



Hanselminutes

Hanselminutes is a weekly audio talk show with noted web developer and technologist Scott Hanselman and hosted by Carl Franklin. Scott discusses utilities and tools, gives practical how-to advice, and discusses ASP.NET or Windows issues and workarounds.

Text transcript of show #109

April 17, 2008

ASP.NET Dynamic Data with Scott Hunter

Following up on the announcement from last week on ASP.NET Dynamic Data, Scott sits down with yet-another-Scott, in this case Scott Hunter, a Program Manager on the ASP.NET team and tries to get his mind wrapped around Dynamic Data.

(Transcription services provided by [PWOP Productions](#))



Our Sponsors

 **telerik**
deliver more than expected
<http://www.telerik.com>

 **nsoftware**
<http://www.nsoftware.com>

NET 
DEVELOPER'S JOURNAL
<http://dotnet.sys-con.com>





Lawrence Ryan: From hanselminutes.com, it's Hanselminutes, a weekly discussion with web developer and technologist, Scott Hanselman, hosted by Carl Franklin. This is Lawrence Ryan, announcing show #109, recorded live Wednesday, April 16, 2008. Support for Hanselminutes is provided by Telerik RadControls, the most comprehensive suite of components for Windows Forms and ASP.NET web applications, online at www.telerik.com. Support is also provided by .NET Developers Journal, the world's leading .NET developer magazine, online at www.sys-con.com. In this episode, Scott discusses dynamic data with Microsoft ASP Senior Program Manager, Scott Hunter.

Scott Hanselman: Hi, this is Scott Hanselman and this is another episode of Hanselminutes. I'm sitting here at the MVP Summit in Redmond, Washington, with Scott Hunter, a Senior Program Manager on the ASP.NET team. Scott is working on dynamic data right now. We just actually came from his dynamic data talk that he gave to all the ASP.NET MVPs and I had recently done a post on dynamic data and I was joking in the post that I thought a small percentage of people were going to care about ASP.NET MVC, but a huge percentage of people were going to care about dynamic data. People are using that percentage, I said 5% and 95%, but numbers I just made up, but it seems like a lot of people are using the data grid and there was a gap. The data grid didn't do something it needed to do. The dynamic data just completely takes that concept, fills it up and extends it.

Scott Hunter: Yeah. I think the big thing that we do in dynamic data is we take some of the controls today that are fairly limited. I mean if you take a grid, a grid is a great example, if you want to customize it, really it requires a lot of work in that, you know, by default you get this great, very quick page, but as soon as you decide you want to go and change the way something works or you want to add some validation, there's no validation in the grid. So, I think that's our strongest point is if you want to go modify a field and a grid, how do you do that? You got to go template each field. You got to put all these validation controls in there. One of the big things that we try to do with dynamic data is look at your data model and learn from which data model it's at. It's going to tell us what fields are required. It's going to show us foreign keys and the tables they associate with so we can do lookups and show the nice name, not the ID. We can show a dropdown list. We can do things like look at the string lengths and make sure that the UI matches the string length. If you just do those three things across a website today, it means you're going to take a grid, you're going to turn every column into a template column, you're going to drag a textbox in or a required validator in, maybe a compare validator if you're looking for an integer or date or something you want to make sure of the types to write value in and so I think one of the things we really do is

we solve a lot of that for you and we do it without doing it with Magic. A lot of fixes you could do for this would be, "Okay, we're just going to add all of this for free." The problem with that is as soon as you want to change it, what do you do? Well, one of the things you're going to notice with a dynamic data project is there will be a folder in your application called dynamic data and if you drill into that, there's a directory called field templates and that's where all the logic that I just talked about is going to exist. So, if you go into that directory, in the case of like editing an integer, there will be an integer editor, integer under edit, ASCX file in there and inside that file is going to be a textbox where somebody is going to type the integer. There will be a required field validator. There will be a compare validator and make sure it's actually an integer and there will probably a regular expression validator in there as well. Let's say you want to change that. You got some third party control vendors reach edit control that automatically prevents you from typing anything other than digits and enforces no decimal points. What you do is you go take the integer edit, you drag that stuff in there and wire it together and that change persist across your entire application versus today, you would have a grid or a detailed view and you'd go in and template every column and then go fix it across your entire app. So, by centralizing it in one spot, I think we really help things.

