Лабораторная работа №6. Проектирование приложений WPF

Введение

В университете регулярно проводятся открытые лекции по различным тематикам. Для удобства организации данных мероприятий администрация университета решила использовать информационные терминалы. С помощью такого терминала предполагается отображать расписание ближайших лекций, а также производить регистрацию участников.

Для записи на мероприятие участнику необходимо выбрать его из списка и ввести свои персональные данные.

Часть I. Подготовка к работе

Задание 1: В среде Microsoft Workbench создайте базу данных с помощью словаря данных и ER-диаграммы (**RegAppDataDictionary.pdf** и **RegAppERD.pdf**), наполните таблицу Мероприятия несколькими записями.

Часть II. Создание интерфейса пользователя

Задание 2: Спроектируйте графический интерфейс пользователя.

1. Создайте новый проект, используя шаблон **Приложение WPF**.
2. На панели **Свойства** измените заголовок окна (**Title – RegApp**);
3. Выполните разметку страницы:
   1) перейдите на вкладку **XAML**;
   2) выделите элемент **Grid** и на вкладке свойства измените цвет фона на темный;
   3) для элемента **Grid** добавьте строки и задайте их размер по высоте:

```
<Grid>
   <Grid.RowDefinitions>
         <RowDefinition Height="*"></RowDefinition>
         <RowDefinition Height="3*"></RowDefinition>
   </Grid.RowDefinitions>
</Grid>
```

   4) в верхнюю строку с **Панели элементов** перенесите компонент типа **TextBlock**;
   5) задайте следующие значения свойств текстового блока, используя вкладку свойства или редактор кода:

```
<TextBlock x:Name="textBlock" Margin="0" TextWrapping="Wrap"
Text="Регистрация на открытые лекции университета" Foreground="White"
FontSize="20" TextAlignment="Center" VerticalAlignment="Center"/>
```

   6) во вторую строку компонента **Grid** поместите таблицу **DataGrid**;
   7) установите для таблицы параметры фона и шрифта, размер по размеру ячейки **Grid**;
   8) создайте столбцы таблицы:

```
<DataGrid x:Name="dataGrid" Margin="0" Grid.Row="1" HeadersVisibility="None">
   <DataGrid.Columns>
         <DataGridTextColumn Width="*"/>
         <DataGridTextColumn Width="2*"/>
   <DataGridTextColumn Width="*"/>
</DataGrid.Columns>
```

4. Запустите программу и проверьте правильность отображения элементов интерфейса. Попробуйте изменить размер окна.

Часть III. Реализация шаблона Model-View-View-Model (MVVM)

Задание 3: Создайте модель данных приложения.

1. В **Обозревателе решений** создайте папку **Model** и класс **Event.cs** в ней.
2. Опишите открытый класс **Event** для представления информации о мероприятиях:

```
//поля класса
public int eventId;
private string name;
private DateTime date;
private string location;

//свойства класса для чтения полей
public int EventId
{
   get { return this.eventId; }
}

public string Name
{
   get { return this.name; }
}

public string Date
{
   get { return this.date.ToString("D"); }
}

public string Location
{
   get { return this.location; }
}

//конструктор класса
public Event(int eventId, string name, DateTime date, string location)
{
   this.eventId = eventId;
   this.name = name;
   this.date = date;
   this.location = location;
}
```

3. Выполните привязку свойств класса к таблице в главном окне приложения:
   1) перейдите к редактированию файла **MainWindow.xaml**;
   2) у таблицы **DataGrid** отмените автоматическое создание столбцов, установив свойство:

```
AutoGenerateColumns="False"
```

   3) для строк таблицы определите свойство привязки:

```
<DataGridTextColumn Width="*" Binding="{Binding Path=Date}"/>
<DataGridTextColumn Width="2*" Binding="{Binding Name}"/>
<DataGridTextColumn Width="*" Binding="{Binding Location}"/>
```

Задание 4: Создайте класс для обеспечения доступа к базе данных.

