DİYALOG VE MESAJ PENCERELERİ

İÇİNDEKİLER

- MessageBox
- Diyalog Pencereleri
  - ColorDialog
  - FolderBrowserDialog
  - FontDialog
  - OpenFileDialog
  - SaveFileDialog
- InputBox Kullanımı ve Parametreleri

HEDEFLER

- Bu üniteyi çalıştıktan sonra;
  - MessageBox penceresini kullanabilecek,
  - Diyalog pencere türlerini tanıyabilecek,
  - Diyalog pencerelerini uygulamalarınızda kullanabilecek,
  - C# ile InputBox kullanımını gerçekleştirebileceksiniz.

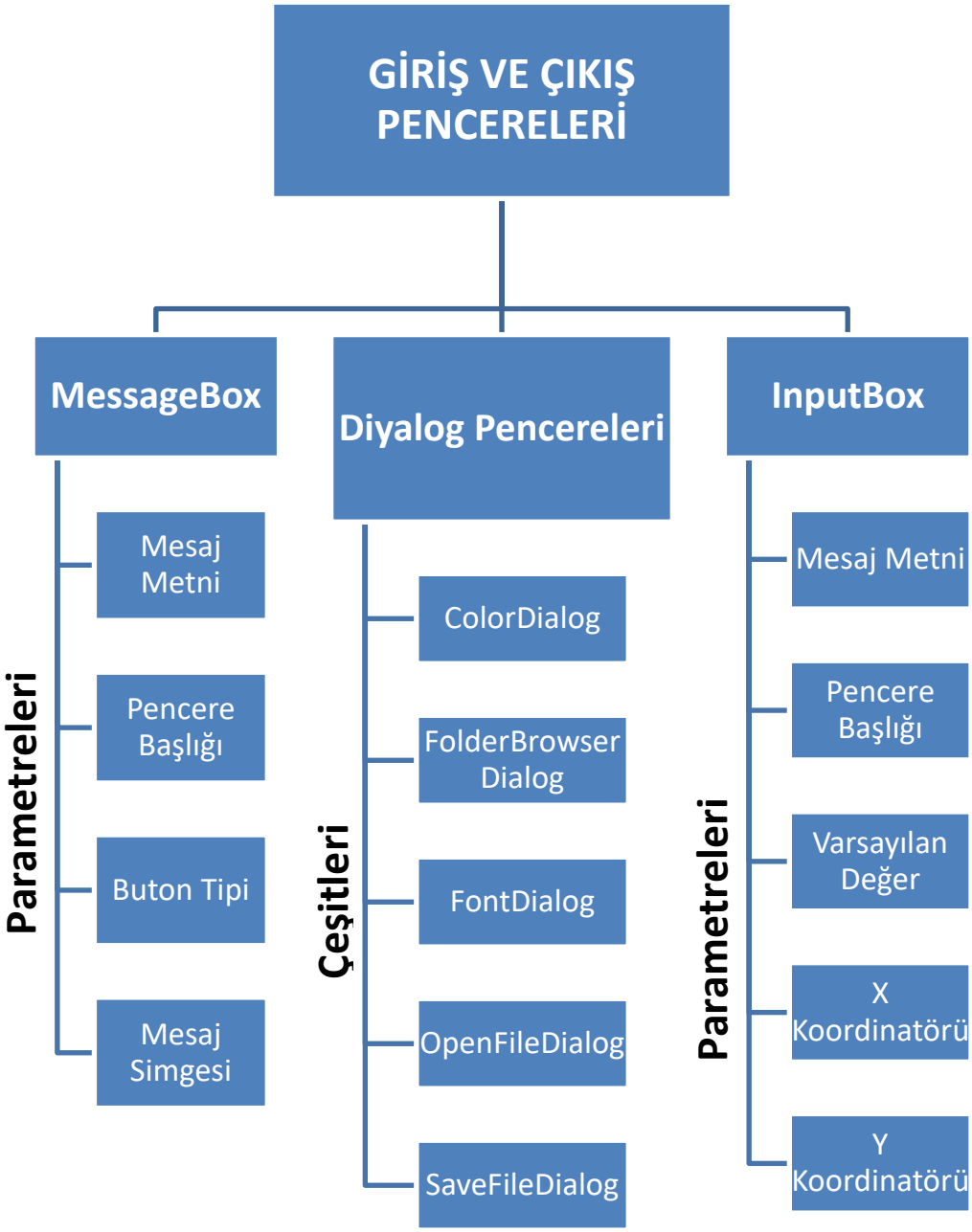
Atatürk Üniversitesi
Açıköğretim Fakültesi

GÖRSEL PROGRAMLAMA I

Öğr. Gör.
Daha ORHAN

ÜNİTE

11


GİRİŞ VE ÇIKIŞ PENCERELERİ
MessageBox
Diyalog Pencereleri
InputBox
Parametreleri
Mesaj Metni
Pencere Başlığı
Buton Tipi
Mesaj Simgesi
Çeşitleri
ColorDialog
FolderBrowserDialog
FontDialog
OpenFileDialog
SaveFileDialog
Parametreleri
Mesaj Metni
Pencere Başlığı
Varsayılan Değer
X Koordinatörü
Y Koordinatörü

GİRİŞ

Bilgisayarlar günlük hayatımızdaki işlemleri kolaylaştırmak adına her geçen gün daha hızlı bir şekilde geliştirilmektedirler. Doğru orantılı olarak bilgisayarların etkili bir şekilde kullanılabilmesi ve makine ile kullanıcı arasındaki iletişimin kolaylıkla sağlanabilmesi için programlama dillerine de farklı seçenekler eklenmektedir. Konsol ortamında geliştirilen uygulamalarda bulunan sınırlılıklar, görsel programlama dillerinin geliştirilmesiyle ortadan kalkmıştır. Bu sınırlılıkları kullanıcı ile bilgisayar arasındaki bilgi alışverişinden yazılım geliştiricilere sunulan seçeneklere kadar birçok alanda gözlemleyebiliriz. Aynı zamanda bilgisayar ile kullanıcı arasındaki iletişimin kolaylık sağlanması ve daha göze hitap edecek şekilde tasarlanması, yani görselliğin ve iletişimin yüksek seviyede olması uygulamaların kullanım tercihine de önemli katkı sağlamaktadır.

Windows form uygulamalarında kullanılan diyalog ve mesaj pencereleri görsel programlama dillerinin önemli bir parçası olarak karşımıza çıkmaktadır. Yazılım geliştiricileri, bahsi geçen diyalog ve mesaj pencerelerini tamamen kendileri tasarlayabileceği gibi .Net platformu tarafından kullanıma hazır bir şekilde sunulan seçenekleri de kullanabilmektedir. Projenin özelliğine ve tasarımına bağlı olarak bilgi giriş ve çıkışı için bu pencerelerin özel olarak hazırlanması gerekebilir. Ancak standardın sağlanabilmesi, kullanıcı açısından bilinirliğin göz önünde bulundurulması, tasarımla harcanan zamanın ortadan kaldırılması, ortak metotlarının kolaylıkla kullanılabilmesi gibi seçenekler de hazır pencerelerin kullanım avantajları arasında gösterilebilir.

MessageBox penceresiyle kullanıcıya gösterilecek olan metnin uzunluğu en fazla 1024 karakter olabilmektedir.

Bu ünitede bilgi vermek amacıyla kullanılan Messagebox ve bilgi girişi sağlamak amacıyla kullanılan ColorDialog, FolderBrowserDialog, FontDialog, OpenFileDialog ve SaveFileDialog pencereleri anlatılacaktır. Ek olarak C# içerisinde mevcut olmayan InputBox kontrolünün nasıl kullanılabileceğine yönelik açıklama ve uygulamalara da yer verilecektir.

