UNPRI

UNPRI PRESS

DASAR-DASAR PEMROGRAMAN VISUAL

Christnatalis HS

Marlince Novita Karoseri Nababan

PENERBIT UNPRI PRESS-2023
UNIVERSITAS PRIMA INDONESIA

DASAR-DASAR PEMROGRAMAN VISUAL

PENULIS
Christnatalis HS, M.Kom
Marlince Novita Karoseri Nababan, M.Kom

EDITOR
Yonata Laia, M.Kom

PENERBIT
UNPRI PRESS
(Anggota IKAPI)

ISBN:

Alamat Redaksi
Kampus 2
Jl. Sampul No. 4 Medan

Hak Cipta dilindungi undang-undang
Dilarang memperbanyak sebagian atau seluruh isi buku ini dalam
bentuk dan cara apapun tanpa izin tertulis dari penerbit.

Kata Pengantar

Puji dan syukur kami panjatkan ke hadirat Tuhan Yang Maha Esa, yang atas berkat dan rahmat-Nya telah memberikan kami kesehatan dan kemampuan sehingga kami dapat menyelesaikan penyusunan buku dasar-dasar pemrograman visual ini. Terima kasih banyak juga kami ucapkan kepada para pimpinan dan rekan kerja di Universitas Prima Indonesia, khususnya di Fakultas Teknologi dan Ilmu Komputer yang telah banyak membantu kami dalam penyusunan buku ini.

Pemrograman visual merupakan salah satu cabang dari pemrograman dasar yang bertujuan untuk mengasah ketrampilan dan kemampuan programmer dalam membuat program yang siap pakai untuk dunia nyata. Buku ini akan membahas dasar-dasar pemrograman visual seperti pemrograman event driven, penggunaan controls, pemanfaatan collections dalam pengolahan data, hingga membuat permainan sederhana seperti Tic Tac Toe. Setiap topik yang dibahas dalam buku ini akan disertai penjelasan lengkap sehingga diharapkan pembaca dapat memahaminya dengan jelas dan mampu menghasilkan program yang dapat berguna di lapangan kerja nyata.

Kami menyadari bahwa masih banyak kekurangan di dalam penyusunan buku ini, karena itu kritik dan saran dari para pembaca sangat kami harapkan agar kami dapat menghasilkan karya yang lebih baik lagi di masa mendatang.

Medan,

Penulis

Daftar Isi

Daftar Gambar

Bab 1. Pengenalan

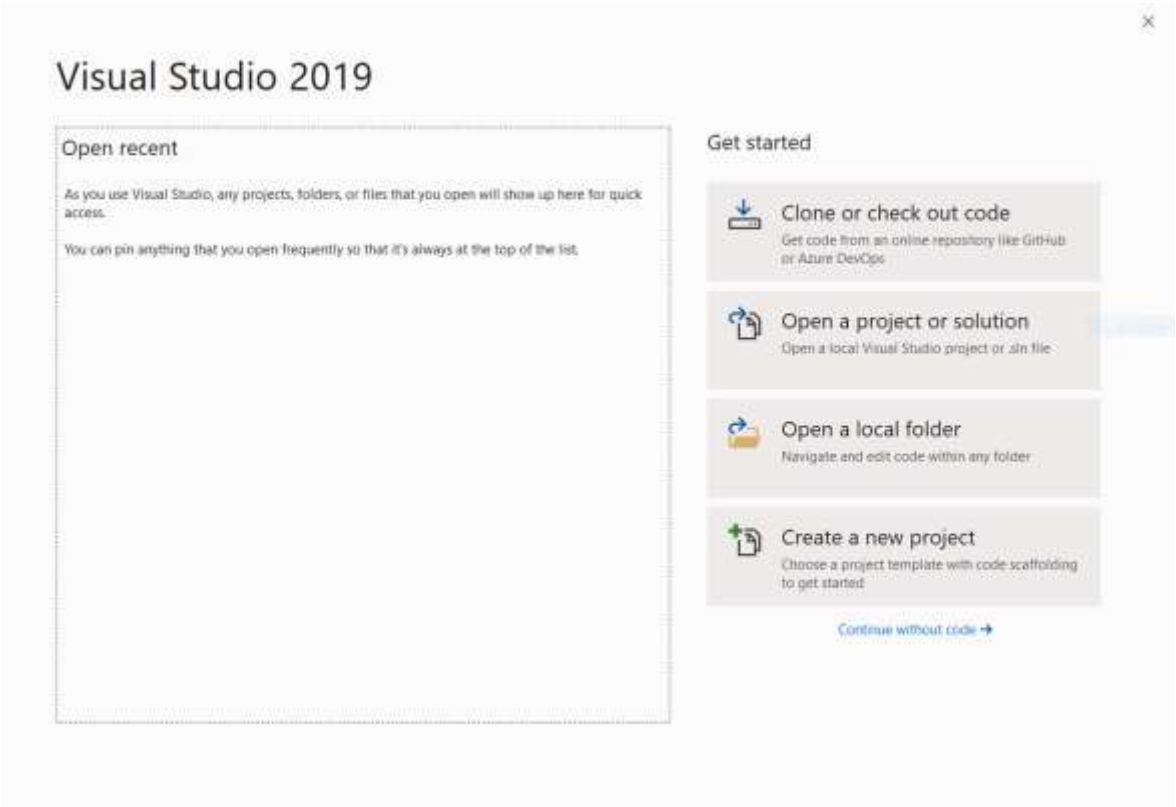
1.1. Pengenalan Interface Microsoft Visual Studio Community 2019.

Pemrograman visual berbeda dengan bahasa pemrograman yang berbasis teks seperti C/C++. Hal ini dikarenakan pada pemrograman visual kita tidak hanya terfokus pada code melainkan juga kepada desain interface (antar muka) dari aplikasi yang hendak kita hasilkan. Secara konsep dan logika dasar pemrograman visual sama halnya dengan pemrograman berbasis teks hanya saja berbeda sintaksnya. Selain itu pada pemrograman visual kita juga harus mengetahui kapan code hendak dieksekusi, hal ini dikarenakan berbeda dengan bahasa C/C++ di mana code akan dijalankan dari awal sampai akhir, di pemrograman visual code akan dijalan sesuai dengan "tindakan" yang kita lakukan terhadap "objek" pada aplikasi kita. Apabila anda sudah mempelajari Pemrograman Berorientasi Objek maka akan lebih mudah memahami hal ini, bila belum juga tidak masalah.

Untuk membuat program yang akan dipelajari di buku ini kita akan menggunakan aplikasi Microsoft Visual Studio Community 2019 yang dapat diunduh secara gratis melalui situs:
https://visualstudio.microsoft.com/vs/community/

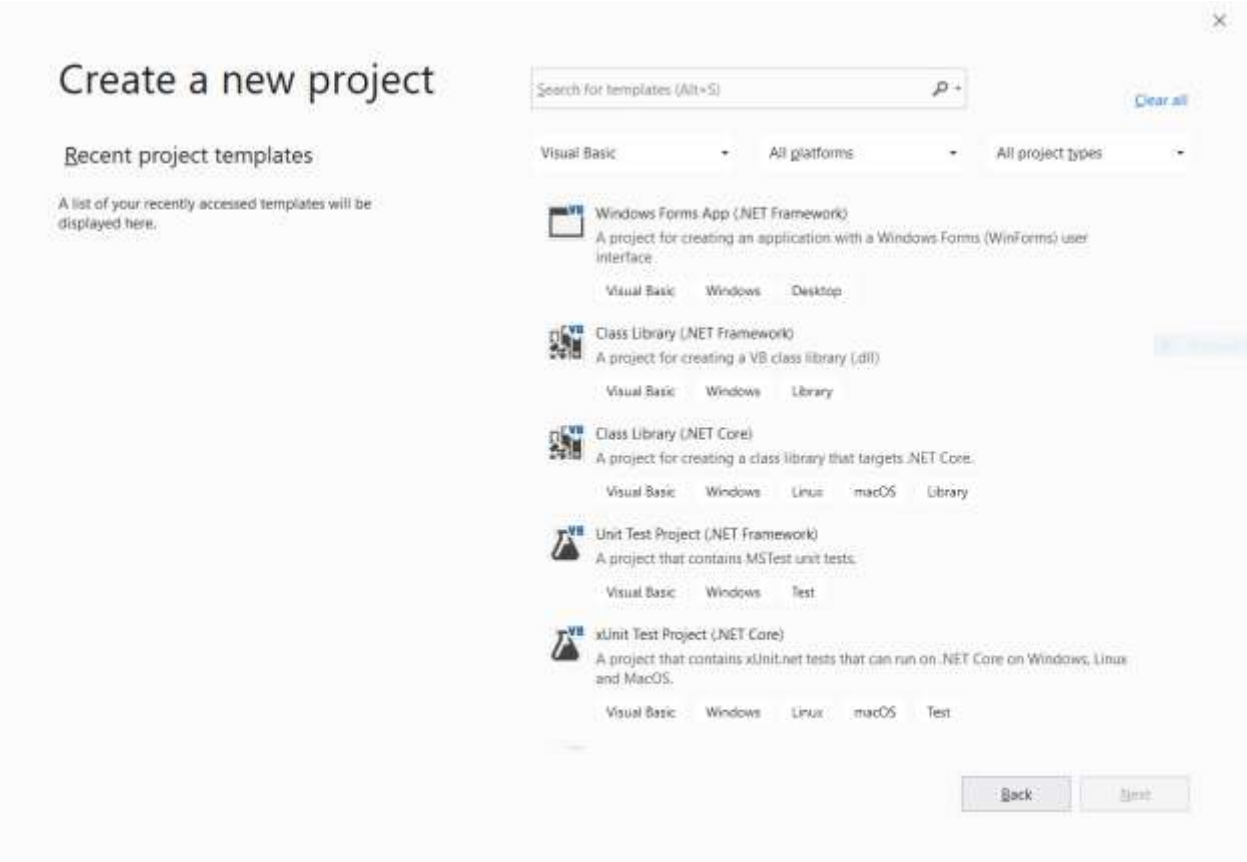
Saran: aplikasi sebaiknya diinstal pada harddisk SSD guna mempercepat proses loading dan compiling.

Setelah diinstall maka jalankan aplikasi Visual Studio 2019 yang akan memunculkan layar seperti berikut ini:



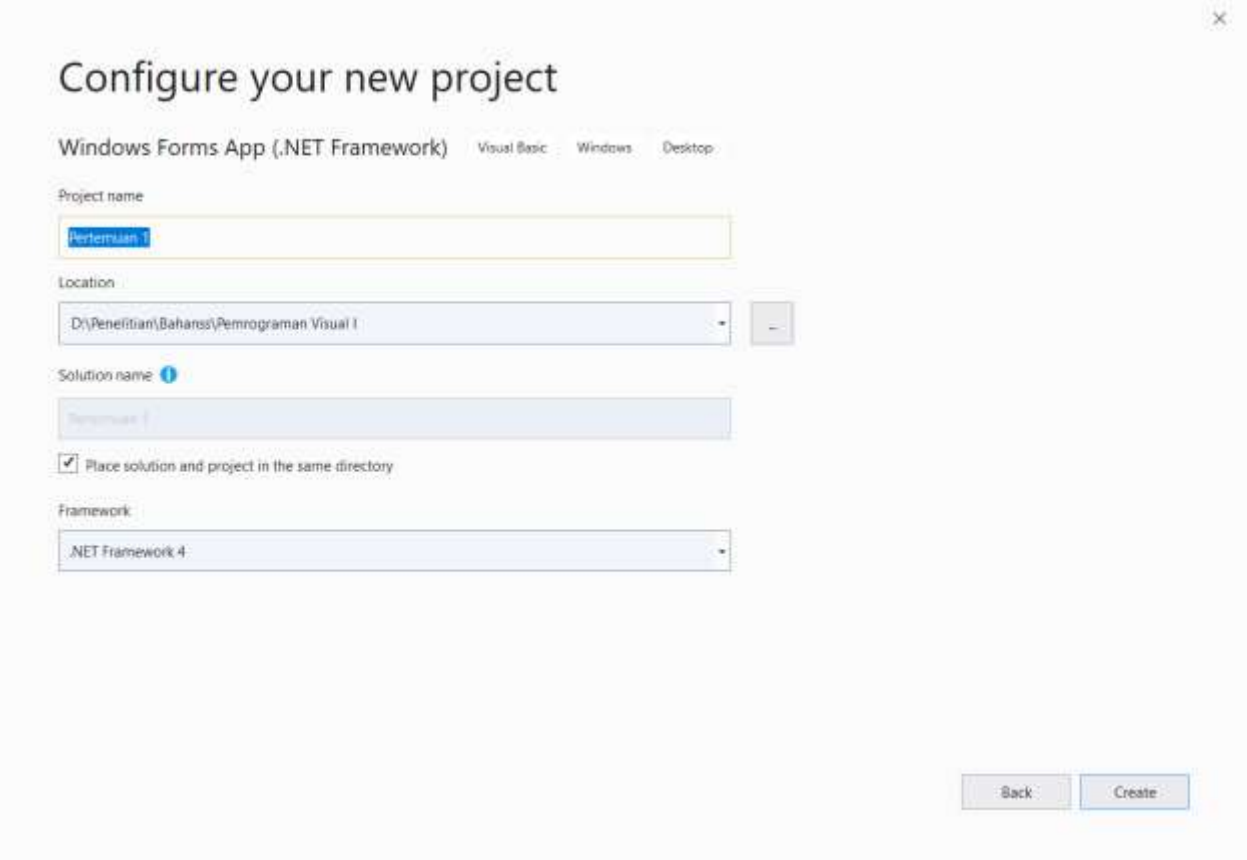
Gambar 1. Tampilan awal Microsoft Visual Studio 2019

Pilih Create a new project maka akan dibawa ke tampilan berikut:



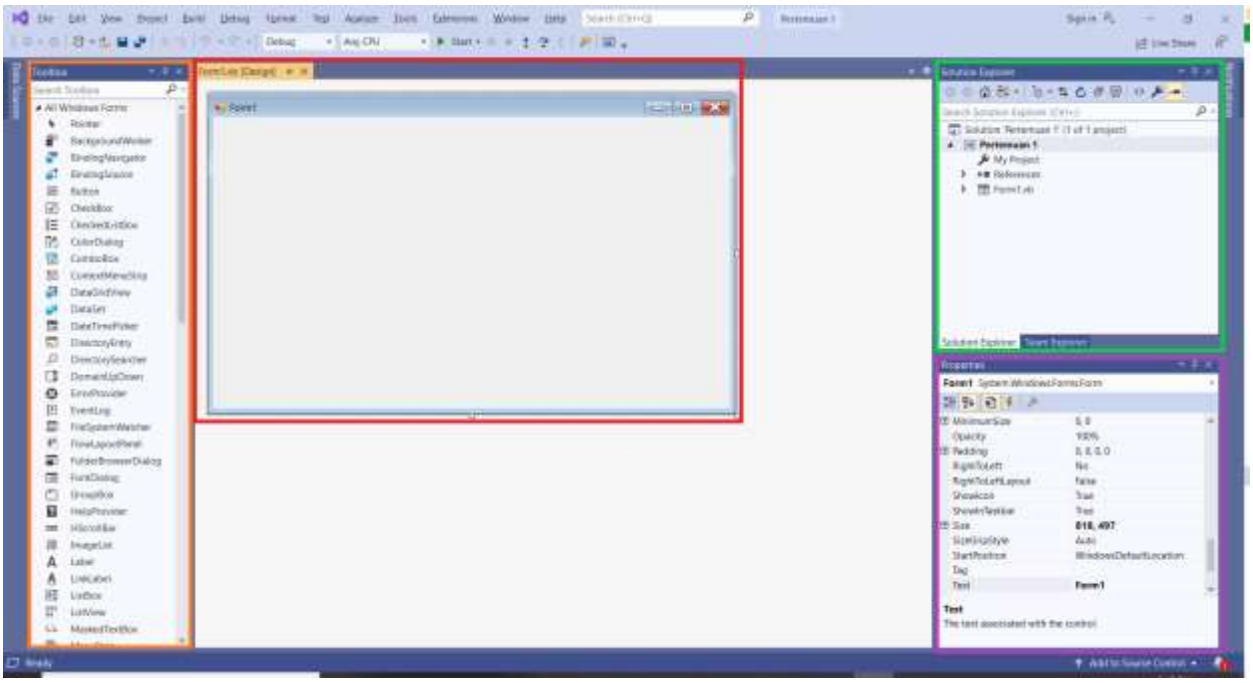
Gambar 2. Pilihan pembuatan project

Pilih Windows Forms App, apabila tidak ditemukan silahkan ketikkan “Windows Forms App” di bagian search for templates. Setelah itu klik tombol next.



