

Writing Better Code: Using Visual Studio to Improve Your Code Base

Jason Bock

Principal Consultant, Microsoft MVP - C#



Agenda

- What?
 - What's good code?
 - What's bad code?
- Why?
 - Why should I care?
- How?
 - How do I go about writing better code?

The “What?”

- What’s good code?
 - Readable – consistent coding style
 - Resilient – lots of unit tests
 - Maintainable – clean API, easy to refactor
- It’s probably easier to analyze bad code! Let’s look at some examples

What's Bad Code?

```
// query and index are method arguments.  
logger.Debug(string.Format("Query " + query +  
    " was processed at index {0}.", index));
```

What if query contains curly braces?

What's Bad Code?

The fix: Use `Format()` correctly

```
// query and index are method arguments.  
logger.Debug(string.Format(  
    "Query {0} was processed at index {1}.",  
    query, index));
```

What's Bad Code?

```
public static double Power(double value,  
    double exponent)  
{  
    return Math.Pow(value, value);  
}
```

Without unit tests, the bug won't be found fast enough.

What's Bad Code?

The fix: Add unit tests

```
[TestClass]
public sealed class PowerTests
{
    [TestMethod]
    public void CheckPower()
    {
        Assert.AreEqual(64.0, Power(4.0, 3.0));
    }
}
```

What's Bad Code?

```
public sealed class Runner
{
    public void Run() { }
    private void RottingCode() { }
}
```

RottingCode() is no longer needed – it's dead code.

What's Bad Code?

The fix: Remove the unused code

```
public sealed class Runner
{
    public void Run() { }
}
```

What's Bad Code?

```
try { // ... }  
catch  
{  
    // ...  
}
```

Catching the general Exception type is bad

What's Bad Code?

The fix: Catch the expected exception(s)

```
try { // ... }  
catch(IOException)  
{  
    // ...  
}
```

What's Bad Code?

```
public void RouteViaDispatcher()  
{  
    Dispatcher d = new Dispatcher();  
    d.Route();  
}
```

It's not obvious, is it?

What's Bad Code?

```
public sealed class Dispatcher : IDisposable
{
    public void Dispose() { // ... }
}
```

The Dispatcher object isn't disposed.

What's Bad Code?

The fix: Add using

```
public void RouteViaDispatcher()  
{  
    using(Dispatcher d = new Dispatcher())  
    {  
        d.Route();  
    }  
}
```

What's Bad Code?

```
using System.ServiceModel;
```

```
[OperationContract(IsOneWay = true)]  
string Process(string data);
```

WCF one-way operations cannot return a value.

What's Bad Code?

The fix: Return `void`

```
using System.ServiceModel;
```

```
[OperationContract(IsOneWay = true)]  
void Process(string data);
```

The “Why?”

"I've lost the will to live, or
at the very least, debug
my coworker's code."

The “Why?”

- Why should I care?
 - We’re professionals
 - We want to go home on time
 - Less stress
 - Better customer satisfaction
 - More time to focus on “fun” coding aspects
- It’s what we do!

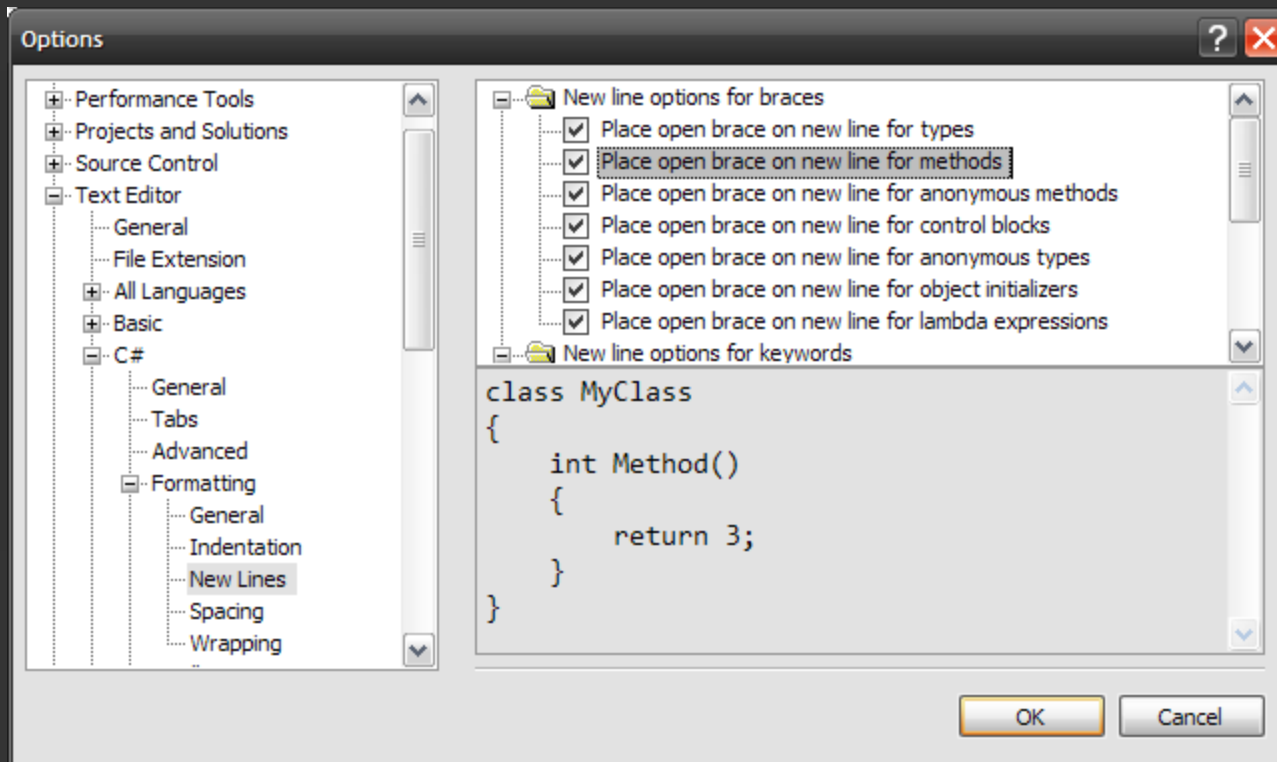
The “How?”

