T.C. MILLI EĞITIM BAKANLIĞI





MESLEKİ EĞİTİM VE ÖĞRETİM SİSTEMININ GÜCLENDIRILMESİ PROJESİ)

BİLİŞİM TEKNOLOJİLERİ

NESNE TABANLI PROGRAMLAMA - 2

ANKARA 2008

Milli Eğitim Bakanlığı tarafından geliştirilen modüller;

- Talim ve Terbiye Kurulu Başkanlığının 02.06.2006 tarih ve 269 sayılı Kararı ile onaylanan, Mesleki ve Teknik Eğitim Okul ve Kurumlarında kademeli olarak yaygınlaştırılan 42 alan ve 192 dala ait çerçeve öğretim programlarında amaçlanan mesleki yeterlikleri kazandırmaya yönelik geliştirilmiş öğretim materyalleridir (Ders Notlarıdır).
- Modüller, bireylere mesleki yeterlik kazandırmak ve bireysel öğrenmeye rehberlik etmek amacıyla öğrenme materyali olarak hazırlanmış, denenmek ve geliştirilmek üzere Mesleki ve Teknik Eğitim Okul ve Kurumlarında uygulanmaya başlanmıştır.
- Modüller teknolojik gelişmelere paralel olarak, amaçlanan yeterliği kazandırmak koşulu ile eğitim öğretim sırasında geliştirilebilir ve yapılması önerilen değişiklikler Bakanlıkta ilgili birime bildirilir.
- Örgün ve yaygın eğitim kurumları, işletmeler ve kendi kendine mesleki yeterlik kazanmak isteyen bireyler modüllere internet üzerinden ulaşılabilirler.
- Basılmış modüller, eğitim kurumlarında öğrencilere ücretsiz olarak dağıtılır.
- Modüller hiçbir şekilde ticari amaçla kullanılamaz ve ücret karşılığında satılamaz.

İÇİNDEKİLER

AÇIKLAMALAR	iii
GİRİŞ	1
ÖĞRENME FAALİYETİ–1	3
1. PENCERELER	3
1.1. Kullanıcı Arabirimi İçin Tasarım Kuralları	3
1.2. Formlar (Pencereler)	4
1.2.1. Basitlik (Simplicity)	4
1.2.2. Kontrollerin Yeri	4
1.2.3. Uyum – Tutarlılık	5
1.2.4. Estetik	5
1.2.5. Renkler	5
1.2.6. Yazı Tipleri	5
1.2.7. Resimler ve Simgeler	6
1.2.8. Şekiller ve Şeffaflık	6
1.3. Formların Kullanımı	6
1.3.1. Projeye Form Eklemek	6
1.3.2. Başlangıç Formu (Start-Up Form) Ayarı	8
1.3.3. Formun Görünümünü Değiştirmek	9
1.3.4. Arka plan Rengi, Ön Plan Rengi ve Metin Özellikleri	.10
1.3.5. Yazı Tipi, İmleç ve Arka Plan Resmi	.11
1.3.6. Opasite (Donukluk)	.11
1.4. Formların Metotlarını (Fonksiyon) Kullanma	.11
1.4.1. Show ve ShowDialog (Göster ve Diyalog Kutusu Göster)	.12
1.4.2. Activate (Aktif Yap)	.12
1.4.3. Hide (Gizle)	.13
1.4.4. Close (Kapat)	.13
1.4.5. Formun Olayları (Form Event)	.13
UYGULAMA FAALİYETİ	.15
ÖLÇME VE DEĞERLENDİRME	.16
ÖĞRENME FAALİYETİ–2	.17
2. KULLANICI ARABİRİMİ NESNELERİ	.17
2.1. Komponent ve Kontrolleri Kullanma	.17
2.1.1. Kontrollerle Çalışmak	.18
2.1.2. Fare Etkileşimleri	.27
UYGULAMA FAALİYETİ	. 28
ÖLÇME VE DEĞERLENDİRME	. 29
ÖĞRENME FAALİYETİ–3	.30
3. MENÜLER	.30
3.1.Menüleri Kullanma	.30
3.1.1. Tasarım Aşamasında Menü Oluşturmak	. 31
3.1.2. Menü Erişimi ve Kısa Yol Tuşları	. 32
3.1.3. Menü Öğelerinin Olaylarını Kullanma	. 33
3.1.4. İçerik Menüsü (Context Menu) Oluşturma	. 33
3.1.5. Çalışma Zamanında Menüleri Değiştirmek	.34
3.1.6. Menü Komutlarını Aktif ve Pasif Yapma	. 34

3.1.7. Menü Öğelerini Gizleme	
3.1.8. Menüleri Kopyalama	35
3.1.9. Çalışma Anında Menü Öğelerini Birleştirme	35
3.1.10. Çalışma Anında Menü Öğeleri Eklemek	35
3.2. Kullanıcı Girişini Onaylamak (Validating)	
3.2.1. Alan Düzeyli Doğrulama	
3.2.2. TextBox Özelliklerini Kullanma	
3.2.3. Alan Düzeyli Doğrulamada Olayları Kullanmak	
3.2.4. Doğrulanan Karakterler	
3.2.5. Odaklama (Focus) Ayarı	
UYGULAMA FAALİYETİ	40
ÖLÇME VE DEĞERLENDİRME	41
MODÜL DEĞERLENDİRME	
CEVAP ANAHTARLARI	43
SÖZLÜK	44
KOD ÖRNEKLERİ	45
ÖNERİLEN KAYNAKLAR	47
KAYNAKÇA	

AÇIKLAMALAR

КОД	482BK0075			
ALAN	Bilişim Teknolojileri			
DAL/MESLEK	Veri Tabanı Programcılığı			
MODÜLÜN ADI	Nesne Tabanlı Programlama 2			
MODÜLÜN TANIMI	Bu modül öğrencinin, gerekli ortam sağlandığında, nesne tabanlı programlamada kullanıcı arabirimi ve menü tasarımı yapabilmesini sağlayan öğrenme materyalidir.			
SÜRE	40/32			
ÖN KOŞUL	Nesne Tabanlı Programlama 1 modülünü bitirmiş olmak			
YETERLİK	Nesne tabanlı programlama dili ile ara yüz yapmak			
MODÜLÜN AMACI	 Genel Amaç Arabirimi ve menü tasarımı yapabileceksiniz. Amaçlar ➢ Pencereler ile çalışabileceksiniz. ➢ Kullanıcı arabirimi nesneleri oluşturabileceksiniz. ➢ Menüler ve kullanıcı girişini yapabileceksiniz. 			
EĞİTİM ÖĞRETİM ORTAMLARI VE DONANIMLARI	Bilgisayar laboratuvarı, nesne tabanlı program, bu programları kullanan işletmeler.			
ÖLÇME VE DEĞERLENDİRME	 Her faaliyet sonrasında o faliyetle ilgili değerlendirme soruları ile kendi kendinizi değerlendireceksiniz. Öğretmen modül sonunda size ölçme aracı (uygulama, soru-cevap) uygulayarak modül uygulamaları ile kazandığınız bilgi ve becerileri ölçerek değerlendirecektir. 			

iv

GİRİŞ

Sevgili Öğrenci,

Windows uygulamaları, özellikle son kullanıcıya hitap eden grafik ara yüzlü programlardır. Her ne kadar son dönemlerde web tabanlı programlama modeli yaygınca kullanılsa da Windows işletim sisteminde tek başına veya diğer servislerle ilişkili bir şekilde çalışacak grafik ara yüzlü programlara ihtiyaç duyulmaktadır. .NET sınıf kütüphanesi sayesinde Windows formlarına yönelik program geliştirmek gerçekten çok kolaylaşmıştır. Form dediğimiz nesneler, Windows tabanlı programlarda gördüğümüz arabirimlerden başka bir şey değildir. Windows formlarına yönelik program geliştirmek, Visual Studio.NET ile çok daha esnek hale getirilmiştir. Burada, bu tür programları herhangi bir metin editöründe geliştiremeyeceğimiz anlamı çıkmamalıdır.

Windows formlarına ilişkin sınıflar System.Windows.Forms isim alanında (namespace) bulunmaktadır. Bu sınıflardan ilk önce öğreneceğimiz Form sınıfıdır. Form sınıfı program çalıştırıldığında ekranda boş bir pencerenin gösterilmesi için gereken özellikleri belirler. Geliştireceğimiz uygulamalarda bu Form sınıfından yeni sınıflar türetip daha gelişmiş form yapıları tasarlayabiliriz. Form sınıflarının birçok özelliği, metodu ve olayı vardır. Modül içinde bu elemanlardan en önemli olanlara değinilecektir.



ÖĞRENME FAALİYETİ–1

AMAÇ

Uygun ortam sağlandığında Pencereler (Formlar) ile çalışabileceksiniz.

ARAŞTIRMA

Bu faaliyet öncesinde aşağıda belirtilen araştırma faaliyetlerini yapmalısınız:

- Size, şu ana kadar kullanımı en kolay gelen programı düşününüz.
- Bu programı kullanmaya sizi çeken en önemli öğenin ne olduğunu yazınız.

1. PENCERELER

1.1. Kullanıcı Arabirimi İçin Tasarım Kuralları

Kullanıcı arabirimi uygulamaların kullanıcılar ile etkileşimini sağlar. Bu nedenle etkili bir tasarımda vazgeçilmez öğedir. Kullanıcı arabirimi tasarlarken birinci önceliğimiz uygulamanın kullanıcılar tarafından kullanılmasıdır, çünkü kullanıcılar bizim hedef kitlemizdir. Hedef kitlenizin kullanıcılar olduğunu bilmek programlarınızın yazımını kolaylaştırdığı gibi, o programı kullanacak kişilere de yardımcı olacaktır. Kötü bir tasarım kullanıcıların programınızı kullanmaktan uzaklaştırabilir.

Formlar (pencereler) Windows uygulamalarının temel elemanlarıdır. Aslında formlar kullanıcıların programla etkileşim evrelerinin temelini kurar. Kontroller ve menüler belirli özellikleri programa katmak için formlara eklenebilir. Buna ek olarak oluşturulan programın fonksiyonel ve daha cazibeli olmasını sağlar. .NET Framework uygulamanızın görsel sunumu için çeşitli grafiksel öğeler, formlar, kontroller, şeffaf elemanlar ve gölgeli elemanlar ihtiva eder. Windows formlarına ilişkin sınıflar, **System.Windows.Forms** isim alanında bulunmaktadır.

Bu dersten sonra aşağıdakileri yapabileceksiniz:

- Kullanıcı arabiriminin önemini tanımlayabileceksiniz.
- Formlar, kontroller ve menülerin kullanıcı arabirimindeki önemini açıklayabileceksiniz.
- > Uygulamalarınızdaki renklerin önemini açıklayabileceksiniz.
- Etkileşimli bir tasarım için gerekli olan resim, ikon ve fontların önemini açıklayabileceksiniz.

1.2. Formlar (Pencereler)

Formlar uygulamalarda ihtiyaç duyulan bilgiler için gerekli olan bir grup öğeden oluşur. Her form aynı zamanda bir sınıftır (class) ve siz bu formun birkaç örneğini oluşturabileceğiniz gibi, bir başka formdan miras (inherit) da alabilirsiniz.

Kontroller, kullanıcıların erişebileceği bilgiler ve seçeneklerdir. Kontroller mesela, etiket veya resim kutuları bilgi gösterebilir. Metin kutuları, liste kutuları, combo box'lar hem bilgi gösterir hem de kullanıcının bilgi girişi yapabilmesini sağlar. Butonlar kullanıcının bir olayı (programı kapatmak gibi) gerçekleştirmesini sağlayabilir. Menüler ve araç çubukları ise, kullanıcıların uygulama içinde kullanabileceği komutları ihtiva eder. Menüler genellikle bir uygulama içindeki tüm formlar için ortak olan bir üst seviye komutları saklar, mesela "save" komutu yardımıyla bilgiler kaydedilip programdan çıkılabilir. Menü elemanları uygulamanın farklı noktalarındaki özelleştirme seçenekleri ile aktif veya pasif edilebilir.

