.NET and C#

Windows Presentation Foundation (WPF)

Indhold

- Windows Presentation Foundation (WPF)
  - XAML
  - Controls
  - Containers
  - EventHandlers
- Events in WPF
  - Raise and Handle events
  - Attaching event handler at Design-time
  - Attaching event handler at Run-time
  - Event types

XAML

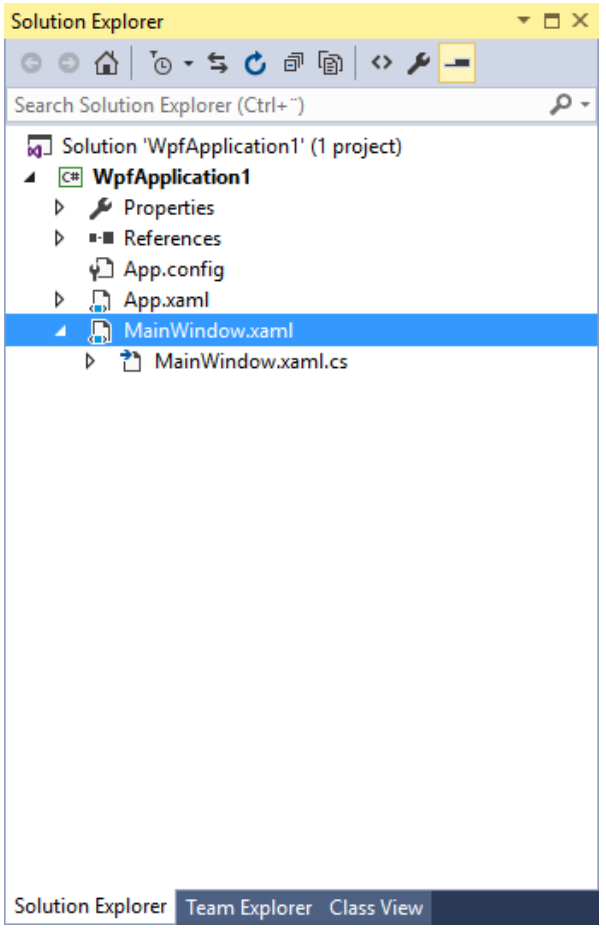
- Udtalt ”zammel”
- XML
- Adskiller GUI fra code

WPF Project

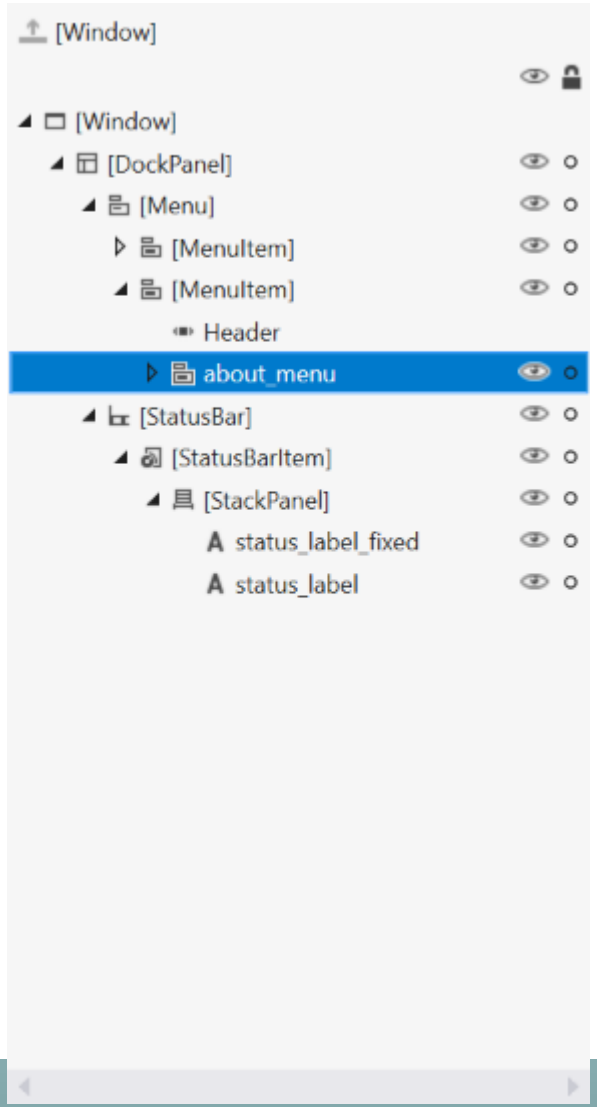
Et nyt WPF project I Visual Studio vil lave en MainWindow.xaml file, som er start vinduet for applikationen.

