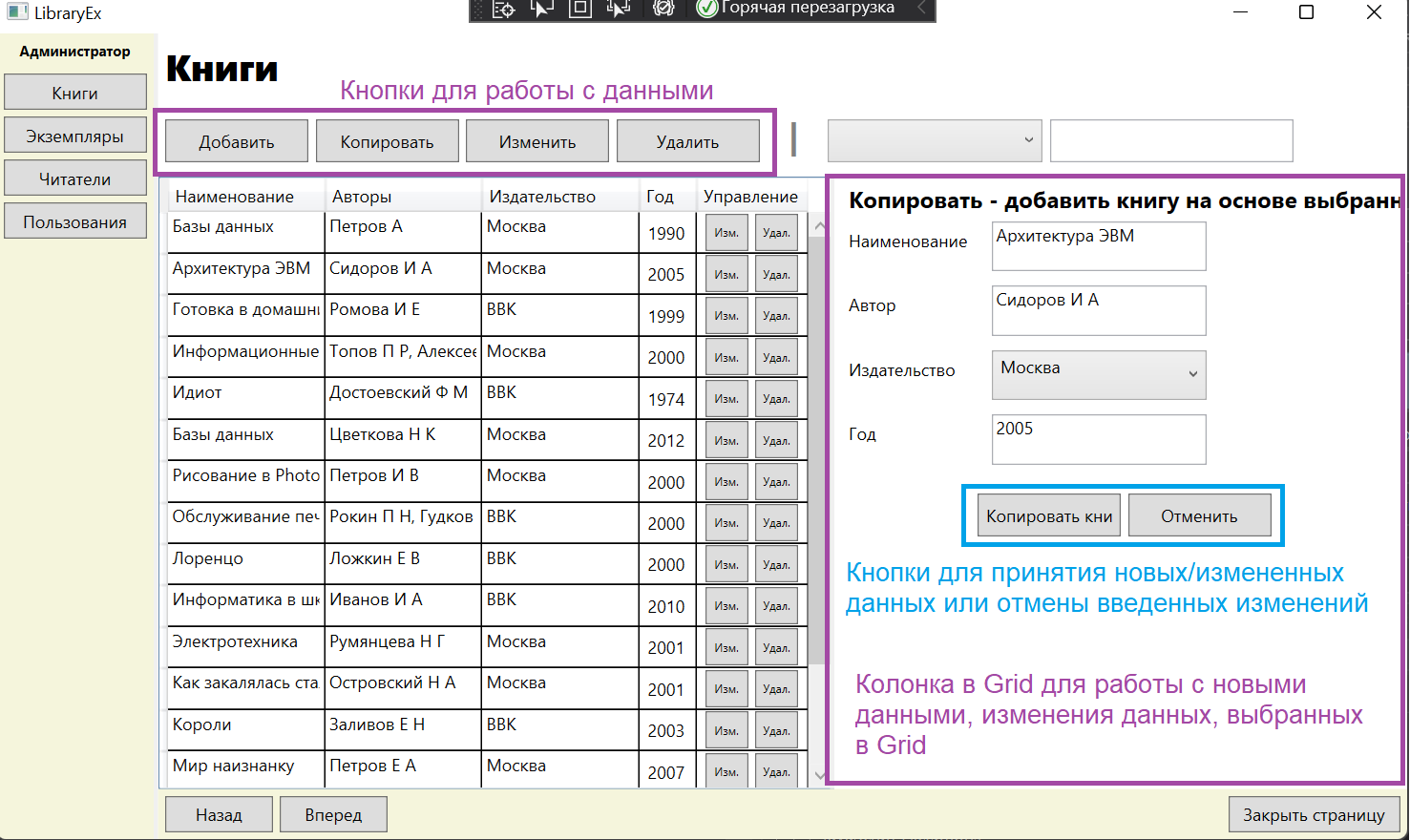
**Работа с данными**

1. Подготовить интерфейс. Для решения задач, связанных с работой данных: добавить (добавить на основе выбранного), используется дополнительная колонка, которая становится доступна к просмотру при выполнении операций с данными.



