Android, iOS and Hybrid Applications

Mobile-Development

Review Day 1

- Create a solution
- Upload to github
- Don’t forget the .gitignore
- If you forgot it:
  - Delete the bin folder
  - Check in the deleted files
  - Check in the .gitignore

https://github.com/github/gitignore/blob/master/VisualStudio.gitignore

Day 2

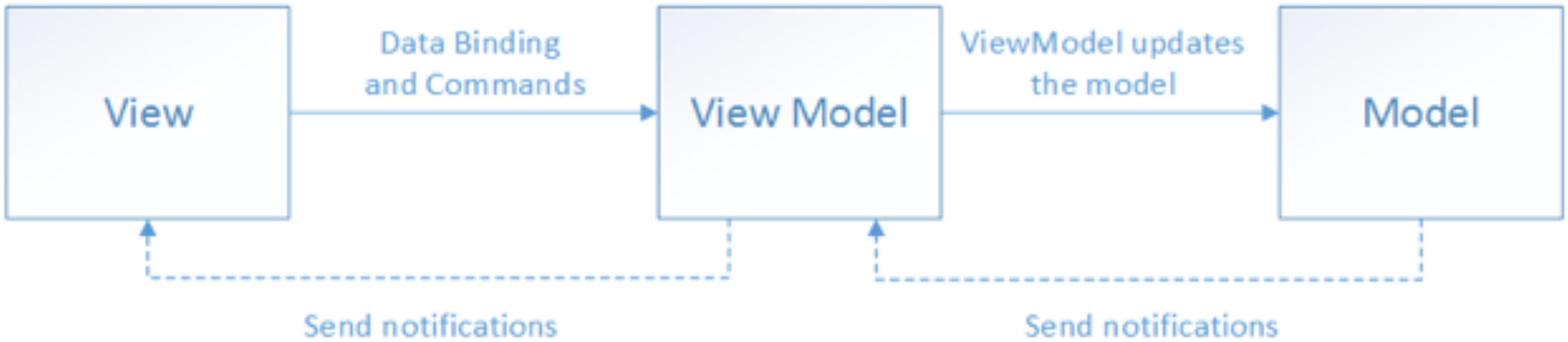
- MVVM
- XAML for Forms
- Controls and differences to WPF
- Bindings
- Commands

MVVM Pattern

- Model View ViewModel
- Decouple the logic from the views
- Use Bindings to connect them
- It’s hard to test code-behind or views in UnitTests
- A ViewModel generally doesn’t know his view

https://docs.microsoft.com/en-us/xamarin/xamarin-forms/enterprise-application-patterns/mvvm

MVVM PATTERN



https://docs.microsoft.com/en-us/xamarin/xamarin-forms/enterprise-application-patterns/mvvm

ViewModels

- The logical part should be in here
  - whether a control is enabled or not
  - what selections are available in lists
  - what is executed when clicking something
  - how a value is calculated (e.g. age from birthdate)

ViewModels – Notifying the View

- INotifyPropertyChanged
- Create a base class for ViewModels

```
protected void RaisePropertyChanged([CallerMemberName] string property = "")
{
  PropertyChanged?.Invoke(this, new PropertyChangedEventArgs(property));
}

public bool IsSelected
{
  get => _isSelected;
  set
  {
    _isSelected = value;
    RaisePropertyChanged();
  }
}
```

https://docs.microsoft.com/en-us/dotnet/api/system.componentmodel.inotifypropertychanged?view=netstandard-2.0

VIEWMODELS – LISTS

- ObservableCollection<T>
- INotifyCollectionChanged (already implemented)

```
public ObservableCollection<TodoItemViewModel> Items
{
  get => _items;
  set
  {
    _items = value;
    RaisePropertyChanged();
  }
}
```

https://docs.microsoft.com/en-us/dotnet/api/system.collections.specialized.inotifycollectionchanged?view=netstandard-2.0

Model – ViewModel – Distinction

- A Model usually does not contain any change notification
- Represents the plain data - what normally is stored in the database
- Should not contain any logic

Questions?

