LINQ TO XML

Nội dung

- Tổng quan về LINQ
- Truy vấn XML bằng LINQ
- Tạo và Biến đổi một tài liệu XML

Giới thiệu về LINQ

- LINQ (Language Integrated Query) – Ngôn ngữ truy vấn tích hợp - Ngôn ngữ thống nhất cách thức truy xuất dữ liệu trong .NET.

- Được phát triển từ năm 2003, và được công bố tại Hội nghị Microsoft Professional Developers vào 2005 do Anders Hejlsberg và nhóm của ông đã trình bày.

- Được tích hợp sẵn trong Visual Studio 2008 (.NET 3.5).

Giới thiệu về LINQ

- **LINQ được phân thành 3 loại:**
  - LINQ to Objects
  - LINQ to XML
  - LINQ to SQL

- **Lợi ích sử dụng LINQ**

- Nó cung cấp một cách chung để truy xuất dữ liệu từ bất kỳ nguồn nào với cùng một cú pháp.
- Giúp thực hiện truy vấn một cách dễ dàng hơn.
- Được tích hợp như cú pháp của ngôn ngữ lập trình.

Nội dung

- Tổng quan về XML và LINQ
- **Truy vấn XML bằng LINQ**
- Tạo và Biến đổi một tài liệu XML
- Danh sách nhóm và tài liệu tham khảo

LINQ TO XML(1)

- LINQ to XML cung cấp một giao diện lập trình XML.

- LINQ to XML sử dụng những ngôn ngữ mới nhất của .NET Language Framework và được nâng cấp, thiết kế lại với giao diện lập trình XML Document object Model (DOM). Tuy nhiên nó cung cấp mô hình đối tượng mới đơn giản hơn và dễ thao tác hơn để làm việc.

- LINQ to XML có cấu trúc truy vấn tương tự SQL.

- Khả năng sử dụng kết quả truy vấn là tham số cho đối tượng XElement và XAttribute cho phép một phương pháp mạnh mẽ để tạo ra cây XML

LINQ TO XML(2)

- Sử dụng LINQ to XML bạn có thể:
  - Load XML từ nhiều file hoặc luồng.
  - Xuất XML ra file hoặc luồng.
  - Truy vấn cây XML bằng những truy vấn LINQ.
  - Thao tác cây XML trong bộ nhớ.
  - Biến đổi cây XML từ dạng này sang dạng khác.

Truy vấn XML bằng LINQ(1)

- IEnumerable<XElement> Descendants(string ElementName)
  - Trả ra một Collections XElement giúp bạn truy xuất đến Element có tên ElementName.
- IEnumerable<XElement> Ancestor();
  - Trả ra một Collections các XElement chứa nó (parent).
- IEnumerable<XElement> ElementBeforeSelf();
  - Trả ra một Collections các XElement đứng trước Element hiện tại.
- IEnumerable<XElement> ElementAfterSelf();
  - Trả ra một Collections các XElement đứng sau Element hiện tại.

Truy vấn XML bằng LINQ(2)

- XElement Parent();
  - Trả ra một XElement giúp bạn truy xuất đến Element cha(chứa nó).
- bool HasElement();
  - Kiểm tra xem Element hiện tại có Element con nào không.
- bool HasAttribute();
  - Kiểm tra xem Element hiện tại có Attribute nào không.
- void Save(string URL);
  - Lưu một tài liệu XML xuống file URL
- bool IsEmpty();
  - Kiểm tra xem Element hiện tại có rỗng không.

Truy vấn XML bằng LINQ(4)

❑**Ba phần cơ bản của một truy vấn XML**

1. **Có được các dữ liệu nguồn.**
2. **Tạo các truy vấn.**
3. **Thực hiện các truy vấn.**

❖Lấy nội dung tag Rows của Element có levelIndex == 2

```
using System.Xml.Linq;

…
XDocument XDoc = new
XDocument.Load(“GameData.xml”);

var query =  from q in
XDoc.Descendants(“Levels”).Elements(“Level”)
where  (int)q.XAttribute(“levelIndex”) == 2
select  q;

foreach( var p in query)
{
    System.Console.WriteLine(“Rows: {0}”,
                p.Element(“Rows”).Value);
}
```

