WINDOWSPROGRAMMERING LEKTION 3

Mahmud Al Hakim
mahmud@alhakim.se
www.alhakim.se

AGENDA

Introduktion till Databindning (Data Binding)

Element Binding

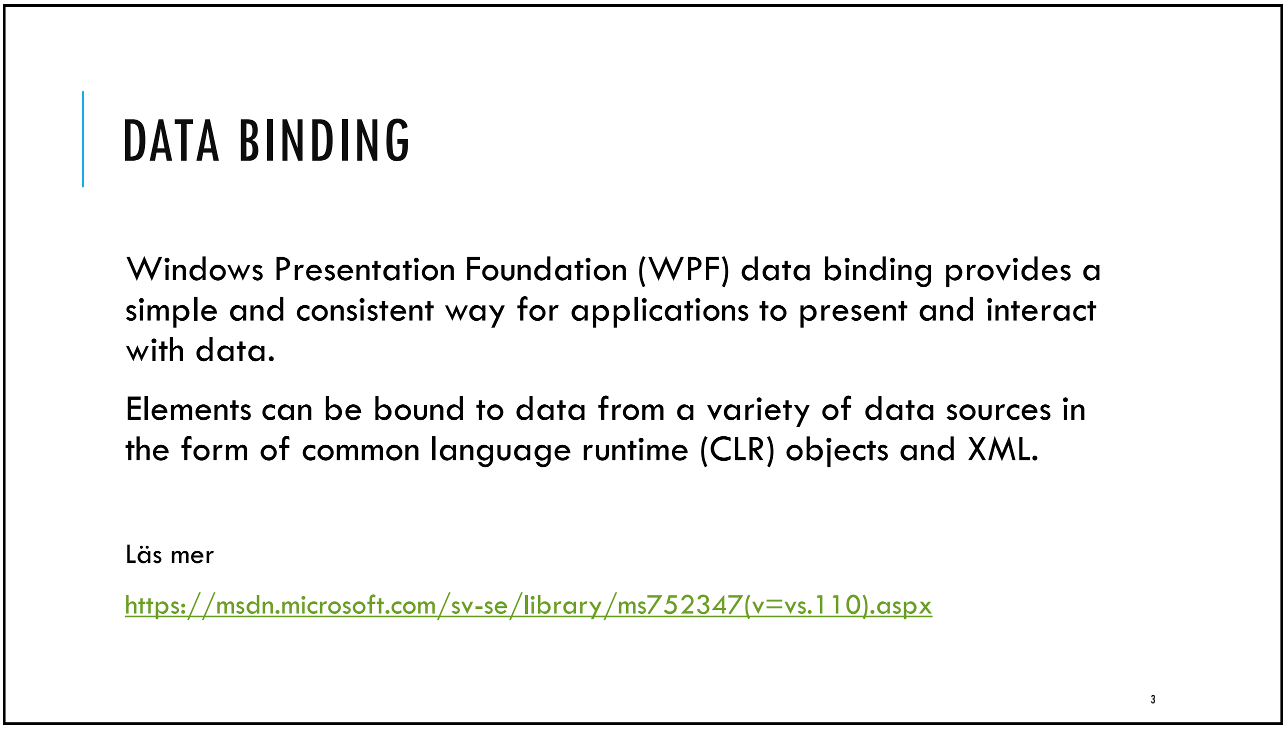
Data Context

Stringformat

Object Binding

Gränssnittet INotifyPropertyChanged (INPC)

ObservableCollection



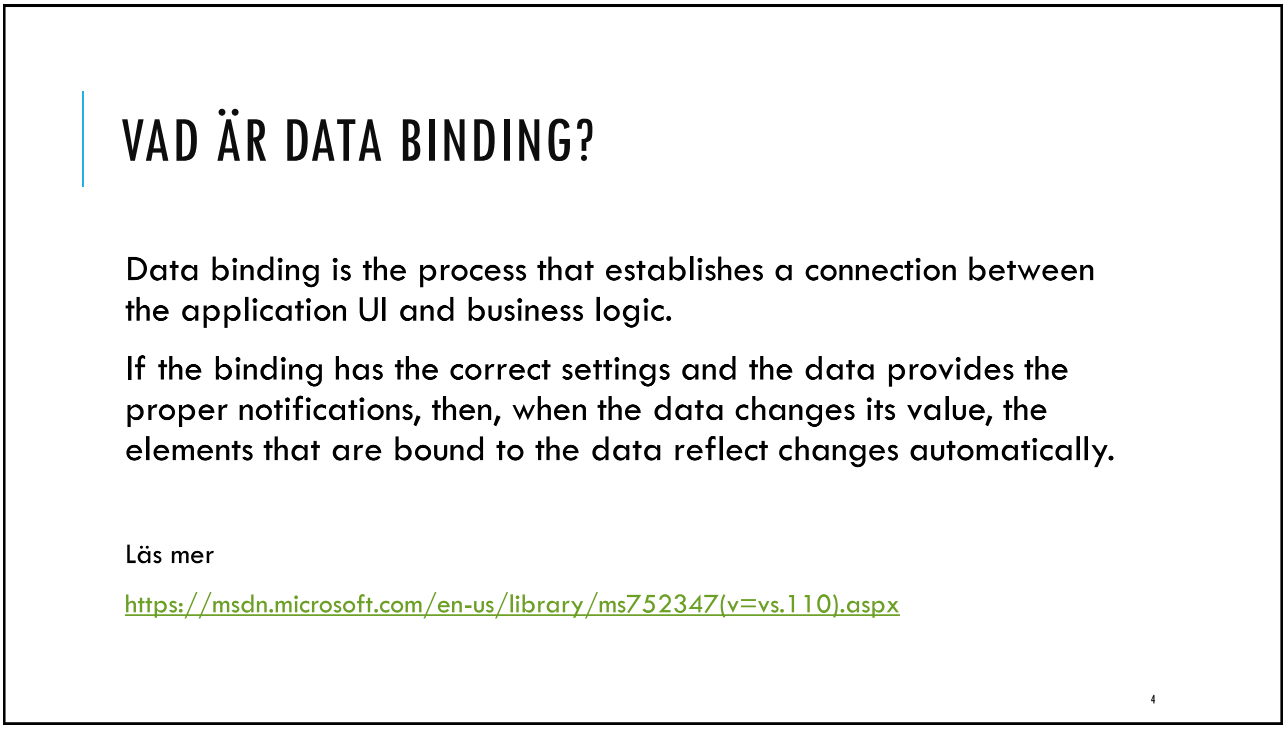
DATA BINDING

Windows Presentation Foundation (WPF) data binding provides a simple and consistent way for applications to present and interact with data.

Elements can be bound to data from a variety of data sources in the form of common language runtime (CLR) objects and XML.

Läs mer

https://msdn.microsoft.com/sv-se/library/ms752347(v=vs.110).aspx



VAD ÄR DATA BINDING?

