Delphi'ye Giriş

Visual dillerden biri olan Delphi Programlama dili ile istediğiniz programları yapmak artık zor değil.

Programlarda kullanılabilecek standart işlemler birer kontrol olarak tasarlanmış olup programcının kullanımına sunulmuştur. Örneğin; Personel bilgileri girilen bir program yapılırsa, personelin adı, soyadı, doğum yeri ve tarihi gibi bilgilerin girileceği Edit kutusu, kullanıcının neyi girmesi gerektiğini belirten Label, değişik seçeneklerden birini seçme imkanı sağlayan aşağı doğru açılan ComboBox kutusu, komut butonları, personelin bir listesini gösterecek bir Liste kutusu (ListBox). Bunları kolayca forma taşıyarak programızın ekranını tasarlayabiliriz. Tek yapılması gereken Form üzerindeki kontrolleri oluşturmak için Ana formun üzerinde bulunan Componentlerden ilgili ikonları seçmek ve formumuza tıklamak.

Object Inspector Penceresi

Formumuza koyduğumuz kontrol elemanlarının özellikleri ve aldığı değerler Object Inspector Penceresinde görülür. Object Inspector; Properties (Özelliğin Adı), Events (Özelliğin Değeri) 2 kısımdan oluşur. Örneğin Formumuza Label1 koyalım. Label1'i Mouse ile seçelim. F11 tuşuna tıklayarak Object Inspector penceresini açalım. Burada Label1'e ait olan Object Inspector penceresinin açıldığını görürüz. Örneğin Label1 ismini değiştirmek istersek Caption özelliğine gelip karşısındaki kutucuktan Label1 sözcüğünü silip istediğimiz kelimeyi yazacağız (Örneğin Personelin Adı).Bunun gibi Object Inspector'deki değerleri değiştirebiliriz. Örneğin Label1'e ait Object Inspector'deki Properties kısmındaki değeri alClient seçtiğimiz zaman kontrol elemanının (Label1) Formun her tarafını kapladığını görürüz. Yine Font kısmında Label1'in yazı karekterini büyütebilir ve rengini değiştirebiliriz. Yine Örnek verecek olursak Formumuza aşağı açılır kutucuk (ComboBox1) koyalım. Bu kontrol elemanına ait Object Inspector'de Items'in yanındaki kutuya tıklayarak açılan formda ComboBox açılırken görmek istediğimiz kelimeleri yazalım.

Programı Çalıştırma

Programı F9 tuşuna basılarak çalıştırılır. Programı çalıştırmak için kodların yazılmasına gerek yoktur. Tasarım halinde de programımızı çalıştırabiliriz.

Kod Yazma

Kod yazmak istenilen kontrol elemanı seçilir ve bu elemana ait Object Inspector'deki Events özelliğinden yararlanılır. Burada ilgili kontrol elemanına tıklanınca programın bitmesi isteniyorsa OnClick kısmına kod yazılır. Eğer iki kez tıklanınca programın bitmesi isteniyorsa OnDblClick kısmına kod yazılır veya mouse ile ilgili kontrol elemanının üzerinden geçerken mesaj vermek istiyorsak, OnMouseMove kısmına kodumuzu yazacağız.

Örnek: Şimdiye kadar öğrendiğimiz bilgilere göre aşağıdaki programı yapalım. Formumuza; Ana Formdaki Standart Component'inden 5 tane Label, 3 tane Edit Text, 2 tane ComboBox, 1 tane ListBox, 5 tane de Buton yerleştirelim. Bu kontrol elemanlarının OnClick kısımlarına aşağıdaki kodları yazalım.

```
procedure TForm1.Button1.Click(Sender: TObject);

begin //Ekle butonuna yazılacak kod

Listbox1.Items.Add ( Edit1.Text+' ' +Edit2.Text+' '+Edit3.Text+' '+ ComboBox1.Text+'
'Combobox2.Text);
```

```
end;

procedure TForm1.Button2.Click(Sender: TObject);

begin // Sil Düğmesi

ListBox1.Items.Delete(ListBox1.ItemIndex);

end;

procedure TForm1.Button3.Click(Sender: TObject);

begin // Kaydet düğmesi

ListBox1.Items.SaveToFile ('liste.dat'); //Liste adlı dosya oluşturarak bilgiyi kaydeder.

end;

procedure TForm1.Button4.Click(Sender: TObject);

begin // Yükle düğmesi

ListBox1.Items.LoadFromFile('liste.dat'); // Bilgileri Listbox1'e yükler.

end;

procedure TForm1.Button5.Click(Sender: TObject);

begin // Programı kapatma düğmesi

Close;

end;
```

Birden Fazla Form Kullanma

Programa yeni bir form eklemek için File menüsünden NewForm seçeneği kullanılır. Programda birden fazla form varsa bu formlara erişmek için View-Forms menüsü kullanılır. (Diğer formlara Shift + F12 kısayol tuşuylada ulaşabilirsiniz)

Component Palet

Form üzerinde oluşturulacak ekran görüntüsü ise Delphi'nin ana formundaki Component Palet üzerindeki kontrol elemanları tarafından yapılır. Component Palet'ten seçilen kontrol elemanları mouse ile Form üzerine kolayca yerleştirilir. Componentler gruplanarak yerleştirilmiştir. Her grup bir çok farklı componentten oluşmaktadır.

Yeni Component'ler Ekleme

Delphi'ye yeni componentler ve activeX kontrolleri eklenebilmektedir. Yeni bir Component eklemek için Component menüsündeki Import ActiveX Control komutu kullanılır. Açılan pencerede yüklü ActiveX kontrolleri listelenir. Eklenilmek istenilen ActiveX kontrolü listede yok ise Add düğmesi ile OCX dosyası bulunup listeye eklenebilir. Penceredeki Palette Page

kutusundan eklemek istenilen kontrolün hangi kısma ekleneceği seçildikten sonra Install düğmesi ile Delphi'ye alınabilir.

Component Şablonları (Template) oluşturmak

Program tasarımında en çok yapılan işlemler, sık kullanılan bileşen yada bileşen grupları bir şablon olarak kaydedilerek aynen standart bir bileşen gibi tekrar kullanılabilir.

Project Manager Penceresi

Project Manager penceresi normalde ekranda görülmez. View / Project Manager menü seçeneği ile görüntülebilir. Bu pencere içinde programımıza ait Formlar ve Unitler gösterilir. Bu pencere içindeki dosyalar DPR uzantılı bir dosyada tutulur. Projeden bir dosyayı çıkarmak için Remove seçeneğini, yeni bir dosya eklemek için New seçeneği kullanılır

Delphi'de Kullanılan Dosya Tipleri	
.pas	Her unitin kaynak kodlarını içeren dosyalardır. Delphi'de oluşturulan formlara ait kodlar da PAS uzantılı dosyalarda saklanır.
.dfm	Form Dosyalarıdır. Delphi'de oluşturulan formlar ve üzerindeki kontrollere ait bilgiler bu uzantıdaki dosyalara kaydedilir.
.dsk	Projemize ait masa üstü ayarları bu dosyaya kaydedilir. Hangi formun açık olduğu , ekrandaki koordinatları gibi bilgiler bu dosyada tutulur.
.dof	Project / Options diyalog penceresindeki ayarları içeren dosya.
.exe	Çalıştırılabilir program dosyası. Application projeleri çalıştırıldığında onlara ait EXE dosya, projeye verilen isimle otomatik olarak oluşturulmaktadır.
.dcu	Derlenmiş unit dosyaları bu uzantı ile kaydedilir. Bir uygulamayı çalıştırdığımız zaman ilk önce unitler derlenir ve EXE hale getirilir.
.dpr	Proje Kaynak Dosyası. Delphi'deki projeler bu tür dosyalara kaydedilir.
.res	Derlenmiş Binary Kaynak (Resource) Dosyaları. Windows tarafından standartlaştırılan bu formattaki dosyalarda resim, ikon, kursor gibi bilgiler bulunur.

