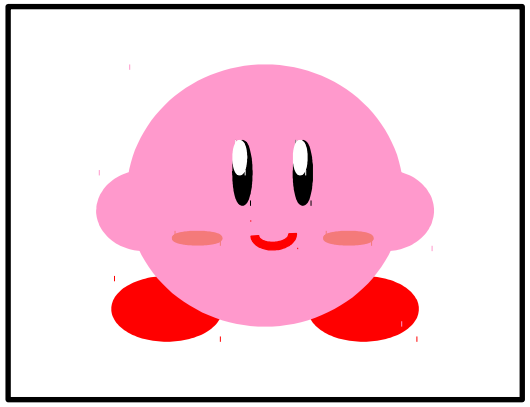
7章 継承について

継承とは？

- 一枚の絵



Aさんが欲しいといった絵を描いた（デジタル）

Bさんも絵が欲しい！+星も付けたして欲しい

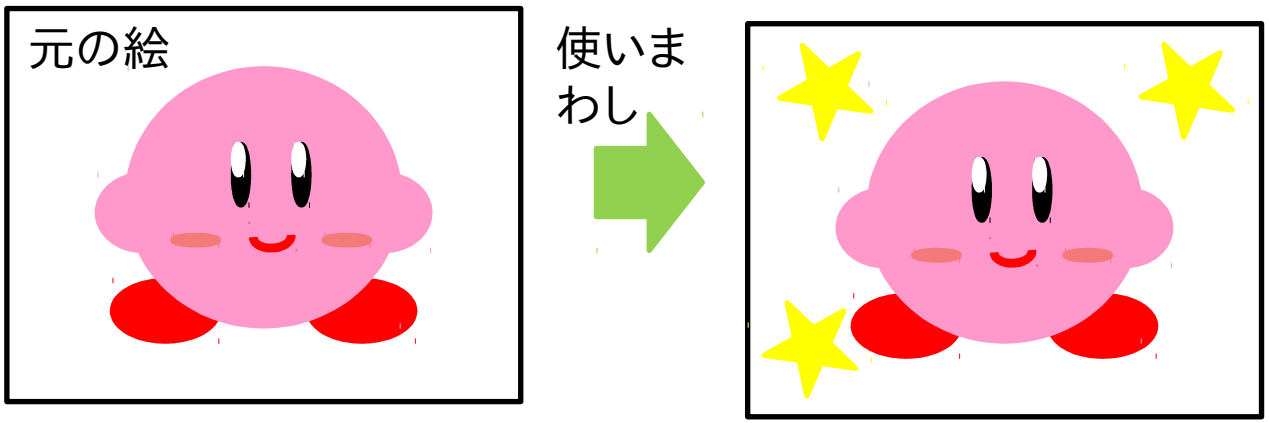
どうする？いちから描く？？

継承とは？

いちから描くのは面倒･･･

だったら････

保存していた絵を使いまわしして星だけ描きたす!!

