# ЛАБ14

1. Что такое процесс, домен, поток? Как они связаны между собой?

Процесс: Процесс представляет собой экземпляр выполняющейся программы. Он содержит исполняемый код, данные, память и другие ресурсы, необходимые для выполнения программы. Процессы изолированы друг от друга и имеют свои собственные адресные пространства.

Домен: Домен является изолированной средой выполнения внутри процесса. Он содержит одну или несколько связанных приложением сборок и предоставляет границу безопасности и изоляции для кода. В домене выполняются CLR (Common Language Runtime) и загружаются и выполняются сборки.

Поток: Поток представляет собой единицу выполнения внутри процесса. Он выполняет инструкции программы последовательно и может работать параллельно с другими потоками внутри процесса. Потоки разделяют адресное пространство процесса и могут обмениваться данными и ресурсами.

Связь между ними: Процесс содержит один или несколько доменов, а каждый домен может содержать один или несколько потоков. Потоки работают внутри домена и разделяют ресурсы, адресное пространство и контекст выполнения процесса.

1. Как получить информацию о процессах?

Для получения информации о процессах в C# можно использовать класс Process из пространства имен System.Diagnostics.

1. Как создать и настроить домен?

Создание и настройка домена осуществляется с помощью класса AppDomain. Для создания нового домена можно использовать метод CreateDomain(), указав имя домена и опциональные параметры. После создания домена можно настроить его свойства, такие как базовый каталог приложения, настройки безопасности, события разрешения сборок и т.д.

1. Как создать и настроить поток?

Создание и настройка потока осуществляется с использованием класса Thread из пространства имен System.Threading. Для создания нового потока можно передать методу Thread.Start() делегат, указывающий на метод, который будет выполняться в потоке. После создания потока можно задать его имя, приоритет, запустить его с помощью метода Start() и т.д.

1. В каких состояниях может быть поток?

Поток может находиться в следующих состояниях:

Unstarted (не запущен): поток создан, но еще не запущен.

Running (работает): поток выполняется в данный момент.

Waiting (ожидание): поток ожидает определенное событие или условие для продолжения выполнения.

Sleeping (сон): поток приостановлен на заданное время.

Suspended (приостановлен): поток приостановлен и ожидает возобновления.

Aborted (прерван): поток был прерван или завершен непредвиденно.

Stopped (остановлен): поток завершил свое выполнение.

1. Какие методы управления потоками вы знаете, для чего и как их использовать?

Некоторые методы управления потоками:

Thread.Sleep(milliseconds): Приостанавливает выполнение текущего потока на указанное количество миллисекунд.

Thread.Join(): Блокирует вызывающий поток до завершения указанного потока.

Thread.Start(): Запускает выполнение потока.

Thread.Abort(): Прерывает выполнение потока.

Thread.Suspend(): Приостанавливает выполнение потока.

Thread.Resume(): Возобновляет выполнение приостановленного потока.

Методы управления потоками используются для контроля над их выполнением, синхронизации и координации работы множества потоков в многопоточном приложении.

1. Какие приоритеты потока вы знаете?

Приоритеты потоков определяют их относительный приоритет выполнения. В C# потоку можно назначить один из следующих приоритетов:

Lowest: Самый низкий приоритет.

BelowNormal: Ниже нормального приоритета.

Normal: Нормальный приоритет (по умолчанию).

AboveNormal: Выше нормального приоритета.

Highest: Самый высокий приоритет.

Приоритеты потоков используются планировщиком потоков для определения порядка выполнения потоков с разными приоритетами.

1. Что такое пул потоков и для чего он используется?

Пул потоков (Thread Pool) представляет собой пул предварительно созданных потоков, которые могут быть повторно использованы для выполнения асинхронных задач. Пул потоков автоматически управляет созданием и уничтожением потоков, что позволяет избежать издержек, связанных с созданием новых потоков каждый раз при выполнении задачи.

Пул потоков обычно используется для выполнения асинхронных операций в параллель, таких как обработка запросов в веб-сервере или выполнение фоновых задач в приложении.

1. Что такое критическая секция? Поясните использование.

Критическая секция (Critical Section) - это участок кода, в котором доступ к общим данным или ресурсам должен быть синхронизирован между потоками. Критическая секция защищает код от одновременного доступа нескольких потоков, что может привести к состоянию гонки (race condition) и непредсказуемому поведению программы.

1. Что такое мьютекс? Поясните использование

Мьютекс (Mutex) - это объект синхронизации, который используется для обеспечения взаимного исключения при доступе к общему ресурсу из нескольких потоков. Мьютекс позволяет только одному потоку захватить его (заблокировать), в то время как другие потоки должны ожидать его освобождения.

1. Что такое семафор? Поясните использование

Семафор (Semaphore) - это объект синхронизации, который используется для ограничения количества потоков, которым разрешен одновременный доступ к определенному ресурсу или критической секции. Семафор содержит счетчик, который указывает количество доступных ресурсов.

1. Что такое неблокирующие средства синхронизации?

Неблокирующие средства синхронизации (Non-blocking synchronization primitives) представляют собой механизмы, которые позволяют потокам совместно использовать ресурсы без блокировки и ожидания друг друга. Они обеспечивают возможность выполнения операций без блокировки и позволяют потокам продолжать работу независимо от состояния других потоков.

1. Для чего можно использовать класс Timer?

Класс Timer в C# представляет собой механизм для выполнения определенного кода через определенный интервал времени или через определенное время после запуска. Он позволяет планировать выполнение задачи в будущем.