Для приведенного выше интерфейса файл-XAML будет описан так:

<Page x:Class="TestApp.Pages.BookPage"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:TestApp.Pages"

mc:Ignorable="d"

d:DesignHeight="450" d:DesignWidth="800"

Title="BookPage">

<Grid x:Name="BookGrid">

**Задание сетки для обеспечения новых задач**

<Grid.RowDefinitions>

<RowDefinition Height="50"/>

<RowDefinition Height="50"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="3"/>

<ColumnDefinition x:Name="BookColumnChange" Width="400" MaxWidth="400"/>

</Grid.ColumnDefinitions>

**Заголовок новой страницы**

<Label Content="Книги" Grid.Row="0" Grid.Column="0" Grid.ColumnSpan="2" FontSize="26" FontWeight="ExtraBold"/>

**Описание кнопок для работы с данными**

<StackPanel Grid.Row="1" Grid.Column="0" Grid.ColumnSpan="3" Orientation="Horizontal">

<Button x:Name="BookAdd" Content="Добавить"

Height="30" Width="100" Padding="5,5,5,5" Margin="5,5,0,5"

Click="BookAdd\_Click"/>

<Button x:Name="BookCopy" Content="Копировать"

Height="30" Width="100" Padding="5,5,5,5" Margin="5,5,0,5"

Click="BookCopy\_Click"/>

<Button x:Name="BookEdit" Content="Изменить"

Height="30" Width="100" Padding="5,5,5,5" Margin="5,5,0,5"

Click="BookEdit\_Click"/>

<Button x:Name="BookDellete" Content="Удалить"

Height="30" Width="100" Padding="5,5,5,5" Margin="5,5,0,5"

Click="BookDelete\_Click"/>

<Label Content="|" FontSize="24" FontWeight="Bold" Foreground="Gray" Margin="15,0,0,0"/>

<ComboBox x:Name="BookFilterComboBox" Width="150" Height="30" Margin="15,0,0,0"/>

<TextBox x:Name="BookFilterTextBox" Margin="5,0,5,0" Width="170" Height="30"/>

</StackPanel>

**DataGrid для отображения данных**

<DockPanel Grid.Row="2" Grid.Column="0" Grid.ColumnSpan="2">

<DataGrid x:Name="BookDataGrid" AutoGenerateColumns="False" CanUserAddRows="False" Background="White">

<DataGrid.Columns>

<!--1-ый способ-->

<DataGridTextColumn Header="Наименование" Binding="{Binding Path=Name}" Width="10\*"/>

<DataGridTextColumn Header="Авторы" Binding="{Binding Path=Authors}" Width="10\*"/>

<DataGridTextColumn Header="Издательство" Binding="{Binding Path=Publishers.PublisherName}" Width="10\*"/>

<!--2-ой способ-->

<DataGridTemplateColumn Width="40">

<DataGridTemplateColumn.Header>

Год

</DataGridTemplateColumn.Header>

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<Label Content="{Binding PublishYear}"/>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

<!--кнопки управления данными выделенной строки-->

<DataGridTemplateColumn Header="Управление">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<StackPanel Orientation="Horizontal">

<Button Content="Изм." Width="30" Margin="5,0,0,0"

FontSize="8" Click="BookEdit\_Click"/>

<Button Content="Удал." Width="30" Margin="5,0,5,0"

FontSize="8" Click="BookDelete\_Click"/>

</StackPanel>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

</DataGrid.Columns>

</DataGrid>

</DockPanel>

**Сплиттер для изменения размеров колонок для работы с новыми данными**

<GridSplitter Grid.Column="1" Grid.Row="2" Width="3" HorizontalAlignment="Stretch" />

**Описание колонки для работы с новыми/измененными данными**

**Старая версия**

<StackPanel Grid.Row="2" Grid.Column="2" Orientation="Vertical">

<Label x:Name="BookLabel" Content="Добавить книгу" FontSize="16" FontWeight="Bold" Margin="5,0,0,0"/>

