Programming the .NET Framework 4.5

Module 03 – Streams and File I/O

In This Chapter

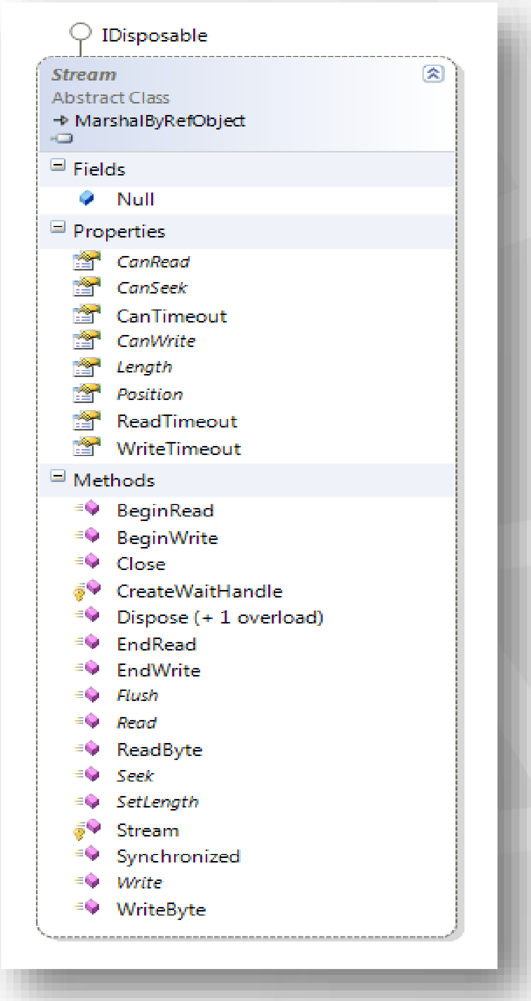
- Streams as data abstraction
- File streams
- Stream readers/writers
- The File and Directory classes
- Lab

Input and Output Abstraction

? What’s common to all input and output sources?

- Stream
  - Length, position, content-type
  - Read, write, seek
  - Open, close

