Modul 8

Object Oriented Programming dalam Visual Basic 2005

Pokok Bahasan:
❑ Class dan Object
❑ Field
❑ Metode
❑ Properti
❑ Deklarasi Objek dan Instantiasi Objek
❑ Constructor
❑ Event
❑ Inheritance

8.1 Class dan Object

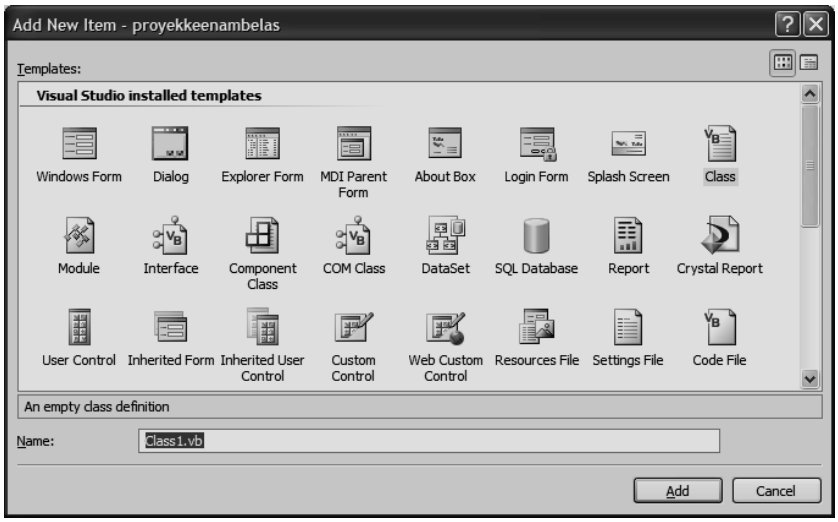
Seperti sudah dijelaskan sebelumnya pada Subbab 4.3, Visual Basic 2005 sangat mendukung *object-oriented programming* (OOP). Dalam konsep OOP, segala sesuatu direpresentasikan dalam bentuk Object. Data dan fungsi digabungkan ke dalam sebuah unit yang disebut Object Object biasanya merupakan representasi dari "dunia nyata". Object merupakan hasil cetakan dari sesuatu, yaitu Class.

Class (kelas) bisa memiliki Field, Property (properti), Method (metode), dan Event. Setiap Object (objek) hasil instantiasi (cetakan) dari sebuah Class juga akan memiliki Field, properti, metode, dan Event dari Class tersebut. Namun, nilai setiap Field atau Property dari masing-masing objek dapat saja berbeda-beda, walaupun mereka berasal dari Class yang sama.

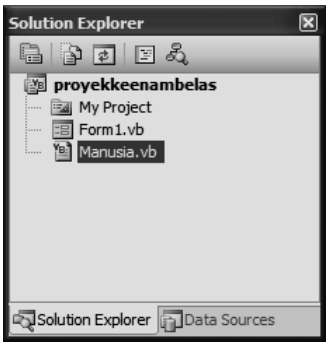
Dalam Visual Basic 2005, kita dapat membuat Class sendiri. Class dibuat di dalam sebuah Modul khusus yang disebut Modul Class

(*Class Module*). Untuk menambahkan sebuah Modul Class ke dalam Project, lakukanlah langkah berikut:

1. Klik menu **Project** > **Add Class..** dalam Menu Bar.
2. Kemudian ketika muncul jendela **Add New Item** seperti di bawah ini, ketiklah nama Class yang ingin Anda buat dalam kotak isian **Name**. Lalu klik tombol **Add**. Nama dari Modul Class tersebut berarti juga nama bagi Class baru yang kita buat.



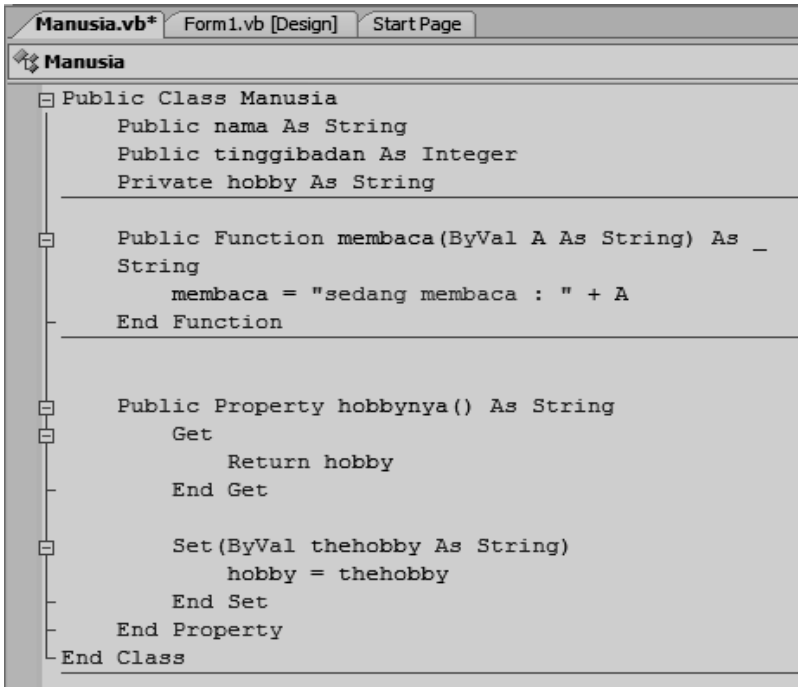
3. Secara otomatis Class baru tersebut ditambahkan dalam Project yang kita buat (lihat dalam bagian **Solution Explorer**). Contoh:



4. Untuk membuat kode program dalam Class, klik ganda terlebih dahulu nama Class tersebut pada bagian **Solution Explorer**.

Sebuah Class umumnya diawali oleh kata **Public Class** yang diikuti nama Class-nya serta diakhiri kata **End Class**. Kode program buatan kita untuk Class itu dituliskan di antara kata-kata tersebut.

Berikut ini contoh sebuah Class bernama **Manusia**, di dalamnya terdapat berbagai kode program yang menunjukkan Field, Property, Metode, atau Event yang dimiliki oleh Class tersebut:



8.2 Field

Field adalah variabel yang dideklarasikan dalam Class. Field disebut juga data yang dimiliki oleh Class. Cara mendeklarasikan sebuah Field hampir sama dengan cara membuat sebuah variabel biasa.

