

Visual Studio

LIVE!

EXPERT SOLUTIONS FOR .NET DEVELOPERS

San Diego

Creating Four Beautiful Apps at Once

An Intro to Xamarin.Forms

Matthew Soucoup




Sr Cloud Developer Advocate

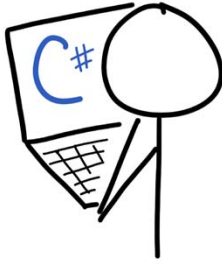
Microsoft

Level: Beginner

 Code Again for the First Time! 







@codemillmatt

## Where to Find Stuff

### The Talk

Code & Slides:

[github.com/codemillmatt/four-beautiful-apps](https://github.com/codemillmatt/four-beautiful-apps)

Matt Soucoup – Sr. Cloud Developer Advocate @ Microsoft

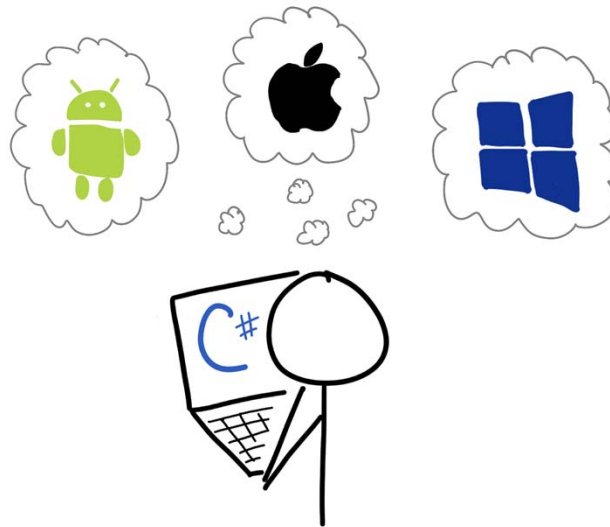
I ❤️ You

[@codemillmatt](https://twitter.com/codemillmatt)

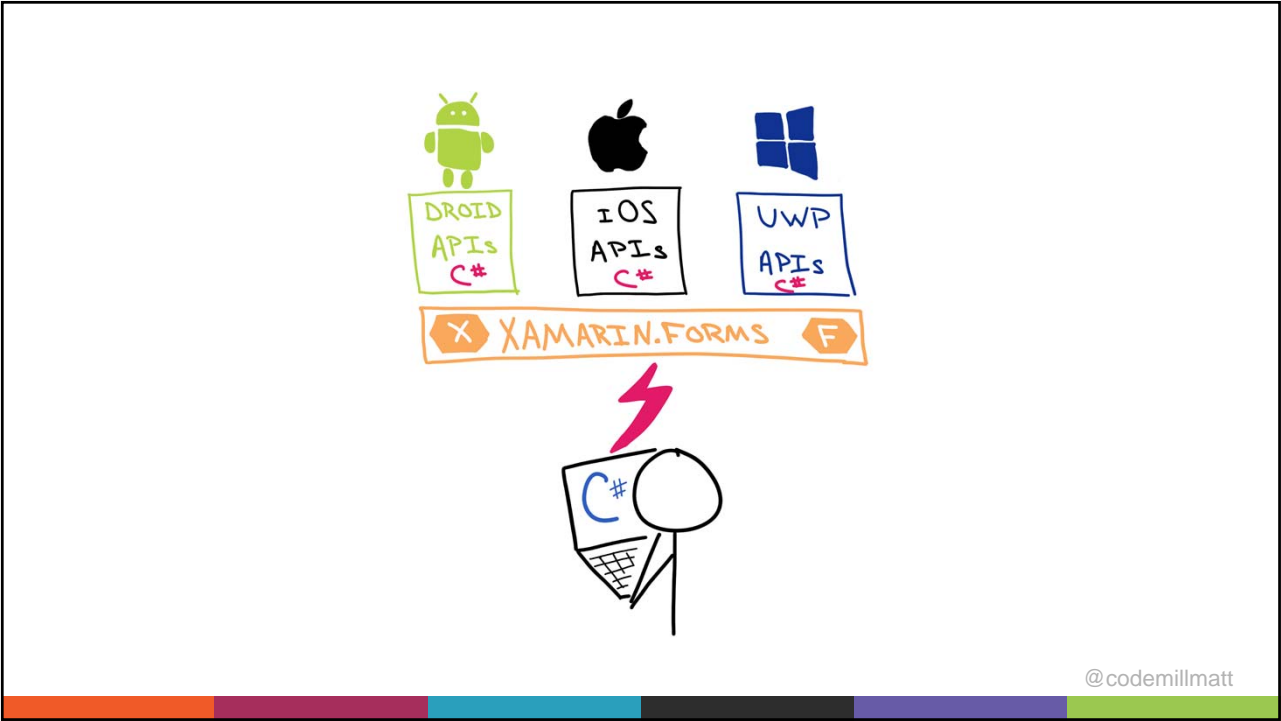
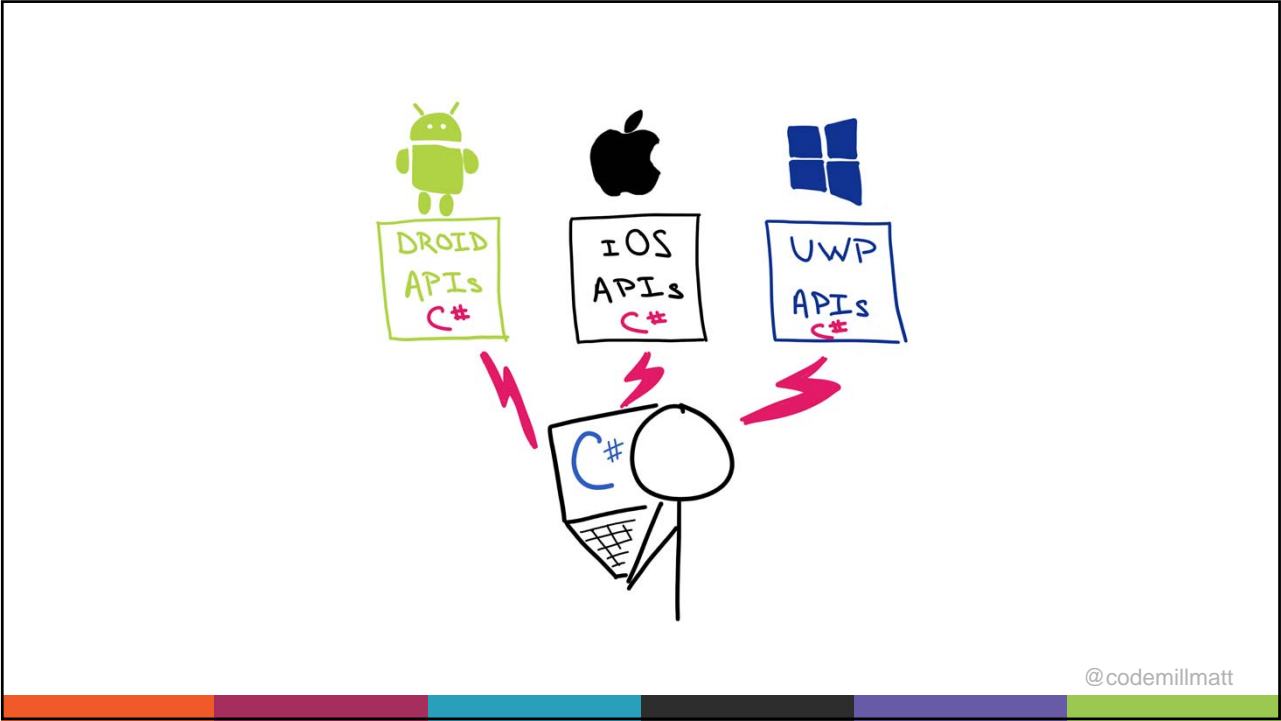
[codemillmatt@microsoft.com](mailto:codemillmatt@microsoft.com)

