Vlaamse Dienst voor Arbeidsbemiddeling en Beroepsopleiding

C#
2013

WPF Taken

Deze cursus is eigendom van de VDAB©

Inhoudsopgave

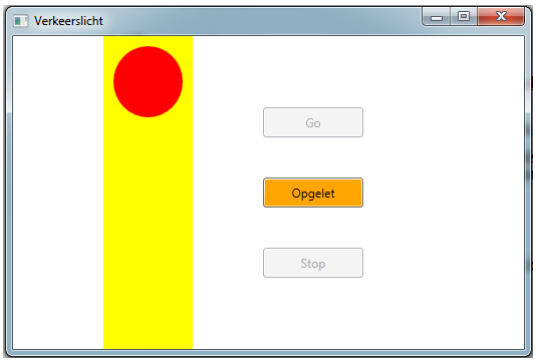
1 VERKEERSLICHT (HOOFDSTUK 1 – 2)

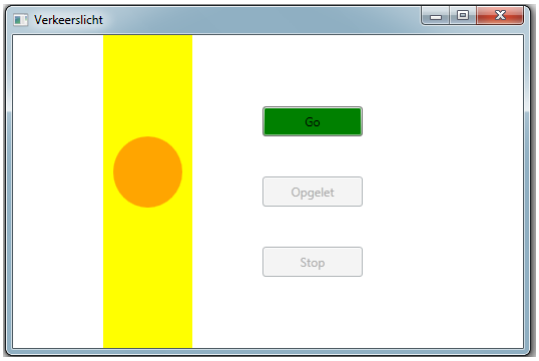
1.1 Toepassing op ...

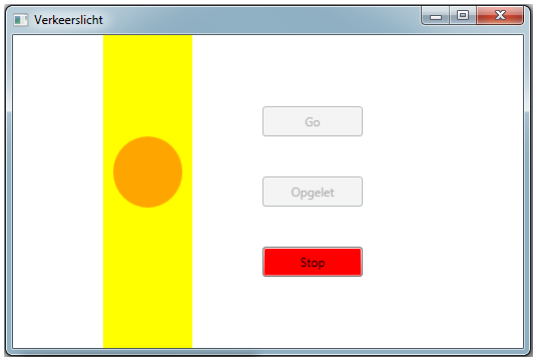
- Panels (Name, Orientation, HorizontalAlignment, VerticalAlignment, Margin, Background)
- Buttons (Name, Content, Height, Width, Background, Margin, IsEnabled, Click)
- Ellipse (Name, Fill, Height, HorizontalAlignment, VerticalAlignment, Width, Opacity = zichtbaarheid ➔ 0 = onzichtbaar, 1 = zichtbaar)

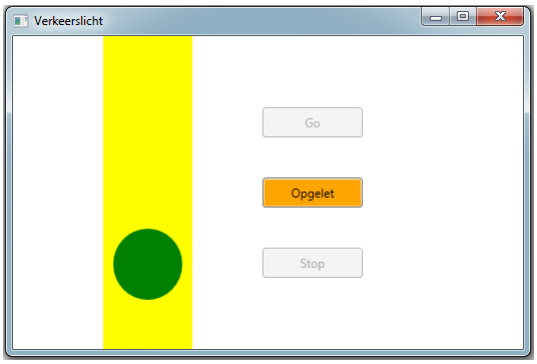
1.2 Opdracht

Maak een nieuwe applicatie met de naam *Verkeerslicht* en bouw via XAML-code het visuele deel op door gebruik te maken van bovenstaande *Controls*.









Functionaliteit:

3 knoppen : Go (groen), Opgelet (oranje), Stop (rood)

- beginsituatie: Rood licht is aan, Opgelet-knop is *Enabled*.

- oranje licht is aan: als het van groen komt, moet het rood worden, als het rood was, moet het groen worden

- groen licht is aan: moet via de oranje knop terug op rood gezet worden

2 BESTELBON PIZZA (HOOFDSTUK 3 - 4)

2.1 Toepassing op …

- Werken met tekst
- Buttons

2.2 Opdracht

Maak een nieuwe applicatie met de naam *PizzaWindow* en bouw via XAML-code het visuele deel op door gebruik te maken van reeds gekende *Controls*.

RadioButtons : small – medium- large : gedragen zich als een groep met *large* in de beginsituatie aangeduid.

De lijn eronder is een platte *Rectangle*.

CheckBoxen : van de ingrediënten zijn *tomaat* en *kaas* in de beginsituatie reeds aangevinkt en niet meer uitzetbaar. De andere kunnen vrij aan en afgevinkt worden.

ToggleButtons : “extra dikke korst” en “extra kaas” zijn *ToggleButtons* die in- of uitgedrukt kunnen worden.

De “hoeveelheid” is een *TextBlock* met daarachter een *Label* met in de beginsituatie een 1.
Daarachter staan 2 *Buttons* met een plusteken en een minteken op. Je kan variëren van 1 tot 10.

De *Button* met de pizza op en de tekst “BESTELLEN” geeft als resultaat onderaan in een *Label* een opsomming van wat je hebt gekozen.

2.3 Tips en Tricks

- Een *Panel* heeft een property *Children* die een collection geeft van alle onderliggende tags. Met een *foreach* kan je bv. door elk *Child* van die *collection* lopen bv. foreach (*FrameworkElement* kind in mijnPanel.Children) . . .
- Je kan het type van element testen door het object te vergelijken met zijn type bv. if (kind is CheckBox)
- *Casten* naar een bepaald type doe je door voor de variabele het gewilde type tussen haakjes te zetten bv. radioKind=(RadioButton)kind zet als het mogelijk is het huidige type van kind om in een *RadioButton* met de naam radioKind.
- Conversie van types: met de functie *Convert* kan je bv. een tekst omzetten naar een *integer* : *Convert.ToInt16(tekst)*
- De *ToString()* functie is bijna overal terug te vinden

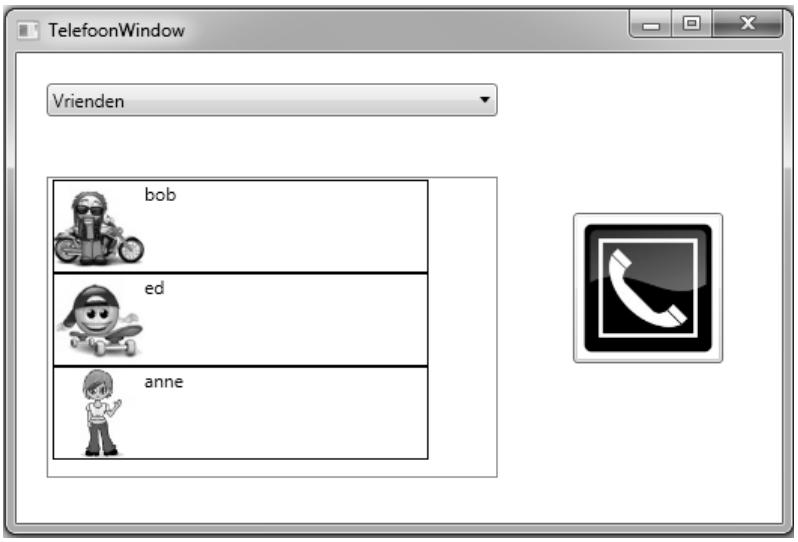
3 TELEFOON (HOOFDSTUK 5 - 6)

3.1 Toepassing op …

- ListBox, ComboBox en MessageBox

3.2 Opdracht

Maak een nieuwe applicatie met de naam *TelefoonWindow*.



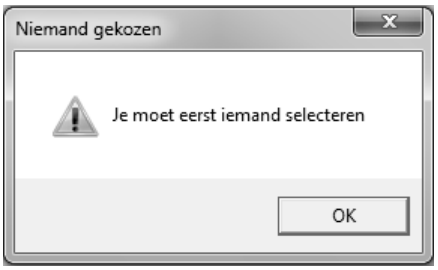
Bovenaan een ***ComboBox*** met als *Items* : Iedereen, Familie, Vrienden en Werk (deze vul je via code in).
Door een keuze te maken, laat je alleen de personen van die groep nog zien, of bij *Iedereen* alle personen.

Je maakt een klasse *Persoon* met volgende properties:
- Naam (= een string)
- Telefoonnr (= een string)
- Groep (= een string namelijk: Familie, Werk of Vrienden)
- Foto (= een *BitmapImage* : voeg daarvoor using System.Windows.Media.Imaging toe)

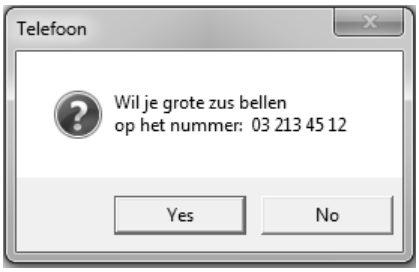
Daaronder een ***ListBox*** met als *Items* personen van de klasse *Persoon* die je met hun foto en naam laat zien. Gebruik voor de foto's de jpg-bestanden uit de map Telefoon door deze in een aangemaakte *Images*-map te kopiëren van het project en vul de andere properties naar eigen believen in.