Setup the Basics

- Short walkthrough
- Example branch: *day2/viewModels_services*

- Setup your ViewModels & Models
- Add the necessary properties
- Raise PropertyChanged where necessary

Xamarin Forms

- Page - represents a Screen
  - ContentPage
  - NavigationPage
  - TabbedPage
  - CarouselPage
  - MasterDetailPage

https://docs.microsoft.com/en-us/xamarin/xamarin-forms/user-interface/controls/pages

XAMARIN FORMS – LAYOUT

- Layouts - Subtypes of View
  - ContentView - ContentControl
  - Frame - Border
  - ScrollView - ScrollViewer
  - StackLayout - StackPanel
  - **Grid**
  - FlexLayout - based on the CSS flexbox-model

https://docs.microsoft.com/en-us/xamarin/xamarin-forms/user-interface/controls/layouts

Xamarin Forms – Layout – Grid

- One of the most powerful layout controls
- Also available in WPF
- Define rows and columns (*Row/ColumnDefinition*)
- Add your children and give them a row or column index

https://docs.microsoft.com/en-us/xamarin/xamarin-forms/user-interface/layouts/grid

Xamarin Forms – Layout – Grid

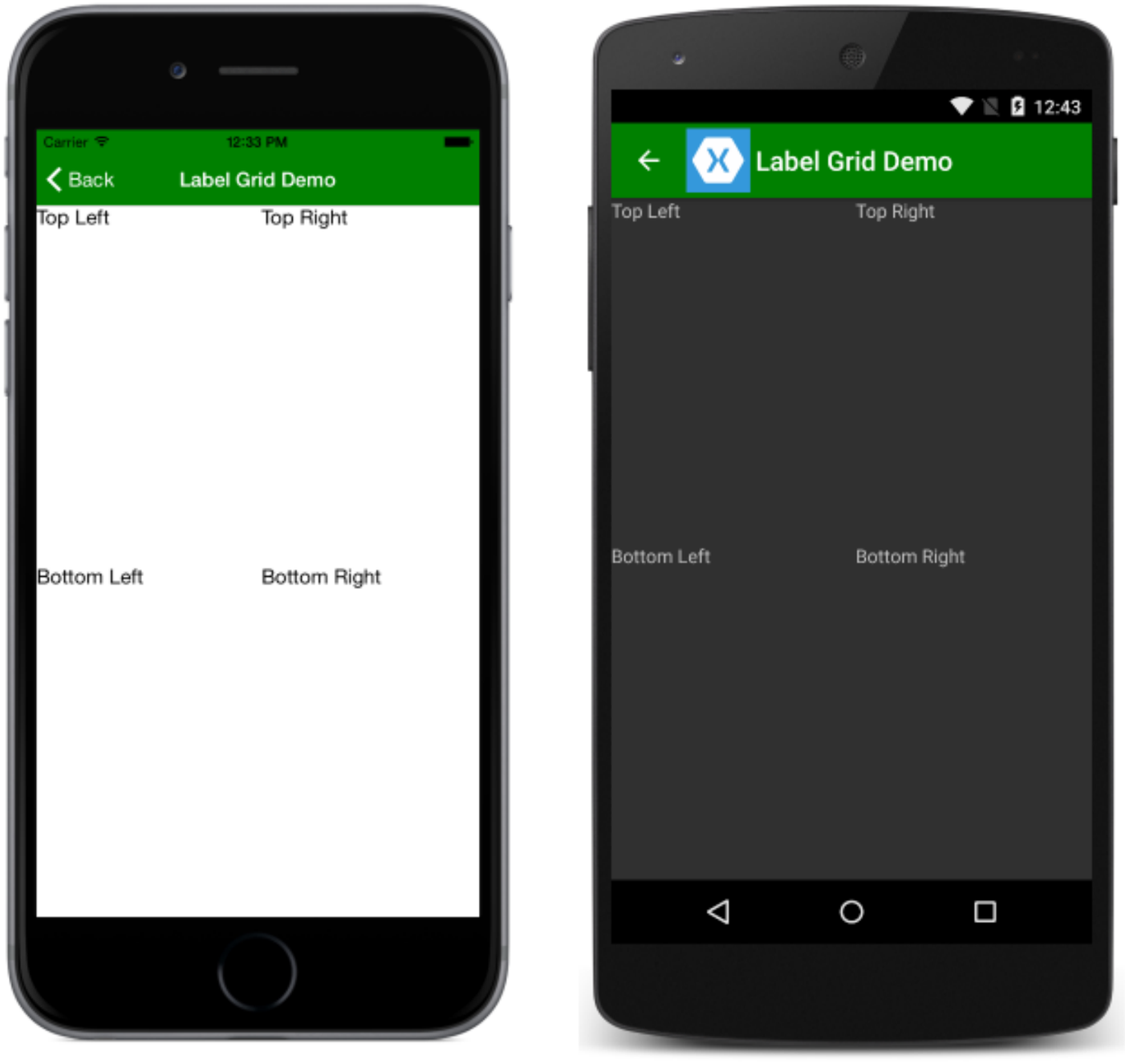
- Sizes
  - Auto - calculated from the children
  - Proportional (*-sized)
  - Absolute (fixed)
- Defaults to *-sized

