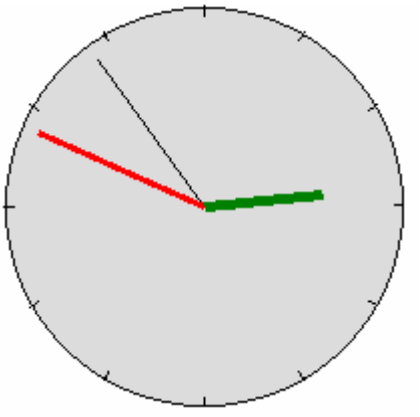
Johanyák Zsolt Csaba: **Analóg óra – oktatási segédlet**
http://www.johanyak.hu
e-mail: johanyak.csaba@gamf.kefo.hu

Analóg óra

Készítsünk egy analóg órát megjelenítő alkalmazást.



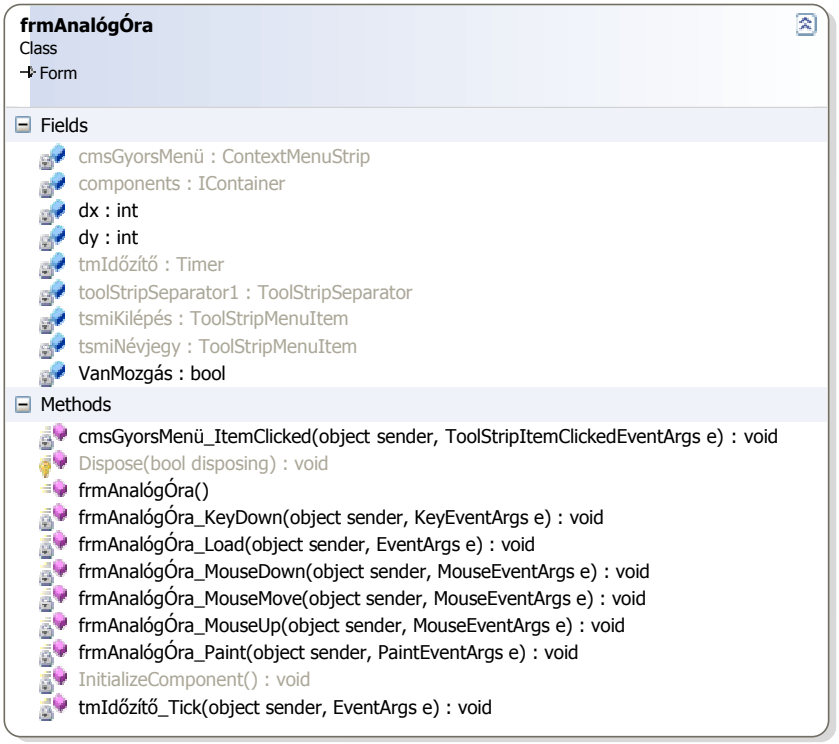
A feladat egy lehetséges megoldása a következő:

1. Az alkalmazás vázának automatikus generálása
   Fájl menü, New, Project …
   Project Types: Visual C# - Windows
   Templates: Windows Application
   Name: AnalogOra
   OK

2. Felület kialakítása
   A Solution Explorerben nevezzük át a formot tartalmazó forrásállományt frmAnalogOra.cs-re.A formot nevezzük át (Name=frmAnalógÓra).
   Nincs szükségünk fejlécre és keretre, ezért FormBorderStyle=None, valamint az ablak legyen mindig látható: TopMost=True.
   Az ablak mérete legyen 200x200-as. Ezt a konstruktorban állítjuk be:

```
public frmAnalógÓra()
{ InitializeComponent();
  Width = 200;
  Height = 200;
}
```

3. Időkezelés. Az óra animációt egy Timer komponens segítségével oldjuk meg. Ez minden másodpercben egy Tick eseményt fog generálni. Ennek feldolgozása során frissítjük (újrarajzoljuk) a rajzterületet.
   Tegyünk a formra egy Timer komponenst. Tulajdonságai: Name=tmIdőzítő és Interval=1000 (ez 1 másodperc). Az ablak konstruktorában engedélyezzük a Timert. Ennek hatására a program indulásakor rögtön aktivizálódik az időzítő, azaz működik az óra.




```
tmIdőzítő.Enabled = true;
```

