Tree: 458626a814 hexo_blog / source / _posts / Bash文档4--shell内置命令-译.md

Find file Copy path

lifayi2008 post modified 8a88664 on Sep 6, 2016

1 contributor

585 lines (445 sloc) | 49.1 KB

Raw Blame History

title	date	tags	categories
Bash文档4--shell内置命令(译)	2016-06-15 04:57:10 -0700	bash	Bash

内置命令是shell自己实现的命令。当内置命令作为简单命令的第一个word时，shell直接执行这个命令，不需要调用其他的程序。内置命令一般用来实现那些不可能或者不便于使用独立的程序实现的功能

本节简单描述那些从bourne shell继承的bash内置命令，和bash添加以及扩展的内置命令

几个内置命令会在其他的章节描述：用来提供作业控制接口的内置命令、目录栈、命令历史和the programmable completion facilities

很多的内置命令已经被POSIX和Bash扩展

除非特别指出，这些内置命令都接受以`-`开始的选项并且以`--`表示选项的结束。`:` `true` `false` `test`内置命令不接受任何参数同时也不会对`--`特殊对待。`exit`、`logout`、`break`、`continue`、`let`和`shift`命令接收`-`开始的参数，但是不需要`--`来表示选项的结束。其他的内置命令接收但是不限于开始提到的参数形式

Bourne Shell内置命令

下面的这些内置命令是从sh继承的。这些命令按照POSIX的标准实现：

`: [arguments]`
除了扩展参数以及进行重定向操作之外不做任何事情，返回值为0

`. filename [arguments]`
在当前的shell上下文中读取并执行filename脚本的内容。如果filename不包含一个斜线，则使用PATH来查找文件名。如果bash未运行在POSIX模式，如果再PATH中未找到filename则会在当前目录中查找。如果提供了arguments则成为filename执行时的位置参数。否则的话当前shell的位置参数也是filename的位置参数。filename执行的返回值是最后一个执行命令的返回值，如果没有执行命令则返回0。如果filename未找到或者不可读则返回非0值。这个内置命令等同于soruce

`break [n]`
从for while until 或者 select循环中退出。如果提供n参数，则第n层的循环被跳出。n必须大于或者等于1。只有在n小于1的情况下break才会返回非0值

`continue [n]`
在for while until或者select循环中开始下一次循环。如果提供了n参数则表示开始下n次循环。n必须大于或者等于1。只有在n小于1的情况下continue才会返回非0值

`exit [n]`
退出shell，返回n给当前shell的父进程。如果n为空则表示上一个执行的命令的返回值。任何关于EXIT的trap在shell终止前被执行

`return [n]`
停止shell函数的执行并且返回n到调用脚本。如果未提供n，则返回值是函数中上一个命令的退出状态。return也可以用来终止使用.（source）内置命令执行的脚本，可以返回n或者脚本中的上一个命令的退出状态。如果提供n则返回值是n的低8位值。任何和RETURN关联的trap都会在返回到调用脚本之前执行。如果n不为一个数字或者将return使用在其他的地方则return返回非0值

`cd [-L|[-P [-e]] [-@] [directory]`
改变当前工作目录到directory。如果directory未指定，则使用`$HOME`的值。directory后面的参数被忽略。如果shell变量CDPATH存在，则使用变量值作为搜索路径：CDPATH变量中的每一个路径名都作为搜索路径，with alternative directory names in CDPATH separated by a colon `:`。如果目录名以一个斜线开始，则不使用CDPATH

`-P`选项表示不跟随符合连接：symbolic links are resolved while cd is traversing directory and before processing an instance of `..` in directory. 默认情况或者指定了-L选项，symbolic links in directory are resolved after cd processes an instance of `..` in directory. 如果directory中使用了`..`， it is processed by removing the immediately preceding pathname component, back to a slash or the beginning of directory. 如果指定`-P`选项的同时指定了`-e`选项并且在目录成功改变之后不能确定当前工作目录，则`cd`会返回一个未成功的状态 在支持的系统上，`-@`选项表示和文件关联的扩展属性 如果directory指定为`-`则在目录改变之前会被更改为`$OLDPWD` 如果使用了CDPATH中的一个非空的目录，或者`-`是第一个参数，并且也成功的改变了目录，则新的工作目录的绝对路径会被打印到标准输出 如果目录改变成功则返回0，否则返回非0

`eval [arguments]`
所有的参数组合在一起成为一个新的命令，并且被执行，*命令的退出状态作为eval的退出状态*。如果没有参数或者有空参数则eval返回0

`exec [-cl] [-a name] [command [arguments]]`
执行command命令而代替当前的shell。如果使用-l选项，shell会在传给command的第0个参数前加一个`-`。login程序就是这样做的。`-c`选项在执行command之前会清空环境。如果使用`-a`选项，则shell将name作为command的第0个参数。如果comand因为某些原因不能执行，则一个非交互式的shell退出，除非使用了execfail shell选项。这种情况下shell返回错误。An interactive shell returns failure if the file cannot be executed。如果未指定command，则可以指定重定向来影响当前shell。如果重定向没有发生错误，则退出状态为0；否则返回一个非0的退出状态

