



.NET Rocks!

The Internet Audio Talk Show
for .NET Developers

With Carl Franklin 
and Rory Blyth

www.franklins.net/dotnetrocks

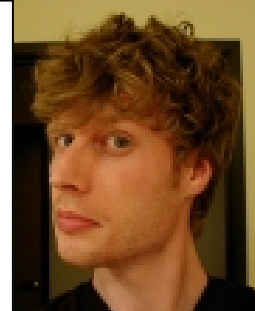
<http://www.dotnetrocks.com/>

Carl Franklin



Carl Franklin and Rory Blyth interview experts to bring you insights into .NET technology and the state of software development. More than just a dry interview show, we have fun! Original Music! Prizes! Check out what you've been missing!

Rory Blyth



Text Transcript of Show # 88

(Transcription services provided by [PWOP Productions](http://www.pwop.com))



Kate Gregory on C++, VB.NET and VSTO
November 08, 2004

Our Sponsors

CoDe

component developer magazine

<http://www.code-magazine.com/>

**DATA
DYNAMICS**



<http://www.datadynamics.com/>



The opinions and viewpoints expressed in .NET Rocks! are not necessarily those of its sponsors, or of Microsoft Corporation, its partners, or employees. .NET Rocks! is a production of franklins.net, which is solely responsible for its content. franklins.net "Training Developers to Work Smarter".

(Music)

Geoff Maciolek: Hey, Rock Heads! The election is over. So listen up, it's time for another stellar episode of .NET Rocks! The Internet, Audio Talk Show for .NET Developers with Carl Franklin and Rory Blyth. This is Geoff Maciolek, here to announce show # 88, with guest Kate Gregory. Recorded live Thursday, November 4th, 2004.

[.NET Rocks!](http://www.franklins.net) is brought to you by [franklins.net](http://www.franklins.net), "Training Developers to Work Smarter", and now offering hands on VB.Net, ASP.Net and C# classes, online at www.franklins.net.

Support is also provided by [CoDe Magazine](http://www.code-magazine.com), "Microsoft Technologies in-depth for IT Managers and Developers", online at www.code-magazine.com.

And now, the man who did not vote for Ralph Nader this time, Carl Franklin.

Carl Franklin: Thanks, Thanks, Geoff. Thanks everybody. I am Carl and welcome to .NET Rocks! The Internet Audio Talk Show for .NET developers, and I am sitting here in New London, Connecticut. All by my lonesome this week, and talk about - it going to extremes. We have last week's show, which would be considered hideously obscene by some and to this show, in which Rory is not here, number one, because we are recording on Thursday, and we usually record on Fridays and Rory is usually doing his presentations on Thursday. We are recording on Thursday because we are going tomorrow to Las Vegas to Dev Connections, and so we won't be here, nobody will be here to do the edit and so we are just making little re-arrangements here. Also seeing as how Rory isn't here, and we are doing it on Thursday, we thought, hey we might as well use this opportunity to go back to the original format of .NET Rocks!, where it's just going to be an hour, hour and fifteen minute interview. So, no news of the week, no Weird Wide Webb, no Ask Rory, no Richard-the Toy Boy, no mail this week either, because we didn't really get any, I mean, we are going back to our roots here folks, and tune in next week, when Rory will be here, and we'll be doing a regular show but of course it's just going to be Rory and I talking to the guest, we are not going to do the funny stuff. And not to say that we are not doing any funny stuff anymore, we are, we're doing it

on a new show called Monday's!. And you can listen to that online at Mondays.pwop.com. That's "PWOP" the sound of a forehead slap ".com". And now that we have all that out of the way, I would like to introduce my guest this week, Kate Gregory.

Kate is a founding partner of "Gregory Consulting Limited". She has over two decades of scientific and engineering programming experience in a variety of writing programming languages including FORTRAN, PL/I, C++, Java, Visual Basic and C#. In 1989, Kate finally started using the Internet after hearing about it for years from friends who were already addicted. In early 1995, Kate co-authored a book on "UseNet for Que", kicking off a writing career, that now covers well over 10 books on programming and related topics, including XML, Most recently Microsoft Visual C++ .Net 2003 Kick Start. She is also a stand up instructor teaching Microsoft .Net, XML, C++ programming, Objects Oriented concepts and UML for several training companies and selected clients. Kate's outstanding energy and knowledge on her subject matter have put her teaching in high demand around the world. Her recent programming work is almost exclusively in Visual C++ and Visual Basic.Net. on a variety of projects and typically features XML. She is a founding member of the Toronto .NET Users Group and the Kawartha Software Developers Association, and a member of the Technology and Development Faculty Board for the CDI Corporate Education Services. She is also an adjunct faculty member of the Department of Computer Studies and Computer Science at Trent University, teaching Object Oriented Design and C++. She writes the Visual C++ column at CodeGuru, covering topics such as Managed and Unmanaged C++, migration and integration, InterOp, and more. And if that wasn't enough, she is a regional director for Toronto and MVP and on the INETA Speakers Bureau. Will you please welcome the amazing Kate Gregory! How are you Kate?

Kate Gregory: Hey Carl! Great, great introduction. My Goodness, you made me sound experienced or something.

Carl Franklin: Well you are. And by the way, that's nothing of my doing. I was just reading your bio that you have published and it is pretty amazing. I mean, for guys like us who started programming on the TRS 80.

Kate Gregory: I had a friend with the TRS 80. We used to tease him about it actually.

Carl Franklin: I am sure you did. Trash 80?

Kate Gregory: Yeah.



Carl Franklin: That's what it stands for.

Kate Gregory: My favorite old computer story, I was talking to Mike Blazac who used to head the OFT team about Sim computers, which sort of had to put together yourself for the soldering iron and stuff.

Carl Franklin: Oh, like Heath kit computers?

Kate Gregory: And this 6502 based and little like 6 LED display, very cool. And I was saying that we had one of these in our first house and we put the power supply in a different room because it was so noisy, and that's how we knew it's really our house, because we didn't have to check with anyone before we drill this hole in the wall. And this little pause and Mike says, "I bought mine with my paper route money". That's when I began to realize that I was little bit older than some of the people in this business.

Carl Franklin: Sahil in the chat room wants to know what you eat for breakfast.

Kate Gregory: I don't do breakfast. Coke, Coco-Cola that's my breakfast.

Carl Franklin: Oh my god! Coca-Cola for breakfast?

Kate Gregory: I am an old-style programmer man, I'm a device for converting Caffeine into code.

(Laugh)

Carl Franklin: I love it. That's pretty good. I would have thought, a brainiac such as you would be eating granola or some hot cereal may be with some oat bran.

Kate Gregory: (Laugh) No.

Carl Franklin: Little honey on it, you know, no?

Kate Gregory: Can of Coke, whatever I can find in the fridge.

Carl Franklin: Macaroni and Cheese.

Kate Gregory: Probably tomorrow morning. Yeah.

Carl Franklin: Speaking of PCs that you put together, I remember as a kid looking at the Heath kit catalog and saying, "wow!, that looks really cool, dad, can we do that?", and I had never even soldered a radio together, and I am like, "Dad lets do this". And he goes, no, you know these things are tweaky enough without the added error quotient of some kid sort of putting

them together. Every time I try to make like a model airplane, I would always end up like breaking in half one critical piece that held the whole thing together and the thing would disintegrate. So I can just imagine you go to boot up your Heath kit PC, and oops you missed to solder joint somewhere and the whole thing smokes.

