

Linux 双引号、单引号、反引号、\$、\$()、\${ }

```
lcc@localhost:~$ value=3
lcc@localhost:~$ echo "$value"
3
lcc@localhost:~$ echo '$value'
$value
lcc@localhost:~$ a='pwd'
lcc@localhost:~$ echo $a
pwd
lcc@localhost:~$ a='pwd'
lcc@localhost:~$ echo $a
/home/lcc
lcc@localhost:~$ b=$(date)
lcc@localhost:~$ echo $b
2022年 08月 29日 星期一 18:59:59 PDT
lcc@localhost:~$ happiness=36
lcc@localhost:~$ echo $happiness
36
lcc@localhost:~$ |
```

单引号与双引号

单引号'剥夺了所有字符的特殊含义，单引号'内就变成了单纯的字符，。

双引号"则对于双引号""内的参数替换(\$)和命令替换(``)是个例外。

```
[lcc@localhost ~]$ value=3
[lcc@localhost ~]$ echo "$value"
3
[lcc@localhost ~]$ echo '$value'
$value
```

反引号与\$()

反引号是命令替换，命令替换是指Shell可以先执行反引号中的命令，将输出结果暂时保存，在适当的地方输出。

\$()也有命令替换的作用

```
[lcc@localhost ~]$ a=`pwd`  
[lcc@localhost ~]$ echo $a  
/home/lcc  
[lcc@localhost ~]$ b=$(date)  
[lcc@localhost ~]$ echo $b  
2022年 08月 29日 星期一 18:59:59 PDT
```

`${ }`与`$`

`${ }`中放的是变量，例如`echo ${hello}`取`hello`变量的值并打印，也可以不加括号比如`$hello`。

`$`的作用相当于引用某变量的值，例如变量`myvar`，我们需要通过`$`取得变量的值，即`$myvar`

```
[lcc@localhost ~]$ happiness=36  
[lcc@localhost ~]$ echo $happiness  
36
```