

C# Mobile

Week 1

- Introductie .NET MAUI
- Introductie XAML



**DE HOGESCHOOL
MET HET NETWERK**

Elfde-Liniestraat 24, 3500 Hasselt, www.pxl.be

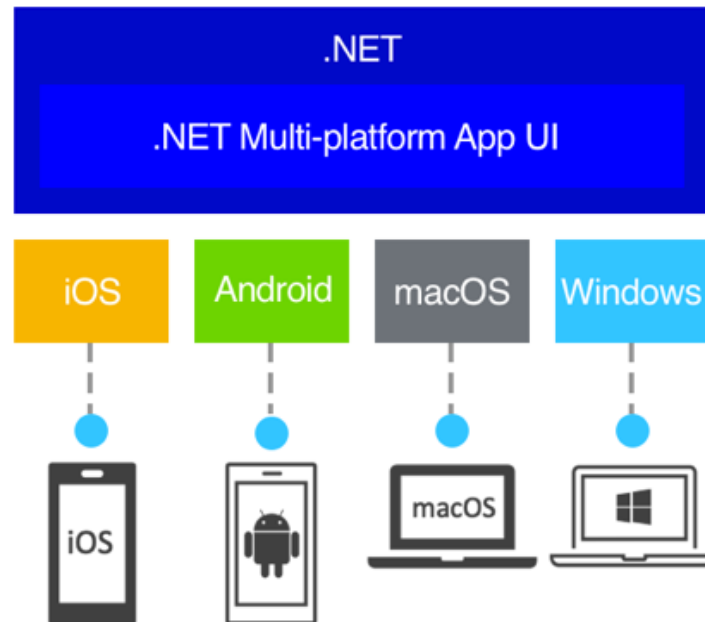




.NET MAUI

.NET MAUI

- Door gebruik te maken van .NET MAUI kan je applicaties ontwikkelen die zowel in Android, iOS, macOS en Windows kunnen worden uitgevoerd en gebruik maken van één en dezelfde basiscode.



