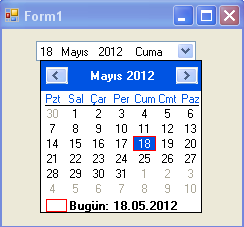
Nesneler Ve Özellikleri

DataTimePicker  
 [DateTimePicker nesnesi](http://www.gorselyazilim.net/csharp-datetimepicker-kullanimi.html) programlarımızda [tarih işlemleri](http://www.gorselyazilim.net/csharp-datetimepicker-kullanimi.html) yapabilmemizi sağlar. Yani uygulamamızda kullanıcıdan bilgi alınırken DateTimePicker nesnesi yardımı ile [tarih](http://www.gorselyazilim.net/csharp-datetimepicker-kullanimi.html) bilgisini de alabiliriz.  
  
**>>>** DateTimePicker nesnesinin görünümü aşağıdaki gibidir.



DateTimePicker- 1

**Name:**Bu özellik kısmına nesnemizin ismini verebiliriz. Dikkat etmemiz gereken buraya gireceğimiz metin nesnemize kod içerisinde ulaşmamızı sağlayacak isimdir. Yani kod içerisinde kullanırken name özelliğindeki ismi kullanırız.

**CalendarFont:**DateTimePicker takvim nesnemizin yazı tipi özelliklerinin ayarlandığı kısım.

**CalendarFontColor:**DateTimePicker takvim nesnemizin ön zemin renginin ayarlandığı kısım.

**CalendarMonthBackground:** DateTimePicker takvim nesnemizin arka plan rengi.

**CalendarTitleBackColor:**DateTimePicker başlığının arka plan renginin ayarlandığı kısım.

**CalendarTitleForeColor:**Başlıktaki yazının yazı renginin ayarlandığı kısım.

**Enabled:**Nesnemizin aktif olup olmayacağının ayarlandığı kısımdır.

**Font:** Bu özellik ise nesnemizin textinin yazı tipi özelliklerinin ayarlandığı kısımdır. Yukarıdaki CalendarFont açılır kutuya tıklandıktan sonra açılan takvimin yazı tipi özelliklerinin ayarlandığı kısım font ise nesnenin textinin yazı tipi özelliklerinin ayarlandığı kısım.

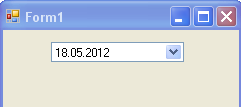
**Format:**Bu özellik ise datetimepicker nesnesindeki tarihin görünüş şeklinin ayarlanabileceği kısımdır.

>>>Format Özelliğini Long Yaparsak (Uzun Tarih)



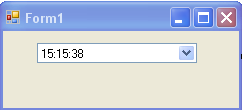
DateTimePicker- 2

**>>>**Format Özelliğini Short Yaparsak (Kısa Tarih)



DateTimePicker- 3

**>>>**Format Özelliğini Time Yaparsak (Sadece Zaman)



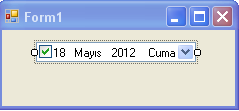
DateTimePicker- 4

>>>Custom seçilir ise Properties penceresinde bulunan CustomFormat kısmında belirttiğimiz değere bağlı olarak değişir.

**MaxDate:**Bu özellik datatimepicker nesnesinde maksimum en fazla gözükecek tarih ve saati belirlediğimiz kısımdır.

**MinDate:**Bu özellik ise minimum en az gözükecek tarih ve saati belirlediğimiz kısımdır.

**ShowCheckBox:**Bu özellik eğer true yapılırsa DateTimePicker nesnemizin başında Check kutusu olur. Bu özelliğin seçili olup olmadığını ise Checked özelliği ile görebiliriz.



DateTimePicker- 5

**UYGULAMA**

Programımıza 2 adet DateTimePicker, 6 adet Label, 1 Adet de Button ekleyelim. Programımızın yapacağı işlem DateTimePicker1 de seçilen tarih ile DateTimePicker2 de seçilen tarih arasındaki gün farkını bulacak ve Label’ e yazacak.

