jeelee

浏览: 207389 次

性别:

来自: 深圳

我现在离线

最近访客 更多访客>>

dyong525 glzgg

Java获取系统信息（cpu，内存，硬盘，进程等）的相关方法

博客分类： java

Java OS Linux C# C++

1.利用jdk自带的API获取信息：（只支持jdk1.60以上的版本啊）

```
import java.io.InputStreamReader;
import java.io.LineNumberReader;
import java.util.ArrayList;
import java.util.List;

import mytools.com.sun.management.OperatingSystemMXBean;
import mytools.java.io.File;
import mytools.java.lang.management.ManagementFactory;
/**
* 获取windows系统信息（CPU,内存,文件系统）
* @author libing
*
*/
```

最新评论

baoyunpeng1116： 很仔细

```
public class WindowsInfoUtil {
    private static final int CPUTIME = 500;
    private static final int PERCENT = 100;
    private static final int FAULTLENGTH = 10;

    public static void main(String[] args) {
    System.out.println(getCpuRatioForWindows());
    System.out.println(getMemery());
    System.out.println(getDisk());
}

//获取内存使用率
public static String getMemery(){
  OperatingSystemMXBean osmxb = (OperatingSystemMXBean)
ManagementFactory.getOperatingSystemMXBean();
  // 总的物理内存+虚拟内存
  long totalvirtualMemory = osmxb.getTotalSwapSpaceSize();
  // 剩余的物理内存
  long freePhysicalMemorySize = osmxb.getFreePhysicalMemorySize();
  Double compare=(Double)(1-freePhysicalMemorySize*1.0/totalvirtualMemory)*100;
  String str="内存已使用:"+compare.intValue()+"%";
  return str;
}

//获取文件系统使用率
```

