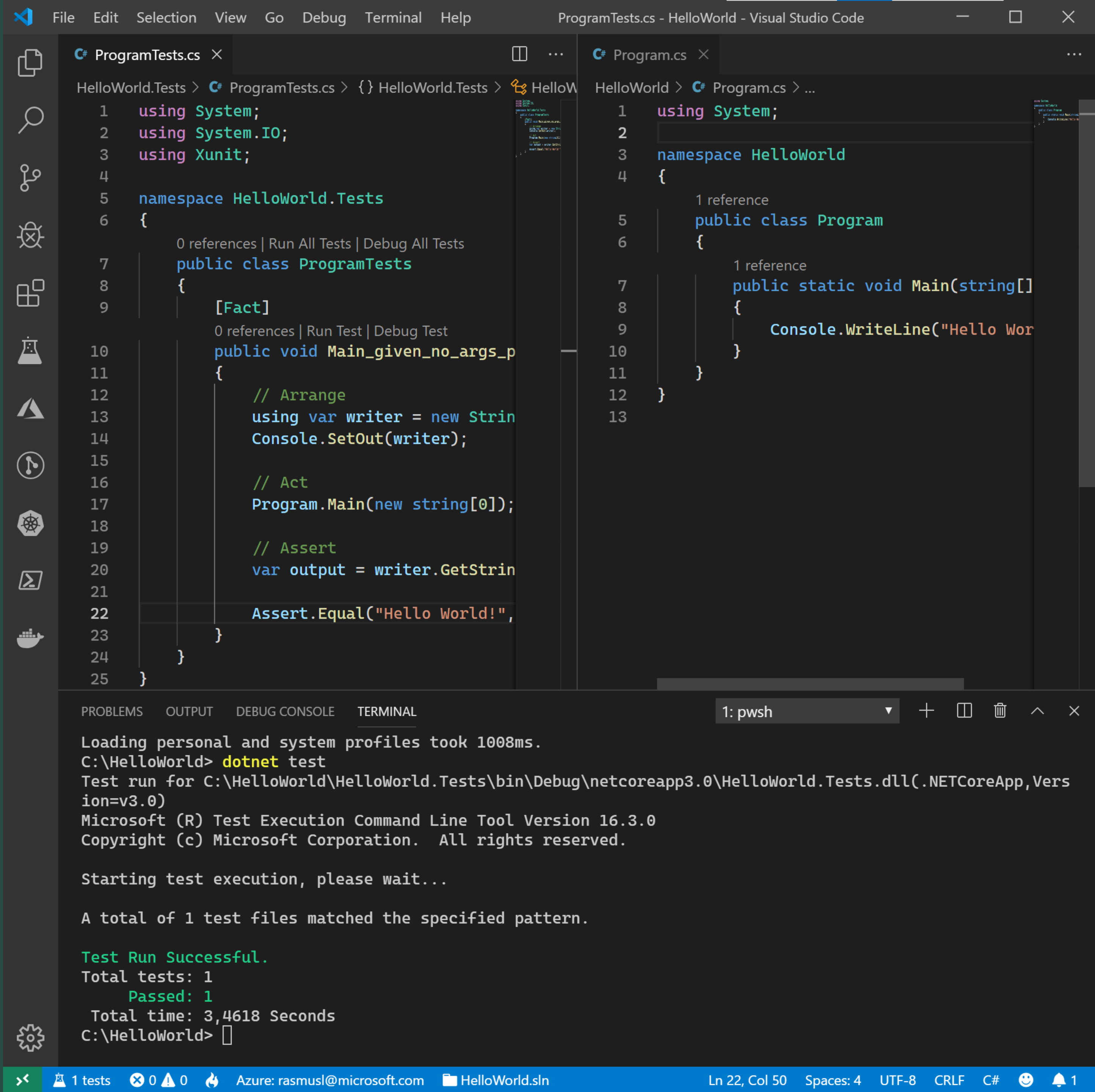
Apps and XAML, UWP and Xamarin.Forms

Rasmus Lystrøm
Associate Professor
ITU
rnie@itu.dk


File Edit Selection View Go Debug Terminal Help
ProgramTests.cs - HelloWorld - Visual Studio Code
ProgramTests.cs
Program.cs
HelloWorld.Tests > ProgramTests.cs > {} HelloWorld.Tests > HelloW
HelloWorld > Program.cs > ...
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
1: pwsh
1 tests
Azure: rasmusl@microsoft.com
HelloWorld.sln
Ln 22, Col 50 Spaces: 4 UTF-8 CRLF C#

```
using System;
using System.IO;
using Xunit;

namespace HelloWorld.Tests
{
    0 references | Run All Tests | Debug All Tests
    public class ProgramTests
    {
        [Fact]
        0 references | Run Test | Debug Test
        public void Main_given_no_args_p
        {
            // Arrange
            using var writer = new Strin
            Console.SetOut(writer);

            // Act
            Program.Main(new string[0]);

            // Assert
            var output = writer.GetStrin

            Assert.Equal("Hello World!",
        }
    }
}
```

```
using System;

namespace HelloWorld
{
    1 reference
    public class Program
    {
        1 reference
        public static void Main(string[]
        {
            Console.WriteLine("Hello Wor
        }
    }
}
```

```
Loading personal and system profiles took 1008ms.
C:\HelloWorld> dotnet test
Test run for C:\HelloWorld\HelloWorld.Tests\bin\Debug\netcoreapp3.0\HelloWorld.Tests.dll(.NETCoreApp,Vers
ion=v3.0)
Microsoft (R) Test Execution Command Line Tool Version 16.3.0
Copyright (c) Microsoft Corporation.  All rights reserved.

Starting test execution, please wait...

A total of 1 test files matched the specified pattern.

Test Run Successful.
Total tests: 1
     Passed: 1
 Total time: 3,4618 Seconds
C:\HelloWorld>
```

Agenda

UI Frameworks for

XAML

Universal Windows Platform (UWP)