- How do I go about writing better code?
 - Coding standards
 - Unit testing
 - Code coverage
 - Code analysis
 - Code metrics

Coding Standards

- A “contract” that all developers agree to adhere to in code
 - Should be short (2-3 pages, tops!)
 - Should be followed (no `m_lastName` in one spot, and `LastName` in another)
 - Should not cause 100-year wars

Coding Standards



Demo: Supporting Coding Standards

Unit Testing

- What is unit testing?
 - “Unit testing is a procedure used to validate that individual units of source code are **working properly**.” (http://en.wikipedia.org/wiki/Unit_testing, emphasis mine)
 - Starting to be a “just-do-it” on software projects
 - Sometimes, it takes a “unit tests just saved my butt” moment to see how unit tests are effective

Demo: Adding Tests to a Class Library

Code Coverage

- Great, you have unit tests in place...now what?
- How much of your code is covered?
- If it isn't...
 - Do you need it?
 - Do you need more tests?

Code Coverage

- How much coverage is enough?
 - AssemblyVerifier: 100%
 - DynamicProxies: 100%
 - EmitDebugger: 40%
 - ExceptionViews: 0%
 - FileGenerator for Reflector: 0%
 - ExceptionFinder for Reflector: 90%
 - Quixo3D: 95%

Demo: Getting Code Coverage Numbers

Code Analysis

- We all want to follow best practices (and sometimes we come up with our own)
- Following them as a manual process is just too hard
- Enter FxCop in Visual Studio
- (Daunting, complex at first, takes time to get used to)

Code Analysis

- Custom Rules
 - Coding standards that can't be handled using standard VS environment settings
 - “All WCF operations should have one argument and one return value”
 - Runtime rules that the compilers don't know about
 - “A one-way WCF operation should not return a value.”

Demos: Using Code Analysis and Creating Custom Rules

Code Metrics

- Akin to code analysis, VS has the ability to generate metrics from your code
 - Maintainability Index
 - Cyclomatic Complexity
 - Depth of Inheritance
 - Class Coupling
 - Lines of Code

Demo: Discovering Code Metrics

A Disclaimer of Sorts...

- Microsoft's tools
 - Pretty much require a Team edition of some sorts
 - Are not very extensible, documented, or just easy to use
 - There are 3rd-party alternatives
- But even if you don't use Microsoft's tools, I strongly encourage you to follow the principles

At the end of the day...

- A perspective on continually improving code is necessary
 - Automate tools when possible
 - Be open to criticism
 - Care about what you do
 - Do you want clear, consistent, and correct code?

References



<http://thedailywtf.com/>

<http://thedailywtf.com/Articles/Theyre-Useful-In-General.aspx>

References

- Books
 - “Code Complete”, by Steve McConnell
 - “Object-Oriented Design Heuristics”, by Arthur J. Riel
 - “Framework Design Guidelines”, by Krzysztof Cwalina and Brad Abrams

References

- Articles
 - Getting Custom Dictionaries in VS 2008:
<http://blogs.msdn.com/fxcop/archive/2007/08/20/new-for-visual-studio-2008-custom-dictionaries.aspx>
 - Writing Custom Rules:
 - <http://www.binarycoder.net/fxcop/>
 - <http://richardsbraindump.blogspot.com/2008/02/how-to-create-wcf-custom-code-analysis.html>

References

- Articles (con't)
 - “Why Does FxCop Warn Against catch(Exception)?”,
<http://blogs.msdn.com/fxcop/archive/2006/06/14/631923.aspx>
 - Interesting Comment on Code Analysis and Phoenix (and why CA2000 was removed):
<http://blogs.msdn.com/fxcop/archive/2008/01/07/faq-which-rules-shipped-in-which-version.aspx>

References

- Articles (con't)
 - Microsoft Source Analysis for C#:
<http://blogs.msdn.com/sourceanalysis/archive/2008/05/23/announcing-the-release-of-microsoft-source-analysis.aspx>

Writing Better Code: Using Visual Studio to Improve Your Code Base

Jason Bock

Principal Consultant, Microsoft MVP - C#

