**Липецкий государственный технический университет**

Факультет автоматизации и информатики

Кафедра автоматизированных систем управления

ОТЧЁТ

об учебной практике в ЛГТУ

Студент Станиславчук С.М.

Группа АС-21-1

Руководитель

Старший преподаватель Муравейко А.Ю.

Липецк 2023 г.

**Задание кафедры:**

Создать структуру данных и реализовать способ их обработки в форме на C#, обеспечить получение данных с веб-сервиса, их отображение на форме, сохранение и загрузка в файл, поиск с помощью регулярных выражений, построение графика и его импорт в графическом формате.

Вариант 7.

Курс валют (https://api.privatbank.ua): получить наименования всех городов, где расположены терминалы самообслуживания данного банка, отсортировать их по алфавиту и сохранить в текстовых файлах в кодировках UTF-8, UTF-16.

Аннотация

С.23. Ил. 11. Литература 5 назв. Прил. 5

Отчет содержит результаты выполнения индивидуального домашнего задания в рамках учебной практики. В работе создана программа на языке С#, которая позволяет получать отсортированные наименования городов, где расположены терминалы самообслуживания банка PrivateBank, а также сохранять их в текстовом файле в кодировках UTF-8, UTF-16.

Представлено описание программы и сопутствующие сведения.

Оглавление

[Ход выполнения задания по учебной практике 5](#_Toc22700)

[1 Получение результата запроса от веб-сервиса 5](#_Toc22701)

[2 Сортировка городов и запись в файл 7](#_Toc22702)

[3 Отображение данных на форме 9](#_Toc22703)

[4 Поиск терминала по городу 10](#_Toc22704)

[Заключение 12](#_Toc22705)

[Библиографический список 13](#_Toc22706)

[Приложение А 14](#_Toc22707)

Приложение Б .................................................................................................... 19

Приложение В .................................................................................................... 20

Приложение Г .................................................................................................... 21

Приложение Д ................................................................................................... 22

# Ход выполнения задания по учебной практике

## 1. Получение результата запроса от веб-сервиса

Для начала на сайте https://api.privatbank.ua/ находим раздел

«Инфраструктура PrivateBank», «Терминалы самообслуживания» и берем оттуда нужное нам API. В итоге, для получения данных с веб-сервиса необходимо отправить следующий запрос:

<https://api.privatbank.ua/p24api/infrastructure?json&tso&address=&city>[=](https://api.privatbank.ua/p24api/infrastructure?json&tso&address&city)

Ответ на данный запрос представлен на рисунке 1.

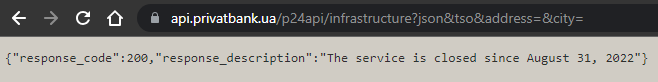


Рисунок 1 – Ответа от веб-сервиса (25.06.2023)

Как видим, на данный момент этот сервис закрыт, но данные все ещё можно просмотреть, если сохранилась резервная копия в глобальном веб-архиве.

Для этого воспользуемся сервисом Wayback Machine (<http://web.archive.org/>).

Ввел интересующий меня URL и мне повезло, что было 2 сохранения этой страницы как раз перед остановкой работы сервиса. Данные представлены на рисунке 2.

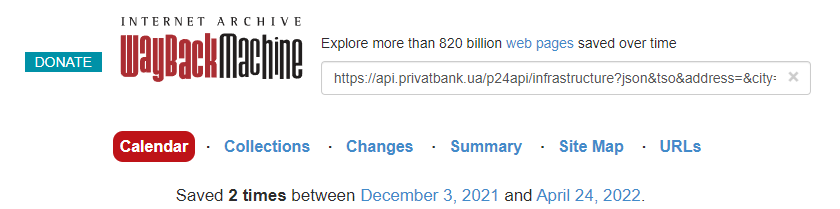


Рисунок 2 – Отчет о сохранениях интересующей меня страницы сервиса Wayback Machine.

Просмотрим данные за 24 апреля 2022 года. Полученные данные представлены на рисунке 3.