1. Добавьте к проекту класс **DataAccess.cs**.
2. В новом классе определите:


```
public static class DataAccess
{
   private static string connectionString = @"Database = eventReg;
                                              Data Source = localhost;
                                              UserID = root; Password = qwerty";
   private static MySqlConnection msConnection;
   private static MySqlCommand msCommand;
   private static MySqlDataAdapter msDataAdapter;
   static public bool IsConnected;      //состояние подключения
   static public string ErrorMessage;   //сообщение об ошибке

   //подключение к БД
   static public void Connect()
   {
         try
         {
               msConnection = new MySqlConnection(connectionString);
               msConnection.Open();
               msCommand = new MySqlCommand();
               msCommand.Connection = msConnection;
               msDataAdapter = new MySqlDataAdapter(msCommand);
               IsConnected = true;
         }
         catch (Exception ex)
         {
               IsConnected = false;
               ErrorMessage = ex.Message;
         }
   }
   //получение списка мероприятий из базы
   static public List<Event> GetEventList()
   {
         List<Model.Event> result = new List<Model.Event>(); // список значений

         DateTime today = DateTime.Today;
         //запрос на выборку данных из базы
         msCommand.CommandText = @"SELECT * FROM Events WHERE Date>='"
                                 + today.ToString("yyyy-MM-dd")
                                 + "' ORDER BY Date";

         //выплонение получение результата запроса
         var reader = msCommand.ExecuteReader();

         //добавление записей из результата запроса в список
         while (reader.Read())
         {
               int id = Convert.ToInt32(reader[0]);
               string name = reader[1].ToString();
               DateTime date = Convert.ToDateTime(reader[2]);
               string location = reader[3].ToString();
               result.Add(new Event(id, name, date, location));
         }
         reader.Close(); //освобождение переменной
         return result;
   }
}
```

3. В файле **MainWindow.xaml.cs** в конструкторе окна выполните подключение к базе данных:

```
DataAccess.Connect();        //установка соединения
if (!DataAccess.IsConnected) //если соединение не удалось
{
   MessageBox.Show(DataAccess.ErrorMessage); //вывод сообщения об ошибке
   this.Close();                             //закрытие приложения
```



```
}
```

4. Запустите программу и проверьте подключение.

Задание 5: Создайте представление для модели.

1. Добавьте в проект класс **ViewModel.cs**.
2. Опишите структуру класса:

```
static public class ModelView
{
   //Список мероприятий
   static public List<Model.Event> EventList = new List<Model.Event>();
   //номер выбранного мероприятия из списка
   static public int currentEventNumber;

   //загрузка списка из БД
   static public void LoadEvents()
   {
      EventList = DataAccess.GetEventList();
      currentEventNumber = 0;
   }

   //получение объекта выбранного мероприятия
   static public Model.Event CurrentEvent()
   {
         //если искомый индекс в границах диапазона списка
         if (currentEventNumber >= 0 && currentEventNumber < EventList.Count)

            //возвращаем объект выбранного мероприятия
            return EventList[currentEventNumber];
         else   //в противном случае возвращаем нулевое значение
             return null;
   }
}
```

3. В конструкторе главного окна выполните привязку данных к таблице:

```
ModelView.LoadEvents();                     //загрузка списка из БД
dataGrid.ItemsSource = ModelView.EventList; //привязка списка к таблице
```

4. Запустите программу и проверьте правильность отображения данных из таблицы Мероприятия.

Часть IV. Страница регистрации

Задание 6: Создайте форму регистрации на мероприятие.

1. Добавьте в проект новое окно **RegWindow**.
2. Спроектируйте интерфейс формы:
   1) разместите текстовый блок для отображения названия выбранного мероприятия;
   2) разместите текстовые поля для ввода имени и фамилии участника;
   3) разместите кнопки **Регистрация** и **Отмена**;
   4) выполните настройку визуального оформления.
3. При щелчке мышью по строкам таблицы главного окна выполните переход на форму с регистрацией:

```
//установка текущего выбранного мероприятия
ViewModel.currentEventNumber = dataGrid.SelectedIndex;

//создание и показ экземпляра окна
RegistrationWindow regWind = new RegistrationWindow();
regWind.ShowDialog();
```

4. В конструкторе формы регистрации отобразите название выбранного мероприятия:

```
textBlock1.Text = ViewModel.CurrentEvent().Name;
```

5. Запустите программу и проверьте правильность ее работы.

Задания для самостоятельной работы

1. Закончите реализацию регистрации участника, добавив механизм записи данных об участнике и регистрации в базу.
2. Дополните приложение возможностью просмотра списка зарегистрированных участников на выбранное мероприятие.

