.NET Gotchas

Venkat Subramaniam

venkats@agiledeveloper.com

http://www.agiledeveloper.com/download.aspx

Code examples from this presentation may be downloaded from the above URL

We will spend most time on code ☺

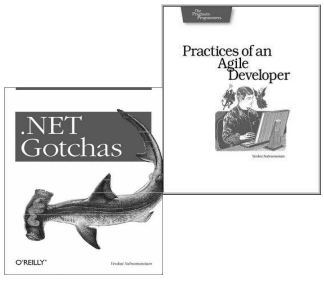
Abstract

Abstract Those of us programming on the .NET framework have come to realize the power and increased productivity that comes with it. Like any development, however, there are things that one should pay attention to while programming on .NET. Are there things in .NET that, if we do not pay attention to, may result in more trouble than it is worth? This session presents Gotchas that a developer needs to know to be productive in the .NET framework. The issues addressed include framework, language, language interoperability, COM interoperability. Most Gotchas are language independent while a few are C# or VB.NET specific. Code examples are presented in C# (download includes examples in VB.NET as well).

About the Speaker *Dr. Venkat Subramaniam*, founder of Agile Developer, Inc., has trained and mentored thousands of software developers in the US, Canada and Europe. He has significant experience in architecture, design, and development of software applications. Venkat helps his clients effectively apply and succeed with agile practices on their software projects, and speaks frequently at conferences.

He is also an adjunct faculty at the University of Houston (where he received the 2004 CS department teaching excellence award) and teaches the professional software developer series at Rice University School of continuing studies.

Venkat has been a frequent speaker at No Fluff Just Stuff Software Symposium since Summer 2002.



.NET Gotchas

- **CLR/Framework Gotchas**
- Visual Studio & Compiler Gotchas
- Language & API Gotchas
- Language Interoperability Gotchas
- GC Gotchas
- Inheritance & Polymorphism Gotchas
- COM Interop Gotchas
- More Gotchas
- Conclusion

Alias Mismatch

- .NET types defined in CTS
- Benefits – no problem of conversion within managed code
- Gotcha – language specific aliases
- Size of types don't match to what you would expect based on familiarity
  - Gets interesting when you call unmanaged code
- Know your types very well

Unhandled Exceptions

- What happens if your application throws an exception
  - If you handle the exception, that's cool
  - What if you don't?
    - Depends on type of application
- Gotcha – displays unfriendly messages
- Provide a handler to take care of all unhandled exceptions (also handle exceptions)

String Immutability & Performance

- Strings are immutable
- All those methods of the String end up creating more strings
- Gotcha – Large overhead of object creation and destruction
  - 100, 000 strings concatenation took 25 seconds while StringBuilder took 0.01 seconds
- Use StringBuilder instead in a performance intense application

.NET Gotchas

- CLR/Framework Gotchas
- **Visual Studio & Compiler Gotchas**
- Language & API Gotchas
- Language Interoperability Gotchas
- GC Gotchas
- Inheritance & Polymorphism Gotchas
- COM Interop Gotchas
- More Gotchas
- Conclusion

Treat Warnings as errors

- Some errors are really not benign
- These must be critical errors
- Gotcha - Computer reports as warning and studio hides it under the rug
  - Trouble waiting to happen
- Set Treat Warnings as errors flag on project
  - Better still modify project templates on your system

Rethrow vs. throw

- You catch an exception
  - Try to handle or
  - Simply log
- You want to send that exception off to the layer above you
- So you say throw ex; where ex is the exception you caught
- Gotcha - What's the origin of this exception?

- Use rethrow instead of throw
  - Don't catch exceptions you can't handle

.NET Gotchas

- CLR/Framework Gotchas
- Visual Studio & Compiler Gotchas
- **Language & API Gotchas**
- Language Interoperability Gotchas
- GC Gotchas
- Inheritance & Polymorphism Gotchas
- COM Interop Gotchas
- More Gotchas
- Conclusion

enum type safety

- enum have make our lives simpler
- They pop up with possible values in IntelliSense
- Gotcha - Gives us an illusion that they are type safe
- Beware of enum type safety – or lack there of

Object initialization sequence

- Your class derives from another class
- In what order is your constructor called and how about initialization of your fields?
- Gotcha - Which language are we talking about?!
- Clearly understand how fields are initialized in the language you are using
  - Also, keep an eye if you mix languages

.NET Gotchas

- CLR/Framework Gotchas
- Visual Studio & Compiler Gotchas
- Language & API Gotchas
- **Language Interoperability Gotchas**
- GC Gotchas
- Inheritance & Polymorphism Gotchas
- COM Interop Gotchas
- More Gotchas
- Conclusion

