Registration ID

109493065

In C#, both Wait and Join are used in multithreading to manage thread execution, but they serve different purposes:

1. **Thread.Join**:
   * **Purpose**: It is used to make one thread wait for another thread to complete.
   * **Usage**: When you call Join on a thread, the calling thread will pause and wait until the thread on which Join was called has finished executing.
   * **Example**:
   * Thread thread = new Thread(SomeMethod);
   * thread.Start();
   * thread.Join(); // Main thread waits for 'thread' to finish
   * Console.WriteLine("Thread has finished execution.");
2. **Monitor.Wait**:
   * **Purpose**: It is used in conjunction with Monitor.Enter and Monitor.Exit to release the lock on an object and wait for a signal.
   * **Usage**: Typically used in scenarios involving producer-consumer problems where a thread needs to wait for a condition to be met before continuing.
   * **Example**:
   * lock (someObject)
   * {
   * Monitor.Wait(someObject); // Releases the lock and waits for a signal
   * // Code here runs after receiving the signal and reacquiring the lock

}In summary, Thread.Join is used to wait for a thread to finish, while Monitor.Wait is used to wait for a condition to be met while releasing the lock on an object

using System;

using System.Diagnostics;

using System.Threading;

namespace joinandsleep {

classProgram {

staticvoid Run() {

for (int i = 0; i < 50; i++) Console.Write("C#corner");

}

staticvoid Main(string[] args) {

Thread th = newThread(Run);

th.Start();

th.Join();

Console.WriteLine("Thread t has terminated !");

Console.Read();

}

}

}

The difference between these two methods is that Task.Run is a static method that both instantiates **and** runs a task, whereas Task.Start is an instance method, which you run on an existing Task instance.

// Create the task, start it and wait for it to finish.

var result1 = await Task.Run(CountCharacters);

Console.WriteLine(result1);

// Create the task

var result2 = new Task<int>(CountCharacters);

// Start it

result2.Start();

// Wait for it to finish

Console.WriteLine(await result2);

int CountCharacters() {

return 3;

}