



.NET Rocks!
 The Internet Audio Talk Show
 for .NET Developers
 With Carl Franklin **msdn**
 and Richard Campbell
<http://www.dotnetrocks.com>

<http://www.dotnetrocks.com>



Carl Franklin

Carl Franklin and Richard Campbell 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!



Richard Campbell

Text Transcript of Show # 313
 (Transcription services provided by [PWOP Productions](#))



Miguel de Icaza and Geoff Norton on Mono
February 5, 2008
Our Sponsors



<http://www.devexpress.com>



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



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



Lawrence Ryan: 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]

Lawrence Ryan: Hey, Rock heads! Scrape that Patriots bumper sticker off your car and listen up! It's time for another stellar episode of .NET Rocks! the Internet audio talk show for .NET developers, with Carl Franklin and Richard Campbell. This is Lawrence Ryan announcing show #313, with guests Geoff Norton and Miguel de Icaza, recorded live, Friday, January 18, 2008. .NET Rocks! is brought to you by Franklins.Net - Training Developers to Work Smarter and now offering SharePoint 2007 video training with Sahil Malik on DVD, dnrTV style, order your copy now at www.franklins.net. Support is also provided by Telerik, combining the best in Windows Forms and ASP.NET controls with first class customer service, online at www.telerik.com. And now, the man who says, "My grandmother can get a touchdown in 1 second," Carl Franklin.

Carl Franklin: Thank you very much, thank you and welcome back to .NET Rocks! the official .NET podcast of Super Bowl 2008.

Richard Campbell: You can't say that. So, did you watch the game?

Carl Franklin: You know, I came in when there was 2 minutes left, so I saw the end drama because I knew that that's what it was going to be all about anyway. I knew it was going to be a close game so I didn't worry about it and I turned it on late.

Richard Campbell: I turned on the whole game. I flipped it on and then I guess within the first 30 seconds of starting, I muted it and popped open my laptop and watched an episode of MythBusters.

Carl Franklin: Yeah, spot the football fan in the crowd.

Richard Campbell: There you go. I sort of watched the last half. Heck, even my wife sat down and watched the last half.

Carl Franklin: Yeah, well, it was the most watched Super Bowl in history.

Richard Campbell: And it was a pretty good ending.

Carl Franklin: 97 million tuned in, I think. Well, anyway, let's get right to Better-Know-A-Framework.

[Music]

Richard Campbell: All right, sir, what have you got for me?

Carl Franklin: I have a really interesting collection, a generic collection, a system collections object model called ObservableCollection(T).

Richard Campbell: Observable?

Carl Franklin: Observable. It represents a dynamic data collection that provides notifications through events when items get added, removed or when the whole list is refreshed. How cool is that?

Richard Campbell: That is pretty cool.

Carl Franklin: Yeah. It was hiding out on me too. I had to dig for that one. ObservableCollection(T), like it, love it, use it, and all that.

Richard Campbell: Your life will be better.

Carl Franklin: Your life will be better. Richard Campbell, what have you got? Besides a bad case of something sick?

Richard Campbell: I've got an email from a new listener.

Carl Franklin: A new listener?

Richard Campbell: Yes, he says, "Richard and Carl, hello guys. Show #310 was my first listening to .NET Rocks!"

Carl Franklin: Well, all right.

Richard Campbell: Yeah. "I enjoyed listening to the show and have a few comments I would like to share." And I love it when a brand new listener sends an email, so welcome George, welcome aboard.

Carl Franklin: Yeah, absolutely, absolutely.

Richard Campbell: "The show was about the functional programming language Haskell but there was no mention of any previous functional programming languages, Lisp for instance has been around since 1958. Had I not known this, I would have gotten the impression from listening to the show #310 that Haskell and functional programming languages were a brand new thing." And I thought



about this for a minute because #310, that was the show with Simon Peyton Jones.

Carl Franklin: Yeah, we've done quite a few shows on functional languages.

Richard Campbell: Yeah, I'm amazed that we didn't say Lisp in that show because we've said Lisp in virtually every other functional programming show we've done.

Carl Franklin: Every other functional programming show, yeah.

Richard Campbell: So, it's an interesting thought to me to think about the show in the context of a brand new listener.

Carl Franklin: Right.

Richard Campbell: And I try -- you know, I really think we work hard at not being too "insidy," like you have to listen to all the shows to make it make sense.

Carl Franklin: Sure.

Richard Campbell: But an interesting twist and he's right. Lisp has been around for a long time and there's plenty of other languages before that and Haskell's relatively new in comparison.

Carl Franklin: Yup, but in the meantime you might want to check out some of those other shows.

Richard Campbell: Yeah, you bet, go back to the catalog. I'm sure you'll find some great things there. One other comment that George made, "I came up with another example of lazy evaluation while listening to the show that could have been mentioned to clarify the concept, Infinite List."

Carl Franklin: Ah, yeah.

Richard Campbell: "Suppose, for example, that you had an infinite list of primes." Boy, does that come right out of a Com. Sci. class or what?

Carl Franklin: Absolutely.

Richard Campbell: "Suppose we had an infinite list of primes..." "Laziness would allow the programmer to pass around this list as an object as if it had already been computed and when the nth element is queried, the nth prime would be computed."

Carl Franklin: Yeah.

Richard Campbell: Okay. It's an interesting thought. I don't know exactly where I'd use it. It sounds very "encryption."

Carl Franklin: Yes, it does. Yes, it does. Boy, encryption is...

Richard Campbell: But it brings up this interesting idea around this lazy evaluation where we're able to say, "There will be something here, but I'm not going to pre-compute everything because it will take forever. Compute only what you need when you need it."

Carl Franklin: Yeah.

Richard Campbell: George closes out with, "Despite all my complaining, I really appreciate what you guys have put together. The sound quality was superb and the topics, as I glanced through past episodes, are extremely exciting. Thanks again."

Carl Franklin: Well.

Richard Campbell: And thank you, George. We'll send you out a mug for your troubles, appreciate your thoughts.

Carl Franklin: Well, I think we got another huge fan on our hands here.

Richard Campbell: I think so.

Carl Franklin: That's awesome. All right, Richard, well, this is indeed a special show because we have with us not only a couple of guys from the Mono Project but none other than Geoff Norton and Miguel de Icaza. Miguel needs no introduction. He is the founder of the Mono Project and his proverbial ass was bought by Novell, sorry Don Box. He has continued to develop the Mono Project which is an open source implementation of the .NET Framework and it works on lots of different platforms which we'll talk about in a little bit. As for Geoff Norton, he's a developer for Novell working also on the Mono Project. He is the founder of the Cocoa# and Objective-C# projects and is the person in charge of the Mac specific development for the Mono Project, including a native System.Windows.Forms driver and native Gtk# support. Prior to working for Novell, he'd been a Mono Project contributor for about 3.5 years. Geoff and Miguel, welcome to the show.

