き

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 最初の位置を設定する部分もインスタンスメソッドにしてしまおう！
  - 初期位置の設定方法は
    - XXXXX.posX = (int)random(width);
    - XXXXX.posY = (int)random(height);
    - XXXXX.speedX = (int)random(1,5);
    - XXXXX.speedY = (int)random(1,5);

void init() で初期化

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

```
class Ball{
    int x;
    int y;
    int speedX;
    int speedY;

    void init(){
        posX = (int)random(width);
        posY = (int)random(height);
        speedX = (int)random(1,5);
        speedY = (int)random(1,5);
    }
```

```
    void move(){
        posX += speedX;
        posY += speedY;
        if ( posX > width-15 ) {
            posX = width-15;
            speedX = -speedX;
        }
        if( posX < 15 ){
            posX = 15;
            speedX = -speedX;
        }
        if( posY > height-15){
            posY = height-15;
            speedY = -speedY;
        }
        if( posY - 15 < 0 ){
            posY = 15;
            speedY = -speedY;
        }
    }
}
```

改良したBallクラスを使うと

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室


```
Ball miyashita;
Ball komatsu;
Ball fukuchi;

void setup() {
  size( 400, 300 );
  fill( 255, 0, 0 );
  miyashita = new Ball();
  komatsu = new Ball();
  fukuchi = new Ball();
  miyashita.init();
  komatsu.init();
  fukuchi.init();
}
```

```
void draw() {
  background(255);
  miyashita.move();
  komatsu.move();
  fukuchi.move();

  ellipse( miyashita.posX, miyashita.posY, 30, 30 );
  ellipse( komatsu.posX, komatsu.posY, 30, 30 );
  ellipse(  fukuchi.posX, fukuchi.posY, 30, 30 );
}
```

setup() もかなり
短くなった！

先ほどのプログラムを改良して
動かしてみよう！

コンストラクタ！

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- コンストラクタは new されたときに呼び出される場所．init() はそこで呼び出したら良いのでは？

```
class Ball{
    int posX;
    int posY;
    int speedX;
    int speedY;
    Ball(){

    }
}
```

コンストラクタ！

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

```
class Ball{
    int posX;
    int posY;
    int speedX;
    int speedY;
    Ball(){
        init();
    }

    void init(){
        posX = (int)random(width);
        posY = (int)random(height);
        speedX = (int)random(1,5);
        speedY = (int)random(1,5);
    }
```

コンストラクタで
initメソッドを呼び出す！

```
    void move(){
        posX += speedX;
        posY += speedY;
        if ( posX > width-15 ) {
            posX = width-15;
            speedX = -speedX;
        }
        if( posX < 15 ){
            posX = 15;
            speedX = -speedX;
        }
        if( posY > height-15){
            posY = height-15;
            speedY = -speedY;
        }
        if( posY - 15 < 0 ){
            posY = 15;
            speedY = -speedY;
        }
    }
}
```

改良したBallクラスを使うと

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室


```
Ball miyashita;
Ball komatsu;
Ball fukuchi;

void setup() {
  size( 400, 300 );
  fill( 255, 0, 0 );
  miyashita = new Ball();
  komatsu = new Ball();
  fukuchi = new Ball();
}
```

```
void draw() {
  background(255);
  miyashita.move();
  komatsu.move();
  fukuchi.move();

  ellipse( miyashita.posX, miyashita.posY, 30, 30 );
  ellipse( komatsu.posX, komatsu.posY, 30, 30 );
  ellipse(  fukuchi.posX, fukuchi.posY, 30, 30 );
}
```

setup() がさらに
短くなった！

先ほどのプログラムを改良して
動かしてみよう！

描画もインスタンスメソッドに

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 下みたいなのはあまり好ましくない
  - ellipse(miyashita.posX, miyashita.posY, 30, 30);
  - ellipse(komatsu.posX, komatsu.posY, 30, 30);
  - ellipse(fukuchi.posX, fukuchi.posY, 30, 30);

```
class Ball{
   int posX;
   int posY;
   int speedX;
   int speedY;
   void display(){
      ellipse( posX, posY, 30, 30 );
   }
```

データはなるべく隠蔽したい（カプセル化）

```
void draw() {
   background(255);
   miyashita.move();
   komatsu.move();
   fukuchi.move();

   miyashita.display();
   komatsu.display();
   fukuchi.display();
}
```

先ほどのプログラムを改良して動かしてみよう！

端で跳ね返る50個のボール

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

400x300の画面内にX座標Y座標ともにランダムに配置された50個の円を描き，それぞれのX，Y方向へのスピードに応じて移動するようにせよ

- posX, posY という座標の配列と，speedX, speedY というXおよびY方向の速度をもつ配列を導入！

クラスを使って実現してみよう！

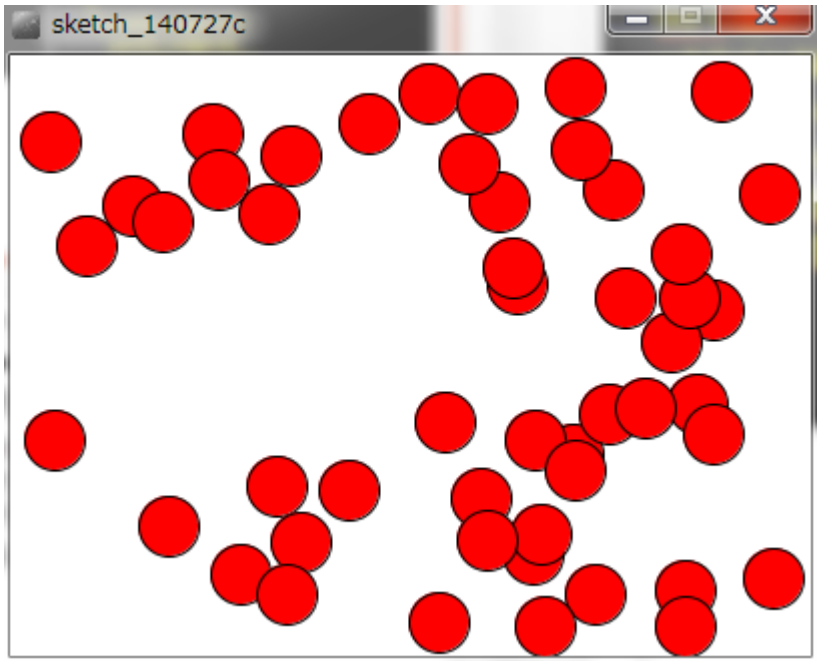
オブジェクト＋配列

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 50個の丸を動かすには配列を使う！

```
Ball [] balls = new Ball [50];

void setup() {
  size( 400, 300 );
  fill( 255, 0, 0 );
  for( int i=0; i<50; i++ ){
    balls[i] = new Ball();
  }
}

void draw(){
  background( 255 );
  for( int i=0; i<50; i++ ){
    balls[i].move();
    balls[i].display();
  }
}
```

型 [] 配列名 = new 型 [要素数];