XAML: Extended Application Markup Language



Handig om visuele blokken (UI) te bouwen



Gebaseerd op XML



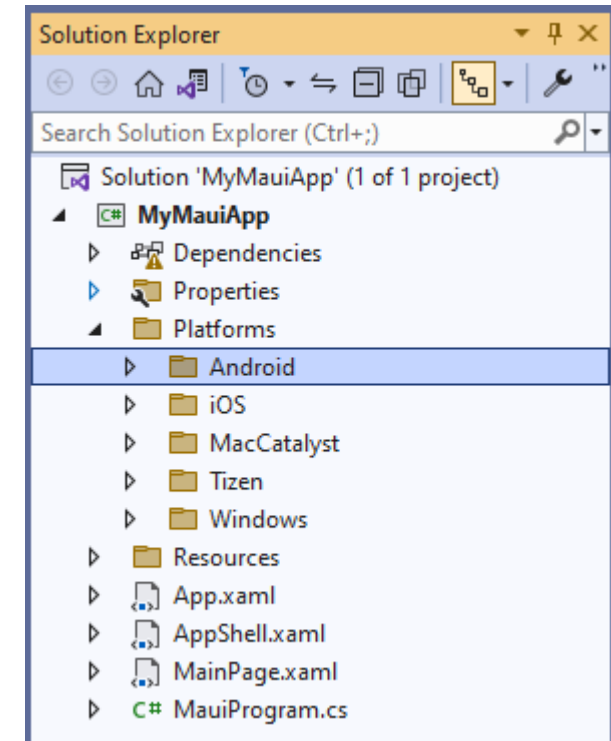
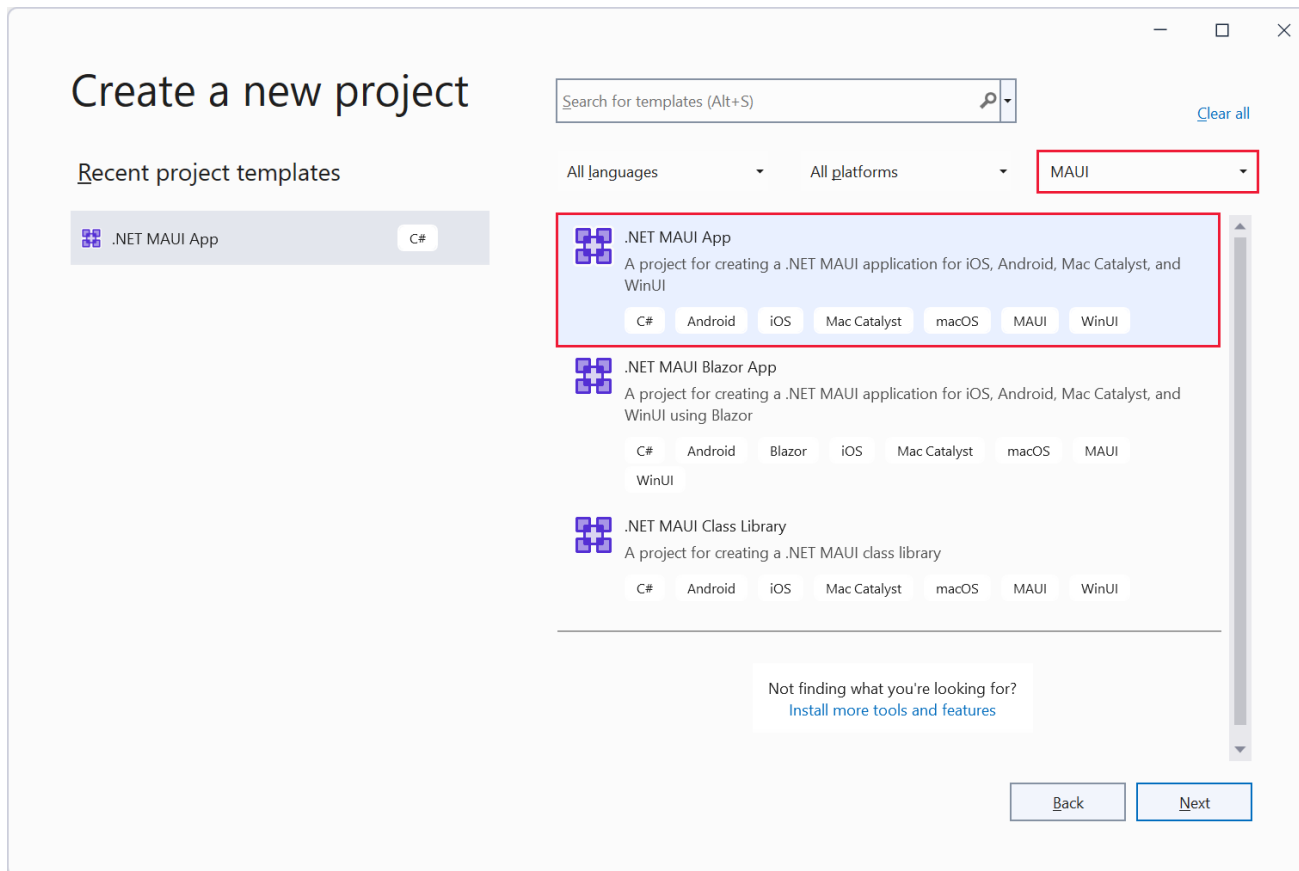
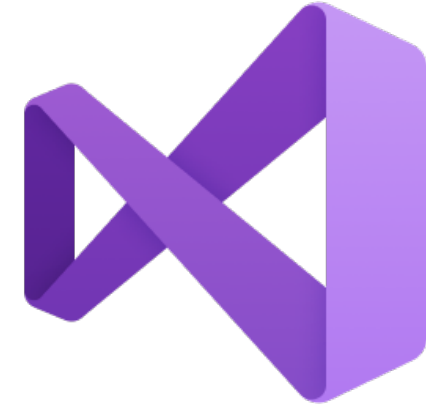
Objecten automatisch in code gegenereerd