Yazılan uygulamanın kullanıcı tarafından hızlı ve kolay öğrenilmesi, kullanıcıyı programı kullanma yönünde teşvik etmesi çok önemlidir. Uygulamanın düzeni konusunda başlıca faktörler şunlardır:

- Basitlik
- Kontrollerin yeri
- Uyum-tutarlılık
- ➢ Estetik

1.2.1. Basitlik (Simplicity)

Kullanıcı arabiriminin en önemli yönü basitliktir. Aşırı derecede karışık bir arabirim programın öğrenilmesini zorlaştırabilir. İyi bir uygulama kullanıcıyla program arasındaki etkileşimi iyi sağlamalı, aynı zamanda uygulamanın her bölümünde fonksiyonel olmalıdır. Uygulama içindeki kontroller form içinde birada gruplanmalıdır. Kontroller – list box, combo box, check box – kullanıcıların önceden belirlenmiş bazı seçenekleri seçebilmelerini sağlar. Sekme sırası (tab order) kullanıcıların alanlar arasında rahat hareket edebilmelerini kolaylaştırır.

Varsayılan değerler uygulamanızın kolay kullanılabilmesini sağlayan bir başka faktördür. Mesela programınız içinde, yaşadığınız şehirleri gösteren combo box'ta İstanbul'u ilk sırada göstermek buna iyi bir örnektir.

1.2.2. Kontrollerin Yeri

Kullanıcı arabirimi içinde kullanılan kontrollerin yerleşimi programın sıklıkla kullanılmasını etkileyen unsurlardandır. Mesela, içinde girilmesi zorunlu ve girilmesi isteğe bağlı olan kontrollerimizin olduğu bir form düşünelim. Genelde Windows tabanlı programlarda form içindeki kontroller soldan sağa ve yukarıdan aşağıya doğru tasarlanır. Çok önemli ya da sıklıkla kullanılan kontroller formun en üstünde yer alır. Herhangi bir olayı tamamlayan bir kontrol (mesela onay butonu), mantıksal bir sırayı takip etmeli ve formun en altında yer almalıdır.

Birbiriyle alakalı bilgileri göstermek için kullanılan kontroller bir grup içinde ele alınmalıdır. Mesela, bir müşteri ile alakalı adresi, telefon numarası, satın aldığı tarih gibi bilgileri bir grup içinde değerlendirebilirsiniz.

1.2.3. Uyum – Tutarlılık

Uygulamanızda kullandığınız her form uyumlu bir tasarım örneği sergilemelidir. Karmaşık bir tasarım uygulamanızı düzensizleştirdiği gibi, hedef kitleniz olan kullanıcıların programa adaptasyonunu da zorlaştırır. Uyumlu bir tasarım için kullanılan renkler, yazı tipleri, boyutlar ve kontrollerin tipi önemlidir. Gerçek bir uygulama meydana getirmeden önce kullanılacak görsel plana karar verilmelidir. Bu işlem sırasında gösterişten kaçınılmalıdır. Konu ile ilgisi olmayan kontroller kullanmak ya da göze çarpan öğeler kullanmak, kullanıcının dikkatini dağıtmaktan başka bir işe yaramaz.

1.2.4. Estetik

Kullanıcı arabirimi mümkün olduğunca hoş ve davetkâr olmalıdır. Ama bu yapılırken programın anlaşılır ve basit olmasından ödün verilmemelidir. Elinizden geldiğince kullanıcıyı programınızı kullanmaktan soğutacak işlerden uzak durunuz.

1.2.5. Renkler

Kullanıcı arabiriminizde kullanılan akıllıca renkler, programınızın çekiciliğini artırır ve kullanıcıyı programı kullanması için davet eder. Program içinde kullanılan renklerin çok canlı olması bazı kullanıcılara hoş gelebilirken bazıları için ters etki yapabilir. Uygulamanızın arka plan renk düzenini seçerken en iyi yöntem yumuşak renkler kullanmaktır.

Uygulamanız hangi konu ile alakalı ise renkleri de ona göre seçiniz. Mesela yerel bir şirket için uygulama yapacaksanız, burada kullanacağınız renk düzeni şirketin kendine has renk düzeni ile uyumlu olmalıdır. Uluslararası bir şirket için tasarım yaparken kullanacağınız renklerin kültürel önemlerinin de olduğunu hesaba katmalısınız. Tutarlı bir tasarım için renkleri abartılı kullanmamak gerekir.

Renklerin nasıl bir etki oluşturduğunu her zaman düşünmek zorundasınız. Mesela mavi zemin üzerine siyah bir yazı yazmak okumayı zorlaştırdığı gibi programın kullanılabilirliğini de zayıflatır. Bazı insanlar kırmızı ve yeşil rengi ayırt edemeyebilir. Bu nedenle yeşil bir zemin üzerine yazılmış kırmızı bir yazı bu kullanıcı tarafından okunamayacaktır.

1.2.6. Yazı Tipleri

Uygulamanız için seçtiğiniz yazı tipleri programın kullanılabilirliğini etkiler. Basit kullanımlar için seçtiğiniz yazı tipleri kolay okunan tipler olmalıdır. Mesela Times New Roman kolay okunan yazı tiplerindendir. El yazısı ya da dekoratif amaçlı yazı tiplerini sadece sayfa başlığı için kullanınız. Önemli bilgiler için kesinlikle bu tip yazı tiplerini tercih etmeyiniz.

1.2.7. Resimler ve Simgeler

Resimler ve simgeler programınızı daha ilgi çekici hale getirir ama bunları ölçülü kullanmak gerekir. Kullanıcının dikkatini dağıtan resimler uygulamanın kullanımını engelleyebilir. Simgeler de bilgi taşır ama bunları da kullanırken dikkatli olunmalıdır. Mesela program icinde, yabancı ülkelerin bir kısmında durma anlamına gelen kırmızı renkli bir sekizgen kullandığımızı farz edelim. Bu işaret birkaç ülke kullanıcısı için durma anlamına gelirken diğer ülke kullanıcıları icin farklı anlamlara gelebilir. Bu da uygulamayı kullanan kullanıcıların bazıları için anlamsız işaretten farklı olmaz. Hedef kitlenizi biliniz, kullanacağınız görsel elemanları ve simgeleri ona göre seçiniz.

1.2.8. Şekiller ve Şeffaflık

.NET Framework form ve kontrolleri olustururken daha önceden kullanılan normal dörtgenler verine seffaflığı ayarlanabilen cesitli araclar sağlar. Bu araclar yardımıyla güclü görsel efektler oluşturulabilir fakat bunların aşırı kullanılmamaları gerekir. Devamlı son kullanıcıları düşünerek bu araçlar kullanılmalıdır. Mesela yarı saydam bir form kullanmak arka planın görünümünü değiştirir. Bununla birlikte nihai amac olan programın kullanılabilirliği unutulmamalıdır.

1.3. Formların Kullanımı

Formlar kullanıcı arabiriminin en temel elemanlarıdır. Formlar uygulama içindeki kontrolleri saklamakla görevlidir. Formlar bilgi sunabildikleri gibi, kullanıcıdan bilgi de alabilir. Windows uygulamaları en az bir adet form ihtiva eder. Karmaşık uygulamalarda ise birden fazla form kullanılır.

Bu dersten sonra aşağıdakileri yapabileceksiniz:

- \triangleright Uygulamalardaki formun rolünü kavramak
- \triangleright Uygulamalara nasıl form eklendiğini açıklamak
- Formun başlangıç yeri ve başlangıç ayarlarını açıklamak
- Formun görsel görünümünü nasıl değiştirebileceğimizi açıklamak
- \triangleright Formun metotlarını nasıl kullanabileceğimizi acıklamak
- Formun olaylarını nasıl kullanabileceğimizi açıklamak

1.3.1. Projeye Form Eklemek

Formlar kullanıcı ile uygulama arasındaki etkileşimi sağlar. Yeni bir Windows uygulaması oluşturduğunuzda "form1" adında bir form varsayılan olarak uygulamanıza eklenir. Tasarım ekranındaki form1'e kontroller, menüler ve görsel öğeler eklenebilir. Bunun için tasarım ekranının sol tarafında yer alan ToolBox (araç kutusu) kullanılır. Bir formun ya da bir elemanın tasarlanması "tasarım-zamanı" (design-time) olarak adlandırılır. Uygulamanızın boyutu büyüdükçe yeni formlar eklemek zorunda kalabilirsiniz.

Projeye yeni form eklemek:

- > Project menüsünden Add Windows Form'a tıklayınız.
- Gelen Add New Item (yeni öğe ekle) ekranında Windows forms'u seçip OK'e tıklayınız.



Resim 1.1: Projeye yeni form eklemek

Add New Item - WindowsApplicati	on3		? ×
Templates:			00 00 00 00 00
Visual Studio installed templa	tes		_
 Class Windows Form Inherited Form Component Class XML File HTML Page Bitmap File Icon File Script File Assembly Information File Settings File Debugger Visualizer 	 ☆ Interface ☑ User Control ☑ Inherited User Control ☑ SQL Database ℤ XML Schema ▲ Style Sheet ♦ Cursor File ₩indows Service ♥BScript File ▲ Application Configuration File MDI Parent ♀ Class Diagram 	 Code File Custom Control DataSet XSLT File Text File Report Installer Class Windows Script Host Resources File About Box 	
A blank Windows Form			
Name: Form2.cs			
		Add	Cancel

Resim 1.2: Projeye yeni öğe ekleme ekranı

Kod kısmını kullanarak da projenize yeni bir form ekleyebilirsiniz.

1.3.2. Başlangıç Formu (Start-Up Form) Ayarı

Uygulamanız içinde birden fazla form mevcut ise bunlardan birini başlangıç formu olarak ayarlamalısınız. Başlangıç formu, uygulamanızın ilk çalıştığı anda karşınıza gelen formdur. Hangi formun başlangıçta ekrana gelmesini istiyorsanız, Main () metodu içinde o formun adını yazmalısınız.

Yukarıdaki program parçasında da görüldüğü gibi, uygulamanız içinde birden fazla form varsa ve bunlardan hangisi ilk olarak ekrana gelecekse, o formun adını Main() metodu içindeki Application.Run(new Form1()) satırında belirtiyoruz.



Resim 1.3: Formun başlama anındaki pozisyonunu ayarlamak

Hazırlanan formun ekrana ilk geliş ayarı yapılabilir. Bunun için ilgili form seçildikten sonra sağ alt köşedeki properties penceresinden ilgili ayar seçilir.

Manual: Form, Location (yer) özelliğinde belirtilen koordinatlar da ekrana gelir.

Center Screen: Ekrana ortalanmış olarak gelir.

WindowsDefaultLocation: Windows'un mevcut koordinatlarına göre ekrana gelir.

WindowsDefaultBounds: Var olan koordinatlar ve mevcut ekran çözünürlüğüne göre ekrana gelir.

CenterParent: Ana formun ortasına denk gelecek şekilde ekrana gelir.

1.3.3. Formun Görünümünü Değiştirmek

Uygulamanızın arabiriminin görüntüsü en önemli öğelerden biridir. Arabirimin kötü olması programın anlaşılabilirliğini etkiler ve bu da harcanan zamanı dolayısıyla da maliyetleri artırır. Özellikler penceresi formunuzun görüntüsünü ayarlamak için çeşitli seçenekler sunar.

Formların birçok özelliği vardır. Aşağıdaki resimde formun çeşitli özellikleri görülmektedir.

