CREATING NEW ROWS WITH THE PROBINDINGSOURCE AND .NET CONTROLS

John Sadd Fellow and OpenEdge Evangelist Document Version 1.0 March 2010





DISCLAIMER

Certain portions of this document contain information about Progress Software Corporation's plans for future product development and overall business strategies. Such information is proprietary and confidential to Progress Software Corporation and may be used by you solely in accordance with the terms and conditions specified in the PSDN Online (http://www.psdn.com) Terms of Use (http://psdn.progress.com/terms/index.ssp). Progress Software Corporation reserves the right, in its sole discretion, to modify or abandon without notice any of the plans described herein pertaining to future development and/or business development strategies. Any reference to third party software and/or features is intended for illustration purposes only. Progress Software Corporation does not endorse or sponsor such third parties or software.

This document accompanies the video presentation that continues the series on updating data using the ProBindingSource. This session shows how to use binding source events to add new rows to a table whose data is displayed in an UltraGrid control in a .NET user interface.

Along with the **AllowEdit** property that determines whether data entry into UI controls will be allowed, If you look at the ProBindingSource properties in the Properties View of Visual Designer, you see that there's a property called **AllowNew** that you can set in the binding source to have it manage whether UI controls that are bound to it will allow a request to create a new row. As with **AllowEdit**, it's **True** by default, so it doesn't need to be changed for this session's example, but you'd want to set it to **False** for a read-only user interface or one that supports updates but not creates.

The other side of supporting creates in the user interface is a key property in the **UltraGrid** that needs to be set to enable the user interface to respond to a request for a new row. This is under **DisplayLayout**, and then **Override** in the **Properties View** for the **UltraGrid**. The property is called **AllowAddNew**. As the name implies, this determines whether – from the grid control's perspective – the user can request a new row through the control, and also how that new row is represented in the grid.

🚰 OpenEdge Editor - Samples/Updal	table/Cu	ustomerUltraGridUpd.cl	s - OpenEdge Arch	itect - C:\Gui4D	aNet					_ O ×
File Edit Design Navigate Search	Project	OpenEdge Run Windov	v Help							
] 🗂 • 🔛 🛆] 🔏 📖 🗞] \$	» • O) • 🏊 • 🛛 🛷 • 🗍 🗄	1 - N - 🌣 🔅	•						
📑 💞 OpenEdge Editor										
Strategy Resources 23	- 0	CustomerModel.ds	Model.ck	Custome	UltraGridUpd.ds (Desi	ign) 🕮	Customer	UbraGridU	pd.ds	- 0
	28 7	·							Toolbox	7
🖻 👹 Samples		CustomerUltra	sGridUpd						CustomConi	trols
B 3 ^Q Referenced Assembles		9		Hedatabi	O Customer Grid				Microsoft C	ontrole
B-JL Procedure Ubraries	_	1		Optication	Costones and				B BoosE dos	Controle
E Code		Text	Test	Text	11		11		G OpenE oge	Controls
🕀 🎃 Updatable		Text	Text	Text		10			Upent dge	Oltra Controls
CustomerModel.cls										
CustomerUltraGridUpd.ct	s									
S IModeLos		1								
🔡 Outine 📅 DB S 💷 Prop 🕮		0								
18 2.	- 📭 🍸									
moUltraGridCustomer: Infragistics.Win.Ult	raWin 💌	1								
Properties Events		1								
E ActiveRovCelAppearar	-						-			
E AddRowAppearance						-				
E AddRowCelAppearance	-	Start!	Click Start! to I	segin designing						
Default		0			0					
Yes		Status:								
No										
TemplateOnTop		1						•		
TemplateOnTop/withTabRepeat										
TemplateOnBottom	- 1	🙄 moBSCustomer								
FieldAdRowOrBoltom	-									
Allowed deliver	_									
Determines whether the user is allowe	d to a	🖸 Console 🔝 Proble	ems 🕄 🧟 Tasks							****
) (* 🕒										

As you can see, there are a number of different options in its enumeration. You can learn about these in the Infragistics documentation, but one that presents a reasonable UI affordance for inserting new rows is **FixedAddRowOnBottom**. Like other Infragistics names and enumeration values, this one states pretty clearly what it does. If you save and compile the form with this property setting, and run the form, you can see below that there's a permanent overlay at the bottom of the grid that lets you request a new record just by clicking in this special row. This seems like a reasonable UI that lets you add a row independent of where it will wind up sorting in the grid itself.

