学生应知信息知识

# Java知识手册（七）

狄登峰 主编

# 目　　录

# Java 编程获取硬盘空间

因为论坛有人问到这个问题,所以就写了这篇文章。希望对大家有所帮助。

一般来讲,要用 java 得到硬盘空间,有 3 种方法:

1. 调用 system 的 command,然后分析得到的结果,这种方法有很强的系统依赖性,linux 下和 win 下要分别写程序。

下面是一个 win 下的例子,编译成功之后,运行 java Diskspace yourdir(比如 c:\)

```
import java.io.BufferedReader;

import java.io.InputStreamReader;

/**

* Determine free disk space for a given directory by

* parsing the output of the dir command.

* This class is inspired by the code at

* Works only under Windows under certain circumstances.
```

```
* Yes, it’s that shaky.

* Requires Java 1.4 or higher.

* @[EMAIL PROTECTED]

*Marco Schmidt

*/

public class Diskspace

{

   private Diskspace()

   {

// prevent instantiation of this class

   }

/**

* Return available free disk space for a directory.

* @[EMAIL PROTECTED]

dirName name of the directory

* @[EMAIL PROTECTED]
```

```
free disk space in bytes or -1 if unknown
*/
    public static long getFreeDiskSpace(String dirName)
    {
try
{
    // guess correct 'dir' command by looking at the
    // operating system name
    String os = System.getProperty("os.name");
    String command;
    if (os.equals("Windows NT") ||os.equals("Windows 2000"))
    {
command = "cmd.exe /c dir " + dirName;
    }
    else
    {
```

```
command = "command.com /c dir " + dirName;
    }
    // run the dir command on the argument directory name
    Runtime runtime = Runtime.getRuntime();
    Process process = null;
    process = runtime.exec(command);
    if (process == null)
    {
return -1;
    }
    // read the output of the dir command
    // only the last line is of interest
     BufferedReader in = new BufferedReader(new InputStreamReader(process.getInputStream()));
    String line;
    String freeSpace = null;
```

```
        while ((line = in.readLine()) != null)
        {
    freeSpace = line;
        }
        if (freeSpace == null)
        {
    return -1;
        }
        process.destroy();
        // remove dots & commas & leading and trailing whitespace
        freeSpace = freeSpace.trim();
        freeSpace = freeSpace.replaceAll("\\.", "");
        freeSpace = freeSpace.replaceAll(",", "");
        String[] items = freeSpace.split(" ");
        // the first valid numeric value in items after(!) index 0
```

```
    // is probably the free disk space
    int index = 1;
    while (index < items.length)
    {
try
{
    long bytes = Long.parseLong(items[index++]);
    return bytes;
}
catch (NumberFormatException nfe)
{
}
    }
    return -1;
}
catch (Exception exception)
```

```
{
return -1;
}
   }
/**
* Command line program to print the free diskspace to stdout
* for all 26 potential root directories A:\ to Z:\
* (when no parameters are given to this program)
* or for those directories (drives) specified as parameters.
* @[EMAIL PROTECTED]
args program parameters
*/

   public static void main(String[] args)
   {
```

```
if (args.length == 0)
{
   for (char c = ’A’; c <= ’Z’; c++)
   {
String dirName = c + ":\\";
System.out.println(dirName + " " +
getFreeDiskSpace(dirName));
   }
}
else
{
   for (int i = 0; i < args.length; i++)
   {
System.out.println(args[i] + " " + getFreeDiskSpace(args[i]));
   }
}
```

```
    }
}
```

方法二:使用 Jconfig,可以跨平台

从 http://www.tolstoy.com/samizdat/jconfig.html 上下载 jconfig。

下载的包的 sample 里有很简单的例子,如果是要得到磁盘空间的话:

用 FileRegistry.getVolumes()得到 DiskVolume。

然后 call getFreeSpace()和 getMaxCapacity()。

就是这么简单。

方法三:jni

这个是解决所有和 os 相关的操作的万能利器了。

例子我也懒得写了。

写一个 dll 然后 call 之即可。

## Java 开发技巧：如何计算对数

毫无疑问，Java 可以计算对数，然而在 API 中却有惊人的误差。但是如果运用了以下的方法，用 Java 处理

数字所遇到的小麻烦就可以轻而易举的解决了。

Sun 的 J2SE 提供了一个单一的对数方法——double java.lang.Math.log(double)，这很容易使用。请看如下代码：

```
double x = Math.log(5);
```