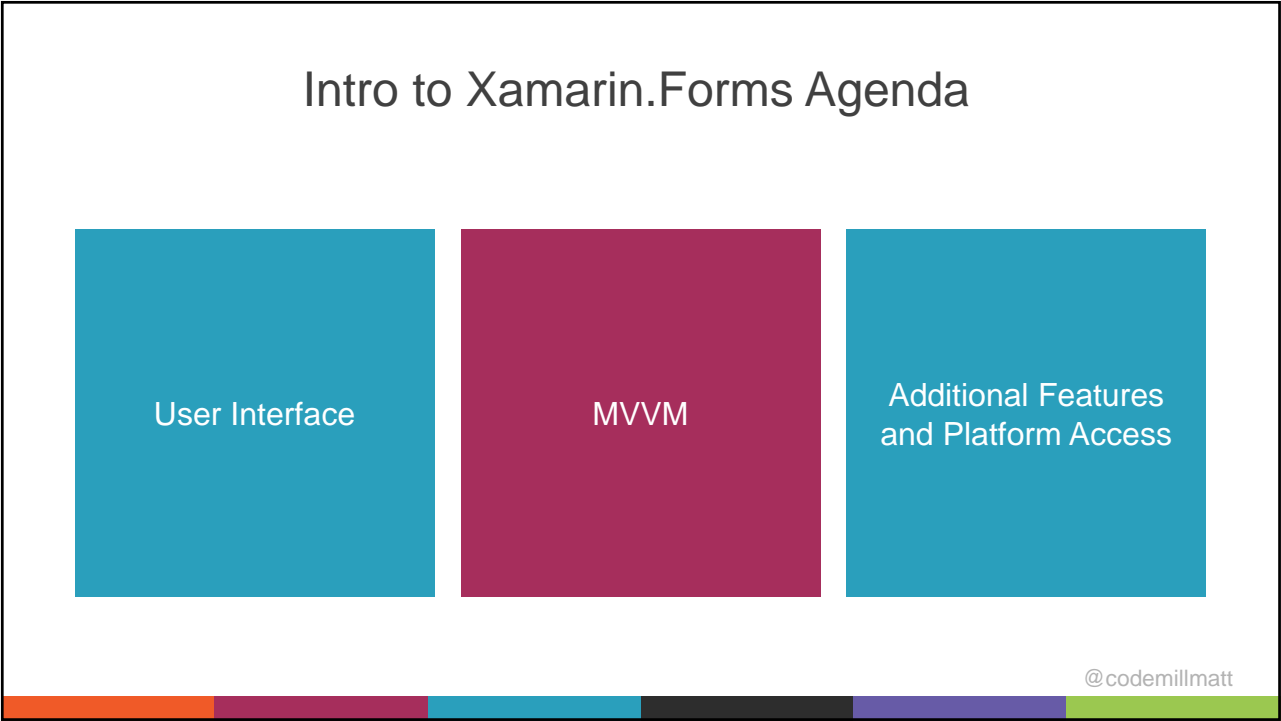
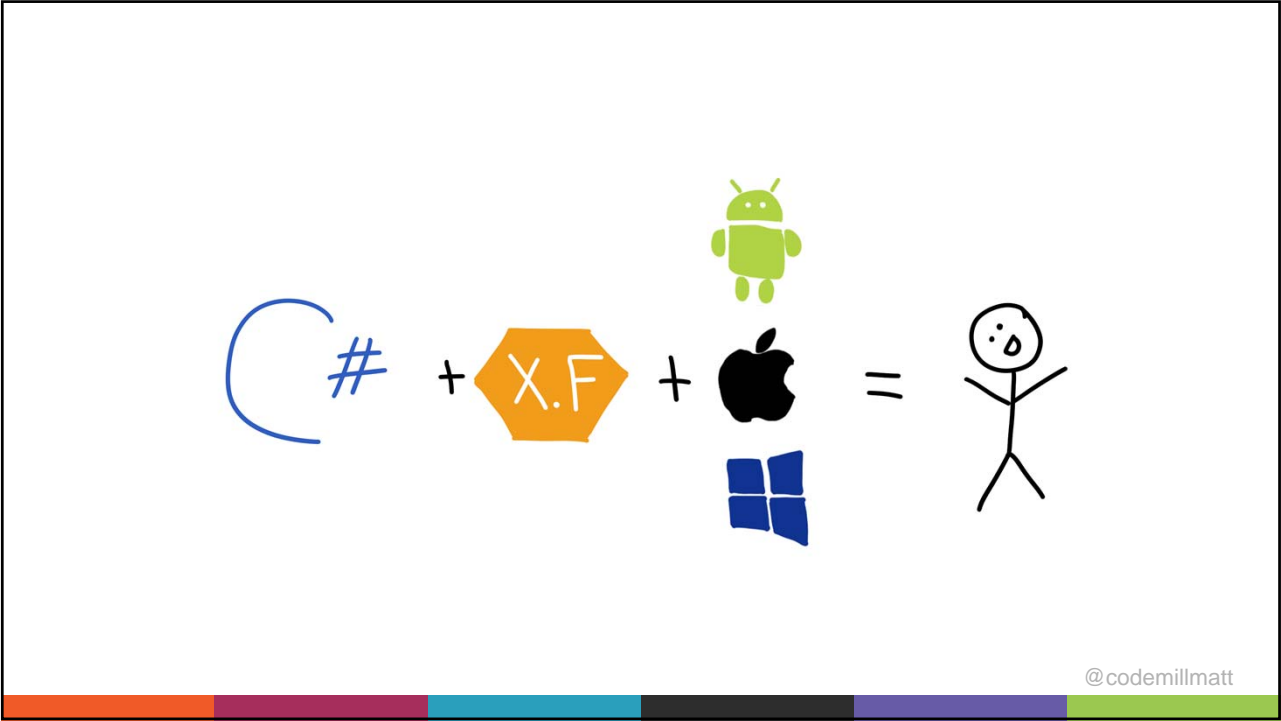
[codemillmatt.com](https://codemillmatt.com)