🗑 OpenEdge Editor - Sam	ples/Upda	tabl	e/EustomerUltraGri	idUpd.cls - OpenEdge	Architect - C:\Gui4Dot!	Net				_0,
File Edit Design Navigab	a Search	Pro)	ect OpenEdge Run	Window Help						
📬 • 🔛 📥 🔏 🗉	1 🕲 🕇	ÿ	0 • 💁 • 🛛 🛷	'•]∄•% •\$	$\phi \rightarrow \phi$					
😭 👫 OpenEdge Editor										
🖑 Resources 🔢		6	° 🗆 💽 CustomerM	odel.ds 💽 IModel.	.ds 🛛 🐇 CustomerUltr		🔀 💽 CustomerUlbraGrid	supd.ds		° (
			~					Tool	box	Ģ
🖂 🐸 Samples			🔺 📄 🛃 Custon	nerUltraGridUpd				1 Ci	stonControls	
🗄 🧬 Referenced Ass	embles	1	ustomerUltraGridU	lpd					Contraction of the Contraction	
I Procedure Librar	ies							. M	crosoft Costrols	
🗄 🗁 rcode				U	pdatable Customer Grid			± 0;	enEdge Contro	ds .
Treeview			CustomerFirstName	CustomerLastName	CustomerBirthCountry	CustomerBirthdate	CustomerGender	± 0;	enEdge Ultra (ontrols
Updatable			Robert	Kennedy	USA.	07/13/1979		-		
Customerik	oer.os raCid led c		Peter	Wagner	Germany	09/14/1907	R			
Model.cls	ratariuupu.i		Alice	Washington	USA	05/19/1902				
Cutine Papers	Dran 5		Patrick	Lewiston	USA	03/19/1970	9			
	A		Eric	Myers	Gemany	06/08/1950	R I			
	# 2		Laura	Torres	Gemany	07/16/1984				
CustomerUltraGridUpd : Prog	ress.Windo		Carol	Murphy	Gemany	12/01/1982				
Properties Events		-	Thomas	Janking	Gemaw	12/29/1954		-		
E Padding	0, 0, (Justin	Watson	USA	01/27/1983		=		
previousState	Norm.		Adhur	Tuner	1164	04/11/1000				
RightToLeft	No		THE P							
Fight I oLeftLayout	P 808	2						-		
ShowInTarkbar	Toue	<u></u>								
E Size	614,	· · · ·					-	-		
SizeGripStyle	Auto _	3180	ur.				12	1		
StartPosition	Windo	Ocres D						2		
StatusBarText on moUlts	aSta									
Tag			- modSU	Justomer						
Text	Custo	Dec	-1							
Toomou	P AUA		-							
The test associated with the	control		E Concole	🕐 Broblens 😚 🎣 Ta	wike .				-	001
	- united to	-			nea				+	
😷 🖻					Writ	able				

You could type in values for a new row here and save it, but that won't do anything meaningful yet. There is some measure of default behavior, but as with the **AutoUpdate** property, it won't get your new row back to the database or execute any other initialization code, so it's necessary to go back to Visual Designer to add some supporting code to provide the behavior needed.

The object-oriented approach to building an application really requires you to work from the bottom up in your design, which is a good discipline, so this means going back to the Interface that defines all the methods a Model class uses in data management. Here I add a method definition named **CreateModelRow** for the code that will handle a request for a new row in any class that manages data. It passes in a buffer name, in case the Model manages data in more than one table.

```
INTERFACE Updatable.IModel:
METHOD PUBLIC VOID FetchData (INPUT pcFilter AS CHARACTER ).
METHOD PUBLIC VOID SortData (INPUT pcSort AS CHARACTER ).
METHOD PUBLIC HANDLE GetQuery().
METHOD PUBLIC LOGICAL SaveData(INPUT pcBufferName AS CHARACTER ).
METHOD PUBLIC LOGICAL FormatColumn (INPUT pcColumnName AS CHARACTER,
INPUT pcColumnValue AS CHARACTER).
METHOD PUBLIC LOGICAL CreateModelRow (INPUT pcBufferName AS CHARACTER ).
END INTERFACE.
```

And now I can implement that method in my one example Model class. As before, from the **Source** menu, I can select **Override / Implement Methods** to get the matching skeleton code generated for me. The binding source is coordinated with its underlying data source to make the method's implementation very straightforward.