GameData.xml

```
<? xml version="1.0"  encoding="utf-8"?>
<!--Thông tin Level -->
<Levels>
  <Level levelIndex="1">
    <ImagePath>Girl.jpg</ImagePath>
    <Rows>3</Rows>
    <Cols>3</Cols>
  </Level>
<Level levelIndex=“2">
    <ImagePath>Boy.jpg</ImagePath>
    <Rows>4</Rows>
    <Cols>4</Cols>
  </Level>
</Levels>
```

Nội dung

- Tổng quan về XML và LINQ
- Truy vấn XML bằng LINQ
- Tạo và Biến đổi một tài liệu XML

Tạo và Biến đổi một tài liệu XML

Tạo một tài liệu XML:

- **Ta sử dụng hàm khởi tạo XDocument();**

  XDocument(XDeclaration xd, param object[] content**);**

  Trong đó:

  - XDeclaration(string version, string encoding, string standalone)
    - version là phiên bản của tài liệu "1.0".
    - encoding là bảng mã lữu trữ "UTF-8".
    - standalone tồn tại một mình hay không "Yes" hoặc "No".

    Thường thì không có thuộc tính này cũng không sao cả.
  - object[] là danh sách các đối tượng cần đưa vào tài liệu:
    - XComment(string comment)

      Chứa chú thích cho tài liệu comment = "Chú thích đây" sẽ cho kết quả <!-- Chú thích đây --!>
    - XElement(XName name, param object[] content)
      - XName là tên của Element <name> </name>
      - object[] tương tự object[] nói ở trên.

- **Dùng hàm Save(string URL) để lưu xuống file URL**

Tạo và Biến đổi một tài liệu XML

Tạo một tài liệu XML:

Code:

```
using System.Xml.Linq;

…
XDocument XDoc = new XDocument(

     new XDeclaration("1.0", "utf-8”),
     new XComment("Thông tin  Level "),
      new XElement(“Levels",
          new XElement(“Level",
             new XAttribute(“levelIndex“,1),
             new XElement("ImagePath", “Girl.jpg”),
             new XElement("Rows", 3),
             new XElement(“Cols", 3)))
);

XDoc. Save(“GameData.xml”);
```

GameData.xml

```
<? xml version="1.0"  encoding="utf-8"?>
<!--Thông tin Level -->
<Levels>
  <Level levelIndex="1">
    <ImagePath>Girl.jpg</ImagePath>
    <Rows>3</Rows>
    <Cols>3</Cols>
  </Level>
</Levels>
```

Tạo và Biến đổi một tài liệu XML

Hoặc táo bạo hơn một chút với phương thức Parse()

Code:

```
using System.Xml.Linq;

…
XDocument XDoc = new XDocument(
   new XDeclaration("1.0", "utf-8", "yes"),
   new XComment("Thông tin  Level "),
   new XElement.Parse(
      “<Levels>
         <Level levelIndex="1">
           <ImagePath>Girl.jpg</ImagePath>
           <Rows>3</Rows>
           <Cols>3</Cols>
           </Level>
       </Levels>”
   )
);
XDoc. Save(“GameData.xml”);
```

GameData.xml

```
<? xml version="1.0"  encoding="utf-8"?>
<!--Thông tin Level -->
<Levels>
  <Level levelIndex="1">
    <ImagePath>Girl.jpg</ImagePath>
    <Rows>3</Rows>
    <Cols>3</Cols>
  </Level>
</Levels>
```

Thêm một Element(1)

Add

- void Add(XElement content);
- Thêm content như là một Node con vào cuối Element hiện tại.

AddFist

- void AddFist(XElement content);
- Thêm content như là một Node con vào đầu Element hiện tại.

AddAfterSelf

- void AddAfterSelf(XElement content);
- Chèn một element vào sau element hiện tại

AddBeforeSelf

- void AddBeforeSelf(XElement content);
- Chèn một element vào trước element hiện tại

Thêm một Element(2)

Add

- void Add(XElement content);
- Thêm content như là một Node con vào cuối Element hiện tại.