Geoff Norton: Thanks guys.

Miguel de Icaza: Thank you guys.

Carl Franklin: So, this is all about development of .NET apps on the Mac. That's what we're going to talk about today. Yeah!

Richard Campbell: How hard could it be? Maybe we should set the stage a bit on how does application development normally happen on the Mac?

Carl Franklin: Yeah, that's a great place to start because we're admittedly clueless there.

Geoff Norton: Well, traditionally, application development on the Mac has gone down one of two paths, well, three paths. Historically, Apple's big focus was on development, send technologies to the windows system they have called Carbon which has now been deprecated. For a while there, Apple was also supporting Java which isn't deprecated yet but they're strongly pushing everyone towards using Objective-C and Cocoa. Objective-C is a language that they've got mostly from neXT, it's derived from Smalltalk, it's a functional language like that and they're really in a big push with they're latest technologies to push everyone down that road. If you look at how Java support was released with 10.5 Leopard, they didn't even have support for Java 6, they just, finally, at the beginning of December released a developer preview which sort of left all of their Java developers out in the lurch there for a while but that sort of, the traditional model is using XCode which is Apple's IDE and developing native applications for Apple. Some of the swing stuff that they've done with Java integrates pretty well so you can do cross-platform with Java but obviously, that's not how we feel the best way to be developing applications for the Mac is that's why we've been focusing on getting some of the Mono Story being a better cross-platform story across all the major operating systems.

Richard Campbell: So Objective-C which you'd immediately think is C but it's got some other elements in it, right? I mean it's very object oriented, it almost seems it's got roots back all the way to The Next.

Geoff Norton: Yeah, yeah, it is. It's a SmallTalk like language, it is compiler compatible with C so you can intermix C code with it in and of itself is definitely nothing like C.

Richard Campbell: Interesting and this is the recommended development platform by Apple?

Geoff Norton: I don't know if they have officially recommended this but I would say, from their positioning and from their SDKs and from everything that they're doing, they're pushing people towards this, for a long time they were recommending that or Java. I'm not sure that they're officially changed their stance on that but they're definitely pushing people down that road.

Carl Franklin: Now you say it's a functional language, Objective-C but does it have any of the kind of robust framework stuff that either Java or .NET has?

Geoff Norton: Well, it does for Mac, the language level, the nice things about .NET and Java are that your application support and your framework support is cross platform. The objective C stuff that Apple gives you, like Apple Kit and Cocoa and things are not, they only run on Apple. Now the GNUstep Project takes a look at implementing a lot of that in an open source manner to run on Windows and Linux, they're lagging behind, Apple's implementation, obviously and it's more an academic project than anything else but it does exist. It should be mentioned for the sake of completeness.

Carl Franklin: But in terms of the robust libraries, what I'm getting at, the development library, how much plumbing code does this, does an Objective-C programmer have to do?

Geoff Norton: I guess it really depends on what you're looking at. For instance, the AppKit framework shipped by Apple does things like, you want to go make an http web request, for instance in .NETSpeak they've got code to do that for you. You don't have to go right from the TCP socket layer like you would in C but that's, there's lots of C libraries that would do the same thing now.

Carl Franklin: So well, that's one thing I mean there's hundreds of thousands of nice little things in the framework, what about the big things like memory management and garbage collection and all those great things that we like about .NET?

Geoff Norton: Well, Apple has just released Objective-C version 2 with 10.5 which does bring in optional garbage collection, you don't have to use it but it is there. In Objective-C 1 they had a reference counting implementation which they called Auto Release Tools so things would be doing simple reference counting, you'd either retain it or release it and if you didn't have to explicitly call release, it would go and do it for you if the reference count was down to zero. So they're partially there, they've integrated the Boehm garbage collector which is actually, we use a packed version or a customized version of the Boehm garbage collector in Mono, as well. They've implemented that for the reference counting, sorry, for the garbage collector in Objective-C 2. So it's there, it's different though, it's not a managed language it does compile down to native code, it's not portable, you get all the same problems you have with other native languages like the incompatibilities that sometimes happen and Apple is pretty good about API but you sometimes have problems moving things back and forth from versions.

Carl Franklin: Yeah.



Geoff Norton: They are usually backwards compatible but not necessarily forward so things compiled on 10.5 unless you specifically link to 10.4 SDK, likely will not run on 10.4.

Richard Campbell: Right. So, where does Cocoa fit into this equation?

Geoff Norton: Well, Cocoa is a set of -- it's a collection of some of their frameworks together which is their new windowing system. Carbon was their C implementation of their windowing system; Cocoa is their Objective-C implementation of their windowing system.

Richard Campbell: Oh, okay.

Geoff Norton: So, Cocoa would be analogous to a System Windows Forms. Cocoa would be analogous to Win32/MSVC windowing and stuff. Cocoa would be analogous to Gtk#, or Gtk+, sorry, rather.

Richard Campbell: All right, so that's one of the sets of libraries that you would use as a developer to build applications on the Mac?

Geoff Norton: Yup.

Richard Campbell: Great.

Carl Franklin: Yeah, now let's talk about Mono on the Mac. What do the Mac people think about Mono in terms of the framework? Let alone the portability but the programmability of it.

Miguel de Icaza: Well, I guess that would depend on who you really ask. Some people value a lot the fact that they can, reuse the same code that they wrote for Windows on the Mac. So you can reuse your Windows Forms code in the Mac so those people don't care as much as the fine look and feel of the application. People that are aiming towards trying to get the UI to look as close as possible to the Mac and follow every guideline on design on the Mac. They typically go for something like Cocoa# which are bindings that will let you write applications that are native to the Mac. The kind of downside with going with Cocoa# is that your application will be tied to the Mac so once you start building applications with the Cocoa# API, you can still build C# code and your continuity is Mono but the application won't be able to move to Linux or to move to Windows but there is a solution, we've got a couple of companies using Cocoa# for that reason. Other people have built their own UIs like the Unity guys. They use Mono, they use some C# but this company is interesting, they actually use a variation of Boo, one of the .NET languages they use a variation of Boo called Unity script that looks and feels a lot like JavaScript but it

has some of the benefits of strong typing that C# has in terms of performance. So they use that for their products and they build that for their own APIs for doing their own interfaces. It doesn't really look like the Mac, it has some elements of the Mac but it's really their own 3D framework and their own 3D UIs.

