

# Cepat Mahir Visual Basic 6.0

**Krisna D. Octovhiana**  
mail4krisna@yahoo.com

## ***Lisensi Dokumen:***

*Copyright © 2003 IlmuKomputer.Com*

*Seluruh dokumen di IlmuKomputer.Com dapat digunakan, dimodifikasi dan disebarkan secara bebas untuk tujuan bukan komersial (nonprofit), dengan syarat tidak menghapus atau merubah atribut penulis dan pernyataan copyright yang disertakan dalam setiap dokumen. Tidak diperbolehkan melakukan penulisan ulang, kecuali mendapatkan ijin terlebih dahulu dari IlmuKomputer.Com.*

## Bab 12

# Penggunaan Procedure

### 1.1 Apa itu Procedure ?

**Procedure** adalah blok kode program yang berisi perintah-perintah untuk mengerjakan tugas tertentu. Bila di dalam kode program yang kita buat ada perintah-perintah untuk melakukan tugas yang sama di beberapa tempat, maka akan lebih baik perintah-perintah tersebut dibuat dalam sebuah procedure. Kemudian, procedure itu bisa di-‘panggil’ bila diperlukan.

Penggunaan procedure sangat menghemat penulisan kode program, karena kode-kode program yang sama di beberapa tempat cukup dibuat pada satu bagian saja. Selain itu, procedure akan memudahkan perbaikan kode program bila terjadi perubahan atau kesalahan, karena perbaikan cukup dilakukan pada satu bagian saja.

## 1.2 Jenis-jenis Procedure

Pada VB6 ada 4 jenis procedure, yaitu :

- **Procedure Sub** – procedure yang tidak mengembalikan nilai setelah ‘tugas’-nya selesai.
- **Procedure Function** – procedure yang mengembalikan nilai setelah ‘tugas’-nya selesai.
- **Procedure Event** – procedure untuk suatu event pada sebuah object. Digunakan di dalam class module.
- **Procedure Property** – procedure untuk mengubah (*let*) atau mengambil (*get*) nilai property pada sebuah object. Digunakan di dalam class module.

Pada bab ini akan dibahas penggunaan procedure sub dan function. Procedure event dan property akan dibahas pada bab selanjutnya.

Bentuk penulisan (*syntax*) procedure sub :

```
[Public | Private] Sub <nama_sub> ([<argumen>])  
    ...  
    <isi procedure>  
    ...  
End Sub
```

Sedangkan bentuk penulisan (*syntax*) procedure function :

```
[Public | Private] Function <nama_function> ([<argumen>]) As <tipe_data>  
    ...  
    <isi procedure>  
    ...  
End Function
```

Pernyataan [Public | Private] menentukan ruang lingkup (*scope*) procedure. Sebuah procedure dengan scope public bisa digunakan dalam lingkup project. Sedangkan procedure dengan scope private hanya bisa digunakan dalam lingkup form saja.

<nama\_sub> atau <nama\_function> dibuat sebagai pengenal procedure saat di-‘panggil’. Aturan penamaan sebuah procedure sama dengan aturan penamaan sebuah variabel. Nama sebuah procedure dibuat unik, tidak boleh ada yang sama.

<argumen> merupakan serangkaian nilai dan tipe data yang dipakai oleh procedure untuk mengerjakan ‘tugas’-nya. Sebuah procedure bisa saja tidak memakai argumen sama sekali.

Pernyataan As <tipe\_data> pada procedure function menentukan tipe data nilai yang akan dikembalikan (*return value*) setelah ‘tugas’-nya selesai.

Untuk menggunakan sebuah procedure, maka procedure tersebut harus di-‘panggil’ pada bagian tertentu dari kode program. Procedure sub di-‘panggil’ dengan pernyataan :