M	ETHOD PUBLIC DEFINE VARIA	LOGICAL CreateModelRow(INPUT pcBuffe BLE cCustomerID AS CHARACTER NO-UNDO.	rN	ame AS CHARACTER):
	DO TRANSAC	TION ON ERROR UNDO, LEAVE :		
	CREATE	ttCustomer.		
	ASSIGN	ttCustomer.CustomerFirstName	=	"First Name"
		ttCustomer.CustomerLastName	=	"Last Name"
		ttCustomer.CustomerBirthCountry	=	"USA"
		ttCustomer.CustomerBirthdate	=	1/1/1950
		ttCustomer.CustomerGender	=	TRUE /* = "male" */
		ttCustomer.CustomerLicenseDate	=	1/1/1950
		ttCustomer.CustomerLicenseExpiryDate	=	1/1/1950
		ttCustomer.CustomerID	=	GUID. /* Unique char ID */
יד	httCus END. /* tr RETURN TRU	tQuery:CREATE-RESULT-LIST-ENTRY (). ansaction */ E. /* No checks for errors yet. */		

First the code creates the new row in the temp-table – the same would work for a database table, but I'm illustrating a separation of the client-side model from the database itself. Then the method assigns whatever initial values are needed. This could include calculating a line number or any other value for the user. In fact, you can see that one line uses the **GUID** built-in ABL function to generate that long character string that the AutoEdge database uses as a unique key field in every table. Finally, the method just executes the ABL **CREATE-RESULT-LIST-ENTRY** function on the query that the binding source is using. This works very neatly, because when you add a new row through a ProBindingSource, it automatically positions the query to the new row being added, so **CREATE-RESULT-LIST-ENTRY** adds it to the end of the query without having to reopen it. For this reason, you shouldn't do anything fancier than this – don't try to explicitly reposition the query or the binding source; just do what is shown here and it will all work properly.

Next we have to go back to the ProBindingSource and its events. First let me point out an event you *don't* want to use.

DpenEdge Editor - Samples/Updatable/Cu	stomerUltraGridUpd.cls	- OpenEdge Archi	tect - C:\Gui4D	ANet					- 0 ×
File Edit Design Navigate Search Project	OpenEdge Run Window	Help							
] 📑 • 🗄 🛆] 🔏 📖 🕲] 🏶 • 🚱	• 💁 • 🛛 🛷 • 🛛 🖄	- 🗄 - 🏠 🔶	• 🕂 •						
🔛 💐 OpenEdge Editor									
🕰 Resources 🖾 🗖 🗖	CustomerModel.cls	IModel.cls	Customert	RraGridUpd.cls (Design) 12 💽	CustomerUltrad	GridUpd.cls		- 0
					_		Toolbox		4
🖻 🖉 Samples 🔺	EustomerUltrat	iridUpd					E Custo	mControls	
∃ 3 ⁹ Referenced Assemblies			Herdalahk	Customer Grid			H Micro	unit Controls	
B-JL Procedure Libraries			0,000,000				a Doen	Edan Controls	
Treeview	Text	Test	Test	1.1		11	C Open	Edge Control	
🗉 🍅 Updatable	Text	Text	Text	_/_/			Upen	coge ditta Co	muois
CustomerModel.cls					100		1-		
CustomerUltraGridUpd.cls									
I PODELOS									
2 Outine 10 06 S D Prop 2 - 0									
₩ 2↓ B3 °									
moBSCustomer : Progress Data BindingSource 💌									
Properties Events				1.4.4	101		-		
AddingNew	*				M.		1-		
BindingComplete					_				
CancelDreateRow	Start!	Click 'Start!' to b	egin designing	the UltraGrid					
DuteriCharoed									
CurrentItemChange	Status:								
DataEnor							_		
DataMemberChang	4						<u>)</u>		
ListChanged	· · · · · ·	h.							
	moBSCustomer								
Pholindratiource Designer									
CurrentChanged									
changes.	🖸 Console 🖹 Problem	is 🖾 🧟 Tasks						10	- 0
1									
J.O. G						1			