<StackPanel Orientation="Horizontal">

<Label Content="Наименование" Margin="5,0,0,0" Width="100"/>

<TextBox x:Name="BookTextName"

Text="{Binding ElementName=BookDataGrid, Path=SelectedItem.Name}" Width="150" Height="35" Margin="5,0,0,0"/>

</StackPanel>

<StackPanel Orientation="Horizontal" Margin="0,10,0,0">

<Label Content="Автор" Margin="5,0,0,0" Width="100"/>

<TextBox x:Name="BookTextAuthors"

Text="{Binding ElementName=BookDataGrid, Path=SelectedItem.Authors}" Width="150" Height="35" Margin="5,0,0,0"/>

</StackPanel>

<StackPanel Orientation="Horizontal" Margin="0,10,0,0">

<Label Content="Издательство" Margin="5,0,0,0" Width="100"/>

<ComboBox x:Name="BookComboBoxPublishers" Height="35" Width="150" Margin="5,0,0,0" DisplayMemberPath="PublisherName"

SelectedItem="{Binding ElementName=BookDataGrid, Path=SelectedItem.Publishers}"/>

</StackPanel>

<StackPanel Orientation="Horizontal" Margin="0,10,0,0">

<Label Content="Год" Margin="5,0,0,0" Width="100"/>

<TextBox x:Name="BookTextPublishYear"

Text="{Binding ElementName=BookDataGrid, Path=SelectedItem.PublishYear}"

Width="150" Height="35" Margin="5,0,0,0"/>

</StackPanel>

|  |
| --- |
| Новая версия (убрана связь по Binding’у)  <StackPanel Orientation="Horizontal">  <Label Content="Наименование" Margin="5,0,0,0" Width="100"/>  <TextBox x:Name="BookTextName" Width="150" Height="35" Margin="5,0,0,0"/>  </StackPanel>  <StackPanel Orientation="Horizontal" Margin="0,10,0,0">  <Label Content="Автор" Margin="5,0,0,0" Width="100"/>  <TextBox x:Name="BookTextAuthors" Width="150" Height="35" Margin="5,0,0,0"/>  </StackPanel>  <StackPanel Orientation="Horizontal" Margin="0,10,0,0">  <Label Content="Издательство" Margin="5,0,0,0" Width="100"/>  <ComboBox x:Name="BookComboBoxPublishers" Height="35" Width="150" Margin="5,0,0,0" DisplayMemberPath="PublisherName"/>  </StackPanel>  <StackPanel Orientation="Horizontal" Margin="0,10,0,0">  <Label Content="Год" Margin="5,0,0,0" Width="100"/>  <TextBox x:Name="BookTextPublishYear" Width="150" Height="35" Margin="5,0,0,0"/>  </StackPanel> |

**Кнопки для принятия или отмены внесенных изменений**

<StackPanel Orientation="Horizontal" HorizontalAlignment="Center" Margin="0,15,0,0">

<Button x:Name="BookAddCommit" Content="Добавить книгу"

Height="30" Width="100" Padding="5,5,5,5" Margin="5,5,0,5" Click="BookAddCommit\_Click"/>

<Button x:Name="BookAddRollback" Content="Отменить"

Height="30" Width="100" Padding="5,5,5,5" Margin="5,5,0,5" Click="BookAddRollback\_Click"/>

</StackPanel>

2. Описать события, связанные с изменением данных

**Описание вспомогательных переменных и процедур (на верхнем уровне класса)**

CS

//описание вспомогательных переменных (на верхнем уровне класса, перед инициализацией)

|  |  |
| --- | --- |
| **Старая версия**  private int DlgMode;  private string Name\_buf;  private string Authors\_buf;  private string Publishers\_buf;  private string PublishYear\_buf; | **Новая версия**  private int DlgMode;  public Base.Books SelectedBook;  public ObservableCollection<Base.Books> Books; |

//уровень инициализации

public BookPage()

