

## BAB II

### PENGENALAN FORM

Form merupakan obyek dalam Visual Basic 6.0 yang pertama kali terlihat dan dikenal oleh pengguna. Visual Basic 6.0 yang merupakan bahasa pemrograman visual menjadikan form sebagai suatu media antarmuka bagi pengguna. Program dan pengguna dapat saling berkomunikasi secara interaktif melalui media form ini. Visual Basic juga dikenal sebagai bahasa pemrograman berbasis obyek, dimana para pengembang / pemrogram bekerja dengan cara membaca dan memodifikasi properti dari obyek, memanggil metode, dan bereaksi terhadap *event* (kejadian) yang dibangkitkan. Nilai properti dari obyek tersebut dapat diisikan oleh pengguna secara langsung pada saat desain (*design-time*) tanpa harus menuliskan kode tertentu.

#### 1. PROPERTI

Properti menunjukkan karakteristik dari form atau obyek yang bersangkutan.

a. Properti *Left*, *Top*, *Width*, dan *Height*.

Properti ini berhubungan dengan pengaturan posisi dan ukuran dari obyek. Isi atau nilai numerik dalam properti bisa dimasukkan ketika saat pendesainan (*design-time*) atau ketika program sedang jalan / dieksekusi (*run-time*).

```
` Menggandakan lebar form, dan memindahkan formnya ke pojok kiri atas
Form1.Width = Form1.Width * 2
Form1.Left = 0
Form1.Top = 0
```

Untuk obyek yang statusnya *invisible* (tak terlihat) ketika program dieksekusi, maka keempat properti di atas tidak bisa diterapkan. Namun ada juga obyek yang tidak mendukung dari properti *Height*, misalnya: *ComboBox*. Kontrol ini properti *Height* bersifat *read-only* (tidak bisa diubah) nilai propertinya. Biasanya tinggi dari *ComboBox* tergantung pada atribut dari font yang diberikan dalam *ComboBox* itu sendiri.

b. Properti *ForeColor* dan *BackColor*

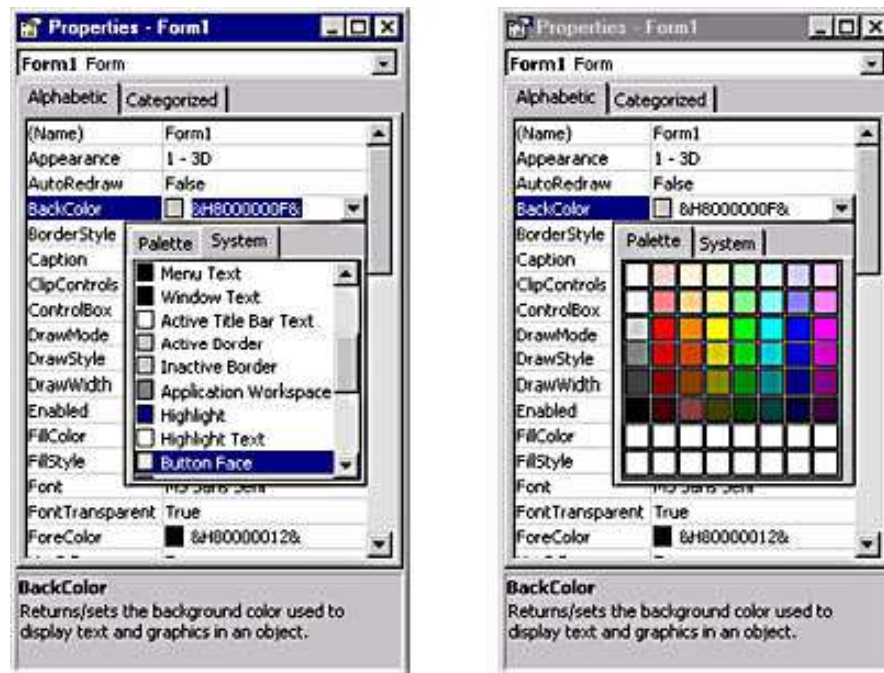
Properti ini berhubungan dengan pengaturan warna dari suatu teks atau latar belakang dari suatu obyek yang *visible* (tampak) atau form. Beberapa kontrol, untuk penggunaan properti *BackColor* bergantung pada pengaturan dari properti lain. Misalnya, kontrol *ScrollBar*, warna dari obyek ini ditentukan oleh warna *theme* dari sistem windows anda. Untuk kontrol *Label*, anda dapat memberi warna latar belakang (*BackColor*) padanya dengan syarat properti *BackStyle* bernilai *1-Opaque*. Karena jika propertinya bernilai *0-Transparent*, maka anda tidak bisa memberikan warna latar padanya.

Untuk menggunakan warna ini anda dapat memberikan warna melalui warna *Windows Standard (System)* atau warna *Custom (Palette)*, seperti terlihat pada gambar 2.1.

Anda dapat memilih dari beberapa cara untuk memasukkan warna dalam kode. Visual Basic menyediakan satu set konstan berupa symbol yang mewakili seluruh warna-warna yang muncul dalam tab *System* dalam jendela properti saat pendesainan.

'Membuat Label1 muncul dalam status yang dipilih.  
 Label1.ForeColor = vbHighlightText  
 Label1.BackColor = vbHighlight

Semua konstan simbolis ditunjukkan di Tabel 2.1, namun anda juga dapat menjelajah mereka di jendela *Object Browser*.



**Gambar 2.1.** Dua cara yang berbeda dalam mengatur nilai *ForeColor* dan *BackColor* pada saat pendesainan (*design-time*).

