VBA voor Doe het Zelvers deel 5

Handleiding van Helpmij.nl

Auteur: leofact

April 2014

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 kunnen leren wordt het aangeraden om hierover boeken te lezen. Bijvoorbeeld uit deze willekeurige selectie.

Vorige aflevering

In de vorige aflevering werd besproken hoe een UserForm kan worden ingezet om tekstinvoer eenvoudiger te maken. Daarnaast werden er verschillende functies besproken om tekst te manipuleren.

In deze aflevering

Neem ik de Event of Gebeurtenis Macro onder de loep met verschillende voorbeelden welke ook in de bijlage worden uitgewerkt. De bijlage bevat onder andere een simpele klok en een manier om ongewenst knippen en plakken te tegen te gaan.

Event Macro

De Event of Gebeurtenis Macro is een macro die reageert op een bepaalde gebeurtenis van het werkboek, een werkblad of een besturingselement. Elk van deze objecten heeft een reeks voor-gedefinieerde Event Macro's. Deze zijn terug te vinden in de Object Browser (**3**).

Na selectie van het betreffende onderdeel links in de Project Verkenner (**1**), kan bij (**2**) het object zelf of één van de leden daarvan worden geselecteerd. Bij (**3**) kan dan één van de bijbehorende Event Macro's worden geselecteerd. De dropdown-lijst geeft overzichtelijk weer welke Event Macro's er beschikbaar zijn. Dit kan verschillen per object. Een Event Macro is altijd een Private Sub. Dat wil zeggen dat de macro alleen geldig is voor het object waar de macro bij hoort. De Event kan in gang gezet (getriggerd) worden door de gebruiker. Bijvoorbeeld door het openen van het werkboek, een ander werkblad selecteren, klikken op een knop, drukken op sneltoetsen enzovoorts. Een Event kan ook door de code zelf worden getriggerd. Bij het schrijven van Event Macro's zijn er onder meer twee belangrijke zaken om rekening mee te houden:

- Tekst invoer blokkeert Event Macro's

Bij het invoeren van tekst in een cel worden alle Events geblokkeerd tot de tekst is ingevoerd. Dit is dus zolang het knipperende dunne cursor streepje zichtbaar is. Pas als de cel wordt verlaten kan er weer een Event worden afgevuurd.

- Events kunnen ongewenste "loops" veroorzaken.

Een Event kan weer een ander Event triggeren. Dat kan ongewenste en oncontroleerbare sprongen opleveren. Excel kan zelfs in een ononderbroken lus terecht komen, welke dan alleen nog onderbroken kan worden met de Break toets. Dit probleem kan vooral optreden bij gebruik van het Change Event. Hier kan mogelijk de ene verandering de volgende weer uitlokken, die op haar beurt weer de volgende uitlokt enz. enz. Het is daarom aan te bevelen om het Change Event zo min

mogelijk te gebruiken.

Soms zijn er toch goede redenen om het Change Event wel te gebruiken, of zijn er andere reden waarbij het gewenst is om het triggeren van Event Macro's tegen te gaan. Dat kan dan als volgt:

Application.EnableEvents = False

Door de waarde True mee te geven worden de Events weer aangezet. Dit gebeurt over het algemeen aan het eind van de routine. Bij een onverwachte fout wordt de routine voortijdig verlaten. De Events blijven dan uitgeschakeld, met als gevolg dat er niet meer op gebruikersinvoer wordt gereageerd. Het is daarom aan te bevelen routines waarin de Events worden uitgezet altijd te voorzien van foutafhandeling en daarin de Events weer aan te zetten. Zo kan worden voorkomen dat een fout een complete stop van de werking van het werkboek veroorzaakt. De routine kan bijvoorbeeld als volgt worden opgebouwd:

```
Worksheet                                   Change
'-------------------------------------------------------------------------------
' Procedure : Worksheet_Change
' Purpose   : Verandering afvangen en daar iets mee doen
'-------------------------------------------------------------------------------
'
Private Sub Worksheet_Change(ByVal Target As Range)

   On Error GoTo Worksheet_Change_Error
        Application.EnableEvents = False
            'de benodigde code
        Application.EnableEvents = True
   On Error GoTo 0
   Exit Sub
'Foutafhandeling
'-------------------------------------------------------------------------------
Worksheet_Change_Error:
    Application.EnableEvents = True
        MsgBox "Error " & Err.Number & " (" & Err.Description & ")" _
            & "in procedure Worksheet_Change of VBA Document Blad1"
End Sub
```

