İÇERİK

ÜNİTE HAKKINDA

Programlamaya yeni bir bakış açısı ile karşı karşıyayız. Daha doğrusu önceden bu teknikle çalışmamış iseniz yeni bir bakış açısı sayılabilir: Nesne Yönelimli Programlama Tekniği.

Bu teknik sizi, klasik yordamsal programlama yöntemlerinden kurtararak, size daha dinamik ve daha modüler bir programlama yapmanıza olanak sağlayacaktır.

Aslına bakarsanız bu üniteye kadar zaten biz bu tekniği bir şekilde kullanıyorduk. Zira birçok örnekte nesneler ve özellikleri ile ilgili programlar yazmıştık. Yalnız, daha çok sözel bir şekilde geçecek olan bu üniteyle birlikte, Görsel Programlama için olmazsa olmazlarından biri olan Nesne Yönelimli Programlama tekniği hakkında daha çok bilgi sahibi olacak ve bu tekniği görece daha bilinçli bir şekilde inceleyerek mantığını kavramaya çalışacağız.

ÖĞRENME HEDEFLERİ

Bu ünite bitiminde;

- Nesne ve Sınıf kavramlarını açıklayabilecek,
- Nesnelerin özelliklerini ve olaylarını tanıyabilecek,
- Nesnelerin temel prensiplerini bilebilecek,
- Delphi'de ki sınıf ve nesneleri ayırt edebilecek ve
- Var olan sınıflardan yararlanarak ve RAD Studio olmadan (sırf Delphi kodlarıyla) çalışma zamanında kendi nesnelerinizi oluşturabileceksiniz.

ÜNİTEYİ ÇALIŞIRKEN

Günümüzdeki programlama dillerinin çoğu NYP tekniğine az ya da çok destek vermektedir. NYP tekniği ile ilgili olarak öğrendiklerinizi Delphi haricinde ki başka programlama dillerinde de (C#, java, C++ gibi) uygulama yoluna gidebilirsiniz.

Bu size dilden bağımsız bir şekilde tekniğin nasıl çalıştığına dair fikir verecektir.

ANA METİN

4. Nesne Yönelimli Programlamaya Giriş

Artık Delphi ortamını ve projelerini tanıdığımıza göre elimizi iyiden iyiye Delphi programlamaya bulaştırmamızın zamanı gelmiştir. İşin aslı zaten elimizi daha ilk üniteyle birlikte Delphi programlamaya bulaştırmıştık, fakat bazı konularda durumun çok da bilincine varmadan yaptığımız örneklerdi bunlar. Artık bu üniteyle birlikte, programcılık hayatımızda önemli bir yeri bulunması gereken yeni bir programlama tekniğinin de farkına varmaya ve bu tekniği kullanarak da bilinçli bir şekilde *görsel programlama* yapmaya başlayacağız.

Görsel programlama kavramını açıklamaya çalışırken yaptığımız tanımlamalarda işletim sistemi tarafından sağlanan *nesnelerden* bahsetmiştik.

? Peki, ama neydi bu nesneler?

Metin giriş kutuları, butonlar, seçenek düğmeleri… dediğinizi duyar gibiyim. Doğrudur, bunlar işletim sistemi tarafından programlarımızda kullanılmak üzere sunulan ve kolaylıkla kullanabildiğimiz *daha doğrusu Delphi gibi bir görsel programlama dili yardımıyla* kolaylıkla kullanabildiğimiz yapılardır. Daha da sayamadığımız yüzlercesi bulunmaktadır.

? Fakat bütün bunlar neden *nesne* olarak ifade edilmektedir ve bu nesnelerin de *gerçek hayattaki nesnelerle* bir ilişkisi var mıdır?

İşte tüm bu soruların cevapları *Nesne Yönelimli Programlama* diye tabir edilen yeni bir programlama tekniğinde yatmaktadır.

4.1. Nesne Yönelimli Programlama (Object Oriented Programming) Tekniği Nedir?

Bu teknik, programlamaya yeni –yeni derken en azından bizim için yeni- bir bakış açısıdır. Uygulamaların tasarımını, yeniden düzenlenmesini ve birçok kişinin aynı uygulama üzerinde birlikte çalışmasını kolaylaştırmak amacıyla geliştirilmiştir. Bu tekniğin avantajları daha çok, *büyük çaplı uygulamalarda* kendisini göstermektedir (Hatta küçük çaplı uygulamaların çoğunda ise "*ben bunu şu yolla daha kolay yapardım*" dedirttiği de olmaktadır. Özellikle de nesneleri kendiniz oluşturmak zorunda kaldığınızda, ama iyi haber: biz daha çok, önceden oluşturulmuş nesnelerle programlama yapacağız, işte bu da görsel programlama olmaktadır).

