ÇOKLU FORMLARLA ÇALIŞMA

İÇİNDEKİLER

- C# Form Yapısı
  - Form Özellikleri
  - Windows Form Kontrolleri ve Özellikleri
- Çoklu Formlar
  - Formlar Arası Geçiş
  - Formlar Arası Veri Aktarımı

HEDEFLER

- Bu üniteyi çalıştıktan sonra;
- İstenilen özellikte formlar tasarlayabilecek,
- Formlar arası geçiş yapabilecek,
- Verileri farklı formlarda göstererek daha etkin uygulamalar geliştirebileceksiniz.

Atatürk Üniversitesi
Açıköğretim Fakültesi

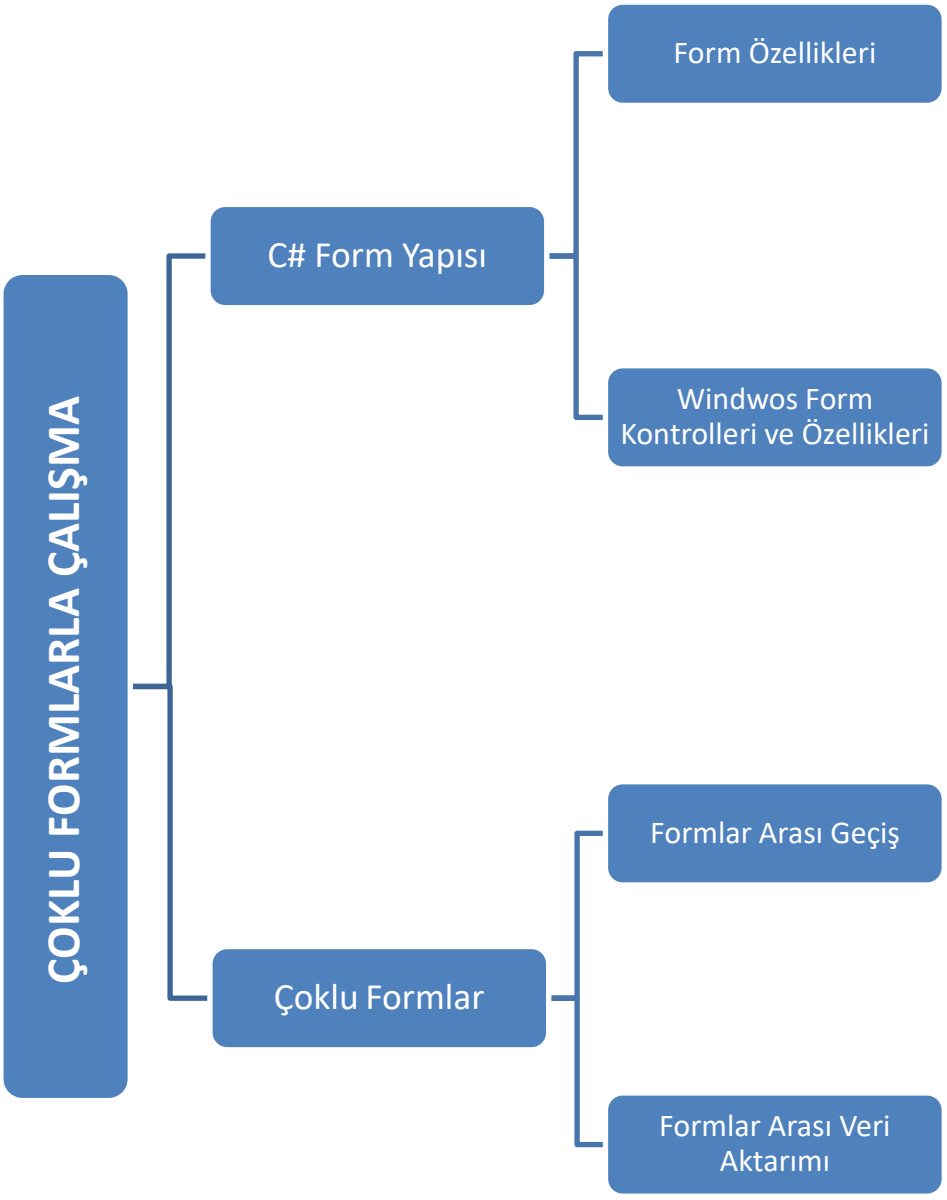
GÖRSEL PROGRAMLAMA I

İshak Metehan SİS

ÜNİTE

14

© Bu ünitenin tüm yayın hakları Atatürk Üniversitesi Açıköğretim Fakültesi'ne aittir. Yazılı izin alınmadan ünitenin tümünün veya bir kısmının elektronik, mekanik ya da fotokopi yoluyla basımı, yayımı, çoğaltımı ve dağıtımı yapılamaz.


ÇOKLU FORMLARLA ÇALIŞMA
C# Form Yapısı
Form Özellikleri
Windwos Form Kontrolleri ve Özellikleri
Çoklu Formlar
Formlar Arası Geçiş
Formlar Arası Veri Aktarımı

GİRİŞ

Programlama, en basit haliyle makine ile insan arasında iletişim kuran yazılım olarak tanımlanabilir. Günümüzde uygulama geliştirmek için birçok programlama dili kullanılmaktadır. Kullanılabilirlik bakımından ise son zamanlarda görsel programlamaya olan ilginin artığı görülmektedir.

Görsel arayüz olmadan da birçok uygulama çeşitli programlama dilleri kullanılarak yapılabilmektedir. Ancak geliştirilen uygulamanın kullanıcılar tarafından daha kolay kullanılabilmesi için görsel programlamaya ihtiyaç duyulur.