Pro	perties	×
Fo	rm1 System.Windows.Forms.F	orm
	2↓ 🔲 🖻	
Ŧ	(DynamicProperties)	A
	(Name)	Form1
	AcceptButton	(none)
	AccessibleDescription	
	AccessibleName	
	AccessibleRole	Default
	AllowDrop	False
	AutoScale	True
	AutoScroll	False
Ŧ	AutoScrollMargin	0, 0
Ŧ	AutoScrollMinSize	0, 0
	BackColor	Control
	BackgroundImage	🗌 (none) 📃 📕
Ba Th co	ickColor e background color used to displ ntrol.	ay text and graphics in the
- [Properties 😢 Dynamic Help	

Resim 1.4: Forma ait özellikler penceresi

1.3.4. Arka plan Rengi, Ön Plan Rengi ve Metin Özellikleri

Kullanıcıların dikkatini ilk önce renkler ve metinler çeker. Metin özelliği formun başlığını gösterir. Arka plan ve ön plan renkleri ise formun renklerini gösterir. Tasarım (design) ekranında forma eklenen her kontrolün – butonlar, etiketler gibi – rengi mevcut forma göre ayarlanabilir. Metin kutusu gibi diğer kontrollerin kendine ait bağımsız ayarlama seçenekleri ile istenen değişiklikler yapılabilir.

Renklerinizi seçerken dikkatli olunuz. Mesela, mavi zemin üzerine kırmızı renkli bir metin yazarsanız programınız çekici gelebilir ama programın kullanımını oldukça zorlaştırır. Bunun için yüksek karşıtlıklı (high Contrast) renkleri tercih etmenizde fayda vardır. Bu ayarları resim 1.4'te de görüldüğü gibi, özellikler (Properties) penceresinden ayarlayabilirsiniz.



Resim 1.5: Renk ve yazı tipi ayarları değiştirilmiş bir form örneği

Örneğin yukarıdaki formun arka plan rengi (BackColor) mavi, ön plan rengi (ForeColor) beyaz, buton kontrolünün rengi ise turuncu seçilmiştir.

1.3.5. Yazı Tipi, İmleç ve Arka Plan Resmi

Uygulamanızın arabiriminin düzenlenmesi için bir başka yol da seçilen yazı tipi, imleç ve arka plan resminin seçilmesidir. Yazı tipi özelliği kullandığınız formdaki metinlerin yazı tipini değiştirmenize izin verir. İmleç özelliği, formunuz üzerinde görünecek olan Mouse işaretçisini değiştirir. Arka plan resmi ise, formunuzun arka planına herhangi bir renk yerine bir resim ekler. Arka plan resmi ekledikten sonra, arka plan rengi özelliğini kullanamazınız.



Resim 1.6: Arka plan resmi eklenmiş bir form örneği

1.3.6. Opasite (Donukluk)

Formunuzun transparanlık (şeffaflık) ayarını özellikler panelindeki "Opacity" seçeneğinden yapabilirsiniz. Code kısmından programınızı yazıyorsanız, buradaki Opacity değeri 0 ile 1 arasında bir değer olarak verilmektedir. Designer (tasarım) ekranındaki özellikler (properties) penceresinde ise varsayılan olarak tüm formların Opacity değeri % olarak ayarlıdır. Buradaki ayarı istediğiniz gibi değiştirerek formunuzu daha dikkat çekici bir hale getirebilirsiniz.

1.4. Formların Metotlarını (Fonksiyon) Kullanma

Metotlar, diğer bir ifadeyle fonksiyonlar herhangi bir olayı gerçekleştirir. Uygulamanıza eklenen her form "System.Windows.Forms.Form" class'ından miras (inherit) alınmıştır. Kullanıcı ortamındaki formlara erişim ve formların gösterimi için çeşitli metotlar vardır. Bu metotları aşağıdaki gibi sıralayabiliriz:

- ➢ Form.Show
- ➢ Form.ShowDialog
- Form.Activate

- ➢ Form.Hide
- ➢ Form.Close

Bu metotları kullanabilmek için, formun bir referansına sahip olmanız gerekir, diğer bir deyişle formun bir örneğinin oluşturulmuş olması ve hafıza yerinin ayrılmış olması şarttır.

Form class'ı (sınıfı) içinde kod yazarken "this" anahtar sözcüğünü kullanarak formun mevcut örneğine gönderme yapabilirsiniz. Mesela, formunuzun "Text" özelliğini değiştirecek bir metot yazdığınızı farz ediniz. Bunun için aşağıdaki gibi bir code yazmanız gerekecektir.

// Bu satır mevcut formun "Text" özelliğini değiştirir. this.Text = "Bu Aktif formdur";

1.4.1. Show ve ShowDialog (Göster ve Diyalog Kutusu Göster)

Formların kullanılabilir olması için görünür (visible) olmaları gerekir. Bir formu görünür yapabilmek için formun "Form.Show" metodunu kullanmanız gerekir. Bu metot form class'ının bir örneğini hafızaya yükler ve ekranda gösterir. Form.Show metodu çağrıldığında formun "visible" özelliği true (doğru) olarak ayarlanır.

Form.ShowDialog hem Form.Show'un işini yapar hem de bir diyalog kutusu görüntüler. Diyalog kutusunun görüntülenmesi, kullanıcının programın geri kalan kısmına devam etmesi için mevcut formu kapatabilmesine imkân tanır. Bu metot genelde özel bir olayı gerçekleştirmek için kullanılır. Mesela, programınızın herhangi bir yerinde Form.ShowDialog metodunu kullanarak kullanıcıya disket sürücüsünde disket olmadığını veya bir parola girmesi gerektiğini söyleyebiliriz.

// DialogForm adında bir form oluşturulduğunu farz ediniz. DialogForm myForm = new DialogForm(); // formu düzenli gösterir. myForm.Show(); // formu modelli olarak gösterir. myForm.ShowDialog();

1.4.2. Activate (Aktif Yap)

Bazen oluşturduğunuz form görünür olmasına rağmen aktif edilmemiş olabilir. Bunun için formların **Form.Activate** metodunu kullanabilirsiniz. Uygulama çalıştırıldığında, Form.Activate metodu formu ekranın en önüne (odak-focus) getirir. Kullanıcı arabirimindeki form aktif değilse (program görev çubuğunda ise), ilgili formun pencere başlığı yanıp sönerek (flash) kullanıcıyı uyarır. Bu metodun görevini yerine getirebilmesi için, formun görünür olması gerekir.

myForm.Activate();

1.4.3. Hide (Gizle)

Form.Hide metodu ilgili formu gizler. Bu form normalde hafızada bir yer işgal etmesine rağmen formun Form. Show metodunun değeri "true" yapılmadıkça form tekrar görünür olmaz. Bu metodu çağırmak formun visible (görünürlük) özelliğini "false" yapmak anlamına gelir.

myForm.Hide();

1.4.4. Close (Kapat)

Herhangi bir formla ilgili işiniz bittiğinde formun Form.Close metodunu kullanarak bu formu kapatabilir ve hafızadan silebilirsiniz. Bu metot formla ilgili kullanılan tüm öğeleri Çöp Toplayıcısına (Garbage Collector) gönderir. Form.Close metodunu çağırdıktan sonra Form.Show metodunu kullanamazsınız. Çünkü formla ilgili tüm kaynaklar artık yoktur.

myForm.Close();

1.4.5. Formun Olayları (Form Event)

Olaylar, form içindeki gerçekleşen bazı işlemleri yapar. Bir önceki konuda bahsedildiği gibi, metotlar uygulandığında bazı olaylar gerçekleşir. Farenin yer değiştirmesi, bir tuşa basılması, formların açılması olaylara birer örnektir. Uygulama içinde kullanılan her form ve kontrolün kendine has olayları vardır. Mesela, **Form.Hide** metodu gerçekleştiğinde, form Deactivate ve VisibleChanged olaylarını gerçekleştirir. Olaylar gerçekleştiğinde programınızın ne yapması gerekeceğini belirtmek için "Event Handler" (Olay Yönlendirici) oluşturabilirsiniz. Olay yönlendiriciler, gerçekleşen olaylara cevap üreten metotlardır. Tasarım ekranında herhangi bir kontrol için Olay Yönlendirici eklemek için ilgili kontrole çift tıklanır ve kod ekranında gerekli olaylar yazılır. Aşağıdaki program kodu Fare kontrolünün Click olayını göstermektedir.

```
private void button1_Click(object sender, EventArgs e)
        {
            //bu alana başka eventler (olay) eklenebilir
            MessageBox.Show("Butonun Click olayı gerçekleşti");
        }
```



Resim1.7: Formun Click (tıklatma) olayı

Programınızı çalıştırıp, butona tıkladığımızda butonun "click" olayı gerçekleşecek ve kod satırında da görüldüğü gibi ekrana bir mesaj verecektir.

UYGULAMA FAALİYETİ

İşlem Basamakları	Öneriler		
Form1 Image: Constraint of the second se	 Yandaki şekilde gördüğünüz gibi bir form tasarlayınız. Form içinde kullanacağınız öğelerin hizalanmalarına dikkat ediniz. Mesela TextBox'ların aralarındaki mesafelerin aynı olması için bu öğeleri seçtikten sonra, Format menüsündeki Vertical Spacing seçeneğini kullanabilirsiniz. 		
Projenize kendi isminizde bir form ekleyiniz. Formunuzun boyutu 200x200 pixel olsun.	Formun özellikler penceresini kullanabilirsiniz.		
Programınızın ilk açılışında aktif olacak formu ayarlayınız.	Main() fonksiyonu size bir şeyler hatırlatabilir.		
Form1 TÜRKİYE CUMHURİYETİ NÜFÜS CÜZDANI SERi Seri Numarası No T.C.Kimik No Soyadı Adı Baba Adı Ana Adı Doğum Yeri Doğum Yeri Doğum Yeri Doğum Yeri Doğum Yeri	Formunuzun arka plan rengini açık mavi olarak ayarlayınız ve programınız çalıştırıldığında formunuzun ekrana ortalanmış olarak gelmesini sağlayınız. Her zamanki gibi özellikler ekranı size yardımcı olacaktır.		

ÖLÇME VE DEĞERLENDİRME

OBJEKTİF TESTLER (ÖLÇME SORULARI)

Aşağıdaki sorulardan; sonunda parantez olanlar doğru / yanlış sorularıdır. Verilen ifadeye göre parantez içine doğru ise "D", yanlış ise "Y" yazınız.

- 1. Form kavramı yerine pencere kavramı da kullanılabilir. ()
- 2. Windows formunun öğeleri System.Windows.Forms isim alanında bulunmaktadır. ()
- 3. Form içinde kullanılan yazı tiplerinin hiçbir önemi yoktur. ()
- **4.** Main() fonksiyonu içine hangi formun ismi yazılırsa ekrana ilk olarak o form gelir. ()
- 5. Form.Hide() belirtilen bir formu kapatır. ()
- 6. Opacity ayarı, formun yazı tipleri ile alakalıdır. ()

DEĞERLENDİRME

Cevaplarınızı cevap anahtarı ile karşılaştırınız. Doğru cevap sayınızı belirleyerek kendinizi değerlendiriniz. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt yaşadığınız sorularla ilgili konuları öğrenme faaliyete geri dönerek tekrar inceleyiniz.

ÖĞRENME FAALİYETİ–2

AMAÇ

Uygun ortam sağlandığında kullanıcı arabirimi nesneleri ile çalışabileceksiniz.

ARAȘTIRMA

Öğrenme faaliyetinden önce aşağıdaki araştırma faaliyetlerini yapmalısınız.

- Program pencereleri üzerinde bulunan öğeleri araştırınız.
- Bu öğelerin programın kullanılabilirliği yönünden önemini araştırınız.
- Programları kullanırken en sık başvurduğunuz kısa yollar nelerdir? Araştırınız.