Değişkenler

Değişken Tanımı

- Delphi'de değişkenler kullanılmadan önce tanımlanma mecburiyeti vardır. Değişken adları 63 karakteri geçmemelidir. 63.karakterden sonrası gözardı edilir.
- Değişken adları sembolleri içermemelidir. ($, * , % vb.)
- Bir değişken tanımlama işi Var bloğunda yapılmalıdır.
- Bir değişkenin tanımı iki kısımdan meydana gelir. 1.kısım değişkenin adını, 2. kısım ise değişkenin tipini belirler.

```
Var
a,b : integer ;
i : integer ;
aciklama : string
;
Medenihal :
Boolean ;
kesirli : real ;
```

Veri Tipleri

Tamsayı Tipleri

ShorInt: 1 Byte'lık işaretli tamsayı tipidir. (- 128) ile 127 arasında değer alır.

SmallInt: 2 Byte'lık işaretli tamsayıdır. (-32768) ile 32767 arasında değer alır.

LongInt: 4 Byte'lık işaretli tamsayı tipidir. (- 2147483647) ile 2147483647 arasında değer alabilir.

Integer: LongInt tipi gibidir. 4 Byte'lık işaretli tamsayı tipidir. (- 2147483647) ile 2147483647 arasında değer alabilir.

Byte: 1 Byte'lık işaretsiz bir tamsayı tipidir. 0 ile 255 arasında değer alabilir.

Word: 2 Byte'lık işaretsiz tamsayı tipidir. 0 ile 65535 arasında değer alabilir.

Cardinal: 4 Byte'lık işaretsiz tamsayı tipidir. 0 ile 2147483647 arasında değer alabilir.

Int64: 64 bitlik tamsayılar tanımlanabilmektedir. (-9 x 10^18) ile 9 x 10^18 arasında değer alabilmektedir.

LongWord: 0 ile 4294967295 arası değerler alır.İşaretsiz tamsayıdır.Delphi 4 versiyonuyla birlikte geliştirilmiştir.

Variant: 16 Byte'lik bir değişken tipidir.Bu tipteki bir değişkene hem sayısal hem de string ifadeler aktarılabilir.

Örnek:

```
Var

i : variant ;

a : integer ;

k : int64 ;

begin

i := 'Bahadir';

a := 12345 ;

k :=123456789 ;

end;
```

Reel Sayı Tipleri

Single: 4 Byte'lık ondalık sayı tipidir. 1.5 x 10^(-45) ile 3.4 x 10^38 arasında değer alır.

Real48: 6 Byte'lık ondalık sayı tipidir. 2.9 x 10^(-39) ile 1.7 x 10^38 arasında değer alabilir.

Real ve Double: 8 Byte'lık ondalık sayı tipidir. 5.0 x 10^(-324) ile 1.7 x 10^308 arasında değer alabilir.

Extended: 10 Byte'lık ondalık sayı tipidir. 3.4 x 10^(-4932) ile 1.1 x 10^4932 arasında değer alabilir.

Comp: 8 Byte'lık ondalık sayı tipidir. -2^63 + 1 ile 2^63 - 1 arasında değer alabilir.

Currency: 8 Byte'lık - 922337203685477.5808 ile + 922337203685477.5807 aralığında işaretli bir sayı tipidir.

Boolean Tipi

Boolean: True ve False değerlerinden birini içerir. 1 Byte'lık yer kaplar.

Karakter Tipleri

Char: Bir karakter içeren 1 Byte'lık veri tipidir. Bu değişkenler sadece bir karakter barındırabilirler. Örnek: ' B ', ' 2 '

AnsiChar: Char tipi ile aynı özelliklere sahiptir.

WideChar: 2 Byte'lık bir karakter tipidir.

String Tipleri

ShortString: 255 karaktere kadar karakter ataması yapabilen veri tipidir. Bellekte 1 Byte'lık yer kaplar.

AnsiString: Bu tip stringler Dinamiktir ve belli bir sınırı yoktur. Yani ne kadar karakter atanırsa o kadar bellekte yer kaplar.

String: AnsiString tipiyle aynı özelliklere sahiptir.

PChar: 64 KByte'a kadar atama yapılabilen #0 karakteri ile biten string tipidir.

Delphi'de Kullanılan Operatörler		
Operatör	Açıklama	Örnek
+	Toplama	a:=b+c;
-	Çıkarma	a:=b-c;
*	Çarpma	a:=b*c;
/	Bölme	a:=b/c;
Div	Tamsayı Bölme	a:=b div 10;
:=	Atama	a:=12;
=	Eşittir	if(a=12) then ShowMessages('a eşit 12');
< >	Eşit Değil	if(a< >12) then ShowMessages('a eşit değil 12');
<	Küçük	if(a<12) then ShowMessages('a küçük 12');
>	Büyük	if(a>12) then ShowMessages('a büyük 12');
>=	Büyük eşit	if(a >=12) then ShowMessages('a büyük eşit 23');
<=	Küçük eşit	if(a<=12) then ShowMessages('a küçük eşit 23');
And	Mantıksal AND	if(a>=12) and (a<15) then ShowMessages('a değerleri gerçekleşti);

Or	Mantıksal Or	if (a>12) or (a<15) then ShowMessages('a değerinin İkisi veya biri gerçekleşti.');
Not	Mantıksal Not	if not (a< 12) then ShowMessages('a değeri 12'den büyük değil.');
^	Pointer Operatörü	a:=b*c^;
@	Pointer Adresi	a:=@b;
$	Heksadecimal Operatörü	a:=$FF;
[]	Dizi Operatörü	a:Array [1..99] of integer;
.	Nokta Operatörü	Personelin.Adı:='Bahadir';

Diziler

Dizi tanımı Var bloğunda aşağıdaki şekilde yapılır.

Var

Diziadi:array[altsınır..üstsınır] of tip

İki boyutlu dizi şöyledir :Diziadi:array[altsınır1..üstsınır1,altsınır2..üstsınır2] of tip

Örnek: Var

a:array[1..50]of integer;

Dinamik Dizi Tanımlama

Dinamik Dizi; boyutunun derleme aşamasında belirlenmesi gerekmeyen dizilere denir ve program çalışırken dizinin boyutu belirlenebilir.

var diziadı:array of tip

Herhangi bir anda bu dizinin boyutu SetLength fonksiyonu ile belirlenir.

Setlength(diziadı,boyutu);

Örnek: SetLength(i,12) ; // i dizisini 12 elemanlı yap.

Çok Boyutlu Dizi

Dizi birden fazla boyutluda olabilir. Bu durumda her boyut için tanım kısmında bir array of ifadesi kullanılır.

Var Diziadı : array of array of array of ... tip

Örnek: 3 boyutlu bir a dizisi tanımlanırken;

Var a : array of array of array of Integer ;

Çok boyutlu dizinin boyutu SetLength (a , 2 , 4 ,6) ; gibi tanımlanır.

Sabitler

Sabitler değişmeyen değerler içerirler. Bu değer programın başından sonuna kadar değeri aynıdır. Bir sabit tanıtımı Const bloğunda yapılmalıdır.

Const

Sabitadı=değeri;

Örnek: Const

i = 100;

adi = 'Bahadir';

İlk Değer Atama ve Static Değişken Tanımı

Değişkenlere ilk değerin atanması ve bu değişkenin değerini prosedürün çalışması bittikten sonra da korunması için Const kısmı altında değişken tanımı yapılır.