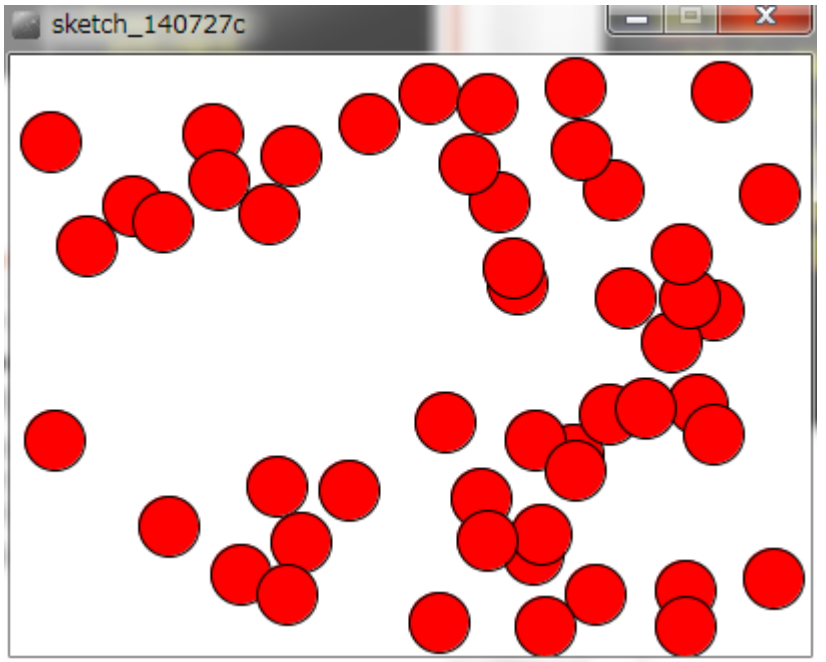
オブジェクト＋配列

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 50個の丸を動かすには配列を使う！

```
Ball [] balls = new Ball [50];

void setup() {
  size( 400, 300 );
  fill( 255, 0, 0 );
  for( int i=0; i<balls.length; i++ ){
    balls[i] = new Ball();
  }
}

void draw(){
  background( 255 );
  for( int i=0; i<balls.length; i++ ){
    balls[i].move();
    balls[i].display();
  }
}
```

配列変数名.length
で配列の長さを取得



オブジェクト＋配列

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 300個の丸を動かすには配列の定義を変更

```
Ball [] balls = new Ball [300];

void setup() {
  size( 400, 300 );
  fill( 255, 0, 0 );
  for( int i=0; i<balls.length; i++ ){
    balls[i] = new Ball();
  }
}

void draw(){
  background( 255 );
  for( int i=0; i<balls.length; i++ ){
    balls[i].move();
    balls[i].display();
  }
}
```

配列変数名.length
で配列の長さを取得

名前を表示してみよう

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 名前用のインスタンス変数を追加
  - String name;
- 名前をセットするインスタンスメソッドを追加
  - void setName(String s);

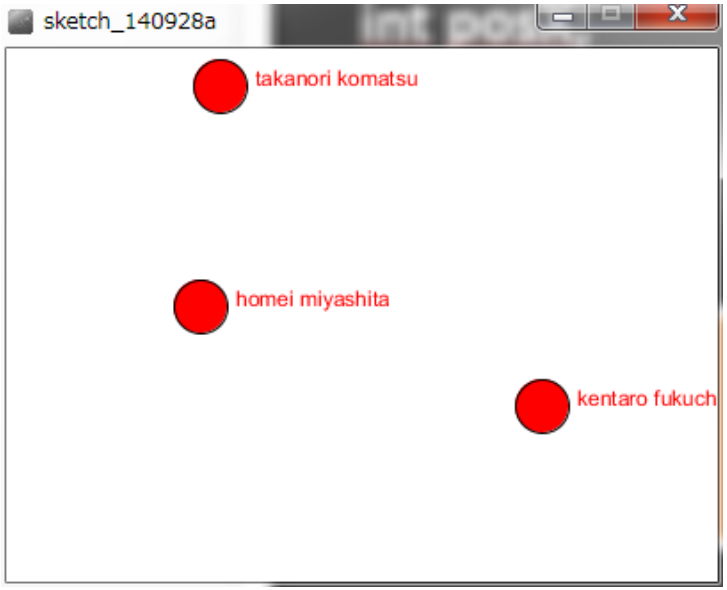
```
class Ball{
  int posX;
  int posY;
  int speedX;
  int speedY;

  String name;
  void display(){
    ellipse( posX, posY, 30, 30 );
    text( name, posX, posY );
  }
  void setName( String s ){
    name = s;
  }
```

名前を表示してみよう

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- setName メソッドを利用して名前を設定

```
Ball miyashita;
Ball komatsu;
Ball fukuchi;
void setup() {
  size( 400, 300 );
  fill( 255, 0, 0 );
  miyashita = new Ball();
  komatsu = new Ball();
  fukuchi = new Ball();
  miyashita.setName( "homei miyashita" );
  komatsu.setName( "takanori komatsu" );
  fukuchi.setName( "kentaro fukuchi" );
}
```



コンストラクタで名前を設定

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- setName で名前を設定したが，コンストラクタ（最初に呼び出される部分）で名前を設定することも可能

```
Ball miyashita;
Ball komatsu;
Ball fukuchi;

void setup() {
  miyashita = new Ball("miyashita homei");
  komatsu = new Ball("komatsu takanori");
  fukuchi = new Ball("fukuchi hiroaki");
}
```

```
class Ball{
  int posX;
  int posY;
  int speedX;
  int speedY;
  int red;
  int green;
  int blue;
  String name;

  Ball( String s ){
    name = s;
    init();
  }
```

コンストラクタの
引数を変更

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

継承

継承？

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 大辞林 第三版
  1. 先の人の身分・権利・義務・財産などを受け継ぐこと。「王位を―する」
  2. インヘリタンス →（オブジェクト指向プログラミングにおいて，クラス間でデータの共有を行う機構。新しく定義するクラスを既存のクラスの下位クラスとして記述し，上位クラスより属性やメソッドを引き継ぐ仕組みをいう。上位クラスに対する差分のみを記述するだけで新しいクラスを定義することが可能となる。）

