Dinamik Kodlama

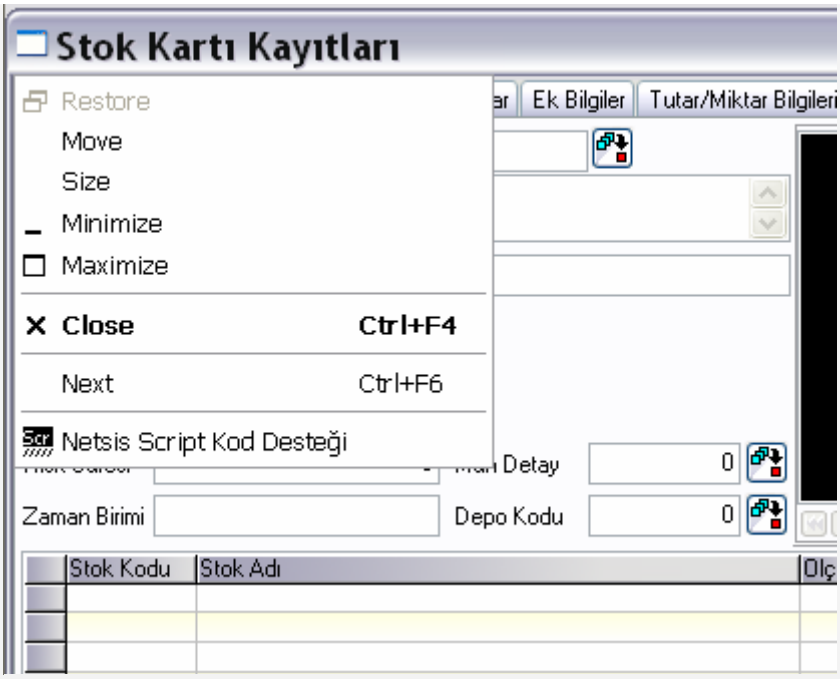
Ürün Grubu	[X] Fusion@6
Kategori	[X] Yeni Fonksiyon
Versiyon Önkoşulu	@6
Uygulama	@6 Serisi ürünlerde, kullanıcı arabirimlerinin her yerine eklenen dinamik kodlama özelliği ile, programın standart davranışını değiştirecek kod yazılması, ekranların istendiği şekilde değiştirilebilmesi, yeni özellikler kazandırılması gibi programlama tekniği ile yapılabilecek fonksiyonlar, kodlanabilmektedir. Dinamik kodlama özelliğinde, kodlama VBScript dili ile yapılmaktadır. Vbscript kodlarını sadece admin olan kullanıcılar tanımlayabilmekte ve gerektiğinde geçersiz hale getirebilmektedirler.

1. Dinamik Kod Desteği Parametresi

Dinamik kod desteğinin programda aktif hale getirilmesi için Yardımcı Programlar \ Kayıt \ Şirket Şube Parametre Tanımları menüsünde Parametreler sekmesinde bulunan Dinamik Kod Desteği parametresinin işaretlenmesi gerekmektedir. Bu parametrenin işaretli olduğu durumlarda, formların sol üst köşesinde bulunan N Harfine tıklandığında, Netsis script kod desteği seçeneği gelecektir. Netsis script kod desteği tıklandığında kod geliştirme ortamı açılacaktır.

2. Dinamik Kodun Yazılma ve Çalışma Yeri

Dinamik kodlama, Netsis programlarının herhangi bir ekranında ya da NDI uygulamasında hazırlanan bir ekranda çalışması için, o ekrana ait script kod girişi bölümünde yazılır. Kod, hangi ekran için yazıldıysa o ekran için çalışır. Yazılan kod, içinde bulunduğu ekranda, programın doğal davranışı dışında başka bir işlemin yapılabilmesini sağlar. Örneğin stok kartı kayıtları ekranında çalışması istenen dinamik kod, "stok kartı kayıtları" menü seçeneği ile açılan pencerenin sol üst köşesindeki menüsünden "*Netsis Script Kod Desteği*" seçeneği ile açılan bölümde yazılmalıdır.

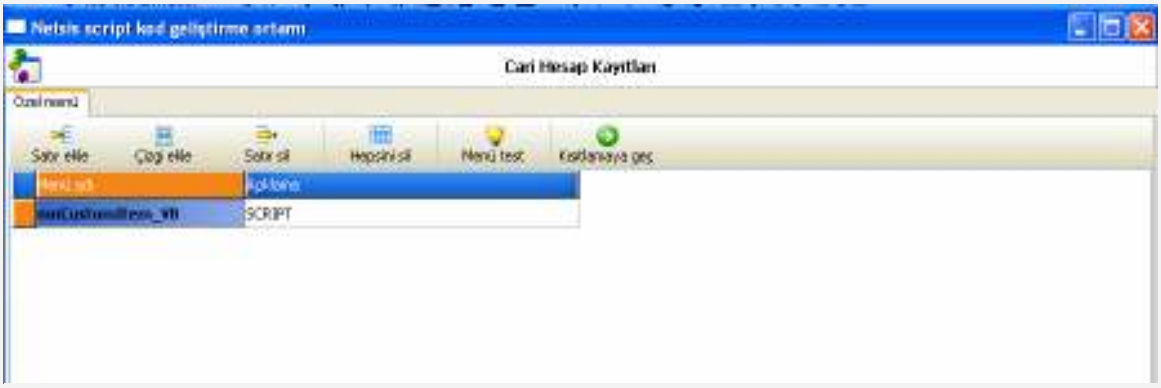


Stok modülü ana menüsünün sol üst köşesinden açılan script kod bölümünde yazılan kod, sadece modül menüsünde çalışacaktır.
NDI uygulamasında hazırlanan bir ekran (form) için, programlama, yine formun sol üst köşesinden açılan menüden ya da form içindeki herhangi bir bileşen üzerinde sağ click menüsünden Script / Dinamik Kod Desteği seçeneği ile gelen ortamda yapılabilir.
Script kodun içinde çalıştığı ekrana "*form*" adı verilmektedir ve dokumanın sonrasında bu şekilde adlandırılacaktır.
Netsis script kod geliştirme ortamı üç bölümden oluşmaktadır. Bunlardan ilki, kod geliştirme ortamına girildiğinde gelen Özel Menü başlıklı bölüdür.

3. NETSİS SCRIPT KOD DESTEĞİ

3.1 Özel Menü Sekmesi

Dinamik kodlama uygulaması ile kullanıcılar tarafından programdaki sağ kliklerdeki menülere eklemeler yapılabilir. Özel Menü bölümünde de farenin sağ tuşunda gelecek menünün adı belirlenmektedir. Kullanıcılar tarafından eklenen bu menülere vbscript kodu yazılabilir.