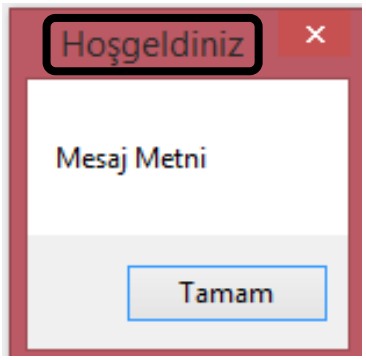
MESSAGEBOX

Uygulamalar çalışırken üretilen sonuçların ve bilgilerin kullanıcıya ve programcıya gösterilmesi amacıyla MessageBox penceresi kullanılmaktadır. Bu pencere yardımıyla kullanıcıya gösterilecek olan metnin uzunluğu en fazla *1024* karakter olabilmektedir. Bu pencerede zorunlu olan ilk parametre yani mesaj bilgisi *String* veri türünde olmalıdır. Kullanıcıya bilgi vermek için program akışı içerisinde MessageBox sınıfına ait Show metodunu kullanmamız gerekmektedir. İlk parametresi zorunlu diğerleri isteğe bağlı olan Show metodunun kullanımı aşağıda gösterilmektedir.

```
MessageBox.Show( MesajMetni, MesajBaşlığı, ButonTipi, MesajSimgesi);
```

Burada zorunlu olan ilk parametreye kullanıcıya iletmek istediğimiz metni yazabiliriz. String türünde olan bu ifade çift tırnak *(“”)* içine alınarak yazılmalıdır. Eğer buraya bir değişken içinden farklı türde bir bilgi aktarmak istiyorsak dönüştürmemiz (convert) gerekmektedir.

Kullanım şekli gösterilen Show metodunun ikinci parametresi kullanıcıya gösterilecek olan mesajın başlığını yazacağımız alanı ifade etmektedir. Bu parametre isteğe bağlı olarak ayarlanmaktadır. Aşağıdaki Şekil 11.1'de pencere başlığı *"Hoşgeldiniz"* olarak ayarlanmış bir MessageBox penceresinin ekran görüntüsü bulunmaktadır.



Şekil 11.1. Messagebox Pencere Başlığı

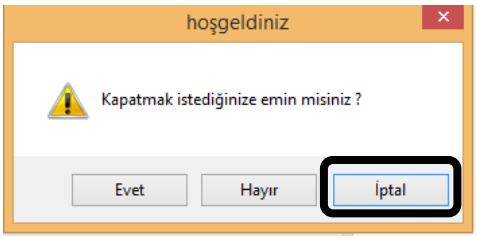
Üçüncü parametre, MessageBox penceresinde bulunan butonun hangi işlemlere izin vereceğine yönelik tipini belirlememize yardım etmektedir. Varsayılan değer olarak çalıştırdığımızda pencere üzerinde sadece *Tamam* butonu görüntülenmektedir. Burada görüntüleyebileceğimiz buton tipleri:

- *MessageBoxButtons.OKCancel*: Bu seçenek ile pencere içinde *Tamam* ve *İptal* butonları bulunmaktadır.
- *MessageBoxButtons.OK*: Bu seçenek ile pencere içinde sadece *Tamam* butonu görüntülenir.
- *MessageBoxButtons.AbortRetryIgnore*: Bu seçenek ile pencere içinde *Durdur*, *Yeniden Dene* ve *Yoksay* butonları bulunmaktadır.
- *MessageBoxButtons.RetryCancel*: Bu seçenek ile pencere içinde *Yeniden Dene* ve *İptal* butonları bulunmaktadır.
- *MessageBoxButtons.YesNo*: Bu seçenek ile pencere içinde *Evet* ve *Hayır* butonları bulunmaktadır.
- *MessageBoxButtons.YesNoCancel*: Bu seçenek ile pencere içinde *Evet*, *Hayır* ve *İptal* butonları bulunmaktadır.

MessageBox pencereleri birden fazla butona sahip olduğunda varsayılan değer olarak ilk buton aktif konumda bulunur.

MessageBox pencereleri birden fazla butona sahip olduğunda varsayılan değer olarak ilk buton aktif konumda bulunur. *Enter* tuşuna bastığımız zaman aktif olan butonun işlevi gerçekleştirilir. Uygulamalarımızda aktif olan butonun farklı bir buton olmasını ayarlayabiliriz. Bunun için DefaultButton kodunu kullanmamız gerekmektedir. Üç butonun olduğu bir pencerede ilk buton için DefaultButton1, ikinci buton için DefaultButton2, üçüncü buton için de DefaultButton3 kodunu yazmamız gerekir. Örneğin Şekil 11.2.'de gösterildiği gibi evet, hayır ve iptal butonlarının bulunduğu bir pencerede iptal butonunu aktif buton olarak ayarlamak için aşağıdaki kodu yazmamız gerekmektedir.

```
MessageBox.Show("Kapatmak istediğinize emin misiniz?", "hoşgeldiniz",
MessageBoxButtons.YesNoCancel,
MessageBoxIcon.Warning,MessageBoxDefaultButton.Button3);
```



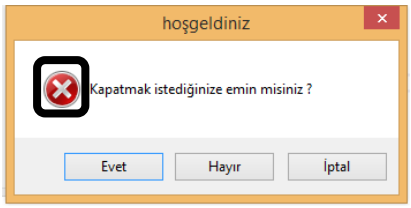
Şekil 11.2. Varsayılan Aktif Buton

MessageBox penceresinde bulunan butonların tepki vermesi için hangi butonun tıklandığını bilmemiz gerekmektedir. Böyle bir durumda *DialogResult* nesnesini kullanıp programın döndürdüğü değer doğrultusunda yapılacak işlemleri belirleyebiliriz.

Show metoduna ait dördüncü parametreyle de MessageBox içinde görüntülenebilecek ikonlar belirlenebilir. Programın bize sunmuş olduğu 4 farklı ikon, bu ikonlara ulaşmak için 8 farklı alternatif bulunmaktadır.

MessageBox içerisinde *Dur* işaretini ifade etmek için kırmızı bir daire içinde beyaz renkli bir çarpı işaretinin bulunduğu ikonu üç farklı alternatifle görüntüleyebiliriz. Bunlar aşağıda maddeler halinde listelenmiştir. Böyle hazırlanmış bir MessageBox penceresine ait ekran görüntüsü de Şekil 11.3.'de gösterilmektedir.

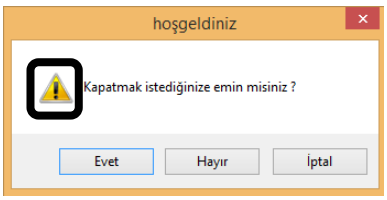
- MessageBoxIcon.Error
- MessageBoxIcon.Hand
- MessageBoxIcon.Stop



Şekil 11.3. Messagebox Dur İşareti

MessageBox içerisinde *uyarı* yapıldığını ifade emek için sarı bir arkaplana sahip üçgen içinde ünlem işaretinin bulunduğu ikonu iki farklı alternatifle görüntüleyebiliriz. Bunlar aşağıda maddeler halinde listelenmiştir. Böyle hazırlanmış bir MessageBox penceresine ait ekran görüntüsü de Şekil 11.4.'de gösterilmektedir.

- MessageBoxIcon.Exclamation
- MessageBoxIcon.Warning

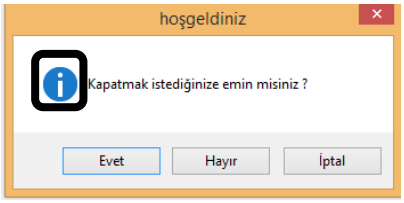


Şekil 11.4. Messagebox Uyarı İşareti

MessageBox içerisinde *bilgilendirme* yapıldığını ifade etmek için mavi arkaplana sahip daire içinde *"i"* harfinin bulunduğu ikonu iki farklı alternatifle

görüntüleyebiliriz. Bunlar aşağıda maddeler halinde listelenmiştir. Böyle hazırlanmış bir MessageBox penceresine ait ekran görüntüsü de Şekil 11.5.'de gösterilmektedir.

