Xamarin.Forms

Fagdag 31.10.14

Agenda

- Intro til Xamarin og Xamarin.Forms
- Pages, layouts og data binding
- Plattformtilpasninger
- Lunsj 11.30-12.30
- Hovedoppgave
- Avslutning og oppsummering ca. 14.30

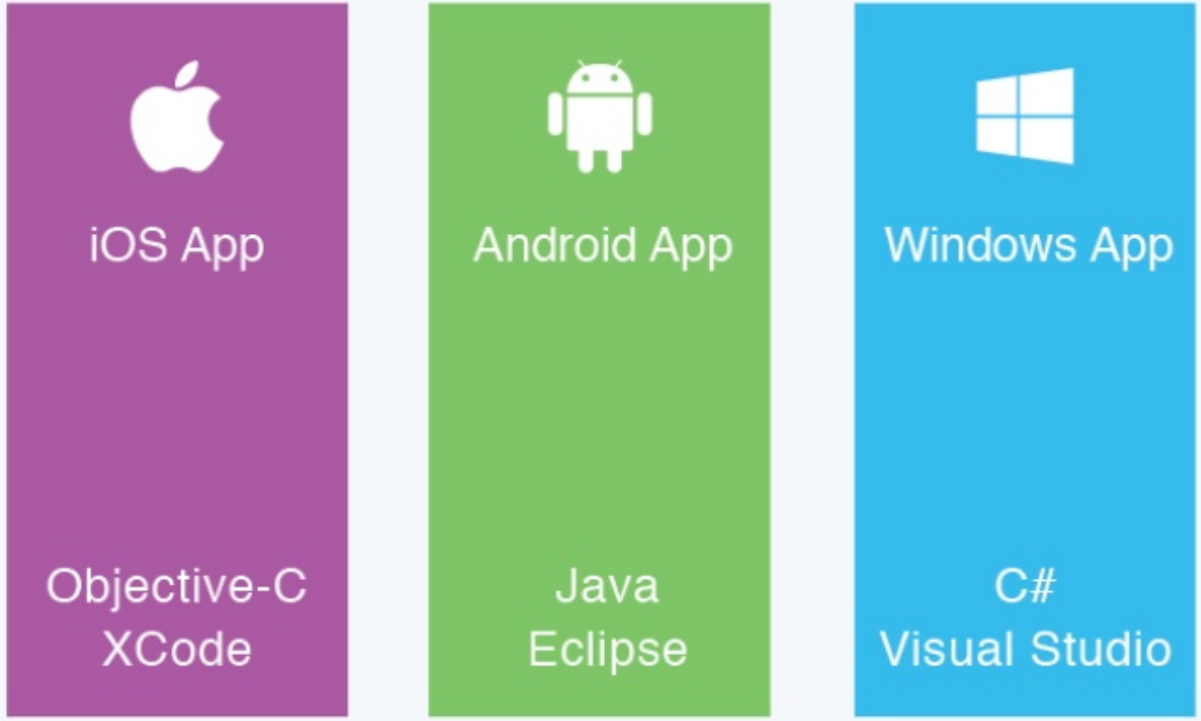
Attachmate

Xamarin

Xamarin

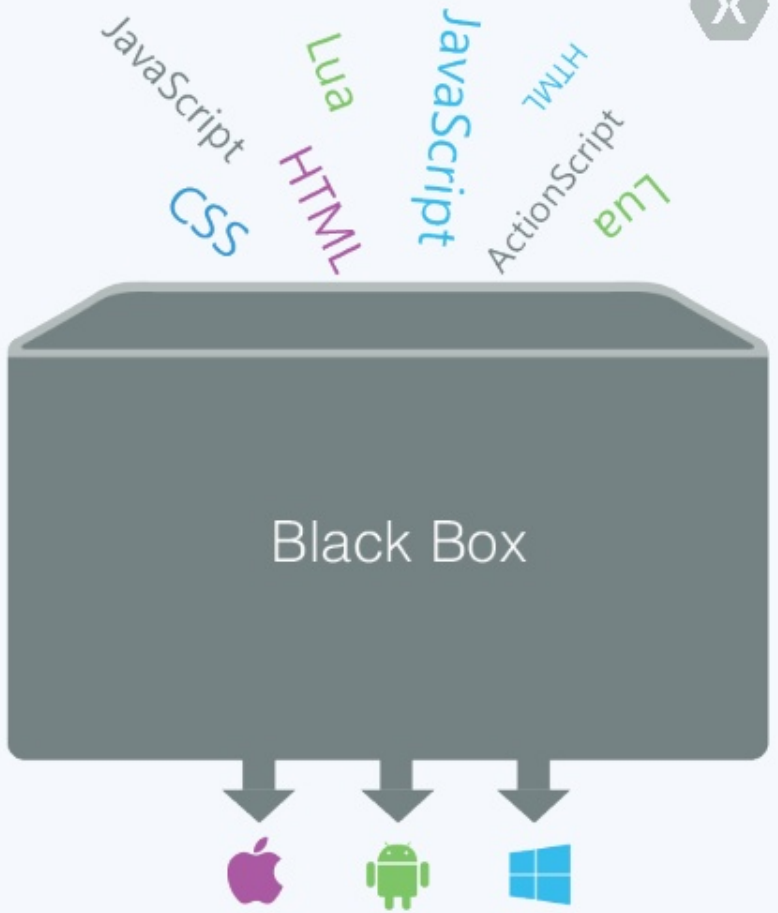
Silo Approach