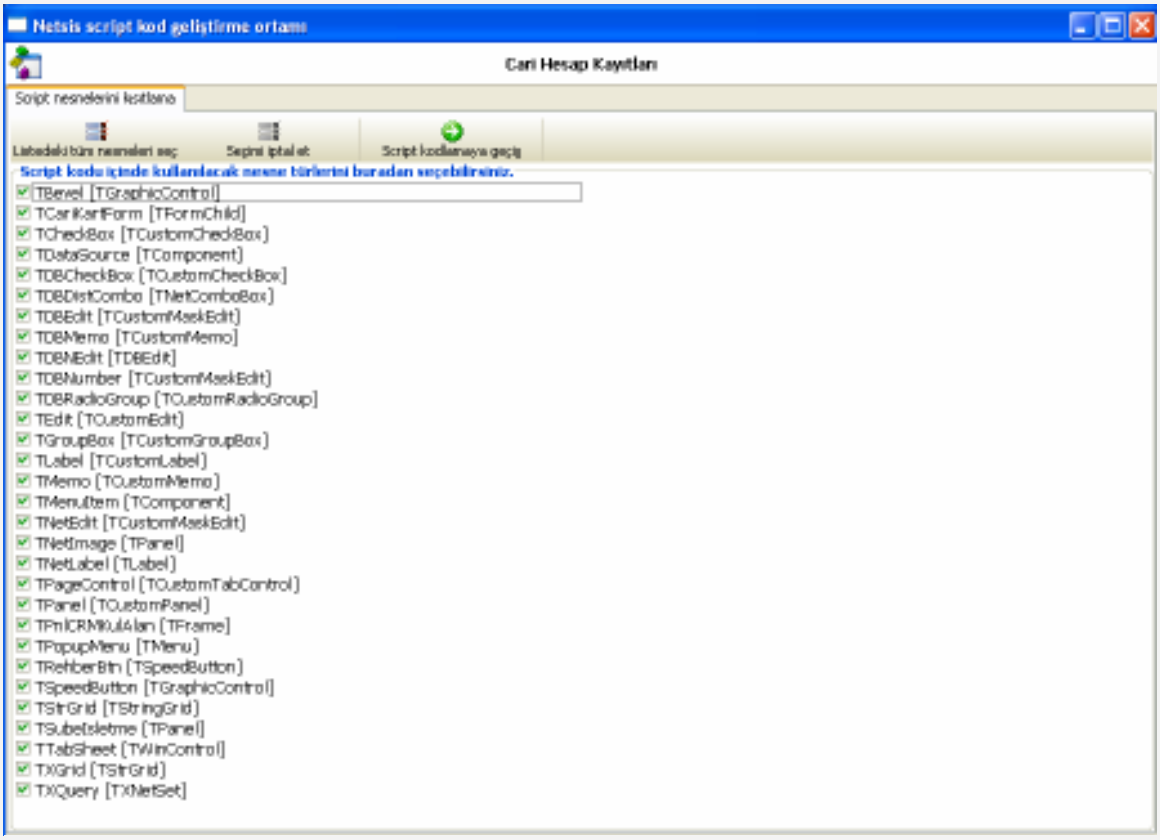


Örneğin SCRIPT isimli bir menü eklemek isteniyorsa Menü adı kısmına kodlama sırasında görülecek açıklama yazıldığında, bu adın başına program tarafından mnCustomItem_ ibaresi ekleyecektir. Açıklama kısmına yazılan değer ise tanımlanan menünün ekranda görülecek ismidir.
Satır Ekle: Yeni bir menü eklemek için boş bir satır açar.

Çizgi Ekle: Menüler arasına yatay çizgi eklenmesini sağlar.
Satır Sil: Üzerinde bulunulan satırı silecektir.
Hepsini Sil: Bu ekranda bulunan tüm tanımlamaları silecektir.
Menü Test: Menünün görünümünün test edilmesini sağlar.
Kısıtlamaya Geç: Özel menü bölümünden çıkarak, Script nesneleri kısıtlama bölümüne geçmek üzere kullanılan butondur.
Bu bölümde yapılan tanımlamalar sonucunda, tanımlamanın yapıldığı ekranda farenin sağ tuşuna basıldığında "Kullanıcı İşlemleri" kısayol seçeneği gelecektir. Bu seçenek altında da tanımlanan özel menüye, belirlenen açıklama bilgisi ile erişilebilir.

3.2 Script Nesnelerini Kısıtlama Sekmesi

Özel Menü bölümündeki "Kısıtlamaya Geç" butonuna basıldığında script girişi esnasında hangi nesnelerin kullanılacağının belirlendiği Script Nesnelerini Kısıtlama bölümü gelecektir. Eğer vbscript kodu, sadece yeni eklenen sağ klik menüsü için yazılacaksa, gelen listeden sadece TMenuItem'ın seçilmesi yeterli olacaktır. Burada bulunan nesnelerin tümü seçilerek de kodlamaya geçiş yapılabilir. Bu durumda, formdaki tüm nesneler için istenirse kodlama yapılabilir. Ancak, tüm nesnelerin eklenmesi, formun açılma hızını etkileyecektir. Bunun sebebi formun her açılışta buradaki nesneleri yükleyerek açmasıdır. Listede yer alan nesneler, scriptin içindeyazıldığı formda bulunan nesnelerdir . *Bknz: Nesne (Object, Instance)*.



Listedeki tüm nesneleri seç: Listelenen nesnelerin tamamını birden seçmek için kullanılan butondur. Seçilen nesnelerin solundaki kutu işaretlenecektir.
Seçimi iptal et: Listelenen nesnelerden seçilmiş olaların seçimini kaldırmak için kullanılan butondur.
Script kodlamaya geçiş: Seçilen nesneler kullanılarak script kodu yazmak için Script Tanımlama bölümüne geçmek için kullanılan butondur.

3.3 Script Kodlama Sekmesi

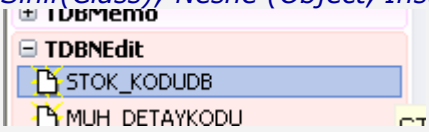
3.3.1 Olay (Event)

Script kodu, formdaki herhangi bir görsel bileşene ait olay (event) gerçekleşmesinde

çalışabilir. Daha doğrusu script kodu, formdaki herhangi bir görsel bileşenin, herhangi bir olayında çalışması için yazılır. Örneğin "kod" isimli alfabetik bilgi giriş sahası (TEdit) görsel bileşeninin çıkışında (onexit), kullanıcının "kod" sahasına girmiş olduğu bilginin kontrolü script içinde yapılarak, buna göre farklı bir davranışta bulunulabilir, mesela girilen kodun kabul edilmemesi, uyarı verilmesi ve tekrar kod okunmak üzere aynı sahaya dönülmesi gibi. *Bknz: Olay Sürmeli (Event Driven) Programlama*

3.3.2 Nesne İzleme Bölümü

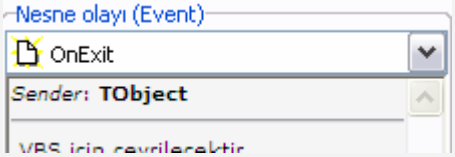
Script kodu yazılmadan önce mutlaka hangi bileşenin hangi olayı için yazılacağı belirlenmelidir. Script kodlama ekranına geçildiğinde, sol bölümdeki "*Nesne İzleme*" penceresinde, scriptin yazıldığı formda bulunan görsel bileşenler listelenir. Listede her bir görsel bileşen, bir nesne olup, kendi türediği sınıf başlığının altında yer alır. *Bknz: Sınıf(Class); Nesne (Object, Instance)*.



Örneğin, TEdit sınıfının altında, script yazılan formda bulunan alfabetik okuma sahalarının isimleri yer alır. TButton sınıfı altında, formdaki butonların isimleri yer alır. Listeden, herhangi bir olayı için script kod yazılmak istenen bileşen, fare ile tek tıklanarak belirlenmelidir.
Burada listelenen bileşenlerden herhangi biri için daha önceden script girişi yapılmış ise, ilgili bileşenin solunda, sağa doğru yeşil bir ok işareti olacaktır. Bu bileşen seçildiğinde ise, girilmiş script, Kod geliştirme bölümüne gelecektir.

3.3.3 Nesne Olayı (Event) Bölümü

"*Nesne İzleme*" penceresinin altındaki bölümde yer alan "*Nesne Olayı (Event)*" bölümünde ise, seçilen bileşen için geçerli olaylar listelenir. Script kodunun, hangi olay gerçekleştiğinde çalışması isteniyorsa, bu olay fare ile tıklanarak belirlenmelidir.



Örneğin TButton sınıfından OkBtn isimli bileşen ve bu bileşene ait olaylardan onclick seçilirse, kullanıcı ekrandaki OkBtn isimli butona bastığında çalışması istenen kod yazılabilir. Örnekte, Stok Kartındaki stok kodu (STOK_KODUDB) sahasının çıkışında (onexit) çalışacak kodun yazılacağı belirtilmiştir. *Bknz: Olay Sürmeli (Event Driven) Programlama*.
Kodlama bölümünde yazılan kod, seçilen nesnenin, seçilen olayı gerçekleştiğinde çalışacaktır. Kod yazılırken sol tarafta seçili nesne ve olayın, doğruluğuna dikkat edilmelidir.

3.3.4 Kodlama Bölümü

3.3.4.1 VBScript Syntax

Ekranın ortasındaki boş alan script girişi için editor olarak tasarlanmıştır. Kodlama, işletim sisteminin doğrudan yorumlayabildiği VBScript komutları ve sentaksı kullanılarak yapılmalıdır. "IF ... THEN ... ELSE ... END IF", "SELECT CASE ... ELSE ... END SELECT", "FOR ... NEXT", "DO WHILE ... LOOP", "MSGBOX", "INPUTBOX" gibi vbscript komutları, temel program mantığının oluşturulmasında kullanılır.

