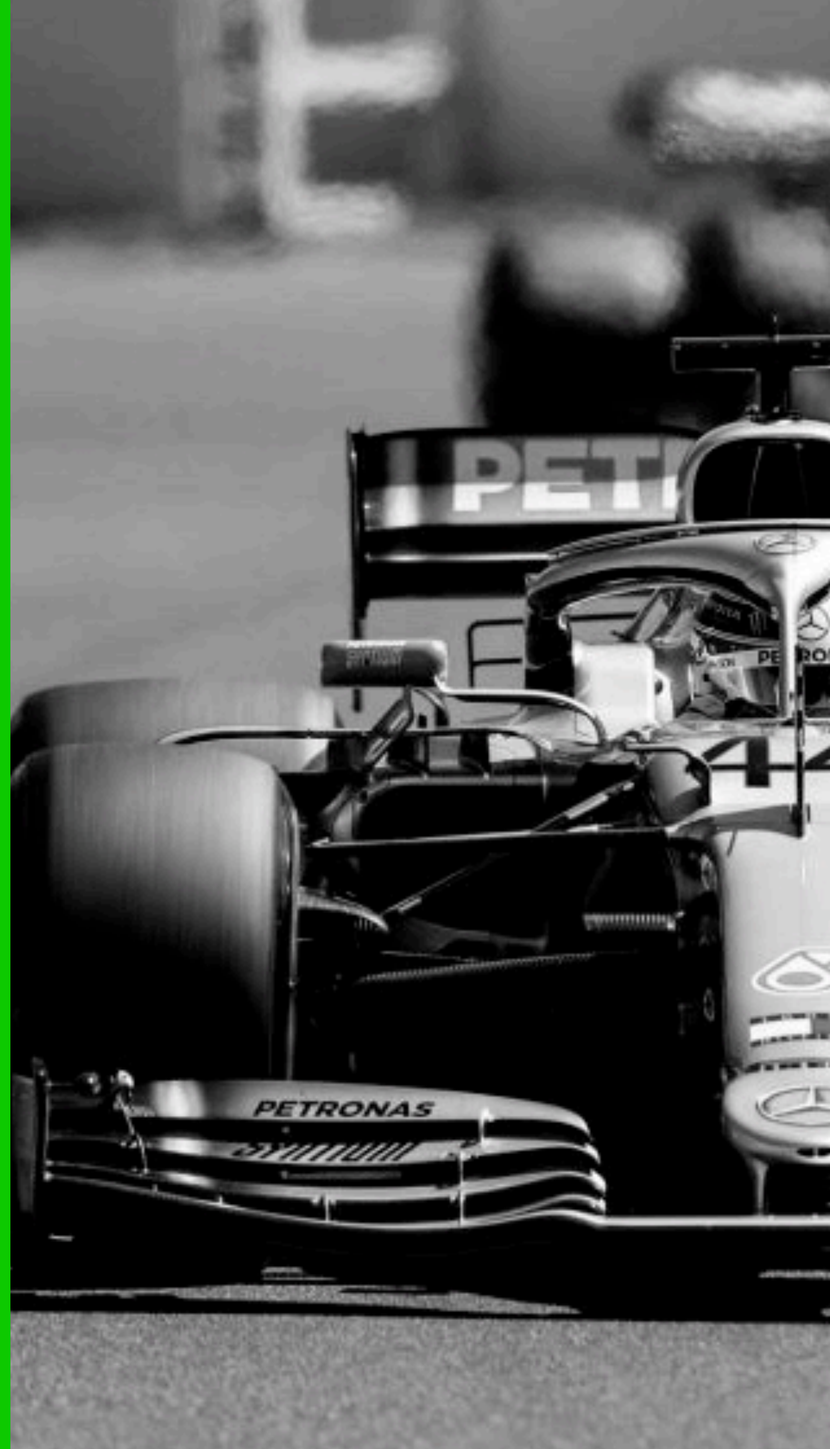