Gambar 3. Konfigurasi project baru

Isikan nama project di project name, lokasi penyimpanan file project di location. Untuk solution name sebaiknya disamakan dengan nama project dan pilihan place solution and project in the same directory dicentang agar file-file yang dibutuhkan disimpan ke lokasi yang sama dan mudah untuk diakses. Untuk framework pilih .NET Framework 4. Apabila hendak membuat aplikasi untuk sistem operasi yang masih menggunakan framework di bawah versi 4, maka dapat dilakukan pemilihan dari sana.
Setelah selesai melakukan setting seperti di atas silahkan klik tombol Create.



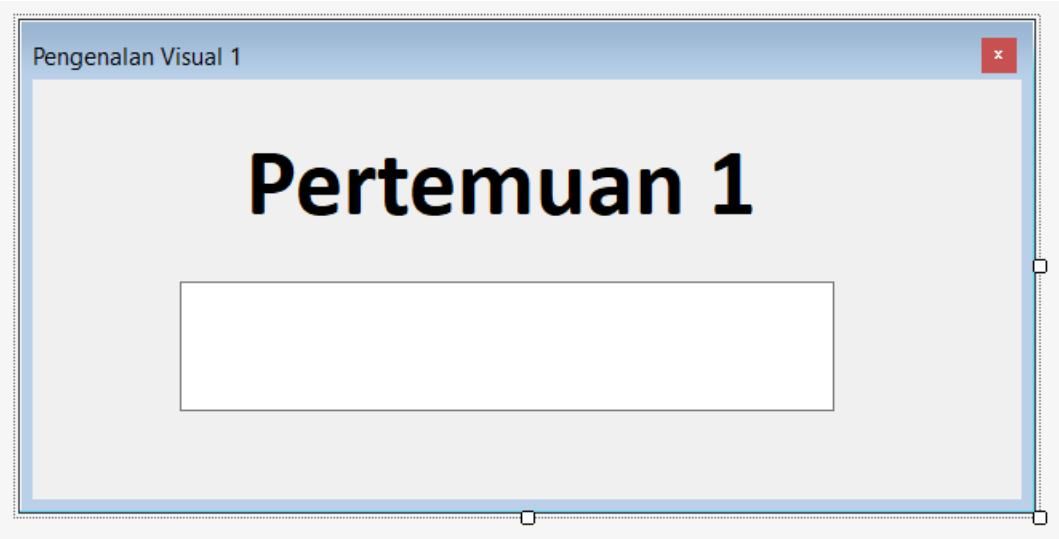
Gambar 4. Tampilan interface

Pada gambar 4. dapat dilihan tampilan interface yang mencakup **toolbox**, **design**, **solution explorer**, dan **properties**.

1. Toolbox berisikan objek yang dapat kita gunakan untuk "menghias" tampilan design aplikasi kita, cukup banyak objek yang terdapat pada toolbox tetapi sebagai dasar kita akan menggunakan beberapa objek yang umumnya digunakan saja.
2. Design merupakan tampilan dari aplikasi yang akan kita hasilkan, untuk menggunakan objek dari toolbox ke design dapat dilakukan dengan cara:
    a. Double click objek yang diinginkan lalu tekan dan tahan objek tersebut setelah muncul di design untuk dipindahkan ke posisi yang kita inginkan.
    b. Click and drag objek tersebut dari toolbox langsung ke design.
    c. Melakukan copy dan paste pada objek yang telah terdapat di design.
3. Solution explorer menampilkan semua file yang digunakan di dalam solution/ project ini. Pada contoh di atas terdapat Form 1.vb yang default akan dihasilkan apabila kita memilih Windows Forms App.
4. Properties menampilkan "sifat" atau "atribut" dari objek, setiap objek memiliki tampilan properties yang berbeda, untuk mengakses properties dari suatu objek cukup dilakukan click pada objek tersebut. Contoh pada gambar di atas menunjukkan properties dari Form 1. Ingat: form juga merupakan suatu objek.

Tips: untuk mengakses objek yang umum digunakan pada design klik tombol segitiga hitam untuk meminimizekan All Windows Forms pada toolbox dan pilih Common Controls.

Untuk program pertama silahkan design tampilan seperti gambar berikut ini:



Gambar 5. Tampilan design form Pengenalan Visual 1

Gunakan objek label untuk tulisan "Pertemuan 1" dan objek textbox untuk kotak putih yang dapat diisikan oleh user nantinya. Adapun properties yang perlu diubah adalah sebagai berikut:

Form:
FormBorderStyle : FixedToolWindow
Text : Pengenalan Visual 1
Size : 636, 309

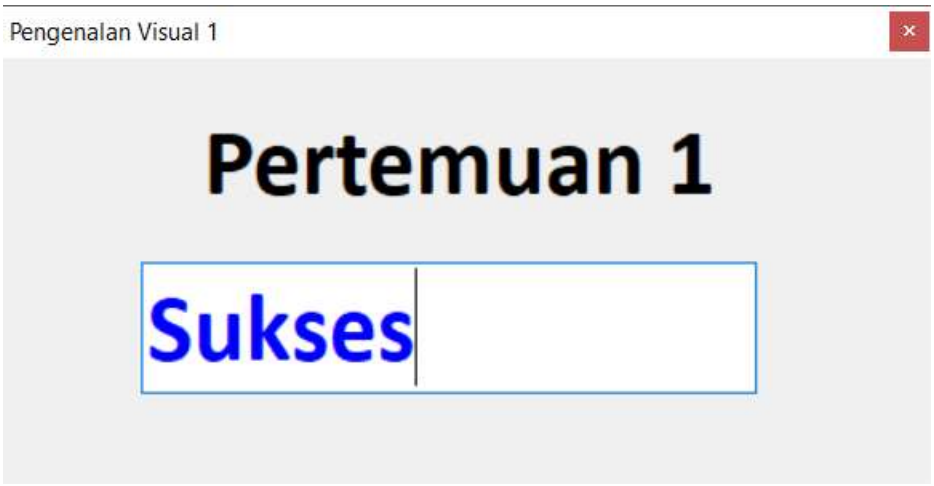
Label:
Font : Calibri, 36pt, style=Bold
Text : Pertemuan 1
Size : 351, 73

Textbox:
Font : Calibri, 36pt, style=Bold
ForeColor : Blue
Size : 409, 81

Tips: untuk size setiap objek dapat dilakukan dengan mengklik objek yang hendak diresize lalu click and drag kotak kecil di bagian kanan bawah objek tersebut.

Saran: cari kegunaan atribut AutoSize pada label di internet.

Setelah design selesai dibuat, tekan tombol F5 pada keyboard atau klik tombol start yang terdapat di toolbar di bagian atas.



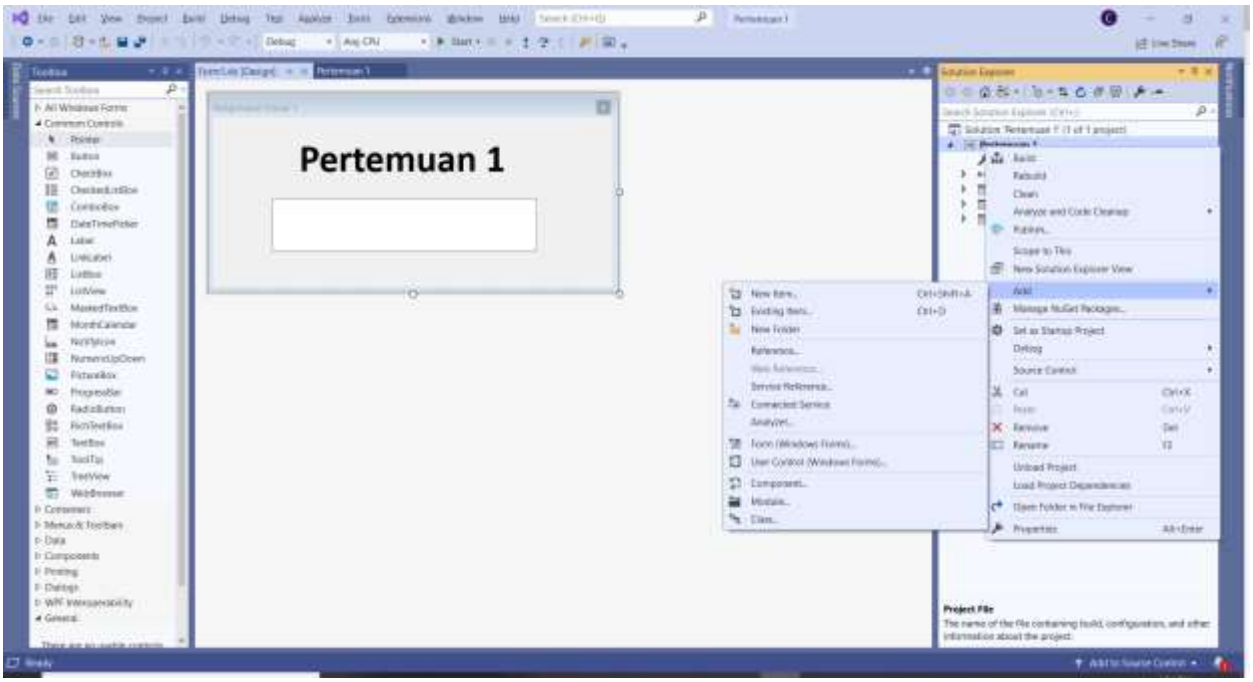
Gambar 6. Tampilan hasil run form Pengenalan Visual 1

Apabila program berhasil dieksekusi maka akan muncul tampilan seperti gambar 6. di atas, pada gambar tersebut penulis menuliskan kata “Sukses” di textbox yang tersedia. Untuk kembali ke layar visual studio dapat dilakukan dengan mengklik tombol “x” di sudut kanan atas atau menekan tombol stop pada layar visual studio.

Silahkan bereksperimen dengan objek yang lain beserta propertiesnya.

1.2. Bermain dengan multi forms.

Kita dapat menggunakan lebih dari satu form di dalam setiap project yang kita bangun. Untuk menambah form dapat dilihat pada gambar 7. berikut ini:



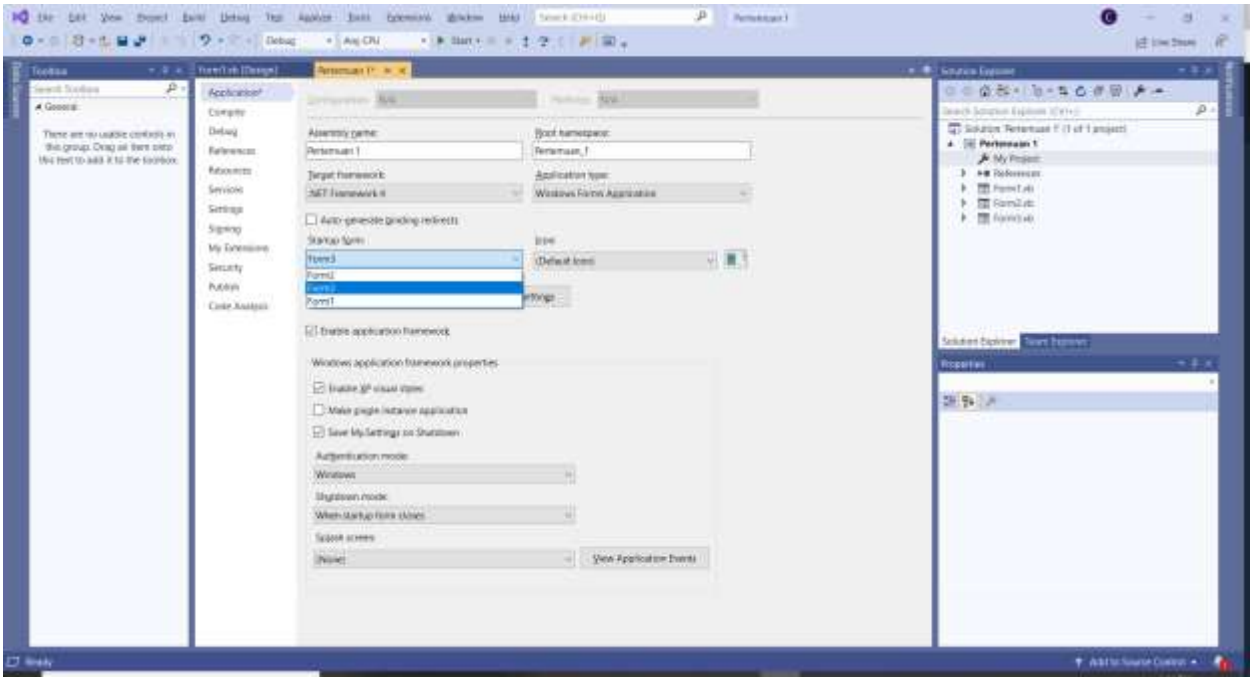
Gambar 7. Menambah form pada project

Langkah-langkah untuk menambahkan form pada project adalah sebagai berikut:

1. Klik kanan pada nama project di solution explorer.
2. Klik add pada menu yang muncul.
3. Klik Form (Windows Forms) pada sub menu yang muncul.
4. Isikan nama form yang hendak ditambahkan di project lalu klik Add.

Setelah selesai langkah-langkah di atas maka akan muncul form tersebut di layar solution explorer kita. Apabila hendak menambahkan form lain maka dapat diulangi langkah-langkah di atas.

Berikutnya untuk menjalankan form yang kita kehendaki apabila terdapat lebih pada satu form di project dapat dilakukan dengan cara berikut ini:



Gambar 8. Mengatur form awal pada project

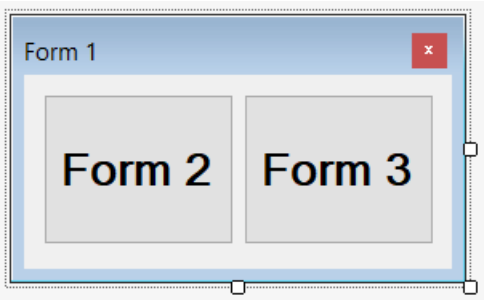
Langkah-langkah untuk mengatur form awal pada project adalah sebagai berikut:

1. Double click pada My Project yang terdapat di solution explorer.
2. Pilih startup form yang diinginkan.

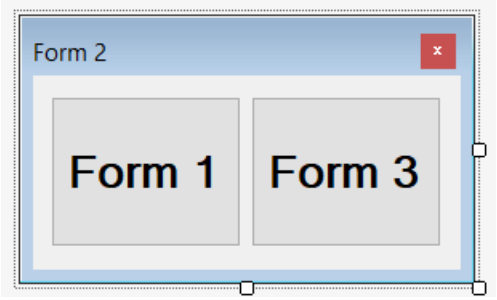
Setelah kedua langkah tersebut dikerjakan maka ketika program dieksekusi form yang dipilih pada startup form yang akan dijalankan.

Tips: untuk berpindah antar form dapat dilakukan dengan shortcut Ctrl+Tab.
Tips2: untuk menyimpan semua form sekaligus dapat dilakukan dengan shortcut Ctrl+Shift+S. (form belum disimpan akan ditandai dengan * di sudut kanan pada tab form tersebut).

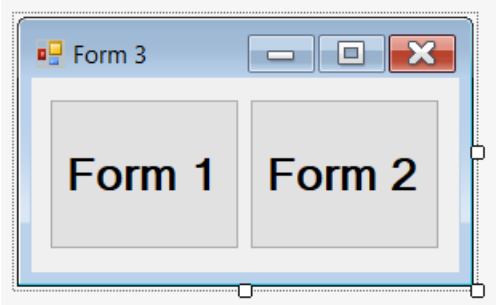
Berikutnya buat 3 form seperti gambar 9 hingga 11 berikut ini:



Gambar 9. Tampilan design form1



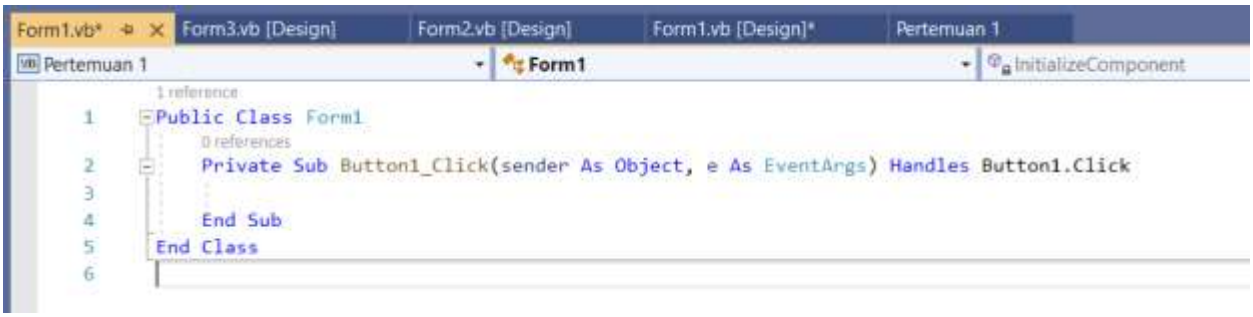
Gambar 10. Tampilan design form2



Gambar 11. Tampilan design form3

Tips: untuk mengeset nilai properties yang sama untuk objek yang berbeda tetapi memiliki tipe yang sama (misalkan mengeset properties Font untuk 2 buah objek Button) maka dapat dilakukan dengan klik objek pertama, tahan Ctrl dan klik objek selanjutnya, setelah selesai pilih properties yang hendak kita set dan isikan.

