# 10.bölüm

**10**

# Formlar

# (UserFormLAR)

Kullanıcılarla interaktivite yöntemlerinden birinin de Formlar olduğunu VBA ana bölümünün başlarında belirtmiştik. Tabi sadece interaktivite diyerek Formların gücünü hafife almamak gerekir. Formlarla Veritabanı uygulamalarından, Yönetim Paneline kadar birçok uygulama arayüzü oluşturulabilir. Bu bölümde bunları göreceğiz.

## Temeller

Bu bölümde küçük bir Userform yaratıp, üzerine bikaç temel kontrol koyacağız. Form eventlerini ve Form özelliklerini inceleyeceğiz.

### Giriş

#### Formlar

Kullanıcımızla iletişim kurmak istediğimizde bazen mesaj kutuları veya inputboxlar yetersiz kalabilmekte. İşte o zaman UserFormları kullanma zamanı gelmiştir. Tabi, bunların tek amacı iletişim kurmak değil. Daha genel amaçlı işler için de kullanılabilir. Örnek olarak;

* Veri Kayıt Arayüzleri
* Grafikler için ek fonksiyonalite
* Senaryo toolları
* Dosya Erişim arayüzü(Kokpit arayüzü)
* Dashboardlar(Kontrol panelleri)
* v.s

Bu arada her ne kadar öyle olmadığını bilsek de worksheetler de form olarak düşünülebilir, zira bu 'form' üzerine çeşitli form kontrolleri konulabilmektedir. O halde, formları, üzerine kontrol konan ve bu kontrollerle çeşitli etkileşimlerde bulunduğumuz arayüzler olarak düşünebiliriz.

#### Kontroller

Programlama camiasına aşina değilseniz, kontrol kelimesini, bir şeyleri kontrol etmeye yarayan nesneler olarak düşünebilirsiniz. Ancak bunlar, formlar üzerindeki görsel nesnelerden başka birşey değildir. Button(Tıklanır düğme), Listbox(liste kutusu), Combobox(açılır liste kutusu) vb.

Bunları, bir makro çalıştırmak, bir hücre grubundaki değerleri liste kutusuna aktarmak, liste kutusundaki değerleri bir hücre grubuna yazdırmak, bir hücreyi seçmek, bir hücrenin değerini 1'er 1'er artırıp azaltmak gibi  amaçlarla kullanırız.

Kontrollerin detaylarına bir sonraki sayfada değineceğiz. Aşağıda ise bazısını küçük bir örnekte kullanacağız.

#### Form Eventleri(Olayları)

Buraya kadar sırayla okuyarak geldiyseniz, olaylar hakkında bilgi sahibi olmuşsunuz demektir. Eğer bilginiz yoksa, Olaylar bölümünden temel bilgileri aldıktan sonra tekrar buraya gelmenizi tavsiye ederim.

Formların kendisi dahil olmak üzere tüm kontrollerin kendine has eventleri vardır. Düğmeye tıklanması için Click eventi, bir listenin güncellenmeden öncesi ve sonrasını gösteren BeforeUpdate, AfterUpdate eventleri gibi. Bunların detaylarını yine sonraki sayfada ele alacağız.

Dikkat: Eventler konusunda gördüğümüz **Application.EnableEvents=False** atama işleminin Userform ve Kontrol eventleri üzerinde bir etkisi yoktur.

### Basit bir UserForm oluşturma

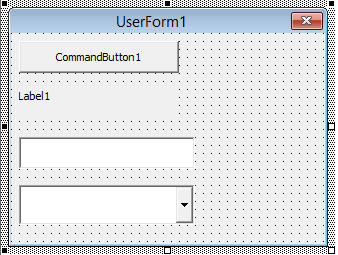
Şimdiye kadar konuları sırayla takip ettiyseniz çoğunlukla Insert Module deyip kodları standart modüller içine yazdığımızı görmüşsünüzdür. Eventleri ele aldığımız sayfalarda ise sheet ve workbook nesneleri içine de kod yazmıştık. Şimdi ise bir başka nesne olan UserFormların içine kod yazmaya geldi sıra.

#### Formu oluşturma aşamaları

Kendinize yeni bir dosya yaratın.

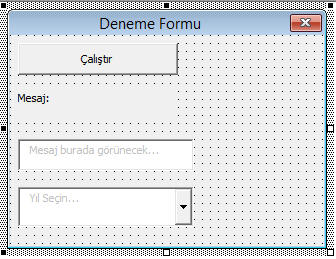
VBE'e geçin ve Project penceresinde Modüllere sağ tıklayıp Insert diyin, sonra da **Userform** seçin.

Karşımıza içi boş bir form gelecektir ve Control ToolBox otomatikman açılacaktır, açılmazsa menüden kendiniz açın. Sonra da toolboxtan aşağıdaki nesneleri formun üzerine sürükleyin ve bırakın: Bi tane commandbutton, bir label, bir textbox, bir de combobox.



Hemen bu noktada isimlendirme standardından bahsedeyim. Properties penceresinden, Form nesnesinin **Name** özelliğini **frm**Deneme olmak üzere diğer nesneleri de sırayla **cmd**Run, **lbl**Mesaj, **txt**Mesaj, **cb**Yıl olarak değiştirin. (Formun özelliklerine erişebilmek için formda boş bir yere tıklamanız yeterlidir.) Böylece kod yazarken nesnelere daha kolay referansta bulunabilirsiniz, özellikle formunuzun üzerinde birçok kontrol olacaksa. Buradaki **standart** şudur: Kontrolün tipinin 2-3 karakterlik bir kısaltması, sonra da anlamlı bir isim.

Ayrıca Form'un, CommandButton'un ve Label'ın **Caption** özellikleriyle TextBox ve ComboBox'ın **Text** özelliklerine de anlamlı bir ifade verelim.



Fark ettiyseniz, birçok form ve web uygulamasında olduğu gibi **grileştirilmiş** metinle kullanıcıya "ipucu/talimat/açıklama içeren mesaj verme" tekniğini kullandım.

#### Formu Çalıştırma

Formunuzun o an itibarıyla canlıda nasıl göründüğünü görmek için **Form seçiliyken** F5 veya yeşil Play tuşuna basabilirsiniz. Bu yöntem developer olan kişinin Form çalıştırma yöntemidir. (Şuan bunu yapabilirsiniz ama bu haliyle bir işe yaramayacaktır. Birazdan formumuz biraz daha işlevsel hale getireceğiz.)

Ama öncesinde formu nihai kullanıcının çalıştıracağı yöntemlere bakalım. Kullanıcılar,

* Bir Worksheet butonuna
* Bir ActiveX butonuna
* Bir Add-in'deki butona
* Ribbona/QAT'ye yerleştirilen bir makro butonuna

bastıklarında Formlar açılırlar. Biz basit ve sık kullanılan bir yöntem olması adına bir Worksheet butonuna tıklandığında aktive edecek kodu yazalım. ActiveX'teki mantık da aynı olacaktır. Sonraki sayfada ActiveX kontrollerin detayını göreceğiz.

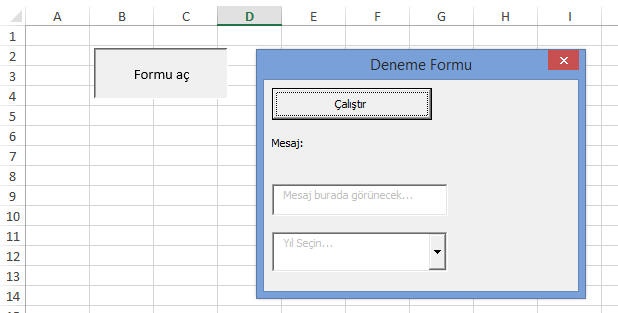
Şimdi sayfamıza bir buton ekleyelim. Ekleyince çıkan dialog kutusunda New'e tıklayalım ve otomatik açılan modüle aşağıdaki kodu yazalım

Sub Button1\_Click()

frmDeneme.Show

End Sub

Evet düğmemizin adını değiştirdikten sonra tıklayalım ve Formumuzu açalım.



##### Modal vs Modeless seçenekleri ile sayfa erişimi

Form bu şekilde açıldığında arkadaki sayfaya erişimimiz engellenmiştir. Sayfayla Form arasında serbest geçiş yapabilmek istiyorsam formu ya **Design aşamasında** properties'ten Modeless**(ShowModal=False)** tanımlarım ya da Button1'e tıkladığımda **Runtime sırasında** Modeless açılmasını sağlarım.

###### Design sırasında



###### Runtime sırasında

Sub Button1\_Click()

frmDeneme.Show vbModeless

End Sub

Design modundayken **Modeless** tanımlanmış bir formu duruma göre modal açmak için ise aşağıdaki kodu yazarız.

Sub Button1\_Click()

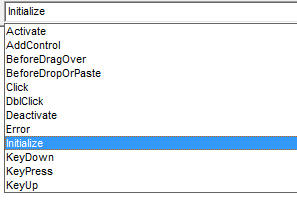
frmDeneme.Show vbModal

End Sub

#### UserForm başlangıç ayarları

Bir form ilk açıldığında çeşitli başlangıç ayarları yapmak iyi bir fikirdir. Bunu formların **Initialize** eventi ile yapıyoruz. Web sitesi tasarlayanlar bilir, bu biraz javascriptin **onload** veya ASP'nin **Page\_Load** veya .Net formlarındaki **Form\_Load** eventlerine benzer.

Bunun için forma sağ tıklayıp **View Code** deyin. Açılan kod sayfasında formu seçince ilk etapta click eventi gelecektir. Siz **Initialize** eventini seçip Click eventini de silin.



Mesela, formumuz açıldığında cbYıl combobox'ını dolduralım. Şimdi aşağıdaki kodu yazalım.(Detay açıklamalara sonraki Kontroller sayfasında gireceğiz)

Private Sub UserForm\_Initialize()

    cbYıllar.List = Array("2017", "2018", "2019")

End Sub

ComboBox'tan seçim yapılacağı sırada da gri olan bilgilendirme yazısı yok olup yılların rengi de siyaha dönsün istiyoruz diyelim.

Private Sub cbYıllar\_DropButtonClick()

    cbYıllar.ForeColor = vbBlack

    cbYıllar.Text = ""

End Sub

Bunun dışında yine çeşitli başlangıç değeri atamaları, kutuların temizlenmesi, varsa statik değişken tanımlamaları v.s bu event içinde yapılabilir.

#### Formları Gizleme ve Kapama

Bazen bir formu geçici olarak gizlemek bazen de tamamen kapatıp başka bir form açmak isteriz. Bunlar için ihtiyacımız olan kodlar şöyle:

Me.Hide 'o an aktif olan formu gizler

Form1.Hide 'Form1i gizler

Unload Me 'aktif formu kapatır

Unload Form1 'Form1'i kapatır

Bir formu kapatmak için sağ üstteki X düğmesine de basılabilir tabiki ama **Unload** fonksiyonu daha çok kapatma işleminin arkasından başka bir iş(ler) yapmak(mesela başka bir formu açmak) istediğimiz zamanlarda kullanılır.

Gizlediğimiz bir formu tekrar aktive etmek için yine **Show** metodu kullanılır, yani bir **Load** fonksiyonu bulunmamaktadır. Özetle elimizdekiler şöyle: İlk kez açma ve yeniden gösterme için **Show** metodu, gizleme için **Hide** metodu, kapatma için **Unload** fonksiyonu.

##### Esc tuşu ile çıkış

ESC tuşuyla çıkış yapmak isterseniz, Form üzerinde bir buton koyun ve bunun **Cancel** özelliğine **True** atayın. Bu sayede Esc tuşuna basıldığında bu düğme odağı almış olur, yani sanki seçilmiş gibi olur, ki bu da Enter eventini tetikler. Şimdi ikinci olarak bu düğmenin Enter eventine formu kapatan kodu yazalım.

Private Sub CommandButton3\_Enter()

Unload Me

End Sub

**NOT:** Modal formlarda, bir form ilk kez açıldığında önce **Initialize** sonra **Activate** eventleri meydana gelir. Sonra bu form gizlenip tekrar açıldığında sadece Activate meydana gelir. Modeless formlarda ise, Initializedan sonra Activate meydana gelmez, iki modeless forma arasında gidip gelince veya gizli olan bir modeless form tekrar gösterildiğinde meydana gelir. Eğer ki yeniden aktive olma durumlarına yaptırmak istediğiniz bir işlem varsa, bu ayrıma dikkat etmelisiniz.

## Form Kontrolleri

### Giriş

Nesne yönelimli programlamanın en somutlaştığı yer olarak ben şahsen kontrolleri görüyorum. Bunlar, gerçek dünya nesnelerine çok benziyorlar. Excel’le çalışırken, bir hücre, bir sayfa veya workbook'un kendisi Excel ile o kadar bütündür ki onları içselleştirmişizdir, bu yüzden onları bir nesne gibi görmek bazen biraz zor olabilir. Ama eminim bu sayfada nesne yönelimli programlama konusunu iyice anlamış olacaksınız.