**Tabel 2.1.** Konstan dalam Visual Basic untuk warna sistem.

Konstan	Nilai Heksa	Deskripsi
vb3DDKShadow	&H80000015	Bayangan paling gelap
vb3Dface	&H8000000F	Warna bayangan gelap untuk penampilan elemen 3-D
vb3Dhighlight	&H80000014	Warna <i>Highlight</i> untuk penampilan elemen 3-D
vb3Dlight	&H80000016	Warna 3-D paling terang kedua setelah vb3Dhighlight
vb3Dshadow	&H80000010	Warna otomatis untuk bayangan jendela ( <i>window</i> )
vbActiveBorder	&H8000000A	Warna pembatas ( <i>border</i> ) untuk jendela yang aktif
vbActiveTitleBar	&H80000002	Warna judul ( <i>caption</i> ) untuk jendela yang aktif
vbActiveTitleBarText	&H80000009	Warna teks di caption yang aktif, ukuran kotak ( <i>box</i> ), kotak

		penunjuk/panah ( <i>arrow</i> ) dari <i>scroll bar</i>
vbApplicationWorkspace	&H8000000C	Warna latar belakang dari aplikasi <i>multiple-document interface</i> (MDI)
vbButtonFace	&H8000000F	Corak pembuatan bayangan pada tombol ( <i>CommandButton</i> ).
vbButtonShadow	&H80000010	Bayangan dari tepi / pinggir tombol ( <i>CommandButton</i> )
vbButtonText	&H80000012	Warna teks ketika tombol ditekan
vbDesktop	&H80000001	Warna <i>Desktop</i>
vbGrayText	&H80000011	Warna keabuan ( <i>disabled</i> )
vbHighlight	&H8000000D	Warna latar belakang dari item yang dipilih dalam kontrol
vbHighlightText	&H8000000E	Warna teks dari item yang dipilih dalam kontrol
vbInactiveBorder	&H8000000B	Warna pembatas ( <i>border</i> ) dari jendela yang tidak aktif
vbInactiveCaptionText	&H80000013	Warna teks dari jendela yang tidak aktif
vbInactiveTitleBar	&H80000003	Warna judul ( <i>caption</i> ) dari jendela yang tidak aktif
vbInactiveTitleBarText	&H80000013	Warna teks di judul jendela yang tidak aktif, ukuran kotak ( <i>box</i> ), kotak penunjuk/panah ( <i>arrow</i> ) dari <i>scroll bar</i>
vbInfoBackground	&H80000018	Warna latar belakang dari <i>ToolTips</i>
vbInfoText	&H80000017	Warna teks dari <i>ToolTips</i>
vbMenuBar	&H80000004	Warna latar belakang dari menu
vbMenuText	&H80000007	Warna teks dalam menu
vbScrollBars	&H80000000	Warna daerah kelabu dari <i>Scroll bar</i>
vbTitleBarText	&H80000009	Warna teks dalam judul ( <i>caption</i> ) yang aktif, ukuran kotak ( <i>box</i> ), kotak penunjuk/panah ( <i>arrow</i> ) dari <i>scroll bar</i>
vbWindowBackground	&H80000005	Warna latar belakang jendela
vbWindowFrame	&H80000006	Warna bingkai dari jendela
vbWindowText	&H80000008	Warna teks di jendela

Ketika anda memasukkan sebuah warna *custom*, anda dapat menggunakan salah satu dari konstan simbolis yang telah ditentukan Visual Basic untuk beberapa warna yang umum. Misalnya: *vbBlack*, *vbBlue*, *vbCyan*, *vbGreen*, *vbMagenta*, *vbRed*, *vbWhite*, and *vbYellow*.

Anda juga bisa menggunakan numeric desimal atau atau konstan heksadesimal.

```
' Pernyataan berikut ini adalah sama artinya.
Text1.BackColor = vbCyan
Text1.BackColor = 16776960
Text1.BackColor = &HFFFF00
```

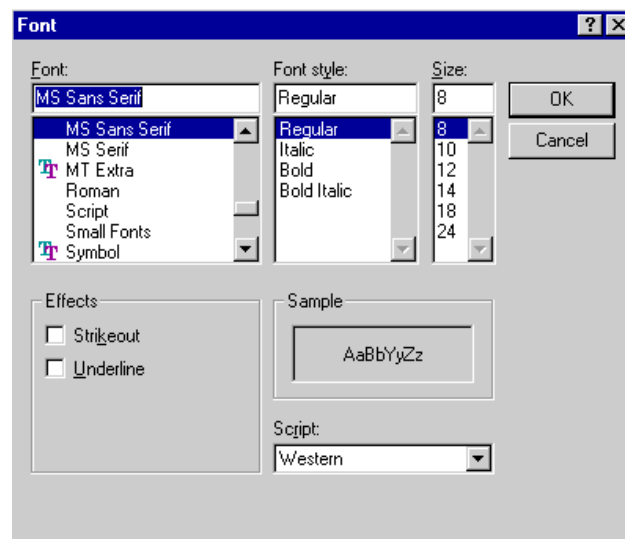
Anda dapat juga menggunakan fungsi RGB untuk membangun sebuah warna yang tersusun atas warna merah, hijau, dan biru (*red, green, blue*).

```
' Pernyataan berikut sama dengan yang di atas.
Text1.BackColor = RGB(0, 255, 255) ' nilai red, green, blue
```

c. Properti *Font*

Properti yang digunakan untuk menampilkan karakteristik dari string (deretan) karakter pada saat pendesainan. Dalam hal ini anda dapat mengatur atribut font dengan menggunakan kotak dialog seperti yang terlihat pada gambar 2.2. Obyek-obyek dari Font meliputi properti *Name, Size, Bold, Italic, Underline, and Strikethrough*.

```
Text1.Font.Name = "Tahoma"
Text1.Font.Size = 12
Text1.Font.Bold = True
Text1.Font.Underline = True
```



