

Async & Await

№ урока: 15 **Курс:** C# Professional

Средства обучения: Компьютер с установленной Visual Studio

Обзор, цель и назначение урока

Данный урок завершает цикл уроков по многопоточности. Учащийся познакомится с новыми возможностями написания асинхронного кода, которые появились с выходом пятой версии языка программирования C#. На уроке рассматриваются примеры применения новой языковой конструкции `async` и `await`.

Изучив материал данного занятия, учащийся сможет:

- Использовать конструкции `async` и `await` для написания многопоточного кода

Содержание урока

1. Рассмотрение проблемы синхронных операций
2. Рассмотрение проблемы обновления GUI из другого потока
3. Знакомство с конструкцией `async` – `await`

Резюме

- Для увеличения скорости отклика приложения и увеличения производительности в программировании используется асинхронность. Поддержка асинхронности оказывается особенно важной для приложений с доступом к потоку пользовательского интерфейса, поскольку все действия, связанные с обновлением GUI, обычно выполняются в одном потоке.
- До выхода C# 5 способы создания асинхронного кода были громоздкими, трудными в написании, отладки и сопровождении.
- С выходом пятой версии C# разработчик получил инструментарий в виде ключевых слов `async` и `await`.
- Ключевое слово `async` указывает компилятору, что метод является асинхронным.
- Ключевое слово `await` указывает компилятору, что в этой точке необходимо дождаться окончания асинхронной операции (при этом управление возвращается вызвавшему методу).
- Асинхронный метод обычно содержит один или несколько вхождений оператора `await`, но отсутствие выражения `await` не вызывает ошибку компилятора.
- Метод, помеченный ключевым словом `async`, и не содержащий ни одного выражения с `await` не является асинхронным.
- По соглашению название асинхронного метода должно заканчиваться словом `Async`.
- Асинхронный метод может иметь тип возвращаемого значения `Task`, `Task<TResult>` или `void`. Метод не может иметь `ref` или `out` параметры, но может вызывать методы, которые имеют такие параметры.
- Исключения «выбрасываются» в месте вызова асинхронной операции, а не `Callback`-метода.
- Метод `Task.WhenAll` создает задачу, которая будет выполнена после выполнения всех предоставленных задач.
- Метод `Task.WhenAny` создает задачу, которая будет выполнена после выполнения любой из предоставленных задач.
- В NET 4 `Task.Factory.StartNew` был основным методом планирования новой задачи. В .NET Framework 4.5 был введен новый метод `Task.Run`. Подробнее читайте по ссылке – <http://blogs.msdn.com/b/pfxteam/archive/2011/10/24/10229468.aspx>
- Метод `Task.Run` ставит в очередь заданную работу для запуска в пуле потоков и возвращает объект `Task`, представляющий эту работу.

Закрепление материала

1. Какие проблемы возникают при синхронном выполнении кода?
2. Каким образом обновить GUI не используя новые конструкции (async – await)?
3. Как работает конструкция async – await?

Дополнительное задание

Переделайте дополнительное задание из урока №11 с использованием конструкции async await.

Самостоятельная деятельность учащегося

Задание 1

Выучите основные конструкции и понятия, рассмотренные на уроке.

Задание 2

Создайте WPF приложение, разместите в окне TextBox и две кнопки. При нажатии на первую кнопку в TextBox выводится сообщение «Подключен к базе данных» при этом в обработчике установите задержку в 3-5 сек для имитации подключения к БД, также данная кнопка запускает таймер, который с периодичностью в несколько секунд выводит в TextBox сообщение «Данные получены». При нажатии на вторую кнопку по аналогии с первой отключаемся от базы (с задержкой), выводим сообщение и останавливаем таймер.

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: async

<https://msdn.microsoft.com/ru-ru/library/hh156513.aspx>

MSDN: await

<https://msdn.microsoft.com/ru-ru/library/hh156528.aspx>

MSDN: Асинхронное программирование с использованием ключевых слов Async и Await

<https://msdn.microsoft.com/ru-ru/library/hh191443.aspx>