Составил: Садовский Р.В.

Практическая работа №23
Передача и обработка данных

1 Цель работы

1.1 Научиться выполнять обработку данных в приложениях Avalonia UI.

1.2 Научиться передавать данные между частями приложения при помощи внедрения зависимостей в приложениях Avalonia UI.

2 Литература

2.1 Avalonia documentation – Текст : электронный // AvaloniaUI, 2024. – URL: https://docs.avaloniaui.net/

3 Подготовка к работе

3.1 Повторить теоретический материал (см.п.2).

3.2 Изучить описание лабораторной работы.

4 Основное оборудование

4.1 Персональный компьютер.

5 Задание

5.1 Конфигурация приложения

5.1.1 Установите пакеты:
Microsoft.Extensions.Configuration,
Microsoft.Extensions.Configuration.Json,

5.1.2 В файле App.axaml.cs добавьте свойство

```
public static IConfiguration Configuration { get; private set; }
```

5.1.3 Добавьте в проект файл appsettings.json

```
{
  "ApiKeys": {
    "SomeApi": "YourApiKey"
  }
}
```

5.1.4 Создайте в файле App.axaml.cs ConfigurationBuilder для настроек вашего приложения

```
ConfigurationBuilder builder = new ConfigurationBuilder();
//добавляем файл с настройками
builder.AddJsonFile("appsettings.json");
Configuration = builder.Build();
```

5.1.5 При помощи обращения к конфигурации приложения отобразите данные из конфигурации на странице приложения

```
var key = App.Configuration.GetSection("ApiKeys")["SomeApi"];
```

5.2 Внедрение зависимостей

5.2.1 Установите пакет Microsoft.Extensions.DependencyInjection

5.2.2 В файле App.axaml.cs добавьте свойство

```
public static ServiceProvider Services { get; private set; }
```

5.2.1 Создайте в файле App.axaml.cs ServiceCollection для вашего приложения

```
var collection = new ServiceCollection();

//Добавить сервисы

Services = collection.BuildServiceProvider();
```

5.3 Модифицировать сервис навигации из ПР№3 с использованием внедрения зависимостей

5.3.1 Добавить перегруженный метод, получающий страницу для навигации через внедрение зависимостей с помощью `GetRequiredService()`

```
public void NavigateTo<T>(Action<T>? action = null) where T : ViewModelBase
{
    var viewModel = App.Services.GetRequiredService<T>();
    NavigateTo(viewModel, action);
}
```

5.3.2 Зарегистрируйте сервис навигации и страницы в приложении в коллекции сервисов

```
//Singleton - один на все приложение
collection.AddSingleton<NavigationService>();

//Transient - создается новый при каждом обращении
collection.AddTransient<RegistrationViewModel>();
collection.AddTransient<AuthorizationViewModel>();
```

5.4 Применение внедрения зависимостей

5.4.1 В конструкторы `RegistrationViewModel` и `AuthorizationViewModel` добавьте параметр `NavigationService navigation`, значение параметра присвойте приватному полю класса. Аналогичные действия выполните в MainWindow.

5.4.2 На страницах расположите кнопки для перехода от одной страницы к другой, к ним необходимо привязать команды, которые будут вызывать метод навигации. В параметры передайте значение свойства для настройки свойств целевой страницы.

```
_navigation.NavigateTo<AuthorizationViewModel>(
    x=>x.Login = this.Login);
```

5.5 Конверторы значений

5.5.1 Для кастомизации отображения и преобразования привязанных данных используются конверторы значений. Создайте новый класс CelsiusToFarenhateConverter, наследующий IValueConverter

```
public class CelsiusToFarenhateConverter : IValueConverter
```

```
{
    public object? Convert(object? value, Type targetType, object?
parameter, CultureInfo culture)
    {
        if (value == null)
            return null;
        return value; //преобразуйте value из градусов C в градусы
F
    }

    public object ConvertBack(object value, Type targetType,
object parameter, CultureInfo culture)
    {
        return value; //преобразуйте value из градусов F в градусы
C
    }
}
```

5.5.2 Для применения конвертера добавьте на страницу

```
<UserControl.Resources>
    <local:CelsiusToFarenhateConverter
x:Key="CelsiusToFarenhate"/>
</UserControl.Resources>
```

Применение в привязке:

```
{Binding Celsius, Converter={StaticResource CelsiusToFarenhate}}
```

5.5.3 Проверьте работу конвертора

6 Порядок выполнения работы

6.1 Выполнить все задания из п.5.

6.2 Ответить на контрольные вопросы.

7 Содержание отчета

7.1 Титульный лист

7.2 Цель работы

7.3 Ответы на контрольные вопросы

7.4 Вывод

8 Контрольные вопросы

8.1 Для чего необходимо внедрение зависимостей

8.2 Как использовать конфигурацию IConfiguration

8.3 Как получить сервис из коллекции сервисов.

8.4 Для чего применяются конверторы значений