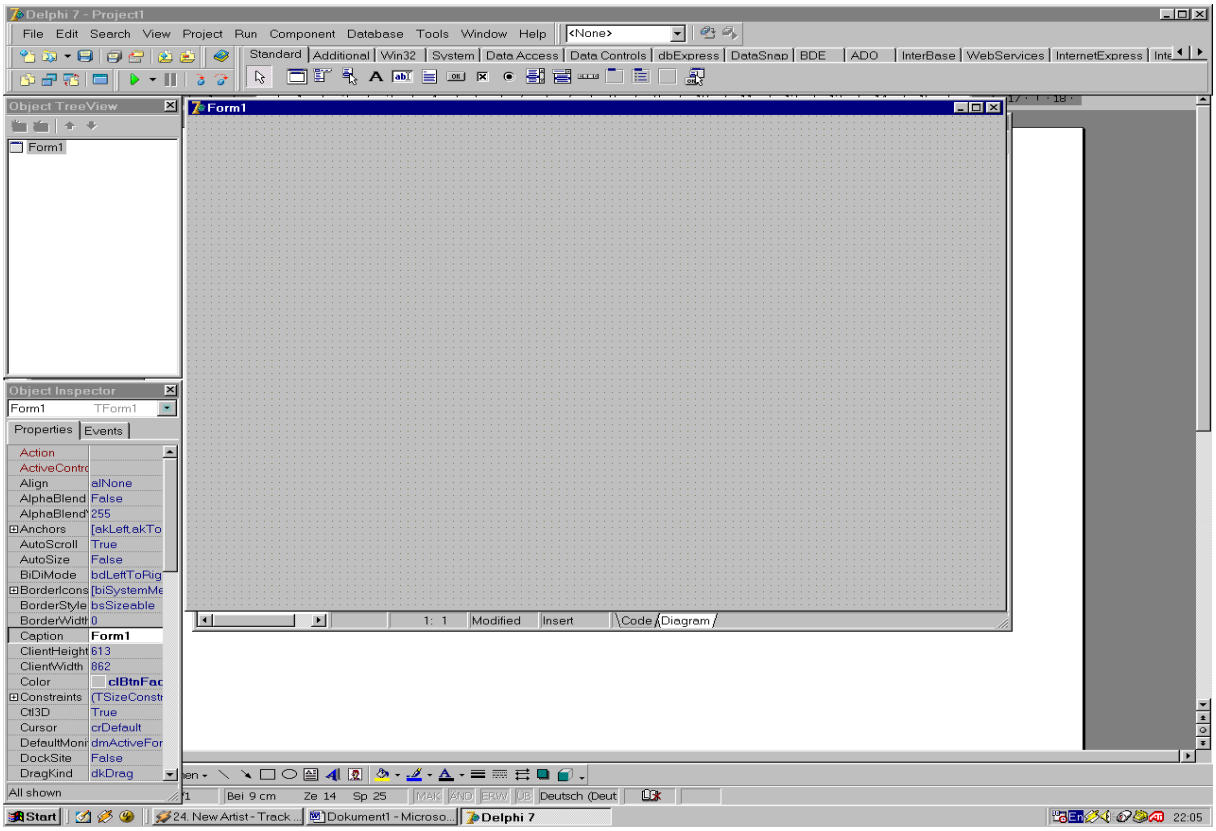
1. Svaka forma je prozor

Kao što kaže naslov, svaka forma jeste prozor, ali obrnuto ne važi. Neke Delfijeve komponente su takođe prozori: dugmad, list boxovi...

1.1 Napravite svoju prvu formu

Kada pokrenete Delfi, on će automatski za vas kreirati novu formu. Ukoliko već imate otvoren projekat, izaberite File/New/Application čime ćete zatvoriti stari projekat (bićete upitani da sačuvate neke od fajlova) i otvoriti novi, prazan projekat. Sada već imate aplikaciju koja može da se izvršava. Možete je pokrenuti Run dugmetom saa toolbara, ili Run/Run komandom menija, čime ćete dobiti standardni Windows program. Ova aplikacija ne radi ništa, pošto ima samo prazan prozor sa nikakvim mogućnostima, sem standardnog ponašanja svakog Windows prozora.



1.1.1. Dodavanje naslova

Naslov forme je Form1. Sa korisnikove tačke gledišta naslov forme je ime aplikacije. Promenimo Form1 u neko drugo ime. Kada pokrenete Delfi, prozor Object Inspector treba da se pojavi sa leve strane forme (ukoliko nije tamo, otvorite ga birajući View/Object Inspector ili pritiskom na F11).

Object Inspector prikazuje osobine selektovane komponente. Prozor ima tab sa dve strane. Prva strana je označena kao Properties – osobine. Druga je Events – događaji i prikazuje listu događaja koji se mogu desiti u formi ili selektovanoj komponenti.

Osobine su navedene po abecednom redu. Formi možemo dati novo ime promenom Caption osobine, koja je inicijalno selektovana selektovana. Primetićete da dok ukucavate novo ime forme Hello, menja se i naslov forme, što se događa sa sveka nekoliko osobina. U opštem slučaju se vrednost menja tek kada pritisnete Enter tipku.. Možete promeniti i interno ime forme koje se nalazi u Name osobini. Ukoliko niste uneli novi naslov forme, vrednost Name osobne biće korišćena i za Caption osobinu.

Bez mnogo rada za sada, imamo aplikaciju sa sistemskim menijem i osnovnim Minimize, Maximize i Close dugmetom. Formi možete promeniti veličinu tako što ćete povlačiti ivice forme, možete je pomeriti povlačenjem naslova, maksimizariti, minimizirati i zatvoriti. Ukoliko pogledate ikonu na task baru, primetićete da ona

prikazuje ime projekta Project1, umesto naslova forme. Davanjem imena projektu pri snimanju, promenićete i tekst na ikoni.

1.1.2. Snimanje forme

Izaberite Save Project ili Save Project As komandu File menija, pri čemu će Delfi od vas zatražiti ime za kod koji je pridružen formi, kao i ime za fajl u kom se nalazi sam projekat. Kako ime projekta treba da da se podudara sa naslovom forme (Hello), nazovimo fajl u kom se nalazi kod forme HELLOF.PAS, od Hello Form. Nazovimo projekat HELLO.DPR.

Ne možemo koristiti isto ime za projekat i za unit koji definiše formu: u svakoj aplikacijiova ova dva fajla moraju imati različita imena. Možete dodati slovo F, reč Form, nazvati svaki unit forme MainForm, ili izmislite svoj način imenovanja. Ovde je ime slično imenu projekta.

Ima koje ste dali fajlu u kom je projekat se po defaultu koristi kao naslov aplikacije koji se tokom izvršavanja vidi u taskbar meniju. Evo još jednog razloga da se ime projekta podudara sa naslovom glavne forme. Možete promeniti naslov aplikacije koristeći Application stranu Project Options dijalog boxa (izaberite Project/Options), ili unosom jedne linije koda koja će promeniti Title osobinu globalnog objekta koji predstavlja aplikaciju - Application (izaberite Project/View Source).

1.2. Korišćenje komponenti

Formu možemo posmatrati kao kontejner za komponente. Svaka forma može biti domaćin velikom broju komponenti i kontrola Komponentu možete izabrati sa Components palete koja se nalazi u Delfijevom prozoru, a iznad forme. Komponentu na formu možete smestiti na jedan od četiri navedena načina. Recimo da smo izabrali komponentu dugme sa standardne strane Components palete:

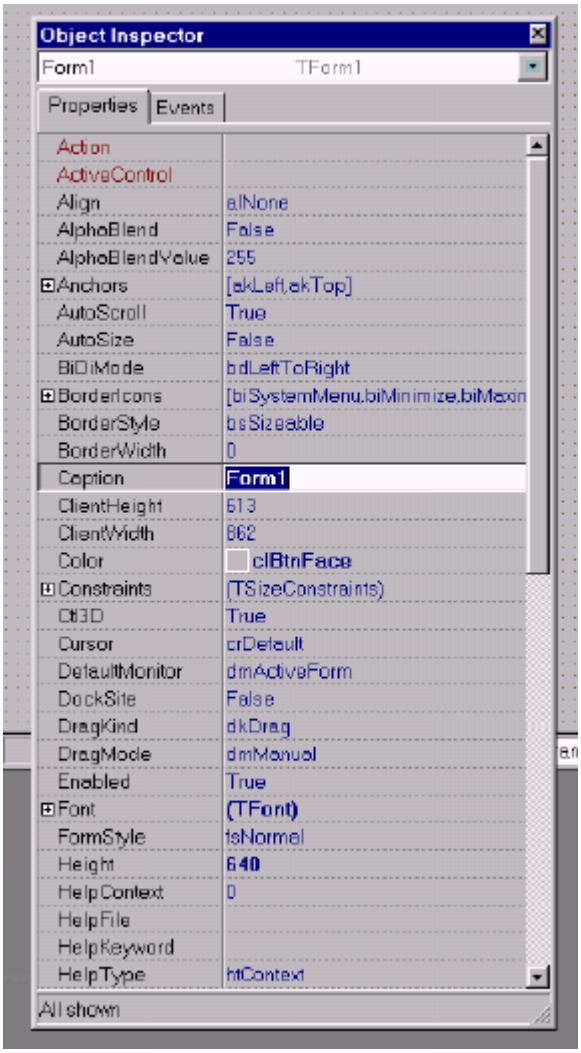
- kliknite na komponentu, pomerite kursor miša do forme, pritisnite levi taster miša da postavite gornji levi ugao dugmeta i povlačenjem miša odredite veličinu dugmeta,
- selektujte komponentu, kao gore, a tada kliknite na formu na mesto na kom želite dugme default veličine,
- duplo kliknite na ikonu na Components paleti i ta komponenta biće postavljena u centar forme,
- Shift-kliknite na ikonu komponente i postavite nekoliko komponenti tog tipa na formu koristeći neki od navedenih metoda.

Postavite jedno dugme u centar forme. Dugme možete centrirati uz pomoć Delfija: izaberite View/Alignment paletu i pojaviće se toolbox koji omogućava poravnanje i pozicioniranje komponenti i kontrola na formi.

1.3. Menjanje osobina

Ukoliko želimo da promenimo naslov dugmeta (naslov dugmeta je ono što na njemu piše), možemo promeniti Caption osobinu, kao i Name osobinu (vidi gore). Ove dve osobine u opštem slučaju treba da su različite. Name je ime kojim se komponenta identifikuje u kodu, a Caption ono što korisnik vidi. Caption je deskriptivna osobina i najčešće se sastoji iz više reči, što Name ne dozvoljava. Name je identifikator koji se sastoji iz slova i cifara i underscore karaktera, koji u sebi ne sadrži praznine i koji ne počinje cifrom.

Postavimo osobine dugmeta i forme:

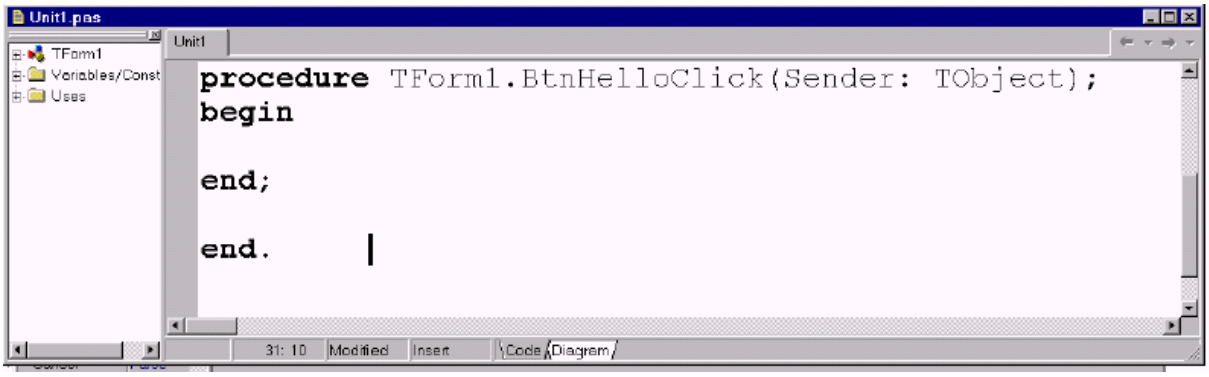


1.4. Reagovanje na događaje

Kada pritisnete dugme miša na formi ili komponenti, Windows obaveštava vašu aplikaciju o događaju slanjem poruke. Delfi odgovara time što primi obaveštenje o događaju i pozivom odgovarajućeg event-handlera (kod koji obrađuje događaj). Delfi za svaku vrstu komponente nudi izvestan broj event-handlera. Listu događaja za komponentu možete pronaći na Events strani Object Inspectora.