Const

değiskeninadi : Tipi = İlkdeğeri;

Örnek: Const

i : integer = 0 ;

Bir Diziye İlk Değer Atama

Const

Diziadi : Array[altsınır..üstsınır] of tipi = (değer1 , değer2 ,...) ; şeklindedir.

Örnek: Const

Mevsimler:Array[1..4] of string = (' İlkbahar ' , ' Yaz ' , ' Sonbahar ' , ' Kış ');

Pointerler

Pointerler bir değeri değil, değerin bulunduğu adresi gösterir.

PAnsiString: AnsiString tipinde bir değişkeni gösteren pointer.

PByteArray: TbyteArray tipinde bir değişkeni gösteren pointer. Çoğunlukla bellekte ayrılmış bölgelerdeki her byte ulaşılabilmek için kullanılır.

PCurrency: Currency tipindeki bir değişkeni gösteren pointer.

PExtended: Extended tipindeki bir değişkeni gösteren pointer.

PShortString: ShortString tipindeki bir değişkeni gösteren pointer.

PWordArray: TWordArray tipindeki bir değişkeni gösteren pointer.

Herhangi bir değişken için ^ karakteri kullanılarak kolayca pointer tanımlanabilir.

Örnek: Var

ptr : PAnsiString ;

p: ^integer ; / / integer tipindeki değişkeni gösteren pointer.

Pointerlerin Barındıracağı Adres

Pointerin barındıracağı adresi belirleme işi @ karakteri ile yapılır.

p := @ i ; / / p pointeri i adresini barındıracaktır.

ptr := @ a[1] ; / / ptr pointeri ile dizinin ilk adresini barındıracak.

Pointer Kullanımı

Pointer, tiplerin önüne ^işareti koyularak tanımlanırlar.

Örnek: var

x : ^integer ;

y : ^string ;

Şartlı Çalıştırma Deyimleri

Programlar normal zamanda satır satır çalışırlar. İsteğe bağlı olarak belli şartlar aranarak programın bir kısmının çalıştırılmasını veya çalıştırılmamasını sağlayabiliriz.

İf Döngüsü

if döngüsünün genel yapısı aşağıdaki gibidir.

If şart then

Komut ;

Else

diğer komut ;

Örnek: var

i : integer ;

begin

i := 0;

i := i + 1 ;

```
if i = 1 then begin

Label1.Caption := 'Doğru ';

end

else

if i<>1 then begin

Label1.Caption := 'Yanlış ';

end;

end;
```

Case Döngüsü

Bir değişkenin aldığı bir çok değere göre ayrı komutların çalıştırılması gereken durumlarda Case döngüsü kullanılır. Genel yapısı aşağıdaki gibidir.

```
Case değişken of

durum1 : Komutlar ;

durum2 : Komutlar ;

.....

durumn : komutlar ;

Else komutlar ;

end;
```

Yukarda dikkat edilirse değişken; durumlara uyan değerler aldığı zaman ilgili komutlar çalışacak eğer değişkenin değeri hiçbirine uymuyorsa Else' den sonraki komut çalışacaktır. Aşağıdaki örneği inceleyelim.

```
Case x of

1 : label1.Caption := ' Merhaba ' ;

2 : label1.Caption := ' Dünya ' ;

3 : Edit1.Text := ' Bahadir Sahin ' ;

else

Edit1.Text := ' Hoşçakal ' ;

end;
```

For Döngüsü

For döngüsünün genel yapısı aşağıdaki gibidir.

```
For i := ilkdeğer to sondeğer do

begin

komut ;

end ;
```

Burada to kullanıldığı için i artarak değerler alır.downto kullanıldığı zaman değer i değerleri azalarak gider.

```
For i := ilkdeğer downto sondeğer do

begin

komut ;

end ;
```

Örnek : var

```
i : array [ 1 .. 20 ] of string ;

k : integer ;

begin

for i := 1 to 20 do

i [ k ] := InputBox( ' i değer girişi ' ,Inttostr (k ) + '.nin adı ' , ' ' ) ;

end ;
```

While - Do Döngüsü

Bir şart gerçekleştiği zaman çalışması gereken program bloklarında kullanılır.Genel yapısı aşağıdaki gibidir.

```
While şart do

Begin

Komutlar ;

end ;
```

Örnek :var

```
i : integer ;
```

```
Begin

i := 0 ;

while i < 10 do

Begin

Label1.Caption := inttostr ( i ) ;

i := i + 1 ;

end ;
```

Repeat - Until Döngüsü

Genel yapısı aşağıdaki gibidir. Şart gerçekleşene kadar çalışması gereken kısımlarda kullanılır.

```
Repeat

Komut ;

Until Şart ;
```

Örnek: var

```
i : integer ;

begin

repeat

i := 50 ;

Until (i>0) and ( i < 50 )

end ;
```

Döngü Kontrol İfadeleri

Bazı durumlarda döngü bitmeden döngüden çıkılmak istenebilir. Bu gibi durumlarda döngü kontrol deyimlerini kullanmak gerekir. Bunlardan bazıları Break, Continue gibi deyimlerdir.

Break

Break kontrol deyimi For,While veya Repeat döngülerinden birinde döngüden çıkmak için kullanılır.Aşağıdaki örneği inceleyelim.

Örnek :

```
procedure TForm1.Button1.Click ( Sender : TObject ) ;

var
```

```
i : array [1 .. 20 ] of string ;

k : integer ;

begin

for i := 1 to 20 do

if i [ k ] = 'Bahadir' then begin

break ;

if > 50 then begin

ShowMessages (' Aranan kayıt bulunamadı. Tekrar deneyiniz...' ) ;

end

else

ShowMessages (Inttostr ( k ) + ' . kişi' ) ;

end ;

end ;
```

Continue

Continue; For, While veya Repeat döngülerinde bazı şartlar gerçekleştiğinde döngünün sonuna gitmeden tekrar başa dönmesini sağlar.

Örnek:

```
var

i : integer ;

Begin

for i := 1 to 20 do

if i<10

continue ;

end ;

ShowMessages( Inttostr ( i ) ) ;

end ;
```

Exit

Exit; mevcut program bloğundan, bloğun sonuna ulaşmadan çıkmaya yarar.

```
Örnek : var i : integer ;

begin

i := strtoint ( Edit1. Text ) ;

if i <10 then begin

exit ;

end

else

Edit1.Text := 'i'nin değerleri 10'dan büyük.' ;

end ;

end;

Halt
```

Programdan çıkış sağlar.

```
Örnek: Begin

Form1.Halt ; / / Programdan çıkış sağlar.

end ;
```

Whit - Do

Herhangi bir kontrol elemanının birden fazla özelliğini değiştirmek için yada metodlara ulaşmak için kullanılır. Genel kullanımım şekli aşağıdaki gibidir.

```
With kontroladi do

Begin

....

end ;
```

Borland Delphi'de Veri Tabanı

Veri Tabanı Dosyası, Delphi ile birlikte gelen Borland Database Desktop ile tasarlanabilir. Bunun dışında Visual Dbase, Dbase for Windows, Paradox, Dbase VI, Access gibi veri tabanı programlarıda desteklenir. Şimdi Database Desktop programı ile veritabanı hazırlamaya başlayalım.

Database Desktop ile Veritabanı Hazırlanması

Delphi ile gelen Database Desktop programını çalıştırın. Daha sonra tablo hazırlayacağız. Bunun için File-New-Table menüsüne tıklayınız. Açılan pencere'de dBASE for Windows'u seçin ve Ok tuşuna tıklayın. Veri tabanındaki alanları tanımlayabilmemiz için pencere açılacaktır. Açılan bu penceredeki sütunlar aşağıdaki özellikleri belirtir.

1) Field Name: Bu sütuna alanın ismi yazılır. (Örneğin Adı Soyadı vb.)