Tasarım ve kodun birlikte kullanılmasıyla programlama yapılması Görsel Programlama olarak tanımlanabilir. C# dilinde bu görsellik için *Windows Form Application* kullanılmaktadır. Bu şablon içinde Form, TextBox, CheckBox, RadioButton, ComboBox, ListBox, DataGridView gibi sınıflar yer almaktadır. Bu sınıflar kullanılarak birçok uygulama kolayca geliştirilebilmektedir.

Bir windows form uygulaması için ilk olarak projeye bir form öğesi eklenir. Daha sonra uygulamanın içeriğine göre TextBox, Label, Button gibi bileşenler form üzerine eklenerek uygulamanın tasarım kısmı tamamlanır.

Çoğu uygulamada farklı bilgiler bir arada kullanılabilir. Bu gibi durumlarda farklı bilgileri tek bir form üzerinde göstermek yerine farklı formlarla çalışmak gerekebilir. Böylece farklı bilgileri farklı formlarda gösterip gerek duyulduğunda kullanıcının bu formlar arasında geçiş yapması sağlanarak daha profesyonel uygulamalar geliştirilebilir.

Bu ünitede ilk olarak C# form yapısı anlatılıp uygulamaya nasıl eklendiğinden bahsedilecektir. Daha sonra form özellikleri ele alınıp form üzerinde yaygın olarak kullanılan bileşenler ve özelliklerine değinilecektir. Son olarak uygulama üzerine birden fazla formun nasıl eklendiği, formlar arasında geçişin nasıl yapıldığı hakkında bilgi verilip konu örnekler ile detaylandırılacaktır.

C# FORM YAPISI

Windows Form uygulaması açıldığında Visual Studio tarafından projeye otomatik olarak bir form atanır.

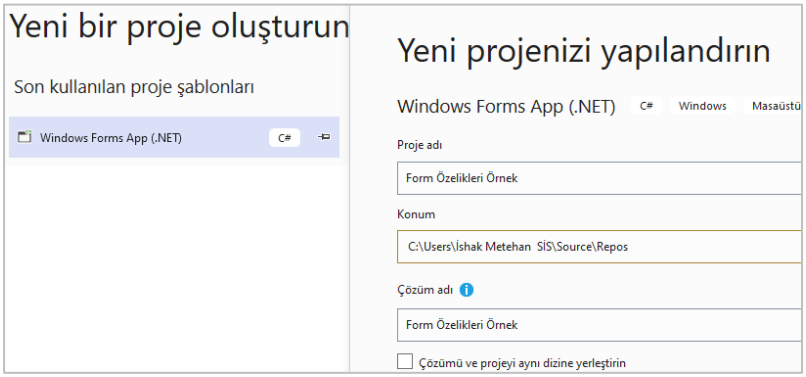
C# Windows Form ile kullanıcı arayüzü tasarlanabilir, arayüzde yer alan bileşenler kolay bir şekilde konumlandırılıp hizalama yapılabilir. Arayüz tasarlanırken çeşitli form kontrollerinden ve bu kontrollerin özelliklerinden yararlanılmaktadır. Kontroller form içinde bulunan nesneler olarak tanımlanabilir. Her bir kontrol belirli bir amaç için oluşturulmuştur ve her birinin kendine ait özellikleri, yöntemleri ve olayları bulunmaktadır. Geliştiriciler bu kontrollerin özelliklerini kullanarak ve değiştirerek Windows Form uygulamalarına dinamiklik kazandırırlar (Alkan, 2018).

Bu bölümde formun uygulama üzerine nasıl eklendiği ve form kontrollerinin özelliklerinden bahsedilecektir.

Form Özellikleri

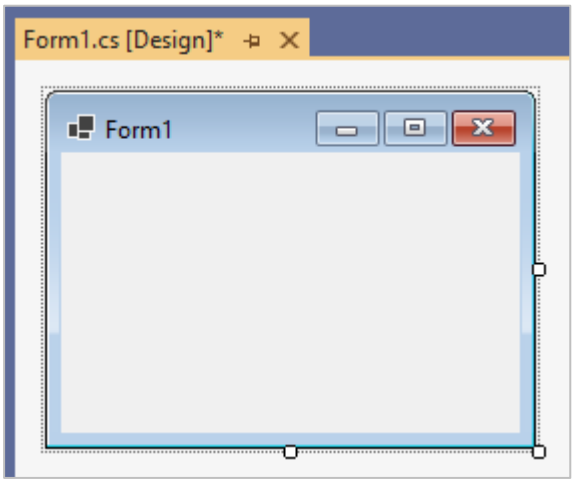
Windows Form uygulaması açıldığında Visual Studio tarafından projeye otomatik olarak bir form atanır. Eğer birden fazla form kullanılacak ise bu geliştirici tarafından daha sonra eklenebilir.

İlk olarak Visual Studio üzerinden bir windows form uygulaması açılır. Şekil 14.1'de uygulamanın nasıl açıldığı gösterilmiştir.



Şekil 14.1. Yeni Proje Açma

Uygulama açıldıktan sonra projeye atanan form Şekil 14.2' de gösterilmiştir.



Şekil 14.2. Form Oluşturma

Daha sonra araç kutusundan gerekli kontrollerin sürükle bırak yöntemi ile forma eklenmesi ile arayüz tasarımı yapılır. Tasarım sırasında kullanılabilirliği ve geliştirici açısından tasarımın kolayca gerçekleştirilebilmesi için kontrollerin özellikleri ve nasıl kullanıldığının bilinmesi gerekir. Form kontrolünün özelliklerine *Özellikler* penceresinden erişilebilmektedir.