Optional parameters

- When mixing languages, you can easily get mixed up
- VB.NET supports optional parameters
- C# does not
- Gotcha – what happens if you derived from a class that uses optional parameters
- As yourself if your code is CLS compliant
- Use CLSCompliant attribute to make sure
- Run it through FxCop

Array allocation

- What's the size of your array?
- Gotcha – Which language you are asking about?
- VB.NET uses 0 based indexing, but…
- Know the size of your array, especially if you will interoperate between languages

Quiz Time

.NET Gotchas

- CLR/Framework Gotchas
- Visual Studio & Compiler Gotchas
- Language & API Gotchas
- Language Interoperability Gotchas
- **GC Gotchas**
- Inheritance & Polymorphism Gotchas
- COM Interop Gotchas
- More Gotchas
- Conclusion

Finalization

- .NET provides automatic garbage collection
- What does that do?
- Cleans up the memory
- But what about unmanaged resources?
- That's taken care by Finalize
- Gotcha – What's Finalize executed?

- Do not rely on the Finalize method

Dispose pattern

- It is your responsibility to cleanup an object when it's no longer needed
- What if an object uses expensive unmanaged resource?
- This is where IDisposable comes in
- Gotcha – Do I need Finalize and Dispose?
- Understand the Dispose Pattern and apply it consistently.

.NET Gotchas

- CLR/Framework Gotchas
- Visual Studio & Compiler Gotchas
- Language & API Gotchas
- Language Interoperability Gotchas
- GC Gotchas
- **Inheritance & Polymorphism Gotchas**
- COM Interop Gotchas
- More Gotchas
- Conclusion

Method hiding

- In C++ you override a method typically by marking base and derived method as virtual (though not required in derived)
- Gotcha - What happens if you carry those features into C#?
- What is the effect of the new keyword?
- When should you use it?
- Ask yourself if you would want to really hide methods – think of OO Practices and Principles

Hiding by signature

- The base class has a method that takes say an int
- I need a method that takes a float in derived
- Gotcha – don't go there!
- Understand how you may end up hiding methods and make your code behavior inconsistent
  - Override and overload if needed

Substitutability issues

- Liskov's substitution principle says that an object of derived must be substitutable where an object of base is used without the user knowing the difference
- Gotcha – Inheriting for the sake of convenience can lead to extensibility problems
- Make your code LSP compliant

.NET Gotchas

- CLR/Framework Gotchas
- Visual Studio & Compiler Gotchas
- Language & API Gotchas
- Language Interoperability Gotchas
- GC Gotchas
- Inheritance & Polymorphism Gotchas
- **COM Interop Gotchas**
- More Gotchas
- Conclusion

COM object cleanup

- In C++, you have to worry about AddRef and Release
- In VB, you have no such concerns
- .NET is somewhat in between
- Gotcha – What happens when you let go of a reference to a Runtime Callable Wrapper

- Understand the lifetime of a COM object
- Look at ReleaseComObject
- Understand the perils of not using it and the consequence of using it

.NET Gotchas

- CLR/Framework Gotchas
- Visual Studio & Compiler Gotchas
- Language & API Gotchas
- Language Interoperability Gotchas
- GC Gotchas
- Inheritance & Polymorphism Gotchas
- COM Interop Gotchas
- **More Gotchas**
- Conclusion

More Gotchas

- What have we discussed?
  - Some interesting Gotchas in different areas
- What we have not discussed?
  - Many more Gotchas (over 5 dozen more)
  - Multithreading Gotchas
  - COM related Gotchas
  - How to structure your COM communication

Quiz Time

.NET Gotchas

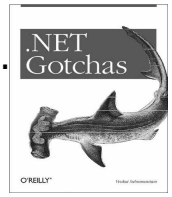
- CLR/Framework Gotchas
- Visual Studio & Compiler Gotchas
- Language & API Gotchas
- Language Interoperability Gotchas
- GC Gotchas
- Inheritance & Polymorphism Gotchas
- COM Interop Gotchas
- More Gotchas
- **Conclusion**

Conclusion

- It is not only important to know how to use the framework/language/API
- We need to know what to avoid
- Very interesting features, but pitfalls as well
- Know your .NET

References

1. .NET Gotchas, Venkat Subramaniam, O'Reilly.



2. Effective C#, Bill Wagner, Addison-Wesley.

3. Microsoft Developer Network (MSDN).
   http://msdn.microsoft.com

4. http://www.AgileDeveloper.com/download.aspx

Thanks!

Please fill out your evaluations!