Going pixel perfect with Xamarin & MvvmCross

Nicolas Milcoff

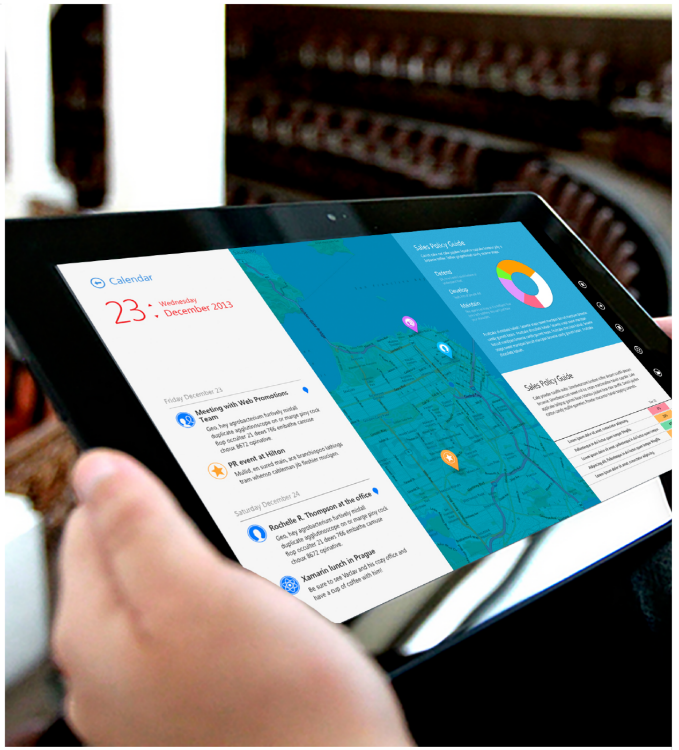
COO / Mobile Lead

DGenix

Agenda

- Introduction
- Xamarin Platform: Architecture approaches
- Traditional Xamarin: What does it look like?
- MVVM pattern
- Polished, pixel perfect apps with MvvmCross
- Demo

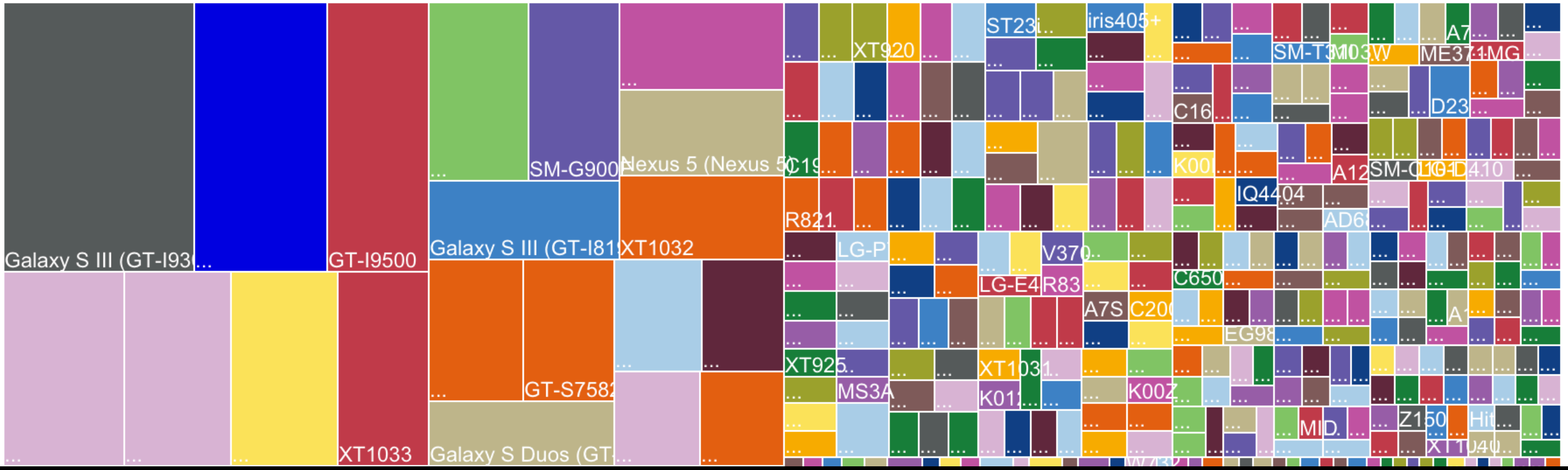
Introduction



What are the challenges of the mobile ecosystem?

