VBA voor Doe Het Zelvers deel 2

Handleiding van Helpmij.nl

Auteur: Leofact

Januari 2014

VBA voor Doe Het Zelvers deel 2

VBA voor Doe Het Zelvers is een reeks artikelen, bedoelt voor mensen die met VBA in Excel aan de slag willen om taken te automatiseren, of om deze toegankelijk te maken voor gebruikers met weinig Excel kennis. VBA is een volwaardige programmeertaal, met een woordenschat en een zinsopbouw (syntaxis). Om deze goed te leren wordt aangeraden om hierover boeken te lezen.

Bijvoorbeeld uit deze willekeurige selectie.

Vorige aflevering

In de eerste aflevering werd uitgelegd wat een macro is en hoe Excel ingesteld moet worden om met macro's te werken. Ook is uitgelegd hoe je met de Macro recorder een macro kunt opnemen en op welke manier deze macro vervolgens zichtbaar wordt in de VBA editor.

In deze aflevering

In deze aflevering wordt de opgenomen code onder de loep genomen. Uitgelegd wordt wat een Procedure is en wat een Function is. In vogelvlucht wordt de opbouw van VBA, als Object Model, uitgelegd met een korte uitleg van de begrippen Methodes en Eigenschappen. Ook wordt besproken hoe en waar code kan worden ingevoerd met uitleg van de auto aanvulling: IntelliSense. De code wordt tot de essentie teruggebracht en er wordt stil gestaan bij het aanpassen van de code.
Als laatste wordt behandeld dat een Procedure kan worden gestart met behulp van sneltoetsen of met een zelfgemaakte knop.

De code

De macro die in de vorige aflevering werd opgenomen, was een rij namen (A2:A8) selecteren en deze vervolgens sorteren. Na opname werd in de Macro editor (te openen met ALT + F11) de volgende code zichtbaar:

```
Sub Macro1()
' Macro1 Macro
'
    Range("A2:A8").Select
      ActiveWorkbook.Worksheets("Blad1").Sort.SortFields.Clear _
        ActiveWorkbook.Worksheets("Blad1").Sort.SortFields.Add _
        Key:=Range("A2:A8"), _
        SortOn:=xlSortOnValues, Order:=xlAscending, DataOption:=xlSortNormal
    With ActiveWorkbook.Worksheets("Blad1").Sort
        .SetRange Range("A2:A8")
        .Header = xlGuess
        .MatchCase = False
        .Orientation = xlTopToBottom
        .SortMethod = xlPinYin
        .Apply
    End With
End Sub
```

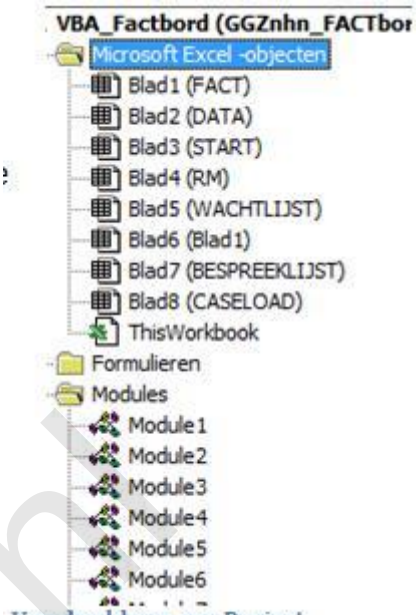
Als eerste valt op dat de simpele handeling, namelijk het selecteren en sorteren, flink wat regels tekst blijkt op te leveren. Even schrikken misschien, maar het is minder ingewikkeld dan het op het eerste gezicht lijkt. De oorzaak hiervan kan gevonden worden in de manier waarop de macrorecorder de code genereert op basis van de handelingen die opgenomen zijn. De recorder weet niet wat er exact bedoelt wordt en genereert daarom de code heel uitgebreid.

Procedures, Functies en het Project

Nu eerst even wat algemene VBA kennis.
De code die je ziet is een Procedure of Subroutine. Deze begint altijd met Sub “naam” en haakjes openen en sluiten. Om ervoor te zorgen dat er geen verwarring kan ontstaan, dient de naam altijd uniek te zijn en dus nooit een term die door Excel gebruikt wordt. Een procedure eindigt altijd met End Sub. Zo weet Excel dat het klaar is met de procedure. Een procedure is een serie handelingen die bepaalde taken uitvoert. Daarnaast is er ook de Function.
Schrijfwijze (Syntax):

Function Naam1()
End Function

De Function kun je vergelijken met een Functie, zoals deze in Excel zelf wordt gebruikt. Een ingevoerd gegeven wordt omgerekend naar een uitkomst.



