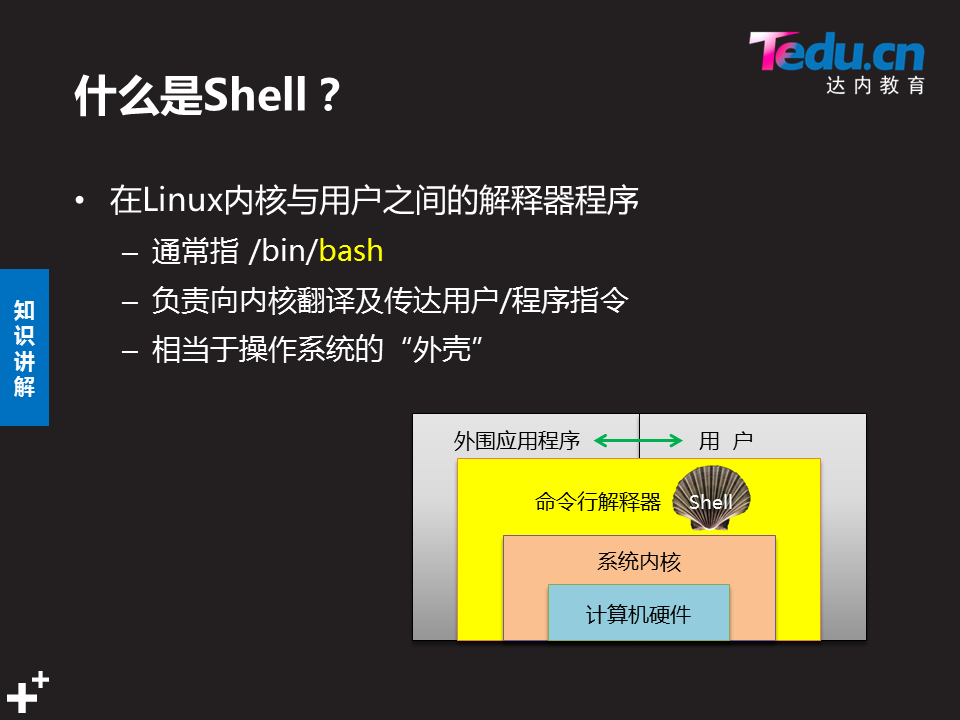
**Linux\_cloud\_shell-day01笔记**

一、什么是SHELL？

在计算机科学中，Shell俗称壳（用来区别于核），是指“提供使用者使用界面”的软件（命令解析器）。它接收用户命令，然后调用相应的应用程序。

1. 在Linux内核与用户之间的**解释器程序**，通常指/bin/bash
2. 负责向内核翻译及传达用户/程序的指令
3. 相当于操作系统内的“外壳”

二、shell的使用方式

shell又是一种[程序设计语言](https://baike.baidu.com/item/%E7%A8%8B%E5%BA%8F%E8%AE%BE%E8%AE%A1%E8%AF%AD%E8%A8%80" \t "https://baike.baidu.com/item/shell/_blank)。作为命令语言，它交互式解释（命令行）和执行用户输入的命令或者自动地解释和执行预先设定好的一连串的命令（脚本）；作为程序设计语言，它定义了各种变量和参数，并提供了许多在[高级语言](https://baike.baidu.com/item/%E9%AB%98%E7%BA%A7%E8%AF%AD%E8%A8%80" \t "https://baike.baidu.com/item/shell/_blank)中才具有的控制结构，包括循环和分支。

Shell的作用

1、自动化批量系统初始化（update、软件安装、时区设置，安全策略）

2、自动化批量软件部署（LAMP/LNMP）

3、管理应用程序（KVM、集群管理扩容、mysql）

4、自动化备份恢复程序（MYSQL完全备份）

5、自动化信息采集及监控程序（收集系统、应用的状态信息，CPU、Apache、mysql等）

6、增加云主机—>业务上线

1. 常见的shell程序的种类

命令行中查看：#cat /etc/shell

常见的种类

/bin/sh

/bin/bash(Linux默认使用)

/bin/tcsh

/bin/csh

/bin/ksh

1. 什么是shell脚本（如演员的剧本）

就是用各类命令预先放入到一个文件中，方便一次性执行的一个[程序文件](https://baike.baidu.com/item/%E7%A8%8B%E5%BA%8F%E6%96%87%E4%BB%B6" \t "https://baike.baidu.com/item/Shell%E8%84%9A%E6%9C%AC/_blank)，主要是方便管理员进行设置或者管理用的。(命令的集合)

1. 脚本名字以.sh结尾

例如：

vim test.sh (新建脚本)

bash test.sh (执行脚本)

1. 脚本的书写格式

-- #！脚本声明

-- #注释信息

-- 可执行语句（命令）

例子：

#!/bin/bash #文件第一行指定运行的shell

echo "hello the world " #内容直接使用命令行的命令，最好用绝对路径

1. 脚本的执行

（1）赋予权限的执行

赋予权限命令：chmod +x 脚本名

1. 直接输入脚本的绝对路径

#chmod +x test1.sh

#/var/tmp/test1.sh

b.使用相对路径

#cd /var/tmp

#chmod +x test1.sh

#./test1.sh

1. 不赋予权限的执行

#cd /var/tmp

1. bash test1.sh(还可以使用其他解释程序，如sh,ksh...)
2. . test1.sh
3. source test1.sh

五、重定向

1. 覆盖重定向（符号：>）

将原本的内容删除然后添加现在的内容

例子：

（1）将指定内容添加到已创建文件

#touch 1.txt

#echo 123 > 1.txt

#cat 1.txt #在命令行查看1.txt的内容

#echo 456 > 1.txt

#cat 1.txt (查看1.txt内容的变化--覆盖了之前的内容)