Let op: er meerdere XAML dialecten

=> niet helemaal hetzelfde als WPF

.NET MAUI applicatie bouwen

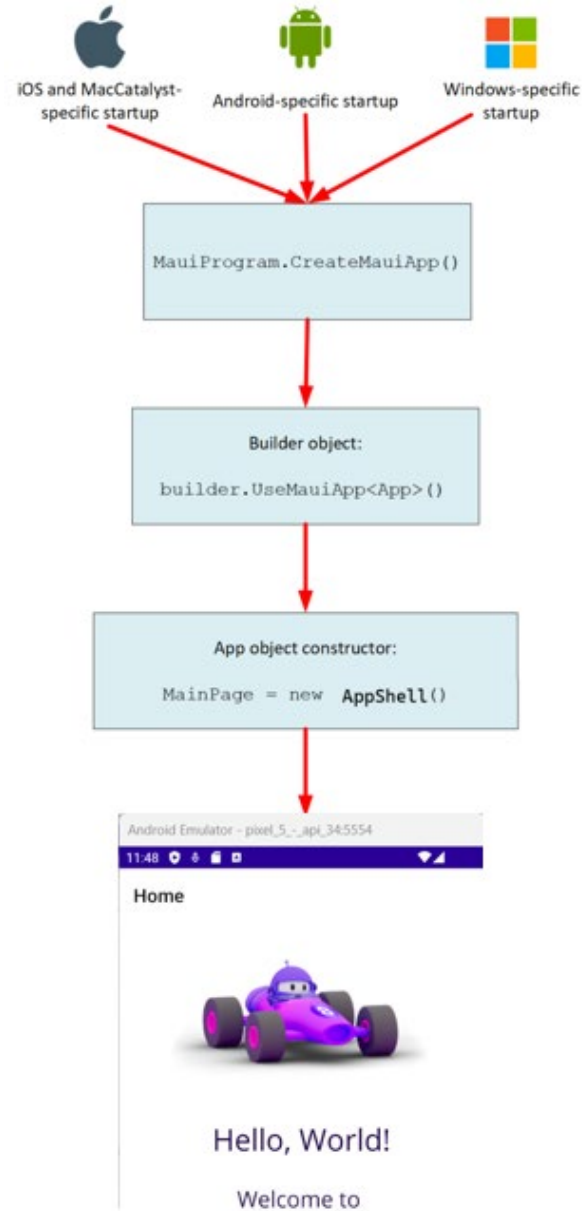
Open Visual Studio 2022



Opbouw project

- Een nieuw .NET MAUI project bevat allerlei standaard files:
 - **MauiProgram.cs**: bevat de code voor de creatie en configuratie van het *Application* object
 - **App.xaml** and **App.xaml.cs**: voorzien de standaard UI resources en maken het initiële venster van de applicatie aan
 - **AppShell.xaml** and **AppShell.xaml.cs**: hier worden de routes van de verschillende pagina's geregistreerd en wordt de startpagina bepaald
 - **MainPage.xaml** and **MainPage.xaml.cs**: wordt meestal gebruikt als startpagina. De XAML code bepaalt de layout en het *code-behind* bestand de UI logic.

Flow of control

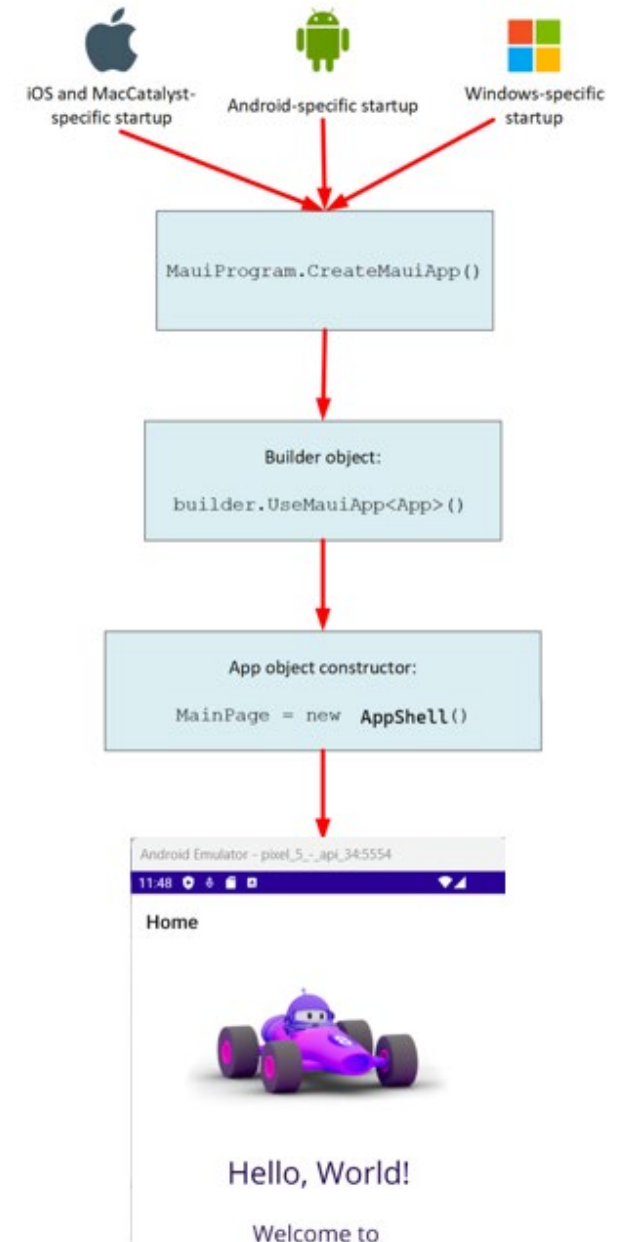


.NET MAUI project

- Er zijn enkele default resources en bootstrap code aanwezig
 - Plaats voor images, icoontjes, lettertypes
 - Hier moet je typisch niet veel aan veranderen
 - Hoe meer je je aan de conventies houdt, hoe makkelijker
- Uiteraard kunnen we zelf extra dingen toevoegen
 - Extra pagina's in de app
 - Eigen C# klassen om onze data en logica in vorm te geven