**Gambar 2.2.** Pada saat pendesainan, kotak dialog Font memungkinkan anda memodifikasi seluruh atribut font dengan segera dan menampilkan hasilnya

d. Properti *Caption* dan *Text*

Properti *Caption* merupakan string (deretan) dari karakter-karakter yang muncul dalam sebuah kontrol (atau di bagian judul dari form) dan pengguna tidak bisa mengubahnya secara langsung. Sebaliknya, properti *Text* berhubungan dengan isi dari kontrol dan biasanya bisa diubah oleh pengguna. Kontrol yang menggunakan properti *Caption* diantaranya

adalah: *Label, CommandButton, CheckBox, OptionButton, Frame, Data*. Sedangkan kontrol yang menggunakan properti *Text* adalah: *TextBox, ListBox, ComboBox*.

```
' Pernyataan berikut ini adalah sama
Text2.Text = Text1.Text
Text2 = Text1
```

e. Properti *Enabled* dan *Visible*

Dalam kondisi awal (*default*) semua kontrol dan form adalah *visible* (tampak) dan *enabled* ketika *run-time*.

```
' Enable atau disable kontrol Text1 control ketika
' pengguna mengklik kontrol CheckBox bernama Check1
Private Sub Check1_Click()
    Text1.Enabled = (Check1.Value = vbChecked)
End Sub
```

f. Properti *MousePointer* dan *MouseIcon*

Properti ini mempengaruhi bentuk dari kursor mouse ketika melalui (*hover*) suatu kontrol. Windows mengijinkan manajemen kursor mouse yang sangat fleksibel dimana tiap form dan kontrol dapat menampilkan kursor yang berbeda.

- Jika properti *Screen.MousePointer* diset ke suatu nilai selain dari nilai *0-vbDefault*, kursor mouse akan berupa sesuai dengan nilai ini dan tidak ada properti lain yang diperhatikan. Tetapi ketika mouse melewati aplikasi yang berbeda (atau *desktop*), penampilan dari kursor tergantung dari status dari aplikasi tersebut, bukan pada status aplikasi anda.
- Jika *Screen.MousePointer* bernilai 0 dan kursor mouse melewati suatu kontrol, Visual Basic akan memeriksa properti *MousePointer* dari kontrol tersebut; jika nilainya berbeda dari *0-vbDefault*, maka kursor mouse diset ke nilai ini.

Jika anda ingin menampilkan kursor hourglass, dimanapun pengguna menggerakkan mousenya, gunakan kode berikut:

```
' Sebuah rutin panjang
Screen.MousePointer = vbHourglass
...
' Tuliskan kode lain sesuai keperluan anda
...
' jangan lupa untuk mengembalikan pointer ke nilai awal (default)
Screen.MousePointer = vbDefault
```

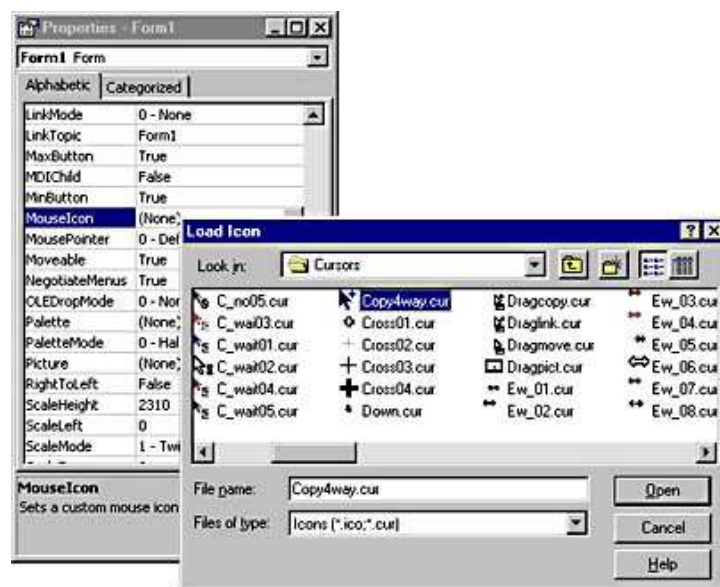
Ini adalah contoh lain:

```
' Menampilkan sebuah kursor crosshair ketika mouse melalui kontrol
' Picture1 dan sebuah hourglass dimanapun di form induk.
Picture1.MousePointer = vbCrosshair
MousePointer = vbHourglass
```

Properti *MouseIcon* digunakan untuk menampilkan kursor mouse yang dapat ditentukan atau dikustomisasi oleh pengguna. Dalam hal ini, anda harus mengatur *MousePointer* ke nilai spesial 99-vbCustom dan memasukkan ikon ke properti *MouseIcon* :

```
' Menampilkan sebuah kursor mouse tanda Stop merah.
' Alamat / jalur dari gambar mungkin berbeda, tergantung dari direktori
' utama dimana anda menginstal Visual Basic
MousePointer = vbCustom
MouseIcon = LoadPicture ("C:\Program Files\Microsoft Visual Studio\
Common\Graphics\Icons\Computer \msgbox01.ico")
```

Untuk memuatkan gambar ikon lebih mudahnya dapat anda masukkan ketika pendesainan daripada saat *run-time*. Yaitu dengan cara memasukkan nilai properti *MouseIcon* pada jendela *Properties*. Dan tentunya properti *MousePointer* diset ke 99-vbCustom.