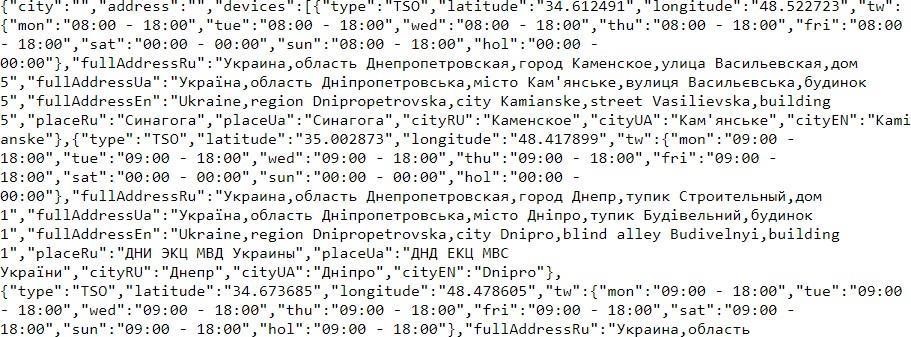


Рисунок 3 – Часть ответа от веб-сервиса (за 24.04.2022)

Создадим модель элемента ответа тех полей, которые нас интересуют.

Модель элемента ответа изображена на рисунке 4.

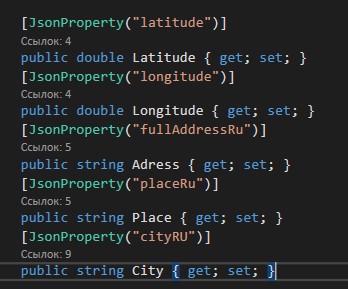


Рисунок 4 – Модель элемента ответа от веб-сервиса

Запишем информацию в массив данных. Фрагмент массива данных отображен на рисунке 5.

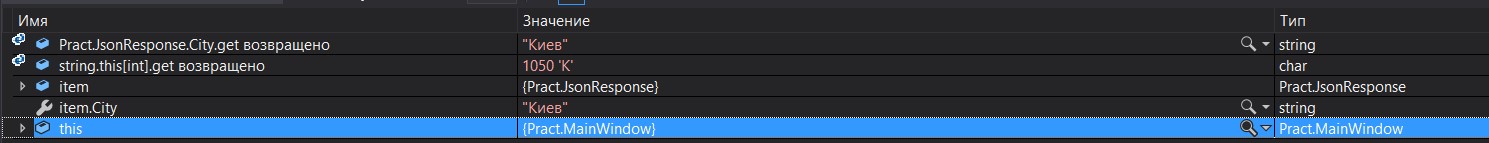


Рисунок 5 – Запись ответа от веб-сервиса в массив данных

## 2. Сортировка городов и запись в файл

Нам необходимо реализовать запись полученных данных в текстовые файлы с кодировкой UTF-8 и UTF-16, но при этом города должны быть отсортированы по алфавиту.

Сортировка данных по алфавиту:

var sortedItems = items.OrderBy(o => o.City).ToList();

Сортировка данных по названию городов представлена на рисунке 6.

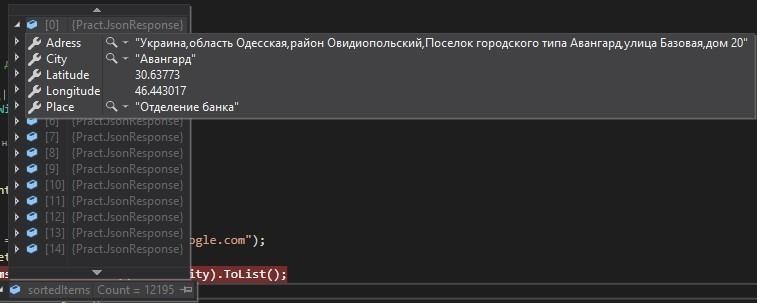


Рисунок 6 – Сортировка по названию города

Для сохранения файлов в форматах UTF-8, UTF-16 был написан метод writeToFile:

private void writeToFile(in List<JsonResponse> jsons)

{ string fileName = "DataTerminal"; using (StreamWriter writeText = new

StreamWriter(File.Open($"{fileName}\_UTF8.txt", FileMode.Create), System.Text.Encoding.UTF8))

{ string line = string.Empty; foreach (var item in jsons)

{

line = $"Город: {item.City} Местоположение: {item.Place} Адрес:

{item.Adress} Широта: {item.Latitude} Долгота: {item.Longitude}";

writeText.WriteLine(line);

} }

using (StreamWriter writeText = new

StreamWriter(File.Open($"{fileName}\_UTF16.txt", FileMode.Create), System.Text.Encoding.Unicode))

{ string line = string.Empty; foreach (var item in jsons)

{ line = $"Город: {item.City} Местоположение: {item.Place} Адрес:

{item.Adress} Широта: {item.Latitude} Долгота: {item.Longitude}";

writeText.WriteLine(line);

}

}

}

Для реализации метода использовано пространство имен StreamWriter, чтобы файл кодировался, как требуется по заданию. Запись в файл происходит автоматически, а в папке с исполняемой программой появятся два текстовых файла DataTerminal(UTF8).txt и DataTerminal(UTF16).txt

На рисунках 7 и 8 изображены записи данных в файлы форматов UTF-8 и UTF-16.

