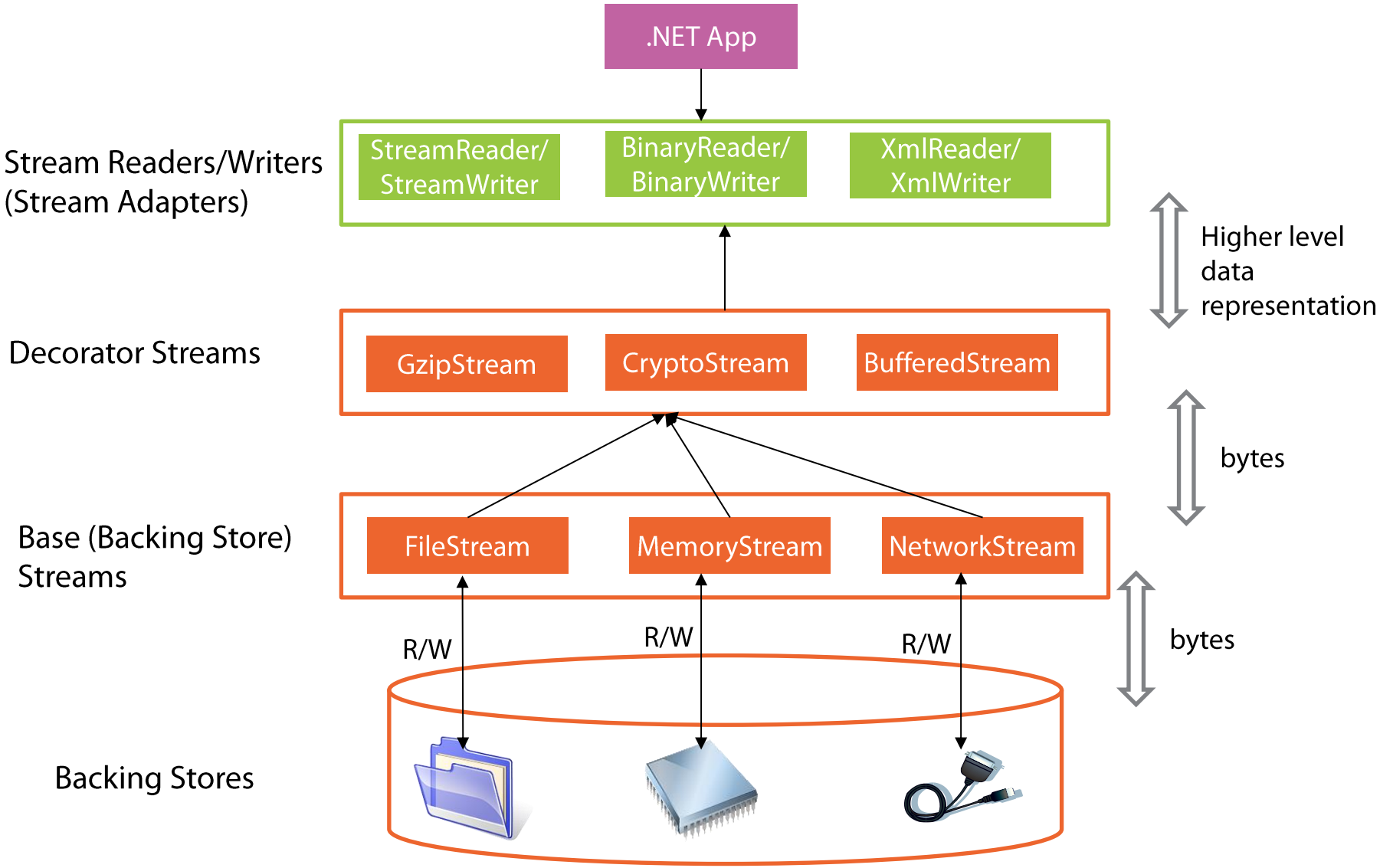
Stream Adapters

Mohamad Halabi
Microsoft Integration MVP
@mohamadhalabi

pluralsight
hardcore dev and IT training

The Overall Architecture



Stream Adapters

- **Text adapters: strings and character-based data**
  - StreamReader/StreamWriter inherit TextReader/TextWriter respectively
- **Binary adapters: primitive types (such int, bool, float) and string**
  - BinaryReader/BinaryWriter abstract classes
- **XML adapters: XML data**
  - XmlReader/XmlWriter

StreamReader

- **Reads bytes from a backing store stream and transforms them into characters**