3.3.4.2 Araç Çubuğu – 1 (Sol Üst)

Kaydet: Script'in saklanmasını sağlar. Yazılan script'ler bu butonla saklandıktan sonra, değişikliklerin geçerli olması için ekranın en alt bölümündeki Tamam butonuna basarak ekranın kapatılması gerekmektedir. Script ekranı Tamam butonu dışında bir şekilde kapatılırsa, değişiklikler saklanmadan çıkılacaktır.

Yenile: Kod geliştirilme ortamının güncellenmesi için kullanılan butondur. Bu butona basıldığında, script girişi en son kaydedilen haline dönecektir.

Temizle: Kod içeriğinin temizlenmesi için kullanılan butondur. Yani butoa basıldığında, ekranda görülen scriptler silinecektir.

Kod Şablonları: Daha önceden saklanmış kod şablonları arasından seçim yapmak için kullanılan butondur. Bu butona basıldığında, Netsis Script Kod Şablon Tanımlamaları rehberi gelecektir.

Form Global:

Session Global:

Uygulama Global: Tüm formlarda, tüm şirketlerde ortak kullanılmak istenilen tanımlamaların girilmesi için kullanılan butondur. Yani eğer yazılan bir fonksiyon veya sub her yerde ortak olarak kullanılmak isteniyorsa bu tanımlamayı uygulama global kısmında yaparak tüm modüllerde ve şirketlerde bu fonksiyona erişerek çalıştırılabilir. Bu özellik ile ortak olan bir fonksiyon tanımlamasının, kullanılacak olan her ekranda tekrar tekrar yazılması önlenmiştir.
Örneğin;

```
sub mail(kime,cc,subject,ek,body)
call NETSISCORE.NetLibEMail.EPostaGonder(kime,cc,subject,ek,body)
end sub
```

şeklinde Uygulama Global'de yapılan bir tanımlamayı bizim hazırladığımız menüde kullanabilmek için;

```
call  appglobal.mail(MAIL,"","MEKTUP","",ICERIK)
```

şeklinde çağırılması gerekmektedir.
Yukarıdaki sub tanımlamasında NETSISCORE.NetLibEMail.EpostaGonder şeklinde bir tanımlama yapılmıştır. Bu tanımlama ile Netsis'in nesnelerinin kullanıldığı belirtilmiştir. Bu nesneler hakkında bilgi almak için Nesne Tarayıcısı Kullanılabilir. *Bknz: Nesne Tarayıcısı*.

3.3.4.3 Nesneye Yönelik Programlama Kavramları

Sınıf (Class)

Nesneye yönelik programlamada temel öğelerdir. Gerçek dünyadaki ya da akıldaki mantıksal nesnelerin programlamda ifade edilmesi için Class yapısı kullanılır. Sınıfların özellik (property) ve yöntemleri (method) vardır. Özellikler, sınıfı tanımlayan verileri olup, yöntemler ise sınıfın yapabileceği işlemler, yani yetenekleridir. Sınıf tanımı içinde sınıfa ait özellik ve yöntemler belirtilir. Programda sınıf tanımlarından nesneler türetilir. O nedenle sınıf tanımı, aynı zamanda nesne şablonu olarak da düşünülebilir.

Nesne (Object, Instance)

Tanımlı bir sınıfın programda kullanılan bir örneğidir ya da olumudur. Sınıfların amacı, bir yapıyı bir kez tanımlamak, isimlendirmek ve tekrar tekrar kullanılmasını sağlamaktır. *Bknz: Sınıf (Class)*. Program yazılırken, bir sınıf tanımına ait gerektiği sayıda nesne yaratılıp kullanılabilir. Bu nesneler, sınıf tanımındaki özelliklere sahiptir ve yöntemleri gerçekleştirebilir. Aynı program parçası içinde, sınıflardan türemiş her bir nesnenin tekrarsız bir adı olmak zorundadır ve nesne bu isimle kullanılır.

Olay Sürmeli (Event Driven) Programlama

Bir program, bir başlangıç noktasından başlayıp, sırayla devam edip bir yerde bitmiyorsa, kod parçacıklarının çalışması olaylara bağlanmışsa buna olay sürmeli programlama diyoruz. Win32 görsel formlarının olay sürmeli programlanması zorunludur. Çünkü kullanıcı form üzerinde işlem yaparken bir sıra takip etmek zorunda değildir ve birbiri arkasından çok farklı olaylar meydana gelebilir. Örneğin bir bilgi giriş sahasına giriş, bilgi giriş sahasından çıkış, başka bir bilgi giriş sahasına uğramadan bir butona basılması gibi. O nedenle, form, üzerinde oluşabilecek her olay için bir kod parçacığı yazılarak yönetilmelidir.
Form ve formun üzerindeki görsel bileşenlerin her biri kendi sınıf tanımının özellik ve yeteneklerini içeren birer nesnedir. *Bknz: Nesne (Object, Instance)*. Her nesnenin olmamakla birlikte çoğunun nesne girişi, çıkışı, tıklanması gibi olayları mevcuttur. Nesneler, sınıflarından gelen yeteneklerinden dolayı bazı işlemleri doğal olarak her seferinde aynı şekilde yapabilir. Örneğin, nümerik veri girişi alanı sınıfından türemiş ve form üzerine konmuş tüm nesneler, nümerik karakterleri alacak, alfabetik karakterleri kabul etmeyecek, belki basamak gruplaması ve ondalık gösterim gibi yetenekleri sergileyecek tir. Ancak bu nesne, Örneğin, iki tane nümerik sahadaki değerlerin çarpılarak başka bir sahada gösterilmesi gibi bir yeteneğe sahip değildir. İstenen bu işlem, nesne çıkışlarına ya da bir buton tıklanması olayı için ayrıca yazılmalıdır.
Dinamik kodlamada, bir form üzerinde mevcut görsel bileşen nesnelerinin olayları

gerçekleşmesi sırasında çalışacak kod parçacıkları yazılabilmektedir. O nedenle kod yazımız öncesinde, hangi nesnenin hangi olayı için kod yazıldığı belirlenir. *Bknz: Nesne İzleme Bölümü, Nesne Olayı (Event) Bölümü*.

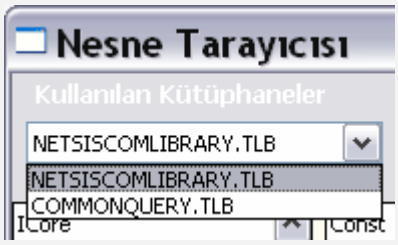
Kalıtım (Inheritance) ve Sınıf Hiyerarşisi
Mevcut bir sınıfın yapısının, yetersiz kaldığı ve sınıftan yaratılacak nesnenin ek özellikler ve yetenekler içermesi gerektiği durumda, mevcut sınıfın yapısını ve mevcut yaratılmış nesnelerdeki kullanımını bozmadan, bu sınıfın yöntem ve özelliklerini miras alan yeni bir sınıf yapısı tanımlanabilmektedir. Yaratılan sınıf, türediği üst sahibinin (owner), özellik ve yöntemlerini miras alır. Yeni sınıfa yeni özellik ve yöntemler ilave edilebilir, sahip sınıf tanımında izin verilen özellik ve yöntemlerde değişiklik yapılabilir. Görsel programlama araçlarında, program yazımında kullanılabilecek sınıf kütüphaneleri mevcuttur ve içinde birçok sınıf barındırır. Bu kütüphanelerde sınıf hiyerarşisi de mevcuttur.

3.3.4.4 Netsis Kütüphanesi

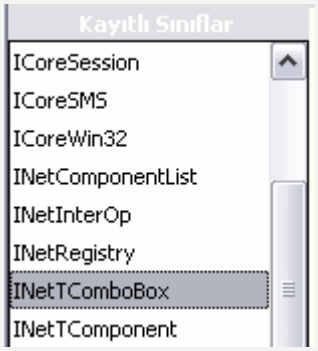
Netsis, dinamik kodlama içinden, içinde bulunulan formda ya da uygulama genelinde mevcut nesnelerin özellik ve yöntemlerine (görsel bileşenlerin içerikleri, global değişkenler, Netsis uygulamasında yapılabilen e-mail, sms gönderme gibi birtakım işlemler) erişimi mümkün kılmak için, ortamda geçerli kütüphaneler sağlar.
Netsis kütüphaneleri, paketin kurulumu sırasında sisteme yüklenir ve kayıt edilir. Ancak kütüphanelerin yenilenmesi ya da tümüne erişim sağlanamaması durumunda, kurulum sırasında yaratılan SERVIS klasöründeki (NETSIS/FUSION06/SERVIS), RegKontrol.Exe çalıştırılarak kütüphanelerin tekrar kayıt edilmesi sağlanabilir.