Build Apps Multiple Times



Write Once, Run Anywhere Approach

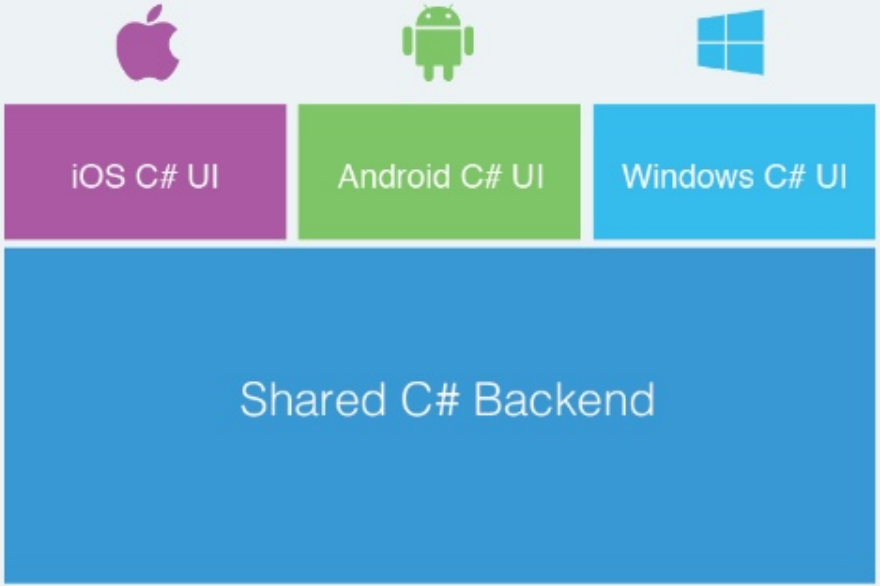
Lowest Common Denominator



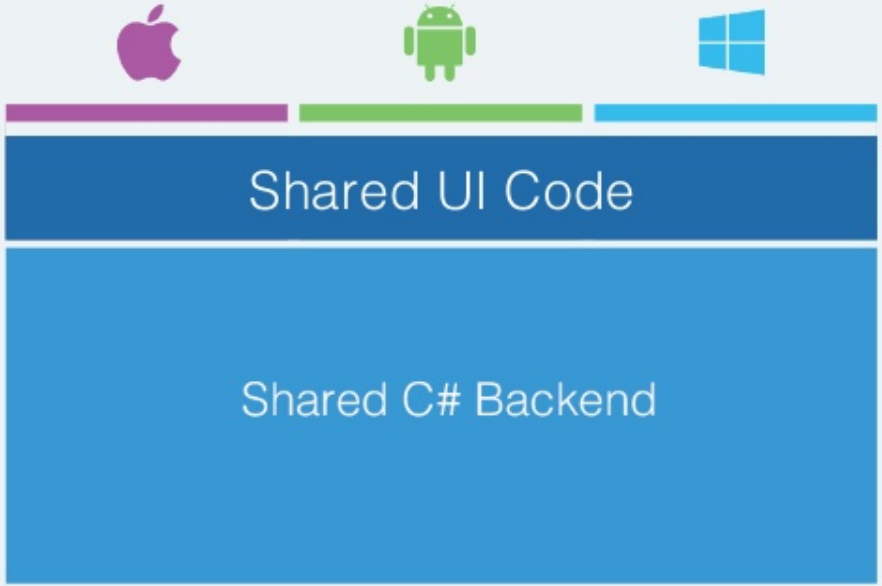
Xamarin + Xamarin.Forms

Xamarin

Traditional Xamarin approach



With Xamarin.Forms:
more code-sharing, native controls



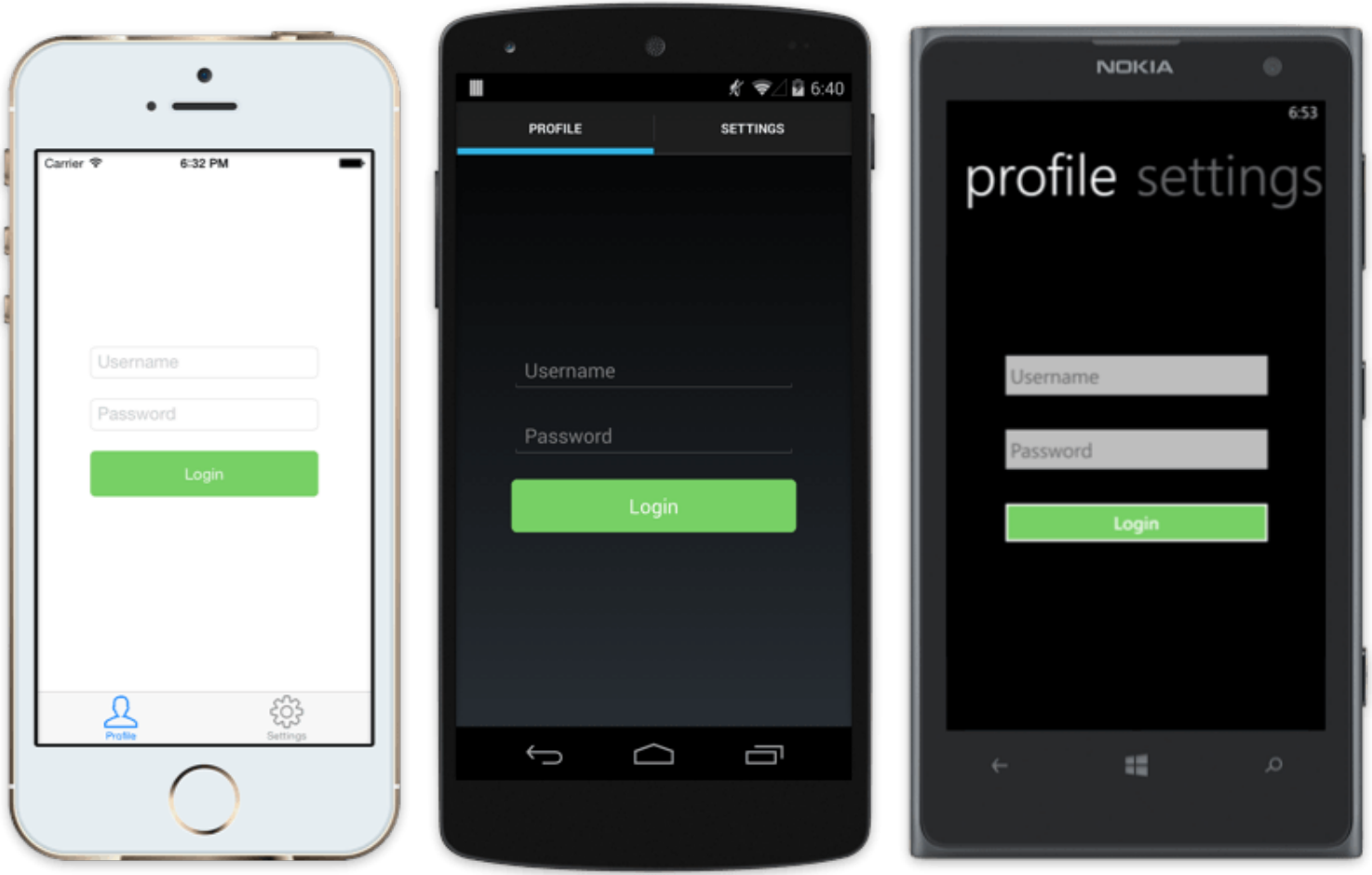
Hvorfor C#?