3つの四角形と3つの丸

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

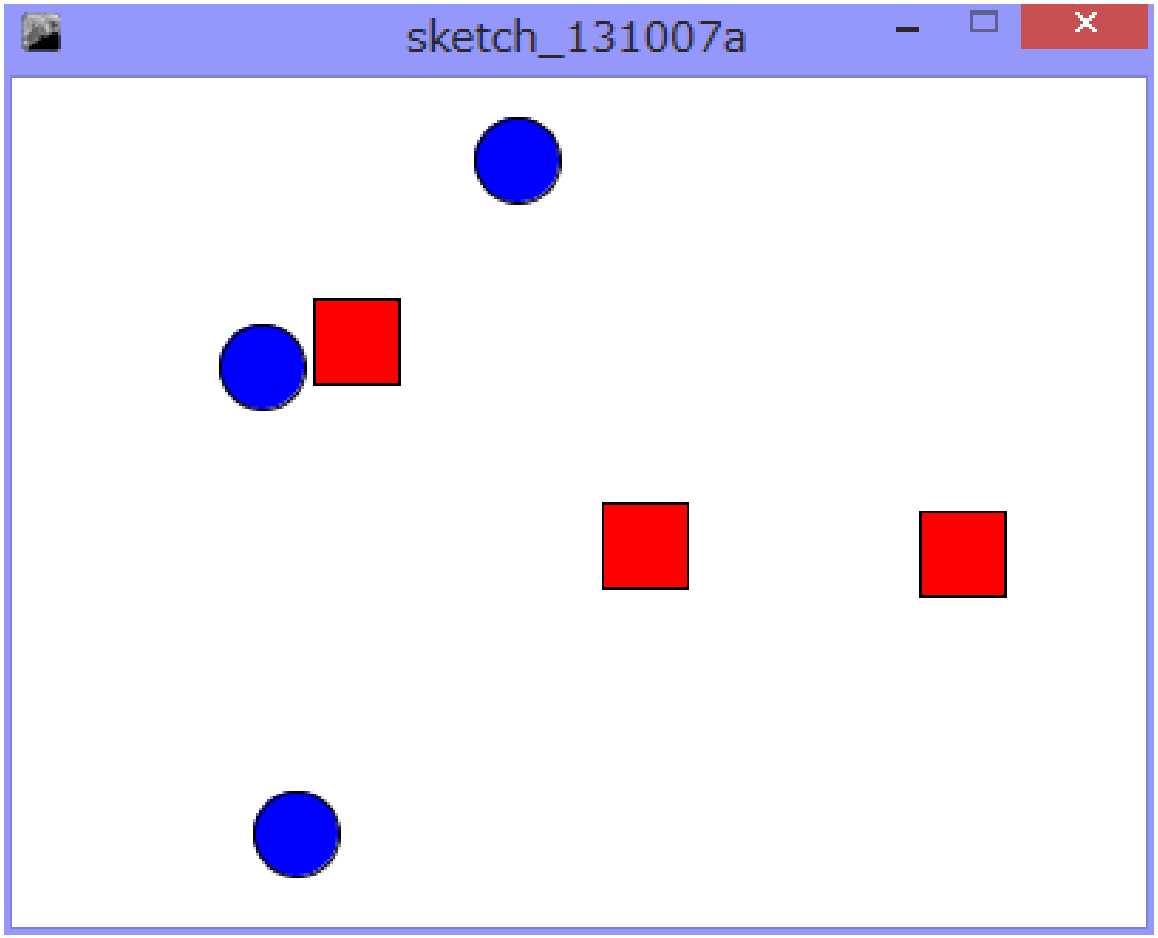
(Q2) 400x300のウインドウ内で，任意の場所から毎フレームx方向，y方向に任意の速度で移動する赤色の3つの丸と，青色の3つの四角形を描画し，右端・左端・上端・下端に来ると跳ね返るようにするには？

- 考え方
  - 右端・左端・上端・下端で衝突する時の条件を整理
  - 衝突した時の速度を反転させる
    - speedX = -speedX;
    - speedY = -speedY;

3つの四角形と3つの丸

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 前に作ったBallクラスと，今回作ったSquareクラスを組み合わせる



Ballクラス

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

```
class Ball{
   int x;
   int y;
   int speedX;
   int speedY;

   Ball(){
      init();
   }
   void init(){
      x = (int)random(width);
      y = (int)random(height);
      speedX = (int)random(5);
      speedY = (int)random(5);
   }
   void display(){
      fill( 255, 0, 0 );
      ellipse( x, y, 30, 30 );
   }
```

```
   void move(){
      x = x + speedX;
      y = y + speedY;
      if ( x+15 > width ) {
         x = width - 15;
         speedX = -speedX;
      }
      if( x - 15 < 0 ){
         x = 15;
         speedX = -speedX;
      }
      if( y + 15 > height ){
         y = height - 15;
         speedY = -speedY;
      }
      if( y - 15 < 0 ){
          y = 15;
          speedY = -speedY;
      }
   }
}
```

Ballを改良しSquareを作る

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

```
class Square{
   int x;
   int y;
   int speedX;
   int speedY;

   Square(){
      init();
   }
   void init(){
      x = (int)random(width);
      y = (int)random(height);
      speedX = (int)random(5);
      speedY = (int)random(5);
   }
   void display(){
      fill( 0, 0, 255 );
      rect( x-15, y-15, 30, 30 );
   }
```

```
   void move(){
      x = x + speedX;
      y = y + speedY;
      if ( x+15 > width ) {
         x = width - 15;
         speedX = -speedX;
      }
      if( x - 15 < 0 ){
         x = 15;
         speedX = -speedX;
      }
      if( y + 15 > height ){
         y = height - 15;
         speedY = -speedY;
      }
      if( y - 15 < 0 ){
          y = 15;
          speedY = -speedY;
      }
   }
}
```

3つの円と3つの四角形を移動

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室


```
Ball miyashita;
Ball komatsu;
Ball kikuchi;
Square fukuchi;
Square nakamura;
Squre hashimoto;
void setup() {
  size( 400, 300 );
  fill( 255, 0, 0 );
  miyashita = new Ball();
  komatsu = new Ball();
  kikuchi = new Ball();
  fukuchi = new Square();
  nakamura = new Square();
  hashimoto = new Square();
}
```

```
void draw() {
  background(255);
  // 移動
  miyashita.move();
  komatsu.move();
  kikuchi.move();
  fukuchi.move();
  nakamura.move();
  hashimoto.move();
  // 描画
  miyashita.display();
  komatsu.display();
  kikuchi.display();
  fukuchi.display();
  nakamura.display();
  hashimoto.display();
}
```

ここで・・・

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- Ballクラスと，Squareクラスはほとんど一緒
- 違いはクラス名とコンストラクタ，そしてdisplayのインスタンスメソッドのみ

無駄じゃね？

ほとんど一緒

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

	Ball	Square	
インスタンス変数	x, y speedX, speedY	x, y speedX, speedY	一致
クラス名	Ball	Square	違う
コンストラクタ	Ball()	Square()	内部は一致
void init()			一致
void move()			一致
void display()	fill(0, 0, 255); ellipse(x, y, 30, 30);	fill(0, 0, 255); rect(x-15, y-15, 30, 30);	違う

一緒の部分をまとめた
スーパークラス（親クラス）を作る

スーパーなObjectクラス

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

```
class Object{
    int x;
    int y;
    int speedX;
    int speedY;

    Object(){
        init();
    }
    void init(){
        x = (int)random(width);
        y = (int)random(height);
        speedX = (int)random(5);
        speedY = (int)random(5);
    }
```

```
    void move(){
        x = x + speedX;
        y = y + speedY;
        if ( x+15 > width ) {
            x = width - 15;
            speedX = -speedX;
        }
        if( x - 15 < 0 ){
            x = 15;
            speedX = -speedX;
        }
        if( y + 15 > height ){
            y = height - 15;
            speedY = -speedY;
        }
        if( y - 15 < 0 ){
            y = 15;
            speedY = -speedY;
        }
    }
}
```

display() は内容が違うので
削除してしまう

一緒の部分をまとめる

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- Ball は Object の変数（x, y, speedX, speedY）や機能（移動や初期化）をもち，独自の表示に関する機能（メソッド）をもつクラス
- Square はObject の変数（x, y, speedX, speedY）や機能（移動や初期化）をもち，独自の表示に関する機能（メソッド）をもつクラス