@codemillmatt



@codemillmatt



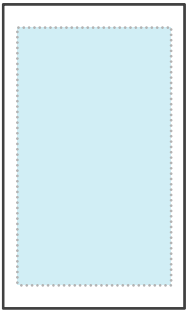


# User Interface

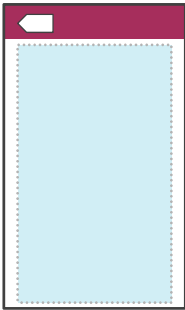
---

@codemillmatt

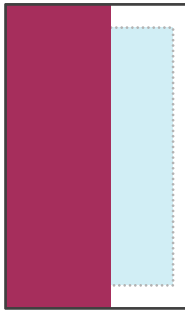
# Pages



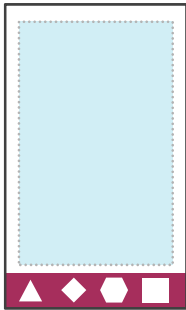
Content



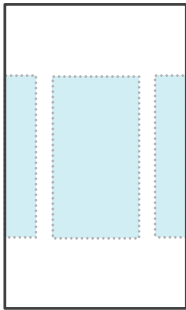
Navigation



Master/Detail



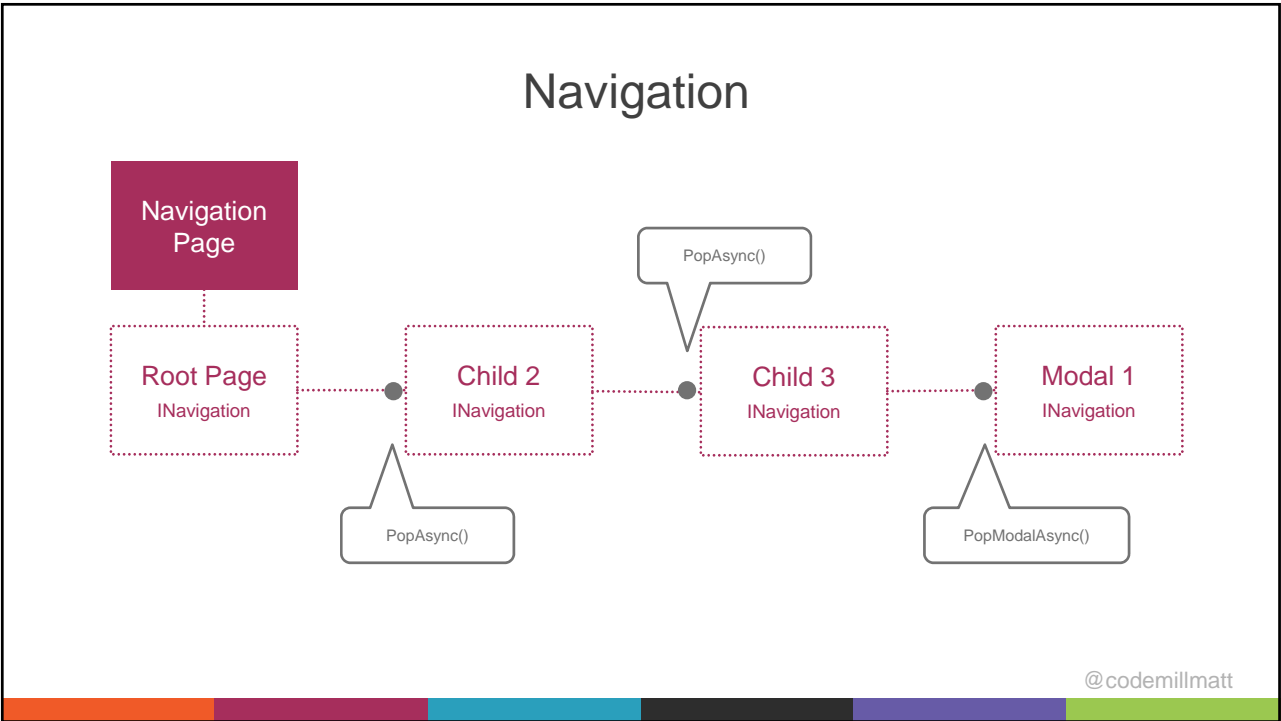
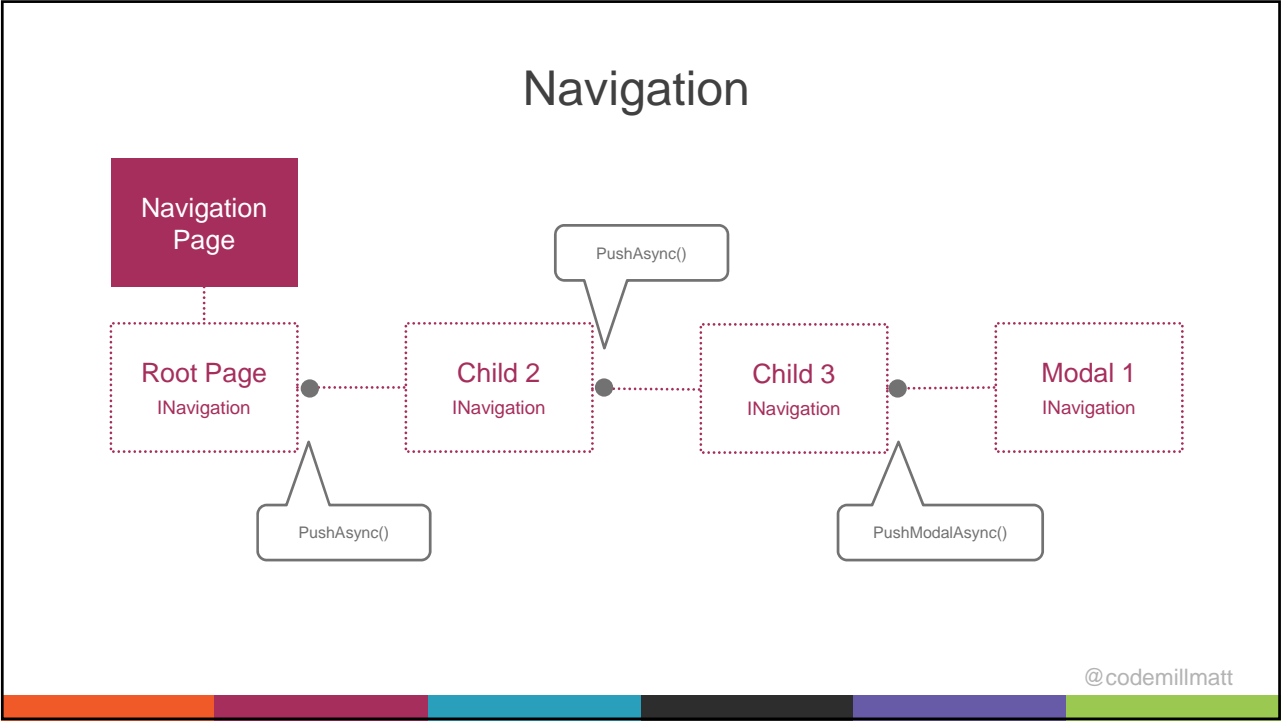
Tabbed