En MainWindow.xaml.cs fil bliver også lavet. Det er koden bagved for xaml filen. .

```
namespace WpfApplication1
{
    /// <summary>
    /// Interaction logic for MainWindow.xaml
    /// </summary>
    public partial class MainWindow : Window
    {
        public MainWindow()
        {
            InitializeComponent();
        }
    }
}
```



XAML træstruktur – Document Outline i VS



Opgave

- Opgave 6.1

Controls

- Button
- CheckBox
- ComboBox
- RadioButton
- Text-controls
- Containers
  - Grid
  - StackPanel
  - WrapPanel
  - DockPanel
  - Canvas
  - For en oversigt over containers se f.eks. https://www.codeproject.com/articles/140613/wpf-tutorial-layout-panels-containers-layout-trans (den er ret gammel, men holder vist stadig)

Text Controls

Text Controls

- TextBox
- RichTextBox
- PasswordBox

Text

- Label
- TextBlock

FORSKELLIGE GUI PROPERTIES OG COMPONENTS

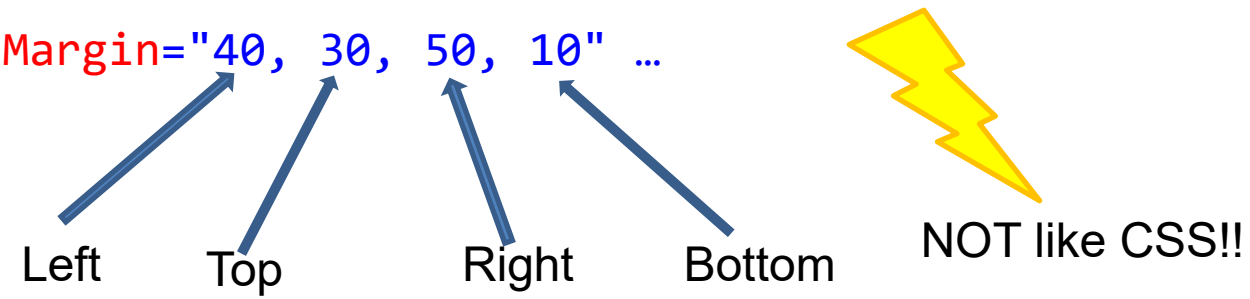
- Layout Properties

  - Margin
  - Padding
  - HorizontalAlignment
  - VerticalAlignment
  - Min/MaxWidth
  - Min/MaxHeight
  - Width, Height