Met het Change Event kan er bijvoorbeeld gebruikersinvoer worden opgevangen en gecontroleerd worden op juistheid. Met de volgende routine wordt er gecontroleerd of de invoer een datum is:

```
Worksheet                                   Change
'-------------------------------------------------------------------------------
' Procedure : Worksheet_Change
' Purpose   : Verandering afvangen en daar iets mee doen
'-------------------------------------------------------------------------------
'
Private Sub Worksheet_Change(ByVal Target As Range)

   On Error GoTo Worksheet_Change_Error
        Application.EnableEvents = False
            If IsDate(Target.Value) = False Then
                MsgBox "Geef de datum uitsluitend in het volgende " _
                    & "formaat; dd-mm-jjjj", vbCritical, "Invoer fout"
                Target.ClearContents
            End If
        Application.EnableEvents = True
   On Error GoTo 0
   Exit Sub
'Foutafhandeling
'-------------------------------------------------------------------------------
Worksheet_Change_Error:
    Application.EnableEvents = True
        MsgBox "Error " & Err.Number & " (" & Err.Description & ")" _
            & "in procedure Worksheet_Change of VBA Document Blad1"
End Sub
```

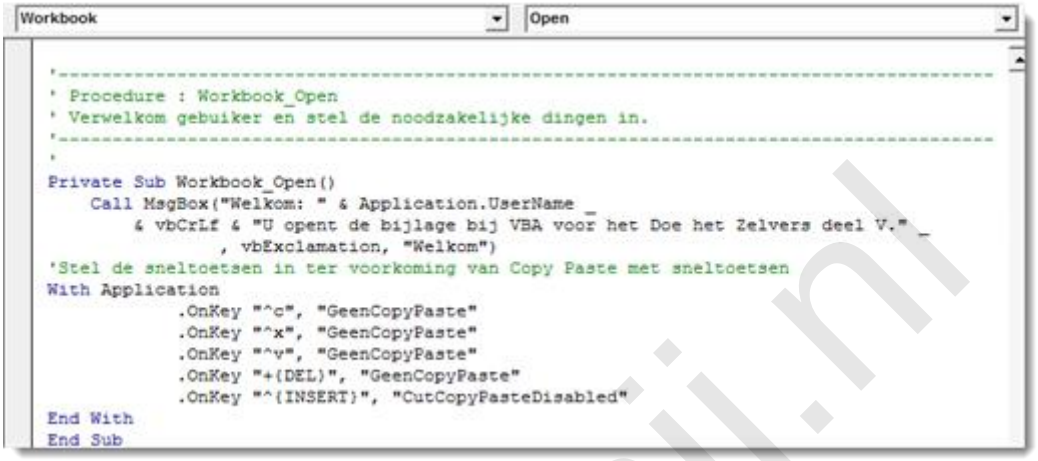
In dit geval is hetzelfde te bereiken met gegevensvalidatie. Dit heeft sterk de voorkeur. In specifieke gevallen kan bovenstaande methode echter wel bruikbare mogelijkheden bieden die anders mogelijk niet zijn te realiseren.

Voorbeelden

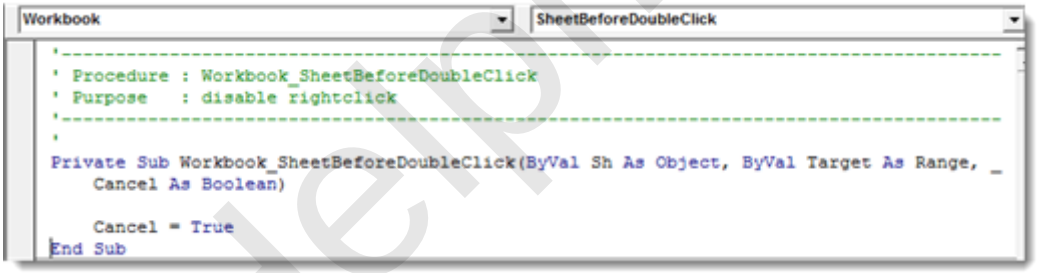
Hieronder volgen nog enkele voorbeelden van het gebruik van Events.