Carl Franklin: So, Cocoa# is the Mono implementation of Cocoa, is that correct?

Miguel de Icaza: It's more like a binding. You know how Windows Forms and Windows is a wrapper around the Win32 API?

Carl Franklin: Yeah.

Miguel de Icaza: It's an object oriented wrapper around a Win32 API, so think of Cocoa# as a .NET binding to the underlying Objective-C Cocoa API.

Richard Campbell: So, the point being, Cocoa# is the way that a C# developer could make a most Mac-like application using the Mac UI?

Miguel de Icaza: Correct, yes.

Richard Campbell: And the sacrifice, of course being there's no equivalent Cocoa library in the Windows world so you're not making compatible apps at this point?

Miguel de Icaza: Correct.

Richard Campbell: Yeah, we're still sort of exploring why would we do this? Are we building original apps on the Mac and we'd rather work in C# or do we have Windows apps that we want to run on the Mac? And I guess Cocoa# is off the table if you're doing that compatibility version?

Miguel de Icaza: Correct, and I would say some people have done, some people that have some C# code, they build native UIs for their applications when they made them cross platform.

Richard Campbell: Right.

Miguel de Icaza: So you can still reuse, let's say 90% of your application, if it's all purely engine, you rewrite the 10% of the GUI sometimes the balance is different and it might be more expensive to do that but depending on how much GUI you have, you might choose to go one path for another. So if you have a lot of GUI code and you just don't want to bother with that, you'll probably go with Windows Forms or Gtk# on the Mac but if you have the budget, the time or the need to have a fully native UI, you'll go with Cocoa#. The idea here is that if you love C# or you like the class libraries or you like that API, you can still keep a high level of familiarity with the language and you can

keep a lot of the code that you already used on Windows. So that might be the reason to go for Mono on the Mac.

Carl Franklin: And also, the other one we mentioned in Geoff's bio is Objective-C#, right?

Miguel de Icaza: Correct.

Carl Franklin: And this is the .NET binding to Objective-C?

Geoff Norton: No, Objective-C# is going the other way. Where Cocoa# is, as we've said a binding straight down to the Cocoa framework to let you leverage those in C# code, Objective-C binds, Objective-C# exposes everything in the common language runtime to your Objective-C application.

Carl Franklin: Oh.

Geoff Norton: So we, it's a set of rules and bindings that do, what I said before that Objective-C is a dynamic language, it lets us do some interesting things with reflection and embeddings. What we actually do with Objective-C# is embed the Mono runtime directly into your Objective-C application without you having to do anything and then you can do things like declare an array list in Objective-C code, put things in it and pass it to other things.

Carl Franklin: And this is the kind of stuff that I was talking about, that did it exist before? To be able to have all those collections and arrays and interfaces and all that stuff that's in the framework, that's really what I'm talking about.

Geoff Norton: No, they have their own implementations and arrays and collections and hash tables and things. The advantage to doing something like Objective-C# is again, if you didn't want to necessarily use Cocoa# or you didn't want it for some other reason or you had some other code written by another group that was done in .NET, using something like this, you can save having to rewrite all of that.

Carl Franklin: I got it.

Geoff Norton: If you've got something customized and specialized to you, you can pass in your native values through the bridge and get the results back out and then present them however you want.

Carl Franklin: Makes sense.

Miguel de Icaza: Yeah, I think what is worth pointing out here is Objective-C# would be used in a case similar to what you would do in Windows with

what Microsoft calls CLR hosting or CLR embedding, I can't remember what they call it, but basically you already have an investment in native code, a lot of C or C++ code and instead of recompiling it with .NET and moving the whole application to .NET, you keep all of that code unmanaged but you want to use .NET in some capacity to extend it or you want to use .NET to script your application, for example, like when you host the CLR and then you use IronPython to script your stuff. So this is similar in that regard but it's aimed at people that have a large investment in Objective-C, in Objective-C applications and they want to expose the inner workings of the Objective-C world to the managed environment and that can be any .NET language.

Carl Franklin: Do you know the perfect formula for building and managing web sites? Follow me here. Zero effort plus Sitefinity CMS equals infinity in website development. That's right. Telerik challenges you to explore its innovative Sitefinity Content Management System and offers you a chance to win a sleek Zune MP3 player or a Sitefinity license. These cool awards could be yours if you only answer a few easy questions about Telerik's Sitefinity CMS. All you have to do is watch five short movies and see how easy it is to build infinitely beautiful websites with zero effort. You'll learn some cool facts about Sitefinity and the effortless creation of websites. So, go to www.sitefinity.com and give it a try. It's fun, it's interesting, and it can get you a free license or a free Zune.

Okay, so now let's talk about the .NET developer, the .NET developer who wants to run applications on the Mac. What's the scope of those types of applications?

Miguel de Icaza: I think that if you're coming from .NET, you are using Windows Forms or WPF for depth of application, the only thing that Mono supports today is Windows Forms and we didn't really have a lot of good support for the Mac for Windows Forms, we used to only have a driver that basically requires a traditional UNIX windowing system on the Mac so your application would require you to start up an X server inside the Mac, it didn't integrate as well as you could and it was cumbersome to use. What Geoff has been working on is basically putting your native Windows Forms driver so that it integrates with the Mac. Now this kind of integrates on the low level aspect and the next stage would be for us to design a seam that makes your Win Forms applications look like Mac OS applications, it's something in our to do list but we're not really actively working on that part yet. So basically, if you're a Windows Forms developer, what you're going to get with Mono is the ability to run these application on the Mac and the last release that we did for Mono on the Mac contain a lot of the work from Geoff but it's not going to be until our



upcoming 1.2.7 release that you're really going to get all of the driver updates that Geoff has been working on and our hope is that by Mono 2.0 which we're hoping to release by the middle of this year, we'll not only have all of the new, this whole new driver integrated with Mono and for every Mac developer to use but we're going to be shipping our IDE co-Mono developed on the Mac.

Carl Franklin: Wow, cool.

Miguel de Icaza: Then basically, we'll bring, because today you kind of have to use eMax or VI or cannot work with Xcode and force it to use Mono, well, what we're going to do with Mono develop, which is an IDE based on #develop that you guys might already know about.

Carl Franklin: Yes.

Richard Campbell: Sure.

Miguel de Icaza: You're going to get IntelliSense, I mean IntelliSense support for Visual Studio Solutions and ASP.NET and GUI designers and all of that stuff and it's really an environment design for the .NET developer on the Mac.

