Chapter 51

perlothrtut

Old tutorial on threads in Perl

51.1 DESCRIPTION

WARNING: This tutorial describes the old-style thread model that was introduced in release 5.005. This model is now deprecated, and will be removed, probably in version 5.10. The interfaces described here were considered experimental, and are likely to be buggy.

For information about the new interpreter threads ("ithreads") model, see the *perlthrtut* tutorial, and the *threads* and *threads::shared* modules.

You are strongly encouraged to migrate any existing threads code to the new model as soon as possible.

51.2 What Is A Thread Anyway?

A thread is a flow of control through a program with a single execution point.

Sounds an awful lot like a process, doesn't it? Well, it should. Threads are one of the pieces of a process. Every process has at least one thread and, up until now, every process running Perl had only one thread. With 5.005, though, you can create extra threads. We're going to show you how, when, and why.

51.3 Threaded Program Models

There are three basic ways that you can structure a threaded program. Which model you choose depends on what you need your program to do. For many non-trivial threaded programs you'll need to choose different models for different pieces of your program.

51.3.1 Boss/Worker

The boss/worker model usually has one 'boss' thread and one or more 'worker' threads. The boss thread gathers or generates tasks that need to be done, then parcels those tasks out to the appropriate worker thread.

This model is common in GUI and server programs, where a main thread waits for some event and then passes that event to the appropriate worker threads for processing. Once the event has been passed on, the boss thread goes back to waiting for another event.

The boss thread does relatively little work. While tasks aren't necessarily performed faster than with any other method, it tends to have the best user-response times.