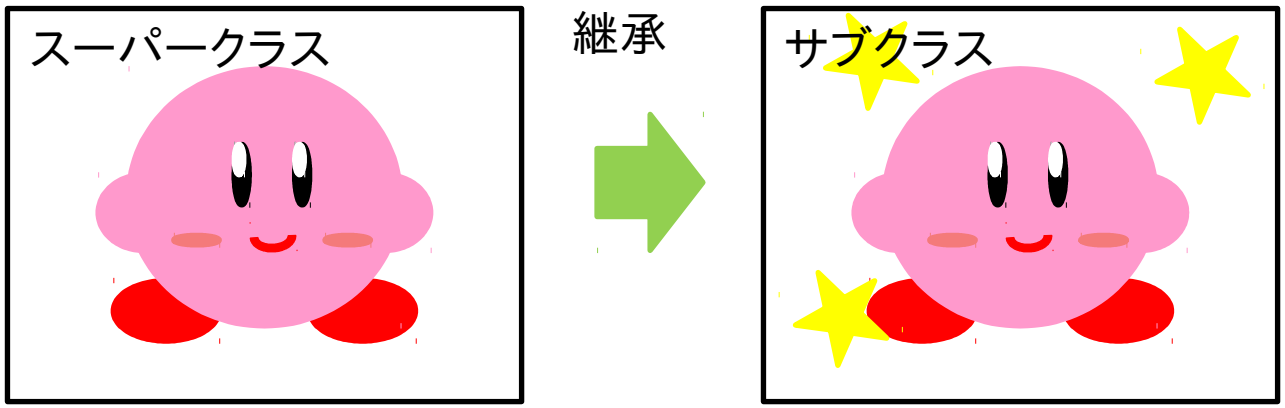


継承とは？

・プログラムでも同じような機能をつくって足りない部分だけつけたしたいと思うことが多々ある。

それを簡単にしたのが・・・・・・「継承」

元の絵となる方を「スーパークラス」、星を追加した方を「サブクラス」と呼ぶ。



継承とは？

継承の構文

class サブクラス　extends スーパークラス{}

継承のメリット
ただ単なるコピー&ペーストで複数つくってしまうと同じ修正を何個も入れないといけない。
→バグが発生しやすくなる！

継承を使用していれば、元のプログラムを修正するだけで自動的に
別のプログラムも同じ修正が反映される！！素敵！！

継承でできないこと

- クラスを継承できるのは1つ(複数は×)