**Gambar 2.3.** Visual Basic 6.0 juga menawarkan banyak gambar kursor, ikon, dan bitmap di subdirektori \GRAPHICS

#### g. Properti Lain

Properti *Value* adalah umum digunakan dalam beberapa kontrol intrinsik, misalnya kontrol *CheckBox*, *OptionButton*, *CommandButton*, and *scroll bar*. Arti dari properti ini bermacam-macam dari kontrol satu dengan yang lain, namun secara umum nilainya berupa *numeric* atau *Boolean*.

Properti *Index* merupakan kunci dalam membangun *control arrays*, salah satu fitur dari Visual Basic yang membantu anda membuat program lebih berguna. (Untuk lebih jelasnya akan diterangkan pada bab berikutnya).

Kebanyakan kontrol intrinsik mendukung properti *Appearance*. Secara *default*, Visual Basic membuat kontrol dengan aspek tiga-dimensi (3D), kecuali anda mengganti nilainya menjadi *0-Flat*.

Properti *BorderStyle* didukung oleh beberapa kontrol intrinsik, misalnya kontrol *TextBox*, *Label*, *Frame*, *PictureBox*, *Image*, dan *OLE*. Anda dapat

mengeset propertinya ke *0-None* untuk menyembunyikan pembatas (*border*) di sekeliling kontrol dan ke *1-Fixed Single* untuk menampilkan pembatasnya.

## 2. METODE

Sebagaimana banyaknya properti yang kebanyakan dimiliki oleh obyek-obyek dalam Visual Basic, mereka juga memiliki banyak metode yang umum.

### a. Metode *Move*

Jika suatu kontrol mendukung properti *Left*, *Top*, *Width*, dan *Height*, maka dia juga mendukung metode *Move*. Hal ini dapat menggantikan beberapa atau keempat properti tersebut dalam satu operasi tunggal. Contoh berikut ini melakukan perubahan pada 3 properti *Left*, *Top*, dan *Width* :

```
' Menduakalikan lebar form, dan memindahkannya ke pojok kiri atas
' dari layar
' Sintak: Move Left, Top, Width, Height
Form1.Move 0, 0, Form1.Width * 2
```

### b. Metode *Refresh*

Metode ini menyebabkan kontrol menjadi digambar ulang. Secara normal anda tidak perlu memanggil metode ini secara eksplisit karena Visual Basic secara otomatis menyegarkan (*me-refresh*) penampilan dari kontrol-kontrol ketika dia memiliki kesempatan (biasanya saat tidak kode yang sedang dieksekusi dan aplikasi biasanya dalam keadaan *idle* (diam),

```
For n=1000 to 1 step -1
    Label1.Caption = CStr(i)
    Label1.Refresh           ' Memperbaharui label dengan segera
Next
```

### c. Metode *SetFocus*

Metode ini memindahkan fokus input ke kontrol yang telah ditentukan. Anda perlu memanggil metode ini hanya jika anda ingin memodifikasi urutan *Tab Order default* yang secara implisit dibuat saat pendesainan dengan mengatur properti *TabIndex* dari kontrol di form. Kontrol dimana properti *TabIndex*-nya diset 0 berarti di menerima fokus ketika form pertama kali dimuatkan (*di-load*).

Metode *SetFocus* ini bisa diterapkan bila kontrol yang dimaksud dalam keadaan *enable* dan *visible*. Sebab bila tidak, akan menimbulkan *run-time error*. Dan juga jangan lupa untuk mengatur nilai properti *TabStop* ke *true*.

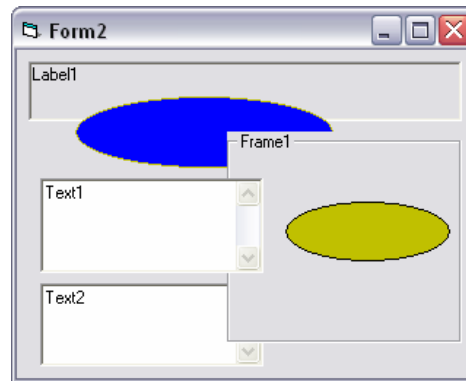
```
' Memindahkan fokus ke Text1
If Text1.Visible and Text1.Enabled then
    Text1.SetFocus
End if
```

### d. Metode *ZOrder*

Metode ini berpengaruh pada penampakan suatu kontrol terhadap kontrol lain. Anda bisa mengeksekusi metode ini tanpa memerlukan argumen apapun untuk memposisikan kontrol berada di depan kontrol lain. Sedangkan untuk memposisikan kontrol di belakang kontrol lain maka *ZOrder* di set 1.

```
' Memindahkan kontrol di belakang kontrol lain dalam form
Text1.ZOrder 1
Text1.ZOrder ` Memindahkan kontrol ke depan
```

Anda juga dapat mengeset *ZOrder* relatif dari suatu kontrol saat pendesainan menggunakan perintah di Sub menu **Order** dari Menu **Format**.



