

Reading and Writing Data Incrementally Using Streams



Jason Roberts

.NET DEVELOPER

@robertsjason dontcodetired.com



Overview



An introduction to streams

The benefits of streams

.NET class hierarchy overview

Using streams to read and write text

Selectively processing part of stream

Using streams to read and write binary data

Using BinaryReader and BinaryWriter

Specifying text encodings

Using streams to append data

Random FileStream access

MemoryStream overview



An Introduction to Streams



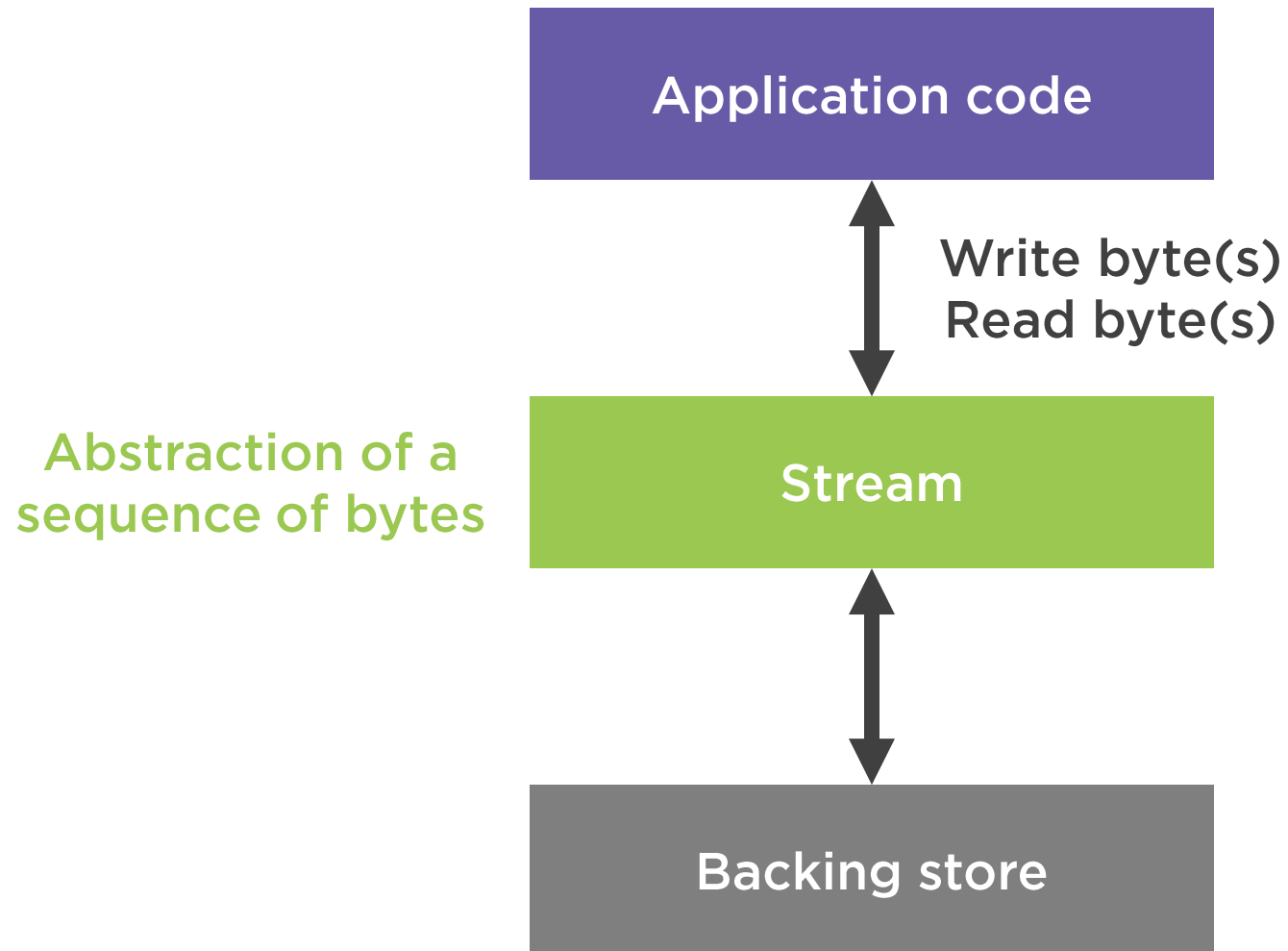
“...a stream is a sequence of bytes that you can use to read from and write to a backing store, which can be one of several storage mediums”