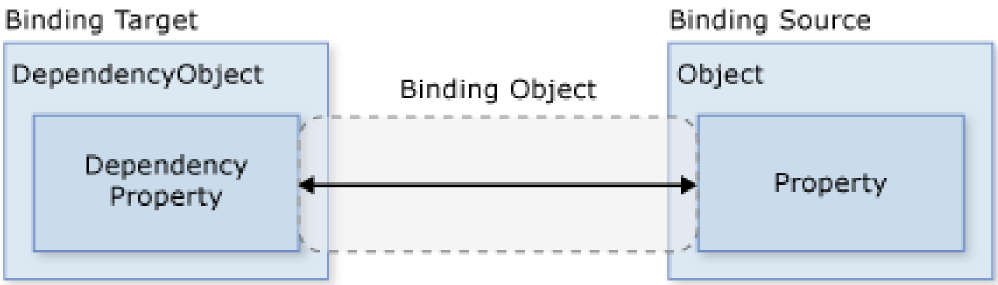
Data binding is the process that establishes a connection between the application UI and business logic.

If the binding has the correct settings and the data provides the proper notifications, then, when the data changes its value, the elements that are bound to the data reflect changes automatically.

Läs mer

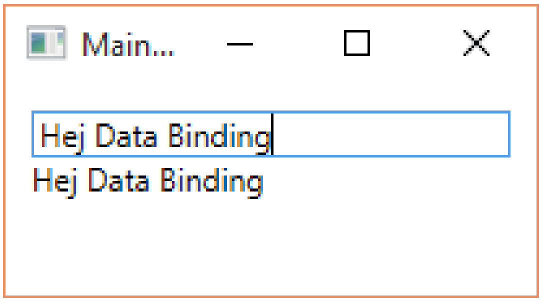
https://msdn.microsoft.com/en-us/library/ms752347(v=vs.110).aspx

DATA BINDING MODEL



Bildkälla:
https://msdn.microsoft.com/sv-se/library/ms752347(v=vs.110).aspx

ELEMENT BINDING



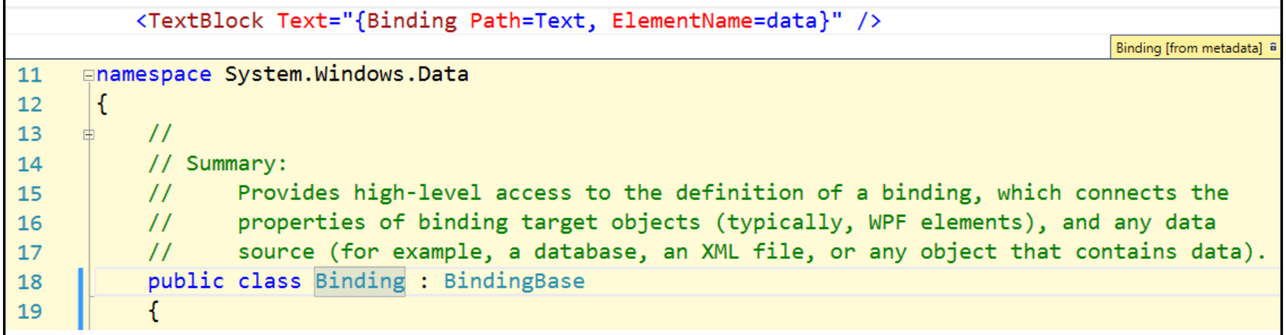
Hej Data Binding

Hej Data Binding

```
<StackPanel Margin="10">
<TextBox Name="data"
         Text="TEST"
 />
<TextBlock Text="{Binding Path=Text, ElementName=data}" />
</StackPanel>
```

{Markup Extension}

KLASSEN BINDING

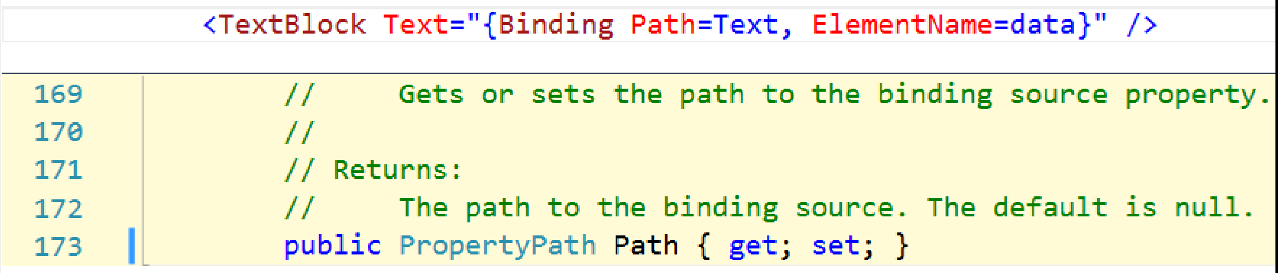


```
<TextBlock Text="{Binding Path=Text, ElementName=data}" />
```

```
namespace System.Windows.Data
{
    //
    // Summary:
    //     Provides high-level access to the definition of a binding, which connects the
    //     properties of binding target objects (typically, WPF elements), and any data
    //     source (for example, a database, an XML file, or any object that contains data).
    public class Binding : BindingBase
    {
```

EGENSKAPEN PATH

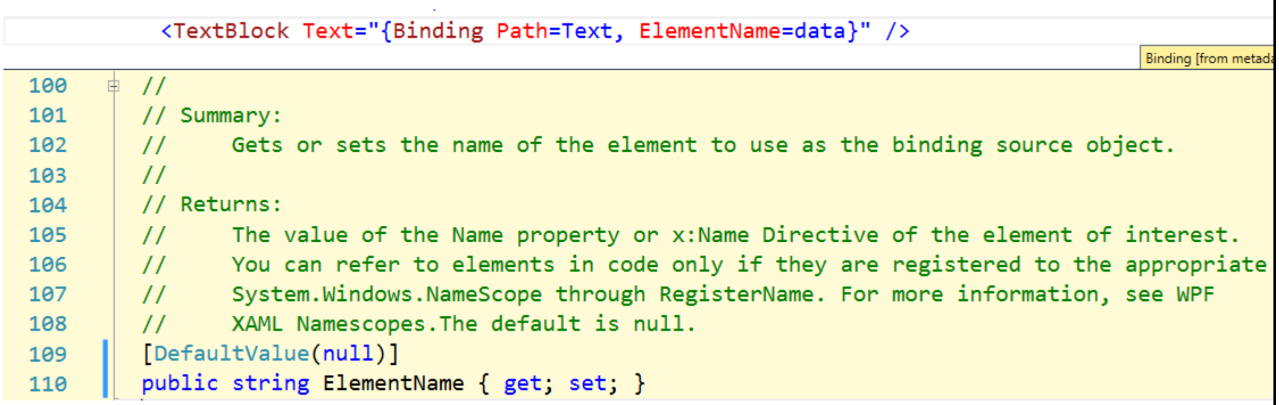
Tips!
Path=
kan utelämnas om källan är den första egenskapen som skrivs direkt efter Binding



```
<TextBlock Text="{Binding Path=Text, ElementName=data}" />
```

```
        //     Gets or sets the path to the binding source property.
        //
        // Returns:
        //     The path to the binding source. The default is null.
        public PropertyPath Path { get; set; }
```

