Preface

The *Multithreaded Programming Guide* describes the multithreaded programming interfaces for POSIX and Solaris threads in the Solaris™ 2.5 system. This guide shows application programmers how to create new multithreaded programs and how to add multithreading to existing programs.

Although this guide covers both the POSIX and Solaris threads implementations, most topics assume a POSIX threads interest. Information applying to only Solaris threads is covered in a special chapter.

To understand this guide, a reader must be familiar with

- A UNIX® SVR4 system—preferably the Solaris 2.5 system
- The C programming language—multithreading is implemented through the `libthread` library
- The principles of concurrent programming (as opposed to sequential programming)—multithreading requires a different way of thinking about function interactions. Some books you might want to read are:
  - *Algorithms for Mutual Exclusion* by Michel Raynal (MIT Press, 1986)
  - *Concurrent Programming* by Alan Burns & Geoff Davies (Addison-Wesley, 1993)
  - *Distributed Algorithms and Protocols* by Michel Raynal (Wiley, 1988)
  - *Operating System Concepts* by Silberschatz, Peterson, & Galvin (Addison-Wesley, 1991)
  - *Principles of Concurrent Programming* by M. Ben-Ari (Prentice-Hall, 1982)

How This Guide Is Organized

Chapter 1, "Covering Multithreading Basics," gives a structural overview of threads implementation in this release.

Chapter 2, "Basic Threads Programming," discusses the general POSIX threads library routines, emphasizing creating a thread with default attributes.

Chapter 3, "Thread Create Attributes," covers creating a thread with nondefault attributes.

Chapter 4, "Programming With Synchronization Objects," covers the threads library synchronization routines.

Chapter 5, "Programming With the Operating System," discusses changes to the operating system to support multithreading.

Chapter 6, "Safe and Unsafe Interfaces," covers multithreading safety issues.

Chapter 7, "Compiling and Debugging," covers the basics of compiling and debugging multithreaded applications.

Chapter 8, "Tools for Enhancing MT Programs," describes some of the tools available for gathering performance and debugging information about your multithreaded programs.

Chapter 9, "Programming with Solaris Threads," covers the Solaris threads (as opposed to POSIX threads) interfaces.

Chapter 10, "Programming Guidelines," discusses issues that affect programmers writing multithreaded applications.

Appendix A, "Sample Application – Multithreaded grep," shows how code can be designed for POSIX threads.

Appendix B, "Solaris Threads Example: barrier.c" shows an example of building a barrier in Solaris threads.

Appendix C, "MT Safety Levels: Library Interfaces" lists the safety levels of library routines.

You might be able to find additional useful information about multithreaded programming by browsing the following World Wide Web (WWW) site:

`http://www.sun.com/sunsoft/Products/Developer-products/sig/threads`

What Typographic Changes and Symbols Mean

Table P-1 describes the type changes and symbols used in this guide.

Table P-1 Typographic Conventions

Typeface or Symbol	Meaning	Example
`AaBbCc123`	Commands, files, directories, and C functions; code examples	The `fork1()` function is new. Use `ls -a` to list all files.
AaBbCc123	Variables, titles, and emphasized words	The *stack_size* value is set by... You *must* specify a zero value.
`AaBbCc123`	What you type, contrasted with on-screen computer output	`system%` **`cc prog.c`**
`page(#)`	The man page name and section in the *Solaris Reference Manual*	See `thr_create`(3T).

Sections of program code in the main text are enclosed in boxes:

```
nt test (100);

main()
{
    register int a, b, c, d, e, f;

    test(a) = b & test(c & 0x1) & test(d & 0x1);
```