Xamarin Forms – Layout – Grid

```
<Grid>
  <Grid.RowDefinitions>
    <RowDefinition Height="*" />
    <RowDefinition Height="*" />
  </Grid.RowDefinitions>
  <Grid.ColumnDefinitions>
    <ColumnDefinition Width="*" />
  </Grid.ColumnDefinitions>

  <Label Text="Top Left" Grid.Row="0" Grid.Column="0" />
  <Label Text="Top Right" Grid.Row="0" Grid.Column="1" />
  <Label Text="Bottom Left" Grid.Row="1" Grid.Column="0" />
  <Label Text="Bottom Right" Grid.Row="1" Grid.Column="1" />
</Grid>
```

XAMARIN FORMS – LAYOUT – GRID



Xamarin Forms – Controls

- Label
- Button
- Entry - TextBox
- ListView - List
- Picker - ComboBox
- DatePicker

https://docs.microsoft.com/en-us/xamarin/xamarin-forms/user-interface/controls/index

Xamarin Forms – Controls & Layouts Examples

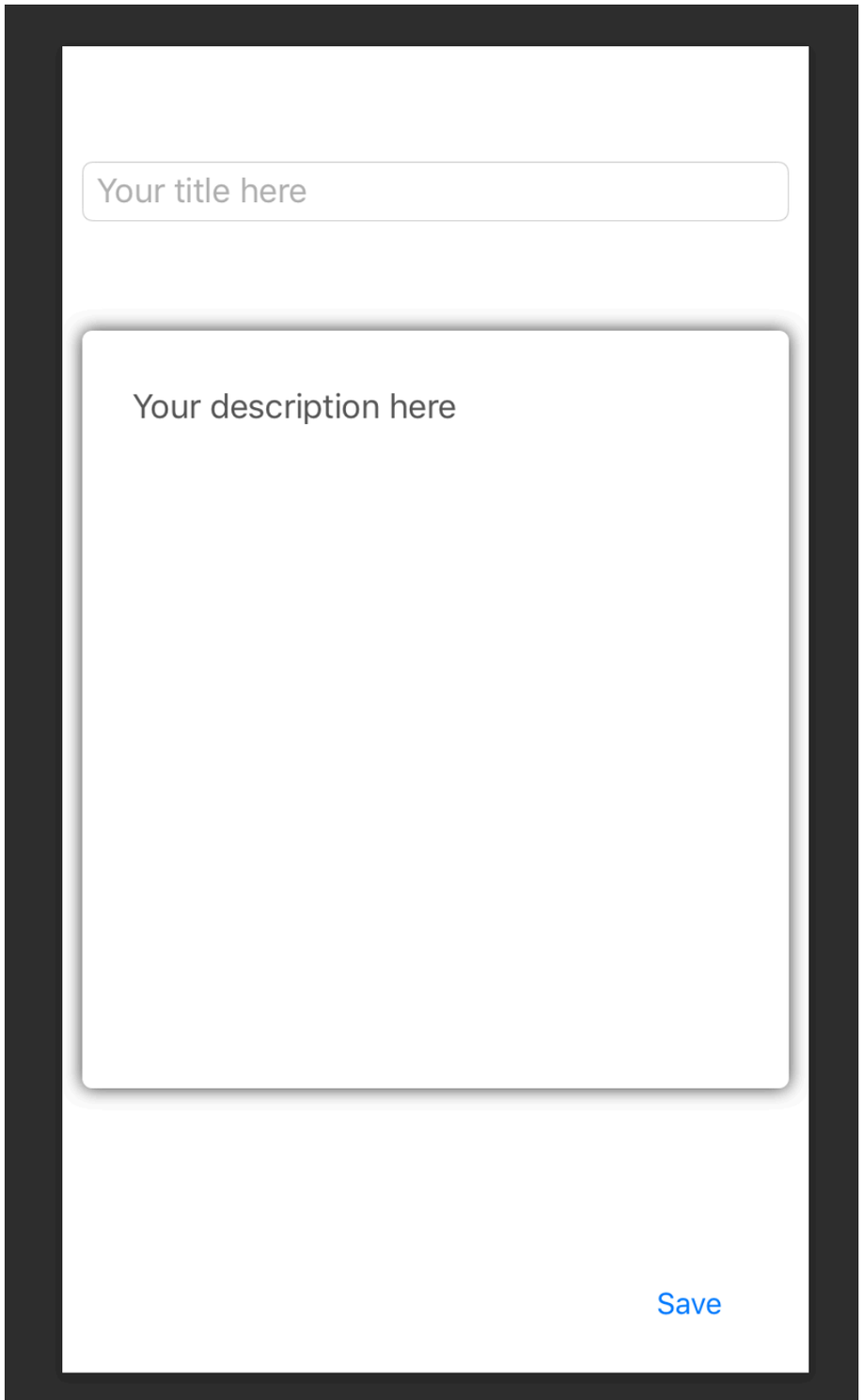
```
<Grid Margin="10">
    <Grid.RowDefinitions>
        <RowDefinition Height="*" />
        <RowDefinition Height="3*" />
        <RowDefinition Height="*" />
    </Grid.RowDefinitions>

    <Entry Grid.Row=“0" VerticalOptions=“Center" Placeholder="Your title here” />

    <Frame Grid.Row="1">
        <Editor VerticalOptions=“FillAndExpand" Placeholder="Your description here” />
    </Frame>

    <Button Grid.Row=“2" HorizontalOptions=“End" VerticalOptions=“End"
            WidthRequest=“100" HeightRequest=“50" Text="Save" />
</Grid>
```

