

BETTER SOFTWARE

BUILD A BRIDGE
And bring agile
to your enterprise

PAGE 26

FOR WHAT IT'S WORTH
The true value of testing

PAGE 32

The Print Companion to

StickyMinds.com



THE MANY LAYERS OF AJAX

*Techniques to Sweeten the
User Experience* **PAGE 18**

A Look at FinalBuilder

by Adam White

I'm not used to finding new tools that become an integral part of my daily work, but I've come across an exception. VSoft Technologies' FinalBuilder (see the StickyNotes for a link to the Web site) is marketed as an automated build and release management tool, but its functionality extends well beyond simple build automation. Think of FinalBuilder as a really great standalone visualization tool for Ant scripts or batch files with a backend workflow engine.

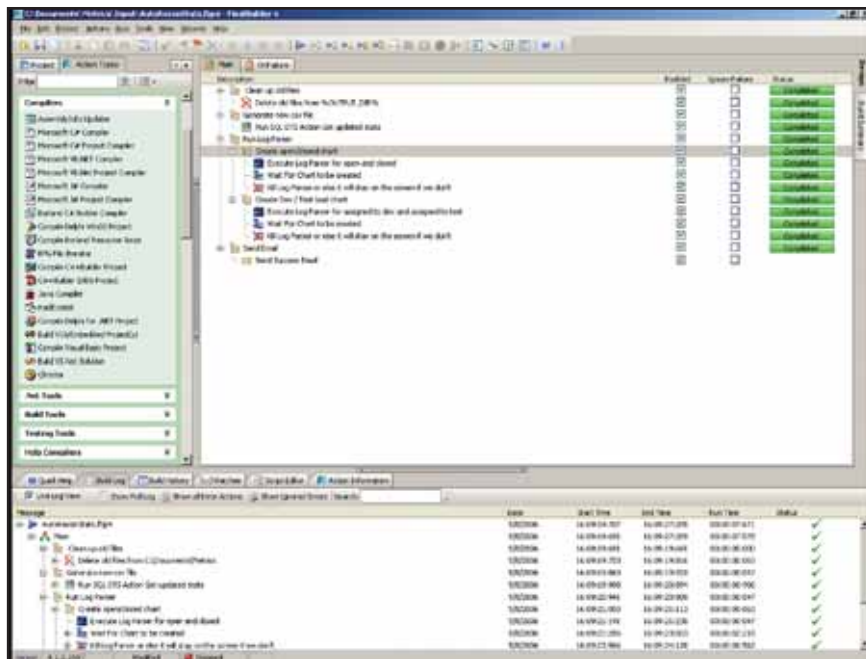
Anatomy of a FinalBuilder File

A FinalBuilder file is made up of Action Lists. Every project by default has a Main and an OnFailure Action List. ActionLists contain actions, which define everything you do and provide the source of ultimate power and flexibility in FinalBuilder. There are built-in actions for almost everything you would want to do—access source control, create/delete files, run unit tests, build installers, and interact with IIS, just to name a few. You can provide variables to actions, allowing you to dynamically change commands at run time. Actions can also be enabled or disabled based on input.

More Control, Less Confusion

FinalBuilder's user interface and logging make troubleshooting much easier to deal with than using batch files. Its user-focused design gives a lot of control without a lot of confusion. For example, my test-engineering team had problems finding efficient solutions to two problems: bootstrapping our automation with something understandable and maintainable by all team members and sending out various metrics to teams in regular intervals.

FinalBuilder solved these problems by providing a rich user experience that made maintenance of our workflow painless. It gave us the ability to change variables in one central location, getting us out of batch file hell and allowing us



to switch projects on the fly using variables on the command line.

(Warning! As with any automation, be careful not to get away from manual testing—especially when an item is important to your customers.)

Asynchronous Actions

FinalBuilder supports asynchronous actions so you can spawn off many things at once instead of executing them serially. This gives you the option to tweak your build or task to make it faster. When we began using FinalBuilder, it took us a day or two to get the builds going after a release was sent out. Our product had grown and changed since the early days and needed to be updated. Thanks to the power and flexibility of FinalBuilder, it now takes us about ten seconds to update the build when we branch/release again.