**Gambar 2.4.** ZOrder relative dari kontrol

### 3. KEJADIAN (EVENTS)

Pada saat sebuah aplikasi Windows dijalankan, akan banyak kemungkinan pekerjaan yang dapat dilakukan oleh pengguna program. Misalnya menekan tombol keyboard, memindahkan mouse, mengklik obyek, dan masih banyak lainnya. Pekerjaan-pekerjaan semacam ini disebut dengan kejadian (*Event*).

a. Kejadian *Click* dan *DbClick*

Kejadian *Click* terjadi ketika pengguna mengklik mouse sebelah kiri sebanyak satu kali, sedangkan kejadian *DbClick* terjadi saat pengguna mengklik ganda (dobel) pada mouse sebelah kiri.

```
' Klik sekali pada form untuk merubah Label1 menjadi "Satu"
' Klik dobel / ganda pada form untuk merubah Label1 menjadi "Dua"
Private Sub Form_Click()
    Label1.Caption = "Satu"
End Sub
Private Sub Form_DbClick()
    Label1.Caption = "Dua"
End Sub
```

b. Kejadian *Change*

Merupakan kejadian paling sederhana yang ditawarkan oleh Visual Basic. Kapanpun isi dari suatu kontrol berubah, maka Visual Basic membangkitkan kejadian *Change*.

```
' Mencegah panjang karakter di textbox lebih dari 3 karakter
' Jika lebih dari 3 karakter, maka textbox dikosongkan
Private Sub text1_Change()
    If len(Text1.Text) > 3 then
        Text1.Text = ""
    End If
End Sub
```



- c. Kejadian *GotFocus* dan *LostFocus*  
Kejadian *GotFocus* dibangkitkan saat suatu kontrol menerima fokus input, sedang *LostFocus* akan dibangkitkan saat fokus input pada kontrol tersebut keluar atau berpindah ke kontrol yang lainnya.
- d. Kejadian *KeyPress*, *KeyDown*, *KeyUp*  
Kejadian ini dibangkitkan saat pengguna menekan kunci tombol pada keyboard saat kontrol memiliki fokus input.  
Urutannya:
- *KeyDown* (pengguna menekan tombol / kunci)
  - *KeyPress* (Visual Basic menerjemahkan kunci ke kode numerik ANSI)
  - *KeyUp* (pengguna melepaskan kunci dari keyboard)
- Kunci yang dapat memanggil kejadian *KeyPress* meliputi *Ctrl+X*, *BackSpace*, *Enter*, *Escape*, dan karakter-karakter yang dapat dicetak. Sedangkan kunci yang tidak bisa memanggil kejadian *KeyPress* dan hanya bisa memanggil kejadian *KeyDown* dan *KeyUp* adalah kunci panah, fungsi, *Alt+X*, dan beberapa kunci lainnya.

```
Private Sub Text1_KeyPress (KeyAscii as integer)
    MsgBox "Anda memasukkan kunci : " & Chr$ (KeyAscii)
End Sub
```

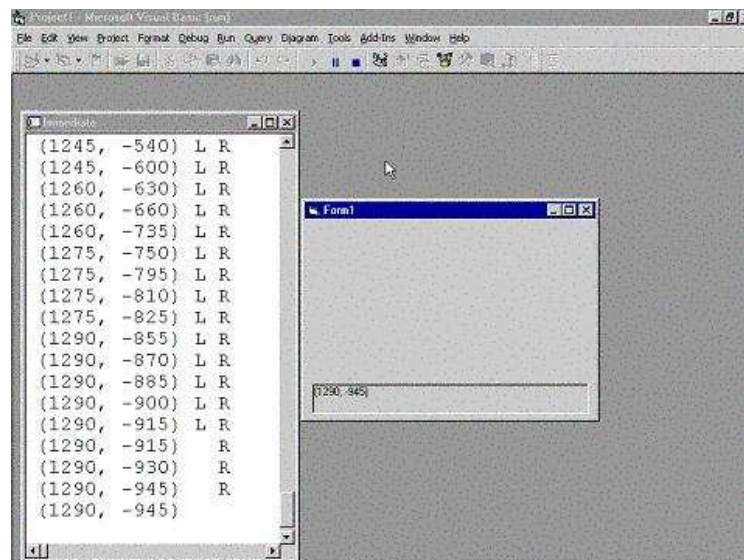
Jika anda memodifikasi parameter *KeyAscii*, maka perubahan yang anda buat akan berpengaruh pada bagaimana program itu menginterpretasikan kunci. Anda dapat juga mencegah munculnya kunci yang diketikkan dengan cara mengeset parameter *KeyAscii* ke 0.

```
Private Sub Text1_KeyPress (KeyAscii as Integer)
    ' Mengubah semua kunci ke huruf besar (UpperCase)
    ' dan menolak karakter kosong
    KeyAscii = Asc ( Ucase$ ( Chr$ ( KeyAscii ) ) )
    If KeyAscii = Asc ( " ") Then KeyAscii = 0
End Sub
```

Untuk kejadian *KeyDown* dan *KeyUp* menerima 2 parameter, yaitu *KeyCode* dan *Shift*.

```
Private Sub Text1_KeyDown (KeyCode as Integer, Shift as Integer)
    ' Jika pengguna menekan Ctrl+F2, maka mengganti isi teks dari
    ' kontrol dengan Date (tanggal hari ini)
    If KeyCode = VbKeyF2 and Shift = VbCtrlMask then
        Text2.Text = Date$
    End If
End Sub
```

- e. Kejadian *MouseDown*, *MouseUp*, *MouseMove*  
Kejadian ini secara berturutan dipanggil ketika mouse diklik, dilepas dan digerakkan melewati suatu kontrol.  
Semuanya menerima parameter-parameter yang sama: status dari tombol mouse, status dari kunci Shift/Ctrl/Alt, dan koordinat x dan y dari kursor mouse. Koordinat tersebut selalu relatif terhadap pojok kiri atas dari kontrol atau form. Pada gambar 2.5 adalah contoh tampilan dari eksekusi kode yang menampilkan status dan posisi dari mouse pada sebuah Label dan membuat *log* (catatan) pada *Immediate window*.



