# Statement Purpose:

Purpose of this lab is to learn Extensible Markup Language (XML) file structure, reading and writing xml files using C#. XML is the universal format for data on the Web. XML allows developers to easily describe and deliver rich, structured data from any application in a standard, consistent way. XML does not replace HTML; rather, it is a complementary format.

# Activity Outcomes:

This lab teaches you the following topics:

* Writing XML files
* Reading XML files

# Stage J (Journey) Introduction

XmlWriter class represents a writer that provides a fast, non-cached, forward-only way to generate streams or files that contain XML data.

The members of the XmlWriter class enable you to:

* + Verify that the characters are legal XML characters and that element and attribute names are valid XML names.
  + Verify that the XML document is well-formed.
  + Encode binary bytes as Base64 or BinHex, and write out the resulting text.
  + Pass values by using common language runtime types instead of strings, to avoid having to manually perform value conversions.
  + Write multiple documents to one output stream.
  + Write valid names, qualified names, and name tokens.

XmlReader class Represents a reader that provides fast, noncached, forward-only access to XML data.

XmlReader uses a pull model to retrieve data. This model:

* + Simplifies state management by a natural, top-down procedural refinement.
  + Supports multiple input streams and layering.
  + Enables the client to give the parser a buffer into which the string is directly written, and thus avoids the necessity of an extra string copy.
  + Supports selective processing. The client can skip items and process those that are of interest to the application. You can also set properties in advance to manage how the XML stream is processed (for example, normalization).

# Stage a1 (apply) Lab Activities:

### Activity 1:

Create following XML file using XmlWriter class:

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

<catalog>

<book id="bk101">

<author>Gambardella, Matthew</author>

<title>XML Developer's Guide</title>

<genre>Computer</genre>

<price>44.95</price>

<publish\_date>2000-10-01</publish\_date>

</book>

</catalog>

Use XmlWriterSettings class to indent the xml.

### Solution:

XmlWriterSettings settings = new XmlWriterSettings(); settings.Indent = true;

settings.IndentChars = "\t";

XmlWriter w = XmlWriter.Create("catalog.xml", settings); w.WriteStartDocument();

w.WriteStartElement("catalog");

w.WriteStartElement("book"); w.WriteAttributeString("id", "bk101"); w.WriteElementString("author", "Gambardella, Matthew"); w.WriteElementString("title", "XML Developer's Guide"); w.WriteElementString("genre", "Computer"); w.WriteElementString("price", "44.95");

w.WriteElementString("publish\_date", "2000-10-01"); w.WriteEndElement();

w.WriteEndDocument(); w.Close();

### Activity 2:

Create following XML file using XmlWriter class:

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

<GPS\_Log>

<Position DateTime="1/26/2017 5:08:59 PM">

<x>65.8973342</x>

<y>72.3452346</y>

<SatteliteInfo>

<Speed>40</Speed>

<NoSatt>7</NoSatt>

</SatteliteInfo>

</Position>

<Image Resolution="1024x800">

<Path>\images\1.jpg</Path>

</Image>

</GPS\_Log>

### Solution:

XmlWriterSettings settings = new XmlWriterSettings(); settings.Indent = true;

settings.IndentChars = "\t";

XmlWriter w = XmlWriter.Create("GPS.xml", settings); w.WriteStartDocument(); w.WriteStartElement("GPS\_Log");

w.WriteStartElement("Position"); w.WriteAttributeString("DateTime", DateTime.Now.ToString()); w.WriteElementString("x", "65.8973342");

w.WriteElementString("y", "72.3452346"); w.WriteStartElement("SatteliteInfo"); w.WriteElementString("Speed", "40");

w.WriteElementString("NoSatt", "7"); w.WriteEndElement(); w.WriteEndElement();

w.WriteStartElement("Image"); w.WriteAttributeString("Resolution", "1024x800"); w.WriteElementString("Path", @"\images\1.jpg");

w.WriteEndDocument(); w.Close();

### Activity 3:

Read xml file in following format using XmlTextReader class. Also, print the xml on console.

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

<bookstore>

<book>

<title>

The Autobiography of Benjamin Franklin

</title>

<author>

<first-name> Benjamin

</first-name>

<last-name> Franklin

</last-name>

</author>

<price> 8.99

</price>

</book>

<book>

<title>

The Confidence Man

</title>

<author>

<first-name> Herman

</first-name>

<last-name> Melville

</last-name>

</author>

<price> 11.99

</price>

</book>

</bookstore>

### Solution:

XmlTextReader reader = new XmlTextReader("books.xml"); while (reader.Read())

{

switch (reader.NodeType)

{

case XmlNodeType.Element: // The node is an element.

Console.Write("<" + reader.Name); Console.WriteLine(">");

break;

case XmlNodeType.Text: //Display the text in each element.

Console.WriteLine(reader.Value); break;

case XmlNodeType.EndElement: //Display the end of the element.

Console.Write("</" + reader.Name); Console.WriteLine(">");

break;

}

}

# Stage v (verify) Home Activities:

### Activity 1:

Generate following xml file using XmlWriter. Format the xml using XmlSettings class.

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

<breakfast\_menu>

<food>

<name>Belgian Waffles</name>

<price>$5.95</price>

<description>

Two of our famous Belgian Waffles with plenty of real maple syrup

</description>

<calories>650</calories>

</food>

<food>

<name>Strawberry Belgian Waffles</name>

<price>$7.95</price>

<description>

Light Belgian waffles covered with strawberries and whipped cream

</description>

<calories>900</calories>

</food>

</breakfast\_menu>

### Activity 2:

Read XML file generated in above activity using XmlReader and display the output xml on console.

### Activity 3:

* Create a class of Food having data members name,price, description, calories and

a parametrized constructor to initialize the values of food objects.

* Now In main generate xml file as shown in above example by writing data of five objects

of food using XMLWriter.

* Read above Generated Xml file by using XMLTextReader class and place values in

objects of food and show these objects values on output screen.