常用Bash技巧总结

Alex / 2018-02-04 / free_learner@163.com / learning-archive.org

更新于2023-07-05，主要是文字排版上的更新，内容基本保持不变。

总结自己在实践中经常用到的一些Bash技巧，包括使用xargs传递参数、命令替换、定义和索引数组、操作行和列、向脚本或函数传递参数等。

一、使用xargs传递参数

管道（pipe）常用来将前一个命令的输出作为后一个命令的输入，比如：

```
## 统计以.txt结尾的文件个数
ls *.txt | wc -l
```

但是有些命令不能将前一个命令的输出直接作为输入，xargs命令可以将前一个命令的输出转换成参数，作为后一个命令的输入，比如：

```
## 删除以.txt结尾的文件
ls *.txt | xargs rm
```

二、命令替换（command substitution）

命令替换就是将一个命令执行的结果作为另一个命令的参数，有两种表达方式：

1. 使用backquotes/backticks

```
## 计算3 + 2，并将结果赋值给变量X
X=`expr 3 + 2`
```

2. 使用dollar-parentheses

```
## 计算3 + 2，并将结果赋值给变量X
X=$(expr 3 + 2)
```

三、定义和索引数组

1. 使用圆括号定义数组

```
## 定义一个包含两个元素的数组
myArr=(Hello World)
## 定义元素含有空格的数组
myArr=("Hello World" 2018)
```

2. 使用方括号索引数组

```
## 索引第一个元素
echo ${myArr[0]}
## 索引所有元素
echo ${myArr[*]}
echo ${myArr[@]}
```

四、操作行和列

1. 统计行数和列数

```
## 统计文本文件的行数
cat myfile.txt | wc -l
## 统计文本文件的列数
head -n 1 myfile.txt | awk '{print NF}'
```

2. 提取行或列

```
## 提取第二行
cat myfile.txt | sed -n 2p
## 提取第二行和第四行
cat myfile.txt | sed -n '2p;4p'
## 提取第二至第四行
cat myfile.txt | sed -n '2,4p'
## 提取第二列，假设分隔符为空格
cat myfile.txt | cut -d ' ' -f2
## 提取第二列
cat myfile.txt | awk '{print $2}'
## 提取第二列或第四列
cat myfile.txt | cut -d ' ' -f2,4
## 提取第二列或第四列
cat myfile.txt | awk '{print $2, $4}'
## 提取第二至第四列
cat myfile.txt | cut -d ' ' -f2-4
```

五、向脚本或函数传递参数

1. 使用位置参数

```
## 定义计算两个数之和的函数
## $1表示第一个参数，$2表示第二个参数
addTwoNum () {
      echo "$1 + $2" |  bc
}
## 调用函数
addTwoNum 1.5 3
```

2. 使用getopts命令

```
## 定义计算两个数之和的函数，选项a表示第一个数，选项b表示第二数
## a:b:表示选项字符串，只支持单字母选项，冒号（:）表示该选项后需要参数
## 自定义变量optname用于存放选项，内置变量OPTARG用于存放选项对应的参数
TwoNumSum () {
      while getopts a:b: optname
      do
       case $optname in
           a) Num1=${OPTARG};;
           b) Num2=${OPTARG};;
           *) echo "invalid arguments";;
        esac
      done
      echo "${Num1} + ${Num2}" | bc
}
##调用函数
TwoNumSum -a 1.5 -b 3
```