`export [-fn] [-p] [name[=value]]`
将环境中的name标记为可导入到子进程中。如果指定`-f`选项，则name表示一个函数；否则的话name表示一个shell变量。-n参数表示name不再标记被导入。如果没有提供name或者使用`-p`选项，则显示所有已导出的变量。`-p`选项将输出组织为可以重新作为输入的形式。如果name后面跟着`=value`，则将name的值设置为value

命令的返回值为0除非提供了非法的参数，或者name是一个非法的变量名，或者使用`-f`选项但是未提供一个函数名

`getopts optstringname [args]`
getopts可以被shell用来分析位置参数。optstring中包含要被识别的选项字符；如果字符后面跟着一个`:`，则表示这个选项期望一个参数值，参数值和选项之间使用空格隔开。冒号`:`和问号`?`不能用作选项字符。每次调用getopts时，都会将下一个选项字符赋值给name，如果name不存在则直接创建这个变量，并且下一个被处理的参数的索引作为OPTIND的值。每次shell或者shell script调用时OPTIND的值被初始化为1。如果选项字符期待有一个参数则getopts在处理是会将这个值赋给OPTANG变量。shell不会自动重置OPTIND；如果在同一个shell使用多组位置参数时，那么在多次调用getopts之间必须手动重置OPTIND变量

如果选项处理完毕，则getopts会退出并且返回一个大于0的数。OPTIND被设置为第一个非选项的参数的索引，并且name设置为`?`

getopts通常用来分析位置参数，但是如果提供了args参数，则getopts会分析args参数而不是位置参数

getopts使用两种方式报告错误。如果optstring参数的第一个字符是`:`，则使用安静的错误报告方式。正常情况下如果指定非法选项或者丢失选项的参数则打印诊断消息。如果变量OPTERR被设置为0，则不打印错误消息，即使optstrings的第一个字符是`:`

如果指定了一个非法的选项，getopts会将name置为`?`，如果没有使用安静的错误报告方式，则打印一个错误消息并且释放OPTARG变量。如果设置为静默方式，则将非法的选项字符被赋给OPTARG变量，也不打印错误消息

如果期望有参数的选项没有获取到参数，并且getopts非静默模式，将name置为`?`，OPTARG被释放，然后打印错误消息。如果使用静默模式则将name置为`:`，并且将OPTARG设置为这个选项字符

`hash [-r] [-p filename] [-dt] [name]`
每次调用hash命令，bash会记住作为参数的name命令的路径，所以下次调用这个命令时不必在目录树中搜索。hash从`$PATH`中搜索查找name命令的完整路径。任何之前记住的路径都会被丢弃。`-p`选项表示使用filename作为name的路径，而不从`$PATH`中搜索。`-r`选项表示让hash忘记之前记住的所有路径。`-d`选项表示忘记之前记住的name的路径。如果使用`-t`选项则name对应的每一个路径都会被打印。如果使用多个name作为参数，并且使用`-t`选择则name先打印然后再打印记住的路径名。如果没有提供选项，或者使用-l选项则会打印所有经过hash的命令和路径名，区别是后者会已可在此作为输入的格式打印。命令返回0除非name命令未找到或者使用了非法的选项

`pwd [-LP]`
打印当前工作的目录的绝对路径。如果使用`-P`选项则会对符号连接进行解引用。如果使用`-L`选项则当前打印的路径名可能会是符号连接。如果提供了非法选项或者不能决定当前目录时返回非0值

`readonly [-aAf] [-p] [name[=value]] …`
将每一个name标记为只读。后面的操作中，这些name的值不能改变。如果使用`-f`选项则表示name是一个函数。`-a`选择表示name是一个索引数组；`-A`选项表示name是一个关联数组；如果两个都提供了则`-A`优先。如果未提供参数或者使用`-p`选项，所有标记为只读的name被打印，如果未提name而使用`-aAf`选项则表示打印相应的子集；使用`-p`可以使用一个可重用(作为输入)的格式打印。如果name后面跟`=value`则表示将name设置为value。如果提供了非法的选项或者name不是一个变量或者函数则返回非0值

`shift [n]`
将位置参数往左移n个。位置参数从`n+1....$#`被重命名为`$1....$#-n`。原来的参数从`$#`到`$#-n+1`被释放。n必须是一个非负数字并且小于或者等于`$#`。如果n是0或者大于`$#`，则位置参数未改变。如果未提供n则默认是1。如果n不合法的返回非0

`test expr`
`[ expr ]`
计算条件表达式expr并且返回0(真)或者1(假)。每个操作符和操作数必须是分开的参数。表达式是后面章节描述符的条件表达式 。test不接受任何选项，也不接受`--`作为参数的结尾，如果有提供则会被忽略

如果使用`[`的形式则最后一个参数必须是`]`

> 注意这是一个命令而不是一种语句结构

表达式可以使用下面的这些操作符结合使用，下面列表按优先级从高到底排列。表达式的计算按照参数的个数，当有五个或者更多的参数时才会使用操作符的优先级

```
! expr               如果expr为假则返回真
( expr )             用来覆盖正常的优先级
expr1 -a expr2       如果两个表达式都返回真则结果为真
expr1 -o expr2       如果两个表达式有一个为真则返回真
```