Kate Gregory: Yeah, but it's great for settling arguments later. Like when you can say in the middle an argument as I am married to my business partner and he is more hardware oriented than I am and in the middle of an argument about programming for a Sys dos when you have to write to the video yourself, he goes, well when I built a video card, these are the guys like, ok, we are done arguing, we are going to do it your way.

Carl Franklin: Things were so much more complex back then, than they are now for the average programmer.

Kate Gregory: Oh for sure, I mean, you had to manage everything, you had to literally figure out what address to write to make the pixels light up, the right color.

Carl Franklin: I remember doing some of that. Some assembly language programming which was hard enough on a PC and I can't imagine mainframe assembler or punch cards or any of that stuff. Yikes!

Kate Gregory: I don't look back with nostalgia on any of that. That's for sure, especially not cards, cards are terrible. It's sure is nice now to have something like GDI+ where you can say, draw me this line that shades from red to blue and curves like this. I mean I like that a lot better.

Carl Franklin: All right. And then there are people that say, Oh, you still program drawing lines and stuff. Why don't you use, shwing O back O wackon swagon backeon, I don't know what the stuff is.

Kate Gregory: Well, I saw this Adobe thing for making user interfaces that just look like space ship, and I was like, oh my gosh another tool that someone thinks that they should learn.

Carl Franklin: Right, well enough about the past, let's talk about what you do now, what's your current passion. I mean there is quite a list of them, but - pick one.

Kate Gregory: Well, right now, I am eagerly looking forward to the next C++ and having fun playing with betas of it. But what I'm doing all day long is mostly VB programming this year because



the current version of C++ for .NET is really not for actually using.

Carl Franklin: That's what I hear and I also hear that the next version is going to be completely revolutionary.

Kate Gregory: it's amazing, I make people come to C++ talks, I like literally pull their elbows until they agree to come in and sit in the room. And then they look at me and then they go, but I can read that code, I understand what that's doing. It's very good.

Carl Franklin: So, if the question obviously on everyone's mind is, why C++, why not C#, and why make C++ more C# like and why not just you know what's the diff.

Kate Gregory: Well, I'll tell you the one line that sort of made people tell me they were going to switch back, it's got to be deterministic destruction. So in C++ for 2005, you can take a reference type like SQL connection, and you can create it on the stack as a local variable, you know SQL connection C, and when you hit your close brace, he goes out of scope.

Carl Franklin: And obviously for anyone who didn't know that that reference types get created on the heap and therefore they're out there garbage collected and you sort of lose control of them.

Kate Gregory: Exactly, and there is a dispose pattern but you have to remember to call dispose.

Carl Franklin: And that doesn't always work either too, right.

Kate Gregory: it's not a 100%, you don't know for sure that say you threw an exception in the middle of the block, you will just pull to get skip, you just pull finally, and instead of drowning and brace bracket and stuff, and C++ has been doing this for a living for a long time. It knows how to still make sure it cleans up deterministically, that is when you can say exactly when it's going to happen, and it actually maps C++ destructors to CLR disposes.

Carl Franklin: Wait a minute what is that? Say that again.

Kate Gregory: Some thing like SQL connection doesn't have a destructor, it's probably not written in C++. But what gets generated for you is a call to dispose. And if you write a C++ class and your write a destructor for it, ~ class name, it will get exposed to the rest of .NET as a dispose method. So, it works in both directions, it's a very slick little decision.

Carl Franklin: And my mind is going here about how to sort of integrate that into VB.Net app or C# app. So, could you do some sort of trickery to give an object that you sub class in C++, deterministic finalization.

Kate Gregory: it's not what the class is in, it's what you are working in.

Carl Franklin: Right, that's true.

Kate Gregory: If you are working in C++, you can have deterministic finalization of every thing regardless of the language they were written in.

Carl Franklin: Right, but then of-course if you make a reference type and create it from VB or C# then it's on the heap anyway. So, what's the point?

Kate Gregory: So you've got to live that "Oh, I hope the garbage collector closes my file pretty soon" life.

Carl Franklin: What about like a static some thing that I mean I am just going nuts here. (Laugh)

What about some sort of static method that...

Kate Gregory: Everybody wants deterministic destruction. They didn't think it was possible with CLR.

Carl Franklin: Well not everybody, Chris sells wants it, but not everybody wants it.

Kate Gregory: Well everyone I show it to, goes, oh! Now, how can I have that VB or C#?

Carl Franklin: Right, that's what I am thinking.

Kate Gregory: That's right, so I have to stick my tongue out and say ha, ha you know.

Carl Franklin: Yeah, you can't do it.

Kate Gregory: That's going to be the C++ offering, that's a huge deal especially if you are writing stuff for the middle tier.

Carl Franklin: Here we go. So you write your object in C++ that does that works with the objects that need to be deterministically finalized. And then you set up some sort of communication between the two and there you go.

Kate Gregory: Like writing your middle tier or your data tier or whatever in C++. You probably still got a C++ person in the back room growing their beard out and...

Carl Franklin: That's a scary thought.

(Laugh)

Kate Gregory: ...write your UI in VB or C#, if you want.

Carl Franklin: Write your middle tier in C++, ouch.

Kate Gregory: And control your object lifetime.

Carl Franklin: OK, so how much more complex is the new C++? How much less complex is the new C++ than its current incarnation.

Kate Gregory: Well, it's readable and it's attractive. So there is no double under scores, step one. That alone is worth major bonus marks, right. Then there is things like, some of the double under scores were these extra bonus keywords like there was one called-property. So I could have a function called, get -- I don't know, temperature and I would stick this under score, under score property qualifier on it, and then thousand lines away elsewhere in the file I could have a function that have to be called "set temperature", and I could also stick an under score, under score property on it, and the compiler would work it out and expose that as a read and write property. Now, you actually type the word 'property', an open brace, a bunch of stuff related to getting and setting and a close brace, it's all together in it and sort of it's more elegant and more readable, it's very intuitive.

Carl Franklin: It's good to hear that it's readable because I always called C++ a write only language.

(Laugh)

Kate Gregory: I reserved that for Pearl and before that ATL.

(Laugh)

So if you already know C++, then C++ flash CLI, which is what they are calling. It's technically not a new language; it's a binding of the language to the runtime.

Carl Franklin: How about compatibility?

Kate Gregory: Well, you are not going to take a bunch of C++/CLI code and then compile it with some 10-year-old Borland compiler and run it on a Unix box, right.

Carl Franklin: All right. So it's sort of breaking a little compatibility.

Kate Gregory: Well, yes and no. If you are not using any CLR features, you will be totally compatible and every thing will be fine.

Carl Franklin: And you could use double under scores if you wanted to do.

Kate Gregory: You need a different compiler flag if you want to use the double under score.

Carl Franklin: Ah, okay all right, that's good, that's a good way to do it.

Kate Gregory: Yeah, but so if you want to write like, "hello world" from C++ 101, it looks exactly the same as it ever did nothing changed. But when you are creating things on the managed heap, there is a different operator instead of new, it's GC new. So, obviously if you run that code through your 10 year old Borland compiler it's going to go I don't know what GC new is.

Carl Franklin: Right, right that's cool.

Kate Gregory: But, as long as you are not using CLR features you are still totally compatible.