Set form 1 sebagai startup form lalu kita akan menambahkan code untuk memanggil form sesuai dengan button yang diklik. Nama button dari ketiga form adalah sama, yaitu Button1 untuk tombol sebelah kiri dan Button2 untuk tombol sebelah kanan. Kita mulai dari Form1 terlebih dahulu. Untuk menambahkan code pada objek (dalam hal ini Button1), double click pada objek tersebut sehingga muncul tampilan pada gambar 12.

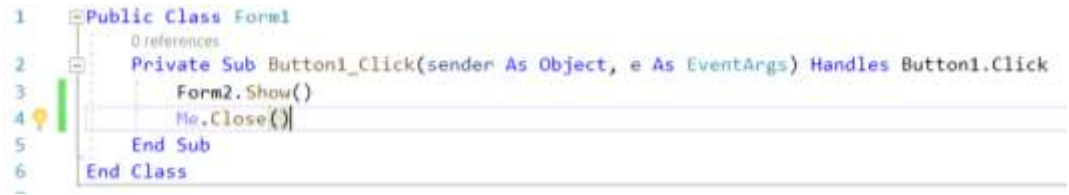


Gambar 12. Tampilan coding untuk form1

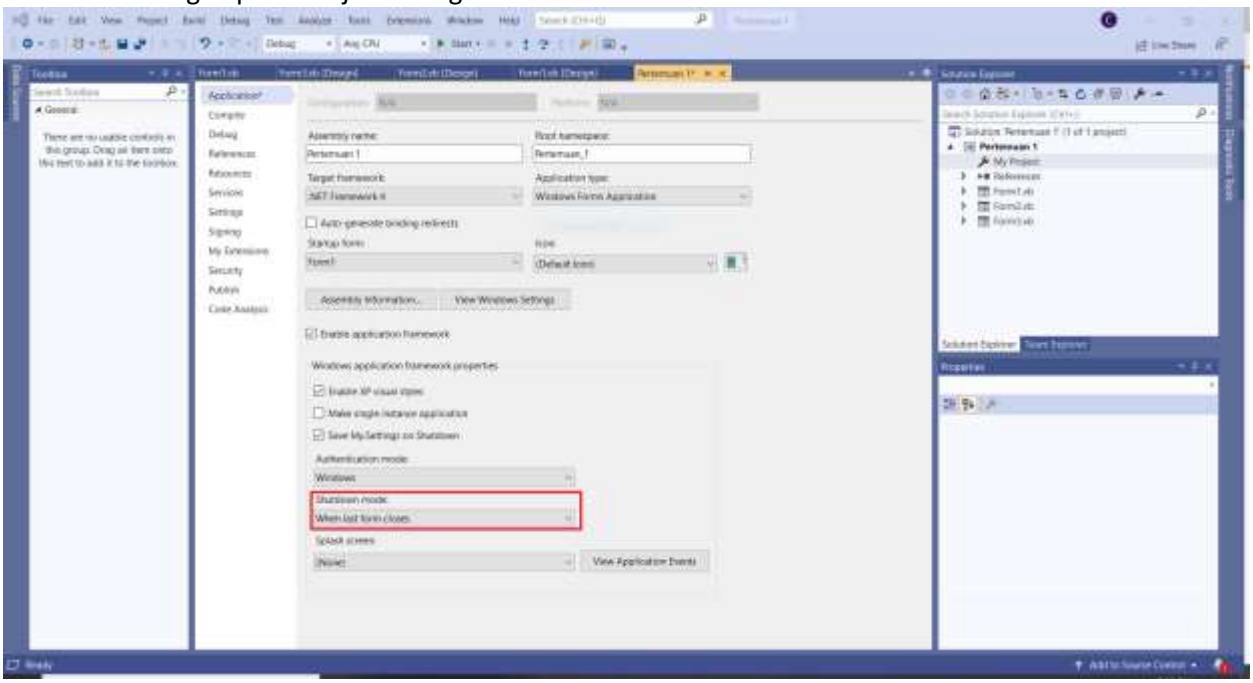
Setiap objek merupakan class termasuk form itu sendiri. Berikut adalah keterangan baris pada coding di atas:

1. Awal deklarasi class untuk Form1
2. Awal deklarasi event Button1_Click untuk objek Button.
    a. Button1_Click merupakan nama dari event.
    b. sender As Object sebagai parameter yang berguna sebagai pembangkit event.
    c. e As EventArgs sebagai pemberi object kepada event untuk ditangani.
    d. Button1.Click merupakan trigger/ pembangkit event tersebut.
3. Isi dari coding yang akan dieksekusi ketika event Button1_Click dipanggil.
4. Akhir dari event Button1_Click.
5. Akhir dari class Form1.

Isikan perintah: Form2.Show() pada baris ketiga di atas. Perintah Form2.Show() berguna untuk menampilkan Form2. Jalankan program lalu klik tombol Form 2 yang terdapat di Form1 maka akan muncul Form2 tetapi Form1 tidak hilang. Untuk menutup Form1 ketika Form2 tampil maka dapat ditambahkan code Me.Close() sehingga coding menjadi seperti ini:



Bagi yang memiliki masalah project akan berhenti apabila tombol Form 2 ditekan bisa diatasi dengan merubah settingan pada Project sebagai berikut:



Gambar 13. Properties shutdown mode

Ubah shutdown mode menjadi when last form closes.
Saran: pelajari di internet mengenai Opening, Closing, and Hiding Forms pada VB. NET.

Latihan: lengkapi coding ketiga form pada Topik 2 sehingga dapat berjalan sebagaimana mestinya.

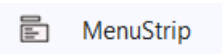
1.3. Penggunaan MDIForm

Kita dapat membuat sebuah “form induk” untuk menampung “form anak” di dalamnya. Dengan menggunakan form induk maka form-form anak di dalamnya merupakan bagian dari form induk tersebut, ini akan membuat form induk sebagai sebuah “penampung” dari form-form anak di dalamnya. MDI sendiri merupakan singkatan dari Multiple Document Interface yang secara sederhana dapat kita artikan sebagai tampilan yang dapat menampung banyak dokumen sekaligus. Di dalam pemrograman dengan Visual Basic .NET hal ini dapat kita lakukan dengan mudah. Pertama-tama sediakan 3 form terlebih dahulu pada project, kosong saja ketiganya tetapi apabila anda mau mengambil dari form-form sebelumnya lebih baik lagi.

Kini form 1 akan kita set sebagai tempat untuk menampung form-form lain atau diistilahkan dengan MDI Parent Form. Form 1 cukup kosong saja tampilannya tetapi property dari **IsMdiContainer** menjadi **True**. Hal ini bertujuan agar Form 1 menjadi Parent dari form 2 dan form 3.
Berikutnya agar ketika Form 1 dibuka otomatis menjadi berukuran penuh (maksimal) maka kita set property dari **WindowState** menjadi **Maximized**. Jalankan program dengan Form 1 sebagai startup form maka kita akan mendapat tampilan seperti ini:

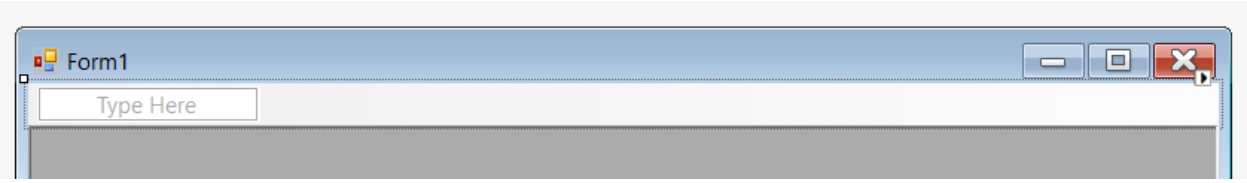
Gambar 14. Tampilan design awal MDIForm

Berikutnya kita akan menambahkan control menu strip yang berfungsi untuk menambahkan menu di bagian atas Form 1.



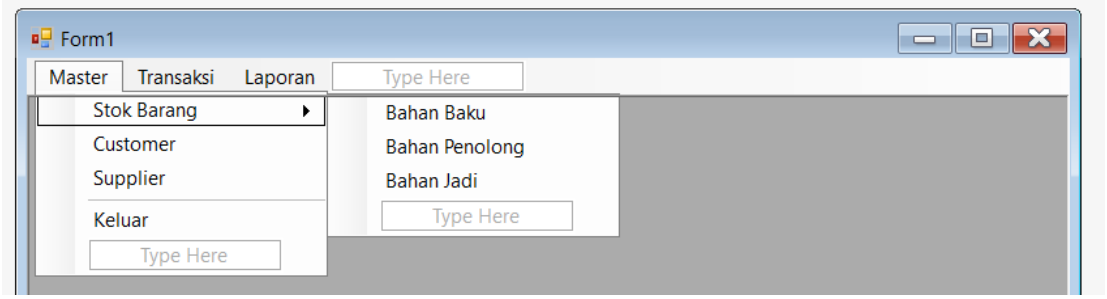
Gambar 15. Tampilan pilihan MenuStrip pada toolbox

Setelah ditambahkan menu strip maka tampilannnya akan berubah menjadi seperti ini:

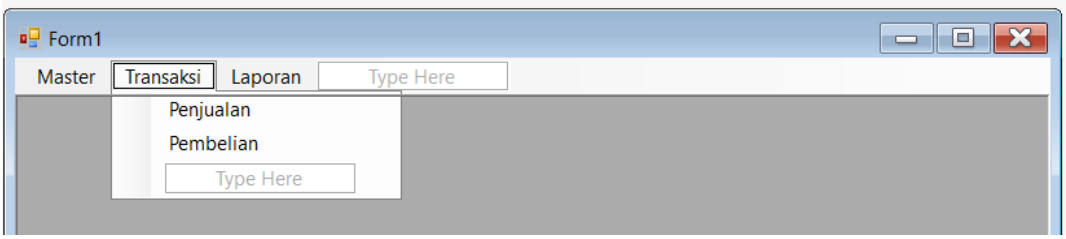


Gambar 16. Tampilan design MDIForm dengan MenuStrip

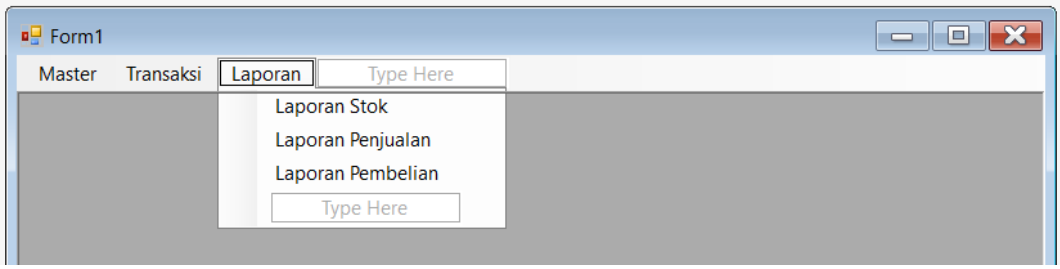
Berikutnya silahkan isikan nama untuk menu di pilihan tersebut, anda juga dapat menambahkan sub menu di dalamnya menjadi seperti ini:



Gambar 17. Tampilan menu Master



Gambar 18. Tampilan menu Transaksi



Gambar 19. Tampilan menu Laporan

Tips: untuk membuat tanda garis di menu, seperti pada tampilan gambar 17, cukup diketikkan – di bagian Type Here.

Berikutnya coba lakukan double click pada salah satu menu, misalkan pada Transaksi -> Penjualan, maka akan muncul program ini di bagian program:

```
Private Sub PenjualanToolStripMenuItem_Click(sender As Object, e As EventArgs) _
    Handles PenjualanToolStripMenuItem.Click

End Sub
```

Isikan saja **Form2.MdiParent = Me** dan **Form2.Show** di dalamnya untuk memunculkan Form 2. Lakukan hal yang sama untuk sub menu lainnya.

Latihan: Buat program dengan MDIForms untuk memasukkan dan mengkategorikan form-form yang telah kita buat sebelumnya. Pengkategorian silahkan anda tentukan dengan bebas.

Bab 2. Event dan Controls.

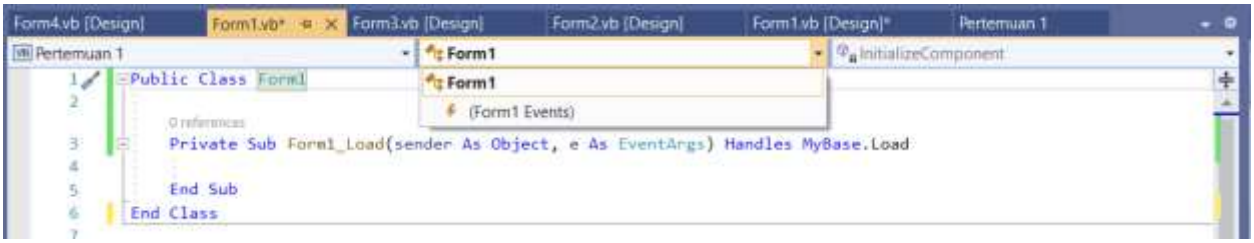
2.1. Bermain dengan event.

Mari kita perdalam lagi sedikit mengenai event. Event adalah keadaan yang terjadi pada suatu objek, baik keadaan dari dalam atau dari luar. Misalkan ketika tombol diklik, textbox diketikkan, checkbox dicentang, kursor mouse melewati objek, form ditutup, form dibuka, dan sebagainya. Setiap jenis objek memiliki event yang berbeda, meskipun ada beberapa event yang sama, seperti mouse melewati objek.

Buat 1 form dan kosongkan, setelah itu double click form tersebut untuk masuk ke mode coding atau dapat juga dengan menggunakan shortcut F7. Untuk kembali ke layar design form bersangkutan dapat digunakan shortcut Shift + F7.

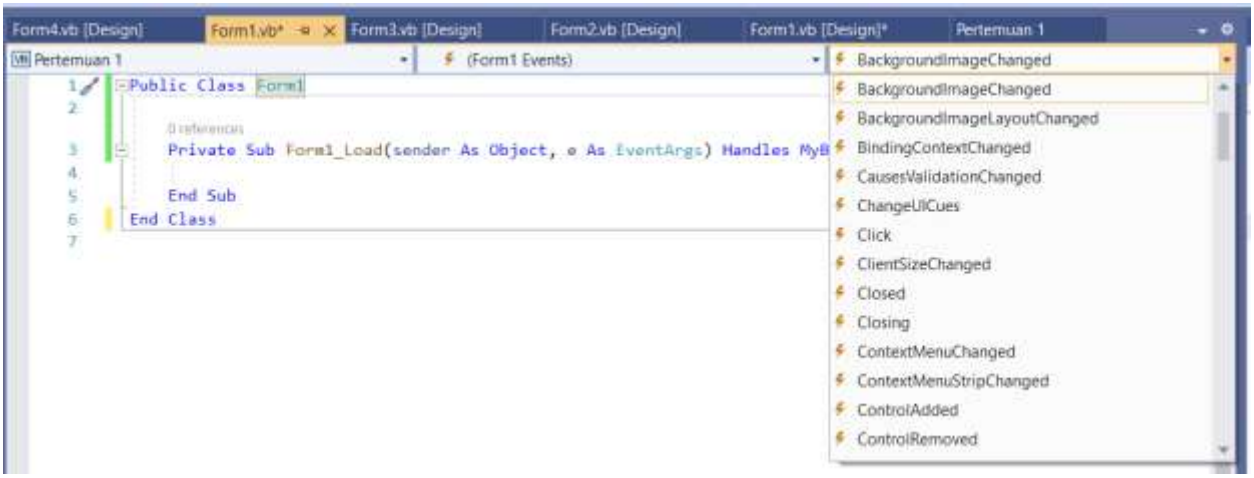
Gambar 20. Tampilan coding event Form Load

Kosongkan baris kedua, ini akan kita gunakan untuk mengisi coding yang baru. Secara default, event untuk form adalah Load (setiap objek memiliki event default, misal button adalah Click). Kita akan membuat program untuk merubah warna form apabila mouse bergerak di atas form tersebut dan merubah warnanya lagi ketika mouse meninggalkan form tersebut.



Gambar 21. Memilih form events

Klik panah seperti ditunjukkan pada gambar di atas dan pilih form1 events.



