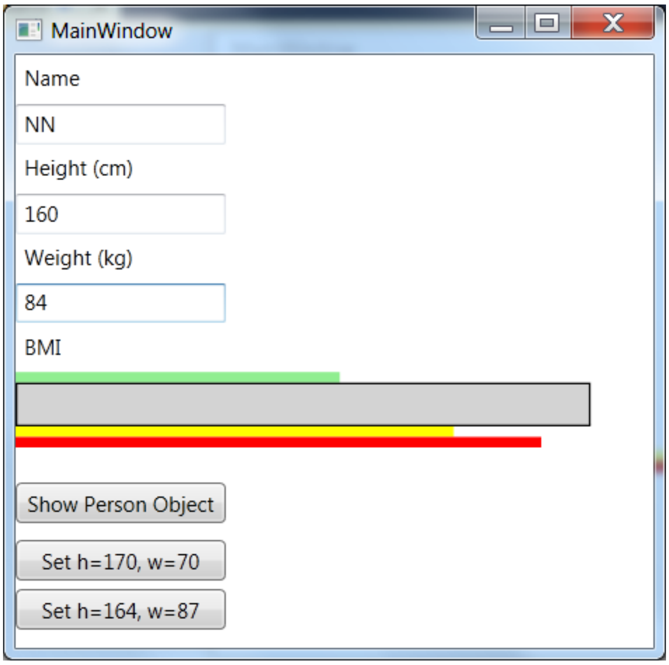
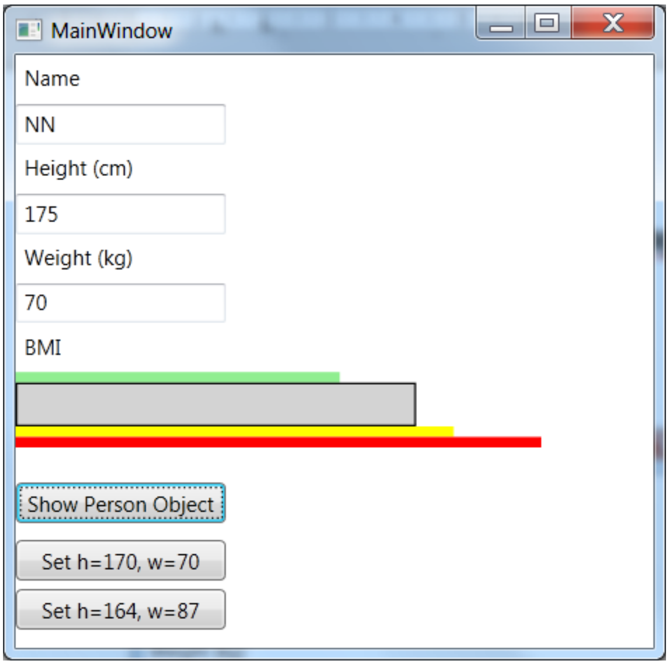
15V Dag 8, Opgaver – DataBinding - DataTempletes

1) *Binding with converters*
The project `dag8opg1start.zip` in the solution dag08databinding contains the start of an application to handle a person object which is bound to some GUI controls. Below you see two examples of the GUI.



Name: NN
Height (cm): 160
Weight (kg): 84
BMI



Name: NN
Height (cm): 175
Weight (kg): 70
BMI

The project contains a MyClasses.cs with a Person class with properties Name (get+set), Height (get+set), Weight (get+set) and BMI (only get).
The three Buttons click events are implemented.

The Text property of three textboxes must be bound to then first three Properties in Person.
The green, yellow and red rectangle are fixed with no bounding.
The width of the lightgrey rectangle must be bound to BMI such that the width is BMI*10.

Important about the Height: Please note the in the GUI the Height is specified in cm, but in the person object the height is a double with the height in m. (In the left example above person.Height=1.60.

a) Implement INotifyPropertyChanged in the Person class. Remember BMI is changed when Height or Weight is changed.

b) Implements the Bindings to the three TextBox’es without any value converters.

c) Implement a value converter for the binding to the height: You need a new class, HeightConverter, which implements IValueConverter. You need to implement both the Convert-method and the ConvertBack-method.

