

---

## MODUL VI

### APLIKASI WINDOWS FORM

#### A. TUJUAN

- ✓ Memahami aplikasi Windows Form dan komponen-komponen didalamnya.
- ✓ Mampu menggunakan kontrol-kontrol dan komponen-komponen Windows Form.
- ✓ Mampu membangun *Graphical User Interface* (GUI) di lingkungan .NET Framework.

#### B. PETUNJUK

- Awali setiap aktivitas dengan doa, semoga berkah dan mendapat kemudahan.
- Pahami tujuan, dasar teori, dan latihan-latihan praktikum dengan baik dan benar.
- Kerjakan tugas-tugas praktikum dengan baik, sabar, dan jujur.
- Tanyakan kepada asisten/dosen apabila ada hal-hal yang kurang jelas.

#### C. DASAR TEORI

##### 1. Windows Form

Windows Form adalah bagian dari .NET Framework dan Visual Studio .NET, yang menggunakan teknologi-teknologi baru meliputi *common application framework*, *managed execution environment*, *integrated security*, dan desain berorientasi objek.

##### 2. Form

Form merupakan bagian utama dalam pengembangan aplikasi Windows. Form adalah elemen dasar antarmuka *user* yang diinstansiasi dari kelas `Form`—yang tersimpan di namespace `System.Windows.Forms`. Di Visual Studio .NET, form direpresentasikan sebagai window yang digunakan oleh aplikasi Windows.

##### 3. Kontrol dan Komponen Windows Form

Kontrol-kontrol Windows Form adalah komponen-komponen *reusable* yang membungkus fungsionalitas antarmuka *user*. Windows Form menyediakan kontrol-kontrol dan komponen-komponen yang berfungsi

untuk menambah fungsionalitas aplikasi. Pada dasarnya, kontrol dan komponen adalah dua objek yang berbeda; di mana kontrol memiliki tampilan visual, sedangkan komponen tidak.

## D. LATIHAN

### 1. Form

Begitu form diciptakan, ia akan mengalami beberapa rangkaian event yang mendeskripsikan siklus hidupnya. Untuk memahami bagaimana siklus hidup suatu form, kita dapat memanfaatkan event-event serta method-method utama yang tersedia.

1. Buat aplikasi Windows (dengan memilih template **Windows Application**)
2. Tambahkan event-event berikut dan lengkapi kode programnya.

```
Private Sub Form1_Load(ByVal sender As Object, _
    ByVal e As System.EventArgs) Handles Me.Load
    Console.WriteLine("Form di-load")
End Sub

Private Sub Form1_Activated(ByVal sender As Object, _
    ByVal e As System.EventArgs) Handles Me.Activated
    Console.WriteLine("Form diaktivasi")
End Sub

Private Sub Form1_FormClosing(ByVal sender As Object, _
    ByVal e As System.Windows.Forms.FormClosingEventArgs) _
    Handles Me.FormClosing
    Console.WriteLine("Form sedang dalam penutupan")
End Sub

Private Sub Form1_Deactivate(ByVal sender As Object,
    ByVal e As System.EventArgs) Handles Me.Deactivate
    Console.WriteLine("Form dideaktivasi")
End Sub

Private Sub Form1_Disposed(ByVal sender As Object, _
    ByVal e As System.EventArgs) Handles Me.Disposed
    Console.WriteLine("Form di-dispose (dihapus dari memori)")
End Sub
```

3. Jalankan aplikasi dan amati hasilnya di window **Output**.

### Menciptakan dan Menutup Form

Pada hakekatnya, form adalah sebuah objek yang dihasilkan dari kelas `Form`. Dengan demikian, kita bisa menciptakan dan menghapus melalui variabel referensi objek.

1. Buat aplikasi Windows.
2. Deklarasikan variabel bertipe `Form`.

```
Private frm As Form
```

3. Tambahkan sebuah button untuk menciptakan dan menampilkan objek form.

```
Private Sub Button1_Click(ByVal sender As System.Object, _
    ByVal e As System.EventArgs) Handles Button1.Click
    frm = New Form
    frm.Show()
End Sub
```

4. Tambahkan lagi sebuah button untuk menutup form aktif.

```
Private Sub Button2_Click(ByVal sender As System.Object, _
    ByVal e As System.EventArgs) Handles Button2.Click
    frm.Close()
End Sub
```

5. Jalankan aplikasi dan amati hasilnya.

### **Konfirmasi Penutupan Form**

Pada saat form ditutup, maka event-event penutupan akan dijalankan. Keadaan ini memungkinkan kita untuk memberikan konfirmasi penutupan form.