- インスタンス変数や，インスタンスメソッドを引き継ぐことを継承と呼ぶ！

Objectクラスを使うと

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室


```
class Ball extends Object {
    void display(){
        fill( 255, 0, 0 );
        ellipse( x, y, 30, 30 );
    }
}

class Square extends Object {
    void display(){
        fill( 0, 0, 255 );
        rect( x-15, y-15, 30, 30 );
    }
}
```

```
Ball miyashita;
Ball komatsu;
Ball kikuchi;
Square fukuchi;
Square nakamura;
Squre hashimoto;
void setup() {
  size( 400, 300 );
  fill( 255, 0, 0 );
  miyashita = new Ball();
  komatsu = new Ball();
  kikuchi = new Ball();
  fukuchi = new Square();
  nakamura = new Square();
  hashimoto = new Square();
}
      :
```

BallクラスとSquareクラスが劇的に短く！

なぜ動くの？

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 継承すると，親の力をすべて引き継ぐ！
- 継承の方法は extends とやるだけ！

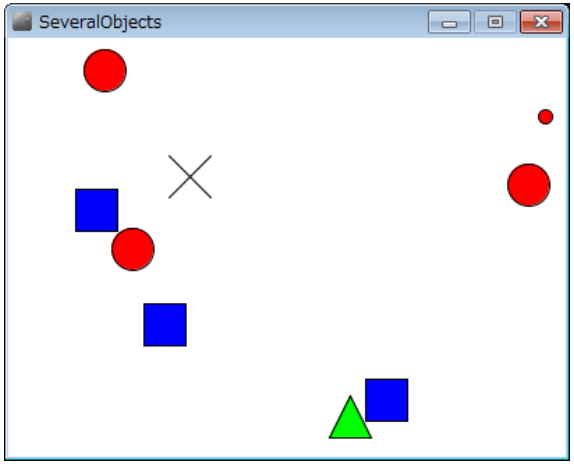
```
class クラス名 extends 親クラス名 {

}
```

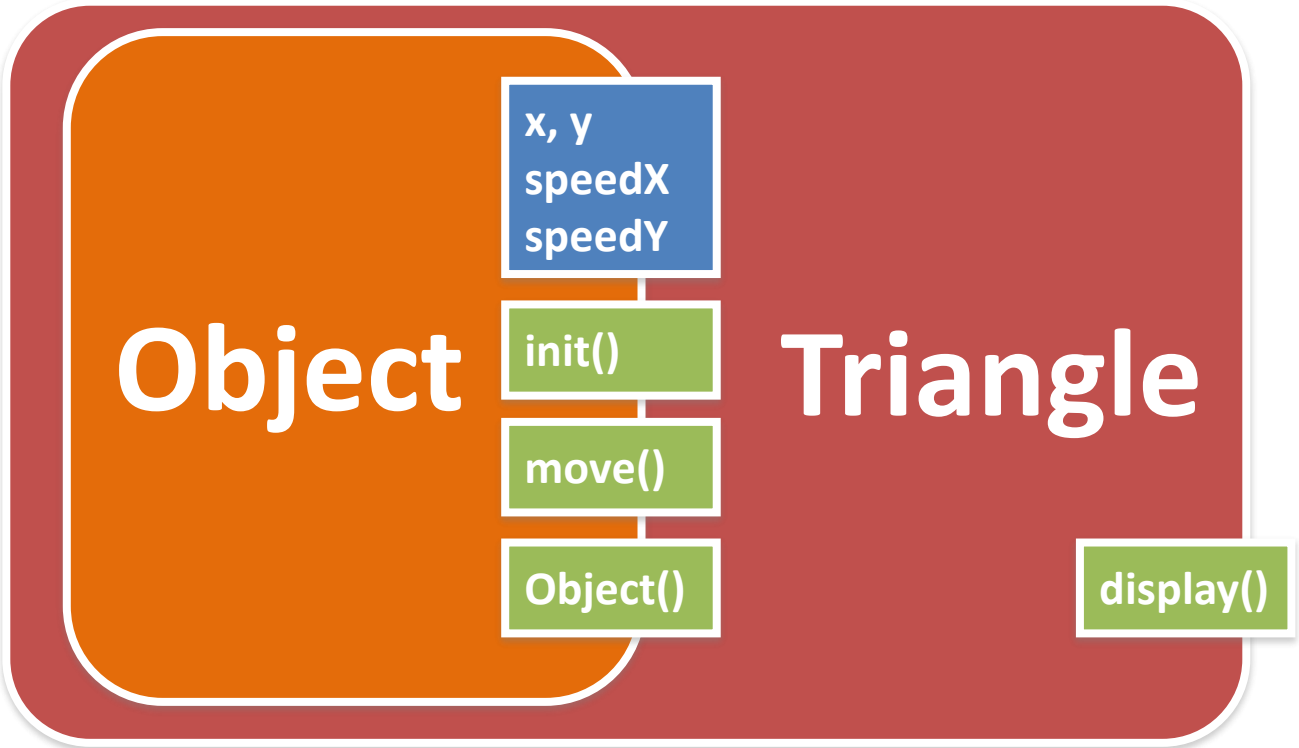
- 継承により親の能力，値はすべて引き継ぎます

じゃあ

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- Object クラスを継承して「三角形」を描画するクラスを作るには？（Triangle）
- Object クラスを継承して「×」を描画するクラスを作るには？（Cross）





三角形と×のクラス

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 動く三角形のクラスと動く×のクラス

```
class Triangle extends Object {
    void display(){
        fill( 0, 255, 0 );
        triangle( x, y-15, x-15, y+15, x+15, y+15 );
    }
}

class Cross extends Object {
    void display(){
        fill( 0, 255, 0 );
        line( x-15, y-15, x+15, y+15 );
        line( x-15, y+15, x+15, y-15 );
    }
}
```

つまり先週の宿題は…

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- Ball クラスを改良し，☆が動き回るStarクラスを作成せよ．また，BallクラスとStarクラスを利用して，50個の丸と50個の星を動かすようにせよ．
  - 星の内部は塗りつぶせるようだったら塗りつぶせ