- Debugging
- Reflection
- Collections
- Globalization
- File I/O
- Networking
- Security
- Lambda
- LINQ
- Threading
- Async/await
- Web services
- XML parsing
- ...

Native Performance

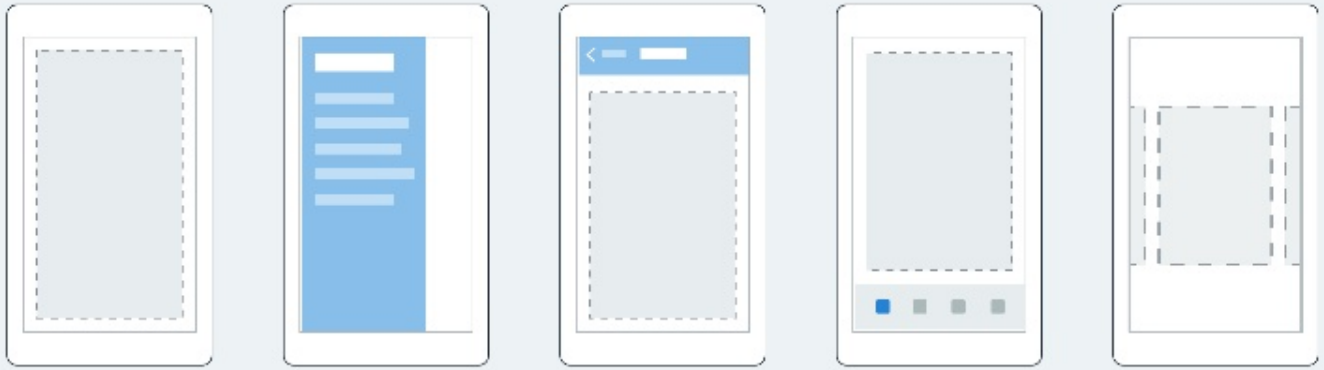
Xamarin.iOS does full Ahead Of Time (AOT) compilation to produce an ARM binary for Apple's App Store.

Xamarin.Android takes advantage of Just In Time (JIT) compilation on the Android device.


Carrier
6:32 PM
Username
Password
Login
Profile
Settings
6:40
PROFILE
SETTINGS
Username
Password
Login
NOKIA
6:53
profile settings
Username
Password
Login

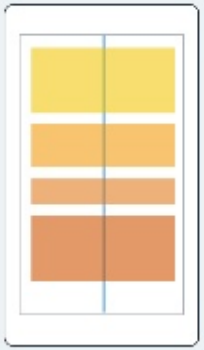
Pages

Xamarin

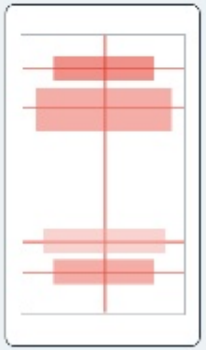


Layouts

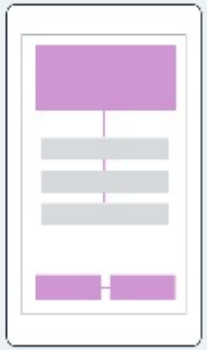
Xamarin



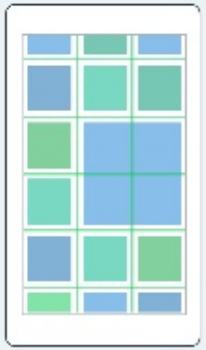
Stack



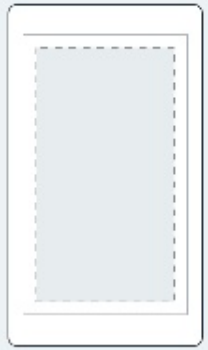
Absolute



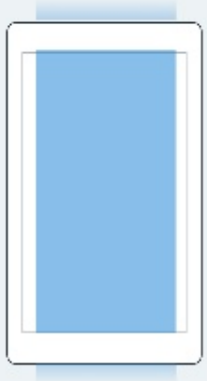
Relative



Grid



ContentView



ScrollView

Frame

Controls

Xamarin

ActivityIndicator	BoxView	Button	DatePicker	Editor
Entry	Image	Label	ListView	Map
OpenGLView	Picker	ProgressBar	SearchBar	Slider
Stepper	TableView	TimePicker	WebView	EntryCell
ImageCell	SwitchCell	TextCell	ViewCell	