Carl Franklin: Define a couple of things for us, Gtk#?

Miguel de Icaza: Yeah? Gtk# is, think of it as Windows Forms for Linux people. Just like you guys have Win32, we have Gtk+, that's what it's called.

Carl Franklin: Okay.

Miguel de Icaza: And just like you guys have Windows Forms, we created a native, well a managed binding to C# called Gtk#.

Carl Franklin: Okay.

Miguel de Icaza: So most applications that you build on Linux today with Mono are with Gtk# because they look native, they integrate deeply into your desktop and so on.

Carl Franklin: Okay.

Miguel de Icaza: And Mono developed in particular is one of those Gtk# applications.

Carl Franklin: Okay.

Miguel de Icaza: So, working together with a company called Imendio, they managed to do a native Gtk port to the Mac so that Gtk integrates directly with Mac OS and that's what we're using to make Mono

developed and their Linux suite of applications run on the Mac as well.

Carl Franklin: And what is Glade?

Miguel de Icaza: Glade is a very old GUI designer. It's like the designer surface that you get o Visual Studio. We have since switched to our own managed implementation of a designer that we call Stetic but they're essentially the same thing, they're GUI designers where you draw your widgets and put your buttons and create some scroll bars and it generates an XML description of the UI.

Carl Franklin: Okay. So Stetic is built into Mono develop then?

Miguel de Icaza: Correct, yes.

Carl Franklin: Yeah. I'm looking at the screen shot. It looks remarkably like Visual Studio.

Geoff Norton: Yeah.

Carl Franklin: It's pretty cool.

Miguel de Icaza: Yeah, we're very happy with it.

Richard Campbell: So, we've got three operating systems in play here, Windows, Mac OS and Linux, maybe we need to dig in a bit to the similarities and differences between what Mac OS is doing and Linux does because obviously Mono started out as a Linux project but was it directly portable to the Mac or was there some work you had to do to get there?

Miguel de Icaza: We had to do some work. Before we were acquired by Novell, we were hired by Systems Control vendor that does embedded systems and they needed a Power PC port so they hired our company at the time to do the port so we did the port to the Power PC and then porting it from their embedded system to the Mac wasn't very easy, this was before the Mac was running on Intel.

Richard Campbell: Right.

Miguel de Icaza: So we did all of the operating system integration work, Geoff was one of the early people, before he joined Novell, he was contributing continuously to the effort and over time there was a Mac community that grew out of that effort so initially Mono was just a command line tool and it has kept drawing, now we have Cocoa#, we have Objective-C#, we're going to be launching an IDE, I mean the same idea that we're using for Mac developers. We're making sure that Windows Forms runs native in the Mac without any third party applications and so on. So there's been a long road and we finally decided to invest, when I say we, I mean Novell, to



invest seriously in making sure that the Mac was on an equal footing as the support for Linux because we know that a lot of developers are now using Macs and we kind of want to attract those developers.

Richard Campbell: For sure. I've got to think that the all out Power PC effort ultimately went to naught when Apple across to the Intel platform, so now you can take your Linux code, I mean doesn't that make things simpler?

Miguel de Icaza: Well, I don't think that it was completely a waste of time because for example today there's still a lot of Power based machines, IBM big servers.

Richard Campbell: Right.

Miguel de Icaza: Using Power and we're hoping to get Mono on the 360, we're already working on a couple of game vendors that they have ported Mono to the 360 but we want to make our own offering of Mono for the 360, some people can develop or extend their existing games with the scripting on Mono.

Richard Campbell: Wow, an alternative to XNA?

Miguel de Icaza: Well, it's different. This is very interesting actually. The situation that many game vendors are facing is that for XNA you basically write your application on top of XNA.

Richard Campbell: Right.

Miguel de Icaza: A lot of these people already have, you know, they're using already existing physics engine or very old that they were using and it's all C and C++ so they don't want to rewrite their code or retarget it to XNA. They want to keep using their existing code but they would still like to, instead of using scripting languages like Lua or REALengine or REALbasic or all of those scripting languages that are quite slow and maybe I got the names of some of them wrong, what they want to do is they want to use Mono as their VM. I don't know if you've seen the demo of the Silverlight Chess demo where they play JavaScript versus .NET.

Carl Franklin: No.

Miguel de Icaza: Yeah, I don't know if you've seen that one, but it's when you make JavaScript fight .NET in the chess game and you know it's a thousand times faster to use a managed language like C# versus JavaScript.

Richard Campbell: Sure.

Miguel de Icaza: So some of these game vendors are actually interested in Mono as a high

performance scripting engine for building the AI for their games.

Carl Franklin: If you're done with that, I have another line of questions to go down which is on the Mac, as I don't have one yet, I plan on buying one this year, believe me, but I've seen Windows apps completely integrated into the desktop with Parallels, this tool, you know what I'm talking about? So it doesn't seem like such a big deal to have Windows apps ported to native Mac OS, OS X, Leopard or whatever it is, is there a performance penalty you pay with Parallels working so well? I mean it just seems to work so well.

Geoff Norton: There's a few things here. One, you are going to pay some of the performance penalty for a virtual machine. It's not so bad anymore because it's the same architecture. The other point is keep in mind licensing talk. If a company or a person has decided to change to a Mac, if the answer for them to run their small .NET app or large .NET app that they need to for work is to go spend \$99 on Parallels and another \$400 on an XP license to be able to run in a virtual machine, that's a lot of money, when you multiply that out by a number of people.

Carl Franklin: Understood, yeah.

Miguel de Icaza: I mean if some people are happy just using Parallels, then I think that's it, I think in particular in this case it's a case where the ISV, not the end user, is making the choice, as we say, we'll support your application natively and it won't require you to run Parallels and it won't require you to go through these extra steps to get it running. I guess what happens here is really the person making the choice is different, in one case it's the end user and in the other case is the ISV.

Richard Campbell: Yeah, \$500 a seat may sound expensive until you look at a \$100,000 development effort.

Carl Franklin: Well, that's true. Good point, Richard.

Richard Campbell: I mean it's not free but it's not that expensive.

Geoff Norton: Yeah, but we're talking about taking your Win Forms application that's been written and running it without having to redevelop.

Carl Franklin: True.

Richard Campbell: I think the big thing here is how much is it going to cost to redevelop like how or what parts do we have to rebuild. If I can really take the core of my code, if I've got a 99% straight cross



compile, I'm pretty excited about that now it is pretty cheap. I'm just trying to get a feel for how much work I have to do, I mean this is where I think Cocoa# is off the table, what's my Windows Form implementation on the Mac?

