МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ОРЛОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИМЕНИ И.С. ТУРГЕНЕВА»

Кафедра программной инженерии

**ОТЧЕТ**

по лабораторной работе № 6

на тему: «Создание и развёртывание web-сервисов»

по дисциплине: «Проектирование и архитектура программных систем»

Выполнили: Кожухова О.А., Карпикова С.П., Макеева Д.С.

Институт приборостроения, автоматизации и информационных технологий

Направление: 09.03.04 «Программная инженерия»

Группа: 71-ПГ

Проверили: Ужаринский А.Ю., Константинов И.С.

Отметка о зачете:

Дата: «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2019 г.

Орел, 2019 г.

**Задание на лабораторную работу.**

Разработать web-сервис, предоставляющий информацию о рейсах и наличии билетов на эти рейсы по запросам пользователей. Должна быть возможность просмотреть все рейсы по заданному маршруту, просмотреть наличие билетов, заказать билет на рейс.

Реализовать клиентское приложение для работы с данным сервисом. (Клиентское приложение и web-сервис могут быть реализованы на разных языках программирования). Для реализации web-сервиса рекомендуется использовать библиотеку WCF платформы .NET.

**Выполнение работы.**

Веб-сервис:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

using System.ServiceModel;

using System.Text;

namespace FlightService

{

// ПРИМЕЧАНИЕ. Команду "Переименовать" в меню "Рефакторинг" можно использовать для одновременного изменения имени интерфейса "IFlightService" в коде и файле конфигурации.

[ServiceContract]

public interface IFlightService

{

[OperationContract]

List<string> ListFlights(string goFrom, string goTo, string date);

[OperationContract]

int GetID(string goFrom, string goTo, string date, string time);

[OperationContract]

int GetCost(int id, int cnt);

[OperationContract]

int Book(int id, int cnt);

[OperationContract]

int GetCnt(int id);

}

}

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Runtime.Serialization;

using System.ServiceModel;

using System.Text;

namespace FlightService

{

public class FlightService : IFlightService

{

public List<string> ListFlights(string goFrom, string goTo, string date)

{

List<string> flights = new List<string>();

FileStream fstream = new FileStream(@"C:\Users\user\Desktop\Универ\Лабораторные\Лабы С#\FlightService\FlightService\App\_Data\flights.txt", FileMode.OpenOrCreate);

string s;

using (StreamReader stream = new StreamReader(fstream, System.Text.Encoding.Default)) {

while((s = stream.ReadLine()) != null)

{

string[] all = s.Split(' ');

if (all[1] == goFrom && all[2] == goTo && all[3] == date)

{

flights.Add(s);

}

}

}

fstream.Close();

return flights;

}

public int GetID(string goFrom, string goTo, string date, string time)

{

FileStream fstream = new FileStream(@"C:\Users\user\Desktop\Универ\Лабораторные\Лабы С#\FlightService\FlightService\App\_Data\flights.txt", FileMode.OpenOrCreate);

string s;

int id = -1;

using (StreamReader stream = new StreamReader(fstream, System.Text.Encoding.Default))

{

while ((s = stream.ReadLine()) != null)

{

string[] all = s.Split(' ');

if (all[1] == goFrom && all[2] == goTo && all[3] == date && all[4] == time)

{

id = Convert.ToInt32(all[0]);

}

}

}

fstream.Close();

return id;

}

public int GetCnt(int id)

{

FileStream fstream = new FileStream(@"C:\Users\user\Desktop\Универ\Лабораторные\Лабы С#\FlightService\FlightService\App\_Data\flights.txt", FileMode.OpenOrCreate);

string s;

int cnt = 0;

using (StreamReader stream = new StreamReader(fstream, System.Text.Encoding.Default))

{

while ((s = stream.ReadLine()) != null)

{

string[] all = s.Split(' ');

if (Convert.ToInt32(all[0]) == id)

{

cnt = Convert.ToInt32(all[5]);

}

}

}

fstream.Close();

return cnt;

}

public int GetCost(int id, int cnt)

{

FileStream fstream = new FileStream(@"C:\Users\user\Desktop\Универ\Лабораторные\Лабы С#\FlightService\FlightService\App\_Data\flights.txt", FileMode.OpenOrCreate);

string s;

int cost = -1;

using (StreamReader stream = new StreamReader(fstream, System.Text.Encoding.Default))

