Thread Encyclopedia Article

Thread

The following sections of this BookRags Literature Study Guide is offprint from Gale's For Students Series: Presenting Analysis, Context, and Criticism on Commonly Studied Works: Introduction, Author Biography, Plot Summary, Characters, Themes, Style, Historical Context, Critical Overview, Criticism and Critical Essays, Media Adaptations, Topics for Further Study, Compare & Contrast, What Do I Read Next?, For Further Study, and Sources.

(c)1998-2002; (c)2002 by Gale. Gale is an imprint of The Gale Group, Inc., a division of Thomson Learning, Inc. Gale and Design and Thomson Learning are trademarks used herein under license.

The following sections, if they exist, are offprint from Beacham's Encyclopedia of Popular Fiction: "Social Concerns", "Thematic Overview", "Techniques", "Literary Precedents", "Key Questions", "Related Titles", "Adaptations", "Related Web Sites". (c)1994-2005, by Walton Beacham.

The following sections, if they exist, are offprint from Beacham's Guide to Literature for Young Adults: "About the Author", "Overview", "Setting", "Literary Qualities", "Social Sensitivity", "Topics for Discussion", "Ideas for Reports and Papers". (c)1994-2005, by Walton Beacham.

All other sections in this Literature Study Guide are owned and copyrighted by BookRags, Inc.



Contents

Thread Encyclopedia Article.....1

Contents.....2

Thread.....3

Thread

A thread is a portion of a program that runs separately during execution. As a program runs it spawns off separate functions as threads. Threads cannot run independently; they remain under the control of the program. Threads are sometimes referred to as "lightweight processes." Both threads and processes are each a sequential flow of control, but a thread is lightweight because it is running within the context of a complete program. It must share system resources belonging to the program of which it is a part.

A thread contains the information needed to serve a particular service request. If multiple users are using the program or concurrent service requests occur, a thread is created and maintained for each of them. Thread information is stored in a special **data** area, the address of which is recorded in the **register**.

Most operating systems support both **multitasking** and multithreading. Multithreading is the ability to run multiple threads simultaneously, with each thread performing a different task. This allows for faster execution of an application, especially when the application is run on a computer with multiple processors.

An application must be written in a certain way to take advantage of a multithreaded operating system. Some **programming** languages have built-in support for threading, and others require the use of **API**s (**Application Programming Interface**s). An example of a multithreaded application is a web **browser** that allows a user to scroll up or down on a web page while downloading additional images.

Multithreaded **applications** are complex to manage. Because threads share system resources, it is easy for one thread to tie up the **CPU** and prevent other threads in the program from executing. Different operating systems have different mechanisms for thread management.