Reflection

认识 java.lang.Class

> Every type is either a reference or a primitive. Classes, enums, and arrays (which all inherit from java.lang.Object) as well as interfaces are all reference types. Examples of reference types include java.lang.String, all of the wrapper classes for primitive types such as java.lang.Double, the interface java.io.Serializable, and the enum javax.swing.SortOrder. There is a fixed set of primitive types: boolean, byte, short, int, long, char, float, and double.

> For every type of object, the Java virtual machine instantiates an immutable instance of java.lang.Class which provides methods to examine the runtime properties of the object including its members and type information. Class also provides the ability to create new classes and objects. Most importantly, it is the entry point for all of the Reflection APIs.

- 每一个类型不是引用就是基本类型
- 每种类型在JVM中都有一个java.lang.Class的实例
- 检查对象运行时的属性：成员和类型
- 具有创建类和对象的能力
- 更重要地，它是Reflection APIs的入口

什么是Reflection：从Class取得类信息的方式

```
package cn.edu.zut.cs;
public class Student {
    private int id;
    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
}
```

使用Class获取类信息

```
import java.lang.reflect.*;
public class ClassInfo {
    public static void main(String... args) {
        Class clz = Student.class;
        clz.getName();
        clz.isInterface();
        clz.getSuperclass().getName();
        ……
    }
}
```

使用Class.forName()

```
public class ClassforName {
    public static void main(String... args) {
        Class clz = Class.forName("cn.edu.zut.cs.Student");
        clz.getName();
        ……
    }
}
```

从Class建立对象

```
public class ClassnewInstance {
    public static void main(String... args) {
        Class clz = Class.forName("cn.edu.zut.cs.Student");
        Object obj = clz.newInstance();
        Method setter = clz.getMethod("setId", Integer.class);
        setter.invoke(obj,12);
        Method getter = clz.getMethod("getId");
        getter.invoke(obj);
    }
}
```

对比熟悉的用法

```
public class Test {
    public static void main(String... args) {
        Student stu = new Student();
        stu.setId(12);
        stu.getId();
    }
}
```