

C# Essential

Потоки

C# Essential

Автор курса



Александр Шевчук
MCT



MCID: 9230440

C# Essential

После урока обязательно



Повторите этот урок в видео формате на [ITVDN.com](http://itvdn.com)

Доступ можно получить через руководство вашего учебного центра



Проверьте как Вы усвоили данный материал на [TestProvider.com](http://testprovider.com)

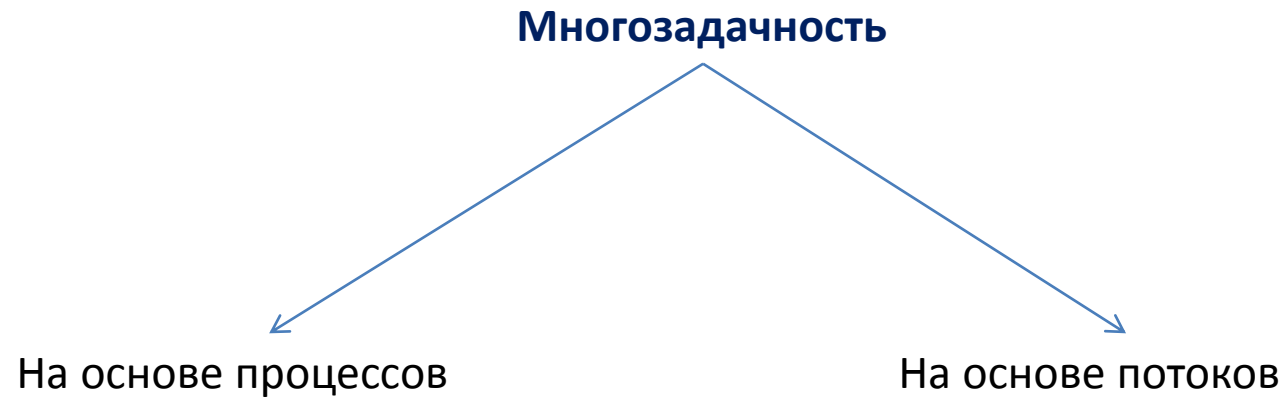
C# Essential

Тема

ПОТОКИ

Многозадачность

Multitasking



Многозадачность – свойство операционной системы или среды программирования обеспечивать возможность параллельной (или псевдопараллельной) обработки нескольких процессов

Многозадачность

На основе процессов

Позволяет выполнять одновременно более одной программы в контексте Операционной Системы.

При использовании многозадачности на основе процессов, программа является наименьшей единицей кода, выполнение которой может контролировать планировщик задач.

Многозадачность

На основе потоков

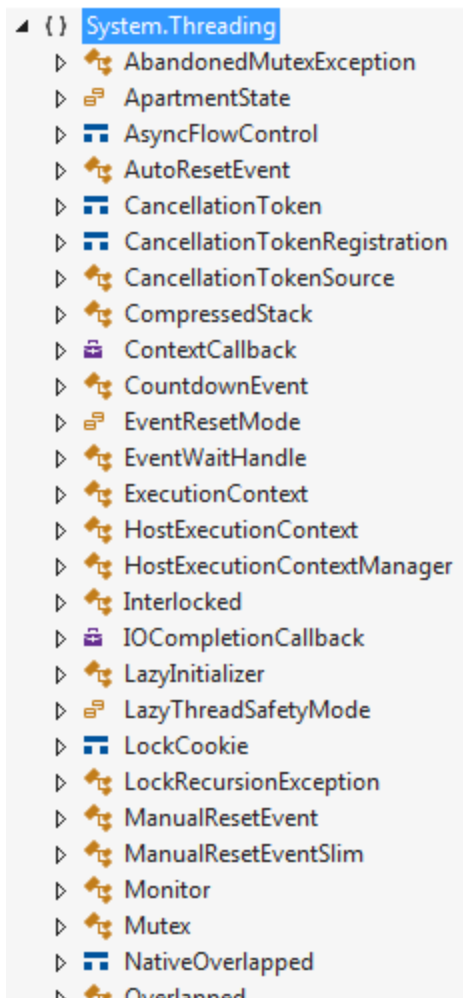
У каждого процесса может быть один или более потоков.

Это означает, что процесс может решать более одной задачи одновременно.

Многозадачность на основе потоков означает параллельное выполнение отдельных частей программы.

System.Threading

Пространство имен

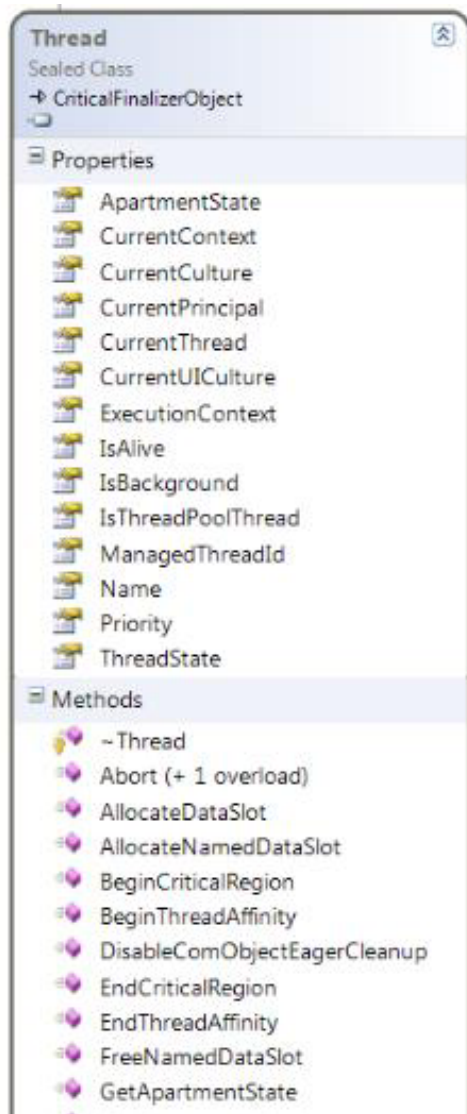


System.Threading – пространство имен для работы с потоками, содержит классы для управления потоками, такие как:

Thread, **ThreadStart**, **ParameterizedThreadStart**, **Monitor**.

