

Moving a form and a panel in parallel should not be as difficult ?!?!?!?

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.framework.windowsforms/2006-11/msg00470.h>

- *From:* "gourmete" <bristian@xxxxxxxxxxxxx>
 - *Date:* 27 Nov 2006 13:13:02 -0800
-

Hi!

I'm developing a screen-capture application.

My Idea is to cover the whole screen with a semi-transparent clickt-hrough layer, solely the Area that will be captured should appear without this layer.

What I did:

I use two forms.

The first form works as the background-layer. It has no border and covers the whole screen. Opacity is set and the ex.Style ist WS_EX_TRANSPARENT to make it click-through. On this Form I have a panel wich bgcolor is set to the forms transparency-key, hence we can look through the panels area.

Second form is not click-through, because i need controls on it. There is also an transparent panel on it, which refers to the area to be captured. When moving and resizing this form I map position and size of both forms transparent panels.

The way it works is exactly what I had in mind, but there is one problem: When moving the front-form I update the location of the transparent panel on the back-form. This update takes to much time, i. e. the second panel moves two slow.

Does anyone can give me an hint how I can move my front-form and the transparent-panel on the back-form in exactly the same time. I read about the DeferWindowPos Win32 Methods, but I didn't get it work.

Since I think nobody understood what I'm talking about I'll provide some code:

-----BACK-FORM-----

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
```

Moving a form and a panel in parallel should not be as difficult ?!?!?!

```
namespace transp
{
public class Back : Form
{
private System.ComponentModel.IContainer components = null;
private Panel transparentArea;
public const int WS_EX_TRANSPARENT = 32;

protected override void Dispose(bool disposing)
{
if (disposing && (components != null))
{
components.Dispose();
}
base.Dispose(disposing);
}

private void InitializeComponent()
{
this.transparentArea = new System.Windows.Forms.Panel();
this.SuspendLayout();
//
// transparentArea
//
this.transparentArea.BackColor =
System.Drawing.Color.Fuchsia;
this.transparentArea.Location = new
System.Drawing.Point(40, 73);
this.transparentArea.Name = "transparentArea";
this.transparentArea.Size = new System.Drawing.Size(200,
100);
this.transparentArea.TabIndex = 0;
//
// Back
//
this.AutoScaleDimensions = new System.Drawing.SizeF(6F,
13F);
this.AutoScaleMode =
System.Windows.Forms.AutoScaleMode.Font;
this.BackColor = System.Drawing.Color.LightSlateGray;
this.ClientSize = new System.Drawing.Size(292, 266);
this.Controls.Add(this.transparentArea);
this.FormBorderStyle =
System.Windows.Forms.FormBorderStyle.None;
this.Name = "Back";
this.Opacity = 0.8;
this.Text = "Form1";
this.TopMost = true;
this.TransparencyKey = System.Drawing.Color.Fuchsia;
this.ResumeLayout(false);
}
```

Moving a form and a panel in parallel should not be as difficult ?!?!?!

Moving a form and a panel in parallel should not be as difficult ?!?!?!?

```
}  
  
public Back()  
{  
InitializeComponent();  
this.Bounds = SystemInformation.VirtualScreen;  
}  
  
public Rectangle TransparentAreaBounds  
{  
get { return this.transparentArea.Bounds; }  
set { this.transparentArea.Bounds = value; }  
}  
  
protected override CreateParams CreateParams  
{  
get  
{  
//make it click-through  
CreateParams cp = base.CreateParams;  
cp.ExStyle |= WS_EX_TRANSPARENT;  
return cp;  
}  
}  
}
```

-----FRONT-FORM-----

```
using System;  
using System.Collections.Generic;  
using System.ComponentModel;  
using System.Data;  
using System.Drawing;  
using System.Text;  
using System.Windows.Forms;  
  
namespace transp  
{  
public class Front : Form  
{  
private System.ComponentModel.IContainer components = null;  
private Back backGroundForm = null;  
  
protected override void Dispose(bool disposing)  
{  
if (disposing && (components != null))  
{  
components.Dispose();  
}  
}
```

Moving a form and a panel in parallel should not be as difficult ?!?!?!?

Moving a form and a panel in parallel should not be as difficult ?!?!?!

```
base.Dispose(disposing);
}

private void InitializeComponent()
{
    this.SuspendLayout();
    this.AutoScaleDimensions = new System.Drawing.SizeF(6F,
    13F);
    this.AutoScaleMode =
    System.Windows.Forms.AutoScaleMode.Font;
    this.ClientSize = new System.Drawing.Size(292, 266);
    this.FormBorderStyle =
    System.Windows.Forms.FormBorderStyle.SizableToolWindow;
    this.Name = "Front";
    this.Text = "Front";
    this.ResumeLayout(false);
}

public Front()
{
    InitializeComponent();
    this.TopMost = true;
    this.BackColor = Color.Fuchsia;
    this.TransparencyKey = Color.Fuchsia;
    this.Move += new EventHandler(UpdateBackGround);
    this.Resize += new EventHandler(UpdateBackGround);
    backGroundForm = new Back();
    this.Show();
    backGroundForm.Show();
    backGroundForm.TransparentAreaBounds = this.Bounds;
}

void UpdateBackGround(object sender, EventArgs e)
{
    backGroundForm.TransparentAreaBounds = this.Bounds;
}

[STAThread]
static void Main()
{
    Application.EnableVisualStyles();
    Application.SetCompatibleTextRenderingDefault(false);
    Application.Run(new Front());
}
}
}
```

I would appreciate any hints. I'm struggling for days now !!!!!
Thanks!

Moving a form and a panel in parallel should not be as difficult ?!?!?!

Moving a form and a panel in parallel should not be as difficult ?!?!?!