XAML List Box Lab - WPF

Perform these labs on your own computer using Visual Studio 2022 or later to ensure you understand the lessons presented in the corresponding videos and lectures.

Lab 1: Create a ListBox of Products

Create a new WPF application named **ListBoxSamples**.

Copy in the **Common.Library**, **EntityClasses**, and **RepositoryClasses** folders.

Right mouse-click on the **Project** and add a new Window named **SimpleListBox**.

Add a new XML namespace.

```
xmlns:data="clr-namespace:AdventureWorks.DataLayer"
```

Modify/Add the following attributes on the <Window> element.

```
WindowStartupLocation="CenterScreen"
Title="Simple List Box"
Height="600"
Width="800"
```

Add a <Window.Resources> element and add an ObjectDataProvider within it.

```
<Window.Resources>
  <ObjectDataProvider x:Key="productList"
                      ObjectType="{x:Type
data:ProductRepository}"
                      MethodName="Get" />
</Window.Resources>
```

Inside the <Grid> add the following XAML.

```
<ListBox Grid.Row="1"
         ItemsSource="{Binding Source={StaticResource
productList}}"
         DisplayMemberPath="Name">
</ListBox>
```

Call this Window

Open the **MainWindow.xaml** file.

Modify/Add the following attributes on the <Window> element.

```
WindowStartupLocation="CenterScreen"
Title="List Box Samples"
Height="600"
Width="800"
```

Replace the <Grid> with the following.

```
<StackPanel HorizontalAlignment="Center">
  <Button Click="SimpleListBox_Click">Simple List
Box</Button>
</StackPanel>
```

Open the **MainWindow.xaml.cs** file and add the SimpleListBox_Click() event procedure.

```
private void SimpleListBox_Click(object sender,
RoutedEventArgs e)
{
  new SimpleListBox().Show();
}
```

Try It Out

Run the application to see the results.

Lab 2: Implement a Simple Search

Open the **SimpleListBox.xaml** file and add the following attribute to the <ListBox> element.

```
TextSearch.TextPath="Name"
```

Try It Out

Run the application and click on a letter to see the list jump to that letter in the list.

Lab 3: Change the Default Data Template

Right mouse-click on the Project and add a new Window named **SimpleDataTemplate**.

Add a new XML namespace.

```
xmlns:data="clr-namespace:AdventureWorks.DataLayer"
```

Modify/Add the following attributes on the <Window> element.

```
WindowStartupLocation="CenterScreen"
Title="Simple List Box with Data Template"
Height="600"
Width="800"
```

Add a <Window.Resources> element and add an ObjectDataProvider within it.

```
<Window.Resources>
  <ObjectDataProvider x:Key="productList"
                      ObjectType="{x:Type
data:ProductRepository}"
                      MethodName="Get" />
</Window.Resources>
```

Inside the <Grid> add the following XAML.

```
<ListBox Grid.Row="1"
          ItemsSource="{Binding Source={StaticResource
productList}}">
  <ListBox.ItemTemplate>
    <DataTemplate>
      <TextBlock Text="{Binding Path=Name}" />
    </DataTemplate>
  </ListBox.ItemTemplate>
</ListBox>
```

Open the **MainWindow.xaml** file and add a new button within the <StackPanel>.

```
<Button Click="SimpleDataTemplate_Click">
  List Box with Simple Data Template
</Button>
```

Open the **MainWindow.xaml.cs** file and add a new event procedure.

```
private void SimpleDataTemplate_Click(object sender,
RoutedEventArgs e)
{
  new SimpleDataTemplate().Show();
}
```

Try It Out

Run the application to see the results.

Lab 4: Two Column List Box

Right mouse-click on the Project and add a new Window named **TwoColumnListBox**.

Add a new XML namespace.

```
xmlns:data="clr-namespace:AdventureWorks.DataLayer"
```

Modify/Add the following attributes on the <Window> element.

```
WindowStartupLocation="CenterScreen"
Title="Two Column List Box"
Height="600"
Width="800"
```

Add a <Window.Resources> element and add an ObjectDataProvider within it.

```
<Window.Resources>
  <ObjectDataProvider x:Key="productList"
                      ObjectType="{x:Type
data:ProductRepository}"
                      MethodName="Get" />
</Window.Resources>
```