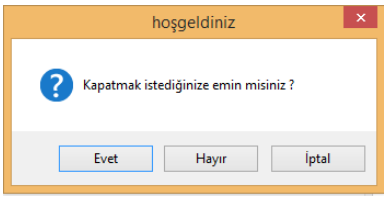
- MessageBoxIcon.Information
- MessageBoxIcon.Asterisk



Şekil 11.5. Meesagebox Bilgilendirme İşareti

Son olarak mavi arkaplana sahip bir daire içinde soru işaretinin bulunduğu ikon ile MessageBox penceresini görüntüleyebiliriz. Bunun için tek bir alternatif bulunmaktadır. Çünkü belirli bir mesaj türünü açık bir şekilde temsil etmemesi ve herhangi bir mesajın soru olarak ifade edilmesi için başka bir seçeneğe gerek duyulmamıştır. Ayrıca soru işareti mesaj sembolü yardım bilgisinin verildiği alanlarda da kullanılmaktadır. Bu nedenle kullanıcıların zihninde karışıklığa yol açmamak için, MessageBox penceresinin içinde bu sembolün kullanılması tavsiye edilmemektedir. Bu ikonu pencere üzerinde görüntülemek için kullanacağınız yöntem aşağıda verilmiştir. Böyle hazırlanmış bir MessageBox penceresine ait ekran görüntüsü de Şekil 11.6.'da gösterilmektedir.

- MessageBoxIcon.Question



Şekil 11.6. Messagebox Soru İşareti

MessageBox kontrolü ile kullanıcının görmesini istediğimiz metni, satırlar halinde sunabiliriz.

Bazı durumlarda kullanıcının görmesini istediğimiz metni, satırlar halinde sunmak isteyebiliriz. Bunun için metin bilgisine yer verdiğimiz parametre kısmında yeni satır anlamına gelen *NewLine* ifadesi veya *"\n"* karakteri kullanılmalıdır. Örneğin Taha ve Orhan isimlerinin alt alta görüntülenmesi için kullanabileceğimiz iki farklı yöntem bulunmaktadır. Bunlar aşağıda gösterilmiştir. Verilen kodların herhangi birini kullanarak hazırlayacağımız programın ekran görüntüsü de Şekil 11.7.'de gösterilmektedir.

- MessageBox.show("Taha \nOrhan");
- MessageBox.show("Taha" + Environment.NewLine + "Orhan");

Şekil 11.7. Messagebox Kontrolünün Çok Satırlı Kullanımı

DİYALOG PENCERELERİ

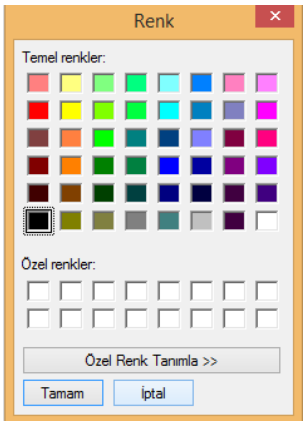
Diyalog pencereleri kullanıcının etkileşime girmesini sağlayan ve açılan pencereyi yanıtlamasını bekleyen yapıdaki grafiksel denetim öğeleridir. Windows form uygulamalarına ekleyebilmek için araç kutusunda bulunan *İletişim Kutuları* sekmesi kullanılabilir. Bu alanda yer alan bütün diyalog pencereleri *ShowDialog* metodu kullanılarak görüntülenmektedir. Aynı zamanda kullanıcıdan gelen yanıt yani basılan tuş *DialogResult* özelliği kullanılarak alınmaktadır.

Uygulamalarımıza diyalog pencerelerini eklediğimiz tasarım alanında çalışırken form üzerinde görüntülenmezler. Visual Studio IDE penceresinin alt tarafında açılan panel üzerinde konumlanmaktadır.

ColorDialog

ColorDialog kontrolü kullanıcının renk seçme işlemini gerçekleştirmesi için kullanılan pencerenin açılmasını sağlamaktadır. Çalışma anında bu pencerenin görüntülenmesine yönelik işlem gerçekleştiğinde Şekil 11.8.'de bulunan ekran görüntüsü elde edilecektir. Renk başlığı adı altında açılan pencere üzerinden temel renklerden biri seçilebileceği gibi *Özel Renk Tanımla* seçeneği de kullanılabilmektedir. Özel renk tanımla seçeneği ile açılan alan üzerinde fare imleci ile herhangi renk seçimi yapılabilmektedir. Ek olarak seçilecek olan renge ait *Kırmızı, Yeşil* ve *Mavi* kodları manuel olarak da girilebilmektedir. Eğer belirlen bu renk sıklıkla kullanılıyorsa *Özel Renklere Ekle* butonu kullanılarak *Özel Renkler* arasına eklenebilir.

ColorDialog kontrolü kullanıcının renk seçme işlemini gerçekleştirmesi için kullanılan pencerenin açılmasını sağlamaktadır.



Şekil 11.8. Colordialog

Çalışma anında ColorDialog penceresi özelleştirilebildiği gibi varsayılan ayarlarına geri dönülmesi de sağlanabilmektedir. Bu işlem için kontrol *Reset* metodunu kullanmak gerekir. Kod editöründe yazabileceğimiz bu komutun örnek gösterimi aşağıda verilmiştir.

- ColorDialogAdi.Reset();

Renk penceresi açıldığı zaman kullanıcının tercihine bırakmadan özel renk tanımla alanının da otomatik olarak açılmasını isteyebiliriz. Böyle bir durumda kontrole ait özellikler panelinde bulunan *FullOpen* seçeneğini kullanmamız gerekmektedir. FullOpen özelliğine *True* değeri aktarıldığında pencere özel renk tanımla alanıyla birlikte açılacaktır.

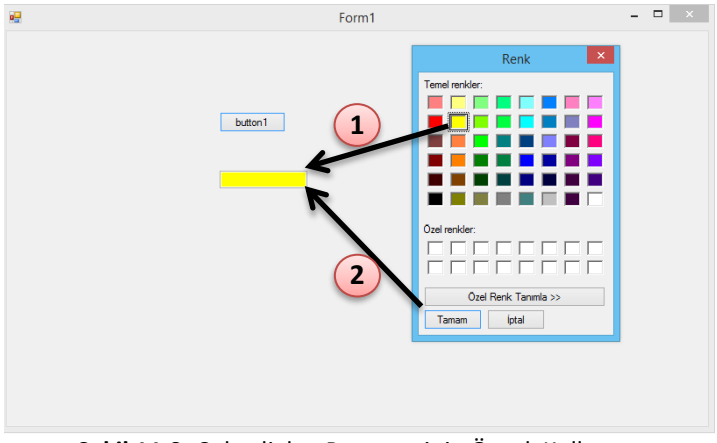
Renk seçim işlemi gerçekleştirildikten sonra *Tamam* butonuna tıklanarak ilgili alana uygulanabilir. Bu pencerede gerçekleştirilen işlemin hatalı olduğu düşünülüyorsa İptal butonu veya pencereye ait kapatma butonu kullanılmalıdır. Konunun daha iyi anlaşılması için aşağıdaki örneği inceleyelim.

Örnek

- Form üzerine yerleştirilen TextBox kontrolünün arkaplan rengini buton yardımıyla colordialog penceresinden seçilen renk ile değiştirme işleminin yapıldığı uygulamaya ait ekran görüntüsü Şekil. 11.9.'da verilmiştir. Buton içine yazılan kodlar da aşağıda verilmiştir.

```
private void button1_Click(object sender, EventArgs e)
        {
        colorDialog1.ShowDialog();
        textBox1.BackColor = colorDialog1.Color;
        }
```

Bilgisayarda bulunan bir klasörü seçmek için FolderBrowserDialog kontrolünü kullanmak gerekmektedir.

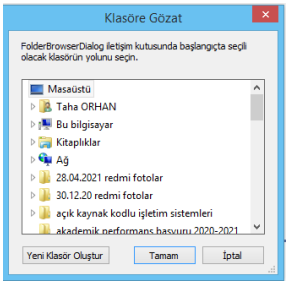


Şekil 11.9. Colordialog Penceresinin Örnek Kullanımı

FolderBrowserDialog

Bilgisayarda bulunan bir klasörü seçmek için FolderBrowserDialog kontrolünü kullanmak gerekmektedir. Bu kontrol sayesinde bilgisayarda seçilen bir

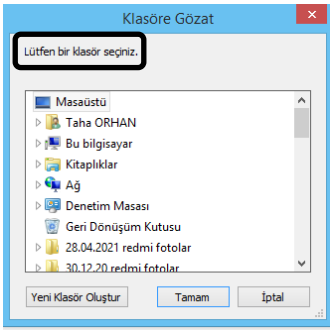
klasörün yolunu özellikler panelinde bulunan *SelectedPath* seçeneğinde tutabiliriz. FolderBrowserDialog uygulamaya eklendikten sonra SelectedPath özelliğine tıklayarak Şekil 11.10'da gösterilen *Klasöre Gözat* penceresi açılmaktadır.