1. 将指定内容添加到未创建的文件

#cat 2.txt #查看当前目录是否存在文件2.txt

#echo 123456 > 2.txt #创建文件2.txt并添加内容123456

#cat 2.txt

1. 追加重定向

原本的内容不删除，从最后一行开始添加现在的内容

#cat 2.txt

#echo 789 >> 2.txt (查看1.txt内容的变化--在文件最后一行添加内容)

1. 标准输出(1),错误输出(2)

标准输出(1):>,>>

将命令行的正常执行覆盖或追加保存到指定文件

错误输出(2):2>,2>>

将命令行的执行结果出现的错误提示信息覆盖或追加保存到指定文件

混合重定向：&>,&>>

将命令行的执行结果和出现的错误提示信息一并覆盖或追加保存到指定文件

练习

#cat 1.txt > log

#cat 123.txt >log2

#cat 2.txt >> log

#ls abc >>log2

#ls /var/tmp abc >log 2>log2

#ls /var/tmp abc &>logs

#ls /var/tmp ttt >mylog 2>&1

#cat 2.txt > /dev/null (/dev/null看作"黑洞". 所有写入它的内容都会永远丢失，一般写入无用的内容)

## 应用与练习：

## 1、使用脚本配置yum源：

#!/bin/bash

rm -rf /etc/yum.repos.d/\*

mkdir /ak47

mount /dev/sr0 /ak47 &>/dev/null

echo '

[ak47]

name=test

baseurl=file:///ak47

enable=1

gpgcheck=0

' >/etc/yum.repos.d/ak47.repo

yum clean all &> /dev/null

六、变量

1、什么是变量

变量是一种使用方便的占位符，能够把程序中准备使用的每一段数据都赋给一个简短、易于记忆的名字。变量可以通过变量名访问。在指令式语言中，变量通常是可变的。

1. 定义变量名的规则

格式：变量名=值

--可以包括数字、字母、下划线、不能以数字开头

--变量名区分大小写

--赋值时等号两边不能有空格

--尽量不要使用关键字和特殊字符

--给同一变量多次赋值时，最后一次的赋值生效

1. 变量的应用

#a=1 (设置变量)

#echo $a (调用变量，显示变量a的值)

#env (查看环境变量)

#set (查看所有变量)

#unset a (取消变量a的值)

练习：

#echo a\_1=34

#echo $a\_1

#echo a = 1

#echo a=1

#echo A=2

#echo $a

#echo $A

#echo a=2

#echo a=3

#echo $a

#echo a1=12

#echo $a1

#ehv

#set

#unset a

#echo $a

1. 变量的种类

--系统环境变量

--预定义变量

--位置变量

--自定义变量

**练习：**

--自定义变量与位置变量的应用

#vim test4.sh ;chmod +x test4.sh #“；”连接前后命令，执行完前一条命令后紧接着执行后一条命令

#!/bin/bash

echo $0 显示脚本的名称

echo $1 显示第一个参数

echo $2 显示第二个参数

echo $3 显示第三个参数

echo $\* 显示所有参数

echo $# 统计参数有多少个

#./test4.sh a b c 1 8 7

1. 常用的预定义变量

$$ 显示当前运行进程的PID号

$? 显示上一次命令执行结果的返回状态，0为正常，1或其他表示异常

$0 显示当前运行的进程号或脚本号

$# 已加载的位置变量的个数

$\* 所有位置变量的值

$! 后台最后一个进程的PID号

例子：

echo $$ (显示当前运行进程的pid号)

echo $? #显示上一次命令执行结果的返回状态

位置变量的应用例子--创建账户并修改密码

#vim test5.sh

#!/bin/bash

useradd $1 #创建用户命令，脚本第一个参数为新用户名字

echo $2|passwd --stdin $1 #给新用户修改密码，脚本第二个参数为新密码

#bash test2.sh tom 123456 #执行脚本并赋予参数，分别为脚本中的$1和$2的值（可变）

1. shell中的引号

双引号"" 引用整体

单引号'' 引用整体,屏蔽特殊含义

反引号`` 引用命令,取命令的执行结果

练习：

#a=12

#echo "$a"

#echo '$a'

#echo a b c

#echo "abc"

#echo 'a b c'

#echo '13$a'

#echo "\$a" #"\"取消符号的特殊含义

#tar -czf log-`date +%Y%m%d`.tar.gz /var/log #将/var/log的内容打包名字为log-当前日期.tar.gz

1. 判断-数字判断

-eq 等于equal

-ne 不等于not equal

-gt 大于greater than

-lt 小于less than

-ge 大于等于

-le 小于等于

**应用与练习：**

1. 修改指定用户的密码，用户名和密码由用户输入（交互式）

补充：read -p "提示符（自定义）" 变量

#vim test6.sh

#!/bin/bash

read -p "请输入用户名:" user

stty -echo

read -p "请输入密码:" pass

stty echo

echo $pass |passwd --stdin $user

#bash test3.sh

1. 测试当前网段有多少主机能ping通

#!/bin/bash

for i in {1..254}

do

ping -c 3 -i 0.1 -w 2 192.168.1.$i &>/dev/null

if [ $? -eq 0 ];then

echo "该主机192.168.1.$i,ping通" > ip\_up.txt

else

echo "该主机192.168.1.$i,无法ping通" > ip\_down.txt

fi

done