Aan de rechterkant een ***Button*** met als uiterlijk het *telefoon.jpg* bestand.

Als je klikt en er is niemand in de *ListBox* geselecteerd, dan wordt dit ook gemeld:



Door te klikken op de *Button*, vraag je de gebruiker of hij deze persoon wil bellen:



In de *MessageBox* staat een titel, een *Yes*- en *No-Button*, een vraagteken en de vraag zelf in 2 regels.
Bij een positief antwoord laat je een beltoon horen op volgende manier:
- het bestand “PHONE.wav” kopiëer je naar de projectmap zelf
- voeg bovenaan de code using System.Media; toe
- de code zelf:

```
SoundPlayer speler = new SoundPlayer("PHONE.wav");
speler.Play();
```

Let op dat je de medecursisten niet laat schrikken !

4 PARKINGBON (HOOFDSTUK 7 - 8)

4.1 Toepassing op …

- MenuBar, ToolBar en StatusBar
- Bestandsoperaties

4.2 Opdracht

Maak een nieuwe applicatie met de naam *ParkingBon*.
Verwijder de standaardpagina en kopieer volgende bestanden vanuit de *TakenBestanden*-map naar de *Projectmap*:
- ParkingBonWindow.xaml
- ParkingBonWindow.xaml.cs

Pas het bestand App.xaml aan naar de juiste opstartpagina.
Maak een *images*-map en kopieer de volledige inhoud van de Parkingbon-images-map naar deze map.

De werking van het programma zou al in orde moeten zijn : `probeer maar uit.`

De inhoud van de

XAML-code:

De window bestaat uit een *Grid* van 4 rijen op 2 kolommen, die allemaal gekende controls bevatten.

Code-behind:

```
public partial class ParkingBonWindow : Window
{
    public ParkingBonWindow()
    {
        InitializeComponent();
        Nieuw();
    }
    private void Nieuw()
```

```
    {
        DatumBon.SelectedDate = DateTime.Now;
        AankomstLabelTijd.Content = DateTime.Now.ToLongTimeString();
        TeBetalenLabel.Content = "0 €";
        VertrekLabelTijd.Content = AankomstLabelTijd.Content;
    }


    private void Window_Closing(object sender,
System.ComponentModel.CancelEventArgs e)
    {
        if (MessageBox.Show("Programma afsluiten ?", "Afsluiten",
MessageBoxButton.YesNo, MessageBoxImage.Question, MessageBoxResult.No) ==
MessageBoxResult.No)
            e.Cancel = true;
    }

    private void minder_Click(object sender, RoutedEventArgs e)
    {
        int bedrag = Convert.ToInt32(TeBetalenLabel.Content.ToString().Replace("
€", ""));
        if (bedrag > 0)
            bedrag -= 1;
        TeBetalenLabel.Content = bedrag.ToString() + " €";
        VertrekLabelTijd.Content =
Convert.ToDateTime(AankomstLabelTijd.Content).AddHours(0.5 *
bedrag).ToLongTimeString();
    }

    private void meer_Click(object sender, RoutedEventArgs e)
    {
        int bedrag = Convert.ToInt32(TeBetalenLabel.Content.ToString().Replace("
€", ""));
        DateTime vertrekuur =
Convert.ToDateTime(AankomstLabelTijd.Content).AddHours(0.5 * bedrag);
        if (vertrekuur.Hour < 22)
            bedrag += 1;
        TeBetalenLabel.Content = bedrag.ToString() + " €";
        VertrekLabelTijd.Content =
Convert.ToDateTime(AankomstLabelTijd.Content).AddHours(0.5 *
bedrag).ToLongTimeString();
    }
}
```

De eigenlijke opdracht:

Er moet gewerkt worden via het *Command* attribuut (dan krijg je de shortcuts er automatisch bij).
- een *MenuBar*

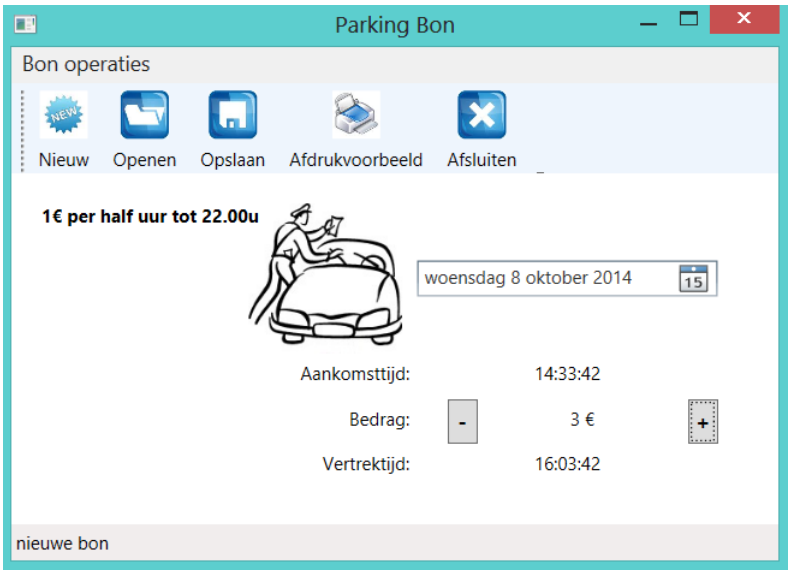
Je mag de bon alleen maar opslaan of afdrukvoorbeeld tonen als het bedrag groter is dan 0 (attribuut IsEnabled true of false) en dat zowel in de *MenuBar* als in de *ToolBar*.

- de *ToolBar*

- de *StatusBar*

Of als het een geopende of opgeslagen bon is:

Resultaat:

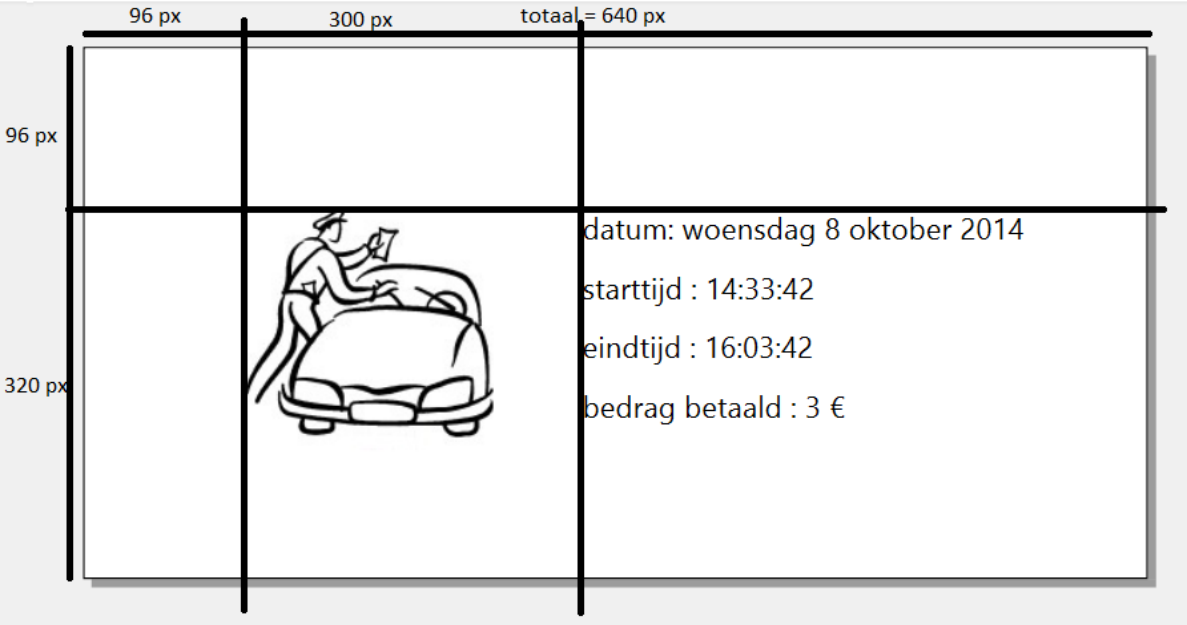


Naar de code toe:

- Bij Nieuw wordt de procedure Nieuw() aangeroepen = klaarzetten voor een nieuwe bon.
- De parkingbon kan opgeslagen en geopend worden door gebruik te maken van de *CommonDialog* objecten. De extensie van de bestanden zijn .bon en je mag alleen bestanden met die extensie laten zien.
  De standaard naamgeving is in volgend formaat:
  de datum met streepjes ertussen + het woordje “om” + de tijd met streepjes ertussen + de extensie “.bon” bijvoorbeeld “8-10-2014om14-33-42.bon”.
  De inhoud van het bestand is de datum, de begintijd, het bedrag en de eindtijd.
  bv. 8/10/2014
  14:33:42
  3 €
  16:03:42

Van zodra een bon hebt opgeslagen of geopend, komt in de *StatusBar* het volledige pad en de naam van het bestand te staan.

- het afdrukvoorbeeld tonen



Het formaat van de bon is 640px op 320px. Er zijn marges van 96px. De tekst staat op een linkermarge van 300px.

De tekst staat geschreven in 18punt met een dubbele regelafstand.