Scott Hanselman: Because now you're doing that in a data grid, either data grid or a repeat, or anything like that. You're doing it on a one instance, by one instance basis. You're pushing this all into one location so it's centralized, the entire app can use it, but I can say I want this column, which is an integer to use the new custom integer stuff, but this other column right next to it not to.

Scott Hunter: That's correct, yeah. So, we have our built-in field templates that exist in this directory and if you modify one of those, this can change your entire website, but at the same time you can also add your own custom one. So, let's say it's Scott Hanselman's integer edit. Then what you can do is go to your data model and by applying an attribute to your data model, you're telling us which table in your database you want this to apply to. You go, "Well, what if I don't want to change it for every time I show that table?" Well, we also let you specify that any time you're in an individual page. So, for example, on a grid, you could go in there and say, "I want the dynamic field in," and one of the parameters for dynamic field is a UI hint and in that UI hint, you can actually override the attribute that was on the model or not even half an attribute on the model and just specify it there.

Scott Hanselman: So, is a UI hint like if I had a string in the database but I'm thinking that that's really a URL, would I say in the UI hint that this is a URL is that a different attribute?



Scott Hunter: We actually have a different attribute for that. We have one called data type. One of the big negatives to me of a database today is if you look at it from the CLR world, once it comes in the database, we just say it's either a date, it's a numeric field or it's a string and string could be a Social Security number, government ID. It could be URL.

Scott Hanselman: Email, a URL.

Scott Hunter: An email address and you probably want to deal with all of this in a different way and so we actually have an attribute called data type and you can put a data type attribute on your data model and that's basically telling us what that should be. Some of the ones that we have out of the box are things like email address, URL, those are already in there. You can also add your own. So, what we do is we actually have an enumeration that has about 15 or 20 built-in values, but the data type attribute also takes a string so you can do custom ones. If you type in like URL, it will actually go to that field template directory and see if we find a control called URL and if we do, it will automatically display it. If not, it will fall back with standard text.

Scott Hanselman: All right. Well, this is a good idea. Let's do this. Let's kind of do a pretend Hello World application here, but I want to -- I say file new and I pick the dynamic data project type. I can use the wizard if I want and it will build a lot of this for me. I can point it to a database and pull some information on it and it will give me -- we've seen this on the websites and people can look at the screen cast that I'll put links up for and we'll get an administration grid, edit, all the crud for free. Let's say that one of those fields is a string, but I want it to be an international phone number. So, what do I need to do to say that this database that has a column that has a string is really an international phone number?

Scott Hunter: Yeah. So, what I would do in that case is the first thing you're going to do is you're probably going to go to your app code directory. You're going to right click and say, "I want to add a new class."

Scott Hanselman: We're here at the MVP Summit. We got a bunch of people who have just apparently successfully compiled an application and decided that that's worth cheering about, so we're filming here in the conference center.

Scott Hunter: Wrote their first Hello World.

Scott Hanselman: Sorry, you were saying?

Scott Hunter: Anyways, so yeah, what you do is you go and you would add a new class to your project. We support LINQ to SQL and Entity Framework. Both of these generic partial classes for

all the tables in your database and so in this case what table are we talking about?

Scott Hanselman: Well, let's just say that I've got a customer and he has a phone number, but let's say that they're international phone numbers. They're an unusual format.

Scott Hunter: We got a customer table so what we do is we would go and override the customer table, the customer object that's automatically created in the data model and so I would go add a new class, call it customer and make it partial.

Scott Hanselman: Oh okay. So, I got a LINQ to SQL model or LINQ to Entities and it's got an automatically generated partial customer.

Scott Hunter: That's correct.

Scott Hanselman: I'm making the rest of it, the other part.

Scott Hunter: That's right and the reason we're creating our own class is because you could go right into the designer file created by the LINQ to SQL editor, but the negative would be then if you recompile your application, you're going to wipe that change out. So, that's the whole idea of partial and why it was added in ASP.NET 2.0.

Scott Hanselman: Exactly. So, we're going to keep that little extra bit of metadata off to the side in another file.

Scott Hunter: That's correct.

Scott Hanselman: Okay.