Carl Franklin: That is pretty cool, yeah, I like that. And it's odd that, I mean it's not odd, it's really cool actually you and Dan Appleman are like hybrid C++, VB people. And so, what is that all about? I mean why not C#, why VB,Net?

Kate Gregory: Well, I got two reasons why not C#. One is that, I am getting older, and I know an awful lot of languages, semicolons and brace-brackets, and it's really embarrassing to forget what language you are in. And so if I spend my day alternating between two different semicolon brace-bracket languages I'd have many, many opportunities to accidentally type things that weren't in that language.

Carl Franklin: Right sure. So you want to make you a clean break.

Kate Gregory: Yeah. So when I am in VB, I know I am in VB. But I still some times put semicolons at the end of my lines. First, you know, you get blue, but, oh yeah, okay right, yeah, I am in VB. You know what the other reason is, that a lot of times I am writing code for my clients to maintain and they are asking me to do it in VB because they believe they can maintain it themselves as little somehow be simpler it is the lot of misconceptions about VB versus C#. And my programs are not that smart or I don't want to spend a lot of money so do it in VB and I am like laughing at them because you know...



Carl Franklin: Yeah, it's silly.

Kate Gregory: ...I have been known to say that C# is just VB with semicolons anyway.

Carl Franklin: Yeah, it's true.

Kate Gregory: There is some little bits of syntax between the two of them.

Carl Franklin: Well the difference is that you can write things that would like an entire if-then structure in one line of code, but come on, is anybody going to really be able to read and decipher that. I love VB.Net because it's readable.

Kate Gregory: That's right and white space is a good thing.

Carl Franklin: White space is good.

Kate Gregory: White space is good.

Carl Franklin: This is no longer 1990, where we got to like take out the white space and all because I have file size. I can make a hundred gig hard drive fit for 50 bucks one of these days.

Kate Gregory: And the other thing is that the different little bits of syntactical sugar that they have tend to come down on the VB side. Like I have been doing a fair amount of Visual Studio tools for office work. And when you call into those methods that are exposed, out of Word and Excel, they might take like 30 parameters and 29 of them are optional. C# doesn't do optional. So, you get to type in the one parameter you really need and then 'type.missing', 'type.missing', 'type.missing'.

Carl Franklin: Right.

(Laugh)

Kate Gregory: And it's worse if it takes it by reference because then you have to actually create like a random variable, set it to 'type.missing' and pass it in by ref. So it's pretty ugly and I don't diff C# for choosing not to support optional parameters, but I'm going to choose VB for all my VSTO work.

Carl Franklin: And it comes down to choice, I mean it's America, right. It's a free world. Use what you want to use.

Kate Gregory: Use what you want to use. And for people who don't already know C++, say people who are coming from Java, I think C# has a ton of appeal for them. It's nice and simple and familiar.

Carl Franklin: Absolutely.

Kate Gregory: Use it. But I already know C++. So if I am going to be typing semi colon, well I am going to do it in C++.

Carl Franklin: Right. Cool. So ASP.Net, is that a part of your life as well?

Kate Gregory: It is, and you can't do that with C++ because of the one file-compiling model. So I do all of that in VB.Net. I'm doing a huge ASP.Net app right now for a big Canadian client.

Carl Franklin: One of the things that I think of reasons to use C++, obviously for the un-managed support when you need things like deterministic destructors and things. But also for doing all those other programs that don't fall in the realm of applications. All the little drivers and things that work with the UI and graphics at a low level. If you are not programming...

Kate Gregory: The seriously geeky stuff where we want to get a little closer to the metal and it's also now really turning into the InterOp language.

Carl Franklin: Interesting, how so?

Kate Gregory: Well, if you want to InterOp from managed to un-managed code, there is three ways to do it from C++ and only two from VB or C# and the third way is of course the fastest of the three. So the way you can only do from C++ is significantly faster.

Carl Franklin: And what is that?

Kate Gregory: It used to be called "It Just Works" InterOp.

Carl Franklin: Oh! Yeah, IJW technology. I remember hearing this.

Kate Gregory: And now it's just called something boring like C++ InterOp, which really doesn't have the same ring.

Carl Franklin: And you know they should really lighten up with the names. I love "It Just Works". That's just so cool!

Kate Gregory: Yeah! And when you show it to the people and they go hey! I know why it's call that now because it's like you include the header file, you link to the lib and you carry on about your life as though it was managed instead of un-managed code.

Carl Franklin: It's very Zen. I love that.



Kate Gregory: And then people are like, where is this scary bit where I have to type a whole bunch of punctuation. No, just call it.

(Laugh)

Kate Gregory: The compiler works it out, it just works, I like that a lot. But now we are being grown up -C++ InterOp, okay.

Carl Franklin: Okay.

Kate Gregory: But it's still the fastest way to get from A to B.

Carl Franklin: I know it's faster obviously it's a little closer to the metal. But what about interacting with COM objects? I know that, the big problem with COM objects and managed code is and especially if you are hooking them in a destructor or finalizer or anything is that you have to call a release COM object all the time to absolutely make sure the COM object is destroyed. Do you have similar issues in...

Kate Gregory: So it's flow, it's really hard to control the lifetime, there is like lifetime and people mismatch. What I'm telling most of my clients to do is if they need to still maintain the COM component as a COM component because typically these people might have like four apps that use it from COM and three apps that use it from .NET. If they can refactor and pull the guts out of the COM component into just an ordinary library, like a C++ DLL if they wrote the thing in C++ in the first place. Then call that from both the COM component and .NET rather than calling COM from .Net. Save themselves an ocean of misery both around performance and around lifetime management.

Carl Franklin: Very cool!

Kate Gregory: So I mean it costs a bit of time to do some refactoring which is always scary because you are touching code that's not broken.

Carl Franklin: But, if you are not refactoring, if you're doing it right from the start and you want to avoid that you should really...

Kate Gregory: That's right. Build it as a library from the first place in treat COM and .NET as peers, have them both called the library.

Carl Franklin: Interesting. So, I'm actually getting very curious about C++, the next generation and think that I might actually check it out. How much of a learning curve from a person who has done a little C++ in the past, kicking and screaming, and is primarily a C# or VB

programmer. What are the kinds of things that we're going to have to know?

Kate Gregory: Well you are actually going to have less learning curve than if you had been part of the Everett, the double underscore generation. Because, for example to make a class abstract you use the incredibly hard to learn keyword "abstract".

(Laugh)

Carl Franklin: I always wondered why we had "must -inherit" in VB, is our VB brain not capable of understanding the abstract word?

Kate Gregory: My favorite is 'shared' instead of 'static'.

Carl Franklin: I mean what was that meeting all about.

(Laugh)

Carl Franklin: No, our research shows that VB programmers will not get 'static'. No, they will not get.

Kate Gregory: But they call it 'shared'. That's so friendly.

Carl Franklin: Yeah. Because sharing is good.

Kate Gregory: Well, this is the language that brought us the 'and' also 'and or else'.

Carl Franklin: Yeah! I know, there are things that even I, Mr. VB don't like about Visual Basic and some of that 'and if', 'not is' 'nothing', 'else'.

(Laugh)

Carl Franklin: Stuff can really get crazy.

Kate Gregory: 'Is not' I think is a new keyword now.

Carl Franklin: Yeah! 'Is not' and 'and else'.

Kate Gregory: Somebody blogged about Int, which I think, contains like an apostrophe right, so it's not a very valid VB keyword.

Carl Franklin: How about, 'something', how about 'if very if' object reference 'is something'. (Laugh) I like that.

