**一.shell脚本**

**1./dev**

  /dev包含了物理设备的条目，这些设备可能以硬件的形式出现，也可能不会。

   /dev/tcp/$host/$port是一个伪设备，只要读取或者写入这个文件，Bash会打开一个TCP连接。

   (echo >/dev/tcp/$host/$port)>/dev/null 2>&1

   /dev/null：所有output到/dev/null中的信息就无法再找到也不会再屏幕上输出，处理不需要输出信息的一种常用手段。

**2.shell脚本的一些特殊符号**

**>**和**>>**：重定向输出stdout，一个覆盖原有内容，一个不覆盖。command > file 2&>1 表示stdout和stderr都重定向到file。

      cmd > file 2>&1与cmd 2>&1 >file

**<**和**<<**：cmd<file表示cmd命令从file中读取，将stdin重定向到file，<<表示将分解符之间的内容作为输入。

**|**：连接上个指令标准输出作为下个指令输入。     如：who | wc -l

**?**和**\***：?可以用来匹配任何一个非null字元，\*可以匹配包括null的任意字元。

**&**：放在指令最后段，表示将该指令放入后台工作。

**#   ;   ;;**(终止case选项)   **"  "**和**' '**(" "中可以有部分特殊字符，而' '中所有的特殊字符都是无效的) **,   \   /   `**command**`**(命令替换)   **:**(等价于nop)

**$**(变量替换)

   []和[[]]**：**[[]]中可以出现&&和||，<，>操作符，在[]中会报错；且[[]]支持字符串模式匹配；[]是bash内部命令。

**3.变量替换**

bash变量不区分类型

|  |  |
| --- | --- |
| $0 | 当前脚本的文件名 |
| $n | 传递给脚本或函数的参数。n 是一个数字，表示第几个参数。 |
| $# | 传递给脚本或函数的参数个数。 |
| $\* | 传递给脚本或函数的所有参数。 |
| $@ | 传递给脚本或函数的所有参数。被双引号(" ")包含时，与 $\* 稍有不同。 |
| $? | 上个命令的退出状态，或函数的返回值。 |
| $$ | 当前Shell进程ID，也就是脚本所在的进程ID。 |

**3.变量截取**

var=http://www.baidu.com/taobao

# 号截取，从左边开始删除第一个 /号及左边的所有字符。${var#\*/}为/www.baidu.com/taobao

      ## 号截取，从左边开始删除最后（最右边）一个 / 号及左边的所有字符。${var#\*/}为taobao

      %号截取，从右边开始，删除第一个 / 号及右边的字符。echo ${var%%/\*}为<http://www.baidu.com/>

      %% 号截取，从右边开始，删除最后（最左边）一个 / 号及右边的字符。echo ${var%%/\*}为http:

      echo ${var:0:5}即http:

      echo ${var:7}即www.baidu.com/taobao

      echo ${var:0-5:3}即tao

      echo ${var:0-7}即taobao

**4.bash命令执行**

   内部命令：shell程序的一部分，在系统启用时就调入内存的，常驻内存，所以执行效率高

                      解析内部命令shell无需创建子进程。

   外部命令：系统的软件功能，需要时才从硬盘调入内存，其执行过程由shell控制。

   内部命令用户输入时系统调用的速率快，不是内置命令，系统将会读取环境变量文件去找PATH路径。

   hash->内置命令->PATH

   shell是用户和[Linux内核](https://www.baidu.com/s?wd=Linux%E5%86%85%E6%A0%B8&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YznvmvrAN9P17hPjKbnWnk0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EPHRYnjbsn1cs)之间的接口程序，如果把[Linux内核](https://www.baidu.com/s?wd=Linux%E5%86%85%E6%A0%B8&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YznvmvrAN9P17hPjKbnWnk0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EPHRYnjbsn1cs)想象成一个球体的中心，shell就是围绕内核的外层。当从shell或其他程序向

