



1. クラス、オブジェクトとはandroid概要
2. 文字と画像の表示
3. Intentにより、画面遷移を実装。

1. クラスとオブジェクト

型の種類

Javaには、大きく分けて変数に以下の2つの型が存在する。

プリミティブ型

参照型

プリミティブ型（基本データ型）

データを格納するための、最も基本的な型である。

整数値や実数値といった、**具体的な「値」**を格納するための変数を作るための型が「プリミティブ型」である。

Javaにおけるプリミティブ型には以下の8つが存在する。

- ・ byte
- ・ short
- ・ int
- ・ long
- ・ char
- ・ float
- ・ double
- ・ boolean

プリミティブ型は頭が小文字である。

それ以外は全て参照型！

クラスを作成しさえすれば、いくらでも参照型とする型を増殖することが出来る。

1. クラス、オブジェクトとは

・クラスについて

クラスとは**物理的に存在する実体**を生成する元となる設計図のようなもの。

・オブジェクトについて

クラスを元に生成された**物理的な実体**をオブジェクトという。

例えば、車を製造する為に必ず設計図が存在するはず。

でも、その設計図が段階では、当然我々は、車に乗ったり操作したりすることは出来ない。

実際に車に乗ったり、操作する為には、設計図から物理的な物体としての車を作る必要があります。

クラスからオブジェクトを作成するには“new”を使います。

newする前は、ただの設計書として存在しているだけ。物理的なプログラムとしての実体はまだない。

オブジェクトが作成されて、初めて**プログラムとなって動かせる状態になっている。**

(設計図の段階では、プログラムは稼働していない。そもそも、プログラムという実体として誕生していない)

メンバー変数:

オブジェクトが持つ属性、特性。

メンバー関数:

オブジェクトが持つ機能。

1. クラス、オブジェクトとは

＜オブジェクトを操作すること。＞

- 一度物体にしてから(オブジェクトにしてから)、その属性を設定する、変更することが可能。(セッターの関数を実行する)
- オブジェクトの属性の情報を取り出すことが可能。(ゲッターの関数を実行する)
- 何かしらの機能を操作すること。(関数を実行する)
- 関数を実行する際、その時点でオブジェクトが持っている特性(属性のデータ)を利用することも可能。(メンバー変数を利用する。)
- クラスからオブジェクトを作成するには“new”を使います。

1. クラス、オブジェクトとは

とりあえず一個、クラスを作ってみましょう！

自分という人間のクラスを作る。
Myselfという名前でクラスを作ってみよう。

- 名前、趣味、年齢という属性がある。
- 名前を決定、変更する機能がある。
- 名前を取る機能がある。(名前を表現する)
- 挨拶をするという機能がある。

1. クラス、オブジェクトとは

```
public class MySelf {  
    private String hobby;  
    private String name;  
    private int age;  
  
    /**  
     * コンストラクタ  
     */  
    public MySelf () {  
        this.age = 0;  
    }  
  
    /**  
     * 自己紹介をする  
     * @return  
     */  
    public String doSelfIntruduction() {  
        String greeting = "こんにちは!"+"¥n";  
        if (this.name != null) {  
            greeting += "私は"+this.name + "です。¥n";  
        }  
        return greeting;  
    }  
  
    /**  
     * 名前を設定する  
     * @param name  
     */  
    public void setName (String name) {  
        this.name = name;  
    }  
  
    /**  
     * 名前を取る  
     * @return  
     */  
    public String getName () {  
        return this.name;  
    }  
}
```

メンバー変数
(クラスが持つ属性)

コンストラクタ
オブジェクトが生成された時に一番最初に必ず実行される関数

セッター関数
メンバー変数の
値を設定する。

ゲッター関数
メンバー変数の
値を取得する。

メンバー関数
(クラスが持つ機能)

1. クラス、オブジェクトとは

//MySelfクラスを実際にプログラムとして動かせる状態にしてみる。
(newする前は、ただの設計書として存在しているだけ。物理的なプログラムとしての実体はまだない)

```
MySelf myself = new MySelf();  
myself.setName("トニー");  
myself.setHobby("音楽");
```

```
String selfIntro = myself.doSelfIntroduction();
```

```
TextView nameView = (TextView) findViewById(R.id.name);  
nameView.setText(selfIntro);
```