**

DateTimePicker- 6

**>>>**DateTimePicker nesnelerimizin **ShowCheckBox** özelliklerini True yapalım.

**>>>**Daha sonra “Hesapla” butonunun **Click**olayına aşağıdaki kodları yazıyoruz.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23 | **private** **void** button1\_Click(**object** sender, EventArgs e)  {  *//Eğer Check kutularının her ikiside işaretli ise işlemi yapar ve if bloğu içerisindeki kodlar çalışır.*  **if** ((dateTimePicker1.[Checked](http://www.google.com/search?q=checked+msdn.microsoft.com) == **true**) && (dateTimePicker2.[Checked](http://www.google.com/search?q=checked+msdn.microsoft.com) == **true**))  {  *//datetimepicker1'de seçilen tarihi ilk değişkenine aktardık.*  DateTime ilk = dateTimePicker1.Value;  *//datetimepicker2'de seçilen tarihi son değişkenine aktardık.*  DateTime son = dateTimePicker2.Value;  *//son değişkeninden ilk değişkeni çıkardık ve sonucu fark değişkenine aktardıkç*  TimeSpan fark = son - ilk;  *//ve son olarak farkı label4 nesnesinin textine yazdırdık.*  label4.Text = "Aradaki Gün Farkı = " + fark.Days.ToString();  }  **else**  {  *//Check kutularından herhangibiri işaretli değilse hata mesajı verir.*  MessageBox.Show("Lütfen Check kutularını işaretleyin..");  }  } |

**>>>**Daha sonra DateTimePicker1 nesnesinin üzerine çift tıklayalım ve ValueChanged olayına aşağıdaki kodları yazalım.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | **private** **void** dateTimePicker1\_ValueChanged(**object** sender, EventArgs e)  {  *//DateTimePicker1 nesnesindeki Checkedın işaretli olup olmadığını kontrol eder.*  **if** (dateTimePicker1.[Checked](http://www.google.com/search?q=checked+msdn.microsoft.com) == **true**)  {  label5.Text = "Check İşaretli";  }  **else**  {  label5.Text = "Check İşaretli Değil";  }  } |

**>>>** Son olarak ise DateTimePicker2 nesnesinin ValueChanged olayına aşağıdaki kodları yazın.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | **private** **void** dateTimePicker2\_ValueChanged(**object** sender, EventArgs e)  {  *//DateTimePicker2 nesnesindeki Checkedın işaretli olup olmadığını kontrol eder.*  **if** (dateTimePicker2.[Checked](http://www.google.com/search?q=checked+msdn.microsoft.com) == **true**)  {  label6.Text = "Check İşaretli";  }  **else**  {  label6.Text = "Check İşaretli Değil";  }  } |

LinkLabel

İçinde hyper link (web sayfası adresleri) bulunan Label kontrolleri oluşturmak için kullanılır.

**Links:** Label içindeki linkleri belirlemek için bu özellik kullanılabilir.

Link.Add(başlangıç, uzunluk, adres)

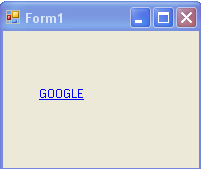
**LinkBehavior :** **LinkLabel**’deki bağlantının görünümünü ayarlamak için kullanılır. (Altı Çizgili, Çizgisiz, Fare İle Üzerine Gelindiği Zaman Çizgi Çıkması)

**LinkColor :** **Link** rengini ayarlamak için kullanılır.

**Text :** **LinkLabel**’de görüntülenecek yazıyı belirtmek için kullanılır.

**UYGULAMA**

Programımıza 1 adet LinkLabel ekliyoruz.



LinkLabel-1

>>>Sonra LinkLabel’in LinkClicked olayına aşağıdaki kodları yazıyoruz.

private void linkLabel1\_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)

{

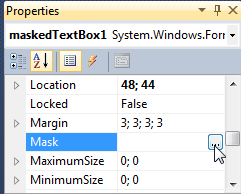
System.Diagnostics.Process.Start("www.google.com");

}

MaskedTextBox

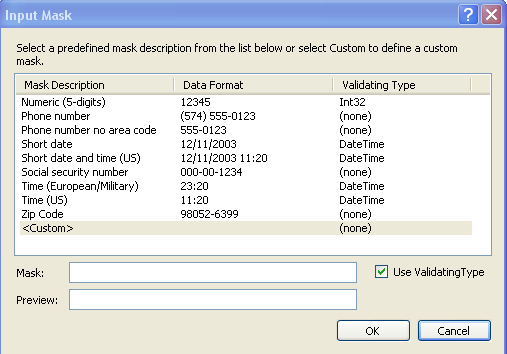
Adından da anlaşılacağı üzere bu kontrolümüz aslında bir TextBox ancak farkı var tabii. Maskeleme yapmayı sağlar. Peki nedir maskeleme, belirli bir formatta veri girişi kuralı tanımlar.