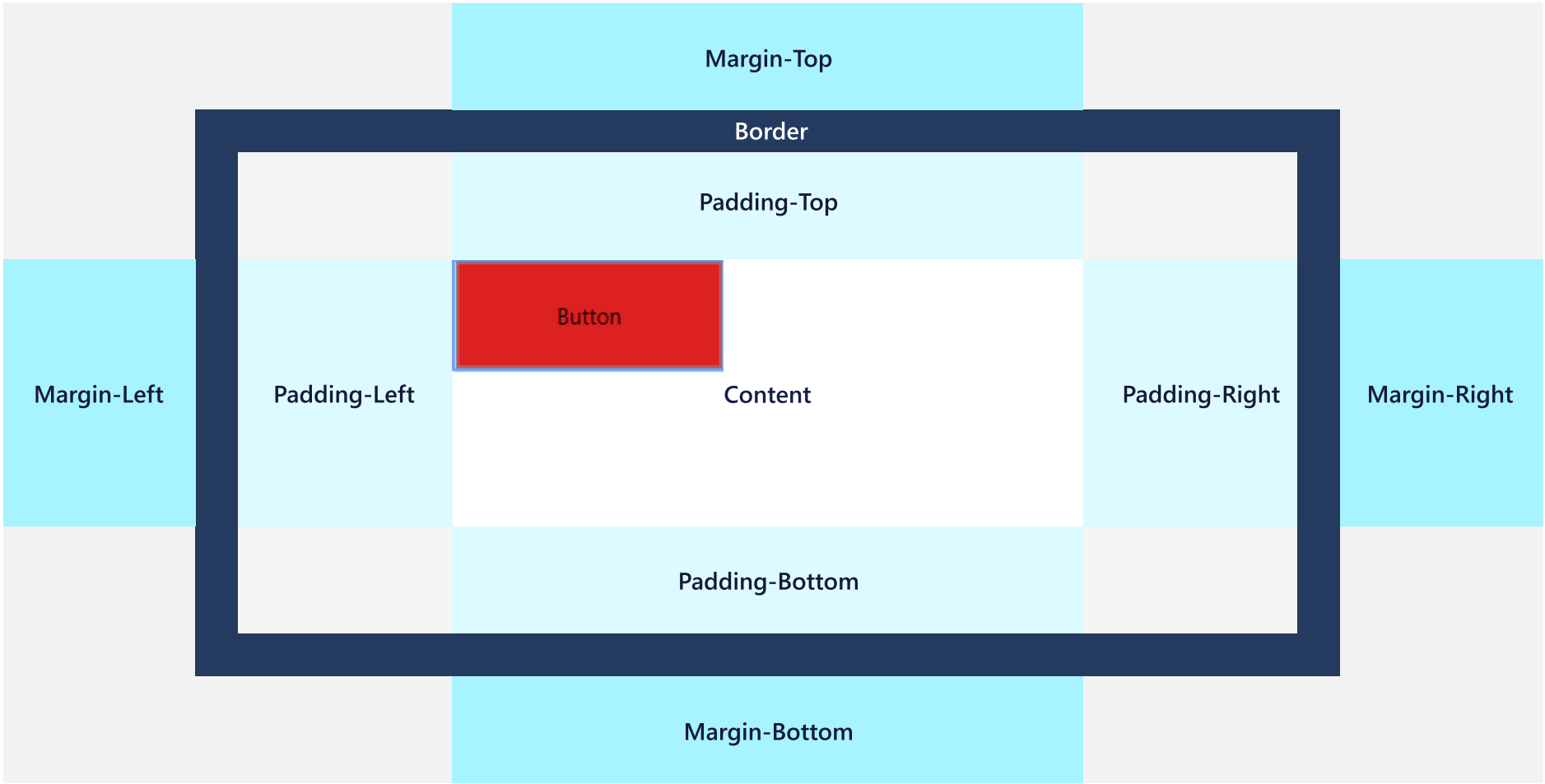
- Margin, Padding

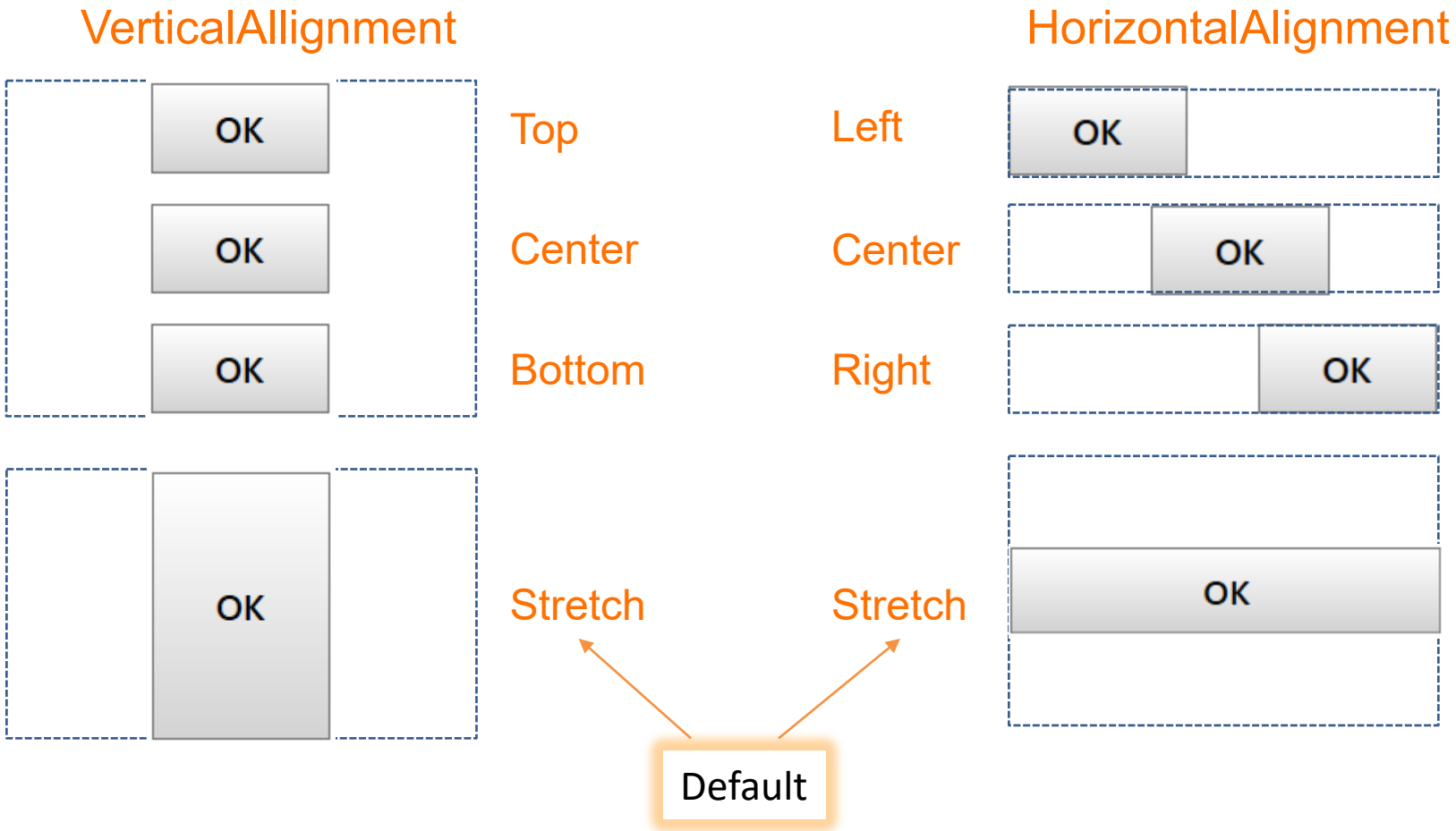
```
<Button Margin="20" …
```

```
<Button Margin="40, 30, 50, 10" …
```



Margin vs padding




VerticalAllignment
HorizontalAlignment
OK
OK
OK
Top
Center
Bottom
Left
Center
Right
OK
OK
OK
OK
Stretch
Stretch
Default

- Alignment

En advarsel ....