× class A extends B,C{}

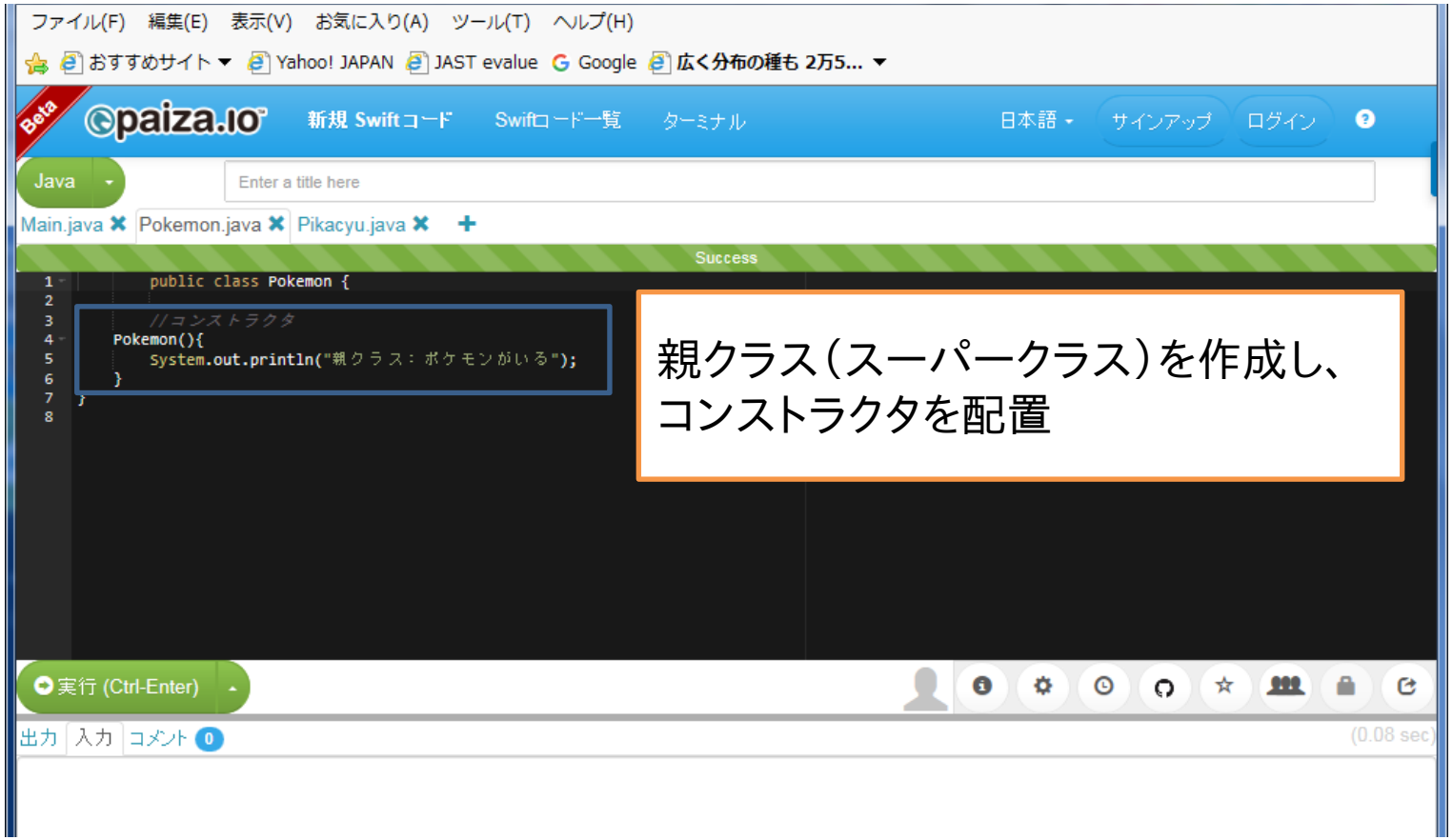
B,Cを継承したいなら・・・

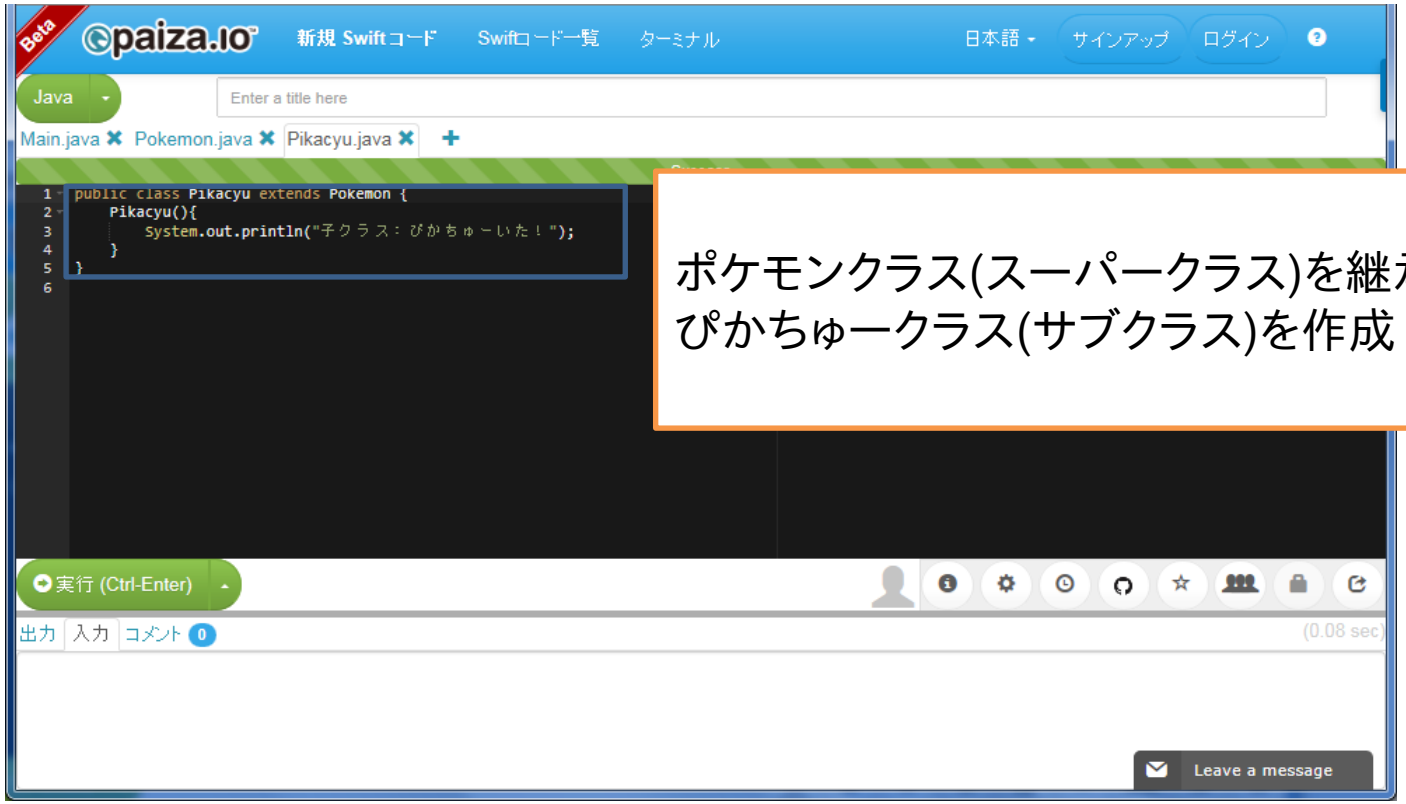