Melding bij het openen van het werkboek:

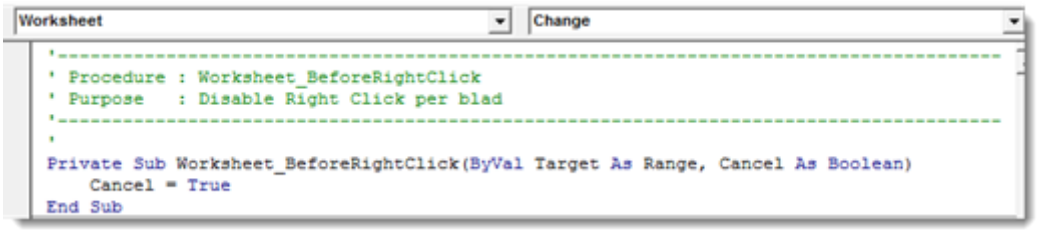
Hiermee wordt bij het openen van het werkboek een messagebox geopend waarin de gebruiker wordt verwelkomd.



Rechtsklik-menu wegvangen:



Deze routine vangt alle rechtsklikken op in het hele werkboek. Indien gewenst kan dat ook per blad, door het Worksheet Event te gebruiken:



Dit is een voorbeeld van een Event welke voor het hele werkboek kan gelden of alleen specifiek per werkblad. Deze keuze is er voor een aantal Events.

Opslaan als voorkomen:

Soms het handig om te voorkomen dat gebruikers het werkboek opslaan onder een andere naam of op een andere locatie. Dat kan worden tegen gegaan door in de het BeforeSave Event na te gaan of er voor “Opslaan Als” is gekozen en dan dit vervolgens als volgt te cancelen:

```
Workbook                                    BeforeSave
'-------------------------------------------------------------------------------
' Procedure : Workbook_BeforeSave
' Purpose   : Opslaan als wordt gecanceld
'-------------------------------------------------------------------------------
'
Private Sub Workbook_BeforeSave(ByVal SaveAsUI As Boolean, Cancel As Boolean)
If SaveAsUI = True Then
    Call MsgBox("Dit werkboek mag niet van naam veranderen " _
                & vbCrLf & "of op een andere locatie worden opgeslagen." _
                , vbExclamation, "Opslaan als")
    Cancel = True
End Sub
```

Met sneltoetsen:

Het indrukken van sneltoetsen is ook een gebeurtenis die kan worden afgevangen. De druk op een toetscombinatie start dan een gespecificeerde macro. Hiervoor is de *OnKey* methode. Zie de VBA help voor de mogelijkheden met de diverse toetscombinaties.

Hieronder volgt een voorbeeld waarmee kopiëren en plakken kan worden tegengegaan. Soms is dat noodzakelijk om bijvoorbeeld te voorkomen dat de opmaak verloren gaat door een kopieer actie.

```
(Algemeen)                                  GeenCopyPaste
'-------------------------------------------------------------------------------
' Procedure : GeenCopyPaste
' Purpose   : Laat de gebuiker weten dat CopyPaste niet kan.
'-------------------------------------------------------------------------------
'
Sub GeenCopyPaste()
    MsgBox "Ter bescherming van de opmaak is kopiëren en plakken niet mogelijk. ", _
        vbExclamation, "Geen Copy Paste mogelijk"
End Sub
```