Scott Hunter: So, we're going to go over there and create that class and one of the negatives of the CLR today is there's not a great way to add metadata to an existing member in a partial class and the way we saw that is we actually created a metadata class that maps to your existing class. So, what you're going to do is after you created this customer class, you're going to create a customer metadata class and you're going to go put an attribute on your customer class that says, "Hey, my metadata is on this customer metadata class."

Scott Hanselman: Which I assume could be anywhere in your assemblies.

Scott Hunter: Anywhere on your assemblies or in your own...

Scott Hanselman: Okay. So, I could put all this metadata in one place.



Scott Hunter: Exactly. You could go take all this metadata, put it in some library somewhere and just go wire it up at the end.

Scott Hanselman: Okay.

Scott Hunter: And then what you would do is take any members from your class that you want to add metadata to. You just create a blank placeholder inside of the temporary class or metadata class.

Scott Hanselman: Okay.

Scott Hunter: So, for example, in our case it would be the...

Scott Hanselman: The international phone number.

Scott Hunter: International phone number so I would create object international phone number get set and then on top of that, I would actually add the data type attribute.

Scott Hanselman: Okay.

Scott Hunter: And since international phone numbers not one of the ones that's part of our built-in enumeration...

Scott Hanselman: Okay.

Scott Hunter: I would open a string up and say international phone number.

Scott Hanselman: Okay. So, at this point, I've taken my data model and I've added a little metadata class and I'm starting to annotate it with hints that I'm going to need because we're trying to keep things dry, we're trying to not repeat ourselves. We're going to stay in one place that this column is an international phone number, but we haven't really declared what an international phone number is.

Scott Hunter: That's correct. We have two choices now. We have, by default, this thing is going to be shown using our text control because technically in the database, it's a text field.

Scott Hanselman: Okay.

Scott Hunter: So we have two choices. We can go modify the existing text template and if I was going to do that, I would load up the text.ascx.cs file and inside of there I can actually write a simple LINQ query to say, "I want to look at the attributes that are specified on this field and see if there's a data type attribute. Ah, I found the data type attribute and it's international phone number. Well, because of that, I'm going to change my rendering now instead of admitting just the actual text value, how about I throw

a hyperlink up and automatically put the right stuff in front of the hyperlink to try to have it dial a phone?"

Scott Hanselman: So, I think I see where you're going here. We've declared that there's this thing, this new data type called international phone number and we're saying we can do one of two things. We can either piggyback it on the existing string_ -- it would be string or textbox.

Scott Hunter: It would be text.ascx.cs.

Scott Hanselman: Okay. So, we could extend the existing textbox, the dynamic textbox, the ASCX and say within there if it's a textbox do the old-fashioned thing versus if it's an international phone number, it needs this one additional piece of work.

Scott Hunter: Correct.

Scott Hanselman: Maybe an extra label or a special validator.

Scott Hunter: Or show it as a URL.

Scott Hanselman: And if I'm guessing where you're going with this, or I could make an entirely separate ASCX that's custom to international phone numbers.

Scott Hunter: That's correct. My other choice would be if this type is so different, I might just say add a new control and so I would go create a control in the field of directory called international phone number, at which point not only do I create the international phone number control, but I can also do an under edit version and an under insert version, which allows you to override how we display that when you edit and how we display that when you insert.

Scott Hanselman: Okay. So, you brought up something interesting in underscore that the idea that there is this convention such that it's data type underscore and then the action?

Scott Hunter: That's correct. So, data type by itself is the display, _edit is for editing and _insert is for inserting.

Scott Hanselman: Okay. So, when the grid is in that state or when the field is in that state, this is what will display.

Scott Hunter: That is correct.

Scott Hanselman: Okay, because I might want to have like a rich text type and have a really rich textbox freetextbox.com or like Telerik, one of our sponsors, use like a really complicated rich editor, but I might want it to be an IFrame when it's not being edited or display it in some other...



Scott Hunter: Exactly. We pretty much give you the ability to customize this as much as you want. I actually want to jump off for a second and go to a different path.

Scott Hanselman: Okay.

Scott Hunter: We've talked a lot about grids and detailed views.

Scott Hanselman: Yeah. There's a whole bunch of stuff underneath this that we're completely crossing over.