NYP (Nesne Yönelimli Programlama) tekniği birkaç küçük dezavantajı yanında yine de birden fazla kişinin aynı proje üzerinde çalışması ve yeniden düzenlenebilirlik söz konusu olunca görsel programlama konusunda vazgeçilmez tekniklerimizden birisi olacaktır.

Programlama konusunda ki bu yeni yaklaşıma göre programlama ortamında ki her şey potansiyel bir nesnedir ve dolayısıyla programlama da bu *Nesne Yönelimli* ortamda yapılmaktadır.

? Peki, ama bu nesnelerin bildiğimiz nesnelerle bir ilişkisi var mıdır?

Bu güne kadar bilgisayar programlama sanatını öğrenmeye çalışırken, hep bizler bilgisayara yaklaşmıştık, yani hep bizler onun o garip dilini öğrenmeye çalışmıştık. Hatırlayın, algoritma derslerimizde problemlerimizin algoritmalarını kurarken *daha böyle bir bilgisayar dili ile düşünmeye gayret edip*,

- Başla,
- Şu oluncaya kadar şu işi şu kadar yap,
- Sonra şu satırdan devam et…
- Dur.

gibi komutlarla derdimizi ona anlatmaya çalışırdık.

İşte şimdi bu teknik yardımıyla bilgisayarı az da olsa bizim tarafa çekmeyi başarabiliyor ve programlama ortamını *nesne yönelimli* bir hale getirerek derdimizi ona biraz daha *bizden bir şeylerle ifade etmeye* çalışıyoruz. Bu anlamda Nesne Yönelimli Programlama tekniğinde bahsedilen nesnelerin *evet*, *gerçek hayattaki nesnelerle bir bağları bulunmaktadır.*

? Peki, nesne nedir?

Aslına bakarsanız nesnenin tam tanımını yapmaya çalıştığımızda, elle tutulabilen, gözle görülebilen, kütlesi ve hacmi bulunan diye bir giriş yapıp, uzun uzadıya da tanımlamak mümkündür. Fakat bizim işimize yarayacak şekilde bir tanımlama yapmaya çalıştığımızda ise işin özünde *algılamak* olduğu ortaya çıkmaktadır. Sonuçta biz *nesneleri algılarız*. Bunu da bazı özelliklerini ortaya koyarak yaparız. Algılamak içinde illaki o nesneyi görmemiz, duymamız veya bir şekilde ona dokunmamız da gerekmez.

Bir şekilde nesnelerin özellikleri çok iyi bir şekilde ortaya konulabilirse -yani tarif edilebilirse- karşı taraf onu rahatlıkla algılayabilecektir. İşte bu algılama da o nesneyi oluşturmuş olacaktır. (Aslında bende felsefeye bu kadar girilmemesi gerektiğini düşünenlerdenim, maazallah felsefeye girip de çıkamayan çok kişi olduğunu da biliyoruz☺)

-Onun için- kısaca şunu demek istiyorum; örneğin şimdi ben, şu an yanımda duran ve sizin hiç görmediğiniz bir nesneyi, size çok iyi bir şekilde tarif edebilsem (yani özelliklerini çok iyi bir şekilde size aktarabilsem, şu renktedir, şu malzemeden yapılmıştır gibi bunun yanında eni-boyu-yüksekliği bilgilerini de size tam olarak verebilirsem eğer) o nesneyi en azından sizin kafanızda canlandırmış olabilirim. Yani bir anlamda o nesnenin özelliklerini ortaya koyarak onu sizin algı dünyanızda oluşturmuş olurum.

Mouse Nesnesi

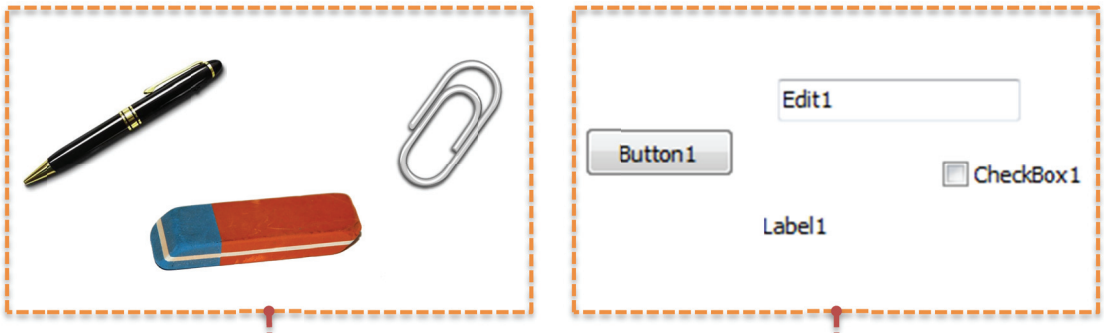
Buradan da anlıyoruz ki bir nesneyi tanımlayan özellikleri bulunmaktadır. Örneğin yukarıdaki gibi bir fareyi,

- Siyah renkli,
- İki temel butonu,
- Bir tekerleği bulunan,
- …
- Genelde sert-parlak bir malzemeden yapılmış olmasına rağmen
- Başparmağımızın denk geldiği yerde daha mat ve kaydırmaz bir madde kullanılarak üretilmiş bir nesnedir şeklinde tanımlamak mümkündür.