Untuk mendeklarasikan sebuah Field gunakan aturan penulisan sebagai berikut:

```
<katakunci> <namafield> As <T_D>
```

Di mana,

- **katakunci** digunakan untuk menentukan sifat bagi Field tersebut. Dalam Visual Basic 2005 ada 5 kata kunci yang bisa digunakan, yaitu :
    - **Public**, artinya Field itu dapat diakses oleh seluruh kode program, baik yang berada di dalam Class tempat Field tersebut berada atau di luar Class.
    - **Private**, artinya hanya dapat diakses oleh kode program yang ada dalam Class.
    - **Protected**, artinya hanya dapat diakses oleh kode program yang ada dalam sebuah Class, serta kode program yang berada di dalam Class lain yang merupakan turunan dari Class tersebut.
    - **Friend**, artinya hanya dapat diakses oleh seluruh kode program yang berada dalam Class serta kode program lain yang berada di luar Class, tapi masih dalam Project yang sama.

      Kata kunci **Friend** memiliki sifat yang hampir sama dengan kata kunci **Public**. Namun, kata kunci Public cakupannya lebih luas. Dalam kata kunci Public, seluruh kode program baik di dalam Project ataupun di luar Project (Project lain) juga dapat mengaksesnya.
    - **ProtectedFriend**, memiliki sifat yang merupakan gabungan dari kata kunci **Protected** dan kata kunci **Friend**.
- **T_D** adalah tipe data bagi Field tersebut. Pada dasarnya Field juga adalah sebuah variabel yang memiliki tipe data.

Misalkan kita telah membuat sebuah Class bernama **Manusia**. Kemudian andaikan Class **Manusia** tersebut memiliki Field **tinggibadan** dan **hobby**, maka kita harus menuliskan kode

program untuk membuat kedua Field tersebut dalam Class **Manusia**.

Misalkan Field bernama **tinggibadan** kita atur agar bersifat **Public**, sedangkan Field **hobby** bersifat **Private**. Andaikan juga Field **tinggibadan** digunakan untuk menampung nilai-nilai bertipe data Integer dan field **Hobby** digunakan untuk menampung String, maka Class Manusia tersebut dapat dilengkapi dengan kode program seperti berikut:

```
Public Class Manusia

       Public tinggibadan As Integer
       Private hobby As String

End Class
```

8.3 Metode

Metode (*Method*) menunjukkan apa saja yang dapat dilakukan oleh Objek hasil instantiasi Class tersebut. Dalam Visual Basic 2005, Metode dapat diimplementasikan dalam bentuk Sub atau Function.

Jika Metode yang dibuat adalah berupa Sub, berarti metode tersebut tidak mengembalikan nilai apa pun ke kode program yang memanggilnya. Hal ini disebut sebagai *Imperative Method*.

Jika Metode yang dibuat berupa Fungsi, berarti metode tersebut mengembalikan sebuah nilai ke kode program yang memanggilnya. Hal ini disebut sebagai *Interrogative Method*.

Sama seperti halnya Field, Metode juga memiliki kata kunci, bisa berupa Public, Private, Protected, Friend, atau ProtectedFriend.

Misalkan sebuah Class bernama **Manusia** memiliki sebuah Metode bernama **Mengangkatbarang**. Maka harus kita buatkan sebuah Sub atau Fungsi untuk merepresentasikan kemampuan **Mengangkatbarang** tersebut.

Andaikan kita buatkan saja sebuah Metode berupa Fungsi bernama **Mengangkatbarang** yang mampu menerima input sebuah angka yang menunjukkan berat barang yang diangkat. Kemudian andaikan Metode tersebut mengembalikan sebuah String yang

merupakan pesan tentang apakah barang diangkat terlalu berat atau tidak. Maka Class **manusia** tersebut dapat dilengkapi menjadi seperti berikut:

```
Public Class Manusia

  Public tinggibadan As Integer
  Private hobby As String

  Public Function Mengangkatbarang(ByVal X As String) As String

        If X >= 100
               Mengangkatbarang="Barangnya sangat berat"
        Else
               Mengangkatbarang="Barangnya tidak berat"
        End If

  End Function

End Class
```

8.4 Properti

Properti (*Property*) adalah sebuah Metode khusus yang digunakan untuk mendapatkan atau mengubah nilai dari sebuah Field dalam Class. Penggunaan *Property* biasanya ditujukan untuk Field yang bersifat **Private**.

Field yang bersifat **Public** dapat diakses **langsung** dari luar Class, sedangkan Field yang bersifat **Private** hanya dapat diakses dari luar Class **melalui *Property***.

Aturan penulisan *Property* adalah sebagai berikut:

```
<katakunci> Property <namaprop>() As <tipedataprop>

       Get
         ...
         Return <namafield>
       End Get

       Set(<T_p> <Arg> As <T_d>)
         ...
         ...
       End Set

End Property
```

Di mana,

- **katakunci** bisa berupa Public, Private, Protected, Friend, atau ProtectedFriend.
- **namaprop** adalah nama *Property*-nya.
- **tipedataprop** adalah tipe data *Property*-nya. Hal ini menunjukkan tipe data dari nilai yang akan dikembalikan oleh *Property* tersebut.
- Bagian **Get** sampai **End Get** adalah bagian *Property* yang digunakan untuk mendapatkan nilai dari Field.
- Perintah **Return** digunakan untuk mengirimkan sebuah Nilai ke kode program yang memanggil *Property* tersebut.
- Bagian **Set** sampai **End Set** adalah bagian *Property* yang digunakan untuk mengatur atau mengubah nilai suatu Field tertentu.
- **T_p** adalah tipe parameter formal bagian **Set**. Biasanya ByVal.
- **Arg** adalah nama parameter formal yang akan digunakan bagian **Set** untuk menampung parameter aktual yang diinputkan padanya.
- **T_d** adalah tipe data dari parameter formal tersebut.