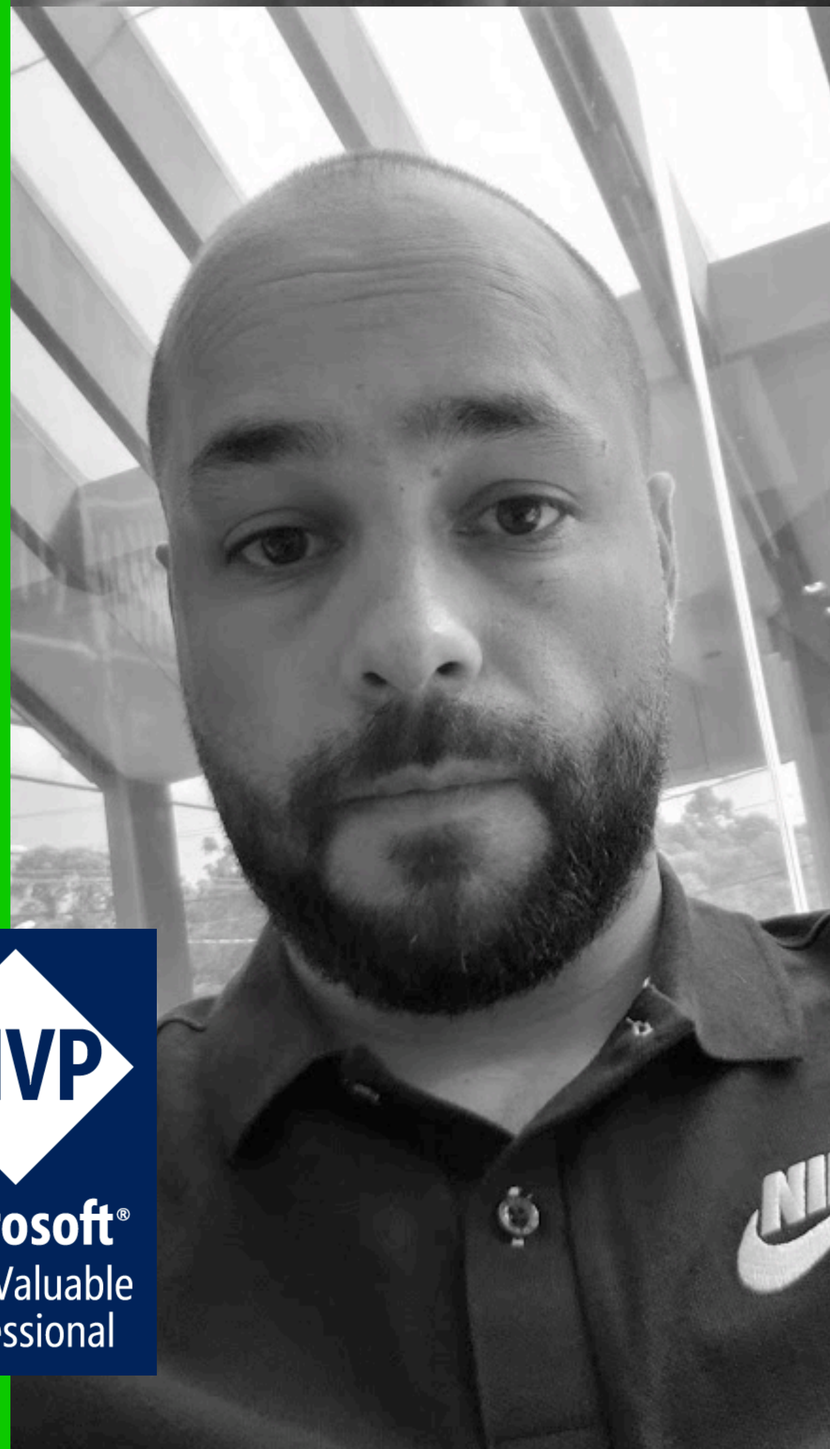
Xamarin Forms Performance Booster

