



Materi Perkuliahan

Pertemuan 05

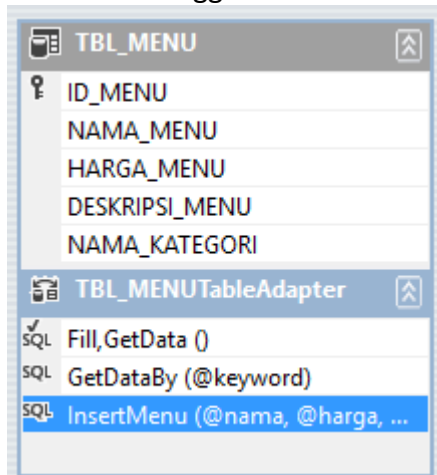
PEMROGRAMAN VISUAL

Semester Genap 2015/2016

Implementasi Add

REVIEW MINGGU LALU

Minggu lalu kita sudah membuat method untuk menginsrtkan data menu.



Selain itu kita juga sudah memanggil Method tersebut di menu control seperti ditunjukkan pada kode berikut :

```
namespace Restoran.Control
{
    class MenuControl
    {
        private TBL_MENUTableAdapter TM = new TBL_MENUTableAdapter();

        public DataTable showMenu()
        {
            return TM.GetData();
        }

        public DataTable searchMenu(string Keyword)
        {
            return TM.GetDataBy(Keyword);
        }

        public void addMenu(Menu M)
        {
            TM.InsertMenu(M.Nama, M.Harga, M.Deskripsi, M.Kategori);
        }
    }
}
```

Pada pertemuan ini kita akan melengkapi fungsi memasukkan data. Akan tetapi kita perlu memahami alur atau skenario dari proses penambahan (add) data.

ALUR SKENARIO PROSES PENAMBAHAN DATA

Alur penambahan data adalah sebagai berikut:

Saat pertama kali Form pengelolaanMenu (Form1.cs) untuk menambahkan data tekan tombol Tambah

The screenshot shows a window titled 'Pengelolaan Menu'. At the top is a search bar labeled 'Pencarian'. Below it is a table with the following data:

	ID	Nama	Harga	Deskripsi	Kategori
▶	4	kerupuk	500	kerupuk ikan	SNACK
	3	Nasi goreng	25000	enak oi	MAKANAN...
	2	Potatoes	5000	kentang goreng	SNACK
	1	Lemon Tea	5000	Minuman lemon deng...	MINUMAN

Below the table are four buttons: 'Tambah' (green), 'Ubah' (yellow), 'Hapus' (red), and 'Batal' (grey). A mouse cursor is pointing at the 'Tambah' button.

Maka akan muncul tampilan untuk menambahkan data sebagai berikut:

The screenshot shows the same 'Pengelolaan Menu' window, but with a modal form overlaid in the center. The modal form has the following fields:

- 'Nama': A text input field.
- 'Harga': A text input field.
- 'Deskripsi': A larger text input field.
- 'Kategori': A dropdown menu currently showing 'MAKANAN UTAMA'.

At the bottom of the modal are two buttons: 'Simpan' (purple) and 'Batal' (grey). The background of the main window is dimmed, and the 'Tambah' button is still visible at the bottom.

Tambahkan data yang diinginkan. Pada saat mengisi di text box harga, kita akan membatasi hanya boleh menerima inputan angka saja. Pada kondisi ini, form dibelakang dari text box search, tombol action dan datagrid dibelakang panel untuk menginput data akan di disable.

Untuk combobox Kategori akan berisi nama kategori yang diambil dari basis data.

Apabila data yang kita inputkan tidak lengkap maka akan keluar tanda peringatan berkedap kedip di setiap samping text box, dan akan memberikan penjelasan kesalahan yang terjadi seperti di bawah ini

The screenshot shows a form with the following fields: 'Nama' (filled with 'Cap cay'), 'Harga' (empty), 'Deskripsi' (filled with 'bergizi'), and 'Kategori' (a dropdown menu showing 'MAKANAN UTAMA'). Below the fields are two buttons: 'Simpan' (blue) and 'Batal' (grey). A red square with a red dot and a cursor is positioned over the 'Harga' field. A tooltip bubble next to it says 'silahkan isi Harga'. A callout box on the right points to this tooltip with the text: 'Tanda peringatan bahwa textbox harga belum di isi'.

Apabila data sudah benar, tekan tombol Simpan, maka panel pengisian data akan menghilang dan form awal akan kembali enable, selain itu data baru akan muncul di datagridview pada posisi paling atas dan akan terseleksi otomatis.

The screenshot shows the 'Pengelolaan Menu' application window. It has a search bar at the top labeled 'Pencarian'. Below it is a data grid with the following data:

	ID	Nama	Harga	Deskripsi	Kategori
▶	5	Cap cay	35000	bergizi	MAKANAN...
	4	kerupuk	500	kerupuk ikan	SNACK
	3	Nasi goreng	25000	enak oi	MAKANAN...
	2	Potatoes	5000	kentang goreng	SNACK
	1	Lemon Tea	5000	Minuman lemon deng...	MINUMAN

Below the grid are four buttons: 'Tambah' (green), 'Ubah' (yellow), 'Hapus' (red), and 'Batal' (grey).

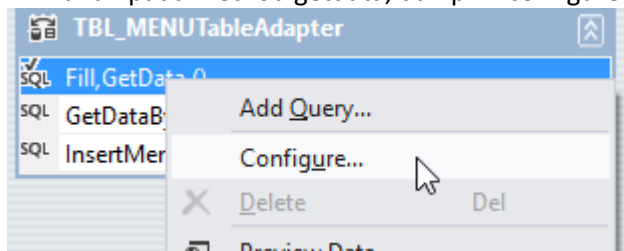
Selanjutnya kita akan memulai membuat dan menambahkan code agar program di atas berjalan sesuai dengan skenario.

