Øvelser dag 7 – WPF – Data binding

Opgave 7.1

Lav et nyt WPF program.
Lav en Person klasse som indeholder Properties: Name, Weight, Age, Score, and Accepted
(Du kan genbruge en Person klasse fra tidligere, det er de samme properties faktisk, så kan du jo bare copy/paste den ind – dog skal der modificeres lidt – se note med bold tekst her).
Lav et nyt Person object med nogle test data.

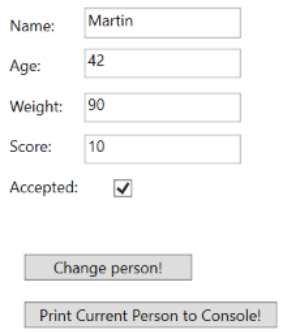
Du skal nu kunne vise properties fra en person af typen Person. Brug data-binding til properties i person – så du skal have noget i stil med dette :

```
 mainGrid.DataContext = person;
//her hedder mit hovedgrid mainGrid og så har jeg defineret et Person object i en person variabel.
```

Du kan bruge Textboxes til at vise den bundne tekst (som skal editeres i næste opgave) og labels til ja, labels!

Du kan se et eksempel her på hvordan det kunne se ud nederst her – her er valgt TextBoxes til alle properties undtagen Accepted. (du behøver ikke have border på som vist her i den højre søjle) - i den venstre er det bare Labels, som jo ikke ændrer sig i forhold til data.
For checkboxen, prøv at binde Checked. Virker det? Hvis ikke, så prøv IsChecked

NB: I denne opgave er det blot data fra EN PERSON
Din UI kunne f.eks. se sådan her ud (der er nogle knapper, som skal bruge i næste opgave)



Opgave 7.2 Notification

Tilret i samme applikation, så hvis properties på person ændres skal det slå igennem i visningen. Brug `InotifyPropertyChanged`.

Opgave 7.3 two way databinding

I den samme WPF applikation skal du nu tilføje en edit mulighed (den mulighed er allerede i en TextBox hvor teksten kan ændres, men nu skal det så virke sådan at data ændres!), sådan

at du kan editere scoren og navnet på den bundne person i TextBoxen og at det så skal ændres i det bundne objekt og også den anden vej! (altså hvis person objektet ændres, så skal gui ændres automatisk)

Nok nemmest så at have two-way binding på…., sådan at data og textbox'ene altid er i sync:

Implement INotifyPropertyChanged interface på Person Klassen og test at GUI'en bliver opdateret når Personen bliver ændret – altså når data i en af dine personer bliver ændret fra koden. (Du kan f.eks. have en knap, som går ind og ændrer data til noget nyt, og så skulle din GUI gerne blive opdateret automatisk – dette tester data -> gui bindingsretning)

Skriv det underliggende Person object ud I consol, sådan at du kan være sikker på at data i det underliggende objekt, faktisk bliver ændret og ikke kun i GUI, når der ændres i Textboxene (dette tester gui -> data bindingsretningen)
Eller du kan skrive det ud i en MessageBox – måske lidt sjovere!: https://wpf-tutorial.com/dialogs/the-messagebox/