Over the past few years we have run into some headaches when it comes to our production build process. Both of my company's products have a mix of .NET, unmanaged C++, and Linux code. We need to build specific versions of

Windows and Linux code apart from building the whole product. The conditional statements featured in FinalBuilder make this very easy, from both an implementation and visual point of view.

Flexibility

So what happens when technology changes and the build process needs to be upgraded? We've encountered two situations regarding this. One was with source control, and the other was with the development environment. Currently we have two source control systems—Source Gear Vault and SVN—but we are soon moving everything to SVN. FinalBuilder has built-in support for both of these source code systems, as well as others like the Visual Studio Team system.

It took about ten minutes to switch the build process from Vault to SVN—simply find the Vault actions and replace them with SVN actions. The support for “well-known” third-party applications gives FinalBuilder ultimate flexibility, because it is (almost) agnostic to what tools you use in your environment. If FinalBuilder doesn't have built-in support,

you can always harness the power of the workflow engine using command line calls. If the application you are trying to use doesn't have a command line, then

of the product with your own custom actions. A newsgroup is available where people can post their own custom actions for use by others (see the StickyNotes for

note is that an individual license is free for Microsoft MVPs. If you go the software assurance route, you will get all updates for free.

The support for “well-known” third-party applications gives FinalBuilder ultimate flexibility, because it is (almost) agnostic to what tools you use in your environment.

you'll need to find another solution.

Another example of the flexibility of FinalBuilder is in its handling of development environment changes. We recently upgraded our development environment to VS 2005 from VS 2003. Upgrading the build was a simple, ten-second task of opening the Visual Studio actions and choosing “Use VS 2005.”

We also use FinalBuilder's built-in NUnit command. When we execute the unit tests using this command, we then use the Transform XML function to apply a stylesheet to the unit test output XML, so we have a pretty HTML file that gets sent out with an email showing all the steps that the build process executed. We also do general administration of machines and servers using the built-in actions. Nightly file backups are extremely easy using the built-in Windows OS actions.

Support

Over the past few years I have found a few issues with the product, but the support staff always responds promptly (whether the issue was my own fault or not). They sometimes provide interim builds to help work around major issues. There are online forums and some of the developers have blogs in which they talk about upcoming features. They also explain items that might not be outlined well in the beta documentation.

FinalBuilder comes with an IDE that you can use to extend the functionality

a link). At my company we've written custom actions to do things such as control VMware ESX servers and virtual machines, as well as other little utility applications like log parser.

Building on FinalBuilder

FinalBuilder's lack of support for Linux causes us to have a separate build process that runs using GCC on a Linux machine. We run builds multiple times a day depending on where we are in the release cycle. It would be great to access the build machine—via a Web-based interface similar to CruiseControl, for instance—in order to do this. We are also experimenting with a CruiseControl .Net FinalBuilder plug-in as a way to run a stripped-down version of the production using FinalBuilder's conditional logic (see StickyNotes for details).

FinalBuilder Specifics

FinalBuilder has a free, thirty-day demo downloadable from the Web site. You can find out more about the two versions of the product, standard and enterprise, on the VSoft Technologies Web site. The licensing model is flexible depending on your needs—you can have a single named user, concurrent users, or site license, and you can opt out of software assurance. An individual license without software assurance costs about \$345 (US). An interesting side

There is a built-in mechanism that checks for updates, which comes in handy for staying up to date with the latest actions. It also helps you make the most of your software assurance contract. The installation is straightforward and quick. For information on system requirements, check the FinalBuilder Web site. **{end}**

Adam White is the manager of test engineering at PlateSpin in Toronto, Ontario. Adam has been testing software as a career for five years and testing the theory of everything for many more. Adam thanks James Bach, Michael Bolton, and Neetu Harish for their feedback on this article.

Sticky Notes

For more on the following topics, go to www.StickyMinds.com/bettersoftware

- FinalBuilder Web site
- FinalBuilder newsgroup
- CruiseControl plug-in