2. KULLANICI ARABİRİMİ NESNELERİ

2.1. Komponent ve Kontrolleri Kullanma

Kontroller görsel arabirimin ikinci önemli elemanlarıdır. Bu grafiksel araçlar –form kontrolleri olarak da adlandırılır– uygulamanın işlevselliğini arttırır. Araçlar Visual Studio ToolBox'ından eklenir. Butonlar ve metin kutuları gibi kontroller, kullanıcıdan bazı bilgileri alıp gerekli yerlere taşır. Diğer kontroller daha karmaşık uygulamalardan oluşan projeler için kullanılmaktadır. Komponentler de kontrollere benzer. Aralarındaki temel fark, kontroller görsel bir sunum sağlarken Komponentler bunu yapamaz.

Bu dersten sonra aşağıdakileri yapabileceksiniz:

- Kontrollerin uygulamanızdaki rolünü tanımlayabileceksiniz.
- Kontroller ve Komponentler arasındaki farkı açıklayabileceksiniz.
- Kontrollerin sekme sırasını açıklayabileceksiniz.
- Hangi kontrollerin diğer kontrolleri de ihtiva edeceğini ve bunların nasıl kullanılacağını tanımlayabileceksiniz.
- Gömülme (docking) ve demir atma (anchoring) terimlerini tanımlayabilecek ve kontrollerle nasıl kullanılacağını açıklayabileceksiniz.
- Kontrolleri dinamik olarak formunuza nasıl ekleyebileceğinizi açıklayacaksınız.
- Araç kutusundan (ToolBox) kontrolleri ekleme aşamalarını tarif edebileceksiniz.
- Kontroller için olay yönlendiricinin (Event Handler) nasıl eklenebileceğini açıklayabileceksiniz.
- Extender'ın ne anlama geldiğini ve nasıl kullanıldığını açıklayabileceksiniz.

2.1.1. Kontrollerle Çalışmak

Visual Studio, uygulamanızı geliştirmek için size önceden belirlenmiş çeşitli kontroller sunar. Kontroller formlarda kullanılmak üzere geliştirilmiştir. Mesela, buton kontrolünü ele alalım. Butonlar form üzerine yerleştirilir ve kendisi için yazılan olaya (ekrana mesaj verme gibi) göre işlevini gerçekleştirir. Butona tıkladığınız zaman Event Handler (Olay Yönlendirici) devreye girerek "click" olayına cevap verir ve ilgili kodu çalıştırır. Etiket (Label) ve resim kutusu (PictureBox) kontrolleri esasında kullanıcıya bilgi gösterir. Diğer kontrollerden metin kutusu (TextBox) ve liste kutusu (ListBox) nun 2 görevi vardır. Bunlar, hem bilgi gösterir hem de kullanıcının bilgi girişi yapmasına izin verir.

Kontroller tasarım ekranında iken eklenebilir. Tasarım ekranı size formun grafiksel durumu sunduğu gibi, çalışma zamanında (RunTime) formun nasıl görüneceğini de gösterir. ToolBox, Fare yardımıyla kontrolleri seçmenizi ve tasarım ekranına getirilmesini sağlar. Uygulamanıza eklediğiniz kontroller kodlarla ilişkilendirilmiştir. Kontrolleri Fare yardımıyla formun istediğiniz bir alanına sürükleyip bırakabilirsiniz. Yine Fare yardımıyla çoğu kontrolü istenen ölçüde boyutlandırabilirsiniz.

2.1.1.1. Uygulamaya Kontrol Eklemek

- > ToolBox'tan eklemek istediğini kontrolü seçiniz.
- Kontrolün form üzerinde olmasını istediğiniz alana tıklayın ve farenin sol tuşunu basılı tutup istediğiniz boyuta geldiği an bırakınız.

Başka bir yöntem olarak da şunu yapabilirsiniz. ToolBox'taki kontrole çift tıklayın. Bu şekilde kontrol, varsayılan boyutta formun üzerine yerleşir.

Fare ile kontrolün boyutlandırmasını istediğiniz gibi yapınız. Bu arada kontrolü yön tuşları yardımıyla hareket ettirebilirsiniz.



Resim 2.1: Uygulamaya yeni bir kontrol ekleme

Properties (Özellikler) penceresi, form üzerinde bulunan kontrollerden hangisi seçili ise onunla ilgili bilgileri görmenizi ve düzenlemenizi sağlar.

2.1.1.2. Kontrolün Özelliklerini Değiştirme

Kontrole sağ tık yapınız ve özellikleri seçiniz. Aynı zamanda kontrolü seçtikten sonra F4'e basarak da aynı işlemi yapabilirsiniz.

Properties	→ ╄ ×		
button1 System.Window	s.Forms.Button 👻		
2↓ □			
🗄 Font	Microsoft Sans Serif; 8,25		
ForeColor	ControlText		
Image	(none)		
ImageAlign	MiddleCenter	🚂 Form2	
ImageIndex	(none)		
ImageKey	(none)		
ImageList	(none)		
RightToLeft	No		
Text N	button1	QO	
TextAlian ゆ	MiddleCenter	o button1	þ
Text		0	
The text associated with the	he control.		

> Özellikler (Properties) penceresinden uygun değerleri seçiniz.

Resim 2.2: Kontrollerin özelliklerini değiştirme

2.1.1.3. Birden Çok Kontrolün Özelliklerini Değiştirme

- Özelliğini değiştirmek istediğiniz kontrolleri Fare yardımıyla seçiniz. Alternatif olarak, kontrol tuşunu basılı tutup fare ile tıklarsanız birden fazla kontrolü seçebilirsiniz.
- Seçimi yaptıktan sonra, properties penceresinden gerekli ayarları yapabilirsiniz.

Komponentler de ToolBox'da yer alır. Daha önce de bahsedildiği gibi komponentlerin kontrollerden tek farkı görsel olmayışlarıdır. Mesela "*Timer*" komponenti belirli zaman aralığında istenen olayı gerçekleştirir. Görsel olmadığı için design (tasarım) ekranında görünmez. Bu elemanlar, formun alt kenarına yakın bir yerde, *Komponent Bölmesi* (Component Tray) adı verilen bir alanda tutulur. Komponentler de kontroller gibi seçildikten sonra, özellikler penceresinden gerekli ayarları yapılabilir.

Form2	button1	
🕐 timer:	I 🐺 backgroundWorker1	N

Resim 2.3: Kontrol ve Komponentlerin form üzerine yerleşimi

2.1.1.4. Kontrollerin Sekme Sırasını (Tab Order) Ayarlamak

Uygulamanız kullanıcılar tarafından çalıştırıldığında, bir kontrolden diğerine hızlı geçiş yapabilmek için Sekme (Tab) tuşunu kullanabilir. Sekme sırasını ayarlamak, ilgili kontrolü odaklamak (focus) veya merkeze almak anlamına gelir. Sekme sırası, özellikler penceresindeki *TabIndex* öğesiyle ayarlanır. Sıralamayı ayarlamak için buradaki değer değiştirilir. TabIndex değeri düşük olan kontroller her zaman öncelikli olarak odaklanır ve değer arttıkça diğer kontroller odaklanır.

🖶 Form2			_ 🗆 ×
ADRES	KAYIT DEFTER	i	
Adı Soyadı Adresi S			
, Temi	zle	Sil	

Resim 2.4: TabIndex değerini ayarlamak

Yukarıdaki resimde bulunan kontrollerden, soyadı etiketinin yanındaki metin kutusunun TabIndex'i en düşük değerli seçilmiştir. Bu yüzden program çalıştırıldığı anda kürsör önce TabIndex değeri küçük olan kontrole odaklanmıştır. Sekme tuşuna basıldığında verilen TabIndex değerlerine göre, sırasıyla diğer kontrollere odaklanma gerçekleştirilir.

Bazı kontrollerin focus (odaklanma) olayı olmadığı için TabIndex özelliği de mevcut değildir.

2.1.1.5. Kapsayıcı Kontroller (Container Control)

Bazı kontroller kendi içlerinde diğer kontrolleri de barındırabilir. Kapsayıcı kontroller içinde Panel, GroupBox ve TabControlleri bulunur. Bu kontrolleri kullanarak, mantıksal bir düzen içinde bulunan kontrolleri bir arada tutabilirsiniz. Mesela, birbiriyle ilişkili birden çok RadioButton kontrolünü GroupBox kontrolü içine alabilirsiniz. GroupBox kontrolünü kullanarak kontrol içinde yer alan kontrolleri programlayabilirsiniz.

Kapsayıcı kontrolün özelliğini değiştirdiğiniz zaman, bu kontrol içinde yer alan diğer kontroller de etkilenir. Örneğin, GroupBox'ın *Enabled* özelliğini *False* yaparsanız, diğer kontroller de *disable* olmuş olur. Benzer şekilde, kullanıcı arabirimi ile alakalı BackColor, ForeColor, Visible gibi özellikleri değiştirdiğinizde, kapsayıcı kontrolün de özellikleri değişir. Bu kullanıcı arabiriminizi kolayca uyumlu bir hale sokar.

Görsel elemanları içinde barındıran bir kapsayıcı kontrolün özelliklerini değiştirdiğinizde, içindeki diğer kontroller de etkilenir ama siz manuel (elle) olarak da bu kontrolleri değiştirebilirsiniz.

🛃 GroupBox	
groupBox1	
-	e-mail gonderilsin
	Şartları Kabul Ediyorum
	Sil Kaydet

Yandaki resimde görüldüğü gibi 2 adet CheckBox ve 2 adet buton GroupBox'ın içine yerleştirilmiştir. Siz burada GroupBox'ı seçip özelikler penceresinden ForeColor seçeneği değiştirirseniz (mesela rengi kırmız seçerseniz) GroupBox içindeki tüm kontrollerin ön plan renginin kırmızı olduğunu görürsünüz. Fakat burada ön plan renginin kırmızı olmasını istemediğiniz bir kontrol varsa, onu elle kendiniz değiştirebilirsiniz.

Resim 2.5: GroupBox uygulaması

2.1.1.6. GroupBox ve Panel Kontrollerini Kullanma

GroupBox ve Panel kontrolleri birbirine oldukça benzerdir. İkisi de kontrolleri mantıksal ve fiziksel olarak bir arada tutar. Bu kontroller formun fiziksel olarak bölümlere ayrılması olarak düşünülebilir. Panel ya da GroupBox'daki özellikleri değiştirdiğiniz zaman kapsamlarında bulunan tüm kontroller de değişir. Panel ya da GroupBox'daki kontroller tasarım ekranının istenen bir yerine taşınabilir. Çalışma anında kapsayıcı kontrolün *Enabled* özelliğini *false* yaparsanız tüm grubu pasif duruma getirmiş olursunuz.

GroupBox kontrolünün Text özelliğini kullanarak kontrolün Caption (Başlık) özelliğini değiştirebilirsiniz. Panel kontrol *kaydırılabilir* kontroldür fakat Caption özelliği yoktur. AutoScroll özelliğini aktif hale getirirseniz, paneldeki kaydırma çubuklarını kullanabilirsiniz.

2.1.1.7. TabControl'ü Kullanma

Üzerinde sekmeler (Tab) olan bir grup kontroldür. TabControl, diğer kontrolleri içinde barındıran bir takım sekme sayfalarıdır. TabControl, herhangi bir uygulamanın özelliklerini gösterebilir. Bu özellikler uygulamanın kendine has özellikleridir.

TabControl'ün en önemli özelliği sekme sayfalarıdır. Sekme sayfaları, sekme sayfası kontrollerinden oluşan bir koleksiyondur ve her sekme sayfasının kendine ait özellikleri vardır. Koleksiyonlar, benzer objelerin mantıksal bir düzen içinde bir araya gelmiş halidir.

