МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ

Національний технічний університет України

"Київський політехнічний інститут імені Ігоря Сікорського"

Факультет інформатики та обчислювальної техніки

Лабораторна робота №5

з дисципліни «Об'єктно оріентовне програмування»

Тема: « **LINQ to XML**»

Виконав:

студенти групи ІП-81

Балачін Петро

Київ-2020

**Мета роботи**: ознайомитися з обробкою XML документів з використанням технології LINQ to XML

**Хід роботи:**

**Варіант 3**

class Program

{

static void Main(string[] args)

{

XmlWriterSettings settings = new XmlWriterSettings();

settings.Indent = true;

using (XmlWriter writer = XmlWriter.Create("cars.xml", settings))

{

writer.WriteStartElement("cars");

foreach (Car car in Cars)

{

writer.WriteStartElement("car");

writer.WriteElementString("idCar", car.id.ToString());

writer.WriteElementString("mark", car.mark);

writer.WriteElementString("type", car.type);

writer.WriteElementString("carPrice", car.carPrice.ToString());

writer.WriteElementString("yearOfProduction", car.yearOfProdaction.ToString());

writer.WriteElementString("priceForDay", car.priceForDay.ToString());

writer.WriteEndElement();

}

writer.WriteEndElement();

}

using (XmlWriter writer = XmlWriter.Create("clients.xml", settings))

{

writer.WriteStartElement("clients");

foreach (Client client in Clients)

{

writer.WriteStartElement("client");

writer.WriteElementString("idClient", client.id.ToString());

writer.WriteElementString("fullName", client.fullName);

writer.WriteElementString("adress", client.adress);

writer.WriteElementString("phone", client.phone);

writer.WriteEndElement();

}

writer.WriteEndElement();

}

using (XmlWriter writer = XmlWriter.Create("transactions.xml", settings))

{

writer.WriteStartElement("transactions");

foreach (Transaction tran in Transactions)

{

writer.WriteStartElement("transaction");

writer.WriteElementString("clientId", tran.clientId.ToString());

writer.WriteElementString("carId", tran.carId.ToString());

writer.WriteElementString("startDay", tran.startDay);

writer.WriteElementString("finishDay", tran.finishDay);

writer.WriteElementString("deposit", tran.deposit.ToString());

writer.WriteEndElement();

}

writer.WriteEndElement();

}

XmlDocument carsDoc = new XmlDocument();

carsDoc.Load("cars.xml");

Console.WriteLine("All cars: ");

foreach (XmlNode node in carsDoc.DocumentElement)

{

Car car = new Car(Int32.Parse(node["idCar"].InnerText), node["mark"].InnerText, node["type"].InnerText,

Int32.Parse(node["carPrice"].InnerText), Int32.Parse(node["yearOfProduction"].InnerText));

Console.WriteLine(string.Format("Id: {0}; Mark: {1}; Type: {2}; Price: {3}; Year of Production: {4}; Daily price {5}", car.id, car.mark,

car.type, car.carPrice, car.yearOfProdaction, car.priceForDay));

}

Console.WriteLine("\n");

XmlDocument clientsDoc = new XmlDocument();

clientsDoc.Load("clients.xml");

Console.WriteLine("All clients: ");

foreach (XmlNode node in clientsDoc.DocumentElement)

{

Client client = new Client(Int32.Parse(node["idClient"].InnerText), node["fullName"].InnerText, node["adress"].InnerText, node["phone"].InnerText);

Console.WriteLine(string.Format("Id: {0}; Name: {1}; Adress: {2}; Phone: {3};", client.id, client.fullName, client.adress, client.phone));

}

Console.WriteLine("\n");

XmlDocument transactionsDoc = new XmlDocument();

transactionsDoc.Load("transactions.xml");

Console.WriteLine("All transactions: ");

foreach (XmlNode node in transactionsDoc.DocumentElement)

{

Transaction transaction = new Transaction(Int32.Parse(node["clientId"].InnerText), Int32.Parse(node["carId"].InnerText),

node["startDay"].InnerText, node["finishDay"].InnerText, Int32.Parse(node["deposit"].InnerText));

Console.WriteLine(string.Format("Client ID: {0}; Car ID: {1}; Start day: {2}; Finish day: {3}; Deposit: {4}", transaction.clientId, transaction.carId, transaction.startDay, transaction.finishDay, transaction.deposit));

}

Console.WriteLine("\n");

XDocument xmlCars = XDocument.Load("cars.xml");

Console.WriteLine("Sorted marks of cars");

var marksSorted = xmlCars.Descendants("car").Select(p => p.Element("mark").Value).OrderBy(p => p.Trim());

foreach (var car in marksSorted)

{

Console.WriteLine(car);

}

Console.WriteLine("\n");

Console.WriteLine("Cars sorted by dayly price");

var carsSorted =

(from car in xmlCars.Descendants("car")

orderby Int32.Parse(car.Element("priceForDay").Value) descending

select new

{

Id = car.Element("idCar").Value,

Mark = car.Element("mark").Value,

Type = car.Element("type").Value,

CarPrice = car.Element("carPrice").Value,

YearOfProduction = car.Element("yearOfProduction").Value,

DaylyPrice = car.Element("priceForDay").Value,

});

foreach (var car in carsSorted)