EGENSKAPEN ELEMENTNAME

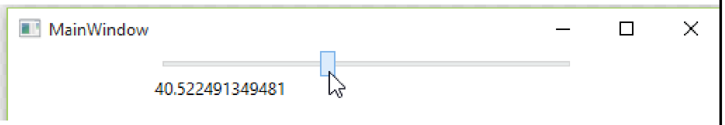


```
    <TextBlock Text="{Binding Path=Text, ElementName=data}" />
```

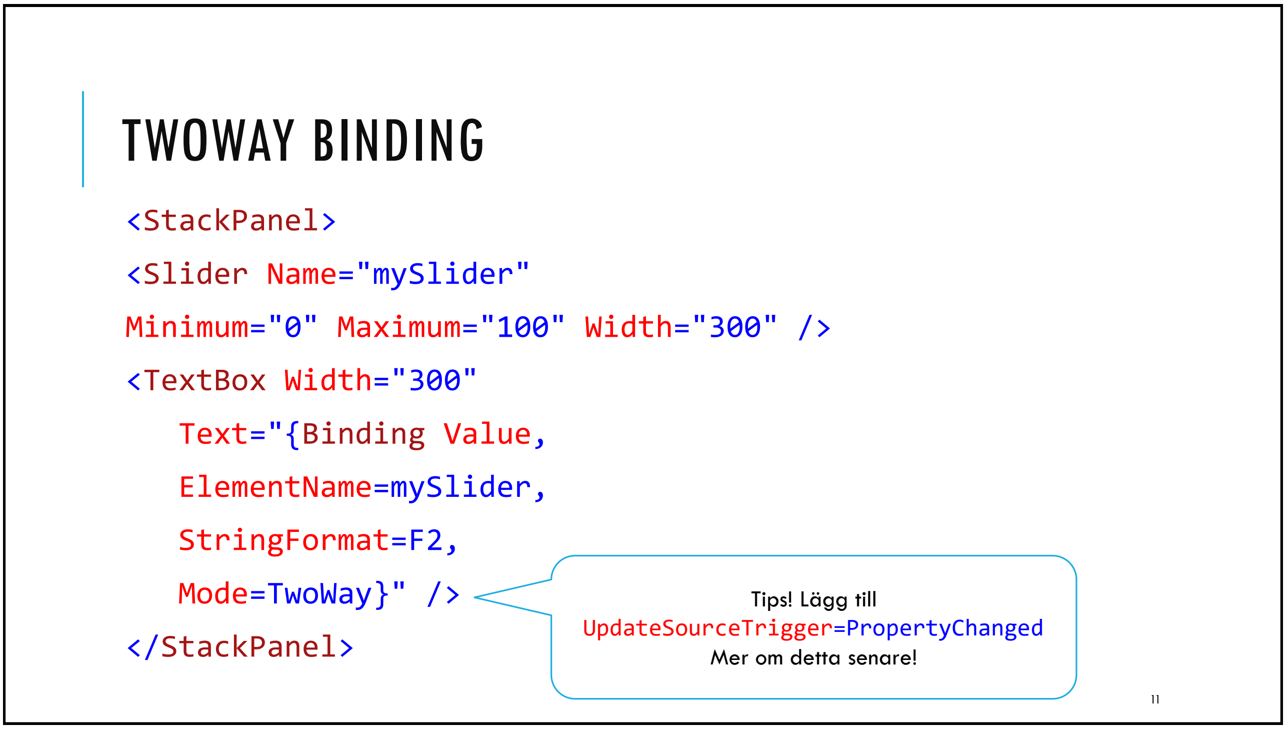
```
//
// Summary:
//     Gets or sets the name of the element to use as the binding source object.
//
// Returns:
//     The value of the Name property or x:Name Directive of the element of interest.
//     You can refer to elements in code only if they are registered to the appropriate
//     System.Windows.NameScope through RegisterName. For more information, see WPF
//     XAML Namescopes.The default is null.
[DefaultValue(null)]
public string ElementName { get; set; }
```

ELEMENT BINDING – EXEMPEL 2

```
<StackPanel>
<Slider Name="mySlider"
        Minimum="0"
        Maximum="100"
        Width="300" />
<TextBox Width="300"
Text="{Binding Value, ElementName=mySlider}" />
</StackPanel>
```



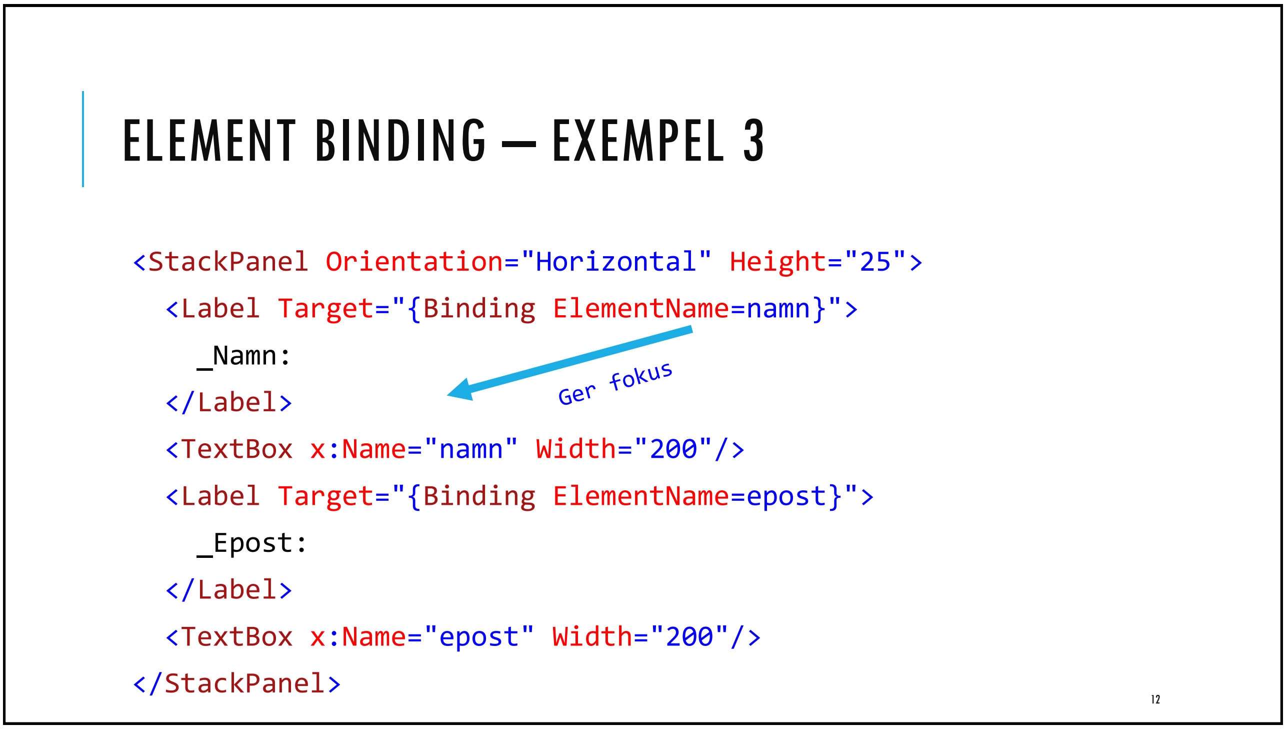
Tips! Lägg till StringFormat=F2
https://msdn.microsoft.com/sv-SE/library/s8s7t687.aspx



TWOWAY BINDING

```
<StackPanel>
<Slider Name="mySlider"
Minimum="0" Maximum="100" Width="300" />
<TextBox Width="300"
   Text="{Binding Value,
   ElementName=mySlider,
   StringFormat=F2,
   Mode=TwoWay}" />
</StackPanel>
```

Tips! Lägg till
`UpdateSourceTrigger=PropertyChanged`
Mer om detta senare!



