


### 3.3.1. Menük használata

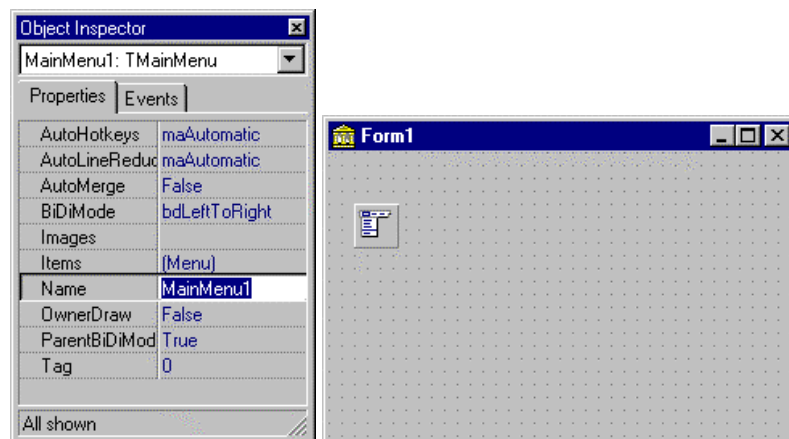
1. Menü tervezésének menete: [menu0](#)
2. A menu0 módosítása újabb menüponttal: [menu1](#)
3. Menüpont kijelölése: [menu2](#)
4. Menüpont változtatása: [menu3](#)
5. Választógombként működő menüpont: [menu4](#)
6. Legördülő menü tervezése: [popup0](#)
7. Menü és popup menü együttes alkalmazása: [popup1](#)
8. Fontválasztás felbukkanó menüből [font](#)




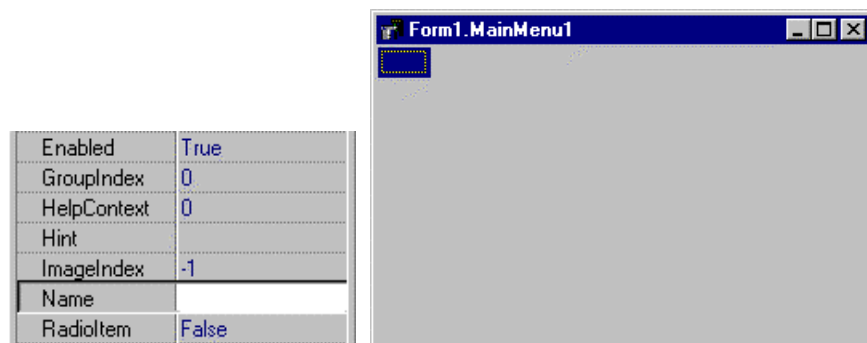
Tervezzünk egy olyan menürendszert, melynek főmenüje a *Menü*, legördülő menüjének két eleme a "Szöveg megjelenítése" és a "Kilépés". A "Szöveg megjelenítése" menüponthoz kételemű almenü tartozik: a "Kisbetűs szöveg" menüpont az <F1> gyorsítóbillentyűvel, a "Nagybetűs szöveg" menüpont pedig az <F2> gyorsítóbillentyűvel! (*menu0*)

Ez a gyakorlat részletesen bemutatja a menürendszer tervezését!

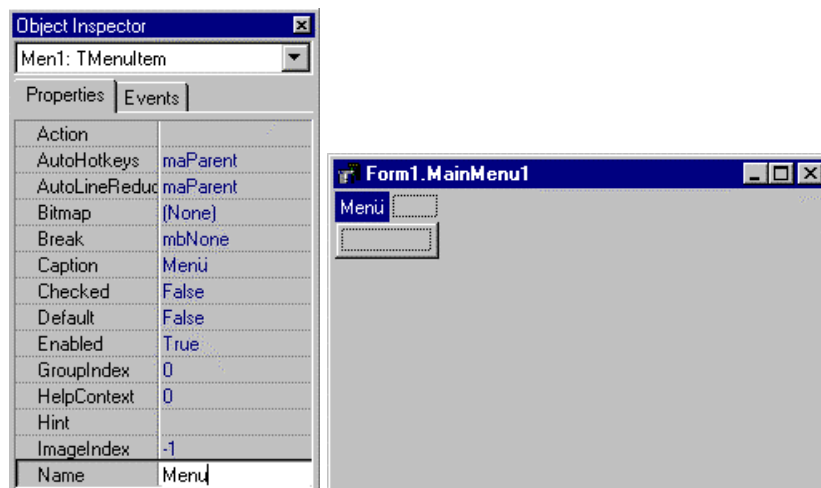
Először kattintsunk a *MainMenu*  ikonon, majd az úrlapon! Az objektum-felügyelő ablakban az objektum a *MainMenu1* nevet kapja, amely a *TMainMenu* osztály objektumpéldánya.



Kétszer kattintva az úrlap  ikonján, a menütervező egy üres menüelemet tartalmazó formot jelenít meg és az objektum-felügyelőben aktívá teszi a *Name* mezőt.

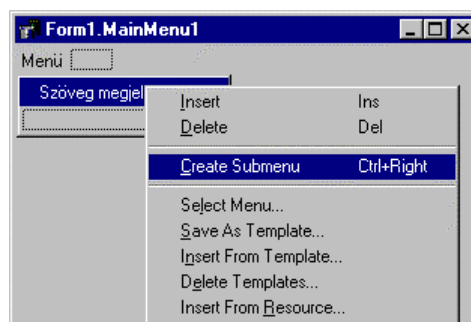


Ezután kitöltjük az objektum-felügyelő *Properties* oldalán a *Name* tulajdonságot. Legyen a menüazonosító *Menu*, a program futása közben ez azonosítja a kiválasztott menüpontot. A *Caption* tulajdonság értéke *Menü*, amely a menüpontban azonnal megjelenik. Ha a menüpont nevében egy & jelet gépelünk bármelyik betű elé, akkor az a betű aláhúzva jelenik meg és a menüpont a billentyűzetről is kiválasztható lesz. Ha nem szerepel a menüpont nevében & jel, akkor az alkalmazás futásakor a menüpontban az első betű vagy az a betű kerül aláhúzásra, amely addig nem szerepelt.