Thread

Класс



Класс `Thread`, представляет собой поток. Он позволяет создавать новые потоки, управлять приоритетом потоков и получать информацию о всех потоках, существующих в рамках приложения.

```
Thread thread = new Thread(writeSecond);
```

```
static void WriteSecond()  
{  
}
```

ThreadStart

Делегат

Делегат **ThreadStart** представляет метод, который выполняется в указанном потоке **Thread**.

```
static void Main()
{
    ThreadStart writeSecond = new ThreadStart(WriteSecond);
    Thread thread = new Thread(writeSecond);
    thread.Start();

    for (int i = 0; i < 10; i++)
    {
        Console.WriteLine("Primary");
    }
}
```



ThreadStart – не позволяет передавать данные в поток

ParameterizedThreadStart

Делегат

Делегат **ParameterizedThreadStart** представляет метод, который выполняется в указанном потоке **Thread**.

```
static void Main()
{
    ParameterizedThreadStart write= new ParameterizedThreadStart(Write);
    Thread thread = new Thread(writeSecond);
    thread.Start("Hello");
}
```



ParameterizedThreadStart позволяет передавать данные в поток упакованные в **object**.

Потоки

Виды потоков

Существуют две разновидности потоков: **приоритетный** и **фоновый**.

Отличие между ними заключается в том, что процесс не завершится до тех пор, пока не окончится приоритетный поток, тогда как фоновые потоки завершаются автоматически после окончания всех приоритетных потоков.

По умолчанию создаваемый поток становится приоритетным.

Для того чтобы сделать поток фоновым, достаточно присвоить логическое значение `true` свойству `IsBackground`.

Логическое значение `false` указывает на то, что поток является приоритетным.

Синхронизация потоков

В случае использования нескольких потоков приходится координировать их действия, такой процесс называется синхронизацией.

Основная причина применения синхронизации – необходимость разделять среди двух или более потоков общий ресурс (разделяемый ресурс), который может быть одновременно доступен только одному потоку.



Синхронный поток – это поток, запущенный в контексте первичного потока.

Асинхронный поток – это поток, запущенный в контексте вторичного потока, относительно текущего.

Потоки

Ключевое слово lock

```
lock (this)
{
}

```

Ключевое слово **lock** не позволит одному потоку войти в важный раздел кода в тот момент, когда в нем находится другой поток.

При попытке входа другого потока в заблокированный код потребуется дождаться снятия блокировки объекта.

Monitor

Класс

```
Monitor.Enter(this);  
  
for (int i = 0; i < 10; i++)  
{  
    Console.WriteLine(i);  
}  
  
Monitor.Exit(this);
```

Класс `Monitor` представляет собой механизм для синхронизации доступа к объектам.

Класс `Monitor` используется для создания критических секций.

Смотрите наши уроки в видео формате

ITVDN.com



Посмотрите этот урок в видео формате на образовательном портале [ITVDN.com](http://itvdn.com) для закрепления пройденного материала.

Все курсы записаны сертифицированными тренерами, которые работают в учебном центре CyberBionic Systematics



Проверка знаний

TestProvider.com

TestProvider

Мы помогаем людям оценить себя

Главная Услуги и цены Центр Тестирования Поддержка О нас

Регистрация Войти

Поиск сертификата

Мы в социальных сетях

Тестирование

Языки программирования и информационные технологии

Microsoft

C# ASP.NET MVC JavaScript Patterns OF Design SQL Architecture Guide WCF HTML&CSS XML SEO WPF HTML5&CSS3 JQuery XNA SharePoint GUI for Android Windows Azure Platform Microsoft Patterns&Practices TFS SCRUM ReSharper TDD WWF LINQ Entity Framework Windows Forms Refactoring Microsoft Expression Blend 4 Windows Phone 8 Windows 8 AppStore Visual Studio Tips&Tricks MSF MEF SilverLight AJAX MEF Service Oriented Architecture

Добро пожаловать на TestProvider.com!

Сайт перенесен на новую облачную платформу с использованием системы единой авторизации Single Sign On. Если вы хотите восстановить статистику по предыдущим экзаменам обратитесь в [службу поддержки](#). Для восстановления информации с предыдущей версии сайта, просба написать в службу поддержки Ваш старый и новый логины.

ITVDN PROMETRIC TEST CENTER CyberBionic Microsoft Partner Windows Azure Cloud Partner EBA

TestProvider – это online сервис проверки знаний по информационным технологиям. С его помощью Вы можете оценить Ваш уровень и выявить слабые места. Он будет полезен как в процессе изучения технологии, так и общей оценки знаний IT специалиста.

После каждого урока проходите тестирование для проверки знаний на TestProvider.com

Успешное прохождение финального тестирования позволит Вам получить соответствующий Сертификат.



C# Essential

Q&A

Информационный видеосервис для разработчиков программного обеспечения