Why you need to go beyond basics?



William S Rodriguez

Master Software Engineer @ ArcTouch



MVP
Microsoft®
Most Valuable Professional

🦤 *https://twitter.com/willbuildapps*

What do you want to achieve?

- 👉🏼 Speed;
- 👉🏼 Responsiveness;
- 👉🏼 60 fps;

First things first

"Xamarin.Forms is Much More Capable Than You Think”

First things first

Xamarin != Xamarin.Forms

First things first

We have hair!
Be bald is more practical!

https://bigthink.com/scotty-hendricks/article-on-the-benefits-of-baldness

First things first

“Update to Xamarin.Forms 3.6”

Emoji Awesome Classification Tool

Fantastic ❤️

Well… you know!

Please DON'T

General

Don’t wait for data to load, 
before you load your UI.

General

Keep number of dependencies or assemblies to a minimum 

General

Push anything you can to a background thread, if its not explicitly needed for app startup.

General

Lazy load anything and everything you can. 

General

Reduce the Size of the  Application

** Link All Assemblies*

General

Enable XAML compilation 🥓

** if you're using XAML*

```
[assembly: XamlCompilation(XamlCompilationOptions.Compile)]
```

General

Use Fast Renderers🥓

** only applicable to the app compat Android backend,*

```
Forms.SetFlags("FastRenderers_Experimental");
```

General

Enable Layout Compression

** not suitable for layouts that have a visual appearance, or that obtain touch input and ++*

```
<StackLayout CompressedLayout.IsHeadless="true"> ... </StackLayout>
```

General

Unnecessary Bindings 💩🤮

** here is no advantage in binding data that doesn't need to be bound.*

```
Button.Text = "Accept"
```

General

Don’t assign default values 💩🤮

** You just need to provide a default property value that's different from the default for the type of the property.*

General

Use async/await 🥓🥓🥓

** avoid blocking the main thread.*

General

Release IDisposable Resources 🥓

👇🏼

Release IDisposable Resources

```
…
using (StreamReader reader = new
StreamReader (filename))
{ text = reader.ReadToEnd (); }
…
```

```
…
… StreamReader reader = null;
try { reader = new StreamReader
(filename); text = reader.ReadToEnd
(); } finally { if (reader != null)
{ reader.Dispose (); } } …
…
```

General

Unsubscribe from Events 🥓

** prevent memory leaks*

Layout

Don’t use a StackLayout to host a single child

** A Layout that's capable of displaying multiple children, but that only has a single child, is wasteful.*

Don’t use a StackLayout to host a single child

```
…
<ContentPage.Content>
  <StackLayout>
    <Image Source="waterfront.jpg" />
  </StackLayout>
</ContentPage.Content>
…
```

```
…
<ContentPage.Content>
  <Image Source="waterfront.jpg" />
</ContentPage.Content>
…
```

Layout

Reduce the Visual Tree Size 🥓

** Reducing the number of elements on a page will make the page render faster*

👇🏼

Reduce the Visual Tree Size

```
…
<ContentPage.Content>
    <StackLayout>
        <StackLayout Padding="20,20,0,0">
            <Label Text="Hello" />
        </StackLayout>
        <StackLayout Padding="20,20,0,0">
            <Label Text="Welcome to the App!" />
        </StackLayout>
        <StackLayout Padding="20,20,0,0">
            <Label Text="Downloading Data..." />
        </StackLayout>
    </StackLayout>
</ContentPage.Content>
…
```

```
…
<ContentPage.Content>
  <StackLayout Padding="20,20,0,0"
Spacing="25">
    <Label Text="Hello" />
    <Label Text="Welcome to the App!" />
    <Label Text="Downloading Data..." />
  </StackLayout>
</ContentPage.Content>
…
```

Layout

Use a Grid when a StackLayout is not enough

** Don't attempt to reproduce a **Grid** layout by using a combination of **StackLayout** instances*

Use a Grid when a StackLayout is enough

```
…
<StackLayout>
  <StackLayout Orientation=“Horizontal">
     <Label Text="Name:" />
     <Entry Placeholder="Enter your name" />
  </StackLayout>
  <StackLayout Orientation="Horizontal">
    <Label Text="Age:" />
    <Entry Placeholder="Enter your age" />
  </StackLayout>
</StackLayout>
…
```

```
...
<Grid>
    <Grid.ColumnDefinitions>
        <ColumnDefinition Width="100" />
        <ColumnDefinition Width="*" />
    </Grid.ColumnDefinitions>
    <Grid.RowDefinitions>
        <RowDefinition Height="30" />
        <RowDefinition Height="30" />
    </Grid.RowDefinitions>
    <Label Text="Name:" />
    <Entry Grid.Column="1"
Placeholder="Enter your name" />
    <Label Grid.Row="1" Text="Age:" />
    <Entry Grid.Row="1" Grid.Column="1"
Placeholder="Enter your age" />
...
```

Layout

Care about Spacing and Padding

** ColumnSpacing or RowSpacing for Grid.*

Care about Spacing and Padding

```
…
<ContentPage.Content>
    <StackLayout>
        <StackLayout Padding="20,20,0,0">
            <Label Text="Hello" />
        </StackLayout>
        <StackLayout Padding="20,20,0,0">
            <Label Text="Welcome to the App!" />
        </StackLayout>
        <StackLayout Padding="20,20,0,0">
            <Label Text="Downloading Data..." />
        </StackLayout>
    </StackLayout>
</ContentPage.Content>
…
```

```
…
<ContentPage.Content>
  <StackLayout Padding="20,20,0,0"
Spacing="25">
    <Label Text="Hello" />
    <Label Text="Welcome to the App!" />
    <Label Text="Downloading Data..." />
  </StackLayout>
</ContentPage.Content>
…
```

