atitit.跨语言执行cmd cli api的原理及兼容性设计草案

[1. 标准输入，标准输出，标准错误与重新定向 1](#_Toc27195)

[2. 常见问题 2](#_Toc8643)

[2.1. 执行bat文件 2](#_Toc14580)

[2.2. 执行bat文件  /c  与/k /k  参数，有利于调试 2](#_Toc25077)

[2.3. Bat参数的隐藏命令：：使用@后头符 2](#_Toc25445)

[2.4. 空格的问题 双引号解决 2](#_Toc20779)

[2.5. 回显乱码 cmd win默认是gbk编码..使用gbk编码读取佐ok 2](#_Toc15174)

[2.6.  双引号转译问题 2](#_Toc1440)

[2.7. Name param的问题 2](#_Toc25533)

[3. Java code 3](#_Toc19539)

[4. C#.net code 4](#_Toc6782)

[5. 参考 5](#_Toc9772)

# ****标准输入，标准输出，标准错误与重新定向****

当Linux执行一个程序的时候，会自动打开三个流，标准输入(standard input)，标准输出(standard output)，标准错误(standard error)。比如说你打开命令行的时候，默认情况下，命令行的标准输入连接到键盘，标准输出和标准错误都连接到屏幕。对于一个程序来说，尽管它总会打开这三个流，但它会根据需要使用，并不是一定要使用。

作者:: 绰号:老哇的爪子 （ 全名：：Attilax Akbar Al Rapanui 阿提拉克斯 阿克巴 阿尔 拉帕努伊 ） 汉字名：艾龙，  EMAIL:1466519819@qq.com

转载请注明来源： http://blog.csdn.net/attilax

# 常见问题

## 执行bat文件

## **执行bat文件  /c  与/k /k  参数，有利于调试**

。如果使用 cmd/c 就可以省掉 EXIT 了

使用process explorer可以看到实际的调用参数

## **Bat参数的隐藏命令：：使用@后头符**

## **空格的问题** 双引号解决

## **回显乱码 cmd win默认是gbk编码..使用gbk编码读取佐ok**

 header('Content-Type:text/html;charset=gbk');

## **双引号转译问题**

好像是三个双引号

## Name param的问题

参考apache cli模块即可。

# Java code

**package** com.attilax.cmd;

**import** java.io.BufferedReader;

**import** java.io.IOException;

**import** java.io.InputStream;

**import** java.io.InputStreamReader;

**import** java.io.UnsupportedEncodingException;

**public** **class** CmdX {

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

// **TODO** Auto-generated method stub

}

**public** **static** **void** redirectEcho2Screen(Process process) {

System.***out***.println("--echo stdin info:");

*echoCmdResult\_out*(process.getInputStream());

System.***out***.println("--echo err info:");

*echoCmdResult\_out*(process.getErrorStream());

}

**private** **static** BufferedReader echoCmdResult\_out(InputStream is)

{

BufferedReader reader = **null**;

// BufferedReader reader;

InputStreamReader inputStreamReader;

**try** {

inputStreamReader = **new** InputStreamReader(is,

"gbk");

} **catch** (UnsupportedEncodingException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

**throw** **new** RuntimeException(e);

}

reader = **new** BufferedReader(inputStreamReader);

String line = **null**;

**try** {

line = reader.readLine();

} **catch** (IOException e1) {

// **TODO** Auto-generated catch block

e1.printStackTrace();

}

**while** (line != **null**) {

System.***out***.println(line);

**try** {

line = reader.readLine();

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

**if** (reader != **null**) {

**try** {

reader.close();

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

**return** reader;

}

}

# C#.net code

参考Atitit.执行cli cmd的原理与调试 - attilax的专栏 - 博客频道 - CSDN.NET.htm

# 参考

Linux文本流和标准输入、标准输出、标准错误与重定向介绍\_LINUX\_操作系统\_脚本之家.htm

标准输入、输出和错误输出 - huaijin - ITeye技术网站.htm

Atitit.执行cli cmd的原理与调试 - attilax的专栏 - 博客频道 - CSDN.NET.htm