Kate Gregory: Sure. I worked with a FORTRAN programmers who, when he went into C, made all these macros so that he could use this FORTRAN. ie and so forth, comparison operators instead of typing...

Carl Franklin: I have no idea with that's. I did take a FORTRAN class in college but that's about it. I really don't know what that is.

Kate Gregory: Well, it turns out that C is so flexible you can make it look like FORTRAN, which is a very sad thing to choose to do.

Carl Franklin: I know FORTRAN has like operators that are good for mathematics and science and things and lot of scientific stuff and that's really the draw of FORTRAN.

Kate Gregory: Totally, scientific programming. I modeled clouds and things like that in FORTRAN.

Carl Franklin: Wow! Modeled clouds?

Kate Gregory: Yes.

Carl Franklin: What's that all about, what's that like?.

Kate Gregory: Oh! That was basically to prove that all the pollution in Canada was America's fault.

(Laugh)

Kate Gregory: Not sure how well the project turned out.

Carl Franklin: But that's not true. You guys still burning coal up there and you are polluting the environment too.

Kate Gregory: No, with some complex thing involving you know the Tennessee valley, coal, all drifting up...

Carl Franklin: In fact you are screwing up our Vermont Maple syrup because of the acid rain coming down from Toronto.

Kate Gregory: Well that's it, it's all a big cycle, the Tennessee smoke comes to Toronto and the Toronto smoke wanders off and... Heaven only knows where the Vermont smoke goes.

Carl Franklin: That's funny. So, object oriented FORTRAN. Been there?

Kate Gregory: No.

Carl Franklin: No.

Kate Gregory: No. The closest I can get to that, I do have a Object Oriented COBOL Joke.

Carl Franklin: Ok.

Kate Gregory: The real name of Object Oriented COBOL should of course be ADD 1 2 COBOL.

(Laugh)

Carl Franklin: Talk about wordiness right? Well, I love these people who say VB is such a wordy language because of things like 'is not', 'nothing' and stuff. But then lets do some serialization in C#, VB, C++ doesn't matter. Dim BF as new system.runtime.serialization.formatters.binary.binaryformatter. Come on. (Laugh) You think VB is, you're complaining about 'is nothing', and yet you have no problem going into nine name spaces deep to get a binary format or something.

Kate Gregory: But thanks to IntelliSense you probably only have to type four characters of that.

Carl Franklin: Right, and especially if you are using code rush, right.

Kate Gregory: Yeah.

Carl Franklin: Poink, poink, poink, done.

Kate Gregory: Absolutely, typing is so 2002.

(Laugh)

Carl Franklin: Oh man! So, we also have something in common that we both wrote books on sockets although yours was probably a lot closer to the metal than mine was. My aim in writing a sockets book was to try to give VB4 and then VB6 programmers' access to tools that were high enough level so that you didn't get mucked in the Mire of sockets and low enough level so that obviously you could program them but you are probably using the sockets API directly. I am sure.

Kate Gregory: I actually wrote my own sockets class for the book because I didn't like the one that was in MFC at the time.

Carl Franklin: Ok. So, socket's class to wrap the sockets APIs.

Kate Gregory: To wrap those APIs.

Carl Franklin: When you said I wrote my own, I thought she didn't write her own stack, which is what I was thinking.

Kate Gregory: (Laugh). No, no, no. I just wrapped up the APIs. I just didn't like the way that the first version of the MFC, C socket class did a few things, so I did differently.



Carl Franklin: I wonder if the Visual Basic sockets, OCX was based on that (laugh) because it didn't do something's right either and I chose not to use it in my book. Have you ever had that happen to you, this happened me. I chose not to use the Microsoft sockets thing because didn't like it and I had heard reports that it was not very good at all when acting as a server. Like it had bugs and stuff. And I heard that from several sources. So I decided not to go there and I used the third party tool that worked really-really well. But it had a splash screen and if you wanted to use it, it's like 75 bucks or something. But I figured, I'll be doing disservice to people if I were to write a book and have great sockets are and then write it all around the tool that sucks.

Kate Gregory: Yeah. And people would, if they had to choose between blaming some author they never heard of before and Microsoft, they are not going on the Microsoft.

Carl Franklin: No, exactly so I thought I was doing people a favor but I am, go read my historical comments on Amazon sometime you'll see.

(Laugh)

Carl Franklin: That was the right decision, right?

Kate Gregory: Absolutely, I mean the early versions of socket support really weren't that good. And I think I managed to come out with some sort of excuse like that if you were still on C++1.5 then you wouldn't have the classes so we could align or we just have to use mine but my motivation was just that this is the better class than that one. So, that's what I went with.

Carl Franklin: Well hang on just one second because I want to tell everybody about our classes coming up and we'll be right back. So stick around.

(Music)

Carl Franklin: So let me tell you that we have added some new classes for 2005 to franklins.net roster. If you've heard about the VB.net master class, may be you have, may you haven't. But it's not for the faint of heart, it's for intermediate to advance VB6 or even people, developers or even people who have been using .Net but just sort of peripherally. We dive in. Good typing skills are going to help a lot and we really do a lot. So check that at www.franklins.net. Class schedule goes as follows for this November 15th obviously coming up here, December 6th, January 10th, February 7th, March 29th and that's the VB.Net master class. The ASP.Net master class which is focused completely on ASP.Net, no Windows

forums and therefore goes into a lot more detail about building user controls and web controls and using the stuff that's there being data driven all that. December 13th, January 24th, February 21st and March 7th. We're also accepting seats for the C# .Net Boot Camp extended with Richard Hale Shaw, which is happening November 29th and his cost is now 2900 bucks and by the way our classes, our price is going up from 2000 to 2300 dollars So, If you want to get in the class for 2000 you want to go to this November or December classes.

Carl Franklin: So Kate, you have been doing some work with Visual Studio Tools for Office, I hear and in fact you were recently talking to the guys. What was it? Today you are telling me; you were on a phone call?

Kate Gregory: Today, I was on a web cast for the 2005 version of the product.

Carl Franklin: So what do you think?

Kate Gregory: Oh! Listen, I was very excited 18 months ago when I started using this because I am not a UI person. And the problem with users is when you give them user interfaces; their minds are contaminated with other stuff they have seen. So, you give them some kind of a grid thing and they are like, wow that's great but I should be able to sort it. So it's not that hard to write some codes to make the grids sort, right and then they say "yeah, I should be able to actually make a graph out of it or I should be able to print it differently than how the screen shot or something". of course this is all because of Excel, right? Everything you can do in Excel, they expect that your app will do. And so the great thing about VSTO is you like, "okay buddy, if you like Excel so much why don't you just use Excel?" And I'll write some managed code to put the numbers in there and then you can go to town and make pivot tables and make graphs and print things and email it to all your buddies and everything will be great.

Carl Franklin: Right.

Kate Gregory: And so the VSTO for 2003 gives you that and you're writing managed codes, you're writing VB, if you're total masochist you can write C # but you're writing VB and you can call web services or go into database or whatever you want. And it all just says here's my user interfaces, it's Excel or it's Word and you already know how to drive it. So, no training.

Carl Franklin: That is a really, really cool feature of this VSTO stuff that I have never done Excel, Macro programming, I was never like a big office programmer in general.



Kate Gregory: No, me neither.