内置命令`test`或者`[`会根据参数个数的不同使用不同的运算规则:

- 0 arguments
  表达式为假
- 1 argument
  当且仅当参数不为空时命令返回真

- 2 arguments
  如果第一个参数是`!`，则当且仅当第二个参数为空时返回真；如果第一个参数是后面描述的一元操作符之一，则一元操作符测试为真时返回真。如果第一个参数不是一个合法的一元操作符则返回假
- 3 arguments
  按照后面描述的依次进行测试。如果第二个参数是二元操作符之一，表达式的结果是第一个和第三个操作数二元测试的结果。如果只有三个参数`-a`和`-o`也被认为是二元操作符。如果第一个参数是`!`，则后面两个参数按照两个参数的情况进行计算，然后对后面两个参数的计算结果取反。如果第一个参数是`(`，第三个参数是`)`则按照一个参数时的情况进行计算。其他情况表达式为假
- 4 arguments
  如果第一个参数是"!"，则后面三个参数按照三个参数的情况计算，在将结果取反。否则的话使用上面描述的优先级规则对表达式进行分析计算
- 5 or more arguments
  使用上面描述的优先级规则对表达式进行分析计算

如果在`[`和`test`中使用`<`或者`>`，则按照ascii的顺序进行计算

`times`
打印出shell及其子进程使用的系统和用户时间，返回值为0

`trap [-lp] [arg] [sigspec …]`
当shell收到sigspec信号是arg会被执行。如果未指定arg(只有参数sigspec)或者使用`-`，则指定的信号处置改为shell被调用时的值。如果arg为空，则当shell或者shell调用的命令执行时sigspec指定的信号被忽略。如果arg未指定并且使用了`-p`选项则shell打印每个sigspec关联的trap。如果没有提供任何参数或者使用了`-p`选项则打印所有设置了trap的信号，并且以可重新作为输入的形式。`-l`选项打印信号名和对应的号码。sigspec可以使用信号名也可以使用对应的号码。信号名大小写无关并且SIG可以省略

如果sigspec是0或者EXIT，则arg在shell退出时执行。如果sigspec是DEBUG，则命令arg在每一个简单命令、for命令、case命令、select命令、每一个算术for命令和shell函数中的第一个命令执行之前之前执行。详细的DEBUG trap

和shell关系的信息可以参考内置命令shopt命令的extdebug选项的描述。如果sigspec是RETURN，则arg在每个函数或者使用`.`(source)内置命令执行的脚步完成时执行

如果sigspec是ERR，则如果一个管道(包括简单命令)，列表或者组合命令返回非0的退出状态时执行arg，但是要受下面条件的制约。如果返回错误的命令是命令列表的一部分并且紧跟着`until`或者`while`关键字，或者是`if`或者`elif`保留字后面的测试命令的一部分，或者是`||` `&&`命令列表的非最后一个命令，或者任何管道线命令的非最后一个命令，或者在命令前面加了`!`来执行，上面几种情况在命令返回错误时都不会执行arg。These are the same conditions obeyed by the errexit (-e) option.

Signals ignored upon entry to the shell cannot be trapped or reset. Trapped signals that are not being ignored are reset to their original values in a subshell or subshell environment when one is created.

如果sigspec未指定一个合法的信号则返回非0值

`umask [-p] [-S] [mode]`
将shell进程的文件创建掩码改为mode。如果mode以数字开始则被解释为8进制数字；如果非数字开始则被解释为和chmod参数类似的符号权限标示。如果mode为空则打印当前的掩码。如果只指定了`-S`选项则打印符号权限标示。如果使用`-p`选项mode为空，则使用可以重新作为输入的格式打印当前的掩码。如果掩码成功更改或者使用正确的选项返回0，其他情况返回非0

注意如果模式为8进制数字的话，则创建文件的权限会使用7减这个数字，所以掩码022表示实际创建文件的权限是755

`unset [-fnv] [name]`
移除变量或者函数name。如果使用`-v`选项，name引用的变量和name都被移除。如果使用`-f`选项则name表示一个函数，并且函数定义被移除。如果使用`-n`选项则name是一个有nameref属性的变量，name被取消，而name引用的变量则不变。如果使用了`-f`选项则`-n`选项无效。如果没有使用选项则name指代一个shell变量；如果不存在这个变量则name指代一个函数名。只读的变量和函数不能被取消

Bash内置命令

本节描述bash扩展并且仅有的命令。其中的一部分是属于POSIX标准

`alias [-p] [name[=value] …]`
没有给出参数或者使用-p选项，则alias打印已创建的别名列表，打印的信息可以重新作为命令的输入。如果给出参数则定义一个将一个name定义为给出的value的别名。如果没有给出value值则打印name对应的别名值。alias详细的描述在后面的章节

`bind`

```
bind [-m keymap] [-lpsvPSVX]
bind [-m keymap] [-q function] [-u function] [-r keyseq]
bind [-m keymap] -f filename
bind [-m keymap] -x keyseq:shell-command
bind [-m keymap] keyseq:function-name
bind readline-command
```

`builtin [shell-builtin [args]]`
使用给出的参数arg运行一个shell内置命令，并且返回。这在定义一个和内置命令同名的函数时有用，因为函数的调用优先于内置命令，所以使用builtin可以调用这个builtin命令而不是同名的函数。如果shell-builting不是一个内置命令则返回非0值

`caller [expr]`
返回任何活动子例程的内容，子例程可以是函数或者使用.（soruce）运行的脚本 Without expr, caller displays the line number and source filename of the current subroutine call. If a non-negative integer is supplied as expr, caller displays the line number, subroutine name, and source file corresponding to that position in the current execution call stack. This extra information may be used, for example, to print a stack trace. The current frame is frame 0. The return value is 0 unless the shell is not executing a subroutine call or expr does not correspond to a valid position in the call stack.