ELEMENT BINDING – EXEMPEL 3

```
<StackPanel Orientation="Horizontal" Height="25">
  <Label Target="{Binding ElementName=namn}">
    _Namn:
  </Label>
  <TextBox x:Name="namn" Width="200"/>
  <Label Target="{Binding ElementName=epost}">
    _Epost:
  </Label>
  <TextBox x:Name="epost" Width="200"/>
</StackPanel>
```

Ger fokus

EGENSKAPEN DATACONTEXT

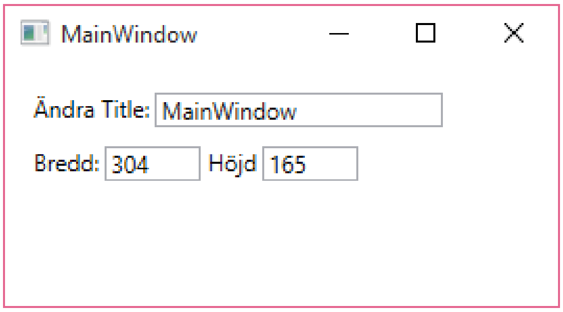
```
public partial class MainWindow : Window
    {
        public MainWindow()
        {
            InitializeComponent();
            DataContext = this;

        }
    }
```

```
Gets or sets the data context for an element
when it participates in data binding.
Returns: The object to use as data context.
```

ÖVNING

```
<StackPanel Margin="15">
 <WrapPanel>
   <TextBlock Text="Ändra Title: " />
   <TextBox Text="{Binding Title}" Width="150" />
 </WrapPanel>
  <WrapPanel Margin="0,10,0,0">
    <TextBlock Text="Bredd: " />
    <TextBox Text="{Binding Width}" Width="50" />
    <TextBlock Text=" Höjd " />
    <TextBox Text="{Binding Height}" Width="50" />
  </WrapPanel>
</StackPanel>
```



MainWindow

Ändra Title: MainWindow

Bredd: 304 Höjd 165

OBS!
Glöm inte att sätta
`DataContext = this;`

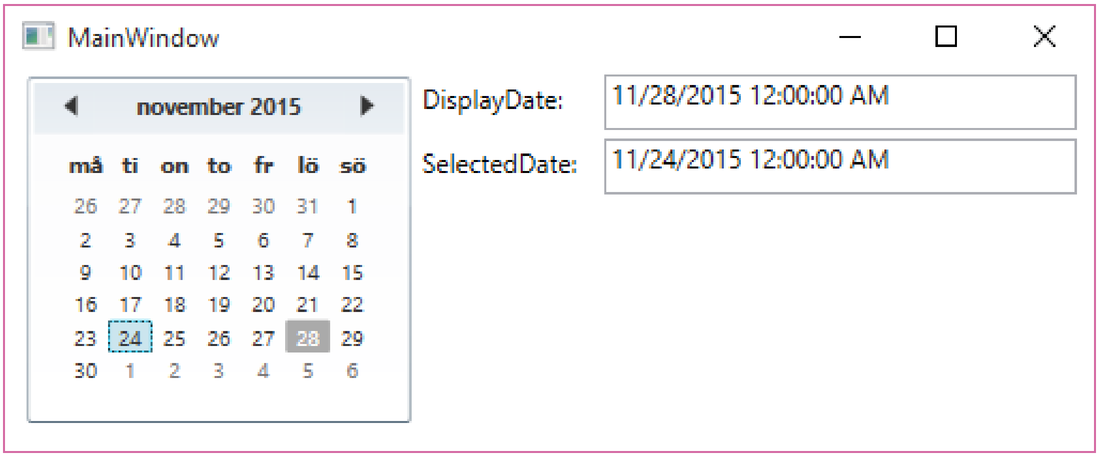
EGENSKAPEN UPDATESOURCETRIGGER

```
<WrapPanel>
<TextBlock Text="Ändra Title: " />
<TextBox Text="{Binding Title,
UpdateSourceTrigger = PropertyChanged}" Width="150" />
</WrapPanel>
```

Gets or sets a value that determines the timing of binding source updates.

Updates the binding source immediately whenever the binding target property changes.

ÖVNING – SKAPA EN KALENDER
BINDA TVÅ TEXTFÄLT TILL DISPLAYDATE OCH SELECTEDDATE

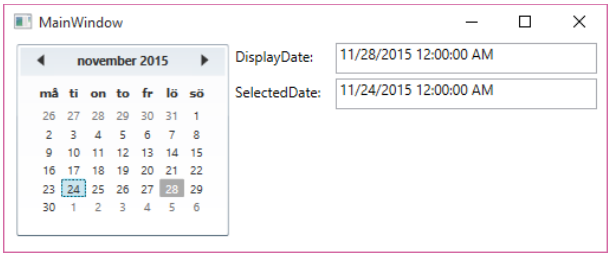


må	ti	on	to	fr	lö	sö
26	27	28	29	30	31	1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	1	2	3	4	5	6

DisplayDate: 11/28/2015 12:00:00 AM

SelectedDate: 11/24/2015 12:00:00 AM

FACIT



november 2015

må	ti	on	to	fr	lö	sö
26	27	28	29	30	31	1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	1	2	3	4	5	6

DisplayDate: 11/28/2015 12:00:00 AM

SelectedDate: 11/24/2015 12:00:00 AM

```
<Grid>
    <Grid.RowDefinitions>
          <RowDefinition Height="30" />
          <RowDefinition Height="30" />
          <RowDefinition Height="*" />
    </Grid.RowDefinitions>
    <Grid.ColumnDefinitions>
          <ColumnDefinition Width="Auto" />
          <ColumnDefinition Width="Auto" />
          <ColumnDefinition Width="200" />
    </Grid.ColumnDefinitions>
</Grid>
```