{

InitializeComponent();

UpdateGrid(null);

BookDlgLoad(false,"");

DataContext = this;

}

//регулирование размера колонки для работы с данными и другие вспомогательные настройки

|  |  |
| --- | --- |
| **Старая версия**  public void BookDlgLoad(bool b){  if (b==true)  {BookColumnChange.Width =  new GridLength(400);  BookDataGrid.IsHitTestVisible = false;  } else  {BookColumnChange.Width =  new GridLength(0);  BookDataGrid.IsHitTestVisible = true;  }  } | **Новая версия**  public void BookDlgLoad(bool b, string DlgModeContent)  {  if (b == true)  {  BookColumnChange.Width = new GridLength(400);  BookDataGrid.IsHitTestVisible = false;  BookLabel.Content = DlgModeContent;  BookAddCommit.Content = DlgModeContent;  BookAdd.IsEnabled = false;  BookCopy.IsEnabled = false;  BookEdit.IsEnabled = false;  BookDellete.IsEnabled = false;  }  else  {  BookColumnChange.Width = new GridLength(0);  BookDataGrid.IsHitTestVisible = true;  BookAdd.IsEnabled = true;  BookCopy.IsEnabled = true;  BookEdit.IsEnabled = true;  BookDellete.IsEnabled = true;  DlgMode = -1;  }  } |

//создание коллекции и списков для визуальных компонентов с отслеживанием позиции курсора

**Обновление данных**

|  |  |
| --- | --- |
| **List<T>** | **ObservableCollection<T>** |
| Класс List<T> из пространства имен System.Collections.Generic представляет простейший список однотипных объектов.  Среди его методов можно выделить следующие:   * **void Add(T item)**: добавление нового элемента в список * **void AddRange(ICollection collection)**: добавление в список коллекции или массива * **int BinarySearch(T item)**: бинарный поиск элемента в списке. Если элемент найден, то метод возвращает индекс этого элемента в коллекции. При этом список должен быть отсортирован. * **int IndexOf(T item)**: возвращает индекс первого вхождения элемента в списке * **void Insert(int index, T item)**: вставляет элемент item в списке на позицию index * **bool Remove(T item)**: удаляет элемент item из списка, и если удаление прошло успешно, то возвращает true * **void RemoveAt(int index)**: удаление элемента по указанному индексу index * **void Sort()**: сортировка списка | ObservableCollection по функциональности похож на список List за тем исключением, что позволяет известить внешние объекты о том, что коллекция была изменена.  Класс ObservableCollection находится в пространстве имен System.Collections.ObjectModel, поэтому в начале подключаем эти пространства имен.  Класс ObservableCollection определяет событие **CollectionChanged**, подписавшись на которое, мы можем обработать любые изменения коллекции.  В обработчике этого события Users\_CollectionChanged для получения всей информации о событии используется объект NotifyCollectionChangedEventArgs e. Его свойство Action позволяет узнать характер изменений. Оно хранит одно из значений из перечисления **NotifyCollectionChangedAction**.  Свойства NewItems и OldItems позволяют получить соответственно добавленные и удаленные объекты. Таким образом, мы получаем полный контроль над обработкой добавления, удаления и замены объектов в коллекции. |

**List<T> vs ObservableCollection<T>**

[**https://utyatnishna.ru/info/98591/observablecollection-vs-list**](https://utyatnishna.ru/info/98591/observablecollection-vs-list)