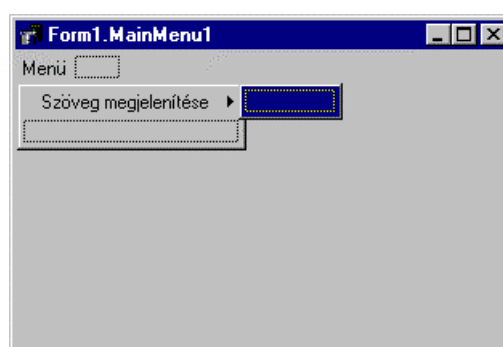



A kitöltés után a *Menü* a főmenü első menüje lesz. A *Menü* menüpontra kattintva megjelenik egy üres hely az első legördülő menü számára.

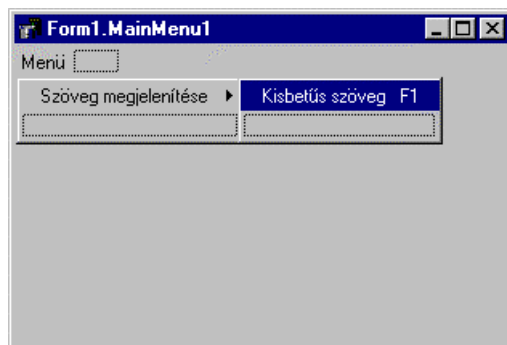
Hasonlóan a főmenühöz, itt is először a **Name** tulajdonság mellett megadjuk a "*Megjelenítés*" azonosítót, majd a **Caption** tulajdonság értékét "*Szöveg megjelenítése*"-re állítjuk. Ha a "*Szöveg megjelenítése*" is láthatóvá válik a menüben, megtervezhetjük az "*Szöveg megjelenítése*" "*Kisbetűs szöveg*" legördülő menüjét. Először kijelöljük a "*Szöveg megjelenítése*" menüpontra a bal egérgombbal kattintva, majd a jobb egérgombbal való kattintás után megjelenik a menütervező felbukkanó menüje.



A gyorsmenü **Create Submenu** menüpontját kiválasztva jön létre az almenü.



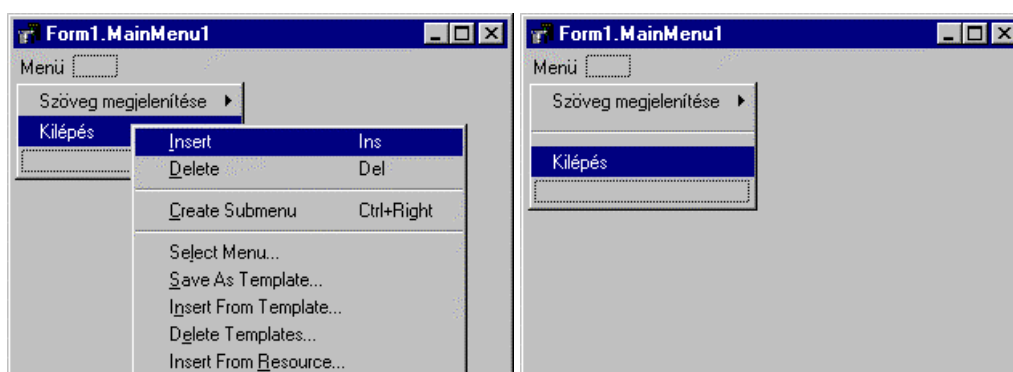
A kijelölés után az objektum-felügyelőben kitölthetjük az almenü elemekre vonatkozó (**Caption** "*Kisbetűs szöveg*" és **Name** "*Kisbetűs*") tulajdonságokat. A kijelölt menüponthoz az <F1> gyorsítóbillentyűt rendeljük. A gyorsítóbillentyű menühöz való rendelése a **Shortcut** tulajdonság  gombján való kattintás után legördülő listaablakból történik. A kiválasztott <F1> billentyű a menüpont mellett meg is jelenik.



A "Kisbetűs szöveg" alatti üres menüponton kattintva kijelöljük a következő menüpontot, és az objektum-felügyelőben itt is megadjuk a **Caption** és a **Name** tulajdonságokat. Ehhez a menüponthoz az <F2> gyorsítóbillentyűt rendeljük.

Az utolsó, **Kilépés** menüpont megtervezése következik, ahol a **Caption** tulajdonság "Kilépés", a **Name** pedig "Kilepes" lesz. Ehhez a menüponthoz nem rendelünk billentyűkombinációt, mivel a felhasználó az <Alt+F4> lenyomásával is lezárhatja az alkalmazás ablakát.

Csinosítsuk tovább a menüt! A "Szöveg megjelenítése" és "Kilépés" menüpontokat válasszuk el tagoló vonallal egymástól. Ez a következőképpen történik. A "Szöveg megjelenítése" menüpontra ráállva az egér jobb gombjával előjövő gyorsmenüből válasszuk ki az **Insert (Beszúrás)** menüpontot. Ezt követve írjunk – (mínusz) jelet az üres menühelyhez tartozó **Caption** mezőbe, a **Name** ekkor automatikusan **N1** lesz. Ennek hatására egy vonal fogja a két tétel elválasztani egymástól.



Még mielőtt megírnánk a program kódját, az ablak *Form1* fejlécét írjuk át "Menü tervezése"-re! Ehhez az objektum-felügyelő legördülő listaablakából válasszuk ki a *Form1* tételt, majd ezt követően írjuk át a **Caption** tulajdonságát a "Menü tervezése" szövegre!

Ha az űrlap felületén kétszer kattintunk, akkor a kódszerkesztő ablakban a *Unit1.pas* fájlban létrejön *FormCreate* eseménykezelő eljárás váza, amelyben megadjuk, hogy a futó alkalmazás ablaka hol jelenjen meg a képernyőn és mekkora legyen a magassága és szélessége (ezt a **Left**, **Top**, **Height** és **Width** tulajdonságoknál tehetjük meg):

```
procedure TForm1.FormCreate(Sender: TObject);
begin
    // az ablak bal felső sarkának koordinátái
    Left   := 200;
    Top    := 100;
    // az ablak magassága és szélessége
    Height := 240;
    Width  := 380;
end;
```

Először a "Kilépés" menüponthoz tartozó eseménykezelő eljárást írjuk meg. Kattintsuk kétszer a "Kilépés" menüpontra, melynek hatására létrejön a *KilepesClick* eseménykezelő eljárás váza! A program futása ennek

a menüpontnak a kiválasztásával fejeződik be, ezért az *Application* objektum a ***Terminate*** metódusát aktivizáljuk.

```
procedure TForm1.KilepesClick(Sender: TObject);  
begin  
    Application.Terminate;  
end;
```