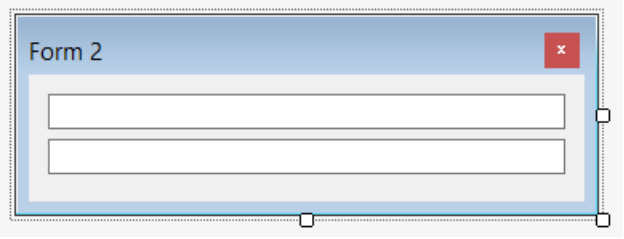
Gambar 22. Pilihan event pada form

Scroll ke bawah dan pilih MouseHover dan MouseLeave. Selanjutnya isikan coding seperti berikut:

```
0 references
Private Sub Form1_MouseHover(sender As Object, e As EventArgs) Handles Me.MouseHover
    Me.BackColor = Color.FromArgb(200, 200, 0)
End Sub

0 references
Private Sub Form1_MouseLeave(sender As Object, e As EventArgs) Handles Me.MouseLeave
    Me.BackColor = Color.FromArgb(0, 200, 0)
End Sub
```

Guna property BackColor adalah untuk warna latar belakang pada form. Untuk mengisi warna kita tidak bisa langsung menggunakan vbBlue misalnya seperti pada VB model lama, kita dapat menggunakan object Color dengan property FromArgb. FromArgb akan mengisi campuran warna berdasarkan informasi Red Green Blue yang berkisar dari 0 sampai dengan 255 untuk masing-masing warna.
Event MouseHover akan dieksekusi ketika cursor mouse melewati bidang form dan event MouseLeave akan dieksekusi ketika mouse meninggalkan bidang layar.

Contoh berikutnya, buat form2 seperti pada gambar berikut ini:



Gambar 23. Tampilan design form2 (Duplikasi Teks)

Textbox pertama bernama textbox1 dan textbox kedua bernama textbox2, selanjutnya ubah coding pada form 2 menjadi seperti ini:

```
Public Class Form2
    0 references
    Private Sub TextBox1_TextChanged(sender As Object, e As EventArgs) Handles TextBox1.TextChanged
        TextBox2.Text = UCase(TextBox1.Text)
    End Sub
End Class
```

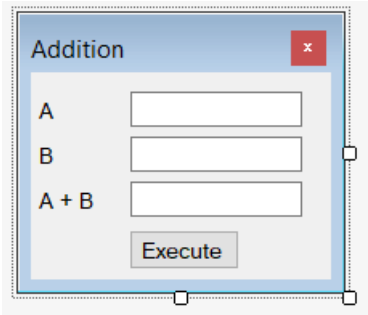
Event TextChanged akan dieksekusi setiap terjadi perubahan teks pada textbox yang bersangkutan. Fungsi UCase adalah untuk merubah tulisan menjadi huruf besar. Sebenarnya nama event, property, dan fungsi di dalam bahasa pemrograman ini sudah cukup jelas dan mudah untuk diingat asalkan kita memiliki sedikit pengetahuan bahasa Inggris. Selanjutnya mari kita coba event berikut ini pada textbox1:

```
Public Class Form2
    'Private Sub TextBox1_TextChanged(sender As Object, e As EventArgs) Handles TextBox1.TextChanged
    '    TextBox2.Text = UCase(TextBox1.Text)
    'End Sub

    0 references
    Private Sub TextBox1_KeyDown(sender As Object, e As KeyEventArgs) Handles TextBox1.KeyDown
        If e.KeyCode = Keys.Enter Then
            TextBox2.Text = TextBox1.Text
        End If
    End Sub
End Class
```

Kini kita akan menggunakan event KeyDown tetapi sebelumnya kita perlu menonaktifkan event TextChanged sebelumnya. Ubah perintah event TextChanged menjadi komentar dengan cara memblok baris-baris tersebut lalu tekan shortchut Ctrl+K, Ctrl+C. **Penting: setelah menekan Ctrl+K lepaskan tangan dari kedua tombol tersebut terlebih dahulu sebelum menekan Ctrl+C.** Untuk merubah kembali menjadi coding maka tekan Ctrl+K, Ctrl+U.

Tips: carilah perbedaan event KeyDown, KeyPress, dan KeyUp
Tips2: untuk mengetik coding dengan cepat, manfaatkan Intellisense dan Shortcut, misalkan setelah memilih property objek dengan menekan titik (.) maka dapat dilanjutkan dengan menekan spasi setelah menemui pilihan property yang dikehendaki tanpa perlu mengetikkan sampai lengkap.

Contoh berikutnya, buat tampilan seperti gambar ini:



Gambar 24. Tampilan design form Penjumlahan

Set nama textbox menjadi txtA, txtB, dan txtC, selanjutnya buat coding seperti berikut ini:

```
Public Class Form3
    Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
        txtC.Text = txtA.Text + txtB.Text
    End Sub
End Class
```

Apabila kita memberikan nilai 2 di A dan 4 di B mana C akan memunculkan nilai 24 ketika tombol Execute ditekan. Hal ini dikarenakan isi dari textbox diperlakukan sebagai string. Untuk merubah isi textbox menjadi nilai dapat maka coding dapat diubah menjadi seperti ini:

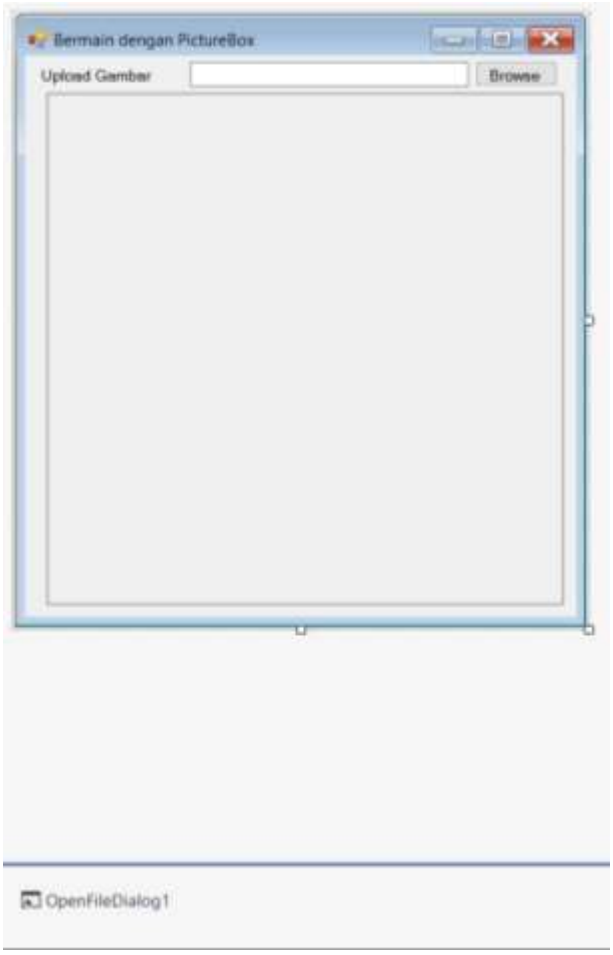
```
Public Class Form3
    Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
        txtC.Text = CInt(txtA.Text) + CInt(txtB.Text)
    End Sub
End Class
```

Fungsi CInt akan merubah nilai di dalam textbox A dan B menjadi integer.

Latihan: Buat program untuk memfilter ketikan dari user sehingga textbox hanya boleh diisi dengan angka.

2.2. Bermain dengan PictureBox

Guna dari picturebox adalah untuk menampilkan gambar dengan berbagai format seperti BMP, PNG, maupun JPG dan format default lainnya. Untuk menggunakan picturebox biasanya kita gabungkan dengan OpenFileDialog. Objek openfiledialog berfungsi seperti ketika kita menekan tombol open pada aplikasi Notepad maka akan diberikan layar pilihan untuk memilih file yang hendak kita buka. Program sederhana berikut berguna untuk memilih gambar yang hendak dibuka dan menampilkan gambar tersebut. Pertama-tama buatlah design seperti gambar berikut ini:



Gambar 25. Tampilan design form Bermain dengan PictureBox

Ubah nama dari objek menjadi seperti ini:

1. Textboxt = txtLokasi
2. Button = btnBrowse
3. PictureBox = PictureBox1
4. OpenFileDialog = OpenFileDialog1

Perhatian: objek OpenFileDialog tidak akan muncul di layar design bagian form melainkan muncul di bagian bawah dari tampilan design form tersebut.

Tips: untuk objek OpenFileDialog apabila tidak terdapat di toolbox bagian "Common Controls", maka coba cari di "All Windows Forms" atau "Dialogs"

Setelah mendesain tampilan seperti gambar di atas maka untuk codingnya kita isikan sebagai berikut:

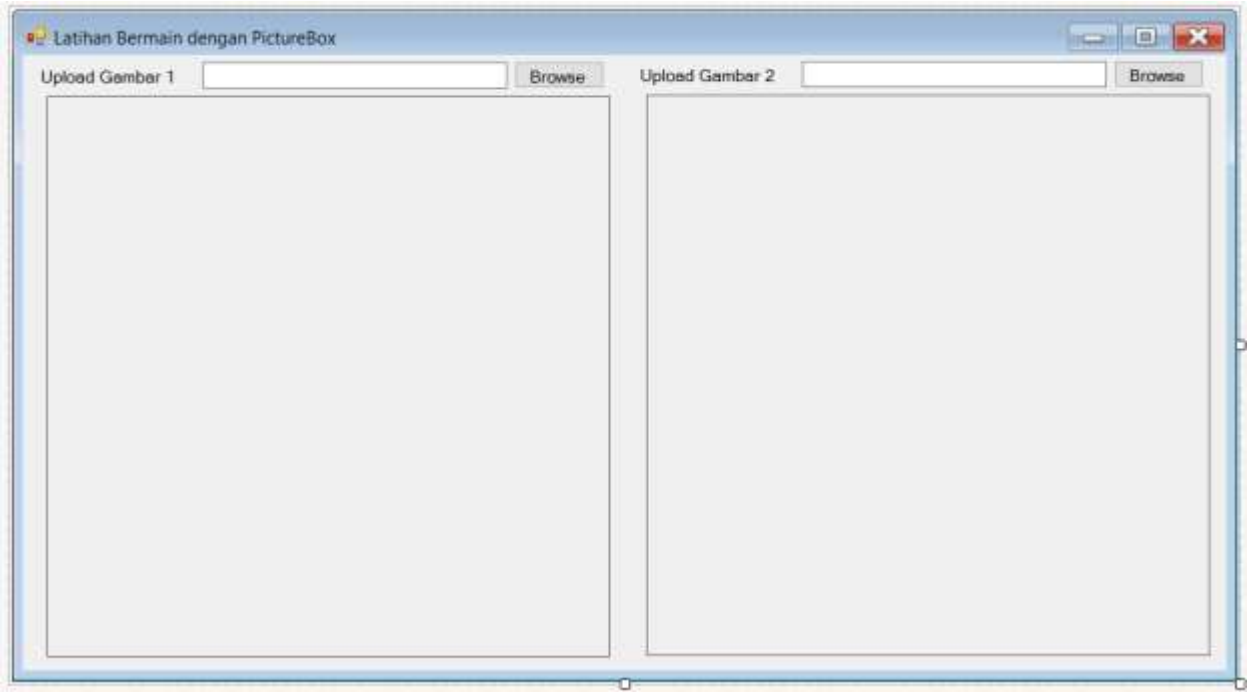
```
Public Class Form1
    Private Sub btnBrowse_Click(sender As Object, e As EventArgs) Handles btnBrowse.Click
        OpenFileDialog1.ShowDialog()
    End Sub

    Private Sub OpenFileDialog1_FileOk(sender As Object, e As System.ComponentModel.CancelEventArgs) _
        Handles OpenFileDialog1.FileOk
        txtLokasi.Text = OpenFileDialog1.FileName
        PictureBox1.Load(txtLokasi.Text)
    End Sub
End Class
```

Cara mengetikkan coding ini cukup gampang karena terletak di bagian event default untuk Button maupun OpenFileDialog, sehingga kita cukup meng-double click objek Button maupun OpenFileDialog di layar design.

Tips: apabila coding dirasakan terlalu panjang maka kita dapat mengetikkannya ke bawah dengan menambahkan simbol garis bawah di belakangnya. Contohnya seperti pada Private Sub OpenFileDialog1_FileOK di gambar 26.

Saran: silahkan cari perbedaan untuk property SizeMode pada PictureBox, seperti: Normal, StretchImage, AutoSize, CenterImage, dan Zoom.

Latihan: buat program untuk membandingkan apa gambar sebelah kiri dengan gambar sebelah kanan sama.



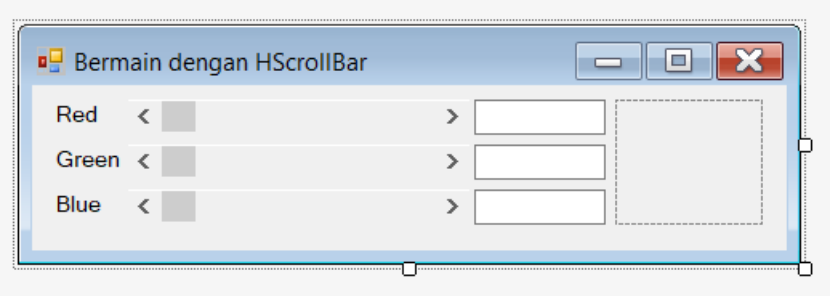
Gambar 26. Tampilan design form latihan Bermain dengan PictureBox

2.3. Bermain dengan Horizontal Scroll Bar

Berikutnya kita akan bermain dengan horizontal scroll bar. Objek ini berguna sebagai slider dan kita dapat menentukan sendiri berapa nilai minimum dan nilai maksimum dari slider tersebut. Variasi lain dari horizontal scroll bar tentunya adalah vertical scroll bar. Adapun property yang perlu kita perhatikan untuk horizontal scroll bar adalah:

1. Minimum, berguna untuk menset nilai minimum dari horizontal scroll bar.
2. Maximum, berguna untuk menset nilai maksimum dari horizontal scroll bar.
3. SmallChange, berguna untuk berpindah sejauh berapa poin apabila panah kanan maupun kiri scroll bar diklik.
4. LargeChange, berguna untuk berpindah sejauh berapa poin apabila bagian tengah antara panah kanan dan kiri diklik.

Sebelum kita lanjutkan coba design tampilan seperti berikut ini:



Gambar 27. Tampilan design form Bermain dengan HScrollBar

Berikutnya ubah nama dari objek menjadi seperti ini:

1. Nama objek horizontal scroll bar secara berurutan dari atas ke bawah: hsbR, hsbG, hsbB.
2. Nama objek textbox secara berurutan dari atas ke bawah: txtR, txtG, txtB.
3. Nama objek picturebox menjadi picRGB.

Untuk nilai properties kali ini kita akan menset langsung melalui coding pada event Form_Load yang akan dieksekusi ketika form tersebut dimunculkan:

```
Private Sub Form2_Load(sender As Object, e As EventArgs) Handles Me.Load
    hsbR.Minimum = 0
    hsbR.Maximum = 255
    hsbR.LargeChange = 1
    hsbG.Minimum = 0
    hsbG.Maximum = 255
    hsbG.LargeChange = 1
    hsbB.Minimum = 0
    hsbB.Maximum = 255
    hsbB.LargeChange = 1
End Sub
```

Sedangkan untuk merubah warna dari picturebox akan kita lakukan ketika slider dari horizontal scroll bar berubah baik untuk warna merah, hijau, maupun biru:

```
0 references
Private Sub hsbR_Scroll(sender As Object, e As ScrollEventArgs) Handles hsbR.Scroll
    txtR.Text = hsbR.Value
    picRGB.BackColor = Color.FromArgb(hsbR.Value, hsbG.Value, hsbB.Value)
End Sub
```

```
0 references
Private Sub hsbG_Scroll(sender As Object, e As ScrollEventArgs) Handles hsbG.Scroll
    txtG.Text = hsbG.Value
    picRGB.BackColor = Color.FromArgb(hsbR.Value, hsbG.Value, hsbB.Value)
End Sub
```

```
0 references
Private Sub hsbB_Scroll(sender As Object, e As ScrollEventArgs) Handles hsbB.Scroll
    txtB.Text = hsbB.Value
    picRGB.BackColor = Color.FromArgb(hsbR.Value, hsbG.Value, hsbB.Value)
End Sub
```

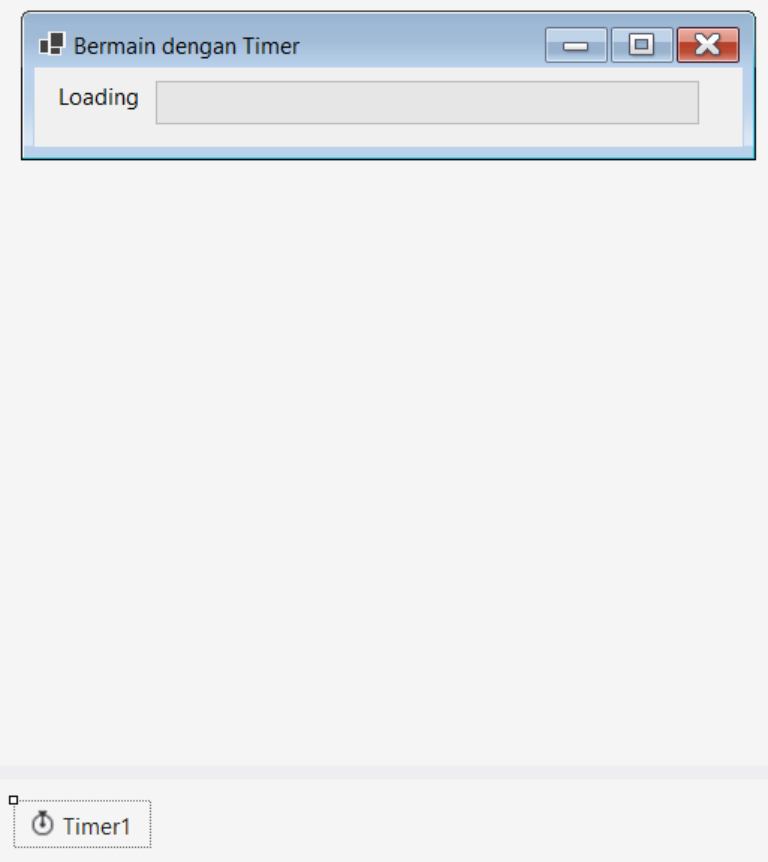
Tips: secara bawaan nilai property LargeChange untuk horizontal scroll bar adalah 10, tetapi kelemahannya adalah apabila slider kita geser rata kanan maka nilai maksimal tidak dapat mencapai 255, coba cari tahu kenapa bisa begitu.
Tips2: untuk menentukan urutan perpindahan fokus antar objek ketika tombol Tab ditekan silahkan revisi nilai dari property TabIndex.

