Werkzitting Visual Basic.NET.

1 Inleiding

Visual Basic.NET laat je toe snel Windows-programma's te schrijven met een grafische interface. Bij het programmeren met een grafische interface gebruikt men geen sequentieel programma (cfr. Pascal) maar werkt men event-driven: gestuurd door gebeurtenissen. Deze gebeurtenissen (we zullen ze in het vervolg events noemen) kunnen veroorzaakt worden door de gebruiker (bv. door een druk op een knop), door een tijdklok die regelmatig een gebeurtenis oproept, of door een programmaonderdeel. Aan elk afzonderlijk event kan vervolgens een stuk code worden gehangen dat beschrijft welke acties er moeten gebeuren wanneer dat bepaald event optreedt.

De bedoeling van deze werksessie is door middel van een aantal geleide oefeningen te experimenteren met deze krachtige omgeving. Je zult zien dat je op een intuïtieve en structurele manier snel kleine programma's kunt opbouwen.

Alhoewel Visual Basic.NET tevens gebruikt wordt om complexe software te ontwikkelen, zal deze oefensessie zich hoofdzakelijk beperken tot in de cursus behandelde concepten.

Vooraf:

- Telkens als je een volgende oefening begint, lees dan eerst de opgave in zijn geheel zodat je een duidelijk beeld hebt van wat eigenlijk gevraagd is.

- Maak zoveel mogelijk gebruik van de Help-functie van Visual Basic.NET, als je een probleem hebt. Als je iets wil weten over een bepaalde procedure, plaats dan de tekstcursor bovenop het corresponderende woord en druk op functietoets F1. De helpfunctie biedt dan de overeenstemmende informatie. Het is mogelijk dat in een pcklas niet altijd lukt. Dezelfde informatie vind je ook terug via
  `http://msdn.microsoft.com/en-us/library/d11h6832(VS.90).aspx`
  maar deze is moeilijker doorzoekbaar.

- Als het aantal elementen dat je aan een formulier toevoegt groot is, kies je best een zinvolle naam voor deze elementen. Best volg je ook een aantal stijlregels i.v.m. naamgeving. Elke naam wordt voorafgegaan door een bepaalde prefix, afhankelijk van het type object. Zo geef je bv. aan een knop (Engels: Button) een naam met als prefix *btn*, aan

een label een naam met prefix *lbl*, aan een horizontale schuifbalk een naam met prefix *hsb*, enz..

- Start Visual Basic.NET nu op. Dit vind je in het start-menu onder

  `"Start > Alle programma's > Microsoft Visual Studio 2012 > Visual Studio 2012".`

Bij de eerste start vraagt Visual Studio welk profiel hij moet gebruiken. **Kies hier voor het C#-profiel!** Anders ga je problemen hebben met het kiezen waar Visual Studio het project aanmaakt (en standaard kiest hij een foute directory).

Wanneer Visual Studio opgestart is klik je op `Tools > Options...`. Aan de linkerkant van het scherm klik je onder de sectie `Projects and Solutions` op `General`. Vul hier voor de opties `Projects location`, `User project templates location` en `User item templates location` de folder `C:\Workdir\Oefeningen\Visual Basic` in[1]. **Vink tevens ook de optie `Save new projects when created` aan** en klik op OK. Deze opties hoef je niet noodzakelijk te veranderen als je Visual Studio op je eigen computer uitvoert.

Wanneer deze opties ingevuld zijn klik je op 'New Project'. Klik daarna op de 'Other Languages', 'Visual Basic' en kies 'Windows Forms Application'. Daarna vul je de locatie in waar je het project wil opslaan. **Zorg ervoor dat het project opgeslagen wordt onder 'c:\WorkDir'.**[2] Anders zal je bij de uitvoering van je project een fout krijgen!

2 Inleidende oefening: Hallo Wereld

Open een nieuw project en bewaar het onder de naam Hello. De bedoeling van deze oefening is om de boodschap "Hallo Wereld!" af te drukken op het scherm door gebruik te maken van een label-object. De bedoeling is wel dat deze boodschap in verschillende talen kan worden afgedrukt.