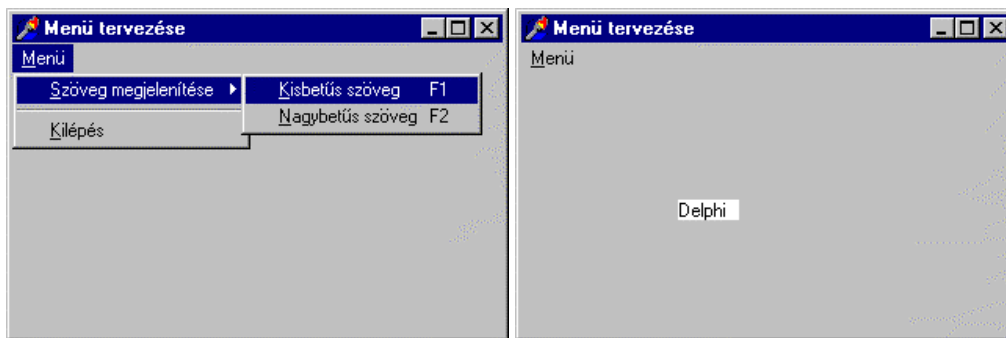
Írjuk meg a "*Kisbetűs szöveg*" menüpont eseménykezelő eljárását! A menüpontra kétszer kattintva megjelenik a *KisbetusClick* eljárás, ahol a ***TextOut*** metódussal kiírjuk a kisbetűs szöveget a (100,80) koordinátájú pozícióban.

```
procedure TForm1.KisbetusClick(Sender: TObject);  
begin  
    Canvas.TextOut(100,80,'Delphi  ');  
end;
```

Hasonlóan megírjuk a "*Nagybetűs szöveg*" menüpont *NagybetusClick* eseménykezelő eljárását. A szöveget nagybetűkkel az (100,80) pozícióban jelenítjük meg.

```
procedure TForm1.NagybetusClick(Sender: TObject);  
begin  
    Canvas.TextOut(100,80,'DELPHI ');  
end;
```

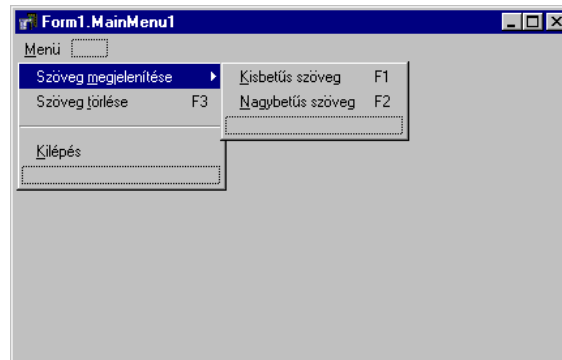
A program egy futási képe:



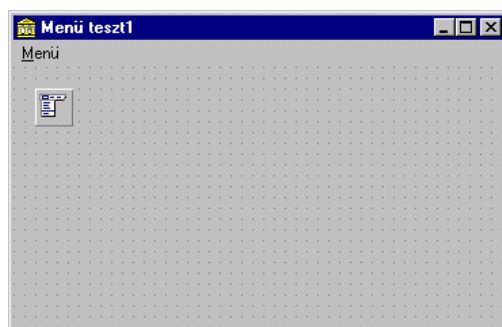


Készítsünk olyan menüvezérelt programot, amely színes szöveget jelenít meg, illetve töröl. A feladatot a MENU0 program módosításával oldjuk meg! (*menu1*)

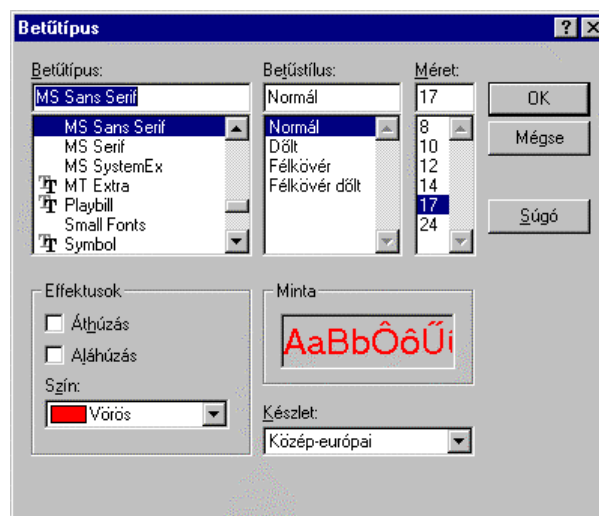
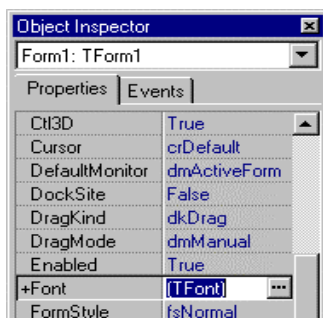
Az alkalmazás menürendszere hasonló, mint a MENU0 programnál volt, azonban a szöveg törlésére vonatkozó menüpont is megjelenik:



A szöveg megjelenítése



A megjelenítendő szöveg színét és a karaktereinek nagyságát beállíthatjuk a Form1 **Font** tulajdonságán kattintva a *Betűtípus* párbeszédablakban. A betű méretét 17 pontnagyságra és a színét pedig *Vörösre* állítjuk.



A *TForm1* osztálya:

```
type
  TForm1 = class(TForm)
    MainMenu1: TMainMenu;
    Menu: TMenuItem;
    Megjelenites: TMenuItem;
    Kisbetus: TMenuItem;
    Nagybetus: TMenuItem;
    Torles: TMenuItem;
    N1: TMenuItem;
    Kilepes: TMenuItem;
    procedure KisbetusClick(Sender: TObject);
    procedure NagybetusClick(Sender: TObject);
    procedure TorlesClick(Sender: TObject);
    procedure KilepesClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);
  end;
```

Eseménykezelő eljárások:

Ha futás közben szeretnénk a szöveg megjelenését szabályozni, akkor azt a *FormCreate* eseménykezelő eljárásban is megtehetjük. A megjelenítendő szöveg karaktereinek színét vörösre (*clRed*) és nagyságát (*17*), valamint a szöveg háttérszínét sárgára (*clYellow*). A szöveg hátterének a fehér alapértelmezés szerinti színe.

```
procedure TForm1.FormCreate(Sender: TObject);
begin
  // az ablak bal felső sarkának és méretének alapértéke
  Height := 240;
  Left := 200;
  Top := 100;
  Width := 380;
  // a szöveg háttérszíne
  Canvas.Brush.Color := clYellow;
  // a szöveg színe
  Canvas.Font.Color := clRed;
  // a szöveg karaktermagassága
  Canvas.Font.Size := 17;
end;
```

A "Szöveg megjelenítése|Kisbetűs szöveg" menüpont kiválasztásakor meghívódó *KisbetusClick*, valamint a "Szöveg megjelenítése|Nagybetűs szöveg" menüpont kiválasztásakor meghívódó *NagybetusClick* eseménykezelő eljárás nem változik MENU0 feladatnál ismertettekhez képest.