Možete definisati handler za OnClick događaj dugmeta na nekoliko načina:

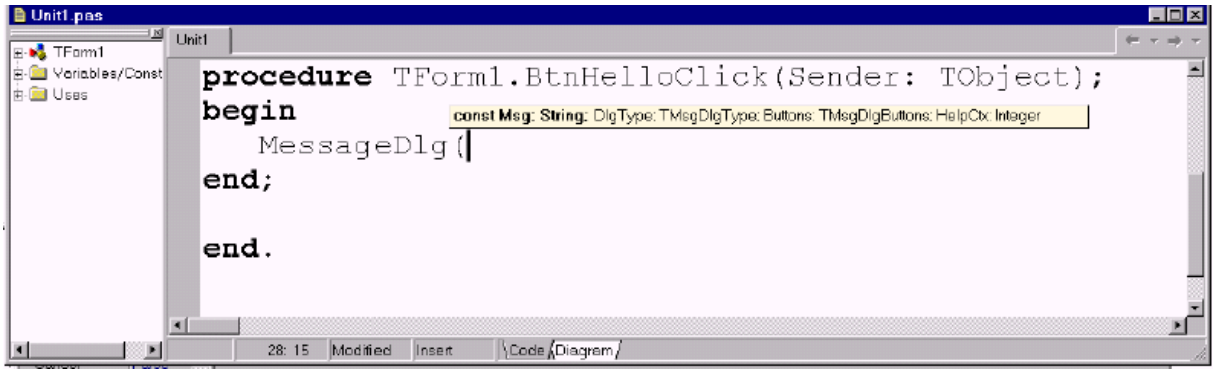
- selektujte dugme, ili na formi, ili korišćenjem combo boxa Object Inspectora, izaberite Event stranicu i duplo kliknite na prazan prostor na desnoj strani OnClick događaja, pojaviće se novo ime metoda, BtnHelloClick,
- izaberite dugme, izaberite Events stranicu i unesite ime novog metoda u prazan prostor na desnoj strani OnClick događaja. Pritisnite Enter tipku za kraj,
- duplo kliknite na dugme i Delfi će izvršiti default akciju za ovu komponentu, što je dodavanje handlera za OnClick događaj. Druge komponente imaju druge default akcije.



Bilo koji način za definisanje handlera rezultiraće procedurom BtnHelloClick u kodu forme i otvoriće fajl u kom je kod na mestu same procedure. Ukoliko ste kreirali proceduru za događaj koji ne želite da obradite, nemojte je ručno brisati. Delfi će sve prazne metode ukloniti pri sledećem snimanju. Kod za handler unosite između ključnih reči begin i end koje ograničavaju proceduru.

```
procedure TForm1.BtnHelloClick(Sender: TObject);
begin
  MessageDlg ('Hello guys!', mtInformation, [mbOK], 0);
end;
```

Kod sadrži samo poziv funkcije MessageDlg, koja će prikazati dijalog box. Kada ukucate otvorenu zagradu, Delfi će vam u posebno prozoru prikazati listu parametara metoda. Ukoliko su vam potrebna dodatna objašenja za funkciju, klinknite na nju, a zatim pritisnite tipku F1, čime ćete pokrenuti Delfijev help sistem, koji će vam dati detaljno objašnjenje funkcije.



Svaki put kada korisnik pritisne dugme na formi, poruka će biti prikazana. Šta se dešava ako korisnik pritisne tipku miša van dugmeta? Ništa, sve dok ne dodamo kod koji će obraditi taj događaj:

```
procedure TForm1.FormClick(Sender: TObject);
begin
  MessageDlg ('You have clicked outside of the button',
    mtWarning , [mbOK], 0);
end;
```

U novoj verziji programa će korisniku biti prikazana poruka upozorenja ukoliko pritisne taster miša van dugmeta. U gornjem kodu je poziv funkcije razbijen u dva reda, ali nove linije Pascal kompajler ignoriše, kao i praznine, razmake (tab), i slične znake za formatiranje. Naredbe se razdvajaju tačka zarezima.

1.5. Kompajliranje i pokretanje programa

Kada izaberete Run/Run, Delfi rade sledeće stvari:

- kompajlira Pascal izvorni kod koji opisuje formu,
- kompajlira fajl u kom se nalazi projekat,
- napravi izvršni (EXE) fajl i povezuje potrebne biblioteke i
- pokreće izvršni fajl, običnu u debug modu.

1.6. Menjanje osobina tokom izvršavanja

Ukoliko želimo da promenimo neke osobine ponašanja dugmeta tokom izvršavanja, recimo, da promenimo poruku iz Say hello u Say hello again nakon što korisnik prvi put pritisne dugme, dovoljno je dodati ovaj kod:

```
procedure TForm1.BtnHelloClick(Sender: TObject);
begin
  MessageDlg('Hello guys!', mtInformation, [mbOK], 0);
  btnHello.Caption := 'Say hello again';
end;
```

Većina osobina može biti promenjena tokom izvršenja, a neke mogu biti promenjenje SAMO tokom izvršavanja. Ove osobine se ne nalaze u Object Inspectoru, ali ćete ih naći u Helpu fajlu za datu komponentu.

1.7. Dvosmerni alat

Kod koji smo do sada napisali biće snimljen u Pascal fajl, koji smo nazvali HELLOF.PAS. U ovom fajlu se ne nalaze samo hadleri za razne događaje koji se mogu desiti na formi i njenim komponentama, već i podaci o tome koje sve komponente forma sadrži. Osobine forme i njoj pridruženih komponenti nalaze se u drugom fajlu koji se zove HELLOF.DFM. Delfi je dvosmerni alat jer sve što uradite u vizuelnom alatu ima svoju kodnu predstavu. Možete menjati kod i te promene će biti vidljive i u vizuelnom alatu, sve dok poštujete neka jednostavna pravila.

1.7.1. Pogled na izvorni kod

Pascal programi su podeljeni u module koji se zovu uniti. Kada kreirate novi projekat, Delfi generiše programski modul i unit koji definiše glavnu formu. Svaki put kada dodate formu, dodajete jedan novi unit. Uniti imaju ekstenziju PAS, a program DPR. Fajl Unit1 koristi niz paketa i definiše novi tip podataka (klasu) i novu promenjivu (objekat te klase). Klasa se zove TForm1 i nasleđuje TForm. Objekat je Form1 i tipa je TForm1. Kada preimenujete unit, promeniće se i kod:

```
unit HelloF;
```

Dodavanjem novih komponenti menja se deklaracija klase. Dodavanjem dugmeta, npr. dobijemo:

```
type
    TForm1 = class (TForm)
        Button1: TButton;
        ...
```

Kada preimenujete dugem, dobijete:

```
type
    TForm1 = class (TForm)
        BtnHello: TButton;
        ...
```

Menjanje drugih osobina, sem imena nema efekta na izvorni kod. Promene drugih osobina vidljive su u fajlu u kom se nalazi opis forme (sa ekstenzijom DFM). Najveći uticaj na kod ima dodavanje novih handlera za događaje:

```
unit HelloForm;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls;

type
  TForm1 = class(TForm)
    BtnHello: TButton;
    procedure BtnHelloClick(Sender: TObject);
    procedure FormClick(Sender: TObject);
    procedure FormResize(Sender: TObject);
  private
    { Private declarations }
```

```
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.dfm}

procedure TForm1.BtnHelloClick(Sender: TObject);
begin
   MessageDlg('Hello guys!', mtInformation, [mbOK], 0);
   btnHello.Caption := 'Say hello again';
end;

procedure TForm1.FormClick(Sender: TObject);
begin
  MessageDlg ('You have clicked outside of the button',
    mtWarning , [mbOK], 0);
end;

procedure TForm1.FormResize(Sender: TObject);
begin
  BtnHello.Top := Form1.ClientHeight div 2 -
    BtnHello.Height div 2;
  BtnHello.Left := Form1.ClientWidth div 2 -
    BtnHello.Width div 2;
end;

end.
```

Izvorni kod definiše reakcije sistema, a fajl koji sadrži opis forme određuje početno stanje sistema. Tekstualnu reprezentaciju (inače je u binarnom obliku) DFM fajla dobićete ako desnim dugmetom miša kliknete na formu i izaberete View As Text komandu. Možete menjati osobine forme menjajući tekstuelnu reprezentaciju njenih osobina, ali se ovo ne preporučuje.

Fajl u kom se čuva Delfi projekat sa ekstenzijom DPR je treći i najznačajniji fajl aplikacije. U DPR fajlu se nalazi Pascal izvorni kod koji opisuje opštu strukturu programa i startup kod:

```
program Project1;

uses
  Forms,
  HelloForm in 'HelloForm.pas' {Form1};

{$R *.res}

begin
  Application.Initialize;
  Application.CreateForm(TForm1, Form1);
  Application.Run;
end.
```

Ovaj kod možete videti izabiranjem View/Project Source komande.

2. Delfi okruženje

2.1 Delfijevi meniji i komande

U Delfiju možete izdati komandu na tri načina:

- koristeći meni,
- koristeći tool bar i
- koristeći jedan od lokalnih menija kji se aktiviraju pritiskom na desno dugme miša.

2.1.1. File meni

File meni je pull down meni koji sadrži komande koje se odnose na projekte i na fajlove sa izvornim kodom. Komande koje se odnose na projekat su New, New Application, Open, Save Project As, Save All, Close All, Add to Project i Remove from Project. Pored ovih komandi, komande koje se odnose na projekat sadrži i Project

meni. Komande koje se odnose na izvorni kod su New, New Form, New Data Module, Open, Reopen, Save, Save As, Close i Print. New je komanda koja otvara dialog box koji omogućava korisnuku da bira šta novo želi: formu, aplikaciju, CLX aplikaciju, unit...

2.1.2. Edit meni

Meni za editovanje ima standardne mogućnosti, kao što su Undo, Redo, Cut, Copy, Paste komande. Važno je znati da tastaturne prečice Ctrl+Z, Ctrl+V, Ctrl+X i Ctrl+C rade i sa tekstom i sa komponentama forme.

2.1.3. Meni za pretraživanje – Search meni

Meni za pretraživanje ima standardne komande Search and Replace i Find in Files sa posebnim opcijama. Incremental Search komanda prebacuje kontrolu u editor i traži reč koja se podudara sa rečju koju ukucavate. Reč koju ukucavate se nigde pritom ne vidi. Prečica je Ctrl+E.

2.1.4. View meni

View meni kombinuje osobine koje nalazite u View i Window menijima. Komanda Toggle Form/Unit (ili F12) prebacuje kontrolu sa forme na njen izvorni kod. Komanda New Edit Window otvara drugi prozor za editovanje, što je jednini način u Delfiju da imate dva fajla jedan pored drugog.

2.2. Delfijev dizajner za forme

2.2.1. Paleta komponenti

Paleta komponenti ima veći broj stranica, kao što su Standard, Additional, Win32, System, Data Access, Dialogs itd. Možete proširiti paletu dodavanjem svojih komponenti, kao što možete preurediti raspored komponenti u postojećim stranicama biranjem Tool/Environment Options i stranice Palette.

2.2.2. Object Inspector

Tokom dizajniranja forme koristićete Object Inspector da postavite vrednosti osobina forme ili komponente. Prozor Object Inspectora prikazuje osobine (i događaje) selektovane komponente, kao i vrednosti osobina (imena handlera za događaje). U zavisnosti od osobine, kao vrednost možete uneti string ili broj, možete birati među ponuđenim opcijama, ili pokrenuti određeni editor. Postoje i osobine koje imaju svoje podosobine (kao npr. Font), koje takođe možete podesiti.

2.2.3. Paleta za poravnanje

Paletu za poravnanje možete otvoriti pomoću Alignment Palette View menija, kao i pomoću komande Align lokalnog menija, koji će se pojaviti desnim klikom miša na selektovane komponente.

3. Prikaz osnovnih komponenti

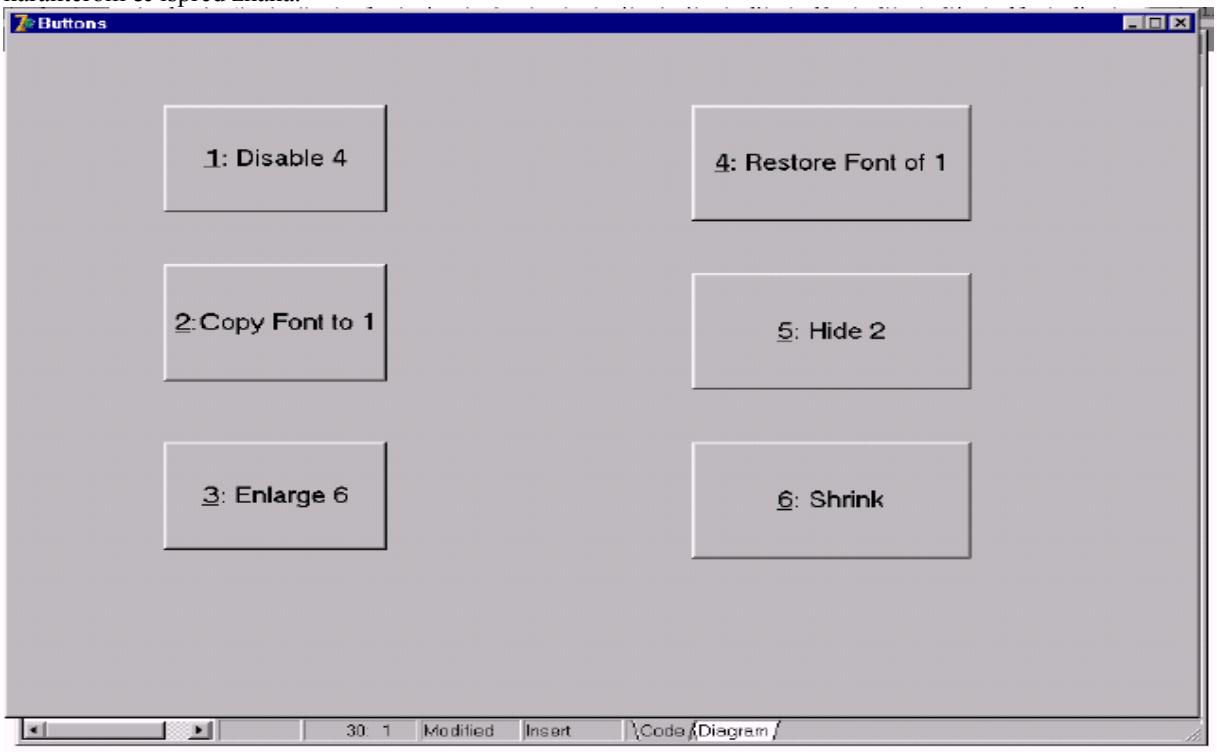
3.1 Klik na dugme

Već smo napravili aplikaciju koja ima dugme. Nova aplikacija treba da ima nekoliko dugmadi, pri čemu će klik na jedno dugme izazvati promenu na drugom.