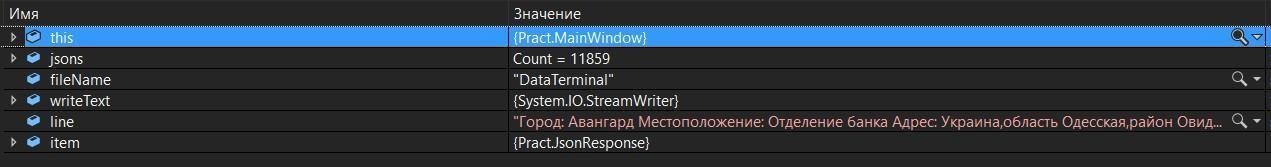


Рисунок 7 – Запись данных в формат UTF-8

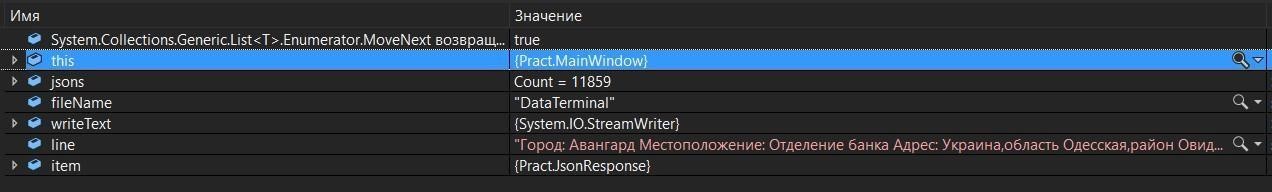


Рисунок 8 – Запись данных в формат UTF-16

Результат сохранения данных изображен на рисунке 9.



Рисунок 9 – Результат сохранения данных

Примеры сохранения данных в текстовые файлы в кодировках UTF-8 и UTF-16 представлены соответственно на рисунках 10 и 11.