Scott Hunter: I also want to talk about the fact that we do template controls as well. So, one of the grid controls that was brought out in ASP.NET 3.5 was the list view control which for all the people that, you know, a lot of folks have been using our controls, they wanted to use a repeater because it will give them customization but as soon as you start trying to do things like sorting and paging and stuff like that, well, you end up writing, you know, 200 or 300 lines of blue code.

Scott Hanselman: That's a good point.

Scott Hunter: And the next thing you know is the repeater has gotten unwieldy and I think that's what list view really helps you because it gives you all the power of repeater where basically all the market is controlled by you, but you still get all the features like sorting and paging and stuff part of the control. We also support list view and form view, which are the two templated controls that we support, which means you don't have to feel like you're limited to just a grid.

Scott Hanselman: That's a good point. I think that people when they initially saw data grid, myself included, thought, "Oh, it's auto generating a website and its scaffolding." That's one way of pigeonholing it, but it's bigger than that and then I've indirectly pigeonholed you into the data grid, but it's bigger than that. I was talking with Peter Blum, Peter Blum who makes the validators and he was explaining to me that there's so much underlying support for metadata that one could build something entirely new from just the underlying classes you guys have within dynamic data, but don't let me get too far off the path. So, within a list view, I have control over my own markup. I could make some kind of a list of stuff that is not at all a data grid. It doesn't look anything like a data grid.

Scott Hunter: That's correct and that's the point I want to make sure is I don't want people to pigeonhole this technology and go, "Oh, we're just grids, we're just detailed views." No. If you want to control all the markup, you can control all the markup. The form view allows you to place the controls

anywhere on the page you want. List view repeats over data.

Scott Hanselman: So, if I got a list view and I'm doing the markup maybe with I want to do it all CSS friendly and I'm using an ordered list, list item, controlling it myself. What do I put in my template to get dynamic data features?

Scott Hunter: Right. So, what you're going to do is earlier when we're talking, we're talking about in grids and detailed view, we use what's called a dynamic field and we do that because if the existing controls have a bound field or a checkbox field, so we were trying to follow that pattern. Once you get to the templated controls, we have a dynamic control and the dynamic control, basically you specify the column that you're interested in and you can also specify the mode and obviously we have three modes. We have a read only mode, we have an edit mode, and an insert mode. This makes a lot of sense in the case if you're building a template for like a list view, it's got different templates for each of those modes.

Scott Hanselman: Right, because a new is different than an edit.

Scott Hunter: That's correct.

Scott Hanselman: Hold that thought. I'm just going to take a quick second to thank our sponsors and we'll be right back.

Hi, it's Scott Hanselman here. Hope you're enjoying the show so far. I'm coming at you from another place and time. Sorry to interrupt the show, but I wanted to let you know that putting together a podcast like this every week isn't free. Folks that pay the bandwidth bill is Telerik. They make the show possible and they also make some pretty cool products like Telerik Sitefinity. It's a development platform for constructing websites, community portals and intranets built all on ASP.NET 2.0, so you're using the various well-known goodies like master pages, membership services, data model provider, things that you already know. It's pretty flexible. You've got a very robust core that you can customize. You can plug in anything that you want from complex applications for CRM or just a little widget that displays the weather. If you're not big into the code thing, that's cool too. You get a full set of features out of the box like workflow, multilingual sites, content versioning that can all be edited without code. There's also a whole bunch of pluggable modules and components for news, blogs, forms, polls, lists, this is all stuff that you can do without code and it's a pretty good looking product as well. You've got a nice web 2.0 administrative interface that lets you as well as your boss who's not technical be really productive. So, check out sitefinity.com and we'll get you right back to the show. Thanks a lot.



So, we're talking about the list view and you were heading in that direction.

Scott Hunter: Yeah. I was going to talk about dynamic control and what it really does for you.

Scott Hanselman: Because that's the heart of the system, isn't it?