Microsoft Documentation

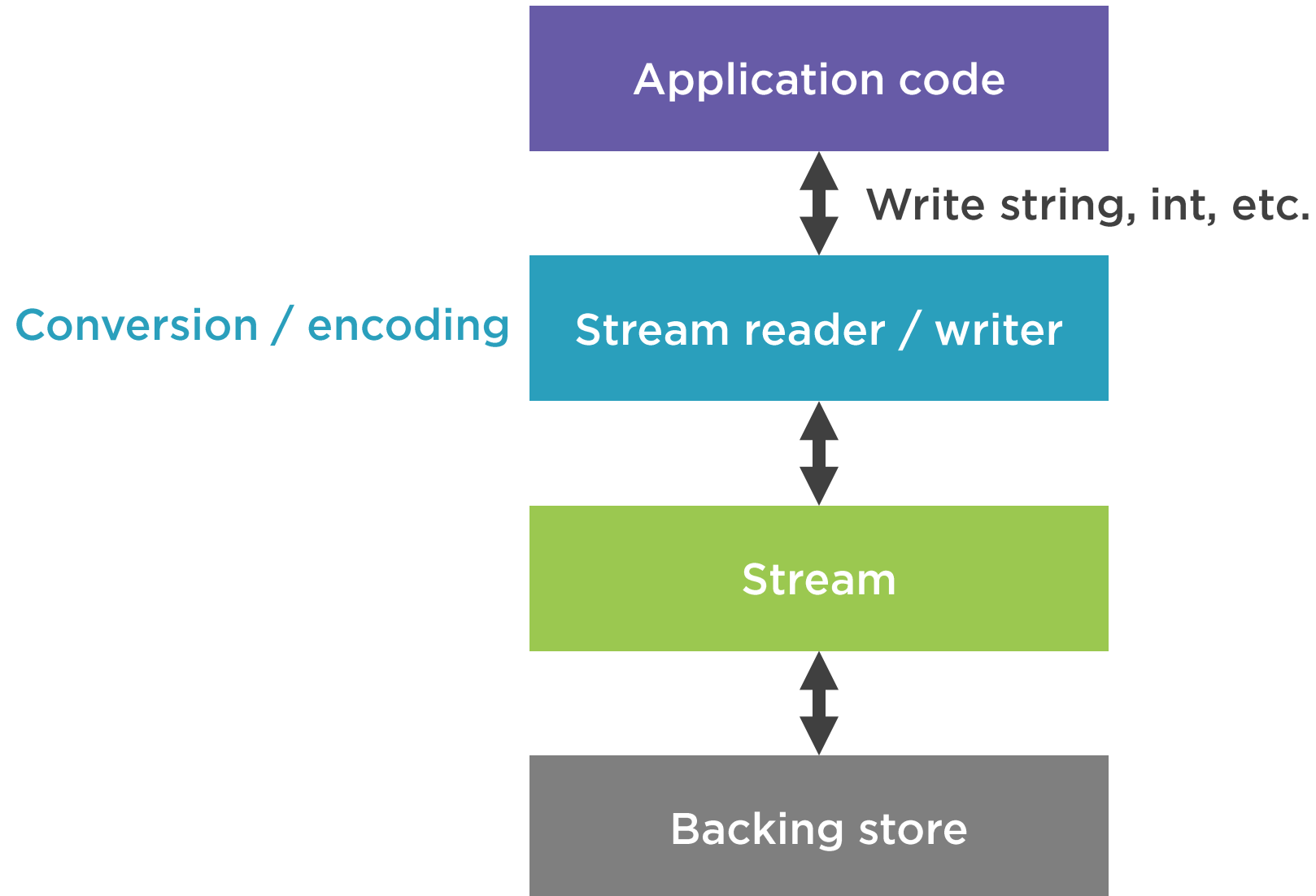
<https://docs.microsoft.com/en-us/dotnet/standard/io/>



