# ЛАБ12

1. Какие классы содержаться с System.IO?

В пространстве имен System.IO содержатся различные классы для работы с вводом-выводом, файлами и директориями. Некоторые из них включают:

- File: Предоставляет статические методы для работы с файлами, такие как чтение, запись, удаление и перемещение файлов.

- FileInfo: Предоставляет экземплярные методы и свойства для работы с информацией о конкретном файле.

- Directory: Предоставляет статические методы для работы с директориями, такие как создание, перемещение и удаление директорий.

- DirectoryInfo: Предоставляет экземплярные методы и свойства для работы с информацией о конкретной директории.

- Path: Предоставляет методы для работы с путями к файлам и директориям, такие как объединение, разделение и нормализация путей.

2. Для чего используются классы Directory и DirectoryInfo? В чем отличие?

Классы Directory и DirectoryInfo используются для работы с директориями. Они позволяют создавать, перемещать, удалять и получать информацию о директориях. Главное отличие между ними заключается в способе использования: Directory содержит статические методы, которые вызываются без создания экземпляра класса, в то время как DirectoryInfo предоставляет экземплярные методы и свойства для работы с конкретной директорией.

3. Для чего используются классы File и FileInfo? Какие методы они содержат.

Классы File и FileInfo используются для работы с файлами. Они предоставляют методы для чтения, записи, удаления и перемещения файлов. Главное отличие между ними заключается в способе использования: File содержит статические методы, которые вызываются без создания экземпляра класса, в то время как FileInfo предоставляет экземплярные методы и свойства для работы с конкретным файлом.

4. Для чего используются классы StreamReader и StreamWriter?

Классы StreamReader и StreamWriter используются для чтения и записи текстовых данных в файлы. StreamReader позволяет читать данные из файла, а StreamWriter позволяет записывать данные в файл. Они автоматически обрабатывают кодировку и обеспечивают удобный способ работы с текстовыми файлами.

5. Для чего используются классы BinaryWriter и BinaryReader?

Классы BinaryWriter и BinaryReader используются для чтения и записи бинарных данных в файлы. BinaryWriter позволяет записывать примитивные типы данных и другие структуры данных в бинарном формате, а BinaryReader позволяет читать эти данные из файла. Они обеспечивают более низкоуровневый доступ к данным, чем StreamReader и StreamWriter, и могут быть использованы для работы с бинарными файлами или форматами данных.

6. Как можно сжимать и восстанавливать файлы?

Для сжатия и восстановления файлов можно использовать классы из пространства имен System.IO.Compression, такие как GZipStream и DeflateStream. GZipStream позволяет сжимать и распаковывать файлы с использованием алгоритма сжатия GZip, а DeflateStream предоставляет алгоритм сжатия Deflate. Эти классы позволяют уменьшить размер файлов для экономии места на диске или передачи данных по сети.

7. Расскажите алгоритм сжатия GZip.

1. Инициализация (создание заголовка GZIP, содержит имя, дату создания, другие метаданные)

2. Сегментация (данные разбиваются на сегменты фиксированного размера)

3. Сжатие (Хаффман или LZ77)

4. Образование потока (объединение сжатых частей в одну)

5. Добавление заголовка Gzip к потоку

6. Запись в файл с расширением .gz

8. Для чего служит класс Path?

Класс Path предоставляет методы для работы с путями к файлам и директориям. Он позволяет объединять пути, получать информацию о расширении файла, нормализовать пути, проверять существование файлов и директорий, а также выполнять другие операции связанные с путями.

9. Что такое тестовый и бинарный файл. Как можно выполнить чтение и запись в эти файлы.

Тестовый файл - это файл, который содержит текст или данные, предназначенные для чтения и анализа человеком. Бинарный файл - это файл, который содержит данные в бинарном формате, которые могут быть прочитаны и интерпретированы компьютером.

Для чтения и записи в эти файлы можно использовать классы StreamReader и StreamWriter для текстовых файлов, а для бинарных файлов - классы BinaryReader и BinaryWriter. StreamReader и StreamWriter предоставляют удобные методы для чтения и записи текстовых данных. BinaryReader и BinaryWriter позволяют работать с бинарными данными, записывать их в файл и считывать из файла.

10. Что такое произвольный доступ к файлу? Приведите пример.

Произвольный доступ к файлу означает возможность чтения или записи данных в любой произвольной позиции в файле, а не только последовательно от начала до конца. Например, при доступе к бинарным файлам или файлам с фиксированной структурой, где данные разделены на записи или блоки.

Для примера, можно использовать класс FileStream в сочетании с классом BinaryReader или BinaryWriter для чтения и записи данных в определенной позиции в файле. Например, можно переместить указатель позиции в файле с помощью метода Seek и затем выполнить чтение или запись данных в текущей позиции.

11. Как применяется конструкция using (не директива) при работе с файловыми потоками? Для чего ее используют.

Конструкция using используется при работе с файловыми потоками для обеспечения автоматического освобождения ресурсов после использования. Когда объект, реализующий интерфейс IDisposable, используется в конструкции using, его метод Dispose вызывается автоматически по завершении блока using.

Например, при работе с файловым потоком FileStream в конструкции using, после окончания области видимости блока using вызывается метод Dispose, который закрывает файловый поток и освобождает связанные ресурсы. Это гарантирует правильное освобождение ресурсов и предотвращает утечки памяти.

using (FileStream fileStream = new FileStream("filename.txt", FileMode.Open))

{

// Ваши операции чтения или записи с использованием fileStream

}

После завершения блока using объект fileStream будет автоматически закрыт и освобожден.