Demo

Oppgave: Magic 8 Ball

Lag en Magic 8 Ball-app som inneholder

- Et StackLayout
- En Label
- En Button som endrer teksten på labelen

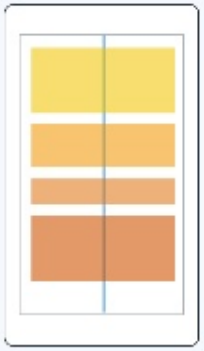
git@github.com:lsmeby/XamarinWorkshop.git

Oppgave: Magic 8 Ball i XAML

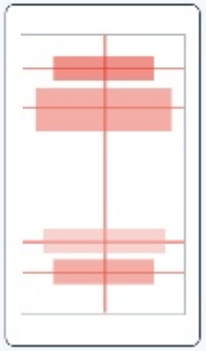
Lag Magic 8 Ball-appen på nytt, denne gangen med XAML

Layouts

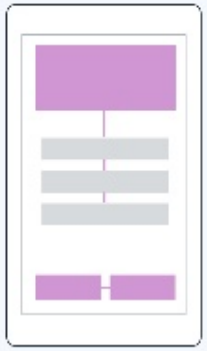
Xamarin



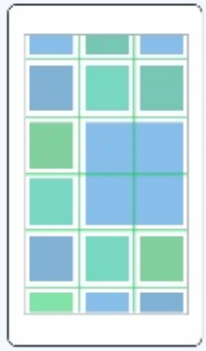
Stack



Absolute

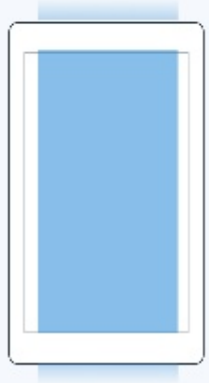


Relative



Grid

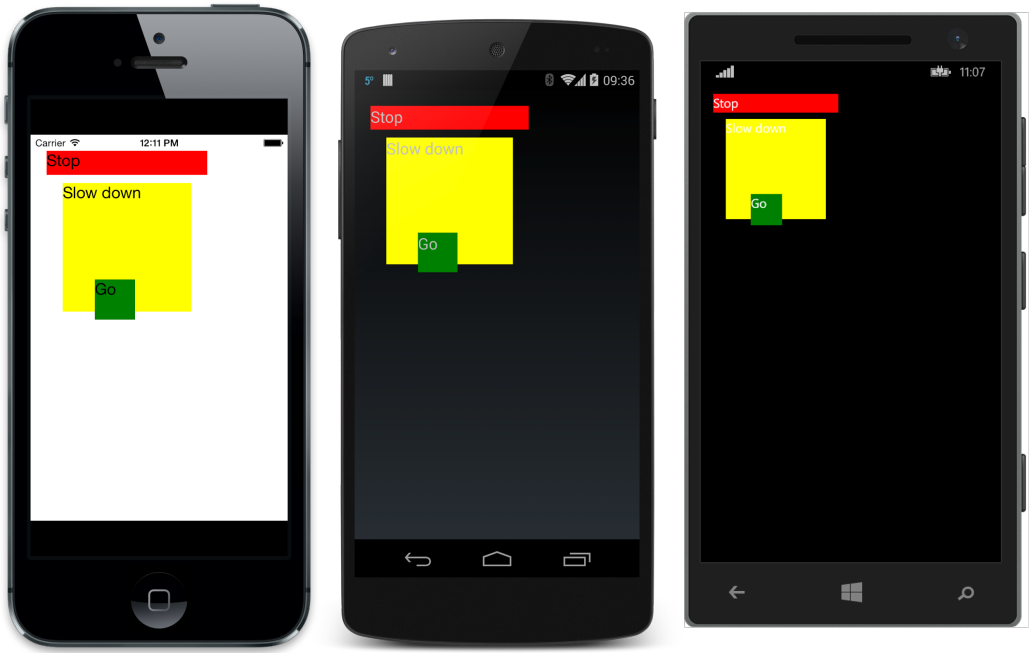
ContentView



ScrollView

Frame

AbsoluteLayout



AbsoluteLayout

```
public class MyAbsoluteLayoutPage : ContentPage
{
    public MyAbsoluteLayoutPage()
    {
        var red = new Label
        {
            Text = "Stop",
            BackgroundColor = Color.Red,
            Font = Font.SystemFontOfSize (20),
            WidthRequest = 200,
            HeightRequest = 30
        };

       // Creating yellow and green label

        var absLayout = new AbsoluteLayout();
        absLayout.Children.Add(red, new Point(20,20));
        absLayout.Children.Add(yellow, new Point(40,60));
        absLayout.Children.Add(green, new Point(80,180));

        Content = absLayout;
    }
}
```

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
  x:Class="HelloXamarinFormsWorldXaml.AbsoluteLayoutExample"
  Padding="20">

  <AbsoluteLayout>

    <Label Text="Stop"
           BackgroundColor="Red"
           Font="20"
           AbsoluteLayout.LayoutBounds="20,20,200,30" />

    <Label Text="Slow down"
           BackgroundColor="Yellow"
           Font="20"
           AbsoluteLayout.LayoutBounds="40,60,160,160" />

    <Label Text="Go"
           BackgroundColor="Green"
           Font="20"
           AbsoluteLayout.LayoutBounds="80,180,50,50" />