Je kan een *Image* op dezelfde manier aan de afdrukpagina toevoegen als een *TextBlock.*

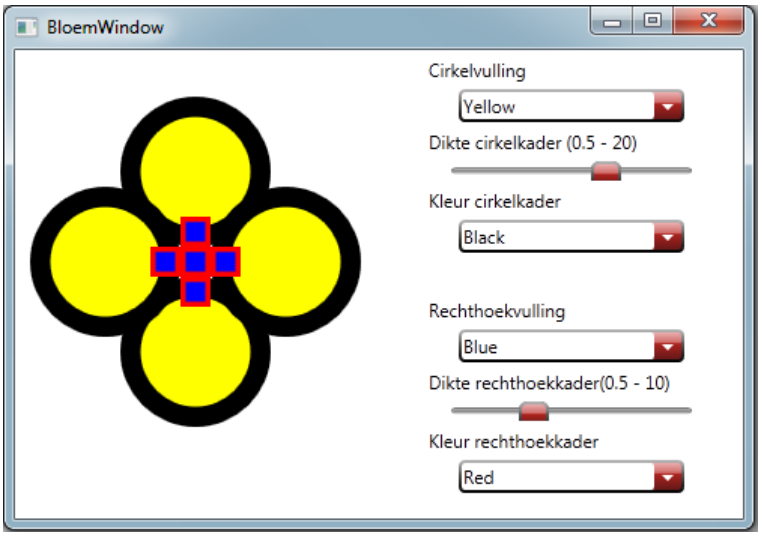
5 BLOEMSAMENSTELLING (HOOFDSTUK 9 - 10)

5.1 Toepassing op …

- Colors, Brushes enResources
- DataBinding

5.2 Opdracht

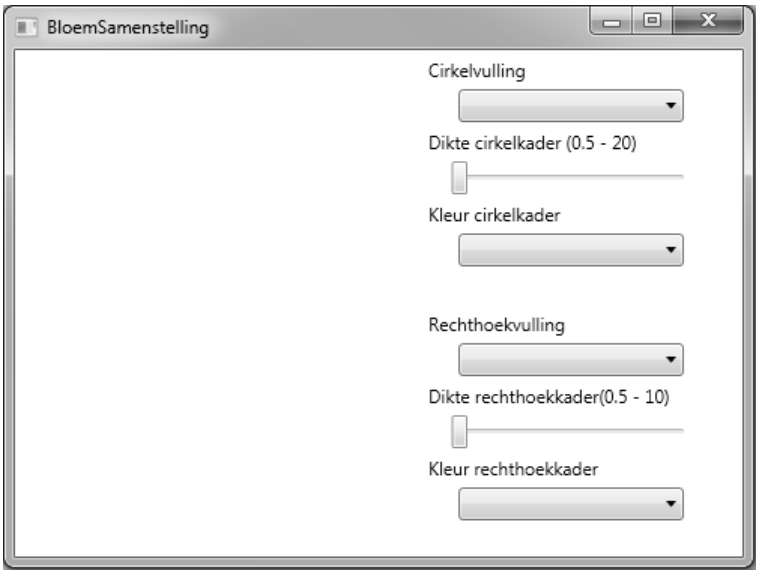
Eindresultaat:



Maak een nieuwe applicatie met de naam *BloemSamenstelling*.
Verwijder de standaardpagina en kopieer volgende bestanden vanuit de *TakenBestanden*-map naar de *Projectmap*:
- BloemWindow.xaml
- BloemWindow.xaml.cs
- Kleur.cs
- ShinyRed.xaml

en pas het bestand App.xaml aan naar de juiste opstartpagina.

Resultaat :



Aan de linkerzijde staan wel cirkels en rechthoeken, maar ze worden niet getoond omdat de kleuren niet gespecificeerd zijn.

Stap1. Vul via code de *ComboBoxen* met de “gekende kleuren”. Gebruik hiervoor de klasse *Kleur* en zorg ervoor dat bij het uitklappen de naam van de kleur getoond wordt (hiervoor kan je de XAML-code aanpassen).

Je kan gaan spieken in de cursus (want daar heb je iets gelijkaardig gedaan)

Stap2. Maak een *Resource Dictionary* aan met de naam *Stijlen.xaml* met als inhoud de *Style* van de
***Ellipse** : Height=100, Width=100, Fill* en *Stroke* gekoppeld aan de *ComboBoxen* en *StrokeThickness* gekoppeld aan de *Slider*.
Rectangle. *Height=20, Width=20, Fill* en *Stroke* gekoppeld aan de *ComboBoxen* en *StrokeThickness* gekoppeld aan de *Slider*.
Label : *ForeGround* op *Black*.

Stap3. Zorg ervoor dat deze *Resource Dictionary* (*Stijlen.xaml)* en de gekopieerde *Resource Dictionary (ShinyRed.xaml)* in de ganse applicatie gekend zijn.

Deze stappen zouden bij uitvoering het eindresultaat moeten tonen !

6 SCHUIFSPEL (HOOFDSTUK 11)

6.1 Toepassing op …

- Drag and Drop

6.2 Opdracht

Maak een nieuwe applicatie met de naam *SchuifSpelWindow*.
Verwijder de standaardpagina en kopieer volgende bestanden vanuit de *TakenBestanden*-map naar de *Projectmap*:
- SchuifSpelWindow.xaml
- SchuifSpelWindow.xaml.cs

Pas het bestand App.xaml aan naar de juiste opstartpagina.
Maak een *images*-map en kopieer de volledige inhoud van de Schuifspel-images-map naar deze map.

Om niet te lang met deze oefening bezig te zijn en alleen de essentie van *DragEnDrop* te programmeren is de volledige xaml-code en een gedeelte van de code-behind al toegevoegd:

XAML:

De window bestaat uit een *Grid* van 4 rijen op 4 kolommen, een *Button* om de *Images* door elkaar te zetten, en een *Button* om de oplossing te bekijken.

In het **code-behind** gedeelte is volgende code al klaargezet:

```
private void Window_Loaded(object sender, RoutedEventArgs e)
{
    Shuffle();
}

private void Check()
{
    int irij, ikolom, grij, gkolom;
    int aantalfout = 0;
    foreach (Image stukje in puzzelGrid.Children)
    {
         irij = Convert.ToInt16(stukje.Name.Substring(4, 1));
         ikolom = Convert.ToInt16(stukje.Name.Substring(5, 1));
         grij = Grid.GetRow(stukje);
         gkolom = Grid.GetColumn(stukje);
         if ((irij != grij) || (ikolom != gkolom))
         {
             aantalfout++;
         }
    }
    if (aantalfout == 0)
      Oplossing();
}
```

Check() voert een controle uit of alle stukjes op hun plaats staan. Is dit het geval dan wordt de procedure Oplossing() opgeroepen die het laatste stukje erbij zet.

```
private void OplossingButton_Click(object sender, RoutedEventArgs e)
{
    Oplossing();
}

private void Oplossing()
{
    puzzelGrid.Children.Clear();
    for (int r = 0; r <= 3; r++)
    {
       for (int k = 0; k <= 3; k++)
       {
            Image stuk = new Image();
            BitmapImage bi = new BitmapImage(new Uri(@"images/vdab" + r + k +
             ".jpg", UriKind.Relative));
            stuk.Name = "stuk" + r + k;
            stuk.Source = bi;
            zetImage(r, k, stuk);
        }
    }
}
```

Oplossing() zet alle stukjes op hun plaats, inclusief het laatste ontbrekende stukje.

```
private void zetImage(int rij, int kolom, Image zetstuk)
{
    Image stuk = new Image();
    stuk = zetstuk;
    Grid.SetColumn(stuk, kolom);
    Grid.SetRow(stuk, rij);
    if (stuk.Name == "stuk33")
    {
        stuk.Drop += puzzelGrid_Drop;
        stuk.AllowDrop = true;
    }
    else
    {
       stuk.MouseMove += stuk_MouseMove;
       AllowDrop = false;
    }
    puzzelGrid.Children.Add(stuk);
}
```

zetImage zet een *Image* (zetstuk) op een bepaalde *rij* en *kolom* van het *Grid*.
Als de *Image* het leeg stukje is, dan is het de bedoeling dat daar gedropt kan worden, dus wordt de *Drop-EventHandler* eraan gekoppeld, alsook de *AllowDrop* op *true* gezet.
Is het een stuk van de foto, dan wordt de *MouseMove-EventHandler* gekoppeld om het slepen mogelijk te maken.

```
private void Shuffle()
{
    puzzelGrid.Children.Clear();
    int[,] checken = new int[4, 4];
    for (int r = 0; r <= 3; r++)
    {
        for (int k = 0; k <= 3; k++)
        {
```

```
            checken[r, k] = 0;
        }
    }
    checken[3, 3] = 1;
    Random rnd = new Random();
    int rij, kolom;
    for (int r = 0; r <= 3; r++)
    {
        for (int k = 0; k <= 3; k++)
        {
            if (k < 3 || r < 3)
            {
                do
                {
                    rij = rnd.Next(0, 4);
                    kolom = rnd.Next(0, 4);
                } while (checken[rij, kolom] == 1);

                checken[rij, kolom] = 1;
                Image stuk = new Image();
                BitmapImage bi = new BitmapImage(new Uri(@"images/vdab" + r + k
                  + ".jpg", UriKind.Relative));
                stuk.Name = "stuk" + r + k;
                stuk.Source = bi;
                zetImage(rij, kolom, stuk);
            }
        }
    }

    Image leegstuk = new Image();
    BitmapImage bl = new BitmapImage(new Uri(@"images/leeg33.jpg",
        UriKind.Relative));
    leegstuk.Name = "stuk33";
    leegstuk.Source = bl;
    zetImage(3, 3, leegstuk);
}
```

Shuffle() bevat de code om de stukjes foto willekeurig (*Random*) in het *Grid* te plaatsen (behalve in de rechter onderhoek). Als laatste wordt het lege stukje in de rechter onderhoek geplaatst.

```
private void ShuffleButton_Click(object sender, RoutedEventArgs e)
{
    Shuffle();
}
```

Je kan de applicatie uitproberen (dit zou geen fouten mogen opleveren). Het *Drag* en *Droppen* gaat natuurlijk nog niet lukken.