class B extends C{}

class A extendsB(Cを継承したBを使えます) {}

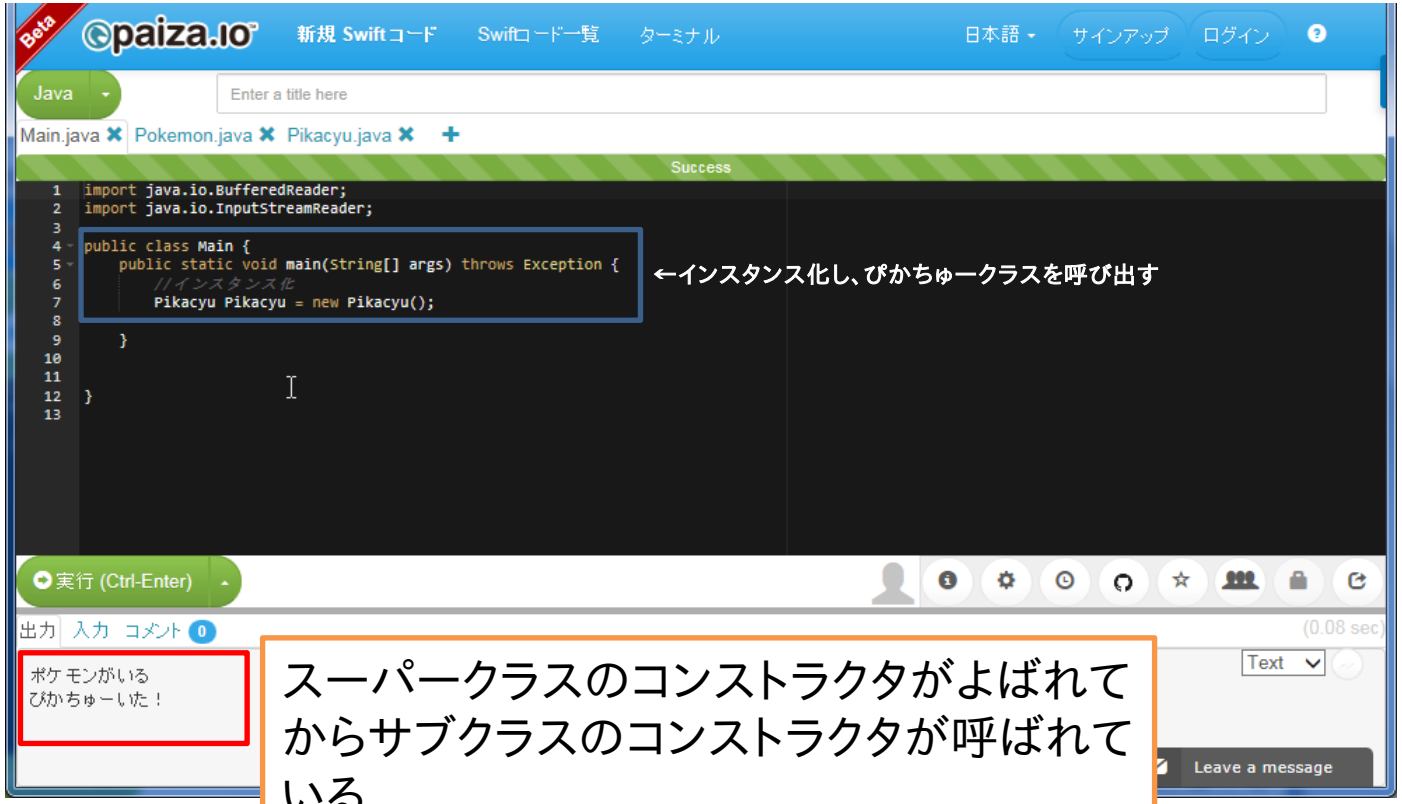
