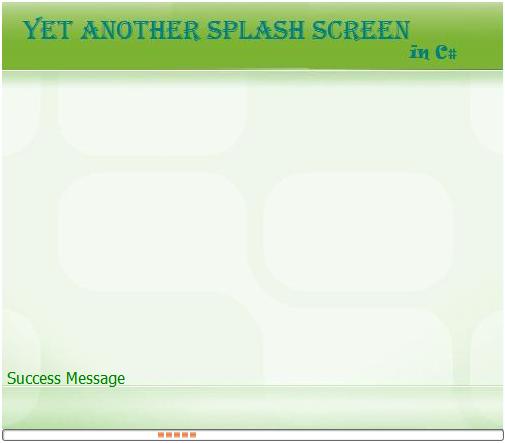
Yet Another Splash Screen in C#



Introduction

The article describes creating a simple splash screen in C#. The intent of this article is to create a simple quick splash screen in a separate thread.

Lot of code is available to build fancy looking splash screens with lot of animation. I have not spent time adding animation, etc. I just took a simple picture for splash screen over which a status message is displayed along with the progress bar in marquee mode. **A good thing which I learnt was to make the label’s background transparent when overlaying over the picture box control.**

Additionally I have created the splash screen in a separate thread and calls for updating the splash screen status are thread safe using BeginInvoke. I have seen people preferring splash screen in a separate thread and some in the main thread. In my opinion, it's better to create it in a separate thread and expose some functions which can be called by the main thread to show/hide/update the splash screen.

Background

The code is simple using threading concepts in C#. The code has been compiled using Visual Studio Express edition 2008 compiled with C# 2.0. One can use these files in Visual Studio 2005 by creating a blank solution and then selecting Windows application in application type.

Using the Code

I have created the following functions for exposing the Splash Screen functionality in the main form. All these functions are static functions, meaning you would not require an instance of the class to be initiated for using it:

* Hide   Copy Code

public static void ShowSplashScreen()

* Hide   Copy Code

public static void CloseSplashScreen()

* Hide   Copy Code

public static void UdpateStatusText(string Text)

* Hide   Copy Code

public static void UdpateStatusTextWithStatus(string Text,TypeOfMessage tom)

I made a static class named SplashScreen which is basically a wrapper around the Splash Screen form. To modify the look and feel of the splash screen, the user can editSplashScreenForm form and add more functionality to it. The SplashScreenForm exposes functions to update the status text on the form.

The code below is used in the constructor of the MainForm where I initiate the Splash screen and hide the form.

Hide   Copy Code

this.Hide();

Thread splashthread = new Thread(new ThreadStart(SplashScreen.ShowSplashScreen));

splashthread.IsBackground = true;

splashthread.Start();

The code below is used to make the label’s background transparent on a picturebox control. This is used in the SplashScreenForm form.

Hide   Copy Code

this.label1.Parent = this.pictureBox1;

this.label1.BackColor = Color.Transparent;

I made two functions to update the status on the splash screen. I have set the default color for label text to green and the user can call update using the function:

Hide   Copy Code

public static void UdpateStatusText(string Text)

I also extended the update function to change the font color based on type of status message.

* Green for success
* Yellow for warning
* Red for error

So I create an enum structure as below for the public static void UdpateStatusTextWithStatus(string Text,TypeOfMessage tom) where I change the color of the label.

Hide   Copy Code

public enum TypeOfMessage

{

Success,

Warning,

Error,

}

Another important thing to keep in mind is that the user should not be able to close the application when splash screen is active. So I create a flag (bool CloseSplashScreenFlag) which is checked whenever the splash screen is sent a message to close by trapping FormClosing event. Hence a simple call like this.close()would not close the form. One would require public static void CloseSplashScreen() function to be called which sets the flag and then close the splashscreen.

The code below is for handling the FormClosing event where if the flag is set to false, it bypasses the close event.

Hide   Copy Code

private void SplashForm\_FormClosing(object sender, FormClosingEventArgs e)

{

if (CloseSplashScreenFlag == false)

e.Cancel = true;

}

Points of Interest

1. Making label’s background transparent over a picture box control
2. Handing close event on the splash screen
3. Exposing static functions to call splashscreen