Carl Franklin: Although, I realize the value, I mean the value is that the business world uses these documents for everything. And if you can provide some functionality behind their spreadsheets and behind their code, behind their documents that they are already using. To a business user, that's like glorious.

Kate Gregory: That's right. So they are happier and at the same time the programmers are happier. And it sounds like a great combination to me. And I don't have to learn VBA; I can use the whole .Net framework instead of try and learn some new set of techniques.

Carl Franklin: And if you never heard of Visual Studio tools for Office you can go back and listen to the show we did with Robert Green on that, when it just came out. But what separates this from previous sort of Office Programming Technologies is that the code does not exist in the documents. Right. It exists in an assembly, which is in a shared place. And this is critical because you don't want to send out a new document just to update the code to people.

Kate Gregory: Well and even if you did send out a new document, people don't always listen. I mean when I was writing books, the publisher sends you this file word document with all the templates in it to do things and you write, like, 11 chapters and then they say "Here's the new template", and then you are suppose to start using that for all your other chapters. You've just got such a mish mash of code in which macros are layered.

Carl Franklin: Right and you take all that.

Kate Gregory: It's more about deployment focus that, that we are seeing in the whole .Net Universe of things. How can we actually deal with the way people really use stuff as opposed to the way we thought it might be is...

Carl Franklin: Have you developed anything with it?

Kate Gregory: I did a few things for customer of mine because they really needed to make that macro dialog go away. They had this gorgeous work flow thing going on and it worked that all the low levels and when it went up to the top they just e-mailed the doc, the word documents around and when you open them up, a code ran, and the highest ranking people, the people with Windows in their offices, when they open that word document and saw that warning dialog saying, "this document contains macros". They had been

well trained by IT they would say, "no don't run the macros". And the whole workflow app failed because the top guys would not run the macros.

Carl Franklin: Oh, man.

Kate Gregory: So that was quickly converted into a VSTO solution.

Carl Franklin: How did you deal with the security issue?

Kate Gregory: You have to get IT to put a code access security policy on everybody's machine.

Carl Franklin: And they understood that and they figured it out? Did you write the policy?

Kate Gregory: No I just told them what to do. I said make it like this and they did something with a lot of initials in it that IT people are cool about. Some automated, "oh, we'll just push that out and insert initials here". And I said, "okay that's fine, you guys are in the basement". But, that is a real issue, right, everyone has to have the .Net framework, everyone one has to have Office 2003 and everyone has to have the right security policies because by default none of this code is trusted.

Carl Franklin: So just by out of curiosity did you do the signing things? Or did you sign your assembly with the strong name and then trust all assemblies with that strong name?

Kate Gregory: Well, with that key, yeah.

Carl Franklin: Yeah, That's what I do. I love that, I love that, solution.

Kate Gregory: If it comes from us, it's good.

Carl Franklin: It's safe, 100% right?

Kate Gregory: Yeah.

Carl Franklin: Yeah. And getting that policy out to all the users, you just created a set up program that administrators could run.

Kate Gregory: Well I have to say, I let the basement guys do it, but that's my guess. And you don't have to do it on a 30,000 desktops; you have to do it on the desktops of the people who are using the app.

Carl Franklin: Do you know of any resources that you can send people to for creating those code access security policies?

Kate Gregory: There is a whole office developer center before I say URL, may be I should type it.



It's something brilliant like
msdn.microsoft.com/office.

Carl Franklin: We'll provide a link to that but anything particularly on doing code access security policies that you know of and if you do we can find it.

Kate Gregory: No, not at the top of my head. No, because I am lucky enough to have basement people at the client.

Carl Franklin: You throw some meat down to them every once a while, some coke.

Kate Gregory: Roll cans of coke down the stairs.

Carl Franklin: Slide a pizza under the door.

(Laugh & sound)

Kate Gregory: You got it.

Carl Franklin: Those people are invaluable and we love them dearly.

Kate Gregory: They run the firewall too, you have to love them.

Carl Franklin: Back to the sockets thing, what kind of implementations have you done with the sockets because to tell you the truth that's where the really cool and interesting programming happens, in my opinion, is when you have a raw communications channel from program A to program B and you can do whatever you want. What kinds of cool things have you done?

Kate Gregory: Well for the books, for the heck of it, we just sort of implemented the clients and servers that were in common use sense. So we did like a mail clients and FTP clients and news and so forth, antiquated things that no one does anymore.

Carl Franklin: I wrote a Gopher client.

Kate Gregory: I think we did have Gopher in there actually. A Gopher was the first time that the Internet had any kind of magic to it for me.

Carl Franklin: Yeah, Me too.

Kate Gregory: You are following things and you don't know where you're going to go and that everyone thinks that it's normal thing with the web but that was a whole new thing with gopher.

Carl Franklin: Well it was the web, essentially except that there was a folder metaphor that's all, a file folder metaphor.

Kate Gregory: But then you would go somewhere and there would be this list of things and the women look interesting and you go somewhere else, you kind of look up and say, "I don't know where I am". And that sort of serendipity thing, we all got hooked on that, we just found it another way to achieve that.

Carl Franklin: Sahil from the chat room says and he is our resident question asker by the way. What are your feelings about WinFx in comparison to Win32 if that doesn't open up a can of worms, I don't know...

Kate Gregory: Really, the you decided page is necessary. Let's see. I am looking forward, perhaps very forward to getting to use that eventually. But in some ways Win32 is always going to be with us, right. When we think we are calling some exciting .Net base class library today, guess what it turns around and does, right. So for everything to be completely redone, a 100% managed is a good thing to look forward to. Because I have always been C++ person, the integration bizarrely enough between old stuff like MFC is better in newer technologies like in WinFx and an Avalon in the future than it is in what we have going on right now, for example with Win forms and with the base class library. So, today if you want to call the base libraries from C++ old, some unmanaged C++ competitive code you have to go through COM. But in days to come there will be better ways to do that.

Carl Franklin: How about in Enterprise Services? What do we know about that? Don't you work with Enterprise Services?

Kate Gregory: I wrote a whole course on Enterprise Services.

Carl Franklin: Oh wow! What's your take on that? Because you know we had Juval on. He was the first guy to talk about it on our show. And he was also the last guy to talk about. I mean not a lot of people and we also as I said on the last show Juval that we did some sessions at Dev connections on that and nobody showed up. So, what can you say about that?

Kate Gregory: You should probably be scared of Enterprise Services because the rules for it is-don't use it unless you really-really need it.

Carl Franklin: Really? So you are the first person I have ever heard say that.

Kate Gregory: If you need, say, distributed transactions across like SQL and Oracle. Then you need Enterprise Service, if you need object pooling and all the different slice dice and different ways of rolling object pooling that it



gives you, then okay go and do it but otherwise don't.

Carl Franklin: Just in time activation.

Kate Gregory: There's just in time activation and a pooling...

Carl Franklin: All those other things...

Kate Gregory: Yeah. It's very complicated and there is prices to pay, like some of your classes have to inherit from service component. You don't take it on just because oh yeah! I heard the transactions are cool. You know because it may be, you can just use transactions in ADO or you can just use pooling if all you want to pool as your database connections there is pooling in AADO.NET.

Carl Franklin: Now this new system.transactions namespace which Juval talked about couple of weeks ago. That's not Enterprise Services. Is it?