Carousel

@codemillmatt

T02 - Creating Four Beautiful Apps At Once with Xamarin.Forms - Matthew Soucoup

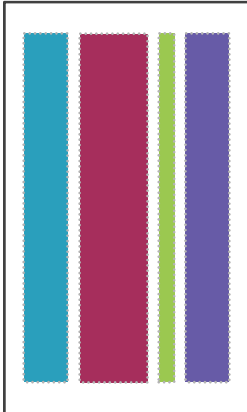


# Layouts - Stack



A diagram showing a vertical stack layout. A container box contains six horizontal bars stacked on top of each other. From top to bottom, the bars are: light blue, maroon, light green, purple, orange, and light blue. The bars have varying heights, with the light green bar being the tallest and the purple bar being the shortest.

Vertical



A diagram showing a horizontal stack layout. A container box contains four vertical bars placed side-by-side. From left to right, the bars are: light blue, maroon, light green, and purple. The bars have varying widths, with the maroon bar being the widest and the light green bar being the narrowest.

Horizontal

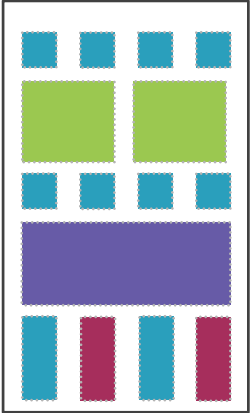
@codemillmatt

# Layouts - Grid

A diagram illustrating a grid layout. A large rectangle is divided into a 4x4 grid of smaller squares. The squares are colored: the top row has four blue squares; the second row has four red squares; the third row has a large green square on the left, a small green square in the middle, and a small orange square on the right; the bottom row has a large purple square on the left, a small green square in the middle, and a small orange square on the right.

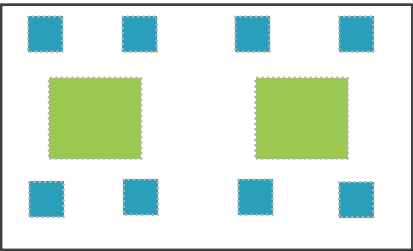
@codemillmatt

Layouts - Flex



@codemillmatt

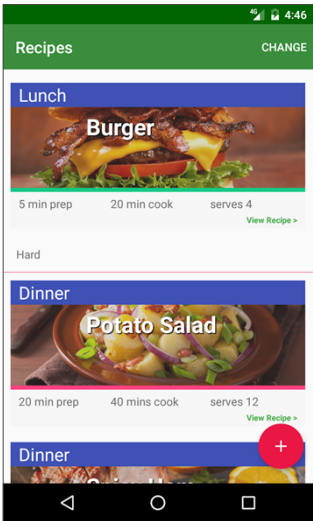
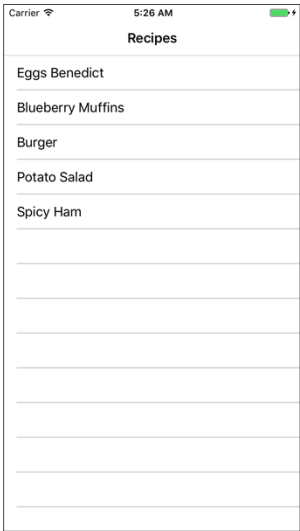
Layouts - Flex