Perhatikan contoh pada Subbab 8.2, telah diasumsikan bahwa terdapat sebuah Field bernama **hobby** yang bersifat **Private**. Sehingga isi Field tersebut tidak dapat langsung diubah atau diakses dari luar Class. Oleh karena itu, harus dibuatkan sebuah *Property* khusus untuk mengubah atau mengakses Field bernama **hobby**. Andaikan *Property* tersebut diberi nama **thehobby**, Class Manusia dapat kita lengkapi sebagai berikut:

```
Public Class Manusia

  Public tinggibadan As Integer
  Private hobby As String


  Public Function Mengangkatbarang(ByVal X As String) As String
       If X >= 100
              Mengangkatbarang="Barangnya sangat berat"
```

```
      Else
            Mengangkatbarang="Barangnya tidak berat"
      End If
  End Function

  Public Property thehobby() As String
      Get
        Return hobby
      End get

      Set(ByVal Z As String)
        hobby=Z
      End Set
  End Property

End Class
```

8.5 Deklarasi Objek dan Instantiasi Objek

Membuat sebuah Objek, berarti mendeklarasikan dan menginstantiasi objek. Objek dideklarasikan dan diinstantiasi di dalam kode program yang akan menggunakan objek tersebut, misalnya di dalam **Form1**. Cara untuk mendeklarasikan sebuah Objek hampir sama dengan cara mendeklarasikan sebuah variabel biasa.

Untuk mendeklarasikan sebuah Objek gunakan aturan penulisan sebagai berikut:

```
Dim <namaobjek> As <NamaClass>
```

Setelah objek dideklarasikan, kemudian objek tersebut diinstantiasi menggunakan perintah **New**. Aturan penulisannya sebagai berikut:

```
<namaobjek> = New <namaClass>()
```

Andaikan kita akan mendeklarasikan dan menginstantiasi sebuah **Objek** bernama **Adit** dari **Class** bernama **Manusia**, maka dapat kita tuliskan kode program berikut:

```
Dim Adit As Manusia
Adit = New Manusia()
```

Selain dengan cara penulisan di atas, objek Adit tersebut dapat juga dibuat dengan cara berikut:

```
Dim Adit As New Manusia()
```

Atau,

```
Dim Adit As Manusia = New Manusia()
```

Jika sebuah Objek sudah tidak dipakai lagi dalam kode program, maka kita dapat menghancurkannya dengan aturan penulisan berikut:

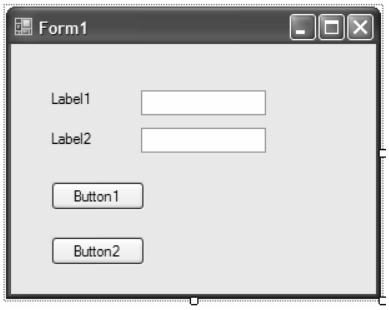
```
<namaobjek> = Nothing
```

Contoh:

```
Adit = Nothing
```

Lakukanlah langkah-langkah berikut untuk mencoba menggunakan dan membuat Class, Field, Properti, Metode, dan Objek dalam kode program Anda:

1. Buatlah sebuah Project baru bernama **proyekkeenambelas**.
2. Tambahkan dua buah **Button** serta dua buah **Label** dan **TextBox** ke dalam Form sehingga menjadi seperti berikut.

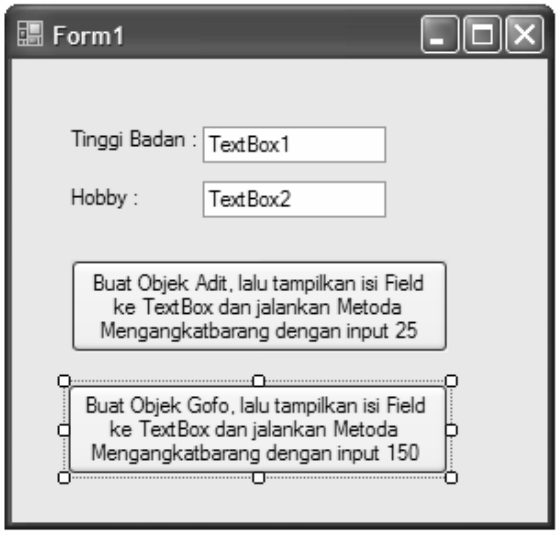


3. Atur properti masing-masing **Label**, **TextBox**, dan **Button** tersebut seperti berikut.

Kontrol	Properti	Nilai
Label1	Text	Tinggi Badan:
Label2	Text	Hobby:
TextBox1	(Name) Text	TextBox1 TextBox1

TextBox2	(Name) Text	TextBox2 TextBox2
Button1	(Name) Text	Btn_adit Buat Objek Adit, lalu tampilkan isi Field ke TextBox dan jalankan Metode Mengangkatbarang dengan input 25
Button2	(Name) Text	Btn_gofo Buat Objek Gofo, lalu tampilkan isi Field ke TextBox dan jalankan Metode Mengangkatbarang dengan input 150

4. Aturlah kembali posisi **Label**, **TextBox**, dan **Button**-nya sehingga menjadi seperti berikut.



5. Klik menu **Project** > **Add Class...** untuk menambahkan Class baru. Beri nama Class tersebut dengan nama **Manusia**.

6. Klik ganda nama Class **Manusia** pada **Solution Explorer** untuk membuka jendela kode program Class tersebut. Lalu lengkapi dengan kode program berikut:

```
Public Class Manusia

  Public tinggibadan As Integer
  Private hobby As String


  Public Function Mengangkatbarang(ByVal X As String) As String

       If X >= 100
              Mengangkatbarang="Barangnya sangat berat"
       Else
              Mengangkatbarang="Barangnya tidak berat"
       End If

  End Function

  Public Property thehobby() As String
       Get
         Return hobby
       End get

       Set(ByVal Z As String)
         hobby=Z
       End Set
  End Property

End Class
```

