一、Java Properties类

Java中有个比较重要的类Properties（Java.util.Properties），主要用于读取Java的配置文件，各种语言都有自己所支持的配置文件，配置文件中很多变量是经常改变的，这样做也是为了方便用户，让用户能够脱离程序本身去修改相关的变量设置。

像Python支持的配置文件是.ini文件，同样，它也有自己读取配置文件的类ConfigParse，方便程序员或用户通过该类的方法来修改.ini配置文件。

在Java中，其配置文件常为.properties文件，格式为文本文件，文件的内容的格式是“键=值”的格式，文本注释信息可以用"#"来注释。

Properties类继承自Hashtable，如下：

它提供了几个主要的方法：

1. getProperty (String key)，用指定的键在此属性列表中搜索属性。也就是通过参数 key ，得到 key 所对应的 value。
2. load (InputStream inStream)，从输入流中读取属性列表（键和元素对）。通过对指定的文件（比如说上面的 test.properties 文件）进行装载来获取该文件中的所有键 - 值对。以供 getProperty (String key) 来搜索。
3. setProperty (String key, String value) ，调用 Hashtable 的方法 put 。他通过调用基类的put方法来设置 键 - 值对。
4. store (OutputStream out, String comments)，以适合使用 load 方法加载到 Properties 表中的格式，将此 Properties 表中的属性列表（键和元素对）写入输出流。与 load 方法相反，该方法将键 - 值对写入到指定的文件中去。
5. clear ()，清除所有装载的 键 - 值对。该方法在基类中提供。

二、Java读取Properties文件

Java读取Properties文件的方法有很多，最常用的还是通过java.lang.Class类的getResourceAsStream(String name)方法来实现，如下可以这样调用：

InputStream in = getClass().getResourceAsStream("资源Name");作为我们写程序的，用此一种足够。

或者下面这种也常用：

InputStream in = new BufferedInputStream(new FileInputStream(filepath));

三、相关实例

下面列举几个实例，加深对Properties类的理解和记忆。

我们知道，Java虚拟机（JVM）有自己的系统配置文件（system.properties），我们可以通过下面的方式来获取。

1、获取JVM的系统属性

复制代码

```
1 import java.util.Properties;
2
3 public class ReadJVM {
4     public static void main(String[] args) {
5         Properties pps = System.getProperties();
6         pps.list(System.out);
7     }
8 }
```

复制代码

结果：

```
-- listing properties --
java.runtime.name=Java(TM) SE Runtime Environment
sun.boot.library.path=D:\Program Files\Java\jre7\bin
java.vm.version=21.1-b02
user.country.format=CN
java.vm.vendor=Oracle Corporation
java.vendor.url=http://java.oracle.com/
path.separator=;
java.vm.name=Java HotSpot(TM) Client VM
file.encoding.pkg=sun.io
user.script=
user.country=US
sun.java.launcher=SUN_STANDARD
sun.os.patch.level=Service Pack 1
java.vm.specification.name=Java Virtual Machine Specification
user.dir=D:\eclipse workplace\Test
java.runtime.version=1.7.0_01-b08
java.awt.graphicsenv=sun.awt.Win32GraphicsEnvironment
java.endorsed.dirs=D:\Program Files\Java\jre7\lib\endorsed
os.arch=x86
java.io.tmpdir=C:\Users\bycer\AppData\Local\Temp\
line.separator=
```

2、随便新建一个配置文件（Test.properties）

```
name=JJ
Weight=4444
Height=3333
```

复制代码

```
public class getProperties {
    public static void main(String[] args) throws FileNotFoundException,
IOException {
        Properties pps = new Properties();
        pps.load(new FileInputStream("Test.properties"));
        Enumeration enum1 = pps.propertyNames();//得到配置文件的名字
        while(enum1.hasMoreElements()) {
            String strKey = (String) enum1.nextElement();
            String strValue = pps.getProperty(strKey);
            System.out.println(strKey + "=" + strValue);
        }
    }
}
```

