A+

bash中 [ ] 与 [[ ]] 的区别

如果直接描述 [ ] 与 [[ ]] 的区别，反而不太容易理解，不如先来看一些应用场景，根据应用场景，反而更容易理解。

场景一：判断变量是否为空

我们可以直接判断变量是否为空，方法如下

```
[www.zsythink.net]# a=abc
[www.zsythink.net]# echo $a
abc
[www.zsythink.net]# [ $a ]
[www.zsythink.net]# echo $?
0
[www.zsythink.net]# [[ $a ]]
[www.zsythink.net]# echo $?
0
[www.zsythink.net]# b=""
[www.zsythink.net]# echo $b

[www.zsythink.net]# [ $b ]
[www.zsythink.net]# echo $?
1
[www.zsythink.net]# [[ $b ]]
[www.zsythink.net]# echo $?
1
[www.zsythink.net]#
```

如上图所示，变量值非空时返回真（即返回值为0），使用上述方法判断变量值是否为空时，[ ] 与 [[ ]] 没有区别，上例中，变量值非空，返回真，我们可以使用"!"进
变量值为空时，返回真，示例如下

```
[www.zsythink.net]# echo $c

[www.zsythink.net]# [ ! $c ]
[www.zsythink.net]# echo $?
0
[www.zsythink.net]# [[ ! $c ]]
[www.zsythink.net]# echo $?
0
[www.zsythink.net]# ! [ $c ]
[www.zsythink.net]# echo $?
0
[www.zsythink.net]# ! [[ $c ]]
[www.zsythink.net]# echo $?
0
[www.zsythink.net]#
```

如上图所示，变量c是一个没有被声明赋值的变量，其值为空，我们可以使用上述语法，判断变量c的值是否为空，变量值为空，返回真，同理，上述示例中，[ ] 与 [

那么在判断变量是否为空时，[ ] 与 [[ ]] 的区别在哪里呢？不要着急，我们慢慢聊。

我们知道，在Linux中，我们可以使用test命令判断一个字符串是否为空，test命令为我们提供了"-z选项"与"-n选项"，使用这两个选项可以判断字符串是否为空。
"-z选项"可以判断指定的字符串是否为空，为空则返回真，非空则返回假，-z可以理解为zero
"-n选项"可以判断指定的字符串是否为空，非空则返回真，为空则返回假，-n可以理解为nozero
示例如下

```
[www.zsythink.net]# test -z ""
[www.zsythink.net]# echo $?
0
[www.zsythink.net]# test -z "abc"
[www.zsythink.net]# echo $?
1
[www.zsythink.net]# test -n "abc"
[www.zsythink.net]# echo $?
0
[www.zsythink.net]# test -n ""
[www.zsythink.net]# echo $?
1
[www.zsythink.net]#
```

正如上图所示，我们通过test命令判断了字符串是否为空，那么我们来尝试一下，使用test命令判断变量的值是否为空，示例如下

```
[www.zsythink.net]# a=abc
[www.zsythink.net]# echo $a
abc
[www.zsythink.net]# echo $b

[www.zsythink.net]# test -z $b
[www.zsythink.net]# echo $?
0
[www.zsythink.net]# test -z $a
[www.zsythink.net]# echo $?
1
[www.zsythink.net]# test -n $a
[www.zsythink.net]# echo $?
0
[www.zsythink.net]# test -n $b
[www.zsythink.net]# echo $?
0
[www.zsythink.net]#
```

上例中，变量b的值为空，按照正常的逻辑来说，使用test -n 命令判断变量b的值是否为空时，应该返回假，因为test命令的-n选项表示指定的字符串非空时，返回
返回假，但是上例中，'test -n $b' 这条命令的返回值却为真（应该为假），这是明显不正确的，所以，为了防止上述情况的发生，在使用test命令的-n选项判断变
时，需要在变量的外侧加上"双引号"，示例如下

```
[www.zsythink.net]# echo $b

[www.zsythink.net]# test -n $b
[www.zsythink.net]# echo $?
0
[www.zsythink.net]# test -n "$b"
[www.zsythink.net]# echo $?
1
[www.zsythink.net]#
```

zsythink.net 来双印博客

好了，我们已经明白了使用test命令判断变量是否为空时的一些注意点，那么话说回来，这篇文章的主题是介绍[ ]与[[ ]]的区别的，为什么我们要先介绍test命令呢
x中，"[ ]"与"test命令"是等效的，比如，我们也可以使用"-n"或者"-z"结合"[ ]"去判断变量是否为空

```
[www.zsythink.net]# a=abc
[www.zsythink.net]# echo $a
abc
[www.zsythink.net]# echo $b

[www.zsythink.net]# [ -n "$a" ]
[www.zsythink.net]# echo $?
0
[www.zsythink.net]# [ -n "$b" ]
[www.zsythink.net]# echo $?
1
[www.zsythink.net]# [ -n $b ]
[www.zsythink.net]# echo $?
0
[www.zsythink.net]#
```