7. Klik Form bernama **Form1** pada **Solution Explorer**. Kemudian klik menu **View** > **Code** pada Menu Bar untuk membuka jendela kode program **Form1**. Lalu lengkapi kode program yang ada menjadi seperti berikut. Kode program Anda ditulis di bagian Event **Click** dari objek **Btn_adit** dan **Btn_gofo**:

```
Public Class Form1

    Private Sub Btn_adit_Click(ByVal sender As Object,
    ByVal e As System.EventArgs) Handles Btn_adit.Click

         Dim adit As Manusia
         adit = New Manusia()

         adit.tinggibadan = 170
```

```
        adit.thehobby = "main game"

        TextBox1.Text = adit.tinggibadan
        TextBox2.Text = adit.thehobby

        Dim pesan = adit.Mengangkatbarang(25)

        MessageBox.Show(pesan, _
         "Hasil dari metoda Mengangkatbarang", _
         MessageBoxButtons.OK, _
         MessageBoxIcon.Information)

    End Sub

    Private Sub Btn gofo Click(ByVal sender As Object,
    ByVal e As System.EventArgs) Handles Btn_gofo.Click

        Dim gofo As Manusia
        gofo = New Manusia()

        gofo.tinggibadan = 165
        gofo.thehobby = "membaca"

        TextBox1.Text = gofo.tinggibadan
        TextBox2.Text = gofo.thehobby

        Dim pesan = gofo.Mengangkatbarang(150)

        MessageBox.Show(pesan, _
         "Hasil dari metoda Mengangkatbarang", _
         MessageBoxButtons.OK, _
         MessageBoxIcon.Information)

    End Sub
End Class
```

8. Klik menu **Build** > **Build proyekkeenambelas** pada menu bar untuk mengompilasi program aplikasi yang baru saja Anda buat.
9. Setelah proses kompilasi selesai, klik menu **Debug** > **Start Debugging** pada menu bar untuk mencoba menjalankan program aplikasi Anda. Klik dua buah tombol yang berada pada **Form1**..

10. Tutuplah aplikasi Anda dengan mengklik menu **Debug** > **Stop Debugging**.
11. Simpan seluruh Form dan Project.

8.6 Constructor

Konstruktor adalah metode yang dijalankan secara otomatis ketika sebuah objek diinstantiasi dari sebuah Class. Konstruktor dibuat dengan cara membuat *M*etode berupa Sub yang bernama **New** di dalam Class.

Berikut ini sebuah contoh kontruktor sederhana beserta beberapa kode program di dalamnya:

```
Public Sub New()

      tinggibadan=0

End Sub
```

Jika sebuah Objek diciptakan dari Class yang memiliki konstruktor seperti di atas, secara otomatis konstruktor akan dijalankan sehingga isi Field bernama **tinggibadan** dari Objek tersebut akan diberi nilai 0.

Konstruktor juga bisa menerima beberapa parameter aktual. Berikut ini sebuah contoh konstruktor yang mampu menerima sebuah parameter aktual bernama X beserta beberapa kode program di dalamnya:

```
Public Sub New(ByVal X As String)

      tinggibadan=0
      hobby=X

End Sub
```

Jika sebuah Objek diciptakan dari Class yang memiliki konstruktor seperti di atas, secara otomatis konstruktor akan dijalankan sehingga isi Field bernama **tinggbadan** dari Objek tersebut akan diberi nilai 0. Isi Field bernama **hobby** secara otomatis akan diberi nilai sesuai input X yang diterima konstruktor tersebut.

Jika sebuah Class memiliki konstruktor yang menerima paramater aktual, maka parameter-parameter aktualnya harus disertakan ketika sebuah Objek diinstantiasi dari Class tersebut. Contoh:

```
Dim Adit As Manusia
Adit= New Manusia("Main Game")
```

Lakukanlah langkah-langkah berikut untuk mencoba menggunakan dan membuat Class yang memiliki konstruktor dalam kode program Anda:

1. Buka kembali Project bernama **proyekkeenambelas** yang pernah Anda buat sebelumnya.
2. Klik ganda nama Class **Manusia** pada **Solution Explorer** untuk membuka jendela kode program Class tersebut. Lalu ubah kode program di dalamnya sehingga menjadi seperti berikut:

```
Public Class Manusia

    Public tinggibadan As Integer
    Public namalengkap As String
    Private hobby As String

    Public Sub New(ByVal X As String)
    tinggibadan = 0
    namalengkap = X

     MessageBox.Show("Field namalengkap otomatis di isi " + X, _
        "Sebuah objek baru diciptakan", _
        MessageBoxButtons.OK, _
        MessageBoxIcon.Information)
    End Sub



   Public Function Mengangkatbarang(ByVal X As String) As String

        If X >= 100 Then
            Mengangkatbarang = "Barangnya sangat berat"
        Else
            Mengangkatbarang = "Barangnya tidak berat"
        End If

    End Function

    Public Property thehobby() As String
        Get
```

```
            Return hobby
        End Get

        Set(ByVal Z As String)
            hobby = Z
        End Set
    End Property
End Class
```

3. Klik Form bernama **Form1** pada **Solution Explorer**. Kemudian klik menu **View** > **Code** pada Menu Bar untuk membuka jendela kode program **Form1**. Lalu ubah kode program yang ada menjadi seperti berikut. Kode program yang diubah ada di bagian Event **Click** dari objek **Btn_adit** dan **Btn_gofo**:

```
Public Class Form1

    Private Sub Btn_adit_Click(ByVal sender As Object,
    ByVal e As System.EventArgs) Handles Btn_adit.Click

        Dim adit As Manusia
        adit = New Manusia("Aditya Nugroho")

        adit.tinggibadan = 170
        adit.thehobby = "main game"

        TextBox1.Text = adit.tinggibadan
        TextBox2.Text = adit.thehobby

        Dim pesan = adit.Mengangkatbarang(25)

        MessageBox.Show(pesan, _
         "Hasil dari metoda Mengangkatbarang", _
         MessageBoxButtons.OK, _
         MessageBoxIcon.Information)

    End Sub

    Private Sub Btn gofo Click(ByVal sender As Object,
    ByVal e As System.EventArgs) Handles Btn_gofo.Click

        Dim gofo As Manusia
        gofo = New Manusia("Abdolgofo")

        gofo.tinggibadan = 165
        gofo.thehobby = "membaca"
```

```
        TextBox1.Text = gofo.tinggibadan
        TextBox2.Text = gofo.thehobby

        Dim pesan = gofo.Mengangkatbarang(150)

        MessageBox.Show(pesan, _
         "Hasil dari metoda Mengangkatbarang", _
         MessageBoxButtons.OK, _
         MessageBoxIcon.Information)
    End Sub
End Class
```

4. Klik menu **Build** > **Build proyekkeenambelas** pada menu bar untuk mengompilasi program aplikasi yang baru saja Anda ubah.
5. Setelah proses kompilasi selesai, klik menu **Debug** > **Start Debugging** pada menu bar untuk mencoba menjalankan program aplikasi Anda. Klik dua buah tombol yang berada pada **Form1**.
6. Tutuplah aplikasi Anda dengan mengklik menu **Debug** > **Stop Debugging**.
7. Simpan seluruh Form dan Project.