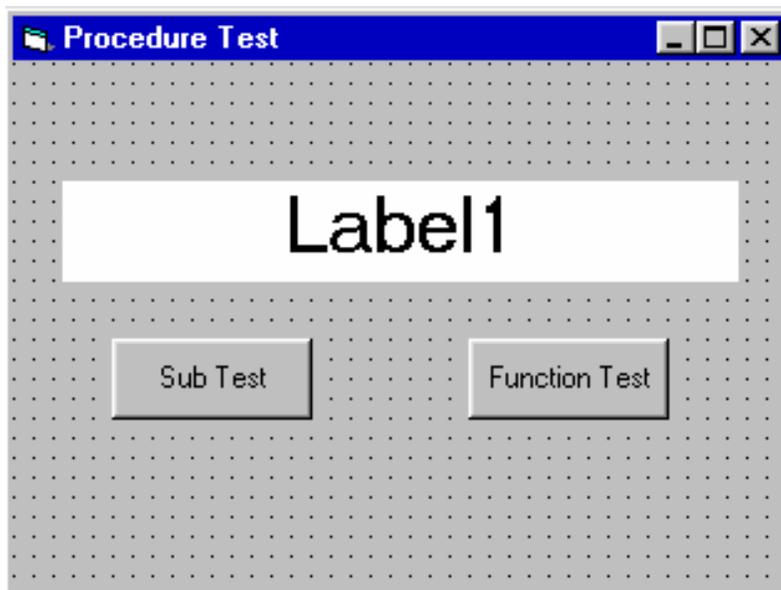
```
Call <nama_sub>([<argumen>])
```

Sedangkan procedure function bisa di-‘panggil’ langsung dengan menyisipkannya di dalam kode program yang memanggilnya.

Procedure boleh ditulis dimana saja dalam kode program, tetapi biasanya ditulis dibagian atas atau bawah agar mudah ditangani.

### 1.3 Contoh Program : Procedure Test

1. Aktifkan VB 6 melalui tombol Start.
2. Buka kembali project **Latihan.vbp**.
3. Tambahkan Form baru ke dalam Project.
4. Pada Jendela Form buatlah UI seperti ini :
5. (Gunakan komponen Label dan CommandButton)



6. Pengaturan property setiap object-nya adalah sebagai berikut :

Object	Properties	Value
Form11	Caption StartPosition	Procedure Test 2 – Center Screen
Label1	Alignment BackColor Font ToolTipText	2 – Center Palette : <putih> Size : 24 Dobel-Klik di Sini
Command1	Caption ToolTipText	Sub Test Klik di Sini
Command2	Caption	Function Test

7. Buka Jendela Code, lalu pada bagian Code Editor ketikkan kode programnya sebagai berikut :

```
Private Sub TulisTeks(teks As String, warna As ColorConstants)
    With Label1
        .Caption = teks
        .ForeColor = warna
    End With
End Sub

Private Function JumlahAngka() As String
    Dim angka1 As String, angka2 As String
    Dim hasil As Single

    angka1 = InputBox("Tulis angka 1 :", "Jumlah Angka")
    angka2 = InputBox("Tulis angka 2 :", "Jumlah Angka")

    If angka1 <> "" And angka2 <> "" Then
        hasil = CSng(angka1) + CSng(angka2)
        JumlahAngka = Cstr(hasil)
    End If
End Function

Private Sub Label1_DblClick()
    Call TulisTeks("Hai", vbBlue)
End Sub

Private Sub Command1_Click()
    Call TulisTeks("Hallo", vbRed)
End Sub

Private Sub Command2_Click()
    Label1.Caption = "Jumlah = " & JumlahAngka()
End Sub
```