|  |
| --- |
| **Старая версия**  public void UpdateGrid(Base.Books books)  {  if ((books == null) && (BookDataGrid.ItemsSource != null))  {  books = (Base.Books)BookDataGrid.SelectedItem;  }  // 1-ый способ, используя коллекцию – РЕКОМЕНДУЕТСЯ  ObservableCollection<Base.Books> Books =  new ObservableCollection<Base.Books>(SourceCore.MyBase.Books.ToList());  BookDataGrid.ItemsSource = Books;  BookDataGrid.SelectedItem = books;  BookComboBoxPublishers.ItemsSource = SourceCore.MyBase.Publishers.ToList();  } |
| **Новая версия (фактически ничего не поменялось. Другая запись при объявлении Books на уровне класса)**  public void UpdateGrid(Base.Books Book)  {  if ((Book == null) && (BookDataGrid.ItemsSource != null))  {  Book = (Base.Books)BookDataGrid.SelectedItem;  }  Books = new ObservableCollection<Base.Books>(SourceCore.MyBase.Books);  BookDataGrid.ItemsSource = Books;  BookDataGrid.SelectedItem = Book;  BookComboBoxPublishers.ItemsSource = SourceCore.MyBase.Publishers.ToList();  } |

**Определение режима работы с данными**



**Добавить книгу – кнопка «Добавить»**

XAML

<Button x:Name="BookAdd" Content="Добавить" Height="30" Width="100" Padding="5,5,5,5" Margin="5,5,0,5" Click="BookAdd\_Click"/>

CS

|  |
| --- |
| **Старая версия**  private void BookAdd\_Click(object sender, RoutedEventArgs e)  {  //процедура, обеспечивающая отображение и скрытие колонки для работы с данными  BookDlgLoad(true);  DlgMode = 0;  BookDataGrid.SelectedItem = null;  BookLabel.Content = "Добавить книгу";  BookAddCommit.Content = "Добавить книгу";  BookTextName.Text = "";  BookTextAuthors.Text = "";  BookComboBoxPublishers.Text = "";  BookTextPublishYear.Text = "";  } |
| **Новая версия**  private void BookAdd\_Click(object sender, RoutedEventArgs e)  {  BookDlgLoad(true, "Добавить книгу");  DataContext = null;  DlgMode = 0;  } |

**Добавить книгу (на основе выбранной) – кнопка «Копировать»**

XAML

<Button x:Name="BookCopy" Content="Копировать" Height="30" Width="100" Padding="5,5,5,5" Margin="5,5,0,5" Click="BookCopy\_Click"/>

CS

|  |
| --- |
| **Старая версия**  private void BookCopy\_Click(object sender, RoutedEventArgs e)  {  if (BookDataGrid.SelectedItem !=null)  {  BookDlgLoad(true);  DlgMode = 0;  BookLabel.Content = "Копировать - добавить книгу на основе выбранной";  BookAddCommit.Content = "Копировать книгу";  //использование буферных переменных для «отрыва» от данных выбранной строки (чтобы не сработал Binding)  Name\_buf = BookTextName.Text;  Authors\_buf = BookTextAuthors.Text;  Publishers\_buf = BookComboBoxPublishers.Text;  PublishYear\_buf = BookTextPublishYear.Text;  //убрать фокус с выделенной строки  BookDataGrid.SelectedItem = null;  BookTextName.Text = Name\_buf;  BookTextAuthors.Text = Authors\_buf;  BookComboBoxPublishers.Text = Publishers\_buf;  BookTextPublishYear.Text = PublishYear\_buf;  } else  {  MessageBox.Show("Не выбрано ни одной строки!","Сообщение",MessageBoxButton.OK);  }    } |
| **Новая версия**  private void BookCopy\_Click(object sender, RoutedEventArgs e)  {  if (BookDataGrid.SelectedItem != null)  {  BookDlgLoad(true, "Копировать книгу");  SelectedBook = (Base.Books)BookDataGrid.SelectedItem;  BookTextName.Text = SelectedBook.Name;  BookTextAuthors.Text = SelectedBook.Authors;  BookComboBoxPublishers.Text = SelectedBook.Publishers.PublisherName;  BookTextPublishYear.Text = SelectedBook.PublishYear.ToString();  DlgMode = 0;  }  else  {  MessageBox.Show("Не выбрано ниодной строки!", "Сообщение", MessageBoxButton.OK);  }  } |

**Изменить книгу – кнопка «Изменить»**

XAML

<Button x:Name="BookEdit" Content="Изменить" Height="30" Width="100" Padding="5,5,5,5" Margin="5,5,0,5" Click="BookEdit\_Click"/>