An Introduction to Streams



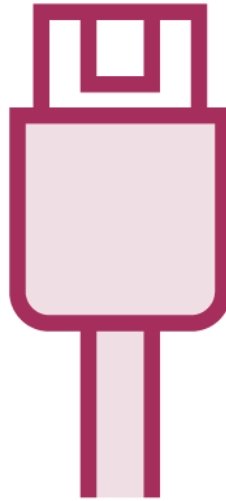
An Introduction to Streams



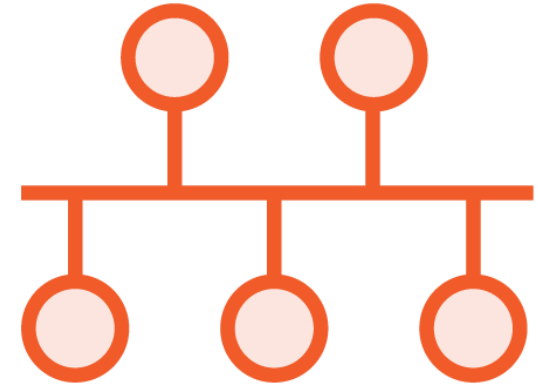
Examples of Backing Stores



Files

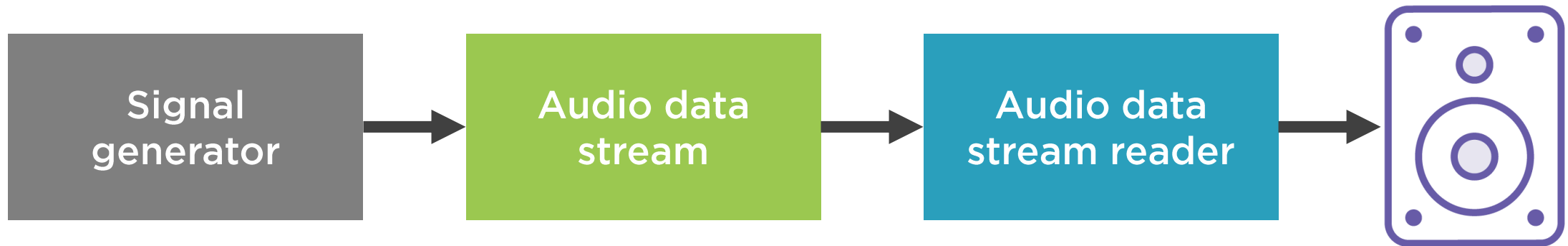


Input/output
device/hardware



TCP/IP sockets

Non-Backing Store Streams



