Day 02 Shell脚本基础 使用变量 条件测试及选择 列表式循环

1. Shell脚本基础
2. 脚本：

· 一个可以运行的文本文件，运行后可实现某种功能

· 提前设计可执行语句，用来完成特定的任务的文件

- 解释型程序

- 顺序、批量执行

· 将命令都写入文本文件，然后赋予文本文件执行权限（命令的堆积）

· 非交互式

- 需要提前设计、智能化难度大

- 批量执行、效率高

- 方便在后台静默执行

· 规范：

- #! 环境声明（声明下列可执行代码用什么程序翻译）

- # 注释文本

- 可执行代码

1. 简单脚本技巧

· 使用 | 管道操作

- 将前一条命令的标准输出交给后一条命令处理

· 免交互处理

- 脚本一般在后台执行，要尽量减少人工交互的语句

· 脚本输出信息的处理

- 记录有价值的信息（>> /var/log/foo.log）

- 屏蔽无价值的、干扰性的信息:黑洞设备(垃圾场w)（&> /dev/null）

- 关于重定向输出：

> ：只收集前面命令的正确输出，将其写入文本文件

2> ：只收集前面命令的错误输出，将其写入文本文件

&> ：收集前面命令的正确和错误输出，将其写入文本文件

- 自定义输出：echo '文本字符串'

3.为了增加脚本适应多变的环境、多变的需求，以及为了方便性，我们要使用变量

- 变量：会变化的量，以不变的名称，存储可以变化的值

- 变量的定义：变量命=变量值

· 为了降低脚本使用的复杂性，使用交互式

- read：产生交互的方式，将用户从键盘的输入记录作为输出赋值给一个变量

read -p '请输入你要创建的用户名:' user

· 设置变量时的注意事项

- 若指定的变量名已存在，相当于为此变量重新赋值

- 等号两边不能有空格

- 变量名由字母/数字/下划线组成，区分大小写

- 变量名不能以数字开头，不要使用关键字和特殊字符

· 基本格式

- 引用变量值：$变量名

- 查看变量值：echo $变量名;和常量混用：echo ${变量名}常量

· 补充：

$[] ：运算

' ' ：让所有的特殊字符变成普通字符

$() ：将命令的输出结果作为参数，与``作用相同

3. 变量的种类

· 环境变量

- 常见的环境变量

PWD、PATH、USER、LOGNAME

SHELL、HOME

USER:当前登陆的用户名

· 位置变量

- 在执行脚本时提供的命令行参数

· 预定义变量

-$# 已加载的位置变量的个属

$\* 所有位置变量的值

$? 程序退出后的状态值，0表示正常，其他值异常

1. 条件测试及选择
2. 条件测试

· 常用的测试选项

-e ：文档存在为真

-d ：存在并且为目录，才为真

-f ：存在并且为文本文件，才为真

-r ：存在并且有读取权限，才为真

-w ：存在并且有写入权限，才为真

-x ：存在并且有执行权限，才为真

· 比较整数（带e字母的都有 等于 二字）

-gt ：大于

-ge ：大于等于

-eq ：等于

-ne ：不等于

-lt ：小于

-le ：小于等于

· 字符串对比

==:相等为真

!=:不相等为真

1. if选择结构

· 双分支：当条件满足/不满足时，分别作xx、yy处理

· 多分支处理：

if [条件测试1];then

命令序列xx

elif [条件测试2];then

命令序列yy

elif [条件测试3];then

命令序列aa

else

命令序列zz

fi

经典脚本之成绩：

用户输入考试成绩

大于等于85 输出 优秀

大于等于70 输出 良好

大于等于60 输出 合格

以上条件均不满足 输出 再牛的肖邦，也弹不出哥的悲伤

1. 列表式循环
2. 适用于反复工作的场景
3. for循环处理

· for 变量名 in 值列表

do

扎针 抽血

done