Дәріс 6. Ағын жұмысын тоқтату. Ағынның күйі. Негізгі ағынмен жұмыс.

Дәрістің мақсаты: Студенттерде ағындардың жұмысын тоқтату және ағындар күйлерін басқару туралы түсінік қалыптастыру.

Дәрісті меңгеру нәтижесінде студенттер келесі қабілеттерге ие болады:

- Ағын жұмысын тоқтату әдістерін түсіну;
- Ағынның күйлерін ажырату;
- Негізгі ағынның ерекшеліктерін түсіну.

Ағындарды тоқтату

Кейде ағынды қалыпты аяқталғанға дейін тоқтату пайдалы. Мысалы, реттеушіге бақылаудан шыққан ағынды үзу қажет болуы мүмкін. Үзілгеннен кейін ағын жүйеден жойылады және қайта басталмайды. Ағынды қалыпты аяқталғанға дейін үзу үшін Thread.Abort() әдісі қызмет етеді. Төменде осы әдістің қарапайым пішіні келтірілген.

public void Abort()

Abort() әдісі қажетті жағдайлар жасайды, Ол шақырылған ағында ThreadAbortException ерекшеліктерді жасау үшін. Бұл ерекшелік ағынның үзілуіне әкеліп соғады және бағдарлама кодында да ұсталуы мүмкін, бірақ бұл жағдайда ол ағынды тоқтату үшін автоматты түрде қайтадан жасалады. Abort() әдісі ағынды тез арада тоқтата алмайды, сондықтан егер ағынды бағдарламаны орындауды жалғастыру алдында тоқтату қажет болса, онда Abort() әдісінен кейін бірден Join() әдісін шақыру керек. Бұдан басқа, ең сирек жағдайларда Abort() әдісі ағынды мүлдем тоқтата алмайды. Бұл, мысалы, finally кодтық блогы шексіз циклге кірген жағдайда болады.

Төменде келтірілген бағдарлама мысалында ағынды үзу үшін Abort() әдісін қолдану көрсетіледі.

```
using System.Threading;
class MyThread {
public Thread Thrd;
public MyThread(string name) {
Thrd = new Thread(this.Run);
Thrd.Name = name;
Thrd.Start();
}
// Ағынға кіру нүктесі.
void Run() {
Console.WriteLine(Thrd.Name + " басталды.");
for(int i = 1; i <= 1000; i++) {
Console.Write(i + " ");
if((i%10)==0) {
Console.WriteLine();
Thread.Sleep(250);
}
}
Console.WriteLine(Thrd.Name + " аяқталды.");
}
}
class StopDemo {
```

```
static void Main() {
MyThread mt1 = new MyThread("Ағын 1");
Thread.Sleep(1000); // туынды ағынның орындалуына рұқсат беру
Console.WriteLine("Ағын жұмысын тоқтату.");
mt1.Thrd.Abort();
mt1.Thrd.Join(); // ағын жұмысының тоқтатылуын күту
Console.WriteLine("Негізгі ағын жұмысы тоқтатылды.");
}
}
```

Кейбір жағдайларда төменде жалпы түрде келтірілген Abort() әдісінің басқа нысаны пайдалы болады:

public void Abort (object stateInfo)

онда ол тоқтаған кезде ағынға беру талап етілетін кез келген ақпаратты білдіреді. Бұл ақпарат ExceptionState алып тастау сыныбынан ThreadAbortException сипаты арқылы қол жетімді. Осы жолмен ағынға аяқталу кодын жіберуге болады. Төменде келтірілген бағдарлама мысалында Abort() әдісінің осы пішінін қолдану көрсетіледі.

```
using System;
using System.Threading;
class MyThread {
public Thread Thrd;
public MyThread(string name) {
Thrd = new Thread(this.Run);
Thrd.Name = name;
Thrd.Start();
}
// Ағынға кіру нүктесі.
void Run() {
try {
Console.WriteLine(Thrd.Name + " басталды.");
for (int i = 1; i <= 1000; i++) {
Console.Write(i + " ");
if((i%10)==0) {
Console.WriteLine();
Thread.Sleep(250);
}
}
Console.WriteLine(Thrd.Name + " қалыпты аяқталды.");
} catch(ThreadAbortException exc) {
Console.WriteLine("Ағын тоқтатылды, аяқтау коды " + exc.ExceptionState);
} }
}
class UseAltAbort {
static void Main() {
MyThread mt1 = new MyThread("Ағын 1");
Thread.Sleep(1000); // туынды ағынның орындалуына рұқсат беру
Console.WriteLine("Ағын жұмысын тоқтату.");
mt1.Thrd.Abort (100);
mt1.Thrd.Join(); // ағын жұмысының тоқтатылуын күту
Console.WriteLine("Негізгі ағын жұмысы тоқтатылды.");
```

```
}
}
```

Ағын күйі

Ағын күйі Thread сыныбында қол жетімді ThreadState сипатынан алынуы мүмкін. Төменде осы сипаттың жалпы пішіні келтірілген.

public ThreadState ThreadState{ get; }

Ағын күйі ThreadState тізімінде анықталған мән түрінде қайтарылады.

Негізгі ағын

C# бағдарламасында негізгі деп аталатын ең болмағанда бір орындау ағыны бар. Бұл ағын орындала бастағаннан кейін автоматты түрде алынады. Негізгі ағынмен барлық қалған ағындар сияқты жұмыс істеуге болады.

Негізгі ағынға қатынасу үшін оған сілтеме жасайтын Thread түріндегі нысанды алу қажет. Бұл Thread сыныбының мүшесі болып табылатын CurrentThread сипатының көмегімен жасалады. Төменде осы сипаттың жалпы пішіні келтірілген.

public static Thread CurrentThread{ get; }

Бұл сипат сілтемені ол пайдаланылатын ағынға қайтарады. Сондықтан, егер CurrentThread сипаты негізгі ағында кодты орындау кезінде пайдаланылса, онда оның көмегімен негізгі ағынға сілтеме алуға болады. Өз иелігінде осындай сілтеме бола отырып, кез келген басқа ағынды басқаруға болады.

```
// Негізгі ағынды басқару.
using System;
using System.Threading;
class UseMain {
static void Main() {
Thread Thrd;
// Негізгі ағынды алу.
Thrd = Thread.CurrentThread;
// Негізгі ағынның атын көрсету.
if(Thrd.Name == null)
Console.WriteLine("Негізгі ағынның атауы жоқ.");
else
Console.WriteLine("Негізгі ағынның атауы: " + Thrd.Name);
// Негізгі ағынның басымдығын көрсету.
Console.WriteLine("Басымдығы: " + Thrd.Priority);
Console.WriteLine();
// Атау мен басымдықты орнату.
Console.WriteLine("Атау мен басымдықты орнату.\n");
Thrd.Name = "Негізгі Ағын";
Thrd.Priority = ThreadPriority.AboveNormal;
Console.WriteLine("Енді негізгі ағын атауы: " + Thrd.Name);
Console.WriteLine("Енді басымдық: " + Thrd.Priority);
}
}
```