Sunil Bishwakarma

INFO 2200

Assignment 7

4/24/2021

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

<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"

xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"

x:Class="SBEarthquakeSearchApp.MainPage">

<StackLayout>

<Label Text="Sunil's Earthquake Search"

HorizontalOptions="Center"

HorizontalTextAlignment="Center"

FontSize="25"

VerticalOptions="CenterAndExpand"

Margin="10"

BackgroundColor="BurlyWood"

/>

<Label Text="Earthquake Between"

HorizontalOptions="Center"

HorizontalTextAlignment="Center"

FontSize="20"

VerticalOptions="CenterAndExpand"

Margin="10"/>

<Label Text="Start Date"

HorizontalOptions="Center"

HorizontalTextAlignment="Center"

FontSize="20"

VerticalOptions="CenterAndExpand"

Margin="10"/>

<Entry x:Name="EntryStartDate"

Placeholder="Enter Start Date"

VerticalOptions="CenterAndExpand"

HorizontalTextAlignment="Center"

BackgroundColor="Bisque"

TextColor="Black"/>

<Label Text="End Date"

HorizontalOptions="Center"

HorizontalTextAlignment="Center"

FontSize="20"

VerticalOptions="CenterAndExpand"

/>

<Entry x:Name="EntryEndDate"

Placeholder="Enter End Date"

VerticalOptions="CenterAndExpand"

HorizontalTextAlignment="Center"

BackgroundColor="Bisque"

TextColor="Black"

Margin="10"/>

<Label Text="Earthquake Size (1-10)"

HorizontalOptions="Center"

HorizontalTextAlignment="Center"

FontSize="20"

VerticalOptions="CenterAndExpand"

Margin="10"/>

<Entry x:Name="EntryEarthquakeSize"

Placeholder="Earthquake Size"

VerticalOptions="CenterAndExpand"

HorizontalTextAlignment="Center"

Margin="10"

BackgroundColor="Bisque"

TextColor="Black"/>

<Button x:Name="BtnFind"

Text="Find"

TextColor="DarkRed"

Padding="30"

VerticalOptions="CenterAndExpand"

Clicked="BtnFind\_Clicked"

Margin="10"/>

</StackLayout>

</ContentPage>

using Newtonsoft.Json.Linq;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Linq;

using System.Net;

using System.Text;

using System.Threading.Tasks;

using Xamarin.Forms;

namespace SBEarthquakeSearchApp

{

public partial class MainPage : ContentPage

{

public MainPage()

{

InitializeComponent();

}

/// <summary>

/// Event Handler for the Find button in this backend code

/// </summary>

/// <param name="sender"></param>

/// <param name="e"></param>

private void BtnFind\_Clicked(object sender, EventArgs e)

{

//Make an API CALL

using (WebClient webClient = new WebClient())

{

//Set the header

webClient.Headers[HttpRequestHeader.ContentType] = "application/x-www-form-urlencoded";

//Create try-catch block. If the code breaks an exception will be handled.

//Get the request for earchquake search.

//String variable to get the json code from the server. DownloadString() is a get request from the header.

//Using geojson format to get into metadata json object

string jsonText =

webClient.DownloadString($"https://earthquake.usgs.gov/fdsnws/event/1/query?format=geojson&starttime={EntryStartDate.Text}&endtime={EntryEndDate.Text}&minmagnitude={EntryEarthquakeSize.Text}&");

//Installed Newtonsoft.Json pacakge. Parse the json text and get the objects

JObject jo = JObject.Parse(jsonText);

//Create jason pbject for metadata object

JObject metadata = JObject.Parse(jo["metadata"].ToString());

//As features json object contains a list of array, i am using JArray class to parse the list of array.

JArray features = JArray.Parse(jo["features"].ToString());

// Creates a list of earthquakes and gets values from the API

List<EarthquakeGV> eqList = new List<EarthquakeGV>();

//Using try-catch block to loop through the features and grab the values.

try

{

int idx = 1;

foreach (var eq in features)

{

//create an instance of JObject for earthquake jason object. Passing in properties from the features jason object.