Memodifikasi Fungsi getdata dan getdataby pada table Menu Adapter

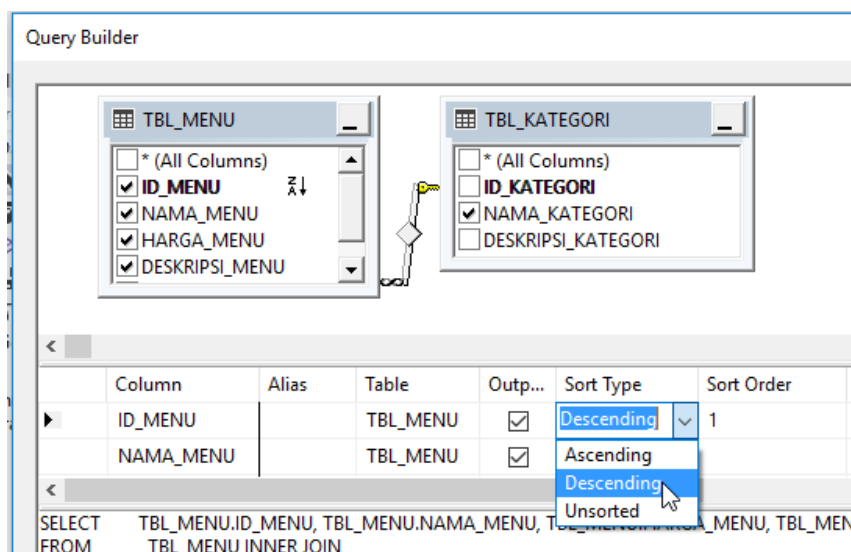
Pada skenario di atas, setiap data yang baru saja ditambahkan akan ditampilkan pada baris paling atas seperti ditunjukkan pada gambar di bawah. Untuk mewujudkannya kita harus memodifikasi fungsi getdata dan getdataby pada table menu adapter.

	ID	Nama	Harga	Deskripsi	Kategori
▶	5	Cap cay	35000	bergizi	MAKANAN...
	4	kerupuk	500	kerupuk ikan	SNACK
	3	Nasi goreng	25000	enak oi	MAKANAN...

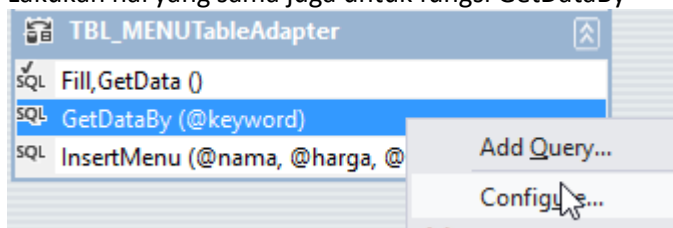
Klikkanan pada method getdata, dan pilih configure



Anda akan masuk Jendela **TableAdapter Configuration wizard**. Kemudian masuk ke menu Query Builder, dan pada ID_MENU pada pilihan Sort Type pilih sort type menjadi Descending dan pastikan sort order menjadi angka 1, kemudian klik Ok, dan finish.



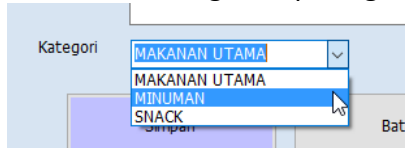
Lakukan hal yang sama juga untuk fungsi GetDataBy



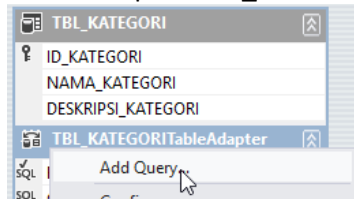
Maka jika anda jalankan, maka data yang tampilkan akan diurutkan secara descending, yang artinya data paling baru akan ditampilkan terdahulu.

Menambahkan fungsi pada TBL_KATEGORITableAdapter

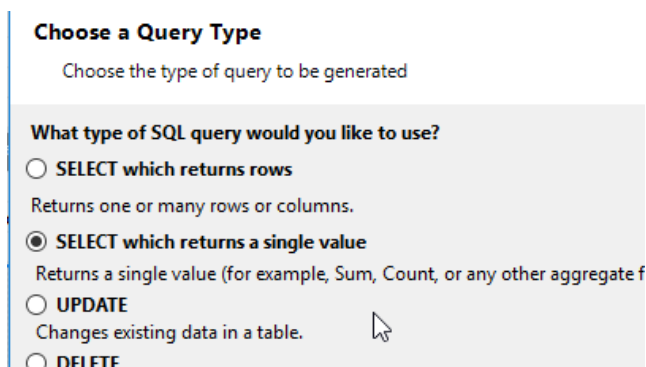
Selanjutnya kita akan menambahkan fungsi GetIdKategori yang akan dipergunakan untuk menambahkan id kategori pada saat kita menambahkan menu. Id kategori didapatkan dari combobox kategori seperti gambar di bawah ini



Klik kanan pada TBL_KATEGORITableAdapter dan pilih Add Query



Kemudian pilih SELECT which returns a single value



Selanjutnya setelah next, tambahkan query seperti contoh berikut

Specify a SQL SELECT statement

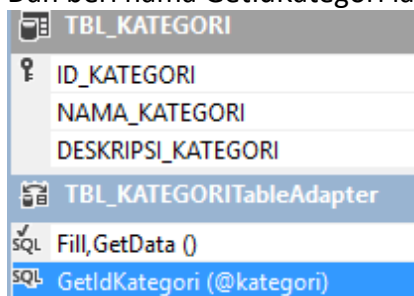
The SELECT statement will be used by the query.

