Subject: Two problems (1 serious) with TreeCtrl
Posted by mrjt on Thu, 17 May 2007 17:07:32 GMT
View Forum Message <> Reply to Message

The first is just annoying, the second is more important:
1) Nodes can be selected (and still trigger WhenSel) when canselect is false. My suggested fix:

```
void TreeCtrl::SetCursorLine(int i, bool sc)
{
 if(nocursor)
  return;
 if(i != cursor) {
  i = minmax(i, 0, line.GetCount() - 1);
  if(i < 0) return;
  Item& m = item[line[i].itemi];
  if (!m.canselect) return; <<= This line added by me
  if(sc)
   sb.ScrollIntoY(line[i].y, m.GetSize().cy);
  RefreshLine(cursor);
  cursor = i;
  RefreshLine(cursor);
  if(m.ctrl && m.ctrl->SetWantFocus())
   return;
  WhenCursor();
  WhenSel();
 }
}
```

2) Opening/closing nodes causes WhenSel to be triggered, which is correct under some conditions but it will be triggered even when the selection doesn't actually change.

What makes this problem worse is that WhenSel is called from SyncTree, which is called by Paint. If you have code that causes a call to Ctrl::WindowProc (say PromptOK) int your WhenCursor or WhenSel callbacks this will trigger the
ASSERT(!sPainting);
macro and presumably crash a non-debug build.

Sorry, I don't have time to post a test case now, but I'll do one in the morning.
James

Subject: Re: Two problems (1 serious) with TreeCtrl
Posted by mirek on Thu, 17 May 2007 17:18:02 GMT
View Forum Message <> Reply to Message

Hopefully, this should fix the second problem:

```
void TreeCtrl::SyncTree()
{
 if(!dirty)
  return;
 if(noroot)
  Open(0);
 Ptr<Ctrl> restorefocus = GetFocusChildDeep();
 hasctrls = false;
 int cursorid = GetCursor();
 for(int i = 0; i < item.GetCount(); i++)
  item[i].linei = -1;
 line.Clear();
 Size treesize = Size(0, 0);
 if(noroot) {
  if(GetChildCount(0))
   treesize.cy = -item[0].GetSize().cy;
  ReLine(0, -1, treesize);
 }
 else
  ReLine(0, 0, treesize);
 treesize.cy = max(0, treesize.cy);
 treesize.cx += levelcx;
 sb.SetTotal(treesize);
 cursor = -1;
 dirty = false;
 if(cursorid >= 0)
  SetCursor(cursorid, false, false, false);
 SyncCtrls(true, restorefocus);
 SyncInfo();
}

void TreeCtrl::SetCursorLine(int i, bool sc, bool sel, bool cb)
{
 if(nocursor)
  return;
 if(sel && multiselect) {
  ClearSelection();
  SelectOne(line[i].itemi, true);
 }
 if(i != cursor) {
  i = minmax(i, 0, line.GetCount() - 1);
  if(i < 0) return;
  Item& m = item[line[i].itemi];
  if(sc)
   sb.ScrollIntoY(line[i].y, m.GetSize().cy);
  RefreshLine(cursor);
  cursor = i;
  RefreshLine(cursor);
  if(m.ctrl && m.ctrl->SetWantFocus())
   return;
  if(cb) {
   WhenCursor();
   WhenSel();
  }
 }
}

void TreeCtrl::SetCursor(int id, bool sc, bool sel, bool cb)
{
 while(id > 0) {
  ASSERT(id >= 0 && id < item.GetCount());
  MakeVisible(id);
  SyncTree();
  const Item& m = item[id];
  if(m.linei >= 0) {
   SetCursorLine(m.linei, sc, sel, cb);
   return;
  }
  id = m.parent;
 }
 SetCursorLine(0, sc, sel, cb);
}
```

Subject: Re: Two problems (1 serious) with TreeCtrl
Posted by mirek on Thu, 17 May 2007 17:19:10 GMT
View Forum Message <> Reply to Message

Add 1:

WhenSel triggers for cursor change too.

I think that selected and cursor are two distinct things..

Subject: Re: Two problems (1 serious) with TreeCtrl
Posted by mrjt on Fri, 18 May 2007 12:58:13 GMT
View Forum Message <> Reply to Message

Thanks, the fix above works perfectly. Unfortunately the more I've tested this ctrl the more issues I have found, including a bug (in the latest dev) where the Ctrl destructor triggers a callback (~Ctrl->Close...->TreeCtrl::ChildRemoved->Dirty->ClearSelection), causing a crash.

Quote:I think that selected and cursor are two distinct things..

True, but that doesn't mean I have to like it

The separation works well if you are using the ctrl (or ColumnList ctrl) with multiselect = true, but causes problems when in single selection mode.

For instance, if multiselect=false and an item is clicked no items are actually selected but the cursor is set (this is why WhenSel is triggered in SetCursor). This is good because unless you are using multi-selection you shouldn't have to scan the whole tree to get the selected node. On the other hand, SetCursorLine must check that the node can be selected and to be consistent SetCursor should NOT trigger WhenSel().

I think I've finally fixed all of the bugs/problems mentioned above, plus quite a few others (including the crash). I've left the WhenSel callback in SetCursor[Line], but prevented triggering when in multisel mode as this is both incorrect and produces multiple callbacks on click events. I've also modified Key handling.

I won't bother listing changes here as they are fairly extensive. I've attached the files and my test package.

James

File Attachments

1) TreeTest.zip, downloaded 596 times
2) TreeCtrl.h, downloaded 708 times
3) TreeCtrl.cpp, downloaded 656 times

Subject: Re: Two problems (1 serious) with TreeCtrl
Posted by mirek on Sun, 20 May 2007 16:45:43 GMT
View Forum Message <> Reply to Message

Well, so far I did not expect 'canselect' checking to happen in '!multiselect' mode....

Subject: Re: Two problems (1 serious) with TreeCtrl
Posted by mirek on Sun, 20 May 2007 21:46:51 GMT
View Forum Message <> Reply to Message

Patches accepted.