8. Simpan Form11 (nama file : **Lat11.frm**).

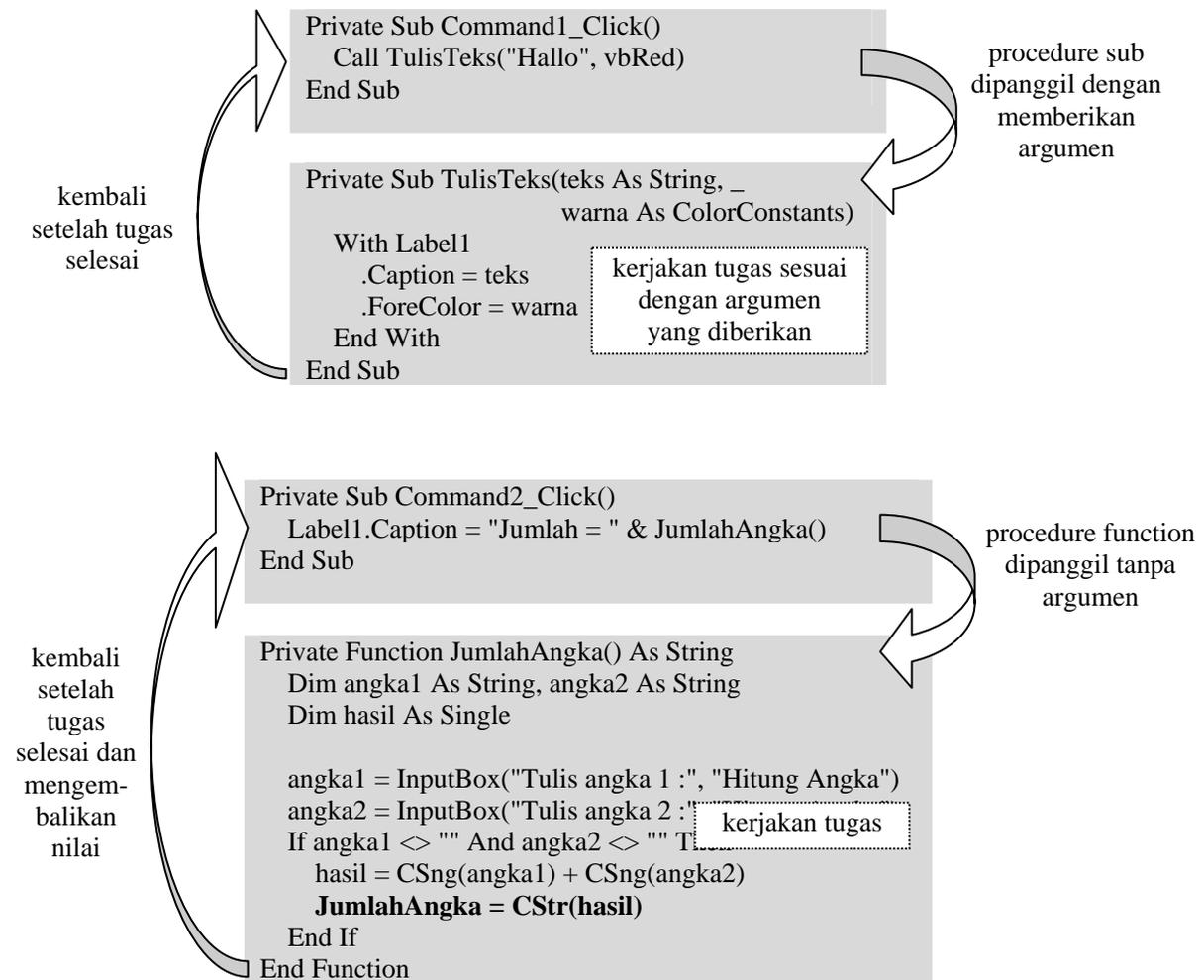
**Catatan** : Anda bisa men-download file Lat1-11.zip.

Klik menu Project > Project1 Properties lalu klik tab General. Gantilah Startup Object-nya menjadi Form11.

Coba jalankan Project-nya :

- Dobel-klik Label1, akan muncul tulisan “Hai” dengan warna biru.
- Klik Command1, akan muncul tulisan “Hallo” dengan warna merah.
- Klik Command2, akan muncul kotak input.
- Ketikkan angka 5, lalu klik tombol OK.
- Ketikkan angka 10, lalu klik tombol OK.
- Akan muncul tulisan “Jumlah = 15”.

### Proses ketika procedure di-'panggil' :



Dapat anda lihat bahwa prosedur Tulisteks dipanggil, dan dimasukkan 2 variabel ke dalamnya yaitu teks dan warna. Selanjutnya kedua argumen tersebut diolah oleh prosedur Tulisteks. Akhirnya ditampilkan tulisan dan warnanya yang merupakan hasil dari properti Caption dan Forecolor dari obyek label.