Şekil 11.10. Klasöre Gözat Penceresi

Klasöre Gözat penceresi bilgisayarda bulunan klasörleri hiyerarşik bir yapı içinde göstermektedir. Bu alan üzerinde klasörlerin yanındaki oklara tıklanarak alt klasörler görüntülenip gizlenebilmektedir. Aynı zamanda bir klasör seçildikten sonra *Yeni Klasör Oluştur* butonuna tıklanarak ilgili klasör altında yeni klasörler oluşturulabilmektedir. Eğer kullanıcının böyle bir yetkiye sahip olmasını istemiyorsanız, özellikler panelinde bulunan *ShowNewFolderButton* seçeneğine *False* değerini atamanız gerekmektedir. Yapılan işlemleri onaylamak için *Tamam*, gerçekleştirmeden çıkmak için *İptal* butonu kullanılmalıdır.

Çalışma anında kullanıcının Klasörlere Gözat penceresini açtığı zaman hiyerarşik yapının üst kısmındaki alan üzerinde bir bilgilendirme metni görmesini sağlayabilirsiniz. Bu işlemi gerçekleştirmek için kontrole ait Description özelliğine görülmesi istenen metin bilgisinin yazılması gerekmektedir. Bu alana "Lütfen bir klasör seçiniz." yazıldıktan sonra elde edilen ekran görüntüsü Şekil 11.11'de gösterilmiştir.



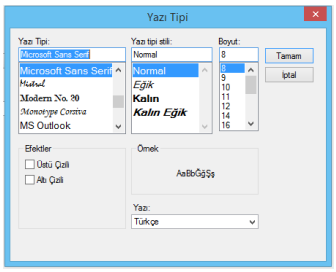
Şekil 11.11. Folderbrowserdialog Kontrolünün Description Özelliği

Kaynak seç penceresinde görüntülenen hiyerarşik yapının hangi klasörden başlanacağı RootFolder özelliğiyle belirlenir.

Kaynak seç penceresinde görüntülenen hiyerarşik yapının hangi klasörden başlanacağı belirlenebilmektedir. Bu işlem için özellikler panelinde bulunan RootFolder seçeneği kullanılmaktadır.

FontDialog

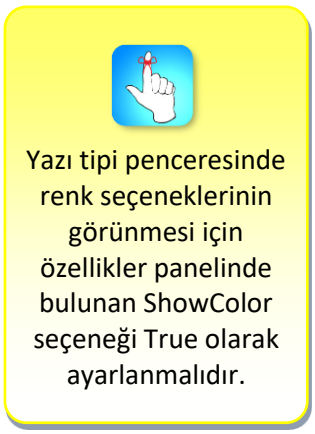
Araç kutusu üzerinde bulunan FontDialog kontrolü sayesinde uygulamamızdaki metinlerin yazı tipi özelliklerini belirleyebilir ve değiştirebiliriz. Kontrolü formumuza ekledikten sonra özellikler panelinde bulunan Font seçeneğine tıklayarak *Yazı Tipi* penceresini açabiliriz. Bu pencerenin ekran görüntüsü Şekil 11.12.'de yer almaktadır.



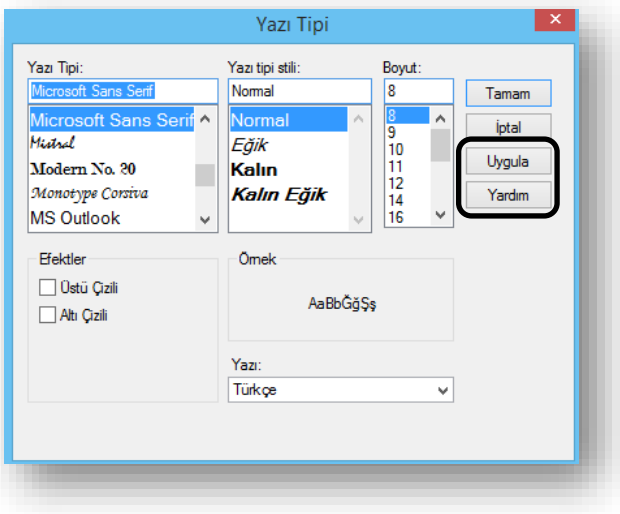
Şekil 11.12. Yazı Tipi Penceresi

Yazı tipi penceresinde metinler üzerinde değişiklik yapabileceğimiz *Yazı Tipi, Yazı tipi stili, Boyut, Efektler, Örnek* ve *Yazı* alanları bulunmaktadır. Bu alanlar sistemde kurulu olan seçenekleri içermektedir. Gerçekleştirilen değişikliklerin ön izlemesi Örnek alanında görüntülenmektedir. Yapılan bu değişiklilerin uygun olduğunu düşünüyorsanız *Tamam* butonuna tıklayarak onaylayabilir veya değişikliklerin etkin hale gelmemesi için *İptal* butonuna tıklayarak formunuza dönebilirsiniz.

Çalışma anında yazı tipi penceresinde yapılan değişikliklerin örnek alanında görüntülenmesi sağlanabildiği form üzerinde de nasıl görüneceği gözlemlenebilir. Bunun için Yazı Tipi penceresinde bulunan Tamam ve İptal butonlarına ek olarak *Uygula* butonunu da eklememiz gerekmektedir. Özellikler panelinde bulunan *ShowApply* seçeneği *True* olarak ayarlandığı zaman bu işlem gerçekleşmiş olur. Aynı zamanda yazı tipi penceresine yardım butonu da eklenebilmektedir. Yardım butonunu ekleyebilmek için özellikler panelinde bulunan *ShowHelp* seçeneği *True* olarak ayarlanmalıdır. Uygula ve Yardım butonlarının yazı tipi penceresine eklenmiş halinin ekran görüntüsü Şekil 11.13.'de gösterilmektedir.



Yazı tipi penceresinde renk seçeneklerinin görünmesi için özellikler panelinde bulunan ShowColor seçeneği True olarak ayarlanmalıdır.



Şekil 11.13. Uygula ve Yardım Butonları

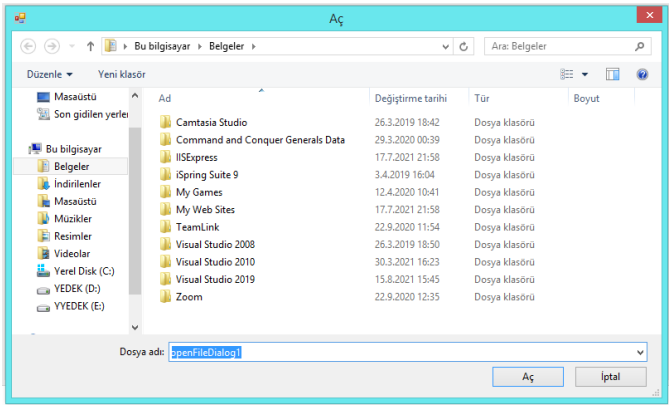
Bazı uygulamalarda metinlerin yazı tiplerinin renklendirilmesine de ihtiyaç duyulabilmektedir. Bu işlem detaylı olarak ColorDialog penceresinden yapılabilmektedir. Ancak ek bir kontrole gerek duymadan FontDialog penceresinin sunduğu ön tanımlı 16 renk kullanılabilir. Bu renklerin Yazı tipi penceresinde görünmesi için özellikler panelinde bulunan *ShowColor* seçeneği *True* olarak

ayarlanmalıdır. Böylelikle yazı tipi penceresinde bulunan *Efektler* alanına *Renk* başlığı altında seçim yapabileceğimiz açılır bir liste eklenmiş olacaktır. Bu listede yaptığımız seçimi *Tamam* butonuna tıklayarak formumuz üzerinde görüntüleyebiliriz.

Uygulamanın çalışma anında yazı tipi penceresinde yapılan değişikliklerle, tasarımın etkilenmemesi için kullanıcıya sınırlamalar getirebilirsiniz. Bazı durumlarda yazı tipi boyutunun çok yüksek seviyelere çıkarılması form üzerinde bulunan kontrollerin hoş olmayan bir düzende görünmesine sebep olabilir. Bu yüzden kullanıcının seçebileceği en büyük ve en küçük yazı tipi boyutu belirlenmelidir. Yapılacak olan sınırlama işlemi özellikler panelinde bulunan *MaxSize* ve *MinSize* alanları kullanılarak ayarlanabilmektedir.

