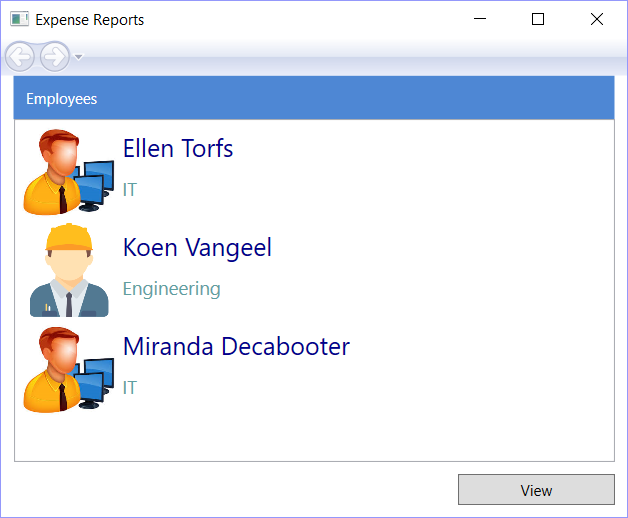
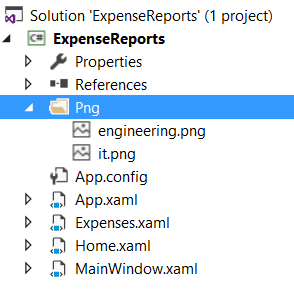
Deze oefening is een uitbreiding van het voorbeeld uit de presentatie.

# Uitbreiding 1

De listbox op de eerste pagina moet naast de naam ook het departement bevatten en per departement een bijhorend icoon. Je krijgt dan bijvoorbeeld:



De iconen (engineering.png en it.png) vind je terug in de map bestanden. Voeg deze toe aan je applicatie in bijvoorbeeld een map “Png”

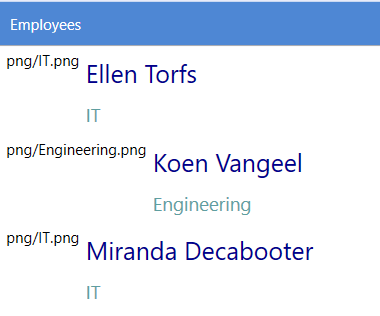


Vervolgens is de eerste stap het toevoegen van de naam van het departement in een extra label. Om de naam van de werknemer en het departement onder elkaar te laten verschijnen, plaats je beide labels in een verticaal georiënteerd stackpanel.

Het toevoegen van de afbeelding vraagt een combinatie van 2 WPF-controls een TextBlock en een Image. Om de afbeelding naast de labels te plaatsen, nest je het vorige stackpanel in een nieuw horizontaal georiënteerd stackpanel.

Stap 1 is het toevoegen van een TextBlock waarin het pad naar het bestand verschijnt.

<TextBlock Name="ImageName" Visibility="Visible"   
 Text="{Binding XPath=@Department, StringFormat=png/{0}.png}" />



Stap 2 bestaat uit het toevoegen van een Image control die als source gebruik maakt van de Tekst-property in de TextBlock-control. ElementName bevat de naam van de wpf-control waaruit de property Text moet worden opgehaald.

<Image Width="75" Source="{Binding Text, ElementName=ImageName}" />

Zorg er tenslotte nog voor dat van de Text-property in de TextBlock-control niet getoond wordt. Merk op dat je de property Visibility niet op ‘Hidden’ kan zetten maar dat je voor de optie ‘Collapsed’ moet kiezen, wil je de insprong bij Koen Vangeel vermijden.

# Uitbreiding 2

Uitbreiding 2 toont een alternatief voor het doorgeven van parameters tussen pages of windows in wpf.

In het voorbeeld wordt het geselecteerde item in de listbox naar de expenses page als parameter doorgegeven aan de constructor. In deze uitbreiding maken we gebruik van een property Employee gedefinieerd in Expenses.xaml en vullen we deze property op met de geselecteerde werknemer.

Stap 1 is het aanmaken van de property Employee in Expenses.xaml. Via de setter wordt net als in het voorbeeld de DataContext van de pagina ingesteld.

public partial class Expenses : Page

{

public Expenses()

{

InitializeComponent();

}

public object Employee

{

set { DataContext = value; }

}

}

Vervolgens krijgt de property Employee een waarde in home.xaml

private void Button\_Click(object sender, RoutedEventArgs e)

{

Expenses expensesPage = new Expenses();

expensesPage.Employee = ListBoxEmployees.SelectedItem;

NavigationService.Navigate(expensesPage);

}

Variant op deze werkwijze is dat je zoals in het voorbeeld een custom constructor gebruikt waarin de property Employee wordt meegegeven als parameter. Je krijgt dan onderstaande code in Expenses.xaml en Home.xaml.

