Subject: Core: Null handling incoherent?
Posted by kohait00 on Wed, 04 May 2011 08:46:33 GMT
View Forum Message <> Reply to Message

hi all,

i'm wondering why

template <class T> bool IsNull(const T& x) { return x.IsNullInstance(); }

and not

template <class T> bool IsNull(const T& x) { return x.IsNull(); }

is there a reason for it? name clashes??
analogue to upp philosophy it should be latter case, (Xmlize() { x.Xmlize() } etc.)

some classes define IsNullInstance(), some do IsNull(), it's kinda 'not clean'

(background: i'm tackling Null handling in terms of extension of Value with other types on user side (i.e. float), where the Null handlig is the major problem.)

EDIT:
it's also the point of what Null actually is.. is it only to be seen in context with Value handling? because all classes interacting with Value the classes need to know about Value, but Value.h defines some of the interface handling with the types as well. so it's mixed. (i.e. String.h has template definition of IsNull())..

maybe to think of null handling as kind a independant from Value and define it in Defs.h

template <class T> void SetNull(T& x) { x.SetNull(); }
template <class T> bool IsNull(const T& x) { return x.IsNullInstance(); }

so both, Value.h and all the others are aware of that concept..
this probably would also make sense to move Nuller concept to Defs.h

i'm currently making some changes, just trying to get trhough it properly..

Subject: Re: Core: Null handling incoherent?
Posted by kohait00 on Wed, 04 May 2011 09:39:50 GMT
View Forum Message <> Reply to Message

attached is a patch for discussion. the IsNullInstance naming issue is not adressed yet.

benefits:

* more clear coherent interface because IsNullInstance, SetNull are obligatory
* users can use RichValue<> as container for even own intrinsic types like float.., specifying themselves the null handling.
(but float f = Null; still wont work, since it has no converter, but SetNull<float>() can, which is a nice price to pay.
* specifying AsString<T> they can use the Value::ToString for it as well, i.e AsString<float> is already specified..

i'd have considered to have

template <class T> void SetNull(T& x) { x = Null; }

to make SetNull method optional, but Null is kind a wired with Value, which, at level of Defs.h isn't known yet..

File Attachments

1) patch0.patch, downloaded 279 times

Subject: Re: Core: Null handling incoherent?
Posted by kohait00 on Wed, 04 May 2011 11:38:46 GMT
View Forum Message <> Reply to Message

tried it with SetNull() { x = Null; } which works fine as well, which eliminates the need of SetNull specialisation for the intrinsic Values.

here is the complete patch for it, it includes the moving of the Nuller stuff to Defs.h

File Attachments

1) patch1.patch, downloaded 329 times

Subject: Re: Core: Null handling incoherent?
Posted by mirek on Fri, 06 May 2011 07:52:59 GMT
View Forum Message <> Reply to Message

kohait00 wrote on Wed, 04 May 2011 04:46hi all, i'm wondering why

template <class T> bool IsNull(const T& x) { return x.IsNullInstance(); }

and not

template <class T> bool IsNull(const T& x) { return x.IsNull(); }

is there a reason for it? name clashes??

You would not be able to use "IsNull" for other objects in any of T methods, without using 'Upp::' prefix -> very annoying...

Subject: Re: Core: Null handling incoherent?
Posted by mirek on Fri, 06 May 2011 07:55:30 GMT

View Forum Message <> Reply to Message

kohait00 wrote on Wed, 04 May 2011 05:39attached is a patch for discussion. the IsNullInstance naming issue is not adressed yet.

benefits:

* more clear coherent interface because IsNullInstance, SetNull are obligatory

How have you defined IsNullInstance method for 'double'?

See, the obligatory interface is IsNull and Null asignment. IsNullInstance is just helper - it is easier to write it as method than as external function. SetNull was sort of forced by you.