Bunun yanında bu fare ile neler yapılabilir sorusuna da:

- Tıklanabilir,
- Çift Tıklanabilir,
- Kaydırma yapılabilir,
- Sağ tıklanabilir… gibi daha birçok olaydan da (iş, oluş, eylemden) oluşan cevaplar verilebilir.

İşte tüm bu özellik ve olaylar bir nesne üzerine atfedilen *tanımlayıcılardır*. Bu tanımlayıcılar sayesinde de nesnelerimiz oluşmaktadır.



Gerçek Hayatta ki Nesneler

Delphi'de ki Nesneler

Nesne Yönelimli Programlama tekniği de programlamaya tamda bu açıdan yaklaşmaktadır. Uygulamalarda kullanılan modüller (parçalar) birer nesne olarak

tasarlanmakta ve o nesnelerinde özellikleri ve olayları kısaca tanımlayıcıları ortaya konmaktadır. Sonrasında ise oluşturulan bu nesneler birleştirilerek uygulamaları meydana getirmektedirler.

Nesnelerden meydana gelen uygulamalar ise nesnelerin özelliklerinden dolayı daha *müdahale edilebilir* olmaktadır.

…1960'lı yılların sonuna doğru ortaya çıkan bu yaklaşım, o dönemin yazılım dünyasında beliren bir bunalımın sonucudur. Yazılımların karmaşıklığı ve boyutları sürekli artıyor, ancak belli bir nitelik düzeyi korumak için gereken bakımın maliyeti zaman ve çaba olarak daha da hızlı artıyordu. NYP'yi bu soruna karşı bir çözüm haline getiren başlıca özelliği, yazılımda birimselliği (modularity) benimsemesidir. NYP ayrıca, bilgi gizleme (information hiding), veri soyutlama (data abstraction), çok biçimlilik (polymorphism) ve kalıtım (inheritance) gibi yazılımın bakımını ve aynı yazılım üzerinde birden fazla kişinin çalışmasını kolaylaştıran kavramları da yazılım literatürüne kazandırmıştır. Sağladığı bu avantajlardan dolayı, NYP günümüzde geniş çaplı yazılım projelerinde yaygın olarak kullanılmaktadır.

NYP'nın altında yatan birimselliğin ana fikri, her bilgisayar programının (izlence), etkileşim içerisinde olan birimler veya nesneler kümesinden oluştuğu varsayımıdır. Bu nesnelerin her biri, kendi içerisinde veri işleyebilir ve diğer nesneler ile çift yönlü veri alışverişinde bulunabilir. Hâlbuki NYP'den önce var olan tek yaklaşımda (Yordamsal programlama), programlar sadece bir komut dizisi veya birer işlev (fonksiyon) kümesi olarak görülmekteydiler. (Nesne Yönelimli Programlama - Vikipedi)

4.2. Nesne Yönelimli Programlama Tekniğinin Avantajları

!*Nesne Yönelimli Programlama Tekniği* ili *Görsel Programlama* kavramlarını karıştırmamak gerekmektedir. Şöyle söyleyebiliriz: Nesne Yönelimli Programlama tekniği sağladığı avantajlardan dolayı görsel programlama sürecinde yararlanacağımız başta gelen tekniklerden olacaktır.

NYP'nin sağladığı avantajlara da kısaca değinmek gerekirse eğer; yukarıdaki Wikipedi'adan yaptığımız alıntıda da bahsedilen özelliklerine göz atmak gerekmektedir. Zira avantajları bu özellikleri içerisinde saklı durumdadır.

NYP tekniği ile oluşturulan nesneler temelde üç özelliğe sahiptirler:

- Bilgi gizleme (information hiding), bazen Veri Soyutlama (data abstraction) da denilebilmektedir,
- Kalıtım (inheritance) ve
- Çok biçimlilik (polymorphism).

İşte tüm bu özelliklerinden dolayı da nesnelerden oluşturulmuş uygulamalar yukarıda da belirttiğimiz gibi daha müdahale edilebilir yapılar olarak karşımıza çıkmaktadırlar.

4.2.1. Nesnelerin Bilgi Gizleme Özelliği

Örneğin aşağıdaki gibi irili ufaklı ve belki de binlerce nesneden oluşan bir otomobil sistemini düşündüğümüzde tüm bu nesnelerin bir şekilde belli bir amaç için bir araya