Manish Jaiswal

Aync Programing Series

MASTERING

Task-Based Asynchronous Pattern (TAP)

swipe

Manish Jaiswal

Aync Programing Series

01

Introduction to TAP

The Task-Based Asynchronous Pattern (TAP) is the standard pattern for asynchronous programming in .NET. TAP uses the Task and Task<TResult> types, making it easier to write, read, and maintain asynchronous code. TAP leverages the async and await keywords to handle asynchronous operations more efficiently.

swipe

Manish Jaiswal

Aync Programing Series

02

Why Choose TAP?

- Simplifies asynchronous code with async and await
- Reduces thread blocking and improves responsiveness
- Enhances code readability and maintainability
- Supports parallel processing and scalability

swipe

Manish Jaiswal

Aync Programing Series

03

Basic TAP Code Example

```
1  public async Task<string> FetchDataAsync(string url)
2  {
3      using (HttpClient client = new HttpClient())
4      {
5          string result = await client.GetStringAsync(url);
6          return result;
7      }
8  }
```

swipe

Manish Jaiswal

Aync Programing Series

04

Explanation:

- **async keyword:** Marks the method as asynchronous.
- **await keyword:** Pauses the method execution until the GetStringAsync task completes.
- Returns a **Task<string>** that represents the ongoing operation.

swipe

Manish Jaiswal

Aync Programing Series

05

Common Use Cases of TAP

- **I/O-bound operations:** Reading and writing files, accessing databases, making HTTP requests.
- **UI applications:** Keeping the UI responsive by offloading tasks to background threads.
- **Long-running tasks:** Executing CPU-intensive operations without freezing the application

swipe