Application class

- “The App class represents the .NET MAUI application as a whole”
- Erft eigenschappen en gedrag van `Microsoft.Maui.Controls.Application`
- Deze klasse wordt geïntantieerd en ingeladen door bootstrap code die specifiek is voor elk platform
- De constructor van App zal typisch een `AppShell` object aanmaken en dit toewijzen aan de `MainPage` property



Application class

- De App class bevat daarnaast:
 - Methods for handling life-cycle events, including when the app is sent to the background
(That is, when it ceases to be the foreground app)
 - Methods for creating new Windows for the application
(The .NET MAUI application by default has a single window, but you can create and launch additional windows, which is helpful in desktop and tablet applications)

Shell

- Shell maakt het makkelijker om apps te bouwen
- Het bevat enkele features die in bijna elke app zitten:
 - (Visuele) beschrijving van de algemene opbouw van de app
 - Herkenbare navigatie hulpmiddelen (back-toets, fly-out menu, ...)
 - Navigatie tussen de verschillende onderdelen van een app via een soort URLs
- We komen later in de cursus terug op Shell



XAML

Voordeel XAML

Please log in

Username

Password

Log in

Voordeel XAML

- Puur in C# code:

```
namespace Sample
{
    public class MyPage : ContentPage
    {
        Button loginButton;
        StackLayout layout;

        public MyPage()
        {
            layout = new StackLayout
            {
                Children =
                {
                    new Label { Text = "Please log in" },
                    new Label { Text = "Username", TextColor = Color.Black },
                    new Entry (),
                    new Label { Text = "Password", TextColor = Color.Black },
                    new Entry { IsPassword = true },
                }
            };

            loginButton = new Button { Text = "Login" };

            layout.Children.Add(loginButton);

            Content = layout;

            loginButton.Clicked += (sender, e) =>
            {
                Debug.WriteLine("Clicked !");
            };
        }
    }
}
```

Voordeel XAML

- Zelfde UI maar dan in XAML (en achterliggende file):