`command [-pVv] command [arguments …]`
使用arguments运行command命令而不管有函数和command同名。*只有内置命令和PATH路径中的命令会被执行*。如果有一个shell函数ls，而在这个函数中运行command ls时会执行外部命令ls而不是递归的调用函数ls。The `-p` option means to use a default value for PATH that is guaranteed to find all of the standard utilities. 如果command不存在或者发生错误返回127，否则的话返回command执行的返回值

如果使用 `-V` 或者 `-v` 选项则会打印command的描述性信息，类似于type命令。如果使用 `-v` 选项则会打印出调用这个命令时实际执行的内容(别名或者可执行文件)；如果使用 `-V` 选项则会打印稍详细点的信息，类似于不带选项的type命令

`declare [-aAfFgilnrtux] [-p] [name[=value] …]`
定义变量并且给他们分配属性。如果未提供name参数，则会打印出给定选项类型的变量和值

`-p` 选项会显示name对应的属性和值。如果使用 `-p` 选项时提供了name则除了 `-f` 和 `-F` 选项其他的选项都被忽略

如果使用-p选项又没有提供name参数，则会显示其他选项对应的所有变量和值以及属性。如果只指定了 `-p` 选项，则会显示所有shell变量的属性和值。 `-f` 选项可以限制只显示函数

`-F` 选择只显示函数名而不显示函数内容和属性。如果启用extdebug选项，则函数定义的文件名和行号也被显示。 `-F` 隐含 `-f`

`-g` 选项可以在全局作用域创建和更改变量，即使在函数中使用declare。其他情况下则会忽略这个选项

下面这些选项可以限制只有特定属性相关的变量被显示或者给变量分配特定的属性:

```
-a    表示name是一个索引数组
-A    表示name是一个关联数组
-f    表示name是一个函数
-i    表示name变量是一个整形的；给变量赋值时会对右值进行算术运算
-l    当给变量赋值时所有的大写字符会被转换为小写。大写属性被禁用
-n    给name分配nameref属性，使name引用另外一个变量。表示这个变量的值是另外一个变量名。所有对变量的引用和赋值，除非
```

```
-r    表示name是一个只读的变量，后续不能再重新赋值和取消
-t    给name分配trace属性。被trace的函数从调用的shell中继承DEBUG和RETURN traps。trace对变量无效
-u    和`-l`相反
-x    通过环境变量将name导入到子进程中
```

使用`+`代替`-`可以关闭这些属性，例外情况是`+a`不会销毁一个数组，而`+r`也不会取消一个变量的只读属性。如果在一个函数中使用declare则和local一样将name标记为局部的，除非使用`-g`选项。如果name后面跟`=value`则变量被赋值

When using `-a` or `-A` and the compound assignment syntax to create array variables, additional attributes do not take effect until subsequent assignments.

使用了不合理的参数或者变量名或者变量值、或者没有使用组合赋值语法给数组分配值则会返回非0值

`echo [-neE] [arg …]`
输出arg，如果有多个arg则输出结果使用空格分隔，并且会在最后添加一个换行。最后返回值为0除非写时发生错误。如果使用-n参数则不会在结尾添加一个换行。如果使用-e参数，则会解释下面这些反斜线转义的字符。-E选项则关闭这些转义字符，即使在那些默认会解释的系统上面。xpg_echo可以用来动态的控制echo命令默认是否解释这些转义字符。echo不会将--作为选项的结尾

echo会解释下面这些转义字符:

```
\a      alert (bell)
\b      backspace
\c      suppress further output
\e
\E      escape
\f      form feed
\n      new line
\r      carriage return
\t      horizontal tab
```

```
  \v      vertical tab
  \\      backslash
  \0nnn   the eight-bit character whose value is the octal value nnn (zero to three octal digits)
  \xHH    the eight-bit character whose value is the hexadecimal value HH (one or two hex digits)
  \uHHHH  the Unicode (ISO/IEC 10646) character whose value is the hexadecimal value HHHH (one to fou
  \UHHHHHHHH  the Unicode (ISO/IEC 10646) character whose value is the hexadecimal value HHHHHHHH (on
```

`enable [-a] [-dnps] [-f filename] [name …]`
启用或者禁用内置命令。禁用一个内置命令可以让和内置命令同名的命令在执行时不必使用完全路径，即使shell会优先扫描内置命令。如果使用`-n`参数则name命令被禁用。否则的话name启用。比如如果想要在$PATH中的一个test命令代替内置的test命令可以使用`enable -n test`