**Objectクラスを継承して
Starクラスを作り，displayだけ
独立させる！**

参考：http://blog.livedoor.jp/reona396/archives/54602822.html

三角形は跳ね返らないように

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

(Q3) 先述のObjectを継承したTriangleクラスを改良し，三角形は跳ね返らず右端→左端，左端→右端，上端→下端，下端→上端と移動するようにせよ

- 考え方
  - Object の move メソッドを Triangle クラス内でオーバーライドして，三角形専用のメソッドを作成する

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

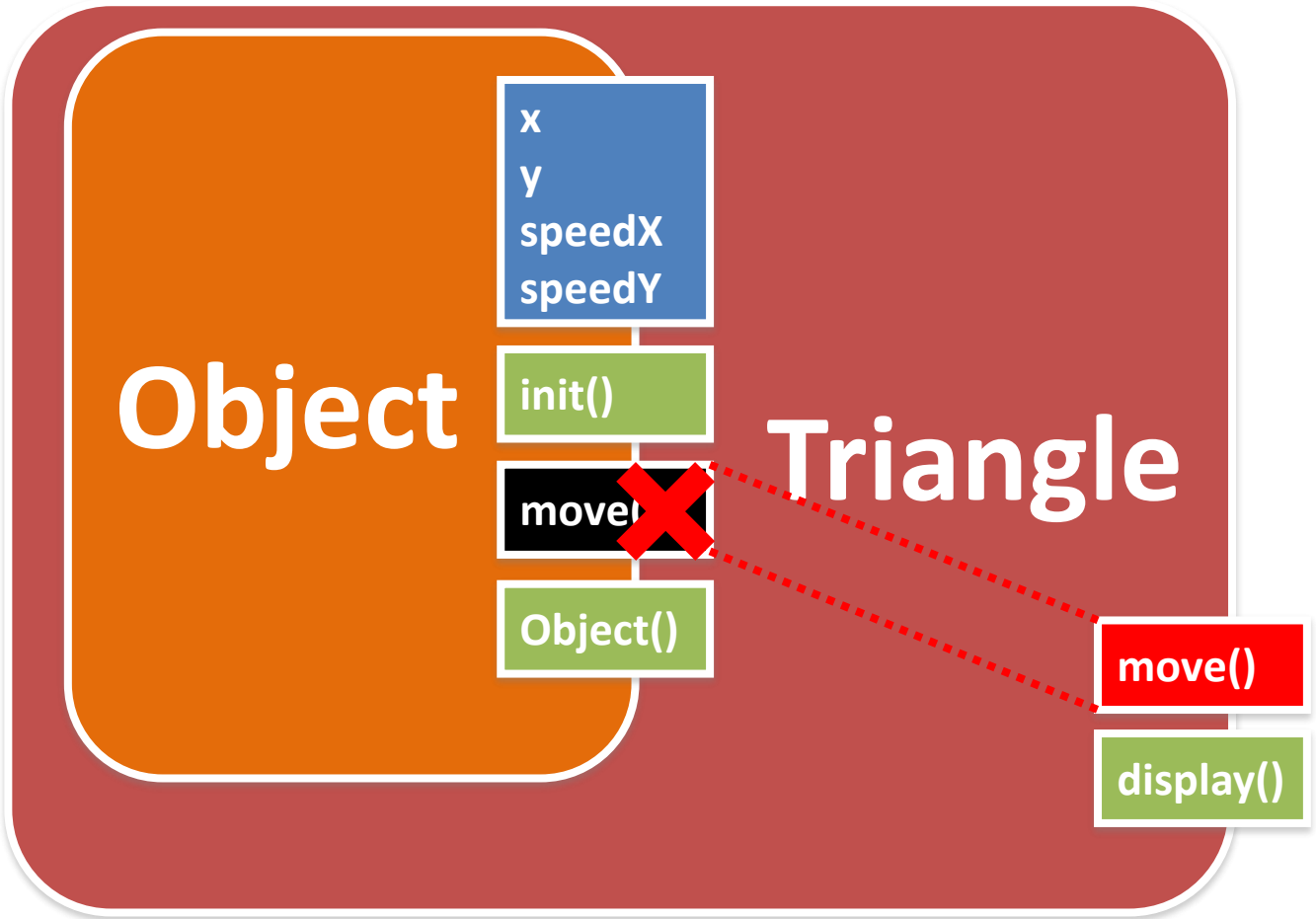
```
class Triangle extends Object {
    void display(){
        fill( 0, 255, 0 );
        triangle( x, y-15, x-15, y+15, x+15, y+15 );
    }
    void move(){
        x = x + speedX;
        y = y + speedY;
        if( x > width ){
            x = x - width;
        } else if( x < 0 ){
            x = width + x;
        }
        if( y > height ){
            y = y - height;
        }
        if( x < 0 ){
            y = width + y;
        }
    }
}
```

move メソッドをオーバーライドして親の move メソッドが呼ばれないようにする

move をオーバーライド

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- Triangle の move で，Object の move を上書きしてしまう！



全体の挙動を変更

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

(Q4) Objectクラスのmoveメソッドを変更し，上端及び下端は跳ね返るが，左端と右端では逆側から現れるようにせよ

- 考え方
  - Object の move メソッドのみ変更
  - y 座標の条件で跳ね返り
  - x 座標の条件で逆側から現れるようにする

Object クラスのみ変更

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

```
class Object{
    int x;
    int y;
    int speedX;
    int speedY;

    Object(){
        init();
    }
    void init(){
        x = (int)random(width);
        y = (int)random(height);
        speedX = (int)random(5);
        speedY = (int)random(5);
    }
```

```
    void move(){
        x = x + speedX;
        y = y + speedY;
        if( x > width ){
            x = x - width;
        } else if( x < 0 ){
            x = width + x;
        }

        if( y + 15 > height ){
            y = height - 15;
            speedY = -speedY;
        } else if( y - 15 < 0 ){
             y = 15;
             speedY = -speedY;
        }
    }
}
```

惑星と衛星の様なオブジェクト

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

(Q5) 400x300のウインドウ内で，任意の場所x，yから任意の速度で移動する3つの赤色の円を描画し，右端・左端・上端・下端に来ると跳ね返るようにする．また，赤色の円には円の中心から30の距離があるところに1つの衛星があり，10度ずつ円の周りを回転するようにせよ

- 考え方
  - 円の中心（x, y）から衛星の方向の角度（0～360度）を theta とすると，衛星の座標は

（x+30*cos(radians(theta)), y+30*sin(radians(theta)) ）

惑星と衛星の様なオブジェクト

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

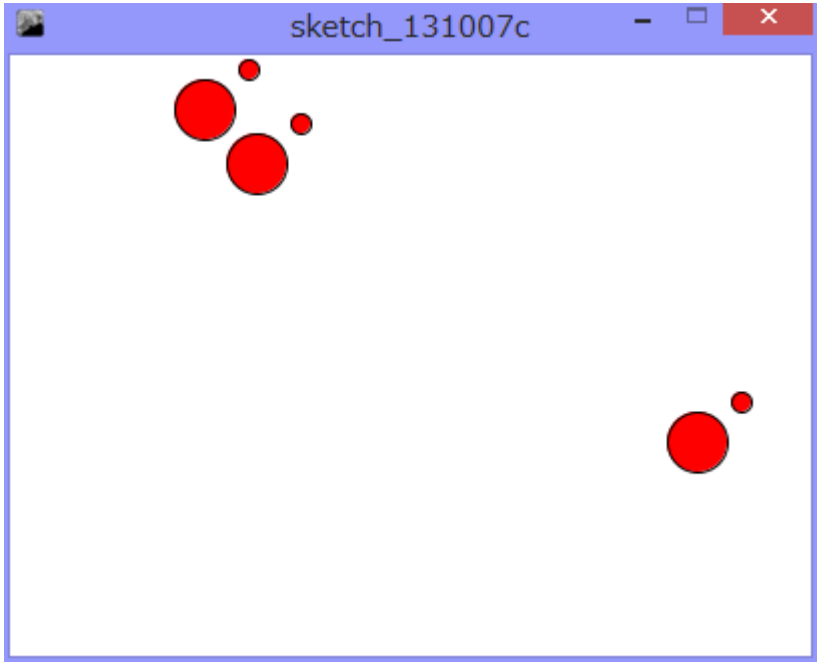
- Objectクラスを継承して，変数を追加する

```
class PlanetSatellite extends Object {
    int theta;
    void display(){
        fill( 255, 0, 0 );
        ellipse( x, y, 30, 30 );
        theta = theta + 10;
        int rx = (int)(x+30*sin(radians(theta)));
        int ry = (int)(y+30*cos(radians(theta)));
        ellipse( rx, ry, 10, 10 );
    }
}
```