🛃 Form2		
	Kullanıcı Bilgileri Sistem Bilgileri Ağ Bilgileri Ağ Bilgileri Ağ Bilgileri Adı	

Resim 2.6: TabControl içindeki sekme sayfaları

Yukarıdaki şekilde, içinde sekme sayfaları bulunan bir TabControl örneği görüyorsunuz. TabControl içindeki sekme sayfalarının özelliğini değiştirmek için, TabControl seçildikten sonra özellikler penceresinin TabPages kısmında bulunan Collections'a tıklanır. Aşağıdaki gibi bir ekran karşımıza gelir.

TabPage Collection Editor						<u>? ×</u>
Members:		tab	Page1 properties:			
0 tabPage1	+	•	2↓ 🖻			
2 tabPage3	+		AccessibleDescrip			
			AccessibleName			
			AccessibleRole	Defa	ault	
		Ξ	Appearance			
			BackColor		Control	
			BackgroundImage		(none)	
			BackgroundImage	Tile		
			BorderStyle	None	3	
			Cursor	Defa	ault	
		Ŧ	Font	Micro	osoft Sans Serif;	
			ForeColor		ControlText	
			RightToLeft	No		
Add <u>R</u> emove			Text	Kull	anıcı Bilgileri	-
			0	<	Cancel	

Resim 2.7: Sekme sayfalarının ayarlarının yapılması

Resimdeki ekrandan da görüldüğü gibi, her sekme sayfasının kendine ait özellikleri bulunmaktadır. Hangi sekme sayfası ile ilgili ayar yapacaksanız, onu seçip sağ taraftaki özellikler kısmından istenilen ayarları yapabilirsiniz.

2.1.1.8. Anchoring (Demir Atma) ve Docking (Gömülme) Kontrolleri

Anchor ve Dock, kontrollerin form içinde nasıl hareket edeceklerini veya davranacaklarını gösteren özelliklerdir. Anchor özelliği, kontrolle formun köşeleri-kenarları arasındaki mesafeyi sabit tutmaya yarar. Böylece kullanıcı çalışma anında formu boyutlandırmaya kalktığında, kontrolün formla daha önceden belirlenen uzaklığını sürdürmesini sağlar. Dock özelliği ise, herhangi bir kontrolü formun belirlenen tarafına tamamen iliştirmek için kullanılır. Form yeniden boyutlandırıldığında, kontrol de ona göre kendini ayarlar.

2.1.1.9. Anchor Özelliğini Kullanmak

Anchor özelliği, herhangi bir kontrolün özelliğini ayarladığımız gibi özellikler penceresinden ayarlanabilir. Özellikler penceresinden Anchor'u seçiniz. Aşağı yönlü oka bastığınızda mevcut özelliklerin bulunduğu pencere açılır.

Properties	Р Х
TextBox2 System.Windows.Forms.Text	
AccessibleName	
AccessibleRole	Default
AllowDrop	False
Anchor	Top, Left 🛛 👻
AutoSize	
BackColor	-
Anchor The anchor of the co to which edges of th	

Resim 2.8: Anchor özelliklerini seçmek

Kontrolün hangi köşeye veya köşelere doğru sabit kalmasını istiyorsanız ilgili seçeneği tıklayınız. Varsayılan değer, Top (üst), Left (sol) olarak ayarlıdır. Bunun nedeni ise, kullanıcıların genelde sağdan sola ve üste doğru pencereyi boyutlandırma alışkanlıklarıdır.

🔡 For	m1		
		_	
	button1		ł
	O	K	4

Resim 2.9: Anchor özelliğinin uygulaması

Yukarıdaki resimde olduğu gibi, siz formunuzu sağ alt köşeden sol üst köşeye doğru boyutlandırdığınızda, buton ile formun sol kenarı ve üst kenarı arasındaki mesafe sürekli sabit kalır.

2.1.1.10. Dock Özelliğini Kullanmak

Docking (yapışma-gömülme) özelliği, bir kontrolü bir üst kontrolün herhangi bir kenarına iliştirmektir. Üst kontrol olarak genelde formlar kullanılır, fakat üst kontroller *Tab* ve *Panel* kontrol gibi kontroller de olabilir. Mesela, bir menünün formun üst kenarına iliştirilmesi örnek olarak verilebilir.

Tasarımın herhangi bir anında *Dock* kontrolünü kullanabilirsiniz. Dock özelliğini seçtikten sonra, kontrolün Docking karakteristiğini seçebileceğiniz bir arabirim karşınıza gelir.

F	roperties	- ₽ ×				
button1 System.Windows.Forms.Button						
G	2↓ 💷 🗲 📄					
E	Focus	A				
	CausesValidation	True				
E	Layout					
	Anchor	Top, Left				
	AutoSize	False				
	AutoSizeMode	GrowOnly	li			
	Dock	None 🗾		🛃 Form1		
Ð	Location				h 1	
Ð	3 Margin				Dutton	
Ð	MaximumSize		1			
I	Dock					
	Defines which borders of the container.					
L		None				

Resim 2.10: Kontrollere Docking özelliği uygulamak

Resimden de görülebildiği gibi, kontrolün arabiriminizle en uygun pozisyonu nasıl olacaksa Dock özelliğinden istediğinizi seçiniz. Mesela, kontrolünüzü formun üst kenarına gömmek-yapıştırmak isterseniz ekrandan en üstteki bölümü tıklayabilirsiniz. Bu işlemden sonra kontrolünüz Resim 10... daki gibi olur.

2.1.1.11. ToolBox'a (Araç Kutusu) Yeni Kontroller Eklemek

Kullandığımız kontroller. NET Framework kütüphanesinde bulunan kontroller ile sınırlı değildir. Üçüncü parti (Third-party) yazılımların yardımıyla ToolBox'a yeni kontroller ekleyebilirsiniz.

ToolBox'a yeni bir kontrol eklemek için:

1. ToolBox tabına geçiniz.

ToolBox görünmüyorsa, View (görünüm) menüsünden ToolBox'ı seçerek ekrana alabilirsiniz.

2. Tab içinde herhangi bir yerde sağ tık yapınız. Gelen menüden *Choose Items*'ı (Öğe seç) seçiniz.

- 3. Choose ToolBox Items isimli bir pencere ekrana gelir. İki bölmeli olan bu pencerenin sağ bölmesinde hâlihazırda ToolBox'da bulunan öğeler ve eklenebilecek öğeler yer alır. Sisteminizde kayıtlı olmayan bir öğe varsa yanındaki kutucuğu işaretleyip Tamam butonuna basınız.
- 4. Eğer eklenmek istenen kontrol başka medyalar (floppy, CD ..vs) içinde yer alıyorsa, pencerenin altındaki göz at'a basılarak ilgili dosya seçilir ve tamam'a tıklanır.

2.1.1.12. Kontroller için Olay Yönlendirici (Event Handler) Oluşturmak

Event'ler programın çalışması esnasında olacak olayları belirtir. Her kontrolün kullanıcı etkileşimine uygun bir olayı mevcuttur. En iyi bilinen örnek butonun *click* (tıklama) olayıdır. Butona tıklandığında program *Buton.Click* olayını çalıştırır ve hangi metodun uygulanacağına karar verir. Şayet birden fazla metot varsa bunlar çalıştırılır. Bu metotlar *Event Handlers* (Olay Yönlendiriciler) olarak isimlendirilir.

Uygulamanızda kullanıcının girdiği veriye cevap veren bir *Event Handler* oluşturabilirsiniz. Her kontrolün varsayılan bir olayı (event) vardır. Örneğin butonun varsayılan olayı Click iken CheckBox (seçme kutusu)'ın varsayılan olayı *CheckChanged*'dir. Tasarım ekranında varsayılan olay için Event Handler oluşturmak için kontrole çift tıklama yapabilirsiniz. Çift tıklamadan sonra, karışınıza gelecek olan kod ekranından uygun Event Handler'ı seçebilirsiniz.

Kontroller değişik amaçlar için kullanılabilecek başka olaylara da sahiptir. *MouseOver* olayını kullanarak, farenin kontrol metni üzerinde hareket etmesiyle bazı işlemleri yapmasını sağlayabilirsiniz.

C#'da Event Handler oluşturmak için:

- Tasarım ekranında Event Handler oluşturacağınız kontrolü seçiniz.
- Özellikler ekranındaki Events butonuna tıklayınız. Mevcut Event'ler karşınıza gelecektir.
- Hangi handler (yönlendirici)'ı uygulayacaksanız ona çift tıklayınız.
- Code ekranında Handler eklemek için kontrolden sonra noktaya basarsanız karşınıza mevcut olaylar listelenir. İlgili olayı buradan seçebilirsiniz.

private (void button1_Click_1	(object sender	, EventArgs	e)
butt	on1.			
}	Dock Dockchanged DobragDrop DoubleClick DragDrop DragEnter DragLeave DragLeave DragLeave DrawToBitmap Enabled	*		

Resim 2.11: Buton kontrolünün çift tıklama olayı

2.1.2. Fare Etkileşimleri

Windows formlarındaki kontroller, fare ile etkileşimli çalışabilir. Örneğin *click* olayı ya da farenin bir kontrol üzerinde hareket etmesi gibi. *Click* ve *DoubleClick* olayları sırasıyla, kontrolün farenin tıklanması ve çift tıklanmasına verdiği cevaplardır. Bu kontroller genelde kullanıcının yaptığı seçime bağlı olarak butona tıklanması olayıdır. Bu kontrollerin olay yönlendiricileri, *EventArgs* sınıfının örnekleri olarak geçmektedir.

Kontroller aynı zamanda fare işaretçisinin bazı hareketlerine de cevap verebilir. Bu olaylar aşağıdaki tabloda açıklanmıştır.

Event (Olay)	Tanımlama	EventArgs'ın Tipi
MouseEnter	Kontrole fare ile giriş yapıldığında yerine getirilecek işleri gösterir.	System.EventArgs
MouseMove	Fare işaretçisinin kontrol üzerinde hareket ettirilmesinden sonra yapılacak işleri gösterir.	System.MouseEventArgs
MouseHover	Fare işaretçisinin kontrol üzerinde dolaşmasından sonra yapılacak işleri gösterir.	System.EventArgs
MouseDown	Fare işaretçisinin kontrol üzerinde hareket ettirilmesi ve butona tıklanmasından sonra yapılacak işleri gösterir.	System.MouseEventArgs
MouseWheel	Fare işaretçisinin kontrolün merkezinde dairesel hareketler yapmasından sonra yapılacak işleri gösterir.	System.MouseEventArgs
MouseUp	Fare işaretçisi kontrolün üzerinde iken butonun bırakılmasından sonra yapılacak işleri gösterir.	System.MouseEventArgs
MouseLeave	Fare işaretçisi kontrol üzerinde hareket ederken yapılacak işleri gösterir.	System.EventArgs

MouseEnter, MouseHover ve MouseLeave olayları, fare işaretçisinin kontrolün herhangi bir bölgesinde iken yapılacak işleri gösterir. MouseMove, MouseDown, MouseWheel olayları ise kullanıcı ile uygulama arasında daha fazla etkileşim olmasını sağlar. Bu olayların tümü, MouseEventArgs olay yönlendiricisinin örnekleridir.

UYGULAMA FAALİYETİ

İşlem Basamakları	Öneriler
Kontrol ve komponent kavramlarını tanımlayınız.	Modül notlarına bakabilirsiniz.
Program içinde kullanılan kontrollere örnek veriniz.	 Her gün kullandığınız programları düşününüz.
Eklediğiniz bir kontrolün özelliğini nasıl değiştirirsiniz.	Modül notlarına bakabilirsiniz.
Kontrollerin TabIndex i neyi ifade eder?	Visual Studio'nun yardımını kullanmayı deneyiniz.
ToolBox'a yeni bir komponent ekleyeniz.	Visual Studio'nun yardımını kullanmayı deneyiniz.