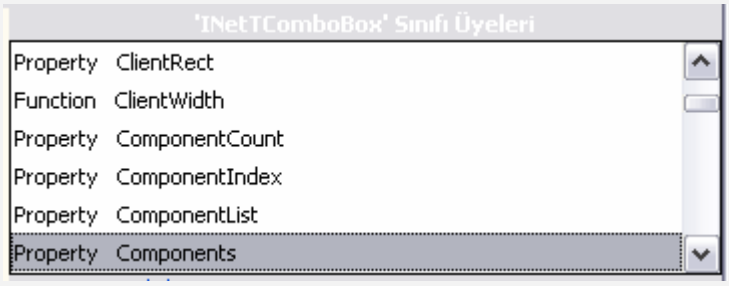
3.3.4.5 Nesne Tarayıcısı ( )

Kullanılan Kütüphaneler
Kodlama ortamında hangi kütüphanelerin kullanılabileceği, kodlama bölümünün araç çubuğundaki "*Nesne Tarayıcısı*" butonu yardımıyla izlenebilir. Nesne Tarayıcısının sol üst köşesinden açılan listede, kullanılabilecek kütüphaneler görülebillir.



Kayıtlı Sınıflar

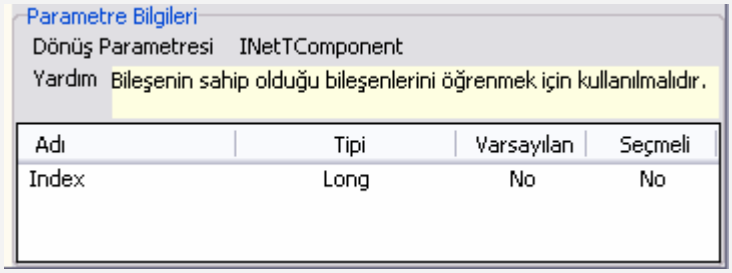




Netsis kütüphaneleri ile kayıt edilen ve script kod içinde kullanılabilecek sınıfların isimleri listede yer alır. Herhangi bir sınıfla ilgili bilgi alabilmek için sınıf üzerine tıklanmalıdır. Seçilen sınıfın özellik (property) ve yöntemleri (function, procedure) sağ taraftaki bölümde listelenir. Herhangi bir özellik ya da yöntemin de tek tıklanmasıyla seçimi sonrasında, aynı pencerenin alt kısmında, yöntemin geriye döndürdüğü parametrenin tipi, yöntemin kısa tanımı ve varsa yöntemi kullanırken gönderilmesi gereken parametrelerin listesi yer alır.

3.3.4.6 Kütüphanelerdeki Sınıflar

Kodlama sırasında kullanılabilecek nesnelerin kısayol yardımı için *bknz: Araç Çubuğu – 2 (Sağ Üst) / Kod İçeriği (Code Insight).*



NetsisCore (ICore): Netsis'in uygulama çalışma sırasında bellekte bulunan aktif kullanıcı, aktif modül, aktif menü seçeneği, entegrasyon tarihi vb. kor (çekirdek) bilgilerine ve Netsis paketinin altyapısında kullanılan E-posta, SMS gönderme fonksiyonları, kredi kartı doğrulama, seslendirme, onaylı sürüm kontrolü, veri tabanı sorgulama nesneleri yaratılması vb. yöntemleri barındıran ana sınıftır. Sınıf ismi "ICore", kodlama ortamında kullanılbilecek nesne ismi ise, "NetsisCore" dur.
Nesne Tarayıcısında "ICore" ile başlayan alt sınıfları içerir. Kodlama ortamında kullanılabilecek nesneleri ise, "ICore" yerine "NetLib" ile başlar, geri kalan bölümü ise aynıdır. Örnek: Sınıf = "IcoreDB", Nesne = "NetLibDB". Her bir alt sınıfın kendi yöntemleri vardır. Yöntemler, NetsisCore.AltSınıf.Yöntem şeklinde kullanılmalıdır. Örnek: "NetsisCore.NetLibDB.GetNewQuery", kor (çekirdek) kütüphanenin veritabanı ile ilgili alt nesnesinin sorgu nesnesi yaratma yöntemini çalıştırır.
NetsisVCL: Birçok görsel bileşen sınıfı içeren Netsis Görsel Bileşen Kütüphanesidir (Visual Component Library). Nesne tarayıcısında, NetsisVCL içindeki görsel bileşen sınıfları "INetT" ile başlar. Kodlama ortamında görsel bileşen nesnesinin kullanılması için doğrudan ilgili nesnenin adı yazılabilir.
CommonQuery: Netsis kor (çekirdek) kütüphanesinin (NetsisCore), veritabanı altnesnesinin (NetlibDB), sorgu yaratma yöntemiyle (GetNewQuery) yaratılan sorgu nesnesidir. Sorgu nesnesinin, SQL cümlesi, kayıt sayısı, aktif olup olmadığı, kayıt kümesinin sonuna gelip gelmediği, saha bilgileri vb. özellikleri ile, sonuç kümesi dödürülmesi, ilk kayıt, sonraki, önceki, son kayıt vb. yöntemleri mevcuttur.
Self: Kodun çalışması sırasında aktif olan nesne için standart kullanımdır. Özellikle, kod yazımı sırasında, kodun hangi nesne için çalıştığı bilinmeyen durumda kullanılabilir. Dinamik kodlama için self, kodlama öncesinde sol bölümdeki "nesne izleme" listesinden seçilmiş olan ve sonrasında, kodun hangi olayı (event) için yazıldığının belirlendiği nesnedir. Self.özellikadı ya da self.yöntemadı formatında yazılarak aktif nesnenin bilgilerine erişmek ya da yöntemini çalıştırmak mümkündür.

3.3.4.7 Araç Çubuğu – 2 (Sağ Üst)

 **Kod Tamamlama (Code Complete):** Belli kalıplarda kod bloklarının otomatik olarak tamamlanması için kullanılabilir. Örneğin "if" ibaresi yazılarak bu tuşa basıldığında, bir if-then-else bloğunun düzgün syntax ile tamamlanması sağlanabilir.

 **Kod İçeriği (Code Insight):** Script içinde kullanılabilecek hazır nesneleri ile nesnelere ait özellik (property) ve yöntemlerin (method) listesinin getirilebileceği tuş. Script giriş editöründe, Ctrl-boşluk tuşu ile aynı işleve sahiptir. Editörde Ctrl-Boşluk tuş takımı ya da araç çubuğundaki bu buton ile, kullanıma açılmış olan nesneler getirilebilmektedir. Ctrl-Boşluk ile açılan "Netsis Code Insight" listesinden bir nesne seçilip tekrar ctrl-boşluk ile seçilen nesnenin özelliklerini getiren yeni bir Code Insight penceresi açılabilir. Buradan da nesnenin istenen kullanılacak özelliği seçilebilir.

 **Git (Go To):** Bu butona basıldığında gitmek istediğiniz satır numarası sorgulanacak ve cursor kod içerisinde verilen satır numarasına gidecektir.

 **Ara:** Kod içerisinde belli bir karakter dizisiniz aramak amacıyla kullanılır. Aranması istenen karakter dizisi, büyük/küçük harf duyarlı, sadece kelime, imleçten itibaren ara, sadece seçili metinde ara seçenekleri ile ileri/geri yönde aranabilmektedir.

 **Ara Sonraki, Önceki:** Kod içerisinde arama yapılıp verilen karakter dizisi bulunduktan sonra, aynı karakter dizisinin, verilen arama kriterleri ile, sonrakini/öncekini bulmaya yarayan butonlardır.

 **Bul/Değiştir:** Arama işlevinin, bulunan karakter dizisi yerine başka bir karakter

dizisini otomatik olarak yerleştirme amacıyla kullanılması içindir. Aranacak karakter dizisi ile yerine yerleştirilecek karakter dizisi belirtilmelidir. Büyük/küçük harf duyarlı, sadece kelime, imleçten itibaren ara, sadece seçili metinde ara seçenekleri ile ileri/geri yönde arama geçerlidir. İlk bul/değiştir işleminden sonra, sonraki/önceki için yukarıda belirtilen butonlar kullanılabilir.

