NAME

Tcl_Panic, Tcl_PanicVA, Tcl_SetPanicProc − report fatal error and abort

SYNOPSIS

#include <tcl.h>

void
Tcl_Panic(*format*, *arg*, *arg*, *...*)

void
Tcl_PanicVA(*format*, *argList*)

void
Tcl_SetPanicProc(*panicProc*)

ARGUMENTS

const char* *format* (in)	A printf-style format string.
arg (in)	Arguments matching the format string.
va_list *argList* (in)	An argument list of arguments matching the format string. Must have been initialized using **va_start**, and cleared using **va_end**.
Tcl_PanicProc **panicProc* (in)	Procedure to report fatal error message and abort.

DESCRIPTION

When the Tcl library detects that its internal data structures are in an inconsistent state, or that its C procedures have been called in a manner inconsistent with their documentation, it calls **Tcl_Panic** to display a message describing the error and abort the process. The *format* argument is a format string describing how to format the remaining arguments *arg* into an error message, according to the same formatting rules used by the **printf** family of functions. The same formatting rules are also used by the built-in Tcl command **format**.

In a freshly loaded Tcl library, **Tcl_Panic** prints the formatted error message to the standard error file of the process, and then calls **abort** to terminate the process. **Tcl_Panic** does not return.

Tcl_SetPanicProc may be used to modify the behavior of **Tcl_Panic**. The *panicProc* argument should match the type **Tcl_PanicProc**:

```
typedef void Tcl_PanicProc(
      const char *format,
      arg, arg,...);
```

After **Tcl_SetPanicProc** returns, any future calls to **Tcl_Panic** will call *panicProc*, passing along the *format* and *arg* arguments. To maintain consistency with the callers of **Tcl_Panic**, *panicProc* must not return; it must call **abort**. *panicProc* should avoid making calls into the Tcl library, or into other libraries that may call the Tcl library, since the original call to **Tcl_Panic** indicates the Tcl library is not in a state of reliable operation.

The typical use of **Tcl_SetPanicProc** arranges for the error message to be displayed or reported in a manner more suitable for the application or the platform. As an example, the Windows implementation of **wish** calls **Tcl_SetPanicProc** to force all panic messages to be displayed in a system dialog box, rather than to be printed to the standard error file (usually not visible under Windows).

Although the primary callers of **Tcl_Panic** are the procedures of the Tcl library, **Tcl_Panic** is a public function and may be called by any extension or application that wishes to abort the process and have a panic message displayed the same way that panic messages from Tcl will be displayed.

Tcl_PanicVA is the same as **Tcl_Panic** except that instead of taking a variable number of arguments it takes an argument list.

SEE ALSO

abort(3), printf(3), exec(3tcl), format(3tcl)

KEYWORDS

abort, fatal, error