

**Second choice is the fastest**

namespace OrderProcessor

{

class Program

{

static void Main(string[] args)

{

string fileName = "Orders.xml";

parseWithXmlDocument(fileName);

parseWithXmlReader(fileName);

parseWithXDocument(fileName);

Console.Write("Press <Enter> to end");

Console.ReadLine();

}

private static void Report (decimal totalOrderPrice, decimal totalFreightCost, decimal orderQty, int Elapsed)

{

Console.WriteLine("Total Order Price: {0:C}", totalOrderPrice);

Console.WriteLine("Total Freight Cost: {0:C}", totalFreightCost);

Console.WriteLine("Average Freight Cost per Order: {0:C}", totalFreightCost / orderQty);

Console.WriteLine("Time to Parse XDocument: {0}", Elapsed);

}

private static void parseWithXmlDocument(string fileName)

{

decimal totalOrderPrice = 0;

decimal totalFreightCost = 0;

decimal orderQty = 0;

var sw = new Stopwatch();

sw.Start();

var doc = new XmlDocument();

doc.Load(fileName);

orderQty = doc.SelectNodes("//Order").Count;

foreach (XmlNode node in doc.SelectNodes("//LineItem"))

{

var freight = decimal.Parse(node.Attributes["Freight"].Value);

var linePrice = decimal.Parse(node.Attributes["Price"].Value) \* decimal.Parse(node.Attributes["Qty"].Value);

totalOrderPrice += linePrice + freight;

totalFreightCost += freight;

}

sw.Stop();

Report(totalOrderPrice , totalFreightCost , orderQty, sw.Elapsed);

}

private static void parseWithXmlReader(string fileName)

{

decimal totalOrderPrice = 0;

decimal totalFreightCost = 0;

decimal orderQty = 0;

var sw = new Stopwatch();

sw.Start();

using (var xmlReader = new XmlTextReader(fileName))

{

while (xmlReader.Read())

{

if (xmlReader.NodeType == XmlNodeType.Element)

{

switch (xmlReader.Name)

{

case "Order":

++orderQty;

break;

case "LineItem":

var qty = decimal.Parse(xmlReader.GetAttribute("Qty"));

var price = decimal.Parse(xmlReader.GetAttribute("Price"));

var freight = decimal.Parse(xmlReader.GetAttribute("Freight"));

totalFreightCost += freight;

totalOrderPrice += (qty \* price) + freight;

break;

}

}

}

}

sw.Stop();

Report(totalOrderPrice , totalFreightCost , orderQty, sw.Elapsed);

}

private static void parseWithXDocument(string fileName)

{

decimal totalOrderPrice = 0;

decimal totalFreightCost = 0;

decimal orderQty = 0;

var sw = new Stopwatch();

sw.Start();

var doc = XDocument.Load(fileName);

foreach (var order in doc.Descendants("Order"))

{

++orderQty;

foreach (var lineItem in order.Descendants("LineItem"))

{

var qty = (decimal)lineItem.Attribute("Qty");

var price = (decimal)lineItem.Attribute("Price");

var freight = (decimal)lineItem.Attribute("Freight");

totalFreightCost += freight;

totalOrderPrice += (qty \* price) + freight;

}

}

sw.Stop();

Report(totalOrderPrice , totalFreightCost , orderQty, sw.Elapsed);

