先请编译和运行下面程序：

import java.util.\*;

import java.io.\*;

public class BadExecJavac2

{

public static void main(String args[])

{

try

{

Runtime rt = Runtime.getRuntime();

Process proc = rt.exec("javac");

int exitVal = proc.waitFor();

System.out.println("Process exitValue: " + exitVal);

} catch (Throwable t){

t.printStackTrace();

}

}

}

我们知道javac命令，当不带参数运行javac

程序时，它将输出帮助说明，为什么上面程序不产生任何输出并挂起，永不完成呢？java文档上说,由于有些本地平台为标准输入和输出流所提供的缓冲区大小

有限,如果不能及时写入子进程的输入流或者读取子进程的输出流,可能导致子进程阻塞,甚至陷入死锁。所以，上面的程序应改写为：

import java.util.\*;

import java.io.\*;

public class MediocreExecJavac

{

public static void main(String args[])

{

try

{

Runtime rt = Runtime.getRuntime();

Process proc = rt.exec("javac");

InputStream stderr = proc.getErrorStream();

InputStreamReader isr = new InputStreamReader(stderr);

BufferedReader br = new BufferedReader(isr);

String line = null;

System.out.println("");

while ( (line = br.readLine()) != null)

System.out.println(line);

System.out.println("");

int exitVal = proc.waitFor();

System.out.println("Process exitValue: " + exitVal);

} catch (Throwable t){

t.printStackTrace();

}

}

}

下面是正确的输出：

D:\java>java MediocreExecJavac

Usage: javac <options>

where possible options include:

-g Generate all debugging info

-g:none Generate no debugging info

-g:{lines,vars,source} Generate only some debugging info

-nowarn Generate no warnings

-verbose Output messages about what the compiler is doing

-deprecation Output source locations where deprecated APIs are used

-classpath Specify where to find user class files

-cp Specify where to find user class files

-sourcepath Specify where to find input source files

-bootclasspath Override location of bootstrap class files

-extdirs Override location of installed extensions

-endorseddirs Override location of endorsed standards path

-d Specify where to place generated class files

-encoding Specify character encoding used by source files

-source Provide source compatibility with specified release

-target Generate class files for specific VM version

-version Version information

-help Print a synopsis of standard options

-X Print a synopsis of nonstandard options

-J Pass directly to the runtime system

Process exitValue: 2

D:\java>

下面是一个更一般的程序，它用两个线程同步清空标准错误流和标准输出流，并能根据你所使用的windows操作系统选择windows命令解释器command.com或cmd.exe,然后执行你提供的命令。

import java.util.\*;

import java.io.\*;

class StreamGobbler extends Thread

{

InputStream is;

String type; //输出流的类型ERROR或OUTPUT

StreamGobbler(InputStream is, String type)

{

this.is = is;

this.type = type;

}

public void run()

{

try

{

InputStreamReader isr = new InputStreamReader(is);

BufferedReader br = new BufferedReader(isr);

String line=null;

while ( (line = br.readLine()) != null)

{

System.out.println(type + ">" + line);

System.out.flush();

}

} catch (IOException ioe)

{

ioe.printStackTrace();

}

}

}

public class GoodWindowsExec

{

public static void main(String args[])

{

if (args.length < 1)

{

System.out.println("USAGE: java GoodWindowsExec ");

System.exit(1);

}

try

{

String osName = System.getProperty("os.name" );

System.out.println("osName: " + osName);

String[] cmd = new String[3];

if(osName.equals("Windows XP") ||osName.equals("Windows 2000"))

{

cmd[0] = "cmd.exe" ;

cmd[1] = "/C" ;

cmd[2] = args[0];

}

else if( osName.equals( "Windows 98" ) )

{

cmd[0] = "command.com" ;

cmd[1] = "/C" ;

cmd[2] = args[0];

}

Runtime rt = Runtime.getRuntime();

System.out.println("Execing " + cmd[0] + " " + cmd[1]+ " " + cmd[2]);

Process proc = rt.exec(cmd);

// any error message?

StreamGobbler errorGobbler = new StreamGobbler(proc.getErrorStream(), "ERROR");

// any output?

StreamGobbler outputGobbler = new StreamGobbler(proc.getInputStream(), "OUTPUT");

// kick them off

errorGobbler.start();

outputGobbler.start();

// any error???

int exitVal = proc.waitFor();

System.out.println("ExitValue: " + exitVal);

} catch (Throwable t){

t.printStackTrace();

}

}

}

下面是一个测试结果：

D:\java>java GoodWindowsExec "copy Test.java Test1.java"

osName: Windows XP

Execing cmd.exe /C copy Test.java Test1.java

OUTPUT>已复制 1 个文件。

ExitValue: 0

D:\java>

下面的测试都能通过(windows xp+jdk1.5)

D:\java>java GoodWindowsExec dir

D:\java>java GoodWindowsExec Test.java

D:\java>java GoodWindowsExec regedit.exe

D:\java>java GoodWindowsExec NOTEPAD.EXE

D:\java>java GoodWindowsExec first.ppt

D:\java>java GoodWindowsExec second.doc

function TempSave(ElementID)

{

CommentsPersistDiv.setAttribute("CommentContent",document.getElementById(ElementID).value);

CommentsPersistDiv.save("CommentXMLStore");

}

function Restore(ElementID)

{

CommentsPersistDiv.load("CommentXMLStore");

document.getElementById(ElementID).value=CommentsPersistDiv.getAttribute("CommentContent");

}