@codemillmatt

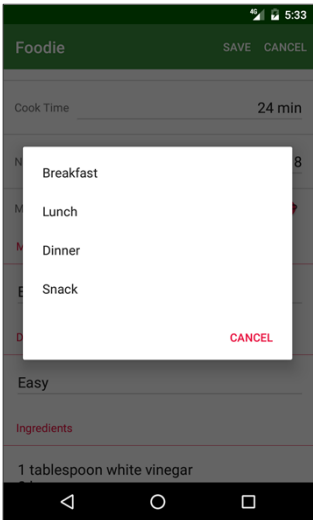
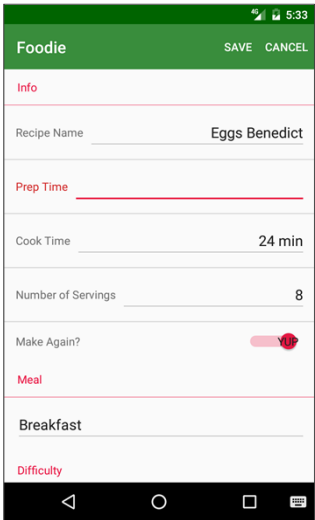


ListView



@codemillmatt

TableView



@codemillmatt

Controls

Image

Label


ToolbarItem

BoxView

Button

Carrier 12:11 PM

< Recipes Details Edit



5 min prep 24 min cook serves 8

Breakfast

\*\* Easy difficulty \*\*

Ingredients

1 tablespoon white vinegar  
8 large eggs  
Hollandaise Sauce  
1/2 pound (16 slices) Canadian bacon  
4 English muffins, split in half, toasted

Directions

1. Fill a large saucepan with about 4 inches of water, add vinegar, and bring to a boil. Fill a shallow dish or pie plate with warm water. Reduce heat under saucepan to medium, so water is just barely simmering. Break 1 egg at a

Make Again!

@codemillmatt

Controls

Entry

Picker

ToolbarItem

Switch

Editor

Carrier 12:12 PM

Foodie Save Cancel

Cook Time 24 min

Number of Servings 8

Make Again? ☒

MEAL

Breakfast

DIFFICULTY

Easy

INGREDIENTS

1 tablespoon white vinegar  
8 large eggs  
Hollandaise Sauce  
1/2 pound (16 slices) Canadian bacon  
4 English muffins, split in half, toasted