De opdracht:

Via *DragEnDrop* kan je een stuk foto verplaatsen naar een langsliggend leeg vakje (dus testen of de *DropZone* er wel naast ligt). De twee stukjes verwisselen dan van plaats. Na elke *Drop* kan je via de *Check()*-procedure testen of alle stukjes juist staan.

Let op: bij elke *DragEnDrop* moet je onthouden vanuit welke cell in de *Grid* je vertrekt(Grid.GetRow(stuk) en Grid.GetColumn(stuk), en waar je het stuk wil plaatsen (om ze te kunnen verwisselen).

Je mag in de meegeleverde code natuurlijk ook code bijschrijven als dit nodig is.

```
private void stuk_MouseMove(object sender, MouseEventArgs e)
{
}
```

Hierin ga je testen of je effectief aan't slepen bent, en maak je het *DataObject* in orde.

```
private void puzzelGrid_Drop(object sender, DragEventArgs e)
{
}
```

Hierin ga je, als alles in orde is, het stuk foto met het lege stukje verwisselen.

7 PARKINGBONMVVM (HOOFDSTUK 12 - 13)

7.1 Toepassing op …

- Ribbon
- MVVM

7.2 Opdracht

Maak een nieuwe applicatie met de naam *ParkingBonMVVM*.

Bereid het project voor om te werken met MVVM Light.

Kopieer volgende bestanden vanuit de *TakenBestanden*-map naar de *Projectmap/View*:
- ParkingBonView.xaml
- ParkingBonView.xaml.cs

Maak in de *View*-map een map *images* en kopieer de Parkingbon-images-map naar deze map.

Wis alles uit de ViewModel-map en wis MainWindow.xaml in de hoofdmap.

Verwijder in het bestand App.xaml de *StartupUri* en alles van de *Application.Resources*.

De eigenlijke opdracht:

Maak een *Ribbon* met als onderdelen:

- QuickAccessToolbar:

  - Open en Opslaan *Buttons*

- ApplicationMenu

  - Nieuw : zet de datum op vandaag en de aankomst- en vertrektijd op de huidige tijd.

  - Opslaan mag alleen actief staan als het bedrag groter is dan 0 (dit geldt voor alle opslaan-mogelijkheden). Tip: als een variabele (Bedrag) gelijk is aan 0, komt dit overeen met *False* alle andere getallen met *True*.

  - Er moet een mogelijkheid voorzien worden om *Nieuw* ook in de *QuickAccesToolbar* op te nemen (dit hoeft niet in de de settings bewaard te worden)

- Bij het *Afsluiten* wordt de vraag gesteld of je de applicatie wil sluiten. Ook bij de sluitknop (x) en Alt F4 moet deze vraag gesteld worden. (Laat je niet vangen aan de functionaliteit : *Afsluiten* is een command, terwijl het effectief sluiten een *Closing*-event is !)

- RibbonTab

De parkingbon kan opgeslagen en geopend worden door gebruik te maken van de *CommonDialog* objecten. De extensie van de bestanden zijn .bon en je mag alleen bestanden met die extensie laten zien.
De standaard naamgeving is in volgend formaat:
de datum met streepjes ertussen + het woordje “om” + de tijd met streepjes ertussen + de extensie “.bon” bijvoorbeeld “16-09-2015om9-33.bon”.
De inhoud van het bestand is de datum, de begintijd, het bedrag en de eindtijd.
bv. 16/09/2015
9:33
6
12:33

8 VOORBEELDOPLOSSING: VERKEERSLICHT

8.1 XAML-code

```
<Window x:Class="Taak1.Verkeerslicht"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="Verkeerslicht" Height="350" Width="525">
<StackPanel Orientation="Horizontal" HorizontalAlignment="Center">
  <StackPanel Background="Yellow">
    <Ellipse Name="RoodLicht" Fill="Red" Height="71"
             HorizontalAlignment="Left" Margin="10" VerticalAlignment="Top"
             Width="69" />
    <Ellipse Name="OranjeLicht" Fill="Orange" Height="71"
             HorizontalAlignment="Left" Margin="10" VerticalAlignment="Top"
             Width="69" Opacity="0" />
    <Ellipse Name="GroenLicht" Fill="Green" Height="71"
             HorizontalAlignment="Left" Margin="10" VerticalAlignment="Top"
             Width="69" Opacity="0" />
  </StackPanel>
  <StackPanel Margin="50,0" VerticalAlignment="Center">
    <Button Name="ButtonGo" Content="Go" Height="30" Width="100"
            Background="Green" Margin="20" Click="ButtonGo_Click"
            IsEnabled="False"></Button>
    <Button Name="ButtonOpgelet" Content="Opgelet" Height="30" Width="100"
            Background="Orange" Margin="20"
            Click="ButtonOpgelet_Click"></Button>
    <Button Name="ButtonStop" Content="Stop" Height="30" Width="100"
            Background="Red" Margin="20" IsEnabled="False"
            Click="ButtonStop_Click"></Button>
  </StackPanel>
</StackPanel>
</Window>
```

8.2 Code-behind

```
namespace Taak1
{
public partial class Verkeerslicht : Window
{
  public Verkeerslicht()
  {
    InitializeComponent();
  }

  private void ButtonGo_Click(object sender, RoutedEventArgs e)
  {
    OranjeLicht.Opacity = 0;
    GroenLicht.Opacity = 1;
    ButtonOpgelet.IsEnabled = true;
    ButtonGo.IsEnabled = false;
  }

  private void ButtonStop_Click(object sender, RoutedEventArgs e)
  {
    OranjeLicht.Opacity = 0;
    RoodLicht.Opacity = 1;
```

```
    ButtonOpgelet.IsEnabled = true;
    ButtonStop.IsEnabled = false;
  }

 private void ButtonOpgelet_Click(object sender, RoutedEventArgs e)
 {
   if (GroenLicht.Opacity==1)
   {
      ButtonStop.IsEnabled = true;
      GroenLicht.Opacity = 0;
   }
   else
   {
      ButtonGo.IsEnabled = true;
      RoodLicht.Opacity = 0;
   }
   OranjeLicht.Opacity = 1;
   ButtonOpgelet.IsEnabled = false;
 }
}
}
```

9 VOORBEELDOPLOSSING: BESTELBON PIZZA

9.1 Xaml-code

```
<Window x:Class="Taak2.PizzaWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="PizzaWindow" Height="370" Width="525">
    <StackPanel Orientation="Vertical" HorizontalAlignment="Center" Width="500">
        <Label Content="Pizza bestellen" HorizontalAlignment="Center"
               FontSize="24"></Label>
        <StackPanel Orientation="Horizontal">
            <StackPanel Name="boxen" Orientation="Vertical" Width="250"
                        Margin="10">
                <RadioButton Name="small" Content="small" Margin="3"
                             GroupName="grootte"></RadioButton>
                <RadioButton Name="medium" Content="medium" Margin="3"
                             GroupName="grootte"></RadioButton>
                <RadioButton Name="large" Content="large" Margin="3"
                             GroupName="grootte" IsChecked="True"></RadioButton>
                <Rectangle Width="250" Height="1" Fill="Black"
                           Margin="0,10"></Rectangle>
                <CheckBox Name="tomaat" Content="tomaat" Margin="3"
                          IsChecked="True" IsEnabled="False"></CheckBox>
                <CheckBox Name="kaas" Content="kaas" Margin="3"
                          IsEnabled="False" IsChecked="True"></CheckBox>
                <CheckBox Name="ham" Content="ham" Margin="3"></CheckBox>
                <CheckBox Name="ananas" Content="ananas" Margin="3"></CheckBox>
                <CheckBox Name="salami" Content="salami" Margin="3"></CheckBox>
            </StackPanel>
            <StackPanel Orientation="Vertical" Width="200">
                <ToggleButton Name="extrakorst" Content="extra dikke korst"
                              Margin="5"></ToggleButton>
                <ToggleButton Name="extrakaas" Content="extra kaas"
                              Margin="5"></ToggleButton>
                <StackPanel Orientation="Horizontal">
                    <TextBlock Text="hoeveelheid:" VerticalAlignment="Center"
                               Width="80"></TextBlock>
                    <Label Name="aantalLabel" Content="1" Width="25"
                           VerticalAlignment="Center"></Label>
                    <StackPanel Margin="5">
                        <RepeatButton Name="meer" Content="+" Height="15"
                                      FontSize="8" Click="meer_Click"></RepeatButton>
                        <RepeatButton Name="minder" Content="-" Height="15"
                                      FontSize="8" Click="minder_Click"></RepeatButton>
                    </StackPanel>
                </StackPanel>
                <Button Width="90" Click="bestellen_Click">
                    <StackPanel>
                        <Image Source="pizza.jpg" Stretch="Fill" Height="75"
                               Width="75" Margin="0,10,0,0"></Image>
                        <Label Content="BESTELLEN" Width="75"
                               HorizontalAlignment="Center"></Label>
                    </StackPanel>
                </Button>
            </StackPanel>
        </StackPanel>
        <Label Name="bestelling" Width="500" Height="66"
               VerticalAlignment="Top"></Label>
```

```
    </StackPanel>
</Window>
```