CS

|  |
| --- |
| **Старая версия**  private void BookEdit\_Click(object sender, RoutedEventArgs e)  {  if (BookDataGrid.SelectedItem != null)  {  BookDlgLoad(true);  BookLabel.Content = "Изменить книгу";  BookAddCommit.Content = "Изменить книгу";  }  else  {  MessageBox.Show("Не выбрано ни одной строки!", "Сообщение", MessageBoxButton.OK);  }  } |
| **Новая версия**  private void BookEdit\_Click(object sender, RoutedEventArgs e)  {  if (BookDataGrid.SelectedItem != null)  {  BookDlgLoad(true, "Изменить книгу");  SelectedBook = (Base.Books)BookDataGrid.SelectedItem;  BookTextName.Text = SelectedBook.Name;  BookTextAuthors.Text = SelectedBook.Authors;  BookComboBoxPublishers.Text = SelectedBook.Publishers.PublisherName;  BookTextPublishYear.Text = SelectedBook.PublishYear.ToString();  }  else  {  MessageBox.Show("Не выбрано ниодной строки!", "Сообщение", MessageBoxButton.OK);  }  }  } |

**Удалить книгу – кнопка «Удалить»**

XAML

<Button x:Name="BookDellete" Content="Удалить" Height="30" Width="100" Padding="5,5,5,5" Margin="5,5,0,5" Click="BookDelete\_Click"/>

CS

|  |
| --- |
| **Старая версия**  private void BookDelete\_Click(object sender, RoutedEventArgs e)  {  if (MessageBox.Show("Удалить запись?","Внимание",MessageBoxButton.OKCancel,MessageBoxImage.Warning) == MessageBoxResult.OK)  {  SourceCore.MyBase.Books.Remove((Base.Books)BookDataGrid.SelectedItem);  SourceCore.MyBase.SaveChanges();  }  } |
| **Новая версия – без определения новой позиции указателя**  private void BookDelete\_Click(object sender, RoutedEventArgs e)  {  if (MessageBox.Show("Удалить запись?", "Внимание", MessageBoxButton.OKCancel, MessageBoxImage.Warning) == MessageBoxResult.OK)  {  SourceCore.MyBase.Books.Remove((Base.Books)BookDataGrid.SelectedItem);  SourceCore.MyBase.Entry((Base.Books)BookDataGrid.SelectedItem).State = EntityState.Deleted;  SourceCore.MyBase.SaveChanges();  UpdateGrid(null);  }  } |
| **Новая версия – с определения новой позиции указателя**  private void BookDelete\_Click(object sender, RoutedEventArgs e)  {  if (MessageBox.Show("Удалить запись?", "Внимание", MessageBoxButton.OKCancel, MessageBoxImage.Warning) == MessageBoxResult.OK)  {  try  {  // Ссылка на удаляемую книгу  Base.Books DeletingBook = (Base.Books)BookDataGrid.SelectedItem;  // Определение ссылки, на которую должен перейти указатель после удаления  if (BookDataGrid.SelectedIndex < BookDataGrid.Items.Count - 1)  {  BookDataGrid.SelectedIndex++;  }  else  {  if (BookDataGrid.SelectedIndex > 0)  {  BookDataGrid.SelectedIndex--;  }  }  Base.Books SelectingBook = (Base.Books)BookDataGrid.SelectedItem;  SourceCore.MyBase.Books.Remove(DeletingBook);  SourceCore.MyBase.SaveChanges();  UpdateGrid(SelectingBook);  }  catch  {  MessageBox.Show("Невозможно удалить запись, так как она используется в других  справочниках базы данных.","Предупреждение", MessageBoxButton.OK, MessageBoxImage.Warning, MessageBoxResult.None);  }  }  } |

