

leren. durven. doen.



C# FUNDAMENTALS

WPF UI

WPF motivatie

- **UI layout en design wordt gescheiden van implementatie functionaliteit en business logica**
- **XML-based Markup language (XAML) voor design, programming language (C#, VB, etc) voor logica in programmeercode**
- **Designers en developers kunnen apart werken op hetzelfde project in andere soorten tools :**
 - Expression Blend voor designers**
 - Visual Studio voor developers**

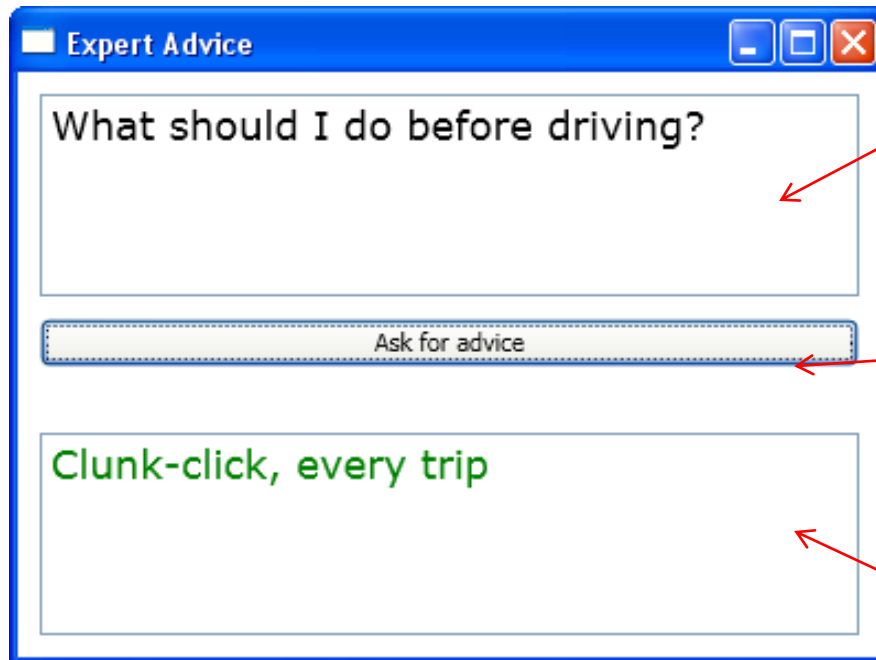
WPF motivatie

- **Dezelfde technologie gebruiken, gebaseerd op XAML en C#/VB, dat gebruikt kan worden voor verschillende soorten projecten:**
 - Windows (WPF)
 - Web (Silverlight)
 - Mobile (Xamarin)
- **Basisidee is om een GUI te maken en gebruik te maken van markup-taal (Xaml) in combinatie met programmeercode, is gelijkaardig aan idee voor web development technologieën, e.g.**
 - HTML & JavaScript
 - ASP.NET & C#

Soorten XAML controls

- **Layout controls**
containers voor andere controls om deze in the User interface te kunnen positioneren
<Frame>, <Grid>, <StackPanel>, ...
- **Interactieve controls**
<Button>, <ComboBox>, <Slider>, ...
- **Display controls**
<Label>, <ListBox>, <Image>, ...
- **Data controls**
<DataGrid>, <ListView>, ...
- **Application controls**
<Menu>, <ToolBar>, ...

Een eenvoudig WPF voorbeeld



text box – gebruiker kan hier tekst ingeven

button – gebruiker kan hier op klikken

Textblock – hier kan een boodschap worden getoond

XAML window

Window gedefineerd in XAML bestand

- **Voorbeeld: Grid control is een container voor andere controls**

```
<Window x:Class="Advice.Advisor"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="Expert Advice" Height="300" Width="400" >
  <Grid>
    <Grid.RowDefinitions>
      <RowDefinition Height="*" />
      <RowDefinition Height="Auto" />
      <RowDefinition Height="*" />
    </Grid.RowDefinitions>
```

Grid heeft 3 rijen, de middelste rij zal zich aanpassen aan zijn inhoud, de andere 2 passen zich aan tot de beschikbare ruimte

Andere controls worden binnen de <Grid> control gedefinieerd

```
</Grid>
</Window>
```

Controls toevoegen aan een window

naam van methode die het click event afhandelt (Event Handler)

```
<TextBox Margin="10,10,10,10" Name="txtQuestion"
    TextWrapping="Wrap" FontFamily="Verdana" FontSize="18"
    Grid.Row="0" >
    [What's your problem?]
</TextBox>
<Button Margin="10,0,10,20" Name="cmdAnswer"
    Click="cmdAnswer_Click"
    Grid.Row="1">
    Ask for advice
</Button>
<TextBox Margin="10,10,10,10" Name="txtAnswer"
    TextWrapping="Wrap" FontFamily="Verdana" FontSize="18"
    Foreground="Green"
    Grid.Row="2">
</TextBox>
```

Grid.Row attribuut specificeert in welke row van de grid de control zal worden getoond

Attributen controleren de presentatie van controls (fonts, margins, ...)