 **Script Sakla/Yükle:** Yazılan scriptin dış ortama text dosya olarak aktarımı ya da dış ortamdaki bir script'in text dosyadan editöre yüklenmesi.

3.4 NETSİS SCRIPT KOD ÖRNEKLERİ

3.4.1 Örnek – 1: Cari Hesap Kayıtları Sağ Click menüsünden anında cari hesaba e-posta gönderimi

Cari Hesap Kayıtlarında, Cari Kod kısmında bulunan cari koduna ait o andaki borç ve alacak bakiyesini carinin mail adresine yollayan bir uygulama.

```
CARI=CARI_KOD.TEXT
set QUERY = NETSISCORE.NetLibDB.GetNewQuery
SORGU="SELECT SUM(BORC),SUM(ALACAK),GETDATE() FROM TBLCAHAR WHERE CARI_KOD='"&CARI&"'"
QUERY.recsql(SORGU)

set QUERY1 = NETSISCORE.NetLibDB.GetNewQuery
SORGU1="SELECT EMAIL,CARI_ISIM FROM TBLCASABIT WHERE CARI_KOD='"&CARI&"'"
QUERY1.recsql(SORGU1)

ICERIK="SAYIN "&QUERY1.fields(1).asstring&"; "&QUERY.fields(2).asstring&" tarihi itibariyle borç bakiyeniz "
ICERIK=ICERIK&QUERY.fields(0).asstring&" ve alacak bakiyeniz "&QUERY.fields(1).asstring&" dir."

MAIL=QUERY1.fields(0).asstring
call  appglobal.mail(MAIL,"","MEKTUP","",ICERIK)
```

Burada öncelikle cari hesap kayıtlarında cari kodu editbox'ında bulunan değeri CARI isimli bir değişkene atılıyor ve Netsis'in nesnelerinden NETSISCORE.NetLibDb.GetNewQuery ile QUERY isimli yeni bir sorgu nesnesi yaratılıyor. SORGU değişkenine çalıştırmak istenilen sorgu yazılıyor. Burada TBLCAHAR tablosundan borç ve alacak toplamı ile günün tarihini çeken bir sorgu yazılmıştır. Ardından QUERY nesnesi ile bu sorguyu çalıştırıyoruz.

İkinci olarak carinin ismi ve mail adresi gerektiğinden bu bilgilerde QUERY1 ile çekiliyor. Ve içerik isminde mailin içeriğini saklayan bir değişken tanımlanıyor. Bu değişkene veritabanından çekilen verilerin de eklenmesi gerekmektedir. QUERY1 ile çekilen değerlerden CARI_ISIM sahasını içeriğe eklemek için QUERY1.fields(1).assstring tanımlaması kullanılabilir. Carinin EMAIL adresini eklemek için ise QUERY1.fields(0).assstring tanımlaması kullanılabilir yani sorgudaki ilk saha için 0 indeks'ten başlanarak veriler alınabiliyor.
ICERIK değişkeni tamamlandıktan sonra uygulama global'de tanımlanan mail sub'ı kullanılarak cariye mail atılması sağlanabiliyor.

3.4.2 Örnek – 2: Girilen Stok Koduna göre Depo Kodunun Oluşturulması

Stok kodu girildiğinde girilen kod'a göre oluşan değerin depo kodu sahasına atılması istenebilir. Stok koduna IZM001,IZM002 gibi kodlar yazıldığında stoğun depo koduna 1 yazılması; IST001,IST002 şeklinde kodlar yazıldığında depo koduna 2 yazılması aksi durumda 3 yazılması isteniyorsa;

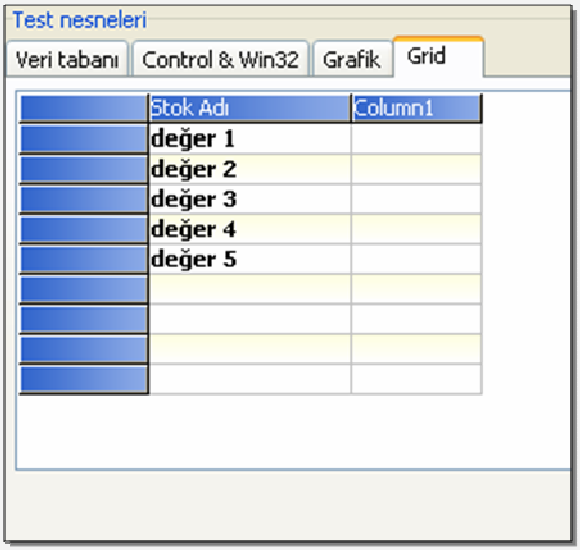
```
IF LEFT(STOK_KODUDB.TEXT,3)="IZM" THEN
DEPO_KODU.Text="1"
ELSEIF LEFT(STOK_KODUDB.TEXT,3)="IST" THEN
DEPO_KODU.TEXT="2"
ELSE
DEPO_KODU.Text="3"
END IF
```

Burada stok kodu bilgisi edit olduğu için TDBNEdit altında bulunabilir. Yukarıdaki tanımlamaya bakıldığında Stok_kodudb'nin OnExit olayında script yazılmıştır ve stok

koduna yazılan değerin ilk 3 hanesini alarak IZM olup olmadığı kontrol ediliyor eğer IZM ise depo koduna 1 yazılıyor. Depo kodu bilgisi de edit olduğu için Depo_Kodu.TEXT olarak atama yapılmıştır. Eğer IZM değilse kontrole devam ederek IST olup olmadığına bakılıyor doğru ise 2 aksi durumda 3 olarak depo kodu belirleniyor.

3.4.3 Örnek – 3: INetStrGrid (Netsis grid nesnesi)

Aşağıdaki sıralama, StrGrid nesnesinin hangi sınıf hiyerarşisinin parçası olduğunu göstermektedir. Bu yapıya göre, StrGrid nesnesi, hiyerarşide kendinden önce gelen(atası olan) tüm sınıfların özelliklerini, otomatik olarak desteklemektedir.



Sınıf Hiyerarşisi

INetTObject → INetTComponent → INetTControl → INetTWinControl → INetStrGrid

StrGrid Nesnesi, tüm Netsis paketlerinde kullanılan *grid* nesnesidir .

***StrGrid* nesnesine ait örnek görünüm**

```
'*****************************************
'StringGrid nesnelerine ait test kodları
'
'sgTest : String Grid nesnesidir
'*****************************************

'İlk kolon uzunlugunu 100 pixel yap
SGTEST.StrColumn(0).Width = 100

'İlk kolonda gösterilen bilgileri bold göster
SGTEST.StrColumn(0).Font.Bold = true
SGTEST.StrColumn(0).Title = "Stok Adı"

'İlk kolonun 1..5 satırlarına değer 1..5 yaz
for I=1 to 5
  sgtest.cells(1,I) = "değer " & i
next

'Grid üzerindeki boyama işlerini gözden geçir
sgtest.Invalidate

'Grid nesnesine ait önemli bilgiler gösteriliyor
GridBilgi = "aktif kolon " & sgtest.Col & chr(13) & chr(10) &_
  "aktif satır " & sgtest.Row & chr(13) & chr(10) &_
  "sabit satır sayısı  " & sgtest.FixedRows & chr(13) & chr(10) &_
  "sabit kolon sayısı " & sgtest.FixedCols & chr(13) & chr(10) &_
  "Solda görünen kolonun numarası " & sgtest.LeftCol & chr(13) & chr(10)
&_
```

```
  "Standart satır yüksekliği " & sgtest.DefaultRowHeight & chr(13) &
chr(10)

MsgBox GridBilgi
```

3.4.4 Örnek – 4: INetStrColumn

Bu tür nesnelere doğrudan erişim yapılamaz. StrGrid veya XGrid nesnelerindeki StrColumns özelliği kullanılarak erişim yapılabilir ve özellikleri kullanılabilir. StrGrid ve Xgrid nesnelerinin kolon bilgilerine erişim için kullanılan bir sınıftır.