Pas når du sætter Height og Width properties specifikt til pixels, da elementet ikke vil automatisk resize, hvis content ændrer størrelse.

En klassisk fejl ....

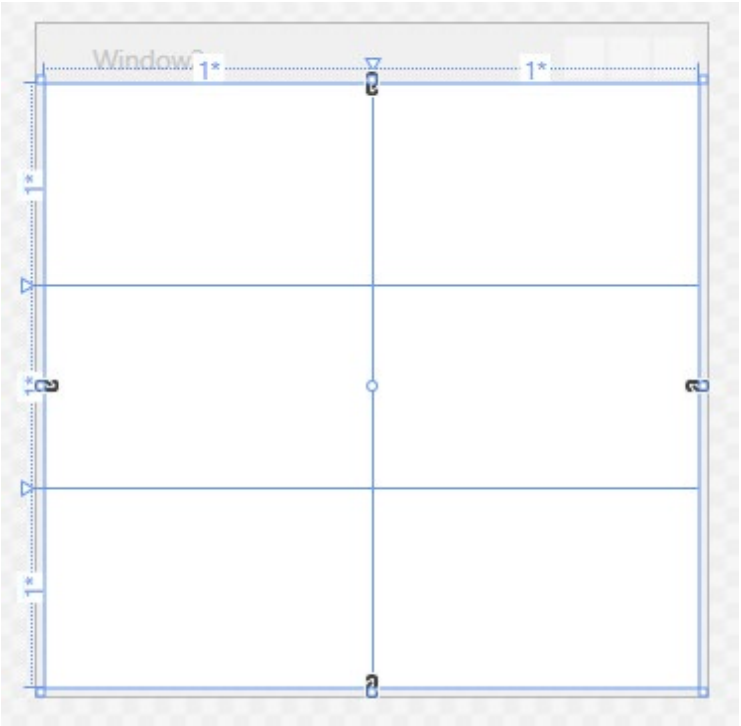
XAML - The property 'Content' is set more than once

Dette sker fordi at nogle components, KUN tillader et child element og du prøver at sætte to ind. I dette tilfælde skal du så bruge en ”collection” component og så sætte de to childs ind der. F.eks. Kan du bruge et StackPanel eller Grid eller WrapPanel som alle kan indeholder flere børn – f.eks. flere knapper. **XAML er en TRÆ-STRUKTUR – se næste slide**

Layouts - Grid

```
<Grid>
   <Grid.RowDefinitions>
      <RowDefinition/>
      <RowDefinition/>
      <RowDefinition/>
    </Grid.RowDefinitions>
    <Grid.ColumnDefinitions>
       <ColumnDefinition/>
       <ColumnDefinition/>
    </Grid.ColumnDefinitions>
</Grid>
```

2 x 3 Grid



RowDefinition/ColumnDefinition

- Auto – Height og width er defineret udfra content
- Fixes Size – Height og width har fast størrelse
- Star Notation – Relative size

Grid.Row og Grid.Column

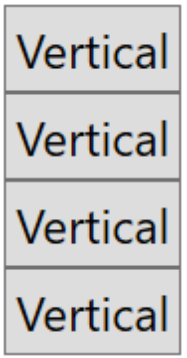
- Definerere i hvilken row and column en control er
- <TextBox Grid.Column=”2" Grid.Row="1"/>

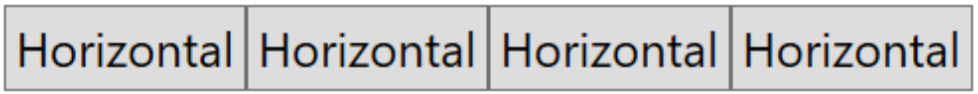
Spanning more rows or columns

- Grid.RowSpan
- Grid.ColumnSpan

StackPanel

- Orientation
  - Horizontal
  - Vertical (Default)





Nyttigt: StackPanels kan ”nestes” inden i et andet stackpanel og orientation kan ændres. Se WpfApp2 StackPanel eksemplet og de næste par slides

- Layouts - StackPanel

Controls

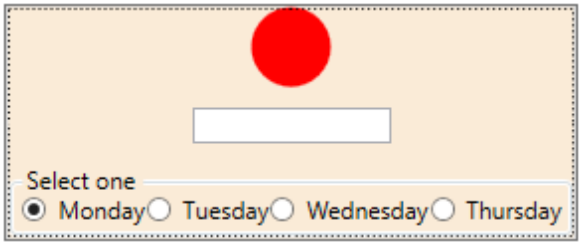
- Button
- CheckBox
- RadioButton

```
<StackPanel>
        <Button HorizontalAlignment="Left" Padding="10" Margin="20">Hello</Button>
        <StackPanel Orientation="Horizontal">
            <CheckBox Margin="10, 10, 0, 0"></CheckBox>
            <Label Margin="0, 5">Select me</Label>
        </StackPanel>
        <GroupBox Header="Select a size" Padding="5" Margin="10"
                  HorizontalAlignment="Left">
            <StackPanel>
                <RadioButton Margin="3" Content="Small" />
                <RadioButton Margin="3" Content="Medium" />
                <RadioButton Margin="3" Content="Large" IsChecked="True" />
            </StackPanel>
        </GroupBox>
</StackPanel>
```