Latihan: lengkapi program pada gambar 27 sehingga ketika nilai textbox kita ketikkan secara manual maka otomatis slider dari warna bersangkutan akan pindah ke posisi sesuai nilai yang kita ketikkan. Buat cek error agar nilai tidak dapat melebihi range 0 sampai dengan 255.

2.4. Bermain dengan Timer

Timer adalah component yang digunakan untuk menjalankan perintah untuk setiap satuan waktu. Lalu kenapa object timer dikatakan component dan bukan control seperti object textbox, label, picturebox, dan sebagainya seperti yang telah dipelajari? Control adalah object yang dapat kita tambahkan di form dan muncul di tampilan form tersebut ketika program dijalankan, sedangkan component adalah objek yang kita tambahkan di form dan tidak muncul di form ketika program dijalankan. Timer sendiri akan berjalan di belakang layar sehingga kita kategorikan sebagai component.

Berikut coba design tampilan form seperti gambar di bawah ini:



Gambar 28. Tampilan design form Bermain dengan Timer

Control di dalam form adalah label dan progress bar, untuk progress bar berikan nama **pb**. Untuk menambahkan timer dapat dengan cara click and drag object timer dari toolbox ke form atau double click object timer pada toolbox. Object timer tersebut tidak akan muncul di form melainkan akan muncul di bawah seperti halnya object openfiledialog yang telah kita pelajari sebelumnya.

Selanjutnya untuk menjalankan animasi seakan-akan proses loading sedang berlangsung kita dapat menggunakan program berikut:

```
Public Class Form1
    Private Sub Form1_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        pb.Maximum = 100
        pb.Value = 1
        Timer1.Start()
    End Sub

    Private Sub Timer1_Tick(sender As Object, e As EventArgs) Handles Timer1.Tick
        If pb.Value < pb.Maximum Then
            pb.Value = pb.Value + 1
        End If
    End Sub
End Class
```

Untuk menset nilai maksimum dari progress bar kita menggunakan property Maximum, pada contoh di atas kita menset nilai maksimum = 100. Ketika form ditampilkan maka kita menset nilai awal dari progress bar adalah 1 dengan cara pb.Value = 1. Setelah itu timer akan mulai dijalankan dengan perintah Timer1.Start().
Event handle untuk timer secara default adalah tick. Tick akan dipanggil setiap 1 interval timer. Secara bawaan interval dari timer adalah 100 yang berarti sebesar 100 miliseconds = 0.1 seconds. Untuk membuat agar animasi berjalan lebih cepat dapat dilakukan dengan mengurangi nilai interval. Tentu saja pada proses loading yang sebenarnya perhitungan dari value progress bar akan dilakukan sesuai dengan berapa banyak data yang telah diloading atau berapa banyak proses loading selesai dilakukan dan dibandingkan dengan total banyak data atau banyak proses yang dibutuhkan sampai operasi selesai.
Pada program di atas dapat dilihat untuk setiap tick maka nilai value dari progress bar akan bertambah. Penting untuk diingat bahwa value dari progress bar tidak boleh melebihi nilai maksimum dari progress bar tersebut oleh karena itu kita menggunakan if condition untuk membandingkan value dengan maksimum dari progress bar tersebut.
Apabila kita hendak menjalankan timer secara manual dengan menggunakan tombol maka pada saat form diload panggil saja fungsi Timer1.Stop(). Lalu pada event penekanan tombol tersebut tambahkan Timer1.Start().
Misalkan kita hendak memiliki sebuah tombol yang dapat menghentikan timer ketika tombol tersebut ditekan dan melanjutkan kembali timer tersebut dengan menekan tombol yang sama, maka kita dapat menggunakan program berikut:

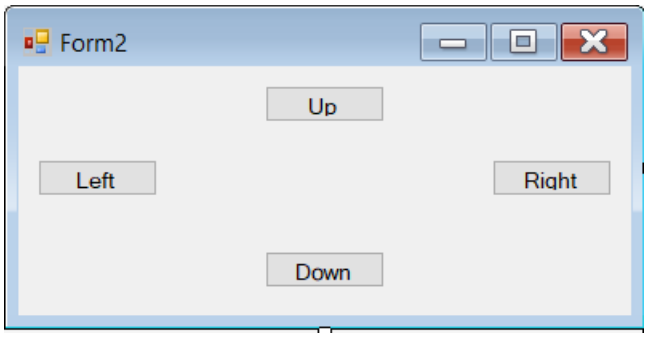
```
Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
    If Timer1.Enabled Then
        Timer1.Stop()
    Else
        Timer1.Start()
    End If
End Sub
```

Untuk mengecek apakah timer tersebut sedang berjalan atau tidak kita dapat menggunakan Timer1.Enabled. Apabila true maka timer tersebut sedang berjalan, sedangkan false berarti timer

tersebut sedang berhenti. Dengan memanfaatkan property enabled tersebut kita juga dapat menjalankan dan menghentikan timer seperti program berikut ini:

```
Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
    If Timer1.Enabled Then
        Timer1.Enabled = False
    Else
        Timer1.Enabled = True
    End If
End Sub
```

Timer juga berguna untuk membuat animasi, misalkan kita hendak menggerakkan form dengan menggunakan 4 buah tombol untuk arah atas, bawah, kiri dan kanan.Pertama-tama kita design tampilan form seperti ini:



Gambar 29. Tampilan design form Menggerakkan Form

Ubah tombol arah menjadi btnUp, btnLeft, btnRight, dan btnDown. Setelah itu tambahkan variabel direction untuk menentukan arah pergerakan dan pada event form load tambahkan program sebagai berikut:

```
Dim direction As String

0 references
Private Sub Form2_Load(sender As Object, e As EventArgs) Handles Me.Load
    Timer1.Interval = 10
    Timer1.Stop()
End Sub
```

Interval dari timer kita ubah menjadi 10 agar pergerakan menjadi lebih cepat. Selanjutnya untuk event click dari setiap tombol kita ketikkan program berikut:

```
0 references
Private Sub btnUp_Click(sender As Object, e As EventArgs) Handles btnUp.Click
    direction = "Up"
    Timer1.Start()
End Sub
```

```
0 references
Private Sub btnLeft_Click(sender As Object, e As EventArgs) Handles btnLeft.Click
    direction = "Left"
    Timer1.Start()
End Sub
```

```
0 references
Private Sub btnRight_Click(sender As Object, e As EventArgs) Handles btnRight.Click
    direction = "Right"
    Timer1.Start()
End Sub
```

```
0 references
Private Sub btnDown_Click(sender As Object, e As EventArgs) Handles btnDown.Click
    direction = "Down"
    Timer1.Start()
End Sub
```

Setiap penekanan tombol kita akan menentukan arah dari pergerakan form, program untuk pergerakan form bukan diletakkan di dalam event click dari tombol melainkan di dalam event tick dari timer seperti berikut ini:

```
Private Sub Timer1_Tick(sender As Object, e As EventArgs) Handles Timer1.Tick
    Select Case direction
        Case "Up" : Me.Location = New Point(Me.Location.X, Me.Location.Y - 1)
        Case "Left" : Me.Location = New Point(Me.Location.X - 1, Me.Location.Y)
        Case "Right" : Me.Location = New Point(Me.Location.X + 1, Me.Location.Y)
        Case "Down" : Me.Location = New Point(Me.Location.X, Me.Location.Y + 1)
    End Select

End Sub
```

Untuk menentukan lokasi dari form kita menggunakan Me.Location di mana Me akan merujuk pada form yang sedang aktif. Untuk nilai kordinat kita perlu menggunakan nilai bertipe data point, oleh karena itu kita menggunakan perintah New Point yang diikuti dengan posisi X dan Y pada layar.
Muncul pertanyaan, apabila timer sudah dijalankan dengan perintah Start pada penekanan tombol arah atas, lalu kita menekan tombol panah kanan dan perintah Start tersebut dipanggil lagi, apakah tidak muncul masalah? Jawabannya adalah tidak. Hal ini dikarenakan apabila perintah Start dipanggil pada saat kondisi Enabled dari Timer adalah True maka perintah Start tersebut hanya akan membuat Enabled dari Timer menjadi True lagi.

Latihan: Lengkapi program pergerakan form sehingga apabila form sudah berada di batas layar monitor maka pergerakan akan berhenti.

2.5. Membuat control dari coding

Kali ini kita akan membahas mengenai bagaimana caranya untuk membuat control dari coding yang akan kita tampilkan ke form. Langkah pertama cukup membuat sebuah form kosong lalu isikan program berikut ini:

```
Public Class Form1
    Private Sub Form1_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        Dim txt As New TextBox
        txt.Size = New Point(200)
        txt.Location = New Point(20, 20)
        Me.Controls.Add(txt)
    End Sub
End Class
```

Program di atas cukup sederhana, berikut adalah penjelasannya:

1. Dim txt as New TextBox berguna untuk mendeklarasikan objek baru berupa textbox, pada contoh di atas kita menggunakan nama variabel txt sebagai objek dari textbox yang akan kita tambahkan ke form.
2. txt.Size berguna untuk menentukan ukuran dari textbox kita (200 menentukan 200 point untuk lebar dari textbox). Dikarenakan pada setting default dari textbox untuk property multiline adalah false maka kita tidak perlu menset tinggi dari textbox tersebut. Untuk menset ukuran atau posisi dari control kita harus menggunakan nilai dengan tipe data Point, itulah sebabnya pada program di atas kita menggunakan New Point.
3. txt.Location berguna untuk menentukan posisi dari textbox tersebut di form.
4. Me.Controls.Add(txt) merupakan perintah yang paling penting, apabila tidak ada perintah ini maka objek tersebut tidak akan ditambahkan ke form.

Pada Visual Basic versi lama control pada form dapat dibuat dalam bentuk array, karena itu ada property Index untuk control. Pada VB .NET 2019 hal ini tidak ada sehingga apabila kita hendak membuat koleksi dari control pada form yang berupa array kita harus melakukannya secara manual melalui program. Berikut ini kita akan membuat 4 buah textbox yang berupa array untuk ditampilkan ke dalam form:

```
Public Class Form2
    Dim txt(4) As TextBox

    0 references
    Private Sub Form2_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        Dim i As Integer
        For i = 0 To 3
            txt(i) = New TextBox
            txt(i).Size = New Point(100, 20)
            txt(i).Location = New Point(20, 20 + i * 30)
            Me.Controls.Add(txt(i))
        Next
    End Sub
```

Perhatian: Untuk deklarasi textbox tersebut kita letakkan di luar fungsi dari form_load untuk mencegah textbox tersebut menjadi variabel lokal dari fungsi tersebut.

Secara garis besar program di atas hampir sama dengan program sebelumnya hanya saja kali ini kita membuatnya dalam bentuk array. Selain itu untuk lokasi kita gunakan sedikit perhitungan matematika agar tampilan textbox tersebut tidak saling bertimpa. Pada contoh di atas juga terdapat sedikit perbedaan pada perintah Size, di sana digunakan New Point (100, 20) tetapi karena property MultiLine masih berupa default yaitu False maka ukuran tinggi (20) tidak akan memberikan dampak pada tampilan textbox.

Lalu bagaimana apabila kita hendak menggunakan event untuk kumpulan textbox tersebut? Kita dapat menambahkan AddHandler di program sehingga menjadi seperti berikut ini:

```
Public Class Form2
    Dim txt(4) As TextBox

    0 references
    Private Sub Form2_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        Dim i As Integer
        For i = 0 To 3
            txt(i) = New TextBox
            txt(i).Size = New Point(100, 20)
            txt(i).Location = New Point(20, 20 + i * 30)
            Me.Controls.Add(txt(i))
            AddHandler txt(i).LostFocus, AddressOf TurnUCASE
        Next
    End Sub

    1 reference
    Private Sub TurnUCASE(ByVal sender As System.Object, ByVal e As System.EventArgs)
        Dim x As TextBox = DirectCast(sender, TextBox)
        x.Text = x.Text.ToUpper
    End Sub
End Class
```

Pada program di atas kita menggunakan perintah AddHandler untuk menghubungkan event LostFocus untuk setiap textbox dengan fungsi TurnUCASE. TurnUCASE adalah fungsi yang kita buat sendiri untuk merubah isi dari textbox menjadi huruf besar semua.

Latihan: Ubah program di atas sehingga merubah isi dari textbox menjadi huruf kecil semua.

Bab 3. Procedure dan Function

3.1. Procedure

Mari kita flashback kembali ke pelajaran kita tentang sub program di algoritma dan pemrograman. Sub program dapat kita bagi menjadi procedure dan function. Pada pemrograman visual hal ini juga kita jumpai, secara garis besar penggunaan procedure dan function adalah sama di mana procedure tidak mengembalikan nilai dan function mengembalikan nilai dengan perintah return. Mari kita mulai dengan procedure.

Berbicara mengenai sub program tidak terlepas dari parameter dan cara passingnya (atau sebagian sumber menyebut dengan passed by). Ada 2 cara passing by, yaitu by value dengan by reference. Sedikit pengingat, passing by value tidak merubah nilai variabel asal sedangkan passing by reference merubah variabel asal. Di dalam bahasa pemrograman ini kedua cara tersebut juga dapat diterapkan untuk sub program yang hendak kita buat. Ingat deklarasi sub program harus terletak di dalam Class Form, misalkan kita tuliskan coding untuk sub program setelah baris Public Class Form1.

Mari kita buat sebuah procedure untuk menukar nilai antara kedua variabel integer:

```
Sub Swap(ByVal A As Integer, ByVal B As Integer)
    Dim Temp As Integer
    Temp = A
    A = B
    B = Temp
End Sub
```

Deklarasi procedure diawali dengan Sub atau Public Sub apabila ingin agar procedure ini dapat dipanggil di form lainnya dalam sebuah project yang sama. Berikutnya kita mendeklarasikan parameter yang kita inginkan yang terdiri atas jenis passing by yang hendak kita gunakan, nama parameter, dan tipe data dari parameter.

Pada contoh di atas kita membuat procedure dengan nama Swap dan kita memiliki 2 buah parameter, yaitu A dan B dengan tipe data integer. Adapun passing by yang kita gunakan adalah by value, tentu saja ini salah mengingat kita hendak menukar nilai kedua buah parameter tetapi mari kita coba saja terlebih dahulu. Pada Form_Load kita ketikkan code sebagai berikut:

```
Private Sub Form1_Load(sender As Object, e As EventArgs) Handles MyBase.Load
    Dim X, Y As Integer
    X = 7
    Y = 9
    MessageBox.Show("X = " & X & " Y = " & Y)
    Swap(X, Y)
    MessageBox.Show("X = " & X & " Y = " & Y)
End Sub
```

Ketika coding di atas kita panggil maka pertama-tama akan ditampilkan " X = 7 Y = 9" untuk nilai X dan Y awal. Setelah kita panggil procedure Swap dengan perintah Swap(X,Y) maka kita tampilkan kembali nilai X dan Y. Output akan tetap menghasilkan " X = 7 Y = 9". Hal ini terjadi karena kita menggunakan passing by value yang tidak merubah nilai X dan Y.