根据上例中的结果可以看出，当"[ ]"中使用"-n"或者"-z"这些选项判断变量是否为空时，必须在变量的外侧加上双引号，才更加保险，与"test命令"的使用方法相同。
不过，使用"[[ ]]"时则不用考虑这样的问题，示例如下

```
[www.zsythink.net]# a=abc
[www.zsythink.net]# echo $a
abc
[www.zsythink.net]# echo $b

[www.zsythink.net]# [[ -n $a ]]
[www.zsythink.net]# echo $?
0
[www.zsythink.net]# [[ -n $b ]]
[www.zsythink.net]# echo $?
1
[www.zsythink.net]# [[ -z $b ]]
[www.zsythink.net]# echo $?
0
[www.zsythink.net]# [[ -z $a ]]
[www.zsythink.net]# echo $?
1
[www.zsythink.net]#
```

综上所述，我们可以得出如下结论：

当使用"-n"或者"-z"这种方式判断变量是否为空时，"[ ]"与"[[ ]]"是有区别的。

使用"[ ]"时需要在变量的外侧加上双引号，与test命令的用法完全相同，使用"[[ ]]"时则不用。

场景二：组合判断条件

在使用shell脚本时，判断几乎是必不可少的，而很多时候，如果想要得到最终的判断结果，可能需要同时对多个条件进行判断，比如，条件一与条件二必须同时为真，再比如，条件一或条件二只要有一个为真，结果即为真。没错，这时，多个条件之间存在"与"或者"或"的关系。

在shell中，我们可以使用"-a"或者"-o"对多个条件进行连接，然后进行"与运算"或者"或运算"，也可以使用"&&"或者"||"对多个条件进行连接，但是，这两种方法对]]"来说，是存在区别的，我们通过一些小例子来了解一下这些区别。

简单示例如下

```
[www.zsythink.net]# [[ 3 -gt 1 && 5 -lt 8 ]]
[www.zsythink.net]# echo $?
0
[www.zsythink.net]# [[ 5 -gt 2 || 9 -lt 3 ]]
[www.zsythink.net]# echo $?
0
[www.zsythink.net]# [[ 3 -gt 1 ]] && [[ 5 -lt 8 ]]
[www.zsythink.net]# echo $?
0
[www.zsythink.net]# [[ 5 -gt 2 ]] || [[ 9 -lt 3 ]]
[www.zsythink.net]# echo $?
0
[www.zsythink.net]#
```

zsythink.net来双印博客

如上图所示，当使用"[[ ]]"对多个条件进行"与运算"或者"或运算"时，可以一次性将多个条件都包含在一个"[[ ]]"中，然后将每个条件用"&&"或者用"||"连接起来，或 别包含在一个"[[ ]]"中，然后再用"&&"或者用"||"连接起来，正如上图所示，这两种写法都是没有问题的。

那么，使用"[[ ]]"时，能否使用"-a"或者"-o"对多个条件进行连接呢？我们来实验一下，示例如下

```
[www.zsythink.net]# [[ 3 -gt 1 -a 5 -lt 8 ]]
-bash: syntax error in conditional expression
-bash: syntax error near `-a'
[www.zsythink.net]# [[ 3 -gt 1 ]] -a [[ 5 -lt 8 ]]
-bash: syntax error near unexpected token `-a'
[www.zsythink.net]# [[ 5 -gt 2 -o 9 -lt 3 ]]
-bash: syntax error in conditional expression
-bash: syntax error near `-o'
[www.zsythink.net]# [[ 5 -gt 2 ]] -o [[ 9 -lt 3 ]]
-bash: syntax error near unexpected token `-o'
[www.zsythink.net]#
[www.zsythink.net]#
```

zsythink.net来双印博客

看来，使用"[[ ]]"时，是不能使用"-a"或者"-o"对多个条件进行连接的。

仍然使用上述套路，我们将"[[ ]]"换成"[ ]"试试。

```
[www.zsythink.net]# [ 3 -gt 1 -a 5 -lt 8 ]
[www.zsythink.net]# echo $?
0
[www.zsythink.net]# [ 3 -gt 1 ] -a [ 5 -lt 8 ]
-bash: [: too many arguments
[www.zsythink.net]# [ 5 -gt 2 -o 9 -lt 3 ]
[www.zsythink.net]# echo $?
0
[www.zsythink.net]# [ 5 -gt 2 ] -o [ 9 -lt 3 ]
-bash: [: too many arguments
[www.zsythink.net]#
[www.zsythink.net]#
```