8.7 Event

8.7.1 Mengenal Event

Seperti pernah dijelaskan sebelumnya, Event adalah segala sesuatu yang dapat dikenakan terhadap sebuah Class atau Objek. Definisi yang lain adalah bahwa ketika sebuah Objek dikenakan sesuatu tindakan, maka Event yang dimilikinya akan dibang-kitkan. Ketika sebuah Event dibangkitkan, maka sebuah Sub yang ditugaskan untuk menangani (meng-*handle*) Event tersebut akan dikerjakan.

Kita sudah sering menggunakan Event, misalnya Event Click yang dimiliki oleh sebuah Button atau Event Load yang dimiliki sebuah Form.

Misalnya Anda ingin menuliskan kode program pada Event **Click** sebuah **Button** bernama **Button1**, maka biasanya Anda menuliskannya di dalam kode program berikut:

```
Private Sub Button1_Click(ByVal sender As Object, ByVal e
    As System.EventArgs) Handles Button1.Click

    ....
    ....
    ....

End Sub
```

Perhatikan bagian kode program yang bergaris bawah, yaitu :

```
Private Sub Button1_Click(ByVal sender As Object, ByVal e
    As System.EventArgs)
```

Bagian kode program di atas memiliki arti bahwa terdapat sebuah Sub bernama **Button1_Click** yang memiliki dua parameter formal, yaitu **sender** dan **e**.

Sekarang perhatikan bagian kode program berikutnya yang bergaris bawah putus-putus, yaitu:

```
Handles Button1.Click
```

Bagian kode program di atas berarti bahwa Sub bernama **Button1_Click** tadi akan menangani (meng-*handle*) sebuah Event bernama **Click** dari sebuah objek bernama **Button1**. Atau dengan kata lain, ketika Event **Click** dibangkitkan oleh **Button1**, maka Sub bernama **Button1_Click** akan dikerjakan beserta kode program di dalamnya.

Dalam menentukan Sub yang akan dikerjakan ketika sebuah Event dibangkitkan, Visual Basic 2005 tidak membedakan nama Sub atau bahkan nama parameter formalnya. Asalkan suatu Sub tersebut memiliki jumlah parameter formal yang sama, tipe data parameter formal yang sama, serta diatur agar meng-*handle* sebuah Event dari suatu Objek, maka ketika Event yang dimaksud dibangkitkan oleh Objek, Sub tersebut akan dikerjakan.

Perhatikan contoh Sub berikut:

```
Private Sub MySubKeren(ByVal X As Object, ByVal Y As
    System.EventArgs) Handles Button1.Click

    ....
    ....
    ....

End Sub
```

Pada kode program di atas, terdapat sebuah Sub bernama **MySubKeren** yang memiliki parameter formal X dan Y. Bagian **Handles Button1.Click** menyatakan bahwa **MySubKeren** akan menangani (meng-*handle*) Event **Click** yang dibangkitkan oleh **Button1**. Artinya, ketika Event **Click** dibangkitkan oleh **Button1**, maka Sub bernama **MySubKeren** akan dikerjakan. Hal ini sama dengan Sub bernama Button1_Click yang tadi.

8.7.2 Membuat Event Sendiri

Kita dapat membuat Event sendiri dalam Class buatan kita. Sebuah Event dideklarasikan dengan kata kunci Event dan memiliki aturan penulisan berikut:

```
[<KKun>] Event <N_event>(<T_p> <Arg> As <T_Arg>)
```

Di mana,

- `KKun` adalah kata kunci yang menandakan sifat bagi Event tersebut.
- `N_event` adalah nama Event yang akan kita buat.
- `T_p` adalah tipe dari parameter formalnya.
- `Arg` adalah parameter formal, yaitu variabel yang akan menerima input bagi Event tersebut, untuk kemudian dikirimkan ke Sub yang menangani (meng-*handle*) Event tersebut.
- `T_Arg` adalah tipe data dari parameter formalnya.

Misalkan Class bernama **Manusia** memiliki sebuah Event bernama **digaji** yang menerima input sebuah angka yang menunjukkan

besar gajinya. Maka kita dapat tambahkan deklarasi Event tersebut ke dalam Class Manusia seperti berikut (lihat yang bercetak miring):

```
Public Class Manusia

    Public tinggibadan As Integer
    Public namalengkap As String
    Private hobby As String

    Public Event digaji(ByVal besarnyagaji As Integer)

    Public Sub New(ByVal X As String)
    tinggibadan = 0
    namalengkap = X

     MessageBox.Show("Field namalengkap otomatis di isi " + X, _
        "Sebuah objek baru diciptakan", _
        MessageBoxButtons.OK, _
        MessageBoxIcon.Information)
    End Sub


   Public Function Mengangkatbarang(ByVal X As String) As String

        If X >= 100 Then
            Mengangkatbarang = "Barangnya sangat berat"
        Else
            Mengangkatbarang = "Barangnya tidak berat"
        End If

    End Function

    Public Property thehobby() As String
        Get
            Return hobby
        End Get

        Set(ByVal Z As String)
            hobby = Z
        End Set
    End Property
End Class
```

Kemudian setelah Event dideklarasikan, kita harus membuat kode program yang akan membangkitkan Event tersebut. Untuk membangkitkan sebuah Event, kita tuliskan sebuah kata kunci **RaiseEvent** diikuti nama Event-nya beserta parameter aktual yang dikirimkan ke Event tersebut.

