

Re: Why doesn't foreach return a value

Source: <http://coding.derkeiler.com/Archive/Tcl/comp.lang.tcl/2008-02/msg00022.html>

- *From:* "tom.rmadilo" <tom.rmadilo@xxxxxxxx>
 - *Date:* Fri, 1 Feb 2008 09:34:14 -0800 (PST)
-

On Feb 1, 1:17 am, Fredderic <my-name-h...@xxxxxxxx> wrote:

[return], [continue], or [break], all send a signal back indicating special action needs to be taken. Each level in turn, then either handles that signal or passes it back. This signal, which includes an "all okay, go get the result" variant, is "returned".

Yes, more or less. Please forgive me for any disrespect I have shown anyone in discussing this topic, I'll try one more time to use the correct language.

Every Tcl command is implemented as a C level function. For now, I'm not talking about Tcl procs, just the stuff implemented in C.

Every C level implementation of a Tcl command (except the one for [exit], which doesn't return), returns a value, 'a standard Tcl completion code', a small integer value. 0 = TCL_OK, 1 = TCL_ERROR, 2 = TCL_CONTINUE, 3 = TCL_BREAK 4 = TCL_RETURN. Greater than 4 are user defined.

No C level implementation of a Tcl command returns any other value, but there can be side effects. Some, but not all, of these command implementations set, reset, append to or modify the Tcl interp's result.

Is there any disagreement with this? Because if I got this wrong, my further reasoning can't be right.

If this is correct, then, yes, every Tcl command is the same, and [exit] isn't really a useful counter example.

But what use is there in talking about the internal return value of a command at the script level? What we want at the script level