Otvorimo novi projekat i imenujmo ga, snimanjem na disk. Unitu dajte ime ButtonF, a projektu Buttons.dpr. Napravimo šest dugmadi. Dugmad se poravnavaju korišćenjem Edit/Align komade. Svih šest dugmadi pre toga moraju biti selektovana. Dugmad ćete selektovati ili povlačenjem pravougaonika oko dugmadi, ili selektovanjem dugmadi sa pritisnutim Shift tasterom. Da biste poravnali dugmad kao na slici selektujte tri po trri.

Imamo formu sa šest dugmadi na njoj. Dajmo formi ime ButtonsForm, a naslov Buttons. Postavimo natpis (Caption) za dugmad. Natpis opisuje funkciju dugmeta. Broj koji se nalazi na početku natpisa svakog dugmeta

nalazi se podvučen broj koji predstvalja tastere skraćenicu. Efekat znaka-skraćenice postiže se ampersend karakterom & ispred znaka.



U nastavku je dat kod koji opisuje osobine dugmadi i forme.

```
object ButtonsForm: TButtonsForm
  Caption = 'Buttons'
  object Button1: TButton
    Caption = '&1: Disable 4'
    OnClick = Button1Click
  end
  object Button2: TButton
    Caption = '&2:Copy Font to 1'
    Font.Color = clBlack
    Font.Height = -20
    Font.Name = 'Arial'
    Font.Style = [fsBold]
    ParentFont = False
    OnClick = Button2Click
  end
  object Button3: TButton
    Caption = '&3: Enlarge 6'
    OnClick = Button3Click
  end
  object Button4: TButton
    Caption = '&4: Restore Font of 1'
    OnClick = Button4Click
  end
  object Button5: TButton
    Caption = '&5: Hide 2'
    OnClick = Button5Click
  end
  object Button6: TButton
    Caption = '&6: Shrink'
    OnClick = Button6Click
  end
end
```

Glavni deo koda se odnosi na programiranje događaja OnClick. Najjednostavniji je kod handlera drugog i četvrtog dugmeta. Kada je pritisnuto dugme 2, program kopira font dugmeta (koji je drugačiji od svih ostalih) na font dugmeta 1, a zatim se onemogući.

```
procedure TButtonsForm.Button2Click(Sender: TObject);
begin
  Button1.Font := Button2.Font;
  Button2.Enabled := False;
end;
```

Pritiskom na dugme 4povratiće se originalni font dugmeta 1. Umesto kopiranja fonta, povraćaj fonta vrši se korišćenjem ParentFont osobine. Događaj će osposobiti dugme 2 za ponovnu upotrebu.

```
procedure TButtonsForm.Button4Click(Sender: TObject);
begin
  Button1.ParentFont := True;
  Button2.Enabled := True;
end;
```

Za implemetiranje Disable i Hide oparcije dugmadi 5 i 1 može se upotrebiti logička promenjiva koja bi čuvala trenutni status. Drugo rešenje podrazumeva određivanje naredne operacije na osnovu trenutnog stanja osobine Enabled dugmeta. Dva metoda koriste različite pristupe:

```
procedure TButtonsForm.Button1Click(Sender: TObject);
begin
  if not Button4.Enabled then
  begin
    Button4.Enabled := True;
    Button1.Caption := '&1: Disable 4';
  end
  else
  begin
    Button4.Enabled := False;
    Button1.Caption := '&1: Enable 4';
  end;
end;

procedure TButtonsForm.Button5Click(Sender: TObject);
begin
  Button2.Visible := not Button2.Visible;
  if Button2.Visible then
    Button5.Caption := '&5: Hide 2'
  else
    Button5.Caption := '&5: Show 2';
end;
```

Dugmad 3 i 6 imaju neograničen kod. Neograničen znači da dugme 6 možete toliko smanjiti da potpuno nestane.

```
procedure TButtonsForm.Button3Click(Sender: TObject);
begin
  Button6.Height := Button6.Height + 3;
  Button6.Width := Button6.Width + 3;
end;

procedure TButtonsForm.Button6Click(Sender: TObject);
begin
  Button6.Height := Button6.Height - 3;
  Button6.Width := Button6.Width - 3;
end;
```

3.1. Klik na dugme miša

Šta je klik? Na prvi pogled, klik podrazumeva pritisak na levi taster miša. To jeste tačno, ali je klik komplikovaniji. Kada korisnik pritisne levi taster miša na komponenti dugme, komponenta će biti i grafički pritisnuta. Ukoliko korisnik pomeri kursor (sve držeći pritisnut levi taster miša) van dugmeta, dugme će ponovo biti »nepritisnuto«. Ukoliko korisnik u ovoj situaciji otpusti taster van površine dugmeta, neće se desiti klik. Ukoliko korisnik vrati kursor na dugme, ono će opet biti pritisnuto, a kada se otpusti taster miša, desiće se klik. Delfi ima događaje koji obaveštavaju o pritisku na taster miša, o otpuštanju tastera miša itd.

3.2. Obojeni tekst na formi

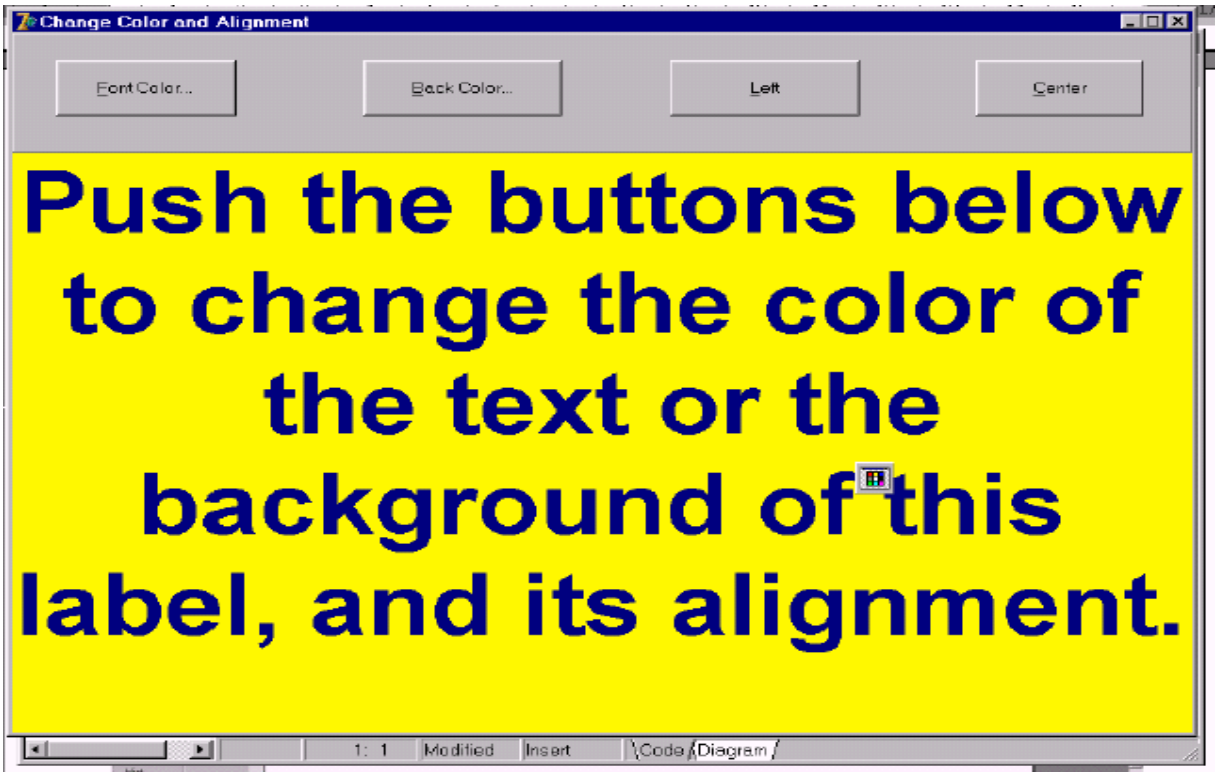
Pozabavimo se labelama. Labele su tekst ili komentari na formi. Korisnici obično ne komuniciraju sa labelama, bar ne direktno, pa stoga nema mnogo smisla kliknuti na labelu. Nije sav tekst na formi u okviru labele. Tekst se može postaviti i direktno na formu, koristeći TextOut metod.

Labele se koriste da opišu druge komponente, obično polja za editovanje, liste ili kombo boxove, jer oni nemaju naslov. Iskoristićemo instancu labele da u sebi čuva tekst kom ćemo tokom izvršavanja programa menjati boju, centriranje, kao i boju pozadine.

Postavimo veeeeliku labelu na formu i unesimo u nju duži tekst. Postavljanjem WordWrap osobine na True omogućimo da labela sadrži nekoliko linija teksta, a postavljanjem AutoSize osobine na False omogućavamo da labela slobodno menja veličinu.

Da bi labela tokom izvršavanja mogla da menja boju fonta i pozadine, kao i centriranje, iskoristićemo dugmad. Mesto da dugmad postavimo direktno na formu, postavimo ih na Panel komponentu, koju smo pre toga smestili na formu. Potrebna su nam četiri dugmeta: dva za boju i po dva za centriranje. Postavljanjem dugmadi na Panel, Panel postaje Parent komponenta za dugmad: kooordinate dugmadi biti će relativne u odnosu na Panel, što znači da će se pomeranjem Panela pomeriti i dugmad, itd.

Panel i labela se mogu poravnati osobinama, za razliku od dugmadi. Panel će biti zalepljen za vrh i raširen po celoj dužini forme ukoliko postavite alTop osobinu na true. Labela će zauzeti sav preostali prostor forme postavljanjem osobine Align osobine na alClient. Potrebna nam je još jedna komponenta: ColorDialog koji se nalazi u Dialogs strani palete komponenti. Ova komponenta poziva standardni Windows dijalog box za boje.



object ColorTextForm: TColorTextForm

```
  Caption = 'Change Color and Alignment'
  object Label1: TLabel
    Align = alClient
    Alignment = taCenter
    AutoSize = False
    Caption =
      'Push the buttons below to change the color of the text or the background ... '
    Color = clYellow
    Font.Color = clNavy
    Font.Name = 'Arial'
    Font.Style = [fsBold]
    WordWrap = True
  end
  object Panel1: TPanel
    Align = alTop
    object BtnFontColor: TButton
    object BtnBackColor: TButton
    object BtnLeft: TButton
    object BtnCenter: TButton
  end
  object ColorDialog1: TColorDialog
end
```

Preostaje pisanje koda za handlere događajima. Metodi za klik na dugme za centriranje su veoma jednostavni (drugi metod koristi TaCEnter umesto laLeftJustify):

```
procedure TColorTextForm.BtnLeftClick(Sender: TObject);
begin
  Label1.Alignment := taLeftJustify;
end;
```

Color dialog box postavite bilo gde na formu. Mesto nije važno, pošto se ova komponenta ne vidi tokom izvršavanja. Komponentu koristi sledeći kod:

```
procedure TColorTextForm.BtnFontColorClick(Sender: TObject);
begin
  ColorDialog1.Color := Label1.Font.Color;
  if ColorDialog1.Execute then
    Label1.Font.Color := ColorDialog1.Color;
end;
```

U prvoj liniji postavljamo boju pozadine za inicijalnu boju koju prikazuje dialog box. U drugoj liniji pokrećemo dialog box, a u trećoj boju koju je korisnik u dialog boxu izabrao dodeljujemo boji fonta labele. Ova zadnja operacije će se izvršiti samo ukoliko korisnik zatvori dialog box pomoću OK dugmeta (jer tada Execute metod vraća True). Za promenu koda poazdine pišemo veoma sličan komad koda, samo u ovom sllučaju mesto Label1.Font.Color pišemo Label1.Color.

3.3. Prihvatanje korisničkog unosa

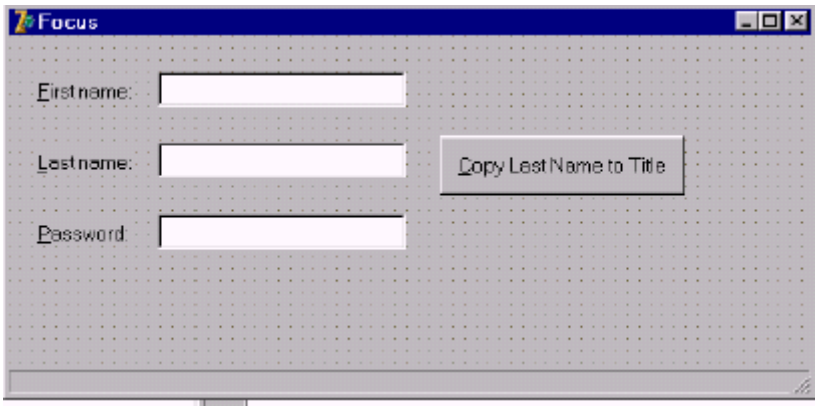
Windows rukuje unosom sa tastature direktno, sem u nekim posebnim slučajevima. Bilo kako bilo, handleri za događaje tastature nisu uobičajeni, jer sistem nudi gotova polja za editovanje i jednostavne tekst editore: Edit, MAskEdit, Memo, RichText... Osnovne su Edit i Memo.

Edit komponente omogućava jednu liniju teksta i ima određene osobine, kao što je ograničavanje broja unetih znakova, ili osobinu koja prikazuje specijalan password karakter umesto stavrnog teksta. Memo može da sadrži više linija teksta.