Scale checkbox: https://stackoverflow.com/questions/13481275/how-to-make-checkbox-bigger

WPF style ….nestede elementer

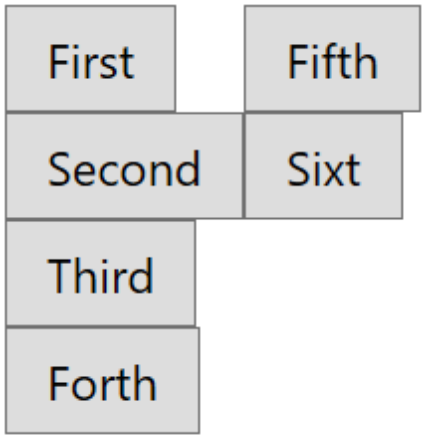
```
<StackPanel Background="AntiqueWhite">
                <Ellipse Height="40" Width="40" Fill="Red" />
                <TextBox Width="100" Margin="10"/>
                <GroupBox Header="Select one">
                    <StackPanel Orientation="Horizontal">
                        <RadioButton Content="Monday" IsChecked="True"/>
                        <RadioButton Content="Tuesday"/>
                        <RadioButton Content="Wednesday"/>
                        <RadioButton Content="Thursday"/>
                    </StackPanel>
                </GroupBox>
            </StackPanel>
```

