**shell概述**

shell环境及特性

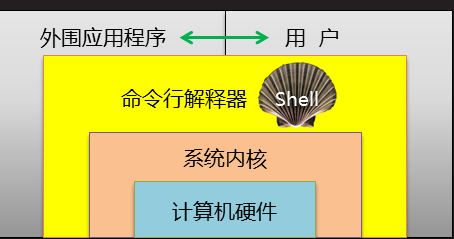
什么是shell

在linux内核与用户之间的解释器程序

通常指/bin/bash

负责向内核解释及传达用户/程序指令

相当于操作系统的“外壳”



shell的使用方式

交互式（命令行）

人工干预、智能化程度高

逐条解释执行、效率低

非交互式（脚本）

需要提前设计、智能化难度大

批量执行、效率高

方便在后台静悄悄的运行

常见的shell程序种类

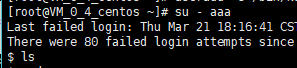
如何切换shell环境

通过usermod、chsh更改登录shell

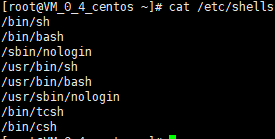
usermod -s 解释器 用户名

useradd -s 解释器 用户名





手动执行目标shell程序



多数unix默认的是第一个shell，在rhel系统中，实际上sh是bash的符号链接，多数linux默认使用的第二个shell，第三个是非登录的shell

bash基本特性

命令行环境回顾

快捷键 tab键补齐

命令历史history

命令别名alias unalias

标准输入输出

重定向

管道操作，可以将一条命令的标准输出交给另一条的标准输入，在一条命令行内可以依次使用多个管道

历史命令

存放位置及数量

默认记录1000条

保存位置：~/.bash\_history

控制历史命令的数量：/etc/profile



操作历史命令

history工具

history：查看历史命令列表

history -c：清空历史命令

调用历史命令

!78：执行历史记录中的第78条命令

!str：执行最近一次以str开头的历史命令

I/O设备与重定向

交互式硬件设备

标准输入：从该设备接收用户输入的数据

标准输出：通过该设备向用户输出数据

标准错误：通过该设备报告执行中的错误信息

|  |  |  |  |
| --- | --- | --- | --- |
| 类型 | 设备文件 | 文件描述号 | 默认设备 |
| 标准输入 | /dev/stdin | 0 | 键盘 |
| 标准输出 | /dev/stdout | 1 | 显示器 |
| 标准错误输出 | /dev/stderr | 2 | 显示器 |

重定向操作

改变标准输入/输出/错误输出的方向

|  |  |  |
| --- | --- | --- |
| 类型 | 操作符 | 用途 |
| 重定向 | < | 将文本输入来源由键盘改为指定的文件 |
| 重定向输出 | > | 将命令行的正常执行输出保存到文件，而不是直接显示在屏幕上 |
| >> | 与>类似，但操作是追加而不是覆盖 |
| 重定向错误 | 2> | 将命令行的执行错误输出信息保存到文件，而不是直接显示在屏幕上 |
| 2>> | 与2>类似，但操作是追加而不是覆盖 |
| 混合重定向 | &> | 相当于>和2>，覆盖到同一个文件 |

**编写及执行脚本**

第一个shell脚本

什么是shell脚本

提前写好可执行语句，能够完成特定任务文件

书序、批量化处理

解释型程序（常见的脚本语言：bash shell python/perl/ruby jsp/php/asp/cgi javascript）

shell版helloworld的诞生

脚本创建“三步走”

1. 新建文本文件
2. 添加可执行的脚本语句（命令行）
3. 添加x执行权限

脚本构成及执行

规范的脚本构成

#!脚本声明（使用哪种解释器）

#注释信息（步骤、思路、用途、变量含义等）

可执行的语句

脚本的执行方式

方法一，作为“命令字”