Alih-alih passing by value, untuk melakukan update terhadap nilai dari parameter, kita dapat menggunakan passing by reference sebagai berikut:

```
Sub Swap(ByRef A As Integer, ByRef B As Integer)
    Dim Temp As Integer
    Temp = A
    A = B
    B = Temp
End Sub
```

Dengan menggunakan passing by reference maka didapatlah hasil yang kita inginkan yaitu nilai X dan Y akan ditukar setelah procedure Swap dipanggil.
Di dalam bahasa pemrograman VB .NET kita tidak diharuskan meletakkan deklarasi sub program sebelum sub program tersebut dipanggil. Misalnya pada contoh di atas bisa saja kita meletakkan baris code untuk Form_Load di atas Sub Swap. Meskipun hal tersebut diijinkan tetapi untuk memudahkan pembacaan code maka sangat disarankan agar meletakkan sub program sesuai dengan urutan pemanggilannya.

3.2. Function

Berikutnya mari kita bermain dengan function. Kita akan membuat sebuah function untuk menjumlahkan kedua buah nilai integer dan mengembalikan hasil penjumlahan tersebut:

```
Function Tambah(ByVal A As Integer, ByVal B As Integer) As Integer
    Return A + B
End Function
```

Misalnya kita hendak menggunakan function ini untuk menjumlahkan isi kedua textbox seperti pada contoh di topik ke- 3, maka kita dapat menggunakan perintah:

```
Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
    txtC.Text = Tambah(CInt(txtA.Text), CInt(txtB.Text))
End Sub
```

Pada code di atas digunakan CInt(txtA.text) yang berguna untuk merubah isi dari textbox txtA menjadi tipe data Integer. Ketika memanggil sub program harus kita pastikan bahwa tipe data dari variabel yang kita isikan harus sama dengan tipe data dari parameter sub program tersebut.

Berikutnya mari kita coba membuat fungsi rekursif, yaitu fungsi yang memanggil dirinya sendiri. Kasus yang akan digunakan adalah bagaimana menghitung faktorial dari sebuah bilangan integer. Cara yang umum digunakan adalah:

```
Function Faktorial(ByVal N As Integer) As Integer
    Dim Temp As Integer = N
    For i = N - 1 To 1 Step -1
        Temp = Temp * i
    Next
    Return Temp
End Function
```

Kita menggunakan variabel Temp sebagai tempat penampungan sementara untuk perkalian di dalam faktorial, untuk menghemat perulangan maka Temp = N dan kita mulai perulangan dari i = N -1, meskipun apabila kita hendak lebih menghemat maka dapat dilakukan For i = N – 1 To 2. Guna dari Step -1 adalah sebagai perulangan mundur atau kurang 1 atas nilai i.

Kini mari kita buat dalam bentuk rekursif:

```
Function Faktorial(ByVal N As Integer) As Integer
    If N > 1 Then
        Return N * Faktorial(N - 1)
    Else
        Return 1
    End If
End Function
```

Fungsi Faktorial akan memanggil dirinya sendiri selama nilai N > 1 dan dikalikan dengan N. Dengan kata lain maka akan dicapai suatu kedalaman hingga N = 1 setelah itu akan dikembalikan lagi hasil dari Faktorial tersebut “ke atas” hingga mencapai “permukaan” dari pemanggilan awal.

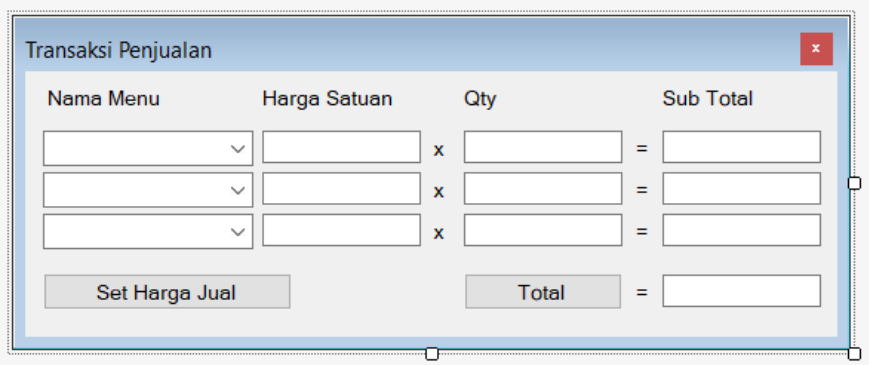
Latihan:

1. Buat procedure untuk menghasilkan deret bilangan Fibonacci.
2. Tambahkan function untuk melakukan penjumlahan pada program Penjumlahan yang terdapat di sub bab 2.1.

Bab 4. Module dan Collection

4.1. Penggunaan module untuk menyimpan variabel global

Bagaimana apabila kita hendak menggunakan 1 buah variabel yang dapat dipanggil di semua form pada sebuah solution? Jawabannya adalah dengan menggunakan variabel global, tetapi variabel global yang kita dideklarasikan di sebuah form hanya dapat dikenali di form itu, cara ini dapat dipecahkan dengan menset variabel tersebut menjadi public (meskipun sebenarnya tidaklah efisien, jauh lebih efisien menggunakan module di dalam solution untuk diakses bersama tetapi agar kita dapat membandingkan kedua cara tersebut marilah kita gunakan dulu cara ini).
Kasus berikut adalah buat program untuk aplikasi penjualan makanan di restoran secara sederhana. Form ada 2, yaitu form penjualan dan form untuk menset nilai harga jual setiap menu. Adapun design dari kedua form tersebut adalah seperti gambar 30 dan 31 di bawah ini:



Gambar 30. Tampilan design form1 (Transaksi Penjualan)

Gambar 31. Tampilan design form2 (Set Harga Jual)

Adapun nama objek di form1 adalah sebagai berikut:

1. Untuk setiap combobox di nama menu dari atas ke bawah: cb1, cb2, cb3.
2. Untuk setiap textbox di harga satuan dari atas ke bawah: txtHarga1, txtHarga2, txtHarga3.
3. Untuk setiap textbox di Qty dari atas ke bawah: txtQty1, txtQty2, txtQty3.
4. Untuk setiap textbox di Sub Total dari atas ke bawah: txtSub1, txtSub2, txtSub3.
5. Untuk tombol Set Harga Jual: Button2.
6. Untuk tombol Total: Button1.

Adapun nama objek di form2 adalah sebagai berikut:

1. Untuk setiap textbox di harga makanan dari atas ke bawah: txt1, txt2, txt3, txt4, txt5, txt6.
2. Untuk tombol Simpan: Button1.

Tips: Agar lebih mudah dibaca set property TextAlign untuk semua textbox yang berisikan angka menjadi Right.
Tips2: Ubah property Enabled untuk setiap textbox Harga Jual, Sub Total, dan Total di Form1 menjadi False guna mencegah user mengubah hasil pemanggilan harga jual maupun hasil perhitungan. Cari di internet perbedaan dari property ReadOnly dengan Enabled.

Pada project kita setting start up form adalah Form1. Apabila hendak mengisi harga jual maka kita akan mengklik tombol Set Harga Jual untuk memunculkan Form2. Karena form2 berfungsi untuk mengeset nilai dari harga jual maka kita akan mendeklarasikan variabel untuk harga setiap menu di dalam Form2 sebagai berikut:

```
1 reference
Public Class Form2
    Public NasiGoreng As Integer = 0
    Public MieGoreng As Integer = 0
    Public KwetiauGoreng As Integer = 0
    Public JusJeruk As Integer = 0
    Public KopiSusu As Integer = 0
    Public AirPutih As Integer = 0

    0 references
    Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
        NasiGoreng = CInt(txt1.Text)
        MieGoreng = CInt(txt2.Text)
        KwetiauGoreng = CInt(txt3.Text)
        JusJeruk = CInt(txt4.Text)
        KopiSusu = CInt(txt5.Text)
        AirPutih = CInt(txt6.Text)
        MessageBox.Show("Harga Jual telah disimpan")
    End Sub

    0 references
    Private Sub Form2_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        txt1.Text = 0
        txt2.Text = 0
        txt3.Text = 0
        txt4.Text = 0
        txt5.Text = 0
        txt6.Text = 0
    End Sub
End Class
```

Di bawah Public Class Form2 langsung kita deklarasikan variabel global/ public yang akan kita gunakan. Apabila hendak dideklarasikan sebagai variabel global untuk digunakan di dalam form saja, dalam artian setiap sub program di dalam form tersebut dapat mengenali variabel ini, maka cukup digunakan perintah **Dim** dengan deklarasi **Dim NasiGoreng As Integer**. Untuk mendeklarasikan sebagai variabel global yang dapat diakses oleh semua form di dalam project yang sama maka kita menggunakan **Public**.

Ketika Button1 di Form2 kita tekan maka akan disimpan harga untuk masing-masing menu ke dalam variabel public tersebut dan akan ditampilkan pesan berhasil penyimpanan dengan menggunakan perintah **MessageBox.Show**.

Penting: setiap kali suatu tombol ditekan maka hendaknya ada perubahan yang terjadi di tampilan Form tersebut agar user mengetahui telah sukses dieksekusi. Apabila tidak ada perubahan tampilan maka hendaknya diberikan suatu pesan dengan fungsi MessageBox.

Ketika Form2 pertama kali dipanggil maka set semua nilai Harga Jual menjadi 0 guna menghindari Error ketika dilakukan penyimpanan.

Untuk coding dari Form1 adalah sebagai berikut:
Kita mulai dari ketika Form1 dipanggil maka akan dieksekusi event Form1_Load sebagai berikut:

```
Private Sub Form1_Load(sender As Object, e As EventArgs) Handles MyBase.Load
    Dim ctrl As Control
    For Each ctrl In Me.Controls
        If (ctrl.GetType() Is GetType(ComboBox)) Then
            Dim cb As ComboBox = CType(ctrl, ComboBox)
            cb.Items.Add("Nasi Goreng")
            cb.Items.Add("Mie Goreng")
            cb.Items.Add("Kwetiau Goreng")
            cb.Items.Add("Jus Jeruk")
            cb.Items.Add("Kopi Susu")
            cb.Items.Add("Air Putih")
        End If
    Next
    txtSub1.Text = 0
    txtSub2.Text = 0
    txtSub3.Text = 0
    txtTotal.Text = 0
End Sub
```

Apabila kita hendak menset satu per satu nama menu ke dalam cb1, cb2, dan cb3 maka kita harus mengulangi pemanggilan .Items.Add sebanyak 3x6 (karena ada 3 combo box dan masing-masing combo box sebanyak 6 menu). Bayangkan apabila ada 10 buah combo box dengan 10 buah menu maka coding akan menjadi semakin tidak efisien. Di sini kita dapat memanfaatkan perintah **For Each**. Perintah for each pada dasarnya sama seperti perintah for hanya saja lebih dirincikan lagi kepada pengecekan "untuk setiap" komponen, control, maupun variabel yang ada. Pada contoh di atas karena kita hendak berurusan dengan combobox maka kita dapat memilih Control. **Me.Controls** merujuk kepada semua control yang ada pada Form di mana coding tersebut dijalankan dengan kata lain Me = Form1, tetapi kita tidak boleh menuliskan Form1.Controls karena akan menyebabkan konflik pemanggilan objek. If di dalam perulangan untuk mengecek apabila control tersebut bertipe combobox maka isi control tersebut dengan menu yang kita inginkan.

Untuk menampilkan Form2 (untuk set harga jual) apabila tombol Set Harga Jual pada Form1 ditekan kita dapat menggunakan coding berikut:

```
Private Sub Button2_Click(sender As Object, e As EventArgs) Handles Button2.Click
    Form2.Show()
End Sub
```

Masing-masing combo box untuk event SelectedIndexChanged kita isikan seperti ini:

```
Private Sub cb1_SelectedIndexChanged(sender As Object, e As EventArgs) Handles cb1.SelectedIndexChanged
    Select Case cb1.Text
        Case "Nasi Goreng" : txtHarga1.Text = Form2.NasiGoreng
        Case "Mie Goreng" : txtHarga1.Text = Form2.MieGoreng
        Case "Kwetiau Goreng" : txtHarga1.Text = Form2.KwetiauGoreng
        Case "Jus Jeruk" : txtHarga1.Text = Form2.JusJeruk
        Case "Kopi Susu" : txtHarga1.Text = Form2.KopiSusu
        Case "Air Putih" : txtHarga1.Text = Form2.AirPutih
    End Select
End Sub
```

Coding di atas untuk mengisikan secara otomatis Harga Jual berdasarkan Nama Menu yang dipilih pada Combo Box pertama, lengkapi event SelectedIndexChanged untuk cb2 dan cb3 serta ubah bagian-bagian yang dibutuhkan (cb1 menjadi cb2, txtHarga1 menjadi txtHarga2)

Untuk merubah nilai dari Sub Total berdasarkan perubahan Qty maka kita gunakan coding berikut pada event TextChanged untuk textbox Qty:

```
Private Sub txtQty1_TextChanged(sender As Object, e As EventArgs) Handles txtQty1.TextChanged
    txtSub1.Text = CInt(txtHarga1.Text) * CInt(txtQty1.Text)
End Sub
```

Textbox Sub Total akan dihitunga berdasarkan hasil perkalian dari textbox Harga Jual dan Qty untuk masing-masing baris yang bersangkutan. Untuk txtQty2 dan txtQty3 silahkan diketik coding di atas dengan penyesuaian seperlunya.

Untuk menghitung Harga Total maka ketika tombol Total diklik akan dipanggil coding sebagai berikut:

```
Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
    txtTotal.Text = CInt(txtSub1.Text) + CInt(txtSub2.Text) + CInt(txtSub3.Text)
End Sub
```

Latihan:

1. Buat pengecekan terhadap Form1 (Form Transaksi Penjualan) agar tidak boleh ada menu yang sama dipilih lebih dari sekali.
2. Sempurnakan program sehingga apabila Harga Jual sudah pernah disimpan maka ketika tombol Set Harga Jual di Form1 ditekan maka Form2 akan muncul lengkap dengan Harga Jual yang sudah disimpan.

4.2. Array

Untuk topik mengenai array kali ini kita akan menggunakan contoh yang telah ada pada sub bab 3.1. Sebelumnya kita menggunakan 6 buah variabel untuk menyimpan harga setiap menu. Dengan memanfaatkan array maka kita cukup menggunakan 1 buah variabel untuk menyimpan harga menu. Selain itu untuk nama menu juga kita akan menggunakan 1 buah variabel array of string sebagai tempat penyimpanan. Kelebihannya lainnya adalah kita dapat menambahkan menu ketika program dijalankan.
Untuk menggunakan array maka kita dapat mendeklarasikan sebagai berikut:

```
Public Class Form2
    Public NamaMenu() As String = {"Nasi Goreng", "Mie Goreng", "Kwetiau Goreng", "Jus Jeruk", "Kopi Susu", "Air Putih"}
    Public HargaMenu() As String
    Public BanyakMenu As Integer = 0
```

Perhatian pada contoh di atas deklarasi dilakukan di form 2 (silahkan baca kembali topik ke-7 apabila anda lupa). Contoh di atas juga mendeklarasikan array secara dinamis artinya banyak kamar pada array tersebut dapat berubah ketika program berjalan. Untuk menggunakan array bertipe statis kita dapat menggunakan program seperti berikut:

```
Public Class Form2
    Public NamaMenu(6) As String
    Public HargaMenu(6) As String
    Public BanyakMenu As Integer = 0
```

Penting: Menggunakan array bertipe statis mengharuskan pengisian nilai array secara terpisah dengan pendeklrasiannya.

Hal ini berarti kita harus mengisi manual NamaMenu(0) = “Nasi Goreng”, NamaMenu(1) = “Mie Goreng” dan seterusnya. Pada program kali ini akan digunakan contoh deklarasi yang paling atas.
Variabel BanyakMenu merupakan berapa banyak menu yang tersedia saat ini, variabel ini belum kita gunakan di contoh pembahasan kali ini.