Layout

Use LayoutOptions.Fill or 🥓 LayoutOptions.FillAndExpand

** You don't need to change the default values* 😀

Layout

Don’t use RelativeLayout 💩🤮

** Just don’t. RelativeLayout adds more work to the CPU (a lot of calculations will happen).*

Layout

Don’t use StackLayout 💩🤮 inside a ScrollView

** Why???*

Layout

Don’t use a ListView inside a ScrollView

** Why???*

Layout

Use the ListViews Header and Footer properties. 🥓

* *Just do it!*

Layout

Use ListViewCachingStrategy.RecycleElement 

** This is not the default*

Layout

Use the ListViews Header and Footer properties.

* *Just do it!*

Layout

Use HasUnevenRows when your ListView has rows of different sizes

* *Just do it!*

Layout

Use data template selectors to bind a collection of objects with different visual representations

* *Just do it!*

Layout

Don’t use TableView where you can use a ListView

** Just don’t!*

Layout

Don’t use multiples labels 💩🤮
to format different parts of a sentence.

** Use Spans and FormattedText*

👇🏼

Use Spans and FormattedText

```
…
        <Label LineBreakMode="WordWrap">
            <Label.FormattedText>
                <FormattedString>
                    <Span Text="Red Bold, " TextColor="Red" FontAttributes="Bold" />
                    <Span Text="default, " Style="{DynamicResource BodyStyle}">
                        <Span.GestureRecognizers>
                            <TapGestureRecognizer Command="{Binding TapCommand}" />
                        </Span.GestureRecognizers>
                    </Span>
                    <Span Text="italic small." FontAttributes="Italic" FontSize="Small" />
                </FormattedString>
            </Label.FormattedText>
        </Label>
…
```

Layout

Disable Label wrapping if possible 🥓

```
LineBreakMode="NoWrap"
```

Layout

Don’t set the vertical alignment property unless required. 💩🤮

`Label.VerticalTextAlignment`

Layout

Don't update any Label instances more frequently than required 💩🤮

* *Update in batch if it’s possible, because the change of size of the label can result in the entire screen layout being re-calculated*