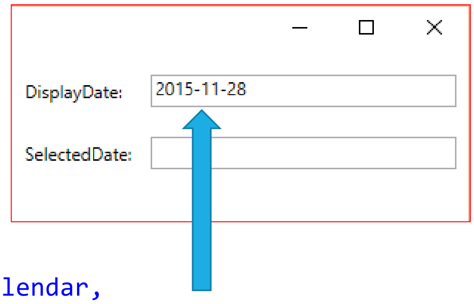
FACIT FORTS...

```
<Calendar x:Name="theCalendar"  Width="180" Height="170"
Margin="10,0,0,0" VerticalAlignment="Top" Grid.RowSpan="3" />

<Label Content="DisplayDate:" Grid.Column="1" Grid.Row="0" />
<TextBox x:Name="displayDateTextBox"
Text="{Binding DisplayDate, ElementName=theCalendar, UpdateSourceTrigger=PropertyChanged}"
Margin="8,2" Grid.Column="2" Grid.Row="0" />

<Label Content="SelectedDate:" Grid.Column="1" Grid.Row="1" />
<TextBox x:Name="selectedDateTextBox"
Text="{Binding SelectedDate, ElementName=theCalendar, UpdateSourceTrigger=PropertyChanged}"
Margin="8,2" Grid.Column="2" Grid.Row="1" />
```

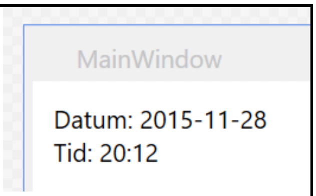
STRINGFORMAT


DisplayDate: 2015-11-28

SelectedDate:

```
<TextBox x:Name="displayDateTextBox"
Text="{Binding DisplayDate, ElementName=theCalendar,
UpdateSourceTrigger=PropertyChanged, StringFormat=yyyy-MM-dd }"
Margin="8,2" Grid.Column="2" Grid.Row="0" />
```

STRINGFORMAT – EXEMPEL 2



Datum: 2015-11-28
Tid: 20:12

OBS! Vi behöver lägga till namnrymden System

```
xmlns:system="clr-namespace:System;assembly=mscorlib"
```

```
<StackPanel Margin="10">
<TextBlock Text="{Binding Source={x:Static system:DateTime.Now},
StringFormat=Datum: {0:yyyy-MM-dd}}" />

<TextBlock Text="{Binding Source={x:Static system:DateTime.Now},
StringFormat=Tid: {0:HH:mm}}" />
</StackPanel>
```

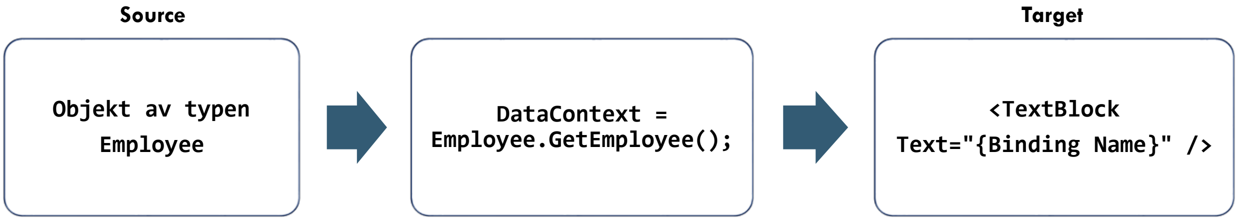
LITE OM X:STATIC

- x:Static är en Markup Extension
- Finns i XAML för att kunna använda statiska medlemmar

```
<StackPanel Margin="10">
    <TextBlock Text="{Binding
        Source={x:Static system:DateTime.Now},
        StringFormat=yyyy-MM-dd}" />
</StackPanel>
```

Tips: https://msdn.microsoft.com/en-us/library/ms742135(v=vs.110).aspx

OBJECT BINDING



Objekt av typen Employee

```
DataContext =
Employee.GetEmployee();
```

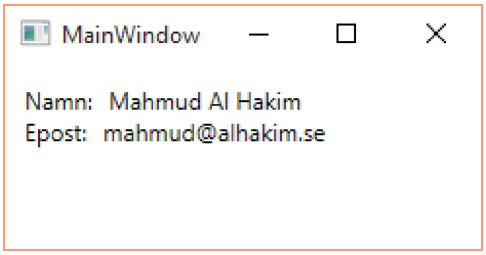
```
<TextBlock
Text="{Binding Name}" />
```

SOURCE – KLASSEN EMPLOYEE

```
public class Employee {
        public string Name  { get; set; }
        public string Email { get; set; }

        public static Employee GetEmployee() {
           var emp = new Employee() {
               Name  = "Mahmud Al Hakim",
               Email = "mahmud@alhakim.se"
           };
           return emp;
        }
}
```

TARGET – UI



Namn: Mahmud Al Hakim
Epost: mahmud@alhakim.se

```
<StackPanel Margin="10">
 <StackPanel Orientation="Horizontal">
    <TextBlock Text="Namn: " />
    <TextBlock Margin="5,0,0,0"  Text="{Binding Name}" />
 </StackPanel>
 <StackPanel Orientation="Horizontal">
    <TextBlock Text="Epost: " />
    <TextBlock Margin="5,0,0,0"  Text="{Binding Email}" />
 </StackPanel>
</StackPanel>
```

DATACONTEXT I CODE BEHIND

```
public partial class MainWindow : Window
    {
        public MainWindow()
        {
            InitializeComponent();
            DataContext = Employee.GetEmployee();
        }
    }
```

INPC

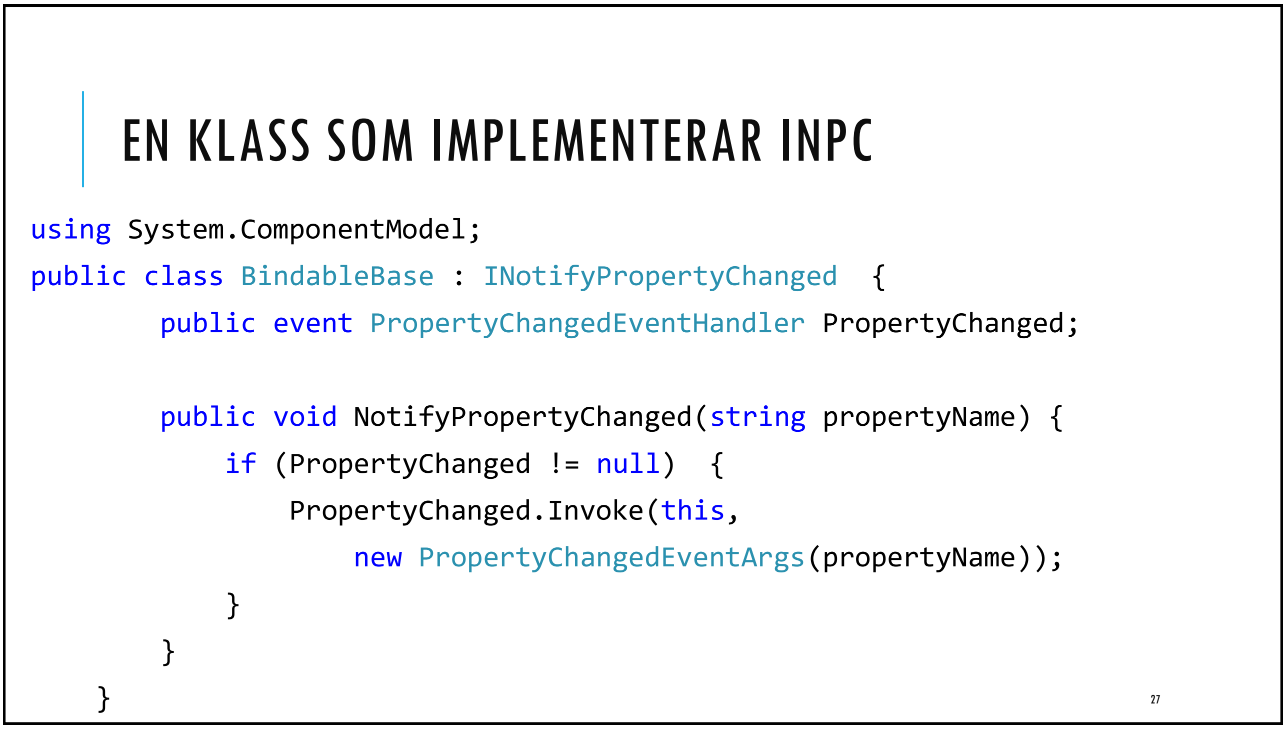
Gränssnittet uppdates inte automatiskt vid förändringar i bakomliggande objekt!