U narednom primeru prikazaćemo osobinu zajedničku mnogim komponentama: ulazni fokus. U windows okruženju je lako odlučiti koji je prozor aktivan, ali nije lako odlučiti koji prozor ili komponenta ima ulazni fokus.Ukoliko korisnik pritisne tipku, koja komponenta će primiti odgovarajuću poruku ulaza sa tastature? To može biti aktivni prozor, ali takože i jedna od njegovih kontrola. Ukoliko imamo formu sa više edit polja, samo

jedno ima ulazni fokus u datom trenutku. Korisnik može da menja fokus pomoću Tab tastera, ili klikom mišem na određeno edit polje.

U sledećem primeru je bitno zapamtiti da svaki put kada komponenta primi ili izgubi ulazni fokus, ona prima i događaj koji je obaveštava da je korisnik pristupio komponenti (OnEnter) ili je napuistio (OnExit).



Pored tri edit polja, form aima i labele koje objašnjavaju smisao edit polja. Za ispisivanje informacije o stanju iskorištena je standardna Windows komponenta StatusBar, koja može biti zamenjena labelom ili panelom. Komponenta StatusBar kao jednolinijsko izalzno polje postavljanjem osobine SimplePanel na True:

```
object FocusForm: TFocusForm
  Caption = 'Focus'
  object Label1: TLabel
    Caption = '&First name:'
    FocusControl = EditFirstName
  end
  object Label2: TLabel
    Caption = '&Last name:'
    FocusControl = EditLastName
  end
  object Label3: TLabel
    Caption = '&Password:'
    FocusControl = EditPassword
  end
  object EditFirstName: TEdit
    TabOrder = 0
    OnEnter = EditFirstNameEnter
  end
  object EditLastName: TEdit
    TabOrder = 1
    OnEnter = EditLastNameEnter
  end
  object EditPassword: TEdit
    PasswordChar = '*'
    TabOrder = 2
    OnEnter = EditPasswordEnter
  end
  object ButtonCopy: TButton
    Caption = '&Copy Last Name to Title'
    TabOrder = 3
    OnClick = ButtonCopyClick
    OnEnter = ButtonCopyEnter
  end
  object StatusBar1: TStatusBar
    SimplePanel = True
  end
end
```

Forma sadrži i dugme koje kopiraja text iz LastNameEdit polja u naslov forme. Dobra praksa podrazumeva da se pre dodele vrednosti naslovu polja proveri da li je korisnik zaista nešto i uneo u edit polje. Status polje će obaveštavati korisnika o promeni fokusa. Za ovo su nam potrebna četiri metoda koja se odnose na OnEnter događaj; po jedan za svako edit polje i jedan za dugme.

```
procedure TFocusForm.ButtonCopyClick(Sender: TObject);
begin
  if EditLastName.Text <> '' then
    FocusForm.Caption := EditLastName.Text;
end;

procedure TFocusForm.EditFirstNameEnter(Sender: TObject);
begin
  StatusBar1.SimpleText := 'Entering the first name...';
end;
```

Testirajte program korišćenjem miša, , tastature ili Tab tastera. Pritiskom na Tab taster, ulazni fokus kružiće između edit komponenti i dugmeta, ali ne i labela. Da osigurate određeni redosled, promenite TabOrder osobinu komponenti u Object Inspectoru, ili korišćenjem lokalnog menija i njegove komande Tab Order koja poziva Edit Tab Order dijalog box. Drugi način da selektujete komponentu jeste upotreba tastera za prečicu. Taster za prečicu se lako postavlja na dugme, ali sa edit poljima to nije moguće direktno izvesti. Možete dodati taster za prečicu (& karakter) labeli, a zatim postaviti FocusControl osobinu labele za odgovarajuću edit komponentu.

U ovom primeru smo pisali četiri različita OnEnter handlera, kopirali četiri stringa u text StatusBar komponente. Sa više edit polja situacija se značajno komplikuje. Alternativa podrazumeva pisanje jednog OnEvent handlera za sva edit polja i dugme. Potrebo je samo da poruku za StatusBar postavimo u osobinu , a kasnije da pogledamo u tu osobinu Sender objekta. Pogodna osobina je Hint, koja je namenjena korisniku i nosi opis:

```
procedure TFokusForm.GlobalEnter(Sender: TObject);
begin
  StatusBar1.SimpleText := (Sender as TControl).Hint;
end;
```

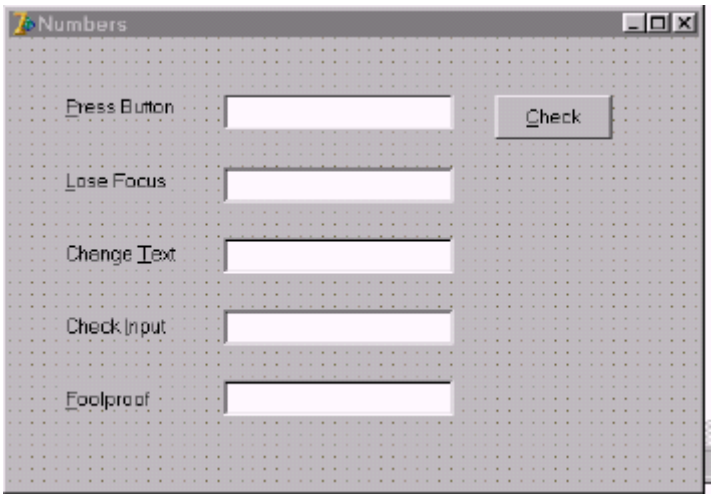
Sender nije mogao biti posmatran kao TEdit tip, jer sender može biti i dugme. Rešenje je posmatrati komponente kao klasu prvog zajedničkog roditelja klasa TButton i TEdit, a koja ima Hint osobinu.

3.4. Unošenje brojeva

Za unošenje brojeva možete iskoristiti MAskEdit komponentu (Additional strana palete komponenti), ili koristiti Edit komponentu, a posle konvretovati uneti string u integer, pomoću standardne Pascalske Val procedure ili Delfijeve IntToStr funkcije. Problem nastaje kada korisnik ukuca slovo mesto cifre. Funkcije za konvreziju vratiće kod za grešku, pa ih možemo iskoristiti i za testiranje unosa. Drugo je pitanje kada izvršiti ovaj test? Kada se vrednost u edit polju promeni, kada komponenta izgubi fokus, ili kada korisnik klikne na određeno dugme? Nisu svi načini podjednako dobri, kao što će pokazati sledeći primer.

Postoji i drugo rešenje: posmatrati ulazni tok u edit polje i zaustaviti svaki nenumerički unos. Ova tehnika je podlžna greškama, jer korisnik uvek može kopirati pogrešan string u edit polje, ali se jednostavno implementira i u kombinaciji sa drugim tehnikama greška se može odkloniti.

Primer prikazuje nekoliko tehnika za unos broja sa tastature, kao i osvrt na ulazni fokus. Za rukovanje numeričkim unosom koristićete uglavnom određene komponente, kao što je SpinEdit ili UpDown kontrole. Forma ima pet polja za editovanje i pet odgovarajućih labela koje opisuju u kom trenutku i kako se vrši kontrola unosa. Forma sadrži i dugme za proveru sadržaja prvog edit polja.



Sadržaj prvog edit polja se provera kada se pritisne Check dugme. Handler za Check OnClick događaj izgleda ovako:

```
procedure TNumbersForm.CheckButtonClick(Sender: TObject);
var
  Number, Code: Integer;
begin
  if Edit1.Text <> '' then
  begin
    Val (Edit1.Text, Number, Code);
    if Code <> 0 then
    begin
      Edit1.SetFocus;
      MessageDlg ('Not a number in the first edit', mtError, [mbOK], 0);
    end
    else
      MessageDlg ('OK, the number in the first edit box is ' +
        IntToStr (Number), mtInformation, [mbOK], 0);
  end;
end;
```

Kada se desi greška, program vraća fokus nazad na edit polje pre nego što pokaže poruku o grešci korisniku, pozivajući ga da unese ispravnu vrednost. Korisnik, naravno, može da ignoriše upozorenje i da se prebaci na sledeće polje za unos. Ista vrsta provere radi se i za drugo polje za editovanje kada ono igubi fokus. U ovom slučaju se poruka prikazuje automatski, ali samo ukoliko je došlo do greške. Kôd nigde ne koristi Edit2 komponentu, nego generičku Sender kontrolu. Edit polja su označena Tag osobinom, tako da svako edit polje ima svoj redni broj. Metod je nešto složeniji, ali će biti moguće ponovo ga koristiti u drugoj komponenti:

```
procedure TNumbersForm.Edit2Exit(Sender: TObject);
var
  Number, Code: Integer;
  CurrEdit: TEdit;
begin
  CurrEdit := Sender as TEdit;
  if CurrEdit.Text <> '' then
  begin
    Val (CurrEdit.Text, Number, Code);
    if Code <> 0 then
    begin
      CurrEdit.SetFocus;
      MessageDlg ('The edit field number ' +
        IntToStr (CurrEdit.Tag) +
        ' does not have a valid number',
        mtError, [mbOK], 0);
    end;
```

```
  end;
end;
```

Treća Edit komponenta testira svoj sadržaj svaki put kada se on promeni (koristi OnChange događaj). Četvrto edit polje koristi potpuno različitu tehniku od prethodna tri, koja proveravaju string nakon što ga je korisnik uneo. Četrvto vrši proveru pre nego Edit uopšte zna da je taster pritisnut. Edit komponente imaju OnKeyPress događaj koji odgovara akcijama korisnika. Korišćenjem ovog metoda i testirati da li je karaker broj ili Backspace taster (numerčke vrednosti 8, pa ga proveravamo kao karakter #8). Ukoliko nije, promenimo vrednost tastera u null karakter (#0), koji edit neće obraditi, i proizvešće upozoravajući zvik:

```
procedure TNumbersForm.Edit4KeyPress(Sender: TObject; var Key: Char);
begin
  {check if the key is a number or backspace}
  if not (key in ['0'..'9', #8]) then
  begin
    Key := #0;
    Beep;
  end;
end;
```

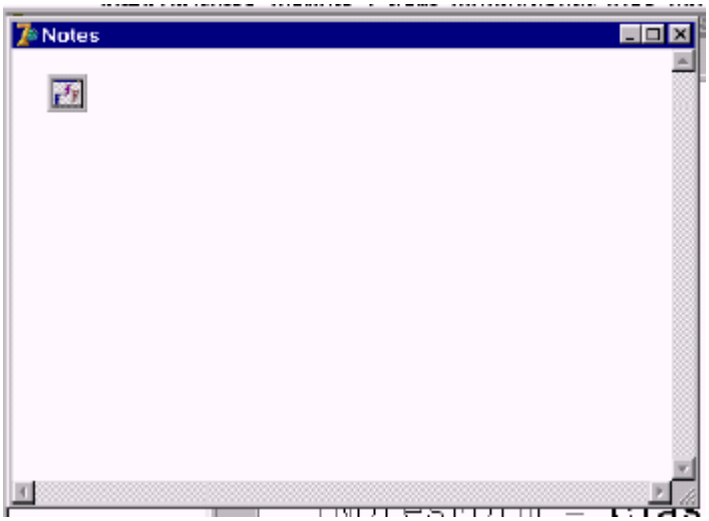
Četvrto edit polje prihvata samo brojeve kao unos, ali se pomoću Copy/Paste može prevariti. Ovaj propust može se prevazići kombinacijom sa metodom drugog ili trećeg polja. Peto polje koje nema propusta: koristi OnKeyPress događaj četvrtog, OnChange trećeg i OnExit drugog polja, pa stoga ne zahteva novi kod, već kombinovanje već postojećeg.

Da bi se iskoristio postojeći metod za novi događaj potrebno je izabrati Events stranicu u Object Inspectoru, izabrati odgovarajuću komponentu, i umesto duplog klika na levu stranu od imena događaja, izabrati combo box na desnoj strani. Pojaviće se lista imena svih postojećih metoda koji su kompatibilni sa trenutnim (imaju isti broj parametara). Izaberite odgovarajuće metode i peta komponenta biće kombinacija treće i četrvte. Zbog toga je u handleru treće i četrvte komponente korišten generički Sender objekat. Ovaj postupak možete primeniti na ovakve metode sve dok se odnose na metode istog tipa komponente.

3.5. Prosti editor

Polja za editovanje mogu da prihvate samo tekst graničene veličine i to u jednoj liniji. Ukoliko je potrebno prihvatiti duži tekst, potrebno je koristiti Memo komponentu, koja je slična Edit komponenti, ali može da se proširi na nekoliko linija, sadrži skrol barove i može da primi više teksta. Prikazaćemo u primeru koji treba da podseća na Windows Notepad, ali koji ima samo opciju podešavanja fonta.

Dovoljno je postaviti Memo komponentu na formu, obrisati njen tekst, ukloniti ivice, postaviti Alignment na alClient. Dodajte oba skrol bara (horizontalni i vertikalni) selektovanjem ssBoth vrednosti za ScrollBar osobinu.