DIRECTIONS

@codemillmatt

UI Demo

Controls

TableView

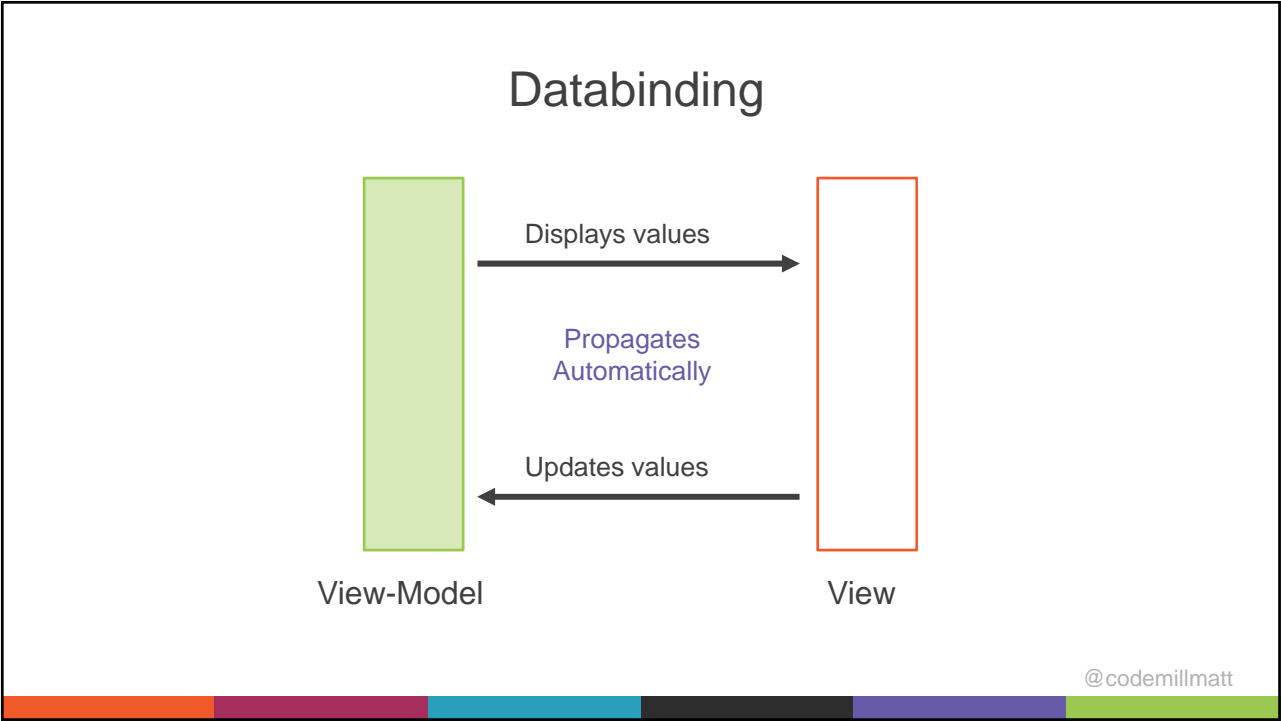
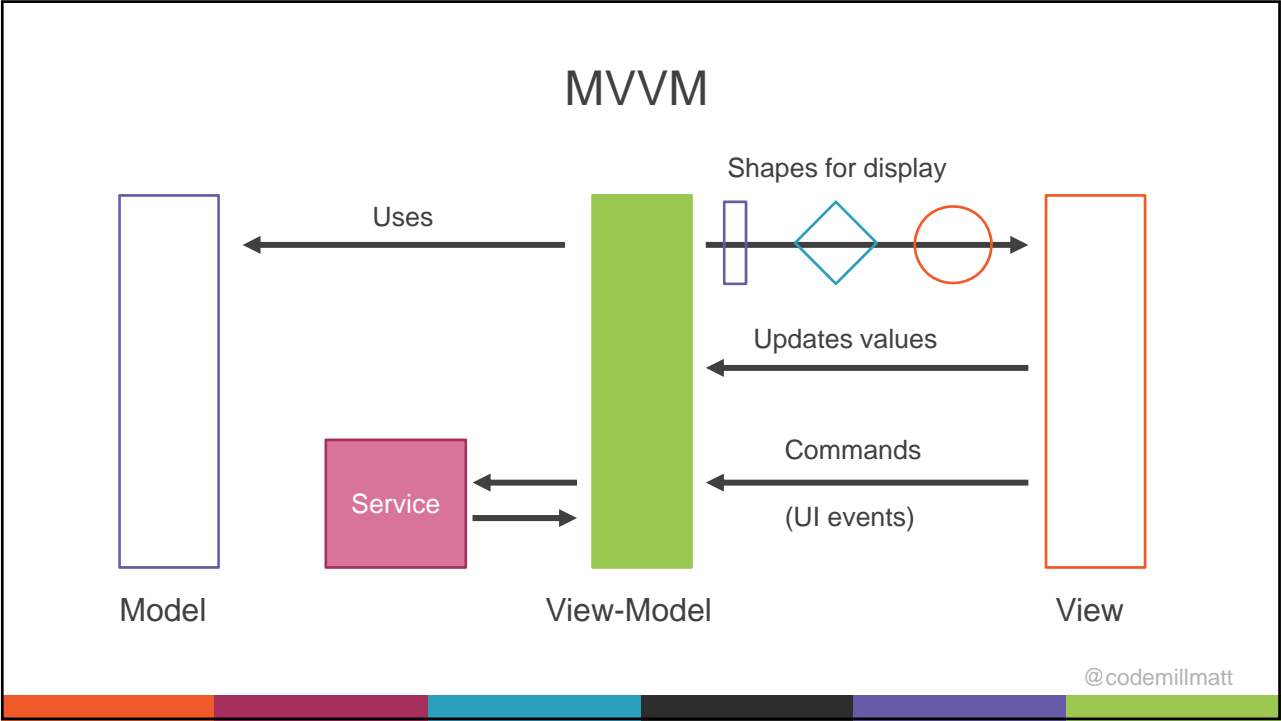
ListView



@codemillmatt

MVVM

@codemillmatt



## INotifyPropertyChanged

```
public class RecipeViewModel : INotifyPropertyChanged
{
    public event PropertyChangedEventHandler PropertyChanged;

    string _recipeName;
    public string RecipeName
    {
        get { return _recipeName; }
        set
        {
            if (_recipeName == value)
                return;

            _recipeName = value;
            OnPropertyChanged();
        }
    }

    public void OnPropertyChanged([CallerMemberName]string memberName = "") =>
        PropertyChanged?.Invoke(this, new PropertyChangedEventArgs(memberName));