Streams Direct Hierarchy



IDisposable

Stream

Abstract Class

→ MarshalByRefObject

⊟ Fields

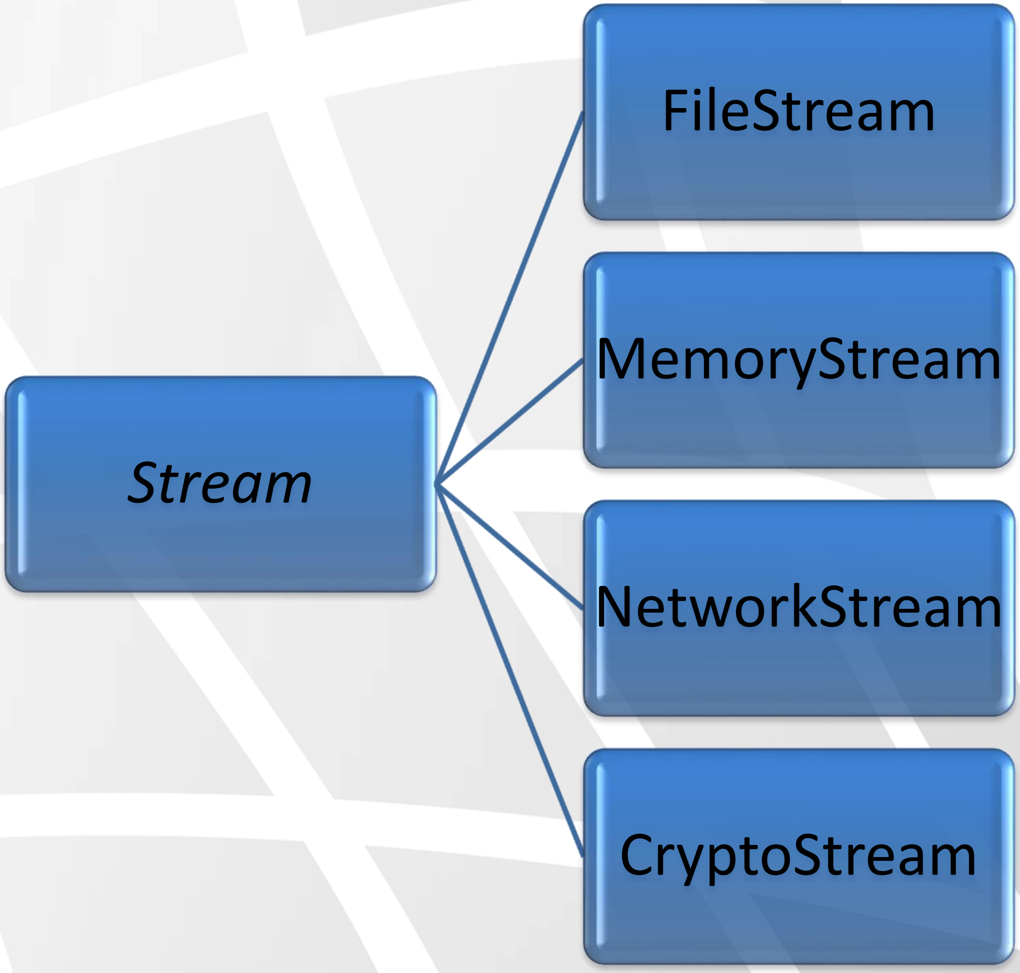
- Null

⊟ Properties

- *CanRead*
- *CanSeek*
- CanTimeout
- *CanWrite*
- *Length*
- *Position*
- ReadTimeout
- WriteTimeout

⊟ Methods

- BeginRead
- BeginWrite
- Close
- CreateWaitHandle
- Dispose (+ 1 overload)
- EndRead
- EndWrite
- *Flush*
- *Read*
- ReadByte
- *Seek*
- *SetLength*
- Stream
- Synchronized
- *Write*
- WriteByte



Decorators and Composites

- Streams can be built on top of streams
- Aggregation, inheritance

CompositeOutputStream
XorEncryptionStream

Demo

File Streams

- File streams enable file I/O

- File name
- File access: Read, write, both
- File open: Create, truncate
- File share: None, read, write, delete

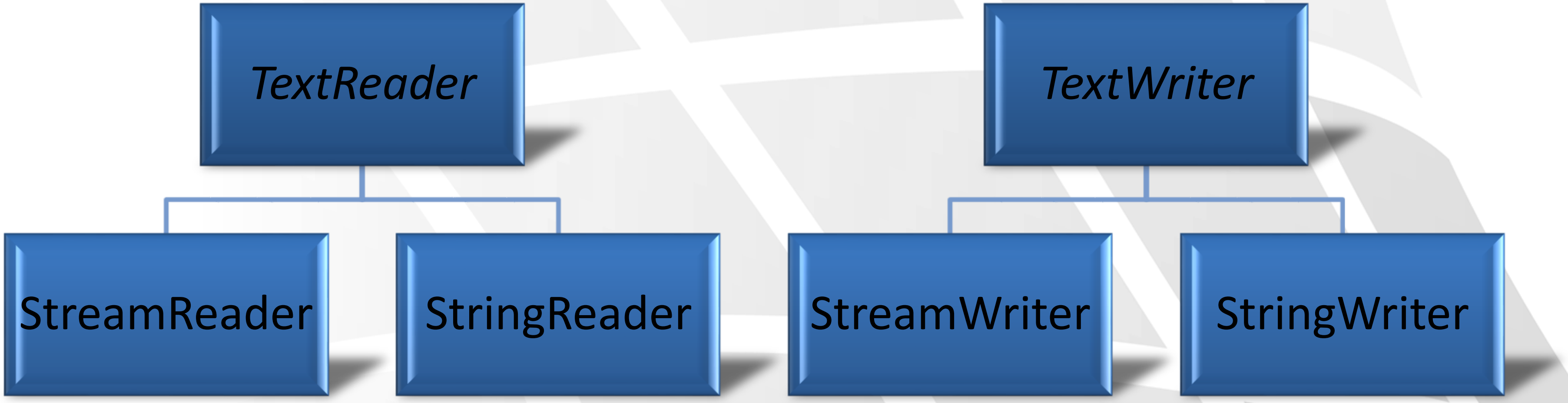
Working with File Streams

```
FileStream myFile = new FileStream("myfile.txt",
    FileMode.Create, FileAccess.ReadWrite);
string greeting = "Good afternoon";
byte[] output = Encoding.Unicode.GetBytes(greeting);
myFile.Write(output, 0, output.Length);
myFile.Close();

myFile = new FileStream("myfile.txt",
    FileMode.Open, FileAccess.Read, FileShare.Read);
byte[] input = new byte[myFile.Length];
myFile.Read(input, 0, input.Length);
myFile.Close();
Console.WriteLine("Read from file: " +
    Encoding.Unicode.GetString(input));
```