Bij het openen van het werkboek dient hiervoor de OnKey ingesteld te worden op de volgende manier:

```
Workbook                                    Open
'-------------------------------------------------------------------------------
' Procedure : Workbook_Open
' Verwelkom gebuiker
'-------------------------------------------------------------------------------
'
Private Sub Workbook_Open()
    Call MsgBox("Welkom: " & Application.UserName _
        & vbCrLf & "U opent de bijlage bij VBA voor het Doe het Zelvers deel V." _
                , vbExclamation, "Welkom")
'Stel de sneltoetsen in ter voorkoming van Copy Paste met sneltoetsen
With Application
            .OnKey "^c", "GeenCopyPaste"
            .OnKey "^x", "GeenCopyPaste"
            .OnKey "^v", "GeenCopyPaste"
            .OnKey "+{DEL}", "GeenCopyPaste"
            .OnKey "^{INSERT}", "CutCopyPasteDisabled"
End Sub
```

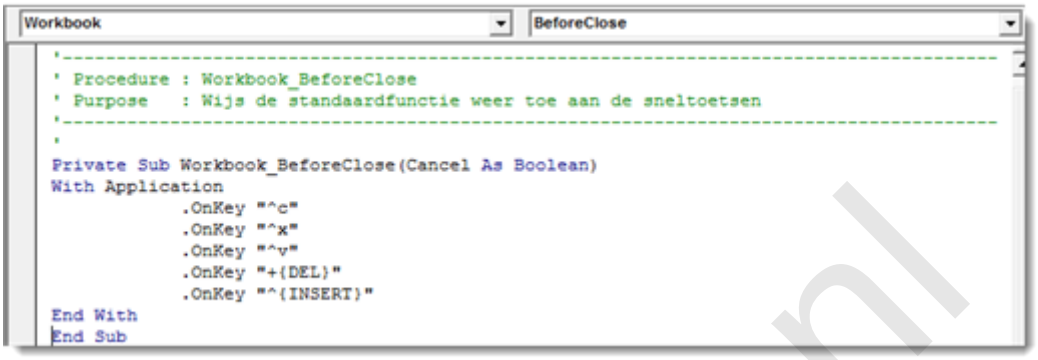
Copy Paste is natuurlijk op verschillende manieren mogelijk. De volgende procedure zorgt ervoor dat ook deze manieren niet te gebruiken zijn:

```
Workbook                                    BeforeSave
'-------------------------------------------------------------------------------
' Procedure : Workbook_SheetSelectionChange
' Purpose   : Zet kopieëren uit als er een cel wordt geselcteerd.
'-------------------------------------------------------------------------------
'
Private Sub Workbook_SheetSelectionChange(ByVal Sh As Object, ByVal Target As Range)
    With Application
        If .CutCopyMode <> False Then .CutCopyMode = False
    End With
End Sub
```

Deze methode is eventueel te combineren met (Public) variabelen die het mogelijk maken om kopiëren op de gewenste momenten wél toe te staan. Dit wordt aan de eigen fantasie overgelaten.

Werkboek sluiten:

Het is altijd netjes om Excel zo achter te laten dat er niets is veranderd voor de gebruiker als hij ons werkboek heeft afgesloten. Dit is nu extra belangrijk, omdat de OnKey Methode op bepaalde momenten de routine ook kan starten als het werkboek al gesloten is. Dat opent dan weer ons werkboek. Dit tot grote (en vast niet blijde) verbazing van de gebruiker die iets probeert te kopiëren in een heel ander werkboek. Gebruik hiervoor het BeforeClose Event. Reset hierin de sneltoetsen als volgt:



Op tijd:

Een enigszins vergelijkbare methode is *OnTime*. Hiermee wordt op de ingestelde tijd een gespecificeerde procedure gestart. Daarmee kan bijvoorbeeld het werkboek ieder kwartier automatisch opgeslagen worden.

Ook aan deze methode zitten een paar mitsen en maren.