Zira birçok programlama dilinde ve onlarla geliştirme yaptığımız IDE'lerde olduğu gibi bu nesnelerin özelliklerini Properties penceresinden değiştirebileceğiz ve bu deneyim de bizi programlama dünyasına biraz daha yakın hissettirecektir. Gerçi kontrollerin propertylerine sadece properties penceresinden(DesginTime) değil kodların çalışması sırasında da (Runtime) erişebileceğiz. Ve yine gerçi Nesne Yönelimli olmak demek, sadece özelliklerin Properties'ten değiştirilebilmesi demek değildir, bundan çok daha büyük bir kavramdır ancak yeni başlayanlar için kolaylık sağladığını düşünebiliriz.

Bu sayfada temel olarak baz alacağımız örnek dosyaya **githubdan(useformlar.xlsm)** ulaşabilirsiniz.

### Kontrol Tipleri

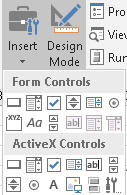
Excel'de 2 tür kontrol bulunmaktadır.

**Form kontrolleri**: Worksheetler üzerine konan ve sınırlı fonksiyonaliteye sahip kontroller.

**ActiveX kontrolleri**: Daha gelişmiş fonksiyonaliteye sahip olan, hem Worksheetler hem de UserFormlar üzerine konan kontrollerdir.

Niye 2 tür kontrol grubu var diye soracak olursanız, önceleri sadece Form kontrolleri vardı, sonra ActiveX kontrolleri geldi diye cevaplanabilir.

Aşağıda Developer menüsünden ikisinin de içeriğini görebilirsiniz. Birbirine çok benzeyen bu kontrollerin temel bazı farkları bulunuyor. Bunlara aşağıda değiniyorum.



#### Worksheet/Form Kontrolleri

Bunlar, Excel arayüzünde **Developer** menüsü altında bulunurlar. Bunların VBA olmadan genel kullanımlarını burada ele almayacağız. VBA'siz de kullanılan bu kontroller oldukça faydalı kontrollerdir ve özellikle dashboard tarzı çalışmaların yaratımında oldukça kullanışlıdırlar.

Bunların VBA'li kullanımında ise ana olay(event) için makro oluşturulur. Mesela sayfa üzerine bir Button(düğme) yerleştirip o düğmenin Click eventinin tetiklenmesiyle(özetle ona tıklayarak) başka bir makroyu çalıştırma amaçlı kullanabiliriz.

Bunların VBA'li kullanımdaki tek avantajları Windows'ta oluşturduğunuz bir dosyanın Apple Mac bir bilgisayardaki Excel'de de çalışacak olmasıdır. Zira Mac işletim sistemi ActiveX kontrolleri desteklemezken bunları destekler.

**NOT**: Gariptir ki Excel’in 5.0 versiyonundan beri kullanılamayan TextBox kontrolü(ve ne olduğunu bilmediğim diğer 2 kontrol) pasif olarak ilgili menüde hala görünmektedir.

##### Makro atama

Bu kontrollere sağ tıklanıp **Assign Macro>New** denince default event için kod ekranı çıkar. Oraya da istediğiniz kodu yazarsınız.

##### Metin değiştirme

Uygun olan kontroller için Sağ tıklanıp **Edit Text** denerek ilgili kontrolün üzerinde görünen metin değiştirilebilir.

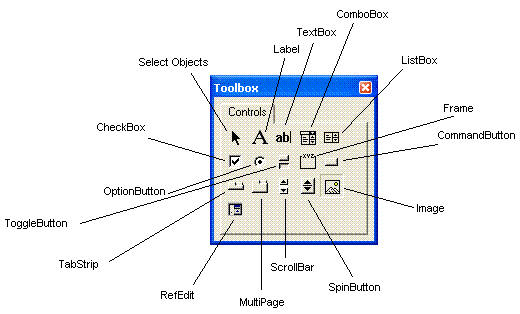
#### ActiveX Kontrolleri

ActiveX kontrolleri hem worksheetlerde hem de VBA UserForm'ları üzerinde kullanılırlar. VBA fonksiyonalitesi olarak worksheet formlarına göre çok daha üstündürler, ancak Excel fonksiyonalitesi olarak ise worksheet form kontrolleri daha kullanışlıdır. O yüzden size tavsiyem bunları **sadece UserFormlar üzerinde kullanın**, diğerlerini de Excel’in bir hücre grubuyla ilişkilendirmek için VBA'siz şekilde kullanın.

Bir düğmeyle bir makro çalıştırmak için de yine worksheet/form kontrollerini kullanabilirsiniz demiştik. Başka neler yapabilirsiniz. Listbox/Combobox'tan seçilen değere göre, seçim yapılır yapılmaz o seçime ait bir veritabanı sorgulaması yapılabilir. Mesela ürün kodlarının olduğu bir Listbox'ta, seçilen ürüne ait özellikler boş sayfaya yazdırılabilir, veya ikinci bir Listbox'ın içeriği doldurulabilir, mesela alt kategorideki ürünlerle.

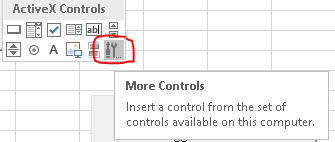
Yukarıda belirttiğim gibi ActiveX kontrollerinin en büyük dezavantajı Mac kullanan bir bilgisayara Windows'ta hazırlanmış bir dosya göndermek olacaktır. Ancak amacımız, ilgili kontrollerin ana eventi dışında bir eventi kullanmaksa o zaman başka çareniz yoktur, mecburen ActiveX kontrolü kullanacaksınız. Mesela, CommandButtonun sadece click eventini kullanacaksanız Worksheet Form kontrolü iş görür, keza Listbox'ın change eventi yeterliyse yine Worksheet Form kontrolü iş görür, ama MouseUp eventini kullanacaksanız ActiveX kullanmak zorundasınız.

Aşağıda, toolboxta default olarak bulunan tüm kontrollerin listesini görebilirsiniz.

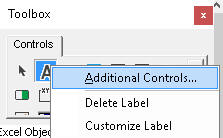


Bunların önemli olanlarının detay özelliklerine aşağıda yer vereceğim, diğerlerini sizin keşfetmeniz gerekiyor.

Bilgisayarlarımızda, Excel ve diğer Microsoft programlarınca kullanılan başka ActiveX kontrolleri de vardır. Bunları, ActiveX kontrollerinin olduğu blokta, sağ alttaki(aşağıda kırmızlı işaretli) buton ile görebilirsiniz ama bunların çoğu worksheetlerde kullanılamaz.  Zaten eklemeye çalışsanız bile bir uyarı çıkacaktır. Hangilerinin kullanılabileceğine dair bir liste var mı, açıkçası bilmiyorum. İlgisini çekenler kurcalayabilir.



Bununla beraber bunların hepsi **Userformlar** üzerinde kullanılabilirler. Bunun için herhangi bir kontrolün üzerine sağ tıklayıp Additional Controls'e tıklamak yeterlidir.



##### Worksheet'te bir kontrole makro atama

Developer'dan Design Mod yapılıp sağ tıklanır. **View Code** denir. İlk başta default(temel) event gelir, istenen event seçilerek kod yazılır.

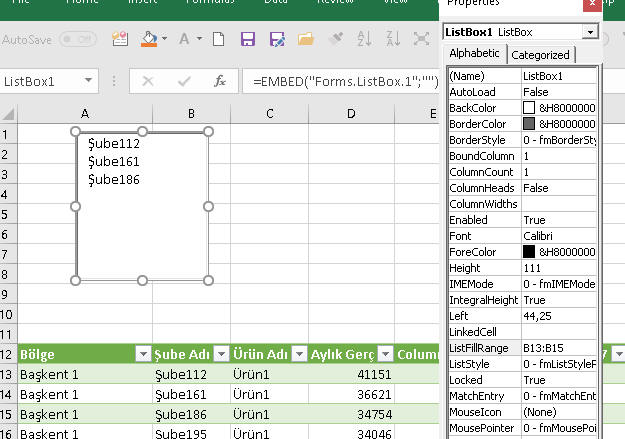
##### Worksheet'te metin değiştirme

Developer'dan Design Mod yapılıp sağ tıklanır. Properties'ten **Caption** veya **Text** özelliği değiştirilir. Veya yine objeye sağ tıklanıp **XXXObject>Edit** denilerek doğrudan metin editlenir.

#### Karşılaştırma

* Excel hücreleriyle etkileşim, Form kontrolleriyle kolayca sağlanır, VBA'siz kullanılır.
* Temel event(button için Click, Listbox için Change) kullanıp Mac bilgisayara gönderme ihtimalimiz varsa: Form kontrol
* Temel event dışındaki eventler için ActiveX kontrolleri kullanılır
* VBA Userformlar üzerinde mecburen ActiveX kontrolleri kullanılır

Sayfanızda 2 tür kontrol de var diyelim. Hangisinin ne tür olduğunu nasıl anlarsınız? Form kontrollerine sağ tıklayabilirken, ActiveX'lere sağ tıklanamaz, bunlara sağ tıklamak için Design Mod'da olmalısınız. Diyelim ki o sırada Design Moddasınız, bu durumda nasıl anlaşılır? Sağ tıklayınca formül çubuğunda EMBED(...) diye bir formül çıkıyorsa ActiveX'tir, çıkmıyorsa Form kontrolüdür. Aynı zamanda ActiveX'e sağ tıklayınca **Properties** ve **View Code** çıkarken diğerinde bunun yerine **Assign Macro** çıkar.



#### Kontrollerin sayfa davranışını yönetmek

Gerek form kontrollerinin gerek ActiveX kontrollerinin sayfa üzerindeki konumu, görünürlüğü, aktif/pasifliği gibi özelliklerini yönetmek için Shape ve OleObject kavramlarını incelemek gerekiyor. Bu bilgiler, kavramsal olarak buraya uygun olmayıp, onları **Shapes ve OleObjects** konusunda inceleyeceğiz.

### Temel Kontroller

#### Command Button

Kontroller arasında en sık kullanılanı ve en aşina olunanı CommandButton'dur.

CommandButon'un default event'i **Click** olmakla birlikte başka eventleri de vardır. Her zamanki yaklaşımımla ben bununla ilgili diğer eventleri şimdiye kadar kullanmadığım için burda da örneklerini vermeyeceğim. Arzu eden ve ihtiyaç duyan araştırabilir.

Click event'i ile bir başka makro çalıştırılabileceği gibi, ekrana bir FileDialog penceresinin gelmesi de sağlanabilir. FileDialog detayına **Dört Temel Nesne>Application** konusunda bakabilirsiniz. Diğer button kullanım amaçları şöyle sıralanabilir:

* MsgBox ile bilgi gösterme
* InputBox ile kullanıcıdan bilgi girmesi/alan seçmesi isteme
* Hücreden bir bilgi okuma
* Hücreye bir bilgi yazdırma
* Veritabanına bilgi yazdırma
* Veritabanından bilgi okuma
* Spin Butonun değerini artırıp/azaltma
* Çeşitli değerleri/nesnelerin içeriklerini resetleme
* v.s

#### TextBox ve Label

##### Label

Label en basit kontroldür. Üzerine genelde ya bir açıklama ya da bir işin sonucunda sonuç mesajı yazdırırız.

##### TextBox

Kullanıcıdan bir şeyler girmesini beklediğimiz kutulardır. Girilen değerin ne olduğunu **Text** ve **Value** özellikleri ile elde ederiz. TextBox'larda bu iki özellik genelde aynı değeri verir. Text ve Value farkını aşağıda daha detaylı göreceğiz.

**ControlSource**: Kutuya, bir hücreden değer ataması yapmak istiyorsak bu özelliği kullanırız. UserFormlarda pek kullanılmaz.

**Multiline**: Kutumuz, birden çok satır içerecekse bu özelliğe True atarız.

**EnterKeyBehaviour**: Buna True atandığı zaman Enter tuşu ile bir alt satıra geçebilirsiniz. False durumundayken alt satıra geçmek için Ctrl+Enter kombinasyonunu kullanmanız gerekir. Tabi alt satıra geçmesi için Multiline özelliğine True atanmış olmasını söylemeye gerek yok sanırım.

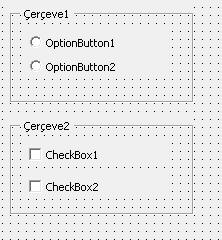
#### OptionButton'ları, Check Box'lar ve Çerçeveler

##### OptionButton ve CheckBoxlar

Option buttonları kullanıcıya **birden çok seçenek içinden birini** seçtirmek için kullanılır. Checkboxlar ise **birden çok seçenek içinden çoklu** seçim yapmaya imkan sağlar. İkisinde de seçenek sayısının az olması tercih sebebedir, çok seçenek olacağı zaman ListBox veya ComboBox kullanılması önerilir.

