Medii vizuale de programare
Curs 5

Conf. dr.ing. GENGE Béla

Universitatea "Petru Maior", Departamentul de Informatică
Tîrgu Mureș, România
bela.genge@ing.upm.ro

GDI+

- Graphic Device Interface (GDI+)
- C# pune la dispoziție un set de clase pentru gestionarea interfeței grafice
- Namespace: `System.Drawing`

Principala clasă

- `System.Drawing.Graphics`
- Metode:
  - `FillRectangle`
  - `DrawString`
  - ...

Întrebare

- Cum accesăm GDI+ sau controalele standard dintr-un alt fir de execuție?
- Creare aplicație WinForm
- Adăugare: ProgressBar & Label
- Crearea a `n` fire de execuție, fiecare fir numără de la 0 la 100 și avansează progress bar
- Lansare cu F5 (cu Ctrl+F5 ar lua mai mult timp să observăm eroarea)

Inițializare ProgressBar

```
pb.Minimum = 0;
pb.Maximum = 100;
pb.Value = 0;
pb.Step = 1;
...
pb.PerformStep();
```

Problema principală

- Accesul la Interfața Utilizator (UI-Engleză) nu este thread-safe
- Aceeași problemă în majoritatea limbajelor de programare, e.g., MS C++ MFC, Qt, Visual Basic, C#, etc..
- Proprietatea `InvokeRequired`. Accesare din firul ferestrei și dintr-un alt fir - valori diferite.

Utilizare InvokeRequired

```
if ( label.InvokeRequired ) {
// Modificarea valorii nu se poate face din alt fir
}
```

Soluția 1

- Utilizarea funcțiilor thread-safe pentru notificarea firului principal prin evenimente
- Se declară un delegat `delegate void UpdateUIStatusDelegate(string status);`
- În Form se declară o metodă publică cu aceeași semnătură ca și delegatul
- În firul apelant se testează dacă e nevoie de apel thread-safe asupra controalelor `if(f.InvokeRequired)`
- În caz afirmativ: `f.BeginInvoke(new UpdateUIStatusDelegate(f.InnerUpdateStatus), str);`
- Altfel: `f.InnerUpdateStatus(str);`

Soluția 1 - sintaxa BeginInvoke

Syntax

C# | C++ | F# | VB

```
public IAsyncResult BeginInvoke(
        Delegate method,
        params object[] args
)
```

Soluția 1 - exemplu

```
public delegate void MyDelegate(Label myControl, string myArg2);

private void Button_Click(object sender, EventArgs e)
{
    object[] myArray = new object[2];

    myArray[0] = new Label();
    myArray[1] = "Enter a Value";
    myTextBox.BeginInvoke(new MyDelegate(DelegateMethod), myArray);
}

public void DelegateMethod(Label myControl, string myCaption)
{
    myControl.Location = new Point(16,16);
    myControl.Size = new Size(80, 25);
    myControl.Text = myCaption;
    this.Controls.Add(myControl);
}
```

Soluția 2

- Fiecare aplicație are un Dispatcher thread - procesează evenimente și reîmprospătează conținutul UI
- Rezolvarea presupune utilizarea unei cozi de mesaje procesate la intervale regulate de firul Dispatcherului
- Se definește un Timer rulat la intervale regulate de Dispatcher (pe același fir)
- Atenție! Timerele uzuale rulează pe un alt fir de execuție!

Pașii

- `DispatcherTimer` este definit în WPA $\Rightarrow$ trebuie adăugat assembly-ul `WindowsBase`
- Click dreapta `References` $\Rightarrow$ `Add Reference` $\Rightarrow$ `.NET` $\Rightarrow$ `WindowsBase`
- În codul Form-ului se creează `DispatcherTimer`
- Se definește funcția callback `private void dispatcherTick(object sender, EventArgs e)`
- Se definește o listă de string-uri cu acces sincronizat (sync_enqueue & sync_dequeue)

Creare DispatcherTimer

```
_timer = new System.Windows.Threading.DispatcherTimer();
_timer.Tick += new EventHandler(dispatcherTick);
_timer.Interval = new TimeSpan(0, 0, 0, 0, 100);
_timer.Start();
```

Blocarea firului principal

- Timpul de execuție al DispatcherThread trebuie să fie minim
- Altfel se blochează UI
- Exemplu: while infinit

Double-Buffering

- O problemă răspândită în GDI+ este cea a pâlpâirii (flickering) ferestrei la desenare
- Problema: ștergerea și redesenarea
- Soluția: desenarea într-un buffer din memorie și copierea conținutului în fereastra principală
- Atenție! Nu toate operațiile de double-buffering sunt thread-safe!!!

Double-Buffering - automat

- În C# double-buffering poate fi activat prin proprietatea ferestrei `DoubleBuffered=true`
- Desenarea se va realiza într-un buffer de memorie care este apoi copiat automat pe UI
- Pentru desenare se suprascrie evenimentul `Paint`
- Desenarea se face prin contextul grafic `PaintEventArgs e;` `e.Graphics`
- Firul apelează `Invalidate()` pentru declanșarea redesenării

Exemplu

```
System.Drawing.Brush br = new System.Drawing.SolidBrush(Color.Red);
System.Drawing.Brush br1 = new System.Drawing.SolidBrush(Color.White);
System.Drawing.Font font = new System.Drawing.Font("Arial", 16);
int y = 100;
e.Graphics.FillRectangle(br1, 0, 0, this.Width, this.Height);
e.Graphics.DrawString("Hello World!", font, br, _nX, y); //_nX modificat de firul de execuție
br.Dispose();
br1.Dispose();

font.Dispose();
```

Double-Buffering - manual

- Crearea unui buffer
- Firul desenează direct pe buffer
- Copierea bufferului pe ecran se realizează prin evenimentul `Paint`
- Firul apelează `Invalidate()` pentru declanșarea redesenării
- Atenție! Operațiile asupra bufferului nu sunt thread-safe!!!

Exemplu

```
private BufferedGraphics _bg = null;// Se declară în form
BufferedGraphicsContext currentContext = BufferedGraphicsManager.Current;//Constructor
_bg = currentContext.Allocate(this.CreateGraphics(), new Rectangle(0, 0, this.Width, this.Height));
//Se adaugă o metodă publică accesibilă firelor de execuție
lock (_bg)//Protejare resursă
{
_bg.Graphics.FillRectangle(br1, 0, 0, this.Width, this.Height);
_bg.Graphics.DrawString("Hello World!", font, br, _nX, y);
}
//În Paint
lock (_bg)
{
_bg.Render(e.Graphics);//Dacă buffering automat e activat va duce la mai multe copieri

}
```