}

}

}

**Orders.xml**

<Orders>

<Order OrderNumber="SO43659">

<LineItem Line="1" PID="349" Qty="1" Price="2024.9940" Freight="50.6249" />

<LineItem Line="2" PID="350" Qty="3" Price="2024.9940" Freight="151.8746" />

<LineItem Line="3" PID="351" Qty="1" Price="2024.9940" Freight="50.6249" />

<LineItem Line="4" PID="344" Qty="1" Price="2039.9940" Freight="50.9999" />

<LineItem Line="5" PID="345" Qty="1" Price="2039.9940" Freight="50.9999" />

<LineItem Line="6" PID="346" Qty="2" Price="2039.9940" Freight="101.9997" />

<LineItem Line="7" PID="347" Qty="1" Price="2039.9940" Freight="50.9999" />

<LineItem Line="8" PID="229" Qty="3" Price="28.8404" Freight="2.1630" />

<LineItem Line="9" PID="235" Qty="1" Price="28.8404" Freight="0.7210" />

<LineItem Line="10" PID="218" Qty="6" Price="5.7000" Freight="0.8550" />

<LineItem Line="11" PID="223" Qty="2" Price="5.1865" Freight="0.2593" />

<LineItem Line="12" PID="220" Qty="4" Price="20.1865" Freight="2.0187" />

</Order>

<Order OrderNumber="SO43660">

<LineItem Line="1" PID="326" Qty="1" Price="419.4589" Freight="10.4865" />

<LineItem Line="2" PID="319" Qty="1" Price="874.7940" Freight="21.8699" />

</Order>

<Order OrderNumber="SO43661">

<LineItem Line="1" PID="300" Qty="1" Price="809.7600" Freight="20.2440" />

<LineItem Line="2" PID="296" Qty="1" Price="714.7043" Freight="17.8676" />

<LineItem Line="3" PID="304" Qty="2" Price="714.7043" Freight="35.7352" />

<LineItem Line="4" PID="223" Qty="4" Price="5.1865" Freight="0.5187" />

<LineItem Line="5" PID="232" Qty="4" Price="28.8404" Freight="2.8840" />

<LineItem Line="6" PID="293" Qty="2" Price="722.5949" Freight="36.1297" />

<LineItem Line="7" PID="348" Qty="3" Price="2024.9940" Freight="151.8746" />

<LineItem Line="8" PID="351" Qty="2" Price="2024.9940" Freight="101.2497" />

<LineItem Line="9" PID="220" Qty="2" Price="20.1865" Freight="1.0093" />

<LineItem Line="10" PID="292" Qty="2" Price="818.7000" Freight="40.9350" />

<LineItem Line="11" PID="349" Qty="4" Price="2024.9940" Freight="202.4994" />

<LineItem Line="12" PID="346" Qty="2" Price="2039.9940" Freight="101.9997" />

<LineItem Line="13" PID="235" Qty="2" Price="28.8404" Freight="1.4420" />

<LineItem Line="14" PID="350" Qty="2" Price="2024.9940" Freight="101.2497" />

<LineItem Line="15" PID="215" Qty="5" Price="20.1865" Freight="2.5233" />

</Order>

<Order OrderNumber="SO43662">

<LineItem Line="1" PID="330" Qty="3" Price="419.4589" Freight="31.4594" />

<LineItem Line="2" PID="342" Qty="5" Price="419.4589" Freight="52.4324" />

<LineItem Line="3" PID="272" Qty="2" Price="183.9382" Freight="9.1969" />

<LineItem Line="4" PID="315" Qty="4" Price="874.7940" Freight="87.4794" />

<LineItem Line="5" PID="262" Qty="3" Price="183.9382" Freight="13.7954" />

<LineItem Line="6" PID="326" Qty="5" Price="419.4589" Freight="52.4324" />

<LineItem Line="7" PID="332" Qty="3" Price="419.4589" Freight="31.4594" />

<LineItem Line="8" PID="338" Qty="2" Price="419.4589" Freight="20.9729" />

<LineItem Line="9" PID="314" Qty="1" Price="2146.9620" Freight="53.6741" />

<LineItem Line="10" PID="317" Qty="1" Price="874.7940" Freight="21.8699" />

<LineItem Line="11" PID="328" Qty="3" Price="419.4589" Freight="31.4594" />

<LineItem Line="12" PID="275" Qty="1" Price="356.8980" Freight="8.9225" />

<LineItem Line="13" PID="319" Qty="6" Price="874.7940" Freight="131.2191" />

<LineItem Line="14" PID="270" Qty="1" Price="183.9382" Freight="4.5985" />

<LineItem Line="15" PID="253" Qty="3" Price="178.5808" Freight="13.3936" />

<LineItem Line="16" PID="310" Qty="1" Price="2146.9620" Freight="53.6741" />

<LineItem Line="17" PID="322" Qty="3" Price="419.4589" Freight="31.4594" />

<LineItem Line="18" PID="264" Qty="1" Price="183.9382" Freight="4.5985" />

<LineItem Line="19" PID="276" Qty="1" Price="356.8980" Freight="8.9225" />

<LineItem Line="20" PID="285" Qty="1" Price="178.5808" Freight="4.4645" />

<LineItem Line="21" PID="334" Qty="3" Price="419.4589" Freight="31.4594" />

<LineItem Line="22" PID="316" Qty="1" Price="874.7940" Freight="21.8699" />

</Order>

<Order OrderNumber="SO43663">

<LineItem Line="1" PID="322" Qty="1" Price="419.4589" Freight="10.4865" />

</Order>

<Order OrderNumber="SO43664">

<LineItem Line="1" PID="345" Qty="1" Price="2039.9940" Freight="50.9999" />

<LineItem Line="2" PID="348" Qty="4" Price="2024.9940" Freight="202.4994" />

<LineItem Line="3" PID="229" Qty="1" Price="28.8404" Freight="0.7210" />

<LineItem Line="4" PID="235" Qty="1" Price="28.8404" Freight="0.7210" />

<LineItem Line="5" PID="350" Qty="2" Price="2024.9940" Freight="101.2497" />

<LineItem Line="6" PID="344" Qty="3" Price="2039.9940" Freight="152.9996" />

<LineItem Line="7" PID="346" Qty="1" Price="2039.9940" Freight="50.9999" />

<LineItem Line="8" PID="351" Qty="1" Price="2024.9940" Freight="50.6249" />

</Order>

</Order>

■ The *XmlDocument* class provides in-memory, random, read-write access to XML nodes.

■ The *XmlReader* class provides fast-streaming, forward-only, read-only access to XML nodes.

■ The *XmlDocument* class is easier to use, whereas the *XmlReader* class is faster.

■ The *XmlDocument* class enables you to retrieve XML nodes by using the element name.

■ The *XmlDocument* class enables you to retrieve XML nodes by using an XPath query.

To run several queries for data from a xml file based on filter criteria the user will be entering for a particular purpose

XmlDocument suports random access

Every day, you recive hundreds of XML files to be read and store in a database

XmlReader is faster

A service that received a very large XML file every month. And you need to retrieve the header information that contains the history date range and the customer information on which this file is based.

XmlReader is stream-based; you can open the file and read until you retrieve the information you need.