如果使用`-p`选项，或者没有提供任何选项，则会打印所有启用的内置命令。而`-a`选项则打印所有启用和禁用的内置命令并且在命令前面加一个enable或者disable

`-f`选项表示如果系统支持动态导入，则从共享对象filename中导入新的内置命令。`-d`命令则删除由`-f`导入的命令

如果未提供参数则打印内置命令的列表。`-s`选项可以控制只显示POSIX的内置命令。如果提供`-f`选项的同时使用`-s`选项则这些命令成为特殊内置命令

如果提供的name不是一个内置命令或者导入命令时出错则返回非0值

`help [-dms] [pattern]`
显示内置命令的帮助信息。如果使用pattern则shell显示匹配pattern的所有命令的帮助信息。如果没有提供参数的话，显示所有内置命令和一个简单的帮助信息。

```
-d    表示显示一个简短的帮助信息
-m    表示显示匹配patthern命令的类似于manpage的帮助信息
-s    表示显示一个简短的usage帮助信息
```

返回值为0除非没有匹配的命令

`let expression [expression …]`
let内置命令可以让shell变量进行数学计算。每个expression都按照arithmetic expansion的规则进行算术运算。如果最后一个expression计算结果为0则返回1；否则的话返回0

`local [option] name[=value] …`
用来创建本地变量，如果有=value部分则同时进行赋值，name可以有多个使用空格分隔。option可以为declare的所有选项。local只能在一个函数内部使用；可以用来限制变量只在本函数（以及本函数调用的函数）中使用。如果在函数外部使用，或者使用非法的name，或者name是一个只读变量则返回非0值

`logout [n]`
退出登录shell，并且将n返回给父shell

`mapfile [-n count] [-O origin] [-s count] [-t] [-u fd] [-C callback] [-c quantum] [array]`
从标准输入读取行，并且赋给索引数组（ *每行作为一个元素* ），如果使用`-u`选项则表示从文件描述符fd获取输入。变量MAPFILE是默认的数组名。可以使用下面的这些选项:

```
-n    表示最多接收n行，如果n为0则表示全部接收
-O    从索引origin开始分配值，默认索引为0
-s    丢弃接收到的第一行内容
-t    移除接收到的行的结尾的换行符
-u    从文件描述符fd接收行而不是标准输入
-C    每次接收到quantum行都调用callback，quantum使用-c参数指定
-c    指定每一次调用callback间隔的读取到的行数
```

如果只使用`-C`选项指定callback，而没有使用`-c`则默认是5000。When callback is evaluated, it is supplied the index of the next array element to be assigned and the line to be assigned to that element as additional arguments. callback is evaluated after the line is read but before the array element is assigned.

如果未提供明确的origin，mapfile在赋值前会清除数组（可以将origin设置为上次调用mapfile的行数，将数组名设置为和上次调用相同来将两次调用赋给同一个数组）

除非是用了错误的选项，数组非法或者只读，或者不是一个索引数组返回非0，否则返回0

`printf [-v var] format [arguments]`
使用format控制的格式将argument打印至标准输出。*使用-v参数可以让结果被赋值给var变量*，而不是打印到标准输出

format是包含三种类型对象的一系列字符串：普通文本字符被简单打印输出；转义字符序列会被转换然后打印到标准输出；格式指定字符，每个会被后面对应的arguments替换。除了和C语言`printf`相同的控制字符之外，shell中的`printf`还会解释下面这些扩展字符:

- ```
  `%b`     转义字符占位符，表示后面对应的参数是一个转义字符，例外是`\c`不会用来结束一行，而是`\'` `\"` `\?`的反
  ```
- ```
  `%q`  Causes printf to output the corresponding argument in a format that can be reused as shel
  ```
- ```
  `%(datefmt)T`        可以让`printf`打印日期时间字符串，datefmt是strftime库函数可识别的字符格式。对应的arg
  ```

Arguments to non-string format specifiers are treated as C language constants, except that a leading plus or minus sign is allowed, and if the leading character is a single or double quote, the value is the ASCII value of the following character.
  ```

如果给出的参数比占位符多，那么占位符和参数不再是一一对应，format会循环消耗参数。如果占位符比参数多，则后面的占位符期望的参数根据情况默认是0或者空。打印成功则返回0，失败则返回非0值

`read [-ers] [-a aname] [-d delim] [-i text] [-n nchars] [-N nchars] [-p prompt] [-t timeout] [-u fd] [name …]`
从标准输入读取一行，如果使用-u参数则表示从文件描述符fd读取，第一个word被赋给第一个name，第二个word赋值给第二个name以此类推，word数多于name则会把后面的word都赋给最后一个name。如果从输入读取到的word少于name则其他的name会被赋空值。IFS变量的值用来将行分隔为word，分隔规则和shell expansion相同。反斜线可以用来转义接下来读到的一个字符或者换行符。如果没有提供name参数，则读到的行被赋值给REPLY变量。除非读到了EOF，或者超时(这时返回值大于128)，或者变量赋值错误，或者是一个非法的文件描述符，否则返回值是0

其他选项的解释如下:

```
-a aname    分隔后的word被赋值给索引数组aname，从索引从0开始。aname中的所有元素在赋值前被移除。如果有name参数被忽
-d delim    使用delim的第一个字符来终止输入的一行内容，而不是换行符
-e          Readline (see Command Line Editing) is used to obtain the line. Readline uses the curre
-i text     如果Readline被用来读取行，会在开始编辑之前将test放入编辑缓存
-n nchars   read读取nchars个字符就停止，如果再读取nchars个字符之前遇到delimter则也作为读取到的一行
-p promp    开始读取输入之前先打印protmpt的内容，尾部不会自动添加换行。只有输入来自于终端时才会打印prompt
-r          如果使用本选项，则反斜线不会被作为转义字符。反斜线作为一个普通字符。当反斜线后面跟换行时被当做续行符
-s          静默模式，如果从终端读取输入，字符不会被回显
-t timeout  如果再timeout秒内没有获取到一个完整的行(或者特定数目的字符)则read会超时并返回错误。timeout可以指定为
-u fd       从文件描述符fd获取输入
```

`readarray [-n count] [-O origin] [-s count] [-t] [-u fd] [-C callback] [-c quantum] [array]`
和mapfile同义

`source filename`
和`.`同义

type [-afptP] [name …]

对每一个name打印出：如果name是一个命令则它会如何被解释。

```
-t    如果使用-t选项，type只打印一个单词标示name是什么类型的命令“alias”“function”“builtin”“file”“keyword”。
-p   如果给出的name使用type -t选项打印出‘file’时，-p选项会打印出这个‘file’的绝对路径；如果-t选项的输出是其他几种
-P   大写的-P选项会强制对每个name进行路径搜索，即使-t的输出不是‘file’，这当name是别名时有用
如果一个命令已经在bash的hash表里面，则直接返回对应的值；这个命令不必一定在$PATH中
-a    如果使用-a选项同时又没有使用-p选项，则type命令会返回name的所有情况，包括别名和函数
-f    如果使用-f选项则type不会查找shell函数
```

typeset [-afFgrxilnrtux] [-p] [name[=value] …]

typeset命令是为了和kshell兼容的命令，和declare同义

ulimit [-abcdefilmnpqrstuvxHST] [limit]

ulimit用来控制shell启动的进程的可用资源数量，系统需要提供这类支持。选项解释如下：

```
-S    更改或者打印和资源关联的软限制值
-H    更改或者打印和资源管理的硬限制值
-a    打印所有的限制值
-b    最大的socket缓存
-c    创建的coredump的最大大小
-d    进程的最大数据段data segment
-e    进程可调度的最大优先级
-f    shell和子进程可创建的最大文件大小
-i    处于pending状态的信号的最大数量
-l    可以锁进内存的最大大小
-m    最大驻留集大小(很多系统未实现本限制)
-n    最大打开文件数量(很多系统不允许设置这个值)
-p    命名管道缓冲区大小
-q    POSIX消息队列最大字节数
-r    最大的实时调度优先级
```

```
-s     最大的栈大小
-t     最大的CPU时间，单位是秒
-u     单个用户最大进程数
-v     shell最大的挂载虚拟内存数量
-x     文件锁的最大数量
-T     最大线程数量
```

如果指定了limit参数，并且未使用-a参数，指定的资源被设置为新的limit值。limit的特殊值hard，soft和unlimited代表当前的硬限制，当前软限制和无限制。非root用户使用的资源不能超过当前设置的硬限制值；软限制值最大可以设置为硬限制值。如果没有给出limit则表示打印指定的资源限制值，除非使用`-H`选项。当设置值时如果未指定`-H`或者`-S`则两者都被设置。如果没有给出任何选项则默认是`-f`。除了`-t`的增量是秒，其他的都是1024byte；`-p`的单位是512-byte blocks；`-T` `-b` `-n`和`-u`都是非度量值

除非指定非法的选项或者值，或者在设置值时出现错误，返回非0

`unalias [-a] [name … ]`
移除alias列表的别名。如果使用-a参数则表示所有的别名都被移除。详细的alias信息可以参考后面的章节

更改**shell**行为的命令

set命令

使用set命令可以更改shell选项和设置位置参数，也可以显示当前的shell变量