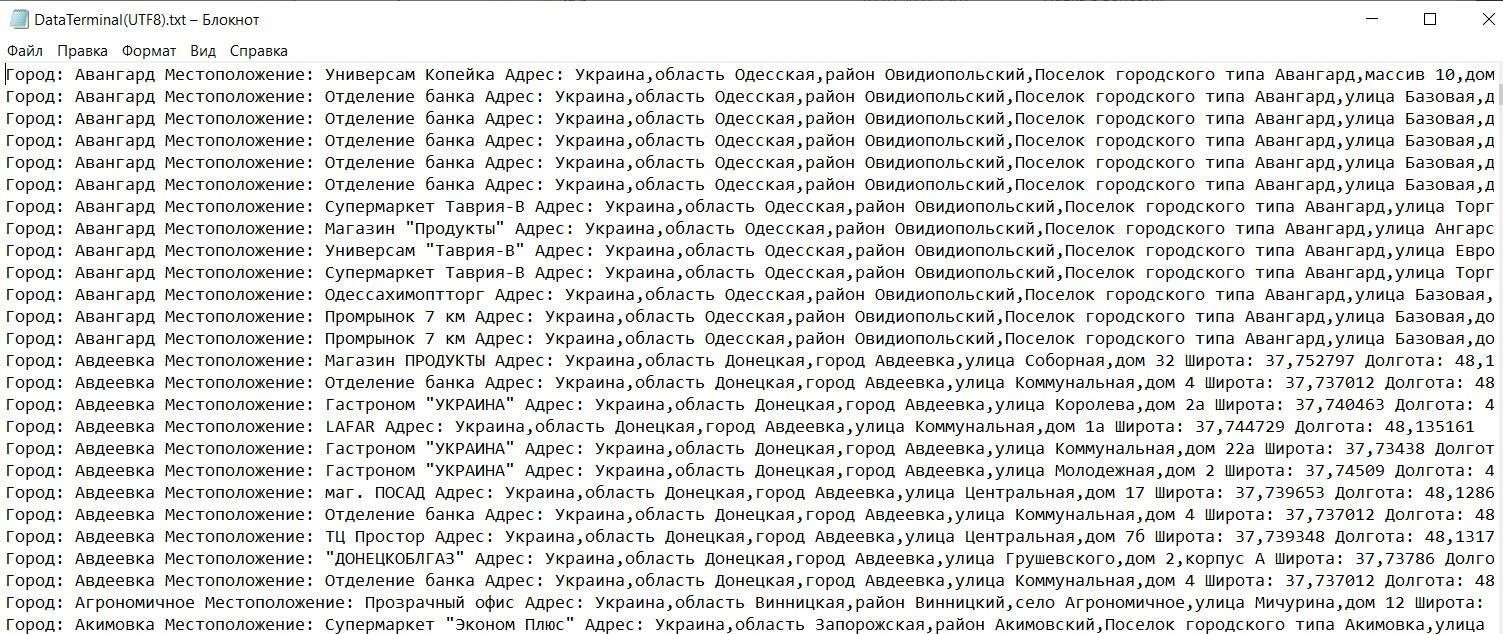


Рисунок 10 – Пример сохранения данных в текстовый файл в кодировке UTF-8

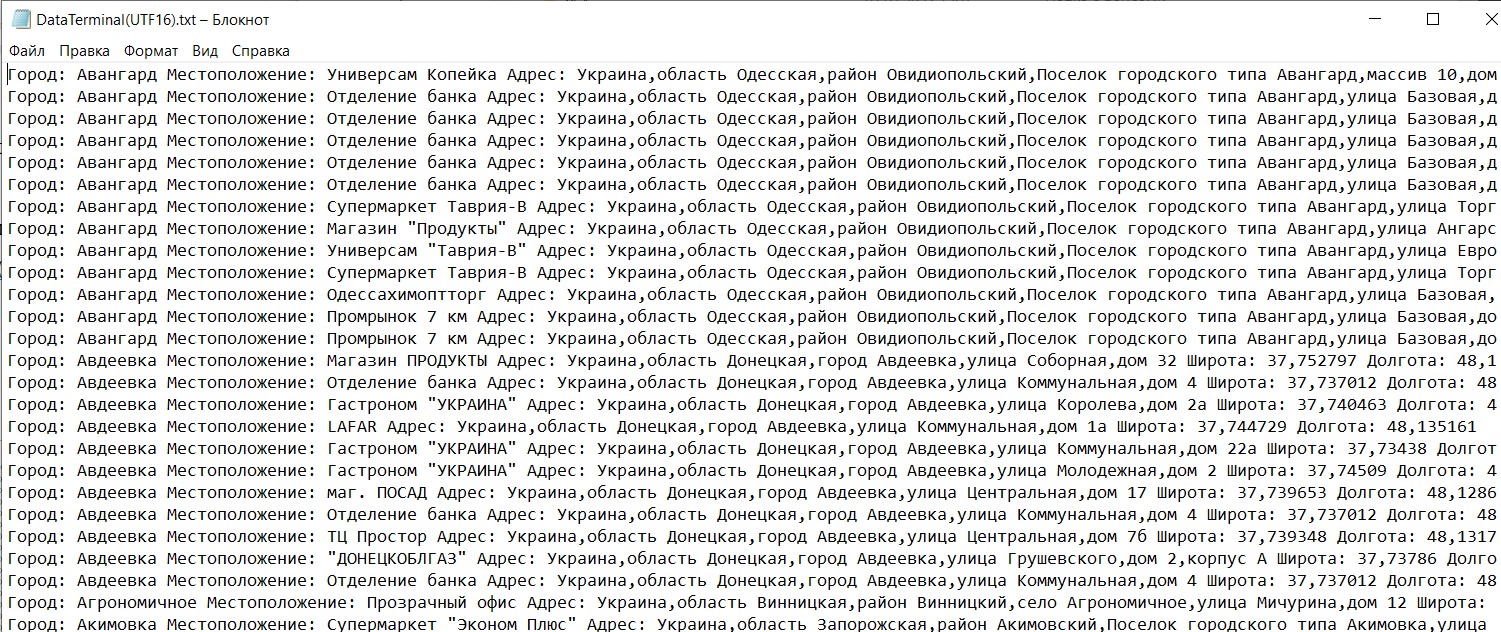


Рисунок 11 – Пример сохранения данных в текстовый файл в кодировке

UTF-16

## 3. Отображение данных на форме

Отображение отсортированных наименований городов, где расположены терминалы самообслуживания банка PrivatBank, будет реализовано в среде разработки Microsoft Visual Studio. Приложение будет состоять из одного окна, на котором будет отображаться результат запросов, и изображения терминалов PrivatBank.

