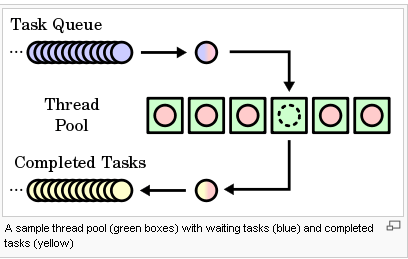
This article describes the definition and uses of Task And Thread:

* What is Task?
* What is Thread?
* Why do we need Task?
* Why do we need Thread?
* How to implement Task
* How to implement Thread
* Differences between Task And Thread

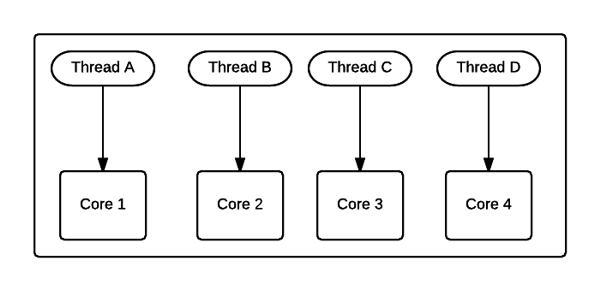
What is Task in C#?

.NET framework provides Threading.Tasks class to let you create tasks and run them asynchronously. A task is an object that represents some work that should be done. The task can tell you if the work is completed and if the operation returns a result, the task gives you the result.



What is Thread?

.NET Framework has thread-associated classes in System.Threading namespace.  A Thread is a small set of executable instructions.



**Why we need Tasks**

It can be used whenever you want to execute something in parallel. Asynchronous implementation is easy in a task, using’ async’ and ‘await’ keywords.

**Why we need a Thread**

When the time comes when the application is required to perform few tasks at the same time.

Here is a beginner tutorial on [Introduction to Threading in C#](https://www.c-sharpcorner.com/article/task-and-thread-in-c-sharp/onference.c-sharpcorner.com)

How to create a Task

1. **static** **void** Main(string[] args) {
2. Task < string > obTask = Task.Run(() => (
3. **return**“ Hello”));
4. Console.WriteLine(obTask.result);
5. }

How to create a Thread

1. **static** **void** Main(string[] args) {
2. Thread thread = **new** Thread(**new** ThreadStart(getMyName));
3. thread.Start();
4. }
5. **public** **void** getMyName() {}

Differences Between Task And Thread

Here are some differences between a task and a thread.

1. The Thread class is used for creating and manipulating a [thread](http://msdn.microsoft.com/en-us/library/windows/desktop/ms684841%28v=vs.85%29.aspx) in Windows. A [Task](http://msdn.microsoft.com/en-us/library/vstudio/system.threading.tasks.task) represents some asynchronous operation and is part of the [Task Parallel Library](http://msdn.microsoft.com/en-us/library/dd460717%28v=vs.110%29.aspx), a set of APIs for running tasks asynchronously and in parallel.
2. The task can return a result. There is no direct mechanism to return the result from a thread.
3. Task supports cancellation through the use of cancellation tokens. But Thread doesn't.
4. A task can have multiple processes happening at the same time. Threads can only have one task running at a time.
5. We can easily implement Asynchronous using ’async’ and ‘await’ keywords.
6. A new Thread()is not dealing with Thread pool thread, whereas Task does use thread pool thread.
7. A Task is a higher level concept than Thread.