Mirek

Subject: Re: Core: Null handling incoherent?
Posted by kohait00 on Fri, 06 May 2011 08:43:00 GMT

View Forum Message <> Reply to Message

that's why i tried a second version using the Null assignment (see the second patch) as the the global SetNull usage. the SetNull method as part of class interface besides IsNullInstance and Nuller ctor is not needed in that way.. it's kind a same approach as with global IsNull template.

the global template SetNull would be used by RichValue<> in GetNull(), which prevented the compile for the custom RichValue<float>.. since i couldnt specify additional operator float() in the Nuller (the idea of a template T() operator for Nuller is not practical for ambiguity reasons).

the IsNullInstance question was why there is no unique verision of IsNull API inside Value handled classes. sometimes they are IsNullInstance (which is dictated by IsNull<>) and sometimes it's IsNull, where a template specialisation is needed, like in Font..
is it legacy or should it be unified?

Subject: Re: Core: Null handling incoherent?
Posted by mirek on Fri, 06 May 2011 11:06:43 GMT
View Forum Message <> Reply to Message

kohait00 wrote on Fri, 06 May 2011 04:43
is it legacy or should it be unified?

In case of Font, unification would perhaps make sense, but it is IMO pretty low on priority.

Subject: Re: Core: Null handling incoherent?
Posted by kohait00 on Fri, 06 May 2011 12:07:38 GMT
View Forum Message <> Reply to Message

yeah, i agree..

but what about the first change...?
the global SetNull..this makes usage of RichValue<> with some intrinsic types possible (to a certain degree at least)..
are you willing to handle it like that?

Subject: Re: Core: Null handling incoherent?
Posted by kohait00 on Mon, 23 May 2011 11:40:30 GMT
View Forum Message <> Reply to Message

here comes a full patch with all i have encountered so far, it's not a lot more..Font, Draw, Painting.
with the changes ide comiles without problems..

File Attachments
1) patch4.patch, downloaded 311 times

Subject: Re: Core: Null handling incoherent?
Posted by kohait00 on Thu, 07 Jul 2011 06:30:51 GMT

View Forum Message <> Reply to Message

hi mirek..

what is your position towards this? this would make useability of custom unchangeable types usable in RichValue, just using the upp gloabal template means.. i think it's worth it..

Subject: Re: Core: Null handling incoherent?
Posted by mirek on Thu, 07 Jul 2011 08:29:53 GMT
View Forum Message <> Reply to Message

IMO, it solves too little for changes so big.

As said before, the standard interface is constructor from Null.

I understand the trouble with preexisting types, but IMO SetNull is only half-way solution.

Mirek

Subject: Re: Core: Null handling incoherent?
Posted by kohait00 on Thu, 07 Jul 2011 08:35:08 GMT
View Forum Message <> Reply to Message

i dont encounter the changes too big, since the standard inline template assigns Null. it merely enables to override the means since for there are cases (like the complex, which we gladfully managed to solve in another way), where you neither can change the class (or dont want to) nor can you add a operator Foo() to Nuller. in those cases you specialize SetNull. to at least be able to use it with RichValue..

please reconsider it. it does not change any semantic.. =Null assignment is still the default..but it offers means to intercept..

what would be the full way solution?

Subject: Re: Core: Null handling incoherent?
Posted by kohait00 on Tue, 13 Sep 2011 16:08:25 GMT
View Forum Message <> Reply to Message

hi mirek..
i'm depending on that a bit.. for my bazaar/LogPosCtrl...

take a look at bazaar/Gen/Vtypes.h/.cpp
there, i have Ctrl::LogPos RichValue'ized, which would not be possible without the SetNull template, since it has no from Nuller ctor.

if you are concerned about the several IsNull->IsNullInstance renames, i can provide a patch without it. but it only code cleanup. think of the last release date we are far away..maybe a good chance to correct things.

for the float problem, i managed to circumvent the Value extension, but this is way more important, cause very general.

moving the things to Defs.h is just clean. since several other classes do depend on the stuff, that otherwise resides scattered in the code. again, the beginner user benefits from it encountering it where expected..