{

while ((s = stream.ReadLine()) != null)

{

string[] all = s.Split(' ');

if (Convert.ToInt32(all[0]) == id)

{

cost = Convert.ToInt32(all[6]) \* cnt;

}

}

}

fstream.Close();

return cost;

}

public int Book(int id, int cnt)

{

FileStream fstream = new FileStream(@"C:\Users\user\Desktop\Универ\Лабораторные\Лабы С#\FlightService\FlightService\App\_Data\flights.txt", FileMode.OpenOrCreate);

string s;

int request = -1;

Random rnd = new Random();

string one = "";

string str = "";

using (StreamReader stream = new StreamReader(fstream, System.Text.Encoding.Default))

{

while ((s = stream.ReadLine()) != null)

{

str += s + Environment.NewLine;

string[] all = s.Split(' ');

if (Convert.ToInt32(all[0]) == id)

{

int cntMax = Convert.ToInt32(all[5]);

if (cnt <= cntMax)

{

request = rnd.Next(500000, 15226262);

cntMax -= cnt;

all[5] = cntMax.ToString();

for (int i = 0; i < all.Length - 1; i++)

{

one += all[i] + ' ';

}

one += all[all.Length - 1];

break;

}

}

}

}

if (request != -1)

{

FileStream fstream1 = new FileStream(@"C:\Users\user\Desktop\Универ\Лабораторные\Лабы С#\FlightService\FlightService\App\_Data\flights.txt", FileMode.OpenOrCreate, FileAccess.Write);

str = str.Remove(str.IndexOf(s), s.Length).Insert(str.IndexOf(s), one);

using (System.IO.StreamWriter file = new System.IO.StreamWriter(fstream1, Encoding.Default))

file.Write(str);

fstream1.Close();

}

return request;

}

}

}

Клиент:

using Client.FlightService;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Client

{

public partial class Form1 : Form

{

private Form2 form2;

FlightServiceClient client = new FlightServiceClient();

public Form1()

{

InitializeComponent();

}

private void getFlights\_Click(object sender, EventArgs e)

{

List<string> flights = new List<string>();

int i = 0;

flights = client.ListFlights(comboBox1.Text, comboBox2.Text, dateTimePicker1.Value.ToString("dd.MM.yyyy")).ToList();

if (flights.Count == 0)

MessageBox.Show("Информация о рейсах по заданному направлению отсутствует");

dataGridView1.Rows.Clear();

btnBook.Enabled = true;

foreach (string s in flights)

{

string[] all = s.Split(' ');

dataGridView1.Rows.Add();

dataGridView1.Rows[i].Cells[0].Value = all[1];

dataGridView1.Rows[i].Cells[1].Value = all[2];

dataGridView1.Rows[i].Cells[2].Value = all[3];

dataGridView1.Rows[i].Cells[3].Value = all[4];

dataGridView1.Rows[i].Cells[4].Value = all[6];

i++;

}

//client.Close();

}

private void btnBook\_Click(object sender, EventArgs e)

{

try

{

int id = dataGridView1.CurrentRow.Index;

int idArrive = client.GetID(dataGridView1.Rows[id].Cells[0].Value.ToString(), dataGridView1.Rows[id].Cells[1].Value.ToString(), dataGridView1.Rows[id].Cells[2].Value.ToString(), dataGridView1.Rows[id].Cells[3].Value.ToString());

int cnt = client.GetCnt(idArrive);

if (cnt == 0)

{

MessageBox.Show("Билетов на рейс больше нет в наличии!");

}

else

{

form2 = new Form2(idArrive) { Visible = false };

form2.Owner = this;

form2.Visible = true;

form2.FormClosed += new FormClosedEventHandler(frm\_FormClosed);

this.Visible = false;

}

}

catch

{

MessageBox.Show("Выберете направление!");

}

}

void frm\_FormClosed(object sender, FormClosedEventArgs e)

{

this.Visible = true;

}

}

}

using System;

using Client.FlightService;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Client

{

public partial class Form2 : Form

{

public int id;

Form1 main;

FlightServiceClient client = new FlightServiceClient();

public Form2(int idArrive)

{

main = this.Owner as Form1;