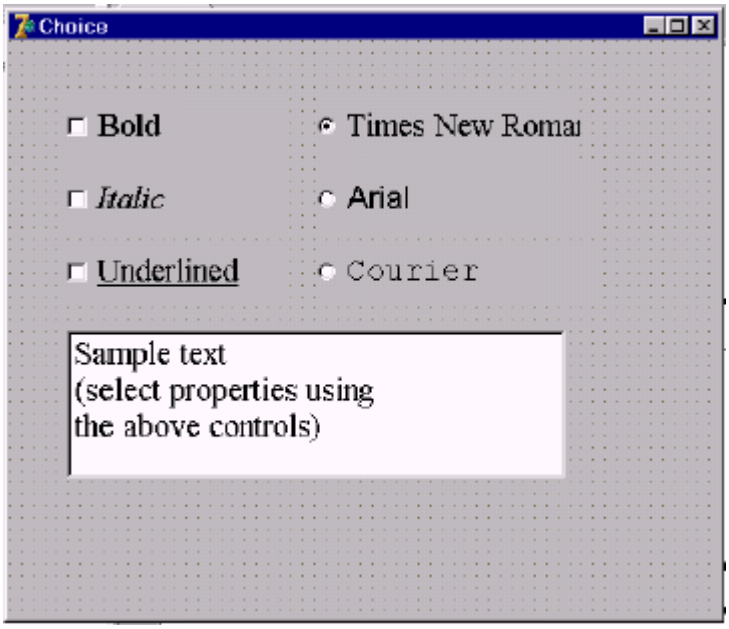
object NotesForm: TNotesForm

```
  Caption = 'Notes'
  object Memo1: TMemo
    Align = alClient
    BorderStyle = bsNone
    Font.Height = -23
    Font.Name = 'Times New Roman'
    ScrollBars = ssBoth
    OnDblClick = Memo1DblClick
  end
  object FontDialog1: TFontDialog
    ...
  end
end
```

3.6. Omogućavanje izbora

Windows nudi dve standardne kontrole koje korisniku omogućavaju da bira ponuđeno: check box koji omogućava slobodno biranje i radio button, koji odgovara ekskluzivnoj selekciji. Ukoliko postoje dva dugmete, A i B, možete selktovati ili A , ili B, ali nikao oba zajedno. Uz to, morate izabrati bar jedno dugme.

Sledeći primer pokazuje razliku između ove dve komponente: check boxovi za biranje stila fonta Bold, Italic i Underlined, i tri radio dugmeta kojim se bira vrsta fonta. Forma sadrži i Memo polje koje sadrži tekst koji će prikazati posledicu korisnikovog izbora.



Svaki put kada korisnik klikne na check box ili radio dugme, treba promeniti tekst Memo komponente na zadati način. Za stilove je potrebno pogledati Check osobinu i dodati ili ukloniti odgovarajući element iz skupa Style osobine Font Memo komponente.

```
procedure TForm1.CheckBoldClick(Sender: TObject);
begin
  if CheckBold.Checked then
    Memo1.Font.Style := Memo1.Font.Style + [fsBold]
  else
    Memo1.Font.Style := Memo1.Font.Style - [fsBold];
end;
```

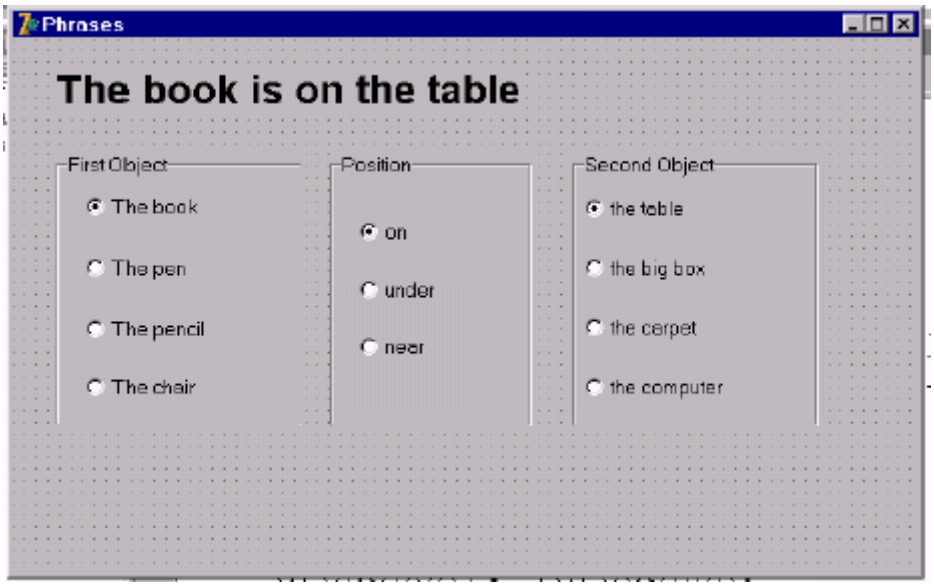
Druga dva check boxa imaju veoma sličan kod za OnClick događaj. Kod za radio dugmad je još jednostavniji:

```
procedure TForm1.RadioArialClick(Sender: TObject);
begin
```

```
  Memo1.Font.Name := 'Arial';
end;
```

Radio dugmad omogućavaju ekskluzivne izbore. Ponekad je potrebno na formi imati nekoliko grupa radio dugmadi. Da bi se Windowsu omogućilo da odredi odnose među različitim radio dugmadima, postoje kontejner komponente. Delfijeva komponenta koja omogućava grupisanje dugmadi, funkcionalno i vizuelno je GroupBox, dok je RadioGroup komponenta kao group box, ali koji sadrži samo radio dugmad.

Napravićemo formu koja sadrži tri grupe radio dugmadi koje omogućavaju kreiranje rečenice na stranom jeziku. Kreirana rečenica pokazaće se u labeli. Prvu grupu dugmadi First Object kreiraćemo tako što najpre na fomru postavimo GroupBox komponentu, a na nju radio dugmad i postavimo nazive za dugmad i za GroupBox. Treća grupa je implementirana RadioGroup komponentom, u kojoj se mogući izbori (radio dugmad navode u Items osobini).



Na formu treba dodati i labelu, izabrati veliki font, i uneti tekst koji odgovara inicijalno selektovanoj dugmadi. U svakoj grupi treba da ibude izabrano po jedno dugme, pri čemu ItemIndex osobina RadioGroup određuje inicijalno izabrano dugme.

```
object Form1: TForm1
  Caption = 'Phrases'
  object Label1: TLabel
    Caption = 'The book is on the table'
    Font.Height = -27
    Font.Name = 'Arial'
    Font.Style = [fsBold]
    ParentFont = False
  end
  object GroupBox1: TGroupBox
    Caption = 'First Object'
    object RadioBook: TRadioButton
      Caption = 'The book'
      Checked = True
      OnClick = ChangeText
    end
    object RadioPen: TRadioButton
      Caption = 'The pen'
      OnClick = ChangeText
    end
    object RadioPencil: TRadioButton
      Caption = 'The pencil'
      OnClick = ChangeText
    end
```

```
    object RadioChair: TRadioButton
      ...
    end
  end
  object GroupBox2: TGroupBox
    Caption = 'Position'
    object RadioOn: TRadioButton
      Caption = 'on'
      Checked = True
      OnClick = ChangeText
    end
    object RadioUnder: TRadioButton ...
    object RadioNear: TRadioButton ...
  end
  object RadioGroup1: TRadioGroup
    Caption = 'Second Object'
    Items.Strings = (
      'the table'
      'the big box'
      'the carpet'
      'the computer')
    OnClick = ChangeText
  end
end
```

U nastavku je dat kod koji omogućava da se fraza prikazana labelom promeni kada korisnik klikne na radio dugme. Jedan način je napisati po jedan OnClick handler metod za svako dugme. U ovom slučaju neophodno je delove fraze čuvati u promenjivama forme, menjati ih u skladu sa dugmadima i na kraju frazu ponovo sastaviti. Drugo rešenje podrazumeva jedan metod koji proverava koja su dugmad trenutno pritisnuta i u odnosu na to napravi frazu. Ovaj metod mora biti povezan sa OnClick događajima svih dugmadi - potrebno je selektovati svu dugmad u Object Inspectoru i imenovati metod u drugoj koloni stranice događaja Object Inspectora:

```
procedure TForm1.ChangeText(Sender: TObject);
var
  Phrase: string;
  I: integer;
begin
  {look at which radio button is selected and add its text to the phrase}
  for I := 0 to GroupBox1.ControlCount - 1 do
    if (GroupBox1.Controls[I] as TRadioButton).Checked then
      Phrase := (GroupBox1.Controls[I] as TRadioButton).Caption;

  {add the verb and blank spaces}
  Phrase := Phrase + ' is ';

  {repeat the operation on the second group box}
  for I := 0 to GroupBox2.ControlCount - 1 do
    with GroupBox2.Controls[I] as TRadioButton do
      if Checked then
        Phrase := Phrase + Caption;

  {retrieve the radiogroup selection, and display the result in the label}
  Label1.Caption := Phrase + ' ' +
    RadioGroup1.Items [RadioGroup1.ItemIndex];
end;
```

ChangeText metod najpre traži koje dugme iz prve grupe je selektovano, pa dodaje glagol i praznine, drugu reč, pa zatim treću, izabranu u RadioGroup komponenti. for petlje počinju 0, i idu do broja dugmadi minus 1, jer niz Controls započinje 0. Cast je neophodan, jer generička komponenta nema osobinu Checked. Kod je znatno jednostavniji za RadioGroup komponentu, jer ona ima ItemIndex osobinu koja označava selektovano dugme.

3.7. Lista sa mnoštvom izbora

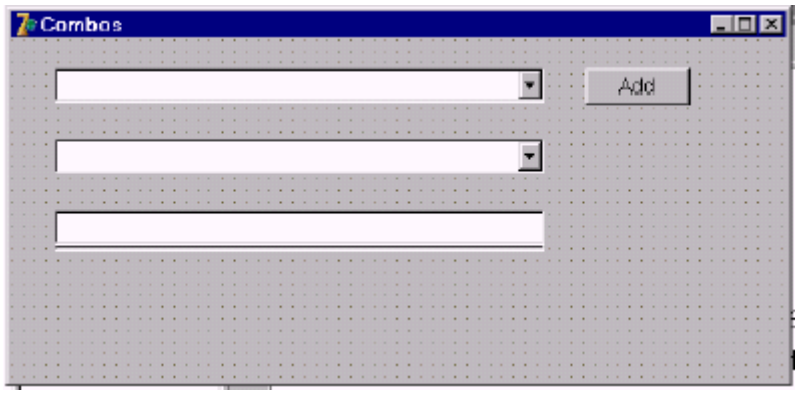
Ukoliko želite da omogućite mnoštvo izbora, Radio buttons nisu rešenje, osim ukoliko želite da napravite baš veliku formu. List box može sadržati veliki broj izbora na malom prostoru, jer sadrži skrol bar. Druga prednost je što se novi izbori mogu relativno lako dodati,a stari izbaciti.

3.8. Mnogo lista, malo prostora

ComboBox je komponenta slična edit komponenti; možete uneti tekst u nju. Slična je i list boxu, jer ima drop-down strelicu kojom se prikazuje sadržaj liste. Ponašanje ComboBox komponente se značajno menja u zavisnosti od toga kako je postavljena osobina Style:

- csDropDown predstvalja tipičan ComboBox, koji dozvoljava direktno editovanje i prikazuje listu na zahtev,
- csDropDownList stil ne dozvoljava editovanje. Pritiskom na taster korisnik selektuje prvi element liste koji započinje tim slovom,
- csSimple definiše ComboBox koji uvek prikazuje listu, i dozvoljava editovanje,
- csOwnerDrawFixed i csOwnerDrawVariable definišu kombo boxove na osnovu liste koja sadrži grafike koje postavlja program, a ne proste stringove.

Dat je primer koji prikazuje kombo boxove tri različita stila: drop-down, drop-down list i simple.



Svaki kombo box ima iste stringove sa imenima više od 20 različitih životinja. Uz prvi kombo box ide i Add dugme. Kada korisnik pritisne ovo dugme, tekst koji je korisnik uneo u kombo box biće dodat listi imena životinja, ukoliko isti string već ne postoji. Kod koji se izvršava kao handler za OnClick događaj:

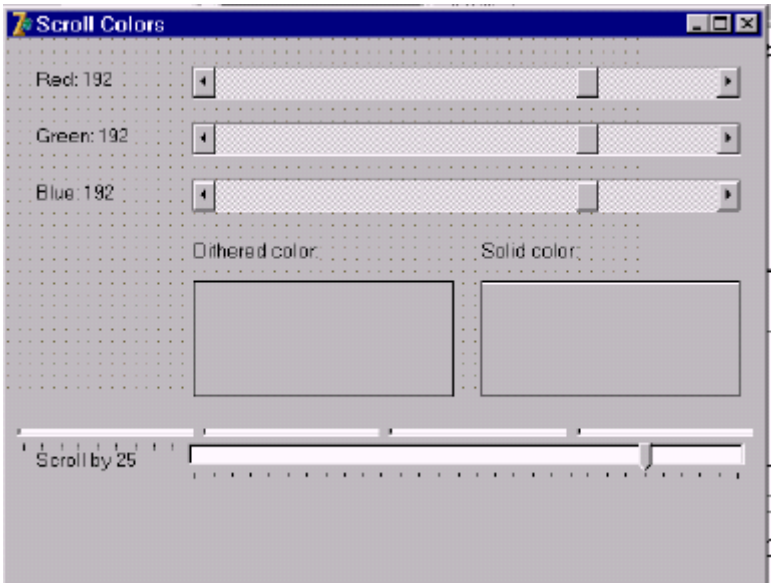
```
procedure TForm1.ButtonAddClick(Sender: TObject);
begin
  with ComboBox1 do
    if (Text <> '') and (Items.IndexOf (Text) < 0) then
      Items.Add (Text);
end;
```

Drgi kombo box možete iskoristiti za eksperimentisanje sa automatskim traženjem: ukucajte neko slovo i biće selektovano prvo ime u listi koje počinje tim slovom. Uz pomoć strelica na tastaturi možete se kretati naviše i naniže u listi. To važi za sve kombo boxove. Treći kombo box je varijanta prvog: novi string se dodaje u listu kada korisnik pritisne tipku Enter.

```
procedure TForm1.ComboBox3KeyPress(Sender: TObject; var Key: Char);
begin
  {if the user presses the enter key}
  if key = chr (13) then
    with ComboBox3 do
      if (Text <> '') and (Items.IndexOf (Text) < 0) then
        Items.Add (Text);
end;
```

3.9. Biranje vrednosti iz opsega

Skrol bar komponente su obično sadržane u nekim drugim komponentama, kao što su Memo polja ili list boxovi i ne može im se direktno prići. Skrol bar se retko koristi sam za sebe, ali se ipak koristi. Tipična upotreba bi bila izbor vrednosti iz opsega.