##### Çerçeveler

Çerçeveler, genelde Option butonları ve CheckBox'ları gruplamak için kullanılmakla birlikte, ortak özelliği olan bütün kontrolleri gruplamakta kullanılabilir. (Bunlar .Net'taki GroupBox'larla aynı işlevi görürler)



Gruplamanın amacı sadece estetik ve anlamsal bir bütünlük katmak değil, aynı zamanda çerçeve içindeki tüm kontrolleri tek seferde enable/disable veya visible/invisible yapmak için de oldukça kullanışlıdır.

##### Frame alternatifi

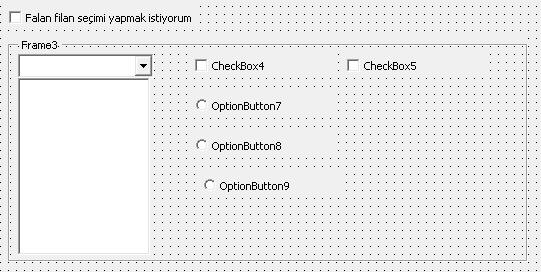
Bir grup OptionButton/CheckBox yaratmanın alternatifi de bu kontrollerin **GroupName** özelliğine ortak bir değer atamaktır. Bu şekilde kullanıldığında biri seçiliyken öbürleri seçimsiz olurlar. Başkaları önerse de ben bu şekilde bir gruplamayı tercih etmiyorum. Zira yukarda belirttiğim gibi gruplamanın amacı kontrolleri sadece aynı çatı altında toplamak değil, tek seferde visible/enable özelliklerini de kontrol etmektir.

GroupName'i önerenler tarafından öne sürülen avantajlarını ve benim yorumlarımı şöyle sayabiliriz.

* Fazladan bir kontrol koymayarak kodun performansını artırırsınız(Ben bunun ihmal edilebileceğini düşünüyorum)
* Frame içindeki tüm kontrollerin frame içine sığdırılması zorunludur, bu da sıkışık bir görüntüye neden olabilir. GroupName kullanımında ise kontroller formun istediğiniz yerinde olabilir(Neden olsun ki, bir seçeneği formun sağ üstüne diğerini sol üstüne koyacak değilsiniz ki!)
* Çerçeveli bir görüntü istemiyorsanız kullanışlıdır. Framede ise transparanlığı bozmuş olursunuz.(Genelde çerçeve sınırı olur, yani Frame tercih edilmelidir)

##### Başlangıç ayarları ve seçimler

Bir CheckBox düşünün, ilk başta seçili değil. Bu checkbox seçildiğinde konuyla ilgili diğer tüm kontrolleri içeren bir çerçeveyi görünür hale getiriyor. İlk başta bu çerçevenin Properties'ten Visible özelliğine False atarız ki bunlar ilk başta görünmesin. Şimdi, Formumuz açıldığında Checkbox'ı seçtiğinizde onla ilgili diğer tüm kontrollerin de visible olmasını, seçimi tekrar kaldırdığınızda ilgili çerçevenin de tekrar gizlenmesini istiyoruz. Formumuz ve kodumuz aşağıdaki gibidir.



Private Sub CheckBox3\_Click()

If CheckBox3.Value = True Then

frAktifPasif.Visible = True

Else

frAktifPasif.Visible = False

End If

End Sub

Yani diyoruz ki, CheckBox3(Falan filan..... yazan) seçiliyorsa frAktifPasif çerçevesini(ve dolayısıyla içindeki tüm kontroller) gizle, seçili değilse göster.

Tabi bunu yapmanın daha basit bir yolu var. Yazım şekli şu şekildedir.

**Kontrol.BooleanÖzellik=Not Kontrol.BooleanÖzellik**

Yani diyoruz ki, kontrolün ilgili özelliğine zıttını ata. Boolean tipteki tüm zıt değer atamalarında bu işlem yapılabilir.

Private Sub CheckBox3\_Click()

frAktifPasif.Visible = Not frAktifPasif.Visible

End Sub

Bu şekilde yukarıda bahsettiğimiz gibi tek seferde tüm frame içindeki kontrolleri yönettik. Frame yerine GroupName özelliğini kullansaydık, bunları tek tek yapmak gerekecekti.

#### Spin Button ve ScrollBar

Kullanıcının bir işlem yaparken değerleri tek tek(5er 5er, 10ar 10ar v.s) artırma/azaltma gibi denemeler yapması sözkonusuysa bu kontrolleri kullanırız. Bunlar, kullanıcıdan Textboxa veya bir hücreye her seferinde bir fazla/eksik değer girmesini beklemenin daha pratik bir yöntemini bize sunar.

Genelde tek başına kullanımları yoktur. VBA tarafında bir Textbox içindeki değeri veya bir değişkenin tuttuğu değeri belli miktarda değiştirmek için kullanılırlar. Excel sayfasında ise, bir hücre içeriğini değiştirmek için kullanılabileceği gibi Çeşitli Örnekler bölümünde göreceğimiz gibi Excel Filtre değerleri arasında dolaşmak için de kullanılabilirler. Biz şu an VBA tarafına odaklanalım.

Bu iki kontrolün de **Orientation** özelliğine Vertical/Horizontal değerlerini atayarak yatay mı dikey mi duracağını belirleyebilirsiniz.

**Min, Max, SmallChange** ikisinde de ortak olup açıklamaları şöyledir:

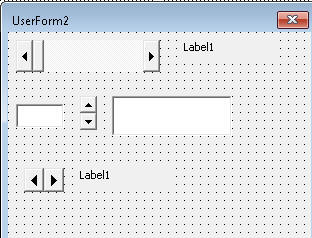
**Min**: Kontrolün alacağı en küçük değerdir, negatif olabilir.

**Max**: Kontrolün alacağı en büyük değerdir, negatif olabilir.

**SmallChange**: Oklara tıklandığında olacak değişim miktarını gösterir.

Scrollda ise fazladan **LargeChange** var. Bunda Scrollbarın ortasına tıklandığında kaçar kaçar değişeceğini belirtiriz. Normal değişim miktarı 1 ise bunu 10 yapabilirsiniz mesela.

Aşağıdaki örneğe bakalım,



Üstteki scroll ve ortadaki spin için şu kodları yazabiliriz.

Private Sub ScrollBar1\_Change()

lblSıra.Caption = ScrollBar1.Value \* 2

End Sub

Private Sub SpinButton1\_Change()

txtNo.Value = SpinButton1.Value

End Sub

Ben bunların min/max özelliğini Properties'ten ayarladım. Tabi istenirse Runtime sırasında da bunlar değiştirilebilir. Mesela bir ComboBox'tan veya Textbox'tan değişimin kaçar kaçar yapılacağını kullanıcıya da bırakabiliriz.

Private Sub TextBox1\_Change()

If TextBox1.Value > 100 Then

MsgBox "1-100 arası değer girilmelidir"

Exit Sub

End If

SpinButton1.SmallChange = TextBox1.Value

End Sub

Şimdi de en alttaki Spine bakalım. Onun için önce modülün başında bir global değişken(Dictionary olacak) tanımlayıp, form yüklenir yüklenmez de içine 5 değer atıyorum.

Public dict As Object

Private Sub UserForm\_Initialize()

Set dict = CreateObject("Scripting.Dictionary")

dict.Add 1, "Volkan"

dict.Add 2, "Ayşe"

dict.Add 3, "Elif"

dict.Add 4, "Murat"

dict.Add 5, "Hakan"

End Sub

Bu sefer değişimi 1er 1er yaptırıp(smallchange özelliği=1) 1-5 arasındaki kişileri öğreniyorum.

Private Sub SpinButton2\_Change()

Label1.Caption = dict(SpinButton2.Value)

End Sub

#### TabStrip ve MultiPage

##### MultiPage

MultiPage'ler, bir veya daha çok Page nesnesini bir arada tutan yapılardır. Framelerin bir üst modeli olarak düşünebilirsiniz. Bir alana sadece 1 frame koyabilirken aynı alana birkaç Page'i olan bir MultiPage koyabilirsiniz. Tek farkı yerden tasarruf değil aynı zamanda daha üst seviyede bir gruplama imkanı da verir. Örneğin oluşturduğunuz form, departmanınızdaki raporlara ulaşmayı sağlayan bir arayüz ise, kullandığınız Multipage'in sayfalarından biri Kredi raporlarını diğeri Mevduat raporlarını v.s gruplamış olabilir. Çeşitli Örnekler bölümünde bununla ilgili bir çalışmamız olacak.

İlk başta bir Multipage içinde iki sayfa bulunur. Yeni sayfalar eklemek için en üste sağ tıklayıp "Add Pages" diyin. Her sayfanın içindeki kontroller, diğer sayfalardan tamamen bağımsızdır.

Sayfalar 0 nolu indexten başlarlar. Bunlara index numarasıyla ulaşabileceğiniz gibi sayfa ismi veya obje ismiyle de ulaşabilirsiniz.

MultiPage1.Pages(0).Caption 'index

MultiPage1.Pages("Krediler").Caption 'sayfa ismi

MultiPage1.Page4.Caption 'obje ismi

Private Sub MultiPage1\_Change()

MsgBox "sayfa indeksi: " & MultiPage1.Value 'seçili sayfanını indexini

MsgBox "SelectedItem.Name yani obje adı: " & MultiPage1.SelectedItem.Name 'seçili sayfanın obje adını

MsgBox "SelectedItem.Caption: " & MultiPage1.SelectedItem.Caption 'seçili sayfanın adını

End Sub

##### TabStrip

TabStrip kontrolü görünüm olarak MultiPage'e çok benzemekle birlikte, bunun içine koyduğumuz kontroller tüm sayfalarda aynen görünür, yani MultiPage'de olduğu gibi farklı sayfalarda farklı kontroller bulunmayabilir. **Ancak burda kritik olan, kontrollerin içeriğinin farklı olmasını sağlıyor olmamızdır**. Bunu da Tab değiştikçe(bunu bir eventle yönetiriz) içeriğin değişmesini sağlayacak bir kodla sağlarız. MultiPage'de ise Event olmasına gerek yok, zaten her sayfa birbirinden bağımsız içeriğe sahiptir.

Tablara erişim şekli MultiPage'de Page'lere erişim ile aynıdır. [http://bigdon-in-VBAland.blogspot.com.tr/2014/01/tabstrip-vs-multipage.html](http://bigdon-in-vbaland.blogspot.com.tr/2014/01/tabstrip-vs-multipage.html) sayfasında iki kontrol arasındaki farkları daha detaylıca görebilirsiniz.

Şimdi kendi **TabStrip** örneğimize geçebiliriz.

Bu örnekte Listbox da var, bunun detayını daha aşağıda göreceğiz, şimdilik ona takılmayın. Sadece listeyi doldurduğumuzu bilin o kadar.

Private Sub UserForm\_Initialize()

'form yüklenir yüklenmez ilk sekme açılır ve 2.sayfadan(0+2) data yüklenir

TabStrip1.Value = 0

Call ListeDoldur(0)

End Sub

Private Sub TabStrip1\_Change()

'her sayfa değişiminde ilgili sayfaya ait ürünler doldurulur

Dim ws As Worksheet

Dim alan As Range

Dim i As Integer

i = TabStrip1.Value 'tab'larda index 0'dan başlar

Call ListeDoldur(i)

End Sub

Sub ListeDoldur(k As Integer)

Set ws = ActiveWorkbook.Worksheets(k + 2) 'worksheetlerde index 1den başlar

Label1.Caption = ws.Name

Set alan = ws.Range("A1").CurrentRegion 'ilgili sayfad

ListBox1.Clear 'önce listeyi boşaltalım

For Each urun In alan

ListBox1.AddItem (urun)

Next urun

End Sub

### Liste Kontrolleri

**Combobox** ve **Listboxlara** önceden belirlenmiş değerler atanabileceği gibi form üzerindeki düğmeler aracılığıyla, bunların içeriği zenginleştirilebilir veya içlerindeki elemanlar silinebilir.

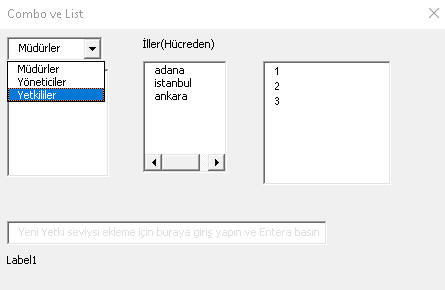
#### Karşılaştırma

Comboboxlar kullanıcıya tek değer gösterirken Listboxlar tüm değerleri tek seferde gösterebilir(hepsi sığmazsa scrollbar çıkar). Eğer amacınız tüm değerleri tek seferde göstermek değilse yerden tasarruf amacıyla Combobox tercih edebilirsiniz.