惑星と衛星の様なオブジェクト

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

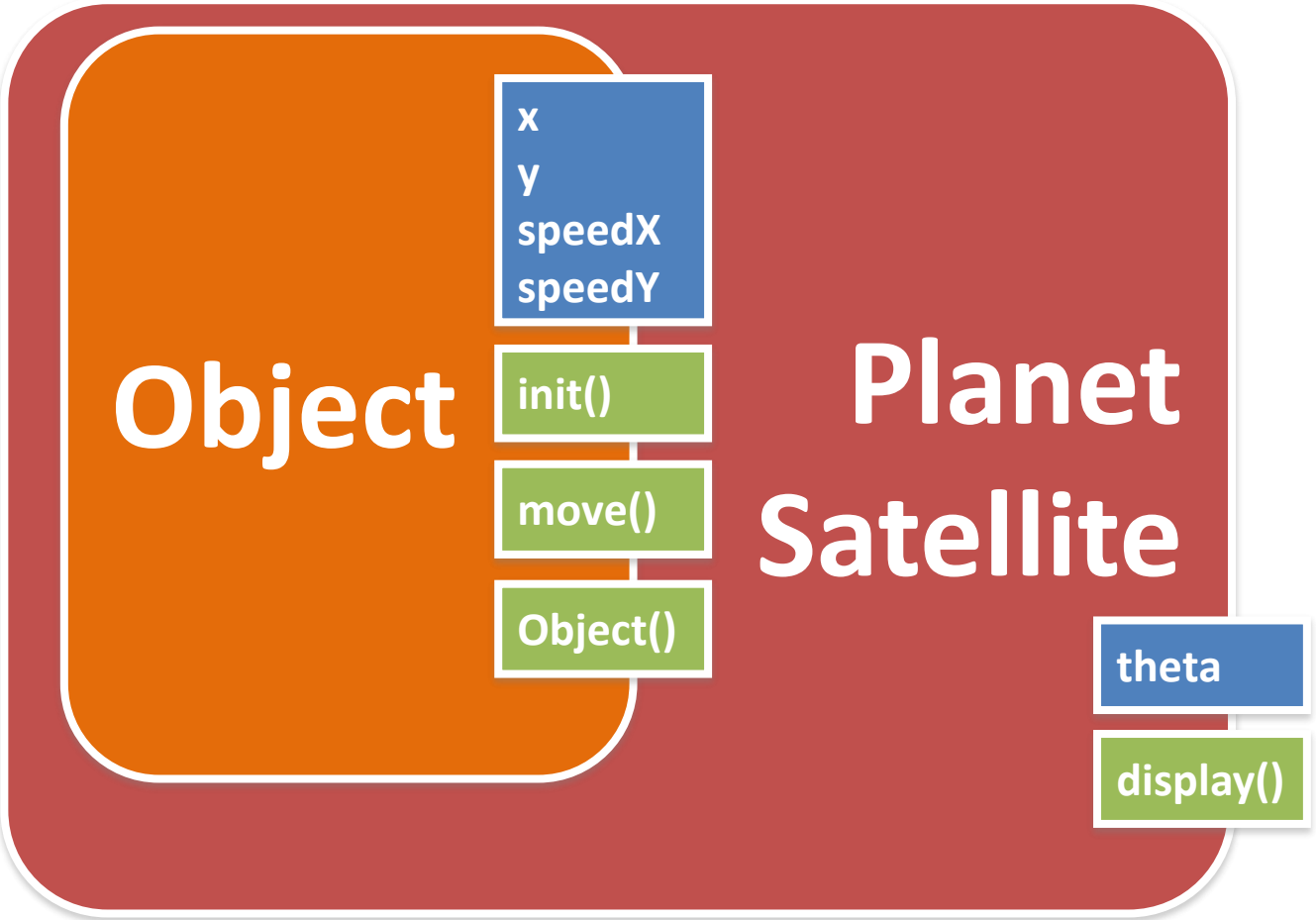
```
PlanetSatellite miyashita;
PlanetSatellite komatsu;
PlanetSatellite kikuchi;
void setup() {
    size( 400, 300 );
    miyashita = new PlanetSatellite();
    komatsu = new PlanetSatellite();
    kikuchi = new PlanetSatellite();
}

void draw() {
    background(255);
    miyashita.move();
    komatsu.move();
    kikuchi.move();
    miyashita.display();
    komatsu.display();
    kikuchi.display();
}
```



継承すると・・・

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- Object を PlanetSatellite として継承し，theta という変数と，display() というメソッドを追加

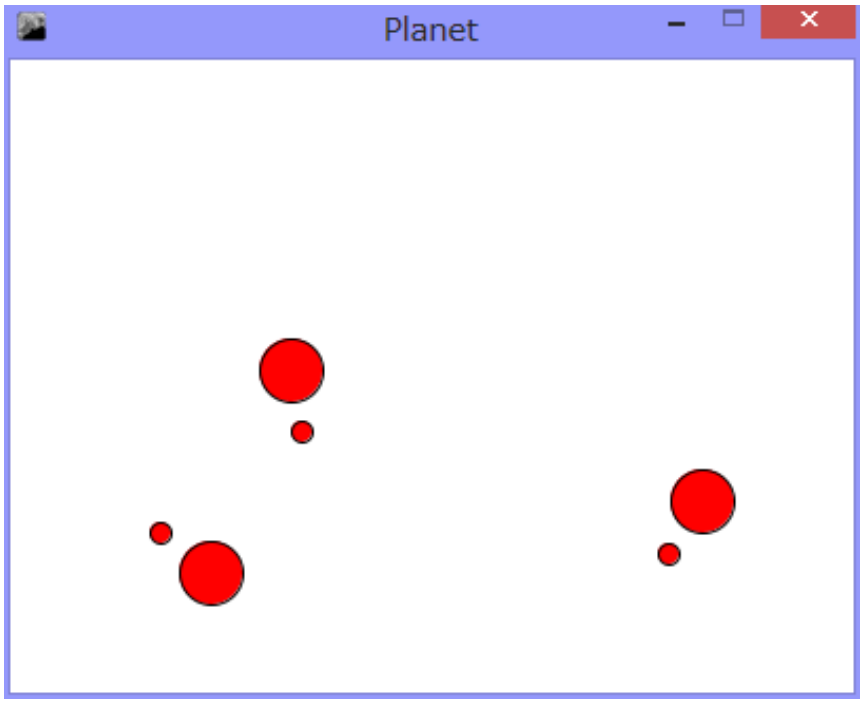


惑星と衛星の様なオブジェクト

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

(Q5) Q4を改良し，衛星の開始角を0～360度の任意の場所にしたい．どうするか？

- 考え方
  - void init() というインスタンスメソッドを追加し，改良したらOK？



やってみる

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

```
class PlanetSatellite extends Object {
    int theta;
    void init(){
        theta  = (int)random(360);
    }
    void display(){
        fill( 255, 0, 0 );
        ellipse( x, y, 30, 30 );
        theta = theta + 10;
        int rx = (int)(x+30*sin(radians(theta)));
        int ry = (int)(y+30*cos(radians(theta)));
        ellipse( rx, ry, 10, 10 );
    }
```