Andaikan Event **digaji** tersebut akan dibangkitkan ketika Metode bernama **Mengangkatbarang** dikerjakan, maka kita dapat tambahkan kode programnya di dalam Metoda **Mengangkatbarang** seperti berikut (lihat bagian yang bercetak miring):

```
Public Class Manusia

    Public tinggibadan As Integer
    Public namalengkap As String
    Private hobby As String

    Public Event digaji(ByVal besarnyagaji As Integer)

    Public Sub New(ByVal X As String)
    tinggibadan = 0
    namalengkap = X

     MessageBox.Show("Field namalengkap otomatis di isi " + X, _
        "Sebuah objek baru diciptakan", _
        MessageBoxButtons.OK, _
        MessageBoxIcon.Information)
    End Sub

   Public Function Mengangkatbarang(ByVal X As String) As String
        If X >= 100 Then

            RaiseEvent digaji(5000)
           Mengangkatbarang = "Barangnya sangat berat"

        Else

            RaiseEvent digaji(2500)
           Mengangkatbarang = "Barangnya tidak berat"
        End If
    End Function

    Public Property thehobby() As String
        Get
            Return hobby
        End Get

        Set(ByVal Z As String)
            hobby = Z
        End Set
    End Property

End Class
```

Sekarang kita sudah deklarasikan Event-nya serta sudah buatkan kode program yang akan membangkitkan Event tersebut. Lalu

selanjutnya kita harus membuat Sub yang akan menangani (meng-*handle*) Event tersebut ketika dibangkitkan.

Sub yang meng-*handle* sebuah Event dituliskan di luar Class, misalnya di dalam **Form1**. Misalkan Sub tersebut bernama **pesangaji**, maka dapat kita tuliskan sebagai berikut:

```
Public Sub pesangaji(ByVal agkgj As Integer) Handles adit.digaji

        Dim pesan As String
        pesan="Objek ini digaji sebesar" + CStr(agkgj)

        MessageBox.Show(Pesan, _
         "dari Sub yang menangani Even digaji", _
        MessageBoxButtons.OK, _
        MessageBoxIcon.Information)

End Sub
```

Sub di atas digunakan untuk meng-*handle* Event **digaji** yang dibangkitkan oleh objek **Adit**. Sub tersebut memiliki sebuah parameter formal bernama **agkgj**. Parameter formal ini akan menerima input dari paramater aktual yang dikirimkan oleh kode program yang membangkitkan Event. Isi parameter formal tersebut kemudian ditampilkan dalam sebuah **MessageBox**.

Jika sebuah Class memiliki Event, maka tambahkan kata kunci **WithEvents** ketika sebuah Objek dideklarasikan dari Class tersebut. Dengan kata kunci **WithEvents**, berarti Objek hasil instantiasi dari Class akan dapat menggunakan Event tersebut. Misalnya kita ingin membuat Objek **Adit** dari Class bernama **Manusia** yang memiliki Event **digaji**, maka gunakan kode program berikut saat mendeklarasikannya:

```
Dim WithEvents adit As Manusia
```

Lakukanlah langkah-langkah berikut untuk mencoba menggunakan dan membuat Class yang memiliki konstruktor dalam kode program Anda:

1. Buka kembali Project bernama **proyekkeenambelas** yang pernah Anda buat sebelumnya.

2. Klik ganda nama Class **Manusia** pada **Solution Explorer** untuk membuka jendela kode program Class tersebut. Lalu ubah kode program di dalamnya sehingga menjadi seperti berikut:

```
Public Class Manusia

    Public tinggibadan As Integer
    Public namalengkap As String
    Private hobby As String

    Public Event digaji(ByVal besarnyagaji As Integer)

    Public Sub New(ByVal X As String)
    tinggibadan = 0
    namalengkap = X

     MessageBox.Show("Field namalengkap otomatis di isi " + X, _
        "Sebuah objek baru diciptakan", _
        MessageBoxButtons.OK, _
        MessageBoxIcon.Information)
    End Sub



   Public Function Mengangkatbarang(ByVal X As String) As String
        If X >= 100 Then

            RaiseEvent digaji(5000)
           Mengangkatbarang = "Barangnya sangat berat"

        Else

            RaiseEvent digaji(2500)
           Mengangkatbarang = "Barangnya tidak berat"
        End If
    End Function

    Public Property thehobby() As String
        Get
            Return hobby
        End Get

        Set(ByVal Z As String)
            hobby = Z
        End Set
    End Property
End Class
```

3. Klik Form bernama **Form1** pada **Solution Explorer**. Kemudian klik menu **View** > **Code** pada Menu Bar untuk

membuka jendela kode program **Form1**. Lalu ubahlah kode program yang ada dan tambahkan sebuah Sub bernama **pesangaji** dalam jendela kode program **Form1**, sehingga menjadi seperti berikut:

```
Public Class Form1

    Dim WithEvents adit As Manusia
    Dim WithEvents gofo As Manusia

Public Sub pesangaji(ByVal agkgj As Integer) Handles adit.digaji

        Dim pesan As String
        pesan = "Objek ini digaji sebesar" + CStr(agkgj)

        MessageBox.Show(pesan, _
        "dari Sub yang menangani Even digaji", _
         MessageBoxButtons.OK, _
         MessageBoxIcon.Information)
End Sub


    Private Sub Btn adit Click(ByVal sender As Object,
    ByVal e As System.EventArgs) Handles Btn_adit.Click


        adit = New Manusia("Aditya Nugroho")

        adit.tinggibadan = 170
        adit.thehobby = "main game"

        TextBox1.Text = adit.tinggibadan
        TextBox2.Text = adit.thehobby

        ' Ketika Metoda bernama Mengangkatbarang
        ' dikerjakan pada Objek Adit, maka secara
        ' otomatis Event bernama digaji akan dibangkitkan
        Dim pesan = adit.Mengangkatbarang(25)

        MessageBox.Show(pesan, _
         "Hasil dari metoda Mengangkatbarang", _
         MessageBoxButtons.OK, _
         MessageBoxIcon.Information)
    End Sub

    Private Sub Btn gofo Click(ByVal sender As Object,
    ByVal e As System.EventArgs) Handles Btn_gofo.Click
```

```
        gofo = New Manusia("Abdolgofo")

        gofo.tinggibadan = 165
        gofo.thehobby = "membaca"
        TextBox1.Text = gofo.tinggibadan
        TextBox2.Text = gofo.thehobby

        Dim pesan = gofo.Mengangkatbarang(150)

        MessageBox.Show(pesan, _
         "Hasil dari metoda Mengangkatbarang", _
         MessageBoxButtons.OK, _
         MessageBoxIcon.Information)
    End Sub
End Class
```

4. Klik menu **Build** > **Build proyekkeenambelas** pada menu bar untuk mengompilasi program aplikasi yang baru saja Anda ubah.
5. Setelah proses kompilasi selesai, klik menu **Debug** > **Start Debugging** pada menu bar untuk mencoba menjalankan program aplikasi Anda. Klik dua buah tombol yang berada pada **Form1**.
6. Tutuplah aplikasi Anda dengan mengklik menu **Debug** > **Stop Debugging**.
7. Simpan seluruh Form dan Project.