Kate Gregory: I don't believe it is. I think Enterprise Services is this for full on distributed transactions across two, completely unrelated providers that don't know anything about each other. Enterprise Service is a good name for because that's where it comes into play. So I have a fair number of clients who've been formed by mergers. And it is really entertaining when all Oracle shops get merged with all SQL shops because the guys who arranged the mergers do not know what these things are. Right? And then you apps have to somehow magically talk to each other. And so if you live that life then you need Enterprise Services. But if you are sitting at home saying, I think it would be really cool if we could write an app that figured out exactly how much to charge for our widget. You are not going to need Enterprise Services.

Carl Franklin: And pooling probably is a lot less required now that the object model or object creation story is a lot better in .NET.

Kate Gregory: Exactly.

Carl Franklin: Machines are faster etc.

Kate Gregory: There was in a chalk talk in South Africa and someone was having an issue with object lifetime in his middle tier and he was saying, he was going to start using Enterprise Services so that he could do some object pooling...

Carl Franklin: Let me guess. They were using a P3 400 megahertz with 256 Megs of RAM.

Kate Gregory: Yes, he had a hardware budget issue.

Carl Franklin: And loading a great big COM object with every request in ASP.

Kate Gregory: Yeah. And so he's having multiple great big COM objects kicking around. So, this is his solution. He was going to get Enterprise Services into the mix, or he was going to do remoting because remoting also has an object lifetime management for you. But you know there are other ways to tackle that particular issue.

Carl Franklin: And watch for Indigo too.

Kate Gregory: I like the promise of Indigo that it's all the good stuff in Enterprise Services, all the good stuff in web services and all the good stuff in remoting and it's not done till it's just fast as the fastest of the three.

Carl Franklin: Right and also with the least amount of code as possible, which is another great thing and I like that. Getting back to WinFx, I am always hearing confusion about this, about what's managed, what's not? Longhorn is supposed to be backward compatible with Win32, yet it's suppose to be a 100% managed, what's the story there, what's managed and what isn't?

Kate Gregory: I just keep hearing that everything is managed. And that's the issue of making yourself available to be called by unmanaged code is solved in a way that's different from the way that it is solved today with among other things interop...

Carl Franklin: So okay. So somehow we are going to have a Win32 DLL's that aren't really Win32 DLL's, that are going to sort of have emulators in a managed world or something or...

Kate Gregory: No, I think the wrappers will turn inside out.

Carl Franklin: Really?

Kate Gregory: So who is wrapping who? I mean it's like the same thing I was talking about refactoring your COM component, right. If you have got something that needs to be available to old or new. Today, you can put a new wrapper around old, but there is nothing to stop you in the future from putting an old style wrapper around the new. Either way you still available to old or new.

Carl Franklin: Well, what about API calls that require pointers and things like that?



Kate Gregory: Oh that's just where C++ comes in, right? Because C++ is the language which is just simple to play with.

Carl Franklin: No, no. I mean you have a Win32 app, right. That is using API calls and Win32 that uses pointers that's managing its own memory. Is that going to run on Longhorn?

Kate Gregory: I think so, that's the plan. I mean you know, what they always say, right, 'it's not done if it can't do that yet'.

Carl Franklin: Right. And it's going to be managed?

Kate Gregory: Yeah.

Carl Franklin: So, the program is going to be run and it's going to be managed and that program doesn't have to do anything special.

Kate Gregory: That's the story.
(Laugh)

Carl Franklin: Doesn't that sound weird?

Kate Gregory: It does. Yes. But then you know well, like a lot of the things that we do today, sounded weird, five years ago.

Carl Franklin: Wow! I have just never thought about it that way before, I always thought like they are going to have some sort of emulators, that will emulate Win32 but keep the whole thing managed but it sounds like, ...wow!

Kate Gregory: Just let your unmanaged code call into managed code and then come on back out. And obviously the Shim that sits in between has got some, may be fancy dancing so do around managing memory but memory is still just memory at the end of the day.

Carl Franklin: It just seems weird to me that if I have a program I wrote in C++ and I 'AllocMem' and allocate and try to lock in an area of memory. And that I have a pointer to that and I start messing around with that memory and unlock it, that something is going to kick in there. You know what I mean?

Kate Gregory: To clean that up for you?

Carl Franklin: To clean that up and make it managed without me having to do anything, doesn't that seem like it would really hurt the performance?

Kate Gregory: Yeah, because you can make things managed by saying, I am going to go put something over on the managed heap that knows

where to find this stuff that's in the unmanaged, not heap but area, right? So, it's just all about layers and shims and wrappers and may be you will take a performance hit. When I am telling people how to work at their InterOp stuff, about whether they should put a wrapper around their old codes so that .Net can call it or put their old code and put a wrapper so their old code can call it. The number one question is, well, who do you want to give the performance hit to? If you want your old COM app to feel clunky and slow and your new .Net app to be zippy and fast, then write all your new stuff managed and that make your old code InterOp. And so just because it can be done doesn't mean it can be done quickly. So, when you say "we still can maintain that compatibility layer for your old programs", it doesn't mean that they will be as fast as they would have been if they were managed.

Carl Franklin: And it's blowing my mind. I have to really think about that. If they are trying to pull that off now and that's what delaying Longhorn, man, I totally understand.

(Laugh)

Carl Franklin: Seriously, that's insane. I mean that's great if they can do that and if the performance is good. There's got be a program out there that can crash it though. Got to be, well they sort of did the same kind of thing with like DOS emulation right, but it was an emulator and so why wouldn't they run it.

Kate Gregory: Well I guess there is a fine line between an emulator and a wrapper.

Carl Franklin: I suppose you are right. But an emulator to me is sort of sounds like a whole separate process environment that is set up to do some hooks. It's sort of more managed underneath. Yeah, wow, Geez! Excuse me while I get my brain back in off the table.

(Laugh)

Kate Gregory: C++ has that affect on some people.

Carl Franklin: What other mind-blowing things have you been thinking about lately, Kate? Or fun? What do you do for fun?

Kate Gregory: What do I do for fun? I just went on vacation. It's one of my first like real big vacation for about 10 years. And I went to England and then France.

Carl Franklin: Went on a 10-year vacation?



Kate Gregory: No, 2-3 weeks if you count TechEd at Africa. I went to England and France and Africa and I had a fantastic time.

Carl Franklin: So what did you do over there?

Kate Gregory: I took my kids to Buckingham Palace, and the Tower of London and Notre Dame, the Louvre and...

Carl Franklin: Cool.

Kate Gregory: Yeah. It was very cool, very educational and hardly any coat at all.

Carl Franklin: Do they still have Mary Queen of Scots beheaded, severed head somewhere over there?

Kate Gregory: Actual severed head-wise it was a little thin on the ground. Although there was a couple skulls in the British museum and some guys had been preserved in a Peat bog. My kids took a lot of pictures of the Peat bog guy.

Carl Franklin: Here is a little trivia thing you may not know about me but when I was 12, I think, 12 or 13, I went with a Westerly Chorus over to England on a tour. I was a boy soprano and we did tours of cathedrals. We did concerts in cathedrals all over England and Scotland. And we were the first; I think one of the first American choirs to sing in Westminster Abbey

Kate Gregory: That would've been cool.

Carl Franklin: Yeah I think that's, I think that's the claim to fame. But that was pretty awesome. I had never seen big churches before. But I had never gone into a cathedral, like you know these are thousands of years old and you walk in and they're like, I remember the most, scariest one was in, Durham cathedral. It was like dark and cold and stone and you go, you clap your hands and it goes. (Sound).