  </AbsoluteLayout>

</ContentPage>
```

RelativeLayout

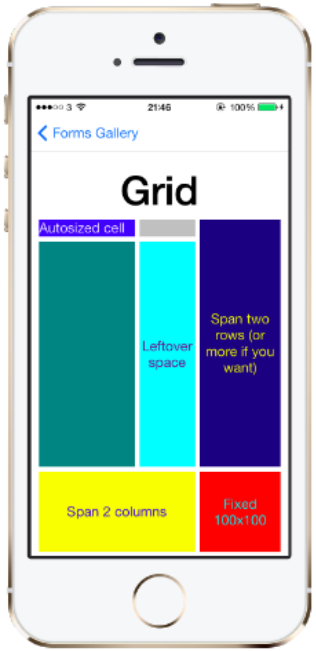
StackLayout

StackLayout

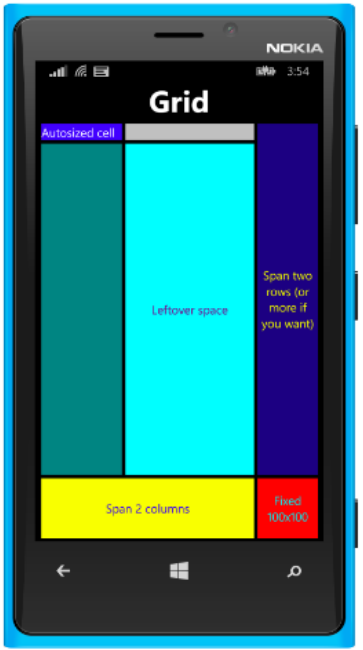
```
public class StackLayoutExample: ContentPage
{
    public StackLayoutExample()
    {
        Padding = new Thickness(20);
        var red = new Label
        {
            Text = "Stop",
            BackgroundColor = Color.Red,
            Font = Font.SystemFontOfSize (20)
        };

       // Creating label yellow and green here

        Content = new StackLayout
        {
            Spacing = 10,
            Children = { red, yellow, green }
        };
    }
}
```

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
  x:Class="HelloXamarinFormsWorldXaml.StackLayoutExample1"
  Padding="20">

  <StackLayout Spacing="10">

    <Label Text="Stop"
           BackgroundColor="Red"
           Font="20" />

    <Label Text="Slow down"
           BackgroundColor="Yellow"
           Font="20" />

    <Label Text="Go"
           BackgroundColor="Green"
           Font="20" />

  </StackLayout>
</ContentPage>
```

Grid


Forms Gallery
Grid
Autosized cell
Leftover space
Span two rows (or more if you want)
Span 2 columns
Fixed 100x100


Grid
Autosized cell
Leftover space
Span two rows (or more if you want)
Span 2 columns
Fixed 100x100


NOKIA
Grid
Autosized cell
Leftover space
Span two rows (or more if you want)
Span 2 columns
Fixed 100x100

Grid

```
public class GridExample: ContentPage
{
    public GridExample()
    {
        // Creating labels here

        var grid = new Grid();

        grid.Children.Add(label1, 0, 0);
        grid.Children.Add(label2, 1, 0);
        grid.Children.Add(label3, 2, 0);
        grid.Children.Add(label4, 0, 1);
        grid.Children.Add(label5, 1, 3, 1, 2);

        Content = grid;

    }
}
```

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
  x:Class="HelloXamarinFormsWorldXaml.GridExample1"
  Padding="20">

  <Grid>

    <Label Text="Label1" Grid.Column="0" Grid.Row="0" />
    <Label Text="Label2" Grid.Column="1" Grid.Row="0" />
    <Label Text="Label3" Grid.Column="2" Grid.Row="0" />
    <Label Text="Label4" Grid.Column="0" Grid.Row="1" />
    <Label Text="Label5" Grid.Column="1"
           Grid.Row="1" Grid.ColumnSpan="2" />