OpenFileDialog

OpenFileDialog kontrolü, kullanıcının bilgisayarda kayıtlı olan dosyaları açması için kullanılan pencerenin ekranda görüntülenmesini sağlamaktadır. Form üzerine eklenen bu kontrolün çalışma anındaki ekran görüntüsü Şekil 11.14.'de gösterilmektedir.



Şekil 11.14. Openfiledialog Aç Penceresi

Çalışma anında pencere başlığı varsayılan değer olarak *"Aç"* adıyla karşımıza gelmektedir. Pencere başlığını değiştirmek için özellikler panelinde bulunan *Text* alanını düzenlememiz gerekir. Aynı zamanda bu pencerede bulunan *Dosya adı* bölümüne atanmak istenen değer, özellikler panelindeki *FileName* seçeneği ile ayarlanabilmektedir.

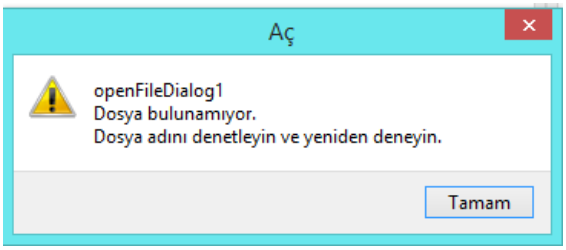
OpenFileDialog kontrolü, kullanıcının bilgisayarda kayıtlı olan dosyaları açması için kullanılan pencerenin ekranda görüntülenmesini sağlamaktadır.

ShowDialog metodu kullanılarak ekrana getirilen dosya açma penceresinde listelenecek dosya türlerinin sınırlı olması sağlanabilmektedir. Bu işlem için *Filter* özelliğini kullanmak gerekir. Filtreleme işleminde öncelikle görüntülenecek dosya türüne ait bir açıklama metni, sonra dikey çubuk, daha sonrada dosya uzantısı belirtilmelidir. Bu yöntem kullanıldığı zaman tek bir dosya türüne ait dosyalar listelenmektedir. Listede farklı dosya türlerinin de görüntülenmesi, dikey çubuk konularak yapılabilmektedir. Bu şekilde yapılacak olan filtreleme işlemi özellikler panelinden ayarlanabileceği gibi kod editöründe de gerçekleştirilmektedir. Kod editörü kullanılarak yapılan filtreleme işleme aşağıda örneklendirilmiştir.

- OpenFileDialogAdi.Filter = "AçıklamaMetni1|*.DosyaTürü1| AçıklamaMetni2|*.DosyaTürü2| AçıklamaMetni3|*.DosyaTürü3 |*.DosyaTürü4";

Örnek incelendiği zaman 4 farklı dosya türünün OpenFileDialog içinde listelenmesi sağlanmaktadır. Burada ilk üç dosya türü için açıklama yazılıp dördüncü için yazılmadığı görülmektedir. Yani kullanılan özelliğin zorunlu olan iki parametresi *Dikey Çubuk* ve *Dosya Türü* olarak karşımıza çıkmaktadır. Ayrıca listeleme işlemlerinde *"*","?"* gibi özel karakterlerin kullanımı ile filtreleme olayı daha detaylı bir şekilde gerçekleştirilebilmektedir.

OpenFileDialog penceresi açıldığı zaman kullanıcı dosya yolunu belirtmeden veya var olmayan bir dosya yolunu belirterek *Aç* butonuna tıklayabilir. Böyle bir durum karşısında kullanıcıya uyarı vermek için özellikler panelinde bulunan *CheckPathExists* seçeneğine *True* değerini atamamız gerekmektedir. Benzer şekilde kullanıcı herhangi bir dosyayı seçmeden veya var olmayan bir dosyayı göstererek *Aç* butonuna tıklayabilir. Bu durum karşısında da kullanıcıya uyarı vermek için *CheckFileExists* seçeneğine *True* değerini atamamız gerekmektedir. CheckFileExists seçeneğinin True olarak ayarlandığı durumda kullanıcıya verilen uyarı mesajının ekran görüntüsü Şekil 11.15.'de gösterilmiştir.



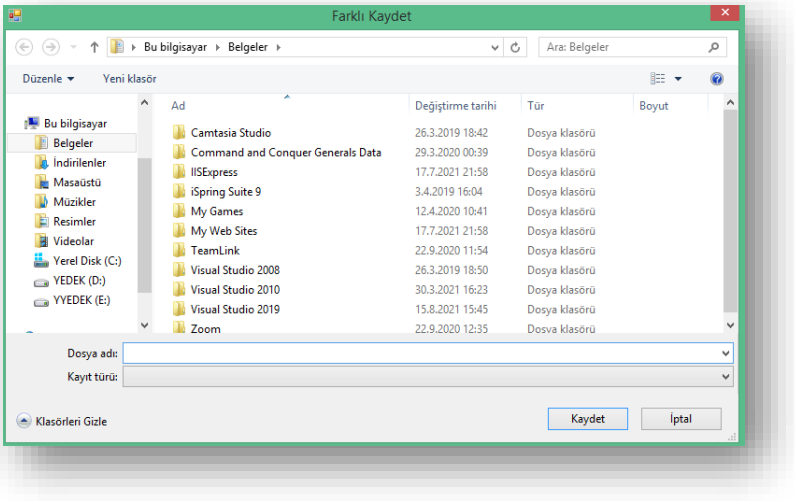
Şekil 11.15. Openfiledialog Kontrolünün Uyarı Mesajı

Bazı durumlarda kullanıcı birden fazla dosya seçmek isteyebilir. Kullanıcının böyle bir işlem gerçekleştirmesine izin vermek için özellikler panelinde bulunan *Multiselect* seçeneğini True olarak ayarlamamız gerekmektedir. Böylelikle çalışma anında açılan pencerede; kullanıcı Fareyi, Ctrl tuşunu veya Shift tuşunu kullanarak birden fazla dosya seçebilmektedir.

SaveFileDialog

SaveFileDialog kontrolü sayesinde kullanıcının herhangi bir dosyayı seçerek kaydetmesini sağlayan bir pencere açılmaktadır.

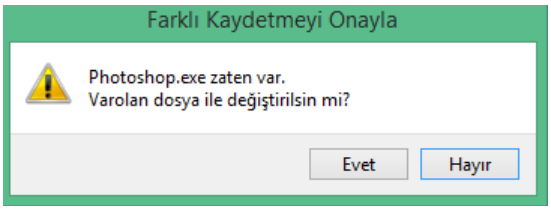
SaveFileDialog kontrolü sayesinde kullanıcının herhangi bir dosyayı seçerek kaydetmesini sağlayan bir pencere açılmaktadır. Dosya açma penceresine benzer görünümde olan kaydet penceresi Şekil 11.16.'da gösterilmektedir.



Şekil 11.16. Savefiledialog Kontrolü Farklı Kaydet Penceresi

ShowDilaog metodu kullanılarak açılan *Farklı Kaydet* penceresi ile var olan bir dosyanın açılması ve var olan bir dosyanın üzerine yeni bir dosya yazılması veya yeni bir dosya oluşturulması sağlanabilir. Aynı zamanda pencere içinde bulunan *Yeni Klasör* butonuna tıklayarak ilgili kök klasörünün içinde yeni klasörler oluşturulabilir.

Kullanıcının farklı kaydet penceresini açtıktan sonra var olan bir dosya adını belirterek kaydet butonuna tıkladığı zaman bir uyarı ile karşılaşması sağlanabilir. Bu işlem için özellikler panelinde bulunan *OverwritePrompt* özelliğini *True* olarak ayarlaması gerekmektedir. Böyle hazırlanmış bir uygulamanın vereceği uyarı penceresi Şekil 11.17.'de gösterilmektedir.



Şekil 11.17. Savefiledialog Kontrolünün Uyarı Mesajı

Kullanıcı farklı kaydet penceresi ile çalıştıktan sonra tekrar aynı pencereyi açarak en son kalmış olduğu klasörden devam etmek isteyebilir. Kullanıcıya böyle bir imkan sunmak için özellikler panelinde bulunan *RestoreDirectory* seçeneği *True* olarak ayarlanmalıdır.