3.4.5 Örnek – 5: INetXGrid (Veri tabanı bağlantılı grid nesneleri)

Sınıf Hiyerarşisi

INetTObject → INetTComponent → INetTControl → INetTWinControl → INetStrGrid →INetTXGrid

Tüm Netsis paketlerinde kullanılan, veri tabanı bağlantılı grid nesnesidir.
NDI paketinde, XGrid nesnesi, hazırlanan ekrandaki değişik amaçlara göre kısıtlanabilir. Eğer stok hareket ekranına benzer bir tasarım yapılacaksa, stok kodu nesnesinin çıkışında xgrid nesnesinin SQL özellik değeri, aktif kod kısıtına göre hazırlanabilir ve kayıtlar gösterilebilir.

```
'****************************************
'XGrid nesnelerine ait test kodları
'
'xgTest : XGrid nesnesidir
'****************************************

'Grid üzerinde gösterilecek kayıtlar için SQL cümlesi hazırlanıyor
SQLStr = "select top 10 * from tblcasabit " &_
      "where cari_kod like 'A%' "
XGTEST.Sql.Text = SQLStr

'Hazırlanan sql cumlesine gore kayıtları göster
XGTEST.Active = true

'grid nesnesindeki kayıtlar tekrar sunucudan alınıyor
XGTEST.Active = false
XGTEST.Active = true
```

3.4.6 Örnek – 6: INetGraphLibrary (Netsis Grafik Kütüphanesi)

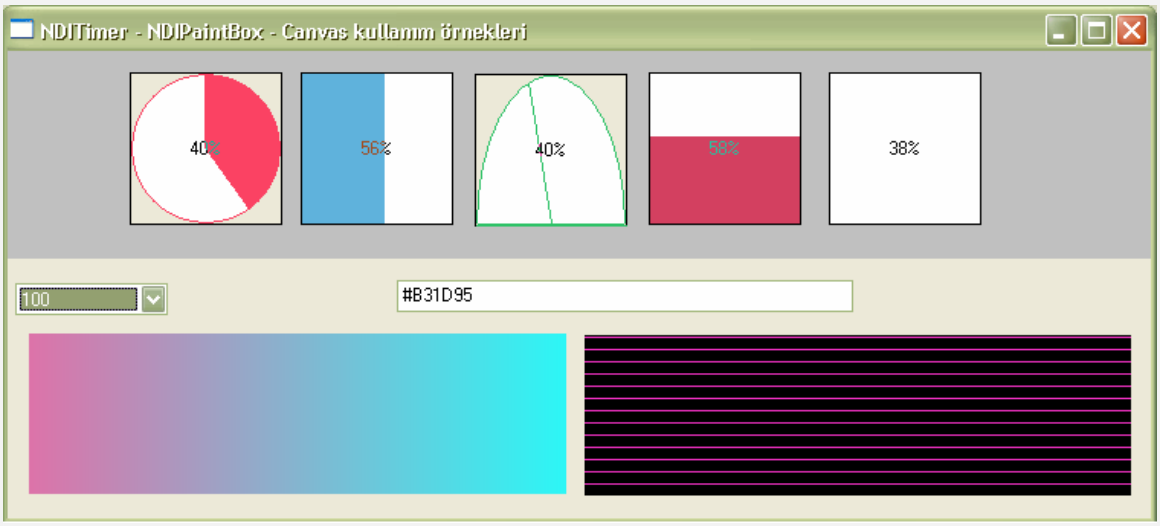
ICanvas, IBrush, IPen, IRec, IPoint ve IGraphUtil arayüzlerini tanımlar. IGraphUtil; ICanvas, IBrush, IPen, IRec nesnelerinin kullanımı için yardımcı yöntemleri tanımlar. Kütüphanenin sağladığı, çizim, boyama ve yazı yazma gibi grafik işlemleri kullanımı örnekleri aşağıdaki gibidir.

1. Aşağıdaki örnek, *NDI* paketi ile hazırlanan bir ekranın, dinamik kodalma ile, grafiksel olarak şekillendirilmesini göstermektedir.

```
'*************************************************************************
*
'FORM GLOBAL TANIMLAMA ALANI
'*************************************************************************
*
'Rastgele bir renk türetmek için kullanılacak olan yöntem, uygulama içinde
birden 'çok yerde kullanılacağı için global olarak tanımlanıyor.
Function RenkTuret()
  RenkTuret = Int(rnd(1)*2^24)
End Function
'TNDITimer bileşeninin zaman aralığını dinamik olarak değiştirmek için
kullanılacak 'yöntem, global olarak tanımlanıyor. Yöntem, ekran üzerindeki grup
kutusundan yeni 'bir aralık seçilmesi durumunda, TNDITimer nesnesi aralık
değerini değiştirir
Sub TimerAta
 NDITIMER1.PropertyValue("TimerInterval") =
CINT(CBTIMERHIZ.propertyValue("Text"))
End Sub
'*************************************************************************
*
```

```
'NDI TIMER "ONTIMER" OLAYI TANIMLAMA ALANI
'TNDITimer nesnesi "OnTimer" olayı ile, belirli aralıklarla(NDITimer.Interval),
'ekran üzerindeki bilgilerin güncellenmesi sağlanıyor
'*************************************************************************
*
'Panel üzerindeki TGauge(sayaç) nesneleri, ölçüm değerlerinin ve renklerinin
değiştirilmesi
For i=1 To PANEL1.ControlCount
  PANEL1.Controls(i-1).PropertyValue("Progress") = Int(rnd(1)*100)
  PANEL1.Controls(i-1).PropertyValue("ForeColor") = RenkTuret
Next
'TNDIPaintBox nesneleri üzerinde grafiksel değişiklikler yapmak için "Canvas"
'nesnelerinin değişkenler aracılığı ile kullanımı için gereken atamalar
Set C1 = NDIPAINTBOX1.PropertyValue("Canvas")
Set C2 = NDIPAINTBOX2.PropertyValue("Canvas")
' IGraphUtil nesnesi GradientFillCanvas yöntemi ile boyama işlemi
Set GL = NETSISCORE.NetLibGraph
Call GL.GradientFillCanvas(C1,RenkTuret,RenkTuret,C1.ClipRect,Int(Rnd(1)))
' IBrush nesnesi ile boyama işlemi.
C2.Brush.Color = RenkTuret
C2.Pen.Color = RenkTuret
C2.Brush.Style =  Int(Rnd(1)*6)
C2.FillRect( C2.ClipRect )
```

Yukarıda dinamik kodlama tanımlamalarının verildiği ve NDI ile tasarlanan ekranın çalışma anındaki görüntüsü aşağıdaki gibidir.



2. Aşağıdaki örnek; *Grid* nesnesi ***OnChangeCanvasRect*** olayı kullanılarak grid hücreleri görünümünde ve içeriğinde değişiklik yapılması için kullanılmaktadır.

```
'Progbil için tasarlanan örnek script kodu, SGBilgi grid nesnesinin
'"OnChangeCanvasRect" olayına atanıyor

if ( Col = 2)  then

  Set GL = NETSISCORE.NetLibGraph

  StartCol = GL.WEBColorToColor("clwebPink")
  StopCol = GL.WEBColorToColor("clwebLightGreen")
  GL.GradientFillCanvas CellCanvas, StartCol, StopCol, Rect, 0

  CellCanvas.Brush.Style = bs_Clear
  CellCanvas.TextOut rect.left, rect.top, SGBILGI.Cells(COL,ROW)

end if

'****NOT:*****
'OnChangeCanvasRect olayı, grid çizimi adımında, hücrelerin, grafik ve içerik
olarak değiştirilebilmesine olanak sağlamaktadır.
```

Uygulamanın çalışması sonucu oluşan çıktı aşağıdaki gibidir.