java.math.BigDecimal类的用法

hiqc：不错！版主可以再小整理一下，期待更完美...
java.math.BigDecimal类的用法

lydia_fly：学习了！不错不错
java.math.BigDecimal类的用法

stuxnet：非常感谢楼主，太棒了，岂是一顶所能抒怀的。
Java获取系统信息（cpu，内存，硬盘，进程等）的相关方法

savasun：
java.math.BigDecimal ？
java.math.BigDecimal类的用法

```
public static List<String> getDisk() {
 // 操作系统
 List<String> list=new ArrayList<String>();
 for (char c = 'A'; c <= 'Z'; c++) {
  String dirName = c + ":/";
  File win = new File(dirName);
       if(win.exists()){
        long total=(long)win.getTotalSpace();
        long free=(long)win.getFreeSpace();
        Double compare=(Double)(1-free*1.0/total)*100;
        String str=c+":盘  已使用 "+compare.intValue()+"%";
        list.add(str);
       }
    }
      return list;
}

//获得cpu使用率
public static String getCpuRatioForWindows() {
       try {
          String procCmd = System.getenv("windir") + "\\system32\\wbem\\wmic.exe process get
Caption,CommandLine,KernelModeTime,ReadOperationCount,ThreadCount,UserModeTime,WriteOperationCoun
          // 取进程信息
          long[] c0 = readCpu(Runtime.getRuntime().exec(procCmd));
          Thread.sleep(CPUTIME);
          long[] c1 = readCpu(Runtime.getRuntime().exec(procCmd));
```

```
            if (c0 != null && c1 != null) {
                long idletime = c1[0] - c0[0];
                long busytime = c1[1] - c0[1];
                return "CPU使用率:"+Double.valueOf(PERCENT * (busytime)*1.0 / (busytime +
idletime)).intValue()+"%";
            } else {
                return "CPU使用率:"+0+"%";
            }
        } catch (Exception ex) {
            ex.printStackTrace();
            return "CPU使用率:"+0+"%";
        }
    }

//读取cpu相关信息
    private static long[] readCpu(final Process proc) {
        long[] retn = new long[2];
        try {
            proc.getOutputStream().close();
            InputStreamReader ir = new InputStreamReader(proc.getInputStream());
            LineNumberReader input = new LineNumberReader(ir);
            String line = input.readLine();
            if (line == null || line.length() < FAULTLENGTH) {
                return null;
            }
            int capidx = line.indexOf("Caption");
```

```
int cmdidx = line.indexOf("CommandLine");
int rocidx = line.indexOf("ReadOperationCount");
int umtidx = line.indexOf("UserModeTime");
int kmtidx = line.indexOf("KernelModeTime");
int wocidx = line.indexOf("WriteOperationCount");
long idletime = 0;
long kneltime = 0;
long usertime = 0;
while ((line = input.readLine()) != null) {
    if (line.length() < wocidx) {
        continue;
    }
    // 字段出现顺序：Caption,CommandLine,KernelModeTime,ReadOperationCount,
    // ThreadCount,UserModeTime,WriteOperation
    String caption =substring(line, capidx, cmdidx - 1).trim();
    String cmd = substring(line, cmdidx, kmtidx - 1).trim();
    if (cmd.indexOf("wmic.exe") >= 0) {
        continue;
    }
    String s1 = substring(line, kmtidx, rocidx - 1).trim();
    String s2 = substring(line, umtidx, wocidx - 1).trim();
    if (caption.equals("System Idle Process") || caption.equals("System")) {
        if (s1.length() > 0)
            idletime += Long.valueOf(s1).longValue();
        if (s2.length() > 0)
            idletime += Long.valueOf(s2).longValue();
```

```
                continue;
            }
            if (s1.length() > 0)
                kneltime += Long.valueOf(s1).longValue();
            if (s2.length() > 0)
                usertime += Long.valueOf(s2).longValue();
        }
        retn[0] = idletime;
        retn[1] = kneltime + usertime;
        return retn;
    } catch (Exception ex) {
        ex.printStackTrace();
    } finally {
        try {
            proc.getInputStream().close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
    return null;
}

/**
 * 由于String.subString对汉字处理存在问题（把一个汉字视为一个字节)，因此在 包含汉字的字符串时存在隐患，现调整如下：
 * @param src 要截取的字符串
```

```
   * @param start_idx 开始坐标（包括该坐标)
   * @param end_idx 截止坐标（包括该坐标）
   * @return
   */
    private static String substring(String src, int start_idx, int end_idx) {
   byte[] b = src.getBytes();
   String tgt = "";
   for (int i = start_idx; i <= end_idx; i++) {
    tgt += (char) b[i];
   }
   return tgt;
  }
}
```