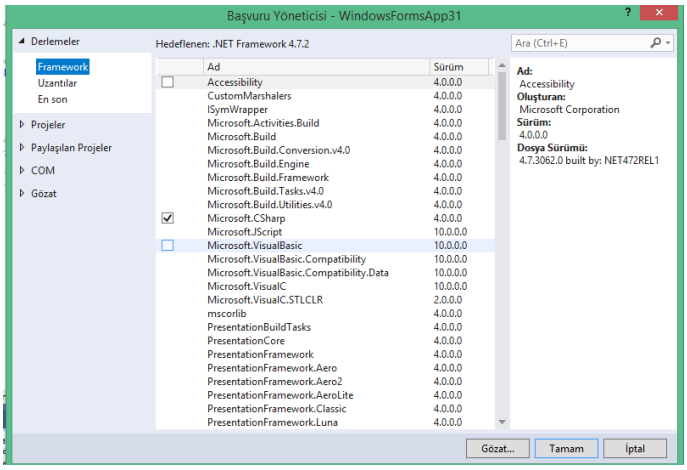
InputBox'ı projelerimizde kullanabilmek için öncelikle referans olarak Microsoft.VisualBasic sınıfını eklememiz gerekmektedir.

Diyalog pencerelerinin tamamı birbirine benzer özellikler taşımaktadır. Diğer pencerelerde kullanılan Title, FileName, Filter gibi özellikler SaveFileDialog penceresinde de aynı yöntemlerle kullanılabilmektedir. Örneğin Title özelliği kullanılarak açılacak olan SaveFileDialog penceresinin başlık bilgisi istenilen bir değer ile düzenlenebilmektedir.

INPUTBOX

Kullanıcı tarafından bilgi girişini sağlayabileceğimiz bir diğer pencere de *InputBox* penceresidir. Ancak C# içinde InputBox penceresinin bulunmadığı

belirtilmişti. Bu pencereyi projelerimizde kullanabilmek için öncelikle referans olarak *Microsoft.VisualBasic* sınıfını eklememiz gerekmektedir. Projemize ekleyebileceğimiz referansların olduğu pencereye *Proje* menüsü altında bulunan *Başvuru Ekle…* seçeneği ile ulaşabiliriz. Bu yol takip edildikten sonra Şekil 11.18.'de gösterilen *Başvuru Yöneticisi* penceresi ekrana gelecektir.



Şekil 11.18. Başvuru Yöneticisi Penceresi

Başvuru yöneticisi penceresinin sol tarafında bulunan Derlemeler ve Framework hiyerarşisi takip edilip, açılan listeden Microsoft.VisualBasic referansı işaretli konuma getirilmelidir. *Tamam* butonuna tıklayarak eklemiş olduğumuz referansları *Çözüm Gezgini* panelinde bulunan *Başvurular* başlığı altında görebiliriz.

Microsoft.VisualBasic referansını ekledikten sonra InputBox penceresi ile işlem yapabilmek için kod editöründe de projemize eklediğimizi belirtmemiz gerekir. Bu işlem için *Using* anahtar kelimesi kullanılmaktadır. Böylelikle istediğimiz sınıfı projemize eklemiş oluruz. Kod editörünün üst kısmına aşağıdaki kod satırını yazmamız yeterli olacaktır.

- Using Microsoft.VisualBasic;

Bu işlemler tamamlandıktan sonra InputBox C# projelerinde kullanılabilmektedir. Şimdi InputBox penceresinin kullanımını ve alacağı parametreleri inceleyelim.

InputBox Kullanımı ve Parametreleri

Veri girişini sağlamak amacıyla kullanılan bu pencere genellikle aldığı değeri bir değişken üzerinde saklamaktadır. Kullanım yöntemi beş farklı parametre alacak şekildedir. Bu parametrelerden ilki zorunlu olup diğerlerinin kullanımı tercihe bağlıdır. Aşağıda kullanım şekli ve parametreleri gösterilmektedir.

- Interaction.InputBox("MesajMetni","PencereBasligi","VarsayilanDeger", XKoordinati, YKoordinati);

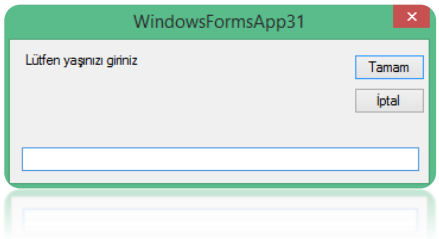
InputBox kullanımının daha iyi anlaşılması için bir örnekle anlatmak daha faydalı olacaktır. Ancak öncelikle parametreleri inceleyelim.

Mesaj metni

Mesaj metni: String veri türünde değer alabilen ve InputBox penceresinin zorunlu olan parametresidir.

String veri türünde değer alabilen ve InputBox penceresinin zorunlu olan parametresidir. Kullanıcının ilgili alana gireceği bilgi hakkında yönlendirme işlemini gerçekleştirmek amacıyla kullanılmaktadır. Eğer bu alana gireceğiniz değer bir değişkenin içinden alınmıyorsa, yazacağınız metni *Çift Tırnak* (*“”*) içine alarak yazmanız gerekmektedir. Bununla beraber ilk parametrenin değerini, farklı türde bir değişkenden alabilmek için dönüştürme (convert) işlemi uygulanmalıdır. Mesaj metnine, “Lütfen yaşınızı giriniz:” yazılmış bir InputBox penceresine ait iki farklı yöntemle hazırlanmış kodlar aşağıda maddeler halinde listelenmiştir. Bu kodlardan herhangi biri kullanılarak oluşturulan InputBox penceresine ait ekran görüntüsü de Şekil 11.19.’da gösterilmektedir.

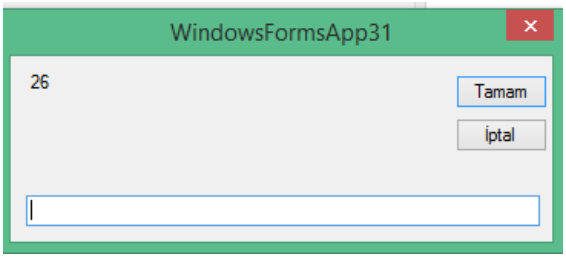
- `Interaction.InputBox("Lütfen yaşınızı giriniz:");`
-
```
string metin;
        metin = "Lütfen yaşınızı giriniz:";
        Interaction.InputBox(metin);
```



Şekil 11.19. Inputbox

Mesaj metnine 26 sayısal değerini taşıyan değişkenin değerini aktarmak için yazılması gereken kod aşağıda listelenmiştir. Bu koda ait InputBox penceresinin ekran görüntüsü Şekil 11.20.’deki gibi olacaktır.

-
```
int a;
    a = 26;
    Interaction.InputBox(Convert.ToString(a));
```

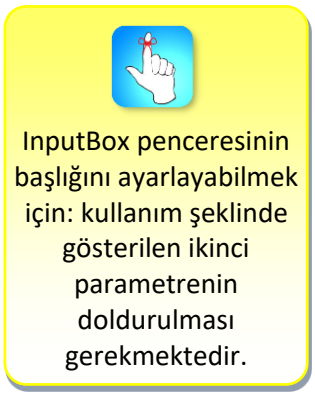


Şekil 11.20. Inputbox Mesaj Metnine Değişken Değeri Yazdırma

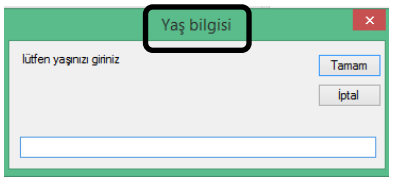
Pencere başlığı

Windows form uygulamalarında çalışılan pencerenin amacını veya özelliğini belirtmek için başlık belirleme işlemi sıklıkla kullanılmaktadır. InputBox penceresinin başlığını ayarlayabilmek için de, kullanım şeklinde gösterilen ikinci parametrenin doldurulması gerekmektedir. Bu parametre de ilk parametrede olduğu gibi *String* türünde değer almaktadır. Pencere başlığı “*Yaş Bilgisi*” olarak düzenlenmiş bir InputBox penceresinin kodu aşağıda verilmiştir. Bu kod yazılarak

çalıştırılan InputBox penceresine ait ekran görüntüsü de kod Şekil 11.21.'de gösterilmektedir.