Окно приложения изображено на рисунке 12.

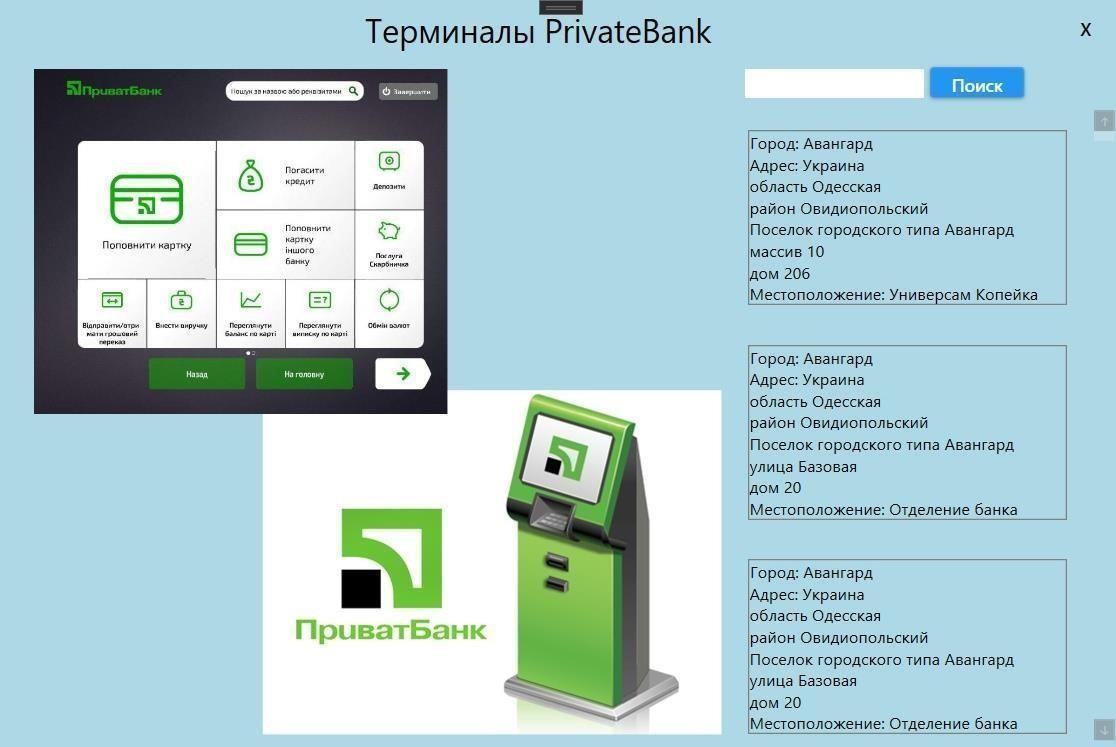


Рисунок 12 – GUI приложения “Терминалы PrivateBank”

## 4. Поиск терминала по городу

Дополнительно реализован поиск терминалов по городу. Из TextBox считывается строка и добавляется к API.

Функция поиска:

private void Search\_Click(object sender, RoutedEventArgs e)

{ try

{ string str = BoxCity.Text.ToString();

IPHostEntry es = Dns.GetHostEntry("["](http://www.google.com/)); var items = GGetItems(str); var sortedItems = items.OrderBy(o => o.City).ToList(); writeToFile(in sortedItems); listView\_items.Items.Clear(); foreach (var item in sortedItems)

{ listView\_items.Items.Add(new CustomListViewItem(new JsonResponse {

Latitude = item.Latitude,

Longitude = item.Longitude,

Adress = item.Adress,

Place = item.Place,

City = item.City

}));

}

}

catch (Exception)

{

MessageBox.Show("Проверьте подключение к сети.", "Ошибка!",

MessageBoxButton.OK, MessageBoxImage.Error);

Application.Current.Shutdown();

}

if(BoxCity.Text ==" ")

{

MaiN();

}

}

Пример поиска терминалов в определенном городе представлен на рисунке 13.



