# C# VA .NET PLATFORMASIDA ASINXRON DASTURLASH: ASYNC/AWAIT MEXANIZMINING SAMARADORLIK TAHLILI

*Abu Rayhon Beruniy nomidagi Urganch davlat unversiteti*
*“Kompyuter ilmlari” kafedrasi stajer-o‘qituvchisi*
**Xusainov Shixnazar Madaminovich**
*Shihnazar4220@gmail.com,+998999604220*

*Abu Rayhon Beruniy nomidagi Urganch davlat unversiteti*
*“Ijtimoiy-iqtisodiy” fanlar fakulteti tyutori*
***Jumaniyozov Boburbek Umidbek o'g'li***
*Boburbek_9111@gmail.com, +998999691994*

*Abu Rayhon Beruniy nomidagi Urganch davlat unversiteti*
*“Kompyuter ilmlari” kafedrasi katta o‘qituvchisi*
***Jumanazarova Onajon Matnazarovna***
*anaxon1954@gmail.com, +998904382759*

**Annotatsiya:** Ushbu maqola C# dasturlash tilida keng qo‘llanilayotgan async/await asinxron mexanizmining samaradorligini o‘rganishga bag‘ishlangan. Maqolada asinxron va sinxron dasturlash modellari solishtirilib, async/await yordamida bajarilgan operatsiyalarning tizim samaradorligiga ta’siri tahlil qilingan. Shuningdek, Thread va Parallel.For kabi boshqa asinxron yondashuvlar bilan funksional va resurs samaradorligi jihatdan solishtiruvlar olib borilgan. Testlar yordamida I/O-bound va CPU-bound vazifalar natijalari keltirilib, texnologiyaning optimal holatlarda qanday ishlashini aniqlashga harakat qilingan.

**Kalit so‘zlar:** C#, .NET, async/await, asinxron dasturlash, samaradorlik, Task Parallel Library, sinxronlik, resurs boshqaruvi

## Kirish

Bugungi kunda dasturiy mahsulotlardan yuqori samaradorlik, tezkor ishlov berish va foydalanuvchiga qulay interfeys talab etiladi. Ayniqsa tarmoqli so‘rovlar, fayl

o‘qish/yozish yoki ma’lumotlar bazasi bilan ishlash kabi resurs talab qiluvchi operatsiyalarda bloklanmagan (responsive) ilovalar muhim ahamiyatga ega. C# tilida bu imkoniyat async va await kalit so‘zlari orqali taqdim etiladi. Ushbu texnologiya yordamida foydalanuvchi interfeysini bloklamasdan fon rejimida operatsiyalarni bajarish mumkin. Mazkur maqola aynan shu texnologiyaning imkoniyatlari, afzalliklari va cheklovlarini ilmiy jihatdan tahlil qiladi.

## Adabiyotlarni O‘rganish

C# tilida asinxron dasturlash 2012-yildan, ya’ni .NET Framework 4.5 bilan async va await funksiyalari joriy etilgach, keng qo‘llanila boshlandi. Stephen Cleary tomonidan yozilgan *Concurrency in C# Cookbook* kitobida ushbu texnologiyaning real ilovalardagi qo‘llanilishi chuqur yoritilgan [1]. Microsoft rasmiy hujjatlarida async/await sintaksisi, foydalanish holatlari va ishlash mexanizmi haqida keng ma’lumotlar mavjud [2].

Shuningdek, Joseph Albahari tomonidan tayyorlangan “Threading in C#” onlayn manbasi orqali Task, Thread, va boshqa parallel texnologiyalar o‘rtasidagi farqlar tahlil qilingan [3]. Ilmiy maqolalarda esa async/await va Thread, Parallel.For modellarining samaradorligi yuzasidan eksperimental solishtiruvlar ko‘p o‘rganilgan [4].

## Asosiy Qism (Metodologiya va Natijalar)

**Metodologiya**: Tadqiqot jarayonida I/O-bound (masalan, HTTP so‘rovlar) va CPU-bound (matematik hisoblashlar) bo‘yicha ikkita test stsenariysi ishlab chiqildi. Har bir stsenariy async/await, Thread va Parallel.For yondashuvlari orqali alohida sinovdan o‘tkazildi. Ishlash tezligi (millisekundlarda) va xotira sarfi o‘lchandi[1].

**Kod namunasi (async/await)**:

```
public async Task<string> GetDataAsync(string url)
{
    using HttpClient client = new HttpClient();
    string data = await client.GetStringAsync(url);
    return data;
}
```

**Test natijalari**:

| Yondashuv | I/O-bound (ms) | CPU-bound (ms) |
|---|---|---|
| async/await | 380 | 710 |
| Thread | 830 | 590 |
| Parallel.For | 760 | 510 |

**Tahlil**: Jadvaldan ko‘rinib turibdiki, async/await I/O operatsiyalar uchun ancha samarali, chunki bu operatsiyalar kutish asosida bajariladi. Boshqa tomondan, CPU-heavy vazifalarda Parallel.For eng tez bajarilgan, chunki bu metod ko‘p oqimli hisob-kitoblarni samarali taqsimlaydi.

**Qo‘shimcha texniklar**: ConfigureAwait(false) parametri yordamida fon jarayonlari UI kontekstiga qaytmasdan bajarilishi mumkin, bu ayniqsa server ilovalari uchun foydalidir. Shuningdek, asinxron metodlarda try-catch bloklari bilan istisnolarni to‘g‘ri tutib olish muhim hisoblanadi.

**Amaliy holatlar**: async/await texnologiyasi ASP.NET Core’da veb-so‘rovlarni qayta ishlash, Xamarinda mobil ilovalar interfeysini bloklamasdan boshqarish, hamda WPF yoki WinForms desktop ilovalarida GUI’ni real vaqtda faol saqlashda keng qo‘llaniladi.

## Xulosa

Tadqiqot natijalari shuni ko‘rsatadiki, async/await texnologiyasi ayniqsa I/O-bound operatsiyalar uchun samarali hisoblanadi. Bu texnologiya tizim resurslarini optimal taqsimlaydi, foydalanuvchi interfeysini bloklamaydi va ilovaning ishlash barqarorligini oshiradi. Biroq, CPU-heavy jarayonlarda Parallel.For va ThreadPool yondashuvlari ba’zan samaraliroq bo‘lishi mumkin. Shu sababli, texnologiyani tanlashda vazifaning turiga qarab yondashuvni moslashtirish zarur. Umuman olganda, async/await — bu zamonaviy ilovalar uchun muhim va kuchli vositadir.

## Foydalanilgan Adabiyotlar Ro‘yxati

1.Cleary, S. (2018). *Concurrency in C# Cookbook* (2nd ed.). O'Reilly Media. https://www.oreilly.com/library/view/concurrency-in-c/9781492054504/

2.Albahari, J. (2021). *Threading in C#*. https://www.albahari.com/threading/

3.Microsoft Docs. (2023). *Asynchronous programming with async and await*. https://learn.microsoft.com/en-us/dotnet/csharp/programming-guide/concepts/async/

4.Harrison, J., & Smith, A. (2019). *Comparative study of multithreading and asynchronous programming models in C#. International Journal of Computer Applications*, 162(8), 10–18.