Detta är bra för …

- Att få bättre prestanda på applikationen.
- Att manuellt kunna notifiera flera kontroller som är beroende av en aktuell uppdatering.

Hur uppdaterar vi gränssnittet då?

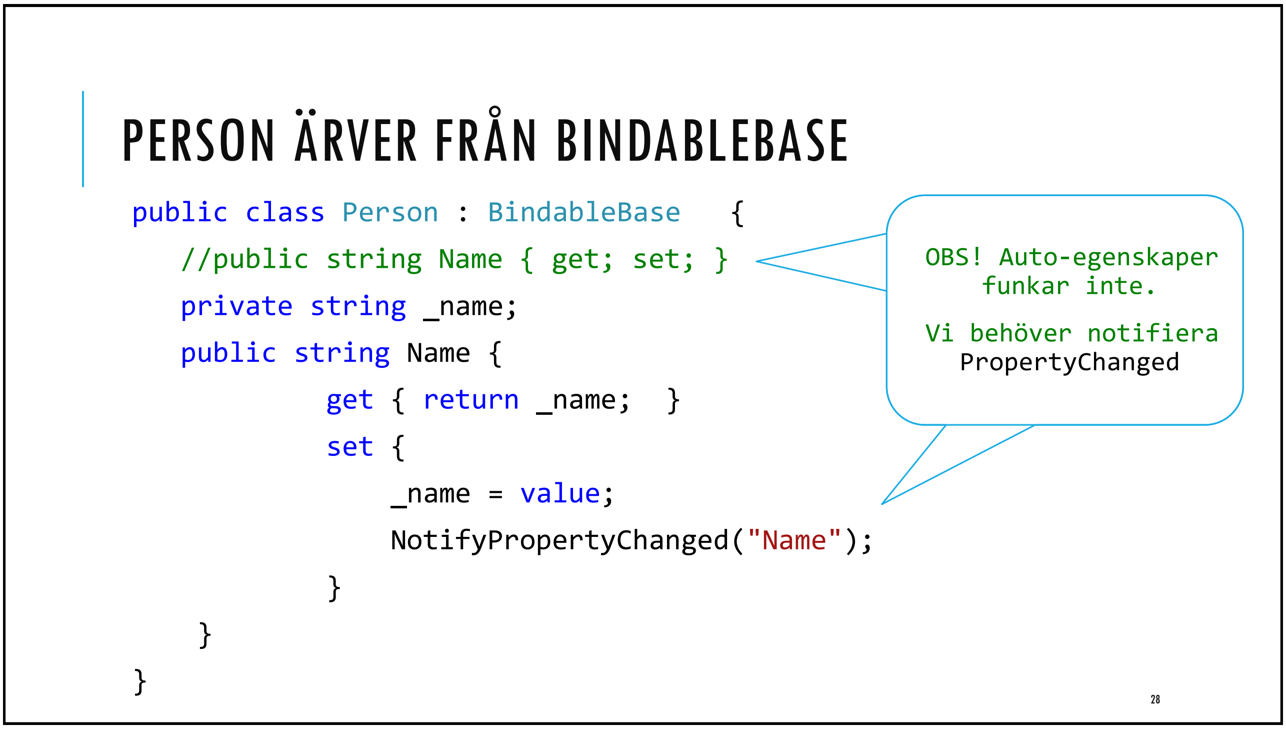
1. Implementera gränssnittet `INotifyPropertyChanged` (INPC).
2. Initiera händelsen `PropertyChangedEventHandler`



EN KLASS SOM IMPLEMENTERAR INPC

```
using System.ComponentModel;
public class BindableBase : INotifyPropertyChanged  {
        public event PropertyChangedEventHandler PropertyChanged;

        public void NotifyPropertyChanged(string propertyName) {
            if (PropertyChanged != null)  {
                PropertyChanged.Invoke(this,
                   new PropertyChangedEventArgs(propertyName));
            }
        }
    }
```