Type your SQL statement or use the Query Builder to build the query.

What data should the table load?

```
SELECT ID_KATEGORI
FROM TBL_KATEGORI
WHERE (NAMA_KATEGORI = @kategori)
```

Dan beri nama GetIdKategori lalu finish



Menambahkan Fungsi getKategori dan getIdKategori pada MenuControl

Selanjutnya kita perlu menambahkan fungsi pada menuControl agar kita bisa menggunakannya untuk menampilkan data kategori ke combo box dan mendapatkan id dari data combobox Pertama pada menuControl tambahkan tbl_kategoriTableAdapter agar kita bisa menggunakan fungsi fungsi dari table adapter tersebut.

```
namespace Restoran.Control
{
    class MenuControl
    {
        private TBL_MENUTableAdapter TM = new TBL_MENUTableAdapter();
        private TBL_KATEGORITableAdapter TK = new TBL_KATEGORITableAdapter();
        ....
    }
}
```

Kemudian tambahkan pada menu control dua buah method yaitu getKategori() dan getIdKategori(string kategori) seperti contoh di bawah

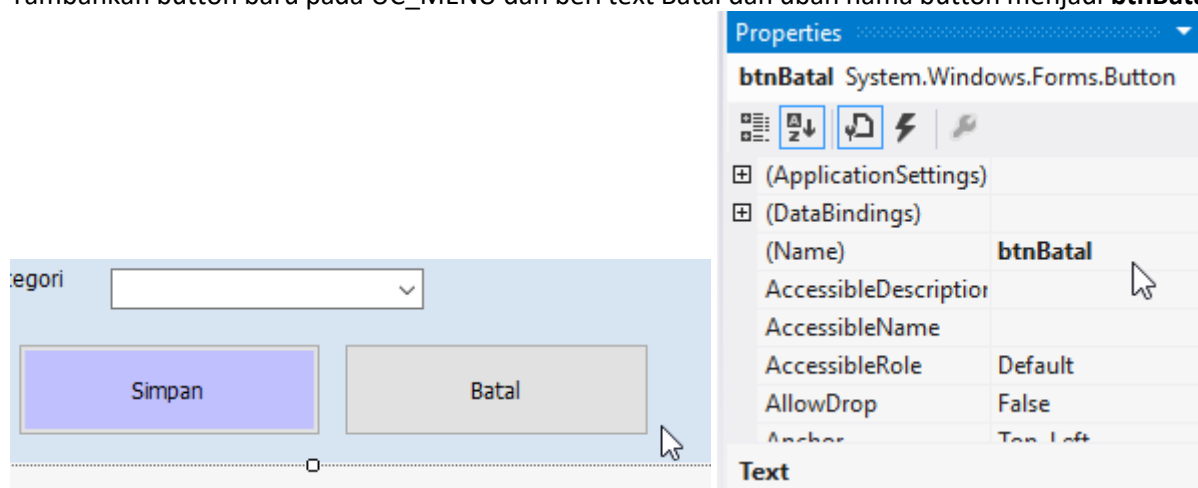
```
public DataTable getKategori()
{
    return TK.GetData();
}

public int getIdKategori(string kategori)
{
    return TK.GetIdKategori(kategori).Value;
}
```

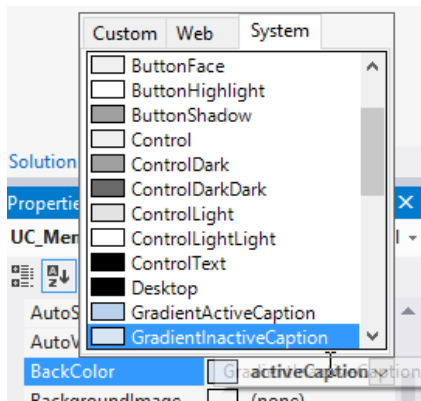
Fungsi getKategori berguna untuk memberi data pada combobox dan getIdKategori berguna untuk mendapatkan id kategori berdasarkan nama kategori

Memodifikasi tampilan dari UserControl UC_MENU

Langkah selanjutnya adalah menambahkan button batal dan merubah warna dari UC_MENU Tambahkan button baru pada UC_MENU dan beri text Batal dan ubah nama button menjadi **btnBatal**

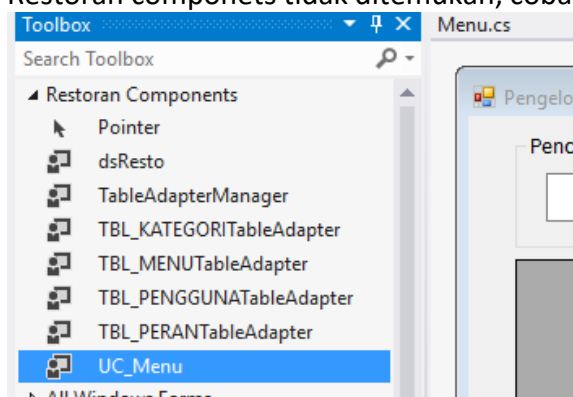


Untuk merubah warna, pilih UC_MENU, dan pada properties UC_MENU, pada pilihan BackColor, masuk ke tab System dan pilih warna : GradientInactiveCaption seperti gambar di bawah .