2) Type: Bu alana girilecek olan bilginin tipini giriyoruz. Aşağıda bu tipleri belirteceğim.

2-1) Character: Bu alana girilen bilgi harflerden veya rakamlardan veya her ikisinin karışımından olabilir.

2-2) Number: Bu alana girilen bilgi sadece rakamlardan ibarettir.

2-3) Date: Tarih bilgileri bu tipte tanımlanır.

2-4) Logical: Evet - Hayır, Açık - Kapalı gibi sadece iki durumdan oluşan olaylar bu tipte tanımlanır.

2-5) Memo: Uzun metinler bu tipte tanımlanır.

2-6) Binary: Resim, ses gibi özel alanlar bu tipte tanımlanır.

3) Size: Bu sütunada karakter sınırı yazılır. (Örneğin 10 yazıldığında ; 10 karaktere kadar klavyeden giriş yapılır.)

Not: Sütunlara bilgiler girildikten sonra, Save As tuşuna basılarak oluşturduğumuz Tablo kaydedilir.

Wizard Kullanılarak Veritabanı Hazırlanması

Borland Delphi bize bir kolaylık daha sunmuştur. DataBase Form Wizard'ı kullanarak veri tabanınızı hazırlayabilirsiniz. Şimdi adım adım Wizard kullanararak Tablomuzu oluşturalım.

İlk önce Delphi programını çalıştırınız. Bundan sonra File menüsünden New menüsüne tıklayın. Açılan pencerede Business kısmına geçiniz. Burada DataBase Form Wizard'ı seçip Ok tuşuna basınız. Karşımıza Database Form Wizard penceresi çıkar.

Bundan sonra; Eğer tek tablonuz varsa Create a simple Form ' u işaretleyin. Eğer birden fazla tabloyu ilişikli şekilde göstermek istiyorsanız; Create a Master / detail Form düğmesini seçin sonra Next tuşuna basınız. Çıkan formda kullanılacak tabloyu belirleyeceğiz. Eğer bir DBF dosyası varsa onun bulunduğu dizine giderek seçebilir veya Driver or Alias Name kutusunda DataBase Explorer ile düzenlenmiş veri tabanından biri seçilebilir. Biz Örnek olarak Borland Shared / Data dizinindeki animals.dbf'yi seçeceğiz.

animals.dbf seçildikten sonra Next tuşuna basınız. Çıkan pencerede >> tuşuna basarak bütün alanları seçeceğiz. Alanları seçtikten sonra Next tuşuna basın. Çıkan pencerede alanları yatay (horizantal), düşey (Vertically) veya in a Grid şeklinde mi sıralanacağını belirleyip Next tuşuna tıklayınız. En son çıkan pencerede Finish tuşuna basınız. Böylece tablomuzu otomatik olarak oluşturmuş olduk.

Veritabanı Bileşenleri

Data Access: kısmındaki kontroller program çalıştığında ekranda gözükmeyen bileşenlerdir. Bu bileşenler Data Controls kısmındaki kontroller ile görüntülenecek veriler için veritabanları ile köprü vazifesi görürler.

TTable: En önemli özellikleri DataBase Name ve Table Name'dir. Database Name 'e BDE içinde tanımladığımız alias verilebilir. Örneğin: Formumuza Data Access kısmından 1 tane TTable yerleştirelim. Bunun Database kısmına DBDEMOS aliasını seçelim. Daha sonra Table Name kısmındaki ComboBox'tan animals.dbf' yi seçelim.

DataSource: Verilerin data controls kısmındaki bileşenler yardımıyla görüntülenmesi için table, query gibi kontrolleri mutlaka Datasource'e bağlanması gerekmektedir. Az önce formumuza Table yerleştirmiştik. Şimdi formumuza Data Access kısmından Data Source kontrolünü yerleştirelim. Object Inspector penceresinden Dataset özelliğini az önce koyduğumuz Table1'e ilişiklendirelim.

DBEdit: Görsel kontrollerden biridir. Data Controls' dan seçip formumuza yerleştirelim. DbGrid'in önemli kısımlarından DataSource ve DataField'dir. Şimdi DataSource kısmında DataSource1'i seçelim. DataField kısmından da Area 'yı seçelim. Gerekli diğer kontrolleride yerleştirdikten sonra; Table1'i seçerek Object Inspector'den Active kısmını True yapalım; sonra programı F9 tuşuna basarak çalıştıralım. Görüleceği gibi animals.dbf'ye ait bütün bilgiler ekranımızda belirdi.

DbImage: Veritabanlarına resim içeren alanların işlenmesi için kullanılır.

DbGrid: Verilerin gösterilmesi için kullanılır. Gösterilecek alanlar ayarlanabilir, verilerin fontu değiştirilebilir. Az önceki örneğimizdeki formumuza Data Controls kısmından DbGrid yerleştirelim. Daha sonra Object Inspector'de DataSource özelliğini DataSource1 değerini verelim. Böylece programı çalıştırdığımız zaman Grid sütunlarında, alanlar ise satırlarında gösterildiğini göreceğiz.

DbNavigator: Veriler üzerinde güncelleme, silme, yeni kayıt ekleme, ileri - geri gitme vb. gibi işlemlerin yapıldığı araç çubuğudur. Örneğimizde formumuza Data Controls kısmından DbNavigator yerleştirelim ve Object Inspector kısmında Data Source özelliğini Data Source1 yapalım. Programı çalıştırdığınız zaman istediğiniz işlemleri DbNavigator çubuğuyla yapabileceğinizi göreceksiniz. Bu çubuktaki özellikleri inceleyelim.

First: İlk kayda gider.

Prior: Bir önceki kayda gider.

Next: Bir sonraki kayda gider.

Last: En son kayda gider.

Insert: Mevcut kayıttan önce araya bir kayıt ekler.

Delete: Mevcut kaydı siler bir sonraki kaydı görüntüler.

Edit: Mevcut kaydın değiştirilmesini sağlar.

Post: Yapılan değişiklikleri veri tabanı dosyasına yazar.

Cancel: Girilen değişiklikleri iptal eder.

Refresh: Mevcut kaydı yeniden görüntüler. Yani Güncelleme yapar.

Not: DbNavigator'ün ShowHint özelliği True yapılarak, bu düğmelerin ne işe yaradığı üzerine gelindiği zaman yazacaktır.

DBEdit: DBText ile aynıdır, ek olarak veriler üzerinde değişiklik yapılabilir. Bunlarda Data Source özelliği Data Source1 kısmına ilişiklenir ve DataField özelliğindede ilgili kısım seçilir.

DbText: Label ile aynıdır. Bağlı olduğu tablodan belirtilen alan bilgisini görüntüler. Genelde üzerinde değişiklik yapılmayacak alanların gösterilmesinde kullanılır.

DBMemo: Birden fazla satırın veya 255 karakterden daha uzun verilerin saklanması ve gösterilmesi için kullanılır.

DBListBox: Verilen alan değeri eğer liste içinde bulunuyorsa otomatik olarak seçilir. Bileşin listesini biz doldurmak zorundayız.Verilen alan ile ilgili değerler otomatik olarak gelmez.

DBComboBox: TListbox ile aynıdır. Bu kontrol aracılığı ile combobox içinde bulunabilecek değerler belirlenir ve kullanıcının bu değerlerden birisini seçmesi sağlanır. Önce DataSource özelliği ile kullanılacak tablo ve DataField özelliği ile de comboBox'un bağlantı kuracağı alan belirlenir. DBComboBox içinde bulunacak değerler ise bu kontrolün Items özelliği ile açılan pencereden belirlenir.