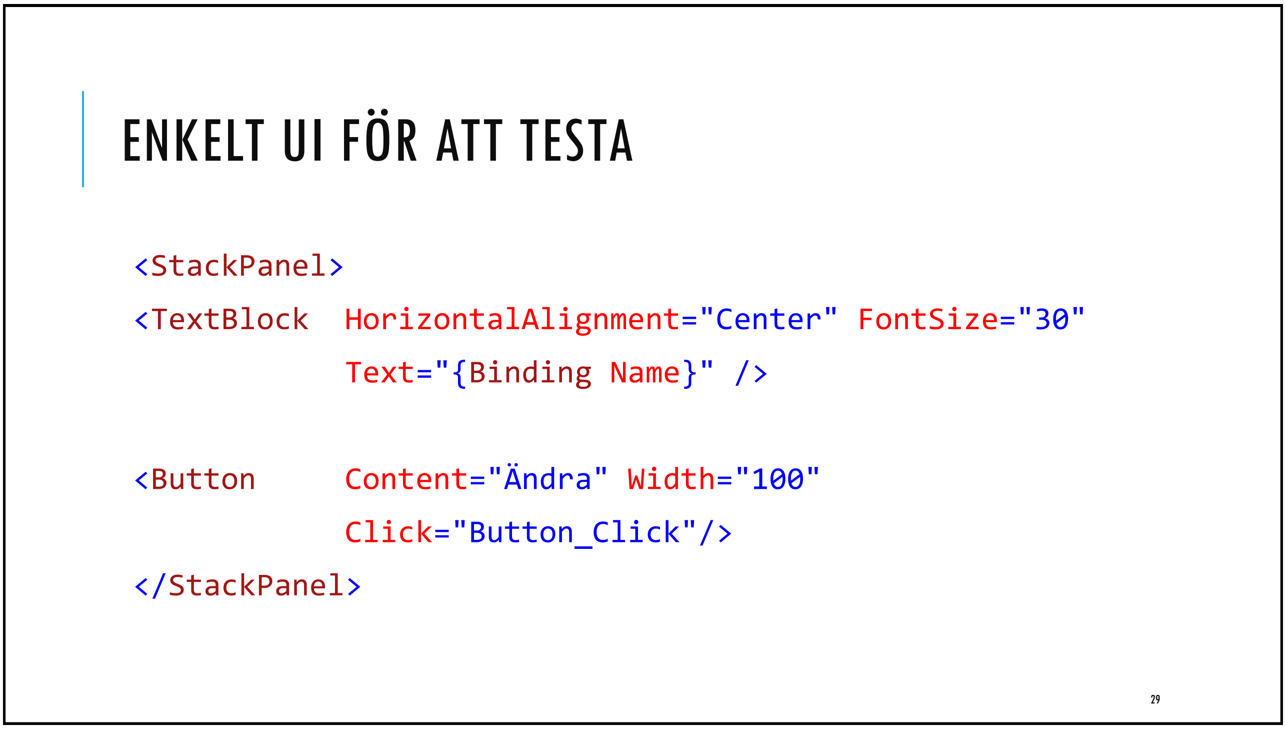


PERSON ÄRVER FRÅN BINDABLEBASE

```
public class Person : BindableBase   {
   //public string Name { get; set; }
   private string _name;
   public string Name {
            get { return _name;  }
            set {
                _name = value;
                NotifyPropertyChanged("Name");
            }
       }
   }
```

OBS! Auto-egenskaper funkar inte.

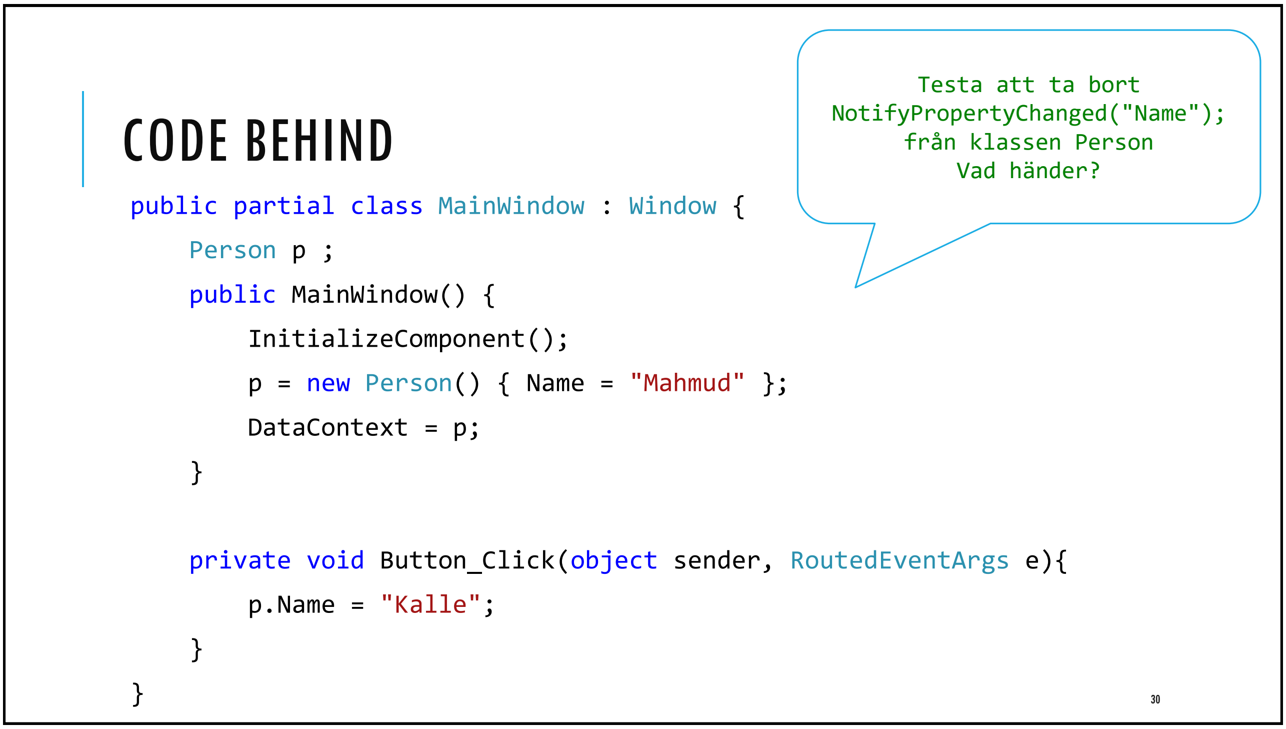
Vi behöver notifiera PropertyChanged



ENKELT UI FÖR ATT TESTA

```
<StackPanel>
<TextBlock  HorizontalAlignment="Center" FontSize="30"
            Text="{Binding Name}" />

<Button     Content="Ändra" Width="100"
            Click="Button_Click"/>
</StackPanel>
```