  </Grid>
</ContentPage>
```

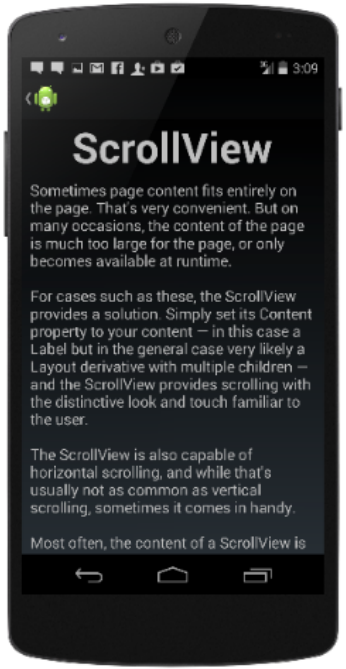
Grid

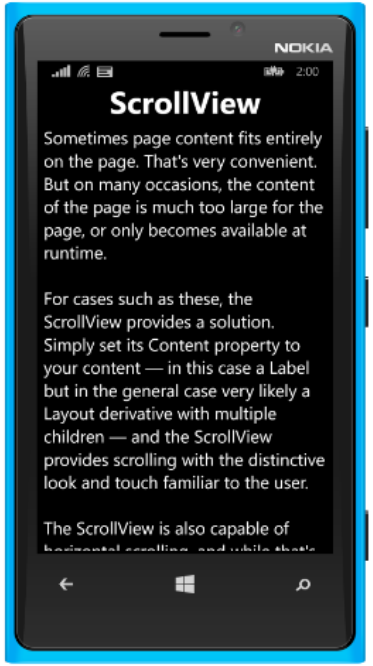
Du kan sette bredde/høyde på hver kolonne og rad med ColumnDefinition og RowDefinition

- Auto - basert på største barn
- Absolute - en gitt numerisk verdi
- Star - proporsjonell fordeling

ScrollView


20:03
50%
Forms Gallery
ScrollView
occasions, the content of the page is much too large for the page, or only becomes available at runtime.
For cases such as these, the ScrollView provides a solution. Simply set its Content property to your content — in this case a Label but in the general case very likely a Layout derivative with multiple children — and the ScrollView provides scrolling with the distinctive look and touch familiar to the


3:09
ScrollView
Sometimes page content fits entirely on the page. That's very convenient. But on many occasions, the content of the page is much too large for the page, or only becomes available at runtime.
For cases such as these, the ScrollView provides a solution. Simply set its Content property to your content — in this case a Label but in the general case very likely a Layout derivative with multiple children — and the ScrollView provides scrolling with the distinctive look and touch familiar to the user.
The ScrollView is also capable of horizontal scrolling, and while that's usually not as common as vertical scrolling, sometimes it comes in handy.
Most often, the content of a ScrollView is


NOKIA
2:00
ScrollView
Sometimes page content fits entirely on the page. That's very convenient. But on many occasions, the content of the page is much too large for the page, or only becomes available at runtime.
For cases such as these, the ScrollView provides a solution. Simply set its Content property to your content — in this case a Label but in the general case very likely a Layout derivative with multiple children — and the ScrollView provides scrolling with the distinctive look and touch familiar to the user.
The ScrollView is also capable of

LayoutOptions

LayoutOptions.Start

LayoutOptions.Center

LayoutOptions.End

LayoutOptions.Fill

LayoutOptions.StartAndExpand

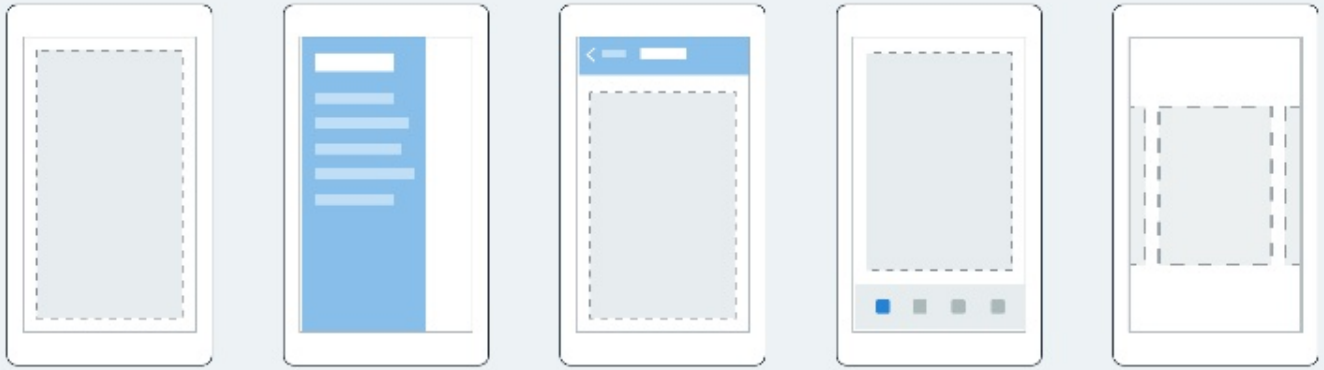
LayoutOptions.CenterAndExpand

LayoutOptions.EndAndExpand

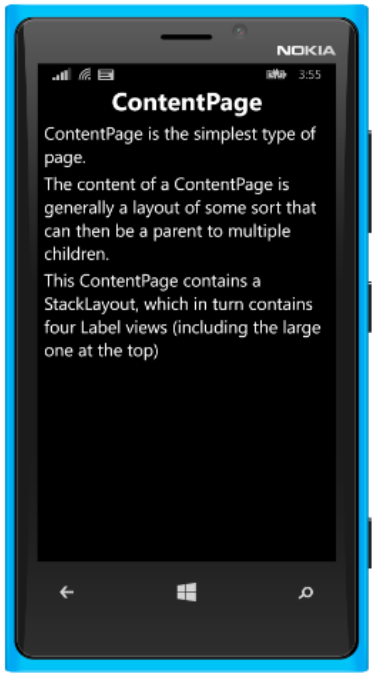
LayoutOptions.FillAndExpand

Pages

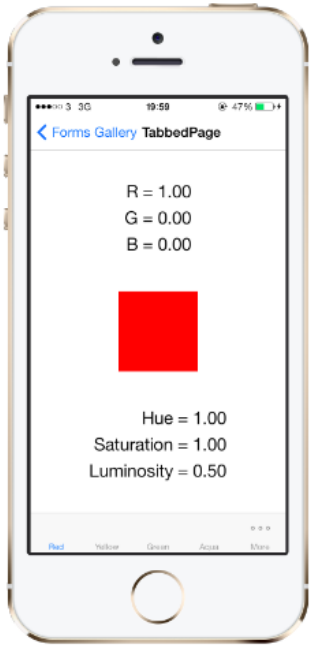
Xamarin



ContentPage



TabbedPage


Forms Gallery TabbedPage
R = 1.00
G = 0.00
B = 0.00
Hue = 1.00
Saturation = 1.00
Luminosity = 0.50


TabbedPage
RED
YELLOW
GREEN
AQUA
BLUE
R = 0.00
G = 0.50
B = 0.00
Hue = 0.33
Saturation = 1.00
Luminosity = 0.25


NOKIA
TabbedPage
red yellow gree
R = 1.00
G = 0.00
B = 0.00
Hue = 1.00
Saturation = 1.00
Luminosity = 0.50

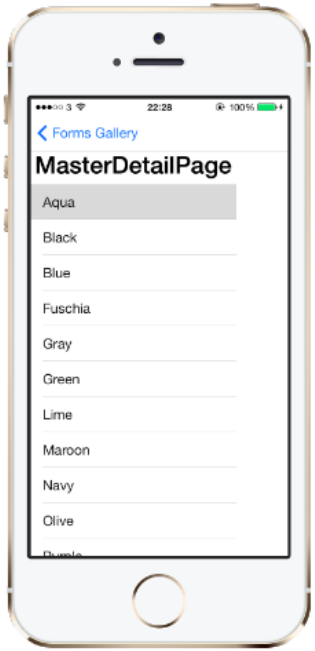
TabbedPage

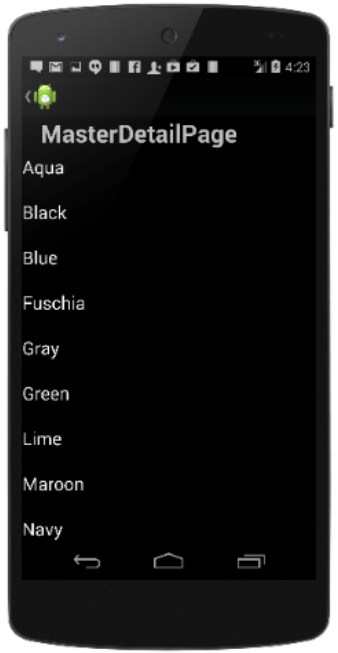
```
class TabbedPageDemoPage : TabbedPage
{
    public TabbedPageDemoPage()
    {
        this.Children.Add(new MyFirstPage());
        this.Children.Add(new MySecondPage());
        this.Children.Add(new MyThirdPage());
        InitializeComponent();
    }
}
```

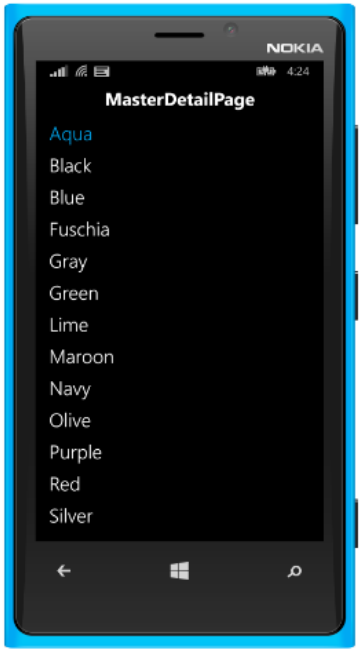
```
<TabbedPage
  xmlns="http://xamarin.com/schemas/2014/forms"
  xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
  xmlns:my="clr-namespace:MyNameSpace;assembly=MyAssembly"
  x:Class="MyNameSpace.TabbedPageDemoPage">

