

Process and Threads

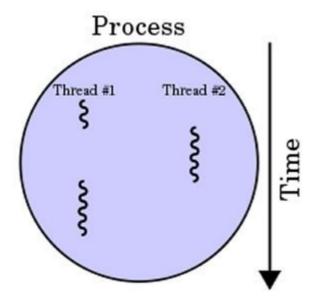
Confidentiality Notice

Copyright (c) 2018 eInfochips. - All rights reserved

This document is authored by eInfochips and is eInfochips intellectual property, including the copyrights in all countries in the world. This document is provided under a license to use only with all other rights, including ownership rights, being retained by eInfochips. This file may not be distributed, copied, or reproduced in any manner, electronic or otherwise, without the express written consent of eInfochips.

Process: In simple words a process is executing a program. But not all, it's only an instance of a computing program. Several processes may be associated with the same program. Process contains program code and its current activity.

Thread: We can say thread is a light weight process. A thread of execution is the smallest sequence of programmed instructions that can be managed independently by scheduler. Threads reside inside the process. Each thread belongs to exactly one process. No thread exists outside the process.



A process is an active program i.e. a program that is under execution. It is more than the program code as it includes the program counter, process stack, registers, program code etc. Compared to this, the program code is only the text section.

A thread is a lightweight process that can be managed independently by a scheduler. It improves the application performance using parallelism. A thread shares information like data segment, code segment, files etc. with its peer threads while it contains its own registers, stack, counter etc.

The major differences between a process and a thread are given as follows:

Comparison Basis	Process	Thread
Definition	A process is a program under execution i.e an active program.	A thread is a lightweight process that can be managed independently by a scheduler.
Context	Processes require more time for	Threads require less time for context
switching time	context switching as they are more	e switching as they are lighter than
J	heavy.	processes.
Memory	Processes are totally independent	t A thread may share some memory with
Sharing	and don't share memory.	its peer threads.
	Communication between processes	S Communication between threads
Communication	requires more time than between	requires less time than between
	threads.	processes .
Blocked	If a process gets blocked, remaining	g If a user level thread gets blocked, all
Diocked	processes can continue execution.	of its peer threads also get blocked.
Resource	Processes require more resources	Threads generally need less resources
Consumption	than threads.	than processes.
Dependency	Individual processes are independent	t Threads are parts of a process and so
Dependency	of each other.	are dependent.
Data and Code	e Processes have independent data and	l A thread shares the data segment, code
sharing	code segments.	segment, files etc. with its peer threads.
TT 1	A 11 41 1°CC 4	
Treatment by	All the different processes are	All user level neer threads are treated as
Treatment by	All the different processes are treated separately by the operating	All user level peer threads are treated as
OS OS	7	All user level peer threads are treated as
OS	treated separately by the operating	All user level peer threads are treated as a single task by the operating system.
OS	treated separately by the operating system.	All user level peer threads are treated as a single task by the operating system.
OS Time for creation	treated separately by the operating system. r Processes require more time for	All user level peer threads are treated as a single task by the operating system. Threads require less time for creation.