A szöveg törlése a "Szöveg törlése" menüpont kiválasztásakor meghívódó *TorlesClick* eseménykezelő eljárásban, az *Invalidate* metódus hívásával történik. Ilyenkor egy üres *OnPaint* eseménykezelő eljárás hívódik meg, amely a frissítés hiányában törli a form területét.

```
procedure TForm1.TorlesClick(Sender: TObject);
begin
  Invalidate;
end;
```



Módosítsuk a MENU1 programot! Bővítsük a menüt egy "Színes szöveg" „kipipálható” menüponttal! A program a "Színes szöveg" kiválasztáskor színes szöveget, illetve pipa nélküli állapotban fekete szöveget jelenítsen meg! (*menu2*)

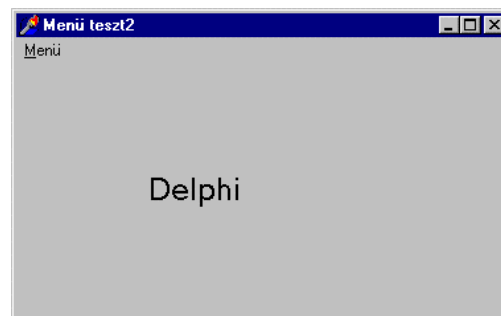
Legyenek a menüpontok az alábbi gyorsítóbíllentyűvel is elérhetők:

"Színes szöveg"	<Ctrl + S>
"Kisbetűs szöveg"	<Ctrl + K>
"Nagybetűs szöveg"	<Ctrl + N>
"Szöveg törlése"	<Ctrl + T>

A feladat megoldását a MENU2 alkalmazás tartalmazza.

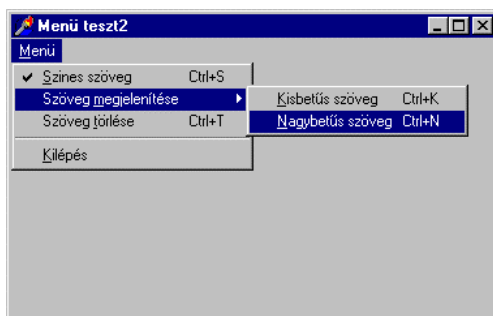
A "Színes szöveg" menüpont pipa nélkül:

fekete színű szöveget jelenít meg.



A pipával jelölt "Színes szöveg" menüpont

piros színű szöveget eredményez.



A *TForm1* osztálya:

```
type
  TForm1 = class(TForm)
    MainMenu1: TMainMenu;
    Menu: TMenuItem;
    SzovegMegjelenites: TMenuItem;
    Kisbetus: TMenuItem;
    Nagybetus: TMenuItem;
    SzovegTorles: TMenuItem;
    N1: TMenuItem;
    Kilepes: TMenuItem;
    SzinesSzoveg: TMenuItem;
  procedure KisbetusClick(Sender: TObject);
  procedure NagybetusClick(Sender: TObject);
  procedure SzovegTorlesClick(Sender: TObject);
  procedure KilepesClick(Sender: TObject);
  procedure FormCreate(Sender: TObject);
  procedure SzinesSzovegClick(Sender: TObject);
  procedure FormPaint(Sender: TObject);
  private
    Szines : boolean;
    Nagybetu: boolean;
    Megjelenites: boolean;
  end;
```



A *Form1* osztályt bővítettük három **boolean** típusú **private** adatmezővel:

<i>A adatmező neve</i>	<i>true érték esetén</i>	<i>false érték esetén</i>
<i>Szines</i>	a szöveg színe piros	a szöveg színe fekete
<i>Nagybetu</i>	a szöveg nagybetűvel jelenik meg	a szöveg kisbetűvel jelenik meg
<i>Megjelenites</i>	a szöveg frissítése	a szöveg törlése

*Eseménykezelő eljárások:*

A *FormCreate* eseménykezelő eljárásban **false** kezdőértéket adunk a *Szines* és a *Megjelenites* adatmezőknek, mellyel a szöveg színét feketére és a törlési módot állítottuk be.

```
procedure TForm1.FormCreate(Sender: TObject);
begin
    Szines := false;
    Megjelenites:= false;
    // az ablak bal felső sarkának koordinátái
    Left := 200;
    Top := 100;
    // az ablak magassága és szélessége
    Height := 240;
    Width := 380;
    // a betű háttere a Form1 háttere lesz
    Canvas.Brush.Color:= Form1.Color;
end;
```

A "Színes szöveg" menütételen való kattintáskor a *SzinesSzovegClick* eseménykezelő eljárásban a *Szines* adatmező értékét ellenkező értékre változtatjuk. Az adatmező **true** állapota a színes (piros) szöveg megjelenítését eredményezi és a menütétel **Checked** tulajdonságának **true** értéke a menütétel „kipipálását” jelenti. Az adatmező **false** értéke fekete szöveg megjelenítését és a menütétel előtt a pipa törlését eredményezi. Az **Invalidate** metódus hívása frissíti az ablakot.

```
procedure TForm1.SzinesSzovegClick(Sender: TObject);
begin
    Szines := not Szines;
    // menüpont kipipálása, illetve
    // megszüntetése
    SzinesSzoveg.Checked := Szines;
    Invalidate;
end;
```

A "Szöveg megjelenítése/Nagybetűs szöveg" menüpont kiválasztásakor a *NagybetusClick* eseménykezelő eljárás hívódik meg, amelyben a *Nagybetu* és a *Megjelenítés* adatmezőket **true** értékre állítjuk. Az **Invalidate** metódus hívásával, a szöveg nagybetűvel jelenik meg a beállított színnel.

```
procedure TForm1.NagybetusClick(Sender: TObject);
begin
    Nagybetu := true;
    Megjelenites:= true;
    Invalidate;
end;
```