- Classの前にfinalをつけると継承はできなくなる

コンストラクタについて





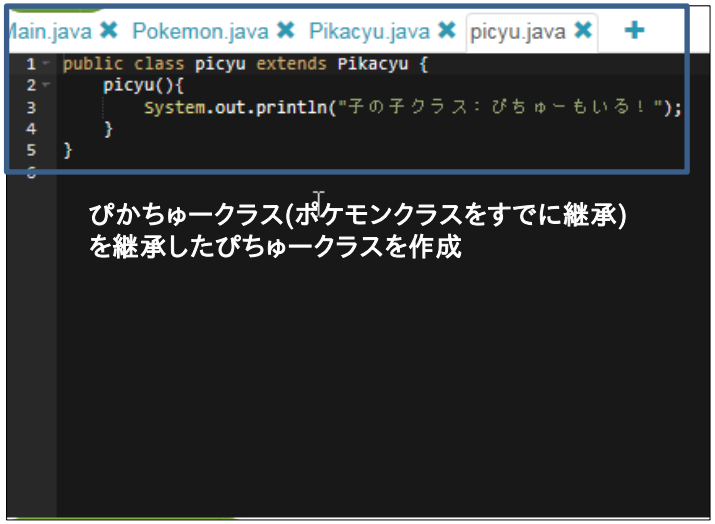
ポケモンクラス(スーパークラス)を継承した
ぴかちゅークラス(サブクラス)を作成

