# Working with Streams

# Project Set Up

## Setup Steps for Visual Studio

From within VS (using point and click):

* Open the WorkingWithDataAndFiles solution
* Add console project called streams-demo.csproj to the solution

## Setup Steps for Visual Studio Code

From Command Prompt:

* Add a new console project to the demos solution by typing:

dotnet new console --output streams-demo

dotnet sln add streams-demo

## Now do the following regardless of project type:

* Replace the Program.cs file’s content with the following:

// Include all relevant namespaces

using System.IO.Compression;

using System.Security.Cryptography;

using System.Text;

namespace streams\_demo

{

internal class Program

{

static void Main(string[] args)

{

// Compress a string into a byte[] using a GZip stream

string originalText =

"It is a period of civil war. Rebel spaceships, striking from a hidden base, " +

"have won their first victory against the evil Galactic Empire. During the battle, " +

"Rebel spies managed to steal secret plans to the Empire’s ultimate weapon, " +

"the DEATH STAR, an armoured space station with enough power to destroy an entire planet. " +

"Pursued by the Empire’s sinister agents, Princess Leia races home aboard her starship, " +

"custodian of the stolen plans that can save her people and restore freedom to the galaxy….";

byte[] compressedBytes;

compressedBytes = CompressIntoByteArray(originalText);

Console.WriteLine(compressedBytes.Length);

Console.WriteLine($"compressed bytes: ");

foreach (byte b in compressedBytes)

{

Console.Write($"{b},");

}

// Uncompress a byte stream back into a string (TO BE DONE IN THE LAB)

string uncompressedText = DecompressByteArrayIntoString(compressedBytes);

System.Console.WriteLine($"\n\n{uncompressedText}");

//Encryption

string message = originalText;

string key = "1234567890123456";

Console.WriteLine($"\nOriginal Message: {message}");

string encryptedString = EncryptString(key, message);

Console.WriteLine($"\nEncrypted Message: {encryptedString}");

Console.WriteLine($"Encrypted Message length (in chars - 2 bytes to a char): {encryptedString.Length}");

compressedBytes = CompressIntoByteArray(encryptedString);

Console.WriteLine($"\nencrypted and compressed message length (in bytes): {compressedBytes.Length}");

uncompressedText = DecompressByteArrayIntoString(compressedBytes); // Currently does nothing. To be completed in lab

System.Console.WriteLine($"\nencrypted and decompressed message length (in chars - 2 bytes to a char): {uncompressedText.Length}");

System.Console.WriteLine($"encrypted and decompressed text: {uncompressedText}");

//N.B. in the LAB you need to pass the uncompressedText variable in place of encryptedString

string decryptedString = DecryptString(key, encryptedString);

Console.WriteLine($"\nDecrypted Message: {decryptedString}");

}

static Byte[] CompressIntoByteArray(string message)

{

byte[] originalBytes = System.Text.ASCIIEncoding.ASCII.GetBytes(message);

//Console.WriteLine($"Length before compression: {originalBytes.Length}");

using MemoryStream inStream = new MemoryStream(originalBytes);

using MemoryStream outStream = new MemoryStream();

using GZipStream zipStream = new GZipStream(outStream, CompressionMode.Compress);

inStream.CopyTo(zipStream);

zipStream.Close();

byte[] compressedBytes = outStream.ToArray();

return compressedBytes;

}

static string DecompressByteArrayIntoString(byte[] compressedBytes)

{

string uncompressedText = "TO BE COMPLETED IN LAB";

return uncompressedText;

}

// Function to encrypt string using AES encryption with a string key

static string EncryptString(string key, string plainText)

{

byte[] iv = new byte[16];

byte[] array;

using (Aes aes = Aes.Create())

{

aes.Key = Encoding.UTF8.GetBytes(key);

aes.IV = iv;

ICryptoTransform encryptor = aes.CreateEncryptor(aes.Key, aes.IV);

using (MemoryStream memoryStream = new MemoryStream())

{

using (CryptoStream cryptoStream = new CryptoStream(memoryStream, encryptor, CryptoStreamMode.Write))

{

using (StreamWriter streamWriter = new StreamWriter(cryptoStream))

{

streamWriter.Write(plainText);

}

array = memoryStream.ToArray();

}

}

}

return Convert.ToBase64String(array);

}

// Function to decrypt a string, using string key

static string DecryptString(string key, string cipherText)

{

byte[] iv = new byte[16];

byte[] buffer = Convert.FromBase64String(cipherText);

using (Aes aes = Aes.Create())

{

aes.Key = Encoding.UTF8.GetBytes(key);

aes.IV = iv;

ICryptoTransform decryptor = aes.CreateDecryptor(aes.Key, aes.IV);

using (MemoryStream memoryStream = new MemoryStream(buffer))

{

using (CryptoStream cryptoStream = new CryptoStream(memoryStream, decryptor, CryptoStreamMode.Read))

{

using (StreamReader streamReader = new StreamReader(cryptoStream))

{

return streamReader.ReadToEnd();

}

}

}

}

}

}

}

* Review the code and try to work out what it’s doing.
* Build and run the code and confirm it behaves as you expected.