InitializeComponent();

id = idArrive;

txtCnt.Text += "1";

int cnt = client.GetCnt(id);

textBox1.Text =cnt.ToString();

int cost = client.GetCost(id, Convert.ToInt32(txtCnt.Text));

textBox3.Text = cost.ToString();

}

private void btnBook\_Click(object sender, EventArgs e)

{

int cnt = Convert.ToInt32(txtCnt.Text);

int request = client.Book(id, cnt);

if (request == -1)

MessageBox.Show("В наличии отсутствует заданное количество билетов!");

else

{

TextBox txtReq = new TextBox();

txtReq.Location = new System.Drawing.Point(372, 236);

txtReq.Size = new System.Drawing.Size(240, 22);

Controls.Add(txtReq);

txtReq.Text = request.ToString();

Label label = new Label();

label.Location = new System.Drawing.Point(200, 236);

label.Size = new System.Drawing.Size(200, 17);

Controls.Add(label);

label.Text = "Номер вашей транзакции:";

}

}

private void button1\_Click(object sender, EventArgs e)

{

int cnt = Convert.ToInt32(txtCnt.Text);

cnt++;

button2.Enabled = true;

txtCnt.Text = cnt.ToString();

textBox3.Text = client.GetCost(id, Convert.ToInt32(txtCnt.Text)).ToString();

}

private void button3\_Click(object sender, EventArgs e)

{

Close();

}

private void button2\_Click(object sender, EventArgs e)

{

int cnt = Convert.ToInt32(txtCnt.Text);

cnt--;

if (cnt == 1)

button2.Enabled = false;

txtCnt.Text = cnt.ToString();

textBox3.Text = client.GetCost(id, Convert.ToInt32(txtCnt.Text)).ToString();

}

}

}

**Контрольные вопросы.**

*1. Какие возможности предоставляет платформа .NET для создания web-сервисов?*

Создание веб-сервисов на платформе .NET – довольно простой процесс. Для этого не требуется знания о протоколах HTTP и SOAP или языке описания веб-сервисов WSDL. Среда разработки, используя библиотеку по работе с веб-сервисами самостоятельно создает прокси-классы веб-сервисов, скрывая, таким образом процесс взаимодействия с ними от разработчика.

*2. Опишите основные принципы работы платформы WCF.*

Принцип 1: границы установлены явно. Этот принцип подчеркивает тот факт, что функциональность службы WCF выражается через четко определенные интерфейсы (т.е. описания каждого члена, его параметров и возвращаемых значений). Единственный способ, которым внешний клиент может связаться со службой WCF, — через интерфейс, при этом оставаясь в блаженном неведении о деталях ее внутренней реализации.

Принцип 2: службы автономны. Говоря о службах, как об автономных сущностях, имеется в виду тот факт, что каждая служба WCF является (насколько возможно) отдельным "островом". Автономная служба должна быть независимой от проблем с версиями, развертыванием и установкой. Чтобы помочь в продвижении этого принципа, мы опять возвращаемся к ключевому аспекту программирования на основе интерфейсов. Как только интерфейс внедрен, он никогда не должен изменяться (или вы рискуете разрушить существующие клиенты). Когда требуется расширить функциональность службы WCF, просто напишите новый интерфейс, который моделирует необходимую функциональность.

Принцип 3: службы взаимодействуют через контракт, а не реализацию.Третий принцип — еще один побочный продукт программирования на основе интерфейсов — состоит в том, что реализация деталей службы WCF (на каком языке она написана, как именно выполняет свою работу, и т.п.) не касается вызывающего ее внешнего клиента. Клиенты WCF взаимодействуют со службами исключительно через их открытые интерфейсы. Более того, если члены службы представляют сложные специальные типы, они должны быть полностью детализированы в виде контракта данных, гарантируя, что клиенты смогут отобразить содержимое на определенную структуру данных.

Принцип 4: совместимость служб базируется на политике.Поскольку интерфейсы CLR предоставляют строго типизированные контракты всем клиентам WCF (и также могут быть использованы для генерации соответствующего документа WSDL на основе выбранной привязки), важно понимать на то, что интерфейсы и WSDL сами по себе недостаточно выразительны, чтобы детализировать аспекты того, что способна делать служба. Учитывая это, SOA позволяет определять политики, которые еще более проясняют семантику службы (например, ожидаемые требования безопасности, применяемые для общения со службой). Используя эти политики, можно отделять низкоуровневые синтаксические описания службы (предоставляемые интерфейсы) от семантических деталей их работы и способов их вызова.