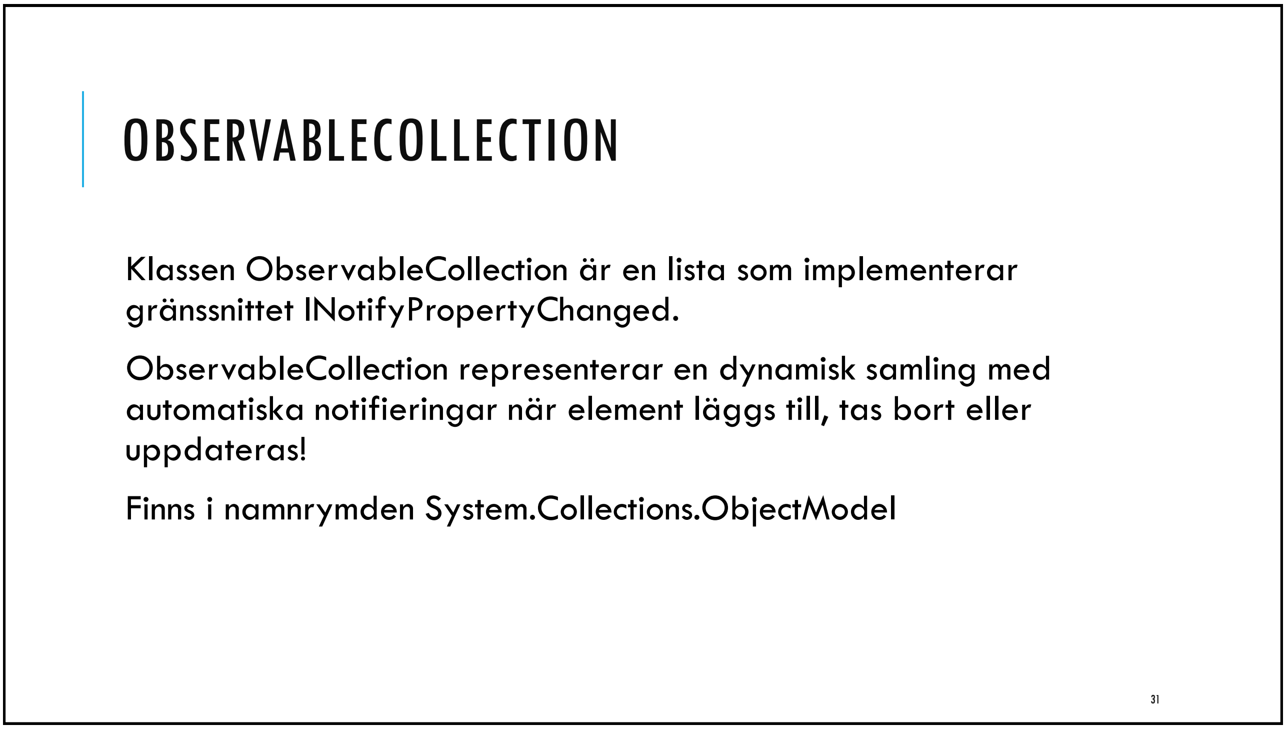


CODE BEHIND

Testa att ta bort
NotifyPropertyChanged("Name");
från klassen Person
Vad händer?

```
public partial class MainWindow : Window {
    Person p ;
    public MainWindow() {
        InitializeComponent();
        p = new Person() { Name = "Mahmud" };
        DataContext = p;
    }

    private void Button_Click(object sender, RoutedEventArgs e){
        p.Name = "Kalle";
    }
}
```

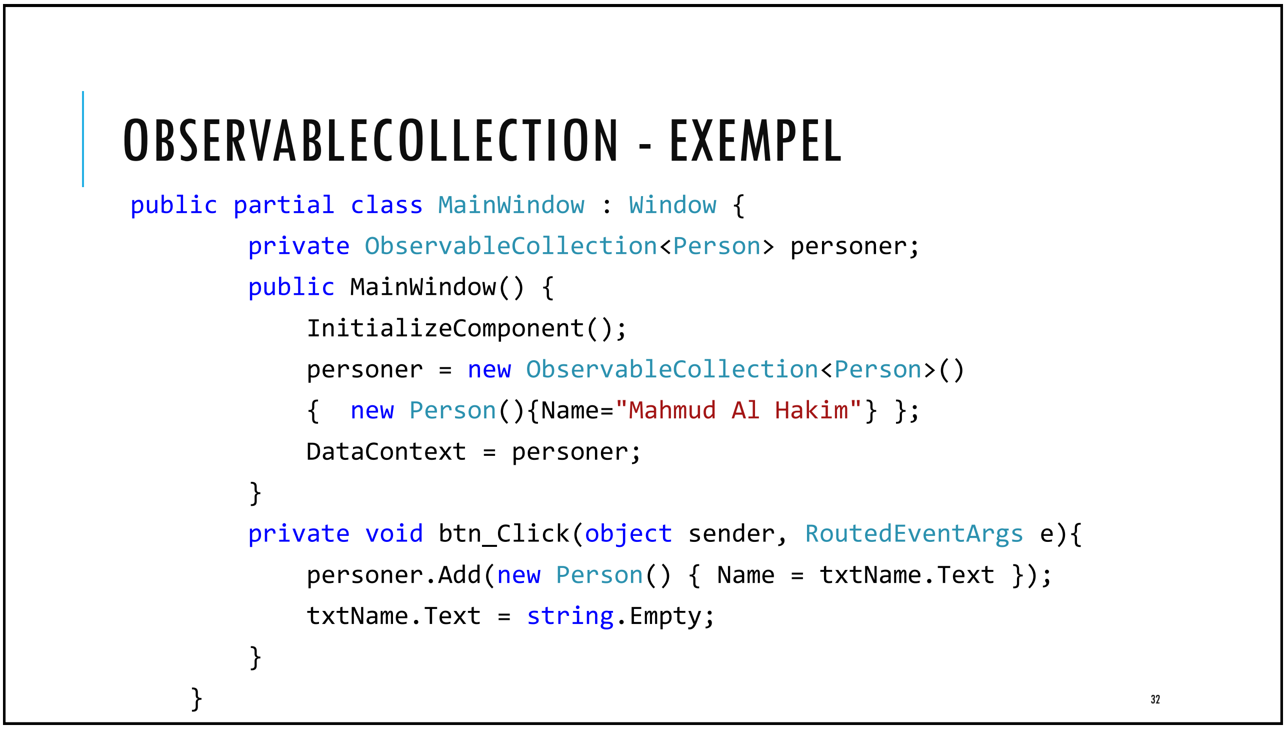


OBSERVABLECOLLECTION

Klassen ObservableCollection är en lista som implementerar gränssnittet INotifyPropertyChanged.

ObservableCollection representerar en dynamisk samling med automatiska notifieringar när element läggs till, tas bort eller uppdateras!

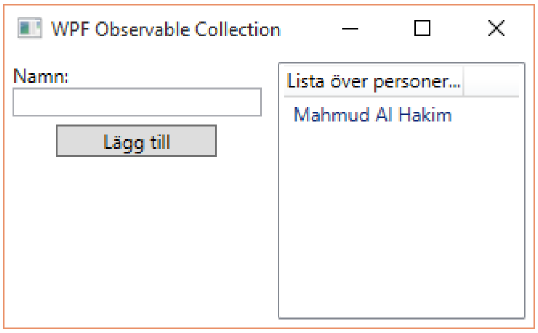
Finns i namnrymden System.Collections.ObjectModel



OBSERVABLECOLLECTION - EXEMPEL

```
public partial class MainWindow : Window {
        private ObservableCollection<Person> personer;
        public MainWindow() {
            InitializeComponent();
            personer = new ObservableCollection<Person>()
            {  new Person(){Name="Mahmud Al Hakim"} };
            DataContext = personer;
        }
        private void btn_Click(object sender, RoutedEventArgs e){
            personer.Add(new Person() { Name = txtName.Text });
            txtName.Text = string.Empty;
        }
    }
```

OBSERVABLECOLLECTION - UI



Mahmud Al Hakim

```
<Grid>
<Grid.ColumnDefinitions> <ColumnDefinition/>
<ColumnDefinition Width="*"/> </Grid.ColumnDefinitions>
<StackPanel Margin="5">
    <TextBlock x:Name="lblName" Text="Namn:"></TextBlock>
    <TextBox x:Name="txtName"></TextBox>
    <Button Width="100" Height="20" Margin="5" Click="btn_Click" Content="Lägg till"></Button>
</StackPanel>
<ListView Margin="5" Grid.Column="1" ItemsSource="{Binding}">
    <ListView.View>
      <GridView><GridViewColumn Header="Lista över personer..."  DisplayMemberBinding="{Binding Name}"/>
     </GridView>
   </ListView.View>
</ListView>
</Grid>
```

ÖVNING

- Utveckla föregående exempel genom att lägga till ett E-post-fält.
- Lägg till en kolumn som visar epost.
- Lägg till en knapp för borttagning av en markerad person.

Tips: http://www.wpf-tutorial.com/data-binding/responding-to-changes/

Tips! Skapa en KeyUp-händelse i TextBox

```
private void txtName_KeyUp(object sender, KeyEventArgs e)
{
            if (e.Key == Key.Enter)
                    btn_Click(sender, e);
}
```