Code: **Add một XElement vào trong <Levels>_</Levels>**

```
using System.Xml.Linq;

…
XDocument  XDoc =
     new XDocument.Load(“GameData.xml”);

XDoc.Element ("Levels“).Add(
    new XElement(
       "Level“,
       new XAttribute(“levelIndex", 2),
       new XElement("ImagePath", “Laydy.jpg”),
       new XElement("Rows“, 3),
       new XElement("Cols", 4)
     )
);
XDoc. Save(“GameData.xml”);
```

GameData.xml

```
<? xml version="1.0"  encoding="utf-8"?>
<!--Thông tin mỗi Level trong game-->
<Levels>
  <Level levelIndex="1">
    <ImagePath>Girl.jpg</ImagePath>
    <Rows>3</Rows>
    <Cols>3</Cols>
  </Level>
</Levels>
```

```
   <Level levelIndex=“2">
     <ImagePath>Lady.jpg</ImagePath>
     <Rows>3</Rows>
     <Cols>4</Cols>
   </Level>
</Levels>
```

XDoc.Add(.....); Error ! → Vị trí cần chèn vào không phải là một XElement

Thêm một Element(3)

AddFist

- void AddFist(XElement content);
- Thêm content như là một Node con vào đầu Element hiện tại.

Code: Add một XElement vào trong <Levels>_</Levels>

```
using System.Xml.Linq;

…
XDocument  XDoc =
     new XDocument.Load(“GameData.xml”);

XDoc.Element ("Levels“).AddFirst(
    new XElement(
        "Level“,
        new XAttribute(“levelIndex", 2),
        new XElement("ImagePath", “Laydy.jpg”),
        new XElement("Rows“, 3),
        new XElement("Cols", 4)
     )
);
XDoc. Save(“GameData.xml”);
```

GameData.xml

```
<? xml version="1.0"  encoding="utf-8"?>
<!--Thông tin mỗi Level trong game-->
<Levels>
  <Level levelIndex="1">
    <ImagePath>Girl.jpg</ImagePath>
    <Rows>3</Rows>
    <Cols>3</Cols>
  </Level>
</Levels>
```

```
      <ImagePath>Girl.jpg</ImagePath>
      <Rows>3</Rows>
      <Cols>3</Cols>
   </Level>
</Levels>
```

❑ XDoc.Add(.....); Error ! → Vị trí cần chèn vào không phải là một XElement.

Thêm một Element(4)

AddAfterSelf

- void AddAfterSelf(XElement content);
- Chèn một element vào sau element hiện tại

Code: Add một XElement sau <Level levelIndex="1“>

```
using System.Xml.Linq;

…
XDocument  XDoc =
      new XDocument.Load(“GameData.xml”);

XDoc.Descendants(“Levels”).Elements
("Level“).Last().AddAfterSelf(
     new XElement(
          "Level“,
          new XAttribute(“levelIndex", 2),
          new XElement("ImagePath", “Laydy.jpg”),
          new XElement("Rows“, 3),
          new XElement("Cols", 4)
      )
);
XDoc. Save(“GameData.xml”);
```

GameData.xml

```
<? xml version="1.0"  encoding="utf-8"?>
<!--Thông tin mỗi Level trong game-->
<Levels>
  <Level levelIndex="1">
    <ImagePath>Girl.jpg</ImagePath>
    <Rows>3</Rows>
    <Cols>3</Cols>
  </Level>
</Levels>
```

```
    <ImagePath>Lady.jpg</ImagePath>
    <Rows>3</Rows>
    <Cols>4</Cols>
  </Level>
</Levels>
```

Thêm một Element(5)

AddBeforeSelf

- void AddBeforeSelf(XElement content);
- Chèn một element vào trước element hiện tại

Code: Add một XElement trước <Level levelIndex="1“>

```
using System.Xml.Linq;

…
XDocument  XDoc =
      new XDocument.Load(“GameData.xml”);

XDoc.Descendants(“Levels”).Elements
("Level“).Last().AddBeforeSelf(
     new XElement(
          "Level“,
          new XAttribute(“levelIndex", 2),
          new XElement("ImagePath", “Laydy.jpg”),
          new XElement("Rows“, 3),
          new XElement("Cols", 4)
      )
);
XDoc. Save(“GameData.xml”);
```