9.2 Code-behind

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;

namespace Taak2
{
    /// <summary>
    /// Interaction logic for PizzaWindow.xaml
    /// </summary>
    public partial class PizzaWindow : Window
    {
        public PizzaWindow()
        {
            InitializeComponent();
        }

        private void meer_Click(object sender, RoutedEventArgs e)
        {
            int aantal = Convert.ToInt16(aantalLabel.Content);
            if (aantal < 10)
                aantal++;
            aantalLabel.Content = aantal.ToString();
        }

        private void minder_Click(object sender, RoutedEventArgs e)
        {
            int aantal = Convert.ToInt16(aantalLabel.Content);
            if (aantal > 1)
                aantal--;
            aantalLabel.Content = aantal.ToString();
        }

        private void bestellen_Click(object sender, RoutedEventArgs e)
        {
            string tekst = " U heeft "+ aantalLabel.Content +" ";

            string ingredienten = string.Empty;
            foreach (FrameworkElement kind in boxen.Children)
            {
                if (kind is RadioButton)
                {
                    if (((RadioButton)kind).IsChecked == true)
                        tekst += kind.Name + @" pizza('s) besteld met: ";
                }
                if (kind is CheckBox)
```

```
                    if (((CheckBox)kind).IsChecked == true)
                    ingredienten += kind.Name + ", ";
            }
            ingredienten = ingredienten.Substring(0, ingredienten.Length - 2);
            int k = ingredienten.LastIndexOf(",");
            ingredienten = ingredienten.Substring(0, k) + " en " +
ingredienten.Substring(k + 2);
            tekst += ingredienten +"\n";
            if (extrakorst.IsChecked == true)
                tekst += " met een extra dikke korst \n";
            if (extrakaas.IsChecked == true)
                tekst += " overstrooid met extra kaas";
            bestelling.Content = tekst;
        }
    }
}
```

10 VOORBEELDOPLOSSING: TELEFOON

10.1 Xaml-code

```
<Window x:Class="TaakTelefoon.TelefoonWindow"
      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
      Title="TelefoonWindow" Height="350" Width="525" Loaded="Window_Loaded">
  <StackPanel Orientation="Horizontal">
    <StackPanel Width="300" Margin="20">
      <ComboBox Name="ComboBoxGroepen"
                SelectionChanged="ComboBoxGroepen_SelectionChanged">
      </ComboBox>
      <Line Margin="0,20"></Line>
      <ListBox Name="ListBoxPersonen" Height="200">
        <ListBox.ItemTemplate>
          <DataTemplate>
            <Border BorderBrush="Black" BorderThickness="1" Width="250">
              <StackPanel Orientation="Horizontal">
                <Image Source="{Binding Path=Foto}" Stretch="Fill" Height="60"
                       Width="60"></Image>
                <TextBlock Text="{Binding Path=Naam}"></TextBlock>
              </StackPanel>
            </Border>
          </DataTemplate>
        </ListBox.ItemTemplate>
      </ListBox>
    </StackPanel>
    <Button Width="100" Height="100" Margin="30" Click="Button_Click">
      <Image Source="images\telefoon.jpg"></Image>
    </Button>
   </StackPanel>
</Window>
```

10.2 Code-behind

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;
using System.Media;


namespace TaakTelefoon
{
    /// <summary>
    /// Interaction logic for TelefoonWindow.xaml
    /// </summary>
    public partial class TelefoonWindow : Window
    {
```

```
        public TelefoonWindow()
        {
            InitializeComponent();
        }

        public List<Persoon> personen = new List<Persoon>();
        private void Window_Loaded(object sender, RoutedEventArgs e)
        {
            personen.Add(new Persoon("kleine zus", "03 213 45 12", "Familie",
new BitmapImage(new Uri(@"images\kleinezus.jpg",UriKind.Relative))));
            personen.Add(new Persoon("grote zus", " 03 213 45 12", "Familie",
new BitmapImage(new Uri(@"images\grotezus.jpg", UriKind.Relative))));
            personen.Add(new Persoon("vader", "02 12 45 78", "Familie", new
BitmapImage(new Uri(@"images\vader.jpg", UriKind.Relative))));
            personen.Add(new Persoon("tante non", "056 78 45 12", "Familie", new
BitmapImage(new Uri(@"images\tantenon.jpg", UriKind.Relative))));
            personen.Add(new Persoon("collega 1", "014 45 16 98", "Werk", new
BitmapImage(new Uri(@"images\collega1.jpg", UriKind.Relative))));
            personen.Add(new Persoon("collega 2", "03 86 54 79", "Werk", new
BitmapImage(new Uri(@"images\collega2.jpg", UriKind.Relative))));
            personen.Add(new Persoon("collega 3", "045 12 45 23", "Werk", new
BitmapImage(new Uri(@"images\collega3.jpg", UriKind.Relative))));
            personen.Add(new Persoon("bob", "012 45 37 58", "Vrienden", new
BitmapImage(new Uri(@"images\bob.jpg", UriKind.Relative))));
            personen.Add(new Persoon("ed", "065 43 29 75", "Vrienden", new
BitmapImage(new Uri(@"images\ed.jpg", UriKind.Relative))));
            personen.Add(new Persoon("anne", "065 43 29 76", "Vrienden", new
BitmapImage(new Uri(@"images\anne.jpg", UriKind.Relative))));
            foreach (Persoon dePersoon in personen)
            {
                ListBoxPersonen.Items.Add(dePersoon);
            }

            ComboBoxGroepen.Items.Add("Iedereen");
            ComboBoxGroepen.Items.Add("Familie");
            ComboBoxGroepen.Items.Add("Vrienden");
            ComboBoxGroepen.Items.Add("Werk");
            ComboBoxGroepen.SelectedIndex = 0;
        }

        private void ComboBoxGroepen_SelectionChanged(object sender,
SelectionChangedEventArgs e)
        {
            ListBoxPersonen.Items.Clear();
            foreach (Persoon dePersoon in personen)
            {
                if (ComboBoxGroepen.SelectedItem.ToString() == "Iedereen")
                {
                    ListBoxPersonen.Items.Add(dePersoon);
                }
                else
                {
                    if ((dePersoon.Groep ==
ComboBoxGroepen.SelectedItem.ToString()))
                    ListBoxPersonen.Items.Add(dePersoon);
                }
            }
        }


        private void Button_Click(object sender, RoutedEventArgs e)
```

```
        {
            if (ListBoxPersonen.SelectedIndex >= 0)
            {
                Persoon beller = (Persoon)ListBoxPersonen.SelectedItem;
                if (MessageBox.Show("Wil je " + beller.Naam + " bellen \nop het
nummer: " +
beller.Telefoonnr,"Telefoon",MessageBoxButton.YesNo,MessageBoxImage.Question) ==
MessageBoxResult.Yes)
                {
                    SoundPlayer speler = new SoundPlayer("PHONE.wav");
                    speler.Play();
                }
            }
            else
            {
                MessageBox.Show("Je moet eerst iemand selecteren","Niemand
gekozen",MessageBoxButton.OK,MessageBoxImage.Warning);
            }
        }
    }
}
```