*3. Что такое контракт в терминологии WCF?*

WCF содержат три вида контрактов:

1) сервисов для описания функциональных операций, реализованных сервисом. Внутри контракта сервиса имеются контракты об операциях, как отдельные операции сервиса, которые реализуют функции;

2) данных, определяющих формат данных, которыми будут обмениваться сервисы. Это относится как к запросу на сервис, так и к ответу сервиса. Если используются примитивные типы - int, string и др., то контракт не нужен, потому что .Net понимается как сериализация и дисериализация типов. В случае применения комплексных типов — Customers, Order и др., необходимо указать принцип сериалиализации и дисериализаии этих объектов;

3) сообщений, как тип контракта, который используется для того, чтобы получить контроль над заглавном SOAP пакета.

*4. Опишите механизм построения клиентского приложения на основе описания web-сервиса.*

Стандарты веб-сервисов обычно используются совместно и согласованно. После обнаружения WSDL в UDDI или другом месте генерируется SOAP-сообщение для отправки на удаленный сайт.

При предоставлении документа по адресу веб-сервиса программа использует XML-схему определенного типа, позволяющую преобразовать данные из ее входного источника и на основе того же WSDL-файла создать экземпляр XML-документа в формате, согласованном с целевым веб-сервисом. WSDL-файл используется для определения как входного, так и выходного преобразования данных.

SOAP-процессор отправляющего компьютера преобразует данные из собственного ("родного") формата в тип данных, предопределенный в соответствии с содержащейся в WSDL-файле XML-схемой на основе таблиц преобразования для текстов, значений с плавающей точкой и других данных. Таблицы преобразования "связывают" собственные типы данных с соответствующими конкретной XML-схеме. SOAP-процессор получающего компьютера выполняет обратное преобразование данных из типов XML-схемы в собственные типы данных.

Файлы описаний веб-сервисов обычно регистрируются с помощью URL. URL, повсеместно используемый в Сети, указывает на IP-адрес, соответствующий веб-ресурсу. Схемы веб-сервисов являются одной из форм веб-ресурса, они содержатся в доступных через Интернет файлах и к ним применим тот же механизм, что используется при загрузке HTML-файлов. Главное отличие между загрузкой HTML-файла и обращением к ресурсу веб-сервиса заключается в том, что веб-сервис оперирует XML-документами, а не HTML-документами и опирается на соответствующие технологии, такие как использование схем, преобразование, проверка подлинности, что и обеспечивает поддержку удаленного соединения приложений. Но способ, согласно которому схемы веб-сервисов публикуются и загружаются, одинаков: HTTP-операция по указанному URL.

Для проверки достоверности сообщений веб-сервисы используют XML-схемы. После получения документа реализация веб-сервиса сначала должна проанализировать XML-сообщение и удостовериться в корректности данных, выполнить проверку качества услуг, такую как проверку политики безопасности или соглашений торговых партнеров, а затем произвести последовательность связанных с данным документом коммерческих операций.

Веб-сервис, доступный по данному интернет-адресу, идентифицируется с помощью публичного WSDL-файла, который может быть загружен на отправляющий компьютер и использоваться при генерации сообщения.

Технологии веб-сервисов сформировались из основных структур. Этих технологий достаточно для построения, развертывания и публикации базовых веб-сервисов. На самом деле, необходим лишь базовый протокол SOAP. С момента появления веб-сервисов к ним все время добавляются другие технологии. Хотя для деловой связи, а также для создания "моста" с несовместимыми технологиями вполне достаточно базовых принципов, данная форма веб-взаимодействия тем не менее была одобрена очень быстро.

*5. Опишите принципы развертывания wcf-службы.*

Развертывание службы Windows Communication Foundation (WCF), который размещается в Internet Information Services (IIS) состоит из следующих задач:

* Убедитесь, что службы IIS, ASP.NET, WCF и компонента активации WCF службы правильно установлен и зарегистрирован.
* Создание нового приложения IIS или повторно использовать существующее приложение ASP.NET.
* Создание SVC-файла для службы WCF.
* Развертывание реализации службы в приложение IIS.
* Настройка службы WCF.