等价于下面的数学方程：

x = ln 5

或

x = loge5

其中 e 是内皮尔数或自然数。

如果你想算底不同的对数又该如何做呢？很遗憾，我们还没有办法计算以 10 为底或以 2 为底的对数。但是它们却是在计算对数时用的最多的。要想解决这个问题，我们就要回想曾经在学校里学过的数学和对数方程：

logx(y) =loge(x) / loge(y)

这只需一段简单的 Java 程序来实现：

```
package com.generationjava.math;

public class Logarithm {
```

```
 static public double log(double value, double base) {
     return Math.log(value) / Math.log(base);
 }
}
```

**计算** 100 **的以** 10 **为底的对数就变为非常简单了：**

```
double log = Logarithm.log(100, 10); // log is 2.0
```

512 **的以** 2 **为底的对数是：**

```
double log = Logarithm.log(512, 2); // log is 9.0
```

**下面的两个简单的方法也都是很有用的：**

```
static public double log2(double value) {
 return log(value, 2.0);
}
static public double log10(double value) {
```

```
    return log(value, 10.0);
}
```

## Java 模式研究之 Flyweight 模式

Flyweight 定义:

避免大量拥有相同内容的小类的开销(如耗费内存),使大家共享一个类(元类).

为什么使用?

面向对象语言的原则就是一切都是对象,但是如果真正使用起来,有时对象数可能显得很庞大,比如,字处理软件,如果以每个文字都作为一个对象,几千个字,对象数就是几千,无疑耗费内存,那么我们还是要"求同存异",找出这些对象群的共同点,设计一个元类,封装可以被共享的类,另外,还有一些特性是取决于应用(context),是不可共享的,这也 Flyweight 中两个重要概念内部状态 intrinsic 和外部状态 extrinsic 之分.

说白点,就是先捏一个的原始模型,然后随着不同场合和环境,再产生各具特征的具体模型,很显然,在这里需要产生不同的新对象,所以 Flyweight 模式中常出现 Factory 模式.Flyweight 的内部状态是用来共享的,Flyweight factory 负责维护一个 Flyweight pool(模式池)来存放内部状态的对象.

Flyweight 模式是一个提高程序效率和性能的模式,会大大加快程序的运行速度.应用场合很多:比如你要从一个数据库中读取一系列字符串,这些字符串中有许多是重复的,那么我们可以将这些字符串储存在 Flyweight 池(pool)中.

如何使用?

我们先从 Flyweight 抽象接口开始:

程序代码:

```
public interface Flyweight
{
public void operation( ExtrinsicState state );
}
//用于本模式的抽象数据类型(自行设计)
public interface ExtrinsicState { }
```

下面是接口的具体实现(ConcreteFlyweight) ,并为内部状态增加内存空间, ConcreteFlyweight 必须是可共享的,它保存的任何状态都必须是内部(intrinsic),也就是说,ConcreteFlyweight 必须和它的应用环境场合无关.

**程序代码**:

public class ConcreteFlyweight implements Flyweight {

private IntrinsicState state;

public void operation( ExtrinsicState state )

{

//**具体操作**

}

} **当然,并不是所有的** Flyweight **具体实现子类都需要被共享的,所以还有另外一种不共享的** ConcreteFlyweight:

**程序代码**:

public class UnsharedConcreteFlyweight implements Flyweight {

public void operation( ExtrinsicState state ) { }

}

Flyweight factory负责维护一个Flyweight池(存放内部状态),当客户端请求一个共享 Flyweight 时,这个 factory 首先搜索池中是否已经有可适用的,如果有,factory 只是简单返回送出这个对象,否则,创建一个新的对象,加入到池中,再返回送出这个对象.池

**程序代码**:

```
public class FlyweightFactory {

//Flyweight pool

private Hashtable flyweights = new Hashtable();

public Flyweight getFlyweight( Object key ) {

Flyweight flyweight = (Flyweight) flyweights.get(key);

if( flyweight == null ) {

//产生新的 ConcreteFlyweight

flyweight = new ConcreteFlyweight();

flyweights.put( key, flyweight );

}

return flyweight;

}
```