   Az időzítő minden másodpercben generál egy Tick eseményt. Készítsünk ehhez egy eseménykezelőt, amiben előírjuk a rajzterület (form) frissítését, azaz egy Paint esemény előidézését. A formra a Paint esemény feldolgozása során fogunk rajzolni.

```
private void tmIdőzítő_Tick(object sender, EventArgs e)
{ this.Refresh();
}
```

4. Rajzolás (mutatók)
   Készítsünk az ablak Paint eseményéhez egy eseménykezelőt. Minden másodpercben keletkezik egy Paint esemény, és ennek feldolgozása során jelenítjük meg az órát. Vonalak rajzolásához minden egyes vonaltípushoz, még ha csak színben térnek is el egymástól, egy külön Pen objektumot kell definiálnunk. A kifestéshez Brush objektumot kell definiálnunk. A rajzolás egy Graphics típusú objektum segítségével lehetséges, ennek metódusai az egyes rajzoló függvények. Referenciáját az eseménykezelő metódus második paraméterének egy adattagjaként kapjuk meg.

```
/// <summary>
/// Megrajzolja az órát és a pontos időt
/// </summary>
private void frmAnalógÓra_Paint(object sender, PaintEventArgs e)
{ // A háttér kifestéshez szükséges ecset objektum definiálása
  SolidBrush sbEcset = new SolidBrush(Color.Gainsboro);
  // Kifestett óra számlap megrajzolása
  e.Graphics.FillEllipse(sbEcset, 1, 1, Width - 2, Height - 2);
  // Toll definiálása a számlap keretvonalához
  Pen pToll = new Pen(Color.Black);
  // Keretvonal megrajzolása
  e.Graphics.DrawEllipse(pToll, 1, 1, Width - 2, Height - 2);
  // Középpont pozíciójának meghatározása
  int cx = Width / 2;
  int cy = Height / 2;
  //
  // Jelzővonalak rajzolása 5 percenként
  //
  for (int i = 0; i < 60; i = i + 5)
  { e.Graphics.DrawLine(pToll, (int)(cx + (cx - 5) *
      Math.Sin(i * Math.PI / 30)),
      (int)(cy - (cx - 5) * Math.Cos(i * Math.PI / 30)),
      (int)(cx + cx * Math.Sin(i *
      Math.PI / 30)),
      (int)(cy - cy * Math.Cos(i * Math.PI / 30)));
  }
  // A lefoglalt erőforrások felszabadítása
  sbEcset.Dispose();
  pToll.Dispose();
  //
  // Idő lekérdezése
  //
  DateTime ido = DateTime.Now;
  //
  // Másodperc mutató megrajzolása
  //
  int RMp = (int)(0.9 * cx);  //a másodpercmutató hossza
  // Toll létrehozása a másodpercmutatóhoz
  Pen pMpToll = new Pen(Color.Black, 1);
  // Vonalhúzás a középpontból
  e.Graphics.DrawLine(pMpToll, cx, cy, (int)(cx + RMp *
    Math.Sin(ido.Second * Math.PI / 30)),
    (int)(cy - RMp * Math.Cos(ido.Second * Math.PI / 30)));
```

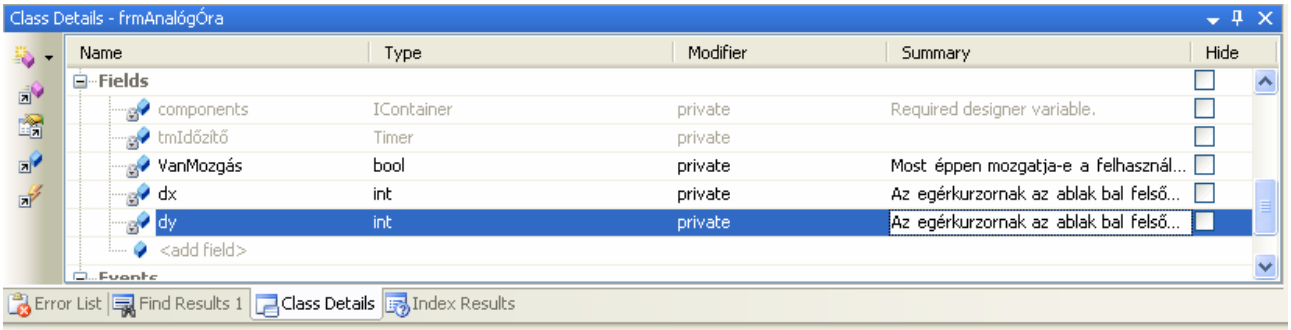
```
        // Erőforrás felszabadítás
        pMpToll.Dispose();
        //
        // Perc mutató megrajzolása
        //
        int RP = (int)(0.9 * cx);  //a percmutató hossza
        // Toll létrehozása a percmutatóhoz
        Pen pPToll = new Pen(Color.Red, 3);
        // Vonalhúzás a középpontból
        e.Graphics.DrawLine(pPToll, cx, cy, (int)(cx + RP * Math.Sin(ido.Minute *
          Math.PI / 30)),
          (int)(cy - RP * Math.Cos(ido.Minute * Math.PI / 30)));
        // Erőforrás felszabadítás
        pPToll.Dispose();
        //
        // Óra mutató megrajzolása
        //
        int RO = (int)(0.6 * cx);  //a percmutató hossza
        // Toll létrehozása az óramutatóhoz
        Pen pOToll = new Pen(Color.Green, 5);
        // Vonalhúzás a középpontból
        e.Graphics.DrawLine(pOToll, cx, cy, (int)(cx + RO * Math.Sin((ido.Hour +
          ido.Minute / 60.0) * Math.PI / 6)),
          (int)(cy - RO * Math.Cos((ido.Hour + ido.Minute / 60.0) *
          Math.PI / 6)));
        // Erőforrás felszabadítás
        pOToll.Dispose();
    }
```