Scroll Color primer predstavlja formu sa tri skrol barai tri odgovarajuće labele, track bar sa sopstvnom labelom, kao i shape komponente koje će prikazivati boju. Svaki skrol bar se odnosi na jednu od tri osnovne boje: crvenu, zelenu i plavu.

Skrol barovi imaju niz neubičajenih osobina. Min i Max su osobine preko kojih možete postaviti opseg vrednosti, Position sadrži trenutnu poziciju, LargeChange i SmallChange osobine ukazuju na inkrementaciju koja se dobija klikom miša na bar ili na strelicu na kraju bara. Skrol barovi u ovom primeru imaju opseg od 0 do 255. Opseg je određen činjenicom da je boja DWORD sa donja tri bajta koji prikazuju boju (Red, Geen, Blue). Osnovna (default) vrednost je 192 za sva tri skrol bara, jer je to tipična svetlo siva boja za formu i shape komponente.

```
object ScrollBarRed: TScrollBar
   LargeChange = 25
   Max = 255
   Position = 192
   OnScroll = ScrollBarRedScroll
 end
```

Slične osobine ima i TrackBar, koji koristimo za podesavanje LargeChange osobine skrol barova:

```
object TrackBar1: TTrackBar
   Max = 30
   Min = 1
   ParentCtl3D = False
   Position = 25
   OnChange = TrackBar1Change
 end

procedure TFormScroll.TrackBar1Change(Sender: TObject);
begin
  LabelScroll.Caption := 'Scroll by ' + IntToStr(TrackBar1.Position);
  ScrollBarGreen.LargeChange := TrackBar1.Position;
  ScrollBarRed.LargeChange := TrackBar1.Position;
  ScrollBarBlue.LargeChange := TrackBar1.Position;
end;
```

Kada se jedan od skrol barova promeni (OnScroll događaj), program menja odgovarajuću labelu i boju shape komponenti. Prva shape komponenta prikazuje boju koju određuju boje koje korisnik odabere skrol barovima, tako što ih RedGreenBlue vrednosti dodele Color osobini četkice (Brush osobina) koja popunjava površinu komponente, čime dobijamo približnu boju koja je aproksimacija zadate. Ista boja se dodeljuje olovci (Pen osobina) drugog shapea, čime se dobija najpribližnija aproksimacija tražene boje.

U kodu se vidi da postoji i treća shape komponenta koja izigrava ivicu drugog shape. Prava ivica ove komponente je povećana da popunu površinu shapea, korišćenjem veoma široke olovke:

```
procedure TFormScroll.ScrollBarRedScroll(Sender: TObject;
  ScrollCode: TScrollCode; var ScrollPos: Integer);
begin
  LabelRed.Caption := 'Red: ' + IntToStr(ScrollPos);
  Shape1.Brush.Color := RGB (ScrollBarRed.Position,
    ScrollBarGreen.Position, ScrollBarBlue.Position);
  Shape2.Pen.Color := RGB (ScrollBarRed.Position,
    ScrollBarGreen.Position, ScrollBarBlue.Position);
end;
```

Kod ostala dva OnScroll događaja druga dva skrol bara je sličan, jedino treba ispraviti ime labele i njen tekst. OnScroll događaj ima tri argumenta: pošiljaoca, vrstu događaja (ScrollCode može označavati da li korisnik povlači dugme: scTrack, scPosition, scEndScroll, da li je kliknuo na jednu od dve krajnje strelice: scLineUp, scLineDown, ili je kliknuo na bar u jednom od dva pravca: scPageUp, scPageDown, ili pokušava da izađe iz opsega: scTop, scBottom), i krajnju poziciju dugmeta (ScrollPos).

4. Kreiranje i rukovanje menijima

Meniji uglavnom imaju dva nivoa: meni bar koji se nalazi ispod naslova prozora i sadrži imena pull-down menija, od kojih svaki može da sadrži više stavki. Meniji su fleksibilni i omogućavaju postvaljanje stavke menija direktno u meni bar i postavljanje jednog pull-down unutar drugog pull-down menija.

4.1 Struktura glavnog menija

Izbegavajte postavljanje komandi direktno u meni bar, jer korisnici očekuju da mogu da izaberu elemente elemente menija da bi mogli da ispitaju njegovu strukturu, a ne da izdaju komandu. POstavljanje jednog pull-down u drugi pull-down meni je uobičajeno, i Windows vizuelno obaveštava korisnika da se radi o ovoj vrsti menija malim trouglom na desnoj strani menija.

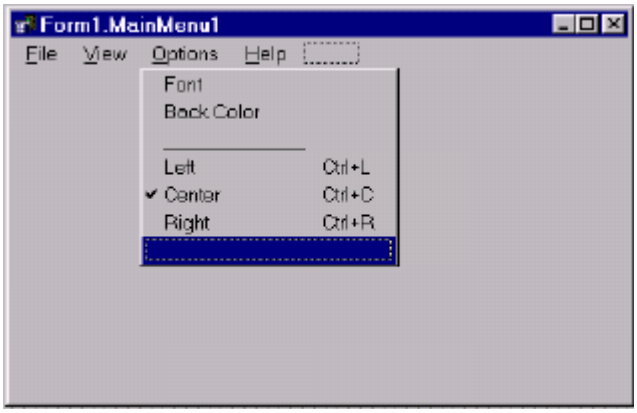
4.1.1. Različite uloge stavki menija

Ako posmatramo stavke menija bez obzira na to gde se nalaze u meniju, primećujemo da postoje tri osnovne vrste stavki u meniju:

- komande izvršavaju akcije i nemaju posebnu oznaku,
- postavljači stanja mogu da uključe i isključe opciju i promene stanje određenog elementa. Ove komande obično imaju znak za štikliranje na levoj strani kada su aktivne. Kada jesu štilirane i kada ih se izabere, dobija se suprotni efekat: deaktiviraju se,
- dijalog stavke menija izazivaju pojavljivanje dijalog boxa. Razlika između ove i drugih stavki menija je ta što korisnik može da isproba moguće posledice dijalog boxa i da čak i odustane izabiranje Cancel dugmeta. Ova vrsta stavki ima oznaku koja se sastoji iz tri tačke.

4.1.2. Kreiranje menija meni dizajnerom

Delfi nudi poseban editor za menije, Menu Designer. Postavite MainManu komponentu na fomru i duplo kliknite na nju. Nemojte brinuti o poziciji menija na fomri: meni će uvek biti isparvno postavljen ispod naslova forme. Meni dizajner omogućava kreiranje menija ukucavanjem teksta komandi , pomeranje stavki ili pull-down menija povlačenjem, kao i jednostavno postavljanje osobina stavki. Takođe omogućava postavljanje komandi direktno u meni bar (ako ne upišete ni jedan element u odgovarajući pull-down meni) i kreiranje drugostepenih pull-down menija izabiranjem Create Submenu komandu meni dizajnerovog lokalnog menija (koji dobijete desnim klikom miša). Meni dizajner omogućava i kreiranje manija prema obrascu.



Meniji slede jednu određenu strukturu: meni počinje File pull-down menijem, koji slede Edit, View, a tada stavke karakteristične za aplikaciju. Poslednji deo uključuje Options, Tools i Window, a uvek se završava Help podmenijem. Svaka od stavki ima standardne elemente, mada oni mogu zavisiti i od aplikacije. File meni, npr, obično sadrži komande kao što su New, Open, Save...

4.1.3. Tasteri za prečice i hotkeys

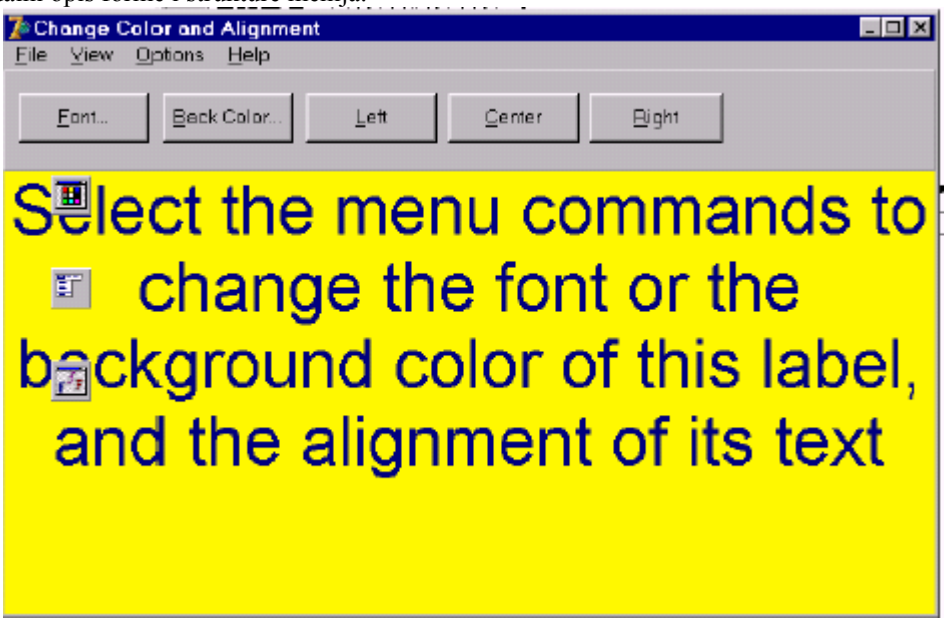
Uobičajena je pojava da stavke menija imaju jedno slovo podvučeno, koje se naziva hotkey. Ovo slovo se koristi za izabiranje stavke menija pomoću tastature, pritiskom na taster Alt i podvučeno slovo. Svaki element menija mora imati drugo slovo podvučeno. Podvučeno slovo se dobija ampersendom - & ispred slova.

Stavke menija mogu imati i tastere za prečicu, kao na slici: Left - Ctrl+L. Tasteri za prečicu se postavljaju izabiranjem odgovarajuće kombinacije tastera u osobini ShortCut. Tsteri za prečicu ne moraju biti vidljivi. Možete napraviti pop-up meni, povezati ga sa formom (postavljanjem PopupMenu osobine forme), postaviti Visible svih elemenata menija na false i dodati elementima tatsre za prečice. Tasteri će operativni, ali se meni neće videti, sem u Help sistemu.

4.2. Odgovor na komande menija

Na formu dodajmo MainMenu komponentu, sa četiri pull-down menija: File pull-down meni sa Exit opcijom, View pull-down sa jednom stavkom: Toolbar, Options meni sa različitim opcijama i Help meni sa About stavkom. Da bi ste dodali separator u Options meni, samo ukucajte _ znake kao tekst komande. Ne dirajte Break osobinu.

Kada kreirate formu, pogledajte listu komponenti koju prikazuje Object Inspector, ili otvorite DFM fajl koji sadrži tekstualni opis forme i strukture menija.



```
 object MainMenu1: TMainMenu
  object File1: TMenuItem
   Caption = '&File'
   object Exit1: TMenuItem
    Caption = 'E&xit'
    OnClick = Exit1Click
   end
  end
  object View1: TMenuItem
   Caption = '&View'
   object Toolbar1: TMenuItem
    Caption = '&Toolbar'
    Checked = True
    OnClick = Toolbar1Click
   end
  end
  object Options1: TMenuItem
   Caption = '&Options'
   object Font1: TMenuItem
    Caption = '&Font...'
    OnClick = Font1Click
   end
   object BackColor1: TMenuItem
    Caption = '&Back Color...'
    OnClick = BackColor1Click
   end
   object N1: TMenuItem
    Caption = '-'
   end
   object Left1: TMenuItem
    Caption = '&Left'
    ShortCut = 16460
    OnClick = Left1Click
   end
   object Center1: TMenuItem
    Caption = '&Center'
    Checked = True
    ShortCut = 16451
    OnClick = Center1Click
   end
   object Right1: TMenuItem
    Caption = '&Right'
    ShortCut = 16466
    OnClick = Right1Click
   end
  end
  object Help1: TMenuItem
   Caption = '&Help'
   object About1: TMenuItem
    Caption = '&About Menu One...'
    OnClick = About1Click
   end
  end
 end
```

OnClick događaj komandi menija za boju pozadine i odgovarajuće dugme toolbara su povezani istim metodom.

```
object BtnBackColor: TButton
  Caption = '&Back Color...'
  OnClick = BackColor1Click
end
```

Komanda menija i dugme koje se odnosi na font su povezani na Font1Click metod, koji prikazuje dijalog za promenu fonta:

```
procedure TFormColorText.Font1Click(Sender: TObject);
begin
  with FontDialog1 do
  begin
    Font := Label1.Font;
    if Execute then
      Label1.Font := Font;
  end;
end;
```

Ostala dugmad i komande menija su povezani sa metodima koji handluju OnClick događaj na isti način. Razlika je u drugom delu Options menija, gde se nalaze (štiklirane) opcije za podešavanje poravnjanja. Svaki put kada korisnik klikne na određeno poravnjanje, potrebno je promeniti poravnjanje teksta, kao i štiklirati to poravnje u meniju (Check osobina stavke menija), a odštiklirati staro poravnanje.

```
procedure TFormColorText.Center1Click(Sender: TObject);
begin
  Label1.Alignment := taCenter;
  Left1.Checked := False;
  Center1.Checked := True;
  Right1.Checked := False;
end;
```

View|Toolbar stavka menija je tipična stavka sa štikliranjem; štiklirana je kada je toolbar vidljiv:

```
procedure TFormColorText.Toolbar1Click(Sender: TObject);
begin
  Panel1.Visible := not Panel1.Visible;
  Toolbar1.Checked := Panel1.Visible;
end;
```

4.3. Promene menija u toku izvršavanja

Stavke menija mogu se menjati tokom izvršavanja: kada komanda ne može, ili ne sme biti aktivirana, siva je i korisnik je ne može pokrenuti. Onesposobljavanje pojedinih komandi stavlja korisniku na znanje koje su mu komande trenutno na raspolaganju. Promenu menija tokom izvršenja sa štikliranjem smo videli u prethodnom primeru.

