

Re: Invalid variant type conversion

Source: <http://coding.derkeiler.com/Archive/Delphi/comp.lang.pascal.delphi.misc/2004-01/0694.html>

From: Bjørge Sæther (*bjorge_at_hahaha_itte.no*)

Date: 01/22/04

Date: Thu, 22 Jan 2004 01:27:33 +0100

Maarten Wiltink wrote:

> "Bjørge Sæther" <*bjorge@hahaha_itte.no*> wrote in message

> *news:K7sPb.2888\$Ed.46110@news4.e.nsc.no...*

>

>> *The NULL value of a database may be used for this purpose, but you*

>> *can't reflect this in application programming unless you limit your*

>> *coding to types that have a NULL value.*

>

> *Not true. Just change TAnyType to PAnyType.*

This indicates no memory is allocated for the variable. Any other meaning is something you put into it. You may return a nil value from a function...meaning ? Why is it that VCL doesn't use this ? I believe it's because within regular programming, there would be no agreement upon what this 'nil' would mean other than "no memory allocated". When using databases, where nil/null values will exist if they're not actively prohibited, they're IMHO (In My Holy Opinion) a constant source of trouble in situations where it's not obvious what they mean. A NULL value in a field related to a field in another table of course means "not related to any foreign record". A NULL in a PRICE field of course means "no price set". In these examples, NULL is certainly different from ZERO. But what does a NULL mean in STREET_ADDRESS field ? Does it mean that the person doesn't *have* a street address (inn small places of Norway, quite a few people don't use any street address) or that it isn't known ?

I would recommend not to trust string value of NULL to mean anything in applications built by standard Delphi controls, at least.

>> *BTW, using database fields for holding CSS data would normally be*

>> *done in a BLOB/MEMO field text, rather than putting CSS' properties*

>> *in dedicated fields. There is no such thing as a NULL value in CSS*

>> *format, only a missing value ("tag"), AFAIK.*

>

> *It was just an example. The example needed the assumption of storing*

> *single CSS properties in a field to make sense. That's not a*

> *limitation of the idea of adding a unique element nil to any domain;*

> *after all, *you* wanted an example and not an academic discussion.*

This is an academic discussion. In this discussion I stated "There is no need for an explicit NULL value in strings, as they *have* a natural NULL value whereas numbers don't." In the context of programming, that is. Strings represent something like "speech". When I ask you: "What is your opinion on the occupation of Iraq?" and you answer "", I would say that "Maarten gave no answer". You say "No, I *did* answer, I answered """. The question is *why* there was no answer from you, and this could be "You was asked, but didn't answer" or "You weren't asked". In this case, a NULL could mean "not asked" and "" could mean "has no opinion". But there are more possibilities, like: "refused to answer", or "the interviewer forgot to ask" or "the operator forgot to type". Whatever, in the Delphi form, it all looks the same – an empty edit or memo box. Now *I* used an example, I just don't understand why this makes the discussion less academic ? Oh, yeah, one could say that "the problem with trying to grasp it all with a few hitting examples is that the conclusion may be wrong.

Just like I believe Windows application would be a lot more useful if they didn't come up with those stupid "do you want to save" – boxes, I believe programming tools should be more oriented towards practical tasks than focusing on providing a totally clean model. Clean models are good only if they're detailed to a useful level.

Another example: Delphi allowing omitting the ^ when accessing what a pointer type points to. Just for the sake of nicer looking code and a few less buttons to press. Brilliant. Most of the time, it's not to be mistaken. While it's of course use for a Pboolean type, so that you may distinguish between 'NULL', 'false' and 'true', making this obligatory would be a step backwards. It doesn't pay off what it costs. And that's the one argument I'm using: NULL is the same as "" in a database. Or – should be. Whenever you need to distinguish, do so by adding a second field or variable. 99.9% of the time, you're only interested in the literal representation, which would be "" for both. For numbers & dates I would need it in *any* application representing real–world data, for booleans I'm not entirely sure. I tend to mean that a simple checkbox should not visually represent the value of "unassigned", as you have a hard time making people understand what a grey square means in practice. A 3–item radio group would be better, because you could then describe the meaning of "unassigned". To illustrate the problem with number's NULL representation: Every second month I fill in a Web form with the period's tax figures. I don't remember from time to time whether leaving a blank makes the validator scream about missing values, or if it will be interpreted as "0". In Excel it will be interpreted as "0", in databases the bottom line will be NULL if there is no explicit handling of setting NULLs to 0. But I *do* understand what leaving a date field blank means – as there is no way a NULL date could be used for any calculations. It *has* to mean "I have no idea", it can't be a certain date.