<https://github.com/naudio/NAudio>



The Benefits of Streams

Incremental data
processing

Abstraction of
backing store

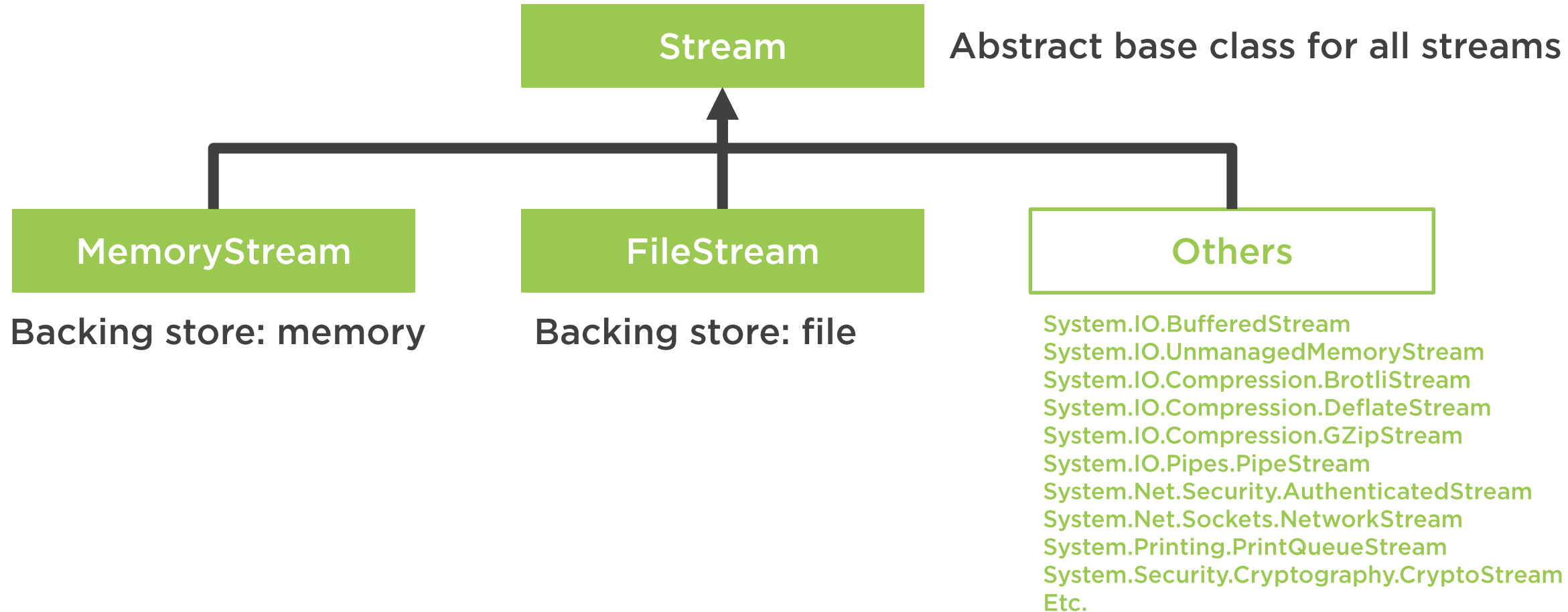
Flexibility /
control

Random access /
seeking

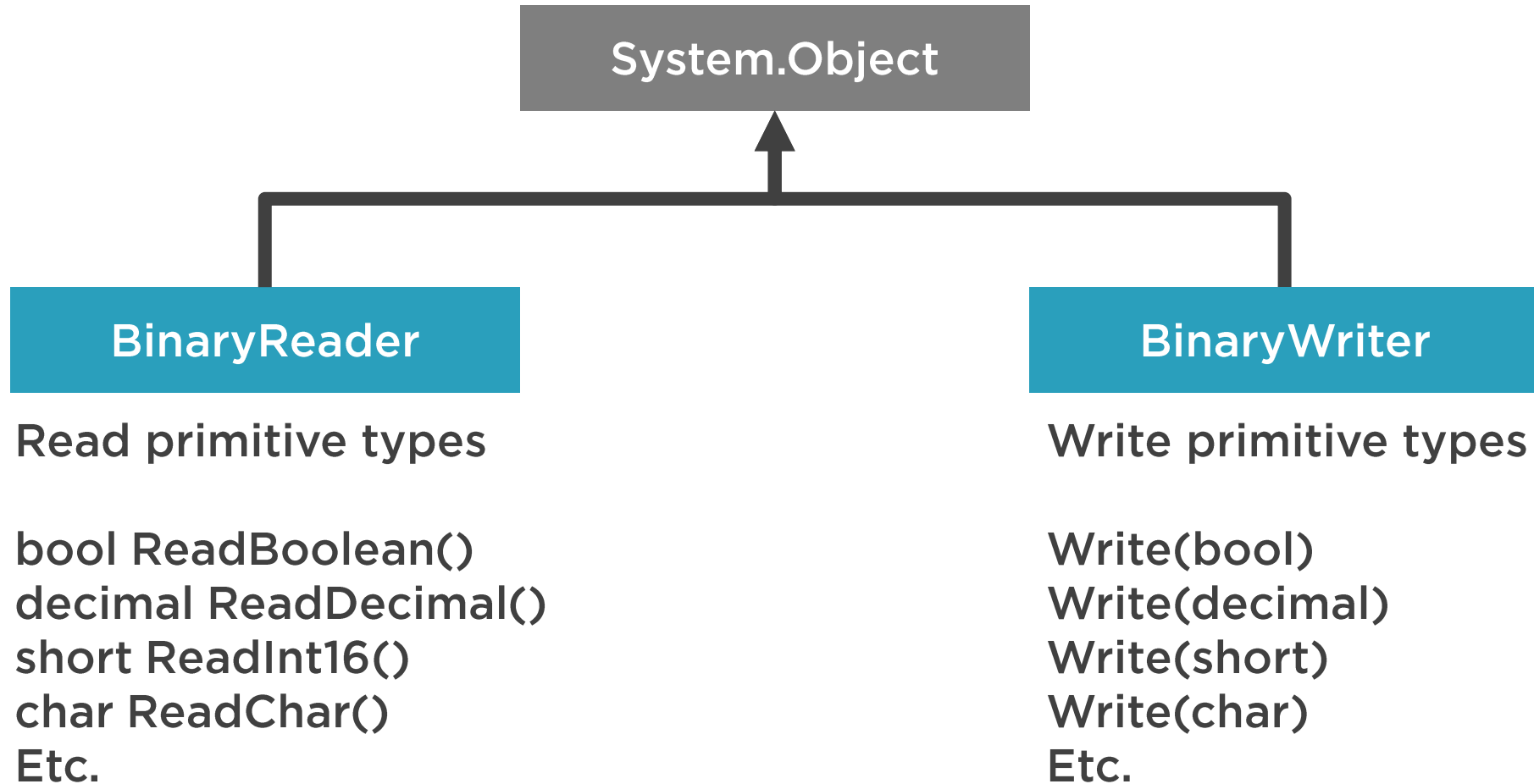
Composability /
pipelines



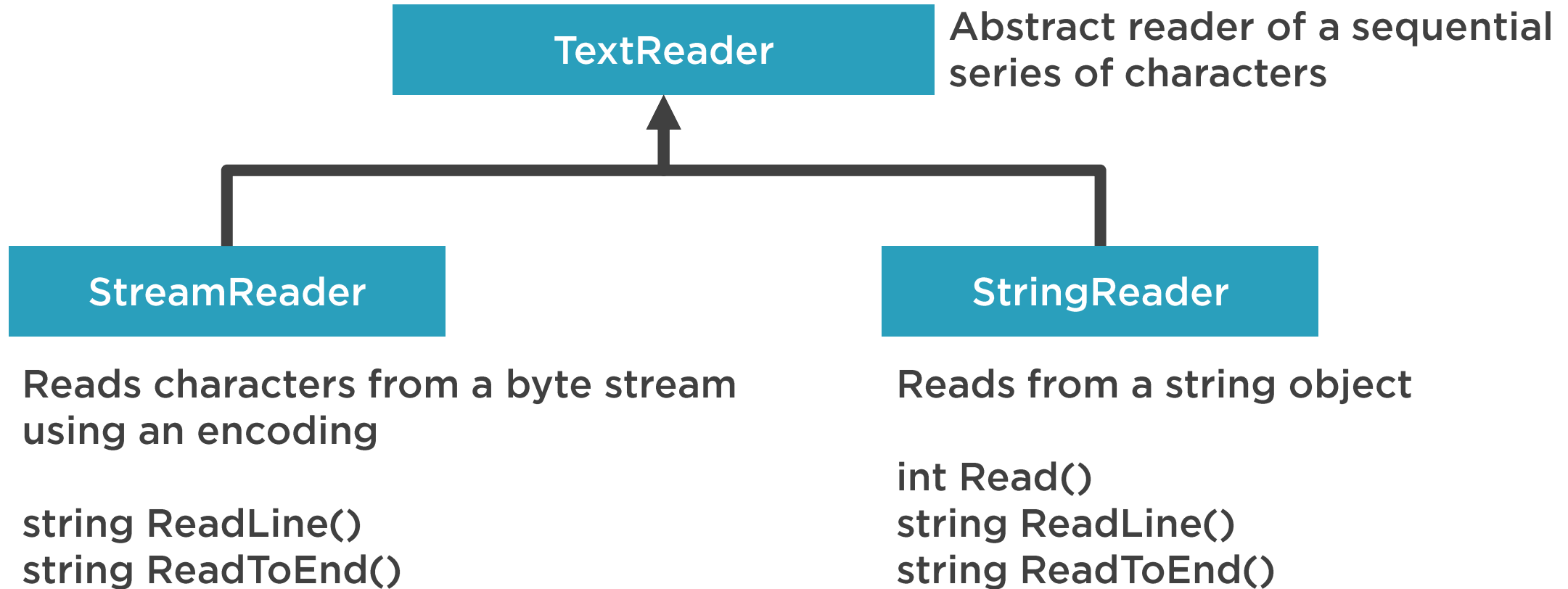