Miguel de Icaza: Yeah, I guess the situation here is, although you will have to pay, I mean depending on how many P/invoke's your application has because that is the only piece that is not portable, right?

Carl Franklin: Yeah.

Miguel de Icaza: If your application is 100% .NET with no P/invoke's, then you have nothing to worry about. The moment you start using P/invoke's then that's an engineering effort that you're going to have to invest money on to make it run on the Mac.

Carl Franklin: Plus, I imagine that you can do more Mac particular things with Mono than you can with .NET.

Miguel de Icaza: Yeah, you could and again I think the issue here is the ISV will choose whether investing whatever amount of dollars is worth for them or if they just pass the consumer and say, "Well, if you want a Mac you should get Parallels plus Windows." So I guess that's truly a decision for the ISV to make.

Carl Franklin: Fair enough. Have you guys talked to Apple at all about how they could snuggle up to Mono?

Miguel de Icaza: We talked to them a long time ago and we really never took up the conversation. We talked to them shortly before they announced the x86 and when I talked to Bob he was incredibly busy with -- the whole engineering team at Apple was incredibly busy with that. I'm not sure they're very excited with Mono, in particular after their Java experience, I don't know the details about that, but I think that they want to encourage a backtrack to C as a platform as opposed to something like Java on Mac OS or .NET on Mac OS.

Carl Franklin: Yeah.

Miguel de Icaza: At the time, I remember we didn't get a lot of attraction. I don't know if that has changed or if it would be worth pursuing it based on what they've done with Java.

Carl Franklin: Yeah.

Richard Campbell: It's such a different world, the Mac world, where the same company that makes the machine provides pretty much the development environment for it and largely dictates a lot of the

rules around it and you're bringing something that's much more of the other side of the community where whatever tools you want to use, whatever languages you want to use in a sort of more open way of building a lot more code. It's a funny thing how different these two things are.

Carl Franklin: Don't you think they might be moving in that direction anyway, though, Richard? And what about you Miguel, do you think?

Miguel de Icaza: Oh I don't know, I mean I guess to a large extent it seems like Apple has had to build all of these things because they had to bootstrap their own ISV ecosystem.

Carl Franklin: Right, out of necessity.

Miguel de Icaza: At some point there was a story, I don't remember where I read it, some of those magazines where they interview Steve Jobs or something, where Steve Jobs went and tried to convince some company to pour their software to the Mac and when the company came back and said, "Well, we're not really going to do it." He decided, "Then we'll just have to build it, right?" So I think the sometimes you have to take things into your own hands. The Mac doesn't have the same ISV vehicle system that Windows has today so a lot of the things need to be bootstrapped by themselves through their own revenue. I don't think that's particularly, I mean I don't think it's them trying to force how the market behaves, I think it's more of a how to reinsure that we can survive.

Richard Campbell: To Apple's credit, one of the features of a Mac is that the software is so consistent and is relatively well written so that the experience is the same, we're all tend to use the same audio editing tools video editing tools and so forth and their quite robust in that sense as opposed to the diversity that's on the PC side that also leads to inconsistency.

Carl Franklin: Right.

Geoff Norton: I think one of the important thing to remember with that comment though is hardware footprint, on the PC side, no matter how the development tools are the underlying stability of an operating system is part of your problem and Windows tends to support every single audio card and video card and whatever out there and Linux has the same problem, where Apple being able to control the operating system and the hardware environment, they've got such a smaller subset of core devices that they can tune and focus their efforts on those and making sure that they've got good quality drivers and good support and that does change those things somewhat.



Richard Campbell: Absolutely.

Carl Franklin: Are there less layers on the Mac software wise between the application and the hardware because they don't have to have...?

Geoff Norton: I wouldn't say so.

Carl Franklin: You wouldn't?

Geoff Norton: I wouldn't say there is less software layers. I just think that, when you have 100 inputs on the top and 100 outputs on the bottom, if you can change that to 10, you've simplified your entire overall ecosystem.

Carl Franklin: Yeah, I'm just thinking of the whole driver infrastructure, having to have so many different layers because of all of the different hardware manufacturers.

Geoff Norton: Most driver implementations now aren't sort of layered they're more pluggable so you got to plug in an internal module or whatever you have for your sound driver. Apple, they use a mock based kernel which is slightly different on how things happen but it's fundamentally the same as how, they're logically the same as how a Linux kernel works.

Carl Franklin: Okay, interesting. All right, Miguel, can you walk us through the way a .NET developer can get started with Mono?

Miguel de Icaza: Yeah. As I mentioned, we're preparing a big update for the Mac users and that's what Geoff has been working on for the past few weeks, we really want to get a polished setup for everyone. Right now the idea is you go to the Mono Project you get our package from there, it's a single click install and that will get you XSP so you can run ASP .NET apps, it will get you compilers to all SDKs to start building applications including Windows Forms and you can start developing there. Now the problem is that we don't have currently an IDE distribution for the Mac and that's what we're working on right now, we're making sure that we package everything in this package and it will come with an IDE. So today, for the .NET developer there, they're going to have to learn a little bit about UNIX and a little bit of how to work in UNIX to do that. Now luckily there's a couple of support groups, there's a couple of chat rooms on the internet where you can get help and people can help you through the process. There's a couple of mailing lists and the people that did integration with Xcode which is the Mac OS X IDE for writing applications have setup manuals and instructions on how to get Mono integrated with Xcode. So there is definitely a learning curve required for starting to develop natively on the Mac. If

you're willing to not develop on the Mac, I mean if you're willing to have two machines, you can develop in Windows, continue to use Visual Studio and what you do is you do an Xcopy deployment so you copy your executables or your ASP.NET sites to your UNIX machine and then you just execute it with Mono, so that's the only difference. Instead of running app.exe, you have to type, mono_app.exe.

Carl Franklin: Is this something you can use Virtual PC for right?

Miguel de Icaza: Yes, absolutely. We also have a VMware machine, a VMware image so you can download from our website a VMware image with the free VMware player.

Carl Franklin: There you go.

Miguel de Icaza: That will get you a full Linux system and that's a Linux system image though. So that will get you a Linux system. That comes with the IDE because we exported IDE in Linux. It comes with examples for Windows Forms, I believe we'll already include Silverlight but I'm not 100% sure if we put it on the VMware image yet and it comes with a configured Apache server that will start serving your ASP.NET apps from a Linux server immediately.