2.利用第三方的jar包：（Hyperic-hq官方网站：http://www.hyperic.com）

通过Hyperic-hq产品的基础包sigar.jar来实现服务器状态数据的获取。Sigar.jar包是通过本地方法来调用操作系统API来获取系统相关数据。Windows操作系统下Sigar.jar依赖sigar-amd64-winnt.dll或sigar-x86-winnt.dll，linux操作系统下则依赖libsigar-amd64-linux.so或libsigar-x86-linux.so

```
import java.net.InetAddress;
import java.net.UnknownHostException;
import org.hyperic.sigar.CpuInfo;
import org.hyperic.sigar.CpuPerc;
import org.hyperic.sigar.FileSystem;
import org.hyperic.sigar.FileSystemUsage;
```

```
import org.hyperic.sigar.Mem;
import org.hyperic.sigar.NetFlags;
import org.hyperic.sigar.NetInterfaceConfig;
import org.hyperic.sigar.NetInterfaceStat;
import org.hyperic.sigar.OperatingSystem;
import org.hyperic.sigar.Sigar;
import org.hyperic.sigar.SigarException;
import org.hyperic.sigar.SigarNotImplementedException;
import org.hyperic.sigar.Swap;

public class SysInfo {

// 1.CPU资源信息

// a)CPU数量（单位：个）
public static int getCpuCount() throws SigarException {
  Sigar sigar = new Sigar();
  try {
   return sigar.getCpuInfoList().length;
  } finally {
   sigar.close();
  }
}

// b)CPU的总量（单位：HZ）及CPU的相关信息
public void getCpuTotal() {
```

```
  Sigar sigar = new Sigar();
  CpuInfo[] infos;
  try {
   infos = sigar.getCpuInfoList();
   for (int i = 0; i < infos.length; i++) {// 不管是单块CPU还是多CPU都适用
    CpuInfo info = infos[i];
    System.out.println("mhz=" + info.getMhz());// CPU的总量MHz
    System.out.println("vendor=" + info.getVendor());// 获得CPU的卖主，如：Intel
    System.out.println("model=" + info.getModel());// 获得CPU的类别，如：Celeron
    System.out.println("cache size=" + info.getCacheSize());// 缓冲存储器数量
   }
  } catch (SigarException e) {
   e.printStackTrace();
  }
}

// c)CPU的用户使用量、系统使用剩余量、总的剩余量、总的使用占用量等（单位：100%）
public void testCpuPerc() {
  Sigar sigar = new Sigar();
  // 方式一，主要是针对一块CPU的情况
  CpuPerc cpu;
  try {
   cpu = sigar.getCpuPerc();
   printCpuPerc(cpu);
  } catch (SigarException e) {
   e.printStackTrace();
```

```
  }
  // 方式二，不管是单块CPU还是多CPU都适用
  CpuPerc cpuList[] = null;
  try {
   cpuList = sigar.getCpuPercList();
  } catch (SigarException e) {
   e.printStackTrace();
   return;
  }
  for (int i = 0; i < cpuList.length; i++) {
   printCpuPerc(cpuList[i]);
  }
}

private void printCpuPerc(CpuPerc cpu) {
  System.out.println("User :" + CpuPerc.format(cpu.getUser()));// 用户使用率
  System.out.println("Sys :" + CpuPerc.format(cpu.getSys()));// 系统使用率
  System.out.println("Wait :" + CpuPerc.format(cpu.getWait()));// 当前等待率
  System.out.println("Nice :" + CpuPerc.format(cpu.getNice()));//
  System.out.println("Idle :" + CpuPerc.format(cpu.getIdle()));// 当前空闲率
  System.out.println("Total :" + CpuPerc.format(cpu.getCombined()));// 总的使用率
}

// 2.内存资源信息
public void getPhysicalMemory() {
  // a)物理内存信息
```

```
  Sigar sigar = new Sigar();
  Mem mem;
  try {
   mem = sigar.getMem();
   // 内存总量
   System.out.println("Total = " + mem.getTotal() / 1024L + "K av");
   // 当前内存使用量
   System.out.println("Used = " + mem.getUsed() / 1024L + "K used");
   // 当前内存剩余量
   System.out.println("Free = " + mem.getFree() / 1024L + "K free");

   // b)系统页面文件交换区信息
   Swap swap = sigar.getSwap();
   // 交换区总量