1. Tambahkan event `FormClosing` pada form, kemudian lengkapi kode konfirmasi penutupan.

```
Private Sub Form1_FormClosing(ByVal sender As Object, _
    ByVal e As System.Windows.Forms.FormClosingEventArgs) _
    Handles Me.FormClosing

    ' Konfirmasi penutupan form
    If MessageBox.Show("Are you sure to exit?", "Exit", _
        MessageBoxButtons.YesNo, MessageBoxIcon.Question) = _
        Windows.Forms.DialogResult.No Then
        e.Cancel = True
    End If
End Sub
```

2. Jalankan aplikasi dan coba tutup form melalui tombol X (di pojok kanan atas).

## **2. ComboBox**

Kontrol `ComboBox`—yang direpresentasikan melalui objek `ComboBox`—merupakan kombinasi dari text box dan list box. Penggunaan kontrol ini diperlihatkan sebagai berikut:

1. Tambahkan item **Windows Form** di project.
2. Tambahkan sebuah kontrol `ComboBox`.
3. Isikan item combo box melalui event `Load` form dan atur behavior-nya.

```
Private Sub Form2_Load(ByVal sender As System.Object, _
    ByVal e As System.EventArgs) Handles MyBase.Load
    ' Menambahkan item
    Me.ComboBox1.Items.Add("Pria")
    Me.ComboBox1.Items.Add("Wanita")
End Sub
```

```

' Menetapkan style untuk item read-only
Me.ComboBox1.DropDownStyle = ComboBoxStyle.DropDownList
' Menetapkan item default berdasar indeks
Me.ComboBox1.SelectedIndex = 0
End Sub

```

4. Tambahkan sebuah button dan isikan kode di event Click untuk mendapatkan nilai combo box.

```

Private Sub Button1_Click(ByVal sender As System.Object, _
ByVal e As System.EventArgs) Handles Button1.Click
    Me.Text = Me.ComboBox1.Text
End Sub

```

5. Jalankan aplikasi.
6. Untuk mengidentifikasi pilihan secara langsung, tambahkan event SelectedIndexChanged milik ComboBox.

```

Private Sub ComboBox1_SelectedIndexChanged( _
ByVal sender As Object, _
ByVal e As System.EventArgs)
Handles ComboBox1.SelectedIndexChanged
    Me.Text = Me.ComboBox1.Text
End Sub

```

### 3. CheckBox

CheckBox merupakan kontrol yang menyediakan opsi—misalnya *Yes/No* atau *True/False*—untuk dipilih.

1. Tambahkan item **Windows Form** di project.
2. Tambahkan sebuah kontrol CheckBox.
3. Tambahkan sebuah button dan isikan kode di event Click untuk mengidentifikasi dan mendapatkan nilai check box.

```

Private Sub Button1_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button1.Click
    If CheckBox1.Checked = True Then
        Me.Text = "Checked"
    Else
        Me.Text = "Unchecked"
    End If
End Sub

```

4. Jalankan aplikasi.
5. Untuk mengidentifikasi pilihan secara langsung, tambahkan event CheckedChanged milik CheckBox.

```

Private Sub CheckBox1_CheckedChanged(ByVal sender As Object, _
ByVal e As System.EventArgs) Handles CheckBox1.CheckedChanged
    If CheckBox1.Checked = True Then
        Me.Text = "Yes"
    Else
        Me.Text = "No"
    End If
End Sub

```

#### 4. RadioButton

Seperti halnya check box, radio button digunakan untuk memilih/mengosongkan opsi. Pada umumnya, radio button digunakan bersama-sama di dalam sebuah grup.

1. Tambahkan item **Windows Form** di project.
2. Tambahkan sebuah kontrol `GroupBox`.
3. Tambahkan dua buah kontrol `RadioButton` ke dalam group box. ubah masing-masing properti `Text` radio button menjadi `Pria` dan `Wanita`.
4. Tambahkan sebuah button dan isikan kode di event `Click` untuk mendapatkan status radio button.

```
Private Sub Button1_Click(ByVal sender As System.Object, _
    ByVal e As System.EventArgs) Handles Button1.Click
    If RadioButton1.Checked = True Then
        Me.Text = "RadioButton1 Selected"
    Else
        Me.Text = "RadioButton2 Selected"
    End If
End Sub
```