XAMARIN FORMS – CONTROLS EXAMPLES



XAMARIN FORMS – CONTROLS ALIGNMENT

- VerticalOptions and HorizontalOptions
  - Start, End, Center and Fill (AndExpand)
- Expansion is used in StackLayouts

https://docs.microsoft.com/en-us/xamarin/xamarin-forms/user-interface/layouts/layout-options

XAMARIN FORMS – CONTROLS ALIGNMENT

Xamarin Forms – Label

- Text - displayed on screen

https://docs.microsoft.com/en-us/xamarin/xamarin-forms/user-interface/text/label

XAMARIN FORMS – BUTTON

- Text - displayed on screen
- Command - executed on click
- CommandParameter - passed to the command

https://docs.microsoft.com/en-us/xamarin/xamarin-forms/user-interface/button

XAMARIN FORMS – ENTRY

- Text - the entered and displayed text
- Placeholder - text shown when empty
- MaxLength - maximum amount of text to fit in
- IsPassword - uses * to display characters
- ClearButtonVisibility - when to display a clear button
- Keyboard - customise the type of keyboard that will show

https://docs.microsoft.com/en-us/xamarin/xamarin-forms/user-interface/text/entry

Xamarin Forms – Editor

- Multiline Text Edit
  - Most of the properties from Entry are supported

https://docs.microsoft.com/en-us/xamarin/xamarin-forms/user-interface/text/editor

XAMARIN FORMS – MISC

- Date Picker

  https://docs.microsoft.com/en-us/xamarin/xamarin-forms/user-interface/datepicker

- Picker (ComboBox)

  https://docs.microsoft.com/en-us/xamarin/xamarin-forms/user-interface/picker/

- CheckBox

  https://docs.microsoft.com/en-us/xamarin/xamarin-forms/user-interface/checkbox

Xamarin Forms – Preview

Tools > Options > Xamarin > Xamarin.Forms XAML Previewer

Check the “Troubleshooting” section in the link

https://docs.microsoft.com/en-us/xamarin/xamarin-forms/xaml/xaml-previewer

Questions?

Practice

- Example
- Extend your app with the necessary controls

Xamarin Forms – Bindings

- Similar to WPF
- Define the “BindingContext“ (ViewModel)
- {Binding Path=Value} Syntax
- BindingMode like WPF (Default, One, Two)
- INotifyPropertyChanged
- IValueConverter to convert “on the fly“

https://docs.microsoft.com/en-us/xamarin/xamarin-forms/app-fundamentals/data-binding/basic-bindings

Xamarin Forms – Bindings

- No support for RelativeSource
- BindingContext is the “Source”
- BindingContext is inherited
- x:Reference to reference a control by name

Xamarin Forms – "Compiled Bindings"

- Will trigger compile errors
- Improve performance
- Use the x:DataType on a Visual to set the context

https://docs.microsoft.com/en-us/xamarin/xamarin-forms/app-fundamentals/data-binding/compiled-bindings