zsythink.net来双印博客

看来，当使用"[ ]"时，如果使用"-a"或者"-o"对多个条件进行连接，"-a"或者"-o"必须被包含在"[ ]"之内，才能够正常使用，否则会报语法错误。

"-a"或者"-o"的使用方法我们已经在"[ ]"中进行了验证，现在，我们来试试"&&"或者"||"在"[ ]"中的使用方法，示例如下

```
[www.zsythink.net]# [ 3 -gt 1 && 5 -lt 8 ]
-bash: [: missing `]'
[www.zsythink.net]# [ 3 -gt 1 ] && [ 5 -lt 8 ]
[www.zsythink.net]# echo $?
0
[www.zsythink.net]# [ 5 -gt 2 || 9 -lt 3 ]
-bash: [: missing `]'
bash: 9: command not found...
[www.zsythink.net]# [ 5 -gt 2 ] || [ 9 -lt 3 ]
[www.zsythink.net]# echo $?
0
[www.zsythink.net]#
```

zsythink.net来双印博客

从上例中可以看出，与"-a"或者"-o"的使用方法正好相反，当使用"[ ]"时，如果使用"&&"或者"||"对多个条件进行连接，"&&"或者"||"必须在"[ ]"之外，否则会报错

综上所述，我们可以总结出如下结论：

在使用"[[ ]]"时，不能使用"-a"或者"-o"对多个条件进行连接。

在使用"[ ]"时，如果使用"-a"或者"-o"对多个条件进行连接，"-a"或者"-o"必须被包含在"[ ]"之内。

在使用"[ ]"时，如果使用"&&"或者"||"对多个条件进行连接，"&&"或者"||"必须在"[ ]"之外。

场景三：某些运算符

如果想要判断变量的值是否满足某个正则表达式，我们可以使用符号"=~"进行判断，示例如下：

```
[www.zsythink.net]# tel=13688888888
[www.zsythink.net]# [[ $tel =~ [0-9]{11} ]]
[www.zsythink.net]# echo $?
0
[www.zsythink.net]# tel=1368888888k
[www.zsythink.net]# [[ $tel =~ [0-9]{11} ]]
[www.zsythink.net]# echo $?
1
[www.zsythink.net]#
[www.zsythink.net]#
```

zsythink.net 朱双印博客

如上图所示，我们通过"=~"，可以判断一个变量的值是否匹配对应的正则表达式，但是细心如你一定发现了，上例中使用了"[[ ]]"，如果把"[[ ]]"替换成"[ ]"，能否呢？我们来试试。

```
[www.zsythink.net]# tel=13688888888
[www.zsythink.net]# [ $tel =~ [0-9]{11} ]
-bash: [: =~: binary operator expected
[www.zsythink.net]#
```

看来是不能这样使用的，所以我们可以得出结论，"=~"只能应用于"[[ ]]"中，不能应用于"[ ]"中。

同样，有些其他符号对于"[[ ]]"或者"[ ]"来说，在使用时也是有区别的，比如">"或者"<"，在之前的文章中，其实已经描述了在使用">"或者"<"时，"[[ ]]"与"[ ]"I你想要了解这些知识点，可以参考如下连接

shell中'-gt'与'>'的区别

小结

当使用"-n"或者"-z"这种方式判断变量是否为空时，"[ ]"与"[[ ]]"是有区别的。

使用"[ ]"时需要在变量的外侧加上双引号，与test命令的用法完全相同，使用"[[ ]]"时则不用。

在使用"[[ ]]"时，不能使用"-a"或者"-o"对多个条件进行连接。

在使用"[ ]"时，如果使用"-a"或者"-o"对多个条件进行连接，"-a"或者"-o"必须被包含在"[ ]"之内。

在使用"[ ]"时，如果使用"&&"或者"||"对多个条件进行连接，"&&"或者"||"必须在"[ ]"之外。

在使用符号"=~"去匹配正则表达式时，只能使用"[[ ]]"，当使用">"或者"<"判断字符串的ASCII值大小时，如果结合"[ ]"使用，则必须对">"或者"<"进行转义。

以上就是我个人总结的一些注意点与使用方法，但是并不一定全面，如果你也有一些其他的使用心得，可以分享出来，我也可以查漏补缺。

上述注意点虽然很细微，但是在不了解这些问题的时候，其实还是比较坑爹的，希望上述总结能够帮助到你~~~

我的微信公众号

关注"实用运维笔记"微信公众号，当博客中有新文章时，可第一时间得知哦~

Shell (bash)