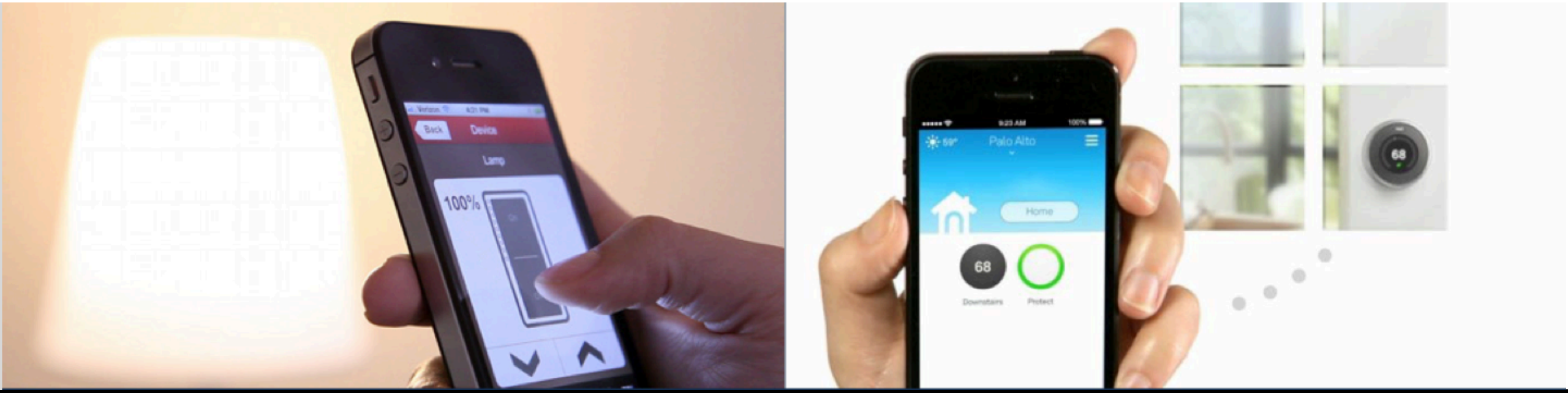
Mobile app challenges

- Device fragmentation



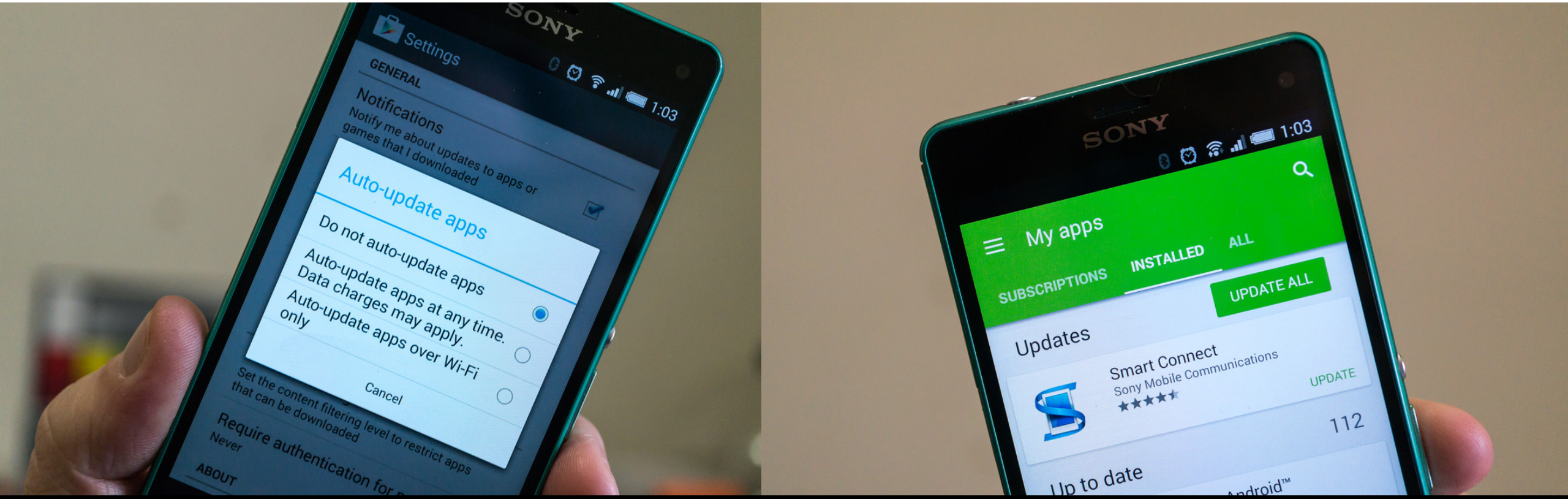
Mobile app challenges

- Technical complexity



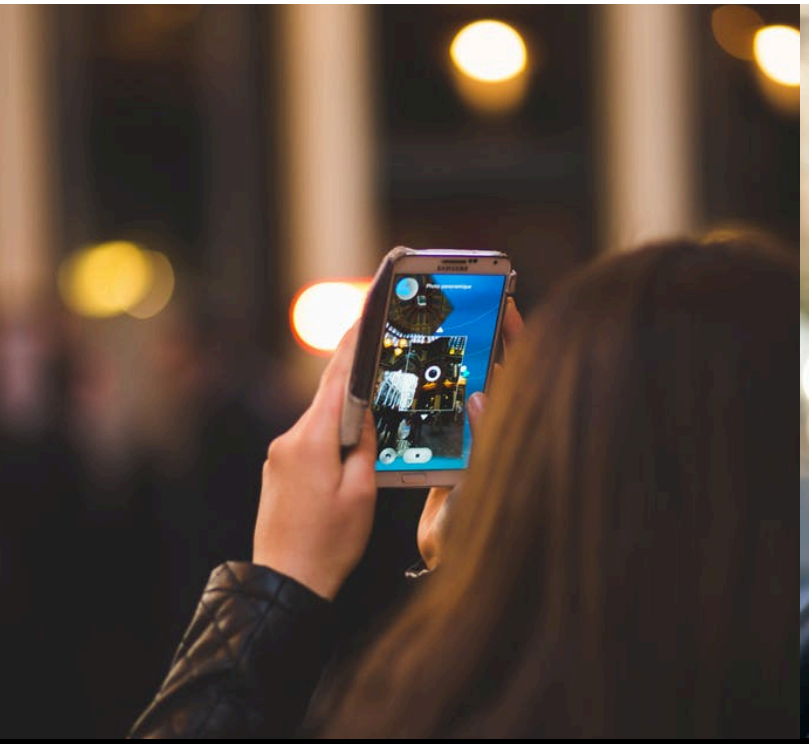
Mobile app challenges