Stream Classes



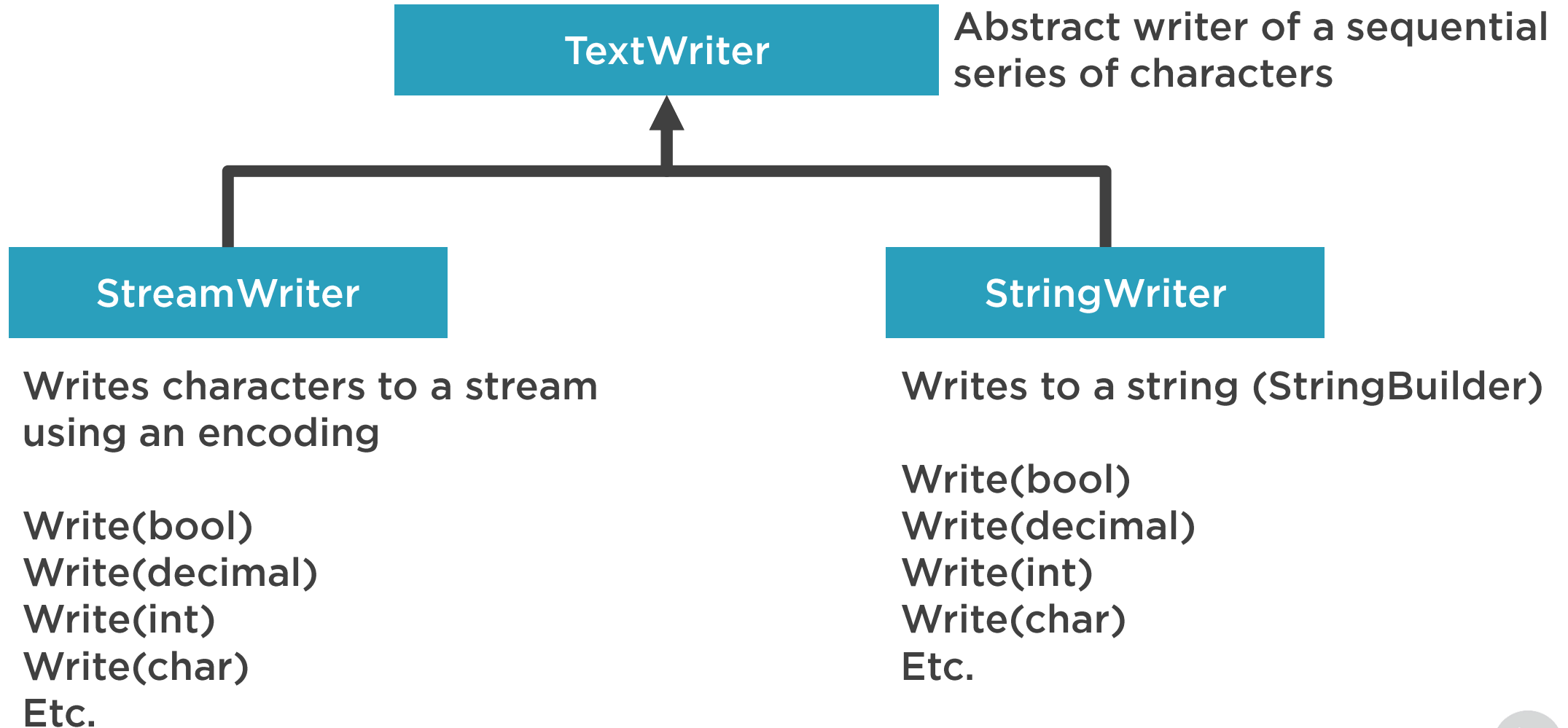
Binary Reader and Writer



TextReaders



TextWriters



Specifying Text Encodings

```
new StreamReader(inputFileStream, Encoding.UTF32)
```

```
new StreamReader(inputFileStream, new UTF8Encoding(true))
```

```
new StreamWriter(OutputFilePath, Encoding.UTF32)
```

```
new StreamWriter(OutputFilePath, new UTF8Encoding(true))
```

```
// No explicit encoding overload
```

```
// Auto-detect with UTF-8 fallback
```

```
File.OpenText(InputFilePath)
```

```
File.CreateText(OutputFilePath) // UTF-8 output encoding
```