      Linux传递命令时，内核会做出相应的反应。 shell是一个[命令语言](https://www.baidu.com/s?wd=%E5%91%BD%E4%BB%A4%E8%AF%AD%E8%A8%80&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YznvmvrAN9P17hPjKbnWnk0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EPHRYnjbsn1cs)解释器，它拥有自己内建的shell命令集，shell也能被系统中其他应用

      程序所调用。用户在提示符下输入的命令都由shell先解释然后传给Linux核心。

**二.csv格式文件**

**文件格式：**

      每条记录占一行，以逗号为分隔符

      逗号前后的空格会被忽略

      字段中包含有逗号，换行符，空格该字段必须用双引号括起来

      第一条记录，可以是字段名

      示例：

      Jack , McGinnis,220 hobo Av.,Phila, PA,09119  
      "John ","Da Man","",Repici,120 Jefferson St.,Riverside, NJ,08075

     使用javacsv的jar包进行解析：

maven导入jar包，进入[http://www.mvnrepository.com/](http://jump.bdimg.com/safecheck/index?url=x+Z5mMbGPAte3G6X0SMjyeANUNnUAlvT3qH6csw2dn8ScA4MGJ4qSl192Cb/yrLE4/SNmlZh8Kz6B+nRSrkXimkC/PTQLCWMjhdN+keQlOelLeO7GUOV/kRjl/zSIMlXVfdkwJqFldMwPGbuJnYGNA==),搜索想要导入的包名，然后就会列出相关的jar，点进去就会有dependency的信息，复制即可。

**Pattern和Matcher类：**

字符串类型的正则表达式必须首先被编译为此类的实例。然后，可将得到的模式用于创建Matcher对象，依照正则表达式，该

   对象可以与任意字符串匹配。

Pattern p = Pattern.compile("a\*b");

Matcher m = p.matcher("aaaaaab");

m.matches();

      pattern方法：Pattern **compile**(String str), String **pattern**(), String**toString**(), Matcher **matcher**(CharSequence c),

                              public [S](http://download.java.net/jdk/jdk-api-localizations/jdk-api-zh-cn/publish/1.6.0/html/zh_CN/api/java/lang/String.html)tring[] **split**(CharSequence input)

      Matcher常用方法：boolean **find**(), String **group**() , Pattern pattern(), **reset**(), int **groupCount**(), int **start**(),

                                       boolean **matches**()

**正则表达式：**

      基础符号：句点符号，匹配任意一个字符

      方括号符号，只有方括号内的字符才可已参与匹配，a[asd]b可以匹配为aab，asb，adb。

      或符号，a(a|s|d|dd)b可以匹配为aab，asb，adb，addb。此处不可用[]。

      匹配次数符号，\* 0次或多次，+ 1次或多次，? 0次或一次， {n} 恰好n此， {n,m} n到m次。

      否符号，[^a]  不匹配a

      \d,\D,\w,\W,\s,\S  \s表示回车(\r)，换行(\n)，换页(\f)等空白。

     （\r ： return 到当前行的最左边。

       \n： newline 向下移动一行，并不移动左右。

[Linux](http://lib.csdn.net/base/linux)中\n表示回车+换行；

       Windows中\r\n表示回车+换行。

       Mac中\r表示回车+换行）

  匹配：   "June 24, 1949"   对应 [a-Z]+\s+[0-9]{1,2},\s\*[0-9]{4}

              "test.test@mail.com"  对应(\w)+(\.\w+)\*@(\w)+((\.\w+)+)

               简单ip匹配    \d+\.\d+\.\d+\.\d+

               ip匹配    ((?:(?:25[0-5]|2[0-4]\d|[01]?\d?\d)\.){3}(?:25[0-5]|2[0-4]\d|[01]?\d?\d))  其中的?:表示非捕获匹配，(?:)中的内容不能进一步处理，即Matcher的group方法获取不到。

               匹配xml格式文件     ([a-zA-Z]+-?)+[a-zA-Z0-9]+\.[x|X][m|M][l|L]