Programming Language - C#

File handling

www.hexaware.com | © Hexaware Technologies. All rights reserved.

File handling

- File handling refers to creating a file, reading from the file, writing to the file, appending the file etc.
- The file becomes streams when we open the file for reading and writing
- A stream is a sequence of bytes which is used for communication
- Input stream and output stream are used for read and write operations
- In C# System.IO namespace contains classes which handle input and output streams

 | © Hexaware Technologies. All rights reserved.

FileStream Class

- To open an existing file or to create a new file, you need to create an object of type FileStream.

- Consider the following syntax for creating the object of type FileStream:

```
FileStream <object name>=new FileStream(<file Name>,<FileMode Enumerator>,<File Access Enumerator>,<FileShare Enumerator>);
```

 | © Hexaware Technologies. All rights reserved.

FileStream Class (Contd.)

- FileMode enumerator defines various methods for opening files.
- The FileMode enumerator parameter is specified in many constructors for the FileStream.
- The parameters to FileMode checks whether a file is overwritten, created, or opened.
- FileAccess enumerator indicates whether you want to read data from the file, write to the file, or perform both the operations.
- The members of the FileAccess enumerator are Read, ReadWrite, and Write.

www.hexaware.com | © Hexaware Technologies. All rights reserved.

FileStream Class (Contd.)

- The FileShare enumerator contains constants for controlling the kind of access that the other FileStream constructors can have to the same file.

- A typical use of this enumeration is to define whether two different applications can simultaneously read from the same file.

 | © Hexaware Technologies. All rights reserved.

Reading and Writing in the Text Files

- The Stream class is used to read from and to write data in the text files.
- It is an abstract class, which supports reading and writing bytes into it.
- If data of a file is only text, then you can use the StreamReader class and the StreamWriter class to accomplish the reading and writing tasks, respectively.

www.hexaware.com | © Hexaware Technologies. All rights reserved.

StreamReader Class

- The StreamReader class is inherited from the abstract class TextReader.
- The TextReader class represents a reader which can read a series of characters.

 | © Hexaware Technologies. All rights reserved.

StreamReader Class (Contd.)

- The following table describes some of the commonly used methods of the StreamReader class.

Methods	Description
Close	Closes the object of StreamReader class and the underlying stream, and releases any system resources associated with the reader
Peek	Returns the next available character but does not consume it
Read	Reads the next character or the next set of characters from the stream
ReadLine	Reads a line of characters from the current stream and returns data as a string
Seek	Allows the read/write position to be moved to any position within the file

www.hexaware.com | © Hexaware Technologies. All rights reserved.

StreamWriter Class

- The StreamWriter class is inherited from the abstract class TextWriter.
- The TextWriter class represents a writer, which can write a series of characters.
- The following table describes some of the commonly used methods of the StreamWriter class.

Methods	Description
Close	Closes the current StreamWriter object and the underlying stream
Flush	Clears all buffers for the current writer and causes any buffered data to be written to the underlying stream
Write	Writes to the stream
WriteLine	Writes data specified by the overloaded parameters, followed by end of line

www.hexaware.com | © Hexaware Technologies. All rights reserved.

FileInfo Class (Contd.)

- The following table describes some of the commonly used methods of the FileInfo class.

Method	Description
Create	Creates a file
AppendText	Appends a text to the file represented by the FileInfo object
Delete	Deletes a file
Open	Opens file
OpenRead	Opens a file in read-only mode

www.hexaware.com | © Hexaware Technologies. All rights reserved.

Anonymous method's

- An anonymous method is a method which doesn't contain any name.
- It is useful to create an inline method.
- It is a block of code that is used as a parameter for the delegate
- The scope of the parameters of an anonymous method is the anonymous method block
- Can use generic parameter types

 | © Hexaware Technologies. All rights reserved.

Lambda expression

- Anonymous method is a little hard to write and manage
- Hence lambda expressions were introduced in the latter C# release
- It provides a simple more concise functional syntax to write anonymous methods.
- A lambda expression is written as parameter list, followed by the +> symbol followed by an expressions block
    - Expression lambda
    - Statement lambda

 | © Hexaware Technologies. All rights reserved.

Contd..

Expression lambda

It has only one expression, with no curly brace or return statement

```
Int res = list.Find(n => n==3)
```

Statement lambda

It is a collection of statements

```
List<int> lst= list.FindAll(n =>
    {
        if(n<=5)
        {  return true; }
        else
        return false;
     }
```

www.hexaware.com | © Hexaware Technologies. All rights reserved.

Thank you

Innovative Services

Passionate Employees

Delighted Customers

Privileged & Confidential Version 1.8
© Copyright 2012 by Hexaware Technologies Limited

All rights reserved. All text included in this document is the exclusive property of Hexaware Technologies Ltd, and may not be copied, reproduced, or used in any way without the express permission in writing of Hexaware Technologies Ltd.