DBRadioGroup: Bir alana girilecek bilgi sayısı sınırlı ise bu kontrol kullanılır. DataSource özelliği ile kullanılacak tablo ve DataField özelliği ile de bağlantı kurulacak alan belirlenir ve Caption ile alanın ismi değiştirilebilir.

DBChart: DBChart kontrolü kullanılarak veri tabanımızda bulunan bilgileri grafiksel olarak ifade edebiliriz. Bu kontrol ile birden fazla grafiği bir arada görebiliriz. Farklı veri tabanlarında bulunan bilgileri bile aynı grafik üzerinde gösterebilmekte ve böylece farklı verilerin analizi grafiksel olarak gerçekleştirilebilmektedir.

TQuery Kontrolü İle Veri Sorgulama

Delphi'de Veri Sorgulama işlemi TQuery kontrolü ile olur. TQuery kontrolüne ait SQL özelliği; hem bir editör olarak ve hem de doğrudan SQL komutları yazılıp çalıştırılmak suretiyle kullanılan bir özelliktir. Şimdi Query1 kontrolüyle sorgulama işlemlerine başlayalım.

Tüm Sütunları Listelemek

Aşağıdaki örneği inceleyelim.

Örnek: liste.dbf adlı veri tabanımız olsun. Bunu Query1 kontrolü ile sorgulayıp listelemek istiyoruz. Bunun için yapmamız gerekenler şunlardır. İlk önce formumuza Data Access kısmından Query1 bileşeni ile DataSource1 bileşeni yerleştirelim. DataSource1'in Dataset özelliğini Query1 yapalım. Daha sonra Data Controls kısmından formumuza DbGrid yerleştirelim ve Object Inspector'den DataSource özelliğini DataSource1 yapalım. Daha sonra Query1'in DataBaseName özelliğini alias olarak tanımlanan önceden oluşturduğumuz liste.dbf'yi seçelim. Sonra Query1'in SQL özelliğine gidip çıkan pencerede aşağıdaki SQL ifadesini yazalım.

```
SELECT * FROM liste
```

Sonra Query1'in Object Inspector'den Active özelliğini True yapalım ve programımızı F9'a basarak çalıştıralım. DbGrid'de görüleceği gibi liste.dbf'deki bütün bilgiler listenmiştir.

İstenilen Sütunları Listelemek

istenilen sütunlar listelenmek istenirse Query1'in SQL özelliğine şu SQL komutu yazılmalıdır.

SELECT adi , soyadi , adresi FROM liste

Not: liste.dbf adlı veri tabanımızdan sadece adi, soyadi, adresi adlı sütunlar ve bu sütunlardaki bilgileri listelemiş olduk.

Tekrarlı Kayıtları Bir Defa Listelemek

Bir tabloda bulunan aynı kayıtları bir kez listelemek için DISTINCT komutu kullanılır. Bunun için; Query1'in SQL özelliğine şu SQL komutu yazılmalıdır.

SELECT DISTINCT adi, soyadi, adresi FROM liste

Şartlı Sorgulama yapmak istersek aşağıdaki gibi SQL komutu kullanmak gerekir.

SELECT adi, soyadi, adresi FROM liste WHERE adi = 'Bahadir '

Örnek1: İsminin başharfi "B" ile başlayanları sorgulayalım. Aşağıdaki SQL komutunu inceleyiniz.

SELECT DISTINCT adi, soyadi, adresi FROM liste WHERE adi LIKE ',B%'

Örnek2: İki şartı aynı anda sağlama. Örneğimizi inceleyelim.

SELECT DISTINCT adi, soyadi, adresi, maas FROM liste WHERE (adi = 'B%' and maas > 10000)

İki şarttan biri gerçekleşmesi isteniyorsa yukardaki örnekte And yerine Or kullanılacak.

Verileri Sıralamak

Tabloda bulunan verileri A-Z'ye veya Z-A'ya sıralayabiliriz. Bunun için aşağıdaki örnekleri inceleyelim.

Örnek1: SELECT DISTINCT adi, soyadi, adresi, maas FROM liste WHERE Order By adi ASC (A-Z'ya sıralar)

Örnek2: SELECT DISTINCT adi, soyadi, adresi, maas FROM liste WHERE Order By adi DESC (Z-A'ya sıralar)

Verileri Gruplandırmak

Bir tabloda yer alan kişilerin farklı zamanlardaki yaptıkları işlerin miktarı gruplandırmak suretiyle tek bir tabloda listelenebilir. Bunun için Group BY komutunu kullanacağız.

SELECT adi, soyadi, SUM (toplam_fiyat)Genel_toplam FROM liste Where fiyat = 'TL' GROUP BY adi

Önemli Not: Şimdiye kadar Query1'in SQL özelliğine SQL komutları yazarak sorgulamalar yaptık. Şimdi kod penceresinde yazacağımız program koduyla veri sorgulayacağız.

Örnek: Parametreye bağlı olarak veri sorgulaması

```
procedure TForm1.Button1Click ( Sender : TObject ) ; // Button1 adlı butona kodumuzu yazıyoruz.

begin

Query1.Close ;

Query1.ParamByNAme( ' Adi ' ).AsString := Edit1.Text ;

Query1.Open ;

end ;
```

Sütun Sorgulama

Bunu yaparken DataSource1 'in DataSet özelliğini Query1 ; Query1'in DataBaseName özelliğini veri tabanınız (bizim örnekte liste.dbf) ; DBGrid1 ' in DataSource özelliğinide DataSource1 yapınız.

Örnek :

```
procedure TForm1.Button1Click ( Sender : TObject ) ;

begin

Query1.SQL.Clear ;

Query1.SQL.Add(' Select adi , soyadi , adresi , maas From liste ' ) ;

Query1.Open ;

end ;
```

Programımızı çalıştırıp Button1'e tıkladığımız zaman DBGrid'de istediğimiz bilgilerin listelendiğini görürüz.

Ekleme Sorgusu

Formumuza Button1 koyalım. Caption özelliğine Ekle yazalım. Ekle butonuna aşağıdaki kodları yazalım. (formumuza Query1, DataSource1 ve DBGrid1 yerleştirilmiş varsayıyorum.)

```
procedure TForm1.EkleClick ( Sender : TObject ) ;

begin

Query1.SQL.Add ('INSERT into liste ' );

Query1.SQL.Add ( ' (ADI , SOYADI ) ' ) ;

Query1.SQL.Add ( 'values (" Bahadir " , " Sahin " ) ' ) ;

Query1.ExecSQL ;
```

```
end;
```

Not : Programı çalıştırıp Ekle butonuna bastığımız zaman, kodda yazmış olduğumuz Bahadir Sahin' i veri tabanına ekler.

Veri Güncellemek

Tablomuzda bulunan verilerimizi güncellemek için UPDATE'i kullanacağız. Bunun için Formumuza Button1 koyalım. Caption özelliğine Güncelle yazalım. Güncelle butonuna aşağıdaki kodları yazalım.

```
procedure TForm1.GuncelleClick ( Sender : TObject ) ;

begin

Query1.SQL.Clear ;

Query1.SQL.Add ( ' UPDATE liste set maas = 10000 where ADI = ' Bahadir ' ) ;

Query1.ExecSQL ;

end ;
```

Veri Silme Sorgusu

```
procedure TForm1.SilClick ( Sender : TObject ) ;

begin

Query1.SQL.Clear ;

Query1.SQL.Add ( ' Delete from liste where ADI = ' Bahadir ' ) ;

Query1.ExecSQL ;

end ;
```

Parametreye Bağlı Olarak Veri Sorgulama

```
procedure TForm1.SorguClick ( Sender : TObject ) ;

begin

Query1.SQL.Clear ;

Query1.SQL.Add('Select ADI, SOYADI from liste Where ADI =:a') ;

Query1.ParamByName ('a').AsString := Edit1.Text;

Query1.Open;

end;
```

Borland Delphi'de QReport Bileşenleri ile Rapor Hazırlama

Daha önceki bölümlerde veri tabanını nasıl oluşturduğumuzu ve bu veri tabanına yazmış olduğumuz verileri nasıl sorgulayıp listelediğimizi görmüştük. Şimdi ise yapmış olduğumuz işlerin meyvesini almaya geldi. Nasıl mı? Tabiki QReport ile...QReport ile Printer'dan çıktı alabiliriz. Şimdi QReport'un özelliklerini inceleyelim.

