



Quarterly Spin



Volume 1, Issue 2

April 1, 2007



New Shower Complete with Cat Option By James Roome

I had been mumbling around the office that our cat was missing. The night before, we couldn't find it.

When I got back from work we started our all-out-search. First we were thinking the cat had curled up and died somewhere in our vast array of junk in the basement, which initiated a long overdue and much procrastinated basement junk reorganization. However, no cat, dead or alive was found.

Then following the theory that maybe our cat had made a run for freedom outside, I braved the freezing subzero temperatures and scanned for cat tracks... nothing.

Well, at that point, I'm ashamed to say I was already calling it quits, and it wasn't until hours later when I was strategizing with my wife

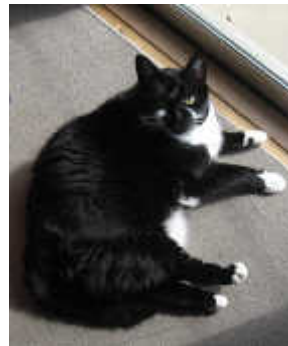
about what we would do with all the space in the basement once we got rid of the cat litter boxes, when a revelation hit me!

A memory, a memory of when I was a child, when one of our cats, Ginger, had inexplicably gone missing. It wasn't until days later, we could hear Ginger meowing faintly, We would rush in the direction of Ginger's weak cries for help, but were unable to locate her.

After a few hours of searching we figured it out, there had been work done on the house recently and poor Ginger had been sealed in under the floor boards!

Blammo! Back to 2007. OMG! OMG!

We are in the process of remodeling our bathroom.

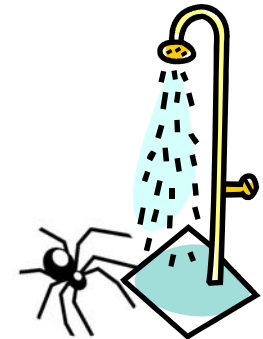


Sure enough, the cat had gone into the (then) ripped out shower and had crawled down beside the bath in the adjacent bathroom. The cat had been dry walled inside the wall for 3 days!!!

Fifteen minutes later, covered in dust and looking quite irritated, we got that darn cat out!

So I guess we'll be stuck with those litter boxes in the basement for another ten years.

See New Shower page 3



Inside this issue:

New Shower Complete with Cat Option 1

MS Build with NUnit 1

Arrogance Leads to Bad Programming 2

Spiderlogic.com Gets Facelift 2

Bars Going High Tech 4



MS Build with NUnit By Geoff Lane

I've written about Unit Testing and Build Automation in the past, but mostly dealing with Java projects and tools (because I usually write about things I'm working on at the time). Well, I've started a .NET project for the first time in a while so I want to solve some of the same problems in this environment.

Why MSBuild?

In the Java world, the natural choice for automation is usually Ant. NAnt was created for the as a work alike for the .NET platform and is a very good tool. I've built custom extensions for database automation in the past and used it successfully. So why would I consider MSBuild? With the advent of Visual Studio 2005,

Microsoft included MSBuild as the build tool for its projects. All of the project files generated by Visual Studio are MSBuild files in disguise. This makes it really easy to leverage the project files to chain together a nice automated build.

See NUnit page 3





Arrogance ... Leads to Bad Programming

By Dan Miser

Surely, you've heard Larry Wall claim that the best traits a great programmer can possess are Laziness, Impatience, and Hubris.

I think it is brilliant analysis on Larry's part to tie all of that together, and I also believe it to be true.

In contrast, let me explain why the vice of arrogance can't actually be turned into a virtue for a programmer. I'll even argue that arrogance in a programmer is inversely proportional to their talent. Now, before the hate mail starts pouring in, I'm talking about extremes here. There is a fine line between self-confidence and arrogance. I think it is vital for a programmer to have an abundance of self-confidence. Self-confidence is good. Programmers can, and do, help shape the world. They build systems that catch criminals, thwart terrorism, pilot airplanes and rockets, map DNA, analyze stock markets, and many other potentially world-changing things. It's only natural to look at what you've created out of thin air and be proud of it, and by extension, gain self-confidence.