The first one in the list here is called **AddingNew**, which sounds like exactly what you would want to use to create a new row. But remember that our OpenEdge-specific ProBindingSource is an extension of a standard .NET binding source object, so it inherits that object's events and properties. Some of those are useful, and some don't really apply to an OpenEdge application. **AddingNew** is one of those that don't apply, because if you look at its **EventArgs**, you'll see that it has a **NewObject** property which is an instance of any System.Object, appropriate for dealing with object-oriented data, perhaps, but not the relational data in your database. We decided not to hide these inherited events and properties from you, and under certain very limited circumstances they might be useful, but generally you should stick to the ones that support OpenEdge applications, as they're described in the OpenEdge documentation and materials such as this.

The event you *do* want to use to handle an add request is **CreateRow**. Let's take a look at its parameters. As always, the ClassBrowser will give us a quick overview of the properties that get passed in. And that's of type **CreateRowEventArgs**.

There are several properties you can use in your event handler. The first is **BandIndex**. If your data is in a grid that supports multiple display bands that display data from different tables, this integer property gives you the zero-based index into the list of bands. That's one way you can identify which table the request is for.

Alternatively, there's a **BufferHdI** property that holds the handle to the buffer for the table, and a **BufferName** property that holds its name. You could use either of these as well.

The last property to look at is **Created**, a logical flag that you set to signal whether the underlying create in your temp-table or database table succeeded or not. This tells the binding source whether to tell the UI controls to cancel the create or not on their end. So all the event handler method has to do is invoke the

Model's **CreateModelRow** method. The call passes in the **BufferName** property value as a placeholder, though remember that in my case I have only one buffer. and if that method returns **True** the caller sets the **Created** property to **True** to signal success. Note that this simplified example doesn't check for specific errors.

```
METHOD PRIVATE VOID moBSCustomer_CreateRow( INPUT sender AS System.Object,
INPUT args AS Progress.Data.CreateRowEventArgs ):
    IF moCustomerModel:CreateModelRow(INPUT args:BufferName) THEN
        args:Created = TRUE.
    ELSE args:Created = FALSE. /* No real checks for errors yet. */
    RETURN.
END METHOD.
```

Now all the code is in place to try out the support for new rows. If run the form again and click in the row that's reserved for new row creates, the first thing I see is that the initial values are automatically transferred by the binding source up to the grid to display. I can select any of the grid's cells and override the default value.

DpenEdge Edite	or - Samples/Updi	diab Dec	le/CustomerUltraGri iest_CoarEdas_Bus	dUpd.cls - OpenEdge (Window - Mala	Architect - C:\Gui4DoU	iet.			_ [] ×			
I P . O A	A Li Qa La	Ha .		- 8 - 5 - 15								
TR M Court for		~	· · · · · · · · / //									
El C openEdge	Editor				-		100					
Sesources 🖾			CustomerM	odel.cis 🔄 IModel.	.cls 🗳 CustomerUltr	aGridUpd.cls (Design)	CustomerUltra	Gndlpd.ck 🗶				
			428	METHOD PRIVATE	VOID Initialized	Component():[*			
🗉 👹 Samples			178									
🖲 🧬 Refere	nced Assemblies		CustomerUltraGridU	pd				1				
B-JL Proced	ure Libraries	_										
E 🗁 rcode				U	pdatable Customer Grid							
18 6 TreeVie	w		CustomerFirstName	CustomerLastName	CustomerBirthCountry	CustomerBirthdate	CustomerGender	L	1/			
Updata	coe toroauthodal de		Robert	Kennedy	USA	07/13/1979	R		1			
	tonerfibra/frid kel e		Peter	Wagner	Germany	09/14/1987 05/19/1982	R	System.Object,				
- IM	del cis		Alice	Washington	USA							
St Cuelina St DB	s Down S		Patrick.	Lewiston	USA	03/19/1970	R	THEN				
	and the Property to		Eric	Myers	Gemany	06/08/1950	V		_			
	14 4		Laura	Tores	Gemany	07/16/1984		s yet. */				
Finite Internet	Value		Carol	Murphy	Gemany	12/01/1982						
derived	false		Thomas	Jenkins	Gemany	12/29/1954	9					
editable	brue		Justin	Watson	USA	01/27/1983	9					
last modifie	January 12, 2010 1		Antes	Travar	11SA	04/11/1990						
basis	CAR APANANE AN	14	NewCustFirst	Last Name	USA	01/01/1950						
Datte	dsCustomer.i	T	1									
path	/Samples/dsCuston							1				
\$12.6	907 bytes	Sta	ture:						*/			
Console © Fiscal results: Table Console © Fiscal results: Table												
) 📬 🗈					Writable	Insert 13	10:1					

