

PowerBuilder 12 Overview

The New Generation of PowerBuilder...
A Bigger, Better Builder

Sybase PowerBuilder 12 is a true technological revolution. Here's why we believe it's arguably the biggest breakthrough since its inception.

Please note in this discourse, we are discussing a beta version of a product, therefore features and capabilities are subject to change.

INTRODUCTION

Sybase PowerBuilder 12 represents a new generation for the beloved, proven, and time-tested development tool. Yes, it's that amazing. It will still have a familiar development environment that is the fastest and easiest way to build applications. And, all you need to use it is your PowerBuilder skill-set. However, this release goes where no other release has gone before. Blending the foundation of Microsoft's Visual Studio Shell and the speed and prowess of PowerBuilder and DataWindow® technologies, PowerBuilder 12 is our throw down for .NET, significantly simplifying development for .NET platform.

So what's it got under the covers? Features and capabilities for PowerBuilder 12 include:

- A new infrastructure, built on the Visual Studio Isolated Shell
- Front-to-back WPF
- Fully managed code @ runtime
- WPF DataWindow
- WCF support
- Open and native RDBMS support as well as access to virtually any DBMS via ODBC, JDBC, or OLE DB

Let's take a high level look at the key new capabilities and features found in PowerBuilder 12.

POWERBUILDER AND THE VISUAL STUDIO SHELL

What is the Visual Studio Shell?

In 2007, Microsoft announced it was making the Visual Studio Shell freely available for use by software vendors. The Shell is a streamlined version of the Visual Studio IDE designed to be used much like the Eclipse development environment, with a "plug and play" infrastructure for custom development tools. The key benefit of the Shell is that it allows developers to quickly build their own development components or stand alone development tools without having to build an entire IDE. By leveraging the Visual Studio Shell, independent software developers can significantly reduce their development time by focusing on their own core functionality instead of building and maintaining an underlying framework.

There are two modes of the Visual Studio Shell: Integrated and Isolated. The Integrated Mode allows applications built upon it to automatically merge with any other editions of Visual Studio installed on the same machine. In other words, tools developed in the Integrated Mode install in the same application environment and would be considered a plug-in. On the other hand, the Isolated Mode facilitates the development of applications built with the Isolated Shell to run independently in their own IDE. Essentially, the Isolated Mode provides the option to customize, configure, and rebrand the Shell of Visual Studio as part of the development environment in which it's being used.

PowerBuilder and the Visual Studio Isolated Shell

Since PowerBuilder is primarily used for development of applications for Microsoft's Windows platform, it is logical for PowerBuilder's roadmap to be closely aligned and in tandem with current and future Microsoft technologies. By building PowerBuilder 12 on the Visual Studio Isolated Shell, PowerBuilder is in technology and capability lock step with the Visual Studio Shell. PowerBuilder is using Visual Studio's Isolated Shell for its baseline infrastructure, thus enabling PowerBuilder to leverage all of the functionality of Visual Studio inside of PowerBuilder, and freeing up our engineering team to focus on building and enhancing differentiating technology, such as the DataWindow. This means that we can focus on making PowerBuilder do what it does best: abstracting development for .NET and getting the job done faster, better.

Sybase's fresh, new version of PowerBuilder—with the famous PowerBuilder productivity, yet built with the Visual Studio Shell framework, is not going to become a plug-in for Visual Studio. PowerBuilder is and will remain a separate tool from Visual Studio; yet, the two IDEs can and will coexist on a developer's machine. Furthermore, use of both IDEs is not required for application development. PowerBuilder developers will not be required to use or buy Visual Studio—they will continue to develop inside of PowerBuilder and enjoy that familiar environment and its unparalleled productivity.

By fusing the Visual Studio Shell within PowerBuilder, Sybase is leveraging both core Visual Studio and core PowerBuilder functionality to provide a best of breed development tool that is the most productive tool on the market today, and into the future. Furthermore, because we are using the Shell as our infrastructure, Sybase can easily leverage and incorporate new capabilities into future releases of PowerBuilder as Microsoft adds functionality to the Visual Studio Shell. This includes things such as mobile targets, modeling, i.e., M/Oslo, Team Foundation Server functionality, as well as Silverlight.

What does the Visual Studio Shell bring to PowerBuilder?

The Visual Studio Shell provides a significant set of new functionality for PowerBuilder. In addition to the core IDE features such as commands, tools, and window framework, the Shell also includes a rich set of component features that PowerBuilder 12 takes advantage of, including the WPF Designer. Additional features that the Visual Studio Shell brings to PowerBuilder include:

- Full featured PowerScript editor that supports color-coding, collapsible sections, bookmarks, and more
- Robust autoscripting
- Dockable and autohide panes

POWERBUILDER AND WINDOWS PRESENTATION FOUNDATION (WPF)

Aside from productivity, the user experience is really what drives developers, and IT managers for that matter, to choose a development tool. Building an application that is easy to use, simple to navigate, and makes the end user's job easier is the goal. A key capability of PowerBuilder 12 is the ability to develop native WPF applications inside the new, Visual Studio based IDE.