ÖLÇME VE DEĞERLENDİRME

OBJEKTİF TESTLER (ÖLÇME SORULARI)

Aşağıdaki sorulardan; sonunda parantez olanlar doğru / yanlış sorularıdır. Verilen ifadeye göre parantez içine doğru ise "D", yanlış ise "Y" yazınız. Şıklı sorularda uygun şıkkı işaretleyiniz.

- 1. Formlarda en çok kullandığımız öğe olan butonlar, bir kontroldür. ()
- 2. Form içinde kullanılan komponentler görsel bir sunum sağlar. ()
- **3.** Dock özelliği, herhangi bir kontrolü formun belirlenen tarafına tamamen iliştirmek için kullanılır. ()
- 4. ToolBox'a yeni komponentler eklenemez. ()
- 5. Label kontrolünün metin özelliğini değiştirmek için Label.Text metodu kullanılır. ()
- 6. Aşağıdakilerden hangisi fare olaylarından biri değildir.?
 A) MouseMove
 B) MouseEnter
 C) MouseActivate
 D) MouseWheel

DEĞERLENDİRME

Cevaplarınızı cevap anahtarı ile karşılaştırınız. Doğru cevap sayınızı belirleyerek kendinizi değerlendiriniz. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt yaşadığınız sorularla ilgili konuları öğrenme faaliyete geri dönerek tekrar inceleyiniz.

ÖĞRENME FAALİYETİ–3

AMAÇ

Uygun ortam sağlandığında menüler ve kullanıcı girişi oluşturabileceksiniz.

ARAȘTIRMA

Öğrenme faaliyetinden önce aşağıdaki araştırma faaliyetlerini yapmalısınız.

- Görsel olmayan programlama dilleriyle (basic, pascal) nasıl menü hazırlandığını araştırınız.
- Menü kullanılmayan bir programın dezavantajları neler olabilir? Araştırınız.
- Microsoft Word programı içinde en sık kullanılan kısa yolları araştırınız.

3. MENÜLER

3.1.Menüleri Kullanma

Menüler, uygulamaların önemli araçlarına ve fonksiyonlarına kolayca erişimi sağlar. Uygun bir menü planlaması ve tasarımı programınızın işlevselliğini artırır.

Bu dersten sonra bunları yapabileceksiniz:

- > Uygulamanızın arabirimini tasarlamada menülerin rolünü açıklayabileceksiniz.
- MenuStrip öğesiyle ana menü tasarımının aşamalarını öğreneceksiniz.
- ContextMenu öğesiyle context (içerik) menüsü tasarımının detaylarını tanımlayabileceksiniz.
- Menü elemanlarının nasıl aktif ya da pasif duruma getirileceğini açıklayabileceksiniz.
- Menü öğeleri için nasıl kısa yol tuşları oluşturabileceğinizi açıklayabileceksiniz.
- Menü öğeleri üzerinde seçme kutusu, radio düğmesi gibi elemanları nasıl oluşturabileceğinizi açıklayabileceksiniz.
- Menü öğelerinin nasıl gizleneceğini açıklayabileceksiniz.
- Menülere dinamik olarak nasıl öğe eklenebileceğini açıklayabileceksiniz.
- > Dinamik olarak bir menüyü nasıl kopyalayacağınızı açıklayabileceksiniz.

Menüler, bilinen üst seviye fonksiyon ve komutların programa eklenmesine izin vererek uygulamanın kolayca anlaşılmasını sağlar. Tasarımı iyi yapılmış bir menü, uygulamanın tüm işlevselliğini mantıksal bir düzen dâhilinde kullanıcıya sunmalıdır. Tersine kötü tasarlanmış menüye sahip bir program sadece zorunlu olunduğu zaman kullanılacak, bu da iyi bir uygulamada olmaması gereken bir özelliktir.

İyi bir menü tasarımında, uygulamanın mantıksal akışı göz önünde bulundurulmalıdır. Menü öğeleri, ilişkili öğeler dikkate alınarak gruplandırılmalıdır. Menülere ulaşım için kısa yol tuşlarının bulunması programın kullanılabilirliğine önemli katkı sağlar.

3.1.1. Tasarım Aşamasında Menü Oluşturmak

Tasarım aşamasında ana menü (MainMenu) oluşturmak için *MenuStrip* öğesi (Component) kullanılır. *MenuStrip* öğesi, *MenuItem* kontrolünün koleksiyonunu saklar ve yönetir. Formlarınız için *MenuStrip* öğesiyle hızlı bir şekilde menü tasarımı yapabilirsiniz.

MenuStrip öğesi aşağıdakileri yapmanıza izin verir:

- Yeni menü ve menü çubukları oluşturabilirsiniz.
- Var olan menüye yeni öğeler ekleyebilirsiniz.
- Özellikler penceresini kullanarak, menü ve menü öğelerinin özelliklerini değiştirebilirsiniz.
- Menü öğeleri için olay yönlendirici (event handlers) ler ekleyebilirsiniz.

Uygulamanıza yeni bir menü eklemek için *MenuStrip* öğesini formunuza eklemelisiniz. Bunun için araç kutusundan (ToolBox) MenuStrip öğesini seçip formunuzun üstüne sürükleyiniz. MenuStrip öğesi formun üstüne geldikten sonra menüyü istediğiniz gibi özelleştiriniz. Bunun için *Type Here* yazan kutucuğa tıklayınız. Alt menü oluşturmak için öğenin sağına tıklayınız.



Resim 3.1: MenuStrip öğesiyle menü oluşturmak

Menüye yeni bir öğe eklendiği zaman, program MenuItem nesnesinin bir örneğini oluşturur. Özellikler penceresinde MenuItem nesnesinden oluşturulan her örneğin özellikleri ve üyeleri vardır. Özellikler penceresindeki Text özelliği, programın çalışması esnasında gösterilen metni ifade eder. Name özelliği ise, objenin referansının kod kısmına nasıl gönderileceğini gösterir. Menü öğelerine ayırıcı çubuk (seperator bar) eklemek için,

- Önce menü öğesini seçiniz.
- Daha sonra menü öğesi üzerinde sağ tık yapınız ve gelen pencerede Insert>Seperator'u seçiniz.

🔛 Menu()lusturn	na						
Dosya	Düzen	Gà	órünüm	Yardım		Туре	e Here	
	ke	es						
	ka	opyala						
	ya	apıştır						
		😰 View Code						
			Insert		•		MenuItem	
		Ж	Cut				_CemboBox	
			Сору		- (Separator	\longrightarrow
		8	Paste			abi	TextBox	
		×	Delete					

Resim 3.2: Menü öğesine ayırıcı çubuk ekleme

3.1.2. Menü Erişimi ve Kısa Yol Tuşları

Menülere erişmek ve kısa yol tuşlarını kullanabilmek için klavye erişimini aktif hale getirebilirsiniz.

Erişim Tuşları

Erişim tuşları, klavyenizdeki Alt tuşu yardımıyla daha önceden belirlenmiş harflere basarak menülere ulaşmanızı sağlar. Menü açıldığında alt tuşu ve belirtilen tuşa basarak ilgili menü öğesini çalıştırabilirsiniz. Mesela, çoğu programda alt tuşu ile birlikte F tuşuna basıldığında (Alt+F) Dosya (File) menüsü aktif olmaktadır. Formlardaki ilgili menü öğesinde, belirtilen erişim tuşunun altı çizilir.

Menü öğeleri farklı gruplarda olmak şartıyla, aynı erişim tuşunu birden fazla menü öğesinde kullanabilirsiniz. Örneğin, File menüsünde Alt+C tuş kombinasyonunu programı kapatmak için kullanabileceğiniz gibi, Edit menüsünde kopyalama işlemi için aynı kombinasyonu kullanabilirsiniz. Aynı tuş kombinasyonunu bir menü içindeki farklı öğelere uygulamaktan kaçınınız. Eğer bunu yaparsanız, program bu öğeler arasında geçiş yapmanıza izin verir ancak ENTER tuşu ile bu öğeleri çalıştıramazsınız.

Menü öğesine erişim tuşu atamak için:

- > Tasarı ekranında erişim tuşu eklemek istediğiniz öğeyi seçiniz.
- Erişim tuşu olarak belirlemek istediğiniz karakterin önüne & (Ampersand) işareti koyunuz.

Kısa yol tuşları

- Kısa yol tuşu eklemek istediğiniz menü öğesini seçerek aktif hale getiriniz.
- Özellikler penceresindeki ShortcutKeys anahtarını seçiniz.
- Sağ taraftaki açılır menüden uygun kısa yol kombinasyonunu seçiniz.

3.1.3. Menü Öğelerinin Olaylarını Kullanma

Diğer kontrollere (buton, TextBox) Event Handler (olay yönlendirici) eklediğiniz gibi, menü öğelerine de Event Handler ekleyebilirsiniz. En sık kullanılan olay Click olayıdır. Click olay yönlendiricisi, menü öğesi tıklandığı zaman gerçekleşmesi istenen kodu ihtiva etmelidir. Bu kod aynı zamanda, kısa yol tuş bileşimi kullanıldığında da çalışmalıdır. Select olayı, bir menü öğesi fare ya da klavye kısa yolu ile seçildiğinde işlev görür.

3.1.4. İçerik Menüsü (Context Menu) Oluşturma

İçerik menüleri genellikle, öğe üzerinde sağ tık yapıldığında açılan menüler için kullanılır. İçerik menüleri, Araç kutusundaki ContextMenuStrip öğesi (component) kullanılarak eklenir. Bu öğeler, MenuStrip (ana menü) oluşturmada izlenen yöntemler gibi uygulanabilir.

İçerik menüleri, ana menünün birçok özelliğini gösterir. Bunlar için kısa yol tuşları atayabilirsiniz ama ana menüde olduğu gibi erişim tuşu veremezsiniz. Bir formdaki kontrolü içerik menüsü ile ilişkilendirmek için kontrolün özellikler penceresindeki *ContextMenuStrip* anahtarını kullanmalısınız.

ContextMenu oluşturmak için:

- Araç kutusundaki ContextMenuStrip öğesine çift tıklayınız ya da formun üstüne sürükleyiniz. ContextMenuStrip öğesi formun üst kenarına yerleşir.
- Menü içerisinde olması gereken öğeleri, ana menüdeki gibi ekleyiniz.
- Eğer gerekiyorsa özellikler menüsünden bir olayı (event) seçiniz.
- İçerik menüsüyle ilişkilendireceğiniz kontrolü seçiniz. Özellikler penceresindeki ContextMenuStrip anahtarının sağında bulunan açılır liste kutusundan ilgili içerik menüsü öğesini seçiniz. Artık uygulamanızın çalışma anında (RunTime) ilgili kontrole sağ tık yaptığınızda hazırlamış olduğunuz içerik menüsü görünecektir.

Properties	- ₽	×
label1 System.Windows.Form	ms.Label	-
2↓ 🗉 🗲 🗀		
TextAlign To	pLeft	
UseMnemonic Tru	ue	
UseWaitCursor Fa	alse	
🗆 Behavior		
AllowDrop Fa	alse	
AutoEllipsis Fa	alse	
🗄 ContextMenuStrip 🛛 on	ntextMenuStrip1 💌	
Enabled (non	ne)	
TabIndex cont	textMenuStrip1	
UseCompatibleTextRent ra	iise	
ContextMenuStrip		
The shortcut menu to display (right-clicks the control.	when the user	

Resim 3.3: Kontrolü ContextMenu ile ilişkilendirme ve içerik menüsünün uygulaması

3.1.5. Çalışma Zamanında Menüleri Değiştirmek

Menüleri çalışma zamanındaki şartlara bağlı olarak dinamik bir şekilde yönetebilirsiniz. Mesela programınız belirli bir görevi yerine getiremediği zamanlarda, çağıracağınız bir komutla bu menüyü pasif duruma getirebilirsiniz. Kullanıcıya bir sonraki menü öğesine geçebilmesi için bir seçme onay imi ya da radio buton gibi seçenekler sunabilirsiniz. Menü öğelerini bazen gizlemek isteyebilirsiniz. Çalışma zamanında bir menü öğesi ekleyebilir veya başka bir menü öğesiyle birleştirebilirsiniz.