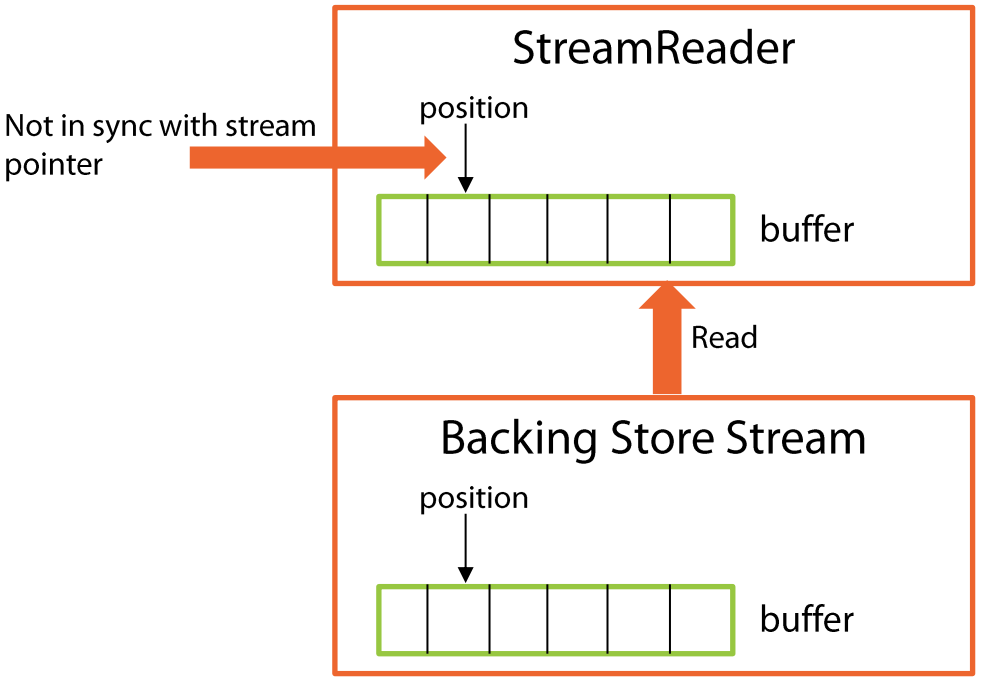
Member	Description
Peek	Returns but does not consume next character StreamReader pointer position is not changed Returns an integer (can be cast back to character) Returns -1 for end of stream or unseekable stream
Read	Reads next character Advances pointer position of StreamReader by one character Returns an integer to cast back to character representation Returns -1 if end of stream is not reached
Read (char[] buffer, int index, int count)	Reads maximum number of characters identified by "count" Reads number of read characters or 0 if end of stream
ReadLine	Returns a line of characters until "\n", "\r", or "\r\n" Line returned does not contain terminating characters Returns null if end of stream

StreamReader

Member	Description
ReadToEnd	Read all characters (from current position) until end of stream Empty string (“”) is returned if position is already at end of stream
Null	Static field. Returns StreamReader around an empty stream
Synchronized	Static method. Inherited from TextReader Creates a thread-safe wrapper around StreamReader

StreamReader Buffer

- **StreamReader has an internal buffer**
  - More data can be read that what is requested
  - Future reads are then served from buffer and not from backing store stream



Call DiscardBufferedData() to reset internal buffer

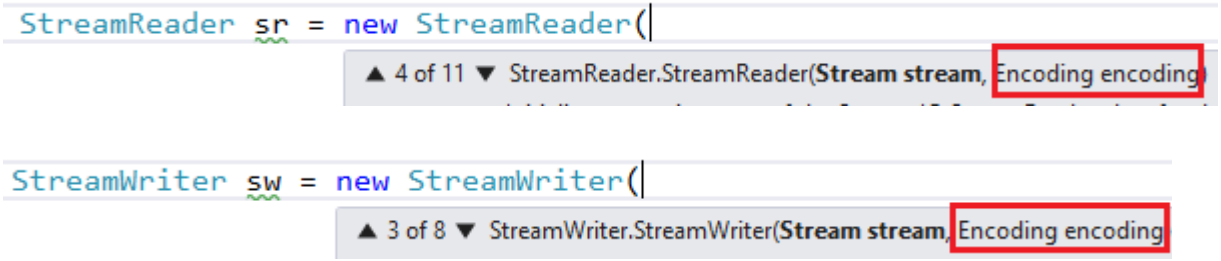
StreamWriter

- **Writes characters into the backing store stream**

Member	Description
Write (…)	Overloaded to write specific types Write(Char): writes a character Write(String): writes a string Write(Decimal): writes text representation of a decimal Write(Int32): writes text representation of a 4 byte signed integer
WriteLine (…)	Multiple overloads Write characters, strings, or string representations of numerical values Different than Write, in that it writes a line terminator at the end
Flush	Clears StreamWriter buffer and writes buffered data to backing store stream Flushes internal buffer of backing store stream (if supported, ex: FileStream)
AutoFlush	Causes StreamWriter to Flush when Write is called