指定脚本文件的路径，前提是有x权限

方法二，作为“参数”（不要求x权限）

bash 脚本文件路径（开子进程）

sh 脚本文件路径（开子进程）

source 脚本文件路径（不开子进程）

. 脚本文件路径（不开子进程）

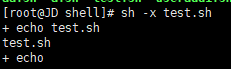
调试shell脚本

主要用途

直接观察执行中的输出、报错信息

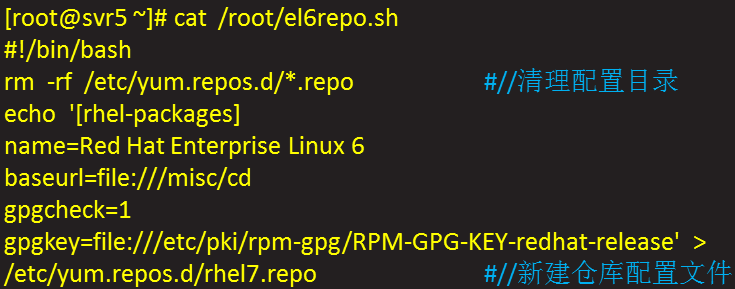
通过sh -x开启调试模式

在可能出错的地方设置echo



简单的脚本应用

快速配置Yum源



快速搭建ftp服务



**shell变量**

变量的设置和取消

什么是变量

以固定名称存放，可能会变化的值

提高脚本对任务需求、运行环境变化的适应能力

方便在脚本中重复使用

定义/赋值/查看变量

定义/赋值变量

变量名=变量值



注意事项：1.若指定的变量名已存在，相当于为此变量重新赋值

1. 等号两边不要有空格
2. 变量名由字母/数字/下划线组成，区分大小写
3. 变量名不能以数字开头，不要使用关键字和特殊字符

查看变量

引用变量值：$变量名

查看变量值：echo $变量名、echo ${变量名}

取消变量

变量的失效

退出定义变量的shell环境时，变量会自动失效

也可以手动取消：unset 变量名

变量的种类

shell变量的分类角度

存储类型

整数型、浮点型、双精度浮点型、字符型、......

shell脚本语言对存储类型要求较松散

使用类型

|  |  |
| --- | --- |
| 类型 | 说明 |
| 环境变量 | 变量名通常都大写，由系统维护，用来设置工作环境，只有个别变量用户可以直接更改 |
| 位置变量 | bash内置，存储执行脚本时提供的参数 |
| 预定义变量 | bash内置，一类有特殊用途的变量，可以直接调用，但不能直接赋值或修改 |
| 自定义变量 | 由用户自主设置、修改及使用 |

环境变量

配置文件：/etc/profile、~/.bash\_profile

相关操作：env：列出所有的环境变量

set：列出所有变量

常见的环境变量：PWD、PATH、USER、LOGNAME、UID

SHELL、HOME、PS1、PS2...

预定义变量

用来保存脚本程序的执行信息

直接使用这些变量

不能直接为这些变量赋值

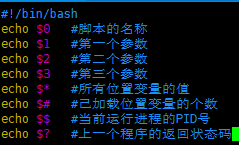
|  |  |
| --- | --- |
| 变量名 | 含义 |
| $0 | 当前所在的进程或脚本名 |
| $$ | 当前运行进程的PID号 |
| $? | 命令执行后的返回状态，0表示正常，1或其他值表示异常 |
| $# | 已加载的位置变量的个数 |
| $\* | 所有位置变量的值 |

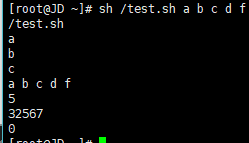
位置变量

在执行脚本是提供的命令行参数

表示为$n，n为序号

$1、$2、....${10}、${11}





变量值及范围控制

扩展赋值操作

区分三种定界符

双引号””：允许特殊符号的功能，以$引用其他变量

单引号’’：禁用特殊符号的功能，即便$也视为普通字符

反撇号``：将命令的执行输出作为变量值（参数，反引号取命令结果）

read标准输入取值

read从键盘读入变量值完成赋值

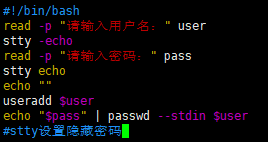
格式：read [-p “提示信息”] 变量名

-p可选，-t可指定超过秒数

终端显示控制

stty -echo：关闭终端输出（无显示）

stty echo：恢复终端输出（显示）



变量的作用范围

局部变量

新定义的变量默认只在当前shell环境中有效

无法在子shell环境中使用

全局变量

全局变量在当前shell及子shell环境中均有效

使用export可将局部变量声明为全局变量

export 局部变量名[=变量值]....：为局部变量添加全局属性

export -n 全局变量名....：取消指定的全局属性