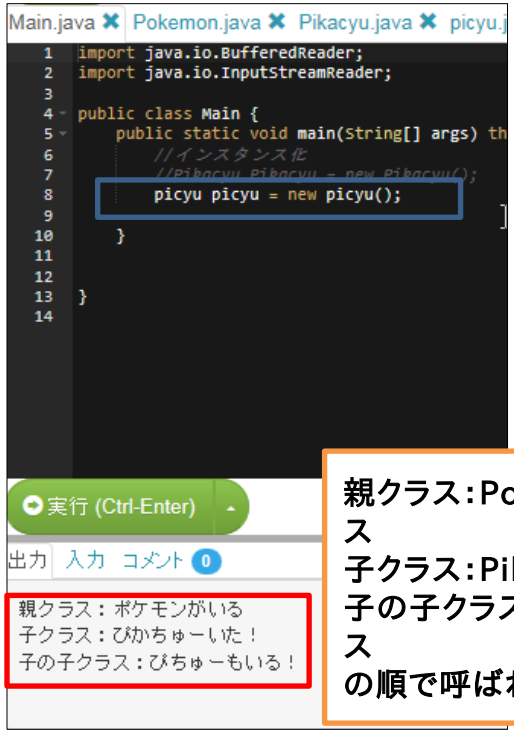


スーパークラスのコンストラクタがよばれてからサブクラスのコンストラクタが呼ばれている

- コンストラクタは、親クラス、子クラスの順で呼び出される

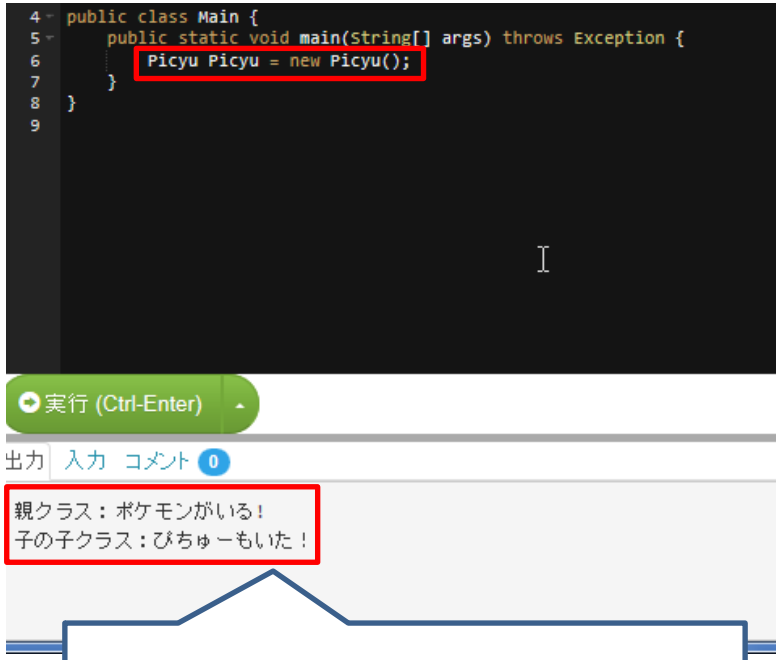
- 複数継承している場合は、一番元となるクラスから準備呼び出される。





- Superのコンストラクタとは？

一番元となるスーパークラスのコンストラクタを呼びだす



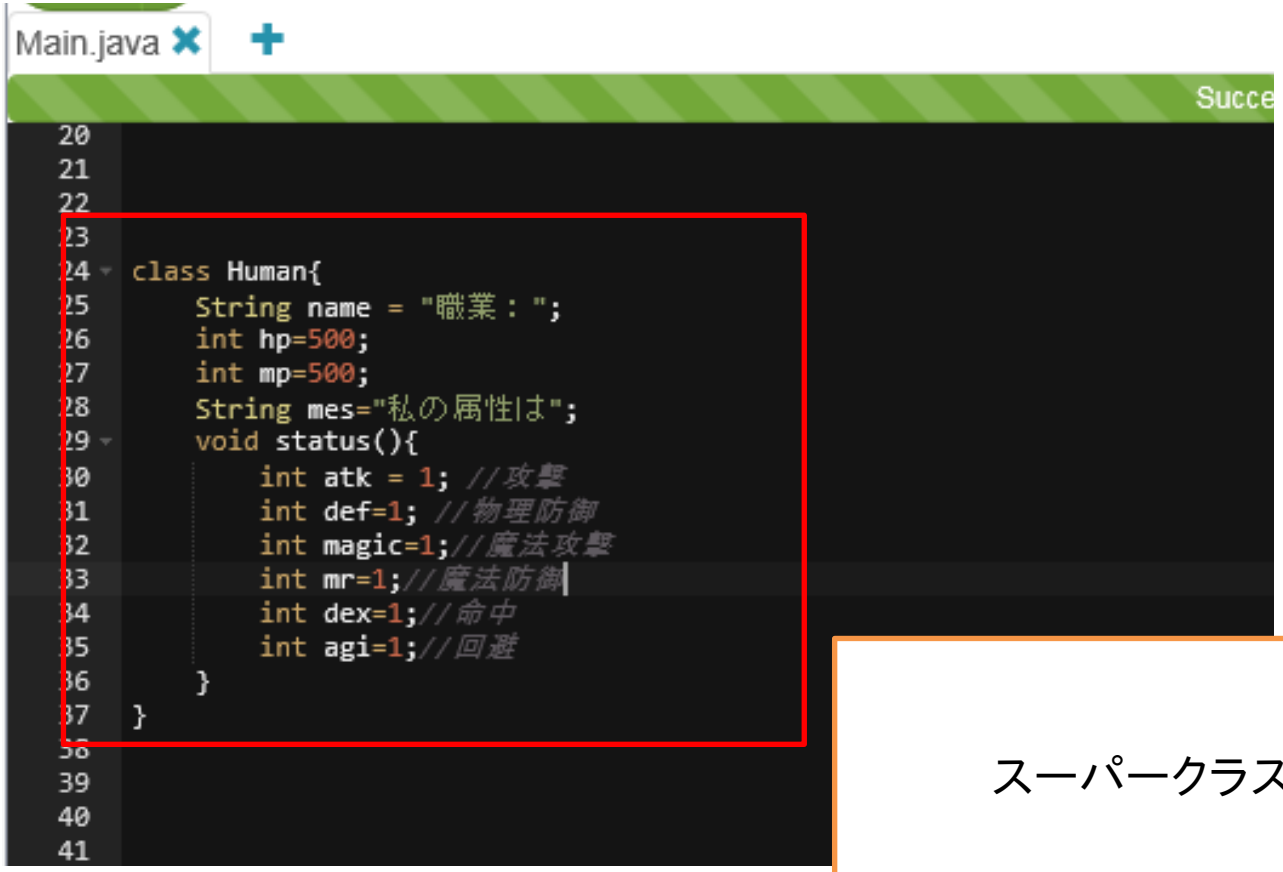
一番元の親クラスのコンストラクタを呼び出し、次に自クラスのコンストラクタを読んでいる

オーバーライドとsuperをメソッド呼び出し

- オーバーライド
  何個も同じようなメソッド書くのがめんどくさい
  →呼び出したら簡単！
  決まり・・・メソッド名、引数リストが同じである
  戻り値は同じもしくは戻り値の型がサブクラスである
  アクセス修飾子はスーパークラスと同じもsしくは
  それより公開範囲が広いこと

・super・・・自オブジェクトから見てスーパークラスのオブジェクトを表現する場合に使用。

super.メソッド名・・・サブクラスから明示的にメソッドを呼び出したい場合に使用。


Main.java
Succe
20
21
22
23
24 class Human{
25 String name = "職業：";
26 int hp=500;
27 int mp=500;
28 String mes="私の属性は";
29 void status(){
30 int atk = 1; //攻撃
31 int def=1; //物理防御
32 int magic=1;//魔法攻撃
33 int mr=1;//魔法防御
34 int dex=1;//命中
35 int agi=1;//回避
36 }
37 }
38
39
40
41

スーパークラス作成

Humanクラスを継承

```
43  class Magic extends Human{
44      String name = super.name+"魔法使い";
45      int hp=super.hp-400;
46      int mp=super.mp-100;
47      String mes= super.mes+"水属性です";
48
49      @Override
50      void status(){
51          int atk = 1; //攻撃
52          int def=1; //物理防御
53          int magic=5;//魔法攻撃
54          int mr=1;//魔法防御
55          int dex=6;//命中
56          int agi=4;//回避
57
58          System.out.println("攻撃力："+atk);
59          System.out.println("物理防御："+def);
60          System.out.println("魔法攻撃："+magic);
61          System.out.println("魔法防御："+mr);
62          System.out.println("命中："+dex);
63          System.out.println("回避："+agi);
64      }
```

Superを使用してHumanクラスの
hp,mpを使用
他も同じように使用

メソッドのオーバーライド

Main.java

Humanクラスを継承

```
68  //剣士
69  class Kenshi extends Human{
70      String name = super.name+"剣士";
71      int hp=super.hp;
72      int mp=super.mp-300;
73      String mes= super.mes+"無属性です";
74      @Override
75      void status(){
76          int atk = 45; //攻撃
77          int def=30; //物理防御
78          int magic=1;//魔法攻撃
79          int mr=3;//魔法防御
80          int dex=16;//命中
81          int agi=4;//回避
82          System.out.println("攻撃力："+atk);
83          System.out.println("物理防御："+def);
84          System.out.println("魔法攻撃："+magic);
85          System.out.println("魔法防御："+mr);
86          System.out.println("命中："+dex);
87          System.out.println("回避："+agi);
88      }
```

Superを使用してHumanクラスの
hp,mpを使用
他も同じように使用

メソッドのオーバーライド

Main.java

Success

```java
class Main {
    public static void main(String[] args)  {
        Magic A = new Magic();
        System.out.println(A.name);
        System.out.println("HP:"+String.valueOf(A.hp));
        System.out.println("MP:"+String.valueOf(A.mp));
        A.status();
        System.out.println(A.mes);
        System.out.println("☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆");
         Kenshi B = new Kenshi();
        System.out.println(B.name);
        System.out.println("HP:"+String.valueOf(B.hp));
        System.out.println("MP:"+String.valueOf(B.mp));
        B.status();
        System.out.println(B.mes);
    }
}
```

各インスタンスを生成し、実行する

それぞれステータスが変わって出力される