Encoding

- Converting from bytes from/to characters requires encoding
- You can pass a System.Text Encoding parameter to StreamReader and StreamWriter constructors



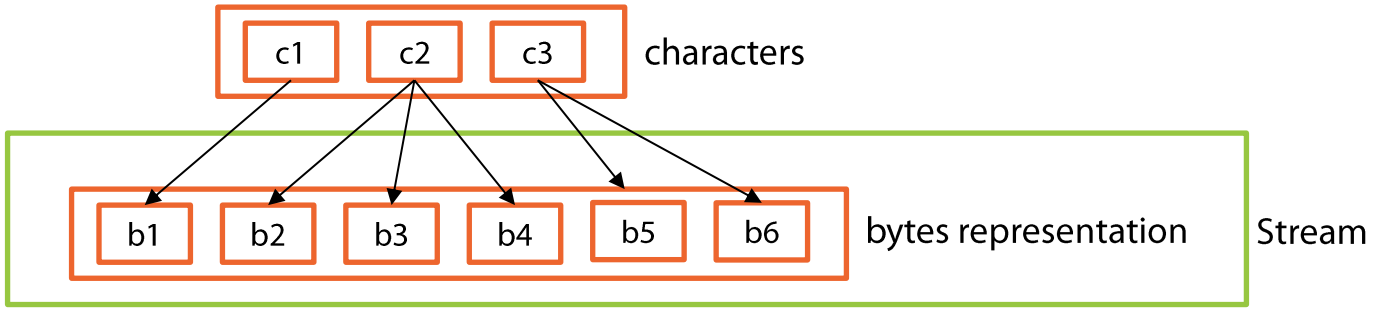
- Default encoding is UTF-8

ASCII vs. Unicode

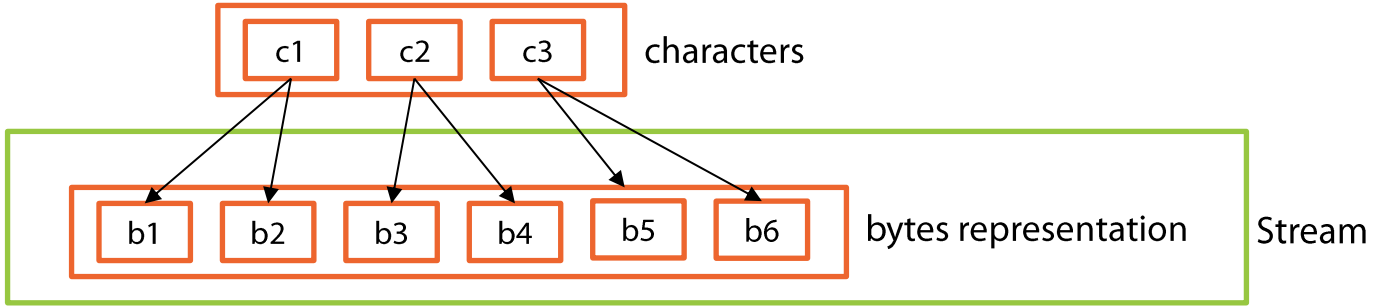
- ASCII efficiently uses 1 byte for each character
- ASCII can represent only the first 127 characters of the Unicode set
- ASCII cannot represent special and non-English characters
- For such special and non-English characters use Unicode encoding

Unicode Bytes Allocation

- **UTF-8:**
  - represents the first 127 characters with 1 byte each (for ASCII compatibility)
  - Remaining characters encoded into variable number of bytes



- **UTF-16:**
  - C# char type (16-bits) will always be represented by 2 bytes



Binary Adapters

- **BinaryReader and BinaryWriter read and write:**
  - Primitive data types (bool, byte, char, decimal, float, double, short, int, long, sbyte, ushort, uint, and ulong)
  - Array of primitive types
  - Strings
- **Primitive types are saved in a compact binary format thus saving storage**

Usage Scenarios

- **Reading and writing primitive types from/into a stream**
- **Reading and writing binary files such as JPEG**
- **How about reading and writing strings? So how is this different than StreamReader and StreamWriter?**
  - Text adapters perform better when dealing with strings (up to 40%)
  - Because they inherit TextReader/TextWriter, Text adapters seems to be optimized for string reading/writing

