**ОСНОВНАЯ ИНФОРМАЦИЯ**

|  |  |
| --- | --- |
| =ВПР(искомое значение; таблица; номер\_столбца; параметр\_просмотра)  Искомое значение — те данные, которые хотим заменить на цифры (id).  Таблица — диапазон ячеек, в котором ищем и из которого затем извлекаем данные (у нас это два столбца: значения из БД и их ID (всегда идет ВТОРЫМ)).  Номер\_столбца — всегда 2  Интервальный\_просмотр — всегда 0 | Алгоритм импорта:   1. Посмотрите на названия файлов — это таблицы, которые нужно заполнить 2. Создайте новый файл Excel, чтобы не испортить данные в исходных файлах 3. Скопируйте заголовки таблиц из БД, чтобы понимать правильный порядок. Столбец ID оставляете пустым 4. Если каких-то данных нет, проверьте, могут ли они быть нулевыми 5. Называйте листы именами таблиц, чтобы не путаться 6. Не забывайте при копировании использовать вставку значений |
| **Installer**  Решение > Добавить > Новый проект  Add > Proj output > Primary output  Add > File > Icon  Создать ярлык для Prim output и в свойствах поменять на Desktop, задать ярлыку icon  Задать проекту свойства  Build > Batch build  Находится в папке Setup > Release | Тест-кейсы |
| Модульные тесты | Команды для GitBash  touch README.md  git init  git remote add origin https://try.gogs.io/Julia\_Zhirnova/Zhirnova\_Praktika  git checkout -b Session1  git add .  git commit -m "first commit"  git push -u origin session1 |
| private static BetarPracticeEntities \_context;  public static BetarPracticeEntities GetContext()  {  if (\_context == null)  {  \_context = new BetarPracticeEntities();  }  return \_context;  }  DGrid.ItemsSource = Entities.GetContext().Products.ToList();  <DataGrid Grid.Row="1" x:Name="DGrid" AutoGenerateColumns="False" IsReadOnly="True" BorderThickness="0" GridLinesVisibility="None" HeadersVisibility="None"> | <Style TargetType="Button" x:Key="btnBorder">  <Setter Property="Control.Template">  <Setter.Value>  <ControlTemplate TargetType="Button">  <Border Background="{TemplateBinding Background}" BorderBrush="{TemplateBinding BorderBrush}" BorderThickness="2">  <ContentPresenter VerticalAlignment="Center" HorizontalAlignment="Center" Margin="15,0,15,0"/>  </Border>  </ControlTemplate>  </Setter.Value>  </Setter>  <Setter Property="Background" Value="Transparent" />  <Setter Property="Foreground" Value="#FF778893"/>  <Setter Property="BorderBrush" Value="#FFCCCCCC"/>  <Setter Property="FontSize" Value="10"/>  <Setter Property="Cursor" Value="Hand"/>  <Style.Triggers>  <Trigger Property="IsMouseOver" Value="True">  <Setter Property="Background" Value="#FF04A0FF"/>  <Setter Property="Foreground" Value="White"/>  <Setter Property="BorderBrush" Value="#FF04A0FF"/>  </Trigger>  </Style.Triggers>  </Style> |
| **Frame**  <Frame Grid.Row="1" Name="MainFrame" NavigationUIVisibility="Hidden"/>  public static Frame MainFrame { get; set; }  MainFrame.Navigate(new DataPage());  Manager.MainFrame = MainFrame;  Update(SortComboBox.Text, (FiltComboBox.SelectedItem as String).ToString(), SearchBox.Text); | var allTypes = BetarPracticeEntities.GetContext().ProductTypes.ToList();  allTypes.Insert(0, new ProductType { Name = "Все виды" });  var Filt = new List<string>() { "Все типы" };  Filt.AddRange(BetarPracticeEntities.GetContext().ProductTypes.Select(c => c.Name).ToList());  FiltComboBox.ItemsSource = Filt;  FiltComboBox.SelectedIndex = 0;  SortComboBox.Items.Add("Цена (по возрастанию)");  SortComboBox.Items.Add("Цена (по убыванию)");  SortComboBox.Items.Add("В наличии (по возрастанию)");  SortComboBox.Items.Add("В наличии (по убыванию)");  SortComboBox.SelectedIndex = 0;  private void DGrid\_IsVisibleChanged(object sender, DependencyPropertyChangedEventArgs e)  {  BetarPracticeEntities.GetContext().ChangeTracker.Entries().ToList().ForEach(p => p.Reload());  DGrid.ItemsSource = BetarPracticeEntities.GetContext().Products.ToList();  Update(SortComboBox.Text, FiltComboBox.Text, SearchBox.Text);  } |
| private void Update(string sort = "", string filt = "", string search = "")  {  var data = BetarPracticeEntities.GetContext().Products.ToList();  if (!string.IsNullOrEmpty(search) && !string.IsNullOrWhiteSpace(search))  {  data = data.Where(p => p.Name.ToLower().Contains(search.ToLower())).ToList();  }  if (!string.IsNullOrWhiteSpace(filt) && !string.IsNullOrEmpty(filt))  {  if (filt != "Все типы")  {  data = data.Where(c => c.ProductType.Name == filt).ToList();  }  }  if (!string.IsNullOrEmpty(sort) && !string.IsNullOrWhiteSpace(sort))  {  if (sort == "Цена (по возрастанию)")  {  data = data.OrderBy(c => c.Price).ToList();  }  if (sort == "Цена (по убыванию)")  {  data = data.OrderByDescending(c => c.Price).ToList();  }  }  DGrid.ItemsSource = data;  } | if (e.Key == Key.Enter)  {  x.Text = "";  Update(SortComboBox.Text, FiltComboBox.Text, SearchBox.Text);  if (DGrid.Items.Count == 0)  {  x.Text = "По данному запросу ничего не найдено";  }  }  SearchBox.Text = "";  x.Text = "";  Update(SortComboBox.Text, FiltComboBox.Text, SearchBox.Text);  <ComboBox Name="SortComboBox"  Height="28"  Width="250"  Margin="5 0"  SelectionChanged="SortComboBox\_SelectionChanged"/> |