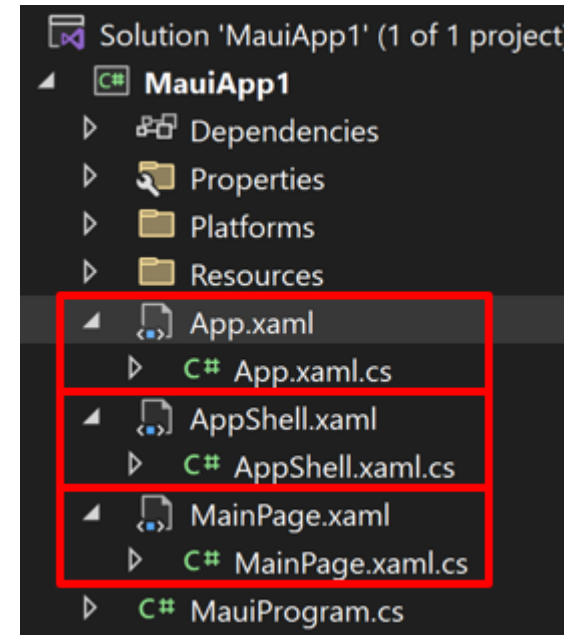
```
<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="Sample.MyPage">
  <ContentPage.Content>
    <StackLayout>
      <Label Text="Please log in" />
      <Label Text="Username" TextColor="Black" />
      <Entry />
      <Label Text="Password" TextColor="Black" />
      <Entry IsPassword="true" />
      <Button Text="Log in" Clicked="LoginButton_Clicked" />
    </StackLayout>
  </ContentPage.Content>
</ContentPage>
```

```
public partial class MyPage : ContentPage
{
    public MyPage()
    {
        InitializeComponent();
    }

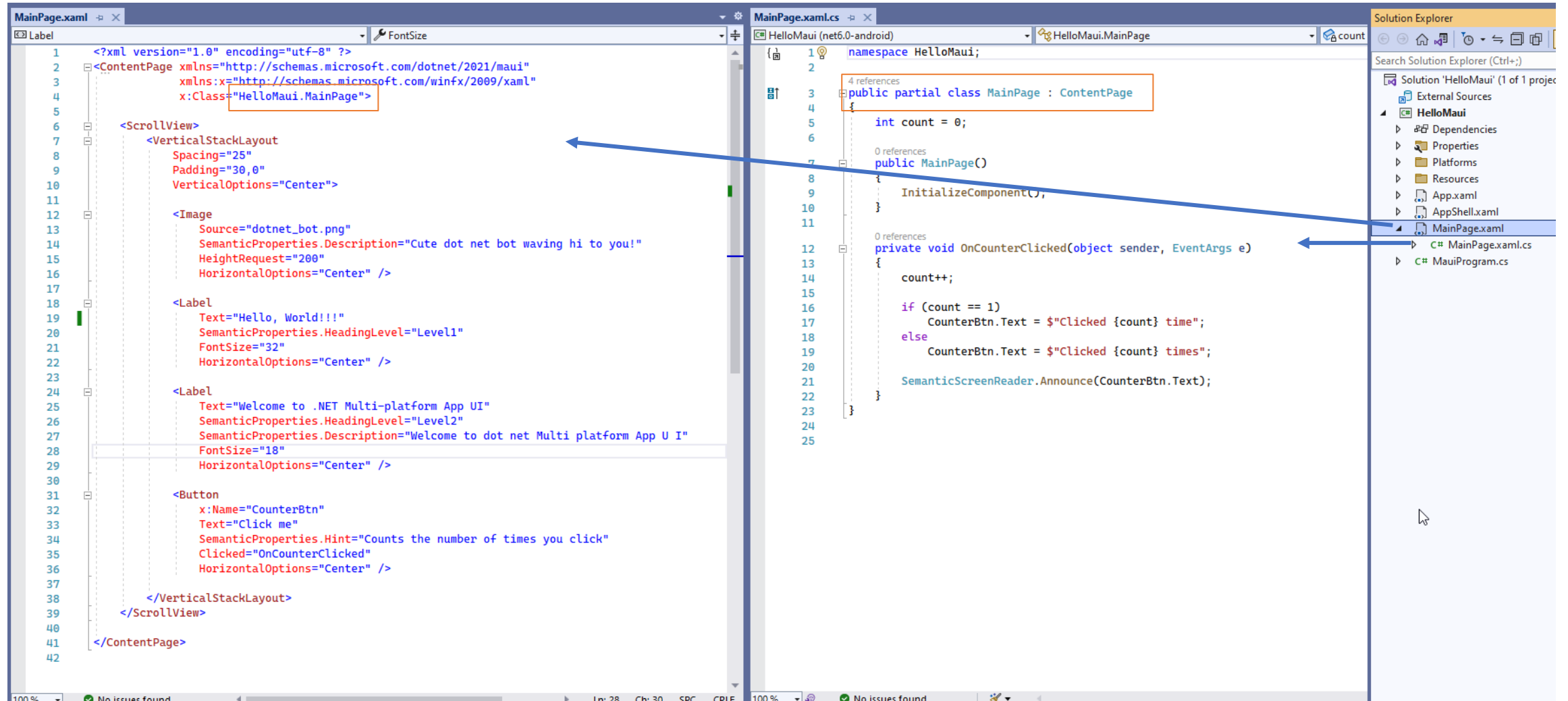
    void LoginButton_Clicked(object sender, EventArgs e)
    {
        Debug.WriteLine("Clicked !");
    }
}
```


XAML files in .NET MAUI

- Elke file bestaat eigenlijk uit een koppel: .xaml en .xaml.cs
 - .xaml bevat XAML code
 - .xaml.cs bevat de achterliggende C# code

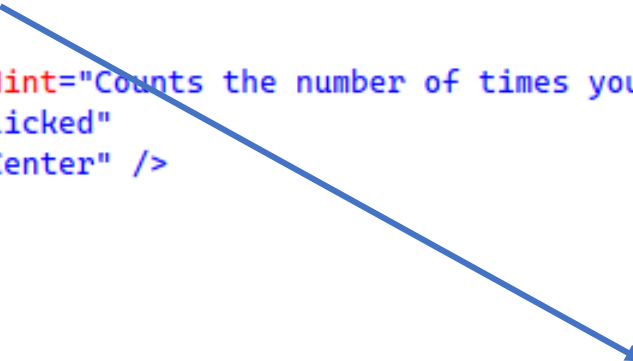


XAML in .NET MAUI



XAML basics

```
<Button  
  x:Name="CounterBtn"  
  Text="Click me"  
  SemanticProperties.Hint="Counts the number of times you click"  
  Clicked="OnCounterClicked"  
  HorizontalOptions="Center" />
```




'x:' is de XAML-namespaces die
bovenaan gedefinieerd is
(Vergelijkbaar met 'using' in C#)

Via x:Name kunnen we elementen
een naam geven om ze in de code
aan te spreken!