BinaryWriter

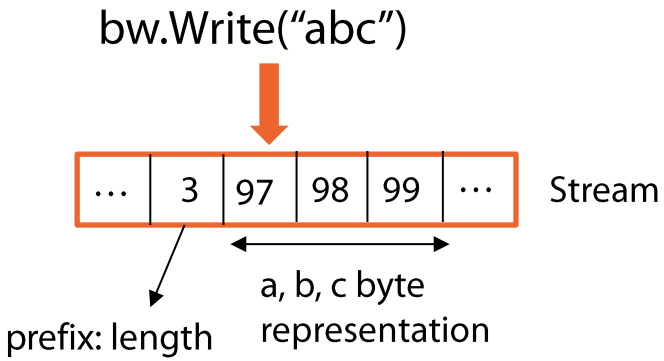
Member	Description
Flush	Calls Flush of the backing store stream
Seek	Internally calls the Seek method of the backing store stream
Write (…)	Overloaded many times for primitive types - Write(Byte[]): writes a byte array to the stream - Write(Double): writes an eight-byte floating-point value to the stream - Write(Int16): writes a two-byte signed integer to the stream Data is not held in an internal buffer Data immediately written to stream and stream position is advanced

BinaryReader

Member	Description
PeekChar	Similar to Peek for StreamReader Returns next available character without advancing position Returns -1 if end of stream or unseekable stream
ReadXXX (primitive types)	Set of methods for reading primitive types - ReadBoolean - ReadByte - ReadDecimal
Read(Byte[], Int32, Int32)	Reads a specified number of bytes
ReadString	Dedicated for string reading

Write(String) and ReadString

- **BinaryWriter.Write(String)**
  - Writes to the stream the length of the string as a prefix



- **BinaryReader.ReadString**
  - First reads the length
  - Then reads number of bytes indicated by the length
    - Ex: reads "3" then thread 3 bytes 97,98, and 99

Implication

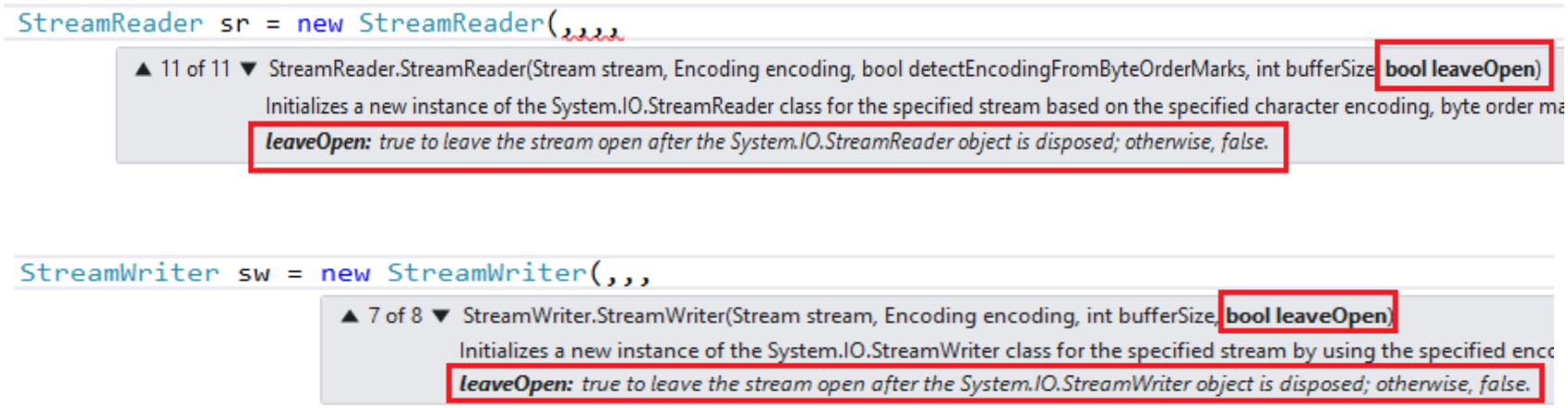
- **BinaryReader.ReadString should be used to read strings written with BinaryWriter.Write(String)**
  - ReadString understands the format written using Write(String)
- **Use BinaryReader.ReadChars if you know characters sequence length**

Xml Adapters

- **XmlReader and XmlWriter are high performance adapters**
- **Forward-only cursors to read/write an XML stream**
  - Forward-only: navigation can be done onwards only
- **Xml adapters are used when the stream contains XML data**
- **Knowledge of XML is required**

Closing Stream Adapters

- Closing a stream adapter closes the backing store stream
- Be careful not to reuse a stream after the corresponding adapter is closed
- In .NET 4.5:



Summary

- **Stream adapters are used to work with high level data representation**
- **Text adapters: read/write string data**
  - StreamReader and StreamWriter
- **Binary adapters: read/write primitive types and strings**
  - BinaryReader and BinaryWriter
  - Text adapters are optimized for strings
- **Xml adapters: read/write XML data**
  - XmlReader and XmlWriter
- **Closing a stream adapter closes the backing store stream**
  - In .NET 4.5 StreamReader and StreamWriter accept a parameter to change this behavior