Scott Hunter: Yes. Actually, underlying everything in dynamic data is a dynamic control even though I was mentioning earlier we had this dynamic field. If you look at the source code for dynamic field, all it really does is take the dynamic control in itself and it's looking at the grid or the details view to figure out which mode it's in. In the case of a list view or a form view, you control the mode. So, as I was going to say is when you have a dynamic control, what you're going to specify is the column you want to work on and then mode, be it read only, insert, edit and what we look at is we look at, okay, you say that the column is X. We're going to look at your database and see what the model says for that column, it's an integer. That's going to tell us to go look up the integer field template inside of our field template directory and then based on the mode, we're going to pull the right version of integer. Basically, that's specifying, you're basically telling us which field template to use. So, you still get all the benefits of being fully templated, control of your markup and whatnot and you get the power of dynamic data still. I think that's really an area that I want people to make sure they understand. I've got one more area that I think people always when they see our first demos, most of the first demos of dynamic data are the scaffolding.

Scott Hanselman: Yeah.

Scott Hunter: Basically, you drag a model in, you register your data model and it's like one line of code, you get this full application built up. The audience we're missing there is "what if I have existing web pages?"

Scott Hanselman: Exactly. That was the question how do I add dynamic data to existing sites without messing stuff up? How do I put my toe in the pool and see if the water is nice without accepting it for all of my data access?

Scott Hunter: Right and I think we give you that benefit. I don't want to go through all the steps of -- actually, what you would do, I mean the overall steps are you drag this dynamic data directory in, you make a few changes in web.config. What I really want to get down to is I've got this page in my application, what do I change in the page?

Scott Hanselman: Exactly.

Scott Hunter: So, we have this new control, kind of like a script manager. We now have a dynamic data manager and you put this control on your page and then any controls you want to have or functionality, you register with this control. So, for example, I'm going to add a dynamic data manager to my page and I've got a grid view and I've got a list view and I want both...

Scott Hanselman: And they work already.

Scott Hunter: They already work.

Scott Hanselman: Okay.

Scott Hunter: And I want to turn on this magic field template stuff that I've been talking about.

Scott Hanselman: Right, but just for a couple of things.

Scott Hunter: Just for a couple of things.

Scott Hanselman: Okay.

Scott Hunter: So, what I'll do is I'll go and let's say I just want to turn dynamic data on for the list view.

Scott Hanselman: Okay.

Scott Hunter: So, in my page_init and my code behind file, I'll say dynamic manager 1, and I'll specify the name of the list view.

Scott Hanselman: Okay.

Scott Hunter: And once you've done that, you basically flipped the switch to turn on dynamic data access for the list view.

Scott Hanselman: So, that I can start using the dynamic fields.

Scott Hunter: Yes, dynamic control.

Scott Hanselman: And the control.

Scott Hunter: Inside of your templates and your list view.

Scott Hanselman: Okay.

Scott Hunter: That's all it takes.

Scott Hanselman: That's it. So, there are a couple of other folders underneath this dynamic data folder. I noticed that when I made a scaffolding, I had paging and filtering and all those things happen automatically.



Scott Hunter: Yeah. We have a couple directories. One of the directories is called content. Content is, as we started building dynamic data, we rendered this problem or we wanted a pager. We wanted some other controls and stuff and so what we did is we created a directory called content and gave you the source code to those controls as well because we said, "Not only do we want to give you a pager, not only do we want to give you a control for doing a dropdown list of filters, but one of goals in dynamic data is never to make you feel like we built something that you're going to fall off the cliff on or it's like 'okay, I've been using it up to this point and I'm going to die.'"

Scott Hanselman: So, you know, that's the thing. Someone who -- I mean like I said, well, I work for the company for six months, but I gave a talk about dynamic data in New York and I was saying how most Microsoft demos involve someone dragging something onto the page, drag a data grid over, hook some stuff up and you wave your hands and you push F5 and like, "Hey, look! I did a complete application," and then by the time the audience is saying, "Hey, but wait! I want to customize... hey!" You've left the stage at that point and it's over. When I think about that kind of canonical, old style example where, "Oh, it's totally customizable until it's completely not customizable." It's cool that there's more and more people. I mean you haven't been on the company all that long either. I guess more and more people are getting to the point that we had this pain, as people on the outside, so now we're coming inside to make sure that that doesn't happen to anyone else.

Scott Hunter: True. It's kind of funny because we actually had some raised eyebrows when we first were working on this because some of the feedback that I know the ASP.NET team had moving into ASP 2.0 was "let's make a blank project, start with the least amount of files as possible."

Scott Hanselman: Exactly.

Scott Hunter: So, here, you create this dynamic data project and we get this directory full of all these files.