  <TabbedPage.Children>
    <my:MyFirstPage />
    <my:MySecondPage />
    <my:MyThirdPage />
  </TabbedPage.Children>

</TabbedPage >
```

CarouselPage

MasterDetailPage







NavigationPage

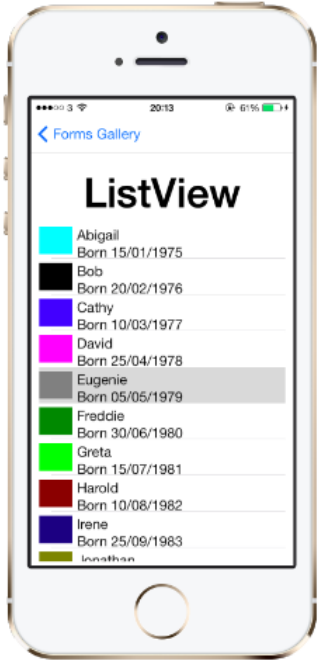
NavigationPage

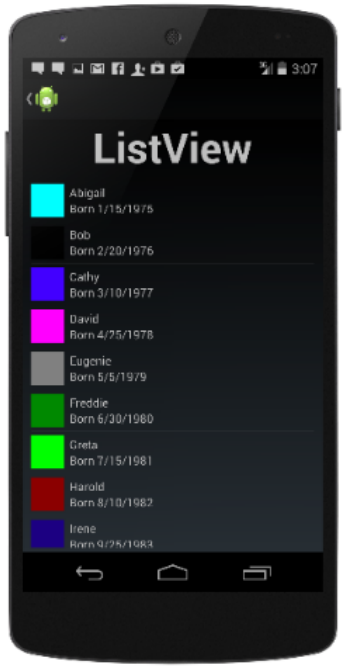
```
class NavigationPageDemo : ContentPage
{
    public NavigationPageDemo()
    {
        Button button = new Button
        {
            Text = "Open another page"
        };

        button.Clicked += async (sender, args) =>
         await Navigation.PushAsync(new
SomeOtherPage());

        this.Content = button;
    }
}
```

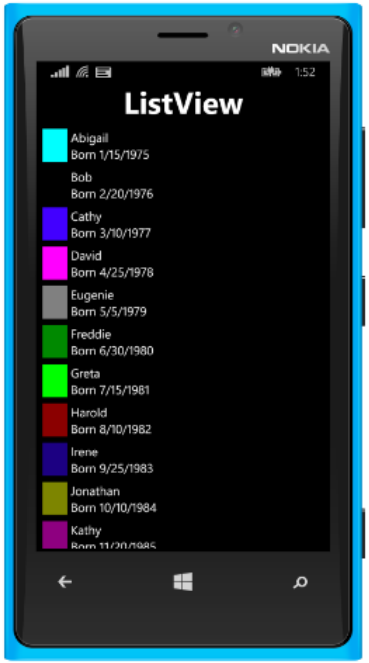
```
<ContentPage
  xmlns="http://xamarin.com/schemas/2014/forms"
  xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
  x:Class="MyNameSpace.NavigationPageDemo">

  <Button Text="Open another page" Clicked="Button_click" />

</ContentPage >
```

```
public class App
{
    public static Page GetMainPage()
    {
        return new NavigationPage(new NavigationPageDemo());
    }
}
```

ListView


20:13
61%
Forms Gallery
ListView
Abigail
Born 15/01/1975
Bob
Born 20/02/1976
Cathy
Born 10/03/1977
David
Born 25/04/1978
Eugenie
Born 05/05/1979
Freddie
Born 30/06/1980
Greta
Born 15/07/1981
Harold
Born 10/08/1982
Irene
Born 25/09/1983


3:07
ListView
Abigail
Born 1/15/1975
Bob
Born 2/20/1976
Cathy
Born 3/10/1977
David
Born 4/25/1978
Eugenie
Born 5/5/1979
Freddie
Born 6/30/1980
Greta
Born 7/15/1981
Harold
Born 8/10/1982
Irene


NOKIA
1:52
ListView
Abigail
Born 1/15/1975
Bob
Born 2/20/1976
Cathy
Born 3/10/1977
David
Born 4/25/1978
Eugenie
Born 5/5/1979
Freddie
Born 6/30/1980
Greta
Born 7/15/1981
Harold
Born 8/10/1982
Irene
Born 9/25/1983
Jonathan
Born 10/10/1984
Kathy

ListView

Innebygde celler:

- EntryCell (label og tekstboks)
- SwitchCell (label og av/på-knapp)
- TextCell (to labeler, primær/sekundær)
- ImageCell (TextCell med bilde)

Eventer:

- ItemSelected
- ItemTapped

ListView