```
set [--abefhkmnptuvxBCEHPT] [-o option-name] [argument …]
set [+abefhkmnptuvxBCEHPT] [+o option-name] [argument …]
```

如果未指定选项和参数，`set`命令显示当前shell所有函数的名字和变量的名称和值，按照服务器地区设置进行排序；并且以可以再作为其他设置变量的命令的输入的格式进行输出。只读变量不能再被设置。在`POSIX`模式下`set`只列出变量名

如果指定选项则表示设置获取取消设置shell属性。下面对选项进行详细解释：

`-a` 将正在创建或者更改的变量或者函数导出到后面命令的执行环境中

`-b` 让后台命令的返回值立即打印到标准输出；默认会等到下一次输出命令提示符的时候才会输出

`-e` 如果一个管道命令（Pipelines，不同于狭义的管道）返回一个非0值则立即退出shell。如果返回错误的命令是在`while`或者`until`后面的命令列表的一部分，或者是`if`语句测试的一部分，或者是`||` `&&`命令列表的一部分并且不是最后一个命令，或者管道（pipeline |）命令的一部分并且不是最后一个，或者命令的返回状态使用`!`取反，则不会退出shell。If a compound command other than a subshell returns a non-zero status because a command failed while -e was being ignored, the shell does not exit. A trap on ERR, if set, is executed before the shell exits

这个选项对当前shell及子shell都有效（即子shell中如果有命令返回非0值则退出子shell返回到当前shell）

`-f` 禁用文件名扩展

`-h` 执行外部命令时记录命令的路径（hash），默认启用

`-k` 命令中的所有赋值语句都作为命令的环境，默认需要放置在命令名之前

`-m` 启用作业控制（job control）。所有进程都在一个单独的进程组中。当后台命令完成时，shell会打印一行内容包含命令的返回值

`-n` 读取命令但是不执行，这样可以用来检查一个脚本的语法是否正确；交互式shell忽略这个选项

`-p` 启用特权模式。在这个模式中`$BASH_ENV`和`$ENV`文件不会被处理，shell函数也不会从环境中继承，环境中的`SHELLOPTS` `BASHOPTS` `CDPATH``GLOBIGNORE`变量被忽略。如果启动shell的有效用户id和真实用户id不同，并且为指定-p选项，则有效用户id被设置为真实用户id。如果使用`-p`选项启动shell则不会重置有效用户id。关闭这个选项则有效用户id和组id被设置为真实用户id和组id

`-t` 读取和执行一条命令就退出

`-u` 使用未设置的变量和参数（`*`和`@`除外）则返回错误。在交互式shell中向标准错误输出打印一条错误信息，非交互式shell中直接退出

`-v` Print shell input lines as they are read

`-x` 打印简单命令、for命令、case命令、select命令、和算术for命令的参数和关联word列表扩展完毕之后执行之前的状态，可以进行命令扩展跟踪。打印的内容使用变量`PS4`的值作为开头

`-B` shell会进行大括号扩展，默认启用

`-C` 阻止输出重定向操作符`>` `>&` `<>`覆盖已经存在的文件

`-E` 如果设置则`ERR上`的trap会被shell函数，命令替换和子shell中的命令执行所继承

`-H` 启用命令历史替换

`-P` 如果设置，在执行cd命令切换当前目录到一个符号链接目录时，直接切换到符号连接指向的实际目录；默认情况下可以切换到符号连接目录（使用`pwd`命令打印可以看出来）

`-T` 如果设置则`DEBUG`和`RETURN`上的trap会被shell函数，命令替换和子shell中的命令执行所继承

`--` 如果这个选项后面未跟任何参数，则表示将位置参数取消设置。否则位置参数被设置为 arguments，即使位置参数使用 - 开始

`-` 标记选项的结束，后面的所有word都被当作是位置参数。The -x and -v options are turned off. If there are no arguments, the positional parameters remain unchanged.

使用**`+`**表示关闭这些选项。当前设置的选项可以使用`$-`查看

剩余的`argument`被当作位置参数依次赋值给`$1 $2 .....$n`，特殊参数`$#`表示参数个数

返回值总是0，除非给出了错误的选项

`-o option-name` option-name可以是：

```
allexport    等同于-a
braceexpand    等同于-B
errexit    等同于-e
errtrace    等同于-E
functrace    等同于-T
hashall    等同于-h
histexpand    等同于-H
keyword    等同于-k
monitor    等同于-m
noclobber    等同于-C
noexec    等同于-n
noglob    等同于-f
nolog    当前这个命令无效
notify    等同于-b
nounset    等同于-u
onecmd    等同于-t
physical    等同于-P
privileged    等同于-p
verbose    等同于-v
xtrace    等同于-x
pipefail    如果设置这个选项则管道命令的退出状态是最后一个非0的退出状态，如果所有命令都返回0则结果也是0。这个选项默
posix    让Bash的行为符合POSIX标准。这样bash的行为就是标准行为的一个严格超集
emacs    使用emac风格的编辑接口。这个选项也会影响read -e命令
vi    使用vi的行编辑接口，同样影响read -e命令
history    启用命令历史（commad history）；在交互式shell中默认启用
ignoreeof    交互式shell在读取到EOF不会退出
```

shopt命令

这个内置命令可以用来更改一些其他的shell行为

```
shopt [-pqsu] [-o] [optname …]
```

如果使用`-o`选项则`optname`必须是`set`命令支持的`option-name`。不加任何选项则在选项后面使用`on`或者`off`表示是否已经开启；`-p`选项则显示出所有可设置的选项（隐含选项是否已经设置），并且输出的格式是可以再次作为命令的输入。`-s`选项表示启用`optname`，`-u`表示禁用`optname`。`-q`表示检查一个选项是否被设置，如果是返回0否则返回1，如果指定多个`optname`则都已经设置时返回0，否则返回1

如果只使用`-u`或者`-s`选项而不指定参数，则分别显示当前未设置或者设置的选项

optname可以是如下：