   System.out.println("Total = " + swap.getTotal() / 1024L + "K av");
   // 当前交换区使用量
   System.out.println("Used = " + swap.getUsed() / 1024L + "K used");
   // 当前交换区剩余量
   System.out.println("Free = " + swap.getFree() / 1024L + "K free");
  } catch (SigarException e) {
   e.printStackTrace();
  }
}

// 3.操作系统信息
```

```
// a)取到当前操作系统的名称：
public String getPlatformName() {
  String hostname = "";
  try {
   hostname = InetAddress.getLocalHost().getHostName();
  } catch (Exception exc) {
   Sigar sigar = new Sigar();
   try {
    hostname = sigar.getNetInfo().getHostName();
   } catch (SigarException e) {
    hostname = "localhost.unknown";
   } finally {
    sigar.close();
   }
  }
  return hostname;
}

// b)取当前操作系统的信息
public void testGetOSInfo() {
  OperatingSystem OS = OperatingSystem.getInstance();
  // 操作系统内核类型如： 386、486、586等x86
  System.out.println("OS.getArch() = " + OS.getArch());
  System.out.println("OS.getCpuEndian() = " + OS.getCpuEndian());//
  System.out.println("OS.getDataModel() = " + OS.getDataModel());//
  // 系统描述
```

```
  System.out.println("OS.getDescription() = " + OS.getDescription());
  System.out.println("OS.getMachine() = " + OS.getMachine());//
  // 操作系统类型
  System.out.println("OS.getName() = " + OS.getName());
  System.out.println("OS.getPatchLevel() = " + OS.getPatchLevel());//
  // 操作系统的卖主
  System.out.println("OS.getVendor() = " + OS.getVendor());
  // 卖主名称
  System.out
    .println("OS.getVendorCodeName() = " + OS.getVendorCodeName());
  // 操作系统名称
  System.out.println("OS.getVendorName() = " + OS.getVendorName());
  // 操作系统卖主类型
  System.out.println("OS.getVendorVersion() = " + OS.getVendorVersion());
  // 操作系统的版本号
  System.out.println("OS.getVersion() = " + OS.getVersion());
}

// c)取当前系统进程表中的用户信息
public void testWho() {
  try {
   Sigar sigar = new Sigar();
   org.hyperic.sigar.Who[] who = sigar.getWhoList();
   if (who != null && who.length > 0) {
    for (int i = 0; i < who.length; i++) {
     System.out.println("\n~~~~~~~~~" + String.valueOf(i)+ "~~~~~~~~~");
```

```
    org.hyperic.sigar.Who _who = who[i];
    System.out.println("getDevice() = " + _who.getDevice());
    System.out.println("getHost() = " + _who.getHost());
    System.out.println("getTime() = " + _who.getTime());
    // 当前系统进程表中的用户名
    System.out.println("getUser() = " + _who.getUser());
   }
  }
 } catch (SigarException e) {
  e.printStackTrace();
 }
}

// 4.资源信息（主要是硬盘）

// a)取硬盘已有的分区及其详细信息（通过sigar.getFileSystemList()来获得FileSystem列表对象，然后对其进行编历）：
public void testFileSystemInfo() throws Exception {
 Sigar sigar = new Sigar();
 FileSystem fslist[] = sigar.getFileSystemList();
 //String dir = System.getProperty("user.home");// 当前用户文件夹路径
 for (int i = 0; i < fslist.length; i++) {
  System.out.println("\n~~~~~~~~~~" + i + "~~~~~~~~~~");
  FileSystem fs = fslist[i];
  // 分区的盘符名称
  System.out.println("fs.getDevName() = " + fs.getDevName());
```

```
// 分区的盘符名称
System.out.println("fs.getDirName() = " + fs.getDirName());
System.out.println("fs.getFlags() = " + fs.getFlags());//
// 文件系统类型，比如 FAT32、NTFS
System.out.println("fs.getSysTypeName() = " + fs.getSysTypeName());
// 文件系统类型名，比如本地硬盘、光驱、网络文件系统等
