Subject: Tree Control - how to access properties of node.ctrl?
Posted by nixnixnix on Mon, 09 Jul 2007 05:10:15 GMT
View Forum Message <> Reply to Message

Hi,

I have my own subclass of Option which I pass into TreeCtrl. However, much later I try to get the control from the TreeCtrl but all I get is a standard Ctrl object and I am not allowed to cast it to the type of control which I originally passed in.

I try this within my TreeCtrl derived class

```
 TreeCtrl::Node node = GetNode(id);
 LayerOption* ptr = (LayerOption*)(node.ctrl);
 Layer* pLayer = ptr->GetLayer();
```

but I am not allowed the cast on the second line. The Ctrl I want to cast to looks like this

```
class LayerOption : public Option {
private:
 virtual void LeftDown(Point p, dword keyflags)
 {
  if(p.x<this->GetSize().cy)
   Option::LeftDown(p, keyflags);
  if(this->GetData()==true)
   SetFocus();

  // now send pass on this event to the parent for possible drag and drop
  Ctrl* pCtrl = this->GetParent();
  Point pt = p + GetRect().TopLeft(); // transform event into parent coordinates
  if(p.x>this->GetSize().cy)
   pCtrl->LeftDown(pt,keyflags);

  Refresh();
 }

 class Layer* m_pLayer; // these classes need to know about each other

public:
 Layer* GetLayer(){return m_pLayer;} // allows the option ctrl to identify its own layer
 void SetLayer(Layer* pLayer){m_pLayer = pLayer;}

 bool DropTarget();
 bool DragObject();
};
```

and is the same type of control I pass into the tree control in the first place and its LeftDown function gets executed so I know its being used.

Is there any way to access this control from the TreeCtrl object please?

Nick

Subject: Re: Tree Control - how to access properties of node.ctrl?
Posted by mirek on Mon, 09 Jul 2007 08:13:29 GMT

View Forum Message <> Reply to Message

```
LayerOption* ptr = (LayerOption*)~node.ctrl;
```

should work.

Anyway, depending on what exactly your code is doing, maybe you should consider accessing your Array<LayerOption> directly (especially if some node removal is involved). You should be able to do it by:

- creating the node without Ctrl just to get node id (which is int)
- then use the node id you got with Array::At method to create (or recycle) the widget, use GetNode/SetNode to assign Ctrl
- then you can use the id of node as index to the array

Mirek

Subject: Re: Tree Control - how to access properties of node.ctrl?
Posted by nixnixnix on Mon, 09 Jul 2007 14:18:17 GMT

View Forum Message <> Reply to Message

Thanks Mirek,

Is this a UPP peculiarity or am I just needing to read a book on the finer points of C++?

Anyway, as usual you cure my headache effortlessly

Cheers,

Nick

p.s. in case you are interested, I don't want to use IDs because the user needs to be able to order and reorder the tree by dragging and dropping nodes so the IDs assigned by the tree are pretty much useless to me as I understand it. This is why I use pointers to my Layer objects (each layer having an array of pointers to child layers and one parent layer pointer). Layers draw their children and so on. The customised option control enables me to communicate between the tree view and the main view which displays the layer content.

Subject: Re: Tree Control - how to access properties of node.ctrl?

Posted by mirek on Mon, 09 Jul 2007 16:42:28 GMT

View Forum Message <> Reply to Message

nixnixnix wrote on Mon, 09 July 2007 10:18Thanks Mirek,

Is this a UPP peculiarity or am I just needing to read a book on the finer points of C++?

Well, yes, that would help as well as looking at Ptr interface

BTW, in U++ we are very often using operator~ in this exact situation - we overload it to provide a "cast to natural type", e.g. for String, it returns const char *....

Quote:
p.s. in case you are interested, I don't want to use IDs because the user needs to be able to order and reorder the tree by dragging and dropping nodes so the IDs assigned by the tree are pretty much useless to me as I understand it. This is why I use pointers to my Layer objects (each layer having an array of pointers to child layers and one parent layer pointer). Layers draw their children and so on. The customised option control enables me to communicate between the tree view and the main view which displays the layer content.

That is OK, as long as the only way how to remove nodes is with the whole tree. If you are allowed to remove individual node, then you have to make sure the node Ctrl instance is destroyed...