ついでに、**MySelf**クラスに性別の属性を追加し、その性別を挨拶文に入れてみましょう！

2. クラス、オブジェクトとは

オブジェクト比較について

```
int[] age = new int[3];  
age[0] = 1;  
age[1] = 2;  
age[2] = 3;
```

```
int[] age2 = new int[3];  
age2[0] = 1;  
age2[1] = 2;  
age2[2] = 3;
```

```
if (age == age2) {  
    Log.d("hello_age_a", 1 + "");  
}
```

```
int[] age3 = age2;  
if (age3 == age2) {  
    Log.d("hello_age_b", 1 + "");  
}
```

```
age3[2] = 4;  
Log.d("hello_age_c", age2[2] + "");
```

2. クラス、オブジェクトとは

基本データ型変数と参照型変数の違い。

基本データ型変数と、オブジェクトのメモリ上の仕組みについて。

http://d.hatena.ne.jp/shuji_w6e/touch/20070211/1171204927

OSには必ずメモリというのがある。

一時的にデータを保持する領域。

スタックとヒープがある。

スタックに実データが格納されているか、(基本データ型)

ヒープに確保されたインスタンスを参照しているか (参照型)

の違い。

ヒープ領域にオブジェクト (実体) が生成され、

その際にオブジェクトにメモリアドレス (ポインタ) が割り振られる。 (オブジェクトのID値のようなもの)

そのメモリアドレスが、スタック内の変数に格納される。

2. クラス、オブジェクトとは

基本データ型の場合

メモリ領域

```
int number = 20;
```

スタック

変数、メソッドが格納される。

number変数

ヒープ領域

オブジェクトが格納される。

基本データ型変数
20という実データ

2. クラス、オブジェクトとは

参照型の場合

メモリ領域

```
int[] age = new int[2];
age[0] = 1;
age[1] = 2;
```

```
int[] age2 = age;
```

スタック

変数、メソッドが格納される。

aaaa

age変数

aaaa

age2変数

bbbb

nameView 変数

参照

参照

参照

ヒープ領域

オブジェクトが格納される。

aaaa

bbbb

参照型変数
格納されたメモリアドレスを元に
ヒープ領域のオブジェクトを参照する。

オブジェクト
new int[3]で実体となった
変数int[2]の実データ。

参照型変数

オブジェクト
(TextView) findViewById(R.id.**name**)で取り
出したTextViewクラスの実データ。

2. クラス、オブジェクトとは

参照型の場合

メモリ領域

//オブジェクトが生成され、オブジェクトへの参照が渡される。(ヒープ内の実データオブジェクトのメモリアドレスが変数myselfに格納される)
MySelf myself = new MySelf();

//ヒープ領域にはオブジェクトの実データがなく、メモリアドレスも変数nameViewには、格納されていない。 => つまり 、NULL
TextView nameView;

//ヒープ領域にオブジェクトの実体はある。空という実データのオブジェクトがヒープに確保され、そのメモリアドレスも生成され、変数nameに格納されている。
String name = "";

スタック

変数、メソッドが格納される。

ヒープ領域

オブジェクトが格納される。

2. クラス、オブジェクトとは

つまり、
スタック領域に格納された基本データ型の実データが定められた容量を超えると
メモリリーク！

ヒープ領域に格納されたオブジェクト（実データ）が定められた容量を超えると
メモリリーク！

アプリが落ちる。

1. クラス、オブジェクトとは

Viewのクラスについて

レイアウトファイルで定義されたViewもオブジェクト。
Activityを起動した時点でオブジェクトとして、メモリ上に配置されている。

1. クラス、オブジェクトとは

Viewのクラスについて

```
TextView nameView = (TextView) findViewById(R.id.name);
```

ー>

レイアウトファイルで定義したviewは全て元々クラスであり、Activityを立ち上げた時点でそれらのviewは全て、オブジェクトとして既に作成されている状態(設計図ではなく、物理的な実体)となっている。ID値nameのTextViewのオブジェクトを取り出している。

(ここでオブジェクト作成している訳ではない)

元となるクラス名が変数nameViewのデータ型となる！

```
TextView nameText = new TextView(this);
```

ー>

新規でTextViewクラスをnewして、オブジェクト化（設計図ではなく、物理的な実体）している。プログラム側から新たにTextViewを作成しActivityに追加したい時に利用する。

Intentにより、画面遷移を実装。