System.out.println("fs.getTypeName() = " + fs.getTypeName());
// 文件系统类型
System.out.println("fs.getType() = " + fs.getType());
FileSystemUsage usage = null;
try {
 usage = sigar.getFileSystemUsage(fs.getDirName());
} catch (SigarException e) {
 if (fs.getType() == 2)
  throw e;
 continue;
}
switch (fs.getType()) {
case 0: // TYPE_UNKNOWN ：未知
 break;
case 1: // TYPE_NONE
 break;
case 2: // TYPE_LOCAL_DISK : 本地硬盘
 // 文件系统总大小
 System.out.println(" Total = " + usage.getTotal() + "KB");
 // 文件系统剩余大小
```

```
    System.out.println(" Free = " + usage.getFree() + "KB");
    // 文件系统可用大小
    System.out.println(" Avail = " + usage.getAvail() + "KB");
    // 文件系统已经使用量
    System.out.println(" Used = " + usage.getUsed() + "KB");
    double usePercent = usage.getUsePercent() * 100D;
    // 文件系统资源的利用率
    System.out.println(" Usage = " + usePercent + "%");
    break;
   case 3:// TYPE_NETWORK ：网络
    break;
   case 4:// TYPE_RAM_DISK ：闪存
    break;
   case 5:// TYPE_CDROM ：光驱
    break;
   case 6:// TYPE_SWAP ：页面交换
    break;
   }
   System.out.println(" DiskReads = " + usage.getDiskReads());
   System.out.println(" DiskWrites = " + usage.getDiskWrites());
  }
  return;
}

// 5.网络信息
```

```
// a)当前机器的正式域名
public String getFQDN() {
  Sigar sigar = null;
  try {
   return InetAddress.getLocalHost().getCanonicalHostName();
  } catch (UnknownHostException e) {
   try {
    sigar = new Sigar();
    return sigar.getFQDN();
   } catch (SigarException ex) {
    return null;
   } finally {
    sigar.close();
   }
  }
}

// b)取到当前机器的IP地址
public String getDefaultIpAddress() {
  String address = null;
  try {
   address = InetAddress.getLocalHost().getHostAddress();
   // 没有出现异常而正常当取到的IP时，如果取到的不是网卡循回地址时就返回
   // 否则再通过Sigar工具包中的方法来获取
   if (!NetFlags.LOOPBACK_ADDRESS.equals(address)) {
    return address;
```

```
    }
  } catch (UnknownHostException e) {
    // hostname not in DNS or /etc/hosts
  }
  Sigar sigar = new Sigar();
  try {
    address = sigar.getNetInterfaceConfig().getAddress();
  } catch (SigarException e) {
    address = NetFlags.LOOPBACK_ADDRESS;
  } finally {
    sigar.close();
  }
  return address;
}

// c)取到当前机器的MAC地址
public String getMAC() {
  Sigar sigar = null;
  try {
    sigar = new Sigar();
    String[] ifaces = sigar.getNetInterfaceList();
    String hwaddr = null;
    for (int i = 0; i < ifaces.length; i++) {
      NetInterfaceConfig cfg = sigar.getNetInterfaceConfig(ifaces[i]);
      if (NetFlags.LOOPBACK_ADDRESS.equals(cfg.getAddress())
        || (cfg.getFlags() & NetFlags.IFF_LOOPBACK) != 0
```

```
     || NetFlags.NULL_HWADDR.equals(cfg.getHwaddr())) {
     continue;
    }
    /*
     *如果存在多张网卡包括虚拟机的网卡，默认只取第一张网卡的MAC地址，如果要返回所有的网卡（包括物理的和虚拟的）则可以修改方法的返回类型为数组或Collection
     *，通过在for循环里取到的多个MAC地址。
     */
    hwaddr = cfg.getHwaddr();
    break;
   }
   return hwaddr != null ? hwaddr : null;
  } catch (Exception e) {
   return null;
  } finally {
   if (sigar != null)
    sigar.close();
  }
}

// d)获取网络流量等信息
public void testNetIfList() throws Exception {
  Sigar sigar = new Sigar();
  String ifNames[] = sigar.getNetInterfaceList();
  for (int i = 0; i < ifNames.length; i++) {
   String name = ifNames[i];
