using Serialization;

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Runtime.Serialization.Formatters.Binary;

using System.Text;

using System.Threading.Tasks;

using System.Xml;

using System.Xml.Serialization;

namespace Cs\_Lesson16

{

//public class Car

//{

// public string Model { get; set; }

// public string Vendor { get; set; }

// public int Year { get; set; }

// public override string ToString()

// {

// return $"{Model} {Vendor} {Year}";

// }

//}

public class Program

{

static void Main(string[] args)

{

//var cars = new List<Car>

//{

// new Car{Model = "Mustang", Vendor = "Ford", Year = 1964 },

// new Car{Model = "Charger", Vendor = "Dodge", Year = 2000 },

// new Car{Model = "Veyron", Vendor = "Bugatti", Year = 2010 },

// new Car{Model = "M5", Vendor = "BMW", Year = 2020 }

//};

#region XML Writer

//using (var writer = new XmlTextWriter("cars.xml", Encoding.UTF8))

//{

// writer.Formatting = Formatting.Indented;

// writer.WriteStartDocument();

// writer.WriteStartElement("Cars");

// foreach (var car in cars)

// {

// writer.WriteStartElement("Car");

// //// Write tag (Element)

// //writer.WriteElementString(nameof(car.Model), car.Model);

// //writer.WriteElementString(nameof(car.Vendor), car.Vendor);

// //writer.WriteElementString(nameof(car.Year), car.Year.ToString());

// // Write Attribute

// writer.WriteAttributeString(nameof(car.Model), car.Model);

// writer.WriteAttributeString(nameof(car.Vendor), car.Vendor);

// writer.WriteAttributeString(nameof(car.Year), car.Year.ToString());

// writer.WriteEndElement();

// }

// writer.WriteEndElement();

// writer.WriteEndDocument();

//}

#endregion

#region XML Reading

//XmlDocument xml = new XmlDocument();

//xml.Load("cars.xml");

//var root = xml.DocumentElement; // root

//if (root.HasChildNodes) // tag, element, node - same things

//{

// foreach (XmlNode car\_node in root.ChildNodes)

// {

// var car = new Car()

// {

// Model = car\_node.Attributes[0].Value,

// Vendor = car\_node.Attributes[1].Value,

// Year = int.Parse(car\_node.Attributes[2].Value)

// };

// Console.WriteLine(car);

// }

//}

#endregion

#region XML Serialize

var army = new TranslatorArmy

{

Name = "Step IT Academy",

Location = "Koroglu Rehimov 71",

Translators = new List<Translator>

{

new Translator

{

Name="Tural",

Surname="Eliyev",

Id=1,

Subjects=new List<Subject>

{

new Subject

{

Name="C++",

Degree=42,

Lessons=30

},

new Subject

{

Name="C#",

Degree=42,

Lessons=35

},

}

},

new Translator

{

Name="Eli",

Surname="Memmedov",

Id=2,

Subjects=new List<Subject>

{

new Subject

{

Name="JS",

Degree=42,

Lessons=30

},

new Subject

{

Name="PHP",

Degree=42,

Lessons=35

},

}

},

new Translator

{

Name="Leyla",

Surname="Mammadli",

Id=3,

Subjects=new List<Subject>

{

new Subject

{

Name="Angular",

Degree=42,

Lessons=30

},

new Subject

{

Name="React",

Degree=42,

Lessons=35

},

}

}

}

};

//var xml = new XmlSerializer(typeof(TranslatorArmy));

//using (var fs = new FileStream("TranslatorArmy.xml", FileMode.OpenOrCreate))

//{

// xml.Serialize(fs, army);

//}

#endregion

#region XML Deserialize

//TranslatorArmy army = null;

//var xml = new XmlSerializer(typeof(TranslatorArmy));

//using (var fs = new FileStream("TranslatorArmy.xml", FileMode.OpenOrCreate))

//{

// army = xml.Deserialize(fs) as TranslatorArmy;

//}

//Console.WriteLine(army);

#endregion

#region Binary Serialization

//var dataBase = new Translator[]

//{

// new Translator()

// {

// Id = 1,

// Name = "Rafiq",

// Surname = "Rafiqli"

// },

// new Translator()

// {

// Id = 2,

// Name = "Cosqun",

// Surname = "Memmedli"

// },

// new Translator()

// {

// Id = 3,

// Name = "John",

// Surname = "Johnlu"

// }

//};

////var bf = new BinaryFormatter();

////using (var fs = new FileStream("file.bin", FileMode.OpenOrCreate))

////{

//// bf.Serialize(fs, dataBase);

////}

//var bf = new BinaryFormatter();

//Translator[] translators = null;

//using (var fs = new FileStream("file.bin", FileMode.OpenOrCreate))

//{

// translators = bf.Deserialize(fs) as Translator[];

//}

//if (translators != null)

//{

// foreach (var item in translators)

// {

// Console.WriteLine(item);

// }

//}

#endregion

#region Json Serialization

#endregion

}

}

}