A "Szöveg megjelenítése/Kisbetűs szöveg" menüpont kiválasztásakor a *KisbetusClick* eseménykezelő eljárás hívódik meg, amelyben a *Nagybetu* adatmezőt **false** és a *Megjelenítés* adatmezőt pedig **true** értékre állítjuk. Az **Invalidate** metódus hívásával, a szöveg kisbetűvel jelenik meg a beállított színnel.

```
procedure TForm1.KisbetusClick(Sender: TObject);
begin
    Nagybetu := false;
    Megjelenites:= true;
    Invalidate;
end;
```

A "Szöveg törlése" menüpont kiválasztásakor meghívódó *SzovegTorlesClick* eseménykezelő eljárásban a *Megjelenites* adatmező **false** értéke jelenti a szöveg törlését, melyet **Invalidate** metódus aktiválásával a **FormPaint** eljárásban hajtunk végre.

```

procedure TForm1.SzovegTorlesClick(Sender: TObject);
begin
    Megjelenites:= false;
    Invalidate;
end;

```

A **FormPaint** eseménykezelő eljárásban a *Megjelenítés* adatmező **true** értéke esetén történik meg a szöveg kiírása a *Szines* és a *Nagybetu* adatmezők állapotának megfelelően. Ha a *Megjelenites* adatmező értéke **false**, akkor nincs megjelenítés és a frissítési kérélem háttérszínnel történő ablaktörlést jelent.

```

procedure TForm1.FormPaint(Sender: TObject);
begin
    if Megjelenites then
        begin
            if Szines then
                Canvas.Font.Color := clRed
            else
                Canvas.Font.Color := clBlack;
            if Nagybetu then
                Canvas.TextOut(100,80,'DELPHI')
            else
                Canvas.TextOut(100,80,'Delphi');
        end;
end;

```

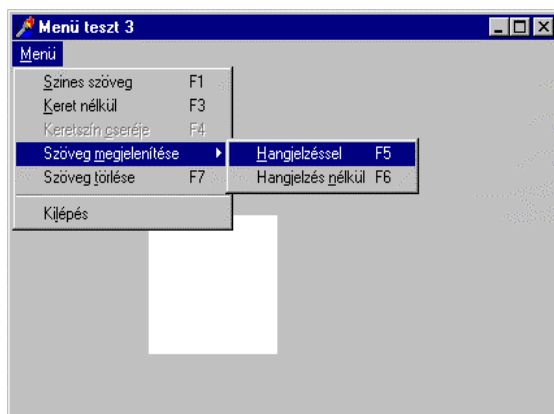


Írjunk olyan programot, amely alkalmas színes, illetve fekete-fehér nagybetűs szöveg hangjelzéssel vagy hangjelzés nélkül, fehér vagy színes szegéllyel ellátott négyzetben történő megjelenítésére! (*menu3*)

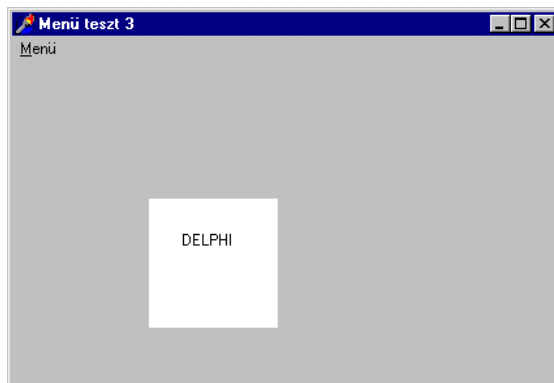
A feladat megoldását a MENU3 alkalmazás tartalmazza. Az alábbi táblázat a menüelemek adatait foglalja össze:

<i>A menü neve</i>	<i>Azonosítója</i>	<i>Gyorsítóbillentyű</i>
Menü	<i>Menu</i>	
Színes szöveg	<i>SzinesSzoveg</i>	F1
Keret nélkül/Kerettel	<i>Keret</i>	F2
Keretszín cseréje	<i>Keretszin</i>	F3
Hangjelzéssel	<i>Hanggal</i>	F4
Hangjelzés nélkül	<i>HangNekul</i>	F5
Szöveg megjelenítése	<i>SzovegMegjelenites</i>	
Szöveg törlése	<i>SzovegTorles</i>	F6

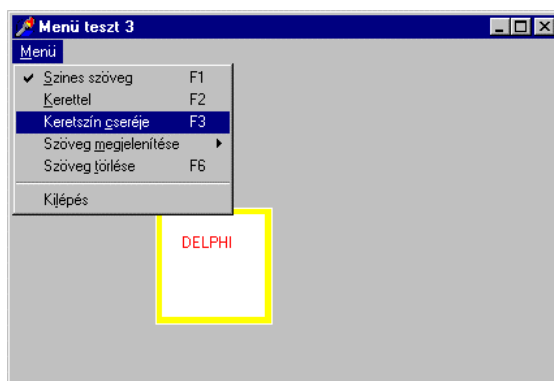
A program futási képei az alábbiak:



A program alapállapota a fehér négyzet.



Fekete szöveg kiírása.



Színes szöveg kiírása.

A "Keret nélkül" menüpont a "Keretszín cseréje" menüpontot letiltja, ez esetben egy fehér négyzet jelenik meg. A "Keret nélkül" menüpontra való kattintáskor a menüpont "Kerettel" menüpontra változik és a négyzet körül sárga keret jelenik meg. Ezek után a "Keretszín cseréje" menüpont is hozzáférhetővé válik. "Keretszín cseréje" menüpont többszöri kiválasztásával, vagy az <F3> gyorsítóbíllentyű nyomogatásával, a négyzet körül a keret színe (sárga, kék, zöld és lila) ciklikusan változik. Ha a "Színes szöveg" menüpont előtt nincs pipa, akkor a szöveg színe fekete, egyébként piros. A "Szöveg megjelenítése" menüpont választásakor a felbukkanó almenüben a "Hangjelzéssel", vagy a "Hangjelzés nélkül" menüpontot választhatjuk ki.

A program működésekor a főablakban egy fehér négyzet jelenik meg, melynek állapotát a menüpontok segítségével, illetve a gyorsítóbíllentyűkkel változtathatjuk meg.