If I scroll across the cells, you can see in the next screenshot the rest of the initial values, including the **GUID** object id field. I can change another value such as one of the dates, and see the calendar control that is automatically applied to a grid column that's bound to a field defined as a date.

😁 • 🗉 👜 J	1001		•] 1 •] 1 •		*			
🝸 🏭 OpenEdge E	ditor							
Resources 😒		C Cu	stomerModel.cls	C Model.cls	CustomerUltraGridUpd.cls (Design)	CustomerU	traGridUpd.cs 🗶	-
		1 🍇 😤 🛛 42	METHOD I	PRIVATE VOID Ini	tializeComponent()	:0		
🗉 🎒 Samples		178						
🖲 📌 Referen	ced Assemblies	🛃 CustomerUlt	raGridUpd				×	
B-JL Procedu	re Libraries							
🗉 🗁 rcode				Updatable Cust	omer Grid			
E E TreeVier	Y .	SustomerBirthdate	CustomerGender	CustomerLicenseDate	CustomerLicenseExpiryOate	CustomerID	•	
Contraction of the contraction o	ee IomerModel de	13/1979		03/28/2002	05/26/2014	1905c74d-1385-a		
	omeri IbraGridi Ind a	14/1907	R	11/06/2000	01/19/2027	1905c74d-1305-a	System.Object,	
 IMo 	sel.ck	19/1982		03/17/2003	08/18/2013	1905c74d-1385-a		
Contine 100 res		19/1970	P	01/22/1994	08/16/2025	1905c74d1385-a	THEN	
Coone 181 ce :		708/1950	R	05/06/1977	06/01/2025	1905c74d-1395-a		-
	19.3	216/1984		09/24/2008	06/23/2024	(905c74d-1385-a	8 yet. */	
hoperty	Value	101/1992	E	10/29/2005	03/29/2025	1905+7441305.h		
E Info	feb a	200 11 05 4		01/21/2000	00/04/2007	1005-7411005-0		
editable	hue	23/1354	M	0173172001	02/24/2017	13056740-1305-8		
last modifie	January 12, 2010 1	2//1363	M	05/24/2008	08/25/2015	19U5C/40-1305-8		
linked	false	211/1980		105./17/2002	108/26/2011	1976-764.1396, s		
location	C:\GuHDotNet\San	01/1950		0120121350	0170172001	d5lac629164912	⊻	
name	dsCustomer.i	1			🐔 January, 2001 🕒	P		
gine	gil2 huter				un Mon Tue Wed Thu Fri Sat			
940	100 01000	Status:			1 2 3 4 5 6		h	/
		201	B METHOD 3	PRIVATE VOID M	7 8 9 10 11 12 13	rCellUndate(
		2.02	INPUT	sender AS Syst	14 15 16 17 18 19 20 21 22 23 24 25 26 27			
		203	INPUT	e AS Infragist	28 29 30 31 1 2 3	ellEventArgs):	
		204	IF 1	soCustomerHode!	4 5 6 7 8 9 10	Cell:Column: H	Kev.	
			•		Today: 1/28/2010			•
		🗉 Co	nsole 💽 Problems	23 🖉 Tasks			le.	V 🗆

Selecting another row causes the **BeforeRowUpdate** event to fire just as it does for any other update, and that saves the changes I made to my new row back to the temp-table and from there to the database table. You can see the update message here in the **StatusBar**, and if I re-sort the data by clicking on a column header, I can scroll down and find my new row in its correct sort position.

