|  |
| --- |
|  |
| Inhoudsopgave |
|  |
|  |

**GROEP 2 LAYOUT**

May 29, 2017

Opgesteld door: Janneke van Veltom

Inhoudsopgave

Bestelapplicatie

Inhoud

[Introductie 2](#_Toc483835080)

[Lijst lay-out 3](#_Toc483835081)

[Scroll view 4](#_Toc483835082)

[Grid lay-out 5](#_Toc483835083)

[Doorsturen naar een nieuw form 6](#_Toc483835084)

[Bronnen 7](#_Toc483835085)

# Introductie

De bestelapplicatie dient mogelijkheden te hebben om uit een reeks van producten te kiezen en die te bestellen. Tijdens deze workshop komt het onderdeel lay-outs in Xamarin naar voren. Wij gaan u vertellen over het gebruik van lijst lay-out, het gebruik van scroll view, het gebruik van een grid lay-out en het doorsturen naar nieuwe form. In deze workshop worden enkele voorbeelden gegeven over het gebruik van deze onderdelen in Xamarin en welke voorbeelden u beter niet kunt gebruiken.

# Lijst lay-out

ListView is een belangrijk UI component van Android applicaties, die overal gebruikt wordt voor 'short lists' vanuit de menu opties tot aan 'long lists' van bijvoorbeeld contacten of internetbrowser favorieten.

ListView biedt een simpele manier om een 'scroll list' van rijen te presenteren die geformateerd met een ingebouwde stijl kunnen worden.

Ook heeft ListView een Adapter nodig om data in te kunnen laden die in rijen worden weergeven.

## Toevoegen items aan lay-out

De makkelijkste manier om een items toe te voegen, doe je doormiddel van een array gevuld met strings te vullen.

**Hier een voorbeeld:**

var listView = new ListView();

listView.ItemsSource = new string[]{

"mono",

"monodroid",

"monotouch",

"monorail",

"monodevelop",

"monotone",

"monopoly",

"monomodal",

"mononucleosis"

};

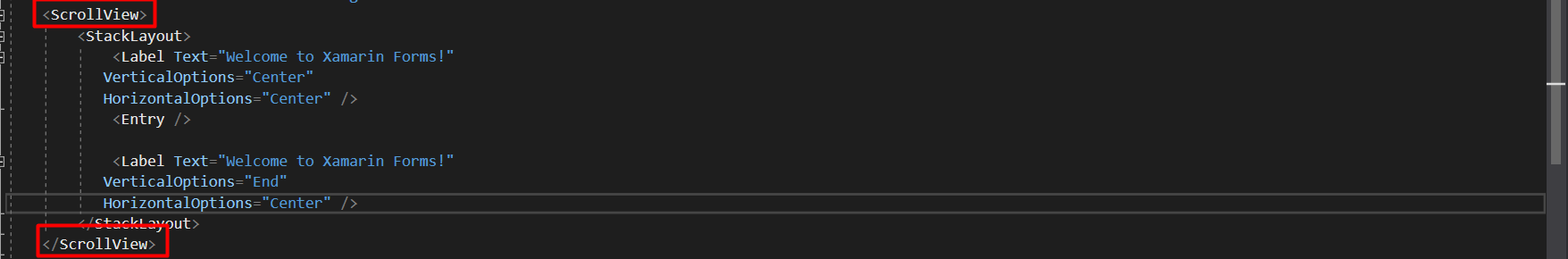
Hier wordt een nieuw ListView aangemaakt, waarbij de ItemsSource bestaat uit een string.

De waardes worden hierbij ingevuld.

# Scroll view

De ScrollView is een geweldige layout-optie één die je bijna op elke pagina wil gebruiken. Het zorgt ervoor dat in principe je pagina zo lang mogelijk kan maken.

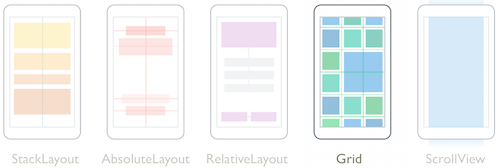
Om een ScrollView te implementeren heb je maar 2 simpele regels code nodig!



Simple! Het resultaat zorgt ervoor dat je naar onder en boven kan swipen!

# Grid lay-out

## Introductie

 Grid lay-out is een mogelijkheid om een pagina weer te geven door middel van rijen en kolommen. Deze rijen en kolommen kun je zelf indelen op grote. Anders dan HTML tabellen, is Grid uitsluitend bedoeld om inhoud uit te leggen.

Grid kan worden gebruikt voor het weergeven van je pagina in een raster. Dit kan handig zijn in de volgende omstandigheden:

* Ordenen van knoppen.
* Organiseren van knoppen / keuzes in een raster, zoals bijvoorbeeld de iOS- of Android startschermen.
* Arrangeren van de weergaven zodat alles op gelijke grootte is, zoals bijvoorbeeld in sommige werkbalken.

Alle code bij deze workshop zal in XAML zijn.

