### Java第一阶段—DAY20-JAVA案例

1. 通过反射的方式调用User类的有参构造器来创建实例。

class User{  
  
 private String name;  
  
 private int age ;  
  
 *// 构造方法是私有的* private User(String name , int age ){  
 this.name = name ;  
 this.age = age ;  
 }  
  
 @Override  
 public String toString() {  
 return "User{" +  
 "name='" + name + '\'' +  
 ", age=" + age +  
 '}';  
 }  
}  
  
public class Demo02Reflect {  
  
 public static void main(String[] args) throws Exception {  
*// 获取对应的Class对象* Class clazz = User.class ;  
*// 获取想要的构造器,这里的参数列表的类型要跟定义的构造方法一致* Constructor c = clazz.getDeclaredConstructor(String.class, int.class);  
*// 由于构造方式是私有的，需要设置权限* c.setAccessible(true);  
*// 通过对应的构造器创建对应的对象，这里的参数列表的值要跟构造方法的类型一致* User user = (User) c.newInstance("blb", 18);  
 System.out.println(user);  
 }  
}

1. 通过反射遍历Student类的所有的字段，并给私有的属性name赋值。

class Student{  
  
 private String name;  
  
 private int age ;  
  
 @Override  
 public String toString() {  
 return "User{" +  
 "name='" + name + '\'' +  
 ", age=" + age +  
 '}';  
 }  
}  
  
public class Demo03Reflect {  
  
 public static void main(String[] args) throws Exception {  
*// 获取对应的Class对象* Class clazz = Student.class ;  
*// 获取对象* Student s = (Student) clazz.newInstance();  
*// 获取当前类定义的所有的属性* Field[] fields = clazz.getDeclaredFields();  
*// 遍历所有的属性* for (Field f: fields) {  
*// 打印字段名称* System.out.println(f.getName());  
 }  
*// 获取属性名为name的属性* Field nameField = clazz.getDeclaredField("name");  
*// 私有的属性需要打开访问权限* nameField.setAccessible(true);  
*// 给对象s的name属性赋值为blb* nameField.set(s,"blb");  
  
 System.out.println(s);  
  
 }  
}

1. 通过反射Student类，调用其所有的方法，包括私有方法。

class Student{  
  
 public void study(){  
 System.out.println("好好学习，天天向上。");  
 }  
  
 private int exam(String name){  
 System.out.println(name+"同学，认真考试");  
 int score = 90 ;  
 return score ;  
 }  
  
}  
  
public class Demo04Reflect {  
  
 public static void main(String[] args) throws Exception {  
*// 获取对应的Class对象* Class clazz = Student.class ;  
*// 获取对象* Student s = (Student) clazz.newInstance();  
*// 获取当前类定义的所有的属性* Method[] methods = clazz.getDeclaredMethods();  
  
*// 遍历所有的方法* for (Method m: methods) {  
*// 打印方法名称* System.out.println(m.getName());  
 }  
  
*// 获取方法名为study的方法* Method study = clazz.getDeclaredMethod("study");  
*// 通过invoke方法调用这个s对象的study方法* study.invoke(s);  
  
*// 获取方法名为exam的方法,参数类型要一致* Method exam = clazz.getDeclaredMethod("exam",String.class);  
*// 私有方法，必须要设置访问权限* exam.setAccessible(true);  
*// 通过invoke方法调用这个s对象的exam方法，接收方法调用的结果* Integer result = (Integer) exam.invoke(s,"blb");  
 System.out.println("考试分数为： "+result);  
  
 }  
}