Code-behind bestand: .xaml.cs

- Bevat C# (partial) class die afgeleid is van Window built-in class

```
public partial class Advisor : Window
{
    public Advisor()
    {
        InitializeComponent();
    }

    private void cmdAnswer_Click(object sender, RoutedEventArgs e)
    {
        AdviceGenerator generator = new AdviceGenerator();
        txtAnswer.Text = generator.GetRandomAnswer(txtQuestion.Text);
    }
}
```

constructor

event handler methode

event handler methode gebruikt bv class *AdviceGenerator* en zet de Text property van de text box *txtAnswer*

Applicatie en windows

- **App.xaml is het startup bestand voor een WPF project**

```
<Application x:Class="WindowsApplication1.App"
  xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
  StartupUri="Advisor.xaml"
>
  <Application.Resources>

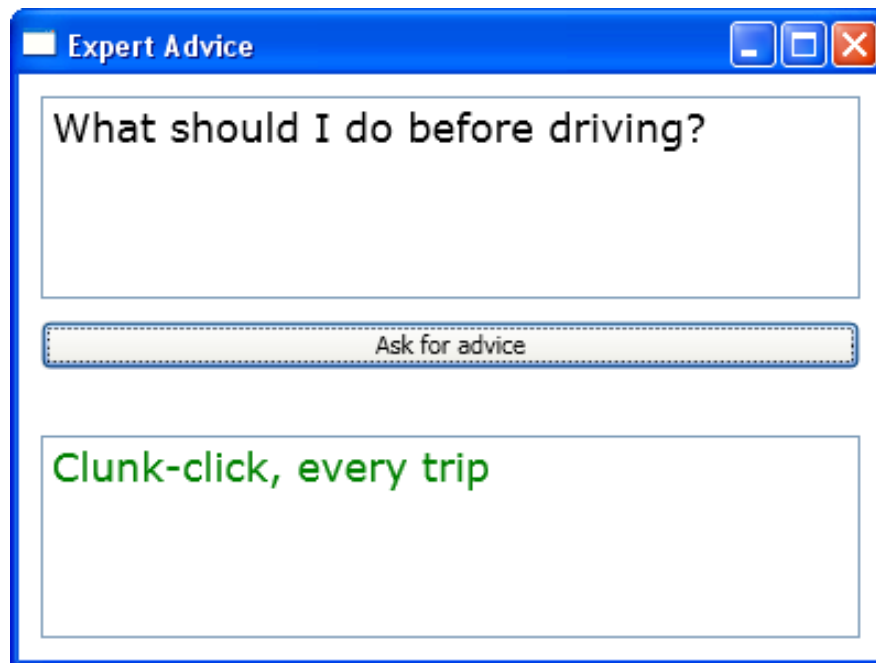
  </Application.Resources>
</Application>
```

window die wordt geopend bij startup

- **Code-behind van App.xaml bevat een lege constructor by default**
- **Waar is de Main method?**
Antw: In autogenerated code in file **App.g.cs** in **obj** folder
- **Additionele windows worden gedefinieerd in aparte XAML files**
Kunnen een instantie aanmaken van een code-behind class en hun Show methode om een nieuw window te openen

Oefening XAML

- Maak dit venster na met behulp van vorige slides



Code en visual designers

- **WPF windows kunnen designed worden dr gebruik te maken van visual design tools in Visual Studio en Expression Blend**
- **Via XAML-markup taal kan men de UI-design maken**
- **Maak gebruik van XAML om layout controls beter te positioneren (slepen van controls van designer toolbox op window, zal geen goed resultaat geven wanneer window wordt resized (vanwege absolute position))**

Zo is het mogelijk via XAML om layouts te maken die goed tonen, ook wanneer de window resized wordt.

Layout controls (containers voor andere controls)

- **Grid**

Zal de child-controls in een tabelvorm structureren

- **Stack Panel, Wrap Panel**

Stack Panel zal de child-controls naast of onder elkaar positioneren, Wrap Panel zal child-controls naar een nieuwe lijn laten springen indien ruimte tekort is

- **Dock Panel**

Zal zijn elementen 'docken' aan zijn linker, rechter, top, bodemkant of in zijn centrum
















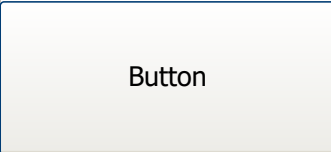
- **Canvas**

Zal zijn elementen positioneren op basis van coördinaten, wordt meestal gebruikt voor 2D drawing

Alignment (Aligneren)

