* 调用指定的构造器

@Test  
public void test3() throws Exception {  
  
 String className = "com.sglm.test.day01.Person";  
 Class clazz = Class.forName(className);  
  
 Constructor cons = clazz.getConstructor(String.class, int.class);  
 cons.setAccessible(true);  
 Person wcy = (Person) cons.newInstance("wcy", 24);  
 System.***out***.println(wcy);  
}

* **调用指定方法**

通过反射，调用类中的方法，通过Method类完成。步骤：

* 通过Class类的**getMethod(String name,Class…parameterTypes)方法取得一个Method对象**，并**设置此方法操作时所需要的参数类型**。
* 之后使用Object invoke(Object obj,Object[]args)进行调用，并向方法中传递要设置的obj对象的参数信息。



@Test  
public void test2() throws Exception {  
  
 Class clazz = Person.class;  
  
 //getMethod(String methodName,Class ... params)  
 //获取运行时类中声明为public的指定的方法。  
 Method m1 = clazz.getMethod("show");  
 Person p = (Person) clazz.newInstance();  
 //调用指定的方法，invoke(Object obj,Object ... obj)  
 Object returnVal = m1.invoke(p);  
 System.***out***.println(returnVal);  
  
 Method m2 = clazz.getMethod("toString");  
 Object returnVal1 = m2.invoke(p);  
 System.***out***.println(returnVal1);  
}

* **调用指定的属性**

在反射机制中，可以直接通过Field类操作类中的属性，通过Field类提供的set()和get()方法就可以完成设置和取得属性内容的操作。

* **public Field getField(String name)**

获取运行时类中声明为**public**的指定属性名为name的属性。

* **public Field getDeclaredField(String name)**

获取运行时类中指定的名为name的属性。

* 在Field中：
* public Object get(Object obj)取得指定对象obj上此Field的属性内容
* public void set(Object obj,Object value)设置指定对象obj上此Field的属性内容
* 注：**在类中属性都设置为private的前提下，在使用set()和get()方法时，首先要使用Field类中的setAccessible(true)方法将需要操作的属性设置为可以被外部访问**。
* public void setAccessible(true)

访问私有属性时，让这个属性可见。

public void test3() throws Exception {  
  
 Class clazz = Person.class;  
  
 //1、获取指定的属性  
 Field name = clazz.getField("name");  
 //2、创建运行时类的对象  
 Person p = (Person) clazz.newInstance();  
 System.***out***.println(p);  
 //3、将运行时类的指定的属性赋值  
 name.set(p,"wcy");  
 System.***out***.println(p);  
  
 System.***out***.println("================");  
  
 //age的权限为private  
 Field age = clazz.getDeclaredField("age");  
 age.setAccessible(true);  
 age.set(p,24);  
 System.***out***.println(p);  
}