Рисунок 13 – Поиск терминалов в городе «Одесса»

# Заключение

В результате выполнения практической работы, была разработана программа на языке программирования высокого уровня C# в среде разработки Microsoft Visual Studio. Данная программа предназначена для получения отсортированных наименований городов, их просмотра и записи в текстовые файлы в кодировках UTF-8, UTF-16. В ходе работы была изучена работа с API веб-сервиса, работа с библиотекой Newtonsoft.Json и функциями, которые позволяют кодировать и раскодировать файлы в различных кодировках. Полученная программа имеет графический интерфейс, реализованный с помощью WPF.

# Библиографический список

1. Работа с библиотекой Newtonsoft.Json на реальном примере. Часть 1 [Электронный ресурс]: коллективный it-блог. – Режим доступа: [WWW.URL:](http://www.url/) https://habr.com/ru/post/481514/, свободный – 30.06.2023.
2. Работа с библиотекой Newtonsoft.Json на реальном примере. Часть 2 [Электронный ресурс]: коллективный it-блог. – Режим доступа: [WWW.URL:](http://www.url/) https://habr.com/ru/post/482042/, свободный – 30.06.2023.
3. Связь между C# и C#: REST, gRPC и все, что между [Электронный ресурс]: коллективный it-блог. – Режим доступа: [WWW.URL:](http://www.url/) https://habr.com/ru/post/488102/, свободный – 30.06.2023.
4. Харрисон Ферроне. Изучаем C# через разработку игр на Unity 2020 (пятое издание) [Книга]

# Приложение А

Файл MainWindow.xaml.cs

using GMap.NET;

using GMap.NET.MapProviders; using Newtonsoft.Json; using Newtonsoft.Json.Linq; using System;

using System.Collections.Generic; using System.IO;

using System.Linq;

using System.Net;

using System.Windows;

using System.Windows.Input;

namespace Pract

{

/// <summary>

/// Логика взаимодействия для MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{ public MainWindow()

{

InitializeComponent();

MaiN();

}

public void MaiN()

{

try

{

IPHostEntry e = Dns.GetHostEntry("[www.google.com"](http://www.google.com/)); var items = getItems();

var sortedItems = items.OrderBy(o => o.City).ToList(); WriteFile(in sortedItems); listView\_items.Items.Clear(); foreach (var item in sortedItems)

{ listView\_items.Items.Add(new DataTerminal (new ResponseProcessingJson

{

Latitude = item.Latitude,

Longitude = item.Longitude,

Adress = item.Adress,

Place = item.Place,

City = item.City

}));

}

}

catch (Exception)

{

MessageBox.Show("Проверьте подключение к сети.", "Ошибка!",

MessageBoxButton.OK, MessageBoxImage.Error);

Application.Current.Shutdown();

}

}

private void Window\_MouseDown(object sender, MouseButtonEventArgs e)

{ if (e.ChangedButton == MouseButton.Left) DragMove();

}

private void Close\_Click(object sender, RoutedEventArgs e)

{

System.Windows.Application.Current.Shutdown();

}

private List< ResponseProcessingJson> GetItems()

{

WebRequest request =

WebRequest.Create("https://api.privatbank.ua/p24api/infrastructure?json&tso&address=& city=");

var response = request.GetResponse(); var resonseStream = response.GetResponseStream(); var reader = new StreamReader(resonseStream); var data = reader.ReadToEnd(); reader.Close();

var innerJson = JObject.Parse(data)["devices"].ToString(); var jsonData = JsonConvert.DeserializeObject<List<ResponseProcessingJson >>(innerJson);

fixCityName(ref jsonData); return jsonData;

}

private void listView\_items\_MouseLeftButtonUp(object sender, MouseButtonEventArgs e)

{ var item = (CustomListViewItem)listView\_items.SelectedItem; if (item != null)

{ var json = item.ResponseProcessingJson

;

}

}

private void fixCityName(ref List<ResponseProcessingJson> jsons)

{ foreach (var item in jsons)

{ if (item.City[0] == ' ')

{ item.City = item.City.Replace(" ", string.Empty); } }

jsons = jsons.Distinct().ToList();

}

private void writeToFile(in List<ResponseProcessingJson> jsons)