GameData.xml

```
<? xml version="1.0"  encoding="utf-8"?>
<!--Thông tin mỗi Level trong game-->
<Levels>
  <Level levelIndex="1">
    <ImagePath>Girl.jpg</ImagePath>
    <Rows>3</Rows>
    <Cols>3</Cols>
  </Level>
</Levels>
```

```
      <ImagePath>Girl.jpg</ImagePath>
      <Rows>3</Rows>
      <Cols>3</Cols>
   </Level>
</Levels>
```

Thêm một Element(6)

Kết luận:

- Trong hai 4 ví dụ trên thì hai hàm Add() và AddAfterSelf() cho cùng một kết quả chỉ khác nhau ở vị trí Add vào:

  - **XDoc.Element ("Levels").**Add()

  - **XDoc.Descendants("Levels").Elements ("Level").Last().**AddAfterSelf()

- Tương tự với hai hàm AddFirst() và AddBeforeSelf().

Cập nhật một Element(1)

Value

- VD: XElement XE; XE.Value = “123”;
- Một dạng Properties giúp đặt giá trị cho một element một cách trực tiếp.

SetValue

- SetValue(object value);
- Đặt giá trị cho element hiện tại giá trị là value vì là kiểu object nên nó cũng có thể là cả một XElement.

SetAttributeValue

- SetAttributeValue(XName name, object value);
- Đặt giá trị value cho “attribute” của element tên name.

SetElementValue

- SetElementValue(XName name, object value);
- Đặt giá trị cho “element” tên name giá trị là value vì là kiểu object nên nó cũng có thể là một XElement.

Cập nhật một Element(2)

Value

- VD: XElement XE; XE.Value = "123";
- Một dạng Properties giúp đặt giá trị cho một element một cách trực tiếp.

❑**Muốn cập nhật phải tìm được XElement trong tài liệu trước.**

Code: **Thay đổi ImagePath của <Level levelIndex="1">**

```
using System.Xml.Linq;
XDocument XDoc =
            new XDocument.Load("GameData.xml");

foreach (XElement x in XDoc.Descendants("Level"))
        {
           if
(x.Attribute("levelIndex").Value.ToString().Equals("1"))
              x.Element("ImagePath").Value =
"BoyNew.jpg";
        }
XDoc. Save("GameData.xml");
```

GameData.xml

```
<? xml version="1.0"  encoding="utf-8"?>
<!--Thông tin mỗi Level trong game-->
<Levels>
  <Level levelIndex="1">
    <ImagePath>Boy.jpg</ImagePath>
    <Rows>3</Rows>
    <Cols>3</Cols>
  </Level>
</Levels>
```

❑**Tương tự với các thành phần khác của <Level levelIndex="1"> _</Levels>**
Như: <Rows> hay <Cols>...

Cập nhật một Element(3)

SetValue

- SetValue(object value);
- Đặt giá trị cho element hiện tại giá trị là value vì là kiểu object nên nó cũng có thể là cả một XElement.

Tương tự như Properties Value:

- Chỉ cần thay x.Element("ImagePath") = "Boy.jpg";

Bằng x.SetValue("Boy.jpg"); sẽ cho kết quả ý hệt.

Cập nhật một Element(3)

SetAttributeValue

- SetAttributeValue(XName name, object value);
- Đặt giá trị value cho "attribute" của element tên name.

❑**Muốn cập nhật phải tìm được XElement trong tài liệu trước.**

Code: **Thay đổi Attribute của <Level levelIndex="1">**

```
using System.Xml.Linq;
…
XDocument  XDoc =
            new XDocument.Load("GameData.xml");

foreach (XElement x in XDoc.Descendants("Level"))
        {
    if (x.Attribute("levelIndex").Value.ToString().Equals("1"))
            x.SetAttributeValue( "Index", 3);
}
XDoc. Save("GameData.xml");
```

GameData.xml

```
<? xml version="1.0"  encoding="utf-8"?>
<!--Thông tin mỗi Level trong game-->
<Levels>
 <Level Index="3">
  <ImagePath>Girl.jpg</ImagePath>
  <Rows>3</Rows>
  <Cols>3</Cols>
 </Level>
</Levels>
```

Cập nhật một Element(4)

SetElementValue

- SetElementValue(XName name, object value);
- Đặt giá trị cho "element" tên name giá trị là value vì là kiểu object nên nó cũng có thể là một XElement.

