**Drag and Drop** shows how to use a **ListBox** to create a simple drag-and-drop example

## Step 1

|  |  |
| --- | --- |
|  | Follow **Setup and Start** on how to Install and/or Get Started with **Visual Studio 2019** if not already or in **Windows 10** choose **Start**, find and select **Visual Studio 2019** then from the **Get started** screen select **Create a new project** |
| A screenshot of a cell phone  Description automatically generated | Then choose **Blank App (Universal Windows)** and select **Next** and then in **Configure your new project** enter the **Project name** as **DragAndDrop** and select **Create** |
| A screenshot of a social media post  Description automatically generated | Finally, in **New Universal Windows Platform Project** pick the **Target version** and **Minimum version** to be at least **Windows 10, version 1903 (10.0; Build 18362)** and then select **OK** |

**Target Version** will control the most recent features of **Windows 10** your application can use. To make sure you always have the most recent version, check for any **Notifications** or **Updates** in **Visual Studio 2019**

## Step 2

|  |  |
| --- | --- |
| A screenshot of a cell phone  Description automatically generated | Choose **Project** then **Add New Item...** from the **Menu** in **Visual Studio 2019** |

## Step 3

|  |  |
| --- | --- |
| A close up of a logo  Description automatically generated | Then choose **Code File** from **Add New Item** in **Visual Studio 2019**, enter the **Name** as **Library.cs** and select **Add** |

## Step 4

In the **Code** View of **Library.cs** will be displayed and in this the following should be entered:

|  |
| --- |
| using System;  using System.Collections.ObjectModel;  using System.Linq;  public class Item  {  public Guid Id { get; set; }  public string Value { get; set; }  }  public class Library  {  public ObservableCollection<Item> Items { get; set; }  = new ObservableCollection<Item>();  public void Add(string value)  {  Items.Add(new Item  {  Id = Guid.NewGuid(),  Value = value  });  }  public void Remove(Guid id)  {  Item result = Items.FirstOrDefault(item => item.Id == id);  if (result != null)  {  Items.Remove(result);  }  }  } |

There is a using statement to include functionality needed for the application. There is a **class** for Item and then there is an ObservableCollection of Item. Add(...) **method** will Add a new Item to the ObservableCollection of Item. Remove(...) **method** will use LINQ to get an Item with FirstOrDefault by Id and if this is not null will Remove it

## Step 5

|  |  |
| --- | --- |
|  | In the **Solution Explorer** of **Visual Studio 2019** select **MainPage.xaml** |

## Step 6

|  |  |
| --- | --- |
| A screenshot of a cell phone  Description automatically generated | Choose **View** then **Designer** from the **Menu** in **Visual Studio 2019** |

## Step 7

In the **Design** View and **XAML** View of **Visual Studio 2019** will be displayed, and in this between the **Grid** and **/Grid** elements enter the following **XAML**:

|  |
| --- |
| <Grid>  <Grid.RowDefinitions>  <RowDefinition Height="Auto"/>  <RowDefinition Height="\*"/>  </Grid.RowDefinitions>  <TextBox Grid.Row="0" Margin="20" Name="Value"/>  <ListView Grid.Row="1" VerticalAlignment="Stretch" AllowDrop="True"  CanReorderItems="True" SelectionMode="Single" Name="Display">  <ListView.ItemTemplate>  <DataTemplate>  <TextBlock FontFamily="Segoe UI" FontSize="16"  Text="{Binding Value}"/>  </DataTemplate>  </ListView.ItemTemplate>  </ListView>  </Grid>  <CommandBar VerticalAlignment="Bottom">  <AppBarButton Icon="Add" Label="Add" Click="Add\_Click"/>  <AppBarButton Icon="Remove" Label="Remove" Click="Remove\_Click"/>  </CommandBar> |

The first block of **XAML** is the main user interface with a **Grid** with two rows – the first contains a **TextBox** for the value to add and the second row is a **ListBox** to show added items with **AllowDrop** and **CanReorderItems** properties set enabling the Control to support drag-and-drop. The second block of **XAML** is the **CommandBar** which contains the **Add** – to add to the **ListBox** and **Remove** - to remove items from the **ListBox**

## Step 8

|  |  |
| --- | --- |
|  | Choose **View** then **Code** from the **Menu** in **Visual Studio 2019** |