3.1.6. Menü Komutlarını Aktif ve Pasif Yapma

Her menü öğesi bir Enabled (seçilir kılınmış) özelliğine sahiptir. Bu özellik false yapıldığında, menü öğesi pasif (disabled) olur ve kullanıcıların erişimini engeller. Erişim tuşları ve kısa yol tuş kombinasyonları da pasif olur ve bu öğeler soluk bir şekilde gözükür. Aşağıdaki program kodu, çalışma anında bir menü öğesinin nasıl pasif yapılacağını göstermektedir.

menuItem1.Enabled = false;

3.1.7. Menü Öğelerini Gizleme

Özellikler penceresindeki Visible anahtarı yardımıyla bir menü öğesini gizleyebilirsiniz. Program içinde değişen şartlara göre de menüleri gizlemek isteyebilirsiniz. Aşağıdaki program kodu bir menü öğesinin nasıl gizleneceğini göstermektedir. menuItem1.Visible = false;

Çalışma zamanında gizlediğiniz bir menü, menü çubuğuyla birlikte gizlenir. Dolayısıyla gizlediğiniz menünün alt menüleri de bu süre zarfında erişilmez olur.

3.1.8. Menüleri Kopyalama

Çalışma anında var olan bir menüye ait olan öğeleri çoğaltabilirsiniz. Mesela bir uygulamaya ait Edit menüsünü bir kontrol için içerik menüsü (context menu) olarak kullanmak isteyebilirsiniz. Yeni bir menü öğesi oluşturmak için CloneMenu metodunu kullanabilirsiniz. Bu metot, bir menüyü üyeleriyle birlikte çoğaltmaya yarar. Böylece yeni oluşturulan menüde de eski menüdeki tüm olaylar da (event) yer alır. Aşağıdaki program parçası, bir menü öğesinden yeni bir içerik menüsünün nasıl oluşturulacağını göstermektedir.

```
// Bu örnekte daha önceden var olan bir menü olduğu farz
edilmiştir.
// fileMenuItem ve kontrol myButton olarak çağrılmıştır.
// Yeni bir içerik menüsü bildirimi yapar ve örneğini oluşturur.
ContextMenu myContextMenu = new ContextMenu();
myContextMenu.MenuItems.Add(fileMenuItem.CloneMenu());
// myButton kontrolüne ContextMenu sü atanır.
myButton.ContextMenu = myContextMenu;
```

3.1.9. Çalışma Anında Menü Öğelerini Birleştirme

Bazen tek menü içinde birden fazla menü öğesini göstermek isteyebilirsiniz. MergeMenu metodu, çalışma anında birden fazla menüyü tek bir menü altında birleştirmenizi sağlar. Bu yolla birden fazla MainMenu (ana menü) ve ContextMenu öğelerini diğer menü öğeleriyle birleştirebilirsiniz.

fileMenuItem.MergeMenu(myContextMenu);

Yukarıdaki program kodunda fileMenuItem, myContextMenu öğesi ile birleştirilmiştir.

3.1.10. Çalışma Anında Menü Öğeleri Eklemek

Çalışma anında dinamik olarak bir menü öğesine yeni bir öğe ekleyebilirsiniz. Örneğin bir programda en son açılan dosyaları menüye eklemek isteyebilirsiniz. Oluşturulan yeni menü öğesinin olay yönlendiricisi yoktur.

Çalışma anında yeni bir menü öğesi eklemek için, yeni bir menü öğesi bildiriminde bulununuz ve bunun bir örneğini oluşturunuz.

```
MenuItem myItem;
myItem = new MenuItem("Item 1");
```

Bu program kodunda ise, MenuItem'e Item 1 adında yeni bir öğe eklenmiştir.

3.2. Kullanıcı Girişini Onaylamak (Validating)

Çoğu uygulamada kullanıcılar bazı bilgileri programa girer. Programın icrası sırasında, kullanıcı tarafından girilen bilgilerin daha önceden belirtilen şekilde olması gerekir. Örneğin kullanıcının kendi adresine ait posta kodunu gireceği bir alanı düşünelim. Buraya girilecek bilginin rakamsal olması ve beş karakteri geçmemesi gerekir. Kullanıcı girişini onaylamak-doğrulamak, veri girişi sırasında olabilecek hataları engellediği için, programın daha sağlam olmasını temin eder.

Bu dersten sonra bunları yapabileceksiniz:

- Form düzeyli doğrulama ile alan düzeyli doğrulamanın farkını açıklayabileceksiniz.
- Kontrol metotlarını ve olaylarını kullanarak doğrudan odaklanmanın (focus) nasıl yapılacağını açıklayabileceksiniz.
- Formlarınıza form düzeyli doğrulama uygulayabileceksiniz.
- Formlarınıza alan düzeyli doğrulama uygulayabileceksiniz.

Kullanıcının bilgi girişinde kullanabileceğiniz iki farklı doğrulama tipi mevcuttur. Form düzeyli (form-level) doğrulama ve alan düzeyli (Field-Level) doğrulama. Form düzeyli doğrulama, kullanıcının tüm alanları doldurulmasında sonra devreye girerek kontrol yapar. Mesela, kullanıcı adını, soyadını ve telefonunu girdikten sonra Tamam'a basılır, form düzeyli doğrulama çalışır ve boş alan bırakılıp bırakılmadığını kontrol eder.

Alan seviyeli doğrulamada ise, bilgi girilen tüm alanlar tek tek kontrol edilir. Şahsi bilgilerimizin istendiği bir formu düşününüz. Form içinde yaşadığımız şehrin alan kodunu isteyen bir alana yanlış bir kod girdiğimizde alan düzeyli doğrulama mekanizması kullanıcıyı uyarır ve bir sonraki alana bilgi girişini engeller.

3.2.1. Alan Düzeyli Doğrulama

Bilgi girişi yapılan her alanı kontrol etmek isteyebilirsiniz. Bu doğrulama metodunda, kullanıcının bilgi girdiği her alan kontrol edilir. Bu bölümde alanlara bilgi girişinde kontrolleri nasıl kullanacağımızı göreceğiz. Aynı zamanda, kullanıcının uygun bilgileri girmesini sağlamak için TextBox kontrolünün özelliklerini nasıl ayarlayabileceğimizi de anlatacağız.

3.2.2. TextBox Özelliklerini Kullanma

Kullanıcı girdilerinde en çok kullanılan kontrol TextBox nesnesidir. Bu nesnenin, kabul edilebilir bilgi girişini sağlamak için birçok özelliği vardır. Bu özelliklerden bazıları şunlardır:

- MaxLength
- PasswordChar
- ➢ ReadOnly
- MultiLine

MaxLength Özelliğini Ayarlamak

Bu özellik TextBox'a (metin kutusu) girilebilecek maksimum karakter sayısını belirler. Eğer belirtilen sayıdan fazla karakter girişi yapılırsa program buna izin vermez ve kullanıcıyı uyarır. Bu özellik posta kodu gibi, her zaman karakter sayısı belli olan alanlar için kullanılır.

PasswordChar Özelliğini Kullanmak

Çalışma anında herhangi bir metin kutusuna girdiğiniz verileri gizler. Buradaki karakteri "*" (asteriks) olarak ayarlarsanız, kullanıcının girdiği veri hangi karakterlerden oluşursa oluşsun, metin kutusuna "*" karakterini basar. Bu özellik genelde, parola bilgisi istenen yerlerde kullanılır. Burada "*" karakteri yerine başka karakterler de kullanabilirsiniz.

ReadOnly Özelliğini Ayarlamak

Bu özellikle, kullanıcının metin kutusundan bulunan veriyi değiştirip değiştiremeyeğini belirleyebiliriz. Şayet bu özellik "true" ise, kullanıcı buradaki veriyi değiştiremez. False ise, normal düzenleme yapabilir.

MultiLine Özelliğini Kullanmak

Metin kutusunda birden fazla satır kullanılıp kullanılmayacağını belirler. Şayet özelliği true ise, kullanıcı metin kutusuna birden fazla satır bilgi girebilir ve program sığmayan karakterleri satır başına otomatik olarak atar.

3.2.3. Alan Düzeyli Doğrulamada Olayları Kullanmak

Alan düzeyli klavye olayları size kullanıcının girdiği veriyi anında doğrulama imkânı verir. Kontrol sizden veriyi alır ve aşağıdaki üç olaya neden olur.

- ➢ KeyDown
- ➢ KeyPress
- ➢ KeyUp

KeyDown ve KeyUp

Bu olaylar klavyeden bir tuşa basıldığı zaman yapılacak olan işleri belirtir. Bu olay gerçekleştiğinde, KeyEventArgs örneği içinde tanımlı olan tuş ya da tuş kombinasyonlarına bakılır. Bir metodun KeyDown ve KeyUp olaylarını gerçekleştirebilmesi için, KeyEventArgs'ın parametreleri gereklidir. KeyEventArgs'ın parametreleri aşağıdaki tabloda özetlenmiştir.

Özellik	Tanımlama
Alt	Alt tuşuna basılıp basılmadığına bakarak bir değer alır.
Control	Control tuşuna basılıp basılmadığına bakarak bir değer alır.
Handled	Olayın yürütülüp yürütülmediğine bakarak bir değer verir veya alır.
KeyCode	Bir tuşa basılıp basılmadığını kontrol eder ve numaralı liste döndürür.
KeyData	Alt, Ctrl, Shift tuşlarıyla birlikte diğer tuşlara basıldığında veri döndürür.
KeyValue	KeyData özelliğinin bir değerini döndürür.
Modifiers	Olayın değiştirici bayrağını elde eder ve hangi tuş kombinasyonlarının kullanıldığını gösterir.

Shift Shift tuşuna basılıp basılmadığına bakarak bir değer alır.

KeyUp ve KeyDown olayları genellikle, Alt,Shift ve Ctrl tuşlarına basılıp basılmadığını kontrol etmek için kullanılır. KeyEventArgs'ın özelliklerinden -Alt, Ctrl ve Shift- Boolean (true-false) değer üretir ve bu tuşlara basılıp basılmadığını kontrol eder. Tuşa basıldığında değeri true olur, tuş bırakıldığında ise false olur. Aşağıdaki program parçası, KeyUp olay yönlendiricisini kullanarak alt tuşuna basılıp basılmadığını kontrol eder.

```
private void textBox1_KeyUp(object sender,
System.Windows.Forms.KeyEventArgs e)
{
    if (e.Alt == true) MessageBox.Show("ALT tuşu basılı durumda");
}
```

KeyPress

Kullanıcı ASCII değerine uygun bir tuşa bastığında KeyPress olayı gerçekleşir. Bu tuşlar ASCII tablosu içinde yer alan harfler, rakamlar (a–z, A–Z ve 0–9), ve Enter, Space gibi özel karakterler olabilir. Şayet ASCII tablosunda karşılığı olmayan bir karakter kullanılmışsa KeyPress olayı gerçekleşmez. Ctrl, Alt, Shift ve fonksiyon tuşları bunlara örnek verilebilir.

Bu olayın en çok kullanıldığı yer tuş vuruşlarını durdurma ve onları değerlendirmektir. Bu olay gerçekleştiğinde KeyPressEventArgs' ın parametreleri devreye girer. KeyPressEventArgs'ın örneği, bilgi girişini doğrulamak için kullanılan tuş vuruşları hakkında bilgi ihtiva eder. KeyPressEventArgs.KeyChar özelliği, ASCII karakterleriyle ilgili bilgileri saklar.

