## 反射常用的api（reflect）

Clsaa类，像是所有类的一面镜子，封装了java类的信息

Filed类，封装了类的属性

Method类，封装了类的方法

Construct类，封装了类的构造方法

Array类，对应一个数组类。操作所有数组对象

## Class类对象的获取方法

每个Java类都有一个对应的Class类对象。当java类被加载后，就会创建这个Java类对应的Class类的对象，这个创建是自动创建的，不是我们手动创建的。

自动创建之后，怎么获得Java类对应的Class类的对象？

方式1：通过Java类的对象，获得已有的Java类对应的Class类对应的对象

Object.getClass();

方式2：通过Java类的class属性（推荐使用）无需调用方法，高效。可以直接检查类是否存在，那就检查对应的Class对象是否存在。安全。 Student.class

方式3：通过Class类的静态方法forName(“类的全限定名”)

Class.forName(“com.zhang.Student”);

这三种方法获得的同一个Class类的对象是同一个对象。所以一个Java类值对应一个Class对象

## Construct类

获取：通过Class类对象

Construct getConstruct(Class[] params) 获得对应的构造方法，参数为构造方法的参数，对应的Class对象。通过 String.class获得

Construct[] getConstructs()获得所有的构造方法

## 获得Java类的具体对象

通过Construct类获得Java类的一个具体对象

Construct c = studentClass.getConsturct(String.class, int.class);

Student s = c.newInstance(“zhangkuan”, “18”); 本质上是通过Student类的构造方法

通过Class类的newInstance() 方法。本质上使用Java类的默认构造方法。

studentClass.newInstance() 使用的是默认构造方法（要存在，否则回报错）

## Field类，访问类的属性

获得，通过Class对象获得，获得一个Field类的对象

Field getField(“属性的名称”) 只能获得公有属性

Field getDeclaredField() 获得各种访问级别的属性

只获得属性，并不意味着能够访问

我们可以修改访问权限（FieldName.setAccessible(true)）

设置类中对象的属性：FieldName.set(对象名称，值)

FieldName.get(对象名称) 获得对象名称：本质是通过get方法获取

## Method类，访问类中的方法

Method getMethod(“方法的名称”, 方法的参数类型.class )

Method.invoke(具体的对象，参数值)

## Array类，访问所有数组对象

Array.set(具体数组名称，索引，值)；

Array.get(具体数组名称，索引，值)；