3.4.7 Örnek – 7: INetComponentList

INetComponent sınıfından türeyen tüm nesnelerde "ComponentList" fonksiyonu kullanıldığında INetComponentList sınıfından bir nesne hazırlanmaktadır. Bu nesne ile form veya panel gibi taşıyıcı nesnelerin içinde yer alan nesneler, sınıf isimi verilerek listelenebilir.

```
'Graphic nesnesi hazırlanıyor
Set GraphLib = NETSISCORE.NetLibGraph

'Ekran nesnesindeki TEdit sınıfına ait örnekler listeye alınıyor
Set EditListesi = Self.ComponentList("TEdit")

'Listede kayıt var mı?
if EditListesi.Count > 0 then
  'Listedeki tüm kayıtları dön
  For i=1 To EditListesi.Count
    'Listedeki aktif nesneye erişim
    Set edObj = EditListesi.Items(I - 1)
    'erişim yapılan nesnenin renk özelliği "domates" rengine atanıyor...
    edObj.PropertyValue("Color") = GraphLib.WEBColorToColor
("clwebTomato")
  Next
End If
```

3.4.8 Örnek – 8: INetTWEBBrowser

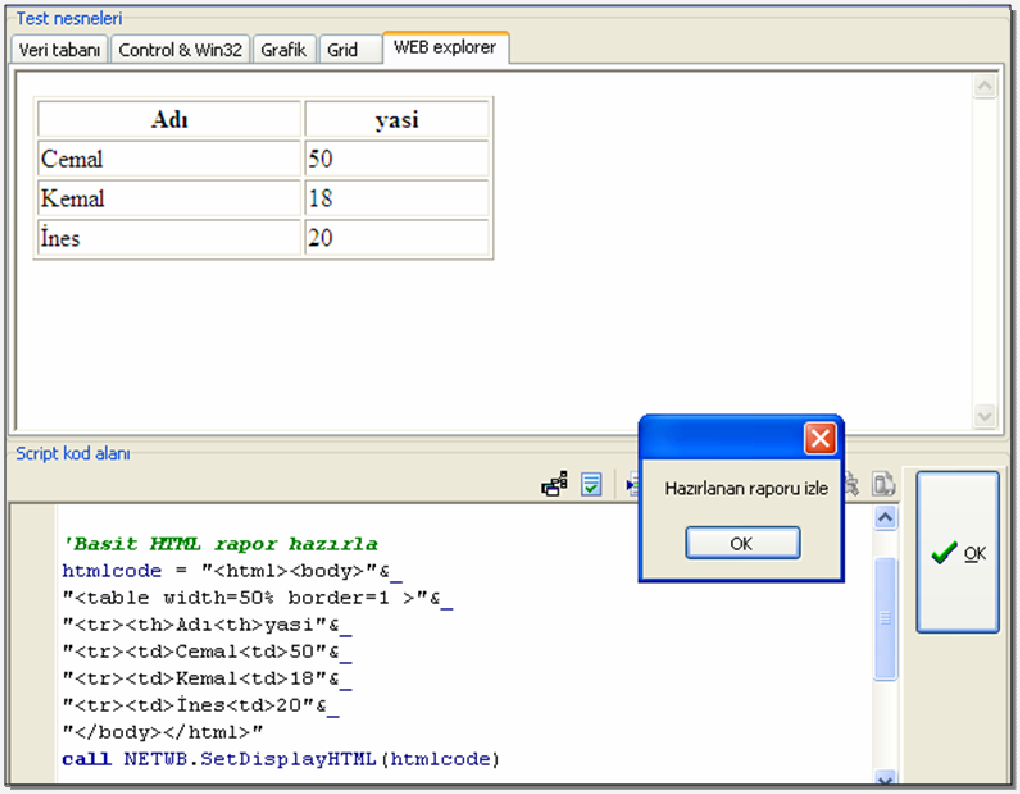
WEB sayfaları ve HTML düzeninde hazırlanan bilgilerin gösterilmesi için tasarlanmıştır. NDI paketinde Win32 ortamında, WEB sekmesinde TNetWEBBrowser nesnesi için kullanılabilir.

```
'WEB nesnesini ilk kullanım için hazırla
call NETWB.Navigate("about:blank")

'Basit HTML rapor hazırla
htmlcode = "<html><body>"&_
"<table width=50% border=1 >"&_
"<tr><th>Adı<th>yasi"&_
"<tr><td>Cemal<td>50"&_
"<tr><td>Kemal<td>18"&_
"<tr><td>İnes<td>20"&_
"</body></html>"
'Hazırlanan HTML bilgisinin ekranda görünmesi için güncelle
call NETWB.SetDisplayHTML(htmlcode)
```

```
msgBox "Hazırlanan raporu izle"

'NetWB nesnesini Netsis ana sayfasına yönlendir
call NETWB.Navigate("http://www.netsis.com.tr")
```



Yukarıdaki örnek koda ait ekran görüntüsü

3.4.9 Örnek – 9: INetControl

"Perform" ve "ClientRect" özellikleri eklendi.

Perform komutuna ait bir örnek

```
'*************************************************************************
'CB_GETITEMHEIGHT windows mesajı nedir?
'*************************************************************************
'
'ComboBox nesnelerinde yer alan satırların yükselik
'değerini geriye dönmektedir.
'Bu örnek, perform komutunun kullanımı için kullanılmıştır.
'Windows API seviyesinde dinamik kodlama için Windows SDK veya benzeri
'kaynaklardan yardım alınması gerekmektedir...
'
'*************************************************************************

'Windows mesaj listesindeki özel değer tanımlanıyor...
Const CB_GETITEMHEIGHT = 340
'Bu tür mesajlar için Windows SDK yardımları kullanılabilir

'Ekrana Windows mesajının cevabını yazdır!
msgbox COMBOBOX1.Perform( CB_GETITEMHEIGHT, 0, 0 )
```

3.4.10 Örnek – 10: INetObject

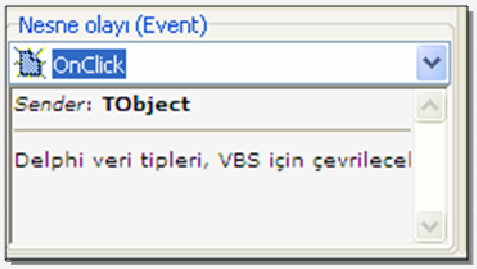
InvokeEvent yöntemi eklendi.

InvokeEvent komutuna ait örnek

```
'Button nesnesindeki "OnClick" olayının (event) tetiklenmesi

'OnClick gibi olay isimlerini öğrenmek için Dinamik kodlama
'tasarım ekranından yardım alınabilir.

BTNSAKLA.InvokeEvent("OnClick")
```



OnClick olayının Dinamik Kodlama Ekranından izlenmesi

Not: InvokeEvent yöntemi sadece tek parametreli olaylar için kullanılmalıdır. (Sender: Tobject) parametre deseninde olmayan olaylarda çalıştırılması sonucunda hata alınacaktır.

3.4.11 Örnek – 11: ICoreSession

"EntegrasyonTarihi" özelliği eklendi.

3.4.12 Örnek – 12: ICoreLocalizationConvert

Genelde Iran kültüründe ihtiyaç duyulan fonksiyonlar için kullanılabilir. Normal takvim sistemindeki bir değer Farsi sisteme, Farsi tarih sistemdeki tarih değeri de normal takvime çevrilebilir.

```
'Localization servis nesnesine erişim yap
Set NetLocalizationService = NETSISCORE.NetLibLocalizationConvert

'Normal takvim bilgisi Iran takvim sistemine çevriliyor
'****NOT:*****
'Iran takvim sistemi, Netsis Çalışma Kültür değerinin Iran'a göre
'ayarlanması durumunda çalışacaktır...
MsgBox
NETSISCORE.NetLibLocalizationConvert.StrDateToStrFarsi("01/01/2007")
```

3.4.13 Örnek – 13: ICoreWin32

KrediKartNoDogrula, OzelParamVarMi , OzelParamDegerOku, YaziIlePara, YaziIleParaIng, WinPostMessage, WinSendMessage yöntemleri eklendi.

Özel Parametre Kontrolü

```
Dim BasimParamAcik
BasimParamAcik = NETSISCORE.NetLibWin32. OzelParamVarMi("FATURA",
"BASIM")
```

WinPostMessage Kullanımı

```
'Windows form kapanış özel mesaj değeri tanımlanıyor...
Const WINDOWS_MESSAGE_FORM_CLOSE = 16

'Windows mesaj kuyruğuna zamanuyumsuz (asynch) mesaj gönderimi için kullanılır

'Zamanuyumlu mesaj gönderimlerinde (synch) WinPostMessage yerine
'WinSendMessage yöntemi kullanılmalıdır.

'*************************************************************************
'Windows mesaj yönetimi için Windows-SDK yardımı kullanılabilir..
'*************************************************************************

CALL NETSISCORE.NetLibWin32.WinPostMessage(SELF.HANDLE,
WINDOWS_MESSAGE_FORM_CLOSE, 0, 0)
```

3.4.14 Örnek – 14: ICoreSMS

ICore. NetLibSMS yöntemi ile yaratılan nesne, tek ya da toplu olarak SMS gönderimi için kullanılmaktadır. Desteklediği yöntemler: AddMessage, SendMessage ve