Scott Hanselman: Right.

Scott Hunter: The reason we get this directory full of all these files is instead of hard coding all these things absolutely under your application, how about we just give you the source code to all the pieces you might want to touch and tweak?

Scott Hanselman: Exactly.

Scott Hunter: So, if you don't like the way our filters look, yeah, go in the content directory, modify the filter. If you don't like the way the pager looks, it's such a user control.

Scott Hanselman: Exactly.

Scott Hunter: Go in there and modify the behavior. If you don't like the field template, that's the whole point of the field template is to actually give you the source code. Right now, a bound field or a checkbox field or an image field, you have no control over those. As soon as you exceed their capabilities, you're done.

Scott Hanselman: You know, speaking of images, you showed me a demo that I thought was pretty cool where you were using a database where one of the columns had a binary format, that had some funky picture in it. It was a non-standard thing. It was like not a .GIF, not a .PNG, an old style thing. So, just assume that you have a database with many columns and one of the columns has something funky in it. It could be PDF, could be a data format that's your own and you want to invent a visualization for that. You made one called DBImage and I thought that the way that you would create that was pretty interesting.

Scott Hunter: Yeah. So, what dynamic data does is as I had mentioned at the beginning of the talk, we were talking about we look at the data model and try to figure out what we should show for each of the fields. Obviously, it's in the binary. We have no way to visualize that and so by default we hide it, but just because we were hiding it by default doesn't mean you can unhide it. The library you're talking about is DBImage and it was basically my attempt at making a nice field template control that display images out of a database.

Scott Hanselman: Yeah.

Scott Hunter: So, the way that works is you go and put your own field template control and understands that type and then on your model is you're going to go and put one of those UI hints we talked about earlier and say, "Hey, when this column shows up, I want to use DBImage for this column," and then instead of hiding that column, now we're going, "Okay, since you've told us you know how to display it, we'll turn that rule off and we'll go find the field template that supports what you said."

Scott Hanselman: Suddenly, you knew what this thing is.

Scott Hunter: Exactly.

Scott Hanselman: But because it was an image, I thought it was interesting because you had to do two things. You had to render the image tag, the image source sequels, but then that source sequels had to point to somewhere, it had to actually go and get the image, so you ended up writing a handler that would go back into the database and get that as well.



Scott Hunter: Yes. I'm actually really proud of the handler. I mean that's one of the things I would recommend somebody. If you want to see some neat code, go look at the handler. What's cool about the handler is we started going to this path of I want to show an image from the database. Well, I don't want to write SQL because as soon as I write SQL, I'm making assumptions about your database and this great feature that came out in .NET 3.5 is LINQ and so suddenly, I can build this LINQ query and by building a LINQ query, I'm not locked to is it SQL Server, is it Oracle, is it MySQL. I don't have to worry about that because whatever product they have on the back-end is going to automatically convert to LINQ query to the correct query.

Scott Hanselman: Right. It will just work. It's a comfortable layer of abstraction that, you know, we know about the database as we're using LINQ to SQL or LINQ to Entities, but we don't know that much, we're not too intimate with the database.

Scott Hunter: That's right, so if you look at the source code for the handler, all we're really doing there is specifying the primary key of the table and the column that contains the image and then if you look at the source code, it's actually dynamically building a LINQ query. So, most of the time when you see people doing LINQ queries, you just build some texturing, but this is actually a dynamic LINQ query. So, if you want to see an example of how you would actually build an expression tree, to build a query by hand, that's what this does, but as I said, the neat thing about that is it means that it's not tied to any particular storage format. This should work on Oracle, it should work on Entity Framework. There is no tie ends, so that's something I don't think you could have done pre-.NET 3.5. Before, I would have to had an Oracle version, a SQL version, a MySQL version, etc., and LINQ has really allowed me to pull that out.

Scott Hanselman: So, all of this stuff that we've been talking about so far is going to be available in an upcoming update to the framework so people will get this in their hands as soon as possible. What are some of the things that you're starting to think about where you could take this? Are we going to be able to use this in Silverlight? Am I going to be able to use this in other places?

