NAME

Tcl_CreatePipeline − create one or more child processes, with I/O redirection

SYNOPSIS

#include <tcl.h>

int
Tcl_CreatePipeline(*interp, argc, argv, pidArrayPtr, inPipePtr, outPipePtr, errFilePtr*)

ARGUMENTS

Tcl_Interp	**interp*	(in)	Interpreter to use for error reporting.
int	*argc*	(in)	Number of strings in *argv* array.
char	***argv*	(in)	Array of strings describing command(s) and I/O redirection.
int	***pidArrayPtr*	(out)	The value at **pidArrayPtr* is modified to hold a pointer to an array of process identifiers. The array is dynamically allocated and must be freed by the caller.
char	**inPipePtr*	(out)	If this argument is NULL then standard input for the first command in the pipeline comes from the current standard input. If *inPipePtr* is not NULL then **Tcl_CreatePipeline** will create a pipe, arrange for it to be used for standard input to the first command, and store a file id for writing to that pipe at **inPipePtr*. If the command specified its own input using redirection, then no pipe is created and -1 is stored at **inPipePtr*.
char	**outPipePtr*	(out)	If this argument is NULL then standard output for the last command in the pipeline goes to the current standard output. If *outPipePtr* is not NULL then **Tcl_CreatePipeline** will create a pipe, arrange for it to be used for standard output from the last command, and store a file id for reading from that pipe at **outPipePtr*. If the command specified its own output using redirection then no pipe is created and -1 is stored at **outPipePtr*.
char	**errFilePtr*	(out)	If this argument is NULL then error output for all the commands in the pipeline will go to the current standard error file. If *errFilePtr* is not NULL, error output from all the commands in the pipeline will go to a temporary file created by **Tcl_CreatePipeline**. A file id to read from that file will be stored at **errFilePtr*. The file will already have been removed, so closing the file descriptor at **errFilePtr* will cause the file to be flushed completely.

DESCRIPTION

Tcl_CreatePipeline processes the *argv* array and sets up one or more child processes in a pipeline configuration. **Tcl_CreatePipeline** handles pipes specified with ''|'', input redirection specified with ''<'' or ''<<'', and output redirection specified with ''>''; see the documentation for the **exec** command for details on these specifications. The return value from **Tcl_CreatePipeline** is a count of the number of child processes created; the process identifiers for those processes are stored in a *malloc*-ed array and a pointer to that array is stored at **pidArrayPtr*. It is the caller's responsibility to free the array when finished with it.

If the *inPipePtr*, *outPipePtr*, and *errFilePtr* arguments are NULL then the pipeline's standard input, standard output, and standard error are taken from the corresponding streams of the process. Non-NULL values may be specified for these arguments to use pipes for standard input and standard output and a file for standard error. **Tcl_CreatePipeline** will create the requested pipes or file and return file identifiers that

may be used to read or write them. It is the caller's responsibility to close all of these files when they are no longer needed. If *argv* specifies redirection for standard input or standard output, then pipes will not be created even if requested by the *inPipePtr* and *outPipePtr* arguments.

If an error occurs in **Tcl_CreatePipeline** (e.g. "|" or "<" was the last argument in *argv*, or it wasn't possible to fork off a child), then -1 is returned and *interp->result* is set to an error message.

SEE ALSO

Tcl_WaitPids, **Tcl_DetachPids**

KEYWORDS

background, child, detach, fork, process, status, wait