11VOORBEELDOPLOSSING: PARKINGBON

11.1 Xaml-code

```
<Window x:Class="ParkingBon.ParkingBonWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        xmlns:custom="clr-namespace:ParkingBon"
        Title="Parking Bon" Height="379" Width="525"
        Closing="Window_Closing">

  <Window.CommandBindings>
    <CommandBinding Command="New"
      Executed="NewExecuted"></CommandBinding>
    <CommandBinding Command="Open"
      Executed="OpenExecuted"></CommandBinding>
    <CommandBinding Command="Save"
      Executed="SaveExecuted"></CommandBinding>
    <CommandBinding Command="PrintPreview"
      Executed="PrintPreviewExecuted"></CommandBinding>
    <CommandBinding Command="Close"
      Executed="CloseExecuted"></CommandBinding>
  </Window.CommandBindings>
  <DockPanel>

    <Menu Height="24" Name="MenuBalk" DockPanel.Dock="Top">
     <MenuItem Name="MenuOperaties" Header="Bon operaties" FontSize="14">
     <MenuItem Name="BonNieuw" Header="Nieuwe Bo_n" Command="New"
        FontSize="14"></MenuItem>
     <MenuItem Name="BonOpenen" Header="Bon _openen" Command="Open"
        FontSize="14"></MenuItem>
     <MenuItem Name="BonOpslaan" Header="Bon _opslaan" Command="Save"
        FontSize="14"></MenuItem>
     <MenuItem Name="BonAfdrukken" Header="Bon afdrukken"
        Command="PrintPreview" FontSize="14" IsEnabled="False"></MenuItem>
     <Separator></Separator>
     <MenuItem Name="Afsluiten" Header="Afsluiten" Command="Close"
        FontSize="14"></MenuItem>
    </MenuItem>
  </Menu>

  <ToolBarTray Height="60" DockPanel.Dock="Top">
  <ToolBar Name="ToolBarOperaties">
  <Button Name="NewButton" Command="New">
    <StackPanel>
      <Image Source="images\new.jpg" Stretch="Fill" Height="32"
          Width="32"></Image>
      <Label>Nieuw</Label>
    </StackPanel>
  </Button>
  <Button Name="OpenButton" Command="Open">
    <StackPanel>
      <Image Source="images\open.jpg" Stretch="Fill" Height="32"
          Width="32"></Image>
      <Label>Openen</Label>
    </StackPanel>
  </Button>
  <Button Name="SaveButton" Command="Save" IsEnabled="False">
    <StackPanel>
      <Image Source="images\save.jpg" Stretch="Fill" Height="32"
          Width="32"></Image>
```

```
      <Label>Opslaan</Label>
    </StackPanel>
  </Button>
  <Button Name="PrintPreviewButton" Command="PrintPreview"
       IsEnabled="False">
    <StackPanel>
      <Image Source="images\print.jpg" Stretch="Fill" Height="32"
          Width="32"></Image>
      <Label>Afdrukvoorbeeld</Label>
    </StackPanel>
  </Button>
  <Button Command="Close" HorizontalAlignment="Right">
    <StackPanel>
      <Image Source="images\close.jpg" Stretch="Fill" Height="32"
          Width="32"></Image>
      <Label>Afsluiten</Label>
    </StackPanel>
  </Button>
</ToolBar>
</ToolBarTray>

<StatusBar Name="BonStatus" DockPanel.Dock="Bottom" Height="24">
  <StatusBarItem Name="StatusItem"></StatusBarItem>
  </StatusBar>

<Grid Margin="20">

. . .

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

11.2 Code-behind

```
private void Nieuw()
{
    DatumBon.SelectedDate = DateTime.Now;
    AankomstLabelTijd.Content = DateTime.Now.ToLongTimeString();
    TeBetalenLabel.Content = "0 €";
    VertrekLabelTijd.Content = AankomstLabelTijd.Content;
    StatusItem.Content = "nieuwe bon";
    SaveEnAfdruk(false);
}

private void SaveEnAfdruk(Boolean actief)
{
    PrintPreviewButton.IsEnabled = actief;
    SaveButton.IsEnabled = actief;
    BonAfdrukken.IsEnabled = actief;
    BonOpslaan.IsEnabled = actief;
}

private void NewExecuted(object sender, ExecutedRoutedEventArgs e)
{
    Nieuw();
}

private void OpenExecuted(object sender, ExecutedRoutedEventArgs e)
{
    try
```

```
    {
        OpenFileDialog dlg = new OpenFileDialog();
        dlg.Filter = "Parkeerbonnen | *.bon";

        if (dlg.ShowDialog() == true)
        {
            using (StreamReader invoer = new StreamReader(dlg.FileName))
            {
                DatumBon.SelectedDate = Convert.ToDateTime(invoer.ReadLine());
                AankomstLabelTijd.Content = invoer.ReadLine();
                TeBetalenLabel.Content=invoer.ReadLine();
                VertrekLabelTijd.Content = invoer.ReadLine();
            }
            StatusItem.Content = dlg.FileName;
            SaveEnAfdruk(true);
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show("opslaan mislukt: " + ex.Message);
    }
}

private void SaveExecuted(object sender, ExecutedRoutedEventArgs e)
{
    try
    {
        SaveFileDialog dlg = new SaveFileDialog();
        DateTime tijd = (DateTime)DatumBon.SelectedDate;
        dlg.FileName = tijd.Day.ToString() + "-" + tijd.Month.ToString() +
          "-" + tijd.Year.ToString() + "om" +
          AankomstLabelTijd.Content.ToString().Replace(":", "-");
        dlg.DefaultExt = ".bon";
        dlg.Filter = "Parkeerbonnen | *.bon";

        if (dlg.ShowDialog() == true)
        {
            using (StreamWriter uitvoer = new StreamWriter(dlg.FileName))
            {
                uitvoer.WriteLine(tijd.ToShortDateString());
                uitvoer.WriteLine(AankomstLabelTijd.Content);
                uitvoer.WriteLine(TeBetalenLabel.Content);
                uitvoer.WriteLine(VertrekLabelTijd.Content);
            }
            StatusItem.Content = dlg.FileName;
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show("opslaan mislukt: " + ex.Message);
    }
}

private double vertPositie;

private void PrintPreviewExecuted(object sender, ExecutedRoutedEventArgs e)
{
    FixedDocument document = new FixedDocument();
    document.DocumentPaginator.PageSize = new Size(640,320);
    PageContent inhoud = new PageContent();
    document.Pages.Add(inhoud);
```

```
    FixedPage pagina = new FixedPage();
    inhoud.Child = pagina;

    pagina.Width = 640;
    pagina.Height = 320;
    Image logo = new Image();
    logo.Source = logoImage.Source;
    logo.Margin = new Thickness(96);
    pagina.Children.Add(logo);
    vertPositie = 96;
    pagina.Children.Add(Regel("datum: "+ DatumBon.Text));
    pagina.Children.Add(Regel("starttijd : "+AankomstLabelTijd.Content));
    pagina.Children.Add(Regel("eindtijd : "+VertrekLabelTijd.Content));
    pagina.Children.Add(Regel("bedrag betaald : "+TeBetalenLabel.Content));

    Afdrukvoorbeeld preview = new Afdrukvoorbeeld();
    preview.Owner = this;
    preview.AfdrukDocument = document;
    preview.ShowDialog();
}

private TextBlock Regel(string tekst)
{
    TextBlock deRegel = new TextBlock();
    deRegel.Margin = new Thickness(300, vertPositie, 96, 96);
    deRegel.FontSize = 18;
    vertPositie += 36; // 18 * 2
    deRegel.Text = tekst;
    return deRegel;
}

private void CloseExecuted(object sender, ExecutedRoutedEventArgs e)
{
    this.Close();
}


    private void minder_Click(object sender, RoutedEventArgs e)
    {
        int bedrag = Convert.ToInt32(TeBetalenLabel.Content.ToString().Replace("
€", ""));
        if (bedrag > 0)
            bedrag -= 1;
        TeBetalenLabel.Content = bedrag.ToString() + " €";
        VertrekLabelTijd.Content =
Convert.ToDateTime(AankomstLabelTijd.Content).AddHours(0.5 *
bedrag).ToLongTimeString();
            SaveEnAfdruk(!(bedrag == 0));
    }

    private void meer_Click(object sender, RoutedEventArgs e)
    {
        int bedrag = Convert.ToInt32(TeBetalenLabel.Content.ToString().Replace("
€", ""));
        DateTime vertrekuur =
Convert.ToDateTime(AankomstLabelTijd.Content).AddHours(0.5 * bedrag);
        if (vertrekuur.Hour < 22)
            bedrag += 1;
        TeBetalenLabel.Content = bedrag.ToString() + " €";
```

```
        VertrekLabelTijd.Content =
Convert.ToDateTime(AankomstLabelTijd.Content).AddHours(0.5 *
bedrag).ToLongTimeString();
        SaveEnAfdruk(!(bedrag == 0));
    }
}
```

11.3 Tweede WPF-Window (Afdrukvoorbeeld)

Je maakt een tweede WPF-Window aan met de naam *Afdrukvoorbeeld*.

XAML-code

```
<Window x:Class="ParkingBon.Afdrukvoorbeeld"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="Afdrukvoorbeeld" Height="300" Width="300">
    <DocumentViewer Name="printpreview"></DocumentViewer>
</Window>
```

Code-behind

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

    public IDocumentPaginatorSource AfdrukDocument
    {
        get { return printpreview.Document; }
        set { printpreview.Document = value; }
    }
}
```