8.8 Inheritance

Selain konsep enkapsulasi, keuntungan lain dari *object-oriented programming* adalah kode program yang bersifat *reusable*. Sehingga waktu pembuatan program dapat dikurangi, serta dapat menghindari berbagai kesalahan yang tidak perlu. Visual Basic 2005 memperkenalkan konsep *reusable* ini dalam sebuah proses yang disebut *inheritance* (pewarisan).

Pada Subbab 8.4, kita pernah membuat sebuah Class bernama **Manusia**. Dalam Class **Manusia** tersebut terdapat Field bernama **tinggibadan** dan **hobby**. Sekarang bagaimana jika ternyata ada **Manusia** yang bekerja sebagai **Pegawai** dan ada yang sebagai

Mahasiswa. Seorang **Pegawai** tentu saja memiliki Nomor Induk Pegawai (**NIP**) sedangkan seorang **Mahasiswa** tentu saja tidak memiliki NIP, tapi memiliki Nomor Induk Mahasiswa (**NIM**). Lalu bagaimana mengakomodasi hal tersebut? Apakah sebaiknya kita tambahkan saja sebuah Field baru bernama **NIP**? Tapi, tentu saja ini bukan solusi yang terbaik, karena jika ternyata objeknya adalah seorang **Mahasiswa** tentu saja dia tidak memiliki **NIP** melainkan memiliki **NIM**.

Ilustrasi yang lain misalnya sebagai berikut. Seorang **Pegawai** memiliki jabatan, sedangkan seorang mahasiswa tidak. Lalu apakah sebaiknya kita tambahkan saja sebuah Field baru dalam Class **Manusia**, yaitu Field **jabatan**? Tapi tentu saja hal tersebut bukan pilihan yang terbaik, karena jika ternyata dia adalah seorang **Mahasiswa**, dia tidak akan membutuhkan Field **jabatan**. Atau ilustrasi yang lain lagi. Seorang Mahasiswa mungkin memerlukan Metode bernama **MengisiKRS**, sedangkan seorang **Pegawai** tidak memerlukannya. Lalu apakah kita tambahkan saja sebuah Metode baru bernama **MengisiKRS** ke dalam Class **Manusia**? Tentu saja hal ini bukan pula jawaban yang terbaik. Lalu sebaiknya bagaimana?

Solusi yang paling tepat dari beberapa ilustrasi di atas adalah Inheritance (Pewarisan). Dengan pewarisan, kita dapat membuat Class baru yang mewarisi sifat dari Class lainnya. Class baru tersebut lebih spesifik dari Class yang diwarisinya. Class baru tersebut disebut sebagai turunan (*child*) dari Class induk yang diwarisinya (*parent*). *Child* Class mewarisi seluruh Field, Properti, Metode, dan Event yang dimiliki oleh *Parent* Class. *Child* Class dapat mengakses Field, Properti, Metode, dan Event yang dimiliki oleh *parent*-nya, asalkan tidak bersifat Private. *Parent* Class disebut juga sebagai *Base Class*, sedangkan *Child* Class disebut juga sebagai *SubClass*.

Oleh karena itu, untuk mengakomodasi beberapa ilustrasi di atas, dapat kita buat dua buah Class baru yang lebih spesifik, yaitu Class **Pegawai** dan Class **Mahasiswa**. Kedua Class tersebut akan mewarisi seluruh isi Class **Manusia**. Selain itu kita dapat menambahkan hal-hal yang lebih spesifik ke dalam kedua Class tersebut.

Kita dapat menambahkan dua buah Field baru ke dalam Class **Pegawai**, yaitu **NIP** dan **jabatan**. Kita dapat menambahkan satu buah Field baru ke dalam Class **Mahasiswa**, yaitu **NIM**. Kita juga dapat menambahkan sebuah Metoda baru ke dalam Class **Mahasiswa** yaitu **MengisiKRS**, sedangkan pada Class **Pegawai** tidak perlu ditambahkan.

Seandainya ada sebuah objek bernama **Adit** yang merupakan hasil instantiasi dari Class **Pegawai**. **Adit** akan memiliki dua buah field, yaitu **NIP** dan **jabatan**. Namun, karena Class **Pegawai** adalah turunan dari Class **Manusia**, maka Adit juga secara otomatis akan memiliki Field **tinggibadan**, dan **hobby**.

Untuk membuat sebuah *Child* Class, tambahkan kata kunci **Inherits** ketika Class tersebut dideklarasikan, diikuti dengan nama *Parent* Class yang akan diwarisi. Contoh:

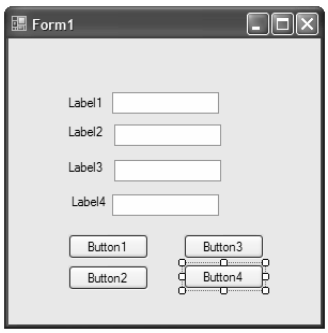
```
Public Class Pegawai
        Inherits Manusia


End Class
```

Pada kode program di atas, dibuat sebuah Class baru bernama **Pegawai** yang mewarisi (*inherit*) Class **Manusia**.

Lakukanlah langkah-langkah berikut untuk mencoba menggunakan Inheritance (Pewarisan) dalam kode program Anda:

1. Buatlah sebuah Project baru bernama **proyekketujuhbelas**.
2. Tambahkan empat buah **Button** serta empat buah **Label** dan **TextBox** ke dalam Form sehingga seperti berikut.