Arrogance leads you to not question your code. The more talent and experience that I obtain as a programmer, the more I look at my code as the source of the problem first. Most of the time, the problem

is in my code. The times that it's a bug elsewhere, I've taken all of the basic steps to gather a test case to prove that it's not my bug. From there, I can either write a workaround, and/or report the bug to the responsible party in a way that allows them to fix the problem easily.

Arrogance leads you to rely solely on yourself. After all, if you truly are that good, why would you want to take advice from anyone?

As a programmer, I relish the opportunity to learn. It doesn't matter where the knowledge comes from. I'm just thankful to gain it. Arrogance closes your options, since the Arrogant Programmer refuses to accept that there are others with an alternative approach.

Arrogance leads you to write "tricky" code. This make perfect sense, since it clearly demonstrates the superior intellect and raw genius of the Arrogant Programmer. Documentation? Comments? Meaningful identifier names? Pshaw. They're for mere mortals. The Arrogant Programmer knows the ins and outs of every system, subsystem, and method, and knows that a call could never fail because it has been given life by the Arrogant Programmer. And heaven help the next programmer who dares disturb us to ask why they did such things!

Arrogance is most likely rooted in a false confidence. The Arrogant Programmer may be afraid of not knowing everything, and even more afraid that others will find out that they don't know everything. As a result, the typical mindset is to tear code and people down, instead of building themselves up. You should love working in a team full of great coders. If you want to rise to the top of the top (in anything, IMO), you need to be pushed and challenged. If you find yourself in an environment where you have others keeping you on your toes, it spurs you on to be better than you were before.

So, let's all strive to live by this motto: Friends don't let friends program arrogantly.

Check out Dan's Blog at:

distribucon.com/blog



Spiderlogic.com Gets Facelift

Spiderlogic.com has recently been given a facelift. The new design gives a more international feel. As a company with multiple offices in different countries it is critical that we provide a look which captures our organization.

The "new look" emphasizes the global nature of SpiderLogic and our commitment to an open and communicative workplace.

Our visitors can still access many of the same features such as the Spider Portal and Planet Spider Blogs

We continue to post our job listings as well as technology specific white papers.

Take a few minutes and browse the new site.



Check out our new website

www.spiderlogic.com

N Unit continued from pg. 1

Why NUnit?

Microsoft is a true believer in Not Invented Here and as such they created their own Unit Testing framework that looks and acts exactly the same as [NUnit](http://nunit.org). (<http://nunit.org>)

The MS XUnit framework only comes with the higher-end Team System versions of Visual Studio. In addition it is very hard to deploy the DLLs to build systems. The only way to get the proper DLLs installed is to install the full Visual Studio. That seems like a really big problem to me. With NUnit you can include the needed DLLs in your version control system and reference them in the project which makes it very easy to build and test code and use a Continuous Integration process.

How to Integrate MSBuild and NUnit

Here's the good news, all of the work has been done for you. The good people at [Tigris.org](http://tigris.org) (the creators of Subversion) have created a series of [MSBuild Tasks](http://msbuildtasks.tigris.org) (<http://msbuildtasks.tigris.org>)

Their hard work and the simple script below should get you started with a build that can run your tests as well.

The "Test" target shown in the example below uses the NUnit task to run all of the NUnit Tests found in the DLL. There are a number of different ways to create Item-Groups ins MSBuild, all of which are odd and confusing. If anyone has any ideas on what the best practices in this regard, let me know.

One final plug for my favorite Visual Studio plugin, [ReSharper](http://www.jetbrains.com/resharper).

(<http://www.jetbrains.com/resharper>)

It includes support for NUnit. It will help you create NUnit Tests and then lets you run them and debug them from within your IDE. It's almost like you have a real IDE all of a sudden!