Form kontrolünün en yaygın kullanılan özellikleri (Salkım, 2012);

- **AcceptButton:** Enter tuşuna basılınca hangi tuşun işlem yapacağını belirtir. Form üzerindeki buttonlardan biri seçilerek o button için özellik aktif hale getirilir.
- **AutoScroll:** Form üzerindeki nesneler forma sığmadığı durumda otomatik olarak scrollbar eklenip eklenmeyeceği bu özellik ile belirlenir. Varsayılan değeri *False'*dır.

- **AutoScrollMargin:** Eğer formda scrollbar var ise scrollbar'ın genişliği ve yüksekliğini ayarlar.
- **AutoSize:** Formun içindeki kontrollere göre formun boyutunun otomatik olarak ayarlanacağı özelliktir.
- **BackColor**: Formun zemin rengini değiştirmek için kullanılır. Custom, Web ve System olmak üzere 3 seçeneği mevcuttur.
- **BackgroundImage:** Formun arka plan resmini ayarlar.
- **BackGroundImageLayout**: Forma eklenen arka plan resminin nerede ve nasıl duracağını belirler. None, Tile, Center, Stretch, Zoom olmak üzere 4 farklı özelliği mevcuttur.
- **CancelButton:** "ESC" tuşuna basılınca form üzerinde çalışacak buttonu ayarlar.
- **Enabled:** Bu özellik ile formun kullanıcı tarafından kullanılıp kullanılmayacağı, form üzerindeki elemanların aktif olup olmayacağı belirlenir. Eğer pasif olursa Form üzerindeki diğer Form nesneleri de pasif olur.
- **Font:** Form üzerine eklenen elemanların yazı tipi özellikleri ayarlanır. Form üzerindeki diğer Form nesnelerinin de fontu değişir.
- **FontDialog**: Sistemde o anda yüklü olan yazı tiplerini gösterir.
- **FormBorderStyle**: Formun boyutlarının değiştirilip değiştirilmeyeceğini ayarlar. Varsayılan değeri None'dır. Eğer formun boyunun değiştirilmesi istenmiyorsa "Fixed " özelliği kullanılmalıdır. None, FixedSingle, Fixed3D, FixedDialog, Sizable, FixedToolWindow, SizableToolWindow gibi özellikleri bulunmaktadır. Bu özellikler ile formun boyutlandırılması ve form simgesinin nasıl gösterileceği ayarlanır.
- **HelpButton:** Başlık çubuğunda *Yardım* düğmesinin gösterilip gösterilmeyeceği ayarlanır.
- **Locked**: Bu özellik ile hem formun boyutu hem de konumu değiştirilmez hale getirilir.
- **Language:** Form'un dilini belirler. Default değeri sistemin kurulu olduğu dildir.
- **Is MDI Container:** MDI (Multiple Document Interface) *Çoklu Form Arayüzü* demektir. "True" değeri seçilirse formun içinde başka bir form açılabilir demektir. Bu durumda ana form Parent Form, ana formun içinde açılan form ise Child Form olarak adlandırılır.
  Bu özellik true olduğunda Visual Studio otomatik olarak içine bir adet Container kontrolü atar. Bu Container kontrolü sayesinde birçok form ile çalışılabilir.
- **TopMost:** Form açıldığında başka form varsa, diğer tüm formların üzerinde gözüküp gözükmeyeceğini ayarlar. Formumuz kapatılmadan diğer form gözükmez. Varsayılan değeri False olmaktadır.
- **Text:** Formun sağ üstte bulunan ikonun yanındaki yazıyı değiştirmek için kullanılır. Bu değer varsayılan olarak Form1 'dir.

Formun FormBorderStyle özelliği FixedSingle yapılarak Formun boyutunun fare ile değiştirilmesi engellenir.

Uygulamada birden fazla form kullanılacak ise Is MDI Container özelliğinin True yapılması gerekir.

- **TopMost:** Form açıldığında başka form varsa, diğer tüm formların üzerinde gözüküp gözükmeyeceğini ayarlar. Default değeri False'dır.
- **WindowsState**: Formun tam ekran mı yoksa normal boyutlarda mı gösterileceğini ayarlar. Normal, Minimized, Maximized özellikleri mevcuttur.
- **StartPosition:** Uygulama çalıştırıldığında Formun, ekranın neresinde açılacağını belirtir. Manuel, CenterScreen, WindowsDefaultLocation, WindowsDefaultBounds, CenterParent gibi seçenekleri mevcuttur.
- **Size:** Formun boyutlarını ayarlar. Float tipindedir.
- **Name:** Bu özellik kullanılarak formun adı değiştirilebilir. Kodlama sırasında okunabilirliği artırmak amacıyla Form1, Form2 yerine forma uygun isim verilir.

Form özelliklerinin yanı sıra işlem yapılmasını sağlayan form olayları mevcuttur. Yaygın olarak kullanılan form olayları Tablo 14.1'de yer almaktadır.

Tablo 14.1. Form Olayları (Alan, 2011)

	Açıklama
Click	Form üzerine tıklandığında gerçekleşen olaydır.
Closing	Form kapanmadan önce gerçekleşen olaydır.
KeyDown	Form üzerinde bir tuşa basılınca gerçekleşen olaydır.
KeyUp	Form üzerine basılan tuşun kaldırılması ile gerçekleşen olaydır.
Load	Form yüklenirken gerçekleşen olaydır.

Arayüz tasarlarken çeşitli kontrollerden ve bu kontrollerin özelliklerinden yararlanılır. Bu bölümde tasarım sırasında yaygın kullanılan birkaç kontrol ve özelliğine değinilecektir.