Xamarin.Forms

MVVM

UI Frameworks for C♯

UI Frameworks for C♯

Windows Forms

Windows Presentation Foundation

Universal Windows Platform

Xamarin.Forms

Blazor

Universal Windows Platform vs. Xamarin.Forms

UWP

Native Windows 10

HoloLens

Surface Hub

Surface Pro X

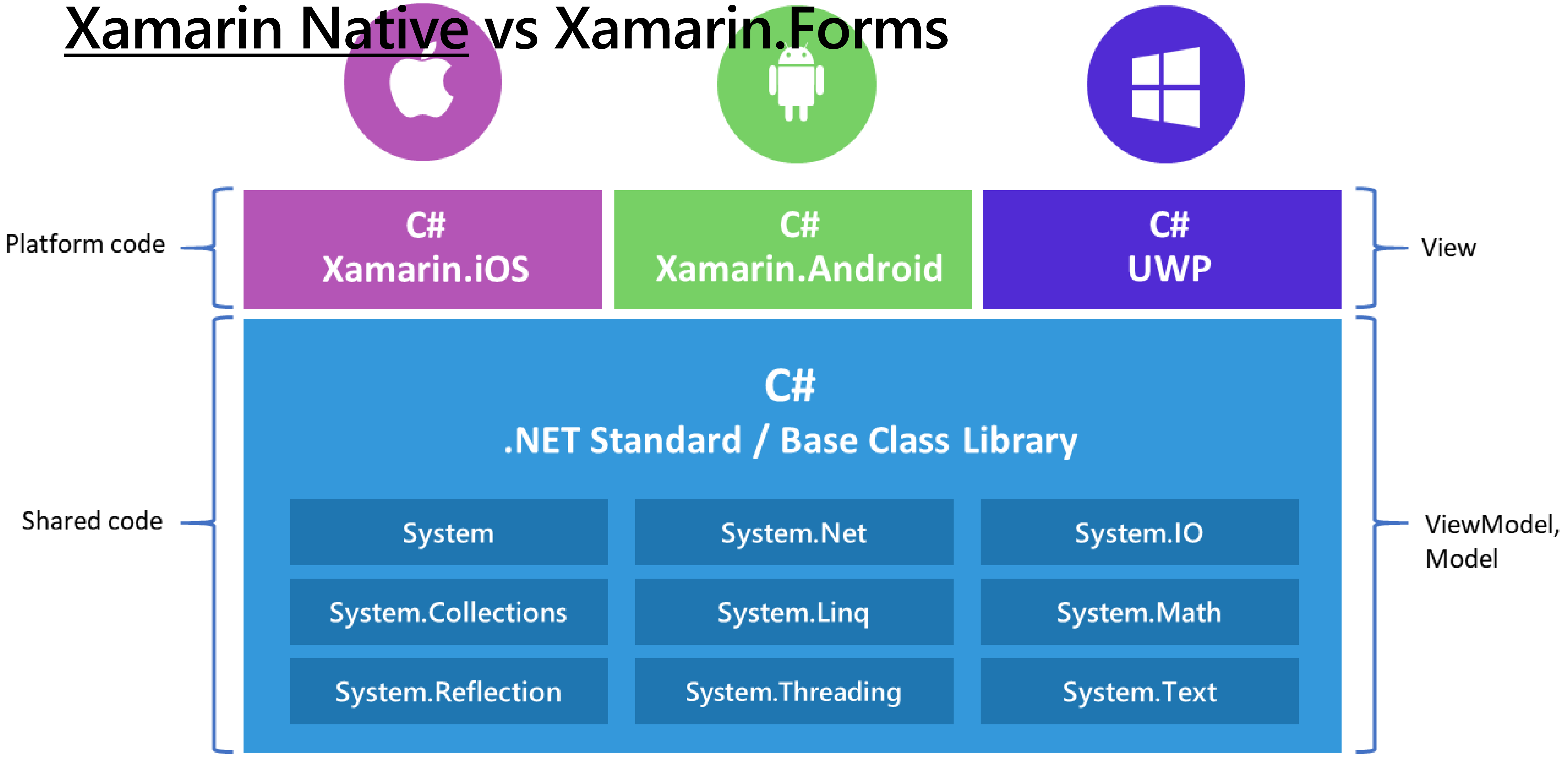
Xamarin.Forms

iOS

Android

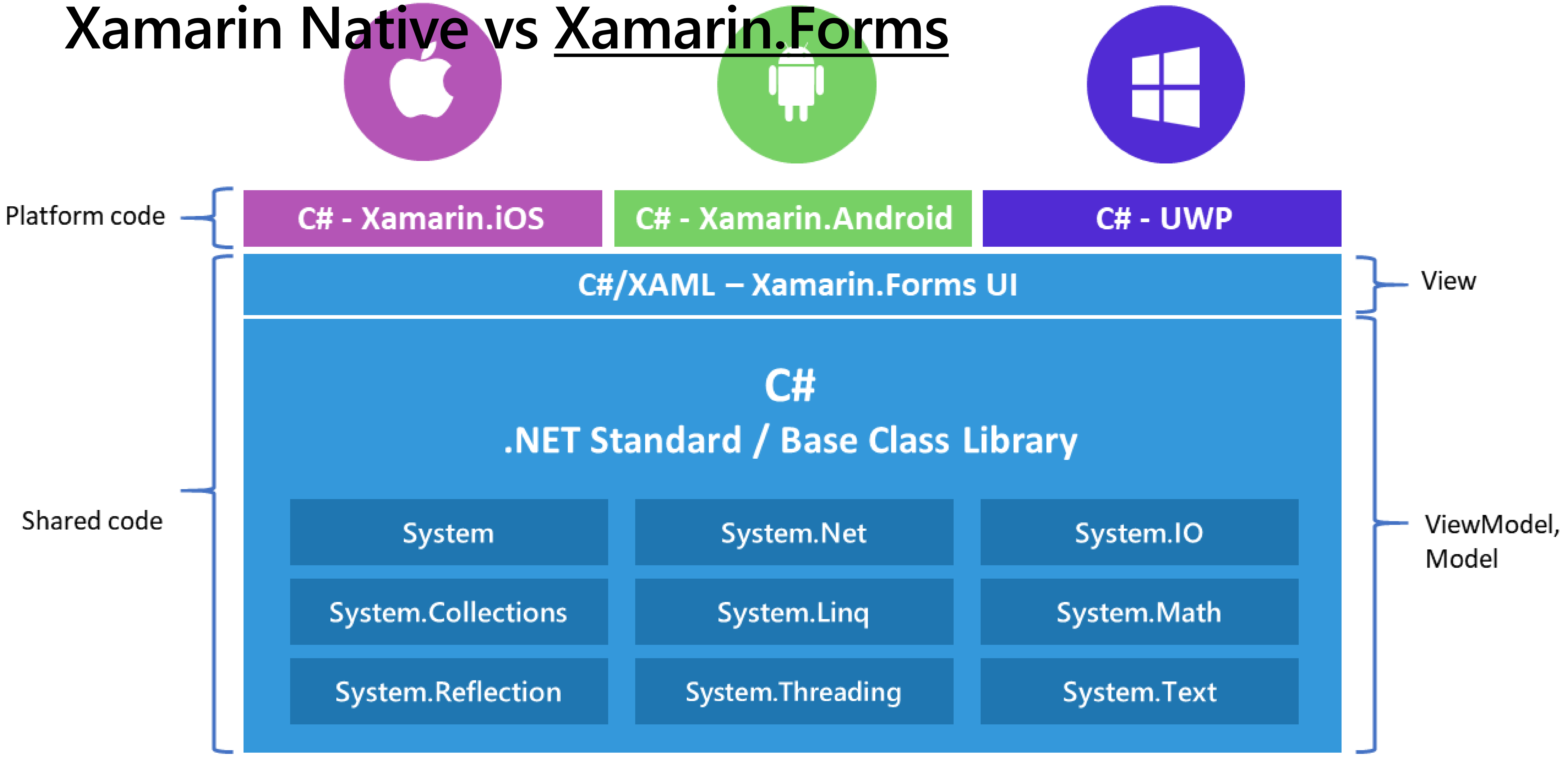
Windows

Xamarin Native vs Xamarin.Forms



Source: https://www.codemag.com/article/1911092

Xamarin Native vs Xamarin.Forms



Source: https://www.codemag.com/article/1911092

XAML

XAML = eXtensible Application Markup Language

Windows Desktop (WPF)

Windows Universal (anything)

Xamarin.Forms (iOS, Android, Windows)

Silverlight (web)

XAML

Markup language for declaratively designing and creating application UIs

XAML maps XML markup to objects in the .NET Framework

Every tag maps to a class and every attribute to a property

Markup and procedural code are peers in functionality and performance

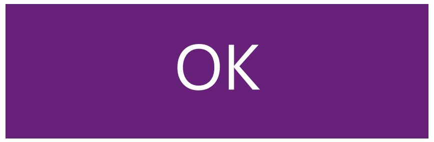
Code and markup are both first class citizens

Consistent model between UI, documents, and media

Compiled to code

XAML Markup vs. Code

```
<Button Width="100">OK
    <Button.Background>
        Purple
    </Button.Background>
</Button>
```



```
var button = new Button();
button.Content = "OK";
button.Background = new SolidColorBrush(Colors.Purple);
button.Width = 100;
```

MainPage.xaml

```xml
<Page>
    <Grid>
        <StackPanel>
            <Ellipse Name="Light" Fill="Red"
                     Height="200" Width="200" Margin="50" />
            <Button Width="150"
                    Content="Change Lights"
                    HorizontalAlignment="Center"
                    Click="Button_Click" />
        </StackPanel>
    </Grid>
</Page>
```