Subject: Re: Core: Null handling incoherent?

Posted by cbpporter on Wed, 14 Sep 2011 07:46:18 GMT

View Forum Message <> Reply to Message

Do you think that adding an IsNull instantiation for unsigned would cause any troubles? Currently it fails to compile.

Subject: Re: Core: Null handling incoherent?

Posted by kohait00 on Wed, 14 Sep 2011 10:46:41 GMT

View Forum Message <> Reply to Message

what exactly did you compile? i cant reproduce any unsigned errors..when i try to compile the LogPosCtrlTest, it fails due to SetNull stuff not present...

imho, unsigned is not neccessary for Value..as mirek already lined out in an older thread, where i proposed to enrich value by all signed type.. Value does not care about the sizes of the representation, it cares only about the logic types, beeing bool, integer, and floating point values (represented by whatever highest available precision, bool, int64, double). int is merely from old days..

did you try the set null patch?

EDIT: heres the patch as svn patch again.. for a quick test
again, the changes are not deep, there are some .IsNull to .IsNullInstance renames for clean code base..the rest is block moving and some extensions

File Attachments

1) nullfixes.svn.patch, downloaded 286 times

Subject: Re: Core: Null handling incoherent?

Posted by mirek on Thu, 15 Sep 2011 10:57:03 GMT

View Forum Message <> Reply to Message

kohait00 wrote on Wed, 04 May 2011 04:46hi all,

i'm wondering why

```
template <class T> bool IsNull(const T& x) { return x.IsNullInstance(); }
```

and not

```
template <class T> bool IsNull(const T& x) { return x.IsNull(); }
```

is there a reason for it? name clashes??

Think about:

```
void Ctrl::TestSomething()
{
    String x = Something();
    if(IsNull(x)) ...
}
```

Mirek

Subject: Re: Core: Null handling incoherent?

Posted by kohait00 on Thu, 15 Sep 2011 12:29:33 GMT

View Forum Message <> Reply to Message

sorry, i cant get the fault   help, hint?

Subject: Re: Core: Null handling incoherent?

Posted by mirek on Thu, 15 Sep 2011 13:34:22 GMT

View Forum Message <> Reply to Message

When IsNull is method, compiler will presume you want to use it instead of global function. That is why method has to have different name.

Well, there might be some IsNull methods somewhere in concrete or special classes still. Not sure this needs fixing. IsNullInstance is contract to make IsNull work without specialized global function.

Mirek

Subject: Re: Core: Null handling incoherent?
Posted by kohait00 on Thu, 15 Sep 2011 14:30:57 GMT

View Forum Message <> Reply to Message

ok i was aware of this, but couldnt figure out what you where aiming at with the example..sorry..

this is totally clean approach. thats why the patch contains some fixes of some classes, (Uuid, Font), which use IsNull as member, which doesnt fit cleanly.

sorry for bothering you with this, but i really heavily rely on flexible Value extension for RichValue..and really hope you are alright with the propose or any alternative. since it does not change any semantic. =Null is still default assignment everywhere (the SetNull template is only used in RichValue..and uses =Null approach as inlined default). it offers possibility to override the =Null assignment and try the SetNull where needed, i.e. when using classes in RichValue, specified all from extern (seriliaze, xmlize, gethashvalue, as template specialisations)

i know it's not the full problem solution..is there any other idea?

Subject: Re: Core: Null handling incoherent?
Posted by mirek on Fri, 16 Sep 2011 08:26:30 GMT

View Forum Message <> Reply to Message

Patch applied.

Mirek

Subject: Re: Core: Null handling incoherent?
Posted by kohait00 on Fri, 16 Sep 2011 13:26:18 GMT

View Forum Message <> Reply to Message

thanks from saving me from heart attack