Subject: Re: Select Grid Row BY ID
Posted by Oblivion on Mon, 08 May 2017 17:09:36 GMT
View Forum Message <> Reply to Message

Quote:
if I use the rows preset ID (grid.AddIndex(ID)Wink
and accessing the row by ID (int rownum = grid.Find(rowid, ID); grid.GetRow(rownum))
the program compiles flawlessly but terminates with a runtime error
(I assume the row Id changes during execution since a concurrent thread deletes a row)

it's a sheer PITA to not being able to just grid[rowid].doStuff() with a ficxed rowid not irritated by sortorder or number of rows or position of row on the grid...

So let me get this straight:

You are trying to access and manipulate a Grid object (in which you store some information) from within worker threads, right?

If this is the case, the first rule you need to remember is that GUI related stuff in U++ should be done in the main thread.
Hence you'll need serialized access. And if you want serialized access to gui elements from within threads, you should consider using PostCallback() function.

GuiMT example in the U++ reference examples in principle demonstrates this behaviour.

E.g.

If you are going to do stuff in a Grid, you can:

```
// Let us assume that we've defined a Grid object as grid in MyApp;
// Then in your worker thread you can call PostCallback().
// Of course you'll need a more complicated version of control code,

//WorkerFoo() represents a  worker thread function.

MyApp::WorkerFoo()
{
	auto rowid = 1;
	PostCallback( [&] { grid(rowid).DoStuff();  } );
}
```

Of course you need to design your code keeping in mind that removing elements from arrayctrls and grids invalidate references and pointers. (see docs and examples)

Regards,

Oblivion.