5. Esc billentyű lenyomására kilépés a programból
   Készítünk egy eseménykezelőt az ablakon történő billentyűlenyomás (KeyDown) eseményhez.

```
/// <summary>
/// ESC billentyű hatására kilépés a programból
/// </summary>
private void frmAnalógÓra_KeyDown(object sender, KeyEventArgs e)
{ if (e.KeyCode == Keys.Escape)
  { tmIdőzítő.Enabled = false;
    Application.Exit();
  }
}
```

6. Legyen az óra ablaka kör alakú
   Állítsuk be az ablak háttérszínét pl. kékre: BackColor=Blue. Legyen a TransparencyKey=Blue, így futásidőben csak a kéktől eltérő rész marad meg az ablakból.

7. Legyen mozgatható az ablak az egér segítségével.
   Létrehozunk egy bool adattagot VanMozgás néven, feladata azon információ tárolása, hogy most éppen mozgatja-e a felhasználó az ablakot. Létrehozunk két int adattagot dx és dy néven, feladatuk az egérkurzornak az ablak bal felső sarkától mért vízszintes és függőleges távolságának tárolása.



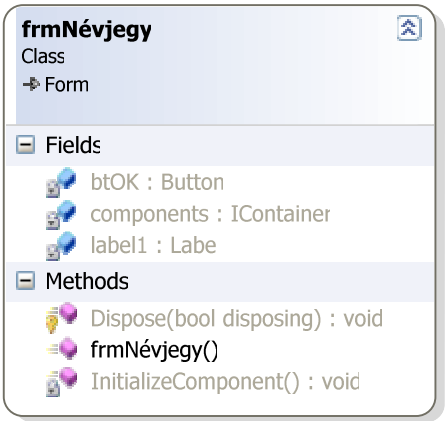
Készítünk egy eseménykezelőt az egérgomb lenyomásához (MouseDown). A bal egérgomb lenyomásakor VanMozgás-t igazra állítjuk, és tároljuk, hogy az egér hol van az ablak bal felső sarkához képest.

```
/// <summary>
/// A bal egérgomb lenyomásakor VanMozgas-t igazra állítjuk, és tároljuk,
/// hogy az egér hol van az ablak bal felső sarkához képest.
/// </summary>
private void frmAnalógÓra_MouseDown(object sender, MouseEventArgs e)
{ if (e.Button == MouseButtons.Left)
  { VanMozgás = true;
    dx = e.X;
    dy = e.Y;
  }
}
```

A bal egérgomb felengedésekor vége az ablakmozgatásnak. Készítünk egy eseménykezelőt az egérgomb felengedéséhez (MouseUp). Ebben a VanMozgás-t hamisra állítjuk.

```
/// <summary>
/// A bal egérgomb felengedésekor vége az ablakmozgatásnak.
/// A VanMozgás-t hamisra állítjuk.
/// </summary>
private void frmAnalógÓra_MouseUp(object sender, MouseEventArgs e)
{ if (e.Button == System.Windows.Forms.MouseButtons.Left)
  { VanMozgás = false;
  }
}
```