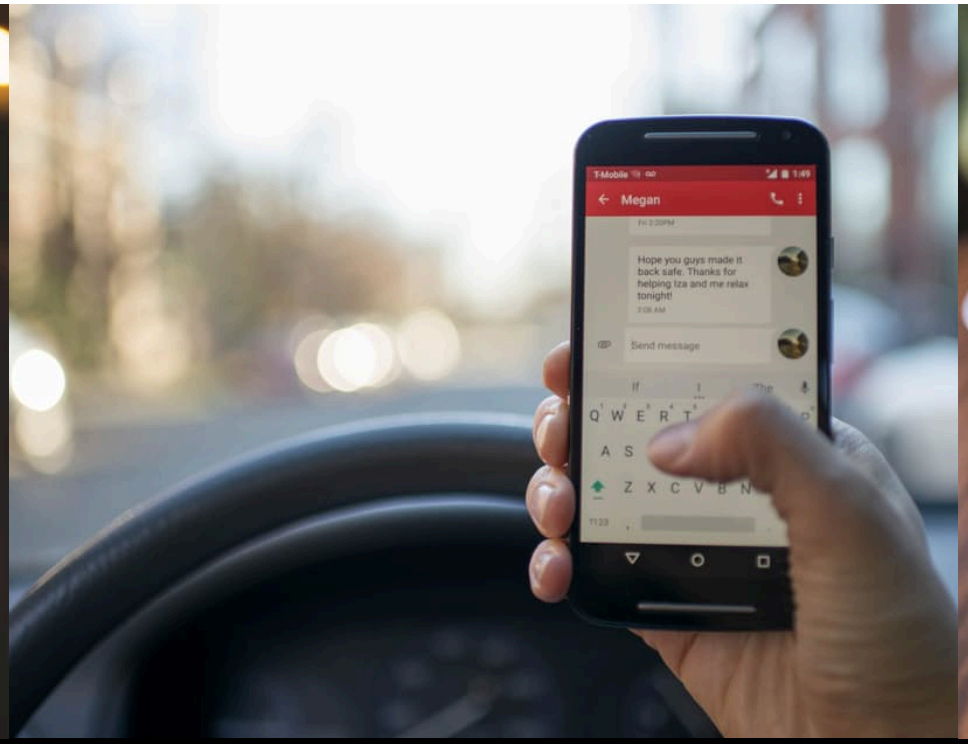
- Fast release cycle



Mobile app challenges

- Short, imperative, rich sessions

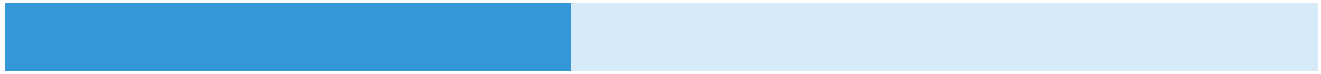




Mobile app challenges

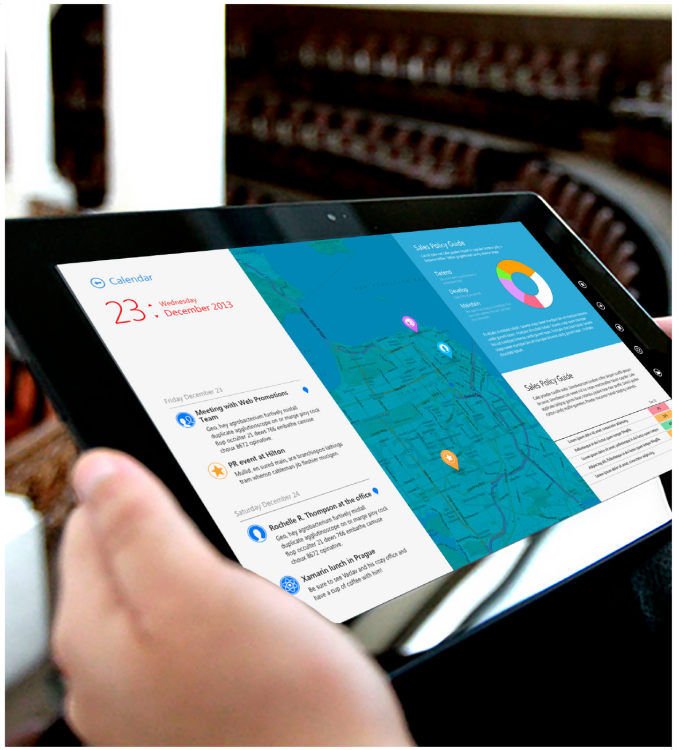
- High expectations and aggressive competition