{ string fileName = "DataTerminal"; using (StreamWriter writeText = new

StreamWriter(File.Open($"{fileName}(UTF8).txt", FileMode.Create), System.Text.Encoding.UTF8))

{ string line = string.Empty; foreach (var item in jsons)

{ line = $"Город: {item.City} Местоположение: {item.Place} Адрес:

{item.Adress} Широта: {item.Latitude} Долгота: {item.Longitude}"; writeText.WriteLine(line);

} }

using (StreamWriter writeText = new

StreamWriter(File.Open($"{fileName}(UTF16).txt", FileMode.Create), System.Text.Encoding.Unicode))

{ string line = string.Empty; foreach (var item in jsons)

{ line = $"Город: {item.City} Местоположение: {item.Place} Адрес:

{item.Adress} Широта: {item.Latitude} Долгота: {item.Longitude}";

writeText.WriteLine(line);

}

} }

private List<ResponseProcessingJson> GGetItems(string str)

{

WebRequest request =

WebRequest.Create("https://api.privatbank.ua/p24api/infrastructure?json&tso&address=& city=" + str);

var response = request.GetResponse(); var resonseStream = response.GetResponseStream(); var reader = new StreamReader(resonseStream); var data = reader.ReadToEnd(); reader.Close();

var innerJson = JObject.Parse(data)["devices"].ToString();

var jsonData = JsonConvert.DeserializeObject<List<ResponseProcessingJson

>>(innerJson);

fixCityName(ref jsonData); return jsonData;

}

private void Search\_Click(object sender, RoutedEventArgs e)

{

try

{

IPHostEntry es = Dns.GetHostEntry("[www.google.com"](http://www.google.com/)); var items = GGetItems(BoxCity.Text.ToString()); var sortedItems = items.OrderBy(o => o.City).ToList(); WriteFile(in sortedItems); listView\_items.Items.Clear();

foreach (var item in sortedItems)

{ listView\_items.Items.Add(new DataTerminal (new ResponseProcessingJson

{

Latitude = item.Latitude,

Longitude = item.Longitude,

Adress = item.Adress,

Place = item.Place,

City = item.City

}));

} }

catch (Exception)

{

MessageBox.Show("Проверьте подключение к сети.", "Ошибка!",

MessageBoxButton.OK, MessageBoxImage.Error);

Application.Current.Shutdown();

}

if(BoxCity.Text ==" ")

{

MaiN();

}

}

}

}

Б

Файл DataTerminal.xaml.cs

using System.Windows.Controls; namespace Pract

{

/// <summary>

/// Логика взаимодействия для DataTerminal.xaml

/// </summary>

public partial class DataTerminal : UserControl

{ public ResponseProcessingJson ResponseProcessingJson { get; set; } public DataTerminal ()

{

InitializeComponent();

}

public DataTerminal (ResponseProcessingJson item)

{

InitializeComponent(); ResponseProcessingJson = item; tb\_city.Text = $"Город: {item.City}"; var adress = item.Adress.Split(','); string line = string.Empty; int index = adress.Length - 1; foreach (var adr in adress)

{ if (adr != adress[index])

{

line += $"{adr}\n";

}

else

{

line += $"{adr}";

}

}

tb\_adress.Text = $"Адрес: {line}"; tb\_place.Text = $"Местоположение: {item.Place}"; }

}

}

В

Файл ResponseProcessingJson.cs

using Newtonsoft.Json; namespace Pract

{

public class ResponseProcessingJson

{

[JsonProperty("Latitude")] public double Latitude { get; set; } [JsonProperty("Longitude")] public double Longitude { get; set; } [JsonProperty("Address")] public string Adress { get; set; } [JsonProperty("Place")] public string Place { get; set; } [JsonProperty("City")] public string City { get; set; }

}

}

Г

Файл DataTerminal.xaml

<UserControl x:Class="Pract.DataTerminal"

xmlns="[http://schemas.microsoft.com/winfx/2006/xaml/presentation"](http://schemas.microsoft.com/winfx/2006/xaml/presentation) xmlns:x="[http://schemas.microsoft.com/winfx/2006/xaml"](http://schemas.microsoft.com/winfx/2006/xaml)

xmlns:mc="[http://schemas.openxmlformats.org/markup-compatibility/2006"](http://schemas.openxmlformats.org/markup-compatibility/2006) xmlns:d="[http://schemas.microsoft.com/expression/blend/2008"](http://schemas.microsoft.com/expression/blend/2008) xmlns:local="clr-namespace:Pract"

mc:Ignorable="d" d:DesignWidth="271.999" d:DesignHeight="68.534"> <ListViewItem Margin="0,0,0.2,-0.2">