- Interaction.InputBox("Lütfen yaşınızı giriniz:",”Yaş Bilgisi”);

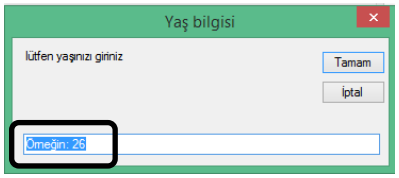


Şekil 11.21. Inputbox Pencere Başlığı

Varsayılan değer

Genellikle doldurulacak alan hakkında kullanıcıya örnek sunmak için kullanılmaktadır. İlk iki parametrede olduğu gibi *String* türünde değer almalıdır. Bu alan doldurularak çalıştırılan uygulamada InputBox penceresi bu bilgiyi kullanıcıya seçilmiş halde göstermektedir. Bunun sebebi girilen değerin hemen üzerine yazılmasını sağlamak içindir. Varsayılan değer olarak, "*Örneğin: 26*" yazılmış bir InputBox penceresinin kodu aşağıda verilmiştir. Bu kod yazılarak çalıştırılan InputBox penceresine ait ekran görüntüsü de kod Şekil 11.22.'de gösterilmektedir.

- Interaction.InputBox("Lütfen yaşınızı giriniz:",”Yaş Bilgisi”,”Örneğin:26”);

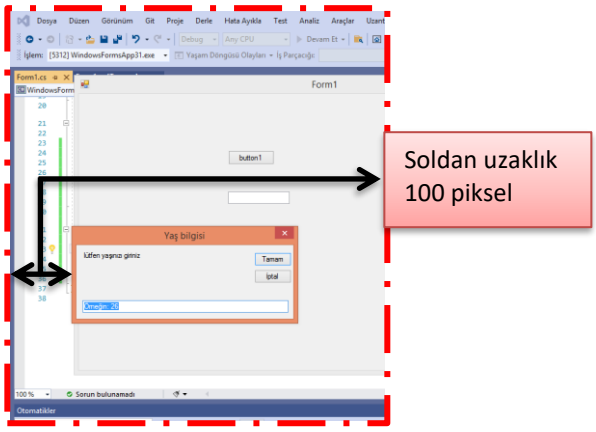


Şekil 11.22. Inputbox Varsayılan Değer

X koordinatörü

InputBox penceresinin ekran üzerindeki konumlanacağı alan belirlenebilir. Bunun için dördüncü parametre, InputBox penceresinin ekranın soluna göre konumlanacağı alanı ayarlamamızı sağlamaktadır. Buraya girilecek olan değer sayısal olmakla beraber mesafe piksel cinsinden ayarlanmaktadır. Ekranın soluna göre 100 piksel uzaklıkta hazırlanmış bir InputBox penceresinin kodu aşağıda verilmiştir. Bu kod yazılarak çalıştırılan InputBox penceresine ait ekran görüntüsü de kod Şekil 11.23.'de gösterilmektedir.

- Interaction.InputBox("Lütfen yaşınızı giriniz:" , ”Yaş Bilgisi” , ”Örneğin:26” , 100);



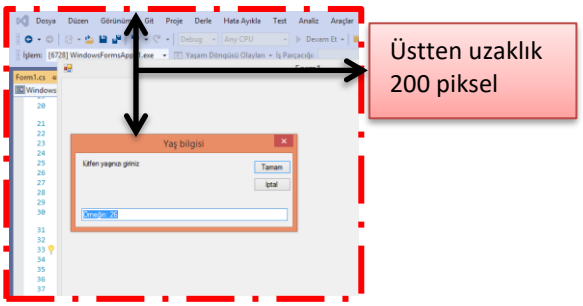
Şekil 11.23. Inputbox Soldan Uzaklık 100 Piksel

Y koordinatörü

InputBox penceresi açılırken ekran üzerindeki konumlanacağı alan belirlenebilir.

InputBox penceresinin ekranın soluna göre konum bilgisi belirlenebildiği gibi ekranın üst kısmına göre hangi konumda açılacağı da belirlenebilmektedir. Bu işlem için InputBox kullanım şeklinde gösterilen beşinci yani son parametre kullanılmalıdır. Dördüncü parametrenin kullanım şekli ile aynıdır. Bu iki parametre boş bırakıldığı takdirde pencere varsayılan bir konum üzerinde açılacaktır. Ekranın üst kısmına göre 200 piksel uzaklıkta görüntülenmesi sağlanan bir InputBox penceresinin kodu aşağıda verilmiştir. Bu kod yazılarak çalıştırılan InputBox penceresine ait ekran görüntüsü de kod Şekil 11.24.'de gösterilmektedir.

- Interaction.InputBox("Lütfen yaşınızı giriniz:" , "Yaş Bilgisi" , "Örneğin:26" , 100 , 200);



Şekil 11.24. Inputbox Üstten Uzaklık 200 Piksel

Bireysel Etkinlik

- InputBox penceresini bütün parametreleriyle beraber kullanarak öğrencinin vizeden aldığı notu öğrenebileceğiniz şekilde hazırlayın. Not girişi yapıldıktan sonra girilen notu MessageBox yardımıyla kullanıcıya gösterin.

Özet

- Diyalog pencereleri kullanıcının etkileşime girmesini sağlayan ve açılan pencereyi yanıtlamasını bekleyen yapıdaki grafiksel denetim öğeleridir. Windows form uygulamalarına ekleyebilmek için araç kutusunda bulunan İletişim Kutuları sekmesi kullanılabilir. Bu alanda yer alan bütün diyalog pencereleri ShowDialog metodu kullanılarak görüntülenmektedir. Aynı zamanda kullanıcıdan gelen yanıt yani basılan tuş DialogResult özelliği kullanılarak alınmaktadır.
- Color Dialog kontrolü kullanıcının renk seçme işlemini gerçekleştirmesi için kullanılan pencerenin açılmasını sağlamaktadır. Renk başlığı adı altında açılan pencere üzerinden temel renklerden biri seçilebileceği gibi Özel Renk Tanımla seçeneği de kullanılabilmektedir. Özel renk tanımla seçeneği ile açılan alan üzerinde fare imleci ile herhangi renk seçimi yapılabilmektedir. Ek olarak seçilecek olan renge ait Kırmızı, Yeşil ve Mavi kodları manuel olarak da girilebilmektedir. Eğer belirlen bu renk sıklıkla kullanılıyorsa Özel Renklere Ekle butonu kullanılarak Özel Renkler arasına eklenebilir.
- Bilgisayarda bulunan bir klasörü seçmek için FolderBrowserDialog kontrolünü kullanmak gerekmektedir. Bu kontrol sayesinde bilgisayarda seçilen bir klasörün yolunu özellikler panelinde bulunan SelectedPath seçeneğinde tutabiliriz. FolderBrowserDialog uygulamaya eklendikten sonra SelectedPath özelliğine tıklayarak Klasöre Gözat penceresi açılmaktadır.
- Araç kutusu üzerinde bulunan FontDialog kontrolü sayesinde uygulamamızdaki metinlerin yazı tipi özelliklerini belirleyebilir ve değiştirebiliriz. Kontrolü formumuza ekledikten sonra özellikler panelinde bulunan Font seçeneğine tıklayarak Yazı Tipi penceresini açabiliriz.
- OpenFileDialog kontrolü, kullanıcının bilgisayarda kayıtlı olan dosyaları açması için kullanılan pencerenin ekranda görüntülenmesini sağlamaktadır. Çalışma anında pencere başlığı varsayılan değer olarak “Aç” adıyla karşımıza gelmektedir. Pencere başlığını değiştirmek için özellikler panelinde bulunan Text alanını düzenlememiz gerekir. Aynı zamanda bu pencerede bulunan Dosya adı bölümüne atanmak istenen değer, özellikler panelindeki FileName seçeneği ile ayarlanabilmektedir.
- SaveFileDialog kontrolü sayesinde kullanıcının herhangi bir dosyayı seçerek kaydetmesini sağlayan bir pencere açılmaktadır. ShowDilaog metodu kullanılarak açılan Farklı Kaydet penceresi ile var olan bir dosyanın açılması ve var olan bir dosyanın üzerine yeni bir dosya yazılması veya yeni bir dosya oluşturulması sağlanabilir. Aynı zamanda pencere içinde bulunan Yeni Klasör butonuna tıklayarak ilgili kök klasörünün içinde yeni klasörler oluşturulabilir.
- Uygulamalar çalışırken üretilen sonuçların ve bilgilerin kullanıcıya ve programcıya gösterilmesi amacıyla MessageBox penceresi kullanılmaktadır. Bu pencere yardımıyla kullanıcıya gösterilecek olan metnin uzunluğu en fazla 1024 karakter olabilmektedir. Bu pencerede zorunlu olan ilk parametre yani mesaj bilgisi String veri türünde olmalıdır. Kullanıcıya bilgi vermek için program akışı içerisinde MessageBox sınıfına ait Show metodunu kullanmamız gerekmektedir.
- Kullanıcı tarafından bilgi girişini sağlayabileceğimiz bir diğer pencere de InputBox penceresidir. Ancak C# içinde InputBox penceresinin bulunmadığı belirtilmişti. Bu pencereyi projelerimizde kullanabilmek için öncelikle referans olarak Microsoft.VisualBasic sınıfını eklememiz gerekmektedir. Projemize ekleyebileceğimiz referansların olduğu pencereye Proje menüsü altında bulunan Başvuru Ekle… seçeneği ile ulaşabiliriz.

