we will discuss **creating an XML document using LINQ to XML.**  
  
  
  
**What is Functional Construction?**  
As far as LINQ to XML is concerned there is a technical term called Functional Construction. First let us understand what this term means with an example. Functional construction is the ability to create an XML tree in a single statement.  
  
  
  
Let us now discuss, **creating an XML tree in a single statement**. We want to create an XML tree that looks as shown below.

<?xml version="1.0" encoding="utf-8" standalone="yes"?>

<!--Creating an XML Tree using LINQ to XML-->

<Students>

  <Student Id="101">

    <Name>Mark</Name>

    <Gender>Male</Gender>

    <TotalMarks>800</TotalMarks>

  </Student>

  <Student Id="102">

    <Name>Rosy</Name>

    <Gender>Female</Gender>

    <TotalMarks>900</TotalMarks>

  </Student>

  <Student Id="103">

    <Name>Pam</Name>

    <Gender>Female</Gender>

    <TotalMarks>850</TotalMarks>

  </Student>

  <Student Id="103">

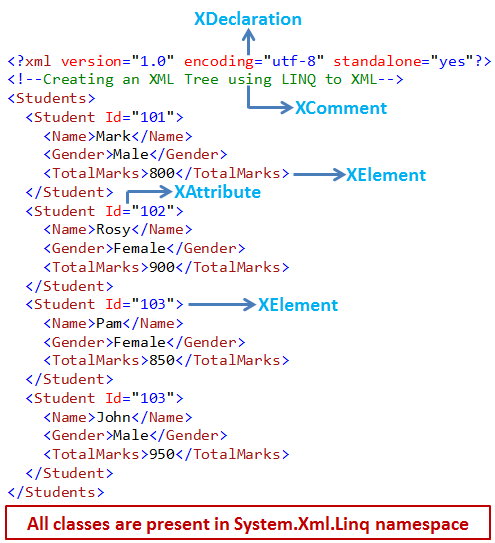
    <Name>John</Name>

    <Gender>Male</Gender>

    <TotalMarks>950</TotalMarks>

  </Student>

</Students>

All the classes to create an **XML document are present in System.Xml.Linq** namespace. To create  
XML Document use XDocument class  
XML Declaration use XDeclaration class  
XML Comment use XComment class  
XML Element use XElement class  
XML Attribute use XAttribute class  
  
  
**Code to create the XML Document**

using System.Xml.Linq;

namespace Demo

{

    class Program

    {

        public static void Main()

        {

            XDocument xmlDocument = new XDocument(

                new XDeclaration("1.0", "utf-8", "yes"),

                new XComment("Creating an XML Tree using LINQ to XML"),

                new XElement("Students",

                    new XElement("Student", new XAttribute("Id", 101),

                        new XElement("Name", "Mark"),

                        new XElement("Gender", "Male"),

                        new XElement("TotalMarks", 800)),

                    new XElement("Student", new XAttribute("Id", 102),

                        new XElement("Name", "Rosy"),

                        new XElement("Gender", "Female"),

                        new XElement("TotalMarks", 900)),

                    new XElement("Student", new XAttribute("Id", 103),

                        new XElement("Name", "Pam"),

                        new XElement("Gender", "Female"),

                        new XElement("TotalMarks", 850)),

                    new XElement("Student", new XAttribute("Id", 104),

                        new XElement("Name", "John"),

                        new XElement("Gender", "Male"),

                        new XElement("TotalMarks", 950))));

            xmlDocument.Save(@"C:\Demo\Demo\Data.xml");

        }

    }

}

Upon running the console application, an XML file with name **Data.xml**should be created in the respective project folder. To see the xml file, click on **"Show All Files"** icon in the solution explorer. Double click on the xml file to open it in visual studio.