Code: **Thay đổi Element <Level levelIndex="1">**

```
using System.Xml.Linq;
...
XDocument  XDoc =
                 new Document.Load("GameData.xml");

foreach (XElement x in XDoc.Descendants("Level"))
        {
if (x.Attribute("levelIndex").Value.ToString().Equals("1"))
      x.SetElementValue( "Level",
          new XElement("Level",
                new XAttribute("levelIndex", 3),
                new XElement("ImagePath", "Laydy.jpg"),
                new XElement("Rows", 5),
                new XElement("Cols", 5)
          )
      );
}
XDoc. Save("GameData.xml");
```

GameData.xml

```
<? xml version="1.0"  encoding="utf-8"?>
<!--Thông tin mỗi Level trong game-->
<Levels>
  <Level Index="3">
    <ImagePath>Laydy.jpg</ImagePath>
    <Rows>5</Rows>
    <Cols>5</Cols>
  </Level>
</Levels>
```

❑ **Bạn cũng có thể thay đổi tất cả các thành phần của Element hiện tại bằng cách tạo một value là một Element mới !**

Xóa Element(1)

Remove	• Remove(); • Xóa node hiện tại từ element parent của nó.
RemoveAll	• RemoveAll(); • Xóa tất cả các Attribute và Node con của tài liệu XML hoặc element hiện tại.
RemoveAttributes	• RemoveAttributes(); • Xóa “attribute” của element hiện tại .
RemoveNodes	• RmoveNode(); • Xóa node con của element hoặc của cả tài liệu XML.

Xóa Element(2)

Remove

- Remove();
- Xóa node hiện tại từ element parent của nó.

Code: Xóa Tag của Level có Attribute == "1“

```
using System.Xml.Linq;
…
XDocument  XDoc =
            new XDocument.Load(“GameData.xml”);

foreach (XElement x in XDoc.Descendants("Level"))
        {
if (x.Attribute("levelIndex").Value.ToString().Equals("1"))
            x.Element(“Cols”).Remove();
}
XDoc. Save(“GameData.xml”);
```

GameData.xml

```
<? xml version="1.0"  encoding="utf-8"?>
<!--Thông tin mỗi Level trong game-->
<Levels>
 <Level levelIndex="1">
   <ImagePath>Girl.jpg</ImagePath>
   <Rows>3</Rows>
</Levels>
```

❑ **Sau khi xác định được Level có Attirbute levelIndex == “1” rồi tìm đến Tag <Cols> ta xóa nó.**

Xóa Element(3)

RemoveAll

- RemoveAll();
- Xóa tất cả các Attribute và Node con của tài liệu XML hoặc element hiện tại.

Code: **Xóa Element Level có Attribute == "1"**

```
using System.Xml.Linq;
...
XDocument  XDoc =
            new XDocument.Load("GameData.xml");

foreach (XElement x in XDoc.Descendants("Level"))
        {
if (x.Attribute("levelIndex").Value.ToString().Equals("1"))
            x.RemoveAll();
}
XDoc. Save("GameData.xml");
```

❑ **Sau khi xác định được Level có Attirbute levelIndex == "1" thì xóa nó.**

Sau khi xóa, file GameData.xml

GameData.xml

```
<? xml version="1.0"  encoding="utf-8"?>
<!--Thông tin mỗi Level trong game-->
<Levels>
  <Level levelIndex="1">
    <ImagePath>Girl.jpg</ImagePath>
    <Rows>3</Rows>
    <Cols>3</Cols>
  </Level>
</Levels>
```

```
<? xml version="1.0"  encoding="utf-8"?>
<!--Thông tin mỗi Level trong game-->
<Levels>

</Levels>
```

Xóa Element(4)

RemoveAttributes

- RemoveAttributes();
- Xóa Attribute của element hiện tại .