Voorbeeld van een Project

Deze uitkomst wordt vervolgens ergens anders in het programma weer gebruikt. Procedures en Functions, die qua werking bij elkaar horen, worden in een Module opgenomen. Er kunnen meerdere Modules zijn. Het geheel van dit alles vormt het “Project”.

Grammatica of Syntaxis

Bij het ontwerpen van VBA had Microsoft een gebruiker in het achterhoofd die niet specifiek een programmeur hoefde te zijn. De taal is daarom behoorlijk vergevingsgezind. Er zijn vele wegen mogelijk die naar hetzelfde resultaat leiden.

Dat is prettig. Tegelijkertijd vormt het ook een valkuil. Het is belangrijk dat je er zorg voor draagt dat je je eigen code goed begrijpt. Ook als je na een paar jaar een Project nog eens onder handen zou moeten nemen. Gebruik daarom steeds namen die voor jou logisch zijn. Hetzelfde geldt voor de opbouw van de code. Dit is soms in strijd met een andere eis: namelijk om de code zo kort mogelijk te houden. Hoe korter de code hoe sneller Excel de opdrachten kan uitvoeren.

Het lijkt me voor de hand liggend, dat je weinig opschiet met snelle code waar je zelf geen weg meer in kan vinden. Excel helpt gelukkig ook nu, want het kan best veel tekst aan voordat er vertraging bij het omzetten (compileren) merkbaar is.

Nog meer taal; over Objecten, Methodes en Eigenschappen.

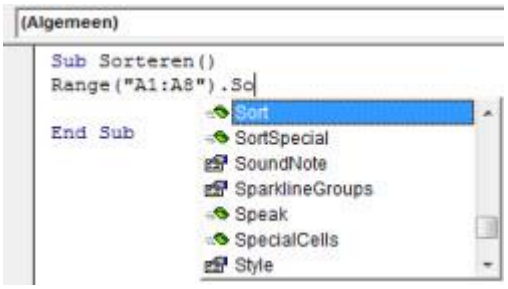
VBA heeft een duidelijke, gestructureerde opbouw. Er is sprake van een Object Model.
De vier belangrijkste Objecten zijn:

- Application
- Workbook
- Worksheet
- Range

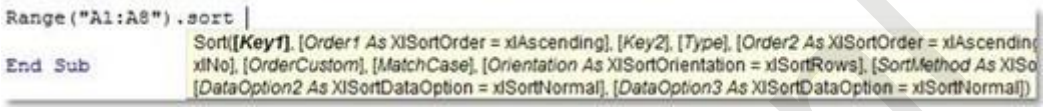
Deze objecten kennen we al van Excel zelf. De Objecten hebben leden, die specifiek bij dat Object horen. Leden kunnen ook weer Objecten zijn, maar ook Methodes en Eigenschappen. Methodes voeren iets uit en bij Eigenschappen gaat het om een instelling. Deze kunnen altijd worden uitgelezen en soms worden ingesteld. Bij het gebruikte code voorbeeld is Sort een methode van het Object

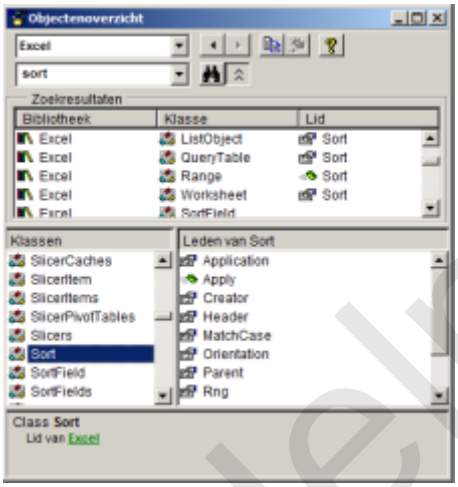
ActiveWorkbook.Worksheets("Blad1")

Eén van de eigenschappen van Sort is Orientation, met als waarde (Constante) xlTopToBottom.
Apply is een Methode, die op Sort wordt uitgevoerd.
Taaie kost. Gelukkig helpt IntelliSense ook hier. Wanneer een punt achter een Object gezet wordt, geeft IntelliSense de mogelijke leden weer.



Wanneer daar achteraan een spatie wordt ingegeven worden de leden en Constanten weergegeven (zie voorbeeld hieronder)





Een andere manier om de Objecten in een overzicht te zien is via het Objecten venster
(druk F2 in de VBA editor). Na selectie van een
Object zie je goed welke leden bij dit Object horen. Hier is ook goed de hiërarchie te zien. Sort kan lid zijn van Range of van Worksheet. Sort heeft zelf ook weer een aantal leden.
Objecten en leden zijn onderdeel van een bibliotheek. In dit geval de Excel bibliotheek. Er zijn meerdere bibliotheken, bijvoorbeeld voor VBA, voor Word etc.