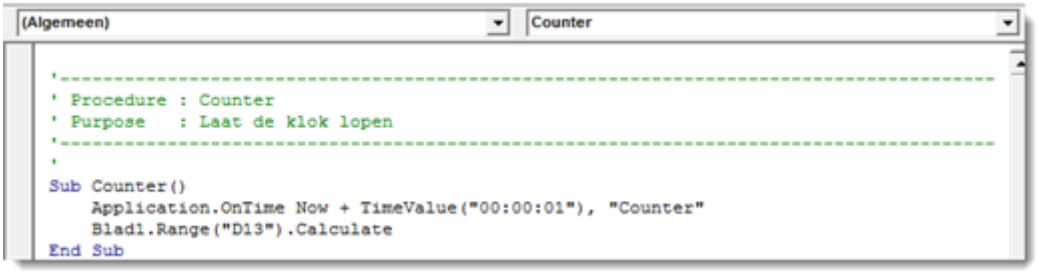
- Tijdens tekstinvoer in een cel wordt er niet naar de gespecificeerde procedure gesprongen. Loopt er bijvoorbeeld een timer mee op basis van OnTime dan stopt deze. Dit wordt dan wel ingehaald wanneer de cel wordt vrijgegeven.
- Tijdens uitvoeren van een andere macro kan de OnTime getriggerd worden, wat tot ongewenste interactie kan leiden.

Deze methode is hierdoor niet geschikt voor echt tijd kritische toepassingen. Ook al omdat de minimum instelbare tijd 1 hele seconde is. Met deze beperkingen in het achterhoofd zijn er toch wel bruikbare toepassingen mogelijk. De tijd instellen is erg gemakkelijk met de Functie TimeValue. Bijvoorbeeld 3 uur 's middags: *TimeValue("15:00:00").* Wanneer je om die tijd de procedure "KoffieTijd" wil starten gebruik je het volgende:

Application.OnTime TimeValue("15:00:00"), "KoffieTijd"

Voor een timer is natuurlijk nodig dat er iedere seconde iets gebeurt. Dat is te bereiken met een procedure waar steeds een seconde na "nu" naar toe wordt gesprongen door middel van een loop:

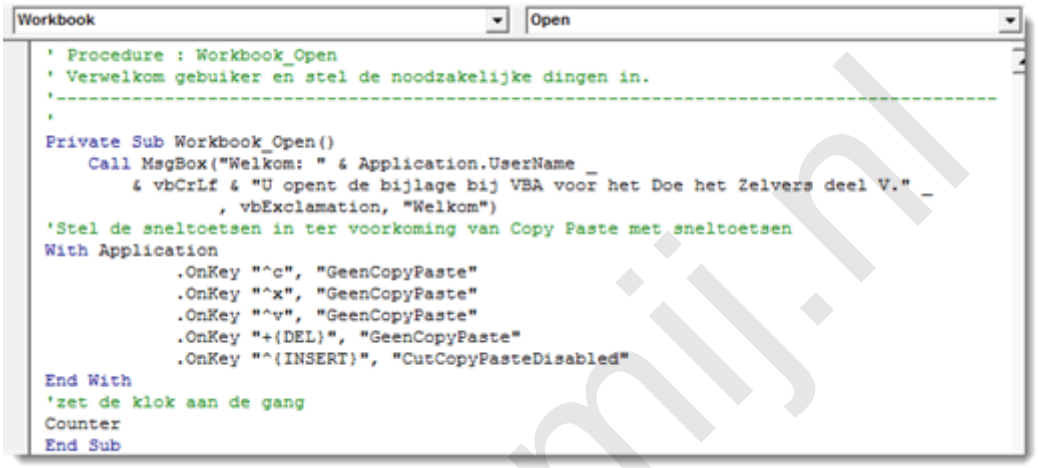
Heel simpel dus. Té simpel, want er gebeurt niets. Behalve dat VBA er iedere seconde naar toe springt. Een klok is met dit principe wel heel gemakkelijk te creëren. Namelijk door op het werkblad NU() als functie te zetten. Met de juiste cel-opmaak krijg je dan een klok. Deze gaat lopen door de betreffende cel elke seconde te laten herberekenen.



```
(Algemeen)                                                    Counter

'---------------------------------------------------------------------------------------
' Procedure : Counter
' Purpose   : Laat de klok lopen
'---------------------------------------------------------------------------------------
'
Sub Counter()
    Application.OnTime Now + TimeValue("00:00:01"), "Counter"
    Blad1.Range("D13").Calculate
End Sub
```