Code: **Xóa Attribute của Level có Attribute == "1“**

```
using System.Xml.Linq;
...
XDocument  XDoc =
            new XDocument.Load(“GameData.xml”);

foreach (XElement x in XDoc.Descendants("Level"))
          { if
(x.Attribute("levelIndex").Value.ToString().Equals("1"))
            x.RemoveAttribute();
}
XDoc. Save(“GameData.xml”);
```

GameData.xml

```
<? xml version="1.0"  encoding="utf-8"?>
<!--Thông tin mỗi Level trong game-->
<Levels>
  <Level >
    <ImagePath>Girl.jpg</ImagePath>
    <Rows>3</Rows>
    <Cols>3</Cols>
  </Level>
</Levels>
```

❑ **Sau khi xác định được Level có Attirbute levelIndex == “1” thì xóa Attribute nó.**

Xóa Element(5)

RemoveNodes

- RmoveNode();
- Xóa node con của một XElement hoặc của tài liệu XML(XDocument).

Code: **Xóa Tag <Size> của Level có Attribute == "1“**

```
using System.Xml.Linq;
...
XDocument  XDoc =
                new XDocument.Load(“GameData.xml”);

foreach (XElement x in XDoc.Descendants("Level"))
          {
if (x.Attribute("levelIndex").Value.ToString().Equals("1"))
                x.Element(“Size”).RemoveNodes();
}
XDoc. Save(“GameData.xml”);
```

GameData.xml

```
<? xml version="1.0"  encoding="utf-8"?>
<!--Thông tin mỗi Level trong game-->
<Levels>
  <Level >
    <ImagePath>Girl.jpg</ImagePath>
</Level>
</Levels>
```

❑ **Sau khi xác định được Level có Attirbute levelIndex == “1” thì xóa Node “Size” nó.**

Nội dung

- Tổng quan về XML và LINQ
- Truy vấn XML bằng LINQ
- Tạo và Biến đổi một tài liệu XML
- **Truy vấn cơ bản trong XML**

Truy vấn cơ bản trong XML

- Tìm một phần tử trong cây XML

Tìm phần tử Address có thuộc tính Type có giá trị là "Billing".

Code:

```
XElement root =
XElement.Load("PurchaseOrder.xml");
IEnumerable<XElement> address =
from el in root.Elements("Address")
where (string)el.Attribute("Type") == "Billing"
select el;
foreach (XElement el in address)
Console.WriteLine(el);
```

File .xml

```
<PurchaseOrder PurchaseOrderNumber="99503"
OrderDate="1999-10-20">
  <Address Type="Shipping">
    <Name>Ellen Adams</Name>
    <Street>123 Maple Street</Street>
    <City>Mill Valley</City>
    <State>CA</State>
    <Zip>10999</Zip>
    <Country>USA</Country>
  </Address>
  <Address Type="Billing">
    <Name>Tai Yee</Name>
    <Street>8 Oak Avenue</Street>
    <City>Old Town</City>
    <State>PA</State>
    <Zip>95819</Zip>
    <Country>USA</Country>
  </Address>
```

Truy vấn cơ bản trong XML

- Lọc phần tử trong cây XML

Ví dụ sau lọc những phần tử có phần tử con <*Type* > có *Value="Yes"*.

Code:

```
XElement root = XElement.Parse(@"<Root>
<Child1>
<Text>Child One Text</Text>
<Type Value=""Yes""/>
</Child1>
<Child2>
<Text>Child Two Text</Text>
<Type Value=""Yes""/>
</Child2>
<Child3>
<Text>Child Three Text</Text>
<Type Value=""No""/>
</Child3>
<Child4>
<Text>Child Four Text</Text>
<Type Value=""Yes""/>
</Child4>
<Child5>
<Text>Child Five Text</Text>
</Child5>
</Root>");
var cList =
from typeElement in root.Elements().Elements("Type")
where (string)typeElement.Attribute("Value") == "Yes"
select (string)typeElement.Parent.Element("Text");
foreach(string str in cList)

Console.WriteLine(str);
```

results

```
Child One Text
Child Two Text
Child Four Text
```

Truy vấn cơ bản trong XML

- **Sắp xếp các phần tử** trong cây XML

Sắp theo giá

Code:

```
XElement root = XElement.Load("Data.xml");
IEnumerable<decimal> prices =
from el in root.Elements("Data")
let price = (decimal)el.Element("Price")
orderby price
select price;
foreach (decimal el in prices)
Console.WriteLine(el);
```

results

```
0.99
4.95
6.99
24.50
29.00
66.00
89.99
```

Thank You !

WINP1.C24