Inside the <Grid> add the following XAML.

```
<ListBox Grid.Row="1"
          ItemsSource="{Binding Source={StaticResource
productList}}">
  <ListBox.ItemTemplate>
    <DataTemplate>
      <StackPanel Orientation="Horizontal">
        <TextBlock Text="{Binding Path=Name}" />
        <Separator Width="100" />
        <TextBlock Text="{Binding Path=ListPrice,
StringFormat={}{0:c}}" />
      </StackPanel>
    </DataTemplate>
  </ListBox.ItemTemplate>
</ListBox>
```

Open the **MainWindow.xaml** file and add a new button within the <StackPanel>.

```
<Button Click="TwoColumn_Click">
  Two Column List Box
</Button>
```

Open the **MainWindow.xaml.cs** file and add a new event procedure.

```
private void TwoColumn_Click(object sender,
RoutedEventArgs e)
{
  new TwoColumnListBox().Show();
}
```

Try It Out

Run the application to see the results.

Change to Grid

Modify the <DataTemplate> within the ListBox to look like the following.

```
<DataTemplate>
  <Grid MinWidth="300">
    <Grid.ColumnDefinitions>
      <ColumnDefinition Width="3*" />
      <ColumnDefinition Width="*" />
    </Grid.ColumnDefinitions>
    <TextBlock Text="{Binding Path=Name}" />
    <TextBlock Grid.Column="1"
               Text="{Binding Path=ListPrice,
StringFormat={}{0:c}}" />
  </Grid>
</DataTemplate>
```

Try It Out

Run the application to see the results.

Lab 5: Multi-Line List Box

Right mouse-click on the Project and add a new Window named **MultiLineListBox**.

Add a new XML namespace.

```
xmlns:data="clr-namespace:AdventureWorks.DataLayer"
```

Modify/Add the following attributes on the <Window> element.

```
WindowStartupLocation="CenterScreen"
Title="Multi Line List Box"
Height="600"
Width="800"
```

Add a <Window.Resources> element and add an ObjectDataProvider within it.

```
<Window.Resources>
  <ObjectDataProvider x:Key="productList"
                      ObjectType="{x:Type
data:ProductRepository}"
                      MethodName="Get" />
</Window.Resources>
```

Inside the <Grid> add the following XAML.

```
<ListBox Grid.Row="1"
          ItemsSource="{Binding Source={StaticResource
productList}}">
  <ListBox.ItemTemplate>
    <DataTemplate>
      <Grid MinWidth="300">
        <Grid.RowDefinitions>
          <RowDefinition Height="Auto" />
          <RowDefinition Height="Auto" />
        </Grid.RowDefinitions>
        <TextBlock FontSize="14"
                   FontWeight="Bold"
                   Text="{Binding Path=Name}" />
        <Grid Grid.Row="1">
          <Grid.ColumnDefinitions>
            <ColumnDefinition Width="Auto" />
            <ColumnDefinition Width="Auto" />
          </Grid.ColumnDefinitions>
          <StackPanel Orientation="Horizontal">
            <Label>Cost:</Label>
            <TextBlock VerticalAlignment="Center"
                       Text="{Binding
Path=StandardCost, StringFormat={}{0:c}}" />
          </StackPanel>
          <StackPanel Grid.Column="1"
                      Orientation="Horizontal">
            <Label>List Price:</Label>
            <TextBlock VerticalAlignment="Center"
                       Text="{Binding Path=ListPrice,
StringFormat={}{0:c}}" />
          </StackPanel>
        </Grid>
      </Grid>
    </DataTemplate>
  </ListBox.ItemTemplate>
</ListBox>
```

Open the **MainWindow.xaml** file and add a new button within the <StackPanel>.

```
<Button Click="MultiLine_Click">
  Multi Line List Box
</Button>
```

Open the **MainWindow.xaml.cs** file and add a new event procedure.

```
private void MultiLine_Click(object sender,
RoutedEventArgs e)
{
  new MultiLineListBox().Show();
}
```

Try It Out

Run the application to see the results.

Lab 6: Horizontal List Box

Add the following to the <ListBox> on the MultiLineListBox window.

```
<ListBox.ItemsPanel>
  <ItemsPanelTemplate>
    <StackPanel Orientation="Horizontal" />
  </ItemsPanelTemplate>
</ListBox.ItemsPanel>
```

Wrap a <Border> around the <Grid> in the DataTemplate. Modify the Width attribute on the Grid to 200.

```
<Border BorderBrush="Black"
        BorderThickness="2"
        Padding="4"
        Margin="4">
  <Grid MinWidth="200">
   // REST OF THE XAML HERE
  </Grid>
</Border>
```

Try It Out

Run the application to see the results.