Benefits of Windows Presentation Foundation

WPF focuses on the user experience (UX.) It empowers developers to build applications with a rich set of features and controls. WPF development focuses on various aesthetic aspects of Windows programming, including unifying application services from user interfaces, 2D and 3D drawings, fixed and adaptive documents, advanced typography, vector graphics, raster graphics, animation, data binding, audio, and video.

Ultimately, PowerBuilder 12 brings these rich features of WPF to the PowerBuilder IDE, and to the PowerBuilder developer, making it a PowerBuilder experience. Moreover, the unmatched DataWindow has been completely rewritten in C# and natively supports WPF.


So what does this mean for you, the PowerBuilder developer? Having native support of WPF in PowerBuilder brings these benefits:

- Declarative Programming
- DirectX for UI rendering through hardware resources, significantly improving performance
- Powerful and modern UI capability
 - 3D graphics
 - resolution independent images
 - animation
- Simplification of the UI creation and the ability to separate the UI creation and application logic
- Deployment of applications as managed code
- Quick transformation of existing applications
- Full .NET interoperability in the UI and PowerScript

Migration of Existing PowerBuilder Applications to WPF

Microsoft has publicly stated they will continue to support Windows Forms, Web Forms, and Win32 well into the foreseeable future, but most enhancements will be focused on technologies such as Windows Presentation Foundation (WPF) and Silverlight. With that being said, Microsoft deems "user experience driven design" as the future of development. WPF is a key pillar to fulfill this goal of user experience driven design for line of business applications.

Like that famous Car Rental Company, we try harder. We know that there are hundreds of thousands of PowerBuilder developers who have built nearly as many business applications that have been developed over the past two decades, most of which are still running. We know that new technologies are compelling and exciting, and



technology decisions are sometimes made in haste because of hype. But, like you, we also know that the reality of business is that every decision must provide core value and a return on your investment. That's why PowerBuilder 12 includes a migration utility to assist developers in bringing their Win32 applications forward to become WPF applications. No other vendor is doing this! Since Win32 and WPF are vastly different technologies, not all functionality will be easily migrated, and customers may need to re-factor some Win32 code that does not exist in WPF. The migration utility found in PowerBuilder 12 identifies unsupported code and will assist developers with re-factoring the old code. Once migrated, PowerBuilder 12 enables customers to easily enhance the UI of their applications with the fresh, new look of WPF.

This migration utility will be part of the new WPF based IDE. Again, our goal is to migrate as much code from Win32 to WPF as possible, and then users can refactor applications to move forward. It should be noted that once Win32 code is migrated to WPF, the original Win32 source code will no longer be backward compatible. It is also important to note that migrating from earlier versions of PowerBuilder to the Win32 based PowerBuilder 12 IDE will be the same as it ever was, a simple, straightforward migration.

MANAGED CODE

Managed code provides a significant degree of deployment flexibility because it's inherently more secure than unmanaged code. Many organizations also require strict usage of managed code not only for security, but also standardization. PowerBuilder 12 will provide fully managed code at runtime, rendering PowerBuilder applications intrinsically more secure and ultimately more convenient to develop. The new PowerBuilder .NET IDE is capable of deploying a .NET managed code application.

WINDOWS COMMUNICATION FOUNDATION (WCF)

WCF is a set of Microsoft technologies designed to build connected systems and services, such as SOAP, that target the .NET framework. PowerBuilder 12 will support WCF enabling applications to consume next generation Web Services.

OPEN DATABASE SUPPORT

OK, so this may not sound exciting. But it is. It's the reason that PowerBuilder came into being; PowerBuilder made it easy build extremely user friendly applications that could access massive amounts of data, wherever it was stored, and sort it, filter it, modify it, update it, and display it in a variety of professionally and visually exciting formats. And, by easy, we mean 5-10 lines of script and declarative programming. PowerBuilder is still the best tool for building data-driven applications, quickly. And, since most organizations run heterogeneous databases, PowerBuilder continues its tradition as an open tool, and version 12 will ship with enhanced integration for the latest RDBMSs, be it SQL Server®, Oracle, or Sybase.

CONCLUSION

There's no conclusion for PowerBuilder—only evolution! PowerBuilder 12 is designed to keep developers as productive as possible, all the while creating robust and visually appealing business applications. A truly innovative advancement for application development, PowerBuilder 12 combines a time-tested, proven tool with hybrid technology to transform PowerBuilder as a next generation tool that continues solves real business problems today and in the future.