Carl Franklin: Oh, that's great.

Richard Campbell: So what about getting to the Mac?

Miguel de Icaza: For the Mac you need to start with our DMG package and as I said, I mean which is an installable package, the only thing that we're missing today is the IDE. Again, the other downside is that we are not allowed to redistribute VMware images that would run Mac OS X right? The only way you could do that is with a hacked Mac OS X and we were not going to do that.

Carl Franklin: Right, right.

Geoff Norton: Our VMware image that we shipped with our Open Tuesday and our Linux development desktop does work in VMware fusion on your Mac so you could run that on your Mac.

Carl Franklin: Okay.

Miguel de Icaza: In that case, you would be basically running Linux inside the Mac to the IDE.

Geoff Norton: I mean the other side from having a second machine, Miguel.

Miguel de Icaza: Oh, oh. All right, yeah, yeah.



Carl Franklin: Yeah.

Miguel de Icaza: Sorry about that. Yes, you can do that too.

Richard Campbell: So it is possible for me to build code on my Windows box and then I can move it over to the Mac and essentially rerun it there under Mono?

Miguel de Icaza: Yeah, the same instructions that we'd posted for doing the Linux transition are exactly the same ones. I mean they apply exactly to Mac OS and in fact the Mono product manager Joseph Hill, when he heard that we're doing this call at .NET Rocks! he was like, "Oh, I wish we were doing this call in a couple of weeks when we had the installers ready," because we're hoping to have in a few weeks the full package that comes with the IDE and everything for Mac users. In a couple of weeks, it will be a one step process as opposed to today, right?

Richard Campbell: Well, by the time this show is published, it will be a couple of weeks and entirely possible that you will have all that ready.

Miguel de Icaza: Oh. Well, then interesting, then the instructions change, so just go get our Mono DMG package from the Mono website which is a Mac installer and that comes with the development tools, it comes with the ASP.NET server and comes with an IDE for building your applications and importing your Visual Studio .NET solutions.

Richard Campbell: Excellent. Of course, this whole scenario gets even more interesting when you consider the idea of just plain old developing on the Mac and developing in the .NET languages and there's a reason why .NET had become so popular, it's a great way to build software.

Miguel de Icaza: Yeah and we're very happy of the compiler support that we have now. We provide C# and Visual Basic compilers but you can use Macsoft F# which is, it seems to be the new hot thing, you can just download that one and run it on the Mac or Linux directly. IronRuby and IronPython run out of the box with Mono. So, there's a bunch of opportunities there and chances. Of course, if you're a Mac developer, you can use Cocoa# for many of these languages as well. I think it's going to be a fun way of prototyping your Cocoa applications with Python and Ruby.

Carl Franklin: Hey, total tangent here, but tell me the Graffiti story.

Miguel de Icaza: Well, that was a little bit surprising. The Taligent guys are building this new CMS called Graffiti and I've been following on and off what they do.

Richard Campbell: This is Rob Howard and Scott Watermasysk and they built Community Server and a bunch of other things too. Graffiti is something new though.

Miguel de Icaza: Yeah, yeah, it's their new product. I mean we've always followed Community Server and I guess some other popular products in the .NET world and recently we saw that they announced support for Mono and I was a little bit surprised because I didn't really know much about that. They asked a few questions but I really didn't think that they were building it then and it turns out that they have been working with Joseph Hill who just joined us at Novell as a Mono Project Manager to make sure that the Graffiti CMS would work on Mono. So I think it would be interesting for the Taligent guys is that now that they officially support Mono to be able to run this stuff in places like Dreamhost where you pay \$5 a month and you get your CMS hosted for you.

Carl Franklin: Wow.

Miguel de Icaza: So I think that was one of the motivations, being able to be on an equal footing or an equal competitive level with PHP or Ruby Frameworks out there and be able to deploy your solutions on these cheap servers.

Carl Franklin: Okay, getting back to the Mac, here. If we're developing with Mono on the Mac and let's say we're using Mono because we want to get the richness that's in the framework, in the framework libraries, can you still write to native Mac code from that if you want to just be *Mac only* and still run in Mono? Like can I still make native calls to the Mac from the Mono app?

Miguel de Icaza: Absolutely. You can use P/Invoke in the same way that you P/Invoke Win32 these days but instead of P/Invoking Win32 you end up calling native UNIX functions or you end up using Cocoa specific functions. In fact, our Windows Forms documentation P/Invokes a lot of functions from the underlying Mac OS implementation, that's how we just implemented the thing.

Carl Franklin: Oh, cool.

Miguel de Icaza: So you can definitely call every API that they document, you can call any of those in the same way that you would do it in P/Invoke.

Geoff Norton: The one point of note though is if you have, as a Mac development company out there and you've written a bunch of code in Objective-C and you want your application to be in Mono to be able to touch that Objective-C code so not Objective-

C# but the other way around. It's a bit unwieldy right now, I'm working on something in a bit of my spare time which is a set of bindings directly to Objective-C2 which will make that a lot easier. No ETA on when that will ship yet but it will make it a lot easier to take sort of an existing Objective-C library out there and leverage it from your managed code.

Carl Franklin: Are you guys working towards making Mono on the Mac a more desirable platform than the Objective-C platform?

Miguel de Icaza: Oh, absolutely and the reason of course, it would be too cocky to say that we're going to get it done this year but...

Carl Franklin: Yeah.

Miguel de Icaza: But that is the objective. I think that with the excitement that there is around the Mac, our ultimate desire is to make sure that we have more applications on Linux and to spread the use of .NET. So I think we can capitalize on the success of the Mac, make sure that we do have the Mac on an equal footing as Linux when it comes to .NET applications and if a .NET developer decides that they want to start targeting the Mac, it would be a lot easier when they've made the decision to make their software cross-platform and isolate Win32 calls from Cocoa calls from Unix calls to get that software onto Linux then convince just the developer from Windows to move directly to Linux. So I think we can capitalize on the excitement around Mac OS X to make .NET a more cross-platform developer environment.

Geoff Norton: I think one of the things that we sort of touched on today but not really stressed, which is an exciting story is with us shipping GTK# for the Mac and a GTK Plus driver which Miguel talked about earlier from the Imendio guys, is regardless where you're developing first on Windows, on Linux or on the Mac, if you're developing your GUI application in GTK# today, you can take that exact same code without changes, again P/invoke independent of this, and then the same GUI with the same interface on your Windows, your Mac and your Linux box which alleviates some of that, especially for the ISVs and other implementations because it's not now just a matter of your Windows Forms app moving over but you can write things in GTK# from the get go which will go everywhere.