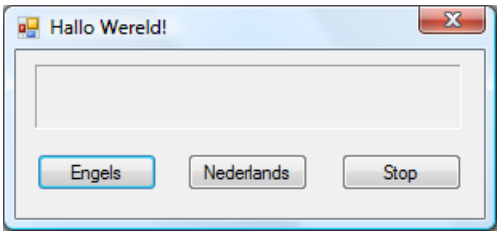
- Elke taal komt overeen met een afzonderlijke knop. Een druk op een bepaalde taalknop zorgt ervoor dat de boodschap "Hallo Wereld!" in de corresponderende taal wordt afgedrukt.

[1] Als deze nog niet bestaat, maak deze dan aan. In feite mag je elke folder gebruiken om je projecten in op te slaan, zolang het maar in de map `C:\Workdir` is. Indien je het niet onder deze map zet, zal je problemen krijgen bij het uitvoeren van je programma.

[2] Dit is niet noodzakelijk als je Visual Studio op je eigen computer uitvoert.

- Voorzie ook een “Stop”-knop.
- Zorg er tenslotte voor dat de grootte van het venster dat bij uitvoer op het scherm getoond zal worden niet aangepast kan worden.

Bij uitvoering moet er een venster verschijnen dat er ongeveer als figuur 1 kan uitzien.



Figuur 1: Initieel venster Hello World.

Als de gebruiker dan bijvoorbeeld op de knop “Engels” drukt, verschijnt de uitvoer weergegeven in figuur 2.

Figuur 2: Resultaat na een druk op knop Engels.

Extra: Zorg er nu ook voor dat dit gebeurt wanneer je je muis gewoon over de knoppen beweegt. Kijk hiervoor naar de `Handles`-instructie bij de procedure-definitie waarin je zojuist je acties specifieërde, en naar de `MouseEnter`-gebeurtenis.

3 Goochelen met fonts

In deze oefening is het de bedoeling om een programma te maken dat de gebruiker toelaat om een font in verschillende groottes en met verschillende effecten te bekijken. Daarvoor heeft hij een tekstvenster dat een voorbeeldtekst toont, enkele radiobuttons om een effect te selecteren en een lijst waaruit hij

de lettergrootte kan kiezen. Een voorbeeld van het programma vind je in figuur 3.

Begin nu met een nieuw project en voer dit formulier in. Let daarbij op de volgende zaken:

- Het is niet de bedoeling dat de gebruiker iets kan typen in het tekstvenster: daarvoor dient immers de knop "Geef tekst in". Zorg er daarom voor dat je de eigenschap `ReadOnly` van het tekstvenster op `True` zet.
- De beginwaarde van de lijst is 10 punten[3].
- Gebruik hiervoor het lettertype *Arial*.
- Zorg ervoor dat de grootte van het venster niet aangepast kan worden tijdens de uitvoering (bekijk hiervoor de *FormBorderStyle* property).

Figuur 3: Het fonts programma.

Wanneer je dit dialoogvenster hebt ingegeven, is het tijd om de code erbij te voegen. Je zult zien dat je voor elk object van het formulier slechts enkele regels nodig hebt. Dit kan dus snel gebeuren.

- Het lettertype van een TextBox kan je veranderen door een nieuw lettertype aan te maken en het toe te kennen aan de TextBox, bvb.

  ```
  txtTekst.Font = New System.Drawing.Font("Arial", grootte).
  ```

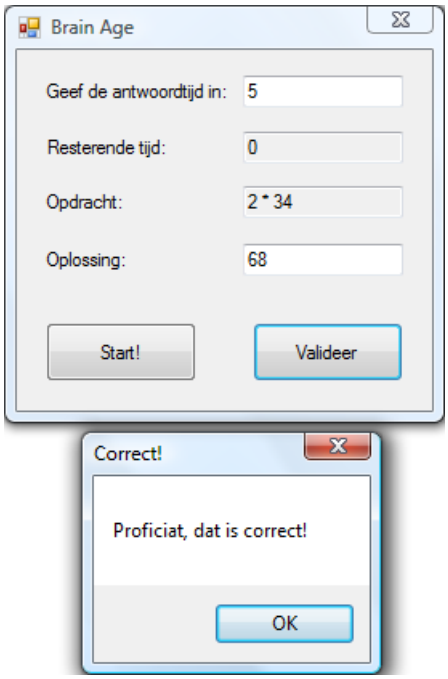
[3]De beginwaarde van de lijst moet je instellen in het `Load`-event van je formulier. Plaats hierin code om de eigenschap `SelectedIndex` van de lijst de juiste waarde te geven.