init() はコンストラクタで呼ばれる
Objectクラス参照

うまく動作しない

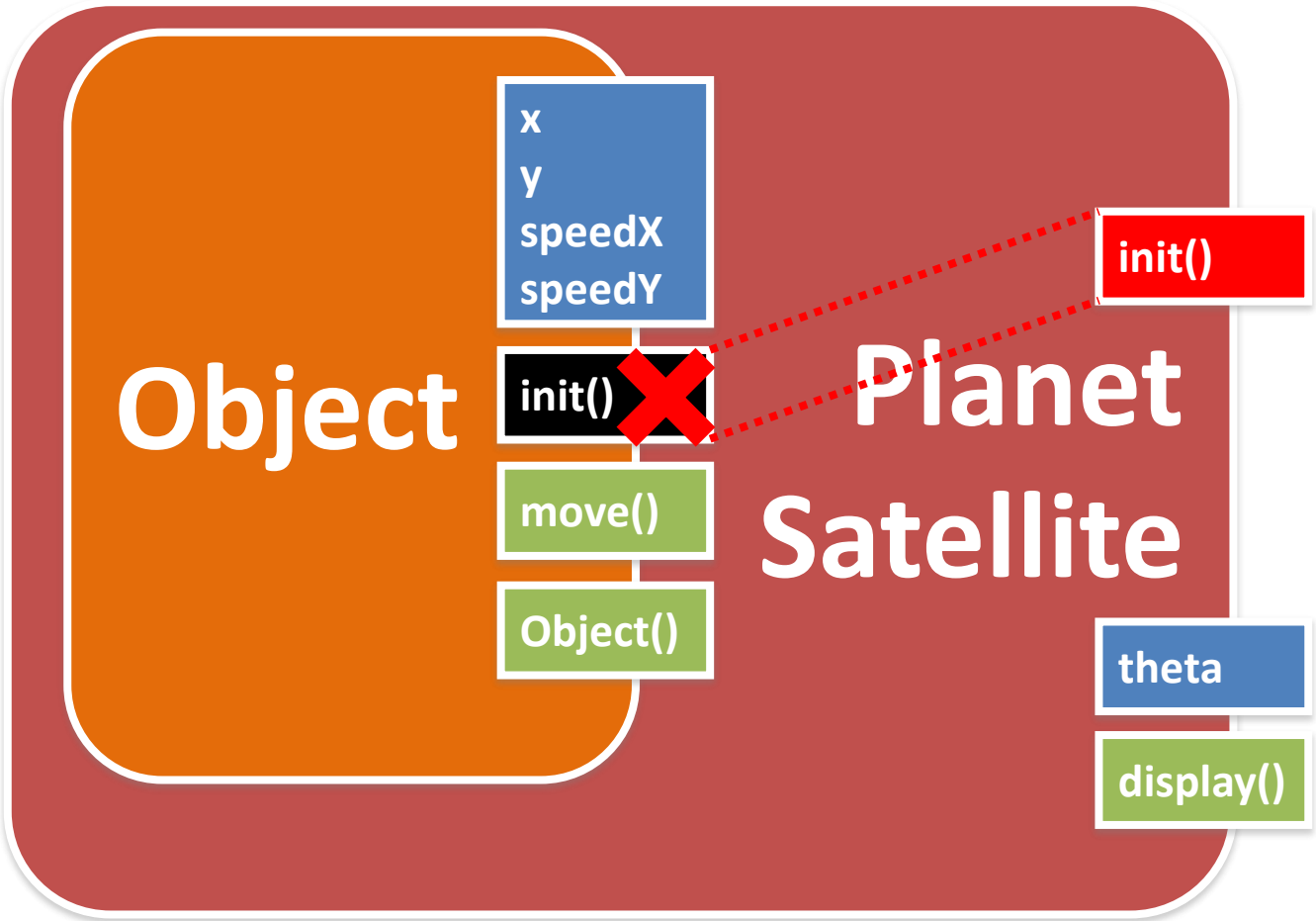
明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 画面の左上から動かない．何故か？

上書きしてしまったから

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- Object の init() を，PlanetSatellite の init() でオーバーライドている（Objectのinit()が呼び出されない）



```
class PlanetSatellite extends Object {
    int theta;
    void init(){
        theta  = (int)random(360);
        x = (int)random(width);
        y = (int)random(height);
        speedX = (int)random(5);
        speedY = (int)random(5);
    }
    void display(){
        fill( 255, 0, 0 );
        ellipse( x, y, 30, 30 );
        theta = theta + 10;
        int rx = (int)(x+30*sin(radians(theta)));
        int ry = (int)(y+30*cos(radians(theta)));
        ellipse( rx, ry, 10, 10 );
    }
}
```

Object の init() にあるのを
そのままコピペする

動くけど，なんだか
無駄が増えている・・・

Objectのinit()も使いたい

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- クラスの中で，super と書くと，継承元の親を呼び出すことができる！

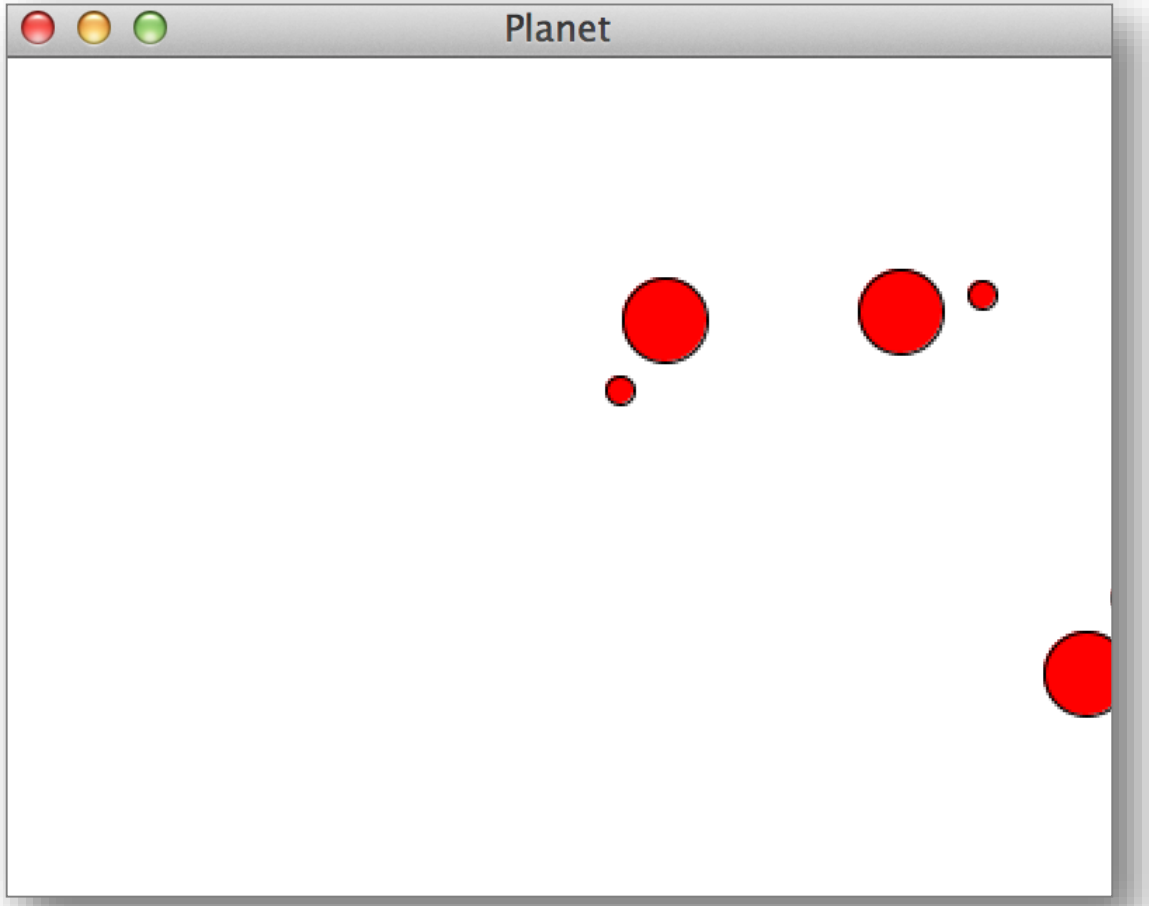
```
class PlanetSatellite extends Object {
    int theta;
    void init(){
        theta  = (int)random(360);
        super.init();
    }
    void display(){
        fill( 255, 0, 0 );
        ellipse( x, y, 30, 30 );
        theta = theta + 10;
        int rx = (int)(x+30*sin(radians(theta)));
        int ry = (int)(y+30*cos(radians(theta)));
        ellipse( rx, ry, 10, 10 );
    }
}
```