Diğer önemli fark ise Combobox’tan sadece 1 eleman seçebilirken Listbox’tan ise çoklu eleman seçimi yapabilirsiniz.

Bir diğer fark ise, Comboboxlar’ın, listede olmayan bir değeri girmeye izin vermesidir. Listbox’ta bu mümkün değildir.

Aşağıdaki görselde, bu iki kontrolü görebilirsiniz. Combobox açılmış durumdadır.



Şimdi de bu kontrollerin çeşitli üyelerine(özellik, metod ve olay) bakalım. Öncelikle şunu söyleyeyim. Bazı özelliklere hem kod yazarken hem properties penceresinden, bazısına ise sadece kod yazarken erişilebilir. "Properteis'ten erişebiliyorken neden kod ile uğraşayım ki?" diye düşünebilirsiniz. Bunun bir cevabı "Eğer form üzerinde birbiriyle aynı türde çok fazla kontrolünüz varsa, mesela 10 tane combobox gibi, her birine tek tek değer atama yerine, döngüsel şekilde tek seferde kod ile yapabilirsiniz." olabileceği gibi, diğer bir cevap ise "runtime sırasında değer atama gerekliliğidir". Mesela bir butona tıkladığınızda başka bir kontrolün **Enabled** özelliğine False değeri atamak gibi.

#### Listeleri doldurma

##### Yöntem1

Liste doldurma yöntemlerinden en bilineni ve basit olanı, Formun **Initialize** eventi içine dizi olarak eklemektir.

Private Sub UserForm\_Initialize()

cbYetkiSeviye.List = Array("Müdürler", "Yöneticiler", "Yetkililer")

Me.cbYetkiSeviye.ListIndex = 0 'ilk eleman seçilir. -1 ile seçili hiç bir eleman olmaz, son eleman için me.cbYetkiSeviye.ListCount - 1

End Sub

Diğer yöntemler arasında Excel’deki bir sayfadan okuma, text doyasından okuma veya Access gibi bir veritabanından okuma olabilir. Bu işlemleri yine Initialize içinde yapabileceğiniz gibi bir Button'a tıklayarak da yapabilirsiniz, tabi pratikte genelde listeler form açıldığında, yani Initialize sırasında, doldurulur. Bunların hepsinde de ilgili kontrolün **AddItem** metodu kullanılır.

##### Yöntem2

İkinci yöntem olarak bir text dosyadan okuma yapalım:

dosya = "C: \....\Ornek\_dosyalar\Makrolar\userformlist.txt"

Open dosya For Input As 1

Do Until EOF(1)

Line Input #1, Content

Me.ListBox2.AddItem Content

Loop

Close #1

##### Yöntem3

Excel sayfadan okuma için aklınıza döngüler gelmiş olabilir, ne var ki buna hiç gerek yok. İlgili alanı Properties'ten **RowSource** özelliğine referans verebilirsiniz. Ör. Sheet1!A17: A19(Sayfa adını ve ! işaretini belirterek) veya runtime sırasında **lbŞehirler.RowSource=Range("A17: A19").Address** diyebilirsiniz.

Değerlerin bulunduğu alan sabit değil de değişkense bunun için aşağıdaki gibi bir kod kullanabilirsiniz.

Private Sub RefEdit1\_Change()

Me.ListBox2.RowSource ="Sheet1!A1: A"& Sheet1.Cells(Rows.Count, "A").End(xlUp).Row

End Sub

Bu işlemi bir **refEdit** elemanına da yaptırabiliriz.

Private Sub RefEdit1\_Change()

Me.ListBox2.RowSource = Me.RefEdit1.Value

End Sub

**NOT**: RefEdit kontrolünü form modal açılmışken kullanmalısınız, modeless açılmış formlarda sıkıntı yaşanmaktadır.

##### Yöntem4

Access'ten okuma yapmak için ya DAO ya ADO tekniklerini biliyor olmak gerekiyor. Bunlar için Veritabanı Programlama konusuna bakınız.

Bu arada belirtmek isterim ki yeni eleman eklemelerini sona yapacaksınızdır ama olur da başa veya arada bir yere yapmak isterseniz,  **AddItem**'ın ikinci parametresini kullanabilirsiniz. **Ve unutmayın ki liste kontrollerindeki indexler 1'den değil 0'dan başlar.**

Me.ListBox2.AddItem Content,0 'ilk sıraya ekledi.

##### AddItem detaylar

AddItem metodu ekleme işini, çok kolonlu bir listenin ilk kolonuna yapar. Daha ileri kolonlara ekleme yapmak için List veya Column propertylerini kullanabilirsiniz. Yine aynı propertyler kullanılarak aynı anda birden fazla satır da ekleyebilirsiniz. Bu da şu anlama gelir: Excel sayfasındaki bir grup hücreyi tek seferde ilgili liste kontrolüne ekleyebilirsiniz.

Bir diziyi olduğu gibi eklemek için List propertysi kullanılırken, transpose halini eklemek için Column propertysi kullanılır. ,Yani dizi(i,j)'yi olduğu gibi eklemek için listbox.List(i,j)  kullanılabilir

Aşağıda bununla ilgili bir örnek bulabilirsiniz, ki bu yöntem elemanların bulunduğu alanı **Rowsource** olarak belirtmenin bir başka yoludur. Ancak AddItem ile ilgili önemli bir detay da, liste kontrolümüz bir datayla ilişkiliyse(Rowsource ile) Additem'ın çalışmayacağıdır. O yüzden sonrasında dinamik şekilde yeni elemanlar eklemek istiyorsak Rowsource ile değil aşağıdaki gibi ilerlemeliyiz.

Private Sub CommandButton10\_Click()

Me.lstÇiftKolon.List = Range("çiftkolon").Value

End Sub

#### Listeleri boşaltma

Listeleri **Clear** metodu ile boşaltırız. Genelde dolu olan bir listeyi tekrardan doldurmadan önce boşaltmak iyi bir fikirdir. Özellikle formu henüz kapatmamışken, işleri baştan almak istediğinizde formun doldurma işlemi varsa, bu mükerrer doldurmaya neden olacağı için kodumuza her zaman (boş bile olsa) önce listeyi boşaltarak başlamak iyi bir pratiktir.  **AddItem** gibi bu metod da, eğer ki listemiz bir veri kümesine bağlıysa çalışmaz. Böyle bir durumda öncelikle **RowSource** özelliğinin temizlenmesi gerekir.

Sub listeboşalt()

lbŞehirler.Rowsource=""

lbŞehirler.Clear

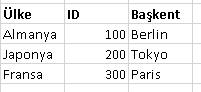
End Sub

Eğer ki sadece belirli bir elemanı çıkarmak istiyorsak **RemoveItem** metodunu kullanırız. Parametre olarak kaçıncı elemanın çıkartılacağı verilir.(Index'in 0'dan başladığını unutmayın).

#### Liste öğelerine erişim

Daha önceki kontrollerde **Text** ve **Value** özelliklerinden bahsetmiştik. Bunlar diğer kontrollerde neredeyse her zaman eşittirler, ancak liste kontrollerinde farklı olma durumları oldukça rastlanan durumlardır.

Liste kontrollerinde **Text**, sizin gördüğünüz değeri verirken, **Value** altta yatan değeri verir. Örneğin listeyi bir veritabanındaki 2 kolonlu bir tablodan(veya Excel’de 2 kolonlu bir alandan) doldurmuşsunuz diyelim. İlk kolon şehir ismi ikinci kolon şehir kodudur. **ColumnCount** özelliğine 1 derseniz, sadece bir kolon gösterilecektir. İlk kolon listbox içinde gösterilecektir, ancak listboxtan şehir seçimi yapıldığında **Value** değerine şehir adı değil de kodu atansın istiyorsak **BoundColumn** özelliğine 2 atarız. **TextColumn** özelliğine ise 1 atarız. Çok kolonlu listelerde TextColumn'un genelde 1 yapıldığı görülür ancak pratikte bunun farklı olduğu durumlar olabilir. Örneğin aşağıdaki örnekte listeye ülkeler yüklenir, Value olarak id tutulur, Text olarak da başkentler tutulabilir.



Bunun için yapılması gerekenler:

**ColumnCount**: 1 (Buradaki 1, **kaç** kolon gösterilecek anlamında) **BoundColumn**: 2 (Buradaki 2, **kaçıncı** kolon Valueolacak **TextColumn**: 3 (**kaçıncı** kolon text değerini tutacak, yani gösterilecek)

Liste kontrollerinde eğer büyük veritabanlarıyla çalışıyorsak performans açısından **Value** özelliği ile işlerimizi halletmeliyiz.

**NOT**: TextColumn'a -1 atandıysa(default budur), Text özelliği seçilen değerin görünen değerini, 0 verilmişse elemanın indexini, 0'dan büyükler için kaç verilmişse o kolondaki değeri verir. Yani ilk kolon için TextValue=1, ikinci kolon için TextValue=2 v.s

##### Eleman erişimi

Peki hangi elemana erişeceğimizi nasıl belirliyoruz? **List** propertysi ve elemanın index numarası ile.

listbox1.List(0) 'ilk eleman

Normalde List property'si iki eleman alır: satır ve sütun. İkinci eleman belirtilmezse ilk kolon baz alınır. Yani yukardaki kod ile listbox1.List(0,0) özdeştir. (Not: List propertysinin parametreleri 0'dan başlar, 1'den değil)

Peki indexi bilmiyorsak, yani dinamik bir şekilde ele almamız gerekiyorsa, onun da yolu var. Aşağıdaki örneğe bakalım:

**ListIndex ve List** özellikleri**:** O an seçili elemana erişim için bu iki özellikle kombine bir şekilde kullanılır. **lbYıl.List(lbYıl.ListIndex)**.

ListIndex bize o an seçili elemanın indexini verirken, bu indexi List propertry'sine parametre gönderince seçili elemanın görünen değerini bize verir. ListIndex 0'dan başlar. (Yukarıda bahsettiğimiz gibi TextColumn özelliğine 0 atayarak da indeksi elde edebiliyoruz)

Aşağıda 3 ayrı değer erişim yöntemi bulunuyor. Farkları inceleyerek anlamaya çalışın. Örnek olarak Japonya seçilyse;

Private Sub CommandButton4\_Click()

MsgBox "Value: " & lstBağımlı.Value '200

MsgBox "Text: " & lstBağımlı.Text 'Tokyo

MsgBox "List&listindex: " & lstBağımlı.List(lstBağımlı.ListIndex) 'Japonya

End Sub

##### Çok kolona erişim

Çok kolona erişmeyi yine **List** özelliği ile yapıyoruz. Bu yöntemi sadece erişim için değil, veri ekleme için de kullanabilirsiniz.

lstSozluk.AddItem "iyi"

lstSozluk.List(0,1)="good" 'ikinci kolona

lstSozluk.List(0,2)="gut" 'üçüncü kolona

Çok kolonlu bir listeye yeni eleman eklemek de şöyle olur

Private Sub CommandButton1\_Click()

Me.lst1.AddItem "kötü"

Me.lst1.List(lst1.ListCount - 1, 1) = "bad"

Me.lst1.List(lst1.ListCount - 1, 2) = "schlecht"

End Sub

##### Listbox'ta çoklu seçim: MultiSelect özelliği

MultiSelect özelliğinin alabileceği 3 değer vardır.

**fmMultiSelectSingle** (numerik değeri 0): Tekli seçim. Her elemana tıklayışta sadece o seçilir.

**fmMultiSelectMulti** (numerik değeri 1): Her tıklamada, tıklanan eleman seçili kalır, tekrar aynı elemana tıklanırsa seçim kalkar.

**fmMultiSelectExtended** (numerik değeri 2): İki eleman arasındaki tüm elemanları tek seferde seçmek için SHIFT tuşuna basılır. CTRL tuşu ile ise fmMultiSelectMulti modu taklit edilebilir.

Çoklu seçimde hangi elemanların seçili olduğunu **Selected** özelliği ile test edebiliriz. Parametre olarak elemanın indexini alır: **Listbox1.Selected(n)**

Mesela aşağıdaki kod ile sadece seçili elemanları bir Collection'a atıyoruz.

Private Sub CommandButton5\_Click()

Dim coll As New Collection

For i = 0 To ListBox3.ListCount - 1

If ListBox3.Selected(i) Then

coll.Add ListBox3.List(i)

End If

Next i

MsgBox "collectionda " & coll.Count & " adet eleman var"

End Sub

##### Listedeki elemanları bir collection'a atama

Yukarıdaki işlemi bir de fonksiyon haline getirirsek bundan sonra ne zaman bir listbox’tan seçili elemanları almamız gerekse bu fonksiyonu kullanabiliriz.

Function ListBoxtakiSeçiliElemanlarıSeç(lst As MSForms.ListBox) As Collection

Dim col As New Collection

If lst.List(lst.ListIndex) = -1 Then GoTo atla

For i = 0 To lst.ListCount - 1

If lst.Selected(i) = True Then col.Add lst.List(i)

Next i

atla:

Set ListBoxtakiSeçiliElemanlarıSeç = col

End Function

'Kullanımı

Sub testListBox()

Dim col As Collection 'new yok, fonksiyonla dolduracağız

Set col = ListBoxtakiSeçiliElemanlarıSeç(UserForm1.LitBox1)

End Sub

##### Listede belirli bir elemanı seçmek(işaretlemek)

Şimdiye kadar elemana erişim ile hep onun değerini elde etmeyi kastettik. Ancak bazen ilgili elemanı seçmek de isteyebiliriz. Bu işlem genelde, listedeki ilk elemanı seçmek için yapılır, ancak tabiki herhangi bir eleman seçiminde de kullanılabilir.

Bunun için iki yöntem var:

Private Sub CommandButton5\_Click()

ListBox3.Selected(0)=True 'Çoklu seçim modunda işe yaramaz

'veya

ListBox3.ListIndex=0

End Sub

##### Kolon gizleme

3 kolonlu bir veri kümemiz olsun. Diyelim ki üçünü değil de baştaki ile sondakini almak istiyorsunuz. Böyle bir durumda üçünü de RowSource'a alırız, ancak ortadakini gizleriz. Gizlemek için **ColumnWidths** özelliğine 0 atarız. Ancak ColumWidths özelliği kullanılırken maalesef tek bir kolona değer ataması yapılamıyor, üç kolon için de değer girmek lazım.

listbox1.ColumnWidths="50;0;50"

##### ListBox'ta dinamik filtreleme

Filtreleme amacı gören bir textbox'a yazacağınız metinlerle bir listbox'taki elemanlarda dinamik filtreleme yapabilirsiniz. Bunun için yol haritası şöyledir:

* Global bir Collection oluşturun
* Bu collection'ı ve ilgili listbox'ı aynı elemanlarla formun başlangıcında doldurun
* Textbox'ın Change eventine de ilgili filtreleme kodunu yazın

Kodlar aşağıdaki gibi olabilir:

'Global değişken

Dim ülkelerCol As New Collection

'Form başlangıcı

Private Sub UserForm\_Initialize()

For Each ülke In Range("ülkeler")

ülkelerCol.Add ülke.Value

Me.lstDinamik.AddItem ülke.Value

Next ülke

End Sub

'TextBox change eventi

Private Sub txtFiltre\_Change()

Dim filtreliÜlkeler As New Collection

Me.lstDinamik.Clear 'önce boşaltıyoruz ki mükerrerlik olmasın

For Each ü In ülkelerCol

If InStr(1, ü, txtFiltre.Text, vbTextCompare) > 0 Then filtreliÜlkeler.Add ü

Next ü

For Each ü In filtreliÜlkeler

Me.lstDinamik.AddItem ü

Next ü

End Sub

### Diğer detaylar

#### Value, Text, Name, Caption

Yukarıda bahsettiğimiz konulara biraz daha detaylı bakalım.

**Text**: Ekranda gördüğümüz metni verir.

**Value**: Arkaplanda tutulan değeri verir.

Bu iki özellik genelde aynı değeri verir. Şu istisnalar hariç:

* Söz konusu kontrol bir listbox veya combobox ise
* Gösterilen değer bound column'dan farklı ise

##### Value detaylar

* Multiselect moddaki listbox’ta Value kullanılamaz
* Multicolumn listbox’ta BoundColumn varsa Value değeri seçili satırdaki bu kolondaki değeri verir
* Multipage'de sayfa indexini verir
* Checkbox OptionButton ve ToggleButtonda ilgili kontrolün seçili olup olmadığını verir. Seçiliyse True, aksi halde False
* Spin ve ScrolBarda o anki değeri verir
* TextBox'ta Text ile aynı değeri verir.

**Caption**: Label'da yazan metni, Form'da ise form başlığını verir. Gariptir ki, Label'da Text veya Value özelliği yerine Caption konmuş.

**Name**: Nesnenin adını verir. Kod sırasında bu nesneye bu isimle başvuru yapılabilir. Ör: Yılları gösteren combobox’a "cbYıllar" diye çağırdığımız gibi. Bu özelliği **If control.Name="cbYıllar"** şeklinde döngüsel bir kod içinde ilgili nesnenin belirli bir nesne olup olmadığını kontrol etmek için de kullanabiliriz.

#### List özellikleri

##### ListCount

Readonly olan bu özellik, ilgili liste kontrolündeki satır sayısını verir. ListRows'daki Rows ifadesi biraz kafa karışıklığı yaratabilir ama satır sayısını ListRows değil ListCount vermektedir. Bu özelliğe sadece kod ortamında ulaşılabilir.

##### ListRows

Sadece comboboxlar’da bulunan bu özellik, combobox’ta gösterilecek eleman sayısını verir. Default değeri 8'dir. Belirtilen değerden daha fazla satır varsa kenarda scrollbar çıkar. Aşağıdaki kod ile dinamik bir şekilde gösterilecek eleman sayısını kontrol edebilirsiniz. Eğer comboboxtaki eleman sayısı 5ten büyükse 5le sınırlayalım, 5ten küçükse kaç satırsa o kadar görünsün.

Private Sub UserForm\_Initialize()

With ComboBox1

If .ListCount > 5 Then

.ListRows = 5

Else

.ListRows = .ListCount

End If

End With

End Sub

#### Me

Üzerinde çalıştığınız formun kendisine **Me** ifadesi ile başvurabilirsiniz. Bu ifade, sadece forma başvuru için faydalı değil aynı zamanda form üzerindeki kontrollere intellisense yardımıyla hızlıca ulaşma imkanı verdiği için de faydalıdır.

#### Kontrolleri tek tek dolaşma

Bazı durumlarda formdaki tüm kontrollerde dolaşıp, onların tipine(TypeNeme), adına(Name) veya başka bir özelliğine bakarak işlem yapmak isteriz. Bunu **Controls** collection'ına For Each uygulayarak yaparız.

Aşağıdaki örnekte Label olan tüm kontrollerin adını yazdırıyoruz.

For Each ctrl In Me.Controls

If TypeName(ctrl) = "Label" Then

Debug.Print ctrl.Name

End If

Next ctrl

Dolaşmak istediğimiz kontroller belli bir çerçeve(Frame) içindeyse;

For Each ctrl In Me.Frame1.Controls

If TypeName(ctrl) = "Label" Then

Debug.Print ctrl.Name

End If

Next ctrl

Tüm framelerde dolaşmak için

For Each cf In Me.Controls

If TypeName(cf) = "Frame" Then

For Each ctrl In cf.Controls

If TypeName(ctrl) = "Label" Then

Debug.Print ctrl.Name

End If

Next ctrl

End If

Next cf

#### Event detayları

##### Mouse eventleri

**MouseDown**: Mouse tuşu basıldığında meydana gelir

**MouseUp**: Mouse tuşu bırakıldığında meydana gelir

**Click**: Mouse ile tıklanabilir bir kontrole tıklandığında meydana gelir

Önce MouseDown olur, sonra MouseUp, en sonra Click. İlk ikisi hem sol hem sağ tuş ile tetiklenebilirken Click sadece sol tuş ile tetiklenir. Mesela bir kontrolün ucuna tıklayıp yeniden boyutlandıracaksanız, tıkladığınız anda MouseDown gerçekleşir, yeniden boyutlandırma bittiğinde ve mousetan elinizi çektiğinizde Up gerçekleşir.

**MouseMove**: Üzerinden geçerken gerçekleşir. Bunu çok kullanma durumum olmadı açıkçası. İlgili kontrolün üzerine gelindiğinde bir mesaj vermek istiyorsanız bunu **ControlTip** özelliği ile de verebilirsiniz.

Buton parametresiyle sol/sağ hangisine basıldığı tespit edilebilir. Mouse tuşlarının nasıl öğrenileceğini aşağıda klavye tuşlarının olduğu bölümde görebilirsiniz.

**X ve Y** parametreleriyle hangi noktalara basıldığı tespit edilebilir, yine bunlar da çok kullandığım özellikler değiller.

**Shift** parametresiyle Shift, Ctrl, Alt tuşlarından birine basılıp basılmadığı kontrol edilebilir.

* 1: SHIFT
* 2: CTRL
* 3: SHIFT+CTRL
* 4: ALT
* 5: ALT+SHIFT
* 6: ALT+CTRL
* 7: üçüne birden

Aşağıda çeşitli örnekler bulunmakta.

Private Sub CommandButton9\_MouseDown(ByVal Button As Integer, ByVal Shift As Integer, ByVal X As Single, ByVal Y As Single)

MsgBox "mousedown-" & Button & "-" & Shift & "-" & X

End Sub

--------------------------

Private Sub CommandButton9\_MouseUp(ByVal Button As Integer, ByVal Shift As Integer, ByVal X As Single, ByVal Y As Single)

MsgBox "mouseup-" & Button & "-" & Shift & "-" & X

End Sub

-------------------------

Private Sub ListBox1\_MouseDown(ByVal Button As Integer, ByVal Shift As Integer, ByVal X As Single, ByVal Y As Single)

If Button = KeyCodeConstants.vbKeyRButton Then

ListBox2.AddItem ListBox1.List(ListBox1.ListIndex)

End If

End Sub

--------------------------

Private Sub cbYıllar\_MouseDown(ByVal Button As Integer, ByVal Shift As Integer, ByVal X As Single, ByVal Y As Single)

cbYıllar.DropDown

End Sub

##### Klavye eventleri

3 adet klavye eventi vardır. Bunlar **KeyDown**, **KeyPress** ve **KeyUp** olup bu sırayla meydana gelirler. KeyDown ve Keyup parametre olarak Keycode alırken, KeyPress KeyAscii alır.

Hangi tuş veya tuş kombinasyonlarına(Ctrl+Enter gibi) basıldığını öğrenmek için kullanılırlar.

Mesela bazen yer tasarrufu yapmak amacıyla Textboxa yazılan metinle ilgili bir iş yapmak için form üzerine button koymak yerine yazmayı bitirdikten sonra Enter'a(veya Ctrl+Enter) basılması durumunda ilgili işlemin yapılmasını sağlayabilirsiniz.

Private Sub TextBox1\_KeyDown(ByVal KeyCode As MSForms.ReturnInteger, ByVal Shift As Integer)

If KeyCode = 13 Then

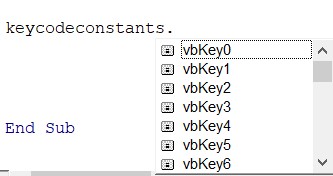
Me.cbYetkiSeviye.AddItem Me.TextBox1.Text

Me.Label1.Caption = "Yetki seviyelerine " & Me.TextBox1.Text & " eklendi"

End If

End Sub

KeyCodelari aşağıdaki linklerden bulabileceğiniz gibi, VBA'de **KeyCodeConstants** yazıp "."'ya basınca intellisense aracılığı ile constant değerlerini de yazabilrsiniz.



**Not: Enter** için **vbKeyReturn** diye bakmak lazım, vbKeyEnter diye bişey bulunmuyor.

* [http: //www.asciitable.com/](http://www.asciitable.com/)
* [https: //msdn.microsoft.com/en-us/VBA/language-reference-VBA/articles/keydown-keyup-events](https://msdn.microsoft.com/en-us/vba/language-reference-vba/articles/keydown-keyup-events)
* <https://stackoverflow.com/questions/1367700/whats-the-difference-between-keydown-and-keypress-in-net> sayfasında ise .Net dilindeki farklar detaylıca anlatılıyor ama bu açıklamaların prensipte VBA için de geçerli olduğunu söyleyebilirim.

##### Birçok kontrol için tek event tanımlama

Formumuzda diyelim ki 10 adet textbox var, ve hepsi için de ortak bir Event tanımlamak istiyorum. Mesela içine girince içindeki yazı silinsin istiyorum. Bunun için tek tek her birine event tanımlamak zahmetli olacaktır. İşte böyle durumlar için custom eventlerden yararlanıyoruz. Örnek dosyayı **githubdan(userform\_multitextbox.xlsm)** indirebilirsiniz.

Adımlarımız şöyle:

* Öncelikle bir Class Modül yaratırız. Tepesine aşağıdaki kodu yazarız. Biz burada TextBox için yazıyoruz ama farklı kontroller için de aynısı uygulanabilir.

Public WithEvents txtGroup As MSForms.TextBox

* Sonra tepeden nesne kutusunda txtGroup seçilir, yandan da mousedown eventi seçilir.(Custom TextBoxlarda Enter eventi bulunmuyor, ama mousedown da aynı görevi görecektir. Tabi ilgili kutulara mouse ile tıklanması kaydıyla, tab tuşuyla ilerlenerek gelinirse tetiklenmez). İçine de aşağıdaki kod yazılır.

Private Sub txtGroup\_MouseDown(ByVal Button As Integer, ByVal Shift As Integer, ByVal X As Single, ByVal Y As Single)

With txtGroup

.Text = ""

.ForeColor = vbBlack 'Form açıldığında gri renkli bişeyler yazıyor olsun

End With

End Sub

* Son olarak Form modülüne gelip tepeye Class1 tipli bir dizi tanımlıyoruz, eleman sayısını bilmediğimiz için boyutsuz tanımlıyoruz. Initialize eventi içinde TextBoxlarda dolaşarak boyutumuzu sürekli artırıyoruz.

'Global değişken bölgesine

Dim controller() As New Class1

Private Sub UserForm\_Initialize()

Dim adet As Integer

Dim ctrl As Control

For Each ctrl In UserForm1.Controls

If TypeName(ctrl) = "TextBox" Then

adet = adet + 1

ReDim Preserve controller(1 To adet)

Set controller(adet).txtGroup = ctrl

End If

Next ctrl

End Sub

Kontrol sayısı çok ise boyutsuz dizi tanımlamak yerine Collection tanımlamak daha doğru bir çözüm olacaktır. Bununla ilgili bir örnek [https://stackoverflow.com/questions/1083603/VBA-using-withevents-on-userforms](https://stackoverflow.com/questions/1083603/vba-using-withevents-on-userforms) sayfasında bulunmaktadır.

##### Diğer Eventler

ListBox'ın Change eventi, Formun Initialize ve Terminate eventleri adları üzerinde olan eventler, bunları kurcalayarak kendinizin görmesinde fayda var. Mesela ListBox'ta bir ana ürün seçildiğinde onun yanındaki listboxa alt ürünlerin gelmesini ilk listbox'ın change eventiyle yaptırabilirsiniz. Form'un Terminate eventi ise Workbookların Close eventine benzer, form kapanırken devreye girerler ve kapanış işlemlerinizi yapmanızı sağlar.

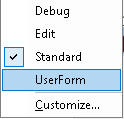
Diğer birçok eventi şimdiye kadar hiç kullanmadım. Arzu eden araştırabilir.

**NOT**: Listbox'ta seçilen bir eleman Excel sayfasındaki bir hücreyi değiştiriyorsa bu değişklik Worksheet'in Change eventini tetiklemez.

#### Hizalama ve Ölçü işlemleri

##### Userformlarda

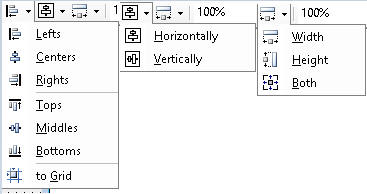
VBA editöründeyken araç çubuğuna sağ tıklayın ve UserForm çubuğunu aktive edin.



Bizim ilgileneceğimiz, kırmızı halka içindekilerdir.



Onların da içerikleri aşağıdaki gibidir. Soldakiyle çeşitli yönlerde hizalama yaparız. Ortadakiyle kontrollerin arasındaki uzaklığı eşit hale getiririz. En sağdakiyle ise kontrollerin ölçülerini eşit hale getiririz. Bunlarla oynayarak ne işe yaradıklarını daha kolay görebilirsiniz.

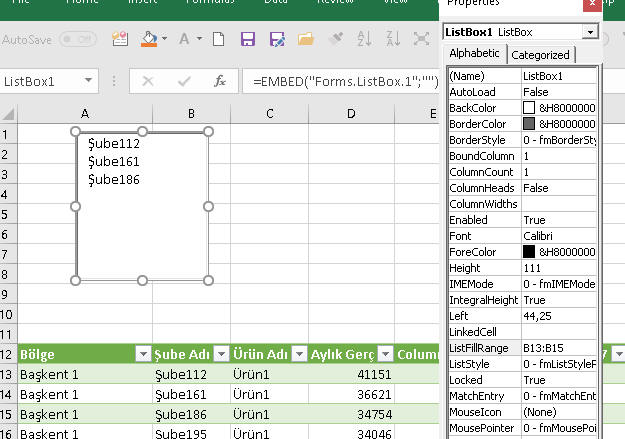


##### Worksheet'te

İlgili kontrolün konumu hücreler üzerinde rasgele durmasın da, uçları hücrelerin köşelerine gelsin istiyorsanız, ilgili kontrol seçiliyken Format menüsünden Arrange Grubundaki Align butonuna tıklayın ve açılır kutudan **Snap to Grid**(kılavuzlara dayandır) diyin, arkasından ilgili kontrolün uçlarını köşelere doğru çekin, otomatikman yerleşir (Siz bu son adımı da yapmadan köşelere otomatikman yerleşmez)

#### WorkSheet'te ActiveX listbox'a hücre bağlama

Önce developer menüsünden design moda geçilir. Sonra **ListFillRange** özelliğine istenen hücre grubu seçilerek aktarılır. Aşağıdaki örnekteki gibi.



#### Diğer özellikler

**WordWrap**: Text veya Caption özelliğine birden fazla satırda yazma özelliği verir.

**ControlTipText**: ilgili kontrolün üzerine gelince onun hakkında kısa bilgi veren, veya birtakım talimatlar içeren bir balon çıkar.

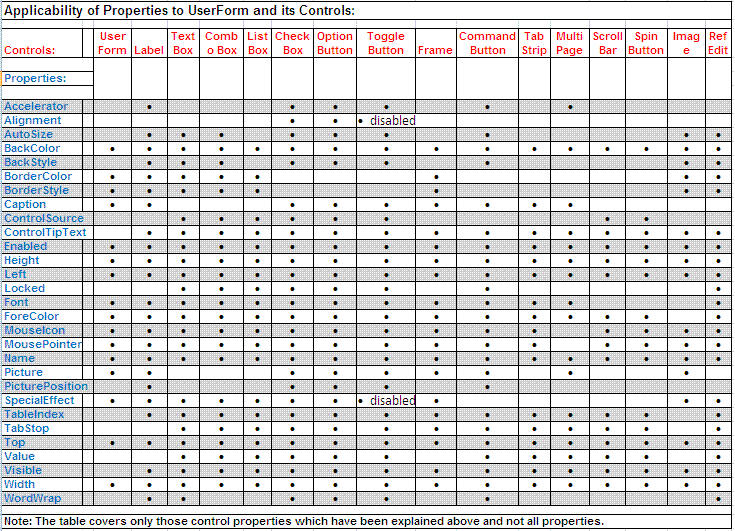
**Enabled**: İlgili elemanla etkileşime geçilip geçilemeyeceğini belirtir. Genelde bir başka kontrolle diğer kontrollerin enabled özelliği kontrol edilir.

**Visible**: Enabled'ın kullanım mantığına benzer. Bu, etkileşimden ziyade ilgili kontrolü gösterir veya gizler.

**TabIndex**: Kontroller arasında Tab tuşu ile gezinebilirsiniz. Hangi sırada gezineceğinizi bu özelliğe atayacağınız değerle yönetirsiniz.

**ControlSource**: Bir kontrolde seçtiğiniz/girdiğiniz değerin Excel’de bir hücreye de yansımasını istiyorsanız bu özelliğe o hücreyi atarsınız. Ör: Listboxtan seçtiğiniz bir şube adı A1 hücresinde de çıksın isterseniz ControlSource özelliğine A1 atayın. Genelde properties'ten designtime sırasında kullanılır.

Aşağıda <http://www.globaliconnect.com> sitesinden aldığım bir kontrol-özellik matrisi var. Bu matristen, hangi kontrolün hangi özellikleri mevcut, onları tek bakışta görebilirsiniz.



#### Çeşitli püf noktaları

UserForm kontrollerini kullanırken bazı püf noktalarını bilmek oldukça faydalı olabilmektedir. Bunlardan birkaçını aşağıda vermeye çalıştım.

**Toggle işlemi**: Bir kontrole tıklandığında onunla ilgili bir boolean işlem yapılacaksa( başka bir kontrolün enabled değerini, kendisinin durumuna veya zıttına ayarlamak gibi) bunu If bloğu içinde yapmak yerine ters/aynı boolena değer atanarak tek satırda yapabilirsiniz.

If Checkbox1.Value= True Then

Frame1.Enabled=True

Else

Frame1.Enabled=False

End If

yerine

Frame1.Enabled=Checkbox1.Value

'ters işlem yapılacaksa başına Not ifadesi konur

Frame1.Enabled= Not Checkbox1.Value

Değer girilmesi gereken yerler için kontrolünüz olsun. Ör: Mail gönderim işlemi yapan bir Formunuz varsa, Subject(Konu) alanı mutlaka dolu olmalı.

If konu.Text ="" Then

MsgBox "Lütfen konu alanını boş bırakmayın"

Exit Sub

End If

Aşağıdaki linklerde hem genel olarak önemli noktalara temas var hem de çeşitli püf noktaları da bulunuyor. Bunları da ayrıca incelemenizi tavsiye ederim.

* [Microsoft userform dokümantasyonu](https://support.microsoft.com/en-us/help/829070/how-to-use-visual-basic-for-applications-vba-to-change-userforms-in-ex)(https://support.microsoft.com/en-us/help/829070/how-to-use-visual-basic-for-applications-vba-to-change-userforms-in-ex)
* [http://what-when-how.com/Excel-VBA/userform-techniques-and-tricks-in-Excel-VBA/](http://what-when-how.com/excel-vba/userform-techniques-and-tricks-in-excel-vba/)
* <https://gregmaxey.com/word_tip_pages/userforms_advanced_tips.html>

**Cheklist**: Formunuz bittikten sonra genel bir kontrol listesine göre eksikleri kontrol etmek güzel bir alışkanlıktır.

* Hizalamalar tamam mı?
* Aynı kümedeki benzer özellikli kontrollerin ölçüleri eşit mi?
* Tab indexler doğru sırada mı?
* Esc tuşuna basılarak formdan çıkılabiliyor mu?
* Form başlığı belirlendi mi?
* Formunuz bir add-in'de kullanılacaksa Add-in'den açılışı test ettiniz mi?

### Çeşitli Örnekler

#### Data Formları

Bu başlık altında bir örnek olmayacak. Birçok yerde bu konu anlatılırken, verilen örneklerde Data Formlarını çok gördüğüm için ben de başlık olarak koydum ama konuyu bir örnekle anlatmak için değil, size bunun için başka bir alternatif önermek için.

Ben bu iş için Access kullanmanızı öneriyorum. Access'in güzelliği, söz konusu datayı gerçek bir veritabanı uygulamasında saklıyor olmasıdır. Bu anlamda Excel'i çok da veritabanı uygulaması gibi kullanmanızı önermiyorum.

Yine de ben illa Excel’i kullanacağım diyenler için Veritabanı Programlama konusunda bir örneğimiz olacak.

#### Kokpit uygulaması

Bu uygulamayı aynen burdaki gibi çalıştırabilmeniz için **githubdan raporlar.zip** dosyasını indirmenizi tavsiye ederim. Ek indikten son içindekileri C: \ sürücüsü altında "raporlar" diye bir klasör oluşturup buraya kopyalayın. Bu ek ile uğraşmak yerine kodlarda gerekli değişiklikleri yaparak da kendi istediğiniz adreslerdeki dosyaların açılmasını sağlayabilirsiniz.

Kokpit dosyasının kendisine de yine **githubdan(Kokpit.xlsm**) ulaşabilirsiniz.

Bu örnek ile departmanınızda/bölümünüzde sık kullanılan dosyalara belli kategoriler aracılığıyla ulaşılmasını sağlayabilecek, kimin ne zaman hangi dosyaya ulaştığının da log kaydını tutmuş olabileceksiniz. Dosya okuma/yazma konusundaki Logger örneğini inceleyerek bu log kaydının nasıl tutulduğunu detaylıca öğrenebilirsiniz.

Ana ekran görüntüsü aşağıdaki gibi olan formumuzda 4 ana sekme bulunuyor. Sekmelerden bazılarında istenilen döneme ait raporun açılmasını sağlayana comboxlar bulunuyor. Ayrıca tüm geçmiş raporların da görüntülenmesini sağlamak için her sekmenin sağında mavi yazılarla yazılmış, üzerine gelindiğinde büyük + işaretine dönen linkler bulunmakta. Örnek olduğu için tüm düğmeler çalışmamaktadır, sadece belli butonlara kod ataması yapılmıştır.



Şimdi kodların üzerinden geçelim:

* Öncelikle, dosya açılır açılmaz çalışacak koda bakalım. Dosya açıldığında, başkalarında açık kalması bazen probleme neden olabildiği için, ilgili kişinin pc'sinde dosyanın gece 00:00’da kapanmasını sağlıyoruz. Sonra Kokpiti kimin ne zaman açtığını kaydedecek log prosedürünü çağırıyoruz. Son olarak da ana formumuzu gösteriyoruz.

Private Sub Workbook\_Open()

Application.OnTime TimeValue("23: 59: 59"), procedure: ="kapat", schedule: =True

Call logkaydı

Anaform.Show vbModeless

End Sub

* Ana form açılır açılmaz çalışacak kodu ise Initialize eventi içine yazıyoruz.
* Küçültme büyütme işlemlerinde kullanmak üzere boy ve üst nokta ölçülerini alıyoruz. Tabi bunlar en tepede global olarak tanımlanan değişkenler olmalı.
* Excel dosyanın kendisi gizli değilse gizliyoruz, ikinci kez açma kapama durumlarında hata almamak için önce gizli olup olmadığını kontrol ediyoruz.
* 2 tane log butonunu sadece sizde(bu örnekte benim pc adım yazılı, siz kendi pc adınızı yazarsınız) açılmasını sağlıyorsunuz. Bu log butonlarında log dosyalarının(txt formatlıdır) içeriğinin aktarıldığı Excel dosyalar açılmaktadır. (Bu örnekte txtden Excel’e alma detayı anlatılmamıştır)
* Sonra da comboboxların ilk değer atamalarını, çeşitli yöntemlerle, yapıyoruz.

Private Sub UserForm\_Initialize()

dHeight = Me.Height

dTop = Me.Top

If Windows("Kokpit.xlsm").Visible Then

Windows("Kokpit.xlsm").Visible = False

End If

'log butonnları benden başkasına görünmesin

If Environ("username") <> "Volki" Then

Me.cmdDetayLog.Visible = False

Me.cmdLogAna.Visible = False

End If

'AddItem ile eleman ekleme

Me.cbYıl.AddItem (Yıl)

Me.cbYıl.AddItem (Yıl - 1)

Me.cbYıl.Text = Yıl 'veya Value

'List ve Array ile eleman ekleme

Me.cbGün.List = Array(1, 2, 3)

Me.cbGün.Value = 1 'veya Text

'düngüsel olarak 12 ayı doldurma

For i = 1 To 12

'Me.cbAy.AddItem i 'bölgesel ayarlarda tarih formatının durumuna göre burası veya aşağısı

Me.cbAy.AddItem IIf(i < 10, "0" & i, i)

Next i

Me.cbAy.Value = "01"

End Sub

Rapor açan düğmelerdeki kodlardan birine örnek aşağıdaki gibidir. Burada önce detay rapor loguna baz teşkil edecek işlemler yapılıyor, sonra, açılacak dosyanın oluşuş oluşmadığı kontrol edildikten sonra rapor açılmaya çalışılıyor. Dosya henüz oluşmadıysa bir uyarı veriliyor. Dosya oluşmasını kontrol eden örneğin detayını buradan inceleyebilirsiniz. (NOT: Benim, kurumumda yaptığım gibi tam otomatik işleyen bir sistemde, ilgili raporlar uygun zamanı bekleyip kendileri çalışır, kendileri uygun yere kaydolur ve ilgili kullanıcılara maille 'raporçıktı' bilgilendirmesi yapılır. O yüzden bu tür bir kontrolün yapılması anlamsız olabilir, ama fazla kontrol göz çıkarmaz desturuyla hareket edelim ve kontrolümüzü yapalım)

Private Sub CommandButton25\_Click()

On Error GoTo hata

rapor = "İşbirimi\_Hacimsel\_Gelişim"

frekans = "Aylık"

Call detayraporlogu(rapor, frekans)

dosya = aylıkyol & Me.cbYıl.Value & "\İşbirimi Hacimsel Gelişim Raporu.xlsx"

If dosyavarmı(dosya) Then

Workbooks.Open Filename: =dosya, ReadOnly: =True

Else

MsgBox "Dosya henüz oluşmamış, Volkanla görüşün"

End If

Exit Sub

hata:

MsgBox "Bi sorun oluştu, Volkanla görüşün"

End Sub

Tüm rapor arşivini gösteren kodumuz aşağıdaki gibidir

Private Sub Label10\_Click()

On Error GoTo hata

Shell "explorer.exe" & " " & günlükyol & "Günsonu Bakiyeler", vbMaximizedFocus

Exit Sub

hata:

MsgBox "Bi sorun oluştu, Volkanla görüşün"

End Sub

Aşağıdaki kodlar ise sırayla, bir access dosyası, bir internet linki ve bir word dosyasını açan düğmelerin kodları bulunmakta

Private Sub CommandButton29\_Click()

On Error GoTo hata

On Error Resume Next

Set ac = GetObject(, "Access.Application")

If ac Is Nothing Then

Set ac = GetObject(, "Access.Application")

ac.opencurrentdatabase "C: \raporlar\hedefler.accdb"

ac.UserControl = True

Set ac = Nothing

End If

Exit Sub

hata:

MsgBox "Bi sorun oluştu, Volkanla görüşün"

End Sub

--------------------------------------------------

Private Sub CommandButton30\_Click()

Shell ("Explorer http: //www.Excelinefendisi.com/Excelent/KullanimKilavuzu.pdf")

End Sub

--------------------------------------------------

Private Sub CommandButton31\_Click()

On Error GoTo hata

Set wordapp = CreateObject("Word.Application")

Set wordDoc = wordapp.documents.Open("C: \raporlar\satış tanımları.docx")

wordapp.Visible = True

Exit Sub

hata:

MsgBox "Bi sorun oluştu, Volkanla görüşün"

End Sub

Formu büyütüp/küçültme işlemi aşağıdaki kodla yapılır.

Private Sub ToggleButton1\_Click()

If Me.ToggleButton1.Value = True Then

Me.Height = dHeight \* 0.1

Me.Top = 0

Me.ToggleButton1.Caption = "Büyüt"

Else

Me.Height = dHeight

Me.Top = 150

Me.ToggleButton1.Caption = "Küçült"

End If

End Sub

Son olarak form kapanırken, dosyayı da kapatıyoruz, kapanırken kaydolmasın istiyoruz ve dosya açılırken schedule ettiğimiz kapat makrosunu devreden çıkarıyoruz.

Private Sub UserForm\_Terminate()

Application.OnTime TimeValue("23: 59: 59"), procedure: ="kapat", schedule: =False

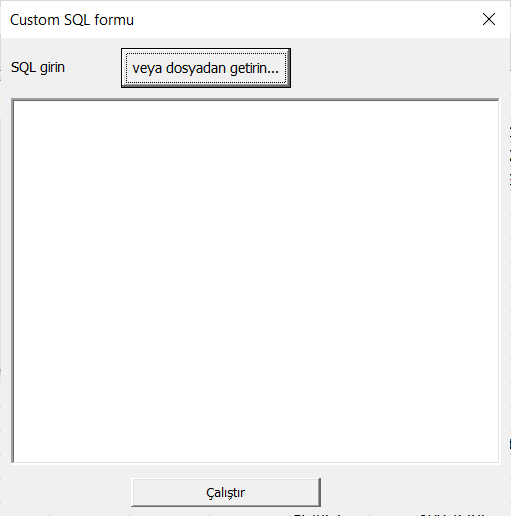
Windows("Kokpit.xlsm").Close savechanges: =False

End Sub

#### Otomatik mail gönderme

Otomatik mail gönderme işlemi Outlook nesne modelini de bilmeyi gerektirdiği için onunla ilgili örneği Outlook Programlama altında bulabilirsiniz.

#### SQL Çalıştırma formu



Bu form ile Toad, AQT, SQL Developer gibi araçlardan çektiğiniz büyük dataları Excel'e yapıştırma zahmetinden kurtulmuş olursunuz, zira bununla, istediğiniz sonuç doğrudan Excel’in içine yerleşir.

Bunun için Veritabanlarıyla ilgili bölümde anlatılan konuları bilmeniz gerekiyor. Bu örneği normalde oraya koymam gerekirdi, ancak userformlarla neler yapılabileceğine ait güzel bir örnek olduğu için buraya koydum.

İlk yapmamız gereken, formu açan bir kod yazmaktır. Aşağıdaki bu mini kodu ya bir add-indeki düğmeye ya da QAT üzerine yerleştireceğimiz bir düğmeye atarız. Siz şimdilik personal.xlsb dosyasında bir modüle koyarak da ilerleyebilirsiniz.

Sub adosql()

frmSQL.Show

End Sub

Sonrasında ise formumuz açılır ve Çalıştır butonundaki kodumuz aşağıdaki gibidir. Aşağıda commentlerde belirtildiği gibi, eğer bağlandığımız veritabanı Oracle veya DB2 gibi şifre kullanımı zorunlu olan bir database ise şifre değişkenini kullanmanız gerekir, ve connection stringinizi de buna göre ayarlamanız gerekir, bunlara ait bilgiler Veritabanı programlama sayfasında bulunuyor. Ancak biz şuan Access gibi şifre zorunluluğu olmayan bir veritabanına bağlandığımız için şimdilik bu değişkeni commentle pasif hale getirdik.

Private Sub CommandButton2\_Click()

'önce tools>references'tan microsoft ado 6.1 seçilmeli

Dim con As New ADODB.Connection

Dim rs As New ADODB.Recordset

Dim strDB As Stream

Dim strSQL As String

Dim constr As String

'Static şifre As String 'her çalıştırma sırasında sormasın diye, eğer şifreniz yoksa uncommentli kalsın, şifreyle ulaştığınız bir database sözkonusuysa comment işaretini kaldırın

On Error GoTo hata

Me.Hide 'formu gizliyoruz

strSQL = frmSQL.TextBox1.Text

If strSQL = "" Then Exit Sub

'şifreli bir veritabanı sözkonusuya aşağıdaki commentleri kaldırın

'If şifre = "" Then

' şifre = InputBox("Şifrenizi giriniz")

'End If

cevap = MsgBox("yeni dosya mı olacak", vbYesNoCancel)

constr = "Provider = Microsoft.ACE.OLEDB.12.0; data source=C: \Users\Volki\Documents\My Web Sites\mysite\Ornek\_dosyalar\Makrolar\VBAusrformsql.accdb"

con.Open ConnectionString: =constr

Application.ScreenUpdating = False

rs.Open Source: =strSQL, ActiveConnection: =con, CursorType: =adOpenKeyset, LockType: =adLockOptimistic

rs.MoveFirst

If cevap = vbYes Then

Workbooks.Add

End If

'önce başlıklar

For i = 0 To rs.Fields.Count - 1

ActiveCell.Offset(0, i).Value = rs.Fields(i).Name

Next i

'şimdi datayı yapıştıralım

ActiveCell.Offset(1, 0).Select

ActiveCell.CopyFromRecordset rs

'burdan sonrasında isterseniz özel tablo formatları da uygulayabilirsiniz

rs.Close

con.Close

Set rs = Nothing

Set con = Nothing

Unload frmSQL 'formu bellekten siliyoruz

Application.ScreenUpdating = True

Exit Sub

hata:

MsgBox Err.Description

Application.ScreenUpdating = True

End Sub

Bu arada kodu elle yazmak yerine hazır kaydedilmiş bir sql dosyasından da getirebilirsiniz, bunun için formdaki ilgili düğmedeki koda atanan kod ise aşağıdaki gibidir.

Private Sub CommandButton1\_Click()

Dim fd As FileDialog

Dim fso As New FileSystemObject

Dim ts As TextStream

Set fd = Application.FileDialog(msoFileDialogFilePicker)

If fd.Show = 0 Then

Exit Sub

End If

Set ts = fso.OpenTextFile(fd.SelectedItems(1))

içerik = ts.ReadAll

ts.Close

Set ts = Nothing

Set fso = Nothing

Me.TextBox1.Text = içerik

End Sub

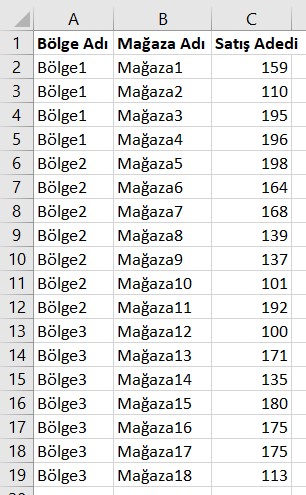
#### Dosya Bölme formu

Bu form, çalıştığım kurumda şuana kadar en çok rağbet gören, **Dosya Bölme** makromu içeren formdur. Aslında favori olma konusunda buna eşlik eden bir de otomatik mail gönderme formu var, ki buna da yukarıda yer verdim. İşte bu meşhur toplu mail gönderme işleminde parametrik ek kullanımı da olacaksa bu makro ile bu ekleri parçalama işlemi yapılmaktadır.

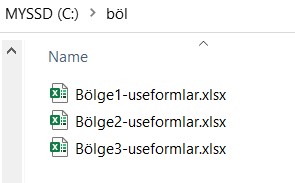
Bölme işleminde temel olarak Dictionary kullanma yoluna gittim. Bunun ilk halinde dictionary kullanmıyordum ve büyük dosyaları bölme işlemi uzun sürüyordu. Sonradan kodu elden geçirip bu hale getirdim.

Evet, şimdi kodları incelemeye başlayabiliriz.

Diyelim ki, elimizde aşağıdaki gibi bir liste var. Her bir bölge için ayrı dosya oluşturmak istiyoruz.



Hedef olarak görmek istediğimiz şey şöyle:



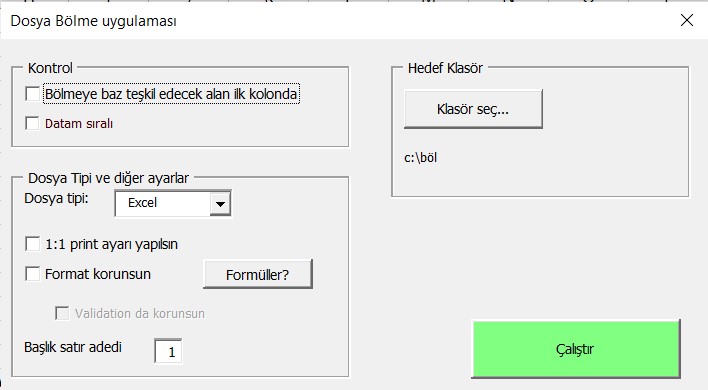
Bölme formumuzu açmak için, ya bir Add-in'deki düğmeye ya da QAT üzerindeki bir düğmeye aşağıdaki kodu atarız. Siz şimdilik personal.xlsb üzerinden veya örnek dosya üzerine gelip, doğrudan forma gelip F5 tuşuna basarak da formu aktive edebilirsiniz.

Sub BölmeAç()

frmBöl.Show

End Sub

Aşağıdaki gibi formumuz açılır.



Bu kontrollere verdiğim isimleri tek tek burda yazmama gerek yok, kod içinden kendiniz de bakabilirsiniz.

Öncelikle form içindeki kodlara bakalım, sonrasında ana kodun bulunduğu modül koduna bakacağız.

İlk olarak Initialize event koduna bakıyoruz. Burada comboların içeriği dolduruluyor ve bir tanesi gizleniyor.

Private Sub UserForm\_Initialize()

Me.cbDosyatip.List = Array("Excel", "PDF")

Me.cbDosyatip.Value = "Excel"

Me.cbPrint.List = Array("Landscape", "Portrait")

Me.cbPrint.Value = "Landscape"

Me.cbPrint.Visible = False

End Sub

Format korunsun checkbox'ına tıklandığında, tick konmuşsa Dosyatip comboboxında seçilen değere göre bir mesaj çıkmakta, bu mesaj her checkbox tıklanışında çıkmasın diye static değişkenle kontrol edilmektedir, ayrıca yine tick konması durumunda Validation checkbox'ı da aktif hale getirilmekte, tick kaldırılınca tekrar pasif olmaktadır.

Private Sub chkFormat\_AfterUpdate()

Static i As Integer 'bu chechkbox her değiştiğinde sürekli bu msgbox çıkmasın diye, bir kere uyarması yeterli

If Me.chkFormat.Value = True Then

If i = 0 Then

If Me.cbDosyatip.Value = "Excel" Then 'formatlı olsa bile pdf hızlı çalışır

MsgBox "Dosya tipi Excel seçildiğinde, format korunursa işlem daha uzun sürecektir." & vbCrLf & \_

"Süre önemliise ya dosya tipini PDF seçin ya da işlemi formatsız yapın"

End If

i = i + 1

End If

Me.chkValidation.Enabled = True

Else

Me.chkValidation.Enabled = False

End If

End Sub

Print checkbox'ı seçildiğinde ise print layoutunun gösterildiği combobox gösterilmekte, seçim kaldırıldığında tekrar gizlenmektedir.

Private Sub chkPrint\_AfterUpdate()

Me.cbPrint.Visible = Me.chkPrint.Value

End Sub

Aşağıdaki kod ise Çalıştır düğmesindeki kod olup, ana kod için ön hazırlık yapmakta ve en son çeşitli parametrelerle ana kodu çağırmakta. Burda iki kontrol bulunuyor. Formun sol üst köşesindeki iki işlemin yapılmış ve bu chekboxların da işaretlenmiş olması lazım, aksi halde bir mesaj gösterilmekte ve kodun çalışması durmaktadır.

Private Sub CommandButton1\_Click()

On Error GoTo hata

Dim printayar As String

'kontroller

If Me.chkKontrolilkkolon.Value = False Then

MsgBox "bölmeye baz teşkil edecek kolon ilk kolonda olmalı." & vbCrLf & \_

"Eğer durum gerçekten böyleyse 'Kontrol' çerçevesi içindeki ilgili checkboxı işaretleyin"

Exit Sub

End If

If Me.chkKontrolSıralı.Value = False Then

MsgBox "Datanız sıralı olmalı. Eğer durum gerçekten böyleyse 'Kontrol' çerçevesi içindeki ilgili checkboxı işaretleyin"

Exit Sub

End If

'böl klasörü yoksa yaratalım

If filefolderexists("C: \böl") = False Then MkDir ("c: \böl")

'A kolonunda / işaretei kontrolü. Zira dosya isimlerinde / işareti olamaz.

On Error Resume Next 'bulamazsa devam etsin diye

Columns("A: A").Select

Selection.Replace what: ="/", replacement: ="-", lookat: =xlPart, \_

searchorder: =xlByRows, MatchCase: =False, searchformat: =False, ReplaceFormat: =False

'başlık satırından sonraki satırda hiç boş hücre olmamalı, space yapalım

Rows(Me.txtBaşlık.Value + 1).Replace what: ="", replacement: =" ", lookat: =xlWhole, \_

searchorder: =xlByRows, MatchCase: =False, searchformat: =False, ReplaceFormat: =False

'hata kontrolünü tekrar getirelim

On Error GoTo hata

If Me.chkPrint.Value = True Then

printayar = Me.cbPrint.Value

End If

Cells(CInt(Me.txtBaşlık.Text), 1).Select

Call filtrekontrol

Application.Wait (Now + TimeValue("00: 00: 02"))

Call bölmekodu(Me.lblKlasör.Caption & "\", printayar, Me.cbDosyatip.Value, Me.chkFormat.Value, CInt(Me.txtBaşlık.Text), Me.chkValidation.Value)

Unload Me

Exit Sub

hata:

MsgBox "bir hata oluştu, volkanla görüşün" & vbCrLf & Err.Description

End Sub

Esas bölmeyi yapan kod biraz uzun olduğu için kitaba koymadım, **githubdan**(CommandButton1\_Click) bakabilirsiniz. Kod içinde yer yer açıklamalar var, ancak ilk etapta F8 ile giderseniz anlaması daha kolay olacaktır.

Aşağıdaki kod ile hedef klasör değiştirilebilmektedir. Default değer [C: \böl](file:///C:\böl) klasörüdür.

Private Sub CommandButton2\_Click()

Dim fd As FileDialog

Set fd = Application.FileDialog(msoFileDialogFolderPicker)

With fd

.Title = "Klasör seçin"

If .Show = True Then

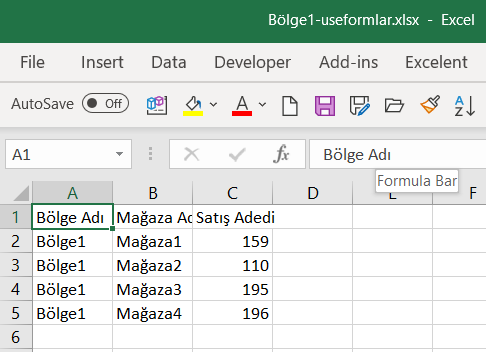
lblKlasör.Caption = .SelectedItems(1)

End If

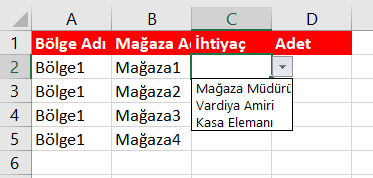
End With

End Sub

Bölme işlemi bittikten sonra kontrol ediyoruz ve gerçekten doğru olarak bölündüğünü görüyoruz.

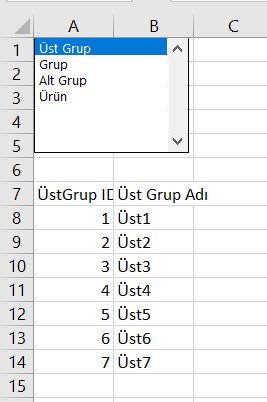


Burda ise, formatın ve validation içeriklerinin korunduğu bir örneği görüyorsunuz.

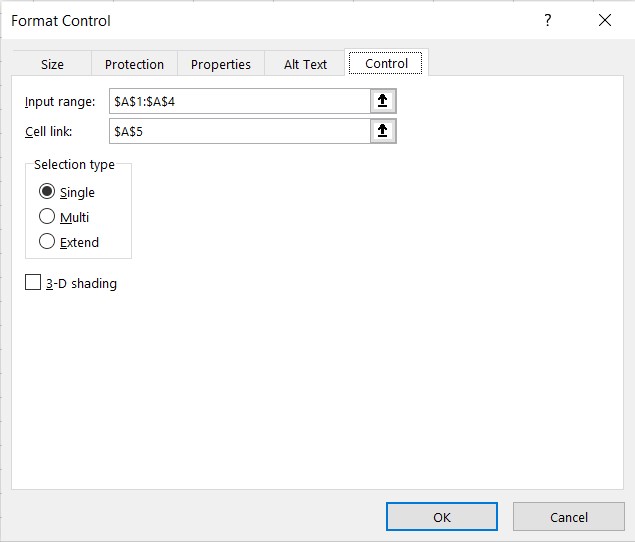


#### Worksheet Formdaki değişime göre bir makronun çalışması

Bu örnekteki form çeşidi her ne kadar worksheet formların konusu olsa da, işin büyük kısmı makro ile yapıldığı için bunu da buraya aldım. Bunun için de biraz veritabanı uygulamalarıyla iletişim bilmek gerekiyor, ancak ben bunu veritabanı konusu yerine bu sefer buraya almayı tercih ettim. Örnek dosyaları **githubdan(worksheet kontrol - VBA.rar)** indirebilirsiniz. Access dosyayı uygun bir klasöre koyup aşağıdaki **constr** değişkenindeki konumunu da değiştirmeniz gerekmektedir.

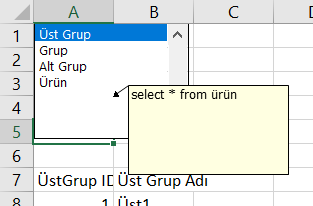


Listbox,'a sağ tıklayıp Control sekmesine geldim ve Input Range ile Cell link özelliklerini aşağıdaki gibi değiştirdim.



A1-A5 arasını tamamen beyaz yaparsanız hiç görünmezler, hatta listbox’ı tamamen A1-A5'i kapatacak şekilde üzerine de taşıyabilirsiniz.

Her hücre içinde comment olarak eklenmiş SQL bulunmakta. Listbox’tan bir ürün seçildiğinde A5'e bu seçimin indeksi gelmekte, buna göre de ilgili SQL çalıştırılmaktadır.



Sub ListBox1\_Change()

Dim con As New ADODB.Connection

Dim rs As New ADODB.Recordset

Dim adet As Integer

Dim constr As String

Dim strsql As String

'Static şifre As String

On Error GoTo hata

strsql = Cells([A5].Value, 1).Comment.Text

If strsql = "" Then Exit Sub

'şifresi olan bir databse ise aşağısı uncommentsiz

'If şifre = "" Then

' şifre = InputBox("şifreyi girin")

'End If

constr = "Provider = Microsoft.ACE.OLEDB.12.0; data source=C: \falanfilanklasör\VBAusrformsql.accdb"

con.Open ConnectionString: =constr

rs.Open Source: =strsql, ActiveConnection: =con, CursorType: =adOpenKeyset, LockType: =adLockOptimistic

rs.MoveFirst

[a7].Select

Selection.CurrentRegion.ClearContents 'bir önceki run sonucunu temizleyelim

'önce başlıkları getirelim

For i = 0 To rs.Fields.Count - 1

ActiveCell.Offset(0, i).Value = rs.Fields(i).Name

Next i

ActiveCell.Offset(1, 0).Select

'şimdi datayı alalım

ActiveCell.CopyFromRecordset rs

rs.Close

con.Close

Set rs = Nothing

Set con = Nothing

Exit Sub

hata:

MsgBox Err.Description

Set rs = Nothing

Set con = Nothing

End Sub

## Neler Öğrendik

Kullanıcılarla MsgBox ve InputBox’a göre daha şık bir araçla iletişime geçmek istediğimizde Userformları nasıl kullanacağımızı öğrendik.

Bir form açıldığında Initialize eventi ile başlangıç işlerini yapmayı, keza kapanırken de kapanış işlemlerini yapmayı gördük.

Userformlar üzerindeki kontrollerin önemli özelliklerine baktık. Kullanımı çok yaygın olmayan özellikleri ise sizlere bıraktık.