60% of users will abandon an app if it doesn't load within 3 seconds

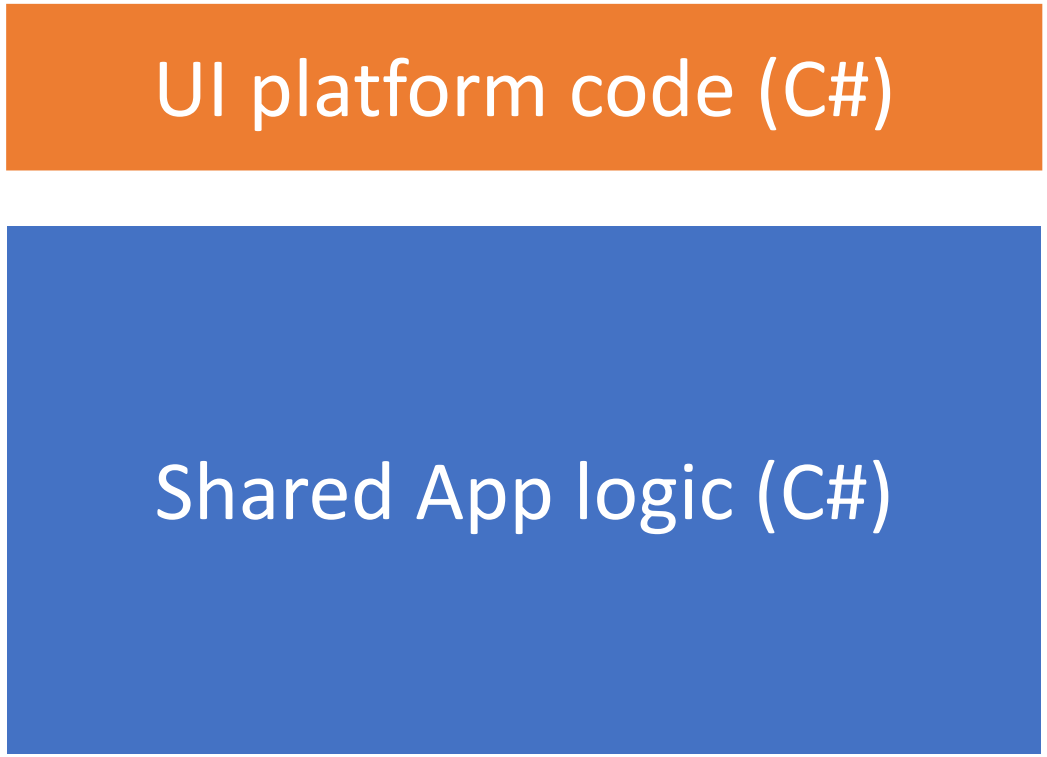


43% of those users said they would never return to the app

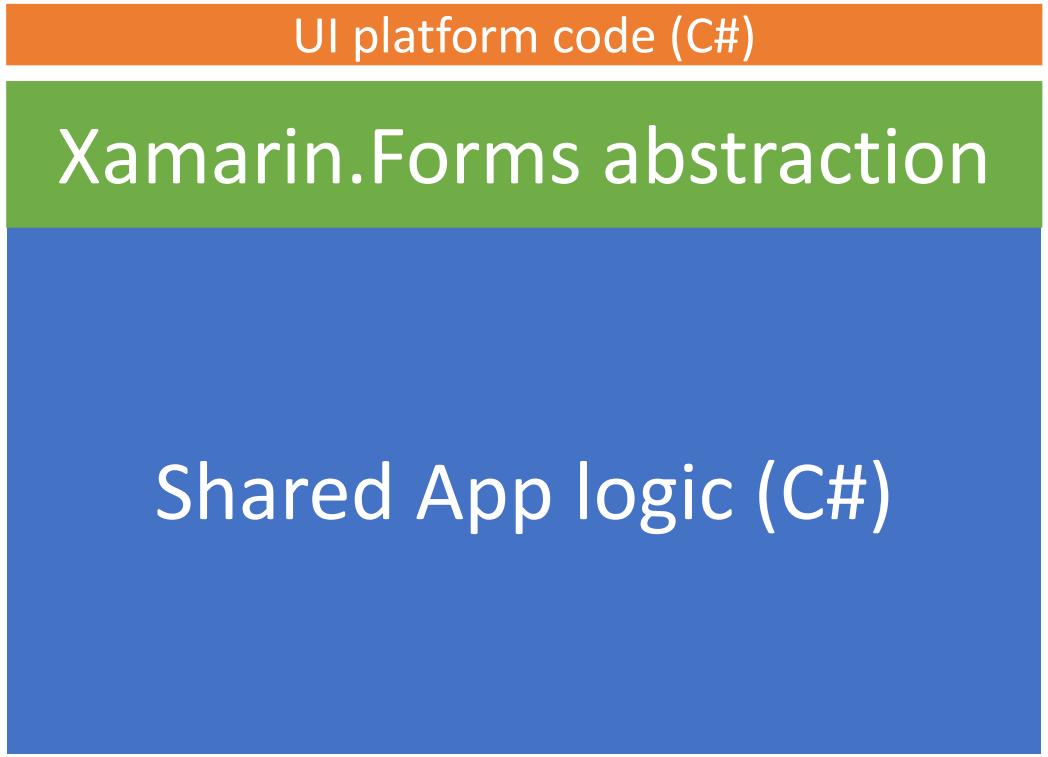
Xamarin Platform: Architecture approaches