Xamarin Forms – “Compiled Bindings”

```
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class=“Todo.Views.TodoDetailPage"

             x:DataType="viewModels:TodoItemViewModel"

             Title="Details">
    <ContentPage.Content>
        <Grid Margin="10">
            <Grid.RowDefinitions>
                <RowDefinition Height="1*" />
                <RowDefinition Height="1*" />
                <RowDefinition Height="3*" />
                <RowDefinition Height="1*" />
            </Grid.RowDefinitions>
            <Entry Grid.Row="0" VerticalOptions="Center" Text="{Binding Title}" />
            <Picker Grid.Row="1" ItemsSource="{Binding Priorities}" SelectedItem="{Binding Priority, Mode=TwoWay}" />
            <Frame Grid.Row="2">
                <Editor VerticalOptions="FillAndExpand" Text="{Binding Description}" />
            </Frame>
            <Button Grid.Row="3" Command="{Binding SaveCommand}" Text="Save" />
        </Grid>
    </ContentPage.Content>
</ContentPage>
```

Xamarin Forms – Commands

- If you want to execute something
- Use the Command class with a delegate
- Don’t use CanExecute for now

```
<Button Command="{Binding Path=MyCommand}" Text="Click me" />
```

```
AddTodoCommand = new Command(() => ...);
```

https://docs.microsoft.com/en-us/xamarin/xamarin-forms/app-fundamentals/data-binding/commanding

XAMARIN FORMS – COMMANDS WITH PARAMETERS

- Use the generic Command class
- Make sure to set the CommandParameter in the code

```
<Button Command="{Binding Path=MyCommand}" CommandParameter="1" Text="Click" />

AddTodoCommand = new Command<int>((parameter) => …);
```

Xamarin Forms – Commands

- The class is already available in Xamarin.Forms
- Don’t write your own implementation

Xamarin Forms – Commands

XAML-File:

```
<Button Command="{Binding SaveCommand}" Text="Save" />
```

ViewModel:

```
public ICommand SaveCommand => new Command(() =>
{
  _listViewModel.Items.Add(this);
  _navigation.PopAsync();
});
```

XAMARIN FORMS – LISTVIEW

- ItemsSource –> use ObservableCollection<T>
- ItemTemplate to override appearance
- SelectedItem to listen for selection

https://docs.microsoft.com/en-us/xamarin/xamarin-forms/user-interface/listview/

Xamarin Forms – ListView

ViewModel:

```
public ObservableCollection<TodoItemViewModel> Items
{
  get => _items;
  set
  {
    _items = value;
    RaisePropertyChanged();
  }
}

private ObservableCollection<TodoItemViewModel> _items;
```

View:

```
<ListView
    ItemsSource="{Binding Items}"
    SelectedItem="{Binding SelectedItem, Mode=TwoWay}" />
```

Xamarin Forms – ListView

```
<ListView ItemsSource="{Binding Path=MyList}">
    <ListView.ItemTemplate>
        <DataTemplate>
            <ViewCell>
                <StackLayout>
                    <Label Text="{Binding Firstname}" />
                    <Label Text="{Binding Lastname}" />
                </StackLayout>
            </ViewCell>
        </DataTemplate>
    </ListView.ItemTemplate>
</ListView>
```

XAMARIN FORMS – VIEWMAPPER

- Simple Service to map a ViewModel to a View

```
public Page Map<TViewModel>(TViewModel viewModel) where TViewModel : class
{
  Page result;
  switch (viewModel)
  {
    case MainViewModel mainViewModel:
      result = new MainPage { BindingContext = mainViewModel };
      break;
    case TodoListViewModel listViewModel:
      result = new TodoListPage { BindingContext = listViewModel };
      break;
    case TodoItemViewModel detailViewModel:
      result = new TodoDetailPage { BindingContext = detailViewModel };
      break;
    default:
      throw new InvalidNavigationException($"Could not map type [{typeof(TViewModel)}]");
  }

  return result;
}
```

XAMARIN FORMS – IVALUECONVERTER

- Change the value of a binding for the control
- Implement the IValueConverter interface

```
<ContentPage.Resources>
  <ResourceDictionary>
    <local:IntToBoolConverter x:Key="Converter" />
  </ResourceDictionary>
</ContentPage.Resources>

<Button
    IsEnabled="{Binding Path=SomeValue, Converter={StaticResource Converter}}”
    Text="Click me" />
```

https://docs.microsoft.com/en-us/xamarin/xamarin-forms/app-fundamentals/data-binding/converters

Practice

- Example
- Extend your app with the basic bindings
- Apply commands for the navigation

Additional Tasks

- Extend your ListView DataTemplate with a converter for color or an icon
  https://docs.microsoft.com/en-us/xamarin/xamarin-forms/user-interface/images
- Think about how you could better organise the services and startup without using statics