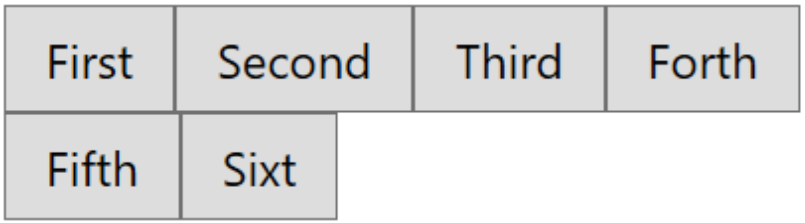


- Layout - WrapPanel

WrapPanel

- Orientation
  - Horizontal (Default)
  - Vertical

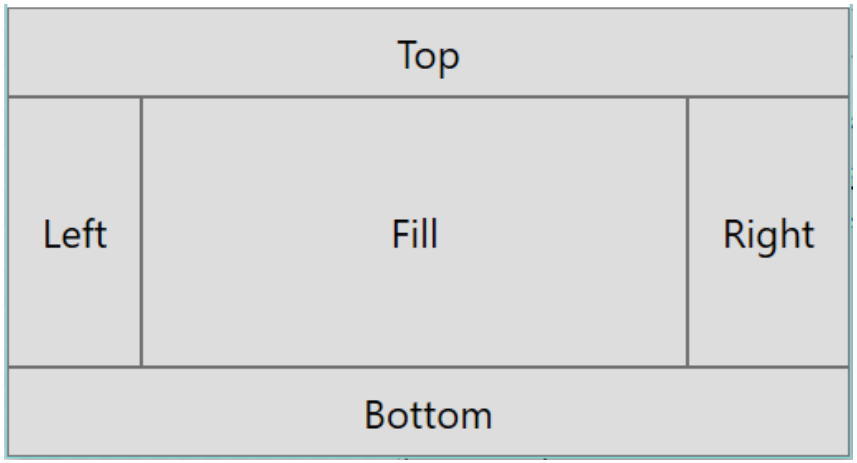




Layout - DockPanel

DockPanel

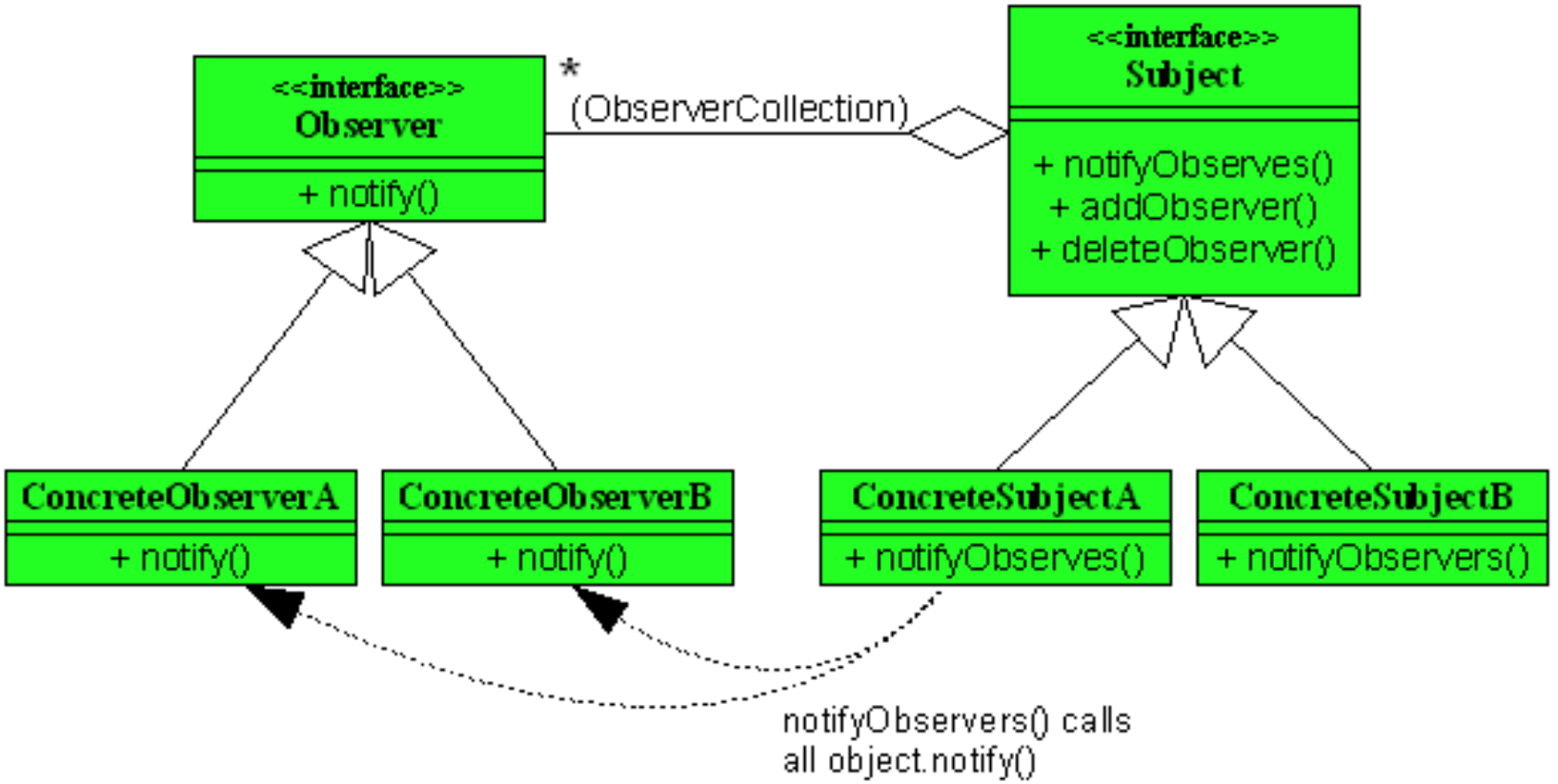
- DockPanel.Dock
  - Left
  - Right
  - Top
  - Bottom



Events in C#

En implementation af Observer-pattern (recap)

Observer pattern – tradionelt implementeret.



Events i WPF og Observer-pattern

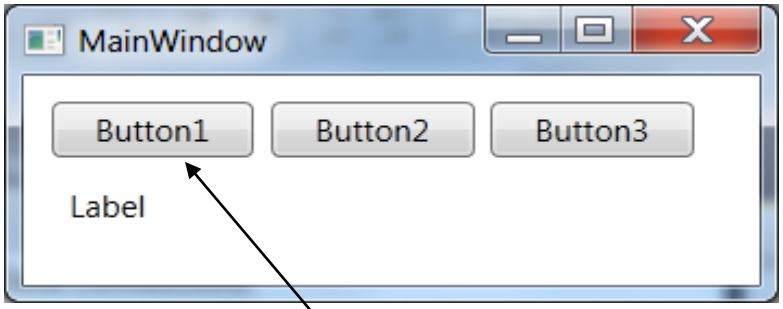
- events i C# er en elegant og fleksibel måde at implementere Observer-pattern på.
- Det kræver ikke to interfaces og disses implementation, men kun:
  - 1 linjes erklæring af delegate typen
  - En simpel erklæring af en event variabel, som så skal være af delegate type.
  - Selve eventhandleren – som så også skal være af samme delegate type. Det er denne kode, som bliver kaldt, når et event udløses.

Events i WPF

Observer-pattern med delegates/events

Programming med events

- Et object kan "raise et *event*".
- Når en event er raised, så vil en eller flere metoder bliver kaldt. Disse kaldes *eventhandlers*.



button1 raiser et Click-event,
når brugeren trykker på knappen

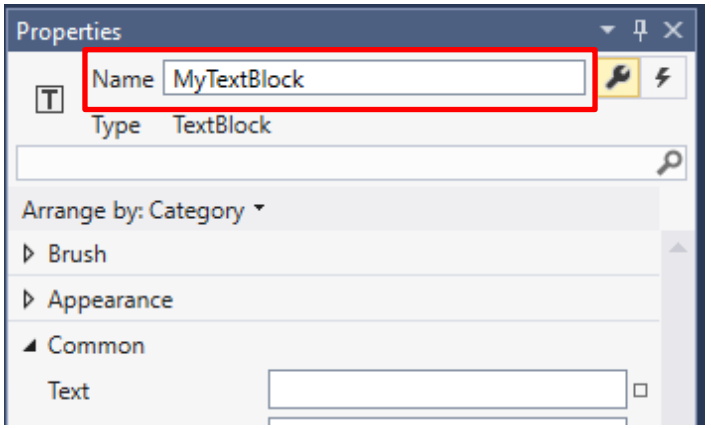
```
public partial class MainWindow : Window  {
   ..
   public void button1_Click(..) {
      ..
   }
}
```