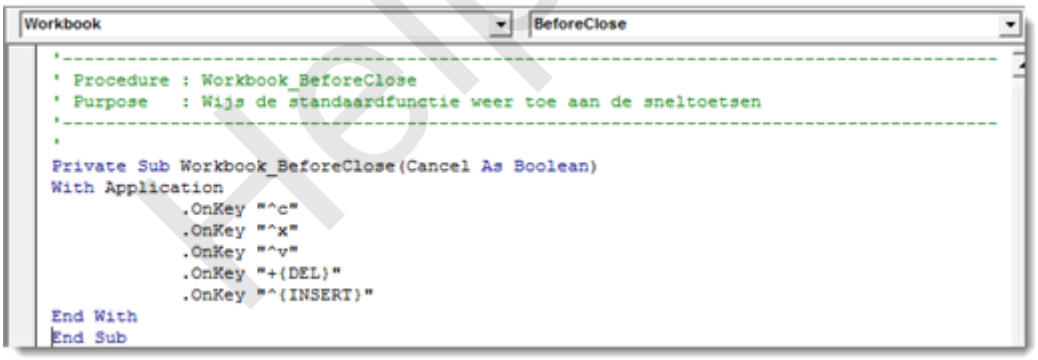
Een regeltje extra is daarvoor voldoende. De procedure moet wel worden gestart bij het openen van het werkboek:



```
Workbook                                                      Open

' Procedure : Workbook_Open
' Verwelkom gebuiker en stel de noodzakelijke dingen in.
'---------------------------------------------------------------------------------------
'
Private Sub Workbook_Open()
    Call MsgBox("Welkom: " & Application.UserName _
        & vbCrLf & "U opent de bijlage bij VBA voor het Doe het Zelvers deel V." _
                , vbExclamation, "Welkom")
'Stel de sneltoetsen in ter voorkoming van Copy Paste met sneltoetsen
With Application
            .OnKey "^c", "GeenCopyPaste"
            .OnKey "^x", "GeenCopyPaste"
            .OnKey "^v", "GeenCopyPaste"
            .OnKey "+{DEL}", "GeenCopyPaste"
            .OnKey "^{INSERT}", "CutCopyPasteDisabled"
End With
'zet de klok aan de gang
Counter
End Sub
```

De OnTIme Methode moet altijd worden afgesloten wanneer er geen gebruik meer van wordt gemaakt. Dit moet ook als het werkboek gesloten wordt en kan dan als volgt:



```
Workbook                                                      BeforeClose

'---------------------------------------------------------------------------------------
' Procedure : Workbook_BeforeClose
' Purpose   : Wijs de standaardfunctie weer toe aan de sneltoetsen
'---------------------------------------------------------------------------------------
'
Private Sub Workbook_BeforeClose(Cancel As Boolean)
With Application
            .OnKey "^c"
            .OnKey "^x"
            .OnKey "^v"
            .OnKey "+{DEL}"
            .OnKey "^{INSERT}"
End With
End Sub
```

Met het afsluiten van het werkboek eindig ik ook deze aflevering.

Samenvattend:

In deze aflevering zijn de Event Macro's besproken met verschillende voorbeelden. Ook werd de OnKey methode uitgelegd en de OnTime methode geïntroduceerd aan de hand van een simpele klok. Dit wordt gedemonstreerd de bijlage, die ***hier*** is te downloaden (Zie de eerste aflevering voor het in gebruik nemen van een werkboek met macro's. Deze kan eventueel worden gedownload in de handleidingsectie).

Volgende aflevering

De volgende keer wordt de OnTIme methode verder besproken met een paar mogelijkheden om deze in te zetten als een timer. Ook wordt een methode gegeven om met behulp van een klein beetje VBA en voorwaardelijke opmaak de geselecteerde rij en kolom zichtbaar te maken. Verder wordt de conversie van werkboeken van oudere (2003 en eerder) Excel versies naar de nieuwere Excelversies besproken. Hier treden nogal eens problemen bij op die ook behandeld zullen worden.