XAML basics

```
<Button
    x:Name="CounterBtn"
    Text="Click me"
    SemanticProperties.Hint="Counts the number of times you click"
    Clicked="OnCounterClicked"
    HorizontalOptions="Center" />
```



Via bijvoorbeeld de 'Clicked' property kunnen we ervoor zorgen dat bepaalde functies worden opgeroepen in de achterliggende code.

```
0 references
private void OnCounterClicked(object sender, EventArgs e)
{
    count++;

    if (count == 1)
        CounterBtn.Text = $"Clicked {count} time";
    else
        CounterBtn.Text = $"Clicked {count} times";

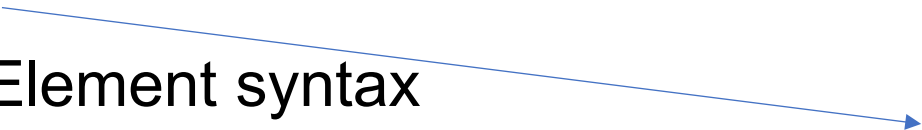
    SemanticScreenReader.Announce(CounterBtn.Text);
}
```

Properties

- In XAML zijn er verschillende soorten properties:
 - Attributes
 - Property Element syntax
 - Attached properties

Properties

- In XAML zijn er verschillende soorten properties:
 - Attributes
 - Property Element syntax
 - Attached properties



```
<Label Text="Hello, XAML!"  
        VerticalOptions="Center"  
        FontAttributes="Bold"  
        FontSize="18"  
        TextColor="Aqua" />
```

Via de XML-attributen van het Label-element kunnen we allerlei dingen instellen

Property Element Syntax

- Wat is het verschil tussen deze twee fragmenten?

```
<Label Text="Username" TextColor=■"Black" />
```

```
<Label>  
  <Label.Text>Username</Label.Text>  
  <Label.TextColor>■Black</Label.TextColor>  
</Label>
```


Property Element Syntax

- Waarom?
 - Soms kunnen we iets niet **inline** schrijven:

```
<Grid>
  <Grid.RowDefinitions>
    <RowDefinition Height="2*" />
    <RowDefinition Height="*" />
    <RowDefinition Height="100" />
  </Grid.RowDefinitions>
  <Grid.ColumnDefinitions>
    <ColumnDefinition Width="*" />
    <ColumnDefinition Width="*" />
  </Grid.ColumnDefinitions>
</Grid>
```

Dit kunnen we niet eenvoudiger schrijven!

~~<Grid RowDefinitions="???">~~

Attached Properties

- Aanspreken properties van andere elementen

```
<Grid>
  <Grid.RowDefinitions>
    <RowDefinition Height="2*" />
    <RowDefinition Height="*" />
    <RowDefinition Height="100" />
  </Grid.RowDefinitions>
  <Grid.ColumnDefinitions>
    <ColumnDefinition Width="*" />
    <ColumnDefinition Width="*" />
  </Grid.ColumnDefinitions>
  <Label Text="I am on the 1 row, second column"
        Grid.Row="0"
        Grid.Column="1" />
</Grid>
```

Het Label 'vraagt' aan het Grid of hij op een bepaalde plaats mag staan

Content properties

- Property die aangeeft wat **getoond** zal worden in een element
- “Hetgeen tussen de open en close tag staat”
- Voorbeeld met Label:
 - De *Text*-property is de *Content property*

```
<Label Text="Please log in" />
```

of

```
<Label>  
  <Label.Text>Please log in</Label.Text>  
</Label>
```



```
<Label>  
  Please log in  
</Label>
```