12 VOORBEELDOPLOSSING: BLOEMSAMENSTELLING

12.1 Xaml-code (stap 1)

```
            . . .
        </Canvas>
        <StackPanel Orientation="Vertical" Width="200">
            <Label>Cirkelvulling</Label>
            <ComboBox Name="cirkelsKleuren" Width="150"
             DisplayMemberPath="Naam" SelectedValuePath="Naam">
            </ComboBox>
            <Label>Dikte cirkelkader (0.5 - 20)</Label>
            <Slider Name="diktecirkelSlider" Margin="20,0" Minimum="0.5"
            Maximum="20" SmallChange="0.5" IsSnapToTickEnabled="True"></Slider>
            <Label>Kleur cirkelkader</Label>
            <ComboBox Name="cirkelKaderKleuren" Width="150"
             DisplayMemberPath="Naam" SelectedValuePath="Naam">
            </ComboBox>
            <Label Margin="0,25,0,0">Rechthoekvulling</Label>
            <ComboBox Name="rechthoekenKleuren" Width="150"
             DisplayMemberPath="Naam" SelectedValuePath="Naam">
            </ComboBox>
            <Label>Dikte rechthoekkader(0.5 - 10)</Label>
            <Slider Name="dikterechthoekSlider" Margin="20,0" Minimum="0.5"
            Maximum="10" SmallChange="0.5" IsSnapToTickEnabled="True"></Slider>
            <Label>Kleur rechthoekkader</Label>
            <ComboBox Name="rechthoekKaderKleuren" Width="150"
             DisplayMemberPath="Naam" SelectedValuePath="Naam">
            </ComboBox>
        </StackPanel>
    </StackPanel>
</Window>
```

12.2 Code-behind (stap 1)

```
public BloemWindow()
{
    InitializeComponent();
    foreach (PropertyInfo info in typeof(Colors).GetProperties())
    {
        BrushConverter bc = new BrushConverter();
        SolidColorBrush deKleur =
            (SolidColorBrush)bc.ConvertFromString(info.Name);
        Kleur kleurke = new Kleur();
        kleurke.Borstel = deKleur;
        kleurke.Naam = info.Name;
        kleurke.Hex = deKleur.ToString();
        kleurke.Rood = deKleur.Color.R;
        kleurke.Groen = deKleur.Color.G;
        kleurke.Blauw = deKleur.Color.B;
        cirkelsKleuren.Items.Add(kleurke);
        rechthoekenKleuren.Items.Add(kleurke);
        cirkelKaderKleuren.Items.Add(kleurke);
        rechthoekKaderKleuren.Items.Add(kleurke);
    }
}
```

12.3 Stijlen.xaml (stap 2)

```
<ResourceDictionary
xmlns=http://schemas.microsoft.com/winfx/2006/xaml/presentation
xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml">

  <Style TargetType="{x:Type Ellipse}">
    <Setter Property="Height" Value="100"></Setter>
    <Setter Property="Width" Value="100"></Setter>

    <Setter Property="Fill"
      Value="{Binding ElementName=cirkelsKleuren,Path=SelectedValue}">
    </Setter>

    <Setter Property="Stroke"
      Value="{Binding ElementName=cirkelKaderKleuren, Path=SelectedValue}">
    </Setter>

    <Setter Property="StrokeThickness"
      Value="{Binding ElementName=diktecirkelSlider, Path=Value}">
    </Setter>
    </Style>

  <Style TargetType="{x:Type Rectangle}">
    <Setter Property="Height" Value="20"></Setter>
    <Setter Property="Width" Value="20"></Setter>

    <Setter Property="Fill"
      Value="{Binding ElementName=rechthoekenKleuren,Path=SelectedValue}">
    </Setter>
    <Setter Property="Stroke"
      Value="{Binding ElementName=rechthoekKaderKleuren, Path=SelectedValue}">
    </Setter>
    <Setter Property="StrokeThickness"
      Value="{Binding ElementName=dikterechthoekSlider, Path=Value}">
    </Setter>
  </Style>

  <Style TargetType="{x:Type Label}">
    <Setter Property="Foreground" Value="Black"></Setter>
  </Style>
</ResourceDictionary>
```

12.4 App.xaml (stap 3)

```
<Application x:Class="BloemSamenstelling.App"
             xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
             xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
             StartupUri="BloemWindow.xaml">
    <Application.Resources>
        <ResourceDictionary>
            <ResourceDictionary.MergedDictionaries>
                <ResourceDictionary Source="ShinyRed.xaml"></ResourceDictionary>
                <ResourceDictionary Source="Stijlen.xaml"></ResourceDictionary>
            </ResourceDictionary.MergedDictionaries>
        </ResourceDictionary>
    </Application.Resources>
</Application>
```

13 VOORBEELDOPLOSSING: SCHUIFSPEL

13.1 Code-behind

```
private int sleeprij, sleepkolom, legerij, legekolom;

private void Shuffle()
{

    ...
    legerij = 3;
    legekolom = 3;
}
```

```
private void stuk_MouseMove(object sender, MouseEventArgs e)
{
    if (e.LeftButton == MouseButtonState.Pressed)
    {
        Image stuk = (Image)sender;
        sleeprij = Grid.GetRow(stuk);
        sleepkolom = Grid.GetColumn(stuk);
        DataObject sleepstuk = new DataObject("sleepstuk", stuk);
        DragDrop.DoDragDrop(stuk, sleepstuk, DragDropEffects.Move);
    }
}
```

```
private void puzzelGrid_Drop(object sender, DragEventArgs e)
{
    if (e.Data.GetDataPresent("sleepstuk"))
    {
        if (geldig())
        {
            Image gesleeptstuk = (Image)e.Data.GetData("sleepstuk");
            Image dropstuk = (Image)sender;
            legerij = Grid.GetRow(dropstuk);
            legekolom = Grid.GetColumn(dropstuk);
            puzzelGrid.Children.Remove(gesleeptstuk);
            puzzelGrid.Children.Remove(dropstuk);
            zetImage(legerij, legekolom, gesleeptstuk);
            zetImage(sleeprij, sleepkolom, dropstuk);
            legerij = sleeprij;
            legekolom = sleepkolom;
            Check();
        }
    }
}

// leeg stuk moet naast het sleepstuk liggen
private Boolean geldig()
{
    if (((sleeprij + 1 == legerij) || (sleeprij - 1 == legerij)) &&
      (sleepkolom == legekolom))
    { return true; }
    else
    {
```

```
        if (((sleepkolom + 1 == legekolom) || (sleepkolom - 1 == legekolom)) &&
          (sleeprij == legerij))
        { return true; }
    }
    return false;
}
```

14 VOORBEELDOPLOSSING PARKINGBONMVVM

14.1 Model – ParkingBon.cs

```
public class ParkingBon
{
    public ParkingBon()
    {
        Datum = DateTime.Now;
        Aankomst = DateTime.Now;
        Vertrek = Aankomst;
        Bedrag = 0;
    }

    public DateTime Datum { get; set; }
    public DateTime Aankomst { get; set; }
    public DateTime Vertrek { get; set; }
    public int Bedrag { get; set; }
}
```

14.2 ViewModel – ParkingBonVM.cs

```
. . .
using System.Windows;
using GalaSoft.MvvmLight;
using GalaSoft.MvvmLight.Command;
using Microsoft.Win32;
using System.ComponentModel;
using System.IO;
. . .
public class ParkingBonVM : ViewModelBase
{
    private Model.ParkingBon parkingbon;

    public ParkingBonVM(Model.ParkingBon deParkingBon)
    {
        parkingbon = deParkingBon;
    }

    public DateTime Datum
    {
        get { return parkingbon.Datum; }
        set
        {
            parkingbon.Datum = value;
            RaisePropertyChanged("Datum");
        }
    }

    public DateTime Aankomst
    {
        get { return parkingbon.Aankomst; }
        set
        {
            parkingbon.Aankomst = value;
            RaisePropertyChanged("Aankomst");
        }
    }

    public DateTime Vertrek
```

```
    {
        get { return parkingbon.Vertrek; }
        set
        {
            parkingbon.Vertrek = value;
            RaisePropertyChanged("Vertrek");
        }
    }

    public int Bedrag
    {
        get { return parkingbon.Bedrag; }
        set
        {
            parkingbon.Bedrag = value;
            RaisePropertyChanged("Bedrag");
        }
    }


    public RelayCommand MeerCommand
    { get { return new RelayCommand(MeerBetalen); } }

    private void MeerBetalen()
    {
        if (Vertrek.Hour < 22)
            Bedrag ++;
        Vertrek = Aankomst.AddHours(0.5 * Bedrag);
    }


    public RelayCommand MinderCommand
    { get { return new RelayCommand(MinderBetalen); } }

    private void MinderBetalen()
    {
        if (Bedrag > 0)
            Bedrag--;
        Vertrek = Aankomst.AddHours(0.5 * Bedrag);
    }


    public RelayCommand NieuwCommand
    { get { return new RelayCommand(NieuweBon); } }

    private void NieuweBon()
    {
        Bedrag = 0;
        Datum = DateTime.Today;
        Aankomst = DateTime.Now;
        Vertrek = DateTime.Now;
    }

    public RelayCommand OpenenCommand
    { get { return new RelayCommand(OpenenBon); } }

    private void OpenenBon()
    {
        try
        {
            OpenFileDialog dlg = new OpenFileDialog();
```

```
            dlg.FileName = "";
            dlg.DefaultExt = ".bon";
            dlg.Filter = "Parkingbonnen |*.bon";

            if (dlg.ShowDialog() == true)
            {
                using (StreamReader bestand = new StreamReader(dlg.FileName))
                {
                    Datum = Convert.ToDateTime(bestand.ReadLine());
                    Aankomst = Convert.ToDateTime(bestand.ReadLine());
                    Bedrag = Convert.ToInt32(bestand.ReadLine());
                    Vertrek = Convert.ToDateTime(bestand.ReadLine());
                }
            }
        }
        catch (Exception ex)
        {
            MessageBox.Show("openen mislukt : " + ex.Message);
        }
    }

    public RelayCommand OpslaanCommand
    { get { return new RelayCommand(OpslaanBon); } }

    private void OpslaanBon()
    {
        try
        {
            string bestandsnaam;
            bestandsnaam = Datum.Day + "-" + Datum.Month + "-" + Datum.Year +
                     "om" + Aankomst.Hour + "-" + Aankomst.Minute + ".bon";
            SaveFileDialog dlg = new SaveFileDialog();
            dlg.FileName = bestandsnaam;
            dlg.DefaultExt = ".bon";
            dlg.Filter = "Parkingbonnen |*.bon";

            if (dlg.ShowDialog() == true)
            {
                using (StreamWriter bestand = new StreamWriter(dlg.FileName))
                {
                   bestand.WriteLine(Datum.ToShortDateString());
                   bestand.WriteLine(Aankomst.ToShortTimeString());
                   bestand.WriteLine(Bedrag);
                   bestand.WriteLine(Vertrek.ToShortTimeString());
                }
            }
        }
        catch (Exception ex)
        {
            MessageBox.Show("opslaan mislukt : " + ex.Message);
        }
    }

    public RelayCommand AfsluitenCommand
    { get { return new RelayCommand(AfsluitenApp); } }


    public void AfsluitenApp()
    {
        Application.Current.MainWindow.Close();}
    public RelayCommand<CancelEventArgs>AfsluitenEvent
```

```
    { get { return new RelayCommand<CancelEventArgs>(OnWindowClosing); } }


    public void OnWindowClosing(CancelEventArgs e)
    {
        if (MessageBox.Show("Afsluiten", "Wilt u het programma sluiten ?",
MessageBoxButton.YesNo, MessageBoxImage.Question, MessageBoxResult.No) ==
MessageBoxResult.No)
            e.Cancel = true;
    }
}
```

14.3 View - ParkingBonView.cs

```
<RibbonWindow x:Class="ParkingBonMVVM.View.ParkingBonView"
  xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
  xmlns:gebruiker="clr-namespace:ParkingBonMVVM.ViewModel"
  xmlns:i="clr-namespace:System.Windows.Interactivity;
    assembly=System.Windows.Interactivity"
  xmlns:gala="http://www.galasoft.ch/mvvmlight”
  Title="ParkingBon" Height="450" Width="525">

    <i:Interaction.Triggers>
        <i:EventTrigger EventName="Closing">
            <gala:EventToCommand Command="{Binding AfsluitenEvent}"
                                 PassEventArgsToCommand="True" />
        </i:EventTrigger>
    </i:Interaction.Triggers>

  <DockPanel LastChildFill="True">
    <Ribbon DockPanel.Dock="Top">

      <Ribbon.QuickAccessToolBar>
        <RibbonQuickAccessToolBar>
          <RibbonButton SmallImageSource="images\open.jpg"
                    Label="Openen"
                    Command="{Binding OpenenCommand}">
           </RibbonButton>
          <RibbonButton SmallImageSource="images\save.jpg"
                    Label="Opslaan"
                    Command="{Binding OpslaanCommand}"
                    IsEnabled="{Binding Bedrag}">
           </RibbonButton>
        </RibbonQuickAccessToolBar>
      </Ribbon.QuickAccessToolBar>

      <Ribbon.ApplicationMenu>
        <RibbonApplicationMenu SmallImageSource="images\bestand.png">
           <RibbonApplicationMenuItem Header="Nieuw"
                    ImageSource="images\new.jpg”
                    QuickAccessToolBarImageSource="images\new.jpg"
                    Command="{Binding NieuwCommand}">
          </RibbonApplicationMenuItem>
          <RibbonApplicationMenuItem Header="Openen"
                    ImageSource="images\open.jpg"
                    Command="{Binding OpenenCommand}">
           </RibbonApplicationMenuItem>
          <RibbonApplicationMenuItem Header="Opslaan"
                    ImageSource="images\save.jpg"
```

```
                Command="{Binding OpslaanCommand}"
                IsEnabled="{Binding Bedrag}">
        </RibbonApplicationMenuItem>
        <RibbonApplicationMenu.FooterPaneContent>
          <RibbonButton SmallImageSource="images\close.jpg"
                HorizontalAlignment="Right"
                Label="Afsluiten"
                Command="{Binding AfsluitenCommand}">
          </RibbonButton>
        </RibbonApplicationMenu.FooterPaneContent>
      </RibbonApplicationMenu>
    </Ribbon.ApplicationMenu>

    <RibbonTab Header="Bestand">
      <RibbonGroup>
        <RibbonButton LargeImageSource="images\new.jpg"
                      Label="Nieuw"
                      Command="{Binding NieuwCommand}">
        </RibbonButton>
        <RibbonButton LargeImageSource="images\open.jpg"
                      Label="Openen"
                      Command="{Binding OpenenCommand}">
        </RibbonButton>
        <RibbonButton LargeImageSource="images\save.jpg"
                      Label="Opslaan" Command="{Binding OpslaanCommand}"
                      IsEnabled="{Binding Bedrag}">
        </RibbonButton>
      </RibbonGroup>
    </RibbonTab>
  </Ribbon>

  <Grid Margin="20">
    <Grid.RowDefinitions>
      <RowDefinition Height="120"></RowDefinition>
      <RowDefinition Height="30"></RowDefinition>
      <RowDefinition Height="30"></RowDefinition>
      <RowDefinition Height="30"></RowDefinition>
    </Grid.RowDefinitions>
    <Grid.ColumnDefinitions>
      <ColumnDefinition Width="250"></ColumnDefinition>
      <ColumnDefinition Width="225"></ColumnDefinition>
    </Grid.ColumnDefinitions>
        <Image Name="logoImage" Source="images\parkingbon.jpg" Grid.Row="0"
          Grid.Column="0" HorizontalAlignment="Right" Margin="0,0,10,0"
          Width="105"></Image>
        <TextBlock Text="1€ per half uur tot 22.00u" FontWeight="Bold">
        </TextBlock>
        <DatePicker Name="DatumBon" Grid.Column="1" ToolTip="datum selecteren"
          SelectedDateFormat="Long" VerticalAlignment="Center"
          Margin="20,45,0,50" SelectedDate="{Binding Datum}"
          Height="24"></DatePicker>
        <Label Name="AankomstLabel" Grid.Row="1" Grid.Column="0"
          HorizontalAlignment="Right" Width="83" Content="Aankomsttijd:"></Label>
        <TextBlock Name="AankomstTextBlock" Grid.Row="1" Grid.Column="1"
          HorizontalAlignment="Center" VerticalAlignment="Center" Width="200"
          Text="{Binding Aankomst, StringFormat=hh:mm:ss}"
          Height="16"></TextBlock>
        <Label Name="BedragLabel" Grid.Row="2" Grid.Column="0"
          HorizontalAlignment="Right" Width="50" Content="Bedrag:"></Label>
        <StackPanel Grid.Row="2" Grid.Column="1" Orientation="Horizontal">
            <RepeatButton Margin="20,0"  Name="minder" Width="20" Content="-"
          FontSize="14" FontWeight="ExtraBold"
```

```
            Command="{Binding MinderCommand}"></RepeatButton>
              <TextBlock Name="TeBetalenTextBlock" Width="100"
                 Text="{Binding Bedrag,
      ConverterCulture={x:Static glob:CultureInfo.CurrentCulture}, StringFormat=c}"
            HorizontalAlignment="Center" VerticalAlignment="Center"></TextBlock>
              <RepeatButton Margin="20,0"  Name="meer" HorizontalAlignment="Center"
            Width="20" Content="+" FontSize="14" FontWeight="ExtraBold"
            Command="{Binding MeerCommand}"></RepeatButton>
          </StackPanel>
          <Label Name="VertrekLabel" Grid.Row="3" Grid.Column="0"
            HorizontalAlignment="Right" Width="68" Content="Vertrektijd:"></Label>
          <TextBlock Name="VertrekTextBlock" Grid.Row="3" Grid.Column="1"
            HorizontalAlignment="Center" VerticalAlignment="Center" Width="200"
            Text="{Binding Vertrek, StringFormat=hh:mm:ss}"
             Height="16"></TextBlock>
    </Grid>
  </DockPanel>
</RibbonWindow>
```

14.4 App.xaml.cs

```
protected override void OnStartup(StartupEventArgs e)
{
    base.OnStartup(e);
    Model.ParkingBon mijnParkingBon = new Model.ParkingBon();
    ViewModel.ParkingBonVM vm = new ViewModel.ParkingBonVM(mijnParkingBon);
    View.ParkingBonView mijnParkingBonView = new View.ParkingBonView();
    mijnParkingBonView.DataContext = vm;
    mijnParkingBonView.Show();
}
```

15 COLOFON

Sectorverantwoordelijke:

Cursusverantwoordelijke:

Medewerkers: Chris Van Loon

Versie: September 2015

Nummer dotatielijst: