新的一年,新的开始.

## 抽象类

- 抽象类和抽象方法用abstract来修饰
- 如果一个类包含抽象方法,那么该类必须是抽象类.
- 抽象类不能被实例化,不能被new.
- 构造方法,用static修饰的方法不能声明为抽象方法
- 抽象类中抽象方法只是声明,不能给出方法的具体实现过程

## Test.java

```
abstract class A {
   A() {
      System.out.println("adstract created");
   }
   abstract void example();//如果一个类包含抽象方法,那么该类必须是抽象类,也就是A必须是抽象类
   void method() {//成员方法,成员变量,构造方法的访问方式都是和普通类一样
      System.out.println("method created");
   }
}
class B extends A {
   void example() {//抽象类子类必须给出抽象中抽象方法的具体实现,除非该类也是抽象类

      System.out.println("adstract method overload of B");
   }
}
class C extends A {
   void example() {
      System.out.println("adstract method overload of C");
   }
}

public class Test {
   public static void main(String []args) {
      //A error = new A(); //抽象类不能被实例化,不能被NEW
      System.out.println("-----B-----");
      A test = new B();
      test.example();
      test.method();
      System.out.println("-----C-----");
      A test2 = new C();
      test2.example();
      test.method();
   }
}
```

## 运行结果

```
jzzh@jizizihe:~/workspace/java/abstract$ java Test
-----B-----
adstract created
adstract method overload of B
method created
-----C-----
adstract created
adstract method overload of C
method created
```

## 总结

抽象类除了不能实例化对象外,其他功能依然存在.成员变量,成员方法和构造方法的访问方式和普通类一样.