A *TForm1* osztálya:

```
type
  TForm1 = class(TForm)
    MainMenu1: TMainMenu;
    Menu: TMenuItem;
    SzinesSzoveg: TMenuItem;
    Keret: TMenuItem;
    Keretszin: TMenuItem;
    S: TMenuItem;
    Hanggal: TMenuItem;
    HangNelkul: TMenuItem;
    Kilepes: TMenuItem;
    SzovegTorles: TMenuItem;
    N1: TMenuItem;
    procedure HanggalClick(Sender: TObject);
    procedure HangNelkulClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure KilepesClick(Sender: TObject);
    procedure SzinesSzovegClick(Sender: TObject);
    procedure KeretClick(Sender: TObject);
    procedure KeretszinClick(Sender: TObject);
    procedure FormPaint(Sender: TObject);
    procedure SzovegTorlesClick(Sender: TObject);
  end;
```

Globális deklarációk:

```
var
  Form1: TForm1;
  Szines      : boolean; // a szöveg színes
  SzovegKeret: boolean; // a négyzet keretezett
  Keretszine  : integer; // a keret színe
  Megjelenites: integer; // 0 : törlés, 1: hanggal, 2: hang nélkül
```

Eseménykezelő eljárások:

A *FormCreate* eljárás, amelyben a menükezeléshez szükséges változókat állítjuk alapállapotra, a *Form1* létrehozásakor aktivizálódik. A *Szines* változó *false* értéke azt jelzi, hogy az ablakban megjelenő négyzet belseje fehér színű. A *SzovegKeret* változó *false* értéke pedig jelzi, hogy a négyzet körül nem lesz szegély. A *Keretszine* változó a szegély színét határozza meg.

```
procedure TForm1.FormCreate(Sender: TObject);
begin
  Left := 200;
  Top := 100;
  Szines := false;
  SzovegKeret := true;
  Keretszine := 0;
  Megjelenites := 0;
  Canvas.Brush.Color := clWhite;
  Canvas.Font.Color := clBlack;
end;
```

A "Színes szöveg" menüpont kiválasztásakor a *SzinesSzovegClick* eseménykezelő eljárásra kerül a vezérlés. A *Szines* változó **true** értéke jelzi a pipa jelenlétét. Az eljárásban a *Szines* változó értéke **true**, illetve **false** lehet, ennek megfelelően a menüpont (*SzinesSzoveg*) **Checked** tulajdonságot **true** értékre állítva a pipát a menüpont előtt megjelenítjük, illetve **false** esetén eltüntetjük. A **Invalidate** metódus hívásával frissítjük az ablakot, hogy a négyzet újrarajzolása megtörténjen.

```
procedure TForm1.SzinesSzovegClick(Sender: TObject);
begin
    Szines := not Szines;
    // a menüpont kipipálása, illetve
    // illetve a pipa megszüntetése
    SzinesSzoveg.Checked := Szines;
    Invalidate;
end;
```

Az "Szöveg megjelenítése|Hangjelzéssel" menüpont kiválasztásakor a *HanggalClick* eljárásban a *Megjelenites* adatmezőt egyre állítjuk, amely a szöveg hangjelzéssel való kijelzését jelenti.

```
procedure TForm1.HanggalClick(Sender: TObject);
begin
    Megjelenites := 1;
    Invalidate;
end;
```

Az "Szöveg megjelenítése|Hangjelzés nélkül" menüpont kiválasztásakor a *HangnelkulClick* eljárásban a *Megjelenites* adatmezőt kettőre állítjuk, amely a szöveg hangjelzés nélkül való kijelzését jelenti.

```
procedure TForm1.HangNelkulClick(Sender: TObject);
begin
    Megjelenites := 2;
    Invalidate;
end;
```

A "Keretszín cseréje" menüpont kiválasztásakor a *KeretszinClick* eseménykezelő eljárás aktivizálódik. Az eljárás *Keretszine* változóba a sárga, kék, zöld és lila színsor következő színének sorszámát tölti be. A **Invalidate** eljárás aktiválásával az ablakot frissítjük a megváltozott keretszín kirajzolásához.

```
procedure TForm1.KeretszinClick(Sender: TObject);
begin
    Keretszine := (Keretszine+1) mod 4;
    // újrafestési kérelem
    Invalidate;
end;
```

A "Keret nélkül/Kerettel" menüpont kiválasztásakor a *KeretClick* eseménykezelő eljárást hívjuk meg. A *SzovegKeret* változó tartalma alapján – amely a menüpont kiválasztásakor mindig ellenkező értékűre változik – a **Caption** tulajdonság módosításával a megfelelő szövegűre változtatjuk a menüpontot. A "Kerettel" menüpont beállításakor a "Keretszín cseréje" menüpontot is hozzáférhetővé tesszük a *Keretszin* **Enabled** tulajdonságának **true** értékűvé tételével.

```
procedure TForm1.KeretClick(Sender: TObject);
begin
    SzovegKeret := not SzovegKeret;
    if SzovegKeret then
    begin
        Keret.Caption := '&Keret nélkül';
        Keretszin.Enabled := true;
    end
    else
    begin
        Keretszin.Enabled := false;
        Keret.Caption := '&Kerettel';
    end;
    Invalidate;
end;
```

Az **Invalidate** metódussal kérjük az ablak újrafestését. A **FormPaint** eseménykezelő eljárást úgy kell megírni, hogy az újrafestési kérelem valóban a változók állapotának megfelelő állapotot adja vissza.

```
procedure TForm1.FormPaint(Sender: TObject);
begin
  Canvas.Pen.Color := clWhite;
  if SzovegKeret then
  begin
    // a keretszin kiválasztása
    case Keretszine of
      0 : Canvas.Brush.Color := clYellow;
      1 : Canvas.Brush.Color := clBlue;
      2 : Canvas.Brush.Color := clGreen;
      3 : Canvas.Brush.Color := clPurple;
    end;
    Canvas.Rectangle(110,110,200,200);
    Canvas.Brush.Color := clWhite;
    Canvas.Rectangle(115,115,194,194);
  end
  else
  begin
    Canvas.Brush.Color := clWhite;
    Canvas.Rectangle(105,105,204,204);
  end;
  if Megjelenites = 0 then
  begin
    // szöveg törlése
    Canvas.Font.Color := clWhite; // az írás színe fehér
    Canvas.TextOut(130,130,'DELPHI'); // fehér szöveg fehér mezőben
  end
  else
  begin
    if Szines then
      Canvas.Font.Color := clRed
    else
      Canvas.Font.Color := clBlack;
      if Megjelenites = 1 then beep;
      Canvas.TextOut(130,130,'DELPHI');
    end;
  end;
end;
```