**Gambar 2.5.** Mengawasi status mouse menggunakan kejadian *MouseDown*, *MouseMove*, dan *MouseUp*. Nilai negatif pada y jika kursor berada di luar area form klien.

```
Private Sub Form_MouseDown(Button As Integer, _
    Shift As Integer, X As Single, Y As Single)
    ShowMouseState Button, Shift, X, Y
End Sub

Private Sub Form_MouseMove(Button As Integer, _
    Shift As Integer, X As Single, Y As Single)
    ShowMouseState Button, Shift, X, Y
End Sub

Private Sub Form_MouseUp(Button As Integer, _
    Shift As Integer, X As Single, Y As Single)
    ShowMouseState Button, Shift, X, Y
End Sub

Private Sub ShowMouseState (Button As Integer, _
    Shift As Integer, X As Single, Y As Single)
    Dim descr As String
    descr = Space$(20)
    If Button And vbLeftButton Then Mid$(descr, 1, 1) = "L"
    If Button And vbRightButton Then Mid$(descr, 3, 1) = "R"
    If Button And vbMiddleButton Then Mid$(descr, 2, 1) = "M"
    If Shift And vbShiftMask Then Mid$(descr, 5, 5) = "Shift"
    If Shift And vbCtrlMask Then Mid$(descr, 11, 4) = "Ctrl"
    If Shift And vbAltMask Then Mid$(descr, 16, 3) = "Alt"
    descr = "(" & X & ", " & Y & ") " & descr
    Label1.Caption = descr
    Debug.Print descr
End Sub
```

Hubungan kejadian *MouseDown*, *MouseUp*, dan *MouseMove* dengan kejadian *Click* and *DoubleClick* :

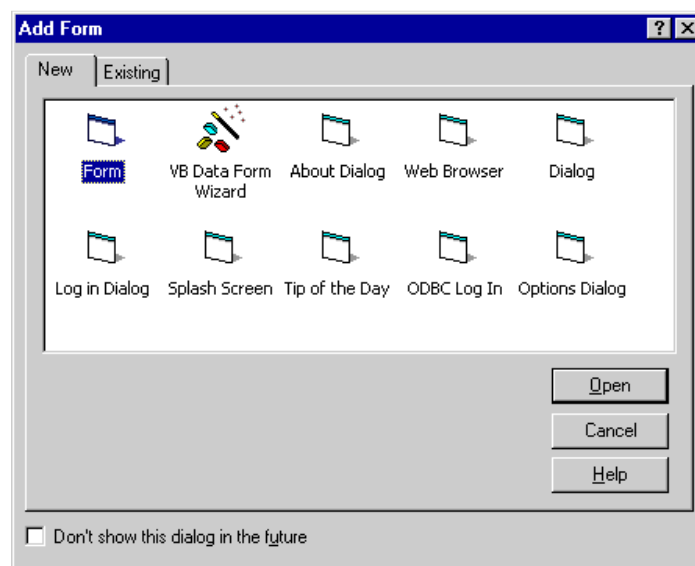
- Kejadian Click terjadi setelah sebuah *MouseDown* ... *MouseUp* secara berturutan dan sebelum kejadian tambahan *MouseMove*.

- Ketika pengguna mengklik ganda pada sebuah kontrol, urutan kejadian lengkapnya adalah sebagai berikut : *MouseDown*, *MouseUp*, *Click*, *MouseMove*, *DbClick*, *MouseUp*, *MouseMove*. Dengan catatan bahwa kejadian *MouseMove* kedua belum dibangkitkan.
- Jika kontrol diklik dan kemudian mouse digerakkan di luar area kliennya, kejadian *Click* tidak akan pernah dipanggil. Bagaimanapun, jika anda mengklik ganda sebuah kontrol dan kemudian anda menggerakkan mouse di luar area klien, urutan kejadian lengkapnya akan terjadi. Tingkah laku ini mencerminkan bagaimana sebuah kontrol bekerja di bawah *Windows* dan seharusnya tidak dianggap sebagai *bug*.

#### 4. OBYEK FORM

Obyek *visible* yang paling penting adalah form. Sebab tanpa adanya form sebagai induk, anda tidak bisa menampilkan obyek lain.

Untuk membuat form baru saat pendesainan (*design-time*) anda bisa menggunakan perintah **Add Form** dari menu **Project** atau dengan mengklik ikon pada Toolbar standar yang mewakili *Add Form*.



**Gambar 2.6.** Template form yang ditawarkan Visual Basic 6.0

##### a. Properti Dasar Form

**BorderStyle** merupakan salah satu properti yang berpengaruh besar pada kelakuan form. Nilai default propertinya adalah *2-Sizeable*, dimana membuat jendela dapat diubah ukurannya oleh pengguna (*resizable*). Untuk membuat form yang *non-resizable*, anda dapat mengeset nilainya ke *1-Fixed Single* atau *3-Fixed Dialog*. Perbedaan di antara keduanya adalah dalam penampilan tombol *Maximize* dan *Minimize*. Untuk membuat form kelihatan mengambang (*float*) seperti *toolbox*, gunakan nilai *4-Fixed ToolWindow* atau *5-Sizeable ToolWindow*. Untuk menghilangkan pembatas (*border*) dan judul (*caption*) gunakan nilai *0-None*.

Untuk menampilkan judul di form adalah dengan mengeset string di properti **Caption**.

Untuk menampilkan/menghilangkan tombol *Minimize/Maximize*, gunakan properti **MinButton** dan **MaxButton**, atur nilainya ke *True* untuk menampilkan atau sebaliknya.