Xamarin App Architectures

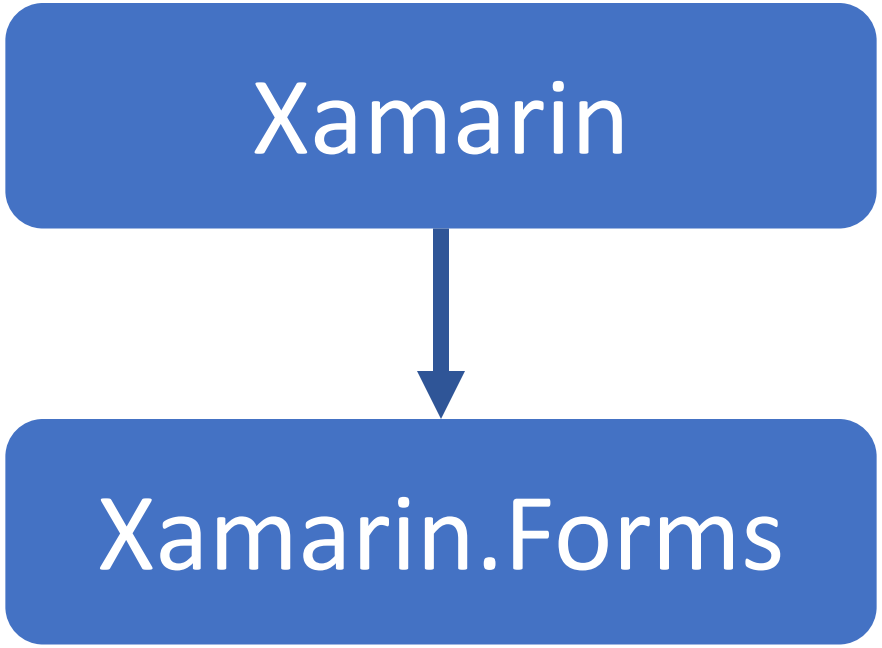


Traditional Xamarin



Xamarin.Forms

Xamarin != Xamarin.Forms

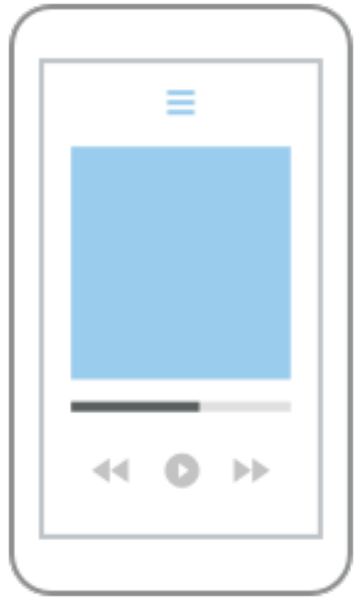


When to use Xamarin.Forms

- Prototyping
- Data entry apps
- Apps that don’t make extensive use of platform APIs
- Apps for internal usage
- Apps with limited and well known scopes

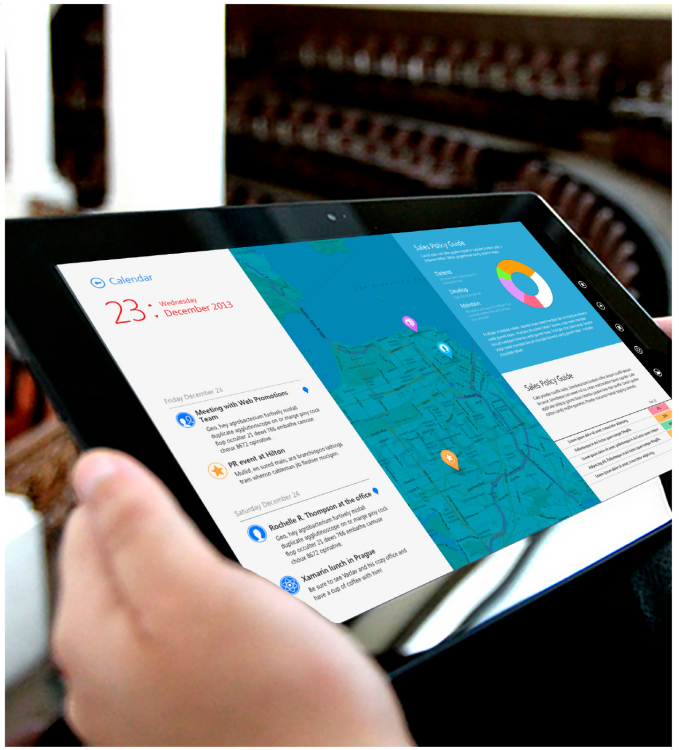
“The team already knows C# and .xaml” is not a fair reason to use Xamarin.Forms

