# 最常用读取properties文件的方法

1. **InputStream in = getClass().getResourceAsStream("资源Name");这种方式要求properties文件和当前类在同一文件夹下面。如果在不同的包中，必须使用：**
2. **InputStream ins = this.getClass().getResourceAsStream("/cn/zhao/properties/testPropertiesPath2.properties");**
3. Java中获取路径方法
4. 获取路径的一个简单实现
5. 反射方式获取properties文件的三种方式

**1 反射方式获取properties文件最常用方法以及思考：**

|  |
| --- |
| Java读取properties文件的方法比较多，网上最多的文章是"Java读取properties文件的六种方法"，但在Java应用中，最常用还是通过java.lang.Class类的getResourceAsStream(String name) 方法来实现，但我见到众多读取properties文件的代码中，都会这么干：    **InputStream in = getClass().getResourceAsStream("资源Name");**    这里面有个问题，就是getClass()调用的时候默认省略了this！我们都知道，this是不能在static（静态）方法或者static块中使用的，原因是static类型的方法或者代码块是属于类本身的，不属于某个对象，而this本身就代表当前对象，而静态方法或者块调用的时候是不用初始化对象的。    问题是：假如我不想让某个类有对象，那么我会将此类的默认构造方法设为私有，当然也不会写别的共有的构造方法。并且我这个类是工具类，都是静态的方法和变量，我要在静态块或者静态方法中获取properties文件，这个方法就行不通了。    那怎么办呢？其实这个类就不是这么用的，他仅仅是需要获取一个Class对象就可以了，那还不容易啊－－**取所有类的父类Object，用Object.class**难道不比你的用你正在写类自身方便安全吗 ？呵呵，下面给出一个例子，以方便交流。    import java.util.Properties;  import java.io.InputStream;  import java.io.IOException;   /\*\*  \* 读取Properties文件的例子  \* File: TestProperties.java  \* User: leizhimin  \* Date: 2008-2-15 18:38:40  \*/  public final class TestProperties {      private static String param1;      private static String param2;       static {          Properties prop = new Properties();          InputStream in = Object.class.getResourceAsStream("/test.properties");          try {              prop.load(in);              param1 = prop.getProperty("initYears1").trim();              param2 = prop.getProperty("initYears2").trim();          } catch (IOException e) {              e.printStackTrace();          }      }       /\*\*       \* 私有构造方法，不需要创建对象       \*/      private TestProperties() {      }       public static String getParam1() {          return param1;      }       public static String getParam2() {          return param2;      }       public static void main(String args[]){          System.out.println(getParam1());          System.out.println(getParam2());      }  }    运行结果：  151  152   Process finished with exit code 0      当然，把Object.class换成int.class照样行，呵呵，大家可以试试。    另外，如果是static方法或块中读取Properties文件，还有一种最保险的方法，就是这个类的本身名字来直接获取Class对象，比如本例中可写成TestProperties.class，这样做是最保险的方法 |

**2 获取路径的方式:**

|  |
| --- |
| **2.1 获取当前类所在的包：**  File fileB = **new** File(**this**.getClass().getResource("").getPath());  System.*out*.println("fileB path: " + fileB);    **2.2 获取当前类所在的工程名：**  System.*out*.println("user.dir path: " + **System.*getProperty*("user.dir"));** |

**3 获取路径的一个简单的Java实现：**

|  |
| --- |
| /\*\*       \* 获取项目的相对路径下文件的绝对路径       \*       \* @param parentDir       \* 目标文件的父目录,例如说,工程的目录下,有lib与bin和conf目录,那么程序运行于lib or       \* bin,那么需要的配置文件却是conf里面,则需要找到该配置文件的绝对路径       \* @param fileName       \* 文件名       \* @return 一个绝对路径       \*/      public static String getPath(String parentDir, String fileName) {          String path = null;          String userdir = System.getProperty("user.dir");          String userdirName = new File(userdir).getName();          if (userdirName.equalsIgnoreCase("lib")                  || userdirName.equalsIgnoreCase("bin")) {              File newf = new File(userdir);              File newp = new File(newf.getParent());              if (fileName.trim().equals("")) {                  path = newp.getPath() + File.separator + parentDir;              } else {                  path = newp.getPath() + File.separator + parentDir                          + File.separator + fileName;              }          } else {              if (fileName.trim().equals("")) {                  path = userdir + File.separator + parentDir;              } else {                  path = userdir + File.separator + parentDir + File.separator                          + fileName;              }          }            return path;      } |

**4 利用反射的方式获取路径：**

|  |
| --- |
| InputStream ips1 = Enumeration.**class**.getClassLoader() .getResourceAsStream("cn/zhao/enumStudy/testPropertiesPath1.properties");    InputStream ips2 =Enumeration.**class**.getResourceAsStream("testPropertiesPath1.properties");    InputStream ips3 =Enumeration.**class**.getResourceAsStream("properties/testPropertiesPath2.properties"); |