Form Kontrolleri ve Özellikleri

TextBox

Kullanıcının çok satırlı veri girmesini sağlar. Girilen bu veriler düzenlenebilir durumdadır. TextBox özellikleri Tablo 14.2'de yer almaktadır.

Tablo 14.2. TextBox Özellikleri

	Açıklama
MultiLine	Bu özelliğin değeri True yapıldığında metin kutusuna birden fazla satır girilmesi sağlanır. False durumunda ise bu işleme izin verilmez metin kutusunun boyutu değişmez.
PasswordChar	Metin kutusuna parola girileceği zaman bu özellik kullanılır. Girilen karakterlerin kullanıcıya hangi karakter olarak görüneceğini belirtir.
ReadOnly	Metin kutusunda yazma işleminin yapılmasına izin verilmez. Sadece okuma yapılabilir.

MaxLenght	Metin kutusuna girilecek maksimum karakter sayısını belirtir.
ScrollBars	Metin kutusuna scrollbar eklenmesini sağlar.

Label kontrolünün TextAlign, Image, ImageAlign gibi özellikleri sayesinde label üzerinde yer alan resim ya da yazının hizalanması sağlanır.

Button

Kullanıcının işlem yapmak için tıklayabileceği düğme olarak tanımlanabilir. Yani kullanıcı ve uygulama arasında etkileşim oluşturmak için kullanılan kontrollerdir.

Label

Kullanıcıya bilgi vermek için kullanılır. Label üzerinde yer alan metin kullanıcı tarafından düzenlenemez.

Tablo 14.3. Label Özellikleri

	Açıklama
TextAligin	Label üzerindeki yazının konumunu belirtir.
Image	Label üzerindeki resmi tutar.
ImageAlign	Label üzerindeki resmin nerede duracağını belirler.

CheckBox

Kullanıcının bir ya da birden fazla seçeneği seçmesine olanak tanır. *Checked* özelliği ile kontrolün seçili olup olmadığı belirlenir.

ComboBox

Açılır bir menüde verilerin görüntülenmesini ve kullanıcıya veri girişi yapmayı sağlar. Kontrol aşağıya doğru açılmakta ve 2 bölümden oluşmaktadır. Üst taraf kullanıcının metin girmesini sağlayan textbox kontrolü gibi ele alınabilir. İkinci kısım kullanıcının seçebileceği öğelerin listesini oluşturan ListBox kontrolü gibi düşünülebilir.

CheckedListbox

Her bir öğenin yanında onay kutusuyla birlikte bir öğe listesi görüntüler. Böylece kullanıcıya çoklu seçme olanağı tanır. Tablo 14.4'te özellikleri yer almaktadır.

Tablo 14.4 CheckedListBox Özellikleri

	Açıklama
Selectedindex	Listeden seçilen elemanın sıra numarasını verir.
Items.Count	Listedeki elemanların toplam sayısını verir.
Checkeditems.count	Seçilmiş eleman sayısını verir.
Selecteditem	Hangi check seçili ise onun adını verir.

DataGridView

Verilerin, hücrelerden ve satırlardan oluşan tablolarda ki gösterimini sağlayan bir kontroldür.

Tablo 14.5 DataGridView Özellikleri

	Açıklama
Rows	Satırlarla ilgili işlem yapmayı sağlar.
Columns	Tablo sütunlarıyla ilgili işlemleri yapmayı sağlar.
DataSource	Veri tabanı bağlantısı yapmayı sağlar.
MultiSelect	Hücrelerde çoklu seçime izin verilip verilmeyeceğini ayarlar.

DataTimePicker

Tarih veya saat bilgisini gösterir.

GroupBox

Form üzerinde yer alan diğer kontrollerin gruplanmasını sağlar.

ListBox

Kullanıcının bir listeden bir veya daha fazla öğe seçmesini sağlayan kontroldür.

OpenFileDialog

Kullanıcıların bir iletişim kutusu kullanarak dosyaları açmasına izin verir.

PageSetupDialog

Kullanıcının yazdırma işlemleri sırasında sayfa yapısını ayarlamayı sağlar.

PrintDialog

Kullanıcılar herhangi bir belgeyi yazdırırken yazdırma özelliklerinin yer aldığı pencerenin açılmasını sağlar.

PrintDocument

Yazdırma işlemleri için kullanılır.

PrintPreviewDialog

Yazdırma işlemi sırasında Baskı Ön İzleme için kullanılır.

RadioButton

Kullanıcıya iki veya daha fazla seçenek sunmayı sağlar.

RichTextBox

Kullanıcıların metin girmesine, görüntülemesine ve güncellemesine izin verir.

MenuStrip

Form üzerine menü eklenmesini sağlar. MenuStrip birçok uygulamada kullanılan genelde üst kısımda yer alan ana menü ve altındaki alt menülerden oluşan bir yapıdır. Çoğu uygulamada Dosya, Düzen, Araçlar gibi seçenekler menülerde ortak olarak kullanılmaktadır. Bu tip ortak menüler MenüStrip kontrolünün sağ üst köşesinde yer alan kulakçığa tıklanıp Standart Öğe Ekle seçeneği seçilerek kolayca oluşturulabilir.

ContexMenuStrip

Form üzerinde, farenin sağ tuşu ile tıklanınca görünen menünün eklenmesini sağlar. Uygulama çalıştırılıp sağ tuşa tıklanınca menünün açılması için oluşturulan ContextMenuStrip kontrolünün bağlantısının yapılması gerekir.

ToolStrip

ToolStrip, genelde uygulamalarda üst kısımda yer alan araç çubuğunu oluşturmak için kullanılan bir kontroldür. ToolStrip kontrolünün sağ üst köşesinde yer alan kulakçığa tıklanıp Standart Öğe Ekle seçeneği tıklanarak hazır araç çubuğu eklenebilir. Bu araç çubuğunda Yeni, Aç, Kaydet, Yazdır, Kes, Kopyala, Yapıştır ve Yardım seçenekleri yer almaktadır.