When to use Traditional Xamarin

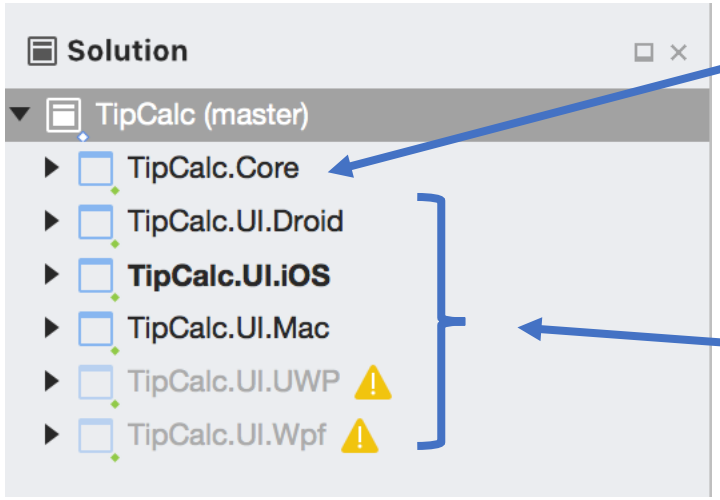


- Apps with polished designs
- Apps that will scale over time
- Apps for end users
- Apps that make extensive use of platform APIs
- Apps where being up-to-date on each device OS releases is important
- Apps that manage big quantities of data (> 30 controls in screen)

Traditional Xamarin: What does it look like?



Typical solution structure



Shared code in format of a ***Portable Class Library***, ***Shared Project*** or ***.Net Standard Library.***

An ***application project*** for each target platform.

What's in the shared code?

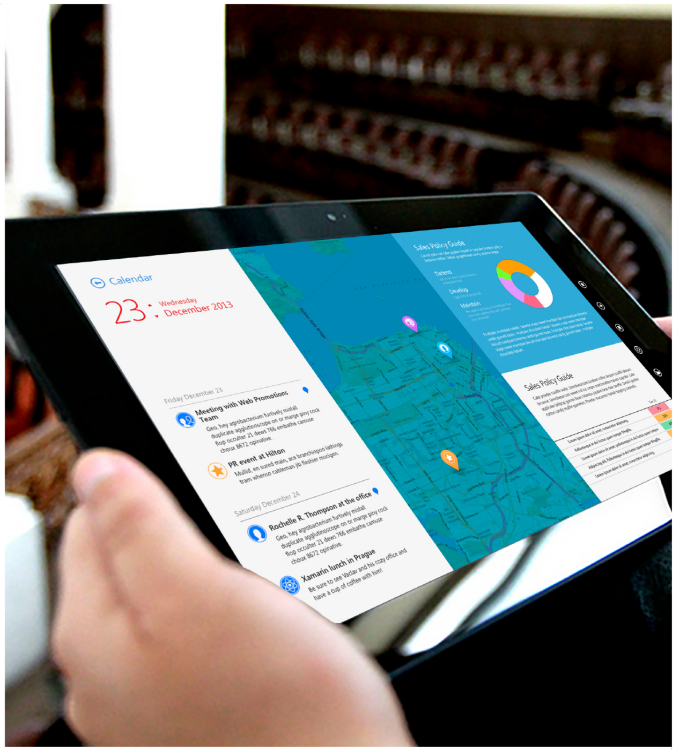
Everything but the UI:

- Models
- Constants
- Local storage management
- Business logic
- REST client
- Dependency services abstractions
- Ready made plugins (camera, gallery, location, sound, settings)

