**What is the purpose of async/await keywords?**

The async and await keywords in C# are used to handle asynchronous programming, allowing you to write non-blocking code in a more readable way.

**Purpose of async and await**

1. **Non-blocking Execution**: They enable asynchronous operations without blocking the main thread, improving application responsiveness, especially in UI and web applications.
2. **Improved Readability**: Instead of using callbacks or complex threading logic, async/await makes asynchronous code look and behave more like synchronous code.
3. **Better Performance**: Increases efficiency by freeing up resources while waiting for I/O-bound operations (e.g., database queries, API calls, file operations) to complete.

**How async and await Work**

* async is used to mark a method as asynchronous.
* await is used to pause execution until an awaited task completes, without blocking the calling thread.

**Example:**

using System;

using System.Threading.Tasks;

class Program

{

static async Task Main()

{

Console.WriteLine("Before async call");

string result = await GetDataAsync();

Console.WriteLine(result);

Console.WriteLine("After async call");

}

static async Task<string> GetDataAsync()

{

await Task.Delay(2000); // Simulating an async operation (e.g., API call)

return "Async operation complete!";

}

}

**Key Points:**

* The Main method calls GetDataAsync and awaits the result.
* The await Task.Delay(2000) simulates an asynchronous delay without blocking.
* The output order:

Before async call

(Waits for 2 seconds)

Async operation complete!

After async call

This approach ensures that applications remain responsive and efficient, especially for tasks like I/O operations or network requests. 🚀