A "Szöveg törlése" menüpont kiválasztásakor a *SzovegTorlesClick* eseménykezelő eljárásban a *Megjelenites* adatmező nullára állítás jelzi a szöveg törlését, amely a **FormPaint** eseménykezelő eljárásban történik meg az **Invalidate** függvény hívásakor.

```
procedure TForm1.SzovegTorlesClick(Sender: TObject);
begin
  Megjelenites := 0;
  Invalidate;
end;
```



Használjunk választógombként működő menüpontot és alkalmazzuk a MENU3 program a "Színes szöveg" menüpontjánál!(*menu4*)

A MENU4 alkalmazás futási képe:



A "Színes szöveg" menüpont *Name* azonosítója *SzinesSzoveg*. A menüpont választógombként való működéséhez a *SzinesSzoveg RadioItem* tulajdonságát *true* értékre állítjuk a *FormCreate* eseménykezelő eljárásban.

```
procedure TForm1.FormCreate(Sender: TObject);
begin
    Left := 200;
    Top := 100;
    // a menüpont választógombként működik
    SzinesSzoveg.RadioItem := true;
    // az adatmezők kezdőértékének beállítása
    Szines := false;
    SzovegKeret := false;
    Keretszine := 0;
    Megjelenites := 0;
    Canvas.Brush.Color := clWhite;
    Canvas.Font.Color := clBlack;
end;
```

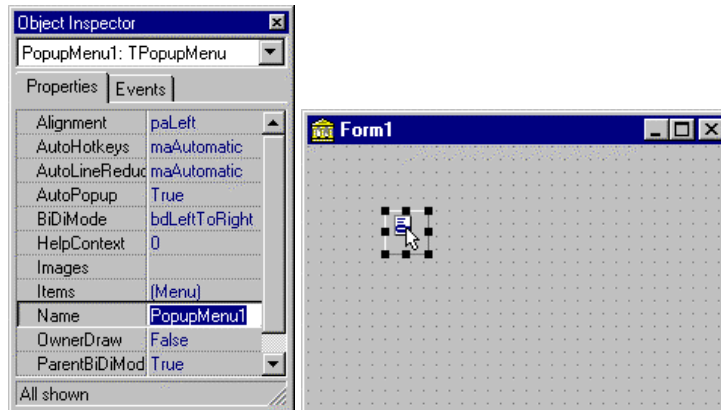


Készítsünk olyan programot, melyben felbukkanó menüből választva kiszámítjuk 2 négyzetét, valamint négyzetgyökét - az eredményt törölhetjük is. (*popup0*)

Válasszuk ki a felbukkanó menü (*PopupMenu*)



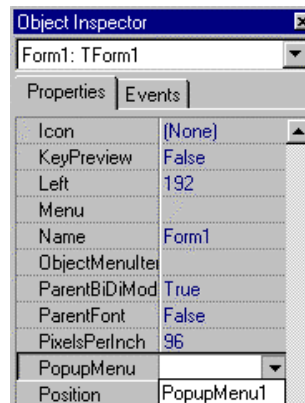
komponenst a **Standard** komponenslapról és utána kattintsunk a formon!



A felbukkanó menü menüpontjai és azonosítói legyenek:

<i><b>Caption</b></i>	<i><b>Name</b></i>
2 négyzete	Negyzet
2 négyzetgyöke	Negyzetgyok
Törlés	Torles

A felbukkanó menüt rendeljük hozzá a *Form1*-hez, a **PopupMenu** tulajdonságnál megjelenő legördülő listából választuk a *PopupMenu1* objektumot!



Ezután a felbukkanó menü menüpontjain kétszer kattintunk, és megírjuk az eseménykezelő eljárásokat. Például az "2 négyzete" menüpont kiválasztásakor a (20,20) koordinátájú pontban 2 négyzetét írjuk.

```
procedure TForm1.NegyzetClick(Sender: TObject);
begin
    Canvas.TextOut(20,20,'2 négyzete      : '+ FloatToStr(sqr(2.0)));
end;
```

A "2 négyzetgyöke" menüpont kiválasztásakor a 2 négyzetgyökét a (20,40) koordinátájú pontban írjuk.

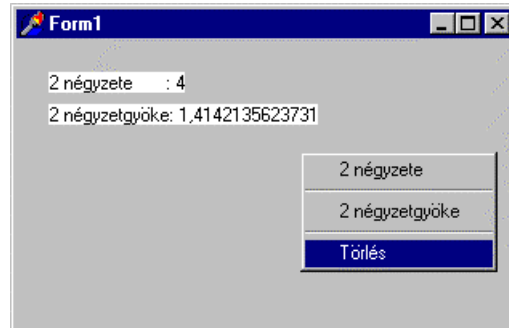
```
procedure TForm1.NegyzetgyokClick(Sender: TObject);
begin
    Canvas.TextOut(20,40,'2 négyzetgyöke: '+ FloatToStr(sqrt(2.0)));
end;
```

A "Törlés" menüpont kiválasztásakor meghívjuk az **Invalidate** metódust, amely a **FormPaint** eseménykezelő eljárás meghívását eredményezi:



```
procedure TForm1.TorlesClick(Sender: TObject);  
begin  
    Invalidate;  
end;
```

Az **Invalidate** hatására törlődik az ablak, mivel nincs **FormPaint** eseménykezelő eljárás, amely az ablakot frissítő utasításokat tartalmazná.



