shell脚本学习指南

CH2 入门

shell语言特性：

- 简单性 shell是一个高级语言。
- 可移植性
- 开发容易

当shell执行一个程序时，会要求UNIX内核启动一个新的进程，以便在该进程里执行所指定的程序。

当一个文件中开头的两个字符是#！时，内核会扫描该行其余的部分，内核会扫描该行的其余的部分，看是否存在可用来执行程序的解释器的完整路径。

shell的几个初级的陷阱

- 当今的系统，对#！这一行的长度限制从63到1024个字符都有，请尽量不要超过64个字符。
- 在某些系统上，命令行部分包含了命令的完整路径名称，不过有些系统却不是这样，命令行的部分会原封不动的传给程序，因此，脚本是否具可移植性取决于是否有完整的路径名称。
- 别在选项之后放置任何空白，因为空白也会跟着选项一起传递给被引用的程序。
- 需要知道解释器的完整路径名称，这可以用来规避可移植性问题，因为不同的厂商可能将同样的东西放在不同的地方。
- 一些较旧的系统上，内核不具备解释#！的能力，有些shell会自行处理，这些shell对于#！与紧随其后的解释器名称之间是否可以有空白，可能有不同的解释。

shell识别三种基本命令：内建命令，shell函数以及外部命令：

- 内建命令是由shell本身所执行的命令。
- shell函数是功能健全的一些列程序代码，以shell语言写成，可以像命令那样引用。
- 外部命令是有shell的副本（新的进程）所执行的命令

> 1. 建立一个新的进程，此进程即为shell的一个副本。
> 2. 在新的进程里，在PATH变量内所列出的目录中，寻找特定的命令。
> 3. 在新的进程里，以所找到的新程序来取代执行中的shell程序并执行。
> 4. 程序完成后，最初的shell会接着从终端读取下一条命令，或执行脚本里的下一条命令。

echo的转义序列

序列	说明
\a	警示字符，，通常是ASCII的BEL字符
\b	退格
\c	输出中忽略最后的换行符，这个参数之后的任何字符，包括接下来的参数，都会被忽略掉
\f	清楚屏幕
\n	换行
\r	回车
\t	水平制表符
\v	垂直制表符
\	反斜杠字符
\0ddd	将字符表示成1到3位的八进制数值

printf

printf format-string [arguments...]

- 第一部分是一个字符串，用来描述输出的排列方式，最好为此字符串加上引号。
- 第二部分是与格式声明相对应的参数列表。（%s %d）

I/O重定向

- 以<改变标注输入
- 以>改变标准输出
- 以>>附加到文件
- 以|建立管道

特殊文件：/dev/null 与 /dev/tty

- /dev/null 是位桶，传送到此文件的数据都会被系统丢掉。
- /dev/tty 当程序打开此文件时，UNIX会自动将它重定向到一个终端或串行端口，也可能是一个通过网络与窗口登陆的伪终端再与程序结合。