Layout

Don't use CarouselPage 💩🤮

* *Use a CarouselView within a ContentPage*

Navigation

Await the PushAsync and PopAsync methods

Navigation

Use the AppCompat 🥓

* *Will improve both performance and the look of the application.*

MessagingCenter

Prefer use something else. 💩🤮

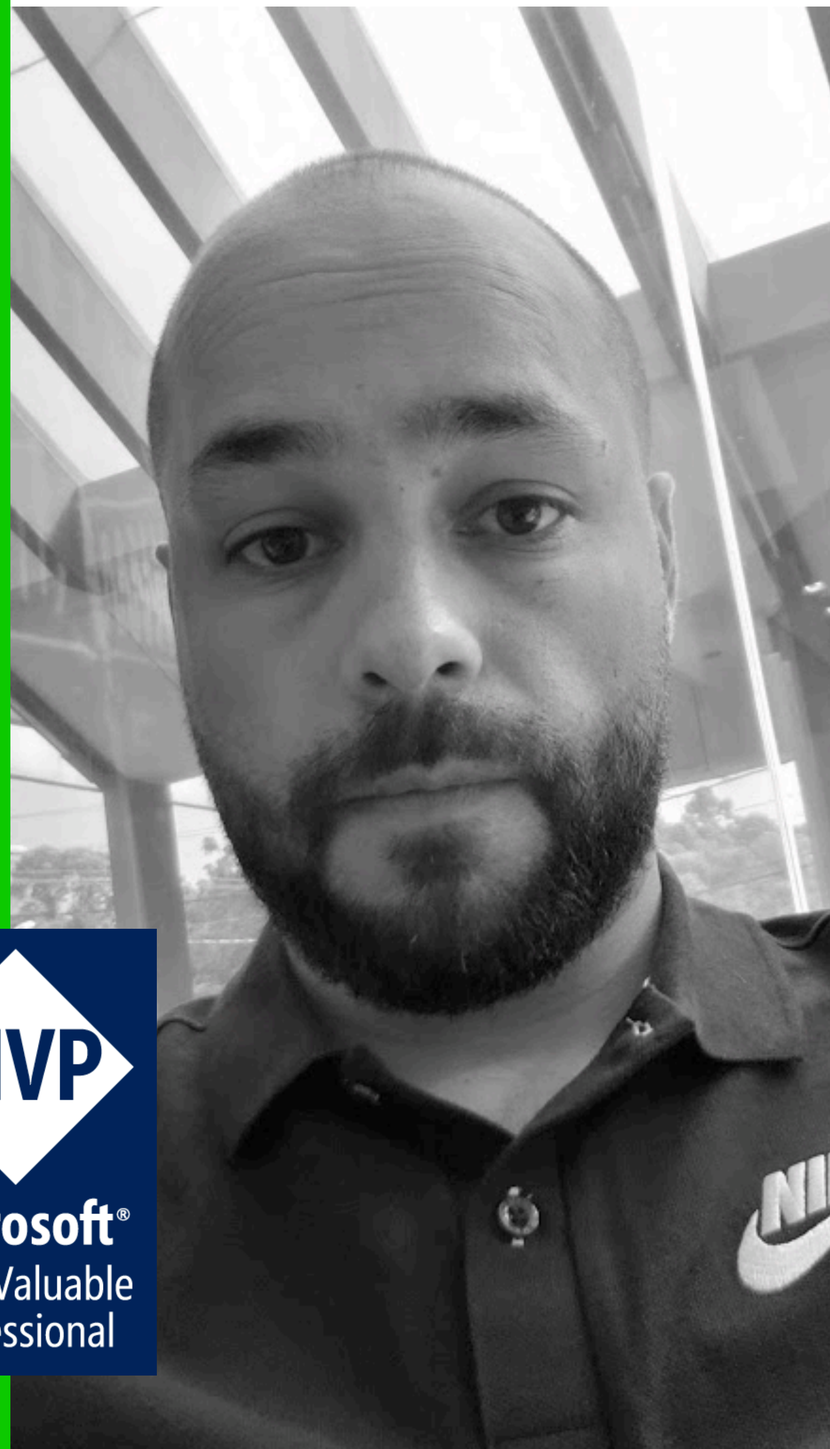
* *Prism, WIP…*

Obrigado 🙏

William S Rodriguez

Master Software Engineer @ ArcTouch

🦤 *https://twitter.com/willbuildapps*



MVP
Microsoft®
Most Valuable Professional

http://bit.ly/xf-booster-feedback

Feedback

https://arctouch.com/careers/

ARCTOUCH

References

👉🏻 Xamarin.Forms Fast Renderers: https://docs.microsoft.com/en-us/xamarin/xamarin-forms/internals/fast-renderers

👉🏻 Enable the XAML Compiler: https://docs.microsoft.com/en-us/xamarin/xamarin-forms/deploy-test/performance#enable-the-xaml-compiler

👉🏻 Layout Compression: https://docs.microsoft.com/en-us/xamarin/xamarin-forms/user-interface/layouts/layout-compression

👉🏻 Bindable Properties: https://docs.microsoft.com/en-us/xamarin/xamarin-fohttps://docs.microsoft.com/en-us/xamarin/xamarin-forms/xaml/bindable-propertiesrms/xaml/bindable-properties

👉🏻 Unsubscribe from Events: https://docs.microsoft.com/en-us/xamarin/cross-platform/deploy-test/memory-perf-best-practices#unsubscribe-from-events

👉🏻 Link All Assemblies: https://docs.microsoft.com/en-us/xamarin/cross-platform/deploy-test/memory-perf-best-practices#reduce-the-size-of-the-application

👉🏻 Misuses Of MessagingCenter: https://xamarinhelp.com/common-misuse-messagingcenter/

References

👉🏻 Layout Options in Xamarin.Forms: https://docs.microsoft.com/en-us/xamarin/xamarin-forms/user-interface/layouts/layout-options

👉🏻 Xamarin.Forms DataTemplateSelector: https://docs.microsoft.com/en-us/xamarin/xamarin-forms/app-fundamentals/templates/data-templates/selector

👉🏻 Jason Smith's Xamarin Forms Performance Tips: https://kent-boogaart.com/blog/jason-smith's-xamarin-forms-performance-tips

👉🏻 Optimizing App Performance with Xamarin.Forms – Jason Smith: https://www.youtube.com/watch?v=RZvdql3Ev0E

👉🏻 Xamarin.Forms Performance: https://docs.microsoft.com/en-us/xamarin/xamarin-forms/deploy-test/performance#enable-the-xaml-compiler

👉🏻 Xamarin.Forms is Much More Capable Than You Think: https://arctouch.com/blog/xamarin-forms-more-capable-than-you-think/

👉🏻 Improving Xamarin.Forms Startup Performance: https://xamarinhelp.com/improving-xamarin-forms-startup-performance/