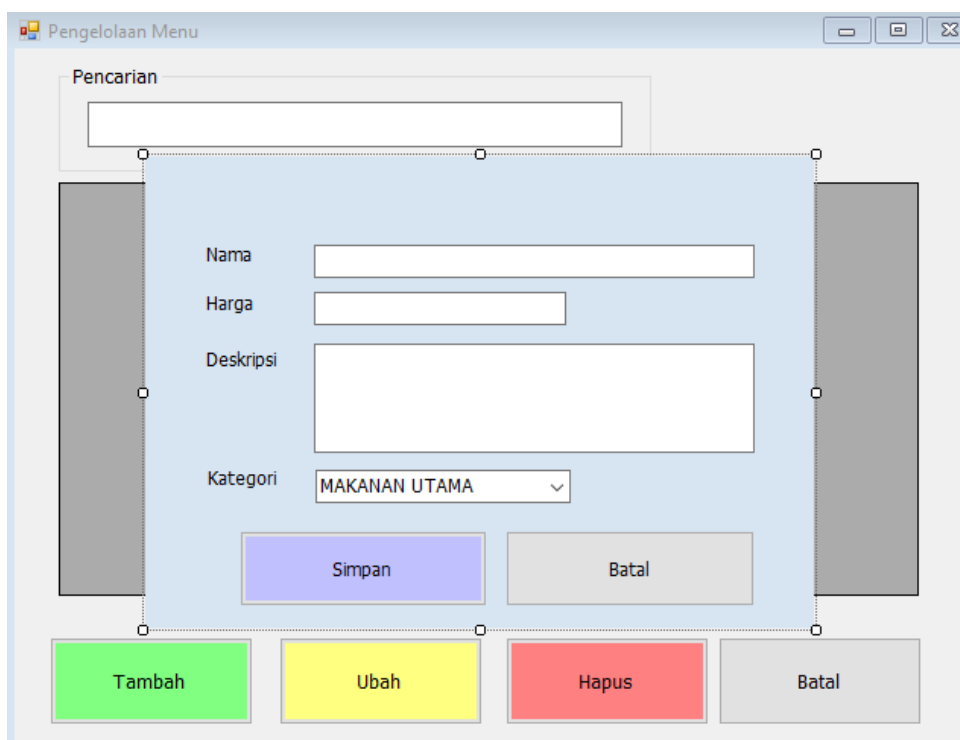


Menambahkan user control yang sudah kita buat (UC_Menu) ke Form 1

Langkah selanjutnya adalah menambah user control UC_Menu ke Form1. Caranya kembali ke tampilan dari Form1 dan pada toolbox sebelah kiri, pada **[nama project anda] Components** (dalam hal ini karena nama proyek penulis adalah restoran maka **Restoran Components**), pilih UC_Menu. Jika Restoran componets tidak ditemukan, coba compile (build) project anda dan cek kembali di toolbox.



Kemudian drag ke form 1 sehingga seperti gambar berikut



Menambah Code untuk menambahkan data di UC_Menu

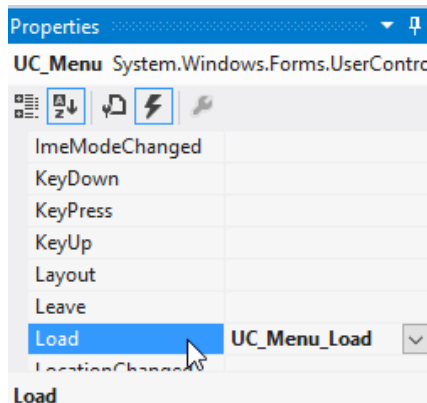
Selanjutnya kita akan menambahkan code pada UC_Menu, pilih UC_MENU dan tekan f7, maka anda akan masuk ke bagian code dari UC_MENU.

Karena UC_Menu akan digunakan untuk dua buah pengelolaan yaitu tambah dan ubah data, maka kita harus memberikan fungsi yang memungkinkan Form1 untuk mengirimkan informasi, apakah hendak menggunakannya untuk menambahkan data atau merubah data. Pada kasus ini kita tentukan bahwa jika yang dikirim adalah nilai 1 berarti untuk tambah data, dan 2 untuk ubah data. Disini kita perlu mendeklarasikan variabel dan public method sbb:

```
namespace Restoran
{
    public partial class UC_Menu : UserControl
    {
        public UC_Menu()
        {
            InitializeComponent();
        }

        int flagperintah = 0;
        public void setFlag(int flag)
        {
            flagperintah = flag;
        }
    }
}
```

Selanjutnya kembali ke tampilan/design dari UC_Menu dan tambahkan event Load melalui properties dari UC_Menu dan pada bagian code akan muncul event UC_Menu_Load

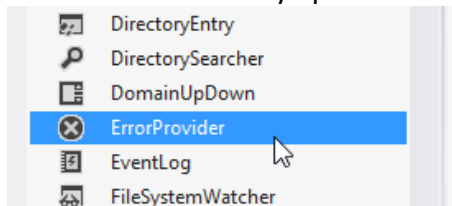


```
private void UC_Menu_Load(object sender, EventArgs e)
{
}
}
```

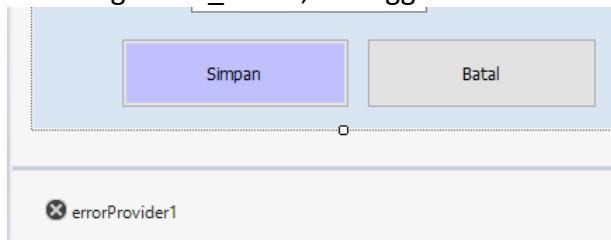
Kali ini kita akan menambahkan code untuk menambahkan data ke combo box kategori, caranya panggil terlebih dahulu MenuControl : MenuControl MC = new MenuControl(); kemudian di dalam event load tambahkan code sbb:

```
MenuControl MC = new MenuControl();
private void UC_Menu_Load(object sender, EventArgs e)
{
    cmbKategori.DataSource = MC.getKategori();
    cmbKategori.DisplayMember = "nama_kategori";
}
```

