Дәріс 8. Dispose() әдісі. TaskFactory класы. Лямбда-өрнек тапсырма ретінде. Тапсырманың (Task) жалғасын құру.

Дәрістің мақсаты: Студенттерде тапсырмалармен жұмысты басқару құралдары туралы түсінік қалыптастыру.

Дәрісті меңгеру нәтижесінде студенттер келесі қабілеттерге ие болады:

- Dispose() әдісінің қызметін түсіну;
- Тапсырманы лямбда-өрнек түрінде жүзеге асыру ерекшеліктерін түсіндіру;
- Тапсырманың жалғасын құру тәсілдерін түсіну;
- TaskFactory класының қызметін түсіндіру.

Dispose() әдісін шақыру

Task класында Dispose() әдісі анықталатын IDisposable интерфейсі іске асырылады. Төменде оның хабарландыру нысаны келтірілген.

public void Dispose()

Dispose() әдісі осы сынып пайдаланатын ресурстарды босатып, Task сыныбында іске асырылады. Әдетте, Task класына байланысты ресурстар автоматты түрде "қоқыс жинау" кезінде (немесе бағдарлама аяқталғаннан кейін) босатылады. Бірақ егер бұл ресурстарды ертерек босату қажет болса, онда Dispose() әдісі осы мақсатқа қызмет етеді.

Бұл тағдырдың еркіндігіне қалдырылатын көптеген міндеттер жасалатын бағдарламаларда ерекше маңызды. Алайда, Dispose() әдісін тек ол аяқталғаннан кейін ғана жеке тапсырма үшін шақыруға болатынын ескеру керек. Демек, Dispose() әдісін шақырмас бұрын жеке тапсырманы аяқтау фактісін анықтау үшін кейбір механизм қажет болады, мысалы, Wait() әдісін шақыру. Сондықтан Dispose() әдісін талқыламас бұрын Wait() әдісін қарастыру аса маңызды болды. Егер әлі де белсенді міндет үшін Dispose() бағдарламасын шақыруға әрекет жасасаңыз, онда InvalidOperationException ерекшелігі жасалынады.

Тапсырманы іске қосу үшін TaskFactory класын қолдану

Жоғарыда келтірілген бағдарламаның мысалдары қажет болғандай тиімді жасалмаған, себебі тапсырманы жасауға және бірден оны орындауды бастауға болады, StartNew сыныбында анықталған TaskFactory() әдісін тудырады. TaskFactory сыныбында міндеттерді жасауды және оларды басқаруды жеңілдететін әртүрлі әдістер ұсынылады. Әдепкі TaskFactory сыныбының нысаны тек Task сыныбында оқуға арналған Factory сипатынан алынуы мүмкін. Осы сипатты пайдалана отырып, TaskFactory сыныбының кез келген әдісін шақыруға болады. Төменде оны хабарландырудың ең қарапайым нысаны келтірілген:

public Task StartNew(Action action)

мұнда action - орындалатын тапсырмаға кіру нүктесі. Алдымен StartNew() әдісінде action параметрімен анықталатын әрекет үшін Task түріндегі нысанның данасы автоматты түрде жасалады, содан кейін тапсырманы орындауға іске қосу жоспарлануда. Демек, Start() әдісін шақыру қажеттілігі енді жойылады.

Мысалы, бұрын қаралған бағдарламаларда StartNew() әдісін келесі шақыру tsk тапсырмасын бір әрекетпен жасауға және іске қосуға әкеледі.

Task tsk = Task.Factory.StartNew(MyTask);

Осыдан кейін оператор бірден MyTask() әдісін орындай бастайды.

Міндет құрылып, бірден орындауға іске қосылатын жағдайларда StartNew() әдісі неғұрлым тиімді көрсетіледі.

Лямбда өрнекті тапсырма ретінде қолдану

Әдеттегі әдісті тапсырма ретінде пайдаланудан басқа, басқа, анағұрлым ұтымды тәсіл бар: лямбда-өрнекті жеке шешілетін тапсырма ретінде көрсету. Естеріңізге сала кетейік, лямбда-өрнектер жасырын функциялардың ерекше түрі болып табылады. Сондықтан олар жеке міндеттер ретінде орындалуы мүмкін. Әдістің жалғыз мақсаты бір реттік міндетті шешу болып табылатын жағдайларда лямбда-өрнектер ерекше пайдалы болып шығады. Лямбда-өрнектер жеке тапсырма жасай алады және басқа әдістерді шақыра алады. Қалай болғанда да, лямбда-өрнекті міндет ретінде қолдану аталған әдіске тартымды балама бола алады.

Төменде келтірілген бағдарлама мысалында лямбда өрнекті тапсырма ретінде қолдану көрсетіледі. Бұл бағдарламада бағдарламалардың алдыңғы мысалдарынан MyTask () әдісінің коды лямбда-өрнекке түрлендіріледі.

Тапсырманың жалғасын құру

Кітапхананың жаңашылдық және өте ыңғайлы ерекшеліктерінің бірі - тапсырманы жалғастыру мүмкіндігі. Жалғастыру - басқа тапсырма аяқталғаннан кейін автоматты түрде басталатын бір тапсырма. Жалғасын, атап айтқанда, Task сыныбында анықталған ContinueWith() әдісінің көмегімен жасауға болады. Төменде оны хабарландырудың қарапайым нысаны келтірілген:

public Task ContinueWith (Action < Task > жалғастыру әрекеті)

мұнда _ жалғастыру әрекеті шақырушы тапсырма аяқталғаннан кейін іске қосылатын тапсырманы білдіреді. Action делегатында Task түрінің жалғыз параметрі бар. Демек, осы әдісте қолданылатын Action делегатының нұсқасы мынадай болып келеді.

public delegate void Action<in T>(T obj)

Бұл жағдайда T жиынтықталған параметрі Task класын білдіреді.
Тапсырманы жалғастыру келесі бағдарламаның мысалында көрсетіледі.

```
using System;
using System.Threading;
using System.Threading.Tasks;
class ContinuationDemo {
// Тапсырма ретінде орындалатын әдіс
static void MyTask() {
Console.WriteLine("MyTask() іске қосылды");
for(int count = 0; count < 5; count++) {
Thread.Sleep(500);
Console.WriteLine("MyTask() әдісіндегі санауыш мәні: " + count );
}
Console.WriteLine("MyTask аяқталды");
}
// Тапсырманың жалғасы ретінде орындалатын әдіс.
```

```
static void ContTask(Task t) {
Console.WriteLine("Жалғасы іске қосылды");
for(int count = 0; count < 5; count++) {
Thread.Sleep(500);
Console.WriteLine("Жалғасында санауыш мәні:  " + count );
}
Console.WriteLine("Жалғасы аяқталды");
}
static void Main() {
Console.WriteLine("Негізгі ағын іске қосылды.");
// Алғашқы тапсырманың объектісін құру
Task tsk = new Task(MyTask);
// Тапсырманың жалғасын құру
Task taskCont = tsk.ContinueWith(ContTask);
// Тапсырмалар тізбегін іске қосу.
tsk.Start();
// Жалғасының аяқталуын күту.
taskCont.Wait();
tsk.Dispose();
taskCont.Dispose();
Console.WriteLine("Негізгі ағын аяқталды.");
}
}
```