Richard Campbell: It's really kind of magical actually and of course the Linux implementation that requires X Windows or is it GNOME?

Miguel de Icaza: Correct, X Windows. GNOME is just -- how do you say that? GNOME is like the Internet shell explorer to some extent and X Windows is kind of Win32.

Richard Campbell: Okay.

Miguel de Icaza: It's not an exact mapping but it's roughly that. Yeah, every Linux machine comes with X so both GTK# and Windows forms will run just with X and kind of the flavor of your UI either it be KDE or GNOME or Enlightenment or anything else, they're all just worked together because they're all really speaking X.

Richard Campbell: Yeah, X is under the hood there so it's always there.

Miguel de Icaza: Correct, yes.

Richard Campbell: That's a tremendous demo to do, to build an application, say all in Windows and then show it under Linux and show it running under the Mac OS.

Miguel de Icaza: Yeah, and we do that with Visual Studio because I think it's a more, I think people are more familiar with doing it with Visual Studio but we can do the same thing with GTK# today but yeah, we generally show the same application running on the Mac and Windows and Linux and it always makes for a good demo or showing what it's like to running on all the platforms, for example. Lutz Roeder was kind enough that now if he doesn't find Internet Explorer, it will fall back into using rich text rendering for running so we can get this really nice tool on UNIX as well.

Carl Franklin: That's great.

Richard Campbell: I'm just fascinated at the prospect that a bunch of guys with Macs building ASP.NET apps that run under Apache.

[Laughter]

Miguel de Icaza: Oh, yeah.

Carl Franklin: It's awesome.

Geoff Norton: Well I mean that's when I started using Mono. I was using a Mac at my old employer and we were doing a lot of ASP.NET work so I worked at making sure that ASP.NET works well on the Mac, it works today. It's worked for several years actually.

Carl Franklin: Wow. ASP.NET on the Mac.

Miguel de Icaza: Did you deploy on Macs or did you deploy on other kind of servers?

Geoff Norton: Primarily we deployed on VPN servers.



Miguel de Icaza: Oh, okay.

Geoff Norton: But we had to test a bunch of things on the Mac so the reason that we had switched some of our developers to the Mac was so that we could test all the major browsers and the implementation outputs of ASP.NET without them having to carry two or three machines around.

Miguel de Icaza: All right, all right.

Geoff Norton: Because we could dual boot into Linux and use Virtual Machines and things like that, especially on the Intel boxes, right?

Carl Franklin: So, what do you have left to do? You mentioned some stuff that's coming out but in a perfect world what would you like to see on the Mono implementation on the Mac side?

Geoff Norton: I was going to say, we're still finishing up some of the corner cases on the Mac driver for Win Forms and that's a big, big thing for us. Getting the IDE shipped and into the hands of people is a big thing for us. There's a couple of things that I'd personally like to do with Cocoa# that I think, especially with the release of 10.5 and Objective C2, we can make that an even better experience for developers. It works quite well right now today, as Miguel said, there's a couple of companies out there using it. I think we could make it a better, more familiar experience because trying to bind a dynamic language like Objective-C from C#, it's got some intricacies and we've learned some things that will make it a lot better. Other than that, there's a lot of work to do, there's always a lot of work to do but I think those are some of the big high level things that I'd like to see, Miguel you'll add a couple of points?

Miguel de Icaza: Yeah, our Windows Forms implementation is missing a couple of things here and there, we've improved typing the work based on reports that we get from a tool, a portability that we ship called MoMA, the Mono Migration Analyzer. So that helps us guide the direction of which APIs were missing and implementing those things. Mono, when it comes to 2.0, we're pretty much complete, we're missing a few, about 100 API calls from Windows Forms or something relatively close to that and we're hoping to have that finished by April so that will basically be the moment where we can call the thing Mono 2.0, which a lot of people have been waiting for but if you don't use Windows Forms, we've been 2.0 enabled for more than a year now.

Richard Campbell: So meantime, Microsoft keeps moving and I guess, 3.5 is the shipping version, when do I get LINQ for my Mac?

Miguel de Icaza: You can already get it. If you get 1.2.6, you get almost all of C# 3.

Richard Campbell: Wow.

Miguel de Icaza: I know that some of the LINQ operators have not been finished so we're missing a couple of LINQ operators and we didn't have the ASP expression generation stuff in place yet so we're working on that right now, as of, that was what I was working on last night with some of the members of the team but 1.2.7 which should be out in the next two months will have pretty much a feature complete LINQ and XML LINQ. We don't have a database LINQ yet so for the time being we're going to recommend people to use an open source project called DbLINQ, it's a LINQ provider for Postgres and MySQL and Oracle and SQL Server, so it's not something we built but it's an open source effort being developed by some great guys out there.

Carl Franklin: What about LINQ to XML and XML literals?

Miguel de Icaza: Oh, that one is implemented. That in fact we've had for almost 6 months now.

Carl Franklin: Do you have the XML literals in there?

Miguel de Icaza: Yeah, yeah, we have that.

Carl Franklin: Sweet!

Miguel de Icaza: Yeah and that was partly driven by our desire to complete the Silverlight support.

Carl Franklin: Yeah.

Miguel de Icaza: A lot of this stuff is being done in support of Silverlight as well.

Richard Campbell: That actually brings up Moonlight.

Miguel de Icaza: Yeah, yeah.

Richard Campbell: Moonlight, of course, I think one of your great success stories, remarkable a bunch of development in stunningly little time.

Carl Franklin: Yeah, how quickly it happened. I think you announced it on MIX where they were announcing Silverlight, didn't you?

Miguel de Icaza: Well, they announced it there and what happened is I think I was blown away just like everybody in the audience was. When they announced C#, I mean a .NET version of Silverlight

and a journalist friend of mine said, "What do you think of this?" I said, "This is awesome." And he said, "Are you going to implement this?" I'm like, "Of course we are." "Can I quote you on this?" I was like, "Yeah, of course you can." And I was like, "Oh, am I making a mistake here?"

Richard Campbell: Now what have you done?

Miguel de Icaza: But then Marc Jalabert from Microsoft France invited me. I mean I met him at MIX and he said, "Well, why don't you come to France and show Mono?" I said, "Okay, maybe." And he kept insisting, "You should really come." I said, "Okay, I will go." He said, "You can come and demo Moonlight at the keynote." I was like, "Huh? Keynote? That sounds good." So, we had three weeks to put Moonlight together to demo it in Paris, so that's what we did. It was really Marc Jalabert's fault that we implemented Moonlight.