> [...]
>> *...but there is one thing I don't need: 'IsNull' for string fields.*
>> *The empty string tells it all. I will not write an application that*
>> *gives NULL value of a string field any meaning.*
>
> *But I will, if it suits me.*

Only you'll have to do something in addition to just distinguishing between the two – you need to **explain** it. Because the GUI would not show it – your NULLs or ""s would not survive a round with ordinary string values, controls or string routines. In the database they would persist. Your CSS field would need to interpret your possible string "states", but might as well would they need a 3rd or 4th state, as your "inherited" is a state giving information about the field, not the string itself. It could be "Inheritable", it could be "Optional", or any other property.

>> *Databases require NULLS for referential integrity, but the
>> distinction is superfluous, since using a " for a key value is
>> madness.*
>
> *Who's talking about key values?*

I was, as database's rules actually impose their own meaning on NULLs: It means "Not related to foreign key field" i the situation where two tables are related via a foreign key, and it means "Not subject to uniqueness evaluation" in (some) databases when checking UNIQUE constraints. It means "Sort first", resp., "Sort last" when sorting.

> *Databases are for storing data, and it
> may suit me to distinguish the cases of a string field having the
> empty string or _no_ string as its value.*

It may suit me in certain cases, too. But – from experience I state that the 0.1% of cases where one would need just 1 "state" indicator – NULL for string fields are too few to justify using this distinction. because it creates trouble & extra work. CHAR fields and Char variables may need the NULL value handling, while VARCHAR fields and string variables don't.

>> *Thus, the "inherited"/"override" problem should be handled otherwise.*
>
> *What's wrong with handling it by using "no value" to denote "not an
> override in place"? You (effectively) asked for an example; I gave you
> one. Don't be too surprised if I give one that would really work.*

Well, I wasn't clear here. Your example was more an illustration of the need for combining a variable with an according 'state' – variable, and that because strings do not have an apparent meaning of NULL, the chance is small that the NULL stuff would solve recurring tasks. About as small or big as the TComponent.Tag solving recurring tasks in Delphi programming.

>> *How to show it in a GUI ? I developed a special vertical grid type,
>> having an extra column of check boxes indicating whether the value
>> was to be left unchanged or specifically set – e.g. to EmptyString.
>> No control can show the difference between the two for a string
>> value, unless some extra functionality is added.*
>
> *Okay, so you don't know of any currently existing (single) control
> that can. Derive one. It's not hard; one of the nice things about*

- > *Delphi is that deriving components isn't hard. Still thinking of my*
- > *CSS example, a different background colour could indicate that the*
- > *displayed value is inherited, and a context menu might have an option*
- > *"revert".*

You need *another* control to do this; e.g. a checkbox, a menu item, a keyboard shortcut + a text telling you what it means or effectively does.

- >> *When you 'Clear' a TMemo using your keyboard, what value do you write*
- >> *in the underlying field ? You have to decide, right ? It's not*
- >> *obvious at all, it's an utterly meaningless distinction.*
- >
- > *You keep saying that, in the face of my example, when **you**, in the*
- > *best biblical tradition, called it "good".*

I didn't mean "You" as in "You, Maarten", I meant "people" or "programmers".

- > *Let's also not forget*
- > *something else: there need not be a one-to-one mapping between what's*
- > *in the database and what's in the GUI. In the simplest case,*
- > *"inherited" is not a CSS property value itself, it's a pointer to*
- > *another property's value, and *_that_* value will be used for*
- > *rendering. In less simple cases, a database may use a very different*
- > *data structure and representation than the one the user eventually*
- > *sees in a presentation layer, with several more layers in between.*

..but NULLs still exist for technical reasons: for a) databases using them as described, b) variants being used e.g. for OLE/COM calls , c) datatypes with dynamically allocated memory, to indicate whether memory is allocated or not. The reason is **not** that all variable handling requires a 'state=NULL' to go along with it. If so, we would have tri-state logic in Delphi, float types/routines handling nulls, etc...

I'm not religious about this. I just try to point out that string types are actually different from other types here, and – if my understanding of the problem is adequate – I think databases should be changed according to this. Nothing would please me more than if someone could convince me I was wrong; Then I would do the NULL handling with Joy & Great Pleasure. Amen.

--
Regards,
Bjørge Sæther
bjorge@haha_itte.no

I'll not spend any money on American Software products
until armed forces are out of Iraq.