Потоки

№ урока: 13 **Курс:** C# Essential

Средства обучения: Компьютер с установленной Visual Studio

Обзор, цель и назначение урока

Рассмотрение многопоточности.
Рассмотрение критических секций.

Изучив материал данного занятия, учащийся сможет:

- Понимать и использовать пространства имен.
- Понимать работу критических секций, как средств синхронизации доступа нескольких потоков к разделяемым ресурсам.

Содержание урока

1. Потоки.
2. Класс `Thread`.
3. Делегат **`ThreadStart`**.
4. Делегат **`ParameterizedThreadStart`**.
5. Критическая секция.
6. Оператор `lock`.
7. Класс - `Monitor`.

Резюме

- C# поддерживает параллельное выполнение кода через многопоточность.
- Потоки и процессы – это связанные понятия в вычислительной технике. Оба представляют собой последовательность инструкций, которые должны выполняться процессором в определенном порядке.
- **Поток** – это независимый путь исполнения, способный выполняться одновременно с другими потоками.
- **Поток (Thread)** – путь выполнения действий внутри исполняемого приложения.
- **Поток** – элементарная единица исполнения, которую можно планировать средствами операционной системы.
- **Потоки ввода-вывода (stream)** – предоставляют возможность писать и читать байты из вспомогательного запоминающего устройства, которым может являться одно из нескольких устройств хранения информации (место на диске, оперативной памяти и т.д.).
- **Задача (Task)** – путь выполнения действий внутри исполняемого приложения. Исполнением задач управляет планировщик задач, а не планировщик потоков (в случае потоков), который работает с пулом потоков. Задачи можно воспринимать как оболочку для пула потоков и предпочтительного способа планирования потоков (хотя и за счет дополнительных накладных расходов). Существующие методы пула потоков продолжат работать, но задачи намного легче использовать, и они предлагают дополнительную функциональность.
- Процессы существуют в операционной системе и соответствуют тому, что пользователи видят как программы или приложения. Поток существует внутри процесса. Каждый процесс состоит из одного или более потоков.
- Программа на C# запускается как единственный поток, автоматически создаваемый CLR и операционной системой (“главный” или первичный поток), и становится многопоточной при помощи создания дополнительных потоков.
- Процесс может создавать один или более потоков для выполнения частей программного кода, связанного с процессом. Следует использовать делегат **`ThreadStart`** или

CyberBionic systematics
CyberBionic Systematics ® 2015
19 Mariny Raskovoy Str., 5 floor
Kyiv, Ukraine
t. +380 (44) 361-8473
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com

`ParameterizedThreadStart` для задания программного кода, управляемого потоком. С помощью делегата `ParameterizedThreadStart` можно передавать данные в потоковую процедуру.

- 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`. А логическое значение `false` указывает на то, что поток
- является приоритетным.
- 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.
- В основу синхронизации положено понятие блокировки, посредством которой организуется управление доступом к кодовому блоку (критической секции). Когда доступный для каждого из потоков объект (объект синхронизации доступа) заблокирован одним потоком, остальные потоки не могут получить доступ к заблокированному кодовому блоку (критической секции). Когда же блокировка снимается одним потоком, объект (объект синхронизации доступа) становится доступным для использования в другом потоке.
- Объектом синхронизации доступа к разделяемому ресурсу считается такой объект, который представляет синхронизируемый ресурс. В некоторых случаях им оказывается экземпляр самого ресурса или же произвольный экземпляр класса, используемого для синхронизации.
- 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.
- Ключевое слово `lock` вызывает `Monitor.Enter()` в начале блока и `Monitor.Exit()` в конце блока.

Закрепление материала

- Что такое поток?
- Какой класс нужно использовать для создания экземпляра потока?
- Какой делегат нужно использовать для передачи метода в поток?
- Каким способом можно передать в поток параметры?
- Что такое критическая секция?
- Что такое разделяемый ресурс?
- Что такое объект синхронизации доступа к разделяемому ресурсу?
- Чем отличается использование оператора `lock` от класса `Monitor`?

Дополнительное задание

Задание 1
Используя Visual Studio 2010, создайте проект по шаблону Console Application.
Напишите программу, в которой метод будет вызываться рекурсивною.
Каждый новый вызов метода выполняется в отдельном потоке.

Задание 2
Выучите основные конструкции и понятия, рассмотренные на уроке.

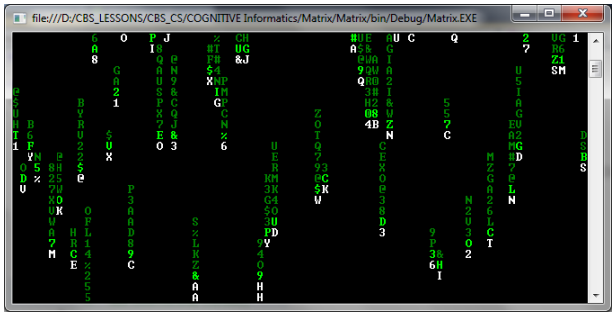
CyberBionic systematics
CyberBionic Systematics ® 2015
19 Mariny Raskovoy Str., 5 floor
Kyiv, Ukraine
t. +380 (44) 361-8473
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com

Самостоятельная деятельность учащегося

Задание 1

Используя Visual Studio, создайте проект по шаблону Console Application.

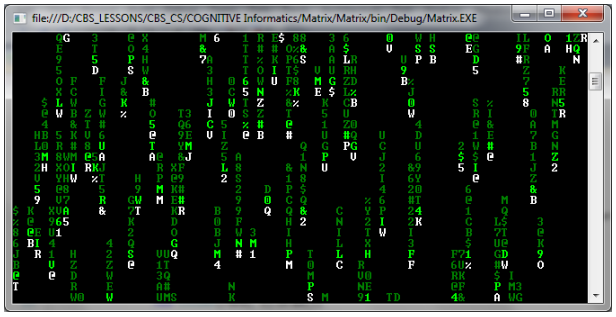
Создайте программу, которая будет выводить на экран цепочки падающих символов. Длина каждой цепочки задается случайно. Первый символ цепочки – белый, второй символ – светло-зеленый, остальные символы темно-зеленые. Во время падения цепочки, на каждом шаге, все символы меняют свое значение. Дойдя до конца, цепочка исчезает и сверху формируется новая цепочка. Смотрите ниже снимок экрана, представленный как образец.



Задание 2

Используя Visual Studio, создайте проект по шаблону Console Application.

Расширьте задание 2, так, чтобы в одном столбце одновременно могло быть две цепочки символов. Смотрите ниже снимок экрана, представленный как образец.



Задание 3

Зайдите на сайт MSDN.

Используя поисковые механизмы MSDN, найдите самостоятельно описание темы по каждому примеру, который был рассмотрен на уроке, так, как это представлено ниже, в разделе «Рекомендуемые ресурсы», описания данного урока. Сохраните ссылки и дайте им короткое описание.

Рекомендуемые ресурсы

MSDN: Потоки и работа с потоками
http://msdn.microsoft.com/ru-ru/library/6kac2kdh.aspx

MSDN: Класс `Thread`
http://msdn.microsoft.com/ru-ru/library/system.threading.thread.aspx

MSDN: Оператор `lock` (Справочник по C#)
http://msdn.microsoft.com/ru-ru/library/c5kehkcz.aspx#Y491

CyberBionic systematics
CyberBionic Systematics ® 2015
19 Mariny Raskovoy Str., 5 floor
Kyiv, Ukraine
t. +380 (44) 361-8473
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com