Wat is Data binding

Data binding is het verbinden van eigenschappen van twee objecten. Dit doe je door de BindingContext van het doelobject in te stellen naar de bron. De bron is je andere object waar de data zich in bevindt.

Een voorbeeld van een label binding met een textbox:

Label label = new Label ();

label.SetBinding (Label.TextProperty, "Name");

label.BindingContext = new {Name = "John Doe", Company = "Xamarin"};

<Label Text="{Binding Value}"

Name=”Label”/>

<Entry Text="{ Binding EntryInput}"

FontAttributes="Bold"

FontSize="Large"

HorizontalOptions="Center"

VerticalOptions="CenterAndExpand"/>

Doordat de label tekst property als waarde de waarde van de binding krijgt, zal de input van de tekst synchroon zijn met de tekst van de label.

In het volgende paragraaf word er uitgelegd hoe dat gebeurt.

Waarom Databinding.

Om te weten waarom je databinding zou gebruiken raden we aan om eerst te kijken naar het hoofdstuk: Wat is databinding?

Er zitten zowel voordelen als nadelen aan databinding, voordat je databinding gaat gebruiken raden we je aan om deze voordelen en nadelen goed door te lezen.

Voordelen:   
- Makkelijk om te gebruiken in .net.  
- Aanrader als je live op de hoogste gehouden wilt worden.   
- Werkt goed bij simpele interfaces die niet veel anders doet dan data weergeven.

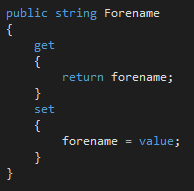
Nadelen:   
- Code kan ongestructureerd worden.   
- Er kunnen met databinding geen berekening uit worden gevoerd.

Voordat je databinding gaat gebruiken is het dus eerst van belang dat je weet wat je gaat maken. Het kan in sommige gevallen meer tijd gaan kosten dan dat het oplevert.

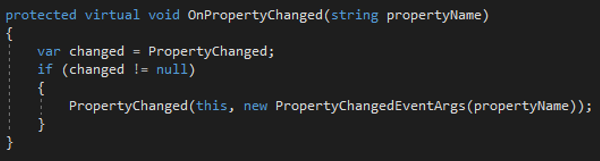
Implementeren van data-binding in Xamarin

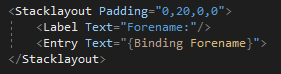
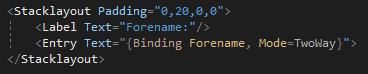
In dit gedeelte gaan we het hebben over hoe je data-binding kan gaan implementeren in Xamarin.

Als eerst gaan we het hebben over de implementatie in de CS file achter je XAML file. Om te beginnen moet we het ViewModel laten overerven van INotifyPropertyChnged. Dit doe je zo:  
  
  
  
Zoals uitgelegd in paragraaf {PLAATS PARAGRAAF NUMMMER HIER!} bind je met data-binding een property aan een andere property. Hierdoor zal je voor elke property die je gebind wilt hebben in je CS file een public property aanmaken met een get set en hier ook een private property voor maken. Dit doe je zo:

(Wanneer we deze property later binden in de XAML zal automatisch de set of get van de gebinde property aanroepen.)

Hierna zul je een methode en een event moeten aanmaken voor wanneer de set aangeroepen wordt van de gebinde variabele. Dit doe je zo:  
  
  
(Dit zal mogelijk al automatische toegevoegd zijn wanneer we de eerste stap hadden uitgevoegd zo niet voeg dit alsnog toen.)  
  
  
(Deze methode zal moeten worden aangeroepen bij elke set van een gebind property en deze updaten voor alle elementen in de view die het gebinde property gebuikt.)  
  
Als laatst zal je in de CS file een aanroep moeten doen aan de methode in elke set van elk gebinde property. Dit doe je zo:  
  
  
(Dit zal de vorige methode die we hebben gemaakt aanroepen.)  
  
Hierna moeten we in de Mainpage.cs dit toevoegen onder de InitializeComponent:  
  
  
(Dit zal er voor zorgen dat de view in staat is data-binding te gebruiken.)

Als laatst moet we in de XAML de property gaan binden aan die in de CS file. Dit doe je zo:  
  
  
(Dit zal ervoor zorgen dat er een label met een textbox verschijnt. In de textbox zal dan de property waarde te zien zijn die momenteel gebind is hieraan, in dit geval “Forename”.)  
  
Data-binding in Xamarin kan one-way of two-way dit wordt uitgelegd in paragraaf {PLAATS PARAGRAAF NUMMMER HIER!}. Om dit aan te geven moet je dit achter de binding aangeven. Dit doe je zo:  
  
  
(Hier geef je dan aan of je het TwoWay of OneWay data-binding wilt hebben. Als je dit niet aangeeft is het standaard TwoWay.)  
  
Als je al deze stappen hebt uitgevoerd heb je een werkend voorbeeld van databinding, en weet je hoe je dit overal kan implementeren.

MVVM :

Zet dit in je Content Page

“xmlns:sys="clr-namespace:System;assembly=mscorlib"”

Dit verbind sys met DLL (mscorlib) windows lib.  
MSCORLIB, (Microsoft Core Library)geeft je de mogelijkheid on in de XAML designer functies te gebruiken die je anders zou gebruiken in de C# code.

het verbonden item SYS wordt gebruikt om de huidige datum op te vragen maar statisch gemaakt en het blijvend binden. Dit houd dus ook in dat het blijft updaten.

Zie de code als volgt.

<StackLayout BindingContext="{x:Static sys:DateTime.Now}">

<Label Text="{Binding Year, StringFormat='The year is {0}'}" />

<Label Text="{Binding StringFormat='The month is {0:MMMM}'}" />

<Label Text="{Binding Day, StringFormat='The day is {0}'}" />

<Label Text="{Binding StringFormat='The time is {0:T}'}" />

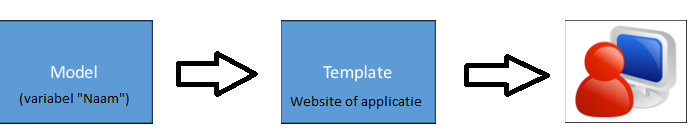
</StackLayout>

Door het aan de Stack Layout het te binden kan je dus alles wat verbonden moet zijn met het sleutelwoord “Binding” de functies gebruiken die het object heeft die je meegeeft.

One way binding.

One way binding betekend simpel gezegt dat data alleen maar opgehaald mag worden. Wanneer de data veranderd, veranderd dit ook in de data die op het beeld staat.

Stel je hebt een variabel genaamt “Naam” waarin een naam is opgeslagen, kan deze dus niet aangepast worden als er een one way binding is toegepast, maar alleen maar opgehaald worden.



Een voorbeeld van hoe one way binding toegepast kan worden:

