# 9.bölüm

**9**

# Olaylar (Eventler)

Kodların nasıl çalıştırılacağı ile ilgili bölümde görmüştük ki, çoğu zaman kodları F5 tuşu ile manuel çalıştırırız. Ancak kodların kendiliğinden çalışmasını sağlamanın da bir yolu var. İşte olaylar, bu kendinden çalıştırma işlemini yapmaktadır.

Olaylar konusuna ben çok önem veriyorum. Bu konuyu ve Application.OnTime konusunu iyi anladığınızda işyerinizde 2-3 ek kişi çalışıyormuş gibi bir verimlilik sağlarsınız.

## Olaylara Genel Bakış

Eventlerle, biz Excel’e "şu olduğunda şu kodu çalıştır" demiş oluruz. Burdaki "şu kod" dediğimiz prosedürlere **Event Handler** prosedürleri denir. Bu prosedürleri yazmak temel olarak normal bir prosedür yazmaya benzer fakat iki farklı yönü vardır.

Bunlar, kendileriyle ilişkili olan nesnenin modülün(Workbook, worksheet) içine yazılır, yani standart bir modül içine yazılmazlar. Bu kuralın 3 istisnası vardır

* Standart bir modül içine konan nesne olmayan olaylar(**OnTime** ve **OnKey**)
* Application eventleri
* Class modüllerine yazılan Chart(sayfa içindeki gömülü olanlardan) eventleri

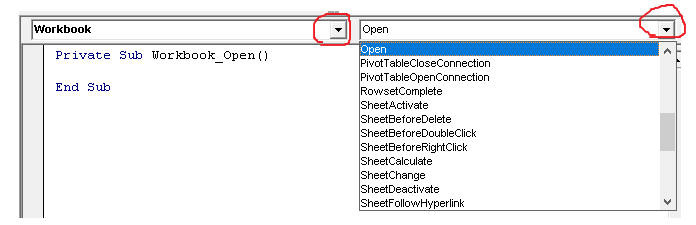
Özel yazım syntaxları vardır.

* Nesne adını takiben "\_" işareti
* Eventin adı
* (Varsa argümanlar)
* Ör: Workbook\_Open(), Worksheet\_Change(ByVal Target As Range)

Tabi bunların özel yazım syntaxı var diyorum ama genelde bunları elle yazmayız. Mesela Projects penceresinde **ThisWorkbook** seçiliyken(çift tıkla seçilmesi gerekir) aşağıdaki resimde göründüğü gibi soldaki nesneler ComboBox'ına tıklanıp **General** olan seçimi,



aşağıdaki gibi Workbook yapınca otomatikman aşağıdaki prosedür oluşacaktır. Sağdaki prosedür ComboBoxına tıklandığında da diğer eventleri görebilirsiniz.



### Kategoriler

Olayları 8 kategoriye ayırabiliriz.

* Workbook olayları
* Worksheet olayları
* Chart olayları
* UserForm olayları
* Worksheet Kontrol olayları(Form ve ActiveX)
* Application olayları
* Özel olaylar
* Belli bir nesneyle ilişkili olmayan olaylar(Bunlar Application.OnTime ve Application.OnKey gibi belirli bir zamanda veya bir tuşa basıldığında meydana gelen olaylardır. İkisi için de Application nesnesiyle ilgili sayfaya bakabilirsiniz)

## Workbook Olayları(Eventleri)

Söz konusu Workbook nesnesi olduğunda, en temel eventler "açılma, kapanma, kaydolma" üçlüsüdür. bunun dışında pek tabiki başka olaylar da bulunmaktadır. Ben burada temelini vermeye çalışacağım. Gerisi kurcalama iştahınıza kalmış.

### Giriş

#### Dosya açılması

Bir dosya açıldığında **Workbook\_Open** olayı devreye girer. Bu event ile kullanıcıya çeşitli mesajlar verebileceğiniz gibi, log kaydı oluşturma, veritabanı işlemleri gibi başka birçok işlem yapabilirsiniz.

Private Sub Workbook\_Open()

MsgBox "x dosyasına hoşgeldiniz. Dosyayı kullanırken şunlara dikkat edin" & vbCrLf & "falan filan" & vbCrLf & "falan filan"

Logger Me.Name, Environ("username"), Date 'log kaydı tutuyoruz, kim ne zaman girmiş diye

End Sub

##### Olay sırası

Bazı olaylar meydana geldiğinde birden çok event tetiklenebilir. Mesela bir dosya açıldığında sadece **Open** eventi devreye girmez, aynı zamanda **Activate** eventi de devreye girer. Bunlardaki sıra önce **Workbook\_Open** sonra **Workbook\_Activate** olacak şekildedir.

##### Auto\_open

Workbook\_Open makrosuna benzer bir de Auto\_open makrosu vardır. Bundan **Dört Temel Nesne>Workbook** konusu içinde bahsetmiştik.

#### Kapanma ve Kaydetme

Bir dosya kapanırken de önce **Workbook\_BeforeClose**, sonra **Workbook\_Deactivate** eventi devreye girer. Kapatma sırasında kaydetme de olacaksa süreç şöyle olur:

Workbook\_BeforeClose

Workbook\_BeforeSave

Workbook\_AfterSave

Workbook\_Deactivate

Private Sub Workbook\_BeforeClose(Cancel As Boolean)

MsgBox "Dosyadan ayrılıyorsunuz. ….. yapmayı unutmayın."

End Sub

##### İptal parametresi

Kapanma ve Kaydetme olayları öncesinde bazen bu olayın geçekleşip gerçekleşmeyeceğini de kontrol altına almak isteyebiliriz. Bunu da varsayılan değeri False olan **Cancel** parametresine kod içinde True değeri atayarak yaparız.

Mesela kaydetme işlemini, sadece belli bir şifreyi bilen kişilere yaptırmak isteyebilirsiniz. Bunun için aşağıdaki gibi bir kod yazılabilir.

Private Sub Workbook\_BeforeSave(ByVal SaveAsUI As Boolean, Cancel As Boolean)

Dim şifre As String

şifre = InputBox("Kaydetmek için yetkili şifresini girin")

If şifre <> "1234" Then

MsgBox "Dosyayı kaydetmeye yetkili değilsiniz."

Cancel = True

Else

MsgBox "dosya başarılı bir şeklide kaydedildi" 'belki bir de hata kontrolü konulabilir buraya

End If

End Sub

##### Kapanma/Deactive olma durumuna göre farklılaşan mesaj gösterme örneği

Aşağıdaki örnekte ise dosya kapanırken farklı, deaktive olurken farklı bir mesaj verme örneği var.

Private Sub Workbook\_BeforeClose(Cancel As Boolean)

Range("Z1000").Value = 1 'hangi hücre müsaitse

Me.Saved = True

End Sub

Private Sub Workbook\_Deactivate()

If Range("Z1000").Value = 1 Then

MsgBox "çıkma mesajı"

Else

MsgBox "deaktive mesajı"

End If

End Sub

Burdaki süreç şöyle işleyecek. Diyelim ki kullanıcı dosya açıldıktan bir süre sonra başka bir dosyaya geçmek istedi, deactivate olayı ilk kez devreye girer, Z1000 hücresinde 1 yazıyor mu diye bakar, yazmadığı için sadece "deaktive mesajını" verir. Sonra diyelim ki yine aktif oldu, bir süre sonra tekrar başka dosyaya geçtiğinde bu süreç aynen tekrar eder. Ne zaman ki dosyadan çıkmak isterse, önce BeforeClose devreye girer ve Z1000'e 1 değerini atar, sora Deactivate olayı devreye girer, Z1000=1 olduğu için sadece çıkma mesajı verilir. Burada BeforeClose içine hiçbir mesaj yazmadık, böyle yapsaydık hem "Çıkma mesajı" hem de "Deactive mesajı" verilmiş olurdu, bu da arka arkaya 2 farklı mesaj kutusu demek olurdu ki pek şık bir durum olmazdı. Burada tabi ben sadece mesaj farklılaşması yaptım, sizin ihtiyacınıza göre farklı işlemler de yaptırılabilir.

##### Auto\_close

Auto\_open için yazılan bilgilerin benzeri aynı mantıkta Auto\_close için de geçerlidir.

Kapanmadan önce sayfayı default ayarlara getirmek

Başlangıç ayarları özellikle belli değerlere ayarlanmış bir dosyanız olsun. Kullanıcılar, bu dosyada çeşitli oynamalar yaptıktan sonra kapatacaklar diyelim. Dosyayı ilk ayarlarına getirmek için BeforeClose olayı kullanılabilir.

Private Sub Workbook\_BeforeClose(Cancel As Boolean)

Application.EnableEvents = False

Range("B1") = vbNullString

Range("B2") = 400

Range("C21") = "Toplam"

'....

Application.EnableEvents = True

End Sub

Application.EnableEvents kullanmamızın sebebi, sayfada Worksheet\_Change eventi varsa bu tetiklenmesin diyedir. Bu konuda daha detay bilgi için bir sonraki konu olan Worksheet olaylarına bakabilirsiniz.

#### Diğer olaylar

##### Yeni sayfa eklenmesi

Yeni bir sayfa yaratıldığında devreye girer. Bazen birilerine gönderdiğiniz bir dosyada yeni sayfa yaratılmasını istemiyor ve kullanıcıların mevcut sayfa(lar) üzerinden çalışmasını istiyorsanız bu eventi aşağıdaki gibi kullanabilirsiniz.

Private Sub Workbook\_NewSheet(ByVal Sh As Object)

Application.ScreenUpdating = False

MsgBox "yeni sayfa yaratamazsın"

Application.DisplayAlerts = False

Sh.Delete

Application.DisplayAlerts = True

Application.ScreenUpdating = True

End Sub

NOT: Screenupdating'i kullanma sebebi, kullanıcının geçici de olsa sayfanın yaratıldığını görmesini engellemek içindir.

Başka olaylar da bulunuyor, ancak bunları araştırmayı size bırakıyorum.

### Detaylar

#### Event tetiklenmesini bastırmak(Geçici olarak durdurmak)

Her ne kadar workbook eventlerinde kısırdöngüye giren event durumu pek rastlanılan bir durum olmasa da, yine de belirli şartlarda eventleri geçici olarak durdurmak, o özel durum geçince tekrar aktive olmalarını sağlamak isteyebilirsiniz.

Ben mesela QuickAccessToolbar üzerine eventleri toggle buton mantığı ile bir aktif bir pasif yapan bir düğme koydum. Zira sıklıkla schedule edilmiş dosyaları açıp içlerinde güncelleme yapma ihtiyacım oluyor. Normalde bunlar açıldıklarında Workbook\_Open makrolarının otomatikman çalışmaları gerekiyor, ama güncelleme yapacaksam bunların çalışmasını engellemek için geçici olarak eventleri pasifleştiriyorum, işimi bitirince tekrar aktifleştiriyorum.

İlgili düğmeye atadığım kod şöyle:

Sub toggle\_event()

If Application.EnableEvents = True Then

Application.EnableEvents = False

Application.StatusBar = "EnableEvents=False\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*EnableEvents=False\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*EnableEvents=False\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*EnableEvents =False\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

Else

Application.EnableEvents = True

Application.StatusBar = "EnableEvents=True\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*EnableEvents=True\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*EnableEvents=True\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*EnableEv ents=True\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

End If

End Sub

#### [Voltran’ı](https://eksisozluk.com/voltrani-olusturmak--395838) oluşturmak

Workbook\_Open eventini, Application.OnTime, Logger fonksiyonu ve Veritabanı(Connection veya ListObject) refresh işlemleriyle birlikte kullandığınızda Voltran’ı oluşturmuş olursunuz. Bu dörtlü sizi ve kurumunuzu inanılmaz bir verimlilik sürecine sokar. Zira bu şekilde istediğiniz sayıda raporu(yeterli cihazın olması varsayımı altında) otomatiğe bağlayabilirsiniz. Diğer üçüyle ilgili detaylara kendileriyle ilgili konu sayfalarından ulaşabilirsiniz, ben buraya sadece küçük bir örnek koymak istiyorum.

Application.OnTime ile schedule edilmiş bir makro düşünün. Bu, Voltran’ın 1.üyesinin devreye girdiği andır. Bu makro çalıştığında(ve çalışırken kod boyunca önemli yerlerde) Log kaydı oluşturacaktır: 2.üye de tamam. Bu makro ile belirli bir Excel dosya açılmaktadır ve devreye bu dosyanın Workbook\_Open eventi girer, işte 3.üye. Son olarak da bu makro içinden çeşitli veritabanından data çekme işlemi yapılarak 4.üye de devreye sokulmuş olur. Son 2 üye aşağıda gösterilmiştir.

Private Sub Workbook\_Open()

'Önceki kodlar

'Çeşitli ön kontroller(kullanıcı, pc adı v.s gbi) burada yapılabilir

For Each cn In ActiveWorkbook.Connections

cn.ODBCConnection.BackgroundQuery = False

cn.Refresh

Next

'Sonraki kodlar

'dosyayı kaydetme ve alıcılara mail gönderme kodları burada yer alabilir

End Sub

#### Kısıtlama uygulama ve Veri Güvenliği sağlama

Diyelim ki gizli bilgiler içeren bir dosyayı birine gönderdiniz, veya genel kullanım için bir add-in yazdınız ve bu add-in gizli/hassas bilgiler içeren bir veritabanı doyasından belli bir sorgu çalıştırıp getiriyor. Her neyse, bu bilgilerin kullanıcılar tarafından yazdırılmasını, copy/paste ile başka dosyaya kopyalanıp orada da yazdırılmasını engellemek istiyorsunuz ve herhangi bir şekilde mail olarak gönderilmesini de engellemek istiyorsunuz. Aşağıda, bunlara ait çözümleri bulabilirsiniz.

Sayfanın yazdırılmasını engelleme

Bunu aşağıdaki basit kod ile yapmak mümkündür.

Private Sub Workbook\_BeforePrint(Cancel As Boolean)

Cancel = True

MsgBox "Bu dosyayı print almanıza izin verilmemiştir."

End Sub

Eğer belli bir kişinin/kişilerin print alma yetkisi olsun isterseniz belli bir sicili/sicilleri kontrol edebilirsiniz. Aşağıda tek sicil numarasının kontrol örneği var, siz isterseniz bir diziye(veya collection) tüm yetkilileri atıp o dizi içinde var mı diye de kontrol edebilirsiniz.(Username'in sicil döndürdüğü varsayımı ile hareket edilmiştir. Sizin kurumda isim de kullanılıyor olabilir, o zaman daha farklı bir yöntem denemeniz gerekebilir. Sadece özel bir şifreyi bilenlerin kaydetmeye yetkisi olması gibi)

Private Sub Workbook\_BeforePrint(Cancel As Boolean)

If Environ("username")<>12345 Then

Cancel = True

MsgBox "Bu dosyayı print almanıza izin verilmemiştir."

End If

End Sub

##### Cut/Copy engelleme

Cut/Copy işlemlerini engellemek için biraz daha uğraşmamız gerekiyor. Öncelikle yasaklama işlemini hem Workboook\_Open hem de Workboook\_Activate eventlerinde kullanmamız gerekiyor. Ayrıca kullanıcı başka bir dosyayı açtığında o dosyada yasakların kalkıp normale dönmesi gerektiği için de işlemlerin tersinin Workboook\_Deactivate eventinde yapılması gerekiyor.

Aşağıdaki örnek kullanıcıya bir de mesaj veriyoruz. Mesaj vermek istemiyorsak aşağıda "cutcopyengel" olan her şeyi "" olarak değiştirmeniz yeterlidir.

Private Sub Workbook\_Open()

With Application

.CutCopyMode = False

.OnKey "^c", "cutcopyengel" 'Ctrl+C ile copy

.OnKey "^x", "cutcopyengel" 'Ctrl+X ile cut

.OnKey "^{INSERT}", "cutcopyengel" 'Ctrl+Insert ile copy

.OnKey "+{DELETE}", "cutcopyengel" 'Shift+Delete ile cut

.OnKey "+{DEL}", "cutcopyengel" 'Shift+Del ile cut

'mousela taşıma iptali

.CellDragAndDrop = False

'sağ tıklama ile engel

.CommandBars("Cell").Controls(1).Enabled = False 'cut

.CommandBars("Cell").Controls(2).Enabled = False 'copy

End With

End Sub

Private Sub Workbook\_Activate()

With Application

.CutCopyMode = False

.OnKey "^c", "cutcopyengel" 'Ctrl+C ile copy

.OnKey "^x", "cutcopyengel" 'Ctrl+X ile cut

.OnKey "^{INSERT}", "cutcopyengel" 'Ctrl+Insert ile copy

.OnKey "+{DELETE}", "cutcopyengel" 'Shift+Delete ile cut

.OnKey "+{DEL}", "cutcopyengel" 'Shift+Del ile cut

'mousela taşıma iptali

.CellDragAndDrop = False

'sağ tıklama ile engel

.CommandBars("Cell").Controls(1).Enabled = False 'cut

.CommandBars("Cell").Controls(2).Enabled = False 'copy

End With

End Sub

'Deactivate eventinde OnKey'in ikinci parametresini boş bırakarak

'işlemi tersine çeviriyoruz

Private Sub Workbook\_Deactivate()

With Application

.CutCopyMode = True

.CellDragAndDrop = True

.OnKey "^c"

.OnKey "^x"

.OnKey "^{INSERT}"

.OnKey "+{DELETE}"

.OnKey "+{DEL}"

.CommandBars("Cell").Controls(1).Enabled = True

.CommandBars("Cell").Controls(2).Enabled = True

End With

End Sub

'Bu da ayrı bir modül içindeki kodumuz

Sub cutcopyengel()

MsgBox "Bu dosyada cut/copy yapamazsınız"

End Sub

##### Mail göndermeyi engelleme

Dosyayı mail ile göndermeyi engellemek için iki iş yapmak lazım.

* Excel içinden File menüsündeki Send As Attachment düğmesini kullanmayı engellemek
* Dosyayı kaydedip Windows Explorer üzerinden veya Outlook ile göndermeyi engellemek

1.yöntemde Ribbonu özelleştirmek gerekiyor, aynı zamanda ilgili mail gönderme butonu QAT(QuickAccessToolbar) üzerinde de olabilir diye QAT'ı göstermeyi de engellemek lazım. Bunları İleri Seviye konular bölümünde göreceğiz.

2.yöntemde ise dosyanın kaydolmasını engellemeliyiz.

Private Sub Workbook\_BeforeSave(ByVal SaveAsUI As Boolean, Cancel As Boolean)

Cancel = True

MsgBox "Bu dosyayı kaydedemezsiniz"

End Sub

Tabi yine belli kişiler için kaydetme izni vermek isterseniz Environ("username") özelliğini kullanabilirsiniz.

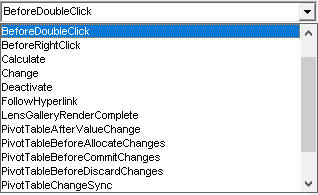
#### Tüm sayfaları ilgilendiren olaylar

Her ne kadar tüm sayfaları ilgilendiren olaylar Workbook eventleri olsa da onları burada değil Worksheet olayları içinde anlatmayı uygun buldum. Bu sayfaya alttaki İleri butonuna tıklayarak ulaşabilirsiniz.

## Worksheet Olayları(Eventleri)

### Giriş

Bir workbook'un sayfalarındaki çeşitli olaylara tepki vermek adına devreye giren olaylar Worksheet olayları olarak adlandırılır. Bunları da yine Worbook olaylarını seçer gibi seçip içlerini doldurmaya başlayabilirsiniz. İlgili combobox seçildiğinde aşağıdaki gibi eventlerin bir kısmı görünür.



Bunlardan en sık kullanacaklarımız;

* Change
* SheetChange
* BeforeDoubleClick
* Activate/Deactivate
* Calculate

eventleridir.

Pivot tablolarla ilgili olanlar da önemli olup bunlara **İleri Seviye Konular>Pivot İşlemleri** konusunda değineceğiz. Şimdi sırayla önemli eventlere bakalım.

### Temel olaylar

#### Worksheet\_Change Event

Kuşkusuz en önemli sayfa olayı sayfada bir hücrenin değişimiyle meydana gelen **Change** olayıdır. (Bu eventin adını AfterChange gibi düşünmeniz yerinde olur. Zira olay, hücre içi değiştikten sonra meydana gelir. Microsoft geliştiricileri olayın adını keşke böyle yapsalarmış. Ne de olsa After ve Before ile başlayan bir sürü event var.) Syntax'ı aşağıdaki gibidir.

Private Sub Worksheet\_Change(ByVal Target As Range)

End Sub

Küçük bir örnek yapalım. Bu örnekte, her değişim oldukça sayfanın rengi değişsin. Bu örneği alıp istediğiniz bir dosyanın Sheet1 modülüne yapıştırın ve sonra gidip sayfada rasgele hücrelere birşey girin. Her Enter'a basışınızda sayfa rengi değişecektir.

Private Sub Worksheet\_Change(ByVal Target As Range)

x = WorksheetFunction.RandBetween(1, 1000000)

ActiveSheet.Cells.Interior.Color = x

End Sub

##### Tetikleyiciler ve özel hususlar

Change olayı kullanıcının manuel bir işlemi sonucunda tetiklenebileceği gibi bir makro kodu sonucunda da tetiklenebilir.

Bazı özel durumlar da vardır:

* Manuel hesaplama durumundan otomatik hesaplama durumuna geçildiğinde de hücrelerin içi değişir ama bu durum Change olayını tetiklemez. Yine de yeni duruma göre içerik kontrolü yapacaksanız bu sefer Calculateolayını kullanmanız gerekir.
* Bir hücrenin içini silmek de değişiklik olduğu için **Change** olayı tetiklenir.
* Merge butonu ile hücre birleştirmek tetiklemez.
* Bir alanı sıralamak tetiklemez
* Goal Seek kullanarak bir hücrenin değişimi tetiklemez

##### Target Parametresi

Target parametresi, belli bir hücrenin içeriğini değişip değişmediği öğrenmek amacıyla kullanılabileceği gibi ilgili hedefin tek bir hücre mi yoksa bir range mi olduğunu belirlemek için de kullanılabilir. Aslında Range nesnesinin tüm özelliklerini kontrol etmek için kullanılabilir.

If Target.Address="$A$1" Then 'bu bir adres kontrolüdür

If Target.Cells.Count=1 Then 'bu da tek bir hücre mi yoksa bir range mi kontrolüdür

Target'ın belirli bir aralıkta olup olmadığını öğrenmek için özel bir kullanım şekli vardır: **If Not Intersect(Target, Range("..")) Is Nothing Then**

Aşağıdaki örnekte değişen hücrenin C3 veya C4'te olması beklenmektedir. Bununla ilgili daha detaylı örnek biraz ileride Çeşitli Örnekler kısmında yapılacaktır.

Private Sub Worksheet\_Change(ByVal Target As Range)

If Not Intersect(Target, Range("C3: C4")) Is Nothing Then

'ana kod bloğu

Else

MsgBox "Yanlış yerden seçim yapıyorsunuz, sadece C3 ve C4 hücrelerini kullanınız"

End If

End Sub

##### Aynı hücredeki değişimlerde bir önceki değeri elde etme

Değişen hücrenin bir önceki değerini elde etmek istiyorsak **Statik** değişken kullanırız.

Private Sub Worksheet\_Change(ByVal Target As Range)

Static öncekiDeğer As String

Static öncekiAdres As String

If öncekiDeğer <> "" And öncekiAdres = Target.Address Then

MsgBox "Önceki: " & öncekiDeğer

End If

öncekiDeğer = Target.Value

öncekiAdres = Target.Address

MsgBox "yenisi: " & Target.Value

End Sub

Bu örnekte statik değişkenlerimiz ilk başta boş olacaktır, zira henüz "öncesi" yoktur. İlk işlemden sonra önceki statik değişkenler dolmaya başlayacaktır. Akabinde, yeni hücre ile öncekinin aynı olup olmadığı kontrol edilir.

#### Worksheet\_SelectionChange

Seçili hücre her değiştiğinde bu event oluşur.

Private Sub Worksheet\_SelectionChange(ByVal Target As Range)

End Sub

Target, seçilen hücreyi gösterir.

Aşağıdaki örnekte, seçilen hücre pencerenin sol üst köşesindeki ilk hücre olacak şekilde ayarlanır.

Private Sub Worksheet\_SelectionChange(ByVal Target As Range)

With ActiveWindow

.ScrollRow = Target.Row

.ScrollColumn = Target.Column

End With

End Sub

##### Önceki seçimi elde etme

Seçimden bir önceki hücreye de ihtiyacımız olacaksa **Statik** bir değişken kullanırız. İlk seçimde çalışmaz, sonrakilerde çalışır, çünkü ilk seçimde henüz "öncesi" yoktur.

Private Sub Worksheet\_SelectionChange(ByVal Target As Range)

Static öncekiRange As String

If öncekiRange <> "" Then

MsgBox "önceki: " & Range(öncekiRange).Address

End If

öncekiRange = Target.Address

MsgBox "yenisi: " & Target.Address

End Sub

Daha farklı bir örnek ise, önceki hücre ile yeni hücre arasındaki alanı kırmızıya boyamak olabilir. "Ne işimize yarayacak" diye sormayın, bu haliyle bir işinize yaramaz, ama farklı bir konuda size fikir verebilir.

Private Sub Worksheet\_SelectionChange(ByVal Target As Range)

Static öncekiRange As Range

If Not öncekiRange Is Nothing Then

Range(öncekiRange, Target).Interior.Color = vbRed

End If

Set öncekiRange = Target

End Sub

#### Worksheet\_BeforeDoubleClick

Bir hücreye çift tıklandığında bu olay olur ve Excel’in o anda nasıl davranmasını istiyorsak bu prosedüre bunları yazarız. Syntaxı aşağıdaki gibidir.

Private Sub Worksheet\_BeforeDoubleClick(ByVal Target As Range, Cancel As Boolean)

End Sub

Target'ı şimdiye kadar öğrenmiş olmalısınız; kullanım mantığı yine yukardakilerle aynı. Cancel parametresine ise True değerini atayarak eylemi iptal edebiliriz, yani Excel’e çift tıklama olmamış gibi davrandırtabiliriz.

En sık kullandığım örneklerden birisi, toplanmış verileri tutan bir listede ilgili hücreye çift tıklama sonucunda o grubun alt detayını gösteren verilerin uygun miktarda satır açılarak araya eklenmesi; aynı hücreye tekrar çift tıklanması durumunda ise bu kayıtların animasyonlu bir şekilde silinip listenin ilk hale gelmesidir. Böyle bir örnek kullanımı ADO içermesi sebebiyle bu sayfada vermeyip bunları veritabanı uygulamaları bölümünde ele alıyor olacağım. İlgili örneğe **Veritabanı Programlama** konusunda bakacağız. Aynı örneği ilgili veriyi aynı sayfada gizlenmiş bir şekilde dururken unhide ederek de yapabilirsiniz. Ancak az önce bahsettiğim, Veritabanı Programlama konusundaki örnekteki liste dinamik bir yapıya sahip olduğu için hide etmek bir uygun bir çözüm olmamaktadır.

Başka bir örneği ise burada ele alabiliriz. Bunda da yine gruplu bir liste var. Bu listede bir hücreye çift tıklayınca bu hücreye ait alt veriler ayrı bir dosya olarak açılıyor olsun. Ör: En çok kredi düşüşü yaşayan şube listesinde şube koduna çift tıklayınca bize en çok düşüş yaşayan müşteriler dosyasını açıp bu şubeyi filtrelesin.

Private Sub Worksheet\_BeforeDoubleClick(ByVal Target As Range, Cancel As Boolean)

adres = "C: \...\"

If Not Intersect(Target, Range("B2: B20")) Is Nothing Then

sb = Target.Value

Workbooks.Open adres + "Kredisi en çok düşen müşteriler - " & Date - 1 & " Sonuçları.xlsm", ReadOnly: =True

ActiveSheet.ListObjects("Query\_from\_DWH").Range.AutoFilter Field: =2, Criteria: =sb

End If

End Sub

Şahsen ben bu eylemi çok önemsiyorum. Bununla ADO'yu birleştirerek yüksek ücretli programlara alternatif programlar yazabilirsiniz. ADO kısmında diğer detayları bulabilirsiniz.

#### Worksheet\_Activate/Deactivate

Belli bir sayfa (yeniden) aktif(veya inaktif) olduğunda çalışmasını istediğiniz kodları bu olayla tetiklenen olay prosedürleri içine yazabilirsiniz.

Private Sub Worksheet\_Activate()

End Sub

Örneğin, ana menü sayfası gibi bir sayfanız var ve buna sadece diğer sayfalardaki **Anamenü** linki aracılığı ile ulaşmak istiyorsunuz, ve bu sayfalar açıkken bu menü sayfası görünmesin istiyorsanız, işte bu menü sayfasından ayrılırken sayfanın gizlenmesini sağlayacak bir kod yazabilirsiniz.

Private Sub Worksheet\_Deactivate()

Me.Visible = xlSheetHidden

End Sub

'aşağıdaki kodu da diğer sayfalardaki Selection\_Change eventine yazarsınız

If Target.Value = "Anamenü" Then

Sheets("Anamenü").Visible = xlSheetVisible

Sheets("Anamenü").Select

End If

#### Worksheet\_Calculate

Bu event, sayfadaki formüller yeniden hesaplandığında tetiklenir. Özetle o formülü etkileyen hücrelerden birinde değişiklik olursa tetiklenir. Mesela Bir hücre grubunun altında SUBTOTAL formülü ile toplam/ortalama v.s alınmışsa ve hücre grubundaki filtrede bir değişiklik yapılırsa formülün içeriği de değişeceği için bu event tetiklenir.

Bu eventte hedef bir hücre(Target) bulunmaz, zira tüm hücreler yeniden hesaplanmıştır.

NOT: Sayfa için aynı zamanda Change eventi de varsa kod bloğu içine eventleri geçici olarak bastıran kodları eklemeyi unutmayın.(Bu konuyu hemen aşağıda inceleyeceğiz)

Private Sub Worksheet\_Calculate()

'çeşitli işlemler

End Sub

Bu konuya ait güzel bir örneği **İleri Seviye Konular>Pivot Table ve Slicer** konusunda ele aldığımız Örnek Uygulamada bulabilirsiniz.

### Diğer Hususlar

#### Event tetiklenmesini bastırmak(Geçici olarak durdurmak)

Makronuzda, bir yerlerde ilgili eventi tekrar tetikleyecek bir kod varsa bu kod sonsuz döngüye girer ve Excel çökebilir(veya ayarlarınıza göre 100 civarı iterasyon sonucunda durabilir, bende 78.iterasyonda duruyor). Change eventi içinde bir hücrenin içeriği değiştirilmesi veya SelectionChange eventi içinde başka bir hücre seçilmesi gibi.

Mesela aşağıdaki örneği F8 ile deneyip görün, her F8 yapışınızda kod hiç durmadan bir aşağı inecektir.

Private Sub Worksheet\_SelectionChange(ByVal Target As Range)

Target.Offset(1, 0).Select

End Sub

Aşağıdaki kodda ise sürekli olarak Change olayı kendisini tetikliyor.

Private Sub Worksheet\_Change(ByVal Target As Range)

Target.Offset(1, 0).Value = Target.Row

End Sub

İşte bu tür durumları önlemek için eventin başında **Application.EnableEvents = False** diyerek eventleri geçici olarak askıya alırız, sonra işlemleri yaptırır, sonra da **Application.EnableEvents = True** diyerek evetnleri tekrar devreye sokarız. Tabi olur da kodumuzda bir hata oluşur da sona gelmeden durursa Eventler askıda kalabilir, bu yüzden bir hata yönetimi bloğu yazıp eventleri burda da tekrar aktive etmeliyiz.

Private Sub Worksheet\_Change(ByVal Target As Range)

On Error GoTo hata

Application.EnableEvents = False

'tetiklemeye neden olabilecek işlemler

Application.EnableEvents = True

Exit Sub

hata:

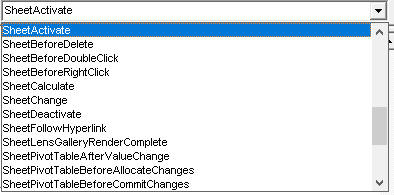
Application.EnableEvents = True

End Sub

#### Workbook'un sheet eventleri

Workbook eventleri workbookla ilgili bir eylem gerçekleşince devreye giriyordu, Worksheet eventleri de sayfayla ilgili bir eylem gerçekleşince. Bir de ikisinin karışımı gibi olan ama aslında bir Workbook eventi olan event grubu var.

Bunların bir listesi aşağıdaki gibi olup, **belli bir sayfada değil de herhangi bir sayfada** bir eylem gerçekleştiğinde tetiklenirler.



Mesela aşağıdaki kod ile hangi sayfa seçilirse onun adı bize MsgBox ile gösterilir.

Private Sub Workbook\_SheetActivate(ByVal Sh As Object)

MsgBox Sh.Name

End Sub

#### Farklı kullanıcılarda eventlerin tetiklendiğinden emin olmak

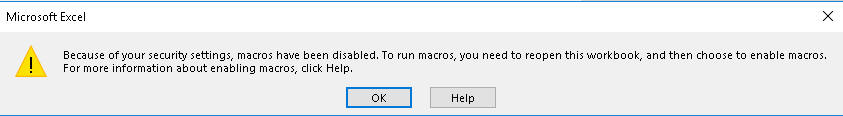
Giriş bölümündeki Güvenlik ayarları bölümünü okumadıysanız öncelikle orayı okumanızı öneririm. Orada belirtildiği gibi makro ayarları **Disable All** şeklindeyse sonuçta bir makro olan Event Prosedürleriniz de devreye girmez.

##### Örnek senaryo

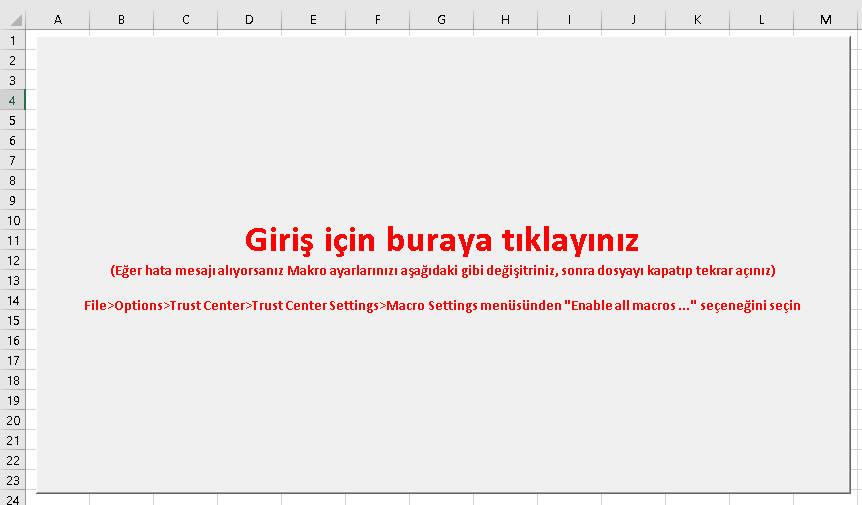
Hazırladığınız bir dosyanın anlamlı olabilmesi için eventlerin çalışması gerekmekte olsun. Ancak kullanıcının makro ayarları Disabled ise kullanıcı dosyadan istenen verimi alamayacaktır, üstelik sizin istemediğiniz şekilde yetkisi olmayan görüntülemeler bile yapabilecektir.(Farklı şubenin rakamlarını görmek gibi)

Bunu engellemek için benim geliştirdiğim yöntem aşağıdaki gibidir(Daha iyi veya daha kötü yöntemler var olabilir, ben araştırdığımda hiçbir şeyle karşılaşmadığım için kendi çözümümü böyle geliştirmiştim)

Çalışmanın tam üstüne denk gelecek şekilde bir düğme koyarım ve bu düğme için bir kod yazarım. Eğer makrolar enable ise düğme kaybolur, makrolar disabled ise aşağıdaki gibi bir hata alır.



Ayrıca düğmeyi silmesin veya başka bir yere taşımasın diye sayfaya protection da koymamız gerekiyor. Makro sırasında dosyayı gizlerken geçici olarak kaldırıyor, gizledikten sonra tekrar koyuyoruz, ki protection'ı başka amaçlar için de kullanabilelim. Buna ait bir örneği **Çeşitli Örnekler** bölümünde 2.örnekte bulabilirsiniz.



Düğmenin Click eventi ise şöyledir.

Sub Button1\_Click()

Sheets(1).Unprotect Password: ="1234"

ActiveSheet.Shapes("Button 1").Visible = msoFalse 'düğmeyi gizler

Sheets(1).Protect Password: ="1234"

End Sub

#### Kısıtlar uygulamak

##### Sayfanın yazdırılmasını engellemek

Diyelim ki kullanıcıların belli sayfaları basmasını istemiyorsunuz. Aşağıdaki kodu ilgili dosyanın Workbook\_BeforePrint eventine yazmanız gerekir.

Private Sub Workbook\_BeforePrint(Cancel As Boolean)

For Each s In ActiveWorkbook.SelectedSheets

If s.Name = "Ham Data" Then

MsgBox ("Bu sayfayı basamazsınız!!!")

Cancel = True

End If

Next

End Sub

Workbook içinde hiçbir sayfanın bastırılmasını istemiyorsanız bu sefer hiç safya kontrolü yapmadan doğrudan MsgBox ve Cancel=True satırları yeterli olacaktır.

Gördüğünüz gibi bu işlemi bir worksheet eventi ile değil workbook eventi ile yapıyoruz.

##### Sayfada cut/copy engellemek

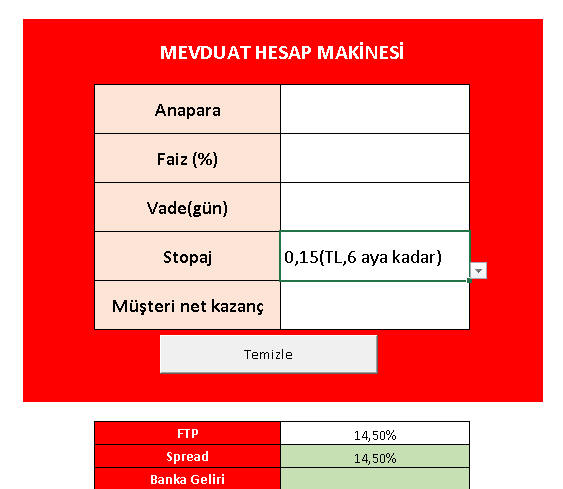
Bu işlemin tüm dosya bazında yapılmasıyla ilgili örnek önceki konuda ele alınmış olup, sayfa bazında yapmak için Worksheet\_Activate ve Worksheet\_Deactivate olaylarında kullanılması yeterlidir.

### Çeşitli Örnekler

#### Mevduat fiyatlama hesap makinası (Animasyonlu)

Bu örnekte, 4 parametreden oluşan bir denklemin herhangi 3'ü bilinirken diğer 4.sünün tespit edilmesine yönelik bir kod yazacağız. Klasik Excel yöntemiyle yapmak istediğinizde 4 ayrı çalışma yapmanız gerekirken VBA ile tek bir format ile tüm senaryoları ele alabileceğiz.

Bunun için aşağıdaki gibi bir form hazırladım. Dosyanın kendisine **github(mevduat makinesi.xlsm)** üzerinden ulaşabilirsiniz.



##### Çalışmaya ait kodlar şöyle:

Önce Sheet1 modülü:

Private Sub Worksheet\_Change(ByVal Target As Range)

On Error GoTo çıkış

Application.EnableEvents = False

If Target = [stopaj] And IsEmpty(Target) Then

Target.Value = "0,15 (TL, 6 aya kadar)"

End If

If Not Intersect(Target, [alan]) Is Nothing Then

ActiveSheet.Unprotect 1234

Call temizlik([alan])

If [alan].Cells.SpecialCells(xlCellTypeBlanks).Count = 1 Then [alan].SpecialCells(xlCellTypeBlanks).Select

Select Case ActiveCell

Case [anapara]

ActiveCell.Formula = "=365\*NetGetiri/(Vade\*Faiz\*(1-value(left(stopaj,4))))"

Case [Faiz]

ActiveCell.Formula = "=365\*NetGetiri/(Vade\*Anapara\*(1-value(left(stopaj,4))))"

Case [Vade]

ActiveCell.Formula = "=365\*NetGetiri/(Anapara\*Faiz\*(1-value(left(stopaj,4))))"

Case [NetGetiri]

ActiveCell.Formula = "=Anapara\*Faiz\*Vade\*(1-value(left(stopaj,4)))/365"

Case Else

MsgBox "Böyle bir seçenek bulunmamaktadır"

End Select

ActiveCell.Font.Color = vbRed

[uyarı].Value = ""

Call Fontsizedeğiş(24, 20)

Call alancopypaste

End If

End If

Application.EnableEvents = True

ActiveSheet.Protect 1234

Exit Sub

çıkış:

If Err.Description = "No cells were found." Then 'blank sayısı 0 ise, count=0 kontrolüne gelmediği için o noktayı kaldırdım

[uyarı].Select

ActiveCell.Value = "Lütfen hangi alanın yeniden hesaplanmasını istiyorsanız onu silin."

Call Fontsizedeğiş(14, 10)

End If

Application.EnableEvents = True

ActiveSheet.Protect 1234

End Sub

'----------------------------------------------------

Sub temizlik(alan As Range)

For Each a In alan

a.Font.Color = vbBlack

Next a

End Sub

Sub Fontsizedeğiş(x As Integer, s As Integer)

For i = 1 To 5

Call Module2.beklet(s)

DoEvents

ActiveCell.Font.Size = x + i \* 2

Next i

For i = 1 To 5

Call Module2.beklet(s)

DoEvents

ActiveCell.Font.Size = x + 10 - i \* 2

Next i

End Sub

'----------------------------------------------------

Sub alancopypaste()

For Each a In [alan]

a.Value = a.Value

Next a

End Sub

###### Standart Modül içeriği

Bunda sleep metodu kullanıldığı için aşağıdaki özel kod en başa eklenmiştir.

#If VBA7 Then

Public Declare PtrSafe Sub Sleep Lib "kernel32" (ByVal dwMilliseconds As LongPtr) '64 Bit Sistemler için

#Else

Public Declare Sub Sleep Lib "kernel32" (ByVal dwMilliseconds As Long) '32 Bit Sistemler için

#End If

Sub beklet(sure As Integer)

Sleep sure

End Sub

###### Temizlik butonunun kodu ise şöyledir.

Sub Button1\_Click()

Range("alan").ClearContents

ActiveSheet.Unprotect 1234

[uyarı].Value = ""

ActiveSheet.Protect 1234

End Sub

##### Bu hesap makinesinin kullanımı şöyledir:

Kullanıcı diyelim ki, bilinen olarak anapara, faiz ve  vadeyi girip müşterinin net kazancını hesaplamak istiyor olsun. Bu üçünü yazınca net kazanç bilgisi otomatik hesaplanır. Bu hesaplamanın sonucu da bir döngü ile font hacminin önce büyüyüp sonra da küçülmesiyle animasyonlu bir şekilde gösterilir.

Kullanıcı diyelim ki sonradan kazanç bilgisini de manuel değiştirdi, o zaman tüm alanlar dolu olacağı için kodumuz neye göre hesaplama yapacağını bilmez ve kullanıcıya "Lütfen hangi alanın yeniden hesaplanmasını istiyorsanız onu silin" mesajını yine animasyonlu bir şekilde gösterir.

Çalışma mantığı ise şöyledir:

Sayfada belli name'ler tanımlanmış durumda. Makronun tetiklenmesi için "alan" isimli namede bir hücrenin değişmesi beklenmekte. Tabi değişiklikler sonucunda başka tetiklenme olmasın diye eventler geçici olarak baskılanmakta. Değişlik sonucunda alan isimli name'de boş hücre sayısının 1 olup olmadığına bakılmaktadır([alan].Cells.SpecialCells(xlCellTypeBlanks).Count = 1 kodu ile). Böylece bu boş olana uygun formül yazılmakta ve sonuç copy-paste yapılmaktadır.

Alan isimli namede 2 hücre doluyken 3.sünün doldurulması durumunda da, 4 hücre doluyken birinin silinmesi durumunda kontrol sonucu 1 dönecek ve esas işi yapan kod bloğu çalışmış olacaktır.

#### Seçimlere göre veritabanından sonuç getirmek

Bu işlem veritabanı kodlama bilgisi de gerektirdiği için Veritabanı Programlama konusu içine konulmuştur.

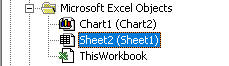
## Application Olayları

### Giriş

Bundan önceki 3 bölümde Workbook, Worksheet ve Chart eventlerini gördük. Hatırlarsanız Worksheet eventleri aynı zamanda Workbook eventleri içinde de geçiyordu. Ör: Workbook\_SheetActivate. Bu şu demekti; ilgili workbook içinde **herhangi** bir sayfayla ilgili olay meydana geldiğinde bu event kodu tetiklenecekti.

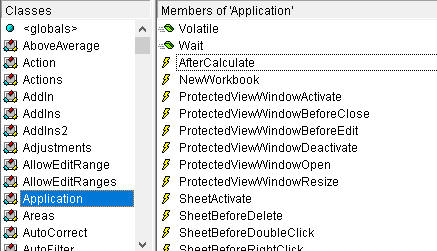
Aynı şekilde açık olan tüm dosya ve sayfaların tamamı için yani Excel’in kendisi için, bir diğer deyişle tüm uygulama(application) seviyesinde geçerli olacak şekilde, herhangi bir Workbook veya Worksheet veya chart eventi meydana geldiğinde bir kodun tetiklenmesini istiyorsak Application eventleri kullanırız.

Ancak Application eventlerinin bir farkı var, bunlar diğer üçü gibi kod penceresinde görünmezler, zira Excel Objects altında böyle bir nesne yoktur. Aşağıda gördüğünüz üzere, Chart, Sheet ve Workbook var, o kadar.



Peki Application eventlerine nasıl ulaşırız? Bunun 2 yolu var:

* İlk yöntemde **Object Browser**'ı kullanıp Application nesnesine gelin, sonra sağ tıklayıp **Group Members** diyin. Propertiler, Metodlar ve Eventler sıralanacaktır. Eventleri en sonda görebilirsiniz.



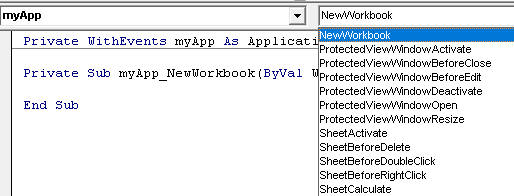
Peki ulaştık ama bunları nasıl kullanacağız? İşte hem bu eventleri görmenin ikinci yöntemi hem de bunları kullanma yöntemine geldi sıra. Bunun da iki yolu var.

#### Class modülde Application nesnesi tanımlama

Şimdi ilgili projeye sağ tıklayıp **Insert>Class Module** diyerek bir class modülü ekleyelim. Evet, Application eventleri class modüller içinde tanımlanır, zaten işlemleri normal modülde yaparsanız hata alırsınız. Sonra da tepeye şu satırları yazın:

Dim WithEvents myApp As Application

Şimdi sol üstteki combodan myApp'i seçin ve sağdaki comboboxta eventleri görün. Böylece eventleri görmenin ikinci yolunu da öğrenmiş olduk. Ama devam edip nasıl kullanacağımıza bakalım.



NOT: Bu iki metodu workbbok ve worksheet nesnelerinde de deneyebilirsiniz ama onların eventi zaten kod penceresinde üstteki comboboxlarda otomatik çıktığı için bu zahmete gerek yok.

Şimdi devam ediyoruz, kulanım için birkaç aşamamız daha var. Öncelikle class modülümüzde şu kodlar olsun. Bu arada properties penceresinde classımızın **Class1** olan adını **myClass** olarak değiştirelim.

Private WithEvents myApp As Application

Private Sub Class\_Initialize()

Set myApp = Application

End Sub

Private Sub myApp\_NewWorkbook(ByVal Wb As Workbook)

MsgBox "Yeni açılan dosya adı: " & Wb.Name

End Sub

Şimdi de **Workbook\_Open** eventine gidip, dosya açılır açılmaz bu classın yaratılmasını sağlayalım.

Dim myClassNesnesi As myClass

Private Sub Workbook\_Open()

Set myClassNesnesi = New myClass

End Sub

Dosyayı kaydedip, kapatalım ve tekrar açalım. Dosya açılınca bu classtan bir nesne yaratılacak, ve nesne yaratılır yaratılmaz da bu classın Initialize eventi sayesinde kendine ait bir eventi olan myApp nesnesi yaratılacaktır. Sonrasında her yeni bir dosya açıldığında(Open değil, New butonu ile) ilgili dosyanın adı görüntülenecektir. Siz isterseniz bu eventin içini özelleştirebilirsiniz, zira bu haliyle pek kullanışlı değil. Aşağıda Çeşitli Örnekler bölümünde biraz daha kullanışlı örnekler bulabilirsiniz.

**NOT**: Buraya kadar olan işlemlerde gördüğünüz gibi Class'ları kullandık. Class ve Class modüller hakkında daha fazla bilgiyi **İleri Seviye Konular>Class ve Class Modüller** konusunda verdiğim linklerden edinebilirsiniz.

#### Workbook modülde Application nesnesi tanımlama

Yukarda dedik ki App nesnesini tanımladığımız modülün Class modül olması lazım. Bildiğiniz gibi Workbook ve Worksheet de aslında bir class olup bunlara ait kod yazdığımız sayfalar da class modül sayfalarıdır. Dolayısıyla bunlara ait sayfalarda da bu nesneyi tanımlayabiliriz.

Şimdi, ThisWorkbook modülüne aşağıdaki kodu yazın.

Public WithEvents myApp As Application

Private Sub Workbook\_Open()

Set myApp = Application

End Sub

Şimdi de yine aynı penceredeyken sol üstte myApp'i seçerek NewWorkbook eventine yine yukardaki gibi aynı kodu yazalım. Bu da bu kadar.

Private Sub myApp\_NewWorkbook(ByVal Wb As Workbook)

MsgBox "Yeni açılan dosya adı: " & Wb.Name

End Sub

#### Son Söz

##### Hangi yöntem seçilmeli

Peki biz bu iki yöntemden hangisini kullanmalıyız. Tamamen kişisel tercih olmakla birlikte üstatlara kulak verebiliriz. Derler ki, ayrı bir class modül kodların derli toplu durması adına daha iyidir. Workbook ve Worksheet sayfalarında onların kendine has kodlarını yazalım.

##### Olayların sırası

Elimizde myApp\_SheetActivate,Workbook\_SheetActivate ve Worksheet\_Activate olaylarının hepsi de varsa, hiyerarşik seviyeye göre işleme girerler. Yani bir sayfa aktive olduğunda önce Application olayı, sonra Workbook olayı, en son da sayfa olayı gerçekleşir.

### Çeşitli Örnekler

#### myApp\_WorkbookOpen ile dış bağlantı tespiti

Aşağıdaki örnekte açılan dosyalarda harici bir bağlantı var mı diye bize söyleyen bir kod bulunuyor.

Bunu yine yukardaki örnekte olduğu gibi Personal.xlsb dosyası içinde ele alalım.

Private Sub myApp\_WorkbookOpen(ByVal Wb As Workbook)

If Wb.Connections.Count > 0 Or Wb.Queries.Count > 0 Then

MsgBox "bu dosya harici bağlantı içeriyor"

End If

End Sub

#### myApp\_WorkbookBeforePrint ile Print alma engeli

Olur da acil bir iş nedeniyle bilgisayarımızı açık bıraktık gittik, birileri gelip bizim iznimiz olmadan bilgisayarımızdan bir şeylerin çıktısını almak isteyebilir. Bunu aşağıdaki kod ile engellemiş oluruz.

Bunu da yine yukardaki örnekte olduğu gibi Personal.xlsb dosyası içinde ele alalım.

Private Sub myApp\_WorkbookBeforePrint(ByVal Wb As Workbook, Cancel As Boolean)

şifre = InputBox("Yazdırma şifresini girin")

If şifre <> "1234" Then

Cancel = True

MsgBox "Şifreyi bilmiyorsanız bu bilgisayardan hiçbir Excel dosyasının çıktısını alamazsınız"

End If

End Sub

## Özel Olaylar

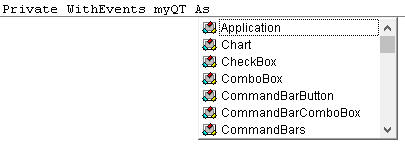
Bundan önceki bölümlerde Excel'in kendi nesnelerinin eventlerinin otomatik olarak kod penceresine eklendiğini (ve bunların değiştirilmemesi gerektiğini) söylemiştik. Bu bölümde kendi nesnelerimize(classlarınıza) ait eventleri nasıl yaratırız bunu göreceğiz. Bunun için Class kavramını biraz biliyor olmanız gerekiyor. Bilmeyenler ön bilgiyi **İleri Seviye Konular>Class ve Class Modüller** konusundaki linkleri ziyaret ederek edinebilirler.

### Kendi Classlarınıza ait olaylar

Bu bölümü yapmaya başladığımda nerden başlayacağıma bir türlü karar veremedim, zira Class konusuna bayağı bir girmek gerekiyordu. Şuan ise sayfaları sırayla hazırladığım için ve henüz Class konusuna girmediğim için biraz anlamsız geldi. O yüzden bu konuyu üstteki paragrafta belirttiğim linke bırakıyorum.

### Diğer özel olaylar

Her ne kadar özel olay olmasa da farklı tanımlama ve erişim şekilleri itibarıyla klasik olaylardan farklı oldukları için **WithEvent** deyimi ile tanımlanan olayları da özel olay gibi düşünebiliriz. Bundan önce **Grafik** ve **Application** için bunları nasıl yazdığımızı görmüştük. Bu tür özel olayları başka nesneler için de kullanabiliyoruz. Tüm kullanılabilir nesnelerin listesini aşağıdaki gibi bir değişken tanımlarken **As**'den sonra boşluk tuşuna basınca görebilirsiniz.



Bunlardan ComboBox, ListBox, Image v.s gibi **ActiveX** kontrollerine ait eventleri zaten Formlar konusunda ayrıca ele alıyor olacağız. Bu sayfadaki konsepte göre tanımlamanın pek bir esprisi yok bence. O yüzden bunları geçiyoruz.

Biz burada sadece **QueryTable** nesnesinin eventlerine bakacağız. ActiveX eventlerine ise Formlar bölümünde bakacağız.

### QueryTable Event örneği

Aşağıdaki kod ile kullanıcıyı uzun bir refresh işlemi için uyarıyoruz, refresh işlemi bitince de bir mesaj kutusu ile haber veriyoruz.

Örnek dosyayı **githubdan(eventQTclass.xlsm)** indirebilirsiniz. Aşamalarımız şöyle:

* Öncelikle yeni bir dosya açın
* VBE'ye geçip bir class modül ekleyin ve propertiesten buna myClass adını verin.
* Sonra aşağıdaki kodu bu modüle yapıştırın.
* Sonrasında dosyanıza bir veri bağlantısı ekleyin(Access, Excel, Oracle, SQL Server v.s olabilir). Örnek olması adına yine **githubdaki pivotdata.xlsx** dosyasından herhangi bir sayfayı ekleyebilirsiniz.(Veri tabanı bağlantılarına aşina değilseniz önce web sitemdeki **Excel>Data Menüsü>Başka Veri Kaynaklarıyla çalışmak** sayfasına ([https://www.Excelinefendisi.com/Konular/Excel/DataMenusu\_BaskaVeriKaynaklariilecalismak.aspx](https://www.excelinefendisi.com/Konular/Excel/DataMenusu_BaskaVeriKaynaklariilecalismak.aspx)) bakın.

Private WithEvents myQT As QueryTable

Private Sub Class\_Initialize()

Set myQT = ActiveSheet.ListObjects(1).QueryTable

End Sub

Private Sub myQT\_AfterRefresh(ByVal Success As Boolean)

If Success = True Then

MsgBox "refresh işlemi bitti"

Else

MsgBox "refresh sırasında bir hata oluştu"

End If

End Sub

Private Sub myQT\_BeforeRefresh(Cancel As Boolean)

cevap = MsgBox("uzun sürecek, iptal edeyim mi", vbYesNo)

If cevap = vbYes Then

Cancel = True

MsgBox "iptal edildi"

End If

End Sub

Son olarak **ThisWorkbook** modülünün içine de şunları yazın.

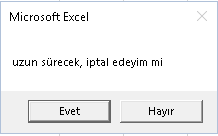
Dim myClassNesnesi As myClass

Private Sub Workbook\_Open()

Set myClassNesnesi = New myClass

End Sub

Bu kadar basit. Şimdi dosyanızı kaydedip kapatın ve tekrar açın, sonra da data üzerinde bir yere gelip, sağ tıklayıp **Refresh** deyin. Aşağıdaki mesajı görmeniz lazım.



Kodun çalışma prensibi şöyle:

Bir class modülümüz var. Dosya açılır açılmaz bu classtan bir nesne yaratılıyor. Bu nesne yaratılınca class'ın Initialize eventi devreye giriyor ve bu sefer de myQT nesnesine sayfadaki 1 indexli QueryTable atanıyor. Sonrasındaki refresh işlemleri ise aşikar.

## Neler Öğrendik

Belirli anlarda belirli eylemleri yerine getirmemizi sağlayan olaylardan nasıl faydalanacağımızı gördük.

Yerel olayları kullanmayı ve kendi olaylarımızı nasıl yazacağımızı öğrendik.