5. Jalankan aplikasi.
6. Untuk mengidentifikasi pilihan secara langsung, tambahkan event `CheckedChanged` milik `RadioButton`.

```
Private Sub RadioButton1_CheckedChanged( _
    ByVal sender As Object, ByVal e As System.EventArgs) _
    Handles RadioButton1.CheckedChanged
    If RadioButton1.Checked = True Then
        Me.Text = "RadioButton1 Selected"
    End If
End Sub

Private Sub RadioButton2_CheckedChanged( _
    ByVal sender As Object, ByVal e As System.EventArgs) _
    Handles RadioButton2.CheckedChanged
    If RadioButton2.Checked = True Then
        Me.Text = "RadioButton2 Selected"
    End If
End Sub
```

#### 5. Kotak Dialog

Kelas `MessageBox` mendefinisikan method statis `Show()` yang di-*overload* guna menyediakan kotak dialog yang variatif. Sintaks umum dari method `Show()` diperlihatkan sebagai berikut:

```
Show(teks [, judul] [, button] [, ikon])
```

Dengan demikian, bentuk kotak dialog yang paling sederhana direpresentasikan melalui kode program berikut:

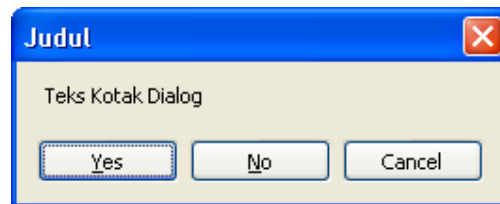
```
MessageBox.Show("Teks Kotak Dialog")
```



*Gambar 1 Kotak dialog sederhana*

Bentuk-bentuk kotak dialog lainnya diperlihatkan sebagai berikut:

```
MessageBox.Show("Teks Kotak Dialog", "Judul", _
MessageBoxButtons.YesNoCancel)
```



*Gambar 2 Kotak dialog Yes/No/Cancel*

```
MessageBox.Show("Teks Kotak Dialog", "Judul", _
MessageBoxButtons.AbortRetryIgnore, MessageBoxIcon.Information)
```



*Gambar 3 Menggunakan Ikon kotak dialog*

Untuk mendeteksi pilihan user, kita memanfaatkan nilai kembalian dari method Show(), yakni enumerasi DialogResult.

```
Dim dlg As DialogResult
dlg = MessageBox.Show("Teks", "Judul", MessageBoxButtons.YesNoCancel)

Select Case dlg
    Case Windows.Forms.DialogResult.Yes
        Console.WriteLine("Yes clicked")
    Case Windows.Forms.DialogResult.No
        Console.WriteLine("No clicked")
    Case Windows.Forms.DialogResult.Cancel
        Console.WriteLine("Cancel clicked")
End Select
```

## 6. Timer

Komponen ini akan *trigger* suatu event pada interval waktu yang dispesifikasikan melalui properti `Interval`. Sebagai contoh, kita dapat memanfaatkan `Timer` untuk menciptakan stopwatch.

1. Tambahkan item **Windows Form** di project.
2. Tambahkan sebuah kontrol `Label`, dua buah `Button`, dan sebuah `Timer`.
3. Misalkan di sini kita tetapkan bahwa nilai properti `Interval` dari `Timer` adalah **100** (dalam satuan milidetik).
4. Lengkapi kode programnya seperti berikut:

```
Private Sub btnStart_Click(ByVal sender As System.Object, _
    ByVal e As System.EventArgs) Handles btnStart.Click
    Me.Label1.Text = "0"
    Me.Timer1.Enabled = True
End Sub


Private Sub btnStop_Click(ByVal sender As System.Object, _
    ByVal e As System.EventArgs) Handles btnStop.Click
    Me.Timer1.Enabled = False
End Sub

' Ini dijalankan setiap interval waktu
Private Sub Timer1_Tick(ByVal sender As Object, _
    ByVal e As System.EventArgs) Handles Timer1.Tick
    Me.Label1.Text = CStr((Cdbl(Me.Label1.Text) + 0.1))
End Sub
```

5. Jalankan aplikasi.



Gambar 4 Simple stopwatch

 Penggunaan kontrol-kontrol dan komponen-komponen Windows Form lainnya bisa dicoba sendiri.