- De knop “Geef tekst in”: wanneer op deze knop wordt gedrukt, moet er een `InputBox` verschijnen, met daarin de vorige waarde van het tekstvak. Een beschrijving van deze functie kan je vinden in de help.
- De knop “Einde”: wanneer hierop wordt gedrukt, moet het programma eerst een bevestiging komen vragen. Doe dit door gebruik te maken van de functie `MsgBox`. Een beschrijving hiervan is te vinden in de help.

4 Werken met tijd

Je kent misschien het Nintendo DS spelletje Brain Age. In dit spel speel je spelletjes die volgens de theorieën van een zekere neuroloog Dr. Kawashima goed zouden zijn voor de hersenen.

In deze oefening maken we een dergelijk spelletje, zie figuur 4.



Figuur 4: Brain Age

De gebruiker kiest zelf een interval waarbinnen hij een rekensom moet oplossen. Na het drukken op **Start** verschijnt de opgave en telt de teller af. Indien de teller afloopt, wordt de juiste oplossing getoond. Indien de gebruiker het antwoord tijdig kan uitrekenen, tikt hij het resultaat in bij

Oplossing:. Daarna klikt hij op **Valideer!** waarna gekeken wordt of dit een goede oplossing is. Zorg dat:

- de gebruiker het vakje 'resterende tijd' en 'opgave' niet kan veranderen
- de resterende tijd iedere seconde verlaagd wordt

We zetten je even op weg.

- een willekeurig getal genereren tussen 0 en x kan je met:

```
Dim R As New Random
operator = R.Next(0, x)
```

  waarbij operator een Integer is.

- Voor het aftellen voeg je een Timer object toe (naar je formulier slepen vanuit de ToolBox) en reageer je op het Tick event.
- Vooraleer je de timer kan gebruiken moet je deze initialiseren. Het initialiseren en starten van de timer Timer1 gaat met `Timer1.Interval = 1000` en `Timer1.Enabled = True`. Het starten van de timer doe je met `Timer1.Start()`.

Succes !

Yolande Berbers
Jasper Bogaerts
Thomas Winant
Job Noorman
Raoul Strackx

Extra: Werken met dynamische Controls*

In de laatste oefening gaan we werken met arrays van objecten. We gaan een geheugenspel maken waarbij de speler - gegeven een matrix van knoppen - in zo weinig mogelijk gokken moet raden achter welke knop een lachend gezichtje schuilt (zie Figuur 5).

Figuur 5: Probeer in zo weinig mogelijk gokken het lachend gezicht te vinden

We willen dit spelen op een matrix van 6x6 rijen. Omdat het manueel tekenen van 36 knoppen een nogal weinig belonende taak is, gaan we ze dynamisch aanmaken. Neem hiervoor als leidraad het volgende stukje code:

```
'Matrix van 6 breed (0 tot en met 5) en 6 hoog (0 tot en met 5)
Dim rooster(5,5) As Button
For i As Integer = rooster.GetLowerBound(0) To rooster.GetUpperBound(0)
    For j As Integer = rooster.GetLowerBound(1) To rooster.GetUpperBound(1)
        rooster(i, j) = New Button
        rooster(i, j).Size = New Size(30, 30)
        rooster(i, j).TextAlign = ContentAlignment.MiddleCenter
        rooster(i, j).Name = "button_" & i & "_" & j
        'x,y locatie in het venster
        rooster(i, j).Location = New Point(XMIN + (30 * i), YMIN + (30 * j))
        'text op de tegel
        rooster(i, j).Text = ""
        ' teken de knop op het formulier
        Me.Controls.Add(rooster(i, j))
    Next
Next
```