Özet (devamı)

- Veri girişini sağlamak amacıyla kullanılan Inputbox penceresi genellikle aldığı değeri bir değişken üzerinde saklamaktadır. Kullanım yöntemi beş farklı parametre alacak şekildedir. Bu parametrelerden ilki zorunlu olup diğerlerinin kullanımı tercihe bağlıdır.
- Mesaj metni: String veri türünde değer alabilen ve InputBox penceresinin zorunlu olan parametresidir. Kullanıcının ilgili alana gireceği bilgi hakkında yönlendirme işlemini gerçekleştirmek amacıyla kullanılmaktadır. Eğer bu alana gireceğiniz değer bir değişkenin içinden alınmıyorsa, yazacağınız metni Çift Tırnak ("") içine alarak yazmanız gerekmektedir. Bununla beraber ilk parametrenin değerini, farklı türde bir değişkenden alabilmek için dönüştürme (convert) işlemi uygulanmalıdır.
- Pencere başlığı: Windows form uygulamalarında çalışılan pencerenin amacını veya özelliğini belirtmek için başlık belirleme işlemi sıklıkla kullanılmaktadır. InputBox penceresinin başlığını ayarlayabilmek için de, kullanım şeklinde gösterilen ikinci parametrenin doldurulması gerekmektedir. Bu parametre de ilk parametrede olduğu gibi String türünde değer almaktadır.
- Varsayılan değer: Genellikle doldurulacak alan hakkında kullanıcıya örnek sunmak için kullanılmaktadır. İlk iki parametrede olduğu gibi String türünde değer almalıdır. Bu alan doldurularak çalıştırılan uygulamada InputBox penceresi bu bilgiyi kullanıcıya seçilmiş halde göstermektedir. Bunun sebebi girilen değerin hemen üzerine yazılmasını sağlamak içindir.
- X koordinatörü: InputBox penceresinin ekran üzerindeki konumlanacağı alan belirlenebilir. Bunun için dördüncü parametre, InputBox penceresinin ekranın soluna göre konumlanacağı alanı ayarlamamızı sağlamaktadır. Buraya girilecek olan değer sayısal olmakla beraber mesafe piksel cinsinden ayarlanmaktadır.
- Y koordinatörü: InputBox penceresinin ekranın soluna göre konum bilgisi belirlenebildiği gibi ekranın üst kısmına göre hangi konumda açılacağı da belirlenebilmektedir. Bu işlem için InputBox kullanım şeklinde gösterilen beşinci yani son parametre kullanılmalıdır. Dördüncü parametrenin kullanım şekli ile aynıdır. Bu iki parametre boş bırakıldığı takdirde pencere varsayılan bir konum üzerinde açılacaktır.

DEĞERLENDİRME SORULARI

1. MessageBox yardımıyla kullanıcıya gösterilecek olan metnin uzunluğu en fazla karakter olabilmektedir?
   a) 256
   b) 512
   c) 1024
   d) 2048
   e) 4096

2. Aşağıdakilerden hangisi MessageBox penceresinde Durdur, Yeniden Dene ve Yoksay butonlarının olmasını sağlar?
   a) MessageBoxButtons.OKCancel
   b) MessageBoxButtons.OK
   c) MessageBoxButtons.AbortRetryIgnore
   d) MessageBoxButtons.RetryCancel
   e) MessageBoxButtons.YesNo

3. Aşağıdakilerden hangisi MessageBox penceresinde *uyarı* yapıldığını ifade emek için sarı bir arkaplana sahip üçgen içinde ünlem işaretinin bulunduğu ikonun olmasını sağlar?
   a) MessageBocIcon.Error
   b) MessageBocIcon.Hand
   c) MessageBocIcon.Stop
   d) MessageBocIcon.Exclamation
   e) MessageBocIcon.Question

4. Aşağıdaki biçim tanımlayıcılarından hangisi Messagebox penceresinde yeni satır tanımlamamızı sağlamaktadır?
   a) \n
   b) \f
   c) \r
   d) \t
   e) \k

5. Diyalog pencerelerinde kullanıcının bastığı tuşu öğrenmek için hangi özelliği kullanmamız gerekmektedir?
   a) Show
   b) DialogResult
   c) ShowDialog
   d) NewLine
   e) Button

6. Aşağıdaki diyalog pencerelerinden hangisi kullanıcının renk seçmesi için bir pencere görüntülemesini sağlamaktadır?
   a) FontDialog
   b) FolderBrowserDialog
   c) OpenFileDialog
   d) SaveFileDialog
   e) ColorDialog

7. FolderBrowserDialog yardımıyla açılan pencerede bilgisayarda seçilen bir klasörün yolu özellikler panelinde bulunan hangi özellik içinde saklanmaktadır?
   a) Text
   b) Name
   c) Check
   d) SelectedPath
   e) Show

8. Aşağıdaki diyalog pencerelerinden hangisi kullanıcının Yazı tipini belirlemesi için bir pencere görüntülemesini sağlamaktadır?
   a) FontDialog
   b) ColorDialog
   c) FolderBrowserDialog
   d) OpenFileDialog
   e) SaveFileDialog

9. ShowDialog metodu kullanılarak ekrana getirilen dosya açma penceresinde listelenecek dosya türlerini belirlemek için hangi özelliği kullanabiliriz?
   a) Name
   b) Filter
   c) Check
   d) Size
   e) Location

10. Aşağıdakilerden hangisi projelerimize referans eklemek için kullanacağımız menü yoludur?
    a) Görünüm – Aç
    b) Dosya – Aç
    c) Dosya – Yeni
    d) Proje – Sınıf ekle
    e) Proje – Başvuru ekle

Cevap Anahtarı

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

YARARLANILAN KAYNAKLAR

Aktaş, V. (2010). *Visual studio 2010 ile her yönüyle C# 4.0* (2. Baskı). İstanbul: Kodlab Yayın.

Aktaş, V. (2020). *Visual studio 2019* (1.Baskı). İstanbul: 01 Yayınları.

Atasever, V. (2017). *C# 7 Uygulamalarla C# programlama dilini keşfedin* (2. Baskı). Kocaeli: Level Kitap.

Demirli, N. & İnan, Y. (2005). *Visual C# .Net* (3. Baskı). Ankara: Palme Yayıncılık.

Sharp, J. (2011). *Adım adım Microsoft visual C# 2010*. (Çev. T. Buldu). Ankara: Arkadaş Yayınevi.

Uzunköprü, S. (2017). *Projeler İle C# 7.0 ve SQL Server 2016* (7. Baskı). İstanbul: Kodlab Yayın.

Yanık, M. (2008). *Visual studio 2008 ile Microsoft visual C# 3.0 for .Net framework 3.5* (1. Baskı). Ankara: Seçkin Yayıncılık.

Yücedağ, M. (2020). *C# Eğitim kitabı* (3. Baskı). İstanbul: Dikeyeksen.