Kada korisnik može da izabere više od dve opcije, bolje je koristiti više stavki sa štikliranjem, ili, još bolje, radio dugme stavku. Tri osobine kojima se najčešće menja stavka tokom izvršenja:

- Check osobina da bi se dodalo ili uklonilo štikliranje,
- Enable osobina kojom se stavka može posiveti i onesposobiti,
- Caption osobina koja nosi natpis stavke, koja može biti promenjena da prikazuje stvaran efekat komande.

Dodajmo na prethodni primer nove stavke u View meni (Hide Label i Fixed Font) i dve stavke u Options meni (Fixed View i Disable Help). Stavka Hide Label prikazuje promenu Caption osobine u skladu sa statusom aplikacije. OnClick handler događaja za stavku:

```
procedure TFormColorText.HideLabel1Click(Sender: TObject);
begin
  Label1.Visible := not Label1.Visible;
  if Label1.Visible then
    HideLabel1.Caption := 'Hide &Label'
  else
    HideLabel1.Caption := 'Show &Label'
end;
```

Druge tri dodate stavke menija onemogućavaju ili sakrivaju stavke menija ili pull-down menija. Komanda Fixed Font Veiw menija onemogućuje Font komandu Options menija:

```
procedure TFormColorText.FixedFont1Click(Sender: TObject);
begin
  ToggleCheck (FixedFont1);
  Font1.Enabled := not Font1.Enabled;
end;


procedure ToggleCheck (Item: TMenuItem);
begin
  Item.Checked := not Item.Checked;
end;
```

Menije ne bi trebalo činiti nevidljivima, jer će to samo zbuniti korisnika. Bolje rešenje je onesposobiti ih, kao u sledećem kodu:

```
procedure TFormColorText.FixedView1Click(Sender: TObject);
begin
  ToggleCheck (FixedView1);
  View1.Visible := not View1.Visible;
end;

procedure TFormColorText.DisableHelp1Click(Sender: TObject);
begin
  ToggleCheck (DisableHelp1);
  Help1.Enabled := not Help1.Enabled;
end;
```

5. Multimedijalna zabava

5.1 Osnovni zvuci Windowsa

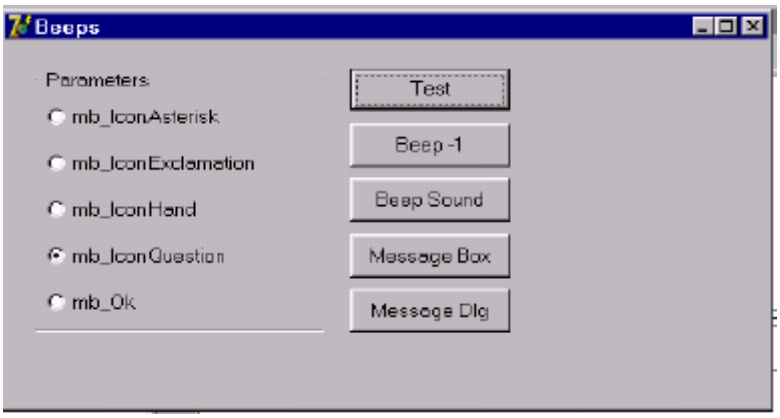
Delfi ima sistemsku proceduru koja generiše zvuk – Beep:

```
procedure Beep;
begin
  MessageBeep(0);
end;
```

Vrednost 0 i –1 kao argumenti rezultuju zvukom internog zvučnika računara. Sa drugim vrednostima kao argumentima MessageBeep proizvodi zvuk pomoću zvučne kartice:

mb_IconAsterisk	SystemAsterisk zvuk
mb_IconExclamation	SystemExclamation sound
mb_IconHand	SystemHand zvuk
mb_IconQuestion	SystemQuestion zvuk
mb_Ok	SystemDefault zvuk

Ove konstante su moguće vrednosti MessageBox funkcije sadržane u MessageBox metodu TApplication klase. Uobičajeno je proizvesti zvuk kada se pojavi poruka, ali MessageDlg nema ovu mogućnost. Rešenje je napraviti svoju funkciju za prikazivanje poruke, koja generiše i zvuk.



Forma Beeps ima nekoliko rdaio dugmadi kojima korisnik može da odabere jednu od 5 konstanti MessageBeep funkcije.

```
object RadioGroup1: TRadioGroup
   Caption = 'Parameters'
   ItemIndex = 0
   Items.Strings = (
     'mb_IconAsterisk'
     'mb_IconExclamation'
     'mb_IconHand'
     'mb_IconQuestion'
     'mb_Ok')
end
```

Program proizvodi zvuk u odnosu na aktuelnu selekciju kada korisnik klikne na Beep Sound dugme. Dugmetov handler za OnClick metod najpre određuje koje je radio dugme selektovano, a zatim proizvodi odgovarajući zvuk.

```
procedure TForm1.BeepButtonClick(Sender: TObject);
var
  Flag: Cardinal;
begin
  case RadioGroup1.ItemIndex of
    0: Flag := mb_IconAsterisk;
    1: Flag := mb_IconExclamation;
    2: Flag := mb_IconHand;
    3: Flag := mb_IconQuestion;
    4: Flag := mb_Ok;
  else
    Flag := 0;
  end;
  MessageBeep (Flag);
end;
```

Else grana služi za sprečavanje kompajlera da izda upozorenje. Za poređenje selektovanog zvuka i default beep zvuka treba kliknuti na Beep-1 dugme.

```
procedure TForm1.BeepOneButtonClick(Sender: TObject);
begin
  MessageBeep (Cardinal (-1));
end;
```

Za testiranje da li je dajver za zvuk instaliran (sa ili bez zvučne kartice, pošto je moguće imati zvuk preko PC zvučnika), kliknuti na dugme Test, koje koristi multimedijalnu funkciju WaveOutGetNumDevs koja izvodi test.

```
procedure TForm1.TestButtonClick(Sender: TObject);
begin
```

```
  if WaveOutGetNumDevs > 0 then
    SoundMessageDlg ('Sound is supported',
      mtInformation, [mbOk], 0)
  else
    SoundMessageDlg ('Sound is NOT supported',
      mtError, [mbOk], 0);
end;
```

Da bi se ova funkcija mogla kompajlirati, potrebno je dodati MmSystem unit u uses klauzulu. Ukoliko računar nema instaliran drajver za zvuk, čuće se samo standardni zvuci, bez obzira koji je zvuk selektovan. Poslednja dva dugmeta imaju sličnu funkciju: prikazuju poruku i genrišu odgovarajući zvuk.

OnClick handler događaja komponente MessageBox koristi standardni Windows pristup. Poziva MessageBeep funkciju , a zatim MessageBox metod Application objekta, što za posledicu ima generisanje zvuka u trenutku pojave poruke.

```
procedure TForm1.BoxButtonClick(Sender: TObject);
var
  Flag: Cardinal;
begin
  case RadioGroup1.ItemIndex of
    0: Flag := mb_IconAsterisk;
    1: Flag := mb_IconExclamation;
    2: Flag := mb_IconHand;
    3: Flag := mb_IconQuestion;
  else {including 4:}
    Flag := mb_Ok;
  end;
  MessageBeep (Flag);
  Application.MessageBox (
   PChar (RadioGroup1.Items [RadioGroup1.ItemIndex]), 'Sound', Flag);
end;
```

Pritiskom na poslednje dugme, program poziva SounMessageDlg funkciju, koja nije interna Delfijeva funkcija. Ona je dodata u program i generiše zvuk, koji određuje AType parametar, a zatim prikazuje Delfijevu standardnu poruku.

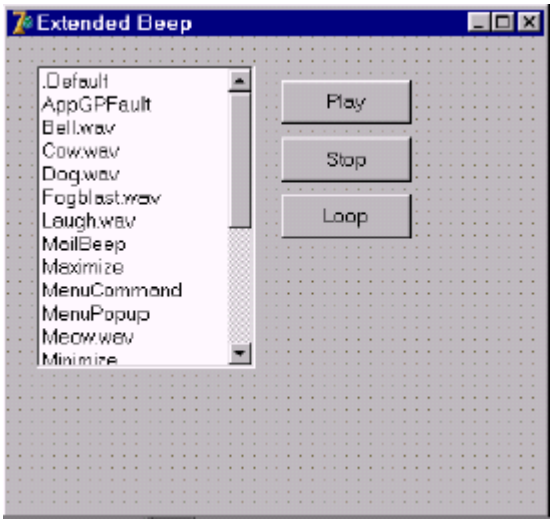
```
function SoundMessageDlg (const Msg: string;
  AType: TMsgDlgType; AButtons: TMsgDlgButtons;
  HelpCtx: Longint): Integer;
var
  Flag: Cardinal;
begin
  case AType of
    mtWarning: Flag := mb_IconExclamation;
    mtError: Flag := mb_IconHand;
    mtInformation: Flag := mb_IconAsterisk;
    mtConfirmation: Flag := mb_IconQuestion;
  else
    Flag := mb_Ok;
  end;
  MessageBeep(Flag);
  Result := MessageDlg (Msg, AType,
    AButtons, HelpCtx);
end;

procedure TForm1.MessDlgButtonClick(Sender: TObject);
var
  DlgType: TMsgDlgType;
begin
  case RadioGroup1.ItemIndex of
    0: DlgType := mtInformation;
```

```
    1: DlgType := mtWarning;
    2: DlgType := mtError;
    3: DlgType := mtConfirmation;
  else {including 4:}
    DlgType := mtCustom;
  end;
  SoundMessageDlg (
    RadioGroup1.Items [RadioGroup1.ItemIndex],
    DlgType, [mbOK], 0);
end;
```

5.1. Od Beepova do muzike

Ako želite muziku, možete koristiti Windows API funkciju PlaySound za bilo koji wave fajl.



List box forme sadrži imena nekih sistemskih i WAV fajlova koji su dostupni u tekućem direktorijumu (onom koji sadrži ExtBeep.exe). Kada korisnik klilkne na Play dugme, poziva se PlaySound funkcija.

```
procedure TForm1.PlayButtonClick(Sender: TObject);
begin
  PlaySound (PChar (Listbox1.Items [ListBox1.ItemIndex]), 0, snd_Async);
end;
```

Prvi parametar je ime zvuka – sistemskog, ime WAV fajla ili određeni izvor zvuka (pogledati Win32 API Help za detalje). Drugi argument određuje gde tražiti sistemski zvuk, a treći sadrži niz flegova, koji u ovom slučaju kažu da funkcija treba odmah da se vrati, a da se zvuk generiše asinhrono (suprotno se postiže snd_Sync). Uz asinhrono generisanje zvuka, dugačak zvuk može se prekinuti pozivom PlaySound funkcije, ali sa prvim parametrom nil, što se vidi iz OnClick handlera za Stop dugme.

```
procedure TForm1.StopButtonClick(Sender: TObject);
begin
  PlaySound (nil, 0, 0);
end;
```

Stop dugme je posebno korisno za prekidanje neprekidnog zvuka koji pokreće Loop dugme pozivom PlaySound metoda kome je poslednji parametar snd_Async ili snd_Loop. Metod FormCreate selektuje prvi element list boxa prilikom startupa, postavljajući ItemIndex na 0. Na ovaj način sprečena je greška koja se može javiti tokom izvršavanja, da korisnik pritisne dugme pre nego izabere neki element list boxa.

5.2. Media Player komponenta

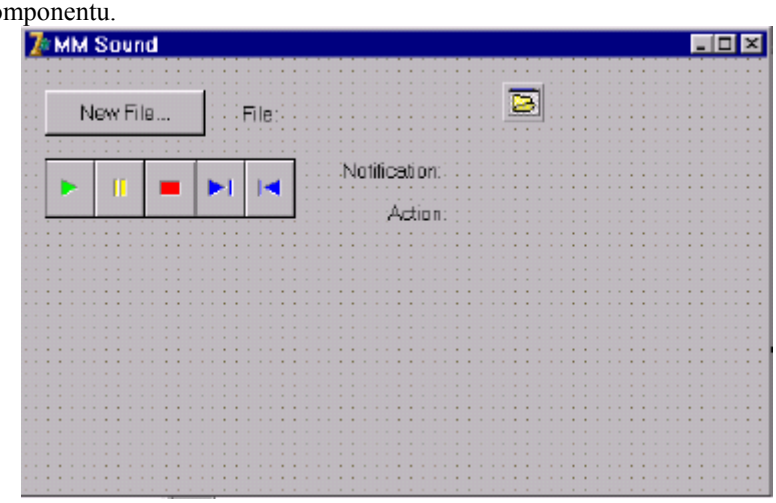
Delfijeva klasa TMediaPlayer učauruje veliki deo funkcionalnosti Windows Media Control Interface (MCI), interfejs visokog nivoa za kontrolisanje internih i eksternih uređaja koji se odnose na medije. Najvažnija osobina

TMediaPlayer komponente je DeviceType, koji može imati sledeće vrednosti: dtAutoSelect, što znači da vrsta uređaja zavisi od ekstenzije fajla (osobina FileName), vrednost koja označava određeni uređaj, npr dtAVIVideo, dtCDAudio, dtWaveAudio itd.