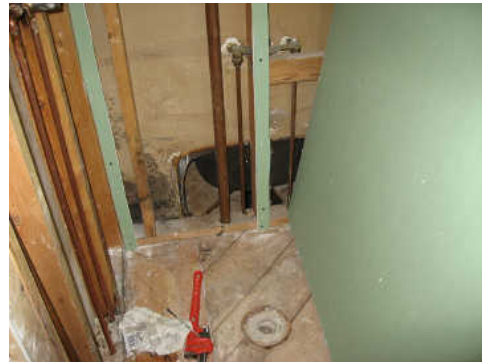
Check out Geoff's blog at zorched.net



New Shower continued from pg. 1



Kitty trapped behind drywall



Post rescue shambles

Example Build File

```
<?xml version="1.0" encoding="utf-8"?>
<Project DefaultTargets="Build" xmlns="http://schemas.microsoft.com/developer/msbuild/2003">
<!-- Import the MSBuild Tasks -->
<Import Project="$(MSBuildExtensionsPath)\MSBuildCommunityTasks\MSBuild.Community.Tasks.Targets" />
<PropertyGroup>
<Configuration Condition=" '$(Configuration)' == '' ">Debug</Configuration>
<ClassLibraryOutputDirectory>bin\$(Configuration)</ClassLibraryOutputDirectory>
<ProjectDir>ProjectName</ProjectDir >
<ProjectTestDir>ProjectNameTest</ProjectTestDir >
<ProjectFile>$(ProjectDir)\ProjectName.csproj</ProjectFile >
<TestProjectFile>$(ProjectTestDir)\ProjectNameTest.csproj</TestProjectFile >
</PropertyGroup>

<!-- Build projects by calling the Project files generated by VS -->
<Target Name="Build">
<MSBuild Projects="$(ProjectFile)" />
<MSBuild Projects="$(TestProjectFile)" />
</Target>

<!-- Run Unit tests -->
<Target Name="Test" DependsOnTargets="Build">
<CreateItem Include="$(ProjectTestDir)\$(ClassLibraryOutputDirectory)\*.Tests.dll">
<Output TaskParameter="Include" ItemName="TestAssembly" />
</CreateItem>
<NUnit Assemblies="@ (TestAssembly)" />
</Target>
</Project>
```



SpiderLogic
10000 Innovation Drive
Milwaukee, WI 53226

Phone: 414.290.8015
E-mail: info@spiderlogic.com

Where architects code and coders
architect!



Bars Going “High Tech” By Chris Peterson

I went to visit some friends and we went out to a bar/pub in their area.

It was a new place. The bar had lots of flat screen TVs on the wall along with a nice internet jukebox, and some gaming machines. Behind the bar were fancy touch screen registers and some top shelf alcohol. I noticed on the bottles there were some weird pour caps, nothing I had seen before.

When I asked the bartender about them he explained they had little computer chips in them that talked to the cash registers. The chips would only allow the bottle to pour if the register said it was OK. The chip would also limit the amount of alcohol that was poured to the perfect shot. I did not believe him until he took out a full bottle and tipped it upside down and nothing came out.

He then went over to the touch screen and put in my order and then tipped the same bottle and this time the rum came out. To prove his point, after the pour he tipped the bottle over again and nothing came out again.

The bar itself was a solid dark wood nothing too surprising, but all the tables at the place had a little built in touch screen. I was curious about it so I went over to a table where I saw some people using one and asked them about it. They told me, from the screen they could place drink orders or call over a waitress. Also they could use the screens for trivia, games, TV, internet, or even a little text chat with the people at another table. they even said these touch screen were spill proof. Although I did not test this feature for the pubs sake I hope it is true.

OK, so this place does not exist yet, but all these concepts are coming soon.

In England some computer science kids who were tired of waiting for a waitress at a busy pub and did not like standing up at the bar and waiting, thought it would be cool if they could just place and order from there table. They came up with the idea of the touch screens and the ability to do all sorts of things including ordering drinks for other tables, and yes even chatting. Currently the students have one system running at a pub in England where it is being product tested.

So are the days of free drinks gone? Who knows, but somebody has to pay for all these improvements.

Checkout Chris' blog:
chriscpetersonblog.blogspot.com