```
class ListViewDemo : ContentPage
{
    public ListViewDemo()
    {
        var listView = new ListView()
        {
            ItemsSource = myCollection,
            ItemTemplate =
                        new DataTemplate(typeof(TextCell))
        }

        listView.ItemTemplate.SetBinding(
                                    TextCell.TextProperty,
                                    "Title");
        Content = listView;
    }
};
```

```
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
 xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
 xmlns:my="clr-namespace:ListViewTest;assembly=ListViewTest"
 x:Class="ListViewTest.ListViewDemo">

<ListView ItemsSource="{x:Static my:MainPage.MyCollection}">
    <ListView.ItemTemplate>
        <DataTemplate>
            <TextCell Text="{Binding Title}" />
        </DataTemplate>
    </ListView.ItemTemplate>
</ListView>

</ContentPage>
```

Data binding

- Oppretter en kobling mellom to objekter
- Som oftest mellom GUI-element og dataobjekt
- To-veis-binding
- Dataobjektet må implementere INotifyPropertyChanged

I Xamarin Forms:

- BindingContext på en Page eller Control settes til objektet
- SetBinding kalles på alle GUI-elementer som skal bindes

Data binding

```
public static Page GetMainPage()
{
    var label = new Label();
    label.SetBinding(Label.TextProperty, "Name");

    return new ContentPage
    {
        BindingContext = new Person("Lars"),
        Content = label
    };
}
```

```
<ContentPage
  xmlns="http://xamarin.com/schemas/2014/forms"
  xmlns:x="http://schemas.microsoft.
com/winfx/2009/xaml"
  x:Class="BindingTestXaml.MainPage">

  <Label Text="{Binding Name}" />

</ContentPage>
```

```
public MainPage()
{
    InitializeComponent();
    this.BindingContext = new Person("Lars");
}
```

Oppgave: ToDo-app

Oppgave: ToDo-app

- Benytt et ListView til todo’s
- Navigasjon til detaljside ved tap
- Benytt binding mellom GUI og model

Device.OnPlatform

```
// left and right padding: 5; top padding: 20 (only on iOS)
layout.Padding = new Thickness (5, Device.OnPlatform(20,0,0), 5, 0),
```

```
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="XamlSamples.GridDemoPage"
             Title="Grid Demo Page">

  <ContentPage.Padding>

    <OnPlatform
      x:TypeArguments="Thickness"
      iOS="5, 20, 5, 0"
      Android="5, 0, 5, 0"
      WinPhone="5, 0, 5, 0" />
  </ContentPage.Padding>

  …

</ContentPage>
```

Custom renderer

```
public class MyEntry : Entry {}
```

```
public class MyEntryRenderer : EntryRenderer
{
  // Override the OnElementChanged method so we can tweak this renderer post-initial setup
  protected override void OnElementChanged (ElementChangedEventArgs<Entry> e)
  {
    base.OnElementChanged (e));
    if (e.OldElement == null) {   // perform initial setup
      // lets get a reference to the native control
      var nativeTextField = (UITextField) Control;
      // do whatever you want to the UITextField here!
      nativeTextField.BackgroundColor = UIColor.LightGray;
      nativeTextField.BorderStyle = UITextBorderStyle.Line;
    }
  }
}
```

```
[assembly: ExportRenderer (typeof (MyEntry), typeof (MyEntryRenderer))]
```

Custom controls

- Samme prinsipp som custom renderers

Native services

```
1  public interface ITextToSpeech
2  {
3      void Speak (string text);
4  }
```

Native services

```
[assembly: Xamarin.Forms.Dependency (typeof (TextToSpeech_iOS))]
```

```
public class TextToSpeech_iOS : ITextToSpeech
{
    public TextToSpeech_iOS () {}

    public void Speak (string text)
    {
        var speechSynthesizer = new AVSpeechSynthesizer ();

        var speechUtterance = new AVSpeechUtterance (text) {
            Rate = AVSpeechUtterance.MaximumSpeechRate/4,
            Voice = AVSpeechSynthesisVoice.FromLanguage ("en-US"),
            Volume = 0.5f,
            PitchMultiplier = 1.0f
        };

        speechSynthesizer.SpeakUtterance (speechUtterance);
    }
}
```

Native service

```
1  public MainPage ()
2  {
3      var speak = new Button {
4          Text = "Hello, Forms !",
5          VerticalOptions = LayoutOptions.CenterAndExpand,
6          HorizontalOptions = LayoutOptions.CenterAndExpand,
7      };
8      speak.Clicked += (sender, e) => {
9          DependencyService.Get<ITextToSpeech>().Speak("Hello from Xamarin Forms");
10     };
11     Content = speak;
12 }
```

Oppgave: OnPlatform

- Endre ToDo-appen så den har forskjellig bakgrunnsfarge på de ulike plattformene

Lunsj

11.30 - 12.30

Forslag til hovedoppgave

- Forbedring av Magix 8 Ball
  - Trigg teksten med akselerometeret ved hjelp av native services
  - Endre GUI til å se ut som en ekte Magic 8 Ball
  - Få spådommen opplest med TextToSpeach
- Fullfør ToDo-appen
  - Lagre todo’ene på telefonen med SQLite
- Lag MineSweeper
  - Benytt ferdig spillmotor under “Oppgave 4” hvis ønskelig
- Lag det du har lyst til

Oppsummering

Tilbakemelding

http://tinyurl.com/xamarin-eval