Selanjutnya kita kembali ke UC_Menu pada bagian design. Disini kita akan membuat pengecekan untuk memastikan bahwa data yang diisikan pengguna tidak kosong. Kita akan menggunakan fitur errorHandler. Caranya pilih errorHandler pada toolbox



Lalu drag ke UC_Menu, sehingga akan muncul pada bagian bawah dari UC_Menu



Selanjutnya kita akan kembali ke bagian code dari UC_Menu kemudian kita tambahkan fungsi yang memanfaatkan errorHandler untuk memastikan inputan dari pengguna harus terisi

```
private bool cektxt()
{
    bool temp = true;

    if (txtNama.Text == "")
    {
        errorHandler1.SetError(txtNama, "silahkan isi Nama Barang");
        txtNama.Focus();
        temp = false;
    }

    if (txtDeskripsi.Text == "")
    {
        errorHandler1.SetError(txtDeskripsi, "silahkan isi Deskripsi");
        txtDeskripsi.Focus();
        temp = false;
    }

    if (txtHarga.Text == "")
    {
        errorHandler1.SetError(txtHarga, "silahkan isi Harga");
        txtHarga.Focus();
        temp = false;
    }

    if (cmbKategori.Text == "")
    {
        errorHandler1.SetError(cmbKategori, "silahkan pilih Kategori");
        cmbKategori.Focus();
        temp = false;
    }
    return temp;
}
```

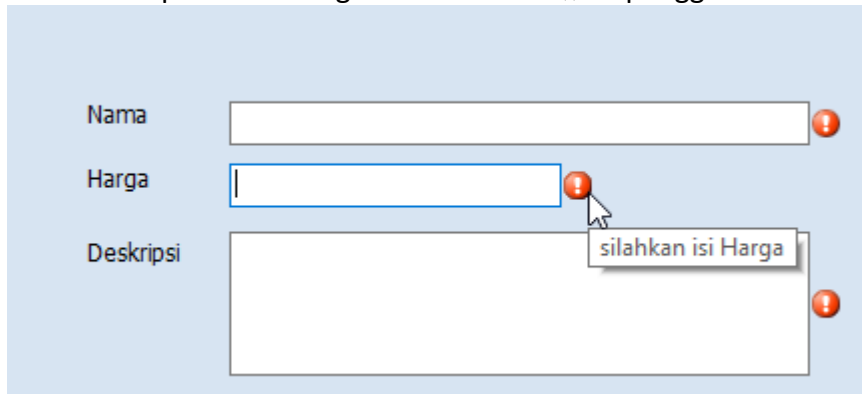
Fungsi di atas berfungsi untuk mereturnkan nilai false apabila data yg dimasukkan ke textbox dari detail data tidak lengkap. Sebagai contoh apabila txtHarga.Text tidak di isi atau sama dengan "" maka kita akan menampilkan tanda error (pentung merah) pada txtHarga dan akan memberikan keterangan kenapa ada kesalahan seperti contoh di bawah. Format code dari error provider cukup sederhana yaitu

Textbox yg di tuju

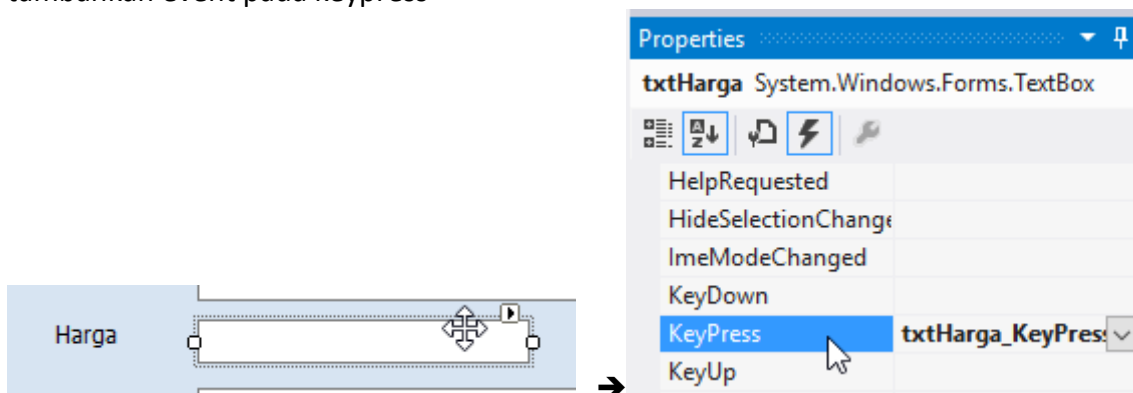
Pesan kesalahan yg hendak ditampilkan

```
errorProvider1.SetError(txtHarga, "silahkan isi Harga");
```

Contoh tampilan saat fungsi `bool cektxt ()` dipanggil



Selanjutnya kita juga akan memastikan agar txtHarga hanya boleh di isi dengan angka saja, jadi inputannya hanya berupa numerik. Caranya adalah, pilih textbox harga, dan pada properties, tambahkan event pada keypress



Akan keluar event sebagai berikut:

```
private void txtHarga_KeyPress(object sender, KeyPressEventArgs e)
{
}
```

Tambahkan code didalamnya sbb:

```
private void txtHarga_KeyPress(object sender, KeyPressEventArgs e)
{
    if (char.IsDigit(e.KeyChar) || (int)e.KeyChar == 8)
        e.Handled = false;
    else
        e.Handled = true;
}
```

Penjelasan dari kode di atas sederhana. Jika masukan dari pengguna (di dapatkan dari `e.KeyChar`) berupa digit angka (`char.IsDigit`) atau inputan dari pengguna berupa “backspace” (diwakili oleh nilai `ascii : 8` , artinya diijinkan untuk menghapus karakter di text box tersebut) maka keypress dari pengguna akan tetap diterima (ditunjukkan dengan code `e.Handled=false`). Sebaliknya selain itu maka inputan dari pengguna akan ditangani oleh sistem sehingga tidak bisa masuk ke `txtHarga` (ditunjukkan melalui kode `e.Handle=true`).