MainWindow.button1_Click(…) kaldes så når button1's click event er raised, fordi det er en eventhandler, der er add'et til click-delegaten

- Naming controls

Hvis man skal referere til en control skal den have et Name

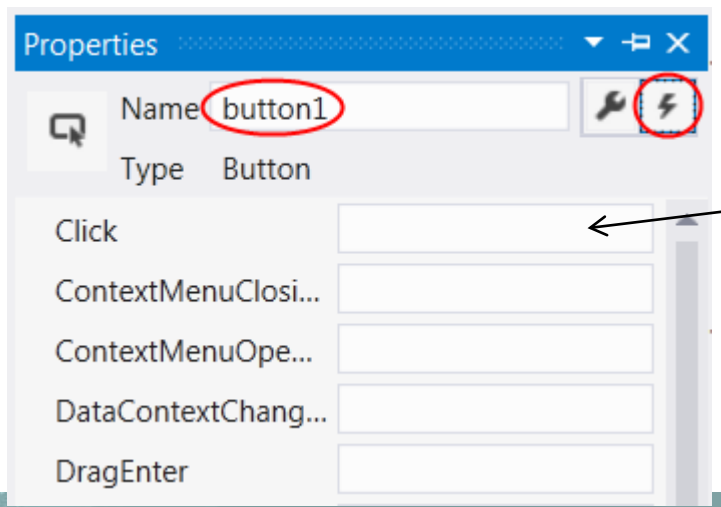
```
<TextBlock x:Name="MyTextBlock">
```




```
private void Button_Click(object sender, RoutedEventArgs e)
{
   MyTextBlock.Text = "New Text";  }
```

Tilføj en event handler til en Button's Click på design time – to måder at gøre det på i GUI

1. DoubleClick på knappen (fordi Click er default event for en Button)
2. DoubleClick i PropertyWindow



Bemærk!! Skifter mellem Properties og EventHandlers

DoubleClick her

Man kan selvfølgelig også editere direkte i xaml filen i stedet for.

Man kan også ændre direkte i XML

3. Eventhandleren tilføjes i XAML (sker automatisk)

```
<Button Name="BtnAdd" ... Click="button1_Click"/>
```

Lille opgave

- Opgaver 6.3 + 6.4

Tilføje en eventhandler via kode (runtime)

```
button1.Click += MyButton_Click;
button2.Click += MyButton_Click;

...

private void MyButton_Click(
          object sender, RoutedEventArgs e)
{
   ...
}
```

Gui-events

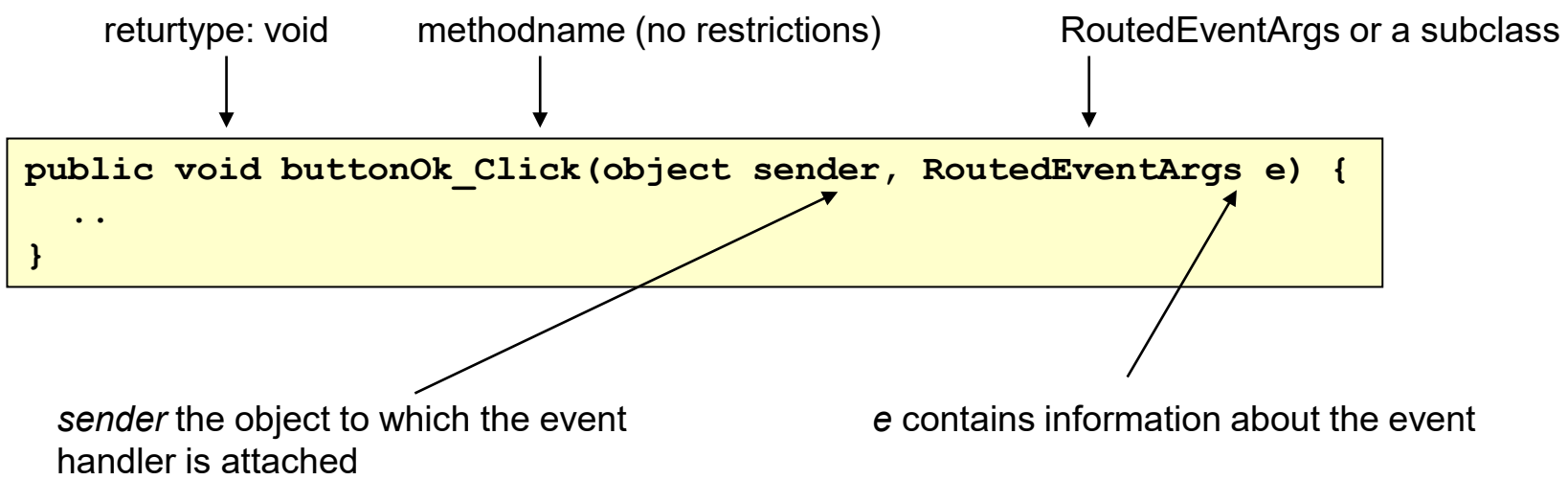
- Hver GUI komponent har sine egne events
  - Button har Click, MouseLeftButtonDown, MouseMove, MouseEnter, MouseLeave etc;
  - Textbox har TextChanged, KeyUp, KeyDown MouseLeftButtonDown, GotFocus, LostFocus etc.