- Bölüm (Band) yapısına dayanan rapor üretici
- QReport bileşenleri ile görsel rapor tasarımı
- Rapor çıktısı verme
- Anında önizleme imkanı
- Limitsiz Memo alanları
- Limitsiz Grup sayısı
- Karmaşık rapor tasarımı
- İsteğe uyarlanabilir rapor tasarımı
- Yazdırılabilir grafik formatları
- Geliştirilmiş hesap ifadeleri
- Tam yazıcı kontrolü vb.diğer özellikler...

Şimdi adım adım Rapor oluşturmaya başlayalım. Bir rapor formu en basit aşağıdaki bileşenlerden meydana gelir.

DataSet (TTable) elemanı

QuickReport (TTable ile bağlantılı)

QuickReport üzerine alınan QRBand bileşeni

TQuickRep elemanının Bands özelliği altındaki HasDetail özelliğinin True yapılması

Detail bölümü üzerine yazılabilir TQRDBText elemanı

Bunları öğrendikten sonra raporumuzu hazırlamaya devam edelim. İlk önce Formumuza Data Access kısmından TTable kontrolünü yerleştirelim. Table1'in Object Inspector'ündeki DatabaseName özelliğini oluşturduğumuz aliası (Ben liste.dbf oluşturdum. Sizde oluşturduğunuz veri tabanını seçin.) seçelim. TableName 'de veritabanının ismini seçelim. Table1'in Active özelliğini True yapalım. Sonra QReport kısmından QuickRep1 elemanın yerleştirelim ve DataSet özelliğini Table1 yapalım. Daha sonra Bands özelliğine tıklayıp HasDetail özelliğini True yapalım. Bu işlem ile form üzerine veri alanlarının yer alacağı bölümü eklemiş olduk. Şimdi bir örnek yapalım.

Örnek: Yukarıdaki işlemleri gerçekleştirdikten sonra QReport kısmından 5 adet QRDBText kontrolü alıp Details kısmına yerleştirelim ve herbir QRDBText'in Dataset özelliğini Table1 ve DataField özelliğinide liste.dbf'de bulunan sütun alanlarından birini seçelim (Mesela ADI, SOYADI, ADRESI). Basit bir rapor yapmış olduk. Şimdi tasarım zamanındaki raporu görebilmek için QuickRep1'in üzerindeyken, Mouse'ın sağ tuşuna tıklayalım. Karşımıza bir menü çıkar. Şimdi bu menüdeki özellikleri inceleyelim.

Report Setting: Tasarım zamanı rapor ile ilgili ayarların yer aldığı pencereye ulaşılır.

Zoom in: Tasarım zamanında rapor üzerindeki bileşenler daha büyük gösterilir.

Zoom Out: Tasarım zamanında rapor üzerindeki bileşenler daha küçük gösterilir.

Rotate Band View: Rapor üzerinde yer alan bileşenlerin yerleri değiştirilir.

Hide Bands: Bu özellikle bileşenler gizlenir.

Reset: Gizlenen bölümleri tekrar gösterir.

Preview: Bu özellik; Raporumuzun ekran görüntüsünü çıkarır.

Raporla İlgili Ayarlar

Tasarım zamanında raporla ilgili ayarların yapılabilmesi için QuickRep1'e sağ tıklanır. Açılan menüde Report Setting seçeneği seçilir. Açılan menüde ;

Paper Size: Kağıdın tipi belirlenir.

Length: Sayfa uzunluğu belirlenir.

Width: Sayfa genişliği belirlenir.

Margins: Kenarlarda bırakılacak boşluk miktarı belirlenir.

Column Space: Sütun genişliği belirlenir.

Number of Column: Sütun sayısı belirlenir.

Other: Bu kısımda raporda kullanılacak font adı, büyüklüğü ve ölçü birimi ayarlanır.

Page Frame: Bu bölümde ise rapor önizleme modunda iken gözükecek çerçeveler, çerçevenin rengi, kalınlığı ayarlanır.

Bands: Rapora eklenecek bölümler (Band) seçilir.

Page Header: Rapora Sayfa Başlığı verilir.

Title: Başlık girilir.

Column Header: Sütun başlıkları girilir.

Detail Band: Veri alanlarının yer aldığı bölümdür.

Page Footer: Sayfanın altında yer alan sayfa alt bilgilerini ekler.

Summary: Raporla ilgili ek açıklamaları ekler.

Çalışma Anında Rapor Önizleme

Bunun için şu işlemleri yapınız. İlk önce Formumuza bir Button ekleyin ve bu button1'in Caption özelliğine Raporla yazınız. Daha sonra butonun OnClick kısmına aşağıdaki kodu yazınız.

```
Procedure TForm1.RaporlaClick ( Sender : TObject ) ;

begin

QuickRep1.Preview ;
```

end;

Programı çalıştırıp Raporla butonuna bastığımız zaman, raporumuzun ekran görüntüsü karşımıza gelir.

Not: Kayıtları direk yazıcıya göndermek istiyorsak, Butonun OnClick olayına şu kodu yazacağız.

QuickRep1.Print ;

Wizard Kullanarak Rapor Hazırlama

- File / New Aplication 'u seçin.
- File / New menüsünden Business kısmına geçiniz. Buradan QuickReport Wizard seçeneğini seçiniz. Sonra Ok düğmesine basınız.
- Çıkan pencerede Start Wizard düğmesiyle bir sonraki adıma geçelim.
- Alias or Directory kutusunda bağlantı kuracağımız veritabanının bulunduğu dizini seçiyoruz. Table Name kısmınada veri tabanımızın ismini seçiyoruz.
- Tablo seçildikten sonra; tablomuzda yer alan sütunlar görülecektir. Raporda bulunmasını istediğimiz alanları belirleyip > veya hepsini seçersek >> tuşuyla sağ taraftaki listeye alalım ve bu işlemi bitirdikten sonra Finish düğmesine basalım. Böylece Wizard kullanarak otomatik olarak raporumuzu oluşturmuş bulunuyoruz.

Veri Süzmek ve Sıralamak

QuickReport; verileri sıralayacak herhangi bir özelliğe sahip değildir. Veriler girildikleri sırada basılırlar. Veri tabanı içindeki veri alanlarının index olarak tanımlanması gerekir. Bu işlemden sonra TTable elemanının IndexName özelliği ile belirtilmesi gerekir. TQuery elemanıylada istenilen şekilde verileri süzmek mümkündür. Veri süzme birkaç şekilde yapılır. TTable.Filter özelliği bir süzme koşulu ekler. Bu sırada TTable.Filtered özelliği True olması gerekir. Diğer yol SQL kullanılarak istenilen kayıtları elde etmektir. Bir diğer seçenek ise Detail bandının PrintEvent olayını kullanmaktır.

Sayfa Büyüklüğü ve Kenar Boşlukların Ayarlama

Bunun için QuickRep1 elemanının Page özelliğine tıklanarak açılan alt özellikler aracılığıyla rapor tasarımında kullanılacak olan sayfanın büyüklüğü ve kenar boşlukları kolay bir şekilde ayarlanır.