JObject eqJObj = JObject.Parse(eq["properties"].ToString());

//Created an instance global variable class

EarthquakeGV earthquake = new EarthquakeGV

{

//assigning global variable properties. Here ID is required for identifying each json objects, but not used in this application

EQid = idx,

//Parse magnitude jason object to string

EQMag = double.Parse(eqJObj["mag"].ToString()),

//parse location jason object to string

EQLocation = eqJObj["place"].ToString()

};

//List also counts of eatchquakes added to the eatchquake variable

eqList.Add(earthquake);

}

//Code to display the labels navigating from the main page asynchronously.

Navigation.PushAsync(new EarthquakePage(eqList));

}

catch (Exception ex)

{

//Alert the user and provide an opton to close the alert message.

DisplayAlert("Request Error", ex.Message, "Close");

}

}

}

}

}

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

<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"

xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"

x:Class="SBEarthquakeSearchApp.EarthquakePage">

<ContentPage.Content>

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="\*"/>

<RowDefinition Height="\*"/>

<RowDefinition Height="\*"/>

<RowDefinition Height="\*"/>

<RowDefinition Height="\*"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="\*"/>

</Grid.ColumnDefinitions>

<Label Text="Results..."

FontSize="Large"

x:Name="lblResults"

Grid.RowSpan="2"

Grid.ColumnSpan="2"

HorizontalTextAlignment="Center"

VerticalTextAlignment="Center"/>

</Grid>

</ContentPage.Content>

</ContentPage>

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using Xamarin.Forms;

using Xamarin.Forms.Xaml;

namespace SBEarthquakeSearchApp

{

[XamlCompilation(XamlCompilationOptions.Compile)]

public partial class EarthquakePage : ContentPage

{

/// <summary>

/// Passing list of earthquakes as eqList from main page. So that the result can be displayed in the label.

/// </summary>

/// <param name="eqList"></param>

public EarthquakePage(List<EarthquakeGV> eqList)

{

InitializeComponent();

//Update the labels to display results

Random random = new Random();

int randeq = random.Next(1, eqList.Count + 1);

if(eqList.Count > 0)

{

EarthquakeGV displayEQ = eqList[randeq];

lblResults.Text = $"There were {eqList.Count} earthquakes during this time.\n\n" +

$"Details of one of them:\nPlace: {displayEQ.EQMag}, \nMagnitude: {displayEQ.EQMag}.";

}

else

{

lblResults.Text = "No earthquakes";

}

}

}

}

using System;

using System.Collections.Generic;

using System.Text;

namespace SBEarthquakeSearchApp

{

/// <summary>

/// This is a custom class prepared to hold teh properties for the global variables

/// Making class public and static, this way the values will stay in the memory as it moves on for the next one.

/// </summary>

public class EarthquakeGV

{

//Global variable for ID event

public int EQid;

/// <summary>

/// Global variable for location: Place

/// </summary>

public string EQLocation { get; set; }

/// <summary>

/// Global variable for magnitude

/// </summary>

public double EQMag { get; set; }

/// <summary>

/// Global variable for how many times the earthquake occured

/// </summary>

public string EQCount { get; set; }

}

}

using System;

using Xamarin.Forms;

using Xamarin.Forms.Xaml;

namespace SBEarthquakeSearchApp

{

public partial class App : Application

{

public App()

{

InitializeComponent();

//Pass in main page as the navigation page

MainPage = new NavigationPage(new MainPage());

}

protected override void OnStart()

{

}

protected override void OnSleep()

{

}

protected override void OnResume()

{

}

}

}

Graphical user interface

Description automatically generated with medium confidence

Graphical user interface

Description automatically generated with low confidenceText, letter

Description automatically generated