Jika anda ingin agar form muncul langsung dalam keadaan *maximize*, set properti **WindowState** ke *2-Maximize*.

Untuk menampilkan form berada di tengah layer gunakan properti **Startup Position** dan set nilainya ke *2-CenterScreen*. Apabila anda ingin agar form ditampilkan di *taskbar*, set properti **ShowInTaskbar** ke True, jika tidak set ke False.

b. Lingkaran kehidupan Form

1. Kejadian *Initialize*

Kejadian pertama dalam kehidupan suatu form adalah *initialize*. Kejadian ini dibangkitkan segera setelah anda mereferensikan nama formnya di dalam kode anda, bahkan sebelum Visual Basic membuat jendela nyata dan kontrol di permukaannya.

2. Kejadian *Load*

Kejadian ini dipanggil saat form sudah dimuatkan (*di-load*)

3. Kejadian *Resize*

Kejadian ini dipanggil saat ukuran form mengalami perubahan.

4. Kejadian *Activate*

Kejadian ini dipanggil saat form menerima fokus lagi dari pengguna setelah sebelumnya kehilangan fokus.

5. Kejadian *Paint*

Kejadian ini dapat dipanggil jika anda mengeset nilai properti *AutoRedraw* pada form ke True.

Dalam prosedur kejadian *Paint*, anda diharapkan menggambar ulang isi dari form menggunakan metode grafik, seperti : *Print*, *Line*, *Cycle*, *Point*, *Cls*, dsb.

6. Kejadian *Deactivate*

Kejadian ini dipanggil saat form kehilangan fokus, dimana fokus telah pindah ke form atau aplikasi lain.

7. Kejadian *QueryUnload*

Ketika suatu form akan ditutup (*Unload*), maka obyek form menerima kejadian *QueryUnload*. Berikut ini contoh potongan kode untuk mencegah pengguna menutup form aplikasi.

```
Private Sub Form_QueryUnload(Cancel As Integer, _
    UnloadMode As Integer)
    ' Jangan biarkan pengguna menutup aplikasi dari form.
    Select Case UnloadMode
        Case vbFormControlMenu, vbAppTaskManager
            Cancel = True
    End Select
End Sub
```

8. Kejadian *Unload*

Merupakan kejadian terakhir yang dikerjakan saat form ditutup (*di-unload*). Diperiksa setelah kejadian *QueryUnload* diperiksa.

c. Koleksi Kontrol

Salah satu properti khusus dari form adalah koleksi kontrol, dimana berisi semua kontrol yang sedang dimuatkan (*di-load*) ke form itu sendiri.

Contoh kode untuk membersihkan isi dari semua *TextBox* dan *ComboBox* dalam form dengan 4 baris kode :

```
On Error Resume Next
For i = 0 To Controls.Count - 1
    Controls(i).Text = ""
Next
```

d. Obyek *Screen*

Seperti kita ketahui bahwa setiap form Visual Basic ditampilkan di layar monitor anda. Properti *Left*, *Top*, *Width*, dan *Height* dari form diekspresikan dengan satuan *Twips*. Merupakan unit pengukuran yang dapat digunakan pada layar dan printer.

Dalam printer, 1 inch = 1440 twips, sedangkan pada layar tergantung pada ukuran layar anda dan resolusi dari kartu VGA anda.

Anda dapat menemukan ukuran layar dalam satuan twips melalui properti *Width* dan *Height* dari obyek *Screen*. Berikut ini contoh kode untuk memindahkan form ke pojok kanan bawah layar :

```
Move Screen.Width - Width, Screen.Height - Height
```

Obyek *Screen* juga membantu anda menampilkan semua font yang dapat ditampilkan di layar.

```
` Panggil semua font yang dapat ditampilkan di layar
` Kemudian tampilkan hasilnya di listbox
Dim i as Integer
For i=0 to Screen.FontCount - 1
    List1.AddItem Screen.Fonts(i)
Next
```

Begitu juga untuk menampilkan jenis pointer dari mouse :

```
Screen.MousePointer = vbHourglass ` Jenis pointer dapat anda pilih
```

e. Metode *Graphic*

Visual Basic menyediakan beberapa metode grafik. Yang paling adalah *Cls*, dimana berfungsi membersihkan permukaan form, dan mengisinya dengan warna latar belakang yang telah ditentukan dalam properti *BackColor*.

## 1. Menggambar Titik

Salah satu metode untuk menggambar titik adalah *PSet*, dimana dengannya dapat membuat satu titik piksel tunggal yang berwarna pada permukaan form.

```
ForeColor = vbRed
PSet (100,100) ` Satu piksel merah
PSet (200,100), vbBlack ` Satu piksel hitam di sebelah kanannya
```

## 2. Menggambar Garis dan Kotak

Metode *Line* berguna untuk menggambar garis lurus, kotak kosong, dan kotak dengan warna solid.

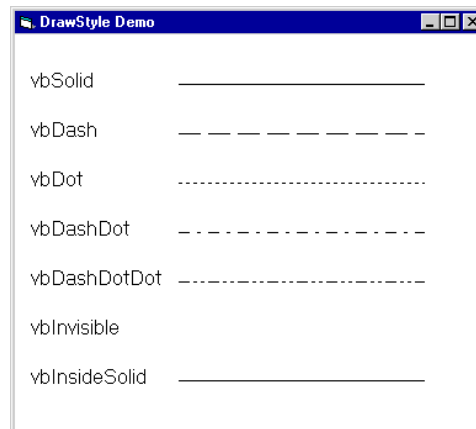
```
` Menggambar diagonal X berwarna merah pada form
DrawWidth = 5
Line (0, 0) - (ScaleWidth, ScaleHeight), vbRed
Line (ScaleWidth, 0) - (0, ScaleHeight), vbRed
```

```

` Menggambar Segi Tiga
Line (1000, 2000)-Step(1000, 0)   ' Garis Horizontal
Line -Step(0, 1000)                ' Garis Vertikal
Line -(1000, 2000)                 ' Menutup segitiga

```

Keluaran (*output*) dari metode Line juga dipengaruhi oleh properti lain, yaitu *DrawStyle*. Nilai default adalah 0 (*vbSolid*), namun anda dapat juga menggambar garis titik-titik dengan berbagai macam gaya.