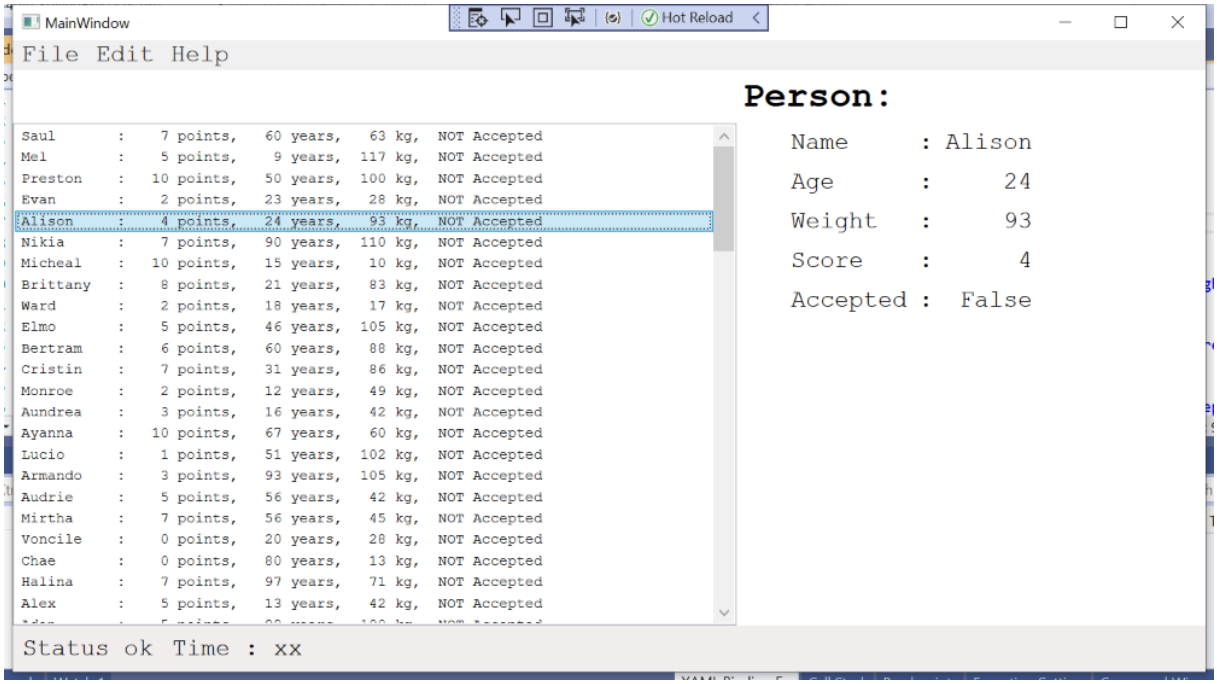
Opgave 7.4 Liste af personer

Lav en ny WPF applikationen og genbrug person klassen i den, som har et ListBox som viser en liste af Personer. Brug binding til en ObservableCollection<Person>.

Prøv f.eks. også at lave en knap som så tilføjer en person til listen – hvis du har implementeret det korrekt med ObservableCollection, så burde listen i gui blive opdateret med den nye person. Det kan f.eks. bare være at knappen tilføjer et fastlagt Person objekt, som du har hardcoded.

Hvordan får du så vist den nuværende valgt person – se screenshot nedenunder?

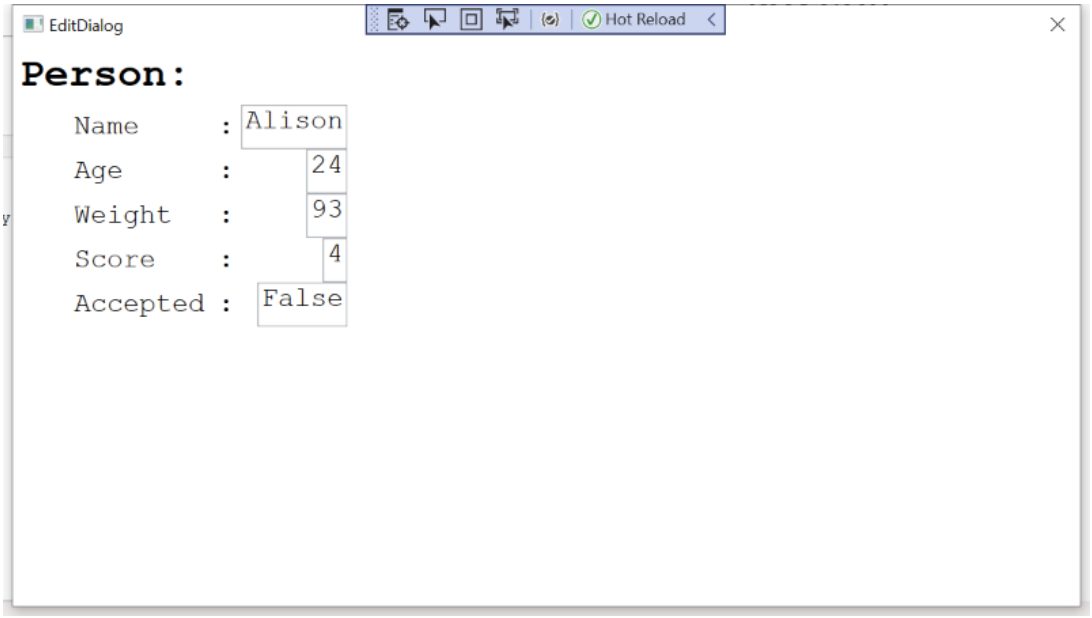
Her er et eksempel på hvordan det kunne se ud i GUI, men du bestemmer selv:



Opgave 7.5 avanceret, men lærerig!

Prøv at lave en edit mulighed. Det kunne være f.eks. være sådan at når man vælger edit (måske fra en menu), så skal man kunne editere den person, som er den nuværende valgte i listen.

Det kan f.eks. laves med at man åbner et separat edit vindue, hvor man kan editere Properties for den valgte person. Når man så er færdig med at editere i den person, så skal ændringer være i listen af personer i hovedvinduet – altså man skal kunne se de ændringer der er lavet i listen jo. Her er et eksempel på hvordan et Edit vindue kunne se ud:



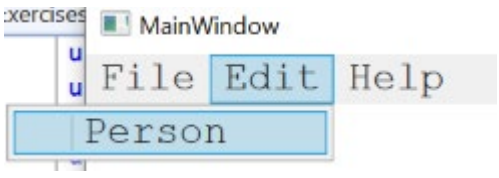
Du kan så bruge two-way databinding til at sørge for at textboxene som kan editeres, også opdaterer det underliggende person objekt:

```
<!–her kun vist for Name – med en label og en TextBox med two-way databinding-->
            <Label Grid.Row="1" Grid.Column="1" Grid.ColumnSpan="1" Content="Name"/>
            <TextBox Grid.Row="1" Grid.Column="3" Grid.ColumnSpan="1" HorizontalAlign-
ment="Right" Text="{Binding Mode=TwoWay, Path=Name}"></TextBox>
```

Du kan lave et separat vindue ligesom du laver hovedvinduet, lad os sige du kalder det Edit-Dialog. Her er en guide til hvordan du laver et ekstra vindue med sin egen xaml fil og tilhørende kode.

https://www.c-sharpcorner.com/blogs/create-a-new-window-in-wpf

Så kan du åbner det på følgende måde fra f.eks. en menu edit option:



Læg mærke til hvordan det nuværende SelectedItem bliver sendt med til constructoren, så du skal lige have en constructor, der kan tage et Person object:

```
  private void MenuItem_Click(object sender, RoutedEventArgs e) {
            var edit = new EditDialog((Person)listBox.SelectedItem);
            edit.Closed += Edit_Dialog_Closed;
        //mærk mærke til denne EventHandler – bruges senere når edit vinduet er lukket
            edit.ShowDialog();

        }
```

Selve Edit dialog klassen er så et nyt vindue med sin egen gui (så er i en ny xaml fil med tilhørende kode):

```
public partial class EditDialog : Window {
        private Person person = null;

        public EditDialog(Person p) { //læg mærke til constructoren her.
            InitializeComponent();

            person = p;
            personDialogGrid.DataContext = person; //initialisering af DataContext
        }
    }
```

Når vi så lukker Edit vinduet igen, så skal vi jo sørge for at listen viser de nye data (som jo kan være ændret i edit dialogen): (her bruges Close Event Handleren som vi tilføjede før):

```
private void Edit_Dialog_Closed(object sender, EventArgs e) {
            listBox.Items.Refresh();
        }
```

Du kan prøve at udelade Refresh() linjen, men så vil du nok se at data i selve listen ikke bliver opdateret.

Opgave 7.6 Datatemplate

Styr visningen i listbox'en vha templates. Det kan f.eks. være, så Accepted vises vha. en checkbox