super . メソッド名

Objectのinit()も使えば…

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- PlanetSatelliteで定義したinit()の中で、Objectのinit()も呼べば、両方の処理が実行される！



明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

他の人が作ったクラスを使う

文字列型

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- String 型は，文字列を扱うためのクラス
  - 「中村聡史」「明治大学 総合数理学部」
- 文字列を扱う際にはどういった機能が必要？
  - 文字列の長さを取得する
  - 文字列にある文字が含まれているかを調べる
  - 文字列を部分的に置き換える
  - 文字列が一致しているか調べる
  - n文字目の文字を取得する
  - などなど

Stringクラスのメソッド

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- charAt(num);　num文字目の文字を返す（0から始まる）
- indexOf(文字列);　　入力された文字列が何文字目か？
- length();　　入力された文字の文字数を返す
- substring(x);　x文字目から最後までを出力
- substring(x, y);　x文字目からy-1文字目までを出力
- toLowerCase();　全てを小文字に変換する
- toUpperCase();　全てを大文字に変換する
- replace(文字列A, 文字列B);
  - 文字列Aを文字列Bに変更する
- split(文字列);　文字列を分割

http://processing.org/reference/String.html
http://docs.oracle.com/javase/6/docs/api/java/lang/String.html

```
String str = "Department of Frontier Media Science (FMS), IMS, Meiji University";

println( str.length() );
println( str.charAt( 11 ) );
println( str.indexOf("F") );
println( str.indexOf("S") );
println( str.indexOf("Meiji") );
println( str.substring( str.indexOf("Meiji") ) );
println( str.toLowerCase() );
println( str.toUpperCase() );
```

```
65
o
14
29
49
Meiji University
department of frontier media science (fms), ims, meiji university
DEPARTMENT OF FRONTIER MEDIA SCIENCE (FMS), IMS, MEIJI UNIVERSITY
```

```
String str = "Department of Frontier Media Science (FMS), IMS, Meiji University";

String [] ret = str.split( " " );
println( ret.length ) ;
int i=0;
while( i<ret.length ){
  println( ret[i] );
  i++;
}
```

```
9
Department
of
Frontier
Media
Science
(FMS),
IMS,
Meiji
U
```

PImage / 画像型

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 画像を格納および描画するクラス
  - .width や .height で画像の縦横のサイズ取得
  - .resize() で画像サイズを変更可能
  - .save() で画像を保存可能
  - .filter() で各種フィルタをかけることが可能

```
PImage img;
size( 400, 400 );
img = loadImage( "画像ファイル名" );
img.filter( BLUR, 6 );
image( img, 0, 0 );
```

フィルタ例 ()内はオプション
THRESHOLD (0-1.0)
GRAY
OPAQUE
INVERT
POSTERIZE (2-255)
BLUR (1以上．半径)
ERODE
DILATE

http://www40.atwiki.jp/spellbound/pages/1800.html
http://processing.org/reference/PImage_filter_.html

StringやPimage以外にも

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

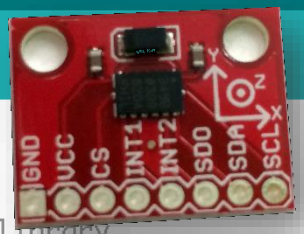
Toast_wADXL345 | Arduino 1.0.5

Toast_wADXL345 §

```
#include <Wire.h>
#include <ADXL345.h>

ADXL345 adxl; //variable adxl is an instance of the ADXL345 library

void setup(){
  pinMode(led, OUTPUT);
  Serial.begin(9600);
  adxl.powerOn();

  //set activity/ inactivity thresholds (0-255)
  adxl.setActivityThreshold(75); //62.5mg per increment
  adxl.setInactivityThreshold(75); //62.5mg per increment
  adxl.setTimeInactivity(10); // how many seconds of no activity is
```

```
void loop(){

  //Boring accelerometer stuff
  int x,y,z;
  adxl.readAccel(&x, &y, &z); //read the accelerometer values and store them in variab

  // Output x,y,z values - Commented out
```

保存しました。



ADXL345

Arduinoの例：

便利なライブラリも
クラスのおかげ

課題4-1 Robotクラスを使う

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- Robot.txtの内容を用いて、Robotクラスを作成し、それを表示するプログラムを作成せよ
- マウスが押されたらdisplayではなく、pressedメソッドを呼ぶこと
  - 考え方
    - 新しいタブにRobotクラスを定義（何も変えないで良い）
    - setup()やdraw()は自分で書いて、そこからRobotクラスを用いてRobotを表示
    - ウィンドウサイズは800×600程度にしておく

課題4-2 Robotクラスを拡張

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 4-1で作成したRobotクラスを継承した、自分のデザインの「マイロボットクラス」を定義し、それを表示するプログラムを作成せよ。
- ロボットクラス作成するクラスの名称は次のようにする（タブの名称も同じ）

  複数作った人は、A,B,Cと変える

  SatoRobo0301A　名字+”Robo”+組+番号+英字

  –考え方
  - 新しいタブを作成してマイロボットクラスを定義（タブの名前はクラス名と同じにする）
  - display()　をオーバーライド
  - 必要なら他もオーバーライド

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

ここまでできたら、プログラムと、自分で作ったマイロボットクラスだけのファイルを提出！

課題4-3 Robotクラスを交換

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 4-2で他の人が作成したマイロボットクラスと、自分のマイロボットクラス両方を用いて、ロボットを表示するプログラムを作成せよ
  - 考え方
    - 資料共有フォルダから他の人のをコピーして使用（基本的に隣の人のを使う）
    - マイロボットクラスを提出する際の気配り：

      (1)基本的なメソッドは、コンストラクタと、void init(), void move(), void display(), void pressed() の4個。他の機能がある場合はコメントに書いて使ってもらおう！

      (2) .pdeファイルだけを共有するので、画像や音声等の外部ファイルは使わない。
    - Enjoy!