```
      try
      {
        writer2.WriteLine("Welcome, .NET!");
      }
      catch(Exception ex)
      {
        WriteLine($"{ex.GetType()} says {ex.Message}");
      }
    } // automatically calls Dispose if the object is not null
  } // automatically calls Dispose if the object is not null
```

9.2.4 压缩流

XML 比较冗长，所以相比纯文本会占用更多的字节空间。可以使用一种名为 GZIP 的常见压缩算法来压缩 XML。

(1) 导入以下名称空间：

```
using System.IO.Compression;
```

(2) 添加 WorkWithCompression 方法，该方法将使用 GZipSteam 的实例创建压缩文件，其中包含与之前相同的 XML 元素，然后在读取压缩文件并将其输出到控制台时对其进行解压，如下所示：

```
static void WorkWithCompression()
{
  // compress the XML output
  string gzipFilePath = Combine(
    CurrentDirectory, "streams.gzip");

  FileStream gzipFile = File.Create(gzipFilePath);

  using (GZipStream compressor = new GZipStream(
    gzipFile, CompressionMode.Compress))
  {
    using (XmlWriter xmlGzip = XmlWriter.Create(compressor))
    {
      xmlGzip.WriteStartDocument();
      xmlGzip.WriteStartElement("callsigns");
      foreach (string item in callsigns)
      {
        xmlGzip.WriteElementString("callsign", item);
      }

      // the normal call to WriteEndElement is not necessary
      // because when the XmlWriter disposes, it will
      // automatically end any elements of any depth
    }
  } // also closes the underlying stream

  // output all the contents of the compressed file
  WriteLine("{0} contains {1:N0} bytes.",
    gzipFilePath, new FileInfo(gzipFilePath).Length);
  WriteLine($"The compressed contents:");
  WriteLine(File.ReadAllText(gzipFilePath));

  // read a compressed file
```