**Подтверждение/отмена введенных изменений**



**Подтверждение внесенных изменений**

|  |
| --- |
| **Старая версия**  private void BookAddCommit\_Click(object sender, RoutedEventArgs e)  {  var NewBook = new Base.Books();  NewBook.Name = BookTextName.Text;  NewBook.Authors = BookTextAuthors.Text;  NewBook.Publishers = (Base.Publishers)BookComboBoxPublishers.SelectedItem;  NewBook.PublishYear = (short?)int.Parse(BookTextPublishYear.Text);  if (DlgMode==0)  {  SourceCore.MyBase.Books.Add(NewBook);  }    SourceCore.MyBase.SaveChanges();  BookDlgLoad(false);  } |
| **Новая версия – без проверок входных данных**  private void BookAddCommit\_Click(object sender, RoutedEventArgs e)  {  if (DlgMode == 0)  {  var NewBook = new Base.Books();  NewBook.Name = BookTextName.Text;  NewBook.Authors = BookTextAuthors.Text;  NewBook.Publishers = (Base.Publishers)BookComboBoxPublishers.SelectedItem;  NewBook.PublishYear = (short?)int.Parse(BookTextPublishYear.Text);  SourceCore.MyBase.Books.Add(NewBook);  SelectedBook = NewBook;  } else  {  var EditBook = new Base.Books();  EditBook = SourceCore.MyBase.Books.First(p => p.IdBook == SelectedBook.IdBook);  EditBook.Name = BookTextName.Text;  EditBook.Authors = BookTextAuthors.Text;  EditBook.Publishers = (Base.Publishers)BookComboBoxPublishers.SelectedItem;  EditBook.PublishYear = (short?)int.Parse(BookTextPublishYear.Text);  }  SourceCore.MyBase.SaveChanges();  BookDlgLoad(false, "");    UpdateGrid(SelectedBook);  } |
| **Новая версия – с проверкой входных данных**  private void BookAddCommit\_Click(object sender, RoutedEventArgs e)  {  StringBuilder errors = new StringBuilder();    if (string.IsNullOrEmpty(BookTextName.Text))  errors.AppendLine("Укажите название книги");  if (string.IsNullOrEmpty(BookTextAuthors.Text))  errors.AppendLine("Укажите название книги");  if (((Base.Publishers)BookComboBoxPublishers.SelectedItem == null) || (BookComboBoxPublishers.Text == " ..."))  errors.AppendLine("Укажите издательство");  if (string.IsNullOrEmpty(BookTextPublishYear.Text))  errors.AppendLine("Укажите название книги");  if (errors.Length > 0)  {  MessageBox.Show(errors.ToString());  return;  }  if (DlgMode == 0)  {  var NewBook = new Base.Books();  NewBook.Name = BookTextName.Text;  NewBook.Authors = BookTextAuthors.Text;  NewBook.Publishers = (Base.Publishers)BookComboBoxPublishers.SelectedItem;  NewBook.PublishYear = (short?)int.Parse(BookTextPublishYear.Text);  SourceCore.MyBase.Books.Add(NewBook);  SelectedBook = NewBook;  } else  {  var EditBook = new Base.Books();  EditBook = SourceCore.MyBase.Books.First(p => p.IdBook == SelectedBook.IdBook);  EditBook.Name = BookTextName.Text;  EditBook.Authors = BookTextAuthors.Text;  EditBook.Publishers = (Base.Publishers)BookComboBoxPublishers.SelectedItem;  EditBook.PublishYear = (short?)int.Parse(BookTextPublishYear.Text);  }  try  {  SourceCore.MyBase.SaveChanges();  BookDlgLoad(false, "");  UpdateGrid(SelectedBook);  }  catch (Exception ex)  {  MessageBox.Show(ex.Message.ToString());  }  } |

**Отмена внесенных изменений**

|  |
| --- |
| **Старая версия**  private void BookAddRollback\_Click(object sender, RoutedEventArgs e)  {  BookDlgLoad(false);  } |
| **Новая версия**  private void BookAddRollback\_Click(object sender, RoutedEventArgs e)  {  BookDlgLoad(false,"");  UpdateGrid(SelectedBook);  } |