SendMessages yöntemleridir. Toplu mesaj gönderimlerinde, mesajlar AddMessage yöntemi ile eklenmeli ve tüm mesajlar eklendikten sonra SendMessages yöntemi çağırılmalıdır. Sadece bir mesaj gönderimi durumunda ise; SendMessage yöntemi çağırılmalıdır.

```
'ICoreSMS nesnesi yaratılıyor
Set SMSObj = ICore. NetLibSMS( "Netsis", now, now+2 )

'Mesaj gönderiliyor
Call
SMSObj.SendMessage("Netsis",now,now+2,"Deneme","00905333333333","01","Netsis")
```

3.5 UYARILAR & EK BİLGİLER

1. Netsis dinamik kodlama sisteminde yer alan Visual Component Library (NVCL) nesne katmanları Borland Delphi mimarisine uyumlu olacak şekilde tasarlanmıştır. (Borland Delphi mimarisi ile ilgili temel bilgiler için http://www.marcocantu.com/edelphi/EssentialDelphi_103.zip bağlantısından yararlanabilirsiniz.)
2. Nesne, sınıf, yöntem ve özellik isimlerinin genelinde Borland isimlendirme tekniği değiştirilmeden kullanılmıştır.
3. Netsis dinamik kodlama sisteminde yer alan örnekler tamamen konunun öğrenilmesine yönelik hazırlanmış çalışmalardır. Buradaki örnek kodların her durumda çalışması ve desteklenmesi söz konusu değildir.
4. Windows mesajlaşma sisteminde uygun mesajın uygun yerlerde kullanılması gerekmektedir. Aksi takdirde uygulamada beklenmeyen sorunlar ortaya çıkabilir.
5. Windows'un bazı özel mesajlarının uygulamada sorun yaratması durumunda daha sonraki sürümlerde kapatılabilir.

3.6 REFERANSLAR

3.6.1 WEB Renk Değerleri

Aşağıdaki tabloda yer alan renk değerlerini NETSISCORE.NetLibGraph .WEBColorToColor yönteminde başına "clWEB" sabit değerini ekleyerek kullanabilirsiniz.

```
edObj.PropertyValue("Color") = GraphLib.WEBColorToColor ("clwebTomato")
```

AliceBlue	Gainsboro	MistyRose
AntiqueWhite	GhostWhite	Moccasin
Aqua	Gold	NavajoWhite
Aquamarine	GoldenRod	Navy
Azure	Gray	OldLace
Beige	Grey	Olive
Bisque	Green	OliveDrab
Black	GreenYellow	Orange
BlanchedAlmond	HoneyDew	OrangeRed
Blue	HotPink	Orchid
BlueViolet	IndianRed	PaleGoldenRod
Brown	Indigo	PaleGreen
BurlyWood	Ivory	PaleTurquoise
CadetBlue	Khaki	PaleVioletRed
Chartreuse	Lavender	PapayaWhip
Chocolate	LavenderBlush	PeachPuff
Coral	LawnGreen	Peru

CornflowerBlue	LemonChiffon	Pink
Cornsilk	LightBlue	Plum
Crimson	LightCoral	PowderBlue
Cyan	LightCyan	Purple
DarkBlue	LightGoldenRodYellow	Red
DarkCyan	LightGray	RosyBrown
DarkGoldenRod	LightGrey	RoyalBlue
DarkGray	LightGreen	SaddleBrown
DarkGrey	LightPink	Salmon
DarkGreen	LightSalmon	SandyBrown
DarkKhaki	LightSeaGreen	SeaGreen
DarkMagenta	LightSkyBlue	SeaShell
DarkOliveGreen	LightSlateGray	Sienna
Darkorange	LightSlateGrey	Silver
DarkOrchid	LightSteelBlue	SkyBlue
DarkRed	LightYellow	SlateBlue
DarkSalmon	Lime	SlateGray
DarkSeaGreen	LimeGreen	SlateGrey
DarkSlateBlue	Linen	Snow
DarkSlateGray	Magenta	SpringGreen
DarkSlateGrey	Maroon	SteelBlue
DarkTurquoise	MediumAquaMarine	Tan
DarkViolet	MediumBlue	Teal
DeepPink	MediumOrchid	Thistle
DeepSkyBlue	MediumPurple	Tomato
DimGray	MediumSeaGreen	Turquoise
DimGrey	MediumSlateBlue	Violet
DodgerBlue	MediumSpringGreen	Wheat
FireBrick	MediumTurquoise	White
FloralWhite	MediumVioletRed	WhiteSmoke
ForestGreen	MidnightBlue	Yellow
Fuchsia	MintCream	YellowGreen

3.6.2 Windows mesajlaşma sistemi (Windows Messaging)

Windows işletim sisteminde kullanılan mesajlar için http://msdn.microsoft.com/library/default.asp?url=/library/en-us/winui/winui/windowsuserinterface/windowing/messagesandmessagequeues.asp adresi kullanılabilir.

3.6.3 GDI (Graphics Device Interface)

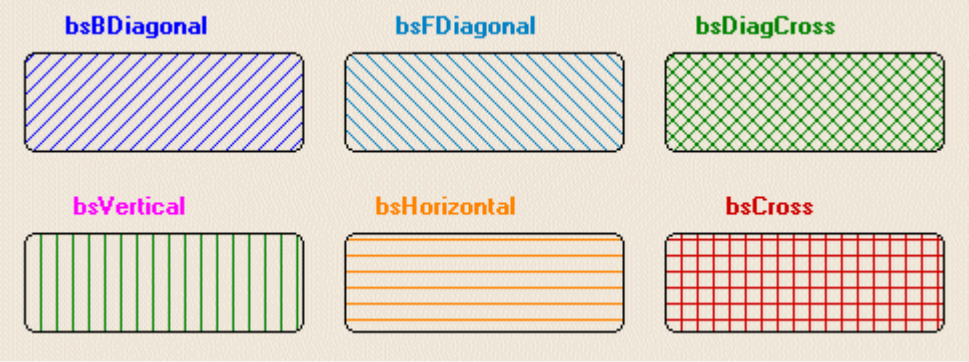
Windows GDI (grafik aygıt arabirimi), grafik çıktıyı görüntülemede kullanılan bir takım API lerden (Application Programming Interface) oluşur.

3.6.3.1 CANVAS

Canvas nesnesi, nesnelerin resimlerini şekillendirmek için kullanılan bir çizim yüzeyi olarak ifade edilmektedir. *Canvas* nesnesi, özellikleri, olayları ve yöntemleri ile, nesnelerin, grafiksel olarak, çizim, boyama ve yazı yüzü özelliklerini belirlemek için kullanılmaktadır

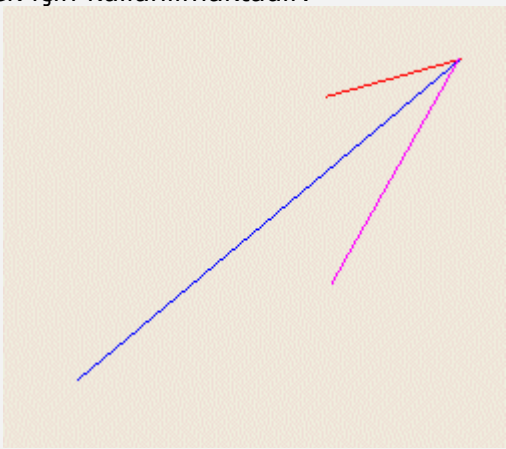
3.6.3.2 BRUSH

Brush, kapalı şekilleri doldurmak için kullanılan bir araçtır. *Brush* nesnesinin taşıdığı renk (brush.color), boyanacak alanın doldurulması için kullanılacaktır ve doldurma işlemi ile(fill), seçilen alan(clientrect, cliprect, vb.) bu renk ile boyanmış olacaktır. *Brush* nesnesi, taşıdığı renk(color), resim(bitmap) ve desen(style) ile tanımlanmaktadır.



3.6.3.3 PEN

Canvas nesnesi aracılığı ile kullanılabilecek olan *Pen* nesnesi, nesneye atanan renk ile(pen.color) çizgi çizmek için kullanılmaktadır.



3.6.3.4 RECT

Rect tipi, bir dikdörtgenin ölçülerini ifade etmektedir.Koordinatlar, sol, üst, sağ ve alt kenarları ifade eden 4 ayrı sayısal değer olarak ya da sol üst köşe ve sağ alt köşe noktaları olarak ifade edilirler.

3.6.3.5 POINT

Point tipi, ekran üzerindeki bir *pixel'* in yerini ifade eder. X, *Point* tipinin yatay koordinatını