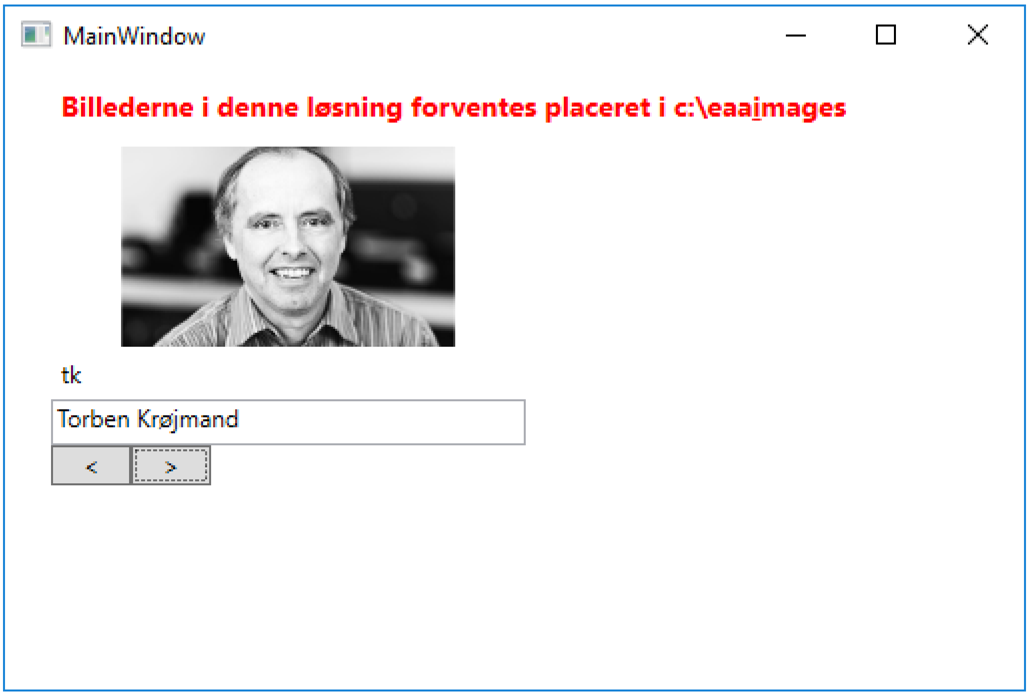
c) Implement a value converter for the binding to the rectangle: You need a new class, BMIConverter, which implements IValueConverter. Here you don’t need to implement the ConvertBack-method.

e) Add another binding to the color of the lightgrey rectangle such that the rectangle changes color depending on the BMI:

When BMI<18.5 the color should be LightYellow
When 18.5<=BMI<25 the color should be LightGreen
When 25<=BMI<30 the color should be LightYellow
When 30<=BMI the color should be Red

Eventually remove the 3 fixed rectangle to make the GUI looks more nice.

2) *Binding to Image Source*
Make a GUI with five control:
An Image
A Label
A TextBox
Two ScrollButtons



Billederne i denne løsning forventes placeret i c:\eaaimages

tk

Torben Krøjmand

Create a class Staff with three properties:
string Filename
string Initials
string FullName

Make an ObservableCollection with 5 Staff objects:

atm.jpg	atm	Arne Tolstrup Madsen
tk.jpg	tk	Torben Krøjmand
kr.jpg	kr	Karsten Rasmussen
haso.jpg	haso	Hanne Sommer
gs.jpg	gs	Gert Simonsen

The GUI shows only one staff object at a time. Let the two ScrollButtons scroll between the 5 objects. The technique was shown in the demo `ListBinding2`, *just without a ListBox*.

The images can be found in `eaa_images.zip`. Unzip them to c:\ such that the images will be placed in c:\eaa_images. Then an ImageSource can be created as

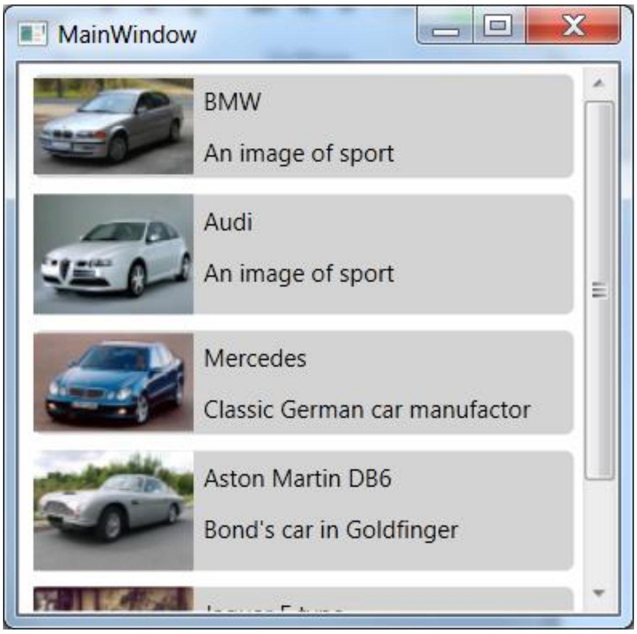
```
Uri uri = new Uri(@"c:\eaa_images\tk.jpg");
BitmapImage img = new BitmapImage(uri);
```

Implement binding between GUI and Staff objects. *The Source property of the Images must be bound to the filename with use of a IValueConverter returning the correct type – find the type of* `Image.Source`.

Select OneWay or TwoWay bindings as appropriate.