//constructor in Expenses.xaml

public Expenses(object employee): this()

{

Employee = employee;

}

//oproepen constructor in Home.xaml

private void Button\_Click(object sender, RoutedEventArgs e)

{

//methode 2 via custom constructor

Expenses expensesPage = new Expenses(ListBoxEmployees.SelectedItem);

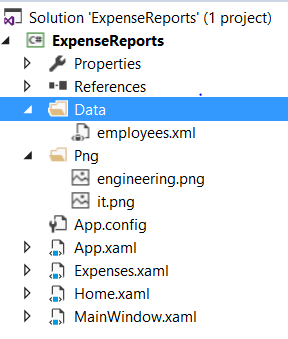
NavigationService.Navigate(expensesPage);

}

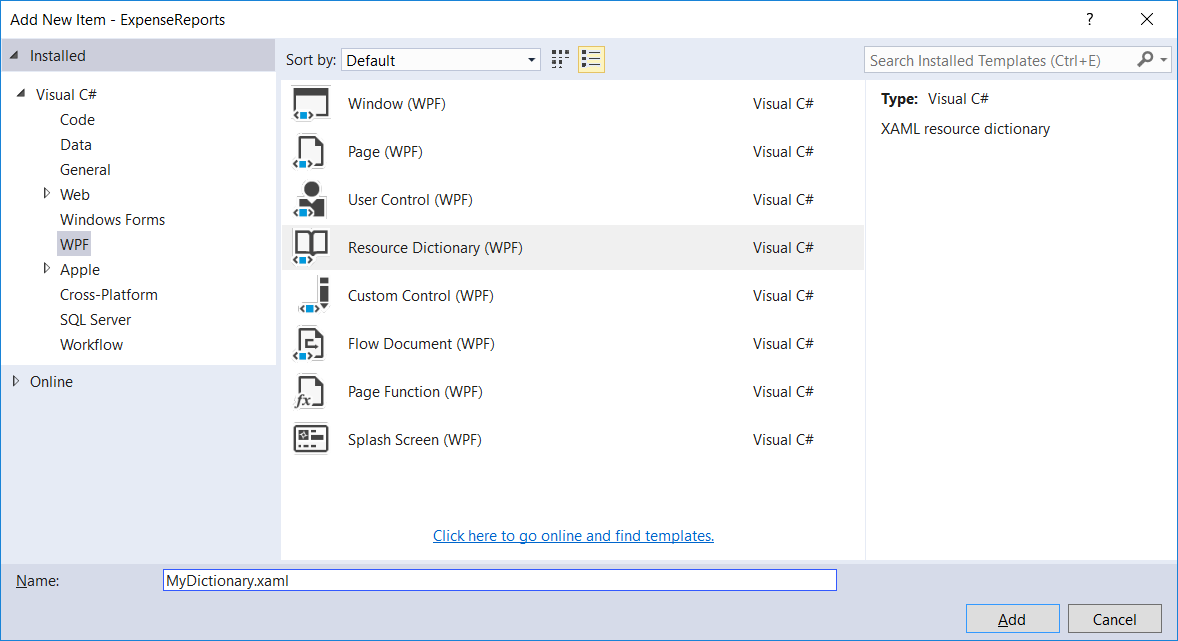
# Uitbreiding 3

Maak een map Data aan in je applicatie. Voeg aan deze map een nieuwe XML-file toe employees.xml met daarin de XML-data die nu inline zijn opgenomen in XAML.

Verwijder de XML-data uit XAML en maak gebruik van het Source-attribuut van de XMLDataProvider om de koppeling naar het externe bestand te leggen.



# Uitbreiding 4

Zorg nu voor 1 stijl in je ganse applicatie. Voeg hiervoor een **Resource Dictionary** toe.  
  


Knip vervolgens de stijlkenmerken weg uit expenses.xaml en plak deze in de dictionary.

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

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:local="clr-namespace:ExpenseReports">

<Style TargetType="{x:Type DataGridColumnHeader}">

<Setter Property="Height" Value="35" />

<Setter Property="Padding" Value="5" />

<Setter Property="Background" Value="#4E87D4" />

<Setter Property="Foreground" Value="White" />

</Style>

</ResourceDictionary>

Knip vervolgens de stijlkenmerken weg uit expenses.xaml en plak deze in de dictionary.

Om de dictionary nu bruikbaar te maken in de applicatie moet deze gedefinieerd worden in app.xaml. Voeg hiervoor onderstaande lijnen toe aan **app.xaml.**

<Application x:Class="ExpenseReports.App"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:local="clr-namespace:ExpenseReports"

StartupUri="MainWindow.xaml">

<Application.Resources>

<ResourceDictionary>

<ResourceDictionary.MergedDictionaries>

<ResourceDictionary Source="MyDictionary.xaml"/>

</ResourceDictionary.MergedDictionaries>

</ResourceDictionary>

</Application.Resources>

</Application>