<Border BorderBrush="Gray" BorderThickness="1 1 1 1" Width="255"> <StackPanel Width="194">

<TextBlock x:Name="tb\_city" TextWrapping="Wrap" Text="Город: "

Margin="-29,0,-27.4,0" HorizontalAlignment="Right" Width="250" Foreground="Black"/>

<TextBlock x:Name="tb\_adress" Text="Адрес: " Margin="-29,0,-27.4,0"

/>

<TextBlock x:Name="tb\_place" TextWrapping="Wrap"

Text="Местоположение: " Height="17" Margin="-29,0,-27.4,0" Foreground="Black"/>

</StackPanel>

</Border>

</ListViewItem>

</UserControl>

Д

Файл MainWindow.xaml

<Window x:Class="Pract.MainWindow" xmlns="[http://schemas.microsoft.com/winfx/2006/xaml/presentation"](http://schemas.microsoft.com/winfx/2006/xaml/presentation) xmlns:x="[http://schemas.microsoft.com/winfx/2006/xaml"](http://schemas.microsoft.com/winfx/2006/xaml) xmlns:d="[http://schemas.microsoft.com/expression/blend/2008"](http://schemas.microsoft.com/expression/blend/2008) xmlns:mc="[http://schemas.openxmlformats.org/markup-compatibility/2006"](http://schemas.openxmlformats.org/markup-compatibility/2006) xmlns:local="clr-namespace:Pract" mc:Ignorable="d" xmlns:gmap="clr-

namespace:GMap.NET.WindowsPresentation;assembly=GMap.NET.WindowsPresentation"

Height="600"

Width="900"

WindowStyle="None"

ResizeMode="NoResize"

WindowStartupLocation="CenterScreen"

Background="LightBlue"

MouseDown="Window\_MouseDown"> <Grid>

<Grid.RowDefinitions>

<RowDefinition Height="45"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="1084.4"/>

</Grid.ColumnDefinitions>

<StackPanel Margin="578,38.2,186.8,6" Grid.Row="1">

<ListView

Foreground="Black"

ScrollViewer.VerticalScrollBarVisibility="Visible"

ScrollViewer.HorizontalScrollBarVisibility="Hidden"

ScrollViewer.CanContentScroll="True"

MouseLeftButtonUp="listView\_items\_MouseLeftButtonUp" SelectionMode="Single" x:Name="listView\_items"

Height="504"

Margin="5" RenderTransformOrigin="2.023,0.489"/>

</StackPanel>

<Button

Content="X"

Foreground="Black"

Grid.Row="0"

Margin="0,7,194.8,5.8" x:Name="btn\_close"

HorizontalAlignment="Right"

Style="{StaticResource MaterialDesignFlatButton}"

Click="btn\_close\_Click" RenderTransformOrigin="0.431,0.35"/>

<Label Content="Терминалы PrivateBank"

Foreground="Black"

HorizontalAlignment="Center"

VerticalAlignment="Center"

FontSize="26"

Grid.Row="0" Margin="289,2,506.8,555" Grid.RowSpan="2" Width="289"/>

<Image Source="C:\Users\Xiaomi\source\repos\Pract\Image\2.jpg"

HorizontalAlignment="Left" Height="282" Margin="211,267.2,0,0" Grid.Row="1"

VerticalAlignment="Top" Width="367"/>

<Image Source="C:\Users\Xiaomi\source\repos\Pract\Image\1.jpg"

HorizontalAlignment="Left" Height="276" Margin="28,10.2,0,0" VerticalAlignment="Top" Width="335" Grid.Row="1"/>

<TextBox Name="BoxCity" HorizontalAlignment="Left" Background="White"

Height="23" Margin="597,10.2,0,0" Grid.Row="1" TextWrapping="Wrap" VerticalAlignment="Top" Width="143" />

<Button x:Name="Search" Content="Поиск" HorizontalAlignment="Left"

Margin="745,9.2,0,0" Grid.Row="1" VerticalAlignment="Top" Width="75" Height="24"

RenderTransformOrigin="0.509,0.292" Click="Search\_Click"/> </Grid>

</Window>