LINQ to XML 提供使用 .NET 语言集成查询 (LINQ) Framework 的内存中 XML 编程接口。 LINQ to XML 使用 .NET 功能，相当于更新的和重新设计的文档对象模型 (DOM) XML 编程接口。

在很多环境中，XML 已广泛采用为格式化数据的方式。 例如，在 Web 上，在配置文件、Microsoft Office Word 文件以及数据库中，都可以看到 XML。

LINQ to XML 经过了重新设计，是最新的 XML 编程方法。 提供文档对象模型 (DOM) 的内存中文档修改功能，并支持 LINQ 查询表达式。 尽管这些查询表达式在语法上与 XPath 不同，但它们提供类似的功能。

**LINQ to XML 开发人员**

LINQ to XML 面向各种开发人员。 对于只想完成某项工作的普通开发人员，LINQ to XML 通过提供与 SQL 相似的查询表达式，使 XML 变得更加简单。 只要稍加学习，程序员就能学会以自己选择的编程语言编写简洁、功能强大的查询。

专业开发人员可以使用 LINQ to XML 来大幅提高他们的工作效率。 通过使用 LINQ to XML，他们可以编写出更少的代码，表达能力更强、更为紧凑、功能更强。 他们可以同时对多个数据域使用查询表达式。

**LINQ to XML 是 XML 编程接口**

LINQ to XML 是一种启用了 LINQ 的内存中 XML 编程接口，支持在 .NET 编程语言中处理 XML。

LINQ to XML 将 XML 文档置于内存中，这一点很像文档对象模型 (DOM)。 您可以查询和修改 XML 文档，修改之后，可以将其另存为文件，也可以将其序列化然后通过 Internet 发送。 但是，LINQ to XML 与 DOM 不同：

* 它提供了一种新的目标模型，更轻、更易操作。
* 它利用 C# 和 Visual Basic 中的语言功能。

LINQ to XML 最重要的优势是它与语言集成查询 (LINQ) 的集成。 由于实现了这一集成，因此，可以对内存 XML 文档编写查询，以检索元素和属性的集合。 LINQ to XML 的查询功能在功能上（尽管不是在语法上）与 XPath 和 XQuery 具有可比性。 C# 和 Visual Basic 中的 LINQ 集成可提供更强的类型化功能、编译时检查和改进的调试器支持。

LINQ to XML 的另一个优势是将查询结果用作 [XElement](https://learn.microsoft.com/zh-cn/dotnet/api/system.xml.linq.xelement) 和 [XAttribute](https://learn.microsoft.com/zh-cn/dotnet/api/system.xml.linq.xattribute) 对象构造函数的参数，实现了一种功能强大的创建 XML 树的方法。 此方法称为*功能构造*，可使开发人员轻松地将 XML 树从一个形状转换成另一个形状。

例如，可能有[示例 XML 文件：典型采购订单](https://learn.microsoft.com/zh-cn/dotnet/standard/linq/sample-xml-file-typical-purchase-order)中所述的典型 XML 采购订单。 通过使用 LING to XML，可以运行以下查询，以获取采购单每个项元素的部件号属性值：

C#复制

// Load the XML file from our project directory containing the purchase orders

var filename = "PurchaseOrder.xml";

var currentDirectory = Directory.GetCurrentDirectory();

var purchaseOrderFilepath = Path.Combine(currentDirectory, filename);

XElement purchaseOrder = XElement.Load(purchaseOrderFilepath);

IEnumerable<string> partNos = from item in purchaseOrder.Descendants("Item")

select (string) item.Attribute("PartNumber");

在 C# 中，这也可采用方法语法形式重写：

C#复制

IEnumerable<string> partNos = purchaseOrder.Descendants("Item").Select(x => (string) x.Attribute("PartNumber"));

另一个示例，您可能需要一个列表，列出值大于 100 美元的项，并根据部件号排序。 若要获取此信息，可以运行下面的查询：

C#复制

// Load the XML file from our project directory containing the purchase orders

var filename = "PurchaseOrder.xml";

var currentDirectory = Directory.GetCurrentDirectory();

var purchaseOrderFilepath = Path.Combine(currentDirectory, filename);

XElement purchaseOrder = XElement.Load(purchaseOrderFilepath);

IEnumerable<XElement> pricesByPartNos = from item in purchaseOrder.Descendants("Item")

where (int) item.Element("Quantity") \* (decimal) item.Element("USPrice") > 100

orderby (string)item.Element("PartNumber")

select item;

同样，在 C# 中，这也可采用方法语法形式重写：

C#复制

IEnumerable<XElement> pricesByPartNos = purchaseOrder.Descendants("Item")

.Where(item => (int)item.Element("Quantity") \* (decimal)item.Element("USPrice") > 100)

.OrderBy(order => order.Element("PartNumber"));

除了这些 LINQ 功能以外，LINQ to XML 提供了改进的 XML 编程接口。 使用 LINQ to XML，可以：

* 从[文件](https://learn.microsoft.com/zh-cn/dotnet/standard/linq/load-xml-file)或[流](https://learn.microsoft.com/zh-cn/dotnet/standard/linq/stream-xml-fragments-xmlreader)加载 XML。
* 将 XML 序列化为文件或流。
* 使用函数构造从头开始创建 XML。
* 使用类似 XPath 的轴查询 XML。
* 使用 [Add](https://learn.microsoft.com/zh-cn/dotnet/api/system.xml.linq.xcontainer.add)、[Remove](https://learn.microsoft.com/zh-cn/dotnet/api/system.xml.linq.xnode.remove)、[ReplaceWith](https://learn.microsoft.com/zh-cn/dotnet/api/system.xml.linq.xnode.replacewith) 和 [SetValue](https://learn.microsoft.com/zh-cn/dotnet/api/system.xml.linq.xelement.setvalue) 等方法对内存 XML 树进行操作。
* 使用 XSD 验证 XML 树。
* 使用这些功能的组合，可将 XML 树从一种形状转换为另一种形状。

**创建 XML 树**

使用 LINQ to XML 编程的一个明显优势是容易创建 XML 树。 例如，若要创建一个小型 XML 树，可以编写以下代码：

C#复制

XElement contacts =

new XElement("Contacts",

new XElement("Contact",

new XElement("Name", "Patrick Hines"),

new XElement("Phone", "206-555-0144",

new XAttribute("Type", "Home")),

new XElement("phone", "425-555-0145",

new XAttribute("Type", "Work")),

new XElement("Address",

new XElement("Street1", "123 Main St"),

new XElement("City", "Mercer Island"),

new XElement("State", "WA"),

new XElement("Postal", "68042")

)

)

);

**备注**

示例 Visual Basic 使用 XML 文本。 还可以在 Visual Basic 中使用 [**XElement**](https://learn.microsoft.com/zh-cn/dotnet/api/system.xml.linq.xelement)，如 C# 版本一样。

有关详细信息，请参见 [XML 树](https://learn.microsoft.com/zh-cn/dotnet/standard/linq/functional-construction)。

**另请参阅**

* [引用](https://learn.microsoft.com/zh-cn/dotnet/standard/linq/reference)
* [LINQ to XML 与DOM](https://learn.microsoft.com/zh-cn/dotnet/standard/linq/linq-xml-vs-dom)
* [LINQ to XML 与其他 XML 技术](https://learn.microsoft.com/zh-cn/dotnet/standard/linq/linq-xml-vs-xml-technologies)
* [System.Xml.Linq](https://learn.microsoft.com/zh-cn/dotnet/api/system.xml.linq)