The caller builtin command

Synopsis

```
caller [FRAMENUMBER]
```

Description

The `caller` builtin command is used to print execution frames of subroutine calls. Without giving a framenumber, the topmost execution frame information is printed ("who called me") wile linenumber and filename.

When an execution frame number is given (0 - topmost), the linenumber, the subroutine (function) and the filename is printed. When an invalid execution frame number is given, it exists `FALSE` . This way it can be used in a loop (see the examples section below).

Examples

Simple stack trace

The code below defines a function `die` that is used to exit the program. It prints a list of execution frames, starting with the topmost frame (0). The topmost frame is the "caller of the die function", in this case function "f1".

This way, you can print a "stack trace" for debugging or logging purposes.

The code is made very simple, just to show the basic purposes.

```
#!/bin/bash

die() {
  local frame=0
  while caller $frame; do
    ((++frame));
  done
  echo "$*"
  exit 1
}

f1() { die "*** an error occured ***"; }
f2() { f1; }
f3() { f2; }

f3
```

Output

```
12 f1 ./callertest.sh
13 f2 ./callertest.sh
14 f3 ./callertest.sh
16 main ./callertest.sh
*** an error occured ***
```

Notes

- `caller` produces no output unless used within a script that's run from a real file. It isn't particularly useful for interactive use, but can be used to create a decent `die` function to track down errors in moderately complex scripts.

  ```
  { bash /dev/stdin; } <<<$'f(){ g; }\ng(){ h; }\nh(){ while calle
  r $((n++)); do :; done; }\nf'
  ```

- For more sophisticated debugging, Bash extended debugging features are available and a number of special parameters that give more detail than caller (e.g. BASH_ARG{C,V}). Tools such as Bashdb (http://bashdb.sourceforge.net/) can assist in using some of Bash's more advanced debug features.
- The Bash manpage and help text specifies that the argument to `caller` is an "expr" (whatever that means). Only an integer is actually allowed, with no special interpretation of an "expression" as far as we can tell.

Portability considerations

- `caller` is not specified by POSIX(R)
- the `caller` builtin command appeared in Bash version 3.0

See also

Discussion

commands/builtin/caller.txt Last modified: 2022/03/20 14:23 by sahirhoda

Bash Hackers Wiki