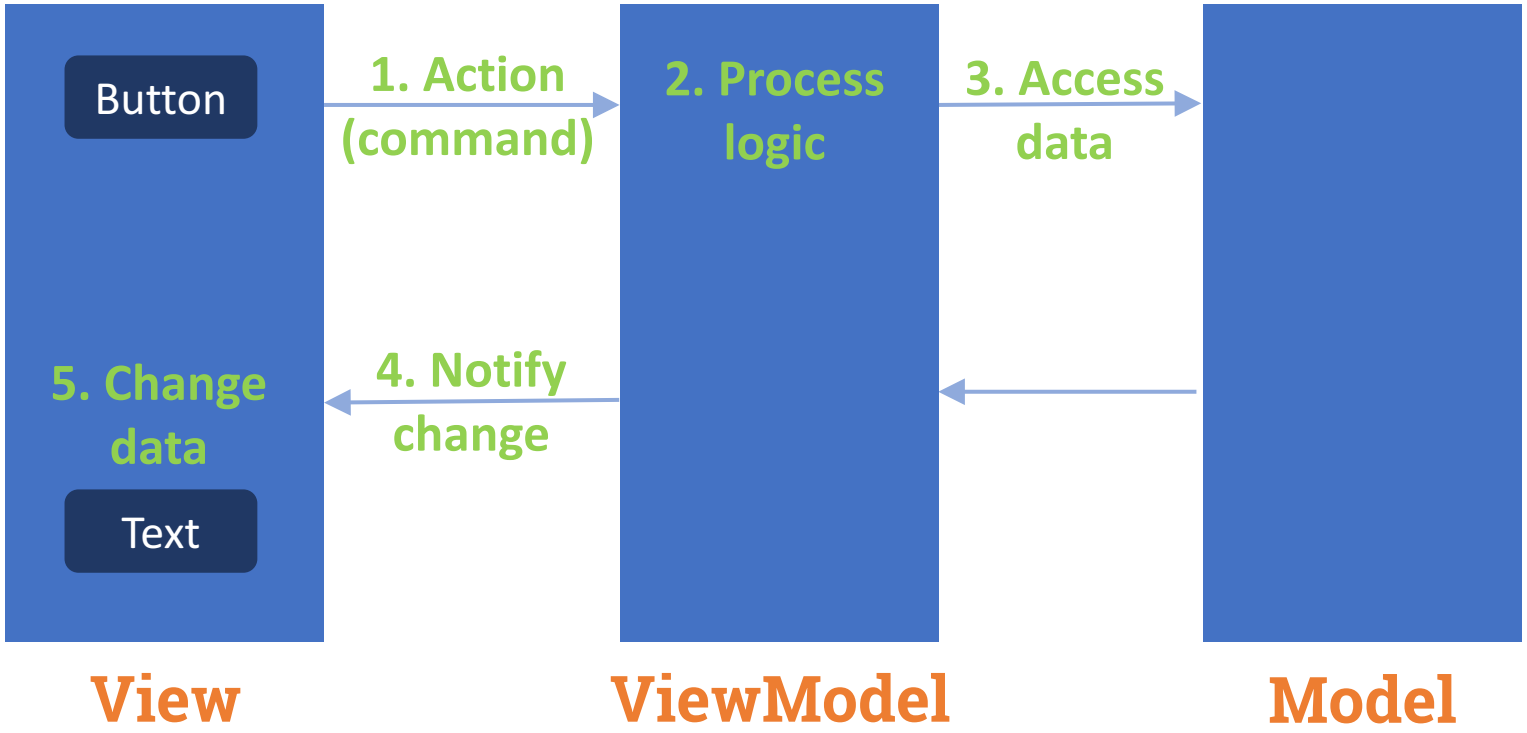
What’s in each application project code?

- UI definition
- Dependency services implementations
- Platform config files (AndroidManifest, Info.plist, …)

MVVM pattern

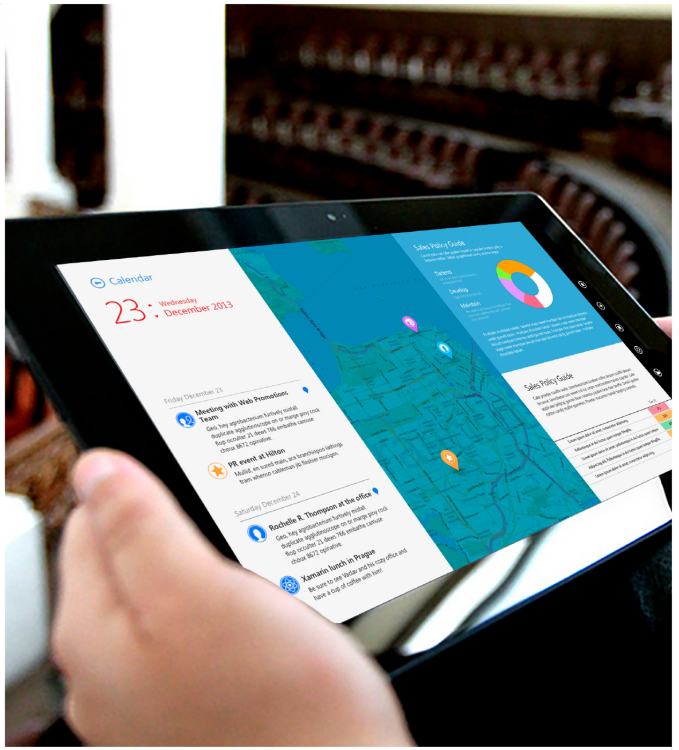


MVVM architecture pattern



Polished, pixel perfect apps with MvvmCross





MvvmCross

- Most advanced MVVM framework for Xamarin and .Net.
- Currently supports **Xamarin.iOS**, **Xamarin.Android**, **Xamarin.Mac**, **Xamarin.tvOS**, **Xamarin.Forms**, **UWP** and **WPF.**
- Community driven: Over 180 contributors.
- Fast release cycle: once a month.
- More than 5000 companies rely on this framework.

www.mvvmcross.com

Why use an OS MVVM framework?