Dengan menggunakan array NamaMenu maka kita dapat menset nama setiap label pada form 2 dengan program seperti berikut ini:

```
Private Sub Form2_Load(sender As Object, e As EventArgs) Handles MyBase.Load
    ReDim HargaMenu(6)
    Dim ctrl As Control
    Dim i As Integer = 0
    For Each ctrl In Me.Controls
        If (ctrl.GetType() Is GetType(Label)) Then
            Dim lb As Label = CType(ctrl, Label)
            lb.Text = NamaMenu(i)
            i = i + 1
        End If
    Next
    txt1.Text = 0
    txt2.Text = 0
    txt3.Text = 0
    txt4.Text = 0
    txt5.Text = 0
    txt6.Text = 0
End Sub
```

Pada baris paling atas dapat kita lihat perintah ReDim HargaMenu(6), ini berarti kita menset jumlah kamar untuk array HargaMenu menjadi 6. Untuk merubah banyak kamar dan mempertahankan nilai yang telah ada kita dapat menggunakan perintah ReDim Preserve HargaMenu(6). Misalkan kita menambah jumlah menu 2 lagi maka kita dapat menggunakan variabel BanyakMenu = 8 lalu diikuti dengan ReDim Preserve HargaMenu(BanyakMenu).
Perintah for each akan dijalankan untuk memberikan nama setiap label sesuai dengan isi dari array NamaMenu. Meskipun begitu kita program dijalankan maka tampilan nama menu di form 2 menjadi seperti ini:

Gambar 32. Tampilan hasil run form2 (Set Harga Jual)

Tampilan menu akan terbalik dikarenakan perintah for each control secara default akan mengakses dari control paling akhir. Untuk membaliknya kita dapat menggunakan beberapa cara, 2 diantaranya adalah sebagai berikut:

Cara pertama yang paling gampang adalah kita membalik urutan pemanggilan array menjadi:

```
Dim i As Integer = 5
For Each ctrl In Me.Controls
    If (ctrl.GetType() Is GetType(Label)) Then
        Dim lb As Label = CType(ctrl, Label)
        lb.Text = NamaMenu(i)
        i = i - 1
    End If
Next
```

Cara kedua yang lebih elegan adalah seperti ini:

```
Dim i As Integer = 0
For Each ctrl As Label In Me.Controls.OfType(Of Label).OrderBy(Function(x) x.Top)
    ctrl.Text = NamaMenu(i)
    i = i + 1
Next
```

Cara ini jauh lebih efisien dibandingkan cara yang sebelumnya dikarenakan pada perintah for each langsung difilter menjadi tipe control tanpa perlu menggunakan perintah if lagi di dalam loopingnya.
Untuk menambahkan nama menu di Form 1 pada combobox kita dapat melakukan hal seperti ini:

```
Private Sub Form1_Load(sender As Object, e As EventArgs) Handles MyBase.Load
    For Each ctrl As ComboBox In Me.Controls.OfType(Of ComboBox)
        For i = 0 To 5
            ctrl.Items.Add(Form2.NamaMenu(i))
        Next
    Next
    txtSub1.Text = 0
    txtSub2.Text = 0
    txtSub3.Text = 0
    txtTotal.Text = 0
End Sub
```

Jauh lebih simpel dan elegan dibandingkan program sebelumnya bukan? Dengan memanfaatkan array dan for each ctrl kita dapat bekerja dengan lebih efisien dan cepat. Selain itu program yang kita tulis juga menjadi lebih rapi.

Latihan:

1. Lengkapi program pada topik sub bab 3.1. dengan array dan sub program sehingga menjadi lebih elegan dan efisien.
2. Modifikasi program yang telah dipelajari sehingga dapat menambah menu baru.

4.3. List

Pada sub bab 4.2. kita telah mempelajari tentang array untuk menyimpan sekumpulan nilai. Selain dengan tipe data array kita juga dapat menggunakan tipe data list. Sama halnya dengan array, list juga hanya dapat menyimpan kumpulan nilai dengan tipe data yang sama, berikut adalah contoh deklarasi dari list yang memuat kumpulan nilai bertipe string:

```
Public Class Form2

    Public NamaMenuList As New List(Of String)()
```

Untuk mengisi nilai ke dalam list kita dapat menggunakan perintah sebagai berikut:

```
Private Sub Form2_Load(sender As Object, e As EventArgs) Handles MyBase.Load
    NamaMenuList.Add("Nasi Goreng")
    NamaMenuList.Add("Mie Goreng")
    NamaMenuList.Add("Jus Jeruk")
```

List memiliki index, artinya item yang pertama diadd akan menempati index ke-0, dan seterusnya. Pada contoh di atas item “Nasi Goreng” menempati list ke-0, “Mie Goreng” pada index ke-1, dan “Jus Jeruk” pada index ke-2. Misalkan kita hendak menyisipkan menu “Kwetiau Goreng” di antara “Mie Goreng” dan “Jus Jeruk”, maka kita melakukannya dengan perintah:

```
NamaMenuList.Insert(2, "Kwetiau Goreng")
```

Untuk mengeluarkan atau menghapus item dari list kita dapat menggunakan perintah Remove seperti berikut:

```
NamaMenuList.Remove("Kwetiau Goreng")
```

Berikutnya bagaimana apabila kita hendak memasukkan sekumpulan nilai langsung ke dalam list dan kita tidak mau mengulang-ulang perintah Add? Kita dapat menggunakan perintah yang hampir sama dengan memasukkan sekumpulan nilai ke dalam array seperti ini:

```
Public NamaMenuList As New List(Of String)() From {"Nasi Goreng", "Mie Goreng", "Kwetiau Goreng" _
                                                   , "Jus Jeruk", "Kopi Susu", "Air Putih"}
```

Tanda _ di bagian belakang baris pertama menandakan bahwa baris tersebut belum selesai dan dilanjutkan ke baris berikutnya.

Lalu bagaimana seandainya kita sudah memiliki array dengan kumpulan nilai dan hendak kita masukkan ke dalam list? Hal ini dengan mudah dapat kita lakukan dengan perintah berikut ini:

```
Public NamaMenu() As String = {"Nasi Goreng", "Mie Goreng", "Kwetiau Goreng",
                               "Jus Jeruk", "Kopi Susu", "Air Putih"}
Public NamaMenuList As New List(Of String)(NamaMenu)
```

Dengan menggunakan list kita dengan mudah dapat menyisipkan item atau nilai ke urutan/ index tertentu, selain itu kita juga dapat menghapus item tertentu di dalam list tersebut. Kita juga dapat mengecek dengan mudah apakah suatu item ada di dalam list atau tidak dengan perintah berikut:

```
If (NamaMenuList.Contains("Nasi Goreng")) Then

End If
```

Di dalam if tersebut kita dapat memasukkan perintah yang akan dikerjakan apabila terdapat item dengan “Nasi Goreng” di dalam list.

Misalkan kita hendak merubah coding pada sub bab 4.2. dari array menjadi list, maka kita cukup mengganti perintah di dalam event Form Load menjadi seperti ini:

```
Private Sub Form2_Load(sender As Object, e As EventArgs) Handles MyBase.Load
    Dim i As Integer = 0
    For Each ctrl In Me.Controls.OfType(Of Label).OrderBy(Function(x) x.Top)
        ctrl.Text = NamaMenuList(i)
        i = i + 1
    Next
    txt1.Text = 0
    txt2.Text = 0
    txt3.Text = 0
    txt4.Text = 0
    txt5.Text = 0
    txt6.Text = 0
End Sub
```

Kita cukup mengganti perintah yang di-highlight saja. Lalu bagaimana apabila kita hendak mengubah Harga pada program di topik ke-9 menjadi list juga? Hal itu mudah dilakukan, cukup ikuti langkah-langkah untuk membuat List NamaMenu di atas.

Meskipun list lebih banyak kelebihan dibandingkan array, tetapi dalam beberapa kasus tetap saja memiliki sejumlah kekurangan. Misalnya pada kasus di topik ke-9 kita hendak memetakan harga menu ke dalam nama menu. Dengan menggunakan array maupun list kita butuh 2 buah array ataupun 2 buah list dengan relasi 1 ke 1. Apabila kita hendak menggabungkan 2 buah nilai ke dalam sebuah index, maka kita dapat menggunakan tipe data Dictionary yang akan dibahas pada pertemuan selanjutnya.

Latihan: ubah penggunaan array menjadi list pada program di sub bab 4.2.

4.4. Dictionary

Pertemuan kali ini akan membahas mengenai tipe data Dictionary. Sama halnya dengan array dan list, dictionary juga merupakan kumpulan data atau yang diistilahkan dengan collection. Sedikit berbeda dengan array dan list, pada dictionary kita juga menentukan kunci primer terhadap data di dalamnya. Maksud dari kunci primer adalah kunci unik yang digunakan sebagai penanda pada record. Record juga dapat disebut dengan 1 "baris" data. Misal kita memiliki data pegawai, maka nomor induk pegawai, nama pegawai, alamat, dan nomor telepon dari setiap individu merupakan satu record.

Mari kita buat program seperti di bawah ini:

```
Private Sub Form2_Load(sender As Object, e As EventArgs) Handles MyBase.Load
    Dim NamaMenuDict As New Dictionary(Of String, Decimal)
    NamaMenuDict.Add("Nasi Goreng", 20000)
    NamaMenuDict.Add("Nasi Goreng", 22000)
    NamaMenuDict.Add("Mie Goreng", 12000)
    NamaMenuDict.Add("Kwetiau Goreng", 12000)
    NamaMenuDict.Add("Jus Jeruk", 11000)
    NamaMenuDict.Add("Kopi Susu", 10000)
    NamaMenuDict.Add("Air Putih", 1000)
End Sub
```

Pada program di atas kita mendeklarasikan variabel bertipe data dictionary di mana kita menggunakan key bertipe string dan nilai bertipe decimal. Decimal adalah tipe data yang umum digunakan untuk menyimpan nilai uang atau currency. Apabila anda hendak memanggil variabel NamaMenuDict di form lain, maka letakkan deklarasi NamaMenuDict di bagian bawah Class Form dan setting menjadi public.

Untuk menambah nilai ke dalam variabel dictionary kita menggunakan perintah .Add yang diikuti dengan nilai kunci dan nilai data yang ingin dimasukkan. Pada contoh di atas terdapat 2 buah baris dengan nilai kunci yang sama, yaitu "Nasi Goreng" sehingga ketika program dijalankan akan muncul pesan kesalahan bahwa key yang sama digunakan lebih dari 1 kali. Silahkan hapus salah satu baris yang berisikan key "Nasi Goreng"

Berikutnya untuk mengambil nilai dari dictionary berdasarkan key dapat kita lakukan dengan perintah berikut:

```
Dim Harga As Decimal
Harga = NamaMenuDict("Nasi Goreng").ToString
```

Bagaimana untuk kasus sebaliknya? Kita hendak mengambil kunci berdasarkan value, dalam kasus ini kita hendak mengetahui nama menu dengan harga 11000 misalnya, kita dapat menggunakan perintah .ContainsValue seperti program di bawah ini:

```
If NamaMenuDict.ContainsValue(11000) Then
    MessageBox.Show(NamaMenuDict.ContainsValue(11000).ToString)
End If
```

Apabila kita jalankan program di atas maka akan ditampilkan “True”. Hal ini dikarenakan output dari perintah .ContainsValue adalah Boolean, yang menandakan apakah nilai tersebut ditemukan atau tidak. Untuk menampilkan nilai kunci yang mengandung value 11000 kita dapat melakukan pengecekan satu per satu seperti program berikut ini:

```
For Each pair As KeyValuePair(Of String, Decimal) In NamaMenuDict
    If pair.Value = 11000 Then
        MessageBox.Show("Menu dengan harga " & pair.Value & " adalah " & pair.Key)
    End If
Next
```

Dengan menggunakan cara di atas maka akan ditampilkan semua kunci dengan value 11000. Untuk lebih jelasnya silahkan ubah 11000 dengan 12000 dan jalankan program tersebut lagi.

Bagaimana apabila kita hendak mengkopi data di dalam dictionary ke dalam array? Kita dapat melakukannya dengan cara berikut ini:

```
Dim test(NamaMenuDict.Count) As String
Dim i As Integer
For Each pair As KeyValuePair(Of String, Decimal) In NamaMenuDict
    test(i) = pair.Key
    i = i + 1
Next

For i = 0 To NamaMenuDict.Count - 1
    MessageBox.Show(test(i))
Next
```

Apabila kita hendak membuat pasangan array nama menu dan harganya maka kita ubah program tersebut menjadi seperti ini:

```
Dim NamaMenu(NamaMenuDict.Count) As String
Dim HargaMenu(NamaMenuDict.Count) As Decimal
Dim i As Integer
For Each pair As KeyValuePair(Of String, Decimal) In NamaMenuDict
    NamaMenu(i) = pair.Key
    HargaMenu(i) = pair.Value
    i = i + 1
Next

For i = 0 To NamaMenuDict.Count - 1
    MessageBox.Show(NamaMenu(i) & " harganya " & HargaMenu(i))
Next
```

Untuk memasukkan nilai dictionary tersebut ke dalam pasangan list NamaMenuList dan HargaMenuList kita cukup mengubah program di atas menjadi seperti ini:

```
Dim NamaMenuList As New List(Of String)
Dim HargaMenuList As New List(Of Decimal)
For Each pair As KeyValuePair(Of String, Decimal) In NamaMenuDict
    NamaMenuList.Add(pair.Key)
    HargaMenuList.Add(pair.Value)
Next

For i = 0 To NamaMenuDict.Count - 1
    MessageBox.Show(NamaMenuList(i) & " harganya " & HargaMenuList(i))
Next
```

Latihan: buat program untuk memasukkan nilai dari pasangan array NamaMenu dan HargaMenu ke dalam variabel dictionary.

Bab 5. Permainan Tic Tac Toe

Bab ini akan membahas mengenai bagaimana membuat permainan Tic Tac Toe dengan menggunakan pemrograman visual. Permainana Tic Tac Toe merupakan permainan yang sederhana dan mudah dimainkan oleh siapa saja. Sebelum masuk ke program untuk permainan tic tac toe mari kita bahas sedikit mengenai konsep permainan ini:

O	X	
	O	
X		O

Gambar 33. Konsep Arena Permainan Tic Tac Toe

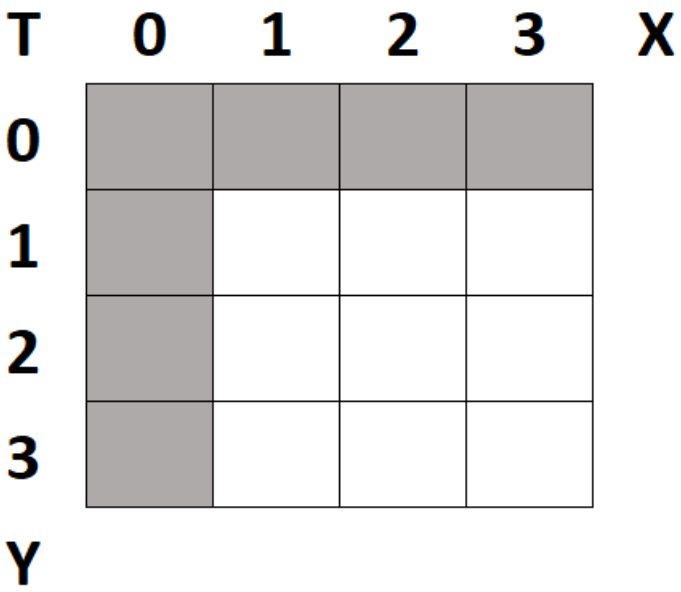
Pada gambar di atas adalah arena permainan game Tic Tac Toe, di mana pemain yang menjalankan O menang. Untuk membuat permainan ini dengan bahasa C kita akan menggunakan konsep array 2 dimensi seperti pertemuan minggu lalu, bedanya karena kali ini kita bermain dengan kordinat di layar maka konsep antara baris dan kolom kita balik, perhatikan gambar di bawah untuk jelasnya:

	1	2	3	X
1	O	X		
2		O		
3	X		O	
Y				

	1	2	3	X
1	1,1	2,1	3,1	
2	1,2	2,2	3,2	
3	1,3	2,3	3,3	
Y				

Gambar 34. Konsep antara baris dan kolom di permainan

O di sudut kiri atas menempati posisi (1,1), X di samping kanannya menempati posisi (2,1), dan seterusnya. Dengan menggunakan kordinat sumbu XY maka kita menuliskan kolom dulu baru baris. Array yang akan kita gunakan misalnya adalah **int T[4][4]**, maka di dalam memory dapat kita gambarkan array T adalah sebagai berikut:

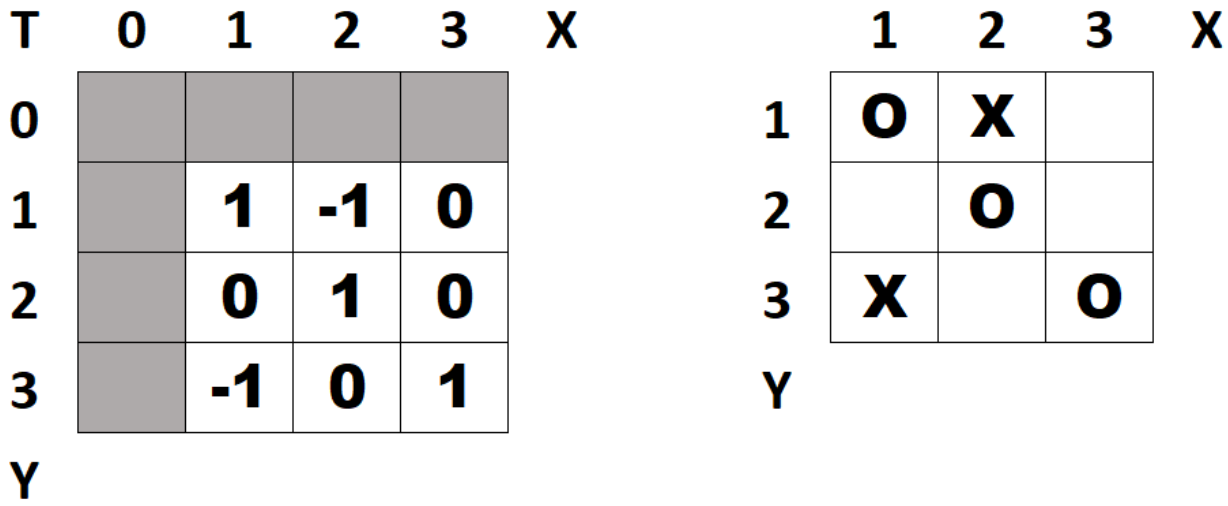


Gambar 35. Konsep array T untuk Tic Tac Toe

Untuk baris ke-0 dan kolom ke-0 tidak akan kita gunakan untuk memudahkan pemetaan. Selanjutnya nilai pada setiap kotak akan mengikuti ketentuan sebagai berikut:

