* Java Reflection
* Reflection（反射）是被视为**动态语言**的关键，反射机制允许程序在执行期借助于Reflection API取得任何类的内部信息，并能直接操作任意对象的内部属性及方法。
* Java反射机制提供的功能
* 在运行时判断任意一个对象所属的类；
* 在运行时构造任意一个类的对象；
* 在运行时判断任意一个类所具有的成员变量和方法；
* 在运行时调用任意一个对象的成员变量和方法；
* 生成动态代理。
* 什么是反射？
* **反射就是把Java类中的各种成分映射成一个个的Java对象**。
* 例如：一个类有：成员变量、方法、构造方法、包等等信息，利用反射技术可以对一个类进行解刨，把各个组成部分映射成一个个对象。
* 反射的原理在于class对象。类的加载过程（Class对象的由来是将class文件读入内存，并为之创建一个Class对象）如下图：



* Java反射机制研究及应用
* 反射相关的主要API：
* java.lang.Class：代表一个类。
* java.lang.reflect.Method：代表类的方法。
* java.lang.reflect.Field：代表类的成员变量。
* java.lang.reflect.Constructor：代表类的构造方法。

/\*\*  
 \* 有了反射，可以通过反射创建一个类的对象，并调用其中的结构。  
 \*/  
@Test  
public void test2() throws Exception {  
 Class<Person> clazz = Person.class;  
  
 //1、创建clazz对应的运行时类Person类的对象。  
 Person p = clazz.newInstance();  
 System.***out***.println(p);  
  
 //2、通过反射调用运行时类的指定的属性。  
 Field f1 = clazz.getField("name");  
 f1.set(p, "吴晨阳");  
 System.***out***.println(p);  
  
 Field f2 = clazz.getDeclaredField("age");  
 f2.setAccessible(true);  
 f2.set(p, 20);  
 System.***out***.println(p);  
  
 //3、通过反射调用运行时类的指定方法。  
 Method m1 = clazz.getMethod("show");  
 m1.invoke(p);  
  
 Method m2 = clazz.getMethod("display", String.class);  
 m2.invoke(p, "China");  
}