3.2.4. Doğrulanan Karakterler

Char veri tipi, KeyPress olayı sonunda veri doğrulama için gerekli olan bazı kullanışlı metotlar ihtiva eder. Bu metotlar şunlardır:

- Char.IsDigit
- ➢ Char.IsLetter
- Char.IsLetterOrDigit

- Char.IsPunctuation
- ➢ Char.IsLower
- Char.IsUpper

Bu metotların her biri isimlerinden de belli olduğu gibi (tabii İngilizce olarak), girilen karakterleri değerlendirerek Boolean türünde bir değer döndürür. Char.IsDigit girilen karakterin sayı olup olmadığını belirten bir fonksiyondur. Karakter sayısal ise true değilse false değeri döndürür. Char.IsLower, girilen karakterlerin büyük harf mi küçük harf mi olduğunu belirler. Diğer metotlar da benzer şekilde davranır.

Aşağıdaki program parçasında, girilen karakterin sayısal olup olmadığı kontrol edilmektedir.

```
private void textBox1_KeyPress (object sender,
   System.Windows.Forms.KeyPressEventArgs e)
{
   if (Char.IsDigit(e.KeyChar) == true)
       MessageBox.Show("sayısal bir tuşa bastınız");
}
```

3.2.5. Odaklama (Focus) Ayarı

Odaklama, klavye ya da fare ile girilen bilgilerin alınmasını sağlayan bir nesnenin yeteneğidir. Formunuz üzerinde birden fazla kontrol olduğu halde, belirli bir zamanda sadece bir kontrolü odaklayabilirsiniz. Uygulamanızın formu üzerinde odaklanan kontrol her zaman aktif olur.

Her kontrol odaklama (focus) olayını yerine getirebilir. Bu metot kontrolün odak ayarını yapar. Metot kontrolün odak ayarının başarılı olup olmamasına bakmadan Boolean bir değer döndürür. Pasif durumda ya da gizlenmiş olan kontroller odaklanmaz. CanFocus özelliğiyle bir kontrolün odaklanıp odaklanmadığını tespit edebilirsiniz. Eğer true değeri alırsa kontrolün odaklanma özelliği alabildiğini gösterir.

Focus olayı aşağıdaki işlemleri de yapabilir:

- ➢ Enter
- ➢ GotFocus
- ➤ Leave
- Validating
- > Validated
- ➢ LostFocus

Enter ve Leave olayları, odaklanmanın bir kontrole ulaşması ya da ayrılmasına bağlı olarak yapılacak işleri belirler. GotFocus ve LostFocus ise, kontrol odaklanma özelliğini alırsa veya kaybederse yapılacak işleri belirler. Göreve bağlı olarak Validated ve Validating olaylarını da kullanabilirsiniz.

UYGULAMA FAALİYETİ

İşlem Basamakları	Öneriler
 İçinde 3 adet menü öğesi bulunan bir form tasarlayınız. Her menü öğesinin 3 tane alt öğesi olsun. 	Visual Studio'nun kendi yardımı ile kaynakçada belirtilen kitap isimleri ve web sitelerini kullanabilirsiniz.
 Ana fomunuza 2 adet Label, 2 adet de TextBox kontrolü ekleyiniz. Label'lerde Bireysel Kullanıcı ve Kurumsal Kullanıcı yazsın. TextBox'lara ise müşteri numaraları girilecektir. 	
Menü öğelerine erişim tuşlarını belirleyiniz.	
 Bireysel kullanıcı için belirlediğiniz TextBox kontrolünü odaklayınız. 	
Müşteri numarası olarak 5 haneden fazla karakter girişine izin vermeyiniz.	
Aynı zamanda rakam dışında bir karakter girilmesine de müsaade etmeyiniz.	

ÖLÇME VE DEĞERLENDİRME

OBJEKTİF TEST (ÖLÇME SORULARI)

Aşağıdaki sorulardan; sonunda parantez olanlar doğru / yanlış sorularıdır. Verilen ifadeye göre parantez içine doğru ise "D", yanlış ise "Y" yazınız. Şıklı sorularda uygun şıkkı işaretleyiniz.

- 1. MenuStrip komponenti tasarım aşamasında kullanılır. ()
- 2. Menülere klavyeden erişmek için kullanılan erişim tuşlarında, istenen erişim karakterinin başına % işareti konur. ()
- 3. İçerik menüsü eklemek için MenuItem kontrolü kullanılır. ()
- 4. Menüler üst seviye fonksiyon ve komutları programa eklememizi sağlar. ()
- 5. Aşağıdakilerden hangisi Alan Düzeyli doğrulamanın olaylarından değildir?
 A) KeyDown
 B) KeyPress
 C) KeyUp
 D) KeyHit

DEĞERLENDİRME

Cevaplarınızı cevap anahtarı ile karşılaştırınız. Doğru cevap sayınızı belirleyerek kendinizi değerlendiriniz. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt yaşadığınız sorularla ilgili konuları öğrenme faaliyete geri dönerek tekrar inceleyiniz.

MODÜL DEĞERLENDİRME

PERFORMANS TESTİ (YETERLİK ÖLÇME)

Modül ile kazandığınız yeterliği, öğretmeniniz işlem basamaklarına göre 0 ile 9 puan arasında olacak şeklinde değerlendirecektir.

DEĞERLENDİRME ÖLÇÜTLERİ	Evet	Hayır
Projeye yeni bir form eklediniz mi?		
Başlangıç formunu ayarlayabildiniz mi?		
Form ve kontrollerin ekran ayarlarını yapabildiniz mi?		
Formun olaylarını kullanabildiniz mi?		
Kontrol ve komponentleri ayırt ettiniz mi?		
Kontrollerin özelliklerini değiştirebildiniz mi?		
Kontroller üzerinde fare etkileşimlerini uygulayabildiniz mi?		
Forma yeni bir menü eklediniz mi?		
Menü öğelerinin erişim tuşlarını belirleyebildiniz mi?		
Menü öğelerine kısa yol tuşları atayabildiniz mi?		
Çalışma anında istenen menü öğesini gizleyebildiniz mi?		

DEĞERLENDİRME

Yaptığınız değerlendirme sonucunda eksikleriniz varsa öğrenme faaliyetlerini tekrarlayınız.

Modülü tamamladınız, tebrik ederiz. Öğretmeniniz size çeşitli ölçme araçları uygulayacaktır, öğretmeninizle iletişime geçiniz.

CEVAP ANAHTARLARI

ÖĞRENME FAALİYETİ-1'İN CEVAP ANAHTARI

1	D
2	D
3	Y
4	D
5	Y
6	Y

ÖĞRENME FAALİYETİ-2'NİN CEVAP ANAHTARI

1	D
2	Y
3	D
4	Y
5	D
6	С

ÖĞRENME FAALİYETİ-3'ÜN CEVAP ANAHTARI

1	D
2	Y
3	Y
4	D
5	D

SÖZLÜK

İsim	Okunuş	Anlam
activate	äk'tıveyt	etkinleştir,harekete geçir
anchor	äng'kır	bağlantı, sabitleyici
caption	käp'şın	başlık, simge yazısı, resim yazısı, düğme yazısı
class	kläs	sınıf,sınıflandır
clone	klon	kopyalama
close	kloz	son vermek, bitirmek; sona ermek, bitmek.
component	kimpo'nint	öğe, unsur, parça, eleman, cüz.
container	kınteynır	Кар, Карѕауісі
design	dîzayn'	motif, tasarım, tasarımlamak
dock	dak	yapışık
event	îvent'	olay, vaka, hadise.
event handler	îvent' hendlır	Olay yönlendirici
false	fôls	Yanlış, hata
focus	fo'kıs	Odak, belirli bir noktaya geitrmek
hide	hayd	Gizle, gizleme
inherit	înher'ît	miras almak, kalıt almak
length	lengkth,	uzunluk
level	levil	düzey, seviye, birim
manual	män'yuwıl	elle yapılan; elle çalıştırılan, el kitabı
merge	mırc	birleşmek; birleştirmek.
method	meth'ıd	Yöntem, metod, usül
namespace	neymsipeys	İsim alanı
opacity	opasity	şeffaf olmayış.
properties	pırapıtiyz	özellikler
runtime	Runtaym	Çalışma zamanı
seperator	Sepireytir	ayırıcı
show	Şo	göstermek
simplicity	sîmplîs'ıti	Sadelik, basitlik
startup	Sıtart-ap	açılış, başlangıç, sistemin açılışı
taborder	tebordır	Sekme sırası
true	tru	Doğru, gerçek
validate	väl'ideyt	onaylamak, tasdik etmek.



Modüllerde gördüğünüz gibi, hemen her şey ingilizce kelimelerden oluşuyor. Dolayısıyla ne kadar çok yabancı kelime bilirseniz programlama dilini kavramanız o ölçüde artacaktır.

KOD ÖRNEKLERİ

```
1- Basit Bir Windows Formu (Penceresi) Oluşturma
using System;
using System.Windows.Forms;
namespace WindowsFormOrnegi
{
    class ilkFormumuz : Form
    {
        static void Main()
        {
            Application.Run(new ilkFormumuz());
        }
    }
}
```

2- Forma buton ekleme ve bazı özelliklerini değiştirme

```
using System;
using System.Windows.Forms;
namespace WindowsFormOrnegi_2
ł
    class ikinciForm : Form
        private Button ilkButon;
        public ikinciForm()
        {
            ilkButon = new Button();
            ilkButon.Text = "Merhaba C#";
            ilkButon.Location = new System.Drawing.Point(50,50);
            ilkButon.Cursor = Cursors.No;
            ilkButon.Size = new System.Drawing.Size(150,50);
            this.Text = "İkinci Windows Uygulamamız";
            this.Controls.Add(ilkButon);
        }
        static void Main()
        {
            Application.Run(new ikinciForm());
        }
    }
```

```
3- Butonun Click Olayını Yazmak
```

```
using System;
using System.Windows.Forms;
namespace ButonlaraOlayEkleme
{
    class ucuncuForm : Form
    {
        private Button ilkButon;
        public ucuncuForm()
        ł
            ilkButon = new Button();
            ilkButon.Text = "Merhaba Windows";
            ilkButon.Location = new System.Drawing.Point(50, 50);
            ilkButon.Cursor = Cursors.Hand;
            ilkButon.Size = new System.Drawing.Size(150, 50);
            ilkButon.Click +=new EventHandler(ButonTiklandi);
            this.Text="Windows Form Uygulaması";
            this.Controls.Add(ilkButon);
        }
        static void Main()
        ł
            Application.Run(new ucuncuForm());
        }
        void ButonTiklandi(object o, EventArgs a)
        ł
            MessageBox.Show("Butona Tiklandi");
    }
```

ÖNERİLEN KAYNAKLAR

- ➢ www.csharpnedir.com
- www.msdn.microsoft.com/vcsharp/
- ➢ www.c-sharpcorner.com/
- ▶ www.csharphelp.com/
- www.csharp-station.com/Tutorial.aspx
- www.codeproject.com/csharp/
- www.functionx.com/csharp/index.htm
- www.msakademik.net/giris.aspx
- www.programmingtutorials.com/csharp.aspx
- www.publicjoe.f9.co.uk/csharp/tut.html
- http://www.yazgelistir.com

KAYNAKÇA

- > ALGAN Sefer, Her Yönüyle C#, Pusula Yayıncılık, 2003.
- **ERIK** Brown, **Windows Form Programming With C#,** Manning Publications, 2002.
- JONES Bradley L., Teach Yourself The C# Language In 21 Days, SAMS Publishing, 2004.