Specifying Text Encodings

```
new BinaryReader(inputFileStream, Encoding.UTF32)
```

```
new BinaryWriter(outputFileStream, Encoding.UTF32)
```

```
// UTF-8 encoding
```

```
new BinaryReader(inputFileStream)
```

```
new BinaryWriter(outputFileStream)
```



Using Streams to Append Data

```
var streamWriter = new StreamWriter(@"C:\data.txt", true)

streamWriter.Write("Content to append...");
streamWriter.WriteLine("Content to append with new line...");
```



Using Streams to Append Data

```
FileStream fs = File.Open(@"C:\data.data", FileMode.Append);  
var binaryWriter = new BinaryWriter(fs);  
  
binaryWriter.Write(42); // append to end of file
```



Random FileStream Access

05 0F FF 5E 2A 00



```
fileStream.Position = 0; // zero-based
```

```
int firstByte = fileStream.ReadByte(); // 05
```



Random FileStream Access

05 0F FF 5E 2A 00



```
fileStream.Position = 0; // zero-based
```

```
int firstByte = fileStream.ReadByte(); // 05
```



Random FileStream Access

05 0F FF 5E 2A 00



```
fileStream.Position = 2;
```



Random FileStream Access

05 0F FF 5E 2A 00



```
fileStream.Position = 2;
```

```
int thirdByte = fileStream.ReadByte(); // FF
```



Random FileStream Access

05 0F FF 5E 2A 00



```
fileStream.Position = 2;
```

```
int thirdByte = fileStream.ReadByte(); // FF
```



Random FileStream Access

05 0F FF 5E 2A 00



```
fileStream.Seek(2, SeekOrigin.Begin);
```



Random FileStream Access

05 0F FF 5E 2A 00



```
fileStream.Seek(2, SeekOrigin.Begin);
```

```
thirdByte = fileStream.ReadByte(); // FF
```



Random FileStream Access

05 0F FF 5E 2A 00



```
fileStream.Seek(2, SeekOrigin.Begin);
```

```
thirdByte = fileStream.ReadByte(); // FF
```



Random FileStream Access

05 0F FF 5E 2A 00



```
fileStream.Seek(1, SeekOrigin.Current);
```



Random FileStream Access

05 0F FF 5E 2A 00



```
fileStream.Seek(1, SeekOrigin.Current);  
int fifthByte = fileStream.ReadByte(); // 2A
```



Random FileStream Access

05 0F FF 5E 2A 00



```
fileStream.Seek(1, SeekOrigin.Current);  
int fifthByte = fileStream.ReadByte(); // 2A
```



Random FileStream Access

05 0F FF 5E 2A 00



```
fileStream.Seek(-3, SeekOrigin.End);
```



Random FileStream Access

05 0F FF 5E 2A 00



```
fileStream.Seek(-3, SeekOrigin.End);
```

```
int threeFromEnd = fileStream.ReadByte(); // 5E
```



Random FileStream Access

05 0F FF 5E 2A 00



```
fileStream.Seek(-3, SeekOrigin.End);
```

```
int threeFromEnd = fileStream.ReadByte(); // 5E
```



Not all streams support
random access / seeking.

Stream.CanSeek



MemoryStream Overview

```
using (var memoryStream = new MemoryStream())  
using (var memoryStreamWriter = new StreamWriter(memoryStream))  
using (var fileStream = new FileStream(@"C:\data.txt", FileMode.Create))  
{  
    memoryStreamWriter.WriteLine("Line 1");  
    memoryStreamWriter.WriteLine("Line 2");  
  
    // Ensure everything written to memory stream  
    memoryStreamWriter.Flush();  
  
    memoryStream.WriteTo(fileStream);  
}
```



Summary



An introduction to streams

The benefits of streams

.NET class hierarchy overview

`new StreamReader(inputFileStream)`

`File.OpenText(InputFilePath)`

Selectively processing part of stream

Using streams to read and write binary data

`outputFileStream.WriteByte(...)`

`new BinaryReader(inputFileStream)`

Text encodings & appending data

Random access & MemoryStreams



Up Next:

Reading and Writing CSV Data