Kate Gregory: You are supposed to be scared in there. The whole building is built to make you feel odd.

Carl Franklin: I know it is, it's jarring and the guy in the Organ lot was practicing (voice) it's just like haunted mansion kind of sick and twisted, and you look down on the floor and it's like, here lies St. Peter or something it's, what? Whoa, whoa.

Kate Gregory: Everybody is in Westminster Abbey.

Carl Franklin: In Westminster Abbey that's right. All the saints in there, they are all dead, buried there under the floor. It was really awesome.

Kate Gregory: The creepiest thing in Westminster Abbey was someone who is not buried there, no relative I am pretty sure but Franklin as in the Northwest Passage, Franklin.

Carl Franklin: Oh well.

Kate Gregory: And he has a little monument there, and underneath that it says, 'not here the ice has him'. Ooh! Creepy, right?

Carl Franklin: That is creepy.

(Laugh)

Kate Gregory: Yeah, it's a little Canadian corner Westminster Abbey they had the battle of the planes of Abraham commemorated with a big statue of somebody dying in somebody else's arms. Very big on that in Westminster Abbey, actually.

Carl Franklin: So what events are you doing these days?

Kate Gregory: So I just did TechEd Africa where it was like a million degrees and you had to wait for to cool off enough to be able to go out for a swim. And then next week to sort of balance that, I am going to go to Winnipeg. Where it will probably be snowing.

Carl Franklin: So, what's going up in Winnipeg?

Kate Gregory: We were doing in Canada, an MSDN user group tour, the Regional Directors are all, basically flying to each others cities and just criss-crossing the country to tell people all about the Smart Clients and the next version of Visual Studio Tools for Office.

Carl Franklin: Great!

Kate Gregory: And there is about 5 or 6 going on at once on November 10th and then we are sort of scattering all through November and doing a bunch more. And, so that's a lot of a fun for us all to be doing the same talks at once and normally when things happen in Canada they tend to happen one city a time so they stretch out over 6-8 week. It's really enjoyable to think we all will be doing it together.

Carl Franklin: Yeah. That is neat.

Kate Gregory: And VSTO 2005 product is going to be just a treat to develop because people who have been doing it the 2003 way are going to say, "Oh, here is all the parts I don't like fixed." And the people who hadn't gotten around to yet are going to feel all smug that they don't have to



learn the hard way to do things. Same with C++, if you didn't do any C++ in the double underscore age you can feel all smug and skip that and go straight with 2005, beautiful and elegant way.

Carl Franklin: I may be actually persuaded to do that.

Kate Gregory: VSTO 2005 does things like hosting Excel in Visual Studio. So when you are doing VSTO project and you hit run, it loads up either Excel or Word depending on which kind of thing you are doing and opens your text documents so forth and so on but it does it in a separate Window because it actually spins up the app. So for the 2005 product that's actually hosted right inside Visual Studio, a little bit more integrated.

Carl Franklin: And if your last experience of hosting Word or Excel was back in the OLE2 days, you need to wake up and smell the coffee, because it's really, really much, much, much, much, much better.

Kate Gregory: For sure, I have seen demos of it and they actually like come up and you can interact with the product. That's an interesting feature that wasn't really there before. And smart documents in VSTO 2005, the current shipping version of smart documents really feels like it is about 0.8 like it's not, it works, it's just not easy.

Carl Franklin: So tell us what smart documents are?

Kate Gregory: Smart documents are a word document, which is secretly, I mean you can show it but people don't, secretly mapped to an XML schema. And so as the person is typing in the document there is things happening over in the action pane to make the document easier to type. So the classic example is some sort of a sales letter where in the action pane there will be drop down box with customer names and you pick one and it would shoot the name, address typical mail merging kind of stuff into the letter for you. And then the action pane would have more information in it, say about products this person has ordered or about their credit limit or whatever, that would help you to compose the document as you are going along.

Carl Franklin: So it sort of a clippie all grown up.

Kate Gregory: And very-very customized, I mean document by document. So here is the code that works with the sales letter, the completely different set of code to work with, I don't know the, "we really need the money, we've asked you for it 6 times" letter. And all of this code is managed, you could also do this in VB6

with unmanaged code, and the experience is pretty much the same in both VB6 and in VB.NET. There is this interface with this, about 400 methods in it. And you had to implement the whole lot of very-very repetitive method. And for the 2005 product, that's basically been replaced with the Visual designer. So, where as in the old product you would say, when you are in this element of the XML there are two controls in the action pane and the name of control no.1 is, I don't know, name, and the type of control no. 1 is, it's a text box, that's all gone and you just sort of drag a text box on to a design surface. The implementation of the 400 functions is done behind the scenes for you.

Carl Franklin: Now I know in the current version of Visual Studio Tools for Office, you basically have Word documents, you have Excel documents and that's it. Are they going to try to attack the other office applications?

Kate Gregory: Yes. I am not sure they have announced which ones. The current version doesn't bring in any body new it's still just Excel, Word and Word template.

Carl Franklin: But their goal is to hit all of them.

Kate Gregory: Yes, it didn't occur to them that there will be a lot of calls for say automatic PowerPoint? But people want help building their PowerPoint apps, the same as they do their word documents.

Carl Franklin: Now of course the real trick will be to get other vendors to use this kind of technology in their applications like, I can think of a couple like Adobe Premier, if you're making movies, which we do.

Kate Gregory: Same thing where you can run off your own code in the middle of what you are doing right with the product.

Carl Franklin: Oh god, I would love to. Or the audio program that we use which is now Adobe Audition, used to be Cool Edit Pro. I love to be able to have some sort of programming interface to these objects that you are using. So what about InfoPath? Have you done much with that?

Kate Gregory: InfoPath is, I love doing demos of InfoPath because it makes people jaws drop.

Carl Franklin: And then they say oh! My god my job is in jeopardy.

(Laugh)



Kate Gregory: Then when you go to try to do it in real life, then you realize your job is not in jeopardy.

Carl Franklin: Good point. I haven't actually used it. But my good friend Tom Robbins has written a book on InfoPath programming and he is really excited about it. We are going to have him on the talk very soon.

Kate Gregory: If you just want to make a form that people are going to fill in, 10 or 15 pieces of information with some pretty obvious validation and then stick it in the data base or stick it in a web service, InfoPath is wonderful for that. If you want a full on workflow situation where depending on what they typed in box 3, you do one of the following ten things. But the first time I used it, I was really bitter, I was trying to map to a SQL database, that here is my database and it had read my schema and made me a form and everything. And my name in the database was only allowed to be twenty characters long, and I was trying to setup my validations. And I couldn't find a way to make a validation to say that you could only type 20 characters. I was bitter. Until I realized, it had done that for me. Because it had read my schema so, of course there is only twenty characters in the database. So it won't let you type more than twenty characters in the form, and I didn't have to set that up and all of that's automatically there. So, if you just want a data entry, oh it's great, and you can make it look beautiful if that's your style. All my apps look like a programmer wrote them. A bunch of gray boxes with gray button on them and text boxes for you to type in. So my Info Path forms tend to look like that too, but I'm apparently you've got sort of cascading style sheets and HTML at Wahoo things happening. So you can make it, fit your corporate look and feel.

Carl Franklin: Well, it's another good segway into user interface. So, you say you're not a user interface design expert, but have you used any third party tools, like Infragistics or component one, or any of those for Window's forms applications to do these kinds of complex things?