Carl Franklin: Wow.

Richard Campbell: I mean once you get to this great experience now where this very Microsoft technology running on other platforms and then the actual execution space is on yet another platform so there I am on a Mac, surfing an Apache site that uses the Moonlight implementation of Silverlight to show that very modern looking thing. Where's WPF going to fit into this now?

Carl Franklin: I can answer that. You guys aren't planning to do WPF are you?

Miguel de Icaza: We're not. I mean the problem with WPF -- I mean we do like WPF. I think it's still big, but we do like it. The problem is that I mean I don't think that we can commit to it and deliver it in any reasonable timeframe. I just think it's too big for us to do. So, either we get a lot of the community excited about building it and that's the way we build it or we partner with someone or we work with someone to do it or WPF becomes incredibly important strategically for...

Richard Campbell: Yeah, we're still waiting to see the potential of WPF be realized.

Carl Franklin: On the .NET side.

Richard Campbell: It's a great concept but where is the app?

Miguel de Icaza: Yeah and the problem I think there is an issue of bootstrapping, right? You already have a lot of code and the WPF proposition is, you need to rewrite some of your code to take advantage of this so it's going to be difficult for existing legacy, well, not legacy but existing apps like let's say

Photoshop or Excel or any of those apps to actually rewrite their code or migrate and I think that's a, I think WPF has a really bright future but it's a few years down the line and right now we're really a small team we can't take on something of that size.

Carl Franklin: What are you guys doing on the concurrency side? Are you working with software transactional memory or anything like that?

Miguel de Icaza: You mean the PLINQ, the PLINQ stuff that was announced?

Carl Franklin: Uh-hmm.

Miguel de Icaza: So, we've looked at it with excitement but I guess we've learned our lesson at this point which is we really should not start work on implementing APIs that are still, I mean there's a word I'm thinking.

Geoff Norton: Influx.

Carl Franklin: Yeah.

Geoff Norton: They're RFPs or CFPs, Community Review or something like that.

Carl Franklin: CTP, Community Technology Preview.

Geoff Norton: That's it, Community Technology Preview.

Miguel de Icaza: Yeah, so we really don't want to do work on CTPs because when they refactor code, we have to do the same refactoring and usually, since we're a smaller team, we're lagging behind so it's kind of very time consuming for us so we tend, with a couple of exceptions, I mean some of my developers just can't stop getting interested in something in particular but we try to stay focused on released APIs. So I think as soon as LINQ becomes a little bit more stable, we'll look into it but we tried using WPF using CTPs and that was, I mean we still did a lot of it but we still wasted a lot of time chasing changes. I think we have learned our lesson now and we'll probably just stick to finish 3.5, which is not that difficult, I mean all the 2.0 operators are relatively simple. So do that, complete LINQ and finish Moonlight and when Microsoft decides to release MVC or PLINQ or ADO.NET Astoria then we'll start looking at it or when they have like a beta like a feature frozen beta.

Carl Franklin: Right.

Richard Campbell: Yeah, that makes sense.

Carl Franklin: So, we're coming to the end of the show here. Is there anything that we missed? Anything that we should be talking about in the last few minutes?

Miguel de Icaza: I don't know. I really wished that we have a Mono session at MIX or a Mono session at the PDC. I'm really looking forward to that or at BOF.

Richard Campbell: Yeah, what's up with that?

Carl Franklin: Yeah.

Miguel de Icaza: I don't know, I don't know. We've never made it to a BOF. I really would like to have a BOF of Mono at the PDC or MIX because it would be a good time to talk to developers and tell them what they can do or how they can migrate their applications and I'd rather have a developer, I mean if I was Microsoft I would rather have a developer move to Mono using .NET than have the developer choose Java, you know?

Carl Franklin: Yes, yes.

Miguel de Icaza: Anyways, I guess we haven't really been convincing enough that that's a good thing. I think it would be great if there was some grassroots support for Microsoft to give us a BOF at the PDC to talk about Mono.

Carl Franklin: Well, hopefully somebody's listening and you'll get a call.

Miguel de Icaza: Yeah, hopefully.

Carl Franklin: Well, the project is Mono. It's at www.mono-project.com. One last thing, is that a monkey or a robot or both? What is that?

Miguel de Icaza: Oh, it's a monkey, it's a monkey.

Geoff Norton: Monkey.

Carl Franklin: Okay.

Geoff Norton: Mono means monkey.

Carl Franklin: It does.

Miguel de Icaza: Yeah, in Spanish, in Spanish.

Carl Franklin: Really?

Geoff Norton: In Spanish.

Miguel de Icaza: Yeah. You know, I'm Mexican and then I didn't know that Mono had such a bad

connotation in English. I always associated it with monochrome or monochromatic, so I figured, "Yeah, Mono. Who's going to..." you know, mono, stereo.

Richard Campbell: You missed out on mononucleosis, huh?

Miguel de Icaza: Yeah.

Carl Franklin: You know I took Spanish in high school for four years and I never learned the word for monkey.

Miguel de Icaza: Oh, it's mono. So yeah, when we launched the project everybody was making some bad jokes and I couldn't get it. I'm like, "I don't understand what's going on." Until somebody spelled it out for me and I'm like, "Oh, yeah, I guess that one's good I guess."

Richard Campbell: Too late now.

Carl Franklin: You're talking about the disease I think is what you're talking about, right?

Miguel de Icaza: Yeah, you don't want to be associated with that so let's just put happy thoughts. Mono is a happy monkey.

Carl Franklin: Okay.

Geoff Norton: But it's the kissing disease, Miguel?

Carl Franklin: Yeah, that's true.

Miguel de Icaza: No, it's a happy monkey.

Geoff Norton: You're weird.

Carl Franklin: All right, guys. Thank you very much. We're out of here.

Geoff Norton: Thank you so much.

Miguel de Icaza: Okay, guys. Thank you so much for the time and the opportunity.

Carl Franklin: You bet.

Miguel de Icaza: Bye-bye. Ciao.

Geoff Norton: Bye.

Carl Franklin: And we'll see you next time on .NET Rocks!

[Music]



Carl Franklin: .NET Rocks! is recorded and produced by PWOP Productions, providing professional audio, audio mastering, video, post production, and podcasting services, online at www.pwop.com. .NET Rocks! is a production of Franklins.Net, training developers to work smarter and offering custom onsite classes in Microsoft development technology with expert developers, online at www.franklins.net. For more .NET Rocks! episodes and to subscribe to the podcast feeds, go to our website at www.dotnetrocks.com.