## Step 9

Once in the **Code** View, below the end of **public MainPage() { ... }** the following Code should be entered:

|  |
| --- |
| Library library = new Library();  protected override void OnNavigatedTo(NavigationEventArgs e)  {  Display.ItemsSource = library.Items;  }  private void Add\_Click(object sender, RoutedEventArgs e)  {  library.Add(Value.Text);  }  private void Remove\_Click(object sender, RoutedEventArgs e)  {  library.Remove(((Item)Display.SelectedItem).Id);  } |

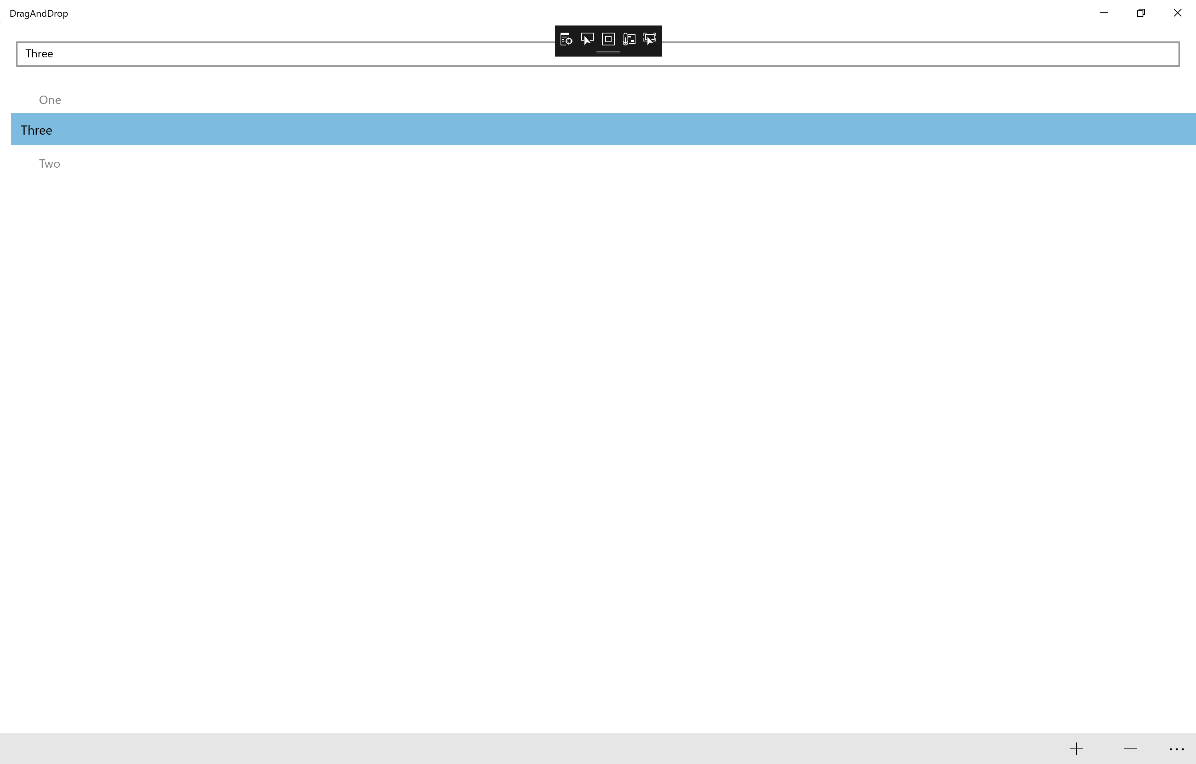
Below the **MainPage(...)** method an instance of the Library **Class** is created. In the OnNavigatedTo **event** handler the ListBox has its ItemsSource set to the ObservableCollection in the Library **class**. Add\_Click(...) will use Add from the Library **class** with the Text from the TextBox and Remove\_Click(...) will use Remove from the Library **class**

## Step 10

|  |  |
| --- | --- |
|  | That completes the **Universal Windows Platform** Application, in **Visual Studio 2019** select **Local Machine** to run the Application |

## Step 11

Once the Application is running you can then type in some text then click **Add**, you can then add multiple items. Then tap and hold or select with a mouse on any of the items to move them up and down the list



## Step 12

|  |  |
| --- | --- |
| A picture containing object  Description automatically generated | To Exit the Application, select the **Close** button in the top right of the Application |