StatusStrip

Genelde kullanıcıya bilgi vermek için kullanıldığından *Bilgi Şeridi* olarak da adlandırılır. Formun en altına konumlanır. Kullanıcıya anlık tarih ve saat bilgisi, aktif kullanıcı bilgisi gibi bilgileri göstermek için kullanılır.

Bireysel Etkinlik

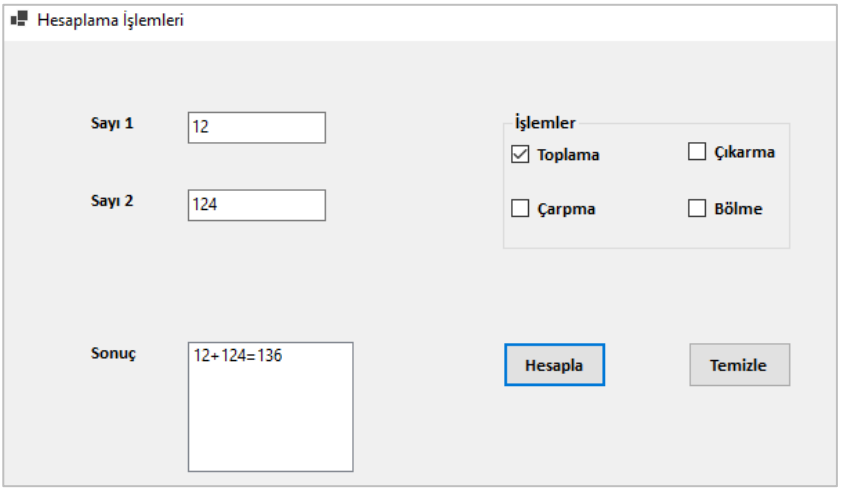
- İllerin listelendiği bir ComboBox'tan, herhangi bir il seçilince o ile ait ilçelerin yine aynı formda yer alan ListBox'a eklendiği bir uygulama geliştiriniz.

Kullanıcıdan TextBox'lar ile aldığı iki sayı için 4 işlem yapan ve işlem sonucunu ListBox'ta gösteren örnek uygulamaya ait kodlar aşağıda yer almaktadır. Dört işlemi seçmek için CheckBox kontrolü kullanılmıştır. Tasarımın daha düzgün olması bakımından dört işlem için kullanılan seçenekler GroupBox içine yerleştirilmiştir.

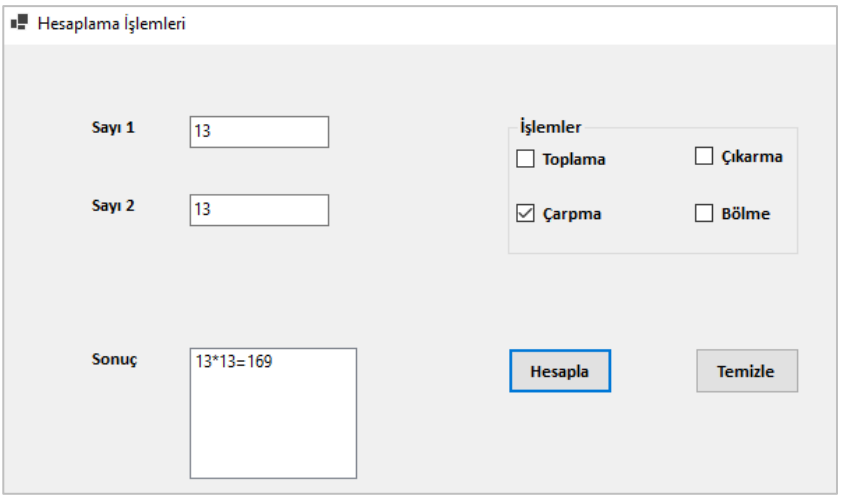
```
private void Hesapla_Click(object sender, EventArgs e)
{
        //TextBox'ta alınan sayılar Double tipine çevrildi.
         double sayi1 = Convert.ToDouble(Sayi1.Text);
         double sayi2 = Convert.ToDouble(Sayi2.Text);

        //Eğer topla checkbox'u seçili ise
         if (checkBoxTopla.Checked == true)
        {
                double sonuc = sayi1 + sayi2;
                list”Box1.Items.Add(sayi1 + "+" + sayi2 + "=" + sonuc);
        }

        //Eğer çıkar checkbox'u seçili ise
        if (checkBoxCikar.Checked == true)
         {
                double sonuc = sayi1 - sayi2;
                listBox1.Items.Add(sayi1 + "-" + sayi2 + "=" + sonuc);
         }
         //Eğer çarpma checkbox'u seçili ise
         if (checkBoxCarp.Checked == true)
         {
                double sonuc = sayi1 * sayi2;
                listBox1.Items.Add(sayi1 + "*" + sayi2 + "=" + sonuc);
         }
         //Eğer bölme checkbox'u seçili ise
         if (checkBoxBol.Checked == true)
         {
                double sonuc = sayi1 / sayi2;
        listBox1.Items.Add(sayi1 + "/" + sayi2 + "=" + sonuc);
         }
}
private void Temizle_Click(object sender, EventArgs e)
{
         //Listbox temizlendi.
         listBox1.Items.Clear();
}
```



Şekil 14.3. Toplama CheckBox Tıklandığı Durum



Şekil 14.4. Çarpma CheckBox Tıklandığı Durum

ÇOKLU FORMLAR

"ShowDialog" ile gösterimde ikinci form kapanmadan birinci form üzerinde kullanıcıya işlem yaptırılmazken "Show" ile aynı anda iki forma da ulaşılabilir.