Constitute Editor	 Consider the de 	hadd be a first	above and the office	and to all a large of	DeserTrian Assi	Back Coldman	No.			L Initi
UpenEugeronce	 Samples, Opera 	Consistent .	stomere	Hindow A	Upentoge and	itett = Li(usianasi	set			ALC: N
ne Lac source	navigace search	Project	upenuage mut	Without r	10p					
📑 - 🗷 🗂 I	70.011	> 0	• 💁 • 🛛 🖉	·) 🖞 -	8 000	9				
😭 👫 OpenEdge E	ditor									
🕼 Resources 🗵		- 0	Customer#	todel.ds	Model.cls	🖏 CustomerLib	aGridUpd.cls (Design	🖇 💽 CustomerUbra	Gridlipd.ds 🕷	° (
		2. 7	42.0	METHOD I	PRIVATE VO	D Initialize	Commonent ()			-
E 😂 Samples		-	178	HUTLESS .	Parmin	1010101100	composition ()			
B 🖓 Referen	ced Assemblies	🔛 Custo	merUltraGrid	Jpd				- I I ×	1	
I JL Procedu	re Libraries	_								
🗉 🗁 rcode					Updat	able Customer Grid				
E C TreeView	W	Custor	nerFisiNane	Customer	rLastName (ustomerBirthCountry	CustomerBithdal	te CustomerGender		
Updacad	No.	Mary		Hunt	66	many	12/23/1971			1
- Cus	tomerUltraGridUpd.c	Melis	58	Geller	Ge	many	03/17/1949		System.Object,	
C IMo	del.cls	Melia	58	Burns	Ge	many	10/25/1959			
				Сок	Ge	mary	05/23/1956	9	THEN	
- course car ee		Mike		Simpson	US	A	04/01/1958	V		_
	14 .4	Narx	7	Miler	US	A	02/25/1957		s yet. "/	
F Info	Yalue	New	CustFirst	NewLast	09	A	01/01/1950	2		
derived	false	Patri	sk	Alexander	Ge	many	02/08/1983		0	
editable	true	Patri	sk	Lewiston	US	A	03/19/1970	V		
last modifie	January 12, 2010 1	Dahi	-6	Caurat	n.,	PT APRI	07/22/1905			
location	C:\GuidDotNet\Sav	*						F		
name	dsCustomer.i	11						•	1	
path	/Samples/dsCuston								1	
size	987 bytes	Status: Ne	enCustFirst Newl	Last update si	uccessful.					*/
			200	SATRONIC	resigner.					
			2018	METHOD 3	PRIVATE VO	ID moUltraGri	dCustomer_Aft	cerCellUpdate(
			202	INPUT	senger As	system.objec	C, UltraNinGrid	CallEvent loga 1		
			204	IFI	moCustomer!	todel:FormatC	olumn (INPUT -	:Cell:Column:Ke	v.	-
			•							•
			Console	Problems	23 🕢 Tasks				3	
- 4 . 573										
) 🚩 🕒						Writable	Insert	178 : 1		

So the support for adding new rows through the grid requires just a few lines of code.

Next I need to look at the code to allow me to cancel the create of a new row. If I were to start to enter a new row and then press Escape to cancel it, the binding source isn't able to do everything automatically that I might need in order to remove the row I no longer want, so I need to put in explicit code to handle that event.

First I add another method to the **IModel** interface to handle a cancel of a new row create, and call it **CancelCreateModelRow**. The new method uses the same buffer name parameter used in the **CreateModelRow** method:

```
INTERFACE Updatable.IModel:
METHOD PUBLIC VOID FetchData (INPUT pcFilter AS CHARACTER ).
METHOD PUBLIC VOID SortData (INPUT pcSort AS CHARACTER ).
METHOD PUBLIC HANDLE GetQuery().
METHOD PUBLIC LOGICAL SaveData(INPUT pcBufferName AS CHARACTER ).
METHOD PUBLIC LOGICAL FormatColumn (INPUT pcColumnName AS CHARACTER,
INPUT pcColumnValue AS CHARACTER).
METHOD PUBLIC LOGICAL CreateModelRow (INPUT pcBufferName AS CHARACTER ).
METHOD PUBLIC LOGICAL CancelCreateModelRow (INPUT pcBufferName AS CHARACTER ).
END INTERFACE.
```