}
```

@codemillmatt

```
public void OnPropertyChanged([CallerMemberName]string memberName = "") =>
    PropertyChanged?.Invoke(this, new PropertyChangedEventArgs(memberName));
```

Cuz it was at the bottom

@codemillmatt

```
<Label Text="{Binding RecipeName}" />
```

...

```
<Label Text="{Binding Time, StringFormat='{0} prep'}" />
```

## Data Binding - XAML

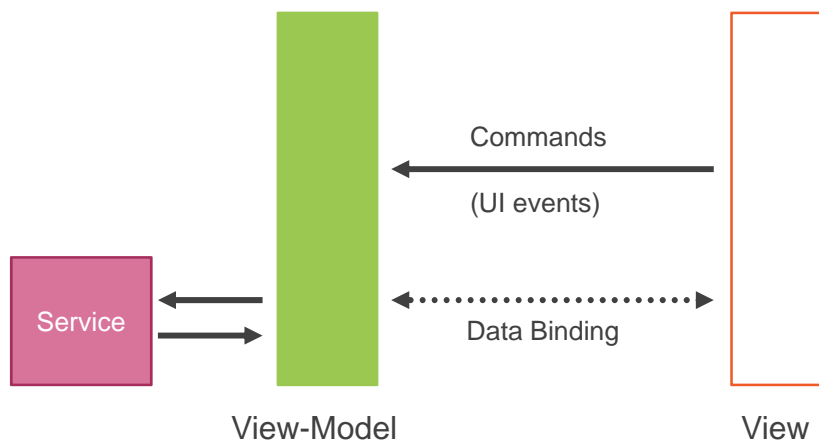
BindingContext set on parent or control

Binding keyword

Reference view model property

@codemillmatt

## Commanding



@codemillmatt

```
public ICommand AddRecipeCommand { get; } =  
    new Command(async (obj) => await Navigation.PushModalAsync(  
        new NavigationPage(new EditRecipePage())))  
    );
```