Afspraken

Om een procedure of een Function overzichtelijk te houden is er een aantal afspraken:

- Een opdracht staat in principe steeds op één regel.
  Wanneer deze zo lang zou worden dat de code onoverzichtelijk wordt, kan de regel worden afgebroken. Hiervoor wordt het liggende streepje gebruikt na een spatie (zie de voorbeeld code).

- Met de apostrof (‘) wordt aangegeven dat de daarop volgende tekst een opmerking is. Excel voert dat niet uit als code. Dit biedt de mogelijkheid om commentaar aan de eigen code toe te voegen, zodat deze duidelijker leesbaar wordt. Commentaar tekst wordt automatisch groen.

- Excel probeert steeds de code die wordt ingevoerd te herkennen. Als dat lukt, vult Excel de tekst aan of geeft de keuze uit bepaalde mogelijkheden (waarden). Bepaalde stappen in de code worden blauw gekleurd en foutieve invoer wordt automatisch rood. Deze functie van intelliSense maakt het leven van een (aankomend) programmeur een stuk aangenamer.

```
If dTijd + TimeValue("00:05:00") >= dEinde Then lKleur = &H80FF&
    If dTijd > dEinde + TimeValue("00:30:00") Then
        TimerReset
            xit Sub
```

Terug naar de code.

Wanneer je nauwkeurig naar de eerder opgenomen code kijkt en wat Engels kent, herken je vast het één en ander.

Range("A2:A8").Select
De cellen A2 tot en met A8 werden geselecteerd, dus dat zal Select heten. Range is het geselecteerde bereik.

```
ActiveWorkbook.Worksheets("Blad1").Sort.SortFields.Clear
ActiveWorkbook.Worksheets("Blad1").Sort.SortFields.Add _
    Key:=Range("A2:A8"), _
    SortOn:=xlSortOnValues, Order:=xlAscending, DataOption:=xlSortNormal
```

Dit wordt iets moeilijker:
Sort is natuurlijk sorteren. De sorteervelden van het werkblad “blad1”, wat onderdeel is van het actieve (geopende) werkboek, worden gereset. Vervolgens wordt er een sorteer veld toegevoegd en we herkennen onze range weer als sleutel. Er gaat op waardes gesorteerd worden en de volgorde zal opklimmend zijn.

```
With ActiveWorkbook.Worksheets("Blad1").Sort
        .SetRange Range("A2:A8")
        .Header = xlGues
        .MatchCase = False
        .Orientation = xlTopToBottom
        .SortMethod = xlPinYin.Apply

End With
```

With geeft aan dat er een blok komt, dat bij het genoemde object hoort. In dit geval dus: ActiveWorkbook.Worksheets("Blad1"). Met een voorafgaande punt wordt aangegeven, dat dit bij het genoemde object hoort.

.SetRange Range("A2:A8"); stelt het ons inmiddels bekende bereik in.
.Header = xlGuess; Excel zal raden of er een kolomkop boven de kolom staat.
.MatchCase = False; Excel maakt bij het sorteren geen onderscheidt tussen hoofd- en kleine letters.
.Orientation = xlTopToBottom; er wordt van boven naar beneden gesorteerd.
.SortMethod = xlPinYin; Chinese karakters worden fonetisch gesorteerd.
.Apply; pas dit alles toe (er wordt eindelijk gesorteerd.)
End With; stopt het blok dat bij ActiveWorkbook.Worksheets("Blad1").Sort hoort.

Moet je dit alles echt uit je hoofd weten? Nee, gelukkig niet. Type het begrip in het Help vak rechtsboven in de VBA editor en je krijgt de benodigde uitleg te zien. Zo kwam ik ook achter de betekenis van XlPinYin. Want eerlijk; ik had geen idee. Is dit alles nu nodig om te sorteren? Nee absoluut niet. De volgende Procedure doet precies hetzelfde:

```
Sub Sorteren()
    Blad1.Range("A2:A8").Sort Range("A2")
End Sub
```

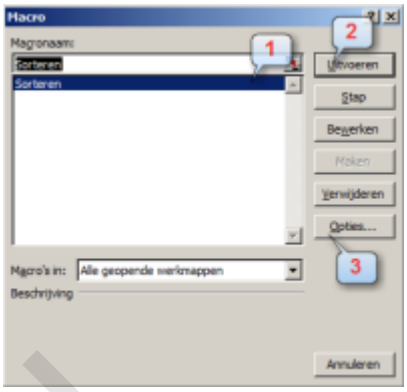
Hier wordt het bekende bereik op de bovenste waarde uit de kolom gesorteerd. Verder wordt alles aan de standaard sorteerinstellingen van Excel over gelaten.
Nu is goed te zien, hoe één en ander naar wens is aan te passen.

```
Sub Sorteren()
    Blad1.Range("A2:C8").Sort Range("B2")
End Sub
```

In deze code worden drie opeenvolgende kolommen gesorteerd. De volgorde wordt bepaald door de gegevens van de middelste kolom. Interessant wordt het als je de Range dynamisch kan aanpassen. Daarvoor kun je variabelen gebruiken.