Content properties

- Source-code Label:



```
namespace Microsoft.Maui.Controls
{
    //
    // Summary:
    //     A Microsoft.Maui.Controls.View that displays text.
    //
    // Remarks:
    //     A Label is used to display single-line text elements as well as multiple lines
    //     of text.
    //     The following example, adapted from the default Microsoft.Maui.Controls
    //     shows a basic use:
    //     public class App : Application
    //     {
    //         public App ()
    //         {
    //             MainPage = new ContentPage {
    //                 Content = new Label {
    //                     Text = "Hello, Forms!",
    //                     FontSize = Device.GetNamedSize (NamedSize.Large, typeof (Label)),
    //                     VerticalOptions = LayoutOptions.CenterAndExpand,
    //                     HorizontalOptions = LayoutOptions.CenterAndExpand,
    //                 },
    //             };
    //         }
    //     }
    //     The FormsGallery sample, which can be found on the Sample Applications
    //     has a LabelDemoPage.cs file. This file contains a longer and more
    //     [ContentProperty("Text")]
    public class Label : View, ILabel, IView, IElement, ITransform, IText, ITextElement
    {
        //
    }
}
```

Content properties

- Werkt dit ook met bijvoorbeeld een Button?

```
<Button
    Clicked="LoginButton_Clicked"
    x:Name="LoginButton">
    Log In
</Button>
```

Content properties

- Soms kan je default zaken weglaten

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://schemas.microsoft.com/dotnet/2021/maui"
  xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
  x:Class="HelloMaui.MainPage">
  <ContentPage.Content>
    <ScrollView>
      <VerticalStackLayout
        Spacing="25"
      />
    />
  />
</ContentPage>
```



```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://schemas.microsoft.com/dotnet/2021/maui"
  xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
  x:Class="HelloMaui.MainPage">
  <ScrollView>
    <VerticalStackLayout
      Spacing="25"
    />
  />
</ContentPage>
```

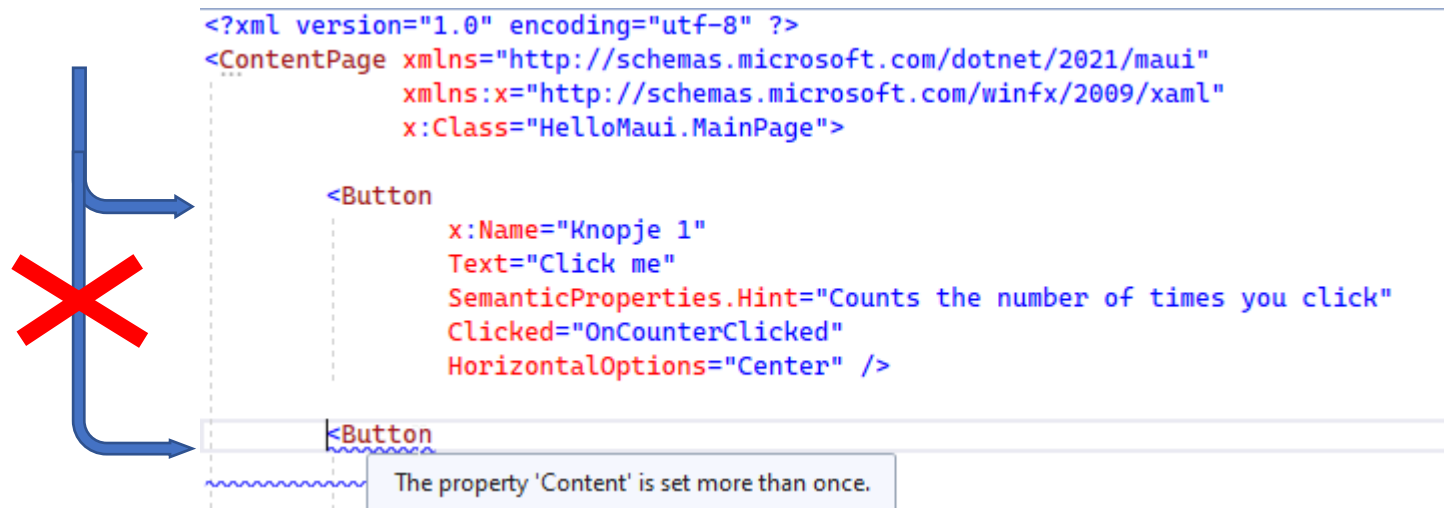


```
//      which is a typical, though basic, use of the Xam
//      The FormsGallery sample, which can be found on t
//      has a ContentPageDemoPage.cs file. This file con
//      example.
```