Kate Gregory: No, I really haven't and I probably should, I get the opportunity to hear about these things from time to time but doesn't always work out in terms of being able to play with them on a project. It's seems like the projects that you've time to try new things on or the project that nobody really cares how they work out.

Carl Franklin: And you really do need time. I mean if you buy a third party solution, you are going to have to spend some significant time figuring out how they do things.

Kate Gregory: I know in the past though the user interface components have been the best money I ever spent, when I was doing more Visual COM work, I hardly do any COM work now for UI. It gives somebody like a hundred dollars for some little control and it's got ten-programmer years of effort in it. I understand they must sell thousands and thousands of them. But boy, the cost of them compared to the price of one or two hours of programmer time.

Carl Franklin: I know it's ridiculous. This is also another good question that, I know that there is still a lot of people sort of on the fence, especially managers who were programmers in the old VB days and the old Visual C++ days. And now they are managers and they are making decision about technologies and stuff and sort of a little afraid of Windows forms, because of their old pain. I'm going to have a column called old pain, which I think is the demise of critical thinking. And if you are going to work in this industry, you have to get over your old pain. You can't apply your experiences of one technology to another technology, even though those technologies come from same company. And so thinking about active X controls in the past, a lot of people got burned, because they would buy these controls, they would have bugs, they didn't ship the source code. You are out of luck trying to talk with some tech support person. I was a tech support person, I know their pain, because I talk to these people and that's an impossible situation for a lot of people. They don't want to get into that. But when you think about .Net controls (a) No DLL hell issues, so versioning issues go away, (b) You have reflector even if you don't have the source code, you have the source code because you can use a tool like...

Kate Gregory: Right! Nothing is hidden.

Carl Franklin: Nothing is hidden, and even if it was written in C# you can read it in VB, right, reflector. You have got dis-assemblers, you've got, your knowledge of the frame work, you have objects that have override able members all over the place So, if you don't like what happens when a particular event gets fired, you can override, you can create your own which you should be creating your own subclasses. You override those on event handlers, and then you just swallow the event, or you do some code before you raise the event, you do some code after, and in that way you get to do exactly what you want and it's better than having the source code. So what do you think of that? There are a lot of people that sort of operate on their old preconceptions and old pain.



Kate Gregory: They sure do and you are totally right. I'll tell you an old pain story. Million years ago, like may be in 1985, I worked with a guy who wouldn't let us use underscores in our variable names.

Carl Franklin: Wait a minute. Is this the Sybase thing?

Kate Gregory: No, it turns out to be based on one particular IBM printer. This particular printer, which this guy had used, to set more lines on a page, strip the bottom pixel out of every other line. And so underscores would become spaces.

Carl Franklin: And therefore...

Kate Gregory: And therefore, underscores were evil.

(Laugh)

Kate Gregory: And we hadn't had one of those printers for like ten years.

Kate Gregory: (Laugh) and he was living on his old pain.

Carl Franklin: Right. That's what I am saying, it's our duty to wake these people up to expose their irresponsible behavior.

Kate Gregory: And what you're saying about .Net components, and hey! If you don't like it just inherit from it and override. I mean, that is such a powerful story that most people don't truly get.

Carl Franklin: I totally agree.

Kate Gregory: It sounds like it couldn't possibly be true, how can that be right, I don't have the source code too and how can I inherit from it, and so they don't stop and sit in that space long enough to realize, if that is true what does that change? Because, it is true.

Carl Franklin: Yes.

Kate Gregory: And it changes everything, about relying on someone else to provide some of your functionality. Because, okay, I like 90% of your things, and I'm going to take care of that 10%, I don't like it, make it my way and I don't need you to help me do that. I am in. I want to use that component because I'm not hostage to it.

Carl Franklin: Yes, wonderful. That's the approach to problem solving that and I'm sure you talk about this in your OOP talks in your classes. I do too, I preach on, and I am really trying to turn people into, from inventors to plumbers, instead of trying to come up with the

right algorithm or the right trick or the right patch to download, to do something, you really just have to find out what's there already, and what inputs to hook up to what outputs, and what to override and to become the object that you are trying to fix and then to fix your internal behavior. That is a problem solving approach that you have to take as an Object Oriented programmer not the, how can I affect this from the outside, how can I kluge around it and you know what I mean.

Kate Gregory: Absolutely, and that anthropomorphism, when I am teaching my old courses, I say to people if you believe there's little guy in your computer who kind of runs around and carries things around for you. Then you are going to have a much better time in my course.

Carl Franklin: (Laugh) That's good.

Kate Gregory: Alright, so I am the COM component and I'm getting a click what I'm going to do, that if you can think that way. And the days are gone when people would just pay you to type code for six years.

Carl Franklin: Right, That's right.

Kate Gregory: There is no budget. So you got to make it work and having a not invented here thing, and say, well I am not going to use that component because I didn't write it, it's wasteful.

Carl Franklin: How many times have you seen also, a new consultant come in and pick up a project that's been done by somebody else and the first thing they say is, we got to re-write it. We are all talking objects, we are all talking .Net, and we don't necessarily need to do that any more.

Kate Gregory: No, if there is one little piece that's not right that you could may be slip that out and slip a new piece in and not have those deployment hassles. The whole idea is not to be so monolithic any more. And to be able to mix and match is a very powerful, very powerful technique. Some of my clients are resisting upgrading from VB 6 because they think they have to port every line of VB 6 code they have to VB.Net in order to make that upgrade.

Carl Franklin: I know, and it doesn't help that we've topics at conferences and things like upgrading from VB 6, like you are going to upgrade, like you are going to run the wizard and then, what do you have to fix and, I get exactly that. I have this VB6 application, since Windows application and I want to convert it, I want to upgrade. What I have to do in? Well, you do have to rewrite.



Kate Gregory: You have to touch every file, potentially introduce bugs by typing all fumble fingeredly.

Carl Franklin: And when you are done, you have got something that isn't, something you want to move forward with, it's got compatibility layer code and it's, Yuck!

Kate Gregory: Yuck! (Laugh). It's interoping to ADO. It's horrible.

Carl Franklin: Well. What else is, any last minute words of wisdom, or advice, or calls to action, or any promotions, or anything that you want to mention, before we hang up.

Kate Gregory: If some one is in Canada they should definitely come to the MSDN user groups tours, the smart client one that's going on and the InterOp one that's following that. And get ready for Whidbey, get a beta somehow and start playing because C++ is not the only technology that's undergoing big changes. For Whidbey, every body should be getting ready.

Carl Franklin: And what is your URL of your home page, if somebody wants to...

Kate Gregory: The best entry point is my blog. So, our website is GregCons, that's the first part is first four letters of Gregory, and the first four letters of consulting, because we are so old, so we used abbreviations, /kateblog (<http://www.gregcons.com/kateblog/>), and from there you can wander around the rest of the site, there is some links.

Carl Franklin: Awesome, Well Kate, I can't tell you how enjoyable this show has been for me and I am sure the listeners too.

Kate Gregory: Great!

Carl Franklin: So, on behalf of myself and I know Rory is out there, somewhere doing a great job in his role as an MSDN guy. Geoff Maciolek out in the sound room and from everybody else here at the .Net Rocks! staff, thanks for coming on the show.

Kate Gregory: Thanks for having me.

Carl Franklin: Right. We will see you soon.

Kate Gregory: Absolutely.

Carl Franklin: Sometime out there. (Laugh). Goodnight.