In the **CustomerModel** class, I then implement that method. The code needed is very simple, and parallel to the statements in **CreateModelRow**. It explicitly deletes the temp-table row that corresponds to the row in the grid, the same row that **CreateModelRow** just created. And to remove the row from the result list, it uses the query's **DELETE-RESULT-LIST-ENTRY** method, returning **True** to signal that the delete succeeded.

```
METHOD PUBLIC LOGICAL CancelCreateModelRow( INPUT pcBufferName AS CHARACTER ):
DELETE ttCustomer. /* That's the only buffer. */
httCustQuery:DELETE-RESULT-LIST-ENTRY ().
RETURN TRUE.
END METHOD.
```

Next I need to return to the form and create an event handler for **CancelCreateRow**. Double-clicking on the event in the **Events** tab of the Visual Designer **Properties View** generates the event handler method skeleton code. Inspecting the parameters that are passed in, I see the **EventArgs** class is another subclass called **CancelCreateRowEventArgs**. This supports the same **BandIndex**, **BufferHdI**, and **BufferName** properties, just as **CreateRowEventArgs** does, but of course no **Created** property, because the create of the row is exactly what I'm canceling out of.

Without adding any special code for unexpected conditions, the event handler just invokes the new cancel method in the model, and display a message if for some reason it doesn't succeed.

METHOD PRIVATE VOID moBSCustomer_CancelCreateRow(INPUT sender AS System.Object, INPUT args AS Progress.Data.CancelCreateRowEventArgs):	
IF NOT moCustomerModel:CancelCreateModelRow (args:BufferName) T MESSAGE "Row cancel did not succeed." VIEW-AS ALERT-BOX.	'HEN
RETURN.	
END METHOD.	

To try this out, I rerun the form and start the add of a new row. Remember that as soon as I click in this special row at the bottom of the grid, the row is added to the temp-table. I can enter a few values before I decide this isn't what I wanted to do.

		_											
📆 OpenEdge Edit] Opentidge Editor - Samples/Updatable/CustomerUltraGridUpd.cls - Opentidge Architect - C:\Gui4DotNet												
File Edit Source	Navigate Search	(Pro)	ect Og	penEdge Run	Window Help								
] 😁 • 🔛 👜	140011	3. •	0 -	Q. •] 🔗	- 🔄 - 🥡 -	$\phi \phi$	• 🔿 •						
📑 👫 OpenEdge	Careful dae Ldtor												
Strate Resources 23		-	° 🗆 🛛	CustomerMo	del.cls 🔄 IModel.cls 🚳 CustomerUltraGridUpd.cls (Desig				CustomerUltra	vändspillets 🕱	- 0		
	← ⇒ & ≡	8	7	41							1		
E Samples A 420 NETHOD PRIVATE VOID InitializeComponent():													
🖲 📌 Refere	inced Assemblies		luston	nerUltraGridUp	юd					1			
B-JL Proces	ture Libraries	_											
🖲 🇁 rcode						Updatab	le Customer Grid						
E 🗁 TreeVi	ew		Custom	erFirstName	CustomerLastNar	ne Cu	tomerBirthCountry	CustomerBitthdate	CustomerGender	1			
B 🧽 Updat	able		NewCi,	utFirst	NewCustLast	USA		01/01/2001	2	1	*/		
	stomerModel.cis		Robert		Kennedy	USA		07/13/1979	3	1			
- R IM	odel.ds		Peter		Wagner	Gem	any	09/14/1987	N				
Se Outine Se Di	5 Prop 5		Alice		Washington	USA		05/19/1982			- 1		
Un control mon			Patrick.		Lewiston	USA		03/19/1970	R				
	14 4		Eric		Mperc		any	06/08/1950	9	cNome) THEN			
Froperty Info	¥809		Laura		Torres	Gemany 07/16/1984			-BOX.				
derived	false		Carol		Murphy	Gem	any	12/01/1982					
edtable	true		Thoma	ŝ	Jenkins	Gem	any	12/29/1954	R				
last modifi	January 12, 2010 1:		Iccobian.		Watson	TIRA		01/27/1982	17				
location	C:\GuHDotNet\Same	*	First No	ame to cancel	Last Name	USA		01/01/1950	• 9	1			
name	dsCustomer.i								•				
path	/Samples/dsCustome												
size	987 bytes	Stat	us:							6	-		
				201 1	VisualDesig	ber.					,		
				2028	METHOD PRIVAT	TE VOID	moBSCustom	r_CreateRow(]	NPUT sender &	S System.Object,			
			- 1	203	INPUT args	AS Pro	gress.Data.(reateRowEvent&	rgs):				
			- 1	204	IF moCust	comer Mo	del:CreateMo	delRov(INPUT a	rgs:BufferNes	e) THEN	. Ľ		
			t l								<u> </u>		
				Console	🕻 Problems 🕄 🏼	Tasks				*	0 0 U		
J 😷 🖪							Writable	Insert 19	1:69				