Kada je određena vrsta uređaja i fajl, taj uređaj se otvara (ili postavljanjem AutoOpen osobine na True) i kontrole MediaPlayer komponente se aktiviraju. Komponenta ima niz dugmadi i ne odgovaraju sva svim vrstama fajlova. Postoje tri osobine koje se odnose na dugmad: VisibleButtons (odlučuje koja su dugmad prisutna na komponenti), EnabledButtons (postavlja aktivnu dugmad) i ColoredButtons (koja dugmad imaju obojene oznake). Događaj OnClick je neuobičajen, jer sadrži parametar koji određuje koje je dugme pritisnuto, a drugi parametar se koristi za onesposobljavanje osnovne funkcije dugmeta. OnNotify govori komponenti da li je događaj koji je generisalo dugme bio uspešan. OnPostClick je poslat pri pokretanju i završetku rada metoda, u zavisnosti od Wait osobine, koja određuje da li je akcija uređaja sinhronizovana ili ne.

5.2.1. Pokretanje muzičkih fajlova

Forma MmSound sadrži labele koje opisuju trenutno stanje, dugme kojim se bira novi fajl, OpenDialog i MediaPlayer komponentu.




```
object MediaPlayer1: TMediaPlayer
    VisibleButtons = [btPlay, btPause, btStop, btNext, btPrev]
    OnClick = MediaPlayer1Click
    OnNotify = MediaPlayer1Notify
  end
```

Kada korisnik otvori novi fajl, wave tabelu ili MIDI fajl, program osopsobljava MediaPlayer, pa možete pustiti muziku i koristiti drugu dugmad.

```
procedure TForm1.NewButtonClick(Sender: TObject);
begin
  if OpenDialog1.Execute then
  begin
    FileLabel.Caption := OpenDialog1.Filename;
    MediaPlayer1.Filename := OpenDialog1.Filename;
    MediaPlayer1.Open;
    MediaPlayer1.Notify := True;
  end;
end;
```

Pošto je osobina Notify postavljena na True u MediaPlayer komponenti, MediaPlayer će pokrenuti određeni handler događajima koji će ispisati informaciju u labelu.

```
procedure TForm1.MediaPlayer1Notify(Sender: TObject);
begin
  case MediaPlayer1.NotifyValue of
    nvSuccessful : NotifLabel.Caption := 'Success';
    nvSuperseded : NotifLabel.Caption := 'Superseded';
```

```
    nvAborted    : NotifLabel.Caption := 'Aborted';
    nvFailure    : NotifLabel.Caption := 'Failure';
  end;
  MediaPlayer1.Notify := True;
end;
```

Osobu Notify treba postaviti na True svaki put kada je pozvan OnNotify handler pozvan, da bi primao i sva dalja obaveštenja. Druga labela prikazuje izdatu naredbu.

```
procedure TForm1.MediaPlayer1Click(Sender: TObject; Button: TMPBtnType;
  var DoDefault: Boolean);
begin
  case Button of
    btPlay: ActionLabel.Caption := 'Playing';
    btPause: ActionLabel.Caption := 'Paused';
    btStop: ActionLabel.Caption := 'Stopped';
    btNext: ActionLabel.Caption := 'Next';
    btPrev: ActionLabel.Caption := 'Previous';
  end;
end;
```

5.2.2. Video

Na računaru postoji video uređaj (ne u smislu kućnog videa, naravno), ali za prikazivanje video fajlova (AVI), potreban je određeni drajver, kojom se pristupa direktno iz Windowsa. Ukoliko računar može da prikazuje video fajlove, omogućiti Delfijevoj aplikaciji isto je jednostavno: postavljanjem MediaPlayer komponente na formu, izabiranjem AVI vrednosti u FileName osobini, postavljanjem AutoOpen osbine na True i pokretanjem programa.

Umesto prikazivanje fajla u svom prozoru, postavimo panel na formu, a ime panela postavimo za osobinu MediaPlayerove osobine Display. Drugo rešenje je određivanje koliki deo prozora video treba da pokriva uz pomoć Display i DisplayRect osobina. Iako je moguće napraviti sličan program ne pišući kod, na taj način bi programer morao tačno da zna koji AVI fajlovi se nalaze na korisnikovom računaru, kao što bi morao da navede punu putanju jednog od fajlova u FileName osobini MediaPlyer komponente. Dole naved kod otvara i počinje prikazivanje fajla automatski. Korisnik samo treba da klikne na panel.

```
procedure TForm1.Panel1Click(Sender: TObject);
begin
  if OpenDialog1.Execute then
  begin
    MediaPlayer1.FileName := OpenDialog1.Filename;
    MediaPlayer1.Open;
    MediaPlayer1.Perform (wm_LButtonDown, 0, $00090009);
    MediaPlayer1.Perform (wm_LButtonUp, 0, $00090009);
  end;
end;
```

Nakon otvaranja MediaPlayera, moguće je odmah pozvati Play metod, ali tada dugmad neće biti adekvatno onesposobljena. Zato je u ovom primeru simuliran klin na poziciji 9 x i y koordinate MediaPlayer prozora (u gornjem primeru koriste se heksadecimalne vrednosti apcise i ordinate). Za prevenciju grešaka onesposobite svu dugmad još u vreme dizajniranja, sve dok se ne desi simulirani klik. Ova aplikacija automatski zatvara MediaPlayer kada se zatvori aplikacija (OnClose handler).

5.2.3. Video u formi

MediaPlayer ima određena ograničenja koja se tiču prozora koji može da koristi za svoj izlaz. Mogu se koristiti mnoge komponente, ali ne sve. MediaPlayer može biti prozor u kom će se prikazivati video, ali ova komponenta ne može biti poravnata, niti joj može biti promenjena velličina. Ukoliko pokušate da koristite veliku dugmad, ona će u trenutku izvršenja biti smanjena. Pritiskom na Pause dugme, ostala dugamd ostaju i dalje skrivena.

Ne možete jednostavno prikazati video u formi, jer u vreme dizajnniranja MediaPlayer ne prihvata formu kao vrednost osobine Display. Međutim, moguće je postaviti formu kao vrendost Display tokom izvršavanja. Postavite skriveni MediaPlayer (Visible je False) i OpenDialog na formu. Izaberita naslov forme i omogućite ShowHints osobinu, i napišite ovaj kod:

```
procedure TForm1.FormClick(Sender: TObject);
begin
  if MediaPlayer1.FileName = '' then
    if OpenDialog1.Execute then
    begin
      MediaPlayer1.FileName :=
        OpenDialog1.FileName;
      MediaPlayer1.Open;
      Playing := False;
    end
    else
      exit; // stop if no file is selected

  if Playing then
  begin
    MediaPlayer1.Stop;
    Playing := False;
    Caption := 'MM Video (Stopped)';
    Hint := 'Click to play video';
  end
  else
  begin
    MediaPlayer1.Display := self;
    MediaPlayer1.DisplayRect := ClientRect;
    MediaPlayer1.Play;
    Playing := True;
    Caption := 'MMV (Playing)';
    Hint := 'Click to stop video';
  end;
end;
```

Playing je privatno polje forme. Program prikazuje video u čtavom klijntskom delu forme. Ukoliko se formi menja veličina, povećajte izlazni pravougaonik:

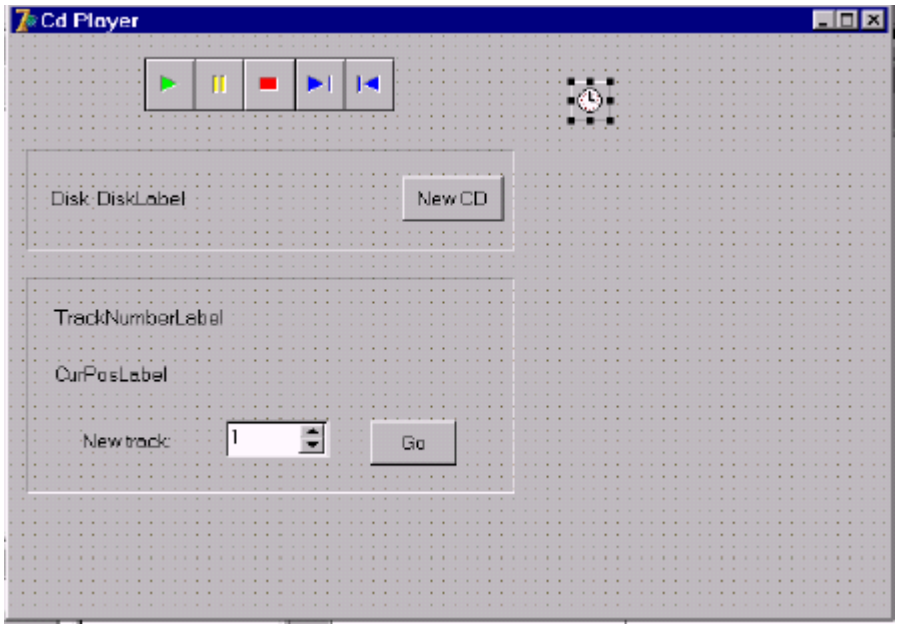
```
procedure TForm1.FormResize(Sender: TObject);
begin
  MediaPlayer1.DisplayRect := ClientRect;
end;
```

MediaPlayer će se zaustaviti kada stigne do kraja fajla, ili u sličaju greške. U oba slučaja, aplikacija prima obaveštenje.

```
procedure TForm1.MediaPlayer1Notify(Sender: TObject);
begin
  Playing := False;
  Caption := 'MMV (Stopped)';
  Hint := 'Click to play video';
end;
```

5.3. Rad sa CD drajvom

Pored rada sa videom i zvukom, MCI se koristi za rad sa spoljašnjim uređajima uopšte. Ukoliko uzmemo za primer CD-ROM drajv, u aplikacijama koje rukuje CDom, dovoljno je angažovati MediaPlayer komponentu čija je osobina DeviceType postavljena na CDAudio, uz uslov da nijedan fajl nije selektovan FileName osobinom.



Forma ima nekoliko dugmadi, labele koje pokazuju trenutno stanje, tajmer i SpinEdit komponentu koja omogućava biranje snimaka sa diska. Labele pokazuju broj snimaka na disku, tekući snimak, poziciju u okviru tekućeg snimka, dužinu snimka i praćenje vremena uz pomoć tajmera. MediaPlayer ima osobinu TimeFormat čija vrednost se može postaviti na tfMSF (Track, Minute, Second, Frame), pristupiti osobinama koje opisuju pozicioniranje (Position i Length). Dobijanje vrednosti nije teško, ako koristite odgovarajuću funkciju

```
CurrentTrack := Mci_TMSF_Track (MediaPLayer.Position);
```

Funkcije koje izračunavaju vrednosti za čitav disk i za trenutnu traku:

```
procedure TForm1.CheckDisk;
var
  NTracks, NLen: Integer;
begin
  NTracks := MediaPlayer1.Tracks;
  NLen := MediaPlayer1.Length;
  DiskLabel.Caption := Format (
    'Tracks: %.2d, Length:%.2d:%.2d',
    [NTracks, Mci_TMSF_Minute (NLen),
    Mci_TMSF_Second (NLen)]);
  SpinEdit1.MaxValue := NTracks;
end;

procedure TForm1.CheckPosition;
var
  CurrentTrack, CurrentPos, TrackLen: Integer;
begin
  CurrentPos := MediaPlayer1.Position;
  CurPosLabel.Caption := Format (
    'Position: %.2d:%.2d',
    [Mci_TMSF_Minute (CurrentPos),
    Mci_TMSF_Second (CurrentPos)]);
  CurrentTrack := Mci_TMSF_Track (CurrentPos);
  TrackLen := MediaPlayer1.TrackLength [CurrentTrack];
  TrackNumberLabel.Caption := Format (
    'Current track: %.2d, Length:%.2d:%.2d',
    [CurrentTrack, Mci_MSF_Minute (TrackLen),
    Mci_MSF_Second (TrackLen)]);
end;
```

Globalne vrednosti za disk se izračunavaju samo pri pokretanju i kada je pritisnuto New CD dugme:

```
procedure TForm1.FormCreate(Sender: TObject);
begin
  MediaPlayer1.TimeFormat := tfTMSF;
  MediaPlayer1.Open;
  CheckDisk;
  CheckPosition;
end;

procedure TForm1.NewButtonClick(Sender: TObject);
begin
  CheckDisk;
  CheckPosition;
end;
```

vrednosti za tekući snimak i poziciju se izračunavaju svaki put kada prođe interval tajmera, pozivom CheckPosition metoda. Ovaj pristup ima manu: ukoliko želite da slušate CD i koristite neke druge programe, tajmer koji pristupa MediaPlayeru će značajno usporiti sistem. pored toga što obaveštava korisnika šta se događa, forma ima MediaPlayer komponentu koja omogućava korisniku da pokrene i zaustavi izvođenje snimka, promeni snikamk, itd. Operacije na ovoj komponenti zaustavljaju tajmer.

```
procedure TForm1.MediaPlayer1PostClick(Sender: TObject;
  Button: TMPBtnType);
begin
  if MediaPlayer1.Mode = mpPlaying then
    Timer1.Enabled := True
  else
    Timer1.Enabled := False;
  CheckPosition;
end;
```

Go dugmetom može se preći na snimak koji je izabran u SpinEdit komponenti, a MaxEdit osobinu postavlja ChekDisk metod:

```
procedure TForm1.GoButtonClick(Sender: TObject);
var
  Playing: Boolean;
begin
  Playing := (MediaPlayer1.Mode = mpPlaying);
  if Playing then
    MediaPlayer1.Stop;
  MediaPlayer1.Position :=
    MediaPlayer1.TrackPosition[SpinEdit1.Value];
  CheckPosition;
  if Playing then
    MediaPlayer1.Play;
end;
```