**Gambar 2.7.** Efek dari berbagai pengaturan dari properti *DrawStyle*

**Tabel 2.2.** Konstan untuk properti *DrawStyle*

Konstan	Nilai	Deskripsi
vbSolid	0	Solid (nilai default)
vbDash	1	Garis terpisah
vbDot	2	Garis titik-titik
vbDashDot	3	Garis dengan kombinasi titik dan potongan garis
vbDashDotDot	4	Garis dgn kombinasi titik ganda & potongan garis
vbInvisible	5	Garis tak terlihat
vbInsideSolid	6	Garis <i>inside</i> (bagian dalam solid)

Dengan catatan bahwa properti *DrawStyle* akan berpengaruh pada keluaran grafik jika *DrawWidth* diset ke 1 piksel, jika tidak, maka properti *DrawStyle* akan dihiraukan dan garis akan selalu digambar dalam mode solid. Menambahkan B pada argumen keempat ke metode *Line* memungkinkan anda untuk menggambar kotak, dalam hal in, 2 titik adalah koordinat dari sudut atau ujung yang berlawanan.

```

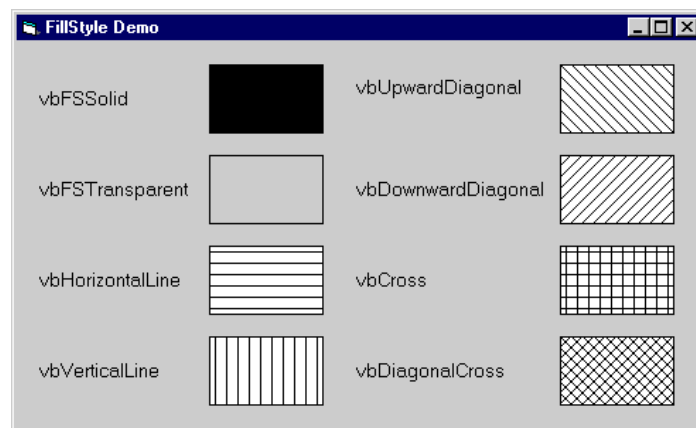
' Sebuah kotak biru, lebar 2000 twips dan tinggi 1000 twips
Line (500, 500)- Step (2000, 1000), vbBlue, B

```

Anda dapat mengaktifkan gaya pengisian solid dengan mengatur properti *FillStyle*, hasilnya dapat anda lihat pada gambar 2.8. Menariknya, Visual Basic menawarkan properti warna terpisah untuk

warna yang digunakan mengisi area, yaitu *FillColor*, yang memungkinkan anda untuk menggambar sebuah bentuk kotak dengan satu warna dan menggambar interiornya dengan warna lain dalam satu operasi tunggal. Berikut ini adalah contoh kodenya :

```
Line (500, 500)-Step(2000, 1000), RGB(64, 64, 64), BF
FillColor = vbFSSolid ' Kita ingin isi kotak yang solid.
FillColor = vbYellow ' Ini adalah warna dari isi kotak
Line (450, 450)-Step(2000, 1000), vbBlack, B
```



**Gambar 2.8.** Delapan gaya yang ditawarkan oleh properti *FillStyle*

Properti *FillStyle* dapat dimasukkan satu dari beberapa nilai, sebagaimana terlihat pada Tabel 2.3.

**Tabel 2.3.** Konstan untuk properti *FillStyle*

Konstan	Nilai	Deskripsi
vbFSSolid	0	Isi Solid
vbFSTransparent	1	Transparan (nilai default)
vbHorizontalLine	2	Garis horizontal
vbVerticalLine	3	Garis vertikal
vbUpwardDiagonal	4	Garis diagonal ke atas
vbDownwardDiagonal	5	Garis diagonal ke bawah
vbCross	6	Garis vertical dan horizontal bersilangan
vbDiagonalCross	7	Garis diagonal saling bersilangan

3. Menggambar Lingkaran, Elip, dan Busur  
Metode grafik yang berikutnya adalah *Circle*.

```
` Menggambar lingkaran dengan radius / jari-jari 1000 Twips
` Digambar di pojok kiri atas form
Circle (1200, 1200), 1000
```

Metode *Circle* dipengaruhi oleh nilai properti *DrawWidth*, *DrawStyle*, *FillStyle*, dan *FillColor*.

```
` Membuat lingkaran dengan lebar 3 piksel dengan pembatas
` berwarna hijau berisi warna kuning solid
DrawWidth = 3
FillStyle = vbFSSolid
FillColor = vbYellow
Circle (1200, 1200), 1000, vbGreen
```

Berikutnya contoh potongan kode untuk menggambar elips :

```
Sub Ellipse(X As Single, Y As Single, RadiusX As Single, _
    RadiusY As Single)
    Dim ratio As Single, radius As Single
    ratio = RadiusY / RadiusX
    If ratio < 1 Then
        radius = RadiusX
    Else
        radius = RadiusY
    End If
    Circle (X, Y), radius, , , , ratio
End Sub
```