参考文档：<http://www.cnblogs.com/XHJT/p/3915221.html>

**一.反射之实例化Class类的5种方式：**

 java的数据类型可以分为两类，即**引用类型和原始类型**（即基本数据类型，c#中是值类型）。   
对于每种类型的对象，java虚拟机会实例化不可变的java.lang.Class对象。   
它提供了在运行时检查对象属性的方法，这些属性包括它的成员和类型信息。

**更重要的是Class对象是所有反射API的入口。**

 Class类是泛型类，可以使用**@SuppressWarnings("unchecked")**忽略泛型或者使用**Class<V>**类型。

**获得Class对象的5种方式：**

**1."Object.getClass()"   :**如果一个类的对象可用，则最简单的获得Class的方法是使用Object.getClass()。当然，此方式只对引用类型有效。

System.*out*.println(class01.getClass().toString());

**2. ".class"------------- :**如果类型可用但没有对象，则可以在类型后加上".class"来获得Class对象。这也是使原始类型，获得Class对象最简单的方式：

System.*out*.println(TestClass01.class.toString());

**3."Class.forName()"---:**如果知道类的全名，则可以使用静态方法Class.forName()来获得Class对象，它不能用在原始类型上，但是可以用在原始类型数组上。(注：此方式会抛出ClassNotFoundException异常)

try{  
 System.*out*.println(Class.*forName*("com.test.fanshe.TestClass01").toString());  
}catch(ClassNotFoundException e){  
 e.printStackTrace();  
}

**4. "包装类的TYPE域"-----:**每一个原始类型和void都有包装类。利用其TYPE域就可以获得Class对象。

System.*out*.println(Integer.*TYPE*); //输出 int

**5."以Class为返回值的方法":** 如获取内部类的getDeclaredClasses()方法，此时返回一个对象数组

 --支持知识共享，转载请标注地址"<http://www.cnblogs.com/XHJT/p/3915221.html>  "——和佑博客园，谢谢~~--

注：除了java.lang.reflect.ReflectPermission和java.lang.reflect包外，所有类都没有公共的构造方法。**为了获得这些类的对象，必须使用Class类中适当的方法**，对于不同的数据类型，Class 对象的获得方式是不同。

**二.反射之查看类的定义：**

**通常类的声明包括常见修饰符(public,protected,private,abstract,static,final,strictfp等)、   
    类的名称、类的泛型参数、类的继承类(实现的接口)、类的注解等信息。** 

**Class类的实例表示正在运行的Java应用程序中的类和接口：**   
    **枚举是一种类（因此可以继承接口和类），注释是一种接口（因此可以实现接口）**。   
    每一个数组属于被映射为Class对象的一个类，所有具有相同元素类型和维数的数组都共享该Class对象；   
    基本数据类型和关键字void也表示为Class对象。   
    Class没有公共的构造方法。   
    Class对象是在加载类时由Java虚拟机以及通过调用类加载器中的defineClass方法自动构造的。   
      
 **Class类常用方法：**    **forName(String className)-:** 根据给定的名称获得Class对象   
   **getAnnotations()------------:** 返回此Class对象上存在的注释   
   **getCanonicalName() --------:** 返回Java Language Specification 中所定义的底层类的规范化名称   
    **getGenericInterfaces()------:** 返回泛型形式的对象类所实现的接口   
    **getGenericSuperclass() -----:** 返回泛型形式的对象类所直接继承的超类   
    **getModifuers() --------------:** 返回此类或接口以整数编码的Java语言修饰符   
    **getTypeParameters() -------:** 按声明顺序返回TypeVariable对象的一个数组

   注：Java语言预定义的注解只有**@Deprecated**可以在运行时获得。