复制代码

3、一个比较综合的实例

根据key读取value

读取properties的全部信息

写入新的properties信息

复制代码

```
//关于Properties类常用的操作
public class TestProperties {
    //根据Key读取Value
    public static String GetValueByKey(String filePath, String key) {
        Properties pps = new Properties();
        try {
            InputStream in = new BufferedInputStream (new
FileInputStream(filePath));
            pps.load(in);
            String value = pps.getProperty(key);
            System.out.println(key + " = " + value);
            return value;
```

```
12
13          }catch (IOException e) {
14              e.printStackTrace();
15              return null;
16          }
17      }
18
19      //读取Properties的全部信息
20      public static void GetAllProperties(String filePath) throws IOException {
21          Properties pps = new Properties();
22          InputStream in = new BufferedInputStream(new
FileInputStream(filePath));
23          pps.load(in);
24          Enumeration en = pps.propertyNames(); //得到配置文件的名字
25
26          while(en.hasMoreElements()) {
27              String strKey = (String) en.nextElement();
28              String strValue = pps.getProperty(strKey);
29              System.out.println(strKey + "=" + strValue);
30          }
31
32      }
33
34      //写入Properties信息
35      public static void WriteProperties (String filePath, String pKey, String
pValue) throws IOException {
36          Properties pps = new Properties();
37
38          InputStream in = new FileInputStream(filePath);
39          //从输入流中读取属性列表（键和元素对）
40          pps.load(in);
41          //调用 Hashtable 的方法 put。使用 getProperty 方法提供并行性。
42          //强制要求为属性的键和值使用字符串。返回值是 Hashtable 调用 put 的结果。
43          OutputStream out = new FileOutputStream(filePath);
44          pps.setProperty(pKey, pValue);
45          //以适合使用 load 方法加载到 Properties 表中的格式，
46          //将此 Properties 表中的属性列表（键和元素对）写入输出流
47          pps.store(out, "Update " + pKey + " name");
48      }
49
50      public static void main(String [] args) throws IOException{
```

```
51          //String value = GetValueByKey("Test.properties", "name");
52          //System.out.println(value);
53          //GetAllProperties("Test.properties");
54          WriteProperties("Test.properties","long", "212");
55      }
56 }
```

复制代码

结果：

Test.properties中文件的数据为：

```
#Update long name
#Sun Feb 23 18:17:16 CST 2014
name=JJ
Weight=4444
long=212
Height=3333
```

```
package com.javademo;

import java.io.BufferedInputStream;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.util.Enumeration;
import java.util.Properties;

import org.junit.Test;

public class PropertiesTest {
```

```
/**
 * @param 获取JVM的系统属性
 */
@Test
public void getParam() {
        Properties properties = System.getProperties();
        properties.list(System.out);
}

@Test
public void getTestProperties() {
        Properties properties = new Properties();
        try {
                properties.load(new FileInputStream("Test.properties"));
                properties.propertyNames();
                Enumeration enumeration = properties.propertyNames();
                while (enumeration.hasMoreElements()) {
                        String key = (String) enumeration.nextElement();
                        String value = properties.getProperty(key);
                        System.out.println(key + " : " + value);
                }

        } catch (FileNotFoundException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
        } catch (IOException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
        }

}

// 根据Key读取Value
public static String GetValueByKey(String filePath, String key) {
        Properties pps = new Properties();
```

```
                try {
                        InputStream in = new BufferedInputStream(new
FileInputStream(
                                        filePath));
                        pps.load(in);
                        String value = pps.getProperty(key);
                        System.out.println(key + " = " + value);
                        return value;

                } catch (IOException e) {
                        e.printStackTrace();
                        return null;
                }
        }

        // 读取Properties的全部信息
        public static void GetAllProperties(String filePath) throws IOException
{
                Properties pps = new Properties();
                InputStream in = new BufferedInputStream(new
FileInputStream(filePath));
                pps.load(in);
                Enumeration en = pps.propertyNames(); // 得到配置文件的名字

                while (en.hasMoreElements()) {
                        String strKey = (String) en.nextElement();
                        String strValue = pps.getProperty(strKey);
                        System.out.println(strKey + "=" + strValue);
                }

        }

        // 写入Properties信息
        public static void WriteProperties(String filePath, String pKey,
                        String pValue) throws IOException {
```

```
            Properties pps = new Properties();

            InputStream in = new FileInputStream(filePath);
            // 从输入流中读取属性列表（键和元素对）
            pps.load(in);
            // 调用 Hashtable 的方法 put。使用 getProperty 方法提供并行性。
            // 强制要求为属性的键和值使用字符串。返回值是 Hashtable 调用 put
的结果。
            OutputStream out = new FileOutputStream(filePath);
            pps.setProperty(pKey, pValue);
            // 以适合使用 load 方法加载到 Properties 表中的格式，
            // 将此 Properties 表中的属性列表（键和元素对）写入输出流
            pps.store(out, "Update " + pKey + " name");
        }

        @Test
        public void getPro() throws Exception {
           // String value = GetValueByKey("Test.properties", "age");
            //System.out.println(value);
            GetAllProperties("Test.properties");
            //WriteProperties("Test.properties", "address", "hangzhou");
        }

}
```