Selanjutnya masih pada `UC_Menu` bagian coding, kita akan membuat Fungsi Clear textbox. Fungsi ini berguna untuk mengembalikan kondisi di detail data menjadi ke keadaan semula. Yaitu dengan membersihkan textbox yaitu nama, harga, deskripsi dan menset combobox kategori agar tidak ada indeks yang terpilih (diset sama dengan -1). Caranya dengan menambahkan fungsi sbb:

```
private void cleartxt()
{
    txtNama.Clear();
    txtHarga.Clear();
    txtDeskripsi.Clear();
    cmbKategori.SelectedIndex = -1;
}
```

Selanjutnya kita akan kembali ke `Form1.cs`.

Menambahkan code untuk mengimplementasikan insert data pada `Form1.cs`

Kita akan kembali ke bagian code dari `Form1.cs`, Karena pada saat `Form1` dijalankan, `UC_Menu` tidak terlihat atau disembunyikan, maka kita perlu menambahkan code untuk menyembunyikannya di awal. Caranya kita letakkan pada event Load Form dengan menambahkan code untuk menyembunyikan `UC_Menu` pada loadForm `Form1.cs` sbb:

```
uC_Menu1.Visible = false;
```

Dengan posisi code tepat ditunjukkan seperti pada gambar di bawah :

```
private void Form1_Load(object sender, EventArgs e)
{
    setDatagridview(this.dataGridView1);
    uC_Menu1.Visible = false;
}
```

Seperti telah dijelaskan pada skenario sebelumnya, pada saat `UC_Menu` muncul maka semua button , datagridview dan text search di `Form1.cs` di disable, dan bila sudah ditambahkan, maka akan kembali di aktifkan / enable. Untuk itu kita harus membuat fungsi Disable dan disable pada `Form1.cs` sbb:

```

private void disable()
{
    txtCari.Enabled = false;
    dataGridView1.Enabled = false;
    btnTambah.Enabled = false;
    btnUbah.Enabled = false;
    btnHapus.Enabled = false;
    btnBatal.Enabled = false;
}

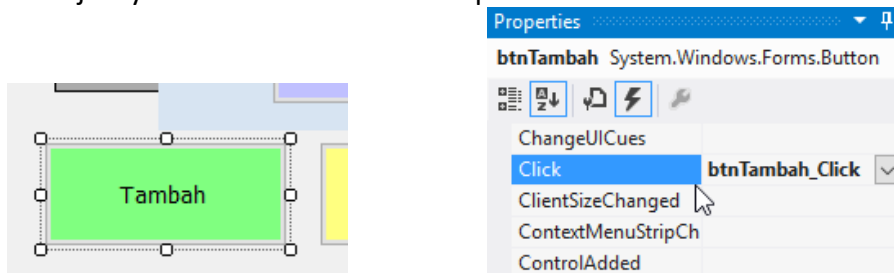
public void Enable()
{
    txtCari.Enabled = true;
    dataGridView1.Enabled = true;
    btnTambah.Enabled = true;
    btnUbah.Enabled = true;
    btnHapus.Enabled = true;
    btnBatal.Enabled = true;

    setDatagridview(this.dataGridView1);
    dataGridView1.Rows[0].Selected = true;
}

```

Pada fungsi disable() kita menset properties enable dari setiap kompone di Form1.cs dengan false yang artinya tidak aktif. Sedangkan pada fungsi Enable(), selain mengaktifkan kembali status enable , kita juga mererefresh datagridview agar menampilkan data terbaru setelah ditambahkan dengan memanggil fungsi setDatagridview kembali. Pada baris terakhir dari fungsi Enable, kita menseleksi data teratas yang merupakan data yang baru ditambahkan yang ditunjukan pada index Rows ke 0.

Selanjutnya tambahkan event Click pada btnTambah



Akan keluar event sebagai berikut

```

private void btnTambah_Click(object sender, EventArgs e)
{
}

```

Selanjutnya tambahkan code didalam event click tersebut sbb:

```

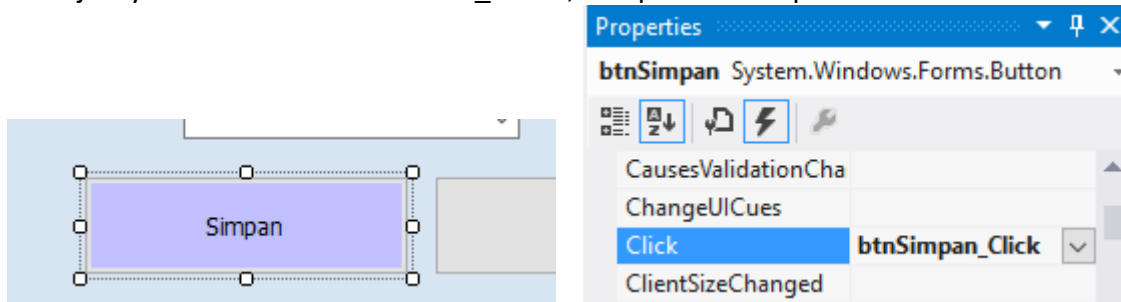
private void btnTambah_Click(object sender, EventArgs e)
{
    uC_Menu1.setFlag(1);
    uC_Menu1.Visible = true;
    disable();
}

```

Fungsi diatas berguna untuk mengirimkan tanda ke UC_Menu bahwa kita hendak menjalankan operasi tambah data yang diwakili dengan kode flag 1. Code tersebut memanggil fungsi public dari UC_Menu yaitu setFlag dan menset flag perintah dengan 1 yang artinya tambah data. Selanjutnya kita menset agar uC_Menu terlihat (Visible) dan muncul dan kita menset Form1.cs menjadi disable.

Menambahkan code tambah data pada UC_Menu

Selanjutnya kita akan kembali ke UC_Menu, dan pilih btnSimpan dan tambahkan event click



Maka akan muncul event click sbb:

```
private void btnSimpan_Click(object sender, EventArgs e)
{

}
```

Tambahkan code didalamnya sebagai berikut

```
private void btnSimpan_Click(object sender, EventArgs e)
{
    if (flagperintah == 1) //tambah data
    {
        if (cektxt() == true)
        {
            errorProvider1.Clear();

            int IDKategori = MC.getIDKategori(cmbKategori.Text);
            Restoran.Entity.Menu M = new Entity.Menu(txtNama.Text, txtDeskripsi.Text, double.Parse(txtHarga.Text), IDKategori);
            MC.addMenu(M);
            cleartxt();
            this.Hide();
            Form1 myParent = (Form1)this.Parent;
            myParent.Enable();
        }
    }
}
```

Logika dari code di atas adalah sbb: Apa bila flagperintah == 1 yang artinya hendak dilakukan tambah data, maka akan di cek terlebih dahulu inputan data dari kita, jika tidak ada isian textbox yang kurang (di cek melalui `if (cektxt() == true)`) maka kita clear kan dulu error Provider (`errorProvider1.Clear();`). Kemudian kita dapatkan id kategori menggunakan menggunakan fungsi `getIDKategori` untuk mendapatkan id kategori yang akan kita insertkan nanti

Selanjutnya kita akan membuat objek dari Entity menu yang atributnya di isi berdasarkan text box yang ada di UC_Menu. Pembuatan objeknya ditunjukkan pada code:

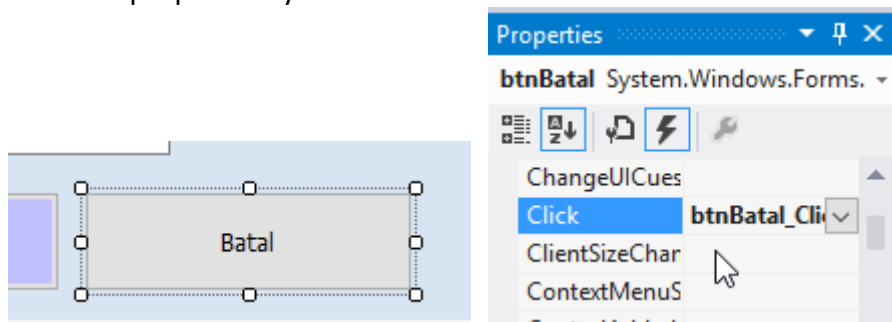
```
Restoran.Entity.Menu M = new Entity.Menu(txtNama.Text, txtDeskripsi.Text,  
double.Parse(txtHarga.Text), IDKategori);
```

Selanjutnya objek M akan digunakan pada fungsi addMenu sebagai parameter. Selanjutnya setelah berhasil menambahkan data, kita akan mengclearkan textbox pada UC_Menu, menyembunyikan UC_Menu agar tidak terlihat (hide) dan kemudian memanggil fungsi dari Form1 yaitu Enable() yang akan mengaktifkan Form1 dan merefresh datagridview dan men-select row teratas yang baru saja ditambahkan.

Untuk memanggil Form1 yang merupakan parent Form dari UC_Menu, digunakan kode sbb:

```
Form1 myParent = (Form1)this.Parent;  
myParent.Enable();
```

Selanjutnya kembali ke tampilan (design) dari UC_Menu, kita pilih btnBatal dan tambahkan event click dari propertiesnya.



Maka akan keluar event click sbb:

```
private void btnBatal_Click(object sender, EventArgs e)  
{  
  
}
```

Tambahkan code di dalamnya sbb:

```
private void btnBatal_Click(object sender, EventArgs e)  
{  
    cleartxt();  
    errorProvider1.Clear();  
    this.Hide();  
    Form1 myParent = (Form1)this.Parent;  
    myParent.Enable();  
}
```

BtnBatal berfungsi membatalkan tindakan dengan mengclearkan textbox dari UC_Menu melalui fungsi cleartxt() yang sudah dibuat sebelumnya, kemudian mengclearkan error Provider. Kemudian menghide / menyembunyikan UC_Menu agar tidak muncul pada form 1. Kemudian memanggil fungsi dari Form1 yaitu Enable() yang akan mengaktifkan Form1 dan merefresh datagridview dan men-select row teratas yang baru saja ditambahkan.

Jika semua sudah tambahkan dengan benar maka Fungsi menambahkan data menu akan berjalan dengan baik. Bersambung ke modul selanjutnya 😊

Rangkuman Keseluruhan code pada Form1.cs sampai minggu 5 :

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using Restoran.Control;

namespace Restoran
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();

            MenuControl MC = new MenuControl();
            public void setDatagridview(DataGridView DG)
            {
                DG.DataSource = MC.showMenu();

                DG.Columns[0].HeaderText = "ID";
                DG.Columns[1].HeaderText = "Nama";
                DG.Columns[2].HeaderText = "Harga";
                DG.Columns[3].HeaderText = "Deskripsi";
                DG.Columns[4].HeaderText = "Kategori";

                DG.Columns[0].Width = 50;
                DG.Columns[1].Width = 100;
                DG.Columns[2].Width = 75;
                DG.Columns[3].Width = 150;
                DG.Columns[4].Width = 80;
            }

            public void searchDataGridView(DataGridView DG, string keyword)
            {
                DG.DataSource = MC.searchMenu(keyword);

                DG.Columns[0].HeaderText = "ID";
                DG.Columns[1].HeaderText = "Nama";
                DG.Columns[2].HeaderText = "Harga";
                DG.Columns[3].HeaderText = "Deskripsi";
                DG.Columns[4].HeaderText = "Kategori";

                DG.Columns[0].Width = 50;
                DG.Columns[1].Width = 100;
                DG.Columns[2].Width = 75;
                DG.Columns[3].Width = 150;
                DG.Columns[4].Width = 80;
            }

            private void Form1_Load(object sender, EventArgs e)
            {
                setDatagridview(this.dataGridView1);
                uC_Menu1.Visible = false;
            }

            private void txtCari_TextChanged(object sender, EventArgs e)
            {
                searchDataGridView(dataGridView1, txtCari.Text);
            }
        }
    }
}
```



```

private void disable()
{
    txtCari.Enabled = false;
    dataGridView1.Enabled = false;
    btnTambah.Enabled = false;
    btnUbah.Enabled = false;
    btnHapus.Enabled = false;
    btnBatal.Enabled = false;
}

public void Enable()
{
    txtCari.Enabled = true;
    dataGridView1.Enabled = true;
    btnTambah.Enabled = true;
    btnUbah.Enabled = true;
    btnHapus.Enabled = true;
    btnBatal.Enabled = true;

    setDataGridView(this.dataGridView1);
    dataGridView1.Rows[0].Selected = true;
}

private void btnTambah_Click(object sender, EventArgs e)
{
    uC_Menu1.setFlag(1);
    uC_Menu1.Visible = true;
    disable();
}
}
}

```

Rangkuman Keseluruhan code pada UC_Menu.cs sampai minggu 5 :

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Drawing;
using System.Data;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using Restoran.Control;
using Restoran.Entity;
using Restoran.dsRestoTableAdapters;

namespace Restoran
{
    public partial class UC_Menu : UserControl
    {
        public UC_Menu()
        {
            InitializeComponent();

            int flagperintah = 0;
            public void setFlag(int flag)
            {
                flagperintah = flag;
            }

            MenuControl MC = new MenuControl();
            private void UC_Menu_Load(object sender, EventArgs e)
            {
                cmbKategori.DataSource = MC.getKategori();
                cmbKategori.DisplayMember = "nama_kategori";
            }
        }
    }
}

```

```

private bool cektxt()
{
    bool temp = true;

    if (txtNama.Text == "")
    {
        errorProvider1.SetError(txtNama, "silahkan isi Nama Barang");
        txtNama.Focus();
        temp = false;
    }

    if (txtDeskripsi.Text == "")
    {
        errorProvider1.SetError(txtDeskripsi, "silahkan isi Deskripsi");
        txtDeskripsi.Focus();
        temp = false;
    }

    if (txtHarga.Text == "")
    {
        errorProvider1.SetError(txtHarga, "silahkan isi Harga");
        txtHarga.Focus();
        temp = false;
    }

    if (cmbKategori.Text == "")
    {
        errorProvider1.SetError(cmbKategori, "silahkan pilih Kategori");
        cmbKategori.Focus();
        temp = false;
    }
    return temp;
}

private void cleartxt()
{
    txtNama.Clear();
    txtHarga.Clear();
    txtDeskripsi.Clear();
    cmbKategori.SelectedIndex = -1;
}

private void btnSimpan_Click(object sender, EventArgs e)
{
    if (flagperintah == 1) //tambah data
    {
        if (cektxt() == true)
        {
            errorProvider1.Clear();

            int IDKategori = MC.getIDKategori(cmbKategori.Text);
            Restoran.Entity.Menu M = new Entity.Menu(txtNama.Text, txtDeskripsi.Text,
double.Parse(txtHarga.Text), IDKategori);
            MC.addMenu(M);
            cleartxt();
            this.Hide();
            Form1 myParent = (Form1)this.Parent;
            myParent.Enable();
        }
    }
}

```

```

private void btnBatal_Click(object sender, EventArgs e)
{
    cleartxt();
    errorProvider1.Clear();
    this.Hide();
    Form1 myParent = (Form1)this.Parent;
    myParent.Enable();
}

private void txtHarga_KeyPress(object sender, KeyPressEventArgs e)
{
    if (char.IsDigit(e.KeyChar) || (int)e.KeyChar == 8)
        e.Handled = false;
    else
        e.Handled = true;
}
}
}

```

Rangkuman Keseluruhan code pada MenuControl.cs sampai minggu 5 :

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Restoran.dsRestoTableAdapters;
using System.Data;
using Restoran.Entity;

namespace Restoran.Control
{
    class MenuControl
    {
        private TBL_MENUTableAdapter TM = new TBL_MENUTableAdapter();
        private TBL_KATEGORITableAdapter TK = new TBL_KATEGORITableAdapter();

        public DataTable showMenu()
        {
            return TM.GetData();
        }

        public DataTable searchMenu(string Keyword)
        {
            return TM.GetDataBy(Keyword);
        }

        public void addMenu(Menu M)
        {
            TM.InsertMenu(M.Nama, M.Harga, M.Deskripsi, M.Kategori);
        }

        public DataTable getKategori()
        {
            return TK.GetData();
        }

        public int getIDKategori(string kategori)
        {
            return TK.GetIdKategori(kategori).Value;
        }
    }
}

```