环境

在 Shell 会话调用环境期间，shell 会存储大量的信息。程序使用存储在环境中的数据来确定配置。

Shell 在环境中存储了两种基本类型的数据：环境变量和 shell 变量。shell 变量是由 bash 存放的少量数据。环境变量就是除此之外的所有其它变量。

1. 检查环境

   printenv 检查出所有的环境信息

   printenv USER 检查出 USER 变量的值

   set 显示所有的 shell 变量，环境变量以及自定义 shell 函数。set 输出的结果按字母排序

   echo $HOME 显示 HOME 变量的值

   alias 显示别名，set 和 printenv 都不能用于显示别名

常见的环境变量

环境变量	说明
DISPLAY	运行图形界面环境时界面的名称。
EDITOR	用于文本编辑的程序名称
SHELL	本机的 Shell 名称
HOME	本机主目录的路径名
LANG	本机语言的字符集和排序规则
OLD_PWD	先前的工作目录
PAGER	用于分页输出的程序名称，通常设置为/usr/bin/less
PATH	以冒号分割的目录列表，当用户输入一个可执行程序名称时，会查找该目录列表
PS1	提示符字符串
PWD	当前工作目录
TERM	终端类型的名称
TZ	本机所处时区
USER	用户名

2. 环境变量是如何建立的？

   用户登录系统后，bash 程序会启动并读取一系列称为启动文件的配置副本。这些脚本定义了所有用户共享的默认配置。接下来，bash 会读取更多存储在目录下的用于定义个人环境的启动文件。

   Shell 会话存在两种类型： login shell 和 non-login shell 会话。

   login shell 会话提示用户名和密码，login shell 会读取一个或多个启动文件：

文件	说明
/etc/profile	适用于所有用户的全局配置脚本
~/.bash_profile	用户个人的启动文件，可以扩展或重写全局配置脚本中的设置
~/.bash_login	若~/.bash_profile 缺失，则 bash 尝试读取此脚本

~/.profile	若前两个均缺失，bash 尝试读取此文件。在 Ubuntu 中，这是默认值

non-loginshell 的启动文件

文件	内容
/etc/bash.bashrc	适用于所有用户的全局配置版本
~/.bashrc	用户的个人启动文件，可扩展或重写全局配置脚本中的设置

3. 激活修改

   因为只有在 shell 启动时才会重新读取.bashrc。所以对上面的文件进行修改后只有在关闭 shell 终端并重启的时候才会生效。但是我们可以使用 source 命令强制 bash 重新读取.bashrc 文件

   ```
   source .bashrc
   ```

4. 注释

   上面文件中，使用#表示注释

5. 提示符的分解

   系统默认的提示符中包含了用户名，主机名和当前的工作目录。提示符是由名为 PS1 的环境变量定义的，echo 命令可以查看到 PS1 的值，如 echo $PS1

shell 提示符中使用的转义字符

转义字符	含义
\a	ASCII 响铃符号，遇到此符号，计算机发出哔哔声
\d	当前日期，以星期，月，日的形式表示，如 Mon May 17
\h	本地机器的主机名，不带域名
\H	完整的主机名
\j	当前 shell 会话进行中的任务个数
\l	当前终端设备的名称
\n	换行符
\r	回车符
\s	shell 程序的名称
\t	当前时间（24 小时制），格式为时:分:秒
\T	当前时间（12 小时制）
\@	当前时间（12 小时制），格式为 AM/PM
\A	当前时间（24 小时制），格式为时:分
\u	当前用户的用户名
\v	shell 的版本号
\V	shell 的版本号和发行号
\w	当前工作目录
\W	当前工作目录的最后一部分

\!	当前命令的历史编号
\#	当前 shell 会话中输入的命令数
\$	非管理员输出$，管理员输出#
\[	标志一个或多个非打印字符序列的开始，用于嵌入非打印的控制字符 比如移动光标或更改文本颜色
\]	非打印字符的结束

6. 设计提示符

```
ps1_old="$PS1"          #保存现有提示符到 ps1_old 变量中
PS1="\A \h \$ "         #将提示符变为时间，主机名
PS1="$ps1_old"          #恢复原始 PS1 提示符
```

7. 添加颜色

字符颜色是由发送到终端仿真器的 ASCII 转义代码控制的，该转义代码嵌入要现实的字符流中。该转义代码由 8 进制代码 033 开始，后面根一个可选的字符属性，之后是一条指令。如将文本颜色设置为正常(属性=0)，黑色的代码是\033[0;30m

字符序列	文本颜色
\033[0;30m	属性 0 表示正常，黑色
\033[0;32m	属性 0 表示正常，绿色
\033[1;30m	属性 1 表示粗体，加粗的黑色
\033[1;32m	属性 1 表示粗体加粗的绿色

```
PS1="\[\033[0;32m\]<\u@\h \W>\$ "#将提示符变为绿色
```

但是此时用户的输入也变成了绿色，要修复这个问题，只要在提示符的末尾插入恢复到原来的颜色即可。

```
PS1="\[\033[0;32m\]<\u@\h \W>\$\[\033[0m\]"
```

设置背景颜色的转义序列

背景颜色不支持粗体属性

字符序列	背景颜色
\033[0:40m	黑色
\033[0:41m	红色
\033[0:42m	绿色
\033[0:47m	淡灰色

8. 移动光标

转义代码也可以用来定位光标。比如在提示符出现的时候，这些转义代码通常用来在屏幕的不同位置显示一个时钟或者其他信息。

光标移动转义序列

转义符	动作
\033[l;cH	将光标移动到 l 行 c 列
\033[nA	将光标向上移动 n 行

\033[nB	光标向下移动 n 行
\033[nC	光标向前移动 n 个字符
\033[nD	光标向后移动 n 个字符
\033[2J	清空屏幕并将光标移动到左上角（0 行 0 列）
\033[K	清空当前光标位置到行末的内容
\033[s	存储当前光标位置
\033[	恢复之前光标位置

```
PS1="\[\033[s\033[0;0H\033[0;41m\033[K\033[1;33m\t\033[0m\033[u\]<\u@\h \W>\$ "
```

9. 永久保存提示符

现在对提示符的修改在再次登录后就会失效，我们可以将其存储在某个位置。如存放在.bashrc 中，在文件末尾可以加入上面的

```
PS1=…
export PS1    #export 命令声明一个变量让其生效。
```