```
autocd
If set, a command name that is the name of a directory is executed as if it were the argument to th
cdable_vars
If this is set, an argument to the cd builtin command that is not a directory is assumed to be the
cdspell
If set, minor errors in the spelling of a directory component in a cd command will be corrected. Th
checkhash
If this is set, Bash checks that a command found in the hash table exists before trying to execute
checkjobs
If set, Bash lists the status of any stopped and running jobs before exiting an interactive shell.
checkwinsize
If set, Bash checks the window size after each command and, if necessary, updates the values of LIN
cmdhist
If set, Bash attempts to save all lines of a multiple-line command in the same history entry. This
compat31
If set, Bash changes its behavior to that of version 3.1 with respect to quoted arguments to the co
compat32
If set, Bash changes its behavior to that of version 3.2 with respect to locale-specific string com
compat40
If set, Bash changes its behavior to that of version 4.0 with respect to locale-specific string com
compat41
```

```
If set, Bash, when in POSIX mode, treats a single quote in a double-quoted parameter expansion as a
compat42
If set, Bash does not process the replacement string in the pattern substitution word expansion usi
complete_fullquote
If set, Bash quotes all shell metacharacters in filenames and directory names when performing compl
direxpand
If set, Bash replaces directory names with the results of word expansion when performing filename c
dirspell
If set, Bash attempts spelling correction on directory names during word completion if the directo
dotglob
If set, Bash includes filenames beginning with a ‘.’ in the results of filename expansion.
execfail
If this is set, a non-interactive shell will not exit if it cannot execute the file specified as ar
expand_aliases
If set, aliases are expanded as described below under Aliases, Aliases. This option is enabled by c
extdebug
If set, behavior intended for use by debuggers is enabled:
The -F option to the declare builtin (see Bash Builtins) displays the source file name and line nur
If the command run by the DEBUG trap returns a non-zero value, the next command is skipped and not
If the command run by the DEBUG trap returns a value of 2, and the shell is executing in a subrouti
BASH_ARGC and BASH_ARGV are updated as described in their descriptions (see Bash Variables).
Function tracing is enabled: command substitution, shell functions, and subshells invoked with ( co
Error tracing is enabled: command substitution, shell functions, and subshells invoked with ( comma
extglob
If set, the extended pattern matching features described above (see Pattern Matching) are enabled.
extquote
If set, $'string' and $"string" quoting is performed within ${parameter} expansions enclosed in dou
failglob
If set, patterns which fail to match filenames during filename expansion result in an expansion err
force_fignore
If set, the suffixes specified by the FIGNORE shell variable cause words to be ignored when perform
globasciiranges
If set, range expressions used in pattern matching bracket expressions (see Pattern Matching) behav
globstar
If set, the pattern ‘**’ used in a filename expansion context will match all files and zero or more
```

```
gnu_errfmt
If set, shell error messages are written in the standard GNU error message format.
histappend
If set, the history list is appended to the file named by the value of the HISTFILE variable when t
histreedit
If set, and Readline is being used, a user is given the opportunity to re-edit a failed history sub
histverify
If set, and Readline is being used, the results of history substitution are not immediately passed
hostcomplete
If set, and Readline is being used, Bash will attempt to perform hostname completion when a word co
huponexit
If set, Bash will send SIGHUP to all jobs when an interactive login shell exits (see Signals).
interactive_comments
Allow a word beginning with '#' to cause that word and all remaining characters on that line to be
lastpipe
If set, and job control is not active, the shell runs the last command of a pipeline not executed i
lithist
If enabled, and the cmdhist option is enabled, multi-line commands are saved to the history with en
login_shell
The shell sets this option if it is started as a login shell (see Invoking Bash). The value may not
mailwarn
If set, and a file that Bash is checking for mail has been accessed since the last time it was chec
no_empty_cmd_completion
If set, and Readline is being used, Bash will not attempt to search the PATH for possible completio
nocaseglob
If set, Bash matches filenames in a case-insensitive fashion when performing filename expansion.
nocasematch
If set, Bash matches patterns in a case-insensitive fashion when performing matching while executin
nullglob
If set, Bash allows filename patterns which match no files to expand to a null string, rather than
progcomp
If set, the programmable completion facilities (see Programmable Completion) are enabled. This opti
promptvars
If set, prompt strings undergo parameter expansion, command substitution, arithmetic expansion, and
restricted_shell
```

```
The shell sets this option if it is started in restricted mode (see The Restricted Shell). The valu
shift_verbose
If this is set, the shift builtin prints an error message when the shift count exceeds the number o
sourcepath
If set, the source builtin uses the value of PATH to find the directory containing the file supplie
xpg_echo
If set, the echo builtin expands backslash-escape sequences by default.
```

特殊内置命令

因为历史的原因，POSIX标准将一些内置命令归类为特殊内置命令。当Bash以`POSIX`模式运行时，特殊内置命令和其他的命令有以下三方面的不同：

1. 在进行命令搜索时，特殊命令先于shell函数
2. 如果一个特殊内置命令返回一个表示错误的状态码，则非交互式shell退出
3. 命令前的变量赋值语句所创建的变量在命令结束之后仍然有效

当Bash没有在`POSIX`模式运行时，这些内置命令的行为和其他内置命令一样

POSIX特殊内置命令如下：

`break` `:`（冒号） `.`（点） `continue` `eval` `exec` `exit` `export` `readonly` `return` `set` `shift` `trap` `unset`