trap

捕捉信号和其他事件并执行命令。

概要

```
trap [-lp] [[arg] signal_spec ...]
```

主要用途

- 用于指定在接收到信号后将要采取的动作。
- 脚本程序被中断时执行清理工作。

选项

```
-l      打印信号名称以及信号名称对应的数字。
-p      显示与每个信号关联的trap命令。
```

参数

arg：接收到信号时执行的命令。

signal_spec：信号名称或信号名称对应的数字。

返回值

如果表达式执行结果为成功时返回0，当参数 `signal_spec` 没有指定有效值时返回1。

关于信号

信号是一种进程间通信机制，它给应用程序提供一种异步的软件中断，使应用程序有机会接受其他程序活终端发送的命令(即信号)。应用程序收到信号后，有三种处理方式：忽略，默认，或捕捉。进程收到一个信号后，会检查对该信号的处理机制。如果是SIG_IGN：就忽略该信号；如果是SIG_DFT：则会采用系统默认的处理动作，通常是终止进程或忽略该信号；如果给该信号指定了一个处理函数(捕捉)，则会中断当前进程正在执行的任务，转而去执行该信号的处理函数，返回后再继续执行被中断的任务。

在有些情况下，我们不希望自己的shell脚本在运行时刻被中断，比如说我们写得shell脚本设为某一用户的默认shell：使这一用户进入系统后只能作某一项工作，如数据库备份， 我们可不希望用户使用 Ctrl+C 等方法进入到shell状态做我们不希望做的事情。这便用到了信号处理。

以下是一些你可能会遇到的常见信号：

```
<table>
<tbody>
<tr>
<th width="100">信号名称</th>
<th width="60">信号数</th>
<th>描述</th>
</tr>
<tr>
```

```
<td>SIGHUP</td>
<td>1</td>
<td>本信号在用户终端连接（正常或非正常）结束时发出，通常是在终端的控制进程结束时，通知同一session内的各个作业，这时它们与控制终端不再关联。登录Linux时，系统会分配给登录用户一个终端(Session)在这个终端运行的所有程序，包括前台进程组和后台进程组，一般都属于这个Session当用户退出Linux登录时，前台进程组和后台有对终端输出的进程将会收到SIGHUP信号。这个信号的默认操作为终止进程，因此前台进程组和后台有终端输出的进程就会中止。对于与终端脱离关系的守护进程，这个信号用于通知它重新读取配置文件。</td>
</tr>
<tr>
<td>SIGINT</td>
<td>2</td>
<td>程序终止(interrupt)信号，在用户键入 Ctrl+C 时发出。</td>
</tr>
<tr>
<td>SIGQUIT</td>
<td>3</td>
<td>和SIGINT类似，但由QUIT字符(通常是Ctrl /)来控制。进程在因收到SIGQUIT退出时会产生core文件，在这个意义上类似于一个程序错误信号。</td>
</tr>
<tr>
<td>SIGFPE</td>
<td>8</td>
<td>在发生致命的算术运算错误时发出。不仅包括浮点运算错误，还包括溢出及除数为0等其它所有的算术错误。</td>
</tr>
<tr>
<td>SIGKILL</td>
<td>9</td>
<td>用来立即结束程序的运行。本信号不能被阻塞，处理和忽略。</td>
</tr>
<tr>
<td>SIGALRM</td>
<td>14</td>
<td>时钟定时信号，计算的是实际的时间或时钟时间alarm 函数使用该信号。</td>
</tr>
<tr>
<td>SIGTERM</td>
<td>15</td>
<td>程序结束(terminate)信号,与SIGKILL不同的是该信号可以被阻塞和处理. 通常用来要求程序自己正常退出kill 命令缺省产生这个信号。</td>
</tr>
</tbody>
</table>
```

例子

当shell收到 HUP INT PIPE QUIT TERM 这几个命令时，当前执行的程序会执行 exit 1

```
[root@pc root]$ trap "exit 1" HUP INT PIPE QUIT TERM
```

1 清理临时文件

下面展示了如果有人试图从终端中止程序时，如何删除文件然后退出：

```
trap "rm -f $WORKDIR/work1 $WORKDIR/dataout; exit" 2
```

执行shell程序，如果程序接收信号为2，那么这两个文件 （work1 和 dataout） 将被自动删除。

添加信号1 SIGHUP）

```
$ trap "rm $WORKDIR/work1 $WORKDIR/dataout; exit" 1 2
```

2 忽略信号

如果陷阱列出的命令是空的，指定的信号接收时，将被忽略：

```
$ trap '' 2
```

忽略多个信号：

```
$ trap '' 1 2 3 15
```

3 重置陷阱

当你改变了收到信号后采取的动作，你可以省略第一个参数来重置到默认行为。

```
$ trap 1 2
```

注意

1. `trap -l` 等价于执行 `kill -l`）
2. 发送信号请查看 `kill` 命令。
3. 该命令是bash内建命令，相关的帮助信息请查看 `help` 命令。
4. 建议您阅读以下参考资料来深入了解该命令：

- GNU 官方手册） trap命令
- Linux Shell的信号trap功能你必须知道的细节
- 阮一峰） Bash 脚本如何创建临时文件）mktemp 命令和 trap 命令教程
- ）Bash百宝箱）shell内建命令之trap

From:
https://rd.irust.top/ - **学习笔记**

Permanent link:
https://rd.irust.top/doku.php?id=command:trap

Last update: **2021/10/15 14:58**