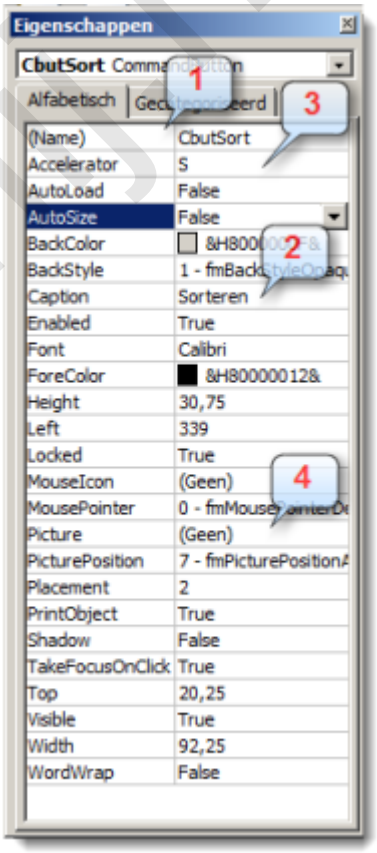
Hierover meer in de volgende aflevering.

Procedure gebruiken.



De simpele procedure is nu klaar. Hoe ga je die nu gebruiken? In de VBA editor kun je op F5 drukken wanneer de cursor ergens in de macro staat. Dat is niet echt handig als je in Excel aan het werk bent.
En nog minder handig als iemand anders jouw macro moet starten. Vanuit Excel zelf kan de macro gestart worden in het venster macro's (ALT + F8). Dubbelklik op de macro (1) of klik op uitvoeren (2) en ja hoor, de kolom wordt gesorteerd. Dat is nog niet echt gebruiksvriendelijk. Het gaat al gemakkelijker door een sneltoetscombinatie aan de macro toe te wijzen.

Druk daarvoor opties (3). Toets vervolgens de gewenste combinatie. Bijvoorbeeld CTRL + SHFT + S. Er kan hier alleen voor Annuleren worden gekozen. De toetscombinatie wordt echter wel opgeslagen. De macro is nu te starten door in het geopende werkblad op de toetscombinatie te drukken. Handig, maar je moet wel de combinatie onthouden.
Om de te starten macro zichtbaar te maken kun je een knop gebruiken. Ga hiervoor naar de ontwikkelaars tab en klik op Ontwerp modus. Klik vervolgens op invoegen en selecteer de knop. Neem wel de knop van de ActiveX objecten. Deze zijn uitgebreider aan te passen. De cursor verandert in een draadkruis en hiermee kun je op de gewenste plek in het werkblad de knop "tekenen". Klik rechts op de knop en selecteer Eigenschappen. Pas de volgende eigenschappen aan: Name, wordt CbutSort (1) (of een voor jou duidelijke naam). Caption (de tekst op de knop) wordt Sorteren (2).
Om het af te maken zet je de Accelerator op S (3). De S op de knop wordt onderstreept en de knop is nu te activeren met de menu toetscombinatie ALT + s. Klik nu weer met rechts op de knop en kies Programmacode weergeven. Er wordt een macro zichtbaar die reageert op een klik op de knop. Dit is een Event, of gebeurtenis macro. Zet daar de eigen code tussen, zoals hieronder. Voortaan zal de code worden uitgevoerd na een druk op de knop.




```
Private Sub CbutSort_Click()
'sorteer macro
    Blad1.Range("A2:C8").Sort Range("B2")
End Sub
```

Tip: Vergeet niet om de ontwerpmodus uit te zetten. Anders gebeurt er niets.

Een andere mogelijkheid is om de knop een afbeelding weer te laten geven.

Maak daarvoor een geschikte afbeelding. Klik naast Picture (4) en vervolgens op het knopje met de drie stippen. Selecteer dan je afbeelding in het bestandsvenster. De knop kan er dan zo uit zien:

Tip: Macro's starten niet als Excel in tekstinvoer modus staat (herkenbaar aan de knipperende tekst cursor in een cel)

Dit was het voor deze aflevering.
De volgende keer gaan we verder met aanpassingsmogelijkheden in de Procedure. Daarvoor worden o.a. de begrippen Range en Variabele behandeld.