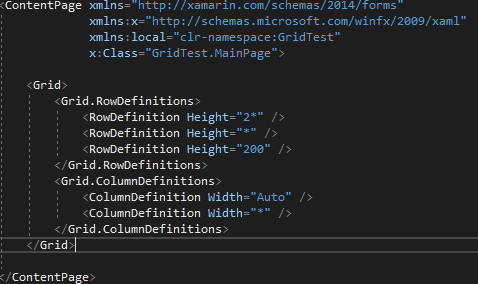
## Rijen en kolommen toevoegen

Rij- en kolominformatie wordt opgeslagen in de eigenschappen van RowDefinitions & ColumnDefinitions van Grid. Dit is een verzameling van RowDefinition en ColumnDefinition-objecten. RowDefinition zorgt de hoogte en de ColumnDefinition zorgt voor de breedte van de tabel. De opties voor hoogte en breedte zijn als volgt:

* Auto  
  Maakt automatisch de maten om de rij of kolom aan te passen. Word gegeven als GridUnitType.Auto in C# of als Auto in XAML.
* Proportional  
  Maakt de maten van de rijen en kolommen als een deel van de resterende ruimte. Gegeven als een waarde en GridUnitType.Star in C# en als #\* in XAmL, waarbij # de gewenste warde is. Een rij of kolom worden opgegeven met \*, en zal er voor zorgen dat de beschikbare ruimte wordt gevuld.
* Absolute  
  Maakt de maten van de rijen en kolommen met een specifieke hoogte en breedte waardes. Word gegeven als GridUnitType.Absolute in C# en als # in XAML, waarbij # uw gewenste waarde is.

Als we ervan uitgaan dat een app 3 rijen en 2 kolommen nodig heeft. De onderste rij wil je bijvoorbeeld 200px lang zijn en de bovenste rij wil je 2 keer zo lang als de middelste rij. De linker kolom moet breed genoeg zijn voor alle inhoud en de rechter kolom zal de resterende ruimte bevatten.

*Hieronder een voorbeeld van bovenstaande opmaak.*

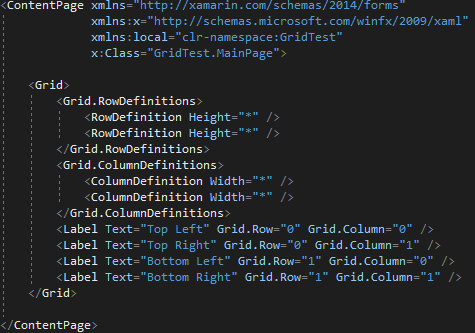


## Weergaven plaatsen in Grid

Om weergaven in een raster te zetten, moet worden deze toegevoegd als ‘kinderen’. Dit betekend dat er moet worden meegeven in welke rij of kolom ze behoren.

In XAML kan er gebruik worden gemaakt van Grid Row en Grid Column. Dit kan helpen met de plaatsing te specificeren. De locatie van Grid Row en Grid Column wordt gebaseerd op lijsten van rijen en kolommen, op basis van nul. Dit betekent dat in een 4x4-rooster de linker bovenste cel (0,0) is en de rechter onderkant (3,3) is.

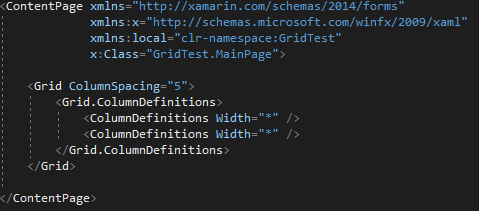
*Hieronder een voorbeeld van bovenstaande opmaak.*



## Ruimte

Columspacing wordt gebruikt om de afstand tussen de rijen en kolommen in het raster aan te passen. RowSparing zorgt voor de hoeveelheid ruimte tussen de rijen. Hieronder zal een stukje voorbeeld volgen wat betreft een raster met 2 kolommen, 1 rij en 5 px tussenruimte tussen de kolommen.

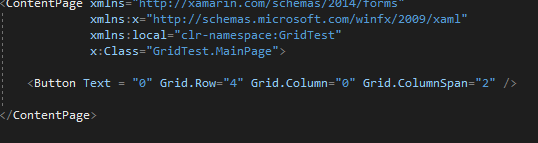
*Hieronder een voorbeeld van bovenstaande opmaak.*



## Gebruik van spans

Vaak bij het werken met een raster is er een element dat meer dan één rij of kolom moet bezetten. In XAML kan een button als volgt worden toegevoegd.

*Hieronder een voorbeeld van bovenstaande opmaak.*



# Doorsturen form

`App.Navigation.PushAsync(new MyCustomContentPage());`

Met dit stukje code kan je een naar een nieuwe page door in de pushasync een nieuwe class aan te maken met de page die je wilt openen. Bijvoorbeeld

als je een class Secondpage hebt kun je App.Navigation.PushAsync(new Secondpage()); gebruiken. De code maakt dan een nieuwe page aan en switched hiernaartoe. Ook kan je new page veranderen in een al bestaande page.

# Bronnen

<https://developer.xamarin.com/guides/xamarin-forms/user-interface/layouts/grid/>