Subject: Re: Message from the system.
Posted by dolik.rce on Sat, 26 Jun 2010 21:28:47 GMT
View Forum Message <> Reply to Message

281264 wrote on Sat, 26 June 2010 22:07Hi,

I am attaching the files of the application in question (please remember that the purpose is to get familiar with U++). The application is very simple: open a dialog by clicking the button in the TopWindow; then type two (double) values in the Edit Fields; then press OK button , close dialog and then compute the multiplication and then show the outcome in the EditField in the TopWindow.

The idea is to practice with dialogs: how to create them, how to transfer values to the TopWindow, etc.

You will see that I still have no idea about how to position the widgets in the dialog, therefore the layout in the dialog is rubbish.

Taking advantage of your help, please allow me to ask some questions:

1.- How to pass the values from the EditFields in the dialog to the TopWindow; so far I have declared public the variables in the dialog so they can be accessed;
2.- How a string captured by a EditField can be converted to a numerical value (let us say double) and vice versa;

guaranteed that the dialog is deleted.
4.- In the manual I have seen expressions of this kind:
virtual Value Scan(const Value& text) const

for been overridden.
4.2.- Why the argument is a &?
4.3.- Any other suggestion related with "good" practices when programming.

I am sorry about my poor knowledge of C++. Your help is very appreciated.

Many thanks.

Best wishes,

Javier

Hi Javier,

First, I didn't get the warning at the end and I didn't even see anything that might cause trouble in your sources. I tested with GCC on Linux, maybe windows compiler behaves bit different... Someone else might be able to help you.

Now to your questions:

1. I use the same way as you. It is the simplest and most elegant solution I know.
2. There are DblStr, StrDbl, IntStr and StrInt functions. If you need something more fancy for number->String conversion, you can use Format* functions.
3. As I said, I didn't see any problems with the app nor in code. All the dialogs in U++ (if used properly, and you did) live until their scopes end. That means that if you create a dialog in your app it exists until it's parent exists or until it's scope (denoted by {}) ends. The default destructor takes care about deleting it properly.
4.1 In C++ virtual means that the function can be overriden, that is all there is to it. The reason for this is that sometimes you need more special behavior in the derived class, so you can just override the functions from base class.
4.2 The & in function argument generaly denotes that the argument should be passed by reference. That means that only address of the object is sent to the function and it operates on the same object as the rest of your program. If you omit the & it will be first copied and this copy will be passed to the function. This have two serious consequences: First, passing by reference is much faster, second, the modifications you do to the object will persist even after the function call.

And at the end few hints for you
1) Sizeable() should add MaximizeBox and MinimizeBox automatically, sou you don't have to call them explicitly.
2) This boton_ok.WhenAction=callback(this,&dialogo::salir_ok); can be written as
 boton_ok.WhenAction=THISBACK(salir_ok); or even as boton_ok<<=THISBACK(salir_ok); All you have to do for this macro to work is to add one typedef to the class definition: class dialogo:public TopWindow{
public:
 typedef dialogo CLASSNAME;
 dialogo();
 ... U++ saves you some typing
3) I see you are not using the layouts yet, so just for future: You can put all your layouts into one .lay file, no need to have a one file per layout.

I hope my answers were at least a bit useful to you.

Honza