## Command – View Model

ICommand interface

Action when invoked

Optional bool function for enabling

@codemillmatt

```
<Button Text="Add" Command="{Binding AddRecipeCommand}" />
```

## Command - XAML

Command property

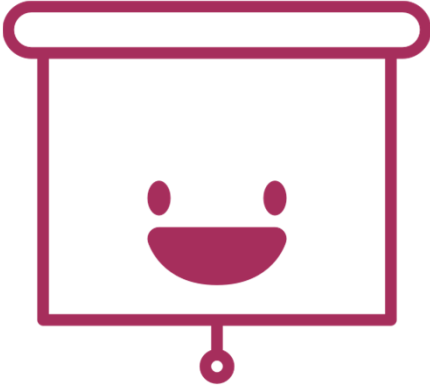
Binding keyword

One command per control


@codemillmatt

## MVVM Demo

Data binding  
Commanding




@codemillmatt



## Additional Features & Platform Access

---

@codemillmatt





## Dependency Service

Platform Specific Functionality in Shared Code

Define interface

Implement on each platform

Register with DependencyService

Shared code calls  
DependencyService

@codemillmatt

```
public interface IFormsCamera
{
    void LaunchCamera();
}
```

## Dependency Service

Define the interface

@codemillmatt

```
[assembly: Dependency(typeof(AppleCamera))]
```

```
namespace Foodie.iOS
```

```
{
```

```
    public class AppleCamera : IFormsCamera
```

```
    {
```

```
        public void LaunchCamera()
```

```
        {
```

```
            // Some camera stuff
```

```
        }
```

```
    }
```

```
}
```

## Dependency Service

Platform implementation

@codemillmatt

```
public void TakePhoto()
```

```
{
```

```
    var camera = DependencyService.Get<IFormsCamera>();
```

```
    camera.LaunchCamera();
```

```
}
```

## Dependency Service - Consume

DependencyService static object

@codemillmatt

### CSS Styling

Similar to web CSS

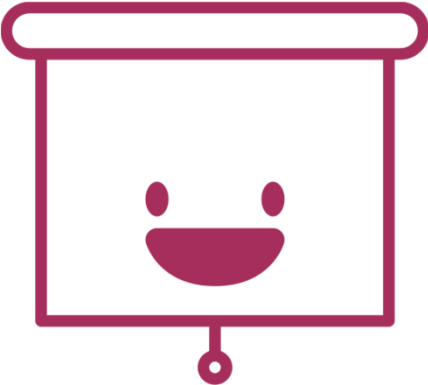
C# or XAML

Select by ID, Class, Type, Base, etc.

Not all Forms properties available... yet

@codemillmatt

CSS Demo



@codemillmatt

# Xamarin.Forms

## User Interface

- Pages
- Navigation
- Layouts
- Controls

## MVVM

- Data binding
- Commanding

## And More!

- Dependency Service
- Platform Access
- CSS

@codemillmatt

The diagram illustrates the components of Xamarin.Forms: C# (represented by a blue C and #), Xamarin.Forms (represented by an orange hexagon with 'X.F'), and mobile platforms (represented by Android, Apple, and Windows icons). These are combined with plus signs and an equals sign to result in a stick figure representing the user.

@codemillmatt

T02 - Creating Four Beautiful Apps At Once with Xamarin.Forms - Matthew Soucoup

## Learn More!

---

Pluralsight: Moving Beyond the Basics with Xamarin.Forms

<https://www.pluralsight.com/courses/xamarin-forms-moving-beyond-basics>

## Where to Find Stuff

### The Talk

Code & Slides:

[github.com/codemillmatt/four-beautiful-apps](https://github.com/codemillmatt/four-beautiful-apps)

Matt Soucoup – Sr. Cloud Developer Advocate @ Microsoft

I ❤️ You

[@codemillmatt](https://twitter.com/codemillmatt)

[codemillmatt@microsoft.com](mailto:codemillmatt@microsoft.com)

[codemillmatt.com](https://codemillmatt.com)

@codemillmatt



Visual Studio **LIVE!** | San Diego  
EXPERT SOLUTIONS FOR .NET DEVELOPERS

# Creating Four Beautiful Apps at Once

An Intro to Xamarin.Forms

**Matthew Soucoup**  
Sr Cloud Developer Advocate  
Microsoft

Level: Beginner

 Code Again for the First Time! 

 Visual Studio 25  
25 YEARS OF CREATING INNOVATION