Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Рязанский государственный радиотехнический университет

имени В.Ф.Уткина»

Кафедра ЭВМ

Отчет по лабораторной работе №13

на тему

«Xamarin.Essentials»

по дисциплине  
**«Визуальное программирование»**

Выполнили:

Студенты группы №140

Бригада 5

Сафаров Д. А.

Тимохин Е. С.

Проверили:  
ст. преп. Хизриева Н.И.

ст. преп. Бастрычкин А.С.

**Цель работы**

Получение навыков работы с библиотекой Xamarin.Essentials.

**Задание**

Разработать приложение, которое будет совершать следующие действия:

1) Вывести на экран информацию о батарее и обновлять её.

2) Вывести на экран информацию об устройстве.

3) Вывести на экран информацию о дисплее.

4) Включить/выключить фонарик.

5) Мигать фонариком с интервалом в 1 секунду.

6) Сделать скриншот.

7) Получить случайный контакт из телефонной книги.

8) Передать полученный случайный контакт в приложение для набора номера.

**Практическая часть**

Код программы представлен в приложении А. Результат её работы представлен на рисунке 1.



Рисунок 1 – Результаты работы программы

Приложение А. Текст программы

**MainPage.xaml**

<?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="App13.MainPage">

<StackLayout>

<Label x:Name="BatteryLabel" Text="Battery Info" />

<Label x:Name="DeviceInfoLabel" Text="Device Info" />

<Label x:Name="DisplayInfoLabel" Text="Display Info" />

<Button Text="Toggle Flashlight" Clicked="OnToggleFlashlightClicked" />

<Button x:Name="ToggleBlinkButton" Text="Start Flashlight Blink" Clicked="OnToggleBlinkingClicked" />

<Button Text="Make Screenshot" Clicked="OnMakeScreenshotClicked" />

<Button Text="Get Random Contact and Dial" Clicked="OnGetRandomContactClicked" />

</StackLayout>

</ContentPage>

**MainPage.xaml.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading;

using System.Threading.Tasks;

using Xamarin.Essentials;

using Xamarin.Forms;

using Xamarin.Forms.PlatformConfiguration;

namespace App13

{

public partial class MainPage : ContentPage

{

public MainPage()

{

InitializeComponent();

UpdateBatteryInfo();

Battery.BatteryInfoChanged += OnBatteryInfoChanged;

DisplayDeviceInfo();

DisplayDisplayInfo();

}

/// Информация о батарее

private void UpdateBatteryInfo()

{

BatteryLabel.Text =

"\tBATTERY\n" +

$"Level: {Battery.ChargeLevel \* 100}%\n" +

$"State: {Battery.State}\n" +

$"Source: {Battery.PowerSource}";

}

private void OnBatteryInfoChanged(object sender, BatteryInfoChangedEventArgs e)

{

UpdateBatteryInfo();

}

/// Информация об устройстве

private void DisplayDeviceInfo()

{

DeviceInfoLabel.Text =

"\tDEVICE\n" +

$"Manufacturer: {DeviceInfo.Manufacturer}\n" +

$"Model: {DeviceInfo.Model}\n" +

$"Version: {DeviceInfo.VersionString}\n" +

$"Platform: {DeviceInfo.Platform}\n" +

$"Idiom: {DeviceInfo.Idiom}";

}

/// Информация о дисплее

private void DisplayDisplayInfo()

{

var mainDisplayInfo = DeviceDisplay.MainDisplayInfo;

DisplayInfoLabel.Text =

"\tDISPLAY\n" +

$"Density: {mainDisplayInfo.Density}\n" +

$"Orientation: {mainDisplayInfo.Orientation}\n" +

$"Rotation: {mainDisplayInfo.Rotation}\n" +

$"Width: {mainDisplayInfo.Width}\n" +

$"Height: {mainDisplayInfo.Height}";

}

/// Кнопка фонарика

private bool isFlashlightOn = false;

private async void OnToggleFlashlightClicked(object sender, EventArgs e)

{

try

{

if (isFlashlightOn)

await Flashlight.TurnOffAsync();

else

await Flashlight.TurnOnAsync();

isFlashlightOn = !isFlashlightOn;

}

catch (Exception ex)

{

throw ex;

}

}

/// Мигать фонариком

private bool isFlashlightBlinking = false;

private CancellationTokenSource cts;

private async void OnToggleBlinkingClicked(object sender, EventArgs e)

{

if (isFlashlightBlinking)

{

cts?.Cancel();

ToggleBlinkButton.Text = "Start Flashlight Blink";

}

else

{

cts = new CancellationTokenSource();

ToggleBlinkButton.Text = "Stop Flashlight Blink";

BlinkFlashlight(cts.Token);

}

isFlashlightBlinking = !isFlashlightBlinking;

}

private async Task BlinkFlashlight(CancellationToken token)

{

try

{

while (!token.IsCancellationRequested)

{

await Flashlight.TurnOnAsync();

await Task.Delay(1000);

await Flashlight.TurnOffAsync();

await Task.Delay(1000);

}

}

catch (Exception ex)

{

throw ex;

}

}

/// Скриншот

public interface IScreenshotService

{

Task<Uri> CaptureAsync();

}

private async void OnMakeScreenshotClicked(object sender, EventArgs e)

{

try

{

var screenshotService = DependencyService.Get<IScreenshotService>();

if (screenshotService == null)

{

Console.WriteLine("Сервис захвата скриншота не доступен");

return;

}

var screenshotPath = await screenshotService.CaptureAsync();

var screenshotPage = new ScreenshotPage(screenshotPath.AbsolutePath);

// Обертываем новую страницу в NavigationPage, чтобы можно было управлять навигацией

var navigationPage = new NavigationPage(screenshotPage);

// Переходим на страницу с скриншотом с использованием NavigationPage

await Navigation.PushModalAsync(navigationPage);

}

catch (Exception ex)

{

Console.WriteLine("Ошибка при создании скриншота: " + ex.Message);

}

}

/// Случайный контакт

private async void OnGetRandomContactClicked(object sender, EventArgs e)

{

var phoneNumber = await DependencyService.Get<IContactService>().GetRandomContactAsync();

if (!string.IsNullOrEmpty(phoneNumber))

{

PhoneDialer.Open(phoneNumber);

}

else

{

await DisplayAlert("Error", "No contacts found or no phone number available.", "OK");

}

}

public interface IContactService

{

Task<string> GetRandomContactAsync();

}

}

}