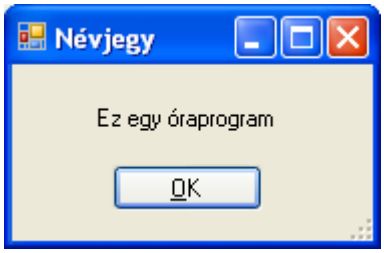
Készítünk egy eseménykezelőt az egér mozgatás (MouseMove) eseményhez. Egér mozgatásakor ellenőrizzük, hogy a mozgatás be van-e kapcsolva. Ha igen, akkor az egér koordinátáit átszámoljuk ablak koordinátarendszerből képernyő koordinátarendszerbe. Az így kapott értékekből levonjuk az előzőekben meghatározott ablak bal felső sarok - egér kurzor távolságokat (dx és dy). A kapott eredmény adja az ablak bal felső sarkának új pozícióját.



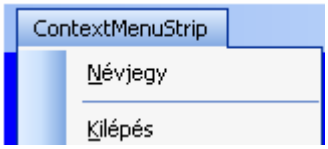

```
    private void frmAnalógÓra_MouseMove(object
sender, MouseEventArgs e)
    { if (VanMozgás)
      { Point p = new Point(e.X, e.Y);
        p = PointToScreen(p);
```

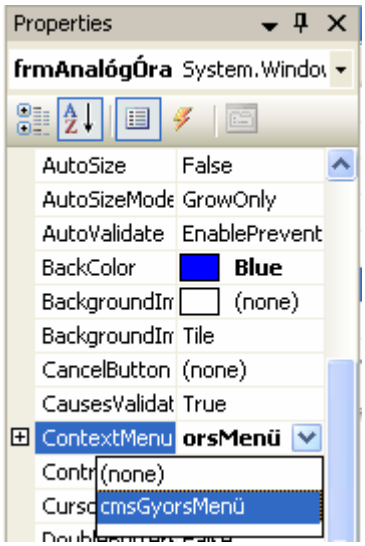
```
            Left = p.X - dx;
            Top = p.Y - dy;
        }
    }
```

8. Névjegy panel készítése.
   Project menü, Add Windows Form...
   Templates: Windows Form
   Name: frmNevjegy.cs
   Text=Névjegy és Name=frmNévjegy
   Elhelyezünk egy címkét (Label) valamilyen felirattal (Text=…), és egy nyomógombot (Button) OK felirattal. A nyomógomb tulajdonságai:
   Text=&OK és DialogResult=OK
   Ez utóbbi azt eredményezi, hogy a gombon kattintva bezáródik a névjegyablak, és az őt eredetileg megjelenítő ShowDialog() metódus DialogResult.OK értékkel tér vissza.



9. Gyorsmenü készítése
   Egy ContextMenuStrip típusú komponenst helyezünk a formra (frmAnalógÓra), neve: Name=cmsGyorsMenü. Kijelöljük, majd a formon megjelenő menüszerkesztőben létrehozunk egy Névjegy (Name=tsmiNévjegy) és egy Kilépés (Name=tsmiKilépés) menüpontot. Köztük legyen egy elválasztó vonal. A form tulajdonságaiban beállítjuk a gyorsmenüt:
   ContextMenuStrip=cmsGyorsMenü





10. Eseménykezelő készítése a gyorsmenühöz.
    Tervezési nézetben kiválasztjuk a gyorsmenüt, majd a Properties ablakban duplán kattintunk az ItemClicked eseményen. A kiválasztott menüpont felirata alapján készítjük a feltételes elágazás szerkezetet az egyes menüpontokhoz tartozó funkcionalitás megvalósítására.

```
private void cmsGyorsMenü_ItemClicked(object sender,
  ToolStripItemClickedEventArgs e)
{ switch (e.ClickedItem.Text)
  { case "&Kilépés":
      Application.Exit();
      break;
    case "&Névjegy":
      frmNévjegy frmNévjegy = new frmNévjegy();
      frmNévjegy.ShowDialog();
      break;
  }
}
```

11. Beállítjuk a főablak kezdőpozícióját a 300,300-as koordinátájú pontba.
    Ehhez eseménykezelőt készítünk a főablak Load eseményéhez.

```
/// <summary>
/// Beállítjuk a főablak kezdőpozícióját a 300,300-as
/// koordinátájú pontba.
/// </summary>
```

```
private void frmAnalógÓra_Load(object sender, EventArgs e)
{ Left = 300;
  Top = 300;
}
```

Feladat

Alakítsa át úgy a programot, hogy a mutatókat ne egy vonal jelképezze, hanem a mellékelt ábrának megfelelő alakjuk legyen.