Scott Hunter: Yeah. That's one of the things that we -- when we started working with this, we actually met with a bunch of other teams at Microsoft and started talking about a lot of the concepts and stuff we were talking about that makes sense way outside of ASP.NET and so all these attributes that you're going to put on your data model let's you tell dynamic data what to do. They don't live in system.web. They live in System.ComponentModel.DataAnnotations and that was done on purpose because we thought other teams would want to use this. So, I can already tell

you that some other teams like the Silverlight team are looking at maybe supporting this in the future where you would take the same attributes, spread them all of your data model and then when you merge your data with Silverlight, you would get the same validation and all the other behaviors that we support.

Scott Hanselman: Very cool. I'd like to be able to do this over maybe an existing data model that I've got. Maybe I have my own ORM framework or I'm using NHibernate or, you know, just plain old objects of some kind and somehow be able to annotate them and enable it such that I could do this kind of dynamic system.

Scott Hunter: Yeah. Well, one of the things that we are working on is, first off, our v1 for sure supports Entity Framework and LINQ to SQL. We are trying to put the right hooks in so if you're SubSonic or somebody else and you have your own data model, you would have a way to write a provider to plug in and we would be able to figure out what your data looks like. To give you an example of what that really means is, obviously, part of what our technology does is it looks at your data model and crawls over all the tables and stuff and then builds an internal abstraction that contains just the information that we need and so that's the hook we're trying to provide is you would write that call where, okay, walk off your model and fill in this data structure we give you and then based on that, it should enable a lot of the other functionality of the dynamic data.

Scott Hanselman: So, it's a generic method to provide the metadata about your model and saying, "Tell me about yourself."

Scott Hunter: That's correct.

Scott Hanselman: Maybe a reflection over the database.

Scott Hunter: That's right.

Scott Hanselman: Very cool.

Scott Hunter: For version 1, it's probably some low level hooks, but for version 2, I think we're going to try to do more.

Scott Hanselman: That's the scenario that you're going to definitely try to push for the average Joe, but maybe we'll figure out some funky advanced way to do it for the first release.

Scott Hunter: Yeah. I think for the first release, if you're SubSonic or NHibernate or whatever, you would be able to write a provider. I don't think the average Joe would be able to write that provider. It's going to require some hardcore technical knowledge to do that.



Scott Hanselman: Okay.

Scott Hunter: But I think when v2 comes around, hopefully we could just consume an arbitrary collection of objects.

Scott Hanselman: Very cool. So, at the time that this podcast is going to be coming out here at the end of the second week of April, folks can go and get a preview of dynamic data up at Code Gallery and I'll put the links to all of the different things, the screen cast, your blog, David's blog up on the Hanselminutes show site, and then soon we'll be able to have this built in to the system so we won't have to download any preview bits.

Scott Hunter: That's correct. You can grab it on Code Gallery right now. It will be up there until we actually RTM.

Scott Hanselman: Okay.

Scott Hunter: And our actual plan is probably to keep updating even past RTM, so we're going to move right into version 2.

Scott Hanselman: Okay, cool.

Scott Hunter: And we're going to continue to use Code Gallery to extend the first version.

Scott Hanselman: So, it's kind of like the ASP.NET AJAX stuff where it was built in, but they're still going and they are making changes and those will be web downloadable at some point.

Scott Hunter: That's correct. I really hope that, you know, as you said, you're new to Microsoft, I'm new to Microsoft and one of the things that I think that Microsoft needs more of is none of these, you know, we ship two CTPs a year and we ship it and don't have a lot of feedback. We would much rather have a lot of our new code out there for the public to see more often and give us more feedback on. Our code can only be as good as the feedback we get.

Scott Hanselman: Certainly, and you and I don't know enough to not try to do these things, so the fact that you're attempting to do this and get that loop tightened up is pretty cool and hopefully they won't beat us down.

Scott Hunter: They've been pretty accepting thus far so I'm happy with it.

Scott Hanselman: I think that dynamic data and kind of juxtaposing it with the MVC is just another example of some of the stuff that you guys are trying to really make happen inside Microsoft, so I'm pretty stoked about it and I'm going to actually try to do a series of screen casts, some videos and stuff and put them up on ASP.NET as we get closer to release.

Scott Hunter: Great. Thanks a lot.

Scott Hanselman: Cool. Thanks a lot. This has been another episode of Hanselminutes and we'll see you again next week.