The way the **UltraGrid** works is that the first time you press **Escape** any change to the value for the current cell is undone. And the next time you press **Escape**, the whole row is undone. This is when the **CancelCreateRow** event fires. So after pressing **Escape** twice, the newly added row is gone, not just from the new row line in the grid, but from the temp-table as well.

DpenEdge Edite	or - Samples/Upda	C.C.	le/Cur	tomerUltraGri	dUpd.cls - OpenEdge a	Architect - C:\Gui4Dot/	iet		_ [] ×
File Edit Source	Navigate Search		iect (penEdge Run	Window Help				
📬 • 🖪 🌰	140013	۶.	0	- 💊 - 🛛 🔗	- 🖗 - 🤤 - 🕸	•			
📑 🖏 OpenEdge I	Editor								
🖏 Resources 🕮		6	• 🗖	CustomerMa	odel.cls 💽 IModel.	.cls 🛛 🆏 CustomerUltr	aGridUpd.cls (Design)	🕙 CustomerUltra	Grdupd.ds 🕷 📃 🗖
				41				_	<u> </u>
🗆 🐸 Samples		_	-	42.	METHOD PRIVATE	VOID Initialize	Component():[]	
🗄 🔗 Refere	nced Assemblies	H	Eusto	merUltraGridU	pd			_ [] ×	1
B-JL Proced	ure Libraries					adatatis Castanaa Raid			
	~		Custo	mer Fruiklama	Curterned autiliane	Customer Customer and	CustomerRithdate	Customerillander	
🗉 🇁 Update	ble		New	ner ssrvane	NewCurif at	LISA	01/01/2001	Customeruender _	
- Cu	stomerModel.cls	Robert			Kennedu	USA	07/13/1979		,,
	stomerUltraGridUpd.c	-	Peter		Wagper	Gemany	09/14/1907	2	
			Alce		Washington	USA	05/19/1982		L _
Pr: Ongue All De	5 Prop 2	H	Patric	k.	Lewiston	USA	03/19/1970	<u> </u>	
,	14 3		Eric		Mperc	Gemany	06/08/1950	2	rNome) THEN
Property	Value	H	Laura	,	Tones	Germany	07/16/1984		-BOX.
derived	false		Carol		Murphy	Gemany	12/01/1982		
editable	true	H	Thom	182	Jenkins	Gemany	12/28/1954	F	
last modifi	January 12, 2010 1:		Look		lu/stron	116A	01/27/1903		
location	C:\Gui4DotNet\Same	٠						F .	d i i i i i i i i i i i i i i i i i i i
name	dsCustomer.i	•						•	[
path	/Samples/dsCustome								
\$426	967 bytes	Stal	Nus:						<pre>////////////////////////////////////</pre>
				201	<pre>@VisualDesigner</pre>				-
				2020	METHOD PRIVATE	VOID moBSCustom	er_CreateRow()	INPUT sender &	S System.Object,
				203	INPUT args AS	Progress.Data.	CreateRowEventJ	trgs):	- many -
				204	IF BOCUSCO	ernodel:createn	DOGETHOM(TWP01 S	args: Burrernas	ej TREN
				Concela (Dubling 17	uta			<u> </u>
		_	_	Console	Process co	1940			* 0
12 🔁						Writable	Insert 19	1:69	

If I had completed the add operation and saved the row, then of course I would have needed support for a row delete operation to undo it at that point. And that is the subject of the next session in this series on ProBindingSource support for row deletes.