3) *Listbox med DataTemplate*

Du skal lave en WPF application med ListBox som vises således



BMW

An image of sport

Audi

An image of sport

Mercedes

Classic German car manufactor

Aston Martin DB6

Bond's car in Goldfinger

Du kan anvende starten på denne opgave i projektet `dag08opg3start` i solution med navnet `dag08datatemplate`.

Billederne er inkluderet i projektet.

Der er programmeret en Car klassen samt en Service klasse (Singleton) med en liste af Car-objekter. Propertien `Service.Cars` returnere denne liste. ListBox'ens ItemSource sættes til denne liste.

Car klassen har erklæringen

```
public class Car
{
  public string Name { get; set; }
  public string Desc { get; set; }
  public ImageSource Image { get; set; }
}
```

ImageSource initialiseres i Service klassen til at være en BitMap source.

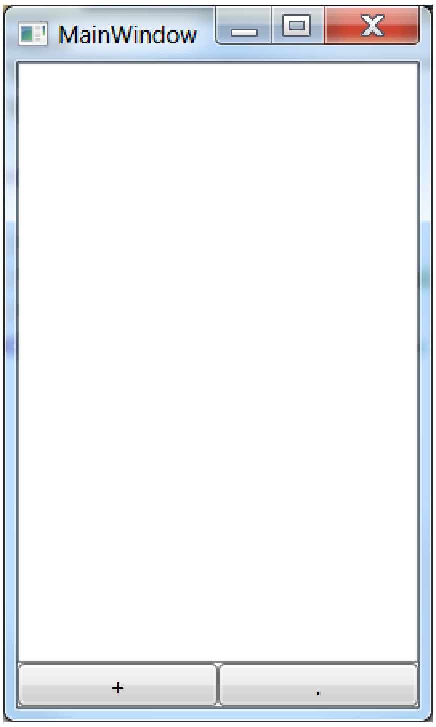
- Sæt ListBox'ens ItemsSource til Service.Cars.
- Tilføje en DataTemplate til ListItem. F.eks. med to Stackpanels, en Image control og to TextBlock controls.

Du kan evt. gå gradsvis frem med din DataTemplate.

4) *DataTemplate*

Du kan anvende starten på denne opgave i projektet `dag08opg4start` i solution med navnet `dag08datatemplate`.

Projekt har fra start følgende Window hvor der er en usynlig ListBox med fast størrelse i midten af vinduet:

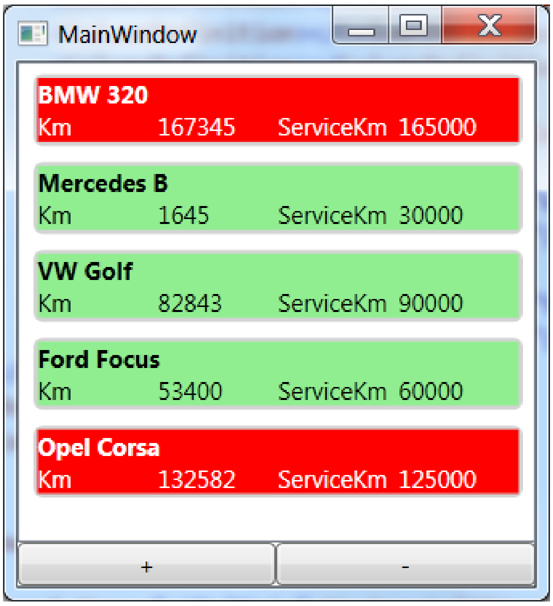


Winduets størrelse tilpasser sig størrelsen af Content.

Der er to knapper nederst til scalering af hele GUI'en – DET VENTER VI MED TIL EN SENERE OPGAVE.

Der er programmeret en Car klassen samt en Service klasse (Singleton) med en liste af Car-objekter. Propertien `Service.Cars` returnere denne liste. ListBox'ens ItemSource sættes til denne liste.

Lav to datatemplate til ListItems, så biler der ikke skal til service vises med én template og de der skal til service vises med en anden:



BMW 320
Km 167345 ServiceKm 165000

Mercedes B
Km 1645 ServiceKm 30000

VW Golf
Km 82843 ServiceKm 90000

Ford Focus
Km 53400 ServiceKm 60000

Opel Corsa
Km 132582 ServiceKm 125000

Begge DataTemplates skal bestå af et grid med 2 rækker. I række 0 vises bilens navn i fed skrift. I række 1 vises Km samt ServiceKm. Anvend 4 kolonner til oplysningerne i nederste række.

I den ene DataTemplate, NormalTemplate, sætter baggrunden for et ListItem til LightGreen.
I den anden, ServicelTemplate, sætter baggrunden for et ListItem til Red.

Programmér en DataTemplateSelector som vælger mellem disse to DataTemplates:
Hvis km<serviceKm skal NormalDataTemplate benyttes, ellers skal ServiceDataTemplate benyttes.