**Mask:** . Properties alanında en önemli özelliği Mask özelliğidir.



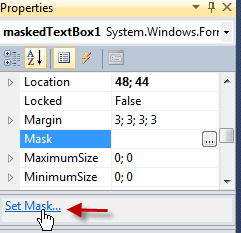
MaskedTextBox-1

Bu özelliği seçtiğimizde (yandaki 3 noktaya tıklayarak) aşağıdaki gibi bir pencere çıkar :



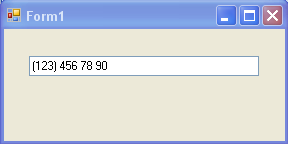
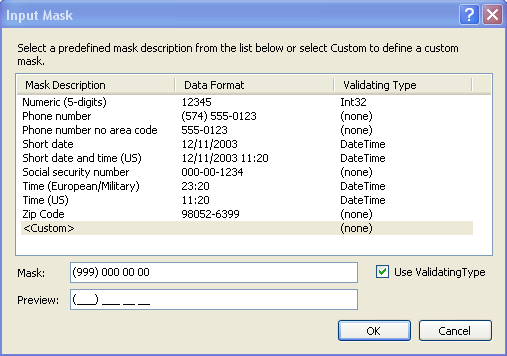
MaskedTextBox-2

Burada görüleceği üzere önceden tanımlanmış belli mask yapıları var. Ayrıca kendimiz de mask yapısı oluşturabiliriz. Bunun için aşağıdaki gibi set mask seçeriz.



MaskedTextBox-3

Örneğin şu şekilde bir telefon numarası girilmesini sağlayalım.(123) 456 78 90



MaskedTextBox-4 MaskedTextBox-5

Burada mask alanında yazdığım (999) girilsede olur girilmesede olur demek. 000 00 00 ise zorunlu giriş demek. Preview alanında bu mask yapısını görebiliyorum.

**BeepOnError:** MaskedTextBox kontrolüne istediğimiz değer haricinde bir veri girilmesine izin verilmeyeceği gibi kullanıcı bir beep sesi ile uyarılır.

ListBox

            Listeleri oluşturabileceğimiz bir kontroldür. Bir grup elmanı listelemek için kullanılır.

**Items**:Liste içinde bulundurulacak elamanlar bu özellik ile belirlenir. Collection tipinde olan bu özellik çok sayıda alt özellik ve metot bulunur. Visual C#’ta diziler sıfırdan başladığı gibi liste içindeki indekslerde sıfırdan başlar.

**Items.Add(Elaman):**Listeye elman eklemek için kullanılan bir metottur. Metottan geriye dönen değer sadece liste içindeki sıra numarasıdır.

**ÖRNEK:**

private void button1\_Click(object sender, EventArgs e)

        {

            listBox1.Items.Add("Ankara");

            listBox1.Items.Add("İstanbul");

            listBox1.Items.Add("İzmir");

        }

**Items.AddRange:** Bir grup elamanı tek seferde listeye eklemek için bu metot kullanılır.

**ÖRNEK:** ListBox1’deki elamanları başka bir liste kutusuna aktaran programı yazınız.

private void button1\_Click(object sender, EventArgs e)

        {

            listBox2.Items.AddRange(listBox1.Items);

        }

**Items.Insert(No,Eleman):** Items.Add  eklenen elemanı listenin sonuna ekler. Araya eleman eklemek için bu metot kullanılır.

**ÖRNEK:** private void button2\_Click(object sender, EventArgs e)

        {

            listBox1.Items.Add("Ankara");

            listBox1.Items.Add("İstanbul");

            listBox1.Items.Add("İzmir");

            listBox1.Items.Insert(1, "Bolu");

        }

**Items.RemoveAt(No):**Numarası verilen elamanı siler.

**ÖRNEK:**

private void button1\_Click(object sender, EventArgs e)

        {

            listBox1.Items.RemoveAt(0); //ilk elamanı siler

        }

**Items.Remove(Eleman):**Verilen elemanı listeden siler. RemoveAt listedeki sıra numarasına göre siler. Remove içeriğe göre siler.

**ÖRNEK:**

listBox1.Items.Remove (“Ankara”);

**Itmes.Clear:**Listedeki bütün elemanları siler.

**ÖRNEK:**

 Listbox1.Items.Clear();

**Items.Count:** Listedaki eleman sayısını verir.

**Items[No]:**Listede numarası verilen elemanın içeriğini öğrenmek veya değiştirmek için kullanılır.

**Items.IndexOf(Eleman):** Listede bir elemanı aramak için kullanılır. Aranan bulunursa sıra numarasını bulunmazsa

-1 değerini verir.