- Når en bruger klikker på knappen, så bliver Click eventet i klassen udført:

```
Button1.Click(sender,e); // invisible code
```

Eventhandlers i WPF

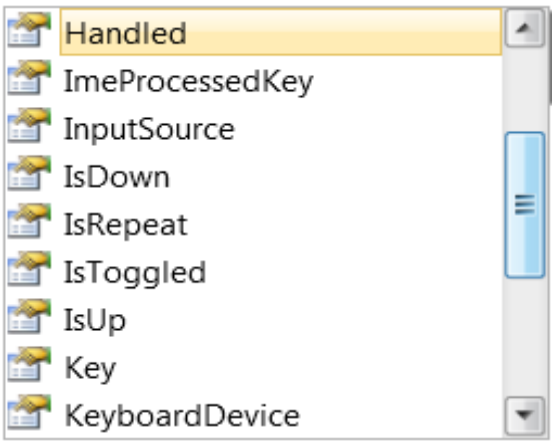
- Eventhandlers i WPF følger en bestemt skabelon:



Den sidste parameter i eventhandlers er EventArgs – dvs. ekstra info om eventen

```
private void textBox1_KeyDown(
    object sender, KeyEventArgs e) { ... }
```

e.

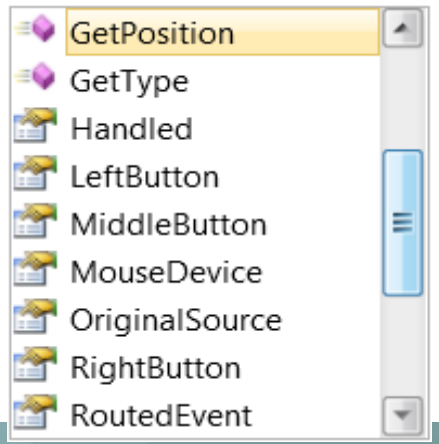



```
Handled: get and set a value indicating
         whether the keypress has been handled
Key:     the character pressed
         if (e.Key==Key.F10 || e.Key==Key.ESC)
           ...
```

Eventargs: Afhænger af typen af event

```
private void textBox1_MouseLeftButtonDown(
   object sender, MouseButtonEventArgs e)
   { ... }
```

e.



GetPosition: returns position of mouse relative to control
Handled: true if the event is handled
LeftButton, MiddleButton, RightButton: the state of a mouse button

Opsummering – eventhandlers

- I en klasse som raiser events (GUI), findes der en event variable for hvert type event.

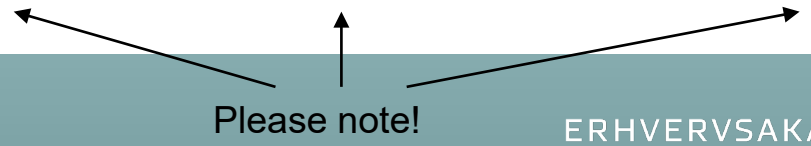
Eksempel: Button har en event-variable **`Click (og mange andre event-variabler)`**

- **`public event RoutedEventHandler Click;`** //multi-cast!!
- En event hører til en *delegate*-definition, som beskriver return type og parameter for den event handler.

En type!

`Click`-event i Button hører til denne delegate-declaration

```
public delegate void RoutedEventHandler(object sender, RoutedEventArgs e)
```



Tilføje en eventhandler

- En eventhandler skal tilføjes til event variable – f.eks. Click her for en knap:

```
button1.Click += this.buttonOk_Click;
```

kun += og -= er legale

- Flere eventhandlers kan tilføjes til den same event-variable.
- *Alle* de tilføjede eventhandlers bliver kaldt når eventet er raised (event er en multi-cast delegate)

Hjælp til event-handlers i VisualStudio

Efter += i koden:

```
weatherStation.sendEvent +=
```

new SendTemp(weatherStation_sendEvent); (Press TAB to insert)

Tryk TAB. Så er new …autogenerated:

```
weatherStation.sendEvent +=new SendTemp(weatherStation_sendEvent);
```

Press TAB to generate handler 'weatherStation_sendEvent' in this class

Navnet kan ændres:

```
weatherStation.sendEvent +=new SendTemp(OnNewTemp);
```

Press TAB to generate handler 'OnNewTemp' in this class

Tryk på TAB igen og metoden er autogeneret i klassen:

```
void OnNewTemp(int newtemp)
{
  throw new Exception("The method or operation is not implemented.");
}
```

OPGAVER