## 1.4 Pernyataan Optional

Nilai argumen di dalam sebuah procedure dapat dibuat menjadi optional (boleh ada atau tidak ada). Bila nilai argumen yang optional tidak ada, maka yang digunakan adalah nilai default-nya. Untuk membuat nilai argumen menjadi optional, tambahkan pernyataan **Optional** di depan argumen-nya dan berikan nilai default-nya, contoh :

```
Private Sub TulisTeks(teks As String, Optional warna As ColorConstants =  
vbMagenta)  
    With Label1  
        .Caption = teks  
        .ForeColor = warna  
    End With  
End Sub
```

Bila nilai argumen warna tidak ada, maka warna teks pada Label1 adalah magenta, contoh :

```
Call TulisTeks("Hallo")
```

## 1.5 Pernyataan ByRef dan ByVal

Nilai argumen yang dikirimkan ke dalam procedure bisa saja berasal dari sebuah variabel. Namun demikian, perlu diperhatikan bahwa nilai argumen yang dikirimkan ke dalam procedure bisa berdasarkan referensi (ByRef) atau berdasarkan nilai (ByVal). Default-nya nilai argumen dikirimkan berdasarkan referensi. Apabila variabel dikirimkan berdasarkan referensi, maka semua perubahan yang dilakukan terhadap argumen akan mempengaruhi variabel tersebut. Hal ini terkadang menimbulkan *logical error*. Sebagai ilustrasi perhatikan contoh berikut ini :

Dim Harga As Currency, Total As Currency

```
Private Sub HitungDiskon(HargaAwal As Currency, Diskon As Single)  
    HargaAwal = HargaAwal * (1 - Diskon)  
    Total = CCur(HargaAwal)  
End Sub  
  
Private Sub Command3_Click()  
    Harga = 10000  
    Total = 0  
    Call HitungDiskon(Harga, 0.1)  
    MsgBox "Harga " & Harga & " setelah diskon 10% menjadi " & Total  
End Sub
```

Hasilnya → **Harga 9000 setelah diskon 10% menjadi 9000**

Variabel Harga dikirimkan berdasarkan referensi (ByRef), sehingga perubahan pada argumen HargaAwal akan mempengaruhi variabel Harga.

Modifikasi procedure sub HitungDiskon menjadi sebagai berikut :

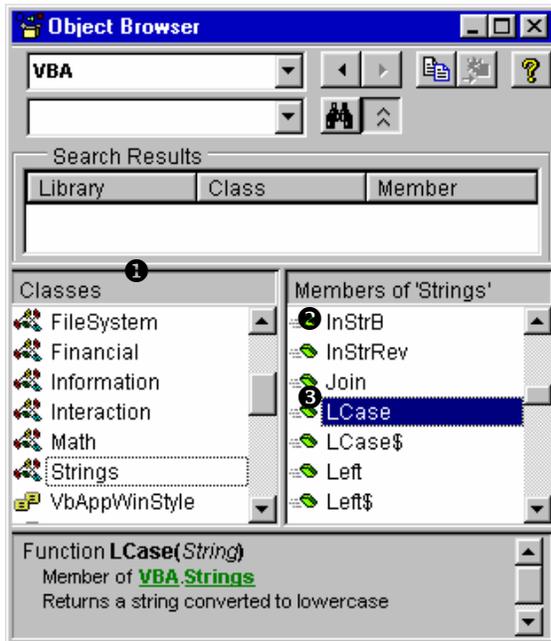
```
Private Sub HitungDiskon(ByVal HargaAwal As Currency, Diskon As Single)  
    HargaAwal = HargaAwal * (1 - Diskon)  
    Total = CCur(HargaAwal)  
End Sub
```

Hasilnya → **Harga 10000 setelah diskon 10% menjadi 9000**

Sekarang variabel Harga dikirimkan berdasarkan nilai (ByVal), sehingga perubahan pada argumen HargaAwal tidak akan mempengaruhi variabel Harga.

## 1.6 Procedure Built-in

VB6 menyediakan cukup banyak procedure yang siap-pakai (*built-in*) untuk berbagai ‘tugas’. Coba klik menu View – Object Browser atau tekan tombol F2 pada keyboard :



1. Cari module Strings pada bagian Classes.
2. Lihat daftar pada bagian Member of ‘Strings’ yang merupakan kumpulan procedure dengan ‘tugas’ pengolahan data string.
3. Klik nama procedure untuk mengetahui apa ‘tugas’-nya.

\*\*\*