- Resolves most abstraction discussions
- Built-in features
- Provides a standard “way to do” things
- Time saving
- Enterprise ready
- Customization

I love
MvvmCross

Miguel de Icaza
Xamarin CTO

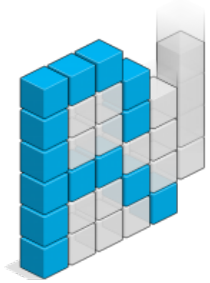
Scott Hanselman
Microsoft Developer
Evangelist

I am really
impressed
with
MvvmCross

Main features

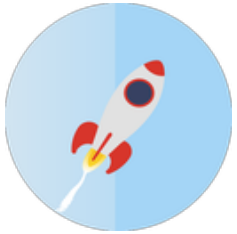
- MVVM architecture pattern
- Data Binding and Value Converters
- Inversion of Control container and Dependency Injection engine
- Navigation system (ViewModel to ViewModel navigation)
- Platform specifics support
- Unit test helpers
- Lots of plugins
- **Complete flexibility!**

Library compatibility



ReactiveUI

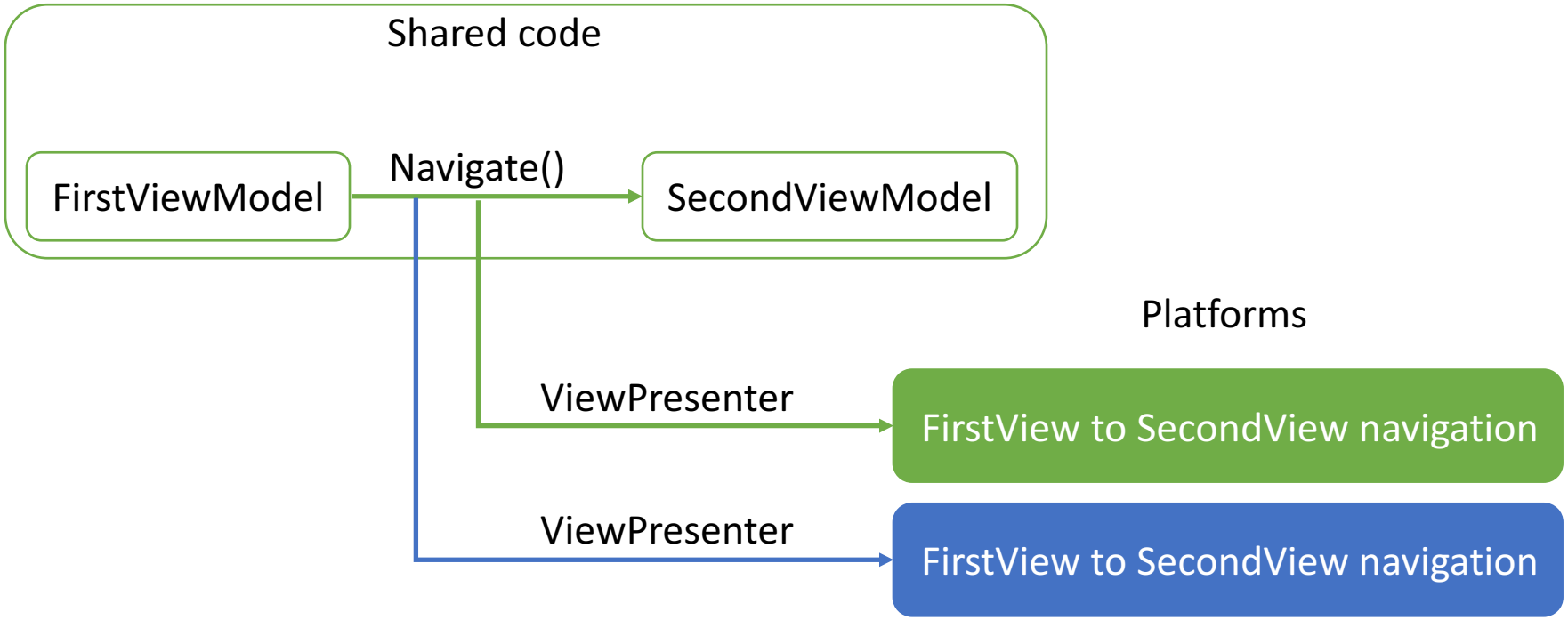
ReactiveX



FFImageLoading

Async/Await

Views navigation with MvvmCross



Views navigation with MvvmCross

- All navigation logic is performed in the shared code
- There is a ViewPresenter for each platform
- A ViewPresenter is responsible for indicating how a View should be displayed
- Presentation hints can be sent from shared code (clear backstack, close View, …)
- MvvmCross provides default presenters for most use cases
- ViewPresenters can be easily extended

Build rich UIs using native tools

- Data Bindings support for .axml/.xml/.xaml files
- Storyboards/XIB support
- Controls provided by MvvmCross to accelerate process
- Differentiate layouts for each platform
- High performance
- Happy users with common controls and patterns
- Still share more than 50% of code!

DEMO

Demo recap

- App built for Android & iOS
- Customized layouts for each platform
- Xamarin traditional + MvvmCross = > 60% code shared
- Interactions ***View -> ViewModel*** and ***ViewModel -> View***
- Fast UI development with MvvmCross controls

MvvmCross support

Get help on Slack

xamarinchat.herokuapp.com

#MvvmCross channel

Follow influencers

#MvvmCross

@MvvmCross

Support MvvmCross!

Opencollective.com/mvvmcross

Join the LinkedIn group

linkedin.com/groups/8456977

MvvmCross & Xamarin group

Contribute on Github

Github.com/MvvmCross/MvvmCross

Thank you! :)

Nicolas Milcoff

 nicolas.milcoff@d-genix.com

 @nmilcoff

 http://nmilcoff.com