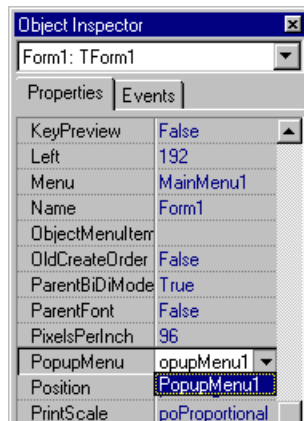
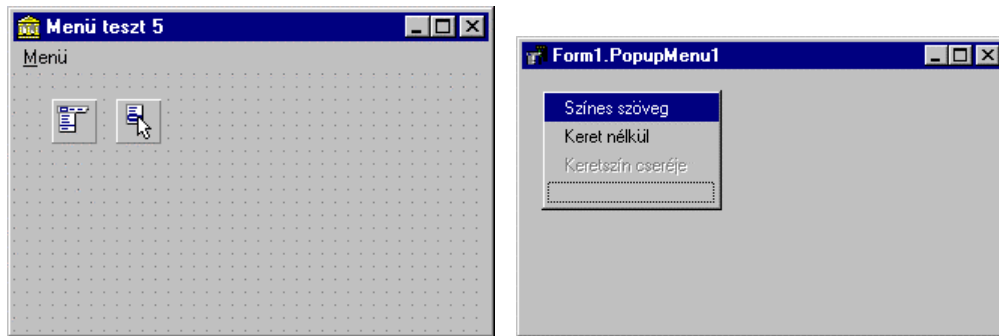
---

 Módosítsuk a MENU3 programot úgy, hogy a „Menü” első három menüpontja legördülő menüből is működtethető legyen! (*popup1*)

---

Gondoskodnunk kell arról, hogy a menü és a legördülő menü menüpontjai szinkronban működjenek. Például az egyiknél kijelölt tétel a másiknál is kijelölt legyen, illetve fordítva.

Feltesszük a Form1 formra a *MainMenu* ikon mellé a *PopupMenu* komponenst. A *PoupMenu* komponensen kattintva a menüpontokat *MainMenu* szerkesztéséhez azonos módon hozzuk létre.



A legördülő menü automatikus működéséhez szükséges, hogy a *Form1 PopupMenu* tulajdonságánál a legördülő ablakból kiválasszuk a *PopupMenu1* tételt. Ezek után a futó alkalmazásnál az egér jobb oldali gombjának megnyomásakor megjelenik az általunk tervezett legördülő menü.

A menük neve és azonosítói táblázata:

<i>A menü</i>			<i>Legördülő menü</i>
<b>Caption</b>	<b>Name és ShortCut</b>		<b>Caption</b>
Menü	Menu		
Színes szöveg	SzinesSzoveg	F1	Színes szöveg
Keret nélkül/Kerettel	Keret	F2	Keret nélkül/Kerettel
Keretszín cseréje	Keretszin	F3	Keretszín cseréje
Hangjelzéssel	Hanggal	F4	
Hangjelzés nélkül	HangNekul	F5	
Szöveg megjelenítése	SzovegMegjelenites		
Szöveg törlése	SzovegTorles	F6	
Kilépés	Kilepes		

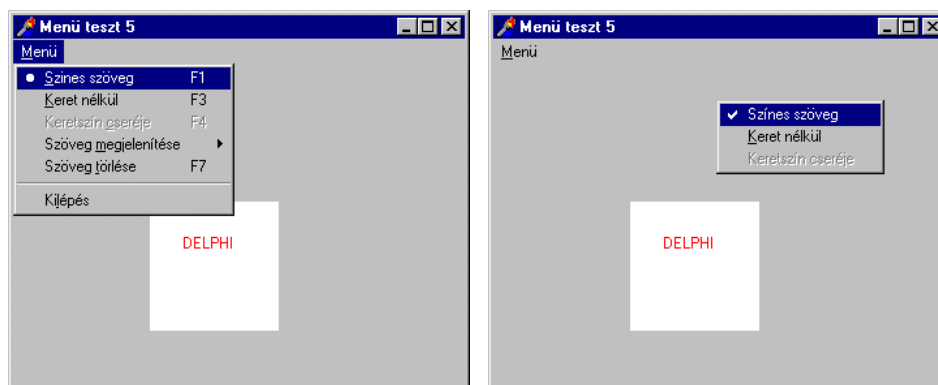
A két menüben a *SzinesSzoveg* kiválasztása mindkét esetben a *SzinesSzovegClick* eseménykezelő eljárást aktivizálja. Az eseménykezelő eljárásban mindkét menüpont kijelölését kezeljük (pötty illetve a pipa jelentését) illetve a kijelölés megszüntetését. Hasonlóan járhatunk el a *Keret* esetén:

```

procedure TForm1.SzinesSzovegClick(Sender: TObject);
begin
    Szines := not Szines;
    // a Menü "Színes szöveg" menüpontja pettyel, illetve
    // illetve petty nélkül
    SzinesSzoveg.Checked := Szines;
    // a legördülő menüben a "Színes szöveg" menüpont pipával,
    // illetve a pipa nélkül
    PopupSzinesSzoveg.Checked := Szines;
    Invalidate;
end;

```

A program futási képei.





Készítsünk programot, mely felbukkanó menüben jeleníti meg a képernyőn használható betűtípusokat. A felbukkanó menü segítségével állítsuk be egy címke betűtípusát! (*font*)

Helyezzük a formra a *PopupMenu1* felbukkanó menüt és a *Label1* címkét, melynek **Caption** tulajdonsága *'Delphi'* és igazodik az ablakhoz (*Align=alClient*)! Válasszuk a *Form1 PopupMenu* tulajdonságát *PopupMenu1*-re!

Kétszer a formra kattintva elkészíthetjük az ablak létrehozásakor (**FormCreate** esemény) a *PopupMenu1* felbukkanó menüt. Használjuk az *ujelem* változót a menüelemek sorrendben való létrehozásához (**TMenuItem.Create** metódus)! A **Screen** objektum **Fonts** tulajdonsága tartalmazza az elérhető fontneveket. Az *Add* metódussal töltjük a menüt

```
procedure TForm1.FormCreate(Sender: TObject);
var ujelem:TMenuItem;
    i:integer;
begin
  for i:=0 to Screen.Fonts.Count-1 do
  begin
    ujelem := TMenuItem.Create(Self);
    ujelem.Caption := Screen.Fonts[i];
    PopupMenu1.Items.Add(ujelem);
    PopupMenu1.Items[i].OnClick:=click;
  end;
end;
```

Készítsük el a menüválasztást kezelő eljárást! Kihhasználjuk, hogy a fontnevet minden menüelem **Caption** jellemzőjében tartalmazza.

```
procedure TForm1.Click(Sender: TObject);
begin
  Label1.font.Name:=TMenuItem(Sender).Caption;
end;
```

Ne felejtsük el a *TForm1* osztályba is deklarálni a **Click** eseménykezelőt!

```
TForm1 = class(TForm)
  PopupMenu1: TPopupMenu;
  Label1: TLabel;
  procedure FormCreate(Sender: TObject);
private
  { Private declarations }
public
  { Public declarations }
  procedure Click(Sender: TObject);
end;
```