Dit stukje code zal een raster van 36 knoppen maken. Daarenboven hebben we voor het verwezenlijken van deze oefening nog enkele bouwblokken nodig:

- We willen natuurlijk dat er iets gebeurt als er op de knoppen wordt geklikt. Hiervoor moeten we een zogenaamde *handler* dynamisch aan de *Click*-event binden. We kunnen dit doen als volgt:
  `AddHandler rooster(i, j).Click, AddressOf ToonInhoud`
  Deze instructie zal ook bij de initializatie moeten worden geplaatst en zal dus zorgen dat de procedure `ToonInhoud` wordt uitgevoerd telkens er op de knop met index (i, j) wordt geklikt. De hoofding van deze methode kan er als volgt uitzien:
  `Sub ToonInhoud(sender As Object, e As MouseEventArgs)`
  Hierbij is `sender` het object waarop de actie wordt uitgevoerd. Binnen deze procedure moeten we dit nog naar een knop converteren om het op die manier aan te kunnen spreken (en dus ook aan alle eigenschappen ervan te kunnen). Zie hiervoor de andere bouwblokken.

- De `Tag`-eigenschap kan je gebruiken om informatie in een knop op te slaan die niet zichtbaar is voor de gebruiker. Gebruik deze om te bewaren of er achter de knop een lachend gezichtje schuilt of niet.

- We willen uiteraard dat het lachend gezichtje achter een willekeurige knop wordt geplaatst. Gebruik hiervoor de volgende code:

  ```
  Dim random As Random = New Random
  Dim x As Integer = random.Next(0, rooster.GetUpperBound(0))
  Dim y As Integer = random.Next(0, rooster.GetUpperBound(1))
  rooster(x, y).Tag = ":-)"
  ```

  Deze code zal op een willekeurige knop de tag ':-)' plaatsen.

- Met behulp van de functie `CType` kan je een object in een andere typering teruggeven. Bijvoorbeeld:
  `Dim knop As Button = CType(object, Button)`
  Dit stukje code zal een variabele *knop* aanmaken die de vorm van *object* aanneemt, maar in de typering van een `Button`. Met andere woorden zal je op knop alle eigenschappen van een `Button` kunnen oproepen, terwijl je dit bij *object* niet kon omdat Visual Basic .NET deze behandelde als een variabele van het type `Object`.

- Met behulp van het `Sleep`-commando kan je de besturing van het programma tijdelijk pauzeren. Bijvoorbeeld:

```
Application.DoEvents()
System.Threading.Thread.Sleep(500)
```

Zal het programma voor een halve seconde blokkeren om erna verder te gaan met de eerstvolgende instructie. Voor we dit doen, zal de `DoEvents`-procedure alles op het formulier hertekenen om de laatste versie te tonen.

Nu je met deze bouwblokken hebt kennisgemaakt, kunnen we de opdracht verder toelichten. Wat we willen zien is het volgende:

- Wanneer een formulier opstart, willen we dat er een matrix van knoppen wordt getekend. Ook (maar niet zichtbaar voor de speler) wordt er één knop uitgekozen achter dewelke een lachend gezichtje zit.
- Elke keer als er op een knop wordt gedrukt zal er (a) een teller met één waarde omhoog gaan. Deze geeft aan hoeveel keer er is geraden, en (b) de knop worden omgedraaid. Indien het geen lachend gezichtje is, moet deze knop de `Text` ':-(' krijgen. Anders wordt het lachend gezichtje achter de knop getoond en komt er een `MessageBox` die aangeeft dat je het lachende gezichtje gevonden hebt.
- Indien er op een knop wordt geklikt waar geen lachend gezichtje op staat, zal dit gedurende één seconde worden getoond om daarna terug van de knop te verdwijnen.
- Alternatief kan je ervoor kiezen om nadat het lachende gezichtje is gevonden, alle knoppen te wissen, een nieuwe gelukkige knop aan te duiden en het spel opnieuw te beginnen.

Veel succes!