3. Atur properti masing-masing **Label**, **TextBox**, dan **Button** tersebut seperti berikut.

Kontrol	Properti	Nilai
Label1	Text	Label1:
Label2	Text	Jabatan:
Label3	Text	Tinggi Badan:
Label4	Text	Hobby:
TextBox1	(Name) Text	TextBox1 TextBox1
TextBox2	(Name) Text	TextBox2 TextBox2
TextBox3	(Name) Text	TextBox3 TextBox3
TextBox4	(Name) Text	TextBox4 TextBox4
Button1	(Name) Text	Btn_adit1 Buat Objek Adit, lalu tampilkan isi seluruh Field miliknya ke TextBox dan jalankan Metode MengisiKRS
Button2	(Name) Text	Btn_gofo1 Buat Objek Gofo, lalu tampilkan isi seluruh Field miliknya ke TextBox
Button3	(Name) Text	Btn_adit2 Buat Objek Adit, lalu jalankan Metode Mengangkatbarang dengan input 25
Button4	(Name) Text	Btn_gofo2 Buat Objek Gofo, lalu jalankan

		Metode Mengangkatbarang dengan input 150

4. Aturlah kembali posisi **Label**, **TextBox**, dan **Button**-nya sehingga menjadi seperti berikut.

5. Klik menu **Project** > **Add Class...** untuk menambahkan Class baru. Beri nama Class tersebut dengan nama **Manusia**.
6. Klik ganda nama Class **Manusia** pada **Solution Explorer** untuk membuka jendela kode program Class tersebut. Lalu lengkapi dengan kode program berikut:

```
Public Class Manusia

  Public tinggibadan As Integer
  Private hobby As String
```

```
  Public Function Mengangkatbarang(ByVal X As String) As String
       If X >= 100
              Mengangkatbarang="Barangnya sangat berat"
       Else
              Mengangkatbarang="Barangnya tidak berat"
       End If
  End Function

  Public Property thehobby() As String
       Get
         Return hobby
       End get

       Set(ByVal Z As String)
         hobby=Z
       End Set
  End Property

End Class
```

7. Klik menu **Project** > **Add Class...** untuk menambahkan Class baru. Beri nama Class tersebut dengan nama **Pegawai**.
8. Klik ganda nama Class **Pegawai** pada **Solution Explorer** untuk membuka jendela kode program Class tersebut. Lalu lengkapi dengan kode program berikut:

```
Public Class Pegawai
    Inherits Manusia

    Public NIP As String
    Public jabatan As String


End Class
```

9. Klik menu **Project** > **Add Class...** untuk menambahkan Class baru. Beri nama Class tersebut dengan nama **Mahasiswa**.
10. Klik ganda nama Class **Mahasiswa** pada **Solution Explorer** untuk membuka jendela kode program Class tersebut. Lalu lengkapi dengan kode program berikut:

```
Public Class Mahasiswa
    Inherits Manusia

    Public NIM As String

    Public Sub MengisiKRS()
        MessageBox.Show("Objek ini mengisi KRS", _
        "Pesan", _
         MessageBoxButtons.OK, _
         MessageBoxIcon.Information)
    End Sub

End Class
```

11. Klik Form bernama **Form1** pada **Solution Explorer**. Kemudian klik menu **View** > **Code** pada Menu Bar untuk membuka jendela kode program **Form1**. Lalu lengkapi kode program yang ada menjadi seperti berikut. Kode program Anda ditulis di bagian Event **Click** dari objek **Btn_adit1** dan **Btn_gofo1** serta objek **Btn_adit2** dan **Btn_gofo2**:

```
Public Class Form1

    Private Sub Btn_adit1_Click(ByVal sender As
    System.Object, ByVal e As System.EventArgs) Handles
    Btn_adit1.Click

        Dim adit As Mahasiswa
        adit = New Mahasiswa()

        adit.NIM = "G64100039"
        adit.tinggibadan = 170
        adit.thehobby = "main game"

        Label1.Text = "NIM :"
        TextBox1.Text = adit.NIM
    TextBox2.Text = "Objek ini tidak punya Field jabatan"
        TextBox3.Text = adit.tinggibadan
        TextBox4.Text = adit.thehobby

        adit.MengisiKRS()

        adit = Nothing

    End Sub
```

```
Private Sub Btn adit2 Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
Btn_adit2.Click

    Dim adit As Mahasiswa
    adit = New Mahasiswa()

    Dim pesan As String
    pesan = adit.Mengangkatbarang(25)

    MessageBox.Show(pesan, _
     "Hasil dari metoda Mengangkatbarang", _
     MessageBoxButtons.OK, _
     MessageBoxIcon.Information)

    adit = Nothing

End Sub

Private Sub Btn gofo1 Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
Btn_gofo1.Click

    Dim gofo As Pegawai
    gofo = New Pegawai()

    gofo.NIP = "132132132"
    gofo.jabatan = "CEO"
    gofo.tinggibadan = 165
    gofo.thehobby = "membaca"

    Label1.Text = "NIP :"
    TextBox1.Text = gofo.NIP
    TextBox2.Text = gofo.jabatan
    TextBox3.Text = gofo.tinggibadan
    TextBox4.Text = gofo.thehobby

    gofo = Nothing

End Sub

Private Sub Btn gofo2 Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
Btn_gofo2.Click

    Dim gofo As Pegawai
    gofo = New Pegawai()
```

```
        Dim pesan As String
        pesan = gofo.Mengangkatbarang(150)

        MessageBox.Show(pesan, _
         "Hasil dari metoda Mengangkatbarang", _
         MessageBoxButtons.OK, _
         MessageBoxIcon.Information)

        gofo = Nothing

    End Sub
End Class
```

12. Klik menu **Build** > **Build proyekketujuhbelas** pada menu bar untuk mengompilasi program aplikasi yang baru saja Anda buat.
13. Setelah proses kompilasi selesai, klik menu **Debug** > **Start Debugging** pada menu bar untuk mencoba menjalankan program aplikasi Anda. Klik empat buah tombol yang berada pada **Form1**.
14. Tutuplah aplikasi Anda dengan mengklik menu **Debug** > **Stop Debugging**.
15. Simpan seluruh Form dan Project.