{

Console.WriteLine(car);

}

Console.WriteLine("\n");

XDocument xmlTransactions = XDocument.Load("transactions.xml");

Console.WriteLine("Available cars");

var idRentedCars = xmlTransactions.Descendants("transaction").Select(p => p.Element("carId").Value);

var carsAv =

from car in xmlCars.Descendants("car")

where !idRentedCars.Contains(car.Element("idCar").Value)

select new

{

Id = car.Element("idCar").Value,

Mark = car.Element("mark").Value,

Type = car.Element("type").Value,

CarPrice = car.Element("carPrice").Value,

YearOfProduction = car.Element("yearOfProduction").Value,

DaylyPrice = car.Element("priceForDay").Value,

};

foreach (var car in carsAv)

{

Console.WriteLine(car);

}

Console.WriteLine("\n");

XDocument xmlClients = XDocument.Load("clients.xml");

Console.WriteLine("Clients that rented cars");

var clientsRenId = xmlTransactions.Descendants("transaction").Select(p => p.Element("clientId").Value);

var clientsRen =

from client in xmlClients.Descendants("client")

where clientsRenId.Contains(client.Element("idClient").Value)

select new

{

Id = client.Element("idClient").Value,

Name = client.Element("fullName").Value,

Adress = client.Element("adress").Value,

Phone = client.Element("phone").Value,

};

foreach (var client in clientsRen)

{

Console.WriteLine(client);

}

Console.WriteLine("\n");

Console.WriteLine("Avarage dayly price: ");

var carsPrice = xmlCars.Descendants("car").Select(p => Int32.Parse(p.Element("priceForDay").Value));

Console.WriteLine(carsPrice.Average());

Console.WriteLine("\n");

Console.WriteLine("Amount of cars: ");

var carsAmount = xmlCars.Descendants("car").Count();

Console.WriteLine(carsAmount);

Console.WriteLine("\n");

Console.WriteLine("The chepiest car: ");

var chipCar = carsSorted.Last();

Console.WriteLine(chipCar);

}

static List<Client> Clients = new List<Client>()

{

new Client(1,"John Andersson","London","2809825659"),

new Client(2,"Vasiliy Ivanov","Vena","4589562045"),

new Client(3,"Arseniy Markow","Krakow","7856235453"),

new Client(4,"Oleg Petrov","Kyiv","380985648216"),

new Client(5,"Ann Black","New York","356498562")

};

static List<Car> Cars = new List<Car>()

{

new Car(1,"Renault Dokker","Hatchback",10846,2013),

new Car(2,"Honda Accord X","Sedan",54580,2018),

new Car(3,"Volkswagen Caddy","Hatchback",17927,2010),

new Car(4,"Audi A8","Sedan",128721,2017),

new Car(5,"Opel Astra J","Hatchback",15841,2012),

new Car(6,"Mercedes-Benz Citan","Minibus",24692,2013)

};

static List<Transaction> Transactions = new List<Transaction>()

{

new Transaction(1,4,"01.06.20","26.06.20",500),

new Transaction(2,2,"01.05.19","12.05.19",200)

};

}

public class Car

{

public int id;

public string mark;

public string type;

public int carPrice;

public int yearOfProdaction;

public int priceForDay;

public Car(int id, string m, string t, int cp, int yp)

{

this.id = id;

this.mark = m;

this.type = t;

this.carPrice = cp;

this.yearOfProdaction = yp;

this.priceForDay = (carPrice / 100) / (DateTime.Now.Year - this.yearOfProdaction);

}

public override string ToString()

{

return "(Mark: " + this.mark + "; Type: " + this.type +

"; Car price: " + this.carPrice + "; Year of production: " + this.yearOfProdaction + "; Price for day:" + this.priceForDay + ")";

}

}

public class Client

{

public int id;

public string fullName;

public string adress;

public string phone;

public Client(int id, string fn, string a, string p)

{

this.id = id;

this.fullName = fn;

this.adress = a;

this.phone = p;

}

public override string ToString()

{

return "(Fullname: " + this.fullName + "; Adress: " + this.adress +

"; Phone number: " + this.phone + ")";

}

}

public class Transaction

{

public int clientId;

public int carId;

public string startDay;

public string finishDay;

public int deposit;

public Transaction(int clientid, int carid, string sd, string fd, int d)

{

this.clientId = clientid;

this.carId = carid;

this.startDay = sd;

this.finishDay = fd;

this.deposit = d;

}

public override string ToString()

{

return "(Client: " + this.clientId + "; Car: " + this.carId +

"; Start Day: " + this.startDay + "; Finish day:" + this.finishDay + "; Deposit: " + this.deposit + ")";

}

}

**Висновок:**Я засвоїв теоретичний матеріал з відповідної теми та закріпив його,виконавши практичні завдання.