Not: QuickRep1'in içinde varsayılan yazı tipleri seçilebilir. Bunun için Font özelliğine tıklanması gerekir. Burada font adı, tipi, rengi fontun altı veya üstü çizili olma durumu ayarlanabilir.

Rapor Başlığı

Rapor tanımlama QuickReport'un Description özelliği ile yapılır. Bu seçeneği tıklayalım ve açılan pencereye raporla ilgili bilgileri girelim. Bir rapor başlığı TQRSysData elemanı aracılığıyla yazdırılabilir. Başlık aşağıdaki kodla oluşturulabilir.

Örnek :procedure TForm1.Button1Click(Sender :TObject) ;

begin

with QuickRep1 do

```
begin

Report Title := ' Personel Bilgisi Raporu ' ;

Bands.HasTitle := True ;

with TQRSysData ( Bands.TitleBand.AddPrintable ( TQRSysData ) ) do

begin

Data := qrsReportTitle ;

AlignToBand := true ;

Aligment := taCenter ;

end ;

Preview ;

end ;

end ;
```

Bir Rapora Bölüm Eklemek

Bir rapora bölüm eklemek için Object Inspector penceresinden TQuickRep elemanına ait Bands özelliğine tıklamak ve çıkan alt bölümdeki özelliklerin değerini true yapmaktır.

Quick Report'ta Kullanılan Operatörler

Operatör	Açıklama
+	Toplama
-	Çıkarma
*	Çarpma
/	Bölme
( )	Parantez
And	Mantıksal And
Or	Mantıksal Or
Not	Mantıksal Not
=	Eşit
<	Küçük
>	Büyük
<=	Küçük eşit
>=	Büyük eşit
< >	Eşit değil

QuickReport Fonksiyonları

Date: Mevcut tarihi string olarak geri gönderir. Rapor üzerine QRExpr elemanı alınarak Expression özelliğine Date verilerek kullanılır.

Time: Mevcut saati stringe çevirir. Rapor üzerine QRExpr elemanı alınarak Expression özelliğine Time verilerek kullanılır.

Str (sayı): Sayısal bir ifadeyi stringe çevirir.

Copy(str,s,l): Bir ifadenin içinde bulunan belli bir bölümü geri gönderir.

Rapora Sayfa Numarası ve Tarih Eklemek

Raporumuza sayfa numarası ve tarih eklemek için; rapor üzerine QRB elemanı alarak BandType özelliğini rbPageFooter yapalım. Daha sonra QRSYSData kontrolünden iki tane alıp birincisinin Data özelliğini qrsDate ve Text özelliğine "Basım Tarihi ", ikincisinin Data özelliğini qrsPageNumber ve Text özelliğini de " Sayfa : " yapalım. Raporu Preview'e tıklayarak önizlemeye alırsak sayfanın sonunda Basım Tarihi ve Sayfa Numarasının yazılmış olduğunu görürüz.

Delphi ile MS Access Database'ine Bağlantı

Delphi'den MS Access Databaseine ulaşabilmek için sırasıyla yapılması gerekenler aşağıdaki gibidir:

1. Windows Control Panel'i açın.

2. ODBC simgesini çift tıklayın.

3. User DSN tabına gelin ve Add butonuna basın.

4. Driver olarak "Microsoft Access Driver (*.mdb)" yi seçin ve Finish butonuna basın.

5. Data Source Name'i olarak kullanmak istediğiniz bir isim verin. Description kısmına isterseniz database'inizle ilgili bir açıklama yazabilirsiniz.

6. Daha sonra Select butonuna basarak database isminizi verin.

7. Advanced butonuna basarak kullanıcı adı ve şifresi tanımlayabilirsiniz. (Zorunlu değil)

8. OK butonuna basıp ODBC ekranını kapatın.

9. Şimdi programınıza bir TDatabase objesi ekleyin.

10. AliasName olarak ODBC ayarlarında verdiğiniz, (5) Data Source Name'i seçin.

11. Database Name olarak istediğiniz bir isim yazın.

12. Eğer 7. basamakta bir kullanıcı adı ve şifresi girmediyseniz login ekranın çıkmaması için LoginPrompt seçeneğini false yapın.

Not: Eğer database'in bir kullanıcı adı ve şifresi varsa LoginPrompt seçeneğini False yapmanız durumunda database'e ulaşamazsınız.

13. Programınıza bit TTable ekleyin.

14. Database Name kısmına TDatabase bileşeninde, Database Name kısmına 11. adımda verdiğiniz ismi seçin.

15. Programınıza bit TDataSource ekleyin.

16. DataSet değeri olarak 13. adımda eklediğiniz TTable adını verin.

Artık MS Access Database'ine bağlı bir table'a sahipsiniz.

Borland Delphi'de Yapılmış Bazı Yararlı Programlar

Database'de Türkçe Problemi

BDE administor programında configuration/dirvers/native/[kullandığın dosya türü] seçeneklerinde langdriver kısmını türkçe bir sürücü olarak değiştir. Bunu yaptıktan sonra delphi ile türkçe karakterleri görebilirsiniz

Bu ayarı yaptıktan sonra Database Desktopta Türkçe karakterler yine görünmeyecek.

Enter Tuşu Kullanımı

Windows programlarında bir alttaki alana geçmek için TAB tuşu kullanılır. Ancak DOS programlarından gelen alışkanlıkla kullanıcılar hep Enter ile alt alana geçmek ister ve bu bir tik olmuştur.

Delphide Enter tuşu ile bir alt alana geçmek için bir yöntem;

- Formun Keypreview olayını True yapılır.
- Form üzerinde herhangiki tüm bileşenlere Default false yaplır.
- formun onKeypres olayına aşağıdaki function ilave edilir.

```
procedure TAdresformu.FormKeyPress(Sender: TObject; var Key: Char);

begin

if Key = #13 then begin

Key := #0;

if (Sender is TDBGrid) then

TDBGrid(Sender).Perform(WM_KeyDown,VK_Tab,0)

else

Perform(Wm_NextDlgCtl,0,0);

end;
```

Açık Windows Uygulamalarının Gösterilmesi

Burada EnumWindows API fonksiyonuyla bir window handle ve pointer parametreleri çağrılarak gizli ve görünür uygulamalar listelenir.

```
//AnaForm üzerine bir ListBox1 ve Button1 ekleyin

//implementation {$R *.DFM} altyna a?a?ydaki fonksiyonu yaz

function EnumWindowsProc(Wnd : HWnd;Form : TForm1) : Boolean;

Export; {$ifdef Win32} StdCall; {$endif}

var

Buffer : Array[0..99] of char;

begin

GetWindowText(Wnd,Buffer,100);

if StrLen(Buffer)

<> 0 then

Form1.ListBox1.Items.Add(StrPas(Buffer));

Result := True;

end;

procedure TForm1.Button1Click(Sender: TObject);

begin

//Tüm gizli ve görünür uygulamalary listele

EnumWindows(@EnumWindowsProc,LongInt(Self));

end;
```

Delphide Yazılan Kodla Exe Çalıştırma

Aşağıdaki örnek kod verilen web sitesini Internet Explorer' ı çalıştırarak açar.

```
procedure TForm1.Button1Click(Sender: TObject);

begin

winexec('C:\Program Files\Internet Explorer\iexplore.exe
www.programlama.com',SW_MAXIMIZE);

end;
```

Bir Harddiskin Seri Numarasının Bulunması

Hard diskin seri numarasını bulur.

```
procedure TForm1.Button1Click(Sender: TObject);
```

```
var

VolumeSerialNumber : DWORD;

MaximumComponentLength : DWORD;

FileSystemFlags : DWORD;

SerialNumber : string;

begin

GetVolumeInformation('C:\', nil, 0, @VolumeSerialNumber,

MaximumComponentLength, FileSystemFlags, nil, 0);

SerialNumber := IntToHex(HiWord(VolumeSerialNumber), 4) + '-' + IntToHex(LoWord(VolumeSerialNumber), 4);

Memo1.Lines.Add(SerialNumber);

end;
```

Windows'un Belgeler Menüsünü Temizleme

Formun uses kısmına ShlOBJ şeklinde ilave yapın; bir butona veya herhangi bir olayın event' ine de ;

SHAddToRecentDocs(SHARD_PATH, nil); şeklinde yazın.