Stream Readers and Writers

- Working with streams is cumbersome
- Readers and writers simplify work
- Writers buffer data



TextReader

StreamReader

StringReader

TextWriter

StreamWriter

StringWriter

Working with Readers / Writers

```
FileStream myFile = new FileStream("myfile.txt",
FileMode.Create);
StreamWriter writer = new StreamWriter(myFile);
writer.WriteLine("From a stream writer!");
writer.Close(); //This is critical!

//Short-hand for creating a file
StreamReader reader = new StreamReader("myfile.txt");
Console.WriteLine(reader.ReadLine());
reader.Close();
```

tips

StreamWriter does not have a finalizer

Closing is critical

Binary Readers and Writers

- Oriented towards binary data
- Methods for various data types (not just strings)

Writing Binary Data

- Note that the matrix bounds are serialized

```
BinaryWriter writer = new BinaryWriter(
    new FileStream("matrix.dat", FileMode.Create));
writer.Write(matrix.GetUpperBound(0));
writer.Write(matrix.GetUpperBound(1));
for (int i = 0; i < 10; ++i)
    for (int j = 0; j < 10; ++j)
        writer.Write(matrix[i, j]);
writer.Close();
```

Reading Binary Data

- The matrix bounds are read first

```
BinaryReader reader = new BinaryReader(
    new FileStream("matrix.dat", FileMode.Open));
int dim0 = reader.ReadInt32();
int dim1 = reader.ReadInt32();
double[,] newMatrix = new double[dim0, dim1];
for (int i = 0; i < dim0; ++i)
    for (int j = 0; j < dim1; ++j)
        newMatrix[i, j] = reader.ReadDouble();
reader.Close();
```

File and Directory Classes

```
void FillTreeViewHelper(string path, TreeNode node)
{
 foreach (string dir in Directory.GetDirectories(path))
 {
    FillTreeViewHelper(dir, node.Nodes.Add(dir));
 }
 foreach (string file in Directory.GetFiles(path))
 {
    currentNode.Nodes.Add(file);
 }
}
```

tips

- Helper classes for common operations.
- File and directory information.

File System Metadata Explorer

Demo

File – static methods

```
FileStream fs = File.OpenRead("ReadMe.txt");
FileStream fs = File.OpenWrite("ReadMe.txt");
StreamWriter sw = File.AppendText("ReadMe.txt");
byte[] buffer = File.ReadAllBytes("ReadMe.txt");
string text = File.ReadAllText("ReadMe.txt");
string[] lines = File.ReadAllLines(); // do not use it
IEnumerable<string> lines =
        File.ReadLines("ReadMe.txt"); // use this one
File.WriteAllBytes("ReadMe.txt", buffer);
File.WriteAllLines("ReadMe.txt", lines);
File.WriteAllText("ReadMe.txt", text);
File.AppendAllText("ReadMe.txt", text);
File.AppendAllLines("ReadMe.txt", lines);
```

Word Count

Lab

Summary

- Streams as a data abstraction
- File streams
- Stream readers / writers
- The File and Directory classes
- Lab

Questions ?