- 0 untuk kotak kosong
- 1 untuk kotak berisi O
- -1 untuk kotak berisi X

Maka berdasarkan aturan tersebut, gambar di awal dapat kita petakan ke dalam array T sebagai berikut:

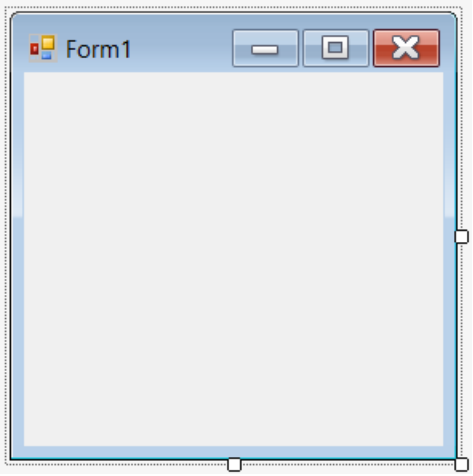


Gambar 36. Pemetaan array T ke papan permainan

Untuk menandakan giliran pemain O atau X kita akan menggunakan variabel **turn** yang bertipe integer. Apabila turn = 1 maka giliran pemain O dan turn = -1 adalah giliran pemain X. Untuk mengecek siapa yang menang kita dapat menjumlahkan setiap baris, setiap kolom, dan 2 buah diagonal untuk mengecek apakah ada yang berjumlah 3 atau -3. Apabila ada yang berjumlah 3 maka O menang, sedangkan -3

adalah X menang. Pada gambar di atas dapat dilihat apabila kita menjumlahkan kotak di posisi diagonal (1,1) + (2,2) + (3,3) akan menghasilkan 3, maka O yang menang.

Sekarang mari kita tuangkan konsep tersebut ke dalam pemrograman visual. Pertama-tama design sebuah form kosong dengan ukuran 280,280 seperti pada gambar berikut:



Gambar 37. Tampilan design form1 (Arena Permainan Tic Tac Toe)

Selanjutnya kita akan mendeklarasikan 3 buah variable public pada form tersebut:

```
Public Class Form1
    Dim picT(4, 4) As PictureBox
    Dim TB(4, 4) As Integer
    Dim turn As Integer
```

Variabel picT berguna untuk menggambarkan pictureBox sebanyak 9 buah di form sebagai arena permainan. Kita akan menggunakan array 2 dimensi untuk hal ini. Ingat, array dimulai dari 0, karena itu apabila kita mendeklarasikan picT(4,4) maka kamar yang tersedia adalah picT(0,0) hingga picT(3,3). Seperti halnya pada permainan Tic Tac Toe terdahulu, kita tidak akan menggunakan baris ke-0 dan kolom ke-0 dari array tersebut melainkan langsung dari kamar di posisi (1,1) sehingga pemetaan antara pictureBox dengan array yang menyimpan nilai turn pemain di belakang layar adalah 1 ke 1.
Variabel TB sama dengan variabel T pada konsep di atas yang berguna untuk menyimpan informasi di setiap posisi arena permainan.
Variabel turn berguna untuk menandakan giliran pemain yang sedang berlangsung dan untuk mengisi variabel TB dengan nilai turn untuk mengecek apakah sudah ada pemain yang menang. Dalam contoh kali ini turn = 1 adalah giliran pemain biru dan turn = -1 adalah giliran pemain merah.

Selanjutnya kita akan membuat sub program DrawTable yang berguna untuk menggambar arena permainan pada form, menambah event handler Click untuk setiap array picT, serta untuk menset nilai awal dari TB. Program dari DrawTable adalah sebagai berikut:

```
Public Sub DrawTable()
    For i = 1 To 3
        For j = 1 To 3
            picT(j, i) = New PictureBox
            picT(j, i).Size = New Point(40, 40)
            picT(j, i).BorderStyle = BorderStyle.FixedSingle
            picT(j, i).Location = New Point((picT(j, i).Width * j), (picT(j, i).Height * i))
            picT(j, i).BackColor = Color.White
            picT(j, i).Name = j.ToString + "," + i.ToString

            Me.Controls.Add(picT(j, i))
            AddHandler picT(j, i).Click, AddressOf Table_Click

            TB(j, i) = 0
        Next
    Next
End Sub
```

Mari kita telaah satu per satu. Kita menggunakan nested loop sebanyak 3x3 untuk menggambarkan picT di form. Di dalam program tersebut kita langsung menset ukuran dari picT (Size), mode pinggirannya (BorderStyle), lokasi dari setiap picT (Location), warna awal dari arena permainan (BackColor), serta nama dari setiap picT (Name). **Perhatikan: kita menggunakan j,i dikarenakan untuk kordinat sumbu layar adalah kolom dahulu baru baris atau yang biasa disebut dengan (X,Y).**
Pemberian nama untuk setiap picT adalah nomor baris, nomor kolom. picT yang berada di posisi 1,1 akan bernama “1,1”, picT di posisi 2,1 akan bernama “2,1”, dan seterusnya. Hal ini penting karena ketika kita memanggil event Table_Click kita harus mengecek apakah picT tersebut sudah ada isinya atau belum.

Setelah menset semua property pada picT, langkah selanjutnya adalah menampilkan picT tersebut ke form dengan perintah Me.Controls.Add.

Langkah selanjutnya adalah menambahkan event Click untuk setiap picT dengan cara AddHandler lalu setelah AddressOf kita menuliskan sub program yang hendak dijalankan setiap picT diklik.

TB(j,i) = 0 berguna untuk menset nilai awal dari TB menjadi 0 untuk menandakan area ini belum diisi oleh pemain manapun.

Berikutnya kita akan menuliskan program untuk Table_Click seperti berikut ini:

```
Private Sub Table_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
    Dim PB As PictureBox = sender
    Dim col As Integer = CInt(PB.Name.ElementAt(0).ToString)
    Dim row As Integer = CInt(PB.Name.ElementAt(2).ToString)

    If TB(col, row) = 0 Then
        If turn = 1 Then
            PB.BackColor = Color.Blue
            TB(col, row) = 1
        Else
            PB.BackColor = Color.Red
            TB(col, row) = -1
        End If
        turn = -turn
        If Win() = 3 Then
            MessageBox.Show("Blue Win")
        ElseIf Win() = -3 Then
            MessageBox.Show("Red Win")
        End If
    End If
End Sub
```

Pertama-tama kita perlu mengetahui picT mana yang diklik oleh user agar kita dapat membandingkan apakah TB di lokasi yang sama dengan picT tersebut sudah ada nilai turnnya atau belum. Untuk lebih gampang kita akan menggunakan variabel dummy bernama PB yang bertipe pictureBox juga di mana variabel ini kita isikan dengan sender yang berupa object pictureBox, tepatnya picT. Selanjutnya kita akan mengambil nilai kolom (col) dari posisi ke-0 nama dari PB dan nilai baris (row) dari posisi ke-2 nama dari PB. Ingat, nama picT misalnya “2,1” berarti adalah kolom ke-2 baris ke-1.

Selanjutnya kita akan mengecek apakah nilai TB di lokasi tersebut = 0 yang berarti belum diisi, apabila belum terisi maka kita cek turn saat ini, 1 untuk pemain biru dan -1 untuk pemain merah. Selanjutnya kita akan ubah warna PB sesuai dengan giliran pemain tersebut dan kita akan isikan turn ke dalam lokasi TB yang sama dengan lokasi picT yang diklik.

Langkah selanjutnya adalah untuk memanggil fungsi Win() yang berguna untuk mengecek jumlah turn untuk 8 kemungkinan di dalam variabel TB, apabila nilai Win = 3 maka pemain biru menang. Bila Win = -3 maka pemain merah yang menang.

Program untuk mengecek apakah sudah ada pemain yang menang dapat kita tuliskan seperti ini:

```
Function Win() As Integer
    Dim Temp As Integer = 0
    For i = 1 To 3
        Temp = 0
        For j = 1 To 3
            Temp += TB(i, j)
        Next
        If Math.Abs(Temp) = 3 Then Return Temp
    Next
    For i = 1 To 3
        Temp = 0
        For j = 1 To 3
            Temp += TB(j, i)
        Next
        If Math.Abs(Temp) = 3 Then Return Temp
    Next
    Temp = 0
    For i = 1 To 3
        Temp += TB(i, i)
    Next
    If Math.Abs(Temp) = 3 Then Return Temp
    Temp = 0
    For i = 1 To 3
        Temp += TB(i, 4 - i)
    Next
    If Math.Abs(Temp) = 3 Then Return Temp
    Return 0
End Function
```

Secara garis besar program di atas sama seperti program pengecek kemenangan yang kita tuliskan dalam bahasa C di mata kuliah sebelumnya. Akan dilakukan pengecekan per baris, per kolom, dan diagonal untuk menemukan apakah sudah pemain yang menang.

Langkah terakhir adalah kita menambahkan program berikut di event Form_Load:

```
Private Sub Form1_Load(sender As Object, e As EventArgs) Handles MyBase.Load
    DrawTable()
    turn = 1
End Sub
```

Program di atas berguna untuk menggambar arena permainan pada form dan selanjutnya menset nilai turn = 1 agar pemain biru jalan duluan.

Latihan: lengkapi program untuk mengecek apakah permainan berakhir dengan seri.

Bab 6. Secangkir Kopi

Selamat, anda telah menyelesaikan membaca buku dasar-dasar pemrograman visual ini, pada bab ini kita akan merangkum mengenai topik apa saja yang sudah kita pelajari.

Berbeda dengan bahasa pemrograman berbasis teks, seperti C/ C++, pada bahasa pemrograman visual kita menggunakan bahasa VB .NET. Perbedaan yang paling mendasar tentu pada konsepnya pembuatan aplikasi atau layoutnya. Dengan bahasa C/ C++ kita hanya perlu fokus pada pemrograman saja sedangkan pada bahasa VB .NET kita juga perlu fokus pada design interfacenya. Setiap object yang kita tambahkan ke form adalah berupa class object dan disebut dengan istilah controls atau components. Perbedaan antara controls dan components adalah sebagai berikut: controls dapat kita lihat ketika aplikasi kita jalankan, misalnya textbox, button, form, dan sebagainya. Sedangkan components tidak dapat kita lihat seperti timer dan openfiledialog. Semua controls adalah components pada dasarnya tetapi tidak sebaliknya.

Setiap component memiliki property atau sifat dari component. Apabila anda sudah belajar mengenai Object Oriented Programming atau setidaknya sudah pernah belajar mengenai Class maka anda akan lebih mudah memahami mengenai konsep component pada bahasa VB .NET.
Property yang terpenting untuk setiap component adalah Name atau nama dari component tersebut. Penulisan Name biasanya diawali dengan tipe component, misal btn untuk Button, txt untuk TextBox, dan sebagainya. Sebaiknya digunakan nama yang mewakili fungsi dari component tersebut misalnya: btnSimpan, txtNamaPemilik, dan seterusnya. Mengenai object atau component ini dapat kita baca kembali bab 2 mengenai event dan controls.

Dengan menggunakan solution explorer dan project, kita dapat dengan mudah menambahkan dan memanage form ataupun file di dalam solution tersebut. Apabila kita hendak bermain dengan banyak form sekaligus di dalam satu solution maka disarankan untuk membuat form induk MDIForm seperti yang telah dijelaskan pada sub bab 1.3. sehingga pengaksesan setiap form menjadi lebih mudah, terkategori, dan efisien.

Setiap component memiliki event, yaitu kondisi atau keadaan yang terjadi terhadap component itu. Misalnya button memiliki event Click yang akan menjalankan program di dalamnya apabila tombol tersebut kita tekan. Untuk penjelasan lebih lengkap mengenai ini dapat dibaca kembali pada topik sub bab 2.1.

Setiap variabel (bahkan component) dapat dibuat sebagai public maupun private. Public berarti variabel ini dapat diakses di luar form tempat dia dideklarasikan, sedangkan private berarti varibel tersebut hanya dapat diakses di mana dia dideklarasikan. Untuk lebih lengkapnya silahkan rujuk kembali sub bab 4.1. **Note: silahkan cari tahu mengenai module untuk penggunaan variabel public yang lebih efisien.**

Agar kita tidak tidak perlu menuliskan program yang sama secara berulang-ulang pada aplikasi, kita dapat menggunakan sub program yang berupa procedure maupun function. Bab 3 membahas hal ini secara gamblang.

Apabila kita hendak menyimpan sejumlah nilai ke dalam sebuah variabel maka kita dapat menggunakan tipe data collections. Tipe data collections dapat berupa Array (sub bab 4.2.), List (sub bab 4.3.), dan Dictionary (sub bab 4.4.).

Kita juga dapat membuat permainan sederhana dengan mudah dengan menggunakan bahasa VB .NET ini seperti halnya yang telah dijelaskan pada bab 5 tentang game Tic Tac Toe.

Akhir kata setelah melalui mata kuliah ini diharapkan kita dapat membuat aplikasi tingkat dasar hingga menengah dengan menggunakan bahasa VB .NET.

Latihan: silahkan modifikasi dan kembangkan program-program yang telah dipelajari sebelumnya guna meningkatkan pemahaman mengenai pemrograman visual.

Daftar Pustaka

Books, G. (2018). *Visual Basic .NET Notes for Professionals.* GoalKicker Books.

Gaddis, T., & Irvine, K. R. (2020). *Starting Out With Visual Basic* (8th ed.). Florida: Pearson.

Grey, D. (2021). *Visual Basic Crash Course, The Ultimate Beginner's Course to Learn Visual Programming in Under 12 Hours.*

Schneider, D. I. (2020). *An Introduction to Programming Using Visual Basic* (11th ed.). Maryland: Pearson.

Yao, R. (2015). *Visual Basic Programming, For Beginners Learn Coding Fast!* Tutorial eBook & Book.

Zak, D. (2018). *Programming with Microsoft Visual Basic 2017* (8th ed.). Boston: Cengage.

Sinopsis

Pemrograman merupakan salah satu kebutuhan dasar di jaman sekarang ini. Banyak lapangan kerja yang membutuhkan jasa programmer baik dalam skala kecil maupun besar. Salah satu ilmu yang mudah dipelajari dan diaplikasikan tapi sulit untuk dikuasai secara mendalam adalah pemrograman visual. Pemrograman visual bertujuan untuk menghasilkan aplikasi yang siap pakai dan dapat digunakan dalam bidang pekerjaan apapun.

Buku ini akan membahas mengenai pemrograman visual dengan bahasa pemrograman Microsoft Visual Basic .Net 2019. Diharapkan sebelum memulai membaca buku ini, pembaca sudah memiliki pengetahuan dasar mengenai bahasa pemrograman, seperti variabel, input, output, kondisi, hingga perulangan. Perjalanan kita akan dimulai dari pengenalan pemrograman visual, penggunaan multi forms dan penguasaan MDI Form untuk memudahkan kita dalam menangani jumlah form yang banyak dalam satu project, selanjutnya kita akan dibawa ke event dan controls, dua hal dasar yang perlu dikuasai untuk menghasilkan aplikasi berbasis pemrograman visual yang mumpuni. Setelah itu kita akan mempelajari tentang sub program untuk memudahkan kita menulis program secara efisien. Pemberhentian selanjutnya adalah bermain dengan collections, seperti array, list, dan dictionary agar aplikasi yang kita buat lebih mudah ditangani dan diolah data-datanya. Di pemberhentian akhir, untuk mengendorkan syaraf kita akan membuat game sederhana, yaitu Tic Tac Toe.

Contoh-contoh yang dibahas dalam buku ini dapat diubah sesuai dengan kebutuhan dan diaplikasikan di lapangan kerja sehingga diharapkan pembaca mendapatkan manfaat penuh setelah menyelesaikan buku ini. Cara penyajian yang sederhana yang disertai dengan gambar dan penjelasan yang lengkap serta tips dan trik membuat buku ini tidak hanya sesuai untuk kalangan akademisi, tetapi juga cocok untuk dibaca oleh kalangan awam yang tertarik di dunia pemrograman visual.