

RE: WPF vs WinForms

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.framework.windowsforms/2008-01/msg00244.h>

- *From:* Robert <Robert@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>
 - *Date:* Mon, 21 Jan 2008 07:29:03 -0800
-

"Keith P" wrote:

I'm a huge fan of WPF, so I'll throw out my impressions of it:
The primary advantage to it is that it doesn't sit on GDI like WinForms, so you've got a vector-based UI that supports virtually any resolution, is accelerated by the 3D card (GDI is accelerated but not to the extent that vector-based systems can). It's a very declarative tech where most of the UI eye candy is written on the page by the designer independent of the coder. Also, the databinding is done in such a way that if you connect the property of one control with another via an event handler (i.e. declare that the color of a control's background is Blue if myButton.IsPressed, your control's background will change to blue when pressed (and back to its original color once the condition is false)...all the magic of DependencyProperty. This means a TON of stuff that would have traditionally required writing event handlers and having several code paths for buttons, etc., is all done inside the WPF code automatically. This is also true for asynchronous databinding. In a nutshell, if you say "IsAsynchronous=true" for your datasource declaration, the dependency property it's bound to listens for the OnDataBindComplete(or something like that...can't recall it right now), and automatically updates from the proper thread while maintaining responsiveness. And from a flexibility standpoint, its styling facilities are a radical departure from WinForms. WPF apps have a logical tree (of buttons, grids, etc) that have no inherent look to them; they just have functionality (a button can be pressed, grid can hold stuff, a textbox has a Text property). But they can look like whatever you want to define their template as. So for example, if you want to draw a tab control, you could implement it as a radio button group where each tab is a radiobutton because a tab control logically functions like a radio button group.

I really wish the WPF cheerleaders would shut up until they have produced a serious application. Yes we know it does really great things with 3D and pictures and you can spin stuff around and make a time line animation, but as

RE: WPF vs WinForms

far as a serious business application goes, that stuff means nothing. If Microsoft wants to help give us a DataGrid (not the grid that it comes with) one that actually holds data. I don't think they have any intention of building one! Give us a toolbar that easy to use instead of the crap thats in it. Give us a Menu that you don't have to edit in Xaml. In my opinion WPF will not be ready for prime time until you able to build an application without touching xaml. PLEASE DON'T TELL ME HOW MUCH YOU LIKE XAML AND HOW ITS THE BEST THING SINCE SLICED WHITE BREAD. The truth is the community is going to be split up into two categories. Developers and Designers. Developers will be able to handle the xaml and probably create some pretty ugly apps along the way. But the designers will be people adopted from the adobe world and are not used to typing anything to create any graphic look. I'm just telling you the way it is. I hope Microsoft is listening because you damn sure can't talk to any of them in person.

.