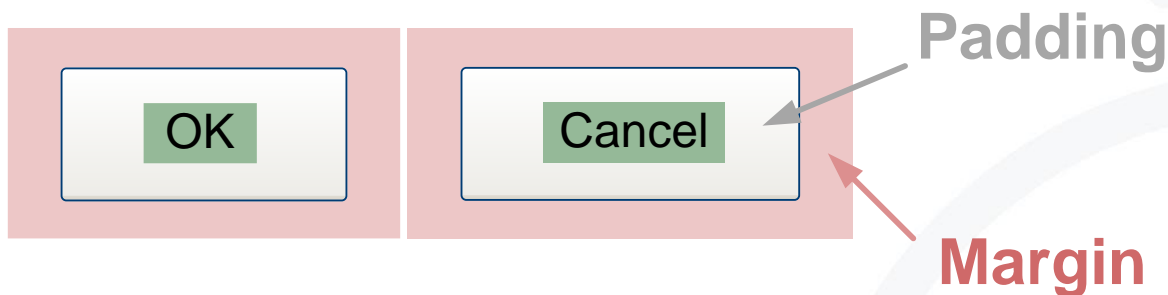
Horizontal Alignment

Vertical Alignment

		Left	Center	Right	Stretch
Vertical Alignment	Top				
	Center				
	Bottom				
	Stretch				

Margin en padding

- De Margin is extra ruimte rond de control
- De Padding is extra ruimte binnen de control
- De Padding van een outer control is de Margin van een inner control



Laying out grid

- **Row en column definities**
- **Sizes:**
 - Fixed:** Vaste grootte
 - Auto:** Control neemt zoveel ruimte in als nodig
 - Star (*):** Neemt zoveel ruimte in als mogelijk
- **Positioneer elke control in de grid via Grid.Column and Grid.Row**
- **Merge grid cellen met Grid.ColumnSpan and Grid.RowSpan**

WPF properties

- **Normale .NET properties**

Value wordt gelezen uit member field in class

- **Dependency properties**

Worden dynamisch gelezen/geschreven, bv dr binding, laat o.a. toe om:

Change notifications te sturen/ontvangen

Overerving van parent elementen

Veel XAML control-properties zijn eigenlijk dependency properties

- **Attached properties**

Laten toe om een value aan een object te hangen

Een child element kan bv een value bijhouden die geassocieerd is met een property die gedefinieerd is voor de parent element

MainWindow

Auto	1*	Auto
House number:	<input type="text"/>	
Post Code:	<input type="text"/>	Look up
Address:	<input type="text"/>	
Check this box to keep details private: <input type="checkbox"/>		

MainWindow

House number:	<input type="text"/>	
Post Code:	<input type="text"/>	Look up
Address:	<input type="text"/>	
Check this box to keep details private: <input type="checkbox"/>		

Layout Voorbeeld - Grid

```
<Grid Margin="3,3,10,3">
  <Grid.RowDefinitions>
    <RowDefinition Height="Auto"></RowDefinition>
    <RowDefinition Height="Auto"></RowDefinition>
    <RowDefinition Height="*"></RowDefinition>
    <RowDefinition Height="Auto"></RowDefinition>
  </Grid.RowDefinitions>
  <Grid.ColumnDefinitions>
    <ColumnDefinition Width="Auto"></ColumnDefinition>
    <ColumnDefinition Width="*"></ColumnDefinition>
    <ColumnDefinition Width="Auto"></ColumnDefinition>
  </Grid.ColumnDefinitions>
```

4 rijen, 3 kolommen

Layout voorbeeld - controls

Column="0" bevat niets voor Row="0"

```
<Label Grid.Row="0" Grid.Column="0" Margin="3"
    VerticalAlignment="Center">House number:</Label>
<TextBox Grid.Row="0" Grid.Column="1" Margin="3"
    Height="Auto" VerticalAlignment="Center"></TextBox>
<Label Grid.Row="1" Margin="3"
    VerticalAlignment="Center">Post Code:</Label>
<TextBox Grid.Row="1" Grid.Column="1" Margin="3"
    Height="Auto" VerticalAlignment="Center"></TextBox>
<Button Grid.Row="1" Grid.Column="2" Padding="2"
    VerticalAlignment="Center">Look up</Button>
<Label Grid.Row="2" Margin="3"
    VerticalAlignment="Top">Address:</Label>
<TextBox Grid.Row="2" Grid.Column="1" Margin="3"
    Height="Auto" VerticalAlignment="Stretch"></TextBox>
<StackPanel Orientation="Horizontal" Grid.Row="3" Grid.Column="0"
    Grid.ColumnSpan="2">
    <Label Margin="3" VerticalAlignment="Center">
        Check this box to keep details private:</Label>
    <CheckBox Margin="3"
        Height="Auto" VerticalAlignment="Center"></CheckBox>
</StackPanel>
</Grid>
```

Oefening XAML Grid container

- Maak dit venster na met behulp van vorige slides

The screenshot shows a WPF application window titled "MainWindow". Inside the window, there is a form with the following elements:

- A label "House number:" followed by a single-line text input field.
- A label "Post Code:" followed by a single-line text input field.
- A label "Address:" followed by a multi-line text area.
- A "Look up" button positioned to the right of the Post Code input field.
- A checkbox at the bottom with the text "Check this box to keep details private:".