```
[ContentProperty("Content")]
public class ContentPage : TemplatedPage
```

Nesting

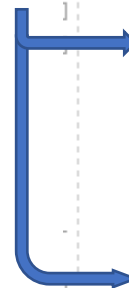
- Veel elementen in .NET MAUI kunnen maar één child-element hebben
- Voorbeeld: ContentPage
 - => Dus er kan maar één knopje getoond worden?



Nesting

- Veel elementen in XAML en XAML.Forms kunnen maar één *child-element* hebben
- Oplossing: Nesting
 - We gebruiken als *child-element* een element dat wél meerdere *children* kan hebben
 - Bijvoorbeeld: StackLayout, Grid, ...

Nesting



```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://schemas.microsoft.com/dotnet/2021/maui"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="HelloMaui.MainPage">
  <StackLayout>
    <Button
      x:Name="Knopje 1"
      Text="Click me"
      SemanticProperties.Hint="Counts the number of times you click"
      Clicked="OnCounterClicked"
      HorizontalOptions="Center" />
    <Button
      x:Name="Knopje 2"
      Text="Click me"
      SemanticProperties.Hint="Counts the number of times you click"
      Clicked="OnCounterClicked"
      HorizontalOptions="Center" />
  </StackLayout>
</ContentPage>
```

Commentaar

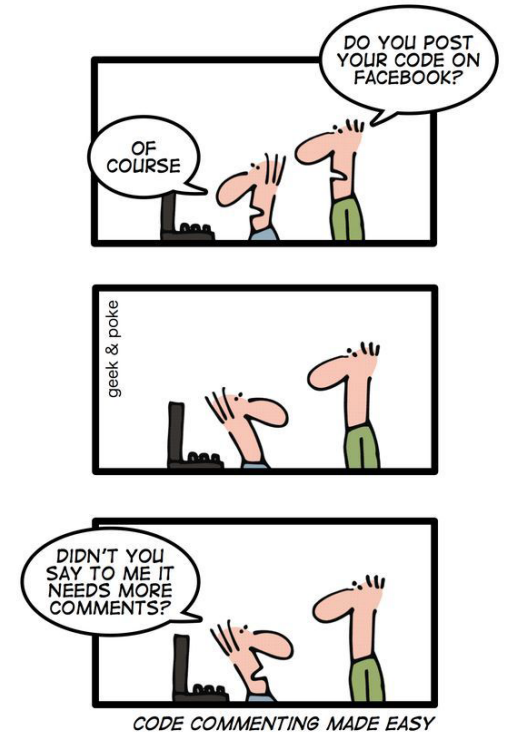
- in XAML:

Zelfde als in HTML: `<!-- Hiertussen commentaar -->`

- in C#:

`// Commentaar op 1 regel`

`/* Commentaar op meerdere regels */`

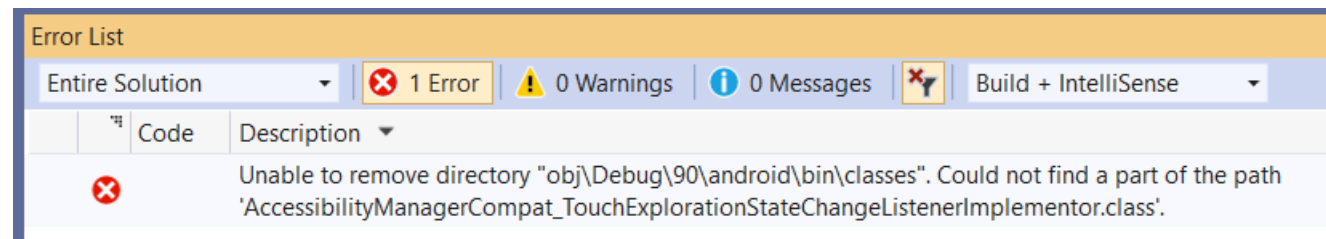


Tegen volgende les

- .NET MAUI 8 geïnstalleerd in Visual studio 2022
- Werkende Android emulator of je eigen device als debug apparaat
- 'Hello world' tutorial doorlopen en werkend op (virtueel) device
- Zie Opdracht 1 uit de oefeningen bundel

Gotcha: Could not find a part of the path

- Vroeger een probleem (potentieel nu nog):
 - Android project compileert niet: 'Could not find part of the path...'



- Oplossing:
 - Het path van de solution is te lang. Plaats je project hoger in de bestandshiërarchie (bv: in C:\Projecten)

Extra: online introductiecursus .NET MAUI

- https://www.youtube.com/watch?v=Hh279ES_FNQ&list=PLdo4fOcmZ0oUBAdL2NwBpDs32zwGqb9DY&index=1&ab_channel=dotNET
- <https://app.pluralsight.com/library/courses/dot-net-maui-big-picture/table-of-contents>