Ekran Görüntüsü Aktarma

Belirttiğiniz sınırlar dahilinde ekranın belli bir alanını formunuzun üzerine koymak istersenız. Formunuza image1 adlı bir resim objesi ekleyin ve daha sonra formunuzun create olayına şu kodu yazın.

```
procedure TForm1.FormCreate(Sender: TObject);

var

DCDesk: HDC;

begin

DCDesk:=GetWindowDC(GetDesktopWindow);

BitBlt(Image1.Canvas.Handle, 0, 0, Screen.Width, Screen.Height,DCDesk, 0, 0,SRCCOPY);

ReleaseDC(GetDesktopWindow, DCDesk);

end;
```

Mouse Pointer1 Gizlemek

Mouse Pointerı gizlemek için ShowCursor(False) komutunu kullanmak yeterli. Tabi yeniden görünmesini sağlamak için ShowCursor(True) komutunu kullanıyoruz.

İmlecin Ekrandaki Yerinin Belirlenmesi

İmlecin o anda ekranın neresinde olduğunu bulan ufak bir kod parçası.

```
procedure TForm1.Button1Click(Sender: TObject);

var Yer:TPoint;

begin

if Assigned(ActiveControl) then

begin

Yer:=Point(0,0); { burda 0,0 imleç'in ekrandaki yeri oluyor }

ActiveControl.ClientToScreen(Yer);

SetCursorPos(Yer.X,Yer.Y);

end;

end;
```

Başlat Menüsü Programlarının Tespit Edilmesi

Başlat menüsünde hangi programların bulunduğunu tespit eden bir kod parçası. Forma bir ListBox, bir Buton, birde DDEClientConv nesnesi ekleyip, Service ve Topic özelligini "Progman" olarak giriniz.

```
var

B:Pchar;

procedure TForm1.Button1Click(Sender: TObject);

begin

ListBox1.Items.clear;

B := DDEClientConv1.RequestData('Groups');

ListBox1.Items.SetText(B);

StrDispose(B);

end;
```

Register Kullanım Örneği

Windows açılırken programınızın otomatik olarak başlatılıp başlatılmamasını nasıl ayarlarsınız. İşte size güzel bir örnek kod. Formunuza CheckBox Ekleyin ve adını Autorun yapın. (veya ne isterseniz.) ve formun Close Olayına Aşağıdaki kodları yazın.

```
procedure Tfilesetup.FormClose(Sender: TObject; var Action: TCloseAction);

var

AppExe :string;

begin

if autorun.Checked=true then begin

with TRegistry.Create do

try

RootKey := HKEY_CURRENT_USER;

if OpenKey ('\SOFTWARE\Microsoft\Windows\CurrentVersion\Run', true) then

AppExe:=#34+Application.Exename+#34;

WriteString('Proton', AppExe);

finally

end;

end;

//DeleteValue('Proton'); İse Programınız Registry den Siler

end;
```

Ekran Çözünürlüğünün Değiştirilmesi

Programınızdan ekran çözünürlüğünü değiştirmek isterseniz yararlı bir kod.

```
{Ekran Çözünürlüğü Örneği-PC'nizin Mevcut Ekran Ayarları}

// AnaForm üzerine ListBox1 ve Button1 bileşenlerini yerleştirin

{Ana formun OnCreate olayı}

procedure TForm1.FormCreate(Sender: TObject);

var

i : Integer;

DevMode : TDevMode;
```

```
begin

Button1.Enabled:=False;

i := 0;

while EnumDisplaySettings(nil,i,Devmode) do begin

with Devmode do

ListBox1.Items.Add(Format('%dx%d %d Colors',[dmPelsWidth,dmPelsHeight,1 shl dmBitsperPel]));

Inc(i);

end;

end;

{ListBox1'in Onclick olayı}

procedure TForm1.ListBox1Click(Sender: TObject);

// Listede istenilen çözünürlük değeri seçildiğinde Button1'de kullanır hale getir

begin

Button1.Enabled := Listbox1.ItemIndex >= 0;

end;

procedure TForm1.Button1Click(Sender: TObject);

// Seçilen çözünürlük değerini değerini uygula

var

DevMode : TDevMode;

begin

EnumDisplaySettings(nil,Listbox1.ItemIndex,Devmode);

ChangeDisplaySettings(DevMode,0);

end;
```

Ekran çözünürlüğünün ekran tarafından desteklenmeyen değerlere ayarlanması ekranınızın bozulmasına neden olabilir.

Mouse'un Sağ Tuşuna Kullanmak

Mouse'un sağ tuşunun kullanımını anlatan küçük bir kod.

```
//bu örnekte form üzerinde mouse'un sağ tuşuna basılınca merhaba yazan bir mesaj çıkar

procedure TForm1.FormMouseDown(Sender: TObject; Button: TMouseButton; Shift: TShiftState; X, Y: Integer);

begin

if ssright in shift then begin

MessageDlg('Merhaba', mtInformation, [mbOk], 0);

end;

end;
```

Varsayılan Browser ile Web Adresi Açmak

Programınızdan herhangi bir internet adresini açmak ister misiniz?

Aşağıdaki kodu girmeden önce programın uses komut satırına "Shellapi" tanımını girmelisiniz.

```
ShellExecute(0, nil, 'http://www.programlama.com', nil, nil, SW_SHOWDEFAULT);
```

Sayıları Formatlı Yazdırmak

Eğer bir ticari program yazıyorsanız, veritabanında bulunmayan bir alanda para ya da benzeri cinsten bir büyüklük göstermek istiyorsanız, ve müşteriniz sizden rahat okuma talep etmişse aşağıdaki kod işinizi görecektir.

```
procedure TForm1.Button1Click(Sender: TObject);

var

a,i:integer;

s:string;

begin

s:=Edit1.Text;

i:=length(s);

if pos(',',s) <>0 then exit;

for a:=1 to i-1 do if (a mod 3)=0 then insert(',',s,i-a+1);

Edit1.text:=s;

end;
```

Final Diyaloğunu Açıp Kapatmak

Find diyalog penceresini açıp kapatan bir kod.

```
H:=FindWindow(PChar('#32770'),Nil); // #32770 Find diyalog'un sınıf adıdır.

If H = 0 Then // eğer Find diyalog'u açık değilse onu aç...

Begin

With dd1 Do

Begin

ConnectMode:=ddeManual;

ServiceApplication:='explorer.exe';

SetLink('Folders','AppProperties');

OpenLink;

ExecuteMacro('[FindFolder(,C:\Dene)]',False);

CloseLink;

End;

H:=FindWindow(PChar('#32770'),Nil);

End;

ShowMessage('Find File Dialogunu gizle...!!');

ShowWindow(H,SW_HIDE);

ShowMessage('Find File Dialogunu göster...!!');

ShowWindow(H,SW_SHOW);
```

Ana Formunuzu Gizleyin

Programınız çalıştığında Ana Formunuzun görünmesini istemiyor musunuz? Çok kolay aşağıdaki kodu projenize ekleyin.

```
program Project1;

uses

Forms,

Unit1 in 'Unit1.pas' {Form1};

{$R *.RES}

begin
```

```
Application.Initialize;

Application.ShowMainForm := False;

Application.CreateForm(TForm1, Form1);

Application.Run;

end;
```