```

```
   NetInterfaceConfig ifconfig = sigar.getNetInterfaceConfig(name);
   print("\nname = " + name);// 网络设备名
   print("Address = " + ifconfig.getAddress());// IP地址
   print("Netmask = " + ifconfig.getNetmask());// 子网掩码
   if ((ifconfig.getFlags() & 1L) <= 0L) {
    print("!IFF_UP...skipping getNetInterfaceStat");
    continue;
   }
   try {
    NetInterfaceStat ifstat = sigar.getNetInterfaceStat(name);
    print("RxPackets = " + ifstat.getRxPackets());// 接收的总包裹数
    print("TxPackets = " + ifstat.getTxPackets());// 发送的总包裹数
    print("RxBytes = " + ifstat.getRxBytes());// 接收到的总字节数
    print("TxBytes = " + ifstat.getTxBytes());// 发送的总字节数
    print("RxErrors = " + ifstat.getRxErrors());// 接收到的错误包数
    print("TxErrors = " + ifstat.getTxErrors());// 发送数据包时的错误数
    print("RxDropped = " + ifstat.getRxDropped());// 接收时丢弃的包数
    print("TxDropped = " + ifstat.getTxDropped());// 发送时丢弃的包数
   } catch (SigarNotImplementedException e) {
   } catch (SigarException e) {
    print(e.getMessage());
   }
  }
}

void print(String msg) {
```

```
    System.out.println(msg);
  }

  // e)一些其他的信息
  public void getEthernetInfo() {
    Sigar sigar = null;
    try {
      sigar = new Sigar();
      String[] ifaces = sigar.getNetInterfaceList();
      for (int i = 0; i < ifaces.length; i++) {
        NetInterfaceConfig cfg = sigar.getNetInterfaceConfig(ifaces[i]);
        if (NetFlags.LOOPBACK_ADDRESS.equals(cfg.getAddress())
            || (cfg.getFlags() & NetFlags.IFF_LOOPBACK) != 0
            || NetFlags.NULL_HWADDR.equals(cfg.getHwaddr())) {
          continue;
        }
        System.out.println("cfg.getAddress() = " + cfg.getAddress());// IP地址
        System.out
            .println("cfg.getBroadcast() = " + cfg.getBroadcast());// 网关广播地址
        System.out.println("cfg.getHwaddr() = " + cfg.getHwaddr());// 网卡MAC地址
        System.out.println("cfg.getNetmask() = " + cfg.getNetmask());// 子网掩码
        System.out.println("cfg.getDescription() = "
            + cfg.getDescription());// 网卡描述信息
        System.out.println("cfg.getType() = " + cfg.getType());//
        System.out.println("cfg.getDestination() = "
            + cfg.getDestination());
```

```
    System.out.println("cfg.getFlags() = " + cfg.getFlags());//
    System.out.println("cfg.getMetric() = " + cfg.getMetric());
    System.out.println("cfg.getMtu() = " + cfg.getMtu());
    System.out.println("cfg.getName() = " + cfg.getName());
    System.out.println();
   }
  } catch (Exception e) {
   System.out.println("Error while creating GUID" + e);
  } finally {
   if (sigar != null)
    sigar.close();
  }
}

}
```


分享到：

JFreeChart教程(一) | 获得IP

2010-04-04 13:57 | 浏览 8300 | 评论(7) | 分类:编程语言 | 相关推荐 MORE

评论

7 楼 stuxnet 2013-02-20

非常感谢楼主，太棒了，岂是一顶所能抒怀的。

6 楼 zhenjw 2012-12-18

灰常感谢楼主

5 楼 lansor2009 2012-05-10

 非常非常感谢楼主！！！！

4 楼 lenka_xiu 2011-09-05

 眞不错的文章，可是我不知道怎么弄。能加QQ教教我不?我的QQ1009610689

3 楼 cnm-111 2011-08-09

楼主你好，能解释下，如果是机子的cpu是双核的话，cpu的各种统计信息，如总的使用率怎么计算呢？是每一个cpu的值接相加吗？
还有就是那个流量统计的信息代表什么含义？ 求解@。。

2 楼 dllk08 2011-04-06

对这篇文章的赞许不是一言两语啊，找了那么久得解决办法，终于在这里找到答案了。嘿嘿！高兴！谢谢！

1 楼 boshding 2010-11-01

太棒了，岂是一顶所能抒怀的。

发表评论

 您还没有登录,请您登录后再发表评论