## 7. Validasi Data

Ada beragam event yang bisa digunakan untuk mengimplementasikan validasi data pada text box, salah satunya adalah event `Leave`. Event ini akan di-trigger manakala fokus (kursor) meninggalkan kontrol. Contoh penggunaan event `Leave` diperlihatkan sebagai berikut:

1. Tambahkan item **Windows Form** di project.
2. Tambahkan sebuah kontrol Label, TextBox, dan Button. Ubah nilai properti Name text box menjadi **txtNama**.
3. Tambahkan event Leave di text box dan lengkapi kode programnya seperti berikut:

```
Private Sub txtNama_Leave(ByVal sender As Object, _
    ByVal e As System.EventArgs) Handles txtNama.Leave
    If (Me.txtNama.Text.Trim.Length = 0) Then
        Me.txtNama.Focus()
    End If
End Sub
```

4. Jalankan aplikasi dan amati hasilnya ketika text box kosong atau ketika berisi teks.

Event-event text box lainnya—seperti KeyDown, KeyPress, dan Validating—bisa dicoba sendiri.

Dalam upaya memudahkan user mendapatkan informasi jika sewaktu-waktu terjadi kesalahan, kita bisa menyediakan fitur spesifik. Salah satu fitur yang efisien adalah memanfaatkan komponen `ErrorProvider`.

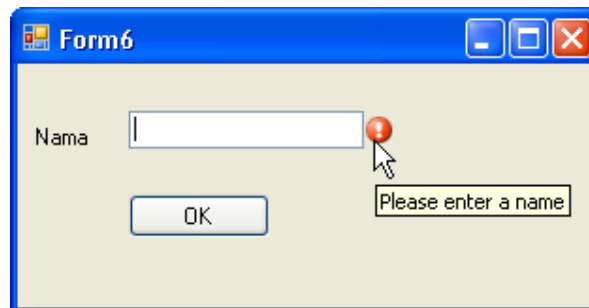
1. Masih melanjutkan project di form sebelumnya, tambahkan sebuah komponen `ErrorProvider`.
2. Modifikasi kode di event Leave seperti berikut:

```
Private Sub txtNama_Leave(ByVal sender As Object, _
    ByVal e As System.EventArgs) Handles txtNama.Leave
    Dim strErr As String = ""

    If (Me.txtNama.Text.Trim.Length = 0) Then
        strErr = "Please enter a name"
        Me.txtNama.Focus()
    End If

    ' Set pesan kesalahan pada text box
    Me.ErrorProvider1.SetError(Me.txtNama, strErr)
End Sub
```

3. Jalankan aplikasi dan amati hasilnya.



Gambar 5 Menggunakan `ErrorProvider`

## 8. Custom Control

Terlepas dari kontrol-kontrol dan komponen-komponen yang tersedia, kita juga bisa menciptakan sendiri kontrol yang spesifik. Salah satu pendekatan paling praktis untuk melakukan hal ini adalah dengan pewarisan (ingat kembali materi pewarisan ☺).

Sekadar contoh sederhana, kita akan memperluas kelas `Label` untuk menghasilkan kelas `RedLabel`.

1. Tambahkan item kelas baru dan simpan dengan nama **RedLabel.vb**.
2. Lengkapi pendefinisian kelasnya seperti berikut:

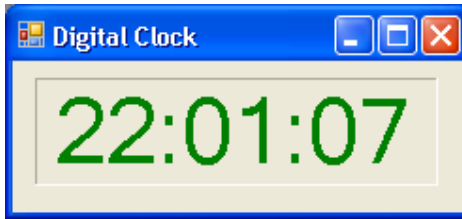
```
Public Class RedLabel
    Inherits Label

    Sub New()
        ' Misal men-set warna label
        Me.ForeColor = Color.Red
    End Sub
End Class
```

3. Untuk memudahkan penggunaan kontrol, eksekusi project. Langkah ini akan mengakibatkan kelas `RedLabel` direpresentasikan dalam bentuk kontrol di **Toolbox**.
4. Sekarang kita bisa menggunakan kontrol `RedLabel` layaknya kontrol-kontrol lainnya.

**E. TUGAS PRAKTIKUM**

1. Buat jam digital dengan memanfaatkan komponen `Timer`.



2. Buat aplikasi voting sederhana dengan melibatkan dua buah pilihan (misal seperti Gambar di bawah). Gunakan variabel untuk menghitung pilihan dari masing-masing item.



3. Buat custom control bernama `NumericTextBox`. Kontrol ini hanya dapat menerima masukan berupa bilangan. Buat juga kode untuk menguji fungsionalitasnya.