MainPage.xaml.cs

```csharp
namespace App
{
    public sealed partial class MainPage : Page
    {
        public MainPage()
        {
            this.InitializeComponent();
        }

        private void Button_Click(object sender, RoutedEventArgs e)
        {
            var current = Light.Fill as SolidColorBrush;

            if (current.Color == Colors.Red)
                Light.Fill = new SolidColorBrush(Colors.Green);
            else
                Light.Fill = new SolidColorBrush(Colors.Red);
        }
    }
}
```

Xamarin.Forms

Demo

Image source: http://lazergaze.tumblr.com/post/26333564955

MVVM

The Model-View-ViewModel Pattern

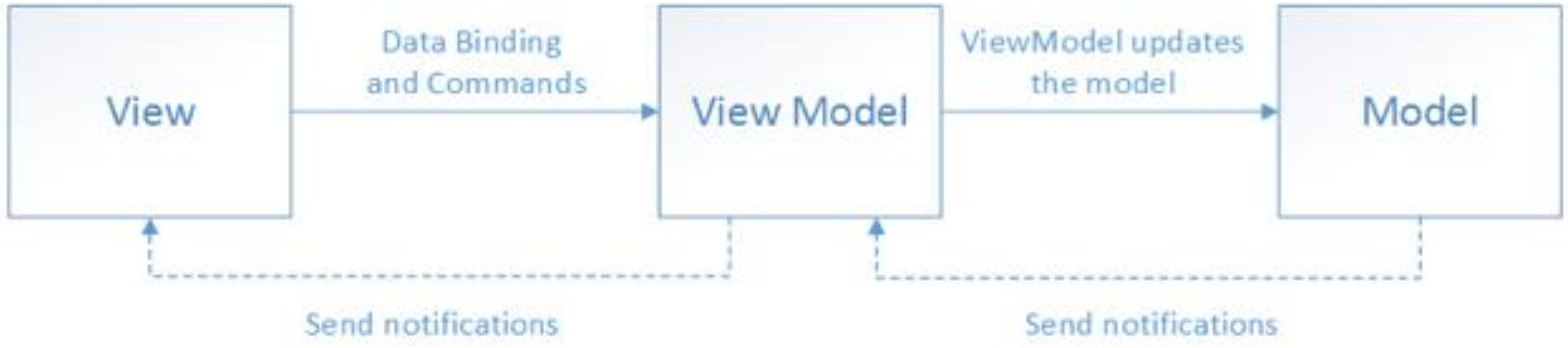
Separation of logic and presentation

Having event handlers in the code-behind is bad for testing, since you cannot mock away the view

Changing the design of the view often also requires changes in the code, since every element has its different event handlers

The logic is tightly bound to the view. It's not possible to reuse the logic in an other view

MVVM


View
Data Binding and Commands
View Model
ViewModel updates the model
Model
Send notifications
Send notifications

MVVM

Demo

MVVM concepts

There is conceptually only ever one MODEL

Code in code-behind should be ABSOLUTELY MINIMAL

A ViewModel should ALWAYS implement `INotifyPropertyChanged`

A ViewModel may be used for more than one view

MVVM Design Patterns

Observer Pattern:

- `INotifyPropertyChanged`
- `ObservableCollection<T>`
- `MessagingCenter`

Command Pattern:

- `ICommand`

Xamarin.Forms / UWP gotchas

- Mobile app must be set to *Build* and *Deploy* in *solution configuration*
- If API and mobile app in same project: Use multiple startup projects
- Source in C:\git or similar
- HTTPS not possible for *localhost*:
  - Manifest: `<application android:usesCleartextTraffic="true"></application>`
- Move `app.UseHttpsRedirection();` to *prod*.
- Run Web API with *kestrel*.
- Don’t `try/catch` until you know what errors you want to handle!
- Enable XAML Hot Reload

MVV

Don’t

MVVM

Templ

Image source: https://dirtyhands.wordpress.com