Daha öncede belirtildiği gibi farklı amaca yönelik bilgileri aynı formda göstermek yerine bu bilgileri farklı formlarda göstermek daha profesyonel çalışmalar yapılmasını sağlayıp kullanımı kolaylaştırmaktadır.

Bu bölümde ilk olarak bir projede birden fazla formun nasıl oluşturulduğu ve bu formlar arasında geçişin nasıl yapıldığına değinilecektir. Daha sonra ise formlar arası veri aktarımı örnekler üzerinden anlatılacaktır.

Formlar Arası Geçiş

Bir Visual Studio projesi otomatik olarak bir form ile açılmaktadır. Projeye ikinci bir form, menüden sırası ile *PROJECT-Add Windows Form* denilip açılan pencereden *Windows Form* seçeneği seçilip formun ismi girilerek eklenmektedir.

Projeye ikinci bir form eklendikten sonra çalışma sırasında iki formunda ekranda görünmesi için ikinci form türünde bir nesne oluşturup bu *Show ya da ShowDialog* komutu ile gösterilmektedir.

Form1'de bulunan buttona tıklanınca Form2'yi açma örneği;

Örnek

```
private void FormGecis_Click (object sender, EventArgs e)
        {
                Form2 f2 = new Form2();
                    f2.Show();
                }
```

Formlar Arası Veri Aktarımı

C# Windows Form uygulamalarının nerede ise tamamında birden fazla formun kullanıldığı ve bu formlar arasında veri alış verişi yapıldığı görülmektedir. Böylece kodun okunabilirliği ve uygulamanın kullanılabilirliği artırılmaktadır.

Formlar arasında veri aktarımı yapılırken ShowDialag(); komutu veri aktarım kodlarından sonra yazılmaz ise veri aktarımı gerçekleşmeden diğer forma geçilmiş olunur.

C#'ta formlar arası veri gönderme işlemi 4 farklı şekilde yapılabilmektedir (Bükülmez, 2018).

- Formu açarken formlar arası veri gönderimi,
- Formu kapatırken formlar arası veri gönderme
- Class kullanarak formlar arası veri gönderme
- Sql ile formlar arası veri gönderme

Formlar açarken formlar arası veri gönderimi

Çok sık kullanılan bir yöntemdir. Burada ilk olarak veriyi alacak formda *public yani global* bir değişken tanımlanır. İşlem sonrası bir sonraki forma veri gönderileceği zaman bu yöntem kullanılmaktadır.

Formlar kapatırken formlar arası veri gönderimi

Bu yöntem genelde işlem sonunda, önceki forma veri göndermek için kullanılır. Örneğin öğrenci bilgilerinin ilk formda, öğrenci kaydetme işleminin ise ikinci formda yer aldığını düşünelim. Bu durumda program çalıştırılıp ikinci formda yeni bir öğrenci kaydı yapıldıktan sonra öğrenci listesinin güncellenmesi gerekir. Yani ikinci formdaki kaydın ilk forma gönderilmesi gerekir. Bu özellik bu tür durumlar için kullanılmaktadır.

Class kullanarak formlar arası veri gönderimi

Genellikle form kapatılırken veri gönderme işlemi sırasında bu yöntem sık kullanılır. Bir class oluşturulur ve bu class içinde diğer forma gönderilecek değişkenler tanımlanır.

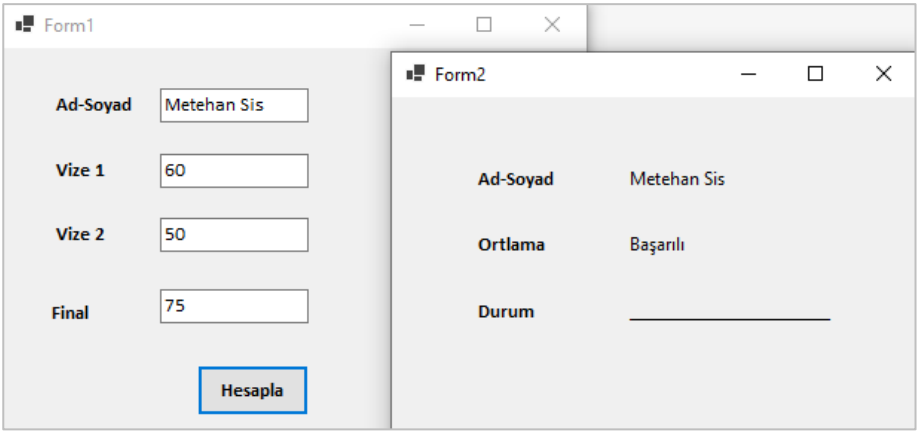
SQL kullanarak formlar arası veri gönderimi

Genellikle 2 farklı proje arasında veri aktarımı yapılacağı durumlarda kullanılır. SQL de yer alan tablodaki aktarılacak veriler diğer formdan çağrılarak veri aktarımı gerçekleştirilmiş olur.

Form1'de gösterilen öğrenci not bilgileri alınıp Form2'de ortalama hesaplaması yapan örnek uygulama aşağıda gösterilmiştir;

TextBox, Button gibi kontrolleri bir başka formda çağırmadan önce Özellikler penceresinden "Modifiers" özelliği Private olarak değiştirilmelidir.

```
private void button1_Click(object sender, EventArgs e)
{
        //Form2 nesnesi oluşturuldu.
        Form2 f2 = new Form2();
        //Form1'de ki ad soyad bilgisi Form2'ye aktarıldı.
        f2.AdSoyad.Text = TexBoxAdSoyad.Text;
        double vize1 = Convert.ToDouble(textBoxVize1.Text);
        double vize2 = Convert.ToDouble(textBoxVize2.Text);
        double final = Convert.ToDouble(textBoxFinal.Text);
        double ortalama = (vize1 + vize2 + final) / 3;
        //Form1'de hesaplanan ortalama Form2'ye aktarıldı.
        f2.labelOrtlama.Text = Convert.ToString(ortalama);
        if(ortalama<50)
        {
                f2.labelOrtlama.Text = "Başarısız";
        }
        else
        {
                f2.labelOrtlama.Text = "Başarılı";
        }
        //Form2 açıldı.
        f2.ShowDialog();
}
```



Şekil 14.5. Formlar Arası Veri Aktarımı

Bireysel Etkinlik

- Uygukamanıza 3 tane form ekleyiniz. İlk forma bir TextBox ve ComboBox yerleştiriniz. Diğer iki forma birer tane ListBox ekleyiniz. Form 1'de yer alan ComboBox'ın ilk seçeneğine tıklnınca TextBox'taki veriyi; Form 2'de ki ListBox'a, ikinci seçene tıklayınca Form3'te ki ListBox'a ekleyen uygulamayı geliştiriniz.

Özet

- C# Form Yapısı
- Arayüz tasarlanırken çeşitli form kontrollerinden ve bu kontrollerin özeliklerinden yararlanılmaktadır. Kontroller form içinde bulunan nesneler olarak tanımlanabilir. Her bir kontrol belirli bir amaç için oluşturulmuştur ve her birinin kendine ait özellikleri, yöntemleri ve olayları bulunmaktadır.
- Form Özellikleri
- AcceptButton: Enter tuşuna basılınca hangi tuşun işlem yapacağını belirtir.
- AutoScroll: Form üzerindeki nesneler forma sığmadığı durumda otomatik olarak scrollbar eklenip eklenmeyeceği bu özellik ile belirlenir.
- AutoScrollMargin: Eğer formda scrollbar var ise scrollbar" ın genişliği ve yüksekliğini ayarlar.
- AutoSize: Formun içindeki kontrollere göre formun boyutunun otomatik olarak ayarlar.
- BackColor: Form'un zemin rengini değiştirmek için kullanılır.
- BackgroundImage: Form'un arka plan resmini ayarlar.
- BackGroundImageLayout: Forma eklenen arka plan resminin nerede ve nasıl duracağını belirler.
- CancelButton: "ESC" tuşuna basılınca form üzerinde çalışacak buttonu ayarlar.
- Enabled: Bu özellik ile formun kullanıcı tarafından kullanılıp kullanılmayacağı, form üzerindeki elemanların aktif olup olmayacağı belirlenir.
- Font: Form üzerine eklenen elemanların yazı tipi özellikleri ayarlanır Form üzerindeki diğer Form nesnelerinin de fontu değişir.
- FontDialog: Sistemde o anda yüklü olan yazı tiplerini gösterir.
- FormBorderStyle: Formun boyutların değiştirilip değiştirilmeyeceğini ayarlar.
- HelpButton: Başlık çubuğunda *Yardım* düğmesinin gösterilip gösterilmeyeceği ayarlanır.
- Locked: Bu özellik ile hem formun boyutu hem de konumu değiştirilmez hale getirilir.
- Language: Form'un dilini belirler. Default değeri sistemin kurulu olduğu dildir.
- Is MDI Container: MDI (Multiple Document Interface) *Çoklu Form Arayüzü* demektir. "True" değeri seçilirse formun içinde başka bir form açılabilmektedir.
- TopMost: Form açıldığında başka form varsa, diğer tüm formların üzerinde gözüküp gözükmeyeceğin ayarlar.
- Text: Formun sağ üstteki ikonun yanındaki yazıyı değiştirmek için kullanılır.
- TopMost: Form açıldığında başka form varsa, diğer tüm formların üzerinde gözüküp gözükmeyeceğin ayarlar.
- WindowsState: Formun tam ekran mı yoksa normal boyutlarda mı gösterileceğini ayarlar.
- StartPosition: Uygulama çalıştırıldığında Formun, ekranın neresinde açılacağını belirtir.
- Size: Formun boyutlarını ayarlar.

Özet (devamı)

- Name: Bu özellik kullanılarak formun adı değiştirilebilir.
- Form özleliklerinin yanı sıra işlem yapılmasını sağlayan "Click", "Closing","KeyDown", "KeyUp","Load" gibi olayları mevcuttur.
- Form Kontrolleri ve Özellikleri
- TextBox :Kullanıcının çok satırlı veri girmesini sağlar.
- Button: Kullanıcının işlem yapmak için tıklayabileceği düğme olarak tanımlanabilir.
- Label: Kullanıcıya bilgi vermek için kullanılır.
- CheckBox:Kullanıcının bir ya da birden fazla seçeneği seçmesine olanak tanır.
- ComboBox: Açılır bir menüde verilerin görüntülenmesini ve kullanıcıya veri girişi yapmayı sağlar.
- CheckedListbox: Her bir öğenin yanında onay kutusuyla birlikte bir öğe listesi görüntüler.
- DataGridView: Verilerin, hücrelerden ve satırlardan oluşan tablolarda ki gösterimini sağlayan bir kontroldür.
- DataTimePicker: Tarih veya saat bilgisini gösterir.
- GroupBox:Form üzerinde yer alan diğer kontrollerin gruplanmasını sağlar.
- ListBox: Kullanıcının bir listeden bir veya daha fazla öğe seçmesini sağlayan kontroldür.
- OpenFileDialog: Kullanıcıların bir iletişim kutusu kullanarak dosyaları açmasına izin verir.
- PageSetupDialog: Kullanıcının yazdırma işlemleri sırasında sayfa yapısını ayarlamayı sağlar.
- PrintDialog: Kullanıcılar herhangi bir belgeyi yazdırırken yazdırma özelliklerinin yer aldığı pencerenin açılmasını sağlar.
- PrintDocument: Yazdırma işlemleri için kullanılır.
- PrintPreviewDialog:Yazdırma işlemi sırasında Baskı Ön İzleme için kullanılır.
- RadioButton:Kullanıcıya iki veya daha fazla seçenek sunmayı sağlar.
- RichTextBox: Kullanıcıların metin girmesine, görüntülemesine ve güncellemesine izin verir.
- MenuStrip: Form üzerine menü eklenmesini sağlar.
- ContexMenuStrip: Form üzerinde, farenin sağ tuşu ile tıklanınca görünen menü eklenmesini sağlar.
- ToolStrip: Form üzerinde araç çubuğu menüsünün eklenmesini sağlar.
- StatusStrip: Kullanıcıya bilgi vermek için kullanılan *Bilgi Şeridi* olarak da adlandırılan menünün eklenmesini sağlar.
- ÇOKLU FORMLAR
- Formlar Arası Geçiş:. Projeye ikinci bir form, menüden sırası ile *PROJECT-Add Windows Form* denilip açılan pencereden *Windows Form* seçeneği seçilip formun ismi girilerek eklenmektedir. İkinci formun açılması için nesne oluşturup bu *Show ya da ShowDialog* komutu ile gösterilmektedir.
- Formlar Arası Veri Aktarımı
- C#'ta Form açarken, Form kapatırken, Class kullanarak, SQL ile bir formdan diğer bir forma bilgi akatrımı yapılabilir.

DEĞERLENDİRME SORULARI

1. Aşağıdaki özelliklerden hangi Form kontrolünün aktif ya da pasif olmasını sağlar?
   a) Enabled
   b) StartPosition
   c) Font
   d) HelpButton
   e) Locked

2. Aşağıdaki özelliklerden hangisi form kontrolüne ait bir özellik değildir?
   a) BackColor
   b) AutoScroll
   c) AcceptButton
   d) Closing
   e) BackgroundImage

3. Aşağıdaki kontrollerden hangisi kullanıcının veri girmesini sağlar?
   a) Button
   b) CheckBox
   c) RadioButton
   d) ListBox
   e) TextBox

4. Aşağıdaki kontrollerden hangisi kullanıcıya bir ya da birden fazla seçeneği seçme olanağı tanır?
   a) ComboBox
   b) CheckBox
   c) TextBox
   d) Button
   e) Label

5. Yazdırma işlemi sırasında yazdırma penceresinin açılması için aşağıdaki sınıflardan hangisi kullanılır?
   a) OpenFileDialog
   b) PageSetupDialog
   c) PrintPreviewDialog
   d) PrintDocument
   e) PrintDialog

6. Aşağıdaki form özelliklerinden hangisi kullanılarak form boyutunun değiştirilmesi önlenir?
   a) Font
   b) AutoSize
   c) FormBorderStyle
   d) StartPosition
   e) TopMost

7. Aşağıdaki özelliklerden hangisi çoklu form kullanılmasını sağlar?
   a) WindowsState
   b) Is MDI Container
   c) Language
   d) CancelButton
   e) FormBorderStyle

8. Aşağıdaki kontrollerden hangisi kullanılarak form üzerine araç çubuğu eklenebilir?
   a) ToolStrip
   b) StatusStrip
   c) MenuStrip
   d) ContextMenuStrip
   e) PrintDialog

9. Aşağıdaki özelliklerden hangisi veri tabanı ile bağlantı yapılmak için kullanılır?
   a) ReadOnly
   b) Items.Count
   c) DataSource
   d) Selecteditem
   e) MultiSelect

10. Kontrollerle ilgili aşağıdakilerden hangisi yanlıştır?
    a) DateTimePicker tarih veya saat bilgisini göstermek için kullanılan bir kontroldür.
    b) StatusStrip kullanıcıya bilgi vermek için kullanılan bir menü kontrolüdür.
    c) Label kullanıcıya bilgi vermek için kullanılır.
    d) ShowDialog() komutu aynı anda iki form üzerinde işlem yapılmasına izin verir.
    e) PrintDocument yazdırma işlemi için kullanılır.

Cevap Anahtarı

1.a, 2.d, 3.e, 4.b, 5.e, 6.c, 7.b, 8.a, 9.c, 10.d

YARARLANILAN KAYNAKLAR

Alan, E. (2011). Formlar ve Windows Form Kontrolleri. 06 Eylül 2021 tarihinde https://doczz.biz.tr/doc/314594/formlar-ve-windows-forms-kontrolleri adresinden erişildi.

Alkan, F. (2018). Windows Form Kontrolleri. 05 Eylül 2021 tarihinde https://alkanfatih.com/windows-form-kontrolleri/ adresinden erişildi.

Bükülmez, M. (2018). C# Formlar Arası Veri Nasıl Gönderilir. 08 Eylül 2021 tarihinde https://mustafabukulmez.com/2018/01/22/c-formlar-arasi-veri-nasil-gonderilir/ adresinden erişildi.

Salkım, G. (2012